U.S. DEPARTMENT OF COMMERCE
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
NATIONAL WEATHER SERVICE
OFFICE OF SYSTEMS DEVELOPMENT
TECHNIQUES DEVELOPMENT LABORATORY

TDL OFFICE NOTE 00-2

COMPUTER PROGRAMS FOR MOS-2000

Edited by

Harry R. Glahn and J. Paul Dallavalle

January 2000

COMPUTER PROGRAMS FOR MOS-2000

Edited by

Harry R. Glahn and J. Paul Dallavalle

INTRODUCTION

This office note TDL ON 00-2 is a companion to TDL ON 00-1, MOS-2000. It contains documentation of most of the programs used in the MOS-2000 system. Programming for this system was underway in 1994. In accordance with TDL ON 00-1, Chapter 3, each main program and subroutine that has lasting value is documented, although for subroutines the documentation may be brief if it is specific to one program and not likely to be used by others.

Each routine has the name of the programmer and the date of original writing. In the upper right of each page is the date the writeup (and possibly the code itself) was last modified; these dates may be prior to April 1, 2000, the issue date of this office note. The numbering scheme follows that specified in TDL ON 00-1, Chapter 9. Two Tables of Contents are provided, one in alphabetical order and one in numerical order; in these Tables of Contents, a "*" following the name means the writeup is very brief. The location of the source code is described in TDL ON 00-1, Chapter 3.

The software for MOS-2000 is written so that it can function on either a 32-bit word workstation (e.g., Hewlett Packard[1] 755), or a 64-bit word machine like the CRAY. This has added some complexity, but was felt desirable because it is uncertain which portions of the MOS-2000 system will be most efficiently and effectively performed on a mainframe as opposed to the workstation; likely, this split will change with time as workstation capabilities continue to increase.[2]

The forerunner to MOS-2000 was a system documented in TDL Office Note 74-14 and its computer programs in TDL Office Note 75-2.

This office note is a "living document" and will be updated as necessary. Programs will be added and modifications made as needed. The header will establish the creation or update date.

[1]No indorsement of specific equipment or companies is made or implied in this document.

[2]The MOS-2000 system has been in the planning and development stages for several years. During that time, the equipment available changed. As of this writing, the National Weather Service CRAY mainframes are being replaced by IBM machines. The 64-bit capability has been retained, although it may not be needed in the future.

CONTENTS

6.5*	WRAA	Writes AA Matrix for U600
6.6*	RDAA	Reads AA Matrix for U600
6.7*	SAVREC	Saves Information for U600 for Restoring
6.8*	RESTOR	Restores Information for U600 Saved by SAVREC
6.9*	SUMRY	Provides Summaries for U600
6.10		
6.11*	RDFORC	Reads a Forced Predictor List
6.12*	XFER	Transfers Cross-Products to Two-Dimensional Array
6.13*	XPROD	Computes Sample Sizes, Sums, and Sums of Cross-Products
6.14*	PRCROS	Prints Sample Size, Sums, and Sums of Cross-Products
6.15*	VRBVEC	Computes a Useability Vector for Variables for U600
6.16*	WREQ	Writes Equations for Printing
6.17*	WRHEAD	Writes Header Records for Operational Equations
6.18*	WROPEQ	Writes Equations for Operational Use
6.19*	THSET	Sets Thresholds
6.20	U660	Collates, Prints, and Packs Data
6.21*	PRU660	Prepares Formats and Writes for Printing for U660
6.22*	RDVR66	Reads Variable List for U660
6.23*	INT660	Initializes from Control File for U660
6.24	U602	Performs Screening Regression Analysis for Multiple Predictands
6.25	U625	Inventories Equation Files Read by U700 and U900
6.26	U630	Edits Station Lists in Equation Files Read by U700 and U900
6.27*	U661	Collates, Prints, and Packs Data
6.28	U662	Merges, Collates, Prints, and Packs Vector Data
6.29	U175	Computes Means in Categories (Not used or distributed)
6.30	U605	Performs Screening Regression Analysis on Ensemble Data
6.31	U655	Performs Logit Analysis.
7.0	U700	Makes Forecasts from Regression Equations
7.1	U710	Postprocesses Forecasts on Sequential Files
7.2	U705	Makes Forecasts and Computes Errors from Regression Equations
8.0	U850	Verifies Forecasts
8.1	U830	Computes Thresholds for Making Categorical Forecasts
8.2	U855	Verifies Gridded Forecasts
9.0	U900	Makes Operational Forecasts from Regression Equations
9.1	U910	Postprocesses Forecasts on Random Access Files
9.2	U911	Inventories Operational Regression Equation Files

CONTENTS (ALPHABETIZED)

2.49*	COMBIN	Combines Data from Two Models into One Predictor
4.87	COMPID	Computes IDs for Data in MOS-2000 External File System
4.97*	CONST	Retrieves Data from MOS-2000 External Random Access File
4.136*	CONSTG	Retrieves Gridded Constant Data from MOS-2000 External Random Access File
4.71*	CONSTX	Retrieves Data from MOS-2000 External Random Access File for Vector-Oriented Programs
4.135*	CONST1	Retrieves Data from MOS-2000 External Random Access File
2.5	COMPID	Computes IDs for Data in MOS-2000 External File System
2.68*	CORDP	Estimates Dew Points During Extremely Cold Conditions or When Relative Humidity is Reported
4.10	DATGEN	Generates Date/Times Between Two Date/Times
4.15	DATPRO	Processes a Date/Time List
2.72*	DEWPCORR	Provides Bias-Corrected Dew Point Forecast from Model Output
2.50*	DEWPT	Computes Dew Point
4.105*	DIFFV	Computes Difference of a Forecast and an Observation
2.1	DIRCMP	Computes Grid Coordinates for Points
4.103*	DIRFUV	Computes Wind Direction from Components
2.71*	DIR2UV	Computes U- and V-Wind Components from Wind Direction and Speed
2.24*	DIVW	Computes Divergence
2.25*	DIVW1	Computes Divergence for DIVW
4.68	DOY	Computes the Day of the Year
4.119*	DPDPRS	Calculates Dew Point Depression
2.70*	DPTDPR	Computes Dew point Depression
2.31*	EOUWND	Computes Earth-Oriented U-Wind
2.32*	EOVWND	Computes Earth-Oriented V-Wind
2.30*	EOWND	Computes Earth-Oriented Wind
2.34*	EXTRAP	Extrapolates a Variable on a Grid
2.130*	FCSTDF	Computes Difference between Local and MOS Forecasts
4.84*	FINDST	Initializes Array with Station Locations
4.63*	FLOPNM	Opens a MOS-2000 File
4.11	FORIER	Computes Harmonics of Day of Year
2.72*	FRZLVL	Computes Freezing Level Pressure at Gridpoints
4.82	FSORT	Completes Sorting Values in an Array
4.109*	FTOKEL	Converts Fahrenheit to Kelvin
4.53*	GCPAC	Compresses Data in Internal MOS-2000 Storage System
4.48	GFETCH	Fetches Data from Internal MOS-2000 Storage System
2.2*	GRCOMB	Stores Grid Characteristics
2.35	GRIDB	Makes a Grid Binary Variable
2.3	GSITB	Smooths, Interpolates, Transforms, and Makes Grid and Point Binaries
4.47	GSTORE	Stores Data into Internal MOS-2000 Storage System
2.29*	GSWIND	Computes Geostrophic Wind Speed
4.91	GTVECT	Returns a Vector Variable from MOS-2000 Internal Storage
2.27*	GUWIND	Computes Geostrophic U-Wind
2.28*	GVWIND	Computes Geostrophic V-Wind
2.26*	GWIND	Computes Geostrophic Wind
4.18	IERX	Writes Message for System Errors
4.8	INTRP	Interpolates into a Precipitation Amount Field to Point Locations
4.6	INTRPA	Interpolates Quadratically into a Grid to Point Locations
4.7	INTRPB	Interpolates Linearly into a Grid to Point Locations
4.131	INTRPC	Finds Value in Grid Closest to Point Location
4.137*	INTRPD	Interpolates to Station by Using Value of Closest Gridpoint
6.1*	INT600	Initializes from Control File for U600

6.23*	INT660	Initializes from Control File for U660
4.108*	INTOMM	Converts Inches to Millimeters
4.17	IPOPEN	Opens Files Used for Output
2.53*	KINDEX	Computes The K-Index
2.84*	KINXRF	Computes Interactive Predictor K-Index * Thunderstorm Relative Frequency
4.110*	KWHTOJ	Converts Kilowatt Hours to Joules
4.116*	LCL	Computes the Pressure and Temperature of the Lifted Condensation Level
2.74*	LINDEX	Computes Lifted Index
4.42	LMIJLL	Converts Grid Coordinates to Latitude/Longitude for Lambert Map
4.41	LMLLIJ	Converts Latitude/Longitude to Grid Coordinates for Lambert Map
2.18*	LMMAPF	Computes Sine of Latitude and Map Factor for Lambert Map
2.75*	LMMAPS	Computes Sine of Latitude and Map Factor for Lambert Conformal Grid
4.51*	LMSTR1	Initializes MSTORE and Prepares LSTORE for GCPAC after Day 1
4.52*	LMSTR2	Prepares LSTORE for GCPAC
2.38*	L1D	Switches to 1-Dimensional Linearization Subroutines
2.40*	L1D1	Supports 1-Dimensional Linearization
2.39*	L1D50CFFF	Linearizes a Variable, 2 Seasons, 4 Projections, by Cycle
2.41*	L12D2	Provides Variable to Linearize for L1D1 or L2D1
2.42*	L1D3	Computes 1-Dimensional Linearized Variable
2.44*	L2D	Switches to 2-Dimensional Linearization Subroutines
2.46*	L2D1	Supports 2-Dimensional Linearization
2.47*	L2D3	Computes 2-Dimensional Linearized Variable
2.45*	L2D601000	Linearizes a 850-mb and 1000-mb Heights
2.48*	L2D600101	Linearizes a 850-mb Geostrophic Vorticity and 1000-mb Height
2.16*	MAPLAT	Computes Sine of Latitude and Map Factor
2.76*	MAXMIN	Computes Daytime Maximum and Nighttime Minimum Temperatures
2.77*	MAXMINB	Computes Daytime Maximum and Nighttime Minimum Temperatures
4.120*	MAXTEST	Ensures Consistency Between 3-h Temperatures and Max or Min Forecasts
2.78*	MCMAPF	Computes Sine of Latitude and Map Factor for Mercator Map
2.79*	MCMAPS	Computes Sine of Latitude and Map Factor for Mercator Grid
2.80*	MDIV	Computes Moisture Divergence
2.81*	MEANRH	Computes Mean Relative Humidity Between Two Isobaric Levels
2.54*	MIXRAT	Computes Mixing Ratio
2.82*	MODELMXMN	Computes 12-H Max or Min Temperature from 6-H Model Values
4.102*	MPSKTS	Converts Meters Per Second to Knots
4.121*	MRFTEST	Ensures Consistency Between Successive Max or Min Forecast Tempera-tures
4.107*	NMLPRB	Normalizes a Set of Probabilities to Lie in the Range 0 to 1
2.52*	NONCNVP	Computes Non-Convective Precipitations Amount
4.98*	NORWND	Modifies Wind Direction
2.124*	OBSADJDIR	Sets the Observed Wind Direction to Zero (Calm) when the Adjusted Wind Speed Equals Zero
2.123*	OBSADJSPD	Adjusts Observed Wind Speeds to a 10-Meter Surface
2.125*	OBSADJUV	Adjusts the Observed U- and V-wind components to a 10-Meter Surface
2.83*	OBSCIGHT	Computes Ceiling Height
2.85*	OBSDMAXT	Computes Daytime Maximum Temperature
2.86*	OBSDPTD	Computes Dew Point Depression from Observational Data
2.87*	OBSMRCLD	Computes Mean Total Sky Cover for a 12-H period
2.88*	OBSMRPTYPE	Computes Precipitation Type for a 12-H Period
2.89*	OBSMRWSP	Computes Observed Maximum Wind Speed for a 12-H Period
2.90*	OBSNMINT	Computes Nighttime Minimum Temperature
2.91*	OBSOBVIS	Assigns Present Weather Reports to Categories of Non-precipitating Obstructions to Vision

2.61*	OBSPOPC	Classifies Present Weather Precipitation Observations into One of Three Possible Categories
2.62*	OBSPOPO	Classifies Present Weather into a Yes/No Designation According to the Occurrence/Nonoccurrence of Precipitation
2.63	OBSPOPO3	Classifies Present Weather into a Yes/No Designation According to the Occurrence/Nonoccurrence of Precipitation over a 3-Hour Period
2.64*	OBSPRWXBIN	Produces Freezing/No Freezing, Snow/No Snow, Rain/No Rain Binaries from Present Weather
4.113*	OBSPTYPE	Classifies Present Weather into Precipitation Type Categories-precipitating Obstructions to Vision
4.114*	OBSTCLD	Computes Total Cloud Coverage
2.92*	OBSTWET	Computes the Average of the Observed Temperature and Dew Point
4.122*	OPFCST	Makes Operational Forecasts from Primary and Backup Forecasts
2.12*	OPTION	Switches to Computational Subroutines
2.13*	OPTN2	Switches Secondarily to Computational Subroutines
4.73*	OPTX	Switches to Computational Subroutines for Vector-Oriented Programs
4.24*	PACK	Packs Data in TDLPACK Format
4.27*	PACKGP	Determines Groups for Packing
2.15	PACKV	Packs and Writes 1-Dimensional Data in TDLPACK Format
4.23	PACKXX	Determines Whether to Pack Spatial Differences With No Missing Values
4.74	PACKYY	Determines Whether to Pack Spatial Differences With Primary Missing Values
4.21	PACK1D	Packs a 1-Dimensional Array
4.22	PACK2D	Packs a 2-Dimensional Array
3.2*	PAWRA	Prepares, Packs, and Writes a Record to a MOS-2000 Random Access File
3.7*	PAWRAG	Prepares, Packs, and Writes a Gridpoint Record to a MOS-2000 Random Access File
2.93*	PBLMIX	Calculates the Combined Influence of the Temperature Profile, Average Wind, and Relative Humidity in the Planetary Boundary Layer
2.127	PCPEFF	Computes Precipitation Efficiency
2.94*	PCPX6	Computes Observed Precipitation Amount
4.28	PKBG	Packs Bits into Words
4.25*	PKMS00	Assists Packing When There Cannot be Missing Values
4.75*	PKMS77	Assists Packing When There Can be Secondary Missing Values
4.77*	PKC4LX	Assists Subroutine PACK in Complex Packing
4.76*	PKS4LX	Assists Subroutine PACK in Simple Packing
4.26*	PKMS99	Assists Packing When There Can be Missing Values
4.100*	POPDIF	Computes difference of Two PoP Forecasts
2.95*	POTEMP	Computes Potential Temperature
6.14*	PRCROS	Prints Sample Size, Sums, and Sums of Cross-Products
2.14	PREDX1	Prepares Title and Gridprints Field
4.36	PRSID	Constructs MOS-2000 ID and Parses It into Component Parts
4.44	PRSID1	Parses MOS 2000 ID into Component Parts
2.10*	PRED21	Computes Predictors for U201 for Day 1
2.11*	PRED22	Computes Predictors for U201 for Days after Day 1
4.33	PRTGR	Contours Two-Dimensional Grid
4.40	PSLLIJ	Converts Latitude/Longitude to Grid Coordinates for Polar Stereographic Map
2.17*	PSMAPF	Computes Sine of Latitude and Map Factor for Polar Stereographic Map
2.96*	PSMAPS	Computes Sine of Latitude and Map Factor for Stations Projected onto a Polar Stereographic Map
6.21*	PRU660	Prepares Formats and Writes for Printing for U660
2.97*	QADV	Computes Advection of specific Humidity

6.6*	RDAA	Reads AA Matrix for U600
4.14	RDC	Reads Formatted Records into a Character Array
4.69	RDDIR	Reads Directory Record
4.62*	RDDATM	Reads a Data Record for RDTM
4.50*	RDDISK	Read Data from Internal MOS-2000 Disk File
4.12	RDI	Reads Formatted Records into an Integer Array
4.13	RDF	Reads Formatted Records into a Real Array
4.61*	RDKEYM	Reads a Key Record
6.11*	RDFORC	Reads a Forced Predictor List
4.90	RDNAME	Reads Plain Language and Scaling Parameters from Variable Constant File
4.88	RDONEV	Reads and Returns a Specific Variable from a TDLPACK Vector File
4.19*	RDPRED	Reads and Orders Predictor List
4.16	RDSNAM	Reads Unit Numbers and Opens Associated Files
4.45	RDSTAD	Reads and alphabetizes a Station List and Directory
4.83	RDSTAL	Reads a Station List and Directory
4.86	RDSTGA	Reads and Alphabetizes One or More Station Lists and Directory Information
4.85	RDSTGN	Reads One or More Station Lists and Directory Information
4.95	RDSTNA	Reads and Alphabetizes One or More Station Lists with Group Names
4.96	RDSTNL	Reads One or More Station Lists with Group Names
4.54*	RDSTRX	Stores all Data for Day 1
2.4*	RDSTR2	Stores All Data for Day 1 for U201
4.59	RDTDLM	Reads Data from MOS-2000 External File System
4.60*	RDTM	Reads Data for RDTDLM
4.89	RDVECT	Reads and Returns a Variable from a List from a TDLPACK Vector File
4.72*	RDVRBL	Reads and Orders Variable List
6.22*	RDVR66	Reads Variable List for U660
6.8*	RESTOR	Restores Information for U600 Saved by SAVREC
4.123*	RHUMID	Calculates Relative Humidity
2.98*	RHVV	Computes Product of Vertical Velocity and Relative Humidity
2.99*	SAMEGR	Compares Grid Number and Grid Extent of Two Grids
6.7*	SAVREC	Saves Information for U600 for Restoring
2.100*	SATLEVNUM	Computes Fraction of Saturated Levels in vertical Sounding
2.101*	SATLEVRH	Computes Height of Saturation Level in Vertical Sounding
4.91	SETPLN	Places Plain Language for Variables
4.35*	SETUP	Sets Options for PRTGR
4.115*	SFCTCLD	Computes Total Cloud Cover
4.43	SKIPR	Skips Records
4.31	SKIPWR	Skips Records and Writes Station List
4.3	SMTH5	Smooths a Field by a 5-Point Smoother
4.4	SMTH9	Smooths a Field by a 9-Point Smoother
4.5	SMTH25	Smooths a Field by a 25-Point Smoother
4.93	SMTH2X	Smooths a Field with a 25-Point Smoother Twice
4.94	SMTH3X	Smooths a Field with a 25-Point Smoother Thrice
2.102*	SNOWFL	Computes Observed Snowfall Amount
2.104*	SNOWEQ	Computes Estimated 24-H Snowfall Amount Prediction
4.126*	SOLAMT	Computes Solar Energy
4.125*	SOLENG	Computes the Fraction of Possible Solar Energy
4.38	SORT16	Sorts Values in an Array
4.134*	STRATVAR	Calculates a Stratification Variable
4.127*	SUNAMT	Computes Amount of Sunshine
4.128*	SUNHRS	Calculates Hours of Sunshine and Extraterrestrial Radiation
2.55*	SPECHUM	Computes Specific Humidity

2.105*	SSR	Computes Single Station Regression (SSR) Predictors
6.9*	SUMRY	Provides Summaries for U600
4.124*	SUNFCT	Calculates Fractional Amount of Sunshine
2.106*	SVRVEC	Creates Thunderstorm and Severe Weather Predictand Data
2.107*	SWEATI	Computes Sweat Index
2.108*	SWTXRF	Computes Product of Sweat Index and Severe Thunderstorm Relative Frequency
4.67*	TDLPRM	Prints Contents of Common Block ARGC in the MOS-2000 File System
2.109*	TDPAVG	Computes Average of LAMP Temperature and Dewpoint Forecasts
4.112*	TEMPAV	Averages Two Variables with the Same Projection
4.138*	TGINTRP	Temporally Interpolates Gridded Model Data to Stations using Quadratic or Linear Interpolation
4.117*	THETAE	Computes Equivalent Potential Temperature
6.19*	THSET	Sets Thresholds
2.37*	TIMEP	Computes Variables as a Function of Time
4.99*	TIMEPV	Computes Time Difference, Mean, Max, or Min of Two Variables
2.110*	TIMGRD	Interpolates Model Grid Fields to Intermediate Hours
4.39	TIMPR	Time Stamps Output
2.211*	TIMTRP	Interpolates Forecasts to Intermediate Hours
2.112*	TMPADV	Computes Temperature Advection
4.111*	TMPCMP	Consistency Checks Temperature and Dew Point
2.113*	TPCP3	Computes 3-Hr Convective Precipitation Amount
2.114*	TPCP6	Computes 6-Hr Convective Precipitation Amount
2.115*	TPCP12	Computes 12-Hr Convective Precipitation Amount
2.116*	TPCP24	Computes 24-Hr Convective Precipitation Amount
4.70	TRAIL	Writes Trailer Record
4.9	TRANS	Transforms a Variable
2.117*	TRPD	Does Calculations for Interpolation Subroutine INTRPD
4.106*	TRUNCP	Truncates a Set of Probabilities to 0 and 1
2.118*	TSLCM	Calculates Upslope in Terrain Elevation
4.132*	TSLOP	Calculates West or South Terrain Slope
2.51*	TTOTALS	Computes Total Totals Index
4.29	UNPACK	Unpacks Data from TDLPACK Format
4.30	UNPKBG	Unpacks Bits from Words
4.81	UNPKLX	Assists in Unpacking of Group Information
4.78	UNPKOO	Assists Unpacking When There Are No Missing Values
4.79	UNPKPO	Assists Unpacking When There Are Primary Missing Values
4.80	UNPKPS	Assists Unpacking When There Are Primary and Secondary Missing Values
4.20	UPDAT	Adds Hours to a Date/Time
2.119*	UPSLCM	Calculates Upslope Winds from U and V Wind Components
4.133*	UPSLOP	Calculates Upslope Motion
2.126*	VARG	Computes Variance at Points on a Grid
2.36*	VERTP	Computes Vertical Variables
4.104*	VERTX	Computes Vertical Difference, Sum, or Mean of Two Variables
2.103*	VORTADV	Computes Absolute Vorticity Advection
2.20*	VORTH	Computes Geostrophic Vorticity
2.21*	VORTH1	Computes Geostrophic Vorticity for VORTH
2.22*	VORTW	Computes Vorticity
2.23*	VORTW1	Computes Vorticity for VORTW
6.4*	VRBLX1	Prints AA Matrix for U600
6.2*	VRBL61	Furnishes Variables to U600 for Day 1
6.3*	VRBL62	Furnishes Variables to U600 for Days After Day 1
6.15*	VRBVEC	Computes a Useability Vector for Variables for U600
2.120*	WETBULBT	Computes Wet Bulb Temperature

2.121*	WINDDR	Computes Wind Direction in Degrees
4.129*	WNDCHL	Calculates Wind Chill
6.5*	WRAA	Writes AA Matrix for U600
4.58*	WRDATM	Writes a Data Record for WRTM
4.49*	WRDISK	Writes Data into Internal MOS-2000 Disk File
6.16*	WREQ	Writes Equations for Printing
6.17*	WRHEAD	Writes Header Records for Operational Equations
4.32	WRITEP	Writes Records and Switches Files
4.57*	WRKEYM	Writes a Key Record
6.18*	WROPEQ	Writes Equations for Operational Use
4.55	WRTDLM	Writes Data into MOS-2000 External File System
4.56*	WRTM	Writes Data for WRTDLM
2.33*	WSPEED	Computes Wind Speed
4.46	XCHANG	Orders Values in an Array
6.12*	XFER	Transfers Cross-Products to Two-Dimensional Array
6.13*	XPROD	Computes Sample Sizes, Sums, and Sums of Cross-Products
4.101*	ZRONEG	Sets Negative Values to Zero
2.122*	ZRPRED	Computes Vertical Profile Index Indicating Occurrence of Freezing Rain

U170

COMPUTES CONSTANTS FROM VECTOR DATA

Mark S. Antolik
May 1, 1999

PURPOSE: U170 calculates constants from MOS-2000 vector data. These constants can be either means or relative frequencies and usually will be calculated for specific stations or groups of stations from climatic (or possibly model) data. These means or relative frequencies will ordinarily then be used as predictors in regression equations, or to assist in regionalization of predictand data during equation development. Constants may be computed over the entire range of possible values of the input vector variable or for exclusive or cumulative categories of variable values. Categories are defined by the real values appearing in word 4 of the MOS-2000 identifier. Due to practical constraints imposed in the interest of code simplicity and relative ease of use, constants from only one "basic" MOS-2000 variable can be computed on a single run. However, constants having the same basic variable ID but different categorical thresholds may be computed in the same U170 run. U170 prints the output constants in an easy-to-read ASCII format, and packs and writes the constants as TDLPACK vector output for the stations designated by control files. Input data must be in TDLPACK format, and because the packed data are self-describing, the input data can come from an essentially unlimited number of various sources. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U170 uses a driver DRU170 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program for every application. U170 is written to run on a 32-bit or a 64-bit word-length machine, which also is controlled by PARAMETER statements in the driver.

Many of the design features, subroutines, and variable names used in U170 were adapted from program U660. Some familiarity with other MOS-2000 documents, especially those pertaining to U660, will be helpful to full understanding of this writeup.

CONTROL FILE INPUT: 'U170.CN' (Unit = KFILDI)

KFILDI is the input unit number as specified in DRU170. As with other MOS-2000 programs, the control file U170.CN must reside in the same directory from which the main program (and driver) is being run. The control file is opened by the driver DRU170 with a standard FORTRAN OPEN statement.

Record Type 1 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 2 - Format (2(I10/),F10.0/,6(I10/),I10) **Run Control Parameters**

This record type contains a series of ten values which provide control over various basic features of each program run. Note that each value is on a separate line. A brief explanation can be put on the same line with each variable to assist the user in knowing which value is for which variable. Effort has been made here to group these values according to functionality.

Error control parameters:

NSKIP - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

PXMISS - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value.

Input Data Print Control Parameters:

The next two parameters control printing of input data as read by the program.

NPRINT - The number of cycles of raw input data to write for printing to unit KFILDO (as specified in the driver DRU170) under the format control and JP(2,) provided with each variable (see Record Type 11). This provides an easy way to examine the raw data for one or more input date/time groups.

LNGTH - The line length for printing the above raw data. For a line printer, 132 is appropriate; for a laser printer, 80 may be desirable.

Station List Control Parameters:

The next three values regulate certain aspects of the program's treatment of the station list.

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the older (3-character) call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

ICHARS - The number of characters of call letters to print when printing is indicated by JP(2,). This is constrained to be between 4 and 8 inclusive.

Date spanning Increment:

INCCYL - Determines the increment between dates when date spanning is used. INCCYL is the number of hours between date/times that are put into IDATE( ) (see Record Type 4, below), as a result of date spanning in subroutine DATPRO. Date/times cannot be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 0600 UTC should never be indicated. This should pose no hardship.

Output Constant Type Control Parameter:

Regulates the type of constants that will be calculated from the input variables of Record Type 11 below.

IMEAN - Indicates whether means (=1) or relative frequencies (=0) will be calculated for the input variable(s) given in Record Type 11. Means or relative frequencies from one "basic" variable only can be calculated in a single run of U170, but constants may be calculated simultaneously for multiple categories of the same input variable. The category breakpoints and orientation are determined by the input in Record Type 11. (see Record Type 11 as well as the Comments section).

Record Type 3 - Format (I3,4X,A60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 3 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and the string "DEFAULT" can be read in its place. (CHARACTER*60)

Record Type 4 - Format (7I10) **Date List**

This group of records determines, together with the parameter INCCYL in Record Type 2, the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, Record Type 4 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as

YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U170 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8 as set in the driver DRU170.

Record Type 5 - Format (I3,4X,A60) **Input Vector Data Files**

This group of records identifies the data sets from which the input vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. (ND6 is set in the driver DRU170.) Record Type 5 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

KFILIN(J) - Unit number for the data file.

NAMIN(J) - Name of file where these data reside. (CHARACTER*60)

Input vector data will usually be periodic weather observations or numerical model data. These data must be packed in the TDLPACK format. When data sets are to be used in sequence, they should be read in the proper chronological order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and the earliest year is on one data set and the other, more recent year is on a second data set, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 6a - Format (I3,4X,A60) **ASCII Output File for U351**

This record (plus the terminator record) identifies the data set which will contain a listing of the stations as well as the output constants in a format suitable for input to u351. Records are read until the terminator KFILAC = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 6a is read by subroutine RDSNAM and will be opened as 'NEW'. This output file is optional; if ASCII output for U351 is not desired, only the terminator need be present.

KFILAC - Unit number for the ASCII output file.

ASCIFM - Name of file for ASCII output. (CHARACTER*60)

Record Type 6b - Format (I3,4X,A60) **ASCII Output File**

This record (plus the terminator record) identifies the data set which will contain a tabular printout of output constants, with stations grouped as specified by the station list (see Record Type 9) and with or without intra-group alphabetization as specified by the parameter NALPH in Record Type 2. Records are read until the terminator KFILAS = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 6 is read by subroutine RDSNAM. This output file is optional; if tabular ASCII output is not desired, only the terminator need be present.

KFILAS - Unit number for the ASCII output file.

ASCIFL - Name of file for tabular ASCII output of constants. (CHARACTER*60)

Record Type 7 - Format (I3,4X,A60) **Packed Output File**

This record (plus the terminator record) identifies the packed vector output file containing the computed means or relative frequencies. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 7 is read by subroutine RDSNAM, and the file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

KFILIO - Unit number for the packed output.

OUTNAM - Name of file where packed constants are to reside. (CHARACTER*60)

Record Type 8 - Format (I3,4X,A60) **Station List and Directory Files**

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 8 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT" (CHARACTER*60), in which case all station directory information is taken from the control file U170.CN.

Record Type 9 - Format (7(A8,1X)) **Station List**

This group of records identifies the stations (and groups of stations) for which constants are desired. If KFILD(1) ≠ KFILDI, this group of records is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read within subroutine RDSTAD by RDC, which eliminates any blanks found in the input. Duplicate stations in the list for a group are kept, but a diagnostic is furnished on unit KFILDO. Note that this diagnostic applies only to duplicates within a group, not from group to group. For

NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty group terminates the station input. That is, the last group and its terminator must be followed immediately by another terminator signifying an empty group. Maximum number of stations, sans terminator, is ND1. ND1 is set in the driver DRU170. (CHARACTER*8)

Record Type 10 - Format (I3,4X,A60) **Variable Constants File**

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 10 is read by subroutine RDSNAM.

KFILCP - Unit number for the variable constants file.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 11 - Format **Input Variables**
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,11X,A1,I2,1XI1,1X30A1)

This group of records contains the ID's of the variables for which means or relative frequencies are to be calculated. Unlike many other MOS-2000 main programs, the variable list may not be taken from any source other than the control file U170.CN. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4 as set in the driver DRU170. More than one record will be read only in cases where constants for multiple categories of the same input variable are to be calculated. See the Restrictions section for important information on the number and ordering of records for input variables. This Record Type 11 is read by subroutine RDVR66; consequently the format and most of the information for ID(,) is retained from U660. This is true even though some of the parameters are either ignored or used for a slightly different purpose than in U660. This format is also compatible with the format for reading predictors in U201. Upon completion of reading this record type, N=1,NVRBL.

ID(J,N) - The first 3 (J=1,3) words of the input variable ID, plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed. Note: Any thresholds specified by Word 4 of ID(,N) are used directly by subroutine CAT170. Thus, the user must take care to specify precisely in Word 4 the value(s) desired for categorization of the input variable. See the Restrictions, below, for further discussion.

JP(J,N) - JP(J,N) indicates for J=1,3:
1 = Not used.
2 = Whether (>0) or not (=0) variable N will be written to Unit KFILDO under the format provided below. This feature is useful for quick viewing of the raw input data. Generally, this printout should not be turned on except for situations in which NPRINT of Record Type 2 is set for printing data from only a small number of input dates. Otherwise, setting JP(2,) ≠ 0 can result in voluminous output, especially when calculations are being performed at a large number of stations.
3 = Whether (>0) or not (=0) input variable N will be written to Unit KFILDO to the resolution packed. This is primarily for checking precision of the input data, and will not often be needed.

Note: The following parameters control print output for ID(,N) under JP(2,N) control as well as printing of the corresponding output constants IDOUT(,N) under KP(2) control (see Record Type 12, below). Printing of input variables and/or output constants is done only if JP(2,N) > 0 (for variables) or KP(2) > 0 (for constants). Since these parameters do "double duty" for the printing of both raw data and output constants, the user should take care to ensure that the values specified are adequate for both applications when printing of both data types is required. When checkout is complete and printing of only the output constants is likely to be necessary, then the user need only be concerned with specifying values for the following parameters which are compatible with the format and scaling of the output constants.

CFMT(N) - The format variable type descriptor for writing input variable and/or output constant N on Units KFILDO and/or KFILAS. Must be either "F" or "I".

IWDTH(N)- The field width for writing input variable and/or constant N to Units KFILDO and/or KFILAS. Must be $\leq$ 30.

IPREC(N)- The precision for writing data on Units KFILDO and/or KFILAS. For an "F" format, this is the number of digits after the decimal point. For an "I" format, it is the number of digits always written. Note that for a "zero" to be printed, IPREC( ) must be > 0; otherwise, zero will be printed as a blank.

HEAD(J,N) - The column heading for writing variable and/or output constant N under the above format control to Units KFILDO and/or KFILAS. Limited to J=1,30 characters. Only IWDTH(N) characters are read, regardless of the length of the string. HEAD(,) is right-justified when writing.

Record Type 12 - Format (2(I9,1X),28X,3I2) **Output Variable Identifier**

This record specifies the "basic" MOS-2000 ID of the output constants to be calculated from the variables listed in Record Type 11. Only one record should be present here, since only Words 1 and 2 of the output variable identifier will differ from ID(,N) above (see the Restrictions section, below). All constant data will have a CCC in Word 1 which falls in the range 400-499. Word 2 of the constant identifier will contain, if required, coded information which pertains to the time period over which the output constants are applicable (see Chapter 4 of TDL Office Note 00-1). Words 3 and 4 of the output variable identifiers are taken directly from the corresponding positions in ID(,N). Although Record Type 12 is not read by subroutine RDVR66, the format and placement of the information needed in this record is compatible to that for the input variables. A terminator of the form IDOUT(1,1) = 999999 is optional. While this final terminator is not actually read by subroutine INT170, the user may desire a final terminator as the last line of the control file U170.CN for consistency with the format of input variables or simply as an aesthetic indication of the end of input data.

IDOUT(J,N) - The first 2 (J=1,2) words of the MOS-2000 identifier of the output variable, as they are to appear in all printed and packed output. Words 3 and 4 of IDOUT are taken from the ID's of the input variables above with one-to-one correspondence, and need not be repeated here.

KP(J) - KP(J) indicates for J=1,3:

1 = Whether (>0) or not (=0) output constants are to be written to Unit KFILIO in TDLPACK format. This is the output which will be used to generate entries in TDL Random Access Constant Files (see Chapter 14 of TDL Office Note 00-1). Thus, if the purpose of a U170 run is to generate records for one or more TDL constant files, the user must ensure that KP(1) is nonzero. Output filename and location will be as specified by Record Type 8. The scaling factor and plain language will be taken from the variable constants file on Unit KFILCP.

2 = Whether (>0) or not (=0) output constants will be written to Unit KFILAS under the format provided with the output variable ID in Record Type 15. Generally, this is the primary output for viewing or printing.

3 = Whether (>0) or not (=0) all output constants will be written to Unit KFILDO to the resolution packed. This is primarily for checkout and quality control of data being written.

If KP(J) > 0, all output constants are printed for all categories, each output constant being associated with the threshold (Word 4) specified in the ID for its corresponding input variable (i.e., having the same value of N).

CONTROL FILE INPUT: (Name read from U170.CN) (Unit = KFILDT)

Record Type 1 - Format (I10/(7I10)) **Date List**

When the dates are not provided in file U170.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 3) = KFILDI (the input unit number as specified in DRU170), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U170 stops. Date/times cannot be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8, as set in the driver DRU170.

CONTROL FILE INPUT: (Name read from U170.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided in file U170.CN, this group of records identifies the stations (or locations) for which constants are to be calculated. This file is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from unit KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which constant values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1, as set in DRU170. (See Record Type 8, control file 'U170.CN', unit KFILDI for additional information.) (CHARAC-TER*8)

CONTROL FILE INPUT: (Name read from U170.CN) (Unit = KFILD(2))

Record Type 1 - Format **Station Locations**
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which constants are desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical within each group no matter the order read in (see NALPH in Record Type 2). [Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. Note that alphabetization is not overall, but by group (see Record Type 2)]. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the older 3-character call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. Either the new ICAO or older call letters can be used according to the value of NEW (see NEW in Record Type 2).

<u>CCALLD</u>(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

<u>NAME</u>(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

<u>NELEV</u>(K) - Elevation of the station. Format is I5.

<u>SIGNLA</u>(K) - Sign of the latitude of the station, read as either "S" or N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

<u>XLAT</u>(K) - Latitude in degrees. Format is F7.4.

<u>SIGNLO</u>(K) - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and increase westward across the United States. Format is A1.

<u>LONDD</u>(K) - Longitude in degrees west. Format is F8.4.

<u>CCALLD</u>(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

<u>IWBAN</u>(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited, and no terminator is used.

CONTROL FILE INPUT: (Name read from U170.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally usable by all U170 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that of the variable constants file read by other programs in the MOS-2000 system.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=2,4) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the data for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer.

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDVR66, much of it associated with the plain language description. See Chapter 4, Variable Identification, of TDL Office Note 00-1 for details.

DATA INPUT:

All data input to U170 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note 00-1). Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. One or more sources of vector data, which can be prepared by U201 (J=1,NUMIN), are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U170.CN', Record Type 5. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each data source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

DATA OUTPUT:

Output that can be printed is put onto Unit KFILDO as described above under control file 'U170.CN' and the definition of KFILDO in the driver DRU170. All errors will be to the default output file. Every effort has been made to notify the user of problems and potential problems and to proceed under user control. There are three forms of data output, besides the diagnostics and other information on Unit KFILDO:

A. ASCII FOR VIEWING OR PRINTING

A maximum of NPRINT (see Record Type 2) or NDATES (see Record Type 4) cycles of data will be written to the file on Unit KFILAS under control of the format provided with each variable N, the variables written controlled by KP(2) (see Record Type 12). The data columns are headed by HEAD(,N) (see Record Type 11). Listing is by station group. If a line length for printing is not sufficient for all variables, then more than one line is used. Line length is specified as LNGTH characters in Record Type 2.

B. ASCII FOR STATIONS, CONSTANT IDS, AND CONSTANTS FOR U351

When KFILAC ≠ 0, information for input to U351 is written to Unit No. KFILAC (see Record Type 6a). This consists of:

(a) Station list - Format (7(A8,1x))

This is the same list of stations read in Record Type 9, CCALL(K), K=1,NSTA, followed by a terminator '99999999'.

(b) Four constant IDs, ISCALE, ICHAR, character definition - Format - (I9,3I10.9,2I4,A32)

See Record Type 12 for a description of the IDs. ISCALD = 3 for the constants, ICAR = 32, and character definition is 'CONSTANT DATA FOR EACH STATION'.

(c) Constants - Format (7F9.3)

Constants for each station for one variable definition followed by a terminator '999999'.

(b) and (c) then repeat for all IDs and constants, followed by a final terminator '999999' (two terminators at the end).

C. BINARY MOS-2000 FORMAT

Constant data for all categories of the input variable will be packed in TDLPACK format and written to unit number KFILIO to the dataset whose name has been provided to OUTNAM (see Record Type 7), unless KFILIO = 0 or KP(1) = 0 in which case data are not output. This feature can be used for checkout or if the user merely wants constants calculated for printing. The data are packed with subroutine PACK1D and its associated subroutines. All constant data and computed variables associated with constant data must have a CCC in the range 400-499.

A primary purpose of U170 is to provide packed vector constant data for other programs, generally via the MOS-2000 External Random Access File System (See Chapter 14 of TDL Office Note 00-1). After generating the TDLPACK output in U170, it is expected that the user will need to write the TDLPACK constant values into one or more Random Access Constant Files. This will be accomplished by subsequently running the programs U350 and U352 on the output data written to Unit KFILIO.

<u>EXAMPLE CONTROL FILE</u>: 'U170.CN'

An example exists as file 'U170.CN' in directory /home21/tdllib/dru170 on blizzard. The easiest way to set up a run is to take an existing control file and modify it; <u>DO NOT START FROM SCRATCH</u>.

<u>RESTRICTIONS</u>:

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. Some machines may not allow this. It is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily. Some compilers treat D and C for Comment, so no change is needed.

It is not necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles, <u>unless</u> the variable is computed from other variables not otherwise used as a "raw" variable. That is, a raw (not computed) variable need not be present for Day 1, but one used <u>only</u> in computations must be present because U170 has no way of knowing that the variable will be needed for future cycles.

<u>Special Note on Thresholds and Constant Variable ID's</u>: The binary indicator "B" in the first word of the input ID of Record Type 14 can be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), 2 (indicating a cumulative variable from below), or 3 (indicating a discrete variable).

For the latter, the <u>upper</u> threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID. In other words, U170 functions as though dealing with a set of variables that are cumulative from below. If there is no lower threshold, the lower one is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999. Since the variables are not ordered by the program (as they are in U600), if constants from a number of discrete binary variables are desired (i.e., with both upper and lower thresholds, often called "exclusive" categorical events), the variable ID's must be input such that the thresholds (word 4) of each ID are in sequence and in the correct order (lower threshold first). In addition, in order to make sure that the upper category completely "exhausts" the entire range of possible input variable values, a variable with a threshold of 99999 is required at the end of the list. When these constants are written out, constants for that event which is cumulative from below the first threshold entered will be output first. Because of the one-to-one correspondence between identifiers for the input variables and output constants, the constant for this first event will be given a MOS-2000 identifier, IDOUT(,1), having the same word 4 as its corresponding input variable ID(,1). Likewise, the last constant output will pertain to input variable values exceeding the last physically meaningful threshold as entered in ID(,N-1). The MOS-2000 identifier of this last output constant will, in reality, have 99999 appearing in IDOUT(4,N). To make sure all is well, check the thresholds as they are echoed on unit KFILDO.

Any thresholds specified in Word 4 of ID(,N) are used directly by subroutine CAT170. Thus, the user must take care to specify precisely in Word 4 the value(s) desired for categorization of the input variable. This may become important in situations where the user wishes to ensure against problems associated with roundoff error in the input variable as values are retrieved from the unpacking routines. For instance, instead of checking against a category bounded by the value X, the user may wish to specify a value of X plus or minus some small offset in Word 4 of the input variable ID so that values of the variable which lie very near the categorical threshold will fall in the proper "bin" when unpacked and subsequently compared against the Word 4 breakpoint in CAT170. For instance, if the user wishes to calculate the relative frequency of observed precipitation greater than or equal to 1.00 inches, it would be advisable to specify a threshold in Word 4 equal to some real value slightly less than 1.00, say 0.9995, to insure that all packed observations of exactly 1.00 inches are included in the calculation. To reiterate, it is important to enter in Word 4 the precise breakpoint value to be checked in CAT170. For the sake of consistency, and because these breakpoints are part of the MOS-2000 variable identifier for the output constants (which later will presumably used as predictors in regression equations), the Word 4 breakpoint values also should be identical to the breakpoints used in U600 during equation development.

COMMENTS

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U170 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there were insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 2). Presumably, the only time 9997 will occur <u>on input</u> is when constants are to be derived from input operational forecasts or U700-produced test forecasts. Other values can be packed as "missing" and will, of course, be returned by the unpacker as such.

U170 will return a value of 9997 <u>as output</u> in situations where mean values for categorical events cannot be calculated for a station (or group of stations) due to the fact that no cases were observed to fall within a particular specified category, but values are otherwise available for this station or group. This will only occur when the particular category is not "exhaustive" (i.e., does not account for all possible values of the input variable); the value of 9997 is used to distinguish this situation from others in which observations of the input variable are truly non-existent at the station or group of stations over the period specified by the input date list.

There will also be occasions where the input TDLPACK data may contain special values (such as "888" for an unlimited ceiling, or "0.004" for a trace of precipitation) which need modification before they can be included in calculations of mean values or relative frequencies. In most cases, these values will probably need to be set to "0" before proceeding with constant calculations. Modifications of this type are handled in subroutine MDFY170. In this subroutine, modifications for each variable requiring special treatment are handled by a unique branch of a FORTRAN IF-THEN-ELSEIF construct. When constants from "new" variables (i.e., those previously not included in the MOS-2000 system) are desired, the user should make sure to check that no special values (other than 9999 or 9997) are encoded in the packed data. If special values are present, then the user will be required to enter the appropriate MOS-2000 identifier for the input variable in a DATA statement at the beginning of subroutine MDFY170, as well as add a corresponding ELSEIF block in the code to perform the necessary data modifications.

<u>Note</u>: When the mean or relative frequency of a grid binary variable is desired it <u>cannot</u> be computed by U170 because this, in essence, demands a

duality of usage for the threshold of the MOS-2000 identifier which is not possible. The reason for this is that Word 4 would need to be used to identify how the grid binary should be created from the original meteorological variable as well as in U170 to determine the threshold for calculation of a constant based on the binary. This type of application is not supported by the MOS-2000 system.

Most errors are output for printing starting with **** to KFILDO. A count is kept for matching with NSKIP and JSTOP (see Record Type 2). Missing variables (but not just a missing value for a station) will usually be counted as an error, and a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors.

On Day 1, GFETCH is entered directly for every variable because it is not yet known when OPTX must be entered. ("Day 1 is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) is not counted as an error in VRBL61 for Day 1. It is counted as an error in VRBL62 (through OPTX), because GFETCH should not be entered directly when OPTX is needed.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage after Day 1.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 9 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list. (In the case of development for gridpoints, the gridpoints are given identifications and treated in the same way as stations.)

SETTING UP THE DRIVER DRU170

The preparation of the driver for a particular U170 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the

directory (read on Unit No. KFILD(2)) unless, of course, the station directory is to be used as the station list (see ND5).

ND2 - Not used.

ND3 - Not used.

ND4 - The maximum number of variables that can be dealt with. In the case of U170, this means the maximum number of categories of the "basic" input variable.

ND5 - Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.

ND6 - Maximum number of all sequential file input sources that can be dealt with. If data from a model are on two files, then this would be counted as two, not one, even though the same unit number might be used.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU170 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

NONSYSTEM ROUTINES USED

On the HP (blizzard), nonsystem MOS-2000 subroutines are available in /home21/tdllib/moslib.

LANGUAGE: FORTRAN77 with some FORTRAN-90 compliant HP extensions.

LOCATION: u170lib. The driver is in dru170. The complete path for U170 code and U170-associated subroutines is /home21/tdllib/u170lib on the HP. The driver (dru170.f) is in /home21/tdllib/dru170 on the HP.

U171

COMPUTES CONSTANTS FROM VECTOR DATA

Harry R. Glahn
November 1, 2003

PURPOSE: U171 calculates constants from MOS-2000 vector data. These constants can be either means or relative frequencies and will be calculated for specific stations or groups of stations usually from observations or model data. These means or relative frequencies will ordinarily then be used as predictors in regression equations, or to assist in regionalization during equation development. Constants can be computed over the entire range of possible values of the input variable or for exclusive or cumulative categories of variable values. Categories are defined by the values appearing in word 4 of the MOS-2000 identifier. U171 prints the output constants in an easy-to-read ASCII format, and packs and writes the constants as TDLPACK vector output for the stations designated by control files to a "sequential" file and/or to a MOS-2000 random access file. Input data must be in TDLPACK format, and because the packed data are self-describing, the input data can come from an essentially unlimited number of sources. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters or gridpoint identifiers. U171 uses a driver DRU171 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program for every application. U171 is written to run on a 32-bit or a 64-bit word-length machine, which also is controlled by PARAMETER statements in the driver.

Many of the design features, subroutines, and variable names used in U171 were adapted from program U660. Some familiarity with other MOS-2000 documents, especially those pertaining to U660, will be helpful to full understanding of this writeup.

CONTROL FILE INPUT: 'U171.CN' (Unit = KFILDI)

KFILDI is the input unit number as specified in DRU171. As with other MOS-2000 programs, the control file U171.CN must reside in the same directory from which the main program DRU171 is being run. The control file is opened by the driver DRU171 with a standard FORTRAN OPEN statement.

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU171.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U171" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U171 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U171HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U171 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). Output has been defined for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU171 DATA statement = 12. Changing the default unit number allows multiple runs of U171 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.
(7) = The variable ID list in summary form including parsed IDs in IDPARS(,)--the 15 components of each ID). If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) = The variable ID list in summary form including parsed IDs and both lower and upper thresholds.
(9) = The variable list in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).
(10) = The variable ID's for the first day (day 1) as read from the archive tapes. This is just a list of all the variables on the input files.
(11) = The variable ID's of the archived data actually needed, in order as they appear on the archive files for day 1.
(12) = The list(s) of stations on the input file(s).
(13) = Not used.
(14) = A diagnostic will be provided when there are no data for a particular input file for a particular date/time. With tape switching, this may not be of much use, and could be misleading.
(15) = Data written in the order packed for each variable indicated by JP(3,) > 0. This is separate from the optional writing associated with JP(2,). Except for a very few days, this would produce voluminous files.
(16) = All or a portion of the data values in the AA(, ,) matrix (N=1,NVRBL) (K=1,NSTA) (L=1 and 3). For each case (cycle), after all input data have been read, the AA(, ,) matrix holds all variable

values for all stations. The print is by variable controlled by JP(2,) (see Record Type 15 below), then the station values are printed in the order dealt with in U171 (see IP(5)). Except for a very few days or a few variables and stations, this would produce voluminous files.

(17) = Quality control information (see Record Type 15, FPHPT1(,), etc.).

(18) = The two maximum and minimum values of each variable in each group of stations.

(19) = All computed data.

(23) =

Information concerning opening and closing of files.

(24) = Information as to where the variables are to be found--directly on input, binary computed from a previous variable, or (at least attempted to be) computed through subroutine OPTX.

For checkout, it may be advisable to set all these values to the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (5(I10/),F10.0/,3(I10/),I10) **Run Control Parameters**

This record type contains values which provide control over various basic features of each program run. Note that each value is on a separate line. A brief explanation can be put on the same line with each variable to assist the user in knowing which value is for which variable.

NSKIP - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - Determines the increment between dates when date spanning is used. INCCYL is the number of hours between date/times that are put into IDATE( ) (see Record Type 4), as a result of date spanning in subroutine DATPRO. Date/times cannot be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 0600 UTC should never be indicated.

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the older (3-character) call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section). Also NEW indicates whether the IDs are gridpoint and duplicates can be accepted (=2) or not accepted (=3). It is important to be correct on this point, because searching on a large number of gridpoints for a large grid is VERY time consuming.

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense. NALPH is not used when NEW = 2 or 3.

PXMISS - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value.

NPRINT - The number of cycles of raw input data to write for printing to unit KFILDO (as specified in the driver DRU171) under the format control and JP(2,) provided with each variable (see Record Type 16). This provides an easy way to examine the raw data for one or more input date/time groups.

ICHARS - The number of characters of call letters to print when printing is indicated by JP(2,). This is constrained to be between 4 and 8 inclusive.

LNGTH - The line length for printing the above raw data. For a line printer, 132 is appropriate; for a laser printer, 80 may be desirable.

NWDATE - Date/time to write in packed recored, format yyyymmddhh.

Record Type 4 - Format (I3,4X,A60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and the string "DEFAULT" can be read in its place. (CHARACTER*60)

Record Type 5 - Format (7I10) **Date List**

This group of records determines, together with the parameter INCCYL in Record Type 3, the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8 as set in the driver DRU171.

Record Type 6 - Format (I3,4X,A60) <u>Input Vector Data Files</u>

This group of records identifies the data sets from which the input vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. (ND6 is set in the driver DRU171.) Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

<u>KFILIN</u>(J) - Unit number for the data file.

<u>NAMIN</u>(J) - Name of file where these data reside. (CHARACTER*60)

Input vector data will usually be weather observations or numerical model data. These data must be packed in the TDLPACK format. When data sets are to be used in sequence, they should be read in the proper chronological order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and the earliest year is on one data set and the other, more recent year is on a second data set, then the first should immediately precede the second in the list and both should have the same unit number. If there are <u>no</u> needed data for the first data available on a file, the file is closed and is not available for the remainder of the run.

Record Type 7 - Format (I3,4X,A60) <u>ASCII Output File for U351</u>

This record (plus the terminator record) identifies the data set which will contain a listing of the stations as well as the output constants in a format suitable for input to u351. Records are read until the terminator KFILAC = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 7 is read by subroutine RDSNAM and will be opened as 'NEW'. This output file is optional; if ASCII output for U351 is not desired, only the terminator need be present.

<u>KFILAC</u> - Unit number for the ASCII output file.

<u>ASCIFM</u> - Name of file for ASCII output. (CHARACTER*60)

Record Type 8 - Format (I3,4X,A60) <u>ASCII Output File</u>

This record (plus the terminator record) identifies the data set which will contain a tabular printout of output constants with colon separators, with stations grouped as specified by the station list (see Record Type 12) and with or without intra-group alphabetization as specified by the parameter NALPH in Record Type 3. Records are read until the terminator KFILAS = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM and will be opened as 'NEW'. This output file is optional; if tabular ASCII output is not desired, only the terminator need be present.

<u>KFILAS</u> - Unit number for the ASCII output file.

<u>ASCIFL</u> - Name of file for tabular ASCII output of constants. (CHARACTER*60)

Record Type 9 - Format (I3,4X,A60) <u>Packed Output File</u>

This record (plus the terminator record) identifies the packed vector output file containing the computed means and/or relative frequencies. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1.

This Record Type 9 is read by subroutine RDSNAM, and the file will be opened as 'NEW'. This file is written with a directory record as a "sequential" file, although the date, being the last date/time in the sample designated in IDATE( ), may not have much relevance. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

KFILIO - Unit number for the packed output.

OUTNAM - Name of file where packed constants are to reside. (CHARACTER*60)

Record Type 10 - Format (I3,4XA60) Random Access Files

This group of records identifies the MOS-2000 random access data sets from (to) which data are read (written). Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 10 is read by subroutine RDSNAM. If no data are to be read from or written to a random access file, only the terminator is necessary. It is not expected reading will be done; however, the calculated data can be written here by using KFILRA( ) = 49.

KFILRA(J) - Unit numbers for the random access constant files (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

RACESS(J) - Names of files of constant data (J=1,NUMRA). (CHARACTER*60)

Record Type 11 - Format (I3,4X,A60) Station List and Directory Files

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 11 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT" (CHARACTER*60), in which case all station directory information is taken from the control file U171.CN. (It would be unusual for KFILD(2) to equal KFILDI.)

Record Type 12 - Format (7(A8,1X)) Station List

This group of records identifies the stations (and groups of stations) for which constants will be computed. If KFILD(1) ≠ KFILDI, this group of records is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). When NEW = 0 or 1, this list is read with subroutine RDSTGN or RDSTGA, depending on NALPH, by RDC, which eliminates any blanks found in the input. Duplicate stations in the list for a group are kept, but a diagnostic is furnished on unit KFILDO. Note that this diagnostic applies only to duplicates within a group, not from group to group. For NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. Alternatively, when NEW = 2 or 3, the gridpoint list is read with subroutines RDSTGD or RDSTGF, respectively. The call letters (or gridpoint values) should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty group terminates the station input. That is, the last group and its terminator must be followed immediately by another terminator signifying an empty group. Maximum number of stations, sans terminator, is ND1. ND1 is set in the driver DRU171. (CHARACTER*8)

Record Type 13 - Format (I3,4XA60) Variable File

This record (plus the terminator record) identifies the file from which the variable ID's are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 13 is read by subroutine RDSNAM.

KFILP -	Unit number for the variable ID's.
PRENAM -	Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 14 - Format (I3,4X,A60) Variable Constants File

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description and most importantly the scaling value for packing the computed values.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 14 is read by subroutine RDSNAM.

KFILCP -	Unit number for the variable constants file.
CONNAM -	Name of file corresponding to KFILCP. (CHARACTER*60)

The next group of records are read in trios, each being the IDs and other information of: (1) the variable for which the relative frequencies or means are to be calculated, (2) the computed result, and (3) a "stratification" variable. Each of the trio has the same format, although not all information is used from all three.

Record Type 15 - Format Input Variables
(I9,1X,I9,1X,I9,1X,I3,1X,A1,1X,I4,1X,I3,4X,3I2,I3,6F6.3,
2I5,2I3,1X,A1,I2,1X,I1,1X,30A1)

The first of each trio contains the ID of the variable for which means or relative frequencies (these together are called "constants") is to be calculated. Records are read until the terminator ID(1, ,) = 999999 is reached. Maximum number of trios, sans the terminator record, = ND4 as set in the driver DRU171. More than one trio will be read when (1) constants for multiple categories of the same basic variable are to be calculated, and/or (2) when constants of different basic variables are to be calculated. Any mix of these is allowed. See the Restrictions section for information on the ordering of records for multiple categories of the same variable. This Record Type 15 is read by subroutine RDVR171. This format is quite compatible with the format for reading variable IDs in other MOS-2000 programs. Upon completion of reading this record type, N=1,NVRBL and L=1,3.

<u>ID</u>(J,N,L) - The first 3 (J=1,3) words of the input variable ID, plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

<u>JP</u>(J,N,L) - (Applies to L = 1 only) indicates for J=1,3:

1 = Whether (>0) or not (=0) the computed data will be packed and written to a "sequential" file and/or to a MOS-2000 external random access file, depending on the unit numbers KFILIO and KFILRA being non zero and 49, respectively.

2 = Whether (>0) or not (=0) variable N will be written to Unit IP(16) with the format specified below. This feature is useful for quick viewing of the raw input data. Generally, this printout should not be turned on except for situations in which NPRINT of Record Type 3 is set for printing data from only a small number of input dates. Otherwise, setting JP(2, ,) ≠ 0 can result in voluminous output, especially when calculations are being performed at a large number of stations.

3 = Whether (>0) or not (=0) input variable N will be written to Unit IP(15) to the resolution packed. This is primarily for checking precision of the input data, and will not often be needed.

Note: Since JP(, ,) is actually used for only L = 1, the values for L = 2 and 3 can be zero or blank, but the format indicated above is maintained.

<u>ITAU</u>(N,L) - The number of hours to add to NDATE (the date being processed, see Record Type 5) to get the variable N. That is, the tau in the ID is left intact, and ITAU is used to "look ahead." Applies to L = 1 and 3. However, if there is an ITAU(,3) > all ITAU(,1), probably some data will not be accessed.

<u>FPHPT1</u>(N,L) - (Applies to L = 1 only) A value of the variable N which when exceeded the user will be notified. Set to -9999 to disable.

<u>FPHPT2</u>(N,L) - (Applies to L = 1 only) A value of the variable N which when exceeded the user will be notified. Set to -9999 to disable. FPHPT1(,) and FPHPT2(,) give the user two levels of notification on values being higher than might be expected.

FPLPT1(N,L) - (Applies to L = 1 only) A value of the variable N which when not exceeded the user will be notified. Set to -9999 to disable.

FPLPT2(N,L) - (Applies to L = 1 only) A value of the variable N which when not exceeded the user will be notified. Set to -9999 to disable. FPLPT1(,) and FPLPT2(,) give the user two levels of notification on values being lower than might be expected.

FPH999(N,L) - (Applies to L = 1 only) A value of the variable N which when exceeded the computed value will be set to 9999. Set to -9999 to disable.

FPL999(N,L) - (Applies to L = 1 only) A value of the variable N which when not exceeded the computed value will be set to 9999. Set to -9999 to disable.

NLPT(N,L) - (Applies to L = 1 only) A value for the number of cases for variable N which when not exceeded the user will be notified. Set to -9999 to disable.

NL99(N,L) - (Applies to L = 1 only) A value for the number of cases for variable N which when not exceeded the computed value will be set to 9999. Set to -9999 to disable.

IBHR(N,L) - (Applies to L = 1 only) The beginning hour over the 24-h period (0 through 23) for which the relative frequency or mean will be calculated.

IEHR(N,L) - (Applies to L = 1 only) The ending hour over the 24-h period (0 through 23) for which the relative frequency or mean will be calculated. When IEHR(N,) < IBHR(N,), the hours IBHR(N,) through 23 and the hours 0 through IEHR(N,) will be used. Both IBHR(,) and IEHR(,) apply to the basic date/time being processed, not with ITAU(,) applied.

CFMT(N,L) - (Applies to L = 1 only) The format variable type descriptor for writing basic variable N on Units KFILDO and/or KFILAS. Must be either "F" or "I".

IWDTH(N,L) - (Applies to L = 1 only) The field width for writing basic variable N to Units KFILDO and/or KFILAS. Must be $\leq$ 30.

IPREC(N,L)- (Applies to L = 1 only) The precision for writing data on Units KFILDO and/or KFILAS. For an "F" format, this is the number of digits after the decimal point. For an "I" format, it is the number of digits always written. Note that for a "zero" to be printed, IPREC( ) must be > 0; otherwise, zero will be printed as a blank.

HEAD(J,N,L) - (Applies to L = 1 and 3 only) The column heading for writing variable N under the above format control to Unit IP(16). Limited to J=1,30 characters. Only IWDTH(N,L) characters are read, regardless of the length of the string. HEAD(, ,) is right-justified when writing.

CONTROL FILE INPUT: (Name read from U171.CN) (Unit = KFILDT)

Record Type 1 - Format (I10/(7I10)) **Date List**

When the dates are not provided in file U171.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 3) = KFILDI (the input unit number as specified in DRU171), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U171 stops. Date/times cannot be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8, as set in the driver DRU171.

CONTROL FILE INPUT: (Name read from U171.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) Station List

When the station list is not provided in file U171.CN, this group of records identifies the stations (or locations) for which constants are to be calculated. This file is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from unit KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which constant values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1, as set in DRU171. (See Record Type 12, control file 'U171.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U171.CN) (Unit = KFILD(2))

Record Type 1 - Format Station Locations
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which constants are desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical within each group no matter the order read in (see NALPH in Record Type 3). [Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. Note that alphabetization is not overall, but by group (see Record Type 3)]. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the older 3-character call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. Either the new ICAO or older call letters can be used according to the value of NEW (see NEW in Record Type 2). When the "stations" are gridpoint locations, the order should be numerical (e.g., 00010001, 00010002, etc.); otherwise, search time is excessive for large numbers of stations.

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA(K) - Sign of the latitude of the station, read as either "S" or N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT(K) - Latitude in degrees. Format is F7.4.

SIGNLO(K) - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and increase westward across the United States. Format is A1.

LONDD(K) - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited, and no terminator is used.

CONTROL FILE INPUT: (Name read from U171.CN) (Unit = KFILP)

Record Type 1 - Format Input Variables
(I9,1X,I9,1X,I9,1X,I3,1X,A1,1X,I4,1X,I3,4X,3I2,I3,6F5.3,
2I4,2I3,1X,A1,I2,1X,I1,1X,30A1)

When the variables to use are not in file 'U171.CN', this group of records contains the variable ID's exactly as specified in Record Type 15 above. When KFILP = KFILDI, this file is omitted, and the variable list is taken from input file KFILDI. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records of each of the three types, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDVR171. Upon completion of reading this record type, N=1,NPRED. See Record Type 16, file 'U171.CN', unit KFILDI, for other details; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U171.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally usable by all U171 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that of the variable constants file read by other programs in the MOS-2000 system.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=2,4) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the data for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer.

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDVR66, much of it associated with the plain language description. See Chapter 4, Variable Identification, of TDL Office Note 00-1 for details.

DATA INPUT:

All data input to U171 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note 00-1). Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. One or more sources of vector data, which can be prepared by U201 (J=1,NUMIN), are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U171.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each data source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

DATA OUTPUT:

Output that can be printed is put onto Unit KFILDO as described above under control file 'U171.CN' and the definition of KFILDO in the driver DRU171. All errors will be to the default output file. Every effort has been made to notify the user of problems and potential problems and to proceed under user control. There are three forms of data output, besides the diagnostics and other information on Unit KFILDO:

A. ASCII DATA FOR VIEWING OR PRINTING

A maximum of NPRINT (see Record Type 3) or NDATES (see Record Type 5) cycles of data will be written to the file on Unit KFILAS under control of the format provided with each variable N, the variables written controlled by JP(,2) (see Record Type 15). The data columns are headed by HEAD(,N) (see Record Type 15). Listing is by station group. If a line length for printing is not sufficient for all variables, then more than one line is used. Line length is specified as LNGTH characters in Record Type 3.

B. ASCII CONSTANTS FOR VIEWING OR PRINTING

When IP(19) ≠ 0, all computed constants are listed by computed variable IDs and by station or by group to Unit No. IP(19). If listing is by group, the group is identified by the first station in the group, as well as group number.

C. ASCII FOR STATIONS, CONSTANT IDS, AND CONSTANTS FOR U351

When KFILAC ≠ 0, information for input to U351 is written to Unit No. KFILAC (see Record Type 7). This consists of:

(a) Station list - Format (7(A8,1x))

This is the same list of stations read in Record Type 12, CCALL(K), K=1,NSTA, followed by a terminator '99999999'.

(b) Four constant IDs ID(, ,2), ISCALE(,2), ICHAR(,2), character definition - Format - (I9,3I10.9,2I4,1X,A32)

See Record Type 15 for a description of the IDs.

(c) Constants - Format (7F9.3)

Constants for each station for one variable definition followed by a terminator '999999'.

(b) and (c) then repeat for all IDs and constants, followed by a final terminator '999999' (two terminators at the end).

D. BINARY MOS-2000 FORMAT

Computed constant data will be put into TDLPACK format and written to unit number KFILIO to the dataset whose name has been provided to OUTNAM (see Record Type 9), unless KFILIO = 0 in which case data are not output. The data are packed with subroutine PACK1D and its associated subroutines. All constant data and computed variables associated with constant data must have a CCC in the range 400-499 (see 2nd of trio in Record Type 15 on U171.CN).

A primary purpose of U171 is to provide packed vector constant data for other programs, generally via the MOS-2000 External Random Access File System (See Chapter 14 of TDL Office Note 00-1). After generating the data in U171, the user can write them onto a MOS-2000 External Random Access File with U352 from the data written to Unit KFILIO, or alternatively the data can be written directly to a MOS-2000 External Random Access file by specifying KFILRA( ) = 49 and JP(1,N,2) = 1 (see E. immediately below).

E. <u>RANDOM ACCESS BINARY MOS-2000 FORMAT</u>

Data for each computed variable N for which JP(1,N,1) > 0 (see Record Type 15) will be put into in TDLPACK format and written to unit number KFILRA( ) = 49, (only) if a file with that number has been provided in Record Type 10. The data are packed with subroutine PACK1D and its associated subroutines.

<u>EXAMPLE CONTROL FILE</u>: 'U171.CN'

An example exists as file 'U171.CN' in directory /home21/tdllib/dru171 on blizzard. The easiest way to set up a run is to take an existing control file and modify it.

<u>RESTRICTIONS</u>:

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. Some machines may not allow this. It is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily. Some compilers treat D and C for Comment, so no change is needed.

It is not necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles, <u>unless</u> the variable is computed from other variables not otherwise used as a "raw" variable. That is, a raw (not computed) variable need not be present for Day 1, but one used <u>only</u> in computations must be present because U171 has no way of knowing that the variable will be needed for future cycles. However, if <u>no</u> data from a particular file is found for day 1, <u>the file is closed</u> and is not available for the rest of the run

<u>Special Note on Thresholds and Constant Variable ID's</u>: The binary indicator "B" in the first word of the input ID of Record Type 15 can be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), 2 (indicating a cumulative variable from below), or 3 (indicating a discrete variable).

For the latter, the <u>upper</u> threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID. In other words, U171 functions as though dealing with a set of variables that are cumulative from below. If there is no lower threshold, the lower one is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999. Since the variables are not ordered by the program (as they are in U600), if constants from a number of <u>discrete</u> binary variables are desired (i.e., with both upper and lower thresholds, often called "exclusive" categorical events), the variable ID's must be input such that the thresholds (word 4) of each ID are in sequence and in the correct order (lower threshold first). In addition, in order to make sure that the upper category completely "exhausts" the entire range of possible input variable values, a variable with a threshold of 99999 is required at the end of the list.

For gridpoint development, the variable NEW indicates whether the IDs are gridpoint and duplicates can be accepted (=2) or not accepted (=3). It is important to be correct on this point, because searching on a large number of gridpoints for a large grid is VERY time consuming. The readers have been optimized for these two possibilities, and assume the directory is in numerical order (e.g., 00010001, 00010002, etc., not 00010001, 00020001, etc.).

COMMENTS

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U171 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there were insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 2). Presumably, the only time 9997 will occur on input is when constants are to be derived from input operational forecasts or U700-produced test forecasts. Other values can be packed as "missing" and will, of course, be returned by the unpacker as such.

There will also be occasions where the input TDLPACK data may contain special values (such as "888" for an unlimited ceiling, or "0.004" for a trace of precipitation) which need modification before they can be included in calculations of mean values or relative frequencies. In most cases, these values will probably need to be set to "0" before proceeding with constant calculations. Modifications of this type are handled through OPTX. When constants from "new" variables (i.e., those previously not included in the MOS-2000 system) are desired, the user should make sure to check that no special values (other than 9999 or 9997) are encoded in the packed data.

Note: When the mean or relative frequency of a grid binary variable is desired it cannot be computed by U171 because this, in essence, demands a duality of usage for the threshold of the MOS-2000 identifier which is not possible. The reason for this is that Word 4 would need to be used to identify how the grid binary should be created from the original meteorological variable as well as in U171 to determine the threshold for calculation of a constant based on the binary. This type of application should not be needed and is not supported by the MOS-2000 system.

Most errors are output for printing starting with **** to KFILDO. A count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables (but not just a missing value for a station) will usually be counted as an error, and a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors.

On Day 1, GFETCH is entered directly for every variable because it is not yet known when OPTX must be entered. ("Day 1 is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) <u>is not</u> counted as an error in VRBL66 for Day 1. It <u>is</u> counted as an error in VRBL67 (through OPTX), because GFETCH should not be entered directly when OPTX is needed.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days (hardwired in subroutine GCPAC). This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1. Even though this careful use of memory seems logical, experience indicates it is not of great importance because of the paging and cashing capabilities of modern workstations.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 12 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list. (In the case of development for gridpoints, the gridpoints are given identifications and treated in the same way as stations.)

When any one of the three types of variables is repeated in the list specified in Record Type 15 is repeated a warning is issued. This may not be an error, but is provided for the user to check for correctness.

<u>SETTING UP THE DRIVER DRU171</u>

The preparation of the driver for a particular U171 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) unless, of course, the station directory is to be used as the station list (see ND5).

ND2 - Not used.

ND3 - Not used.

ND4 - The maximum number of variables of each of 3 "types" that can be dealt with.

ND5 - Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1 and must be $\geq$ the largest number of stations on any input.

ND6 - Maximum number of all sequential file input sources that can be dealt with. If data from a model are on two files, then this would be counted as two, not one, even though the same unit number might be used.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records on all input files for that day.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number may result in wasted memory and possible excess paging (although modern computers may also alleviate this).

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU171 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

NONSYSTEM ROUTINES USED

On the HP (blizzard), nonsystem MOS-2000 subroutines are available in /home21/tdllib/moslib.

LANGUAGE: FORTRAN77 with some FORTRAN-90 compliant HP extensions.

LOCATION: u171lib. The driver is in dru171. The complete path for U171 code and U171-associated subroutines is /home21/tdllib/u171lib on the HP. The driver (dru171.f) is in /home21/tdllib/dru171 on the HP.

U135

CONVERTS TDLPACK DATA INTO GRIB2

David E. Rudack
January 1, 2005

PURPOSE: U135 is designed to convert TDLPACK data into GRIB2. U135 supports the conversion of data on North polar stereographic, Lambert conformal, and Mercator map projections. U135 accommodates one external gridded random access file as data input. U135 also accommodates a bitmap field as input in order to “mask” portions of the processed grid(s) prior to being packed into GRIB2. U135 outputs a GRIB2 file according to the grid specifications set forth in the grid definition file (see below). While U135 currently supports specific Product Definition and Data Representation Templates, other templates may be added if necessary. U135 uses a driver DRU135 so that dimensions of variables may be tailored by PARAMETER statements by the user without requiring a separate copy of the program U135 for every application. U135 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. Some familiarity with GRIB2 Document FM 92 GRIB, and other MOS-2000 documents will be necessary for a full understanding of this write-up.

CONTROL FILE INPUT: 'U135.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) **Output Control**

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU135.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U135" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U135 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'DER2' and IP(2) = 40, the file name for Unit No. 40 = 'U135DER240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U135 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. Values have been defined as indicated below for values of J:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output

file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU135 DATA statement = 12. Changing the default unit number allows multiple runs of U135 or other programs within the same directory without overwriting.

(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(6) = The bitmap variable as it is being read in.
(7) = The bitmap in variable ID in summary form. If there are errors, the ID will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) = The bitmap variable ID in parsed form which includes IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)
(9) = The bitmap variable ID in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID in IDPARS(,), but rather includes the information taken from the MOS-2000 constant file on unit KFILCP (see below).
(12) = The list of values pertaining to sections 0 and 1 that are set in the GRIB2 indicator and identification input file.
(13) = The list of values pertaining to section 2 that are set in the GRIB2 local use section input file.
(14) = The list of values pertaining to section 3 that are set in the GRIB2 Grid Definition input file.
(15) = The list of values pertaining to section 4 that are set in the GRIB2 Product Definition input file.
(16) = The list of values pertaining to section 5 that are set in the GRIB2 Data Representation input file.
(17) = ASCII output of the TDLPACK data field in tabular form. This file may be large for high resolution grids.
(18) = ASCII output of the TDLPACK bitmap data field in tabular form. This file may be large for high resolution grids.
(4),(5),(10),(11),(19)-(25)= Not used.

For checkout, it may be advisable to set all these values to zero or the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) **Run Identification**

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 – (I10/,4X,A6/,F10.4/,I10/,F10.4) **Control Parameters**

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

INCCYL - The increment in hours between date/times that are put into IDATE( ) (see Record Type 5) as a result of date spanning in subroutine DATPRO.

MPROJ - The six character grid designation identifier. The first two characters identify the map region. The following two characters identify the type of map projection and the final two characters represent the mesh length of the processed grid.

XMISSS - The secondary value assigned to the gridpoint data field after it has been unpacked from TDLPACK. In most instances, XMISSS is set to a value of 9997.

JBMAP - Flag indicating whether the supplied bitmap will be packed into the GRIB2 message (1=yes, 0=no).

BMAPVAL - The value assigned to the input TDLPACK data field if the bitmap indicates a gridpoint value should be "masked." The bitmap, containing 0' and 1's, is supplied by the user. (see Record Type 13)

Record Type 4 - Format (I3,4XA60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

NAMDT - Name of file where this date list resides. When KFILDT = KFILDI, NAMDT is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) **Date List**

This group of records determines the date/times for which TDLPACK output is wanted. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J)- Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by

subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL (read in Record Type 1) to the first date/time in IDATE( ).

Record Type 6 - Format (I3,4XA60) **TDLPACK Input Gridded Random Access File**

This group of records identifies the TDLPACK data set from which the data are read. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 6 is read by subroutine RDSNAM.

KFILRA - Unit number for the gridded random access file. Any unit number may be assigned to KFILRA as long it does not conflict with any other unit number being used elsewhere in U135.CN.

RACESS - Name of file corresponding to file KFILRA. (CHARACTER*60)

Record Type 7 - Format (I3,4XA60) **GRIB2 Output File**

This record (plus the terminator record) identifies the GRIB2 sequential output file. Records are read until the terminator KFILGO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 7 is read by subroutine RDSNAM

KFILGO - Unit number for the output file.

OUTGRD - Name of file corresponding to KFILGO. (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) **GRIB2 Sections 0 and 1 Input File**

This record (plus the terminator record) identifies the file from which the GRIB2 indicator and identification values are taken. Records are read until the terminator KFILSC1 = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM.

KFILSC1 - Unit number for the GRIB2 Indicator-identification input file.

IDFNAM - Name of file corresponding to KFILSC1. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) **GRIB2 Section 2 Input File**

This record (plus the terminator record) identifies the file from which the GRIB2 Local Use Section is taken. Records are read until the terminator KFILSC2 = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 9 is read by subroutine RDSNAM. Although this Record Type 9 is not currently supported, a terminator value of “99” must still be present in U135.CN.

KFILSC2 - Unit number for the GRIB2 Section 2 input file.

LCLNAM - Name of file corresponding to KFILSC2. (CHARACTER*60)

Record Type 10 - Format (I3,4XA60) **GRIB2 Section 3 Input File**

This record (plus the terminator record) identifies the file from which the GRIB2 Grid descriptor and grid specifications are taken. Records are read until the terminator KFILSC3 = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 10 is read by subroutine RDSNAM.

KFILSC3 - Unit number for the GRIB2 Section 3 input file.

GDSNAM - Name of file corresponding to KFILSC3. (CHARACTER*60)

Record Type 11 - Format (I3,4XA60) **GRIB2 Section 4 Input File**

This record (plus the terminator record) identifies the file from which the GRIB2 Product Definition values are taken. Records are read until the terminator KFILSC4 = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILSC4 - Unit number for the GRIB2 Section 4 input file.

PDSNAM - Name of file corresponding to KFILSC4. (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) **GRIB2 Section 5 Input File**

This record (plus the terminator record) identifies the file from which the GRIB2 Data Representation values are taken. Records are read until the terminator KFILSC5 = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILSC5 - Unit number for the GRIB2 Section 5 input file.

DATANAM - Name of file corresponding to KFILSC5. (CHARACTER*60)

Record Type 13 - Format (I3,4XA60) **BITMAP Input File**

This record (plus the terminator record) identifies the gridded random access file from which the bitmap data are extracted. This Record Type 13 along with Record Type 14 (see below) is used if the user desires to mask (reassign) some of the data values found on the TDLPACK input file(s). U135 reads the gridded random access file containing a record of “masking” values, that is, a record comprised of binary values. In that record, if a value of zero is encountered at a gridpoint, the data value corresponding to that same gridpoint found on the TDLPACK input file is reassigned a value of BMAPVAL (see Record Type 3). Otherwise, the original data value at the gridpoint is retained. Records are read until the terminator KFILBM = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 13 is read by subroutine RDSNAM.

KFILBM - Unit number for the BITMAP random access file. Any unit number may be assigned to KFILBM as long it does not conflict with any other unit number being used elsewhere in U135.CN.

BMPNAM - Name of file corresponding to file KFILBM. (CHARACTER*60)

Record Type 14 - Format (I3,4XA60) **BITMAP Variable File**

This record (plus the terminator record) identifies the file from which the BITMAP variable ID is to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 14 is read by subroutine RDSNAM.

KFILP - Unit number for the BITMAP variable ID.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 15 - Format (I3,4XA60) **MOS-2000 Variable Constant File**

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (The variable constants include plain language description and grid printing information, the latter of which is not operable in U135.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 15 is read by subroutine RDSNAM.

KFILCP - Unit number for the constant file.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 16 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3) **BITMAP Variable**

This group of records contains the bitmap variable ID to be used when masking GRIB2 data. When KFILP ≠ KFILDI, this group is omitted, and the BITMAP variable is taken from an input file (see Record Type 14). Records are read until the terminator ID(1,) = 999999 is reached. This Record Type 16 is read by subroutine RDPRED. Maximum number of records, sans the terminator record, = 1. If the number of bitmaping variables in the variable list exceeds one, U135 will terminate with an error message.

ID(J,N) - The first 3 (J=1,3) (N=1,ND4) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only) (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed. For the purposes of U135, the 4th word is set to zero.

CONTROL FILE INPUT: (Name read from U135.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) **Date List**

When the dates are not provided in file U135.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU135), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from 'U135.CN') (Unit = KFILSC1)

Record Type 1 - Format (9I10) **Sections 0 and 1 Input File**

This record contains the list of metadata values used to define the GRIB2 Indicator and Identification section (Sections 0 and 1) of the GRIB2 message. The values contained in this Record Type will be assigned to all GRIB2 messages. Entries for this Record Type are read until the terminator '99999999' is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 1 is read by subroutine RDSECT1 and must be placed in a separate file (i.e., KFILSC1 ≠ KFILDI). Below, is a description of the list of the values (in the order read in) expected by subroutine RDSECT1.

Note that if the user is unclear what value to place into a particular entry he/she should check the number of octets designated for that particular value and set that value to the equivalent of all bits turned on. For example, if a value in an entry requires one octet, the user would enter the value of 2^8-1 (255); for two octets, the user would enter the value of 2^{16}-1 (65535), and so on. If the number of values needed for a particular template is less than the total number of possible entries, set the remaining entry values to '-9999.' Note that the information contained in this paragraph is applicable to Sections 3, 4, and 5 discussed below. In addition, all subsequent references to "Code Table..." and "Note..." apply to the GRIB2 FM 92 GRIB Document.

(1) = Message Discipline (see Code Table 0.0 for all possible values)
(2) = GRIB edition number (Currently 2)
(3) = Identification of originating center (see Common Code Table C-1)
(4) = Identification of originating sub-center
(5) = GRIB master tables version number (see Code Table 1.0)
(6) = GRIB local tables version number (see Code Table 1.1)
(7) = Significance of reference time (see Code Table 1.2)
(8) = Production status of the processed data (see Code Table 1.3)
(9) = Type of processed data (see Code Table 1.4)

Below, is an example of a Control Record for Sections 0 and 1.

```
0         2         7         8         1         0         1         0         2
99999999
```

CONTROL FILE INPUT: (Name read from 'U135.CN') (Unit = KFILSC3)

Record Type 1 - Format (A6,9I10,/,10(I10)/,8I10) **Section 3 Input File**

This group of records contains the grid characteristics pertaining to Section 3 in the GRIB2 message. Maximum number of records, sans the terminator record, = ND11. This Record Type 1 is read by subroutine RDSECT3 and must be placed in a separate file, i.e., KFILSC3 ≠ KFILDI. Below is a description of the list of the values (in the order read in) expected by subroutine RDSECT3. If the user is unclear as to what value to place in a particular entry he/she should use a value of '255.' Note that all longitude values defined in GRIB2 must be in positive degrees east.

(1) = Six character map designation identifier. The first two characters identify the map region. The following two characters identify the type of map projection and the final two characters represent the mesh length of the grid. The user should look in the grid definition file for guidance as how to define this six character string. If the six character string defined for "MPROJ" in Record Type 3 is not found in the grid definition file, U135 terminates with an error message.
(2) = Source of the grid information (see Code Table 3.0 and Note 1)
(3) = Number of data points to be processed
(4) = Number of octets for optional list of numbers defining number of points comprising the grid dimensions. Since the user will almost always be using a regular grid, this value is normally set to zero (see Note 2).
(5) = Interpretation of list of numbers defining number of points (see Code Table 3.11)
(6) = Grid Definition Template Number(=N) (see Code Table 3.1)
(7) = Shape of the earth (see Code Table 3.2)

(8) = Scale factor of radius of spherical earth
(9) = Scaled value of radius of spherical earth
(10) = Scale factor of major axis of oblate spheroid earth
(11) = Scaled value of major axis of oblate spheroid earth
(12) = Scale factor of minor axis of oblate spheroid earth
(13) = Scaled value of minor axis of oblate spheroid earth
(14) = Number of points along a parallel or along the X-axis
(15) = Number of points along a meridian or along the Y-axis
(16) = Latitude of the first gridpoint
(17) = Longitude of the first gridpoint
(18) = Resolution and component flags (see Flag Table 3.3 and Note 1)
(19) = Latitude where Dx and Dy are specified

For Mercator projections, supply the additional information:

(20) = Latitude of the last gridpoint
(21) = Longitude of the last gridpoint
(22) = Scanning mode (Flags – see Flag Table 3.4)
(23) = Orientation of the grid, angle between i-direction on the map and the equator (see Note 1)
(24) = Longitudinal direction grid length (see Note 2)
(25) = Latitudinal direction grid length (see Note 2)

For Polar stereographic projections, supply the additional information:

(20) = Orientation of the grid (see Note 2)
(21) = X-direction grid length (see Note 3)
(22) = Y-direction grid length (see Note 3)
(23) = Projection center flag (see Flag Table 3.5)
(24) = Scanning mode (see Flag Table 3.4)

For Lambert conformal projections, supply the additional information:

(20) = Longitude of meridian parallel to Y-axis along which latitude increases as the Y-coordinate increases (i.e., grid orientation)
(21) = X-direction grid length (see Note 1)
(22) = Y-direction grid length (see Note 1)
(23) = Projection center flag (see Flag Table 3.5)
(24) = Scanning mode (see Flag Table 3.4)
(25) = First latitude from the pole at which the secant cone cuts the sphere.
(26) = Second latitude from the pole at which the secant cone cuts the sphere. Note: Although TDLPACK data can only be placed on a tangent cone, both latitude values must be set and must be equal.
(27) = Latitude of the southern pole projection
(28) = Longitude of southern pole projection

Below, is an example of an entry in the grid definition file for a Lambert map projection over the contiguous U.S. with a mesh length of 10 km. (While the spacing between each successive value is not correct, the line designation for each value is correct.) Note that the grid definition file does not require a terminator between entries or at the end of the file.

```
USLC10    0   269180        0        0         30         1        0  6371200             0
0         0        0      626      430   18330000 230740000       48 25000000     252000000
10000000  10000000 0       64  25000000   25000000 -90000000        0
```

CONTROL FILE INPUT: (Name read from 'U135.CN') (Unit = KFILSC4)

Record Type 1 - Format
(3I10.9,1X,I10.10,1X,32A1,/,3(10I10/),7I10,/,2I10) **Section 4 Input File**

This group of records contains values pertaining to Section 4 of the GRIB2 message as well as the MOS-2000 ID identifier for data extraction from TDLPACK. Maximum number of records, sans the terminator record, = ND12. This Record Type 1 is read by subroutine RDSECT4 and must be placed in a separate file, i.e., KFILSC4 ≠ KFILDI. Since U135 provides the option of packing multiple grids in one GRIB2 message, each field within a group must be separated by a value of '999999.' The last field in each group must contain a value of '777777' (placed after the value of '999999') to identify it as the last field in the GRIB2 message.

What follows, is a description of the list of the values (in the order read in) expected by subroutine RDSECT4. The first entry set listed below is mandated for all templates defined in the Product Definition Section 4.N and fulfills the requirements for **Product Definition Template 4.0 (Entries for an analysis or forecast at a horizontal level or in a horizontal layer at a point in time.)**

(1) = MOS-2000 four-word ID (The fourth word must be in integer form). Plain language is optional and may be placed to the right of the MOS-2000 ID.
(2) = Product Definition Template Number (see Code Table 4.0)
(3) = Parameter category (see Code Table 4.1)
(4) = Parameter number (see Code Table 4.2)
(5) = Type of generating process (see Code Table 4.3)
(6) = Background generating process identifier (defined by originating center)
(7) = Analysis or forecast generating processes identifier (defined by originating center)
(8) = Hours of observational data cutoff after reference time (see Note 1)
(9) = Minutes of observational data cutoff after reference time
(10) = Indicator of unit of time range (see Code Table 4.4)
(11) = Forecast time in units defined by (10). (When using Templates 4.0, 4.1 or 4.5, this corresponds to the same value as the projection in the third word of the MOS-2000 ID. If Template 4.8 or 4.9 is used, this value represents the projection in hours that corresponds to the beginning of the forecast period relative to the reference time.)
(12) = Type of first fixed surface (see Code Table 4.5)
(13) = Scale factor of first fixed surface
(14) = Scaled value of first fixed surface
(15) = Type of second fixed surface (see Code Table 4.5)
(16) = Scale factor of second fixed surface
(17) = Scaled value of second fixed surface

Product Definition Template 4.1: Individual ensemble forecast, control perturbed, at a horizontal level or in a horizontal layer at a point in time.

Add the following values:

(18) = Type of ensemble forecast (see Code Table 4.6)
(19) = Perturbation number
(20) = number of forecasts in ensemble

Product Definition Template 4.5: Probability forecasts at a horizontal level or in a horizontal layer at a point in time.

Add the following values:

(18) = Forecast probability number
(19) = Total number of forecast probabilities
(20) = Probability type (see Code Table 4.9)
(21) = Scale factor of lower limit
(22) = Scaled value of lower limit
(23) = Scale factor of upper limit
(24) = Scaled value of upper limit

Product Definition Template 4.6: Percentile forecasts at a horizontal level or in a horizontal layer at a point in time.

Add the following value:

(18) = Percentile value (from 100% to 0%)

Product Definition Template 4.8: Average, accumulation, and/or extreme values or other statistically processed values at a horizontal level or in a horizontal layer in a continuous or non-continuous time interval. Note that the values for entries (18)-(23) refer to the ending time of the overall time interval.

Add the following values:

(18) = Year (Processed internally, assign a value of (65535)=2^{16}-1))
(19) = Month (Processed internally, assign a value of 255)
(20) = Day (Processed internally, assign a value of 255)
(21) = Hour (Processed internally, assign a value of 255)
(22) = Minute (Processed internally, assign a value of 255)
(23) = Second (Processed internally, assign a value of 255)
(24) = Number of time range specifications describing the time intervals used to calculate the statistically processed field
(25) = Total number of data values missing in statistical process
(26) = Statistical process used to calculate the processed field (see Code Table 4.10)
(27) = Type of time increment between successive fields (see Code Table 4.11)
(28) = Indicator of unit of time for time range in 29 (see Code Table 4.4)
(29) = Length of time range over which statistical processing is done, in units described in 28
(30) = Indicator of unit of time for the increment between the successive fields used (see Code Table 4.4)

Product Definition Template 4.9: Probability forecasts at a horizontal level or in a horizontal layer in a continuous or non-continuous time interval. Note that the values for entries (25)-(30) refer to the ending time of the overall time interval.

Add the following values:

(18) = Forecast probability number
(19) = Total number of forecast probabilities

(20) = Probability type (see Code Table 4.9)
(21) = Scale factor of lower limit
(22) = Scaled value of lower limit
(23) = Scale factor of upper limit
(24) = Scaled value of upper limit
(25) = Year (Processed internally, assign a value of (65535)=2^{16}-1))
(26) = Month (Processed internally, assign a value of 255)
(27) = Day (Processed internally, assign a value of 255)
(28) = Hour (Processed internally, assign a value of 255)
(29) = Minute (Processed internally, assign a value of 255)
(30) = Second (Processed internally, assign a value of 255)
(31) = Number of time range specifications describing the time intervals used to calculate the statistically processed field
(32) = Total number of data values missing in statistical process
(33) = Statistical process used to calculate the processed field (see Code Table 4.10)
(34) = Type of time increment between successive fields (see Code Table 4.11)
(35) = Indicator of unit of time for time range over which statistical processing is done (see Code Table 4.4)
(36) = Length of time range over which statistical processing is done, in units defined in 35
(37) = Indicator of unit of time for the increment between successive fields used (see Code Table 4.4)
(38) = Time increment between successive fields, in units defined by 37

Product Definition Template 4.10: Percentile forecasts at a horizontal level or in a horizontal layer in a continuous or non-continuous time interval. Note that the values for entries (19)-(24) refer to the ending time of the overall time interval.

Add the following values:

(18) = Percentile value (from 100% to 0%)
(19) = Year (Processed internally, assign a value of (65535)=2^{16}-1))
(20) = Month (Processed internally, assign a value of 255)
(21) = Day (Processed internally, assign a value of 255)
(22) = Hour (Processed internally, assign a value of 255)
(23) = Minute (Processed internally, assign a value of 255)
(24) = Second (Processed internally, assign a value of 255)
(25) = Number of time range specifications describing the time intervals used to calculate the statistically processed field
(26) = Total number of data values missing in statistical process
(27) = Statistical process used to calculate the processed field (see Code Table 4.10)
(28) = Type of time increment between successive fields (see Code Table 4.11)
(29) = Indicator of unit of time for time range over which statistical processing is done (see Code Table 4.4)
(30) = Length of time range over which statistical processing is done, in units defined in 35
(31) = Indicator of unit of time for the increment between successive fields used (see Code Table 4.4)
(32) = Time increment between successive fields, in units defined by 31

Below is an example of an entry in the Product Definition file for Template 4.0. using a 24-h forecast of surface relative humidity as the processed element. This entry will be packed into GRIB2 as one message.

(While the spacing between each successive value is not correct, the line designation for each value is correct.)

```
003001000 000000002 000000024 0000000000       2m Relative Humidity
    0     1       1      2        0       0       0       0       1       24
    1     0       0   -9999    -9999   -9999   -9999   -9999   -9999    -9999
-9999 -9999   -9999   -9999    -9999   -9999   -9999   -9999   -9999    -9999
-9999
999999 777777
99999999
```

CONTROL FILE INPUT: (Name read from 'U135.CN') (Unit = KFILSC5)

Record Type 1 - Format (2(10(I10)),/),2I10) **Section 5 Input File**

This group of records contains values pertaining to Section 5 of the GRIB2 message. Maximum number of records, sans the terminator record, = ND12. This Record Type 1 is read by subroutine RDSECT5 and must be placed in a separate file, i.e., KFILSC5 ≠ KFILDI. Since U135 provides the option of packing multiple grids in one GRIB2 message, each field within a group must be separated by a value of '999999.' The last field in each group must also contain a value of '777777' (placed after the value of '999999') to identify it as the last field in the GRIB2 message.

What follows, is a description of the list of the values (in the order read in) expected by subroutine RDSECT5. The first entry set listed below is mandated for all templates defined in the Data Representation Section 5.N and fulfills the requirements for **Data Representation Templates 5.0 (Gridpoint data – simple packing) and 5.41 (Portable Network Graphics – PNG)**

(1) = Data Representation Template Number (see Code Table 5.0)
(2) = Reference value (R) (IEEE 32-bit floating-point value) (Set to 0)
(3) = Binary scale factor (E) (set by user)
(4) = Decimal scale factor (D) (set by user)
(5) = Number of bits used for each packed value for simple packing, or for each group reference value for complex packing or spatial differencing. For JPEG 2000 or PNG compression, this refers to the number of bits required to hold the resulting scaled and referenced data values. (i.e., the depth of the image) (Set to 255)
(6) = Type of original field values (see Code Table 5.1)

Data Representation Template 5.2: Gridpoint data - complex packing

Add the following values:

(7) = Group splitting method used (see Code Table 5.4)
(8) = Missing value management used (see Code Table 5.5)
(9) = Primary missing value substitute
(10) = Secondary missing value substitute
(11) = Number of groups of data values into which field is split (Set to 0)
(12) = Reference for group widths (see Note 12) (set to 255)
(13) = Number of bits used for the group widths (after the reference value in octet 36 has been removed) (set to 255)
(14) = Reference for group lengths (see Note 13) (set to 0)
(15) = Length increment for the group lengths (see Note 14) (set to 255)
(16) = True length of last group (set to 0)

(17) = Number of bits used for the scaled group lengths (after subtraction of the reference value given in octets 38-41 and division by the length increment given in octet 42) (set to 255)

Data Representation Template 5.3: Gridpoint data - complex packing and spatial differencing (Include entries (1)–(17) and add the following two entries.)

(18) = Order of spatial differencing (see Code Table 5.6)
(19) = Number of octets required in the Data Section to specify extra descriptors needed for spatial differencing (octets 6-ww in Data Template 7.3) (set to 255)

Data Representation Template 5.40: Gridpoint data – JPEG 2000 Code Stream Format

Add the following values:

(7) = Type of compression used. (see Code Table 5.40)
(8) = Target compression ratio, M:1 (with respect to the bit-depth specified in octet 20 (entry (5)), when octet 22 (entry (7)) indicates Lossy Compression. Otherwise, set to missing.)

Below is an example of an entry in the Data Representation file for Template 5.0. that will contain one GRIB2 message (While the spacing between each successive value is not correct, the line designation for each value is correct.)

```
    0      0      0      0     255      0     -9999    -9999   -9999  -9999
 -9999  -9999  -9999  -9999   -9999   -9999    -9999    -9999   -9999  -9999
 999999 777777
 99999999
```

CONTROL FILE INPUT: (Name read from ‘U135.CN’) (Unit = KFILP)

When the variable to use is not in file 'U135.CN', this record contains the bitmap variable ID. When KFILP = KFILDI, this file is omitted, and the bitmap variable list is taken from input file KFILDI. The record is read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 1 is read by subroutine RDPRED. For the purposes of U135, only one bitmap variable may be defined for this Record Type. Hence, ND4 in the DRU135 should be set to a value of one.

CONTROL FILE INPUT: (Name read from ‘U135.CN’) (Unit = KFILCP)

Record Type 1 - Format
(3(I9,1X),I3,2X,A32,I3,F11.4,F10.4,F9.2,F8.2,2XA12) **MOS-2000 Variable Constants**

This group of records contains information about the bitmap variable, as defined by ID(J,) (see Record Type 16) and IDTEMP(J). This file should be universally useable by all MOS-2000 users and is expected to be a separate file. Note that the format matches that for a file input to other programs such as U600, U660, and U850.

IDTEMP(1) - First word of the bitmap variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=2,4) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALT - When IDTEMP(1) matches an ID(1,N), ISCALT is stored in ISCALD(N). **(Not used)**

SMULTT - When IDTEMP(1) matches an ID(1,N), SMULTT is stored in SMULT(N). This variable is the multiplicative factor (power of 10) to use when gridprinting the gridpoint data. **(Not used)**

SADDT - When IDTEMP(1) matches an ID(1,N), SADDT is stored SADD(N). This variable is the additive factor to use when gridprinting the gridpoint data. **(Not used)**

CONTT - When IDTEMP(1) matches an ID(1,N), CONTT is stored in CINT(N). This variable is the contour interval to use when gridprinting the data. It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT. **(Not used)**

ORIGNT - When IDTEMP(1) matches an ID(1,N), ORIGNT is stored in ORIGIN(N). This variable is the contour origin to use when gridprinting the data It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT. **(Not used)**

UNITST - When IDTEMP(1) matches an ID(1,N), UNITST is stored in UNITST(N). These characters define the units of the data after application of SMULTT and SADDT and are used in the visual inspection of the data in gridprinted maps. **(Not used)**

Other processing occurs with the reading of this record in RDPRED, much of it associated with the plain language description.

DATA INPUT:

All input data to be processed by U135 will be in gridded TDLPACK format. U135 accommodates one external gridded random access input file. The input random access unit number and file name are provided in KFILRA and RACESS (see Record Type 6). If the user desires to mask a portion of the input data, an additional gridded random access file may be defined in U135.CN (on unit number KFILBM) (see Record Type 13).

A. RANDOM ACCESS GRIDPOINT DATA

One source of gridpoint data from an external random access file is accommodated, the data set name and unit number having been provided to RACESS and KFILRA, respectively from the control file 'U135.CN', Record Type 6. It is important to note that all processed data must possess the same grid characteristics within one run of U135.

B. RANDOM ACCESS GRIDDED BITMAP DATA

Only one bitmap (in TDLPACK format) is accommodated, the data set Name and unit number having been provided to BMPNAM and KFILBM respectively, from the control file 'U135.CN', Record Type 13. This data type is used to mask a portion of the TDLPACK data field. The data on this file (generated by U361) **must** have the same output grid characteristics as those stipulated in the grid definition file. Otherwise, U135 gracefully terminates.

DATA OUTPUT:

There is one form of sequential data output, besides the diagnostics and other information on Units KFILDO and IP( ). An output GRIB2 file must be defined in U135.CN; otherwise, U135 will not execute.

BINARY GRIB2 FORMAT

All gridded output data are packed in GRIB2 format. They are packed by using NCEPS’s GRIB2 encoding modules (GRIBCREATE, ADDGRID, ADDFIELD, and GRIBEND). The data are written to KFILGO using subroutine WRYTE. All data are written to the dataset whose name has been provided to OUTGRD and unit number to KFILGO from the control file 'U135.CN', Record Type 7.

EXAMPLE CONTROL FILE: 'U135.CN'

An example exists as file 'U135.CN' in directory /nfsuser/g06/mos2k/tdllib/u135lib/ on the IBM. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more files as described above under control file 'U135.CN', Record Type 1, and the definition of KFILDO in the driver DRU135. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user

of problems and potential problems and to proceed under user control without jeopardizing data.

RESTRICTIONS

Since the input random access file is not processed by ‘CONST’ but rather ‘CONSTG’, the user may use any desired unit number for KFILRA. If the user also supplies a bitmap input file, its unit number KFILBM has no restrictions as well. This statement remains true as long as the random access file unit number does not conflict with an IP( ) value or another other unit number for a different record type.

If a bitmap file (Record Type 13) is listed in the control file, a bitmap variable (Record Type 14) must also be defined in the variable list. If the variable list is omitted when a random access file has been defined (or vice versa), U135 terminates with an error diagnostic notifying the user of this inconsistency.

If specific gridpoints are to be masked (denoted by zeros at gridpoints on the gridded random access file containing the bitmap), U135 assigns these gridpoint values to “BMAPVAL”, read in Record Type 3. Consequently, if the user chooses not to pack a bitmap into the GRIB2 message, the masked data value of “BMAPVAL” will be preserved in the

GRIB2 message. All unmasked data point values remain unchanged when passed to the GRIB2 packer.

U135 supports two types of scanning modes for the packing of data into GRIB2. The first scanning mode (of 64) corresponds to the data being scanned in raster order, beginning at the lower left corner of the grid (i.e., moving left to right, across a row for all rows). The second scanning mode (of 80) corresponds to the data being scanned in boustrophedonic order beginning at the lower left corner of the grid (i.e., moving left to right, then right to left in the above adjacent row, repeating this sequence for all subsequent rows). If the input data are to be packed into GRIB2 with a scanning mode of 80, the data are reordered internally by U135. If a bitmap is also to be packed (JBMAP=1), U135 assumes the user supplied bitmap field has a scanning mode of 64 and will thus reorder the bitmap field to a scanning mode of 80. The user need not concern himself/herself with this processing. If the data should require reordering, a diagnostic will appear in KFILDO.

Options such as reading TDLPACK sequential files as input, the look-back feature, the creation of binaries, the smoothing of data, and the computation of other fields that are found in other MOS-2000 programs are not present in U135. This was done to limit its functionality to a simple data conversion program used in operations.

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. North polar stereographic, Lambert, and Mercator projection gridpoint data are accommodated. Formats and other guidelines in other MOS-2000 documents are followed.

COMMENTS

All data returned from TDLPACK will be packed into GRIBI2 according to the metadata specifications labeled in the Data Representation file (Record Type 12). Note that if the user desires to pack a file containing primary and/or secondary missing values using simple packing, these values are treated in the same manner as all other values in the field. Consequently, simple packing will yield far less compression than complex or second order spatial differencing packing. It is important to stress that if the user wishes to pack primary and/or secondary missing values, complex packing or complex packing with spatial differencing (see Data Representation Templates 5.2 and 5.3 noted above) should be used. In this instance, a bitmap is not needed nor should it be packed into the GRIB2 message.

U135 will check that the input grid characteristics on the TDLPACK file equal those that are defined in the grid definition file. U135's grid characteristic cross checks are performed to the nearest millidegree and millimeter. If the grid characteristics are not equal at this level of precision, U135 terminates with an error message listing the grid characteristics fro both data sources so that the inconsistency can be readily apparent to the user. If a TDLPACK gridded random access file is used, its grid characteristics are also checked against the grid specifications found on the input data TDLPACK file (to the precision noted above). Any grid discrepancies will also result in the termination of U135 with a complete listing of the inconsistencies.

The user has the option of packing as many fields in one GRIB2 message as he/she desires. However, the last grid entry in that particular group (for Sections 4 and 5) must be delineated with a “777777.” An example of this can be seen in the above discussion of Record Type 9. If there is an inconsistency in the number of entries for Sections 4 and 5 for any particular group, U135 terminates with a diagnostic error message.

Since the current MOS-2000 system makes no provision for the use of data expressed to the minute and second, U135 outputs these values as zero in the GRIB2 message.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1).

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities.

SETTING UP THE DRIVER DRU135

The preparation of the driver for a particular U135 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the IBM when used with certain compiler options).
ND4 –	Maximum number of bitmap variables that can be defined in the bitmap variable list. (Maximum should be set to one)
ND5 –	Dimension of IPACK( ), IWORK( ), and DATA( ). Note that ND5 must be greater than or equal to NX*NY.
ND7 –	Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). Should be greater than or equal to 54.
ND8 -	Maximum number of dates that can be dealt with in one run of U135. Should be set to a value of one.
ND10 –	Maximum number of fields allowed in any one specific group (GRIB2 message) in one run of U135.
ND11 –	Maximum number of output grid entries defined in the grid definition file corresponding to Section 3 of the GRIB2 message.
ND12 –	Maximum number of groups (or GRIB2 messages) that can be processed in one run of U135.
LCGRIB –	Maximum length (bytes) of the GRIB2 message (stored in character array CGRIB( )).
IGDSTMPLEN –	Maximum dimension of IGDSTMPL( ) located in the Grid Definition Section of the GRIB2 message (Section 3).
IPDSTMPLEN –	Maximum dimension of IPDSTMPL( ) located in the Product Definition Section of the GRIB2 message (Section 4).
IDRSTMPLEN –	Maximum dimension of IDRSTMPL( ) located in the Data Representation Section of the GRIB2 message (Section 5).
IDEFNUM –	The entries in array IDEFLIST( ); i.e., the number of rows (columns) for which optional gridpoints are defined. (Used if IGDS(3).NE.0)
NUMCOORD –	Number of values in array COORDLIST( ).

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND5). Every effort has been made so that the variables will not be overflowed if values too small are used. The user should **not** change the values of the last five parameters noted above. These values have been carefully chosen to work within the framework of U135.

The "startup" control file "U135.CN" is opened in DRU135, so that this aspect of U135 can be rather easily changed without recompiling or having separate versions of the main (sub)routine U135. This is different on the IBM where unit/file assignments are made differently and possibly when U135 is run in batch mode.

NONSYSTEM ROUTINES USED

U135 is essentially a wrapper program that utilizes NCEP routines that pack and write GRIB2 data. Consequently, U135 is intended to be run on the IBM. The following NCEP libraries are used to link with U135 and must be placed in the makefile:

```
/nwprod/lib/libg2_4.a
/nwprod/lib/libw3_4.a
/nwprod/lib/libbacio_4.a
/usrx/local/64bit/lib/libjasper.a
/usrx/local/64bit/lib/libpng.a
/usrx/local/64bit/lib/libz.a
```

LANGUAGE: FORTRAN

LOCATION: U135lib. The driver is in /mdl1/save/mos/mos2k/u13xlib/RUN/dru135.f

U130

CONVERTS GRIB1 AND GRIB2 DATA TO TDLPACK

David E. Rudack
J. Paul Dallavalle
October 1, 2004

PURPOSE: U130 is designed to convert GRIB1 and GRIB2 data to TDLPACK. U130 supports the conversion of data on North polar stereographic, Lambert conformal, and Mercator projections. The user may use as input a TDLPACK external gridded random access file to mask (i.e., reassign data values to) the output data set. U130 outputs a TDLPACK file according to the grid specifications set forth in the grid list file (see below). Currently, U130 supports the GRIB2 decoding of Product Definition Templates 4.0, 4.1, 4.2, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10, 4.11, and 4.12. U130 uses a driver DRU130 so that dimensions of variables may be tailored by PARAMETER statements by the user without requiring a separate copy of the program U130 for every application. U130 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. Some familiarity with NCEP'S GRIB Office Note 388, GRIB2 Document FM 92 GRIB, and other MOS-2000 documents will be necessary for a full understanding of this write-up.

CONTROL FILE INPUT:'U130.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU130.

IPINIT- 4 characters, usually a user's initials plus a run number, to append to "U130" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U130 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'DER2' and IP(2) = 40, the file name for Unit No. 40 = 'U130DER240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U130 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. Values have been defined as indicated below for values of J:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU130 DATA statement = 12. Changing the default unit number allows multiple runs of U130 or other programs within the same directory without overwriting.

(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).

(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).

(4) = Statements indicating the writing of data to the TDLPACK sequential output file.

(6) = The bitmap variable ID as it is being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better. (See Record Types 7 and 13 for information concerning a bitmap.)

(7) = The bitmap variable ID in summary form. If there are errors, the bitmap variable will be written to the default output file unit KFILDO as well as to unit IP(7).

(8) = The bitmap variable ID in parsed form which includes IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)

(9) = The bitmap variable ID in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID in IDPARS(,), but rather includes the information taken from the MOS-2000 constant file on unit KFILCP (see below).

(11) = ASCII output of the GRIB1 or GRIB2 data field in tabular form. This file may be large for high resolution grids.

(12) = ASCII output of the TDLPACK bitmap data field in tabular form. This file may be large for high resolution grids.

(5),(10),(13)...(25) = Not used.

For checkout, it may be advisable to set all these values to zero or the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (I10/I10/F10.0/F10.0) Control Parameters

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

INCCYL - The increment in hours between date/times that are put into IDATE( ) (see Record Type 5) as a result of date spanning in subroutine DATPRO.

JCONVRT - The flag indicating if GRIB1 (=1) or GRIB2 (=2) data are to be TDLPACKED.

PXMISS - The designated value assigned to missing data on the input GRIB file(s).

XMISSS - The TDLPACK value assigned to all masked gridpoints (see Record Type 13). In most instances, XMISSS is set to a value of 9999.

Record Type 4 - Format (I3,4XA60) Date List File

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

NAMDT - Name of file where this date list resides. When KFILDT = KFILDI, NAMDT is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) Date List

This group of records determines the date/times for which TDLPACK output is wanted. If KFILDT (read in Record Type 4) does not equal KFILDI, this Record Type 5 is omitted.

IDATE(J)- Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL (read in Record Type 1) to the first date/time in IDATE( ).

Record Type 6 - Format (I3,4XA60) GRIB Input Data Files

This group of records identifies the GRIB data sets from which the data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

KFILIN(J) - Unit number for the GRIB input file(s). KFILIN( )is not restricted to the conventional values used for gridded data in the MOS-2000 system. Only values of 97, 99, and those conflicting with an IP( ) value or file unit number other than KFILIN( ) in the .CN file may not be used.

NAMIN(J)- Name of file corresponding to KFILIN( ). (CHARACTER*60)

Record Type 7 - Format (I3,4XA60) TDLPACK Gridded Random Access File

This record (plus the terminator record) identifies the gridded random access file. This record type 7 is only used if the user desires to mask (reassign) some of the data values found on the GRIB file. U130 reads the gridded random access file containing a record of bitmap values, that is, a record comprised of binary values. In that record, if a value of zero is encountered at a gridpoint, the data value corresponding to that same gridpoint found on the GRIB input file is reassigned a value of XMISSS. Otherwise, the original data value at the gridpoints is retained. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 7 is read by subroutine RDSNAM.

KFILRA - Unit number for the gridded random access file.

RACESS - Name of file corresponding to file KFILRA. (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) TDLPACK Output File

This record (plus the terminator record) identifies the gridded TDLPACK sequential output file. Records are read until the terminator KFILIO()=99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. Consequently, if U130 tries to open a pre-existing file with the same name, U130 will terminate.

KFILIO - Unit number for the output file.

<u>OUTNAM</u> - Name of file corresponding to KFILIO. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) <u>GRIB Element List File</u>

This record (plus the terminator record) identifies the file from which the GRIB element list is to be taken. Records are read until the terminator KFILIE = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 9 is read by subroutine RDSNAM.

<u>KFILIE</u> - Unit number for the GRIB element list file.

<u>ELENAM</u> - Name of file corresponding to KFILIE. (CHARACTER*60)

Record Type 10 - Format (I3,4XA60) <u>Grid List File</u>

This record (plus the terminator record) identifies the file from which the GRIB grid list values are to be taken. Records are read until the terminator KFILIC = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 10 is read by subroutine RDSNAM.

<u>KFILIC</u> - Unit number for the grid list file.

<u>CORNAM</u> - Name of file corresponding to KFILIC. (CHARACTER*60)

Record Type 11 - Format (I3,4XA60) <u>Bitmap Variable File</u>

This record (plus the terminator record) identifies the file from which the bitmap variable ID is to be taken. If a bitmap variable file is not needed, this Record Type 11 along with Record Type 13 must be omitted from the control file. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

<u>KFILP</u> - Unit number for the variable ID.

<u>PRENAM</u> - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) <u>MOS-2000 Variable Constant File</u>

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (The variable constants include plain language description and grid printing information, the latter of which is not operable in U130.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

<u>KFILCP</u> - Unit number for the constant file.

<u>CONNAM</u> - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 13 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3) Bitmap Variable

This group of records contains the bitmap variable ID to be used when masking GRIB data. This is needed when some gridpoints on the input grid are to be modified (see XMISSS in Record Types 3 and 7). This ID identifies the variable to be read from the gridded random access file (Record Type 7). When KFILP does not equal KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. This Record Type 13 is read by subroutine RDPRED. Maximum number of records, sans the terminator record = 1. If the number of bitmap variables in the variable list exceeds one, U130 will terminate with an error message.

ID(J,N) - The first 3 (J=1,3) (N=1,ND4) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only) (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed. For the purposes of U130, the 4th word is set to zero.

CONTROL FILE INPUT: (Name read from U130.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) Date List

When the dates are not provided in file U130.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU130), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from ‘U130.CN’) (Unit = KFILIE)

Record Type 1 - Format
(25I6,I6,2(I6,I10)/,3I10.9,I10.10,A4,1X,3(I2,1X),1X,32A1)
(Note that the format changed from 25I5 to 25I6 in 2006, as described below)

Element List File

This group of records contains the element list of values describing the GRIB data to be TDLPACKED as well as the corresponding MOS-2000 ID. The map projection and data scaling factors used during TDLPACKING are also found in this record. Records are read until the terminator ID68(1,) = 99999 is reached. Maximum number of records, sans the terminator record, = ND10. This Record Type 1 is read by subroutine RDELMLST and must be placed in a separate file (i.e. KFILIE does not equal KFILDI). Upon completion of reading this record type, N=1,NFIELDS.

The format of the element list file changed in 2006. Where ID68(J,K) formerly took the format of 25I5, it now takes the format 25I6. A script exists to convert old element lists to the new format. This script is located in [root]mos/mos2k/u13xlib/RUN/5to6.sc. Note that the GRIB and GRIB2 element lists differ in how "missing" unused fields are indicated; GRIB uses -1, GRIB2 uses -9999.

ID68(J,K)- Each value (J=1,30) (K=1,NFIELDS) corresponds to the product definition number assigned to the metadata and data fields. In order to find the relevant information needed for ID68(,) the user can run either "wgrib" (NCEP code for extracting for GRIB1 metadata), "wgrib2" (NCEP code for GRIB2 metadata), or "degrib" (MDL code for GRIB and GRIB2 metadata). The user should familiarize him/herself with the product definition section of NCEP'S GRIB Office Note 388 and/or the GRIB2 documentation FM 92 GRIB for a more thorough explanation of the subsequent descriptors. Note that all array values of "J" **not** defined below are set to a value of -1 for GRIB1 processing and -9999 for GRIB2 processing. In the description below, the corresponding labels from MDL's degrib program are given in parentheses. In October of 2007, the dimension of "J" in array ID68(,) was increased from 25 to 30 to allow for the extraction of probability values as defined in product definition template 4.5

Metadata per element list (degrib metadata nomenclature)

(3,K) = Originating center of the data.

(4,K) = Generation process ID number.

(5,K) = Grid identification number. Not used for GRIB2.

(8,K) = For GRIB1, Indicator of parameter category and units. For GRIB2, Parameter number (Category sub-description - e.g., potential temperature) – see Code Table 4.2). The Category description parameter is placed in (22,K) (see below).

(9,K) = Indicator of type of level or layer being processed. (Type of first fixed surface)

(10,K) = Value 1 of level being processed. (Value of first fixed surface)

(11,K) = Value 2 of level being processed. (Value of second fixed surface [when present])

(17,K) = For GRIB1, forecast time unit (see Table 4 in NCEP Office Note 388 **GRIB**) – For GRIB2, time range indicator (see FM92 code table 4.4). For GRIB1, a value of "1" or "missing" ("-1") causes U130 to use the default value of one hour. For GRIB1 and GRIB2, the coded values of 11 (6-hr) and 12 (12-hr) are supported.

(18,K) = Period 1 of time being processed. (Forecast time in hours).

(19,K) = Period 2 of time being processed. (GRIB2: Projection hours which corresponds to "End time of overall time interval").

(20,K) = For GRIB1, indicator of process [average, accumulation, maximum, minimum, etc.]. For GRIB1, always provide a value. For GRIB2, this corresponds to the statistical process [when present]. ID68(20,K) should be set to "missing" for most weather elements; however, the code currently ignores this field unless the product actually uses a statistical process such as a min or max temperature.

(21,K) = Product Definition Template (Product type). (GRIB2 only)

(22,K) = Category description parameter (e.g., temperature or moisture – see Code Table 4.1). The Parameter number is placed in (8,K) (see above). (GRIB2 only)

(23,K) = For Template 4.1, type of ensemble forecast (see Code Table 4.6). For Templates 4.2 and 4.12, the derived forecast (see Code Table 4.7). For Templates 4.6 and 4.10, the percentile value (from 100% to 0%). For Template 4.9, the forecast probability number. (GRIB2 only)

(24,K) = Perturbation number (Templates 4.1 and 4.11) or number of forecasts in ensemble (Templates 4.2 and 4.12). For ensemble processing only. (GRIB2 only)

(25,K) = Number of forecasts in ensemble (Templates 4.1 and 4.11) For ensemble processing only. (GRIB2 only)

(26,K) = Probability type (Templates 4.5 and 4.9). For probability forecasts only. (see Code Table 4.9) (GRIB2 only)

(27,K) = Scale factor of lower limit. For probability forecasts only. (GRIB2 only)

(28,K) = Scaled value of lower limit. For probability forecasts only. (GRIB2 only)

(29,K) = Scale factor of upper limit. For probability forecasts only. (GRIB2 only)

(30,K) = Scaled value of upper limit. For probability forecasts only. (GRIB2 only)

IDMDL(J,K) - The MOS-2000 ID (J=1,4) corresponding to the GRIB element in ID68(,). As of October 2007, IDMDL(4,K) requires ten digits in defining the 4th word of the MOS-2000 ID. This new requirement was necessary in order to capture probabilities that are defined in terms of a threshold value (e.g., The forecast probability of a ceiling height below 405 Meters).

MODELID(K) - A four character word (character*4) used to identify the model grid type. MODELID( ) is used to create a nexus between the grid on the GRIB file and the output grid. The four character word is arbitrary but must be identical to the grid identifier stored in the grid list file. (See next Record type)

IMAP(K) - The value identifying the map projection of the output grid.

3 - Lambert conformal projection
5 - North polar stereographic projection
7 - Mercator projection

IPWR10(K)- The power of ten scaling factor used to pack the element.

IPWR2(K) - The binary scaling factor used to pack the element. Only 0 is supported.

JPLAIN(MM,K) - The plain language used to describe the element (MM=1,32).

Below, is an example of a GRIB1 entry in the element list file. This variable is a forecast one hour average ground level ozone concentration. Note the four leading spaces in the first field. The first line of the element file is wrapped to the next line here; in the element list file, the first 30 fields are all on the same line.

```
-1    -1     7   211   148    -1    -1   180   107     0 10000    -1    -1    -1
-1    -1    -1     0     1     3    -1    -1    -1    -1    -1    -1    -1    -1
-1    -1
 009400003 000000000 000000001 0000000000OZ5X  3  3  0  POLLUTION
 99999
```

Below, is an example of a GRIB2 entry in the element list file. This variable is an NDFD hourly surface temperature forecast for Alaska, valid for the three-hour projection. Although not evident, there is one leading space before the first field ("-9999"). The first line of the element file is wrapped to the next line here; in the element list file, the first 30 fields are all on the same line.

```
-9999 -9999     8     2 -9999 -9999 -9999     0     1     0 -9999 -9999 -9999
-9999 -9999 -9999 -9999     3 -9999 -9999     0     0 -9999 -9999 -9999 -9999
-9999 -9999 -9999 -9999
 202054079 000000000 000000003 0000000000NDAK 5  1  0   NDFD SFC TEMP(K)
 99999
```

It is important to note that the entries for period 1 and period 2, "ID68(18,K)" and "ID68(19,K)", respectively, have a specific meaning when decoding a GRIB1 or GRIB2 message. Period 1 refers to the projection in hours corresponding to the beginning of the period of some averaged or accumulated variable, for example, QPF relative to the reference time. Period 2 refers to the projection in hours corresponding to the ending time of the period of this averaged or accumulated variable (relative to the reference time). If the user desires to decode a non-averaged variable (e.g., a 3-h temperature forecast), the entry for Period 1 must contain the projection in hours (a value of "3" in the example noted above) and the entry for period 2 must be set to the value of "-1" for GRIB1 or "-9999" for GRIB2. Technically speaking, GRIB2 does not express the ending time of a period in terms of a projection. Rather, the ending time of a period is expressed in terms of a date (i.e., year-month-day-hour-second). By design, U130 ingests the projection corresponding to the end of the period and then internally calculates the corresponding date of the end of the period. This requirement allows U130 to maintain a consistent control file set-up for the processing of either a GRIB1 or GRIB2 message.

When creating an element list for the extraction of probabilities with threshold values it is necessary to place the threshold value in the 4th word of the MOS-2000 ID. Failure to correctly identify this threshold value will cause the user untold headaches. If the user is unsure of the correct representation of the 4th word, he/she is encouraged to run u201 with the MOS-2000 ID placed in IDMDL(,) and check the output of IP(8). This will then supply the user sufficient information to properly define the threshold value with the correct nomenclature. For example, if the user is interested in extracting the probability of visibility below 4827.0 meters at the analysis time, he/she would enter the following:

```
-9999 -9999     7     5 -9999 -9999 -9999     0     1     0     0 -9999 -9999
-9999 -9999 -9999 -9999   000 -9999 -9999     5    19 -9999 -9999 -9999     0
3   4827000     0         0
008110201 000000000 000000000 0482704000AWIR  3  3  0  PROB(VIS < 4827.0 M)
```

CONTROL FILE INPUT: (Name read from 'U130.CN') (Unit = KFILIC)

Record Type 1 - Format
(A4,8I4,4F10.4,4X,F9.4.2F11.6,2X,A3,3X,F8.4) Grid List File

This group of records contains the grid specifications of the TDLPACK output file. Maximum number of records, sans the terminator record, = 50. (This value is hardwired in "GETGRID"). This Record Type 1 is read by subroutine "GETGRID" and must be placed in a separate file, i.e., KFILIC must not equal KFILDI.

MODELID(K)	- A four character word (character*4) used to identify the model grid type. MODELID( ) is used to create a nexus between the grid found on the GRIB file and the output grid (K=1,NFIELDS).
IGRID(K)	- Assigned grid identification number of the output grid (K=1,NFIELDS). The value of IGRID( )should **not** be confused with the MDL's numbering convention used in assigning various types of map projections. This value should be set to equal IGRIB(K).
IGRIB(K)	- Assigned grid identification number of the grid in the GRIB file (K=1,NFIELDS). This value is identical to ID68(5,K).
NXS(K)	- The number of gridpoints in the x-direction on the output grid (K=1,NFIELDS).
NYS(K)	- The number of gridpoints in the y-direction on the output grid (K=1,NFIELDS).
NX(K)	- The number of gridpoints in the x-direction on the input GRIB file (K=1,NFIELDS).
NY(K)	- The number of gridpoints in the y-direction on the input GRIB file (K=1,NFIELDS).
IOFFX(K)	- The number of gridpoints in the x-direction to offset the output grid relative to the input GRIB grid (K=1,NFIELDS). (IOFFX(K)=0 if no offset in the x-direction is desired.)
IOFFY(K)	- The number of gridpoints in the y-direction to offset the output grid relative to the input GRIB grid (K=1,NFIELDS). (IOFFY(K)=0 if no offset in the y-direction is desired.)

POLEX(K) - The x-coordinate of the north pole on the output grid. Although U130 does not actually use this value (the same can be said of POLEY), "POLEX" was not omitted from the grid list file because other MDL programs utilize this same grid list file (K=1,NFIELDS). For U130, this value can be set to zero.

POLEY(K) - The y-coordinate of the north pole on the output grid (K=1,NFIELDS). For U130, this value can be set to zero.

ALAT(K) - The latitude of the lower left corner gridpoint of the output grid (K=1,NFIELDS). The range is 0 to 90 degrees in the northern hemisphere.

ALON(K) - The longitude of the lower left corner gridpoint of the output grid (K=1,NFIELDS). West longitude is 0 through 180 and east longitude is 180 through 359.9999. All values **must** be positive.

ORIENT(K) - Grid orientation (i.e., the vertical longitude) of the output grid in degrees (K=1,NFIELDS). West longitude is 0 through 180 and east longitude is 180 through 359.9999. All values **must** be positive.

DX(K) - The x-increment at XLAT(K) (see below) expressed in kilometers (K=1,NFIELDS).

DY(K) - The y-increment at XLAT(K) (see below) expressed in kilometers (K=1,NFIELDS).

LPROJ(K) - The output map projection abbreviation in ASCII form (K=1,NFIELDS). ("LAM" for Northern hemispheric Lambert, "NPS" for North polar stereographic and "MER" for Mercator projection.)

XLAT(K) - Latitude at which the grid length (DX and DY) applies. The range is 0 to 90 degrees in the northern hemisphere. This is also the latitude of tangency in case of the Lambert map projection. Note that if a GRIB message contains a Lambert projection that has two values of "XLAT", U130 will recalculate an equivalent "XLAT", "DX", and "DY" (the latter two being equal). This computation is necessary because TDLPACK makes no provision for two values of "XLAT" and two different values of "DX" and "DY". U130 will provide a diagnostic notifying the user of the new recalculated values. These new values must equal the grid values of "XLAT", "DX", and "DY" that are stipulated in the grid list file (See the comments section for further details).

Below, is an example of an entry in the grid list file.

ZNEC 146 146 166 142 166 142 0 0 83.5424 683.7399 32.3530 89.9940 79.5000 12.0459 12.0459 LAM 41.0635

CONTROL FILE INPUT: (Name read from 'U130.CN') (Unit = KFILP)

Record Type 1 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2)
Bitmap Variable ID

When the variable to use is not in file 'U130.CN', this record contains the variable ID. When KFILP = KFILDI, this file is omitted, and the bitmap variable list is taken from input file KFILDI. The record is read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 1 is read by subroutine RDPRED. For the purposes of U130, only one bitmap variable may be defined for this Record Type. Hence, ND4 in the DRU130 should be set to a value of one. See Record Type 11, file 'U130.CN', unit KFILDI, for other details; the format is exactly the same.

CONTROL FILE INPUT: (Name read from 'U130.CN') (Unit = KFILCP)

Record Type 1 - Format(3(I9,1X),I3,2X,A32,I3,F11.4,F10.4,F9.2,F8.2,2XA12)
MOS-2000 Variable Constants

This group of records contains information about the bitmap variable, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all MOS-2000 users and is expected to be a separate file. Note that the format matches that for a file input to other programs such as U600, U660, and U850.

IDTEMP(1) - First word of the bitmap variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=2,4) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator, and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALT - When IDTEMP(1) matches an ID(1,N), ISCALT is stored in ISCALD(N). Since the bitmap variable is not placed into tdlpack, the power of ten scaling factor is not needed. **(Not used)**

SMULTT - When IDTEMP(1) matches an ID(1,N), SMULTT is stored in SMULT(N). This variable is the multiplicative factor (power of 10) to use when gridprinting the gridpoint data. **(Not used)**

SADDT - When IDTEMP(1) matches an ID(1,N), SADDT is stored in SADD(N). This variable is the additive factor to use when gridprinting the gridpoint data. **(Not used)**

CONTT - When IDTEMP(1) matches an ID(1,N), CONTT is stored in CINT(N). This variable is the contour interval to use when gridprinting the data. It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT. **(Not used)**

ORIGNT - When IDTEMP(1) matches an ID(1,N), ORIGNT is stored in ORIGIN(N). This variable is the contour origin to use when gridprinting the data. It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT. **(Not used)**

UNITST - When IDTEMP(1) matches an ID(1,N), UNITST is stored in UNITST(N). These characters define the units of the data after application of SMULTT and SADDT and are used in the visual inspection of the data in gridprinted maps. **(Not used)**

Other processing occurs with the reading of this record in RDPRED, much of it associated with the plain language description.

DATA INPUT:

All input data to be processed by U130 will be in GRIB1 or GRIB2 format. The input files can either be sequential or direct access. The unit numbers and file names are provided in KFILIN(J) and NAMIN(J) (J=1,NUMIN), respectively. Reading is done with an NCEP FORTRAN routine called "BAREAD" and the unpacking of data utilizes the NCEP FORTRAN routines of "GETGB" and "GETGB2." If any error messages are returned from these routines, the diagnostics should be clear enough that the user can rectify the problem.

A. GRIDPOINT DATA

One or more sources of gridpoint data are accommodated, the data set names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U130.CN', Record Type 6. It is important to note that all data sources should possess the same grid characteristics within one run of U130. Otherwise, the records on the TDLPACK output file will contain elements possessing different grid characteristics. This result is sometimes undesirable.

B. RANDOM ACCESS GRIDDED DATA

Only one source of gridpoint data (in TDLPACK format) is accommodated, the data set name and unit number having been provided to RACESS and KFILRA, respectively from the control file 'U130.CN', Record Type 7. Gridpoint data in this file are used to mask a portion of the GRIB data. This bitmap record (generated by U361) must have the same output grid characteristics as those stipulated in the input grid list file.

DATA OUTPUT

There is one form of data output, besides the diagnostics and other information on Units KFILDO and IP( ).

SEQUENTIAL BINARY MOS-2000 FORMAT

All gridded output data are packed in the MOS-2000 TDLPACK format. They are packed with subroutine PACK2D and its associated subroutines. The data are then written to KFILIO using subroutine WRITEP. All data are written to the dataset whose name has been provided to OUTNAM and unit number to KFILIO from the control file 'U130.CN', Record Type 8.

EXAMPLE CONTROL FILE: 'U130.CN'

An example exists as file 'U130.CN' in directory [root]mos/mos2k/u13xlib/RUN on the IBM. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U130.CN', Record Type 1, and the definition of KFILDO in the driver DRU130. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control without jeopardizing data.

RESTRICTIONS

Although U130 opens and processes each input file individually and then closes that file prior to opening and processing the next input file, **no input files may share the same unit number; each input file must have a unique unit number.** Furthermore, since the random access file is not processed by ‘CONST’ but rather ‘CONSTG’, the user may use any desired unit number as long as the random access file unit number does not conflict with an IP( ) value or another unit number for a different record type.

All GRIB input files **must** be placed in the control file in chronological order. In addition, all variables that are to be processed for a specific date/time **must** be on the same file. They may not be spread over two or more files. U130 will, however, process multiple data records on an input GRIB file that spans multiple dates.

The user can perform only one type of data conversion in a run of U130. In other words, U130 will not process GRIB1 and GRIB2 data together and place the data into TDLPACK. Each type of GRIB conversion must be done in a separate run of U130.

Prior to the processing of each GRIB input file, a corresponding index is created. The purpose of the index file is to locate the desired record on the GRIB file. The index file is located in the directory where the U130 executable resides and is assigned the reserved unit number of **97**. The index file (named "grib1.index" or "grib2.index") is removed immediately after the related GRIB file is closed and the next GRIB file is to be opened.

In all other MOS-2000 programs, the user may specify a random access file (Record Type 7) in the control file without listing a variable that is defined in the variable list (Record Type 13). U130, however, mandates that if a random access file is listed in the control file, a bitmap variable must also be defined in the variable list. If the variable list is omitted when a random access file has been defined (or vice versa), U130 terminates with an error diagnostic notifying the user of this inconsistency.

If a gridpoint is to be masked (denoted by ones at gridpoints on the gridded random access file), U130 reassigns that gridpoint a value of XMISSS. In most instances, XMISSS should be set to 9999. All unmasked data values remain unchanged. Recall that the value of XMISSS is set at the top of 'U130.CN'.

Options such as the look-back feature, the creation of binaries, the smoothing of data, and the computation of other fields that are found in other MOS-2000 programs are not present in U130. This was done to limit its functionality to a simple TDLPACKING program.

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. North polar stereographic, Lambert, and Mercator projection gridpoint data are accommodated. Formats and other guidelines in other MOS-2000 documents are followed.

COMMENTS

All values in the GRIB data field equaling "PXMISS" (see Record Type 3) are TDLPACKED as 9999. The user must be extremely careful when setting

'PXMISS' because a legitimate data value may be mistakenly TDLPACKED as '9999'. The user should understand that 'PXMISS' applies to all GRIB data processed in one run of U130. Consequently, if the user were to run U130 with two input data sets, each containing a different designated missing value, legitimate values found in one of the data sets (equaling "PXMISS" of the other data set) will be packed incorrectly as '9999'.

The lower left corner of the GRIB input grid need not be coincident with the lower left corner of the output grid. For example, the user may desire a subset of the input grid. In this instance, "NX", "NY", "IOFFX", "IOFFY" "ALAT" and "ALON" must be modified accordingly. If a gridded random access file is to be used in this instance (for the masking of data), its grid characteristics **must** also match the offset grid characteristics. It is important to note that in this case, all other grid characteristics to be placed on the output file must remain equal to those found in the GRIB input file. Other than offsetting the GRIB input grid, no other map manipulation is performed by U130.

U130 will check that the input grid characteristics on the GRIB file equal the output grid characteristics found in the grid list file. If the grid specifications are not equal, U130 terminates with an error message listing the grid characteristics of both files so that the inconsistency can be readily apparent to the user. If a TDLPACK gridded random access file is used, its grid characteristics are also checked against the output grid specifications in the grid list file. Any grid discrepancies will also result in the termination of U130 with a complete listing of the inconsistencies.

U130 supports two types of scanning modes from input as read from the GRIB file(s) but only one on output (scanning mode of 64). The first scanning mode (of 64) corresponds to the data being scanned in raster order, beginning at the lower left corner of the grid (i.e., moving left to right, across a row for all rows). The second scanning mode (of 80) corresponds to the data being scanned in boustrophedonic order beginning at the lower left corner of the grid (i.e., moving left to right, then right to left in the above adjacent row, repeating this sequence for all subsequent rows). If the input data should require reordering, a diagnostic will appear in KFILDO.

MDL's primary source of GRIB data used in development and operations originates from NCEP. And since the grid specifications in NCEP's GRIB files are expressed to the precision of millidegrees and millimeters, U130's grid characteristic cross checks are also performed to this level of precision. As a result, if the input and output grids differ by tenths of a millidegree and/or millimeter (or less), U130 will not alert the user of any discrepancies.

It is important to note that the precision to which the user would like to pack the output grid characteristics is determined by the precision stipulated in the grid list file.

When two “XLAT” values are defined in the GRIB file for a Lambert projection and the equivalent “XLAT”, “DX” and “DY” values are not known *a priori*, U130 may be run with a guess value of “XLAT”, “DX” and “DY” in the grid list file. U130 will terminate and write the correct values into KFILDO. These corrected values can then be substituted into the grid list file and U130 can then be rerun.

Although the current MOS-2000 system makes no provision for the use of data expressed to the minute, U130 outputs metadata that is referenced to the temporal scale of minutes. The minute value is stored in array “IS1(7).”

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1).

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities.

SETTING UP THE DRIVER DRU130

The preparation of the driver for a particular U130 run consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B	-	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the IBM when used with certain compiler options).
ND4	-	The maximum number of bitmap variables in the variable list. Since U130 is designed to use only one bitmap variable, ND4 should be set to one.
ND5	-	The dimension of IPACK( ), IWORK( ), and DATA( ). Since each record size is not known *a priori* to processing, ND5 should be set to a value at least as large as the grid dimension (NX*NY) on the GRIB file.
ND6	-	The maximum number of GRIB input files that can be processed in one run.
ND7	-	The size of ISO( ), IS1( ), IS2( ), and IS4( ). This would normally be 54.
ND8	-	The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.
ND9	-	The maximum number of records that are anticipated to be on the GRIB file(s) being processed.

ND10 - The maximum number of elements that can be processed in the element list file in one run.

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND5). Every effort has been made so that the variables will not be overflowed if values too small are used.

The "startup" control file "U130.CN" is opened in DRU130, so that this aspect of U130 can be rather easily changed without recompiling or having separate versions of the main (sub)routine U130. This is different on the IBM where unit/file assignments are made differently, and possibly when U130 is run in batch mode.

NONSYSTEM ROUTINES USED

U130 is essentially a wrapper program that utilizes NCEP routines that read and retrieve GRIB data. Consequently, U130 is intended to be run on the IBM. If a user would like to run U130 on the HP system, all the source codes including and relating to GETGB, GETGB2, BAOPEN, BAOPENR, BAREAD, and BACLOSE will need to be placed on the HP system.

U130 calls several source codes from the FORTRAN “cmapf” library when a true mesh latitude and its corresponding mesh length need to be calculated for a Lambert conformal grid. These codes have been supplied by the Air Resources Laboratory and reside in [root]mos/mos2k/gridlib/cmapf. These codes do not conform to MOS-2000 standards.

LANGUAGE: FORTRAN

LOCATION: U130lib. The driver is in [root]mos/mos2k/u13xlib/RUN/dru130.f

U155

OBJECTIVE ANALYSIS

Harry R. Glahn
October 1, 2004

PURPOSE: U155 is the driver program for a general objective analysis program. It calls three analysis modules according to what variable is being analyzed:

U405A - Analyzes continuous variables (e.g., temperature).
U405B - Analyzes wind, u and v components and speed together (will likely not be implemented).
U405D - Analyzes discontinuous variables (e.g., visibility) (not yet implemented).

These modules are explained in separate writeups.

U155 is controlled by a control file 'U155.CN'. Also, each major module (e.g., U405A) has one or more control files (e.g., 'U405ATMP^^^.CN' to analyze temperature). The variables to be analyzed in a particular run are determined by 'U155.CN', and the specifics of the analyses are controlled by control files specific to the variable.

U155, in turn, uses a driver DRU155 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U155 (actually, subroutine) for every application; these parameters are called NDx and will be referred to in this writeup. U155 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. Input to U155 includes both gridpoint and vector data in TDLPACK format. TDLPACK is documented in TDL Office Note TDL ON 00-1. U155 was derived from the LAMP program U150; the libraries moslib, lamplib, and u201lib are used in addition to the libraries for U155. The primary output of U155 is a TDLPACK gridpoint dataset containing grids as specified in the control files. In addition, for checkout and quality control, another gridpoint dataset is written containing subsets of the full grid; these grids can be contoured and contain input to U203, a front end to a display program ‘ugem.ksh’. They can also be displayed with ‘plot.sh’. Also, vector records can be written that contain (a) only the data that are accepted by the analysis, (b) the non-accepted data values, and (c) the questionable data values, exclusive of the not accepted ones. KFILDI (the default input unit number) and KFILDO (the default output unit number) are set in DRU155. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

A major use of U155 is foreseen to be the analysis on a fine grid (perhaps 5 km) of MOS forecasts and of hourly observations. A generalized MOS forecast grid or a model field can be used as a first guess. U155 can be used in both development and operations.

CONTROL FILE INPUT: 'U155.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU155.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U155" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U155 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U201HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,23) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U155 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). If another file unit number is designated the same as an IP( ) value, that IP( ) is set to KFILDO. This is ASCII output, generally for diagnostic purposes. Values have been defined as indicated below for values of J:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU155 DATA statement = 12. Changing the default unit number allows multiple runs of U155 or other programs within the same directory without overwriting.

(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).

(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).

(4) = The input station list (call letters only) when the station list is not from the directory [KFILD(1) ≠ KFILD(2)]. If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4) even when KFILD(1) = KFILD(2).

(5) = The station and station directory information in the order to be dealt with in U155. This can be in the order read or alphabetical. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5). This includes 8-character station identifiers, station name,

WBAN number, data quality flags, latitude, longitude, elevation, time zone, and substitute station(s).

(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7) and/or IP(9), may be better. If there are input errors, IP(6) will also contain diagnostic information as will KFILDO.

(7) = The variable ID list in summary form parsed into its 15 components. If there are errors, the predictor list will be written to the default output file unit KFILDO as well as to unit IP(7).

(8) = The list of stations with pairs, and their pairs lists.

(9) = The variable ID list in summary form. This differs from the print in IP(7) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).

(10) = The variable ID's for the first day (Day 1) as read from the archive files. This is just a list of all the packed gridpoint and vector fields on the input files.

(11) = The variable ID's of the archived fields actually needed. This is MSTORE(,), but it is not printed if only one date is processed (see Record Type 3).

(12) = The list(s) of stations comprising each directory record read from a vector input will be output on unit IP(12). For input of hourly data, this creates voluminous output. The print occurs in subroutine FINDST, which also prints a list of stations not found on the input file(s) <u>unless</u> compiled with the /D option.

(13) = The contents of LSTORE(,) after compression after each day (actually cycle) LE LSTPRT. LSTPRT is set in U155 DATA statement to 3.

(14) = Computed lapse rates are listed. Also, stations without good interpolation in subroutine ITRPSL or ITRPSX are identified.

(15) = X and Y positions of the stations for the analysis grid (at mesh length MESHB; see Record Type 3) used on this run are written to unit IP(15).

(16) = Statements indicating the writing of data to a sequential file.

(17) = A list of stations, their X and Y positions, data values, and LTAGS will be written to unit IP(17). The X and Y positions are in reference to the analysis grid length being used on that pass, which could be different from the X an Y positions listed on IP(15). Stations with missing data are not written. For several variables and hours, this will be voluminous output.

(18) = For the analysis of <u>continuous</u> variables, a listing of stations, their X and Y positions, data values, LTAGS, interpolated values for the <u>unsmoothed</u> analyses, and differences between the data and the analysis values <u>for the whole analysis area</u> will be written to unit IP(18) for each pass. The X and Y positions are in reference to the analysis grid length being used on that pass, which could be different from the X an Y positions listed on IP(15). Stations with missing data are not listed.

Also, the data packed by PACKV are listed to the resolution packed when JP(3) ≠ 0. Also, diagnostics are written explaining data that are written to file KFILOV.

(19) = The same as IP(18) except it applies to smoothed analyses. If analyses are not smoothed, IP(19) is not written. IP(19) will probably be smaller than IP(18) because not all passes may not be smoothed.

(20) = For continuous variables, a listing of stations, their X and Y positions, data values, LTAGS, interpolated values for the unsmoothed analysis, and differences between the data and the analysis values will be written to unit IP(20) for each pass for only the subsetted area. In addition, for the subsetted area, the degree of fit (average absolute error) between the unsmoothed analysis and the observations and between the smoothed analysis and the observations will be written to units KFILDO and IP(20) (see Record Type 3 below for description of subsetted area).

(21) = The average degree of fit between the data and the interpolated values from the analysis for the unsmoothed and, if smoothed, the smoothed analysis for each pass over the whole analysis area. This produces one (or two if smoothed) line(s) per pass for each analysis being done (e.g., temperature from U405A). Also, the stations tossed on the last pass are listed.

(22) = The gridprints (zebra maps) for the subsetted area.

(23) = Information concerning opening and closing of sequential files.

(24) = Fit to withheld, if any, and non-withheld stations over the whole analysis area. Used for withheld data tests. When WTHOL1 is used, the list of withheld stations, fit to stations, and any problems with ITRPSL in FITWTH. When WTHOL2 is used, the list of withheld stations, prepared with the filename ‘withheld’ for direct input into U600.

(25) = Bogus station problems are listed.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (6(I10/),F10.0/I10/2(F10.0/),4(I10/),2(F10.0/),12(I10/),I10) Control Parameters

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

KSKIP - When nonzero, KSKIP indicates that the output files KFILIO and KFILQC (see Record Types 8 and 12) are to be moved forward until all data for date KSKIP have been skipped. KSKIP is input as either YYMMDDHH or YYYYMMDDHH and then used as YYYYMMDDHH. All such files should end with a trailer.

NSKIP - The number of errors that will be tolerated on "Day 1" before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results. For operations, only 1 day would occur.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - The increment in hours between date/times that are put into IDATE( ) (see Record Type 5) as a result of date spanning in subroutine DATPRO. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL to the first date/time in IDATE( ).

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section). It is expected that only NEW = 1 will be used.

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

NAREA - The general area over which the run is made. A few options in the analysis are different for the different areas.
1 = CONUS
2 = Alaska
3 = Hawaii

PXMISS - The value to substitute in the output for a secondary missing value of 9997. This allows the 9997 to be maintained if desired, set to zero in the case of the implied value of near zero for forecasts, or even set to some other value. Note that this does not apply to missing values other than 9997.

NPROJ - The map projection to be used:
3 = Lambert,
5 = polar stereograhic, or
7 = Mercator.

ORIENT - The orientation of the grid; the longitude in degrees with which the grid columns are parallel.

XLAT - The latitude in degrees at which the mesh lengths are stated.

<u>MESHB</u> - The "nominal" mesh length in km at latitude XLAT of the grid to which NXL and NYL below refer. These nominal mesh lengths are a way of referring to a mesh length in integer form rather than a very precise and correct value at latitude XLAT. For instance, a nominal mesh length of 80 km is exactly 95.25 km at 60 degrees North Latitude on a polar stereograhic map. All other values of mesh length used in a run will be different from MESHB by one or more factors of 2. MESHB is also the mesh length of the final analysis grid written to KFILIO.

<u>MESHE</u> - The "nominal" mesh length in km at latitude XLAT of the terrain and sea/land grids to be used. The terrain and sea/land grids read in must match this.

<u>NXL</u> - The size (in number of gridpoints) of the analysis grid for this run in the NX direction in MESHB units.

<u>NYL</u> - The size (in number of gridpoints) of the analysis grid for this run in the NY direction in MESHB units.

<u>ALATL</u> - The latitude of the lower left corner of the grid to be used in the analysis. Specify to 3 decimal places.

<u>ALONL</u> - The longitude of the lower left corner of the grid to be used in the analysis. Specify to 3 decimal places.

<u>MESHL</u> - The "nominal" mesh (grid) length in km at latitude XLAT for the quality control (disposable) output grids for <u>continuous</u> variables.

<u>MESHD</u> - The same as MESHL except for <u>discontinuous</u> variables, That is, temperature may be output at MESHL = 5 km, while visibility might be output at MESHD = 10 km. If these are to be overlain for viewing, then MESHL and MESHD should probably be the same.

<u>NXGMIN, NXGMAX, NYGMIN, NYGMAX</u> - Define a subsetted area in terms of the NXL by NYL analysis grid to be gridprinted and written for viewing to Unit No. IP(22) and to be written to a TDLPACK record on Unit No. KFILOG (see the Data Output Section, Subsection B). The analysis grid may be quite large and it may be desired for checkout or quality control to output only a portion of the grid. The same area will be output at the MESHL resolution no matter what resolution is actually used (as specified in specific .CN control files). This means the output for different variables can be overlain for viewing. This facility does not impact the actual output to Unit No. KFILIO, but may require interpolation or thinning to get the areas to match. This same area can also be overlain with discontinuous fields, but only when MESHL = MESHD. NXGMIN = 0 implies no subsetting desired. If these values are internally inconsistent or inconsistent with NXL or NYL, KFILOG is set = 0, there will be no subsetted output, and a diagnostic is written.

See Section "Data Output, Section B" for the subset identifiers.

NCEPNO - The NCEP model number. For GFS MOS, this is 08.

MODNO - The DD in ID(1) of the output.

MINVEC - The number of hours to keep vector data for backup.

MINMOD - The number of hours to keep gridpoint data for backup or for merging 2 or more cycles for the analysis.

IPRTEL - 1 to output the terrain and land/sea mask grids per IP(22) and KFILOG; 0 otherwise.

ISTA - 1 to read station directory; 0 otherwise. This is in connection with sampling a first guess. When sampling, all data can come from the sampled points (ISTA = 0), only from the station directory (ISTA = 1), or both (ISTA = 1). ISTA = 1 is the normal setting.

ISMPL - The maximum number of points to sample from the first guess (see ISTA above) by subroutine SAMPLE. The actual points used is the maximum of ISMPL and IPOINT(J), J=1,NPASS (see U405A). **When ISMPL > 0, the first guess is replaced with the constant GUESS in U405Axxxxxxxxx.CN, Record Type 1.** NSTA from the directory + ISMPL must be $\leq$ ND1 or the number of points used will be truncated to ND1. ISTA and ISMPL must not both be 0. (Not generally used; set to zero.)

Record Type 4 - Format (I3,4XA60) Date List File

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) Date List

This group of records determines the date/times for which output is wanted. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. Note that any negative date must be immediately preceded by a positive date. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL (read in Record Type 1) to the first date/time in IDATE( ). If this is not so, some data may not be saved that will be needed for backup.

Record Type 6 - Format (2I3,1XA60) <u>Gridpoint and Vector Input Data Files</u>

This group of records identifies the data sets from which the gridpoint and/or vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN. If all input data are from a constant file, only the terminator is necessary.

<u>KFILIN</u>(J) - Unit number for the input gridpoint or vector file. KFILIN( ) must be < 80 for gridpoint data and $\geq$ 80 for vector data. Also note that values 97, 99, and 42 through 49 are reserved for other uses.

<u>MODNUM</u>(J) - Source of data that are on this file (e.g., 5 = LAMP forecasts, 8 = GFS model forecasts and GFS-based MOS). For gridpoint data, this number is used to match with the "DD" in the variable ID. It is zero for vector data (see Sequential Vector Data Section for more details).

<u>NAMIN</u>(J) - Name of file where this input resides. (CHARACTER*60)

Note that when data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number. Vector data can either precede, follow, or be mixed with gridpoint data sets in the list read; the only restriction is that those with the same unit number must be proper sequence.

Record Type 7 - Format (I3,4XA60) <u>Random Access Files</u>

This group of records identifies the random access data sets from which constant or other data are read or written. Records are read until the terminator KFILRA( ) = 99 is reached. Maximum number of records, sans the terminator record, = 6. This Record Type 7 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMRA. If no data are needed from a constant file, only the terminator is necessary. If the output is to a random access file, use a KFILRA( ) = 42.

<u>KFILRA</u>(J) - Unit number for the random access constant file (J=1,NUMRA). Unit numbers must be in the range 42 to 49; see "Restrictions" for more information.

<u>RACESS</u>(J) - Name of file corresponding to KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) <u>Gridpoint Output File</u>

This record (plus the terminator record) identifies the primary (archive) output file. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'OLD'. If it does not exist, it is opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

<u>KFILIO</u> - Unit number for the output file.

<u>GOTNAM</u> - Name of file where this output is to reside. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) <u>Vector Data for Plotting</u>

This ASCII output is specifically for use by GMOS_PLOT software. Records are read until the terminator KFILVO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 9 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If ASCII data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

<u>KFILVO</u> - Unit number for the output file.

<u>VOTNAM</u> - Name of file where this output is to reside. (CHARACTER*60)

Record Type 10 - Format (I3,4XA60) <u>Disposable Gridpoint Output File</u>

This record (plus the terminator record) identifies the disposable grid output file. Records are read until the terminator KFILOG( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 10 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

<u>KFILOG</u> - Unit number for the output file.

<u>OUTDIS</u> - Name of file where this output is to reside. (CHARACTER*60)

Record Type 11 - Format (I3,4XA60) <u>Disposable Vector Output File</u>

This record (plus the terminator record) identifies the disposable vector output file. Records are read until the terminator KFILOV( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

<u>KFILOV</u> - Unit number for the output file.

<u>OUTVEC</u> - Name of file where this output is to reside. (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) Quality Controlled Observation Output File

This record (plus the terminator record) identifies the quality controlled observation output file. Records are read until the terminator KFILQC( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If quality controlled data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

KFILQC - Unit number for the output file.

OUTQCV - Name of file where this output is to reside. (CHARACTER*60)

Record Type 13 - Format (I3,4XA60) Station and Location Files

This pair of records (plus the terminator record) identifies the file(s) from which the station (or location) information is obtained. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 13 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters which are to be used in the analyses (J=1) and the station dictionary which holds the latitudes, longitudes, WBAN numbers, elevations, quality control flags (in the block/station number field), and names for each possible station (J=2). KFILD(1) can be the default input file number, KFILDI, in which case DIRNAM(1) is not used. Also, KFILD(1) can equal KFILD(2), in which case the call letters list will consist of all stations on the directory file; note that both a list of stations and a directory cannot exist on unit KFILD(1) = KFILD(2).

DIRNAM(J) - Name of file matching KFILD(J). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 14 - Format (7(A8,1X)) Station List

This group of records identifies the stations for which data are to be used. If KFILD(1) ≠ KFILDI (the default input file) or if KFILD(1) = KFILD(2), this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designators) of stations (or locations) for which data are to be used in the analyses (K=1,NSTA). This list is read within subroutine RDSTQN or RDSTQA by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations in the list are kept, but a diagnostic is furnished on unit IP(5). For NALPH = 1, the stations are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory are put at the end of the list. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank (b) filled on the right, e.g., 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator is

'99999999'. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 15 - Format (7(A8,1X)) <u>Station Pairs File</u>

This record contains the stations for which pairs are available and the associated pairs as determined by U175. Records are read until the terminator KFILLP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 15 is read by subroutine RDSNAM. This file will be opened as 'OLD'. If lapse rates are not to be calculated, only the terminator is necessary here; in that case, a file is not opened and data are not read.

<u>KFILLP</u>	- Unit number for the input file.
<u>STAPRS</u>	- Name of file matching KFILLP. (CHARACTER*60)

Record Type 16 - Format (I3,4XA60) <u>Station Neighbors File</u>

This record (plus the terminator record) identifies the file from which the station neighbors is taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 16 is read by subroutine RDSNAM. If neighbors are not to be used, only the terminator is necessary here; in that case, a file is not opened and data are not read.

<u>KFILNI</u>	- Unit number for the neighbors file.
<u>STANEI</u>	- Name of file matching KFILNI. (CHARACTER*60)

Record Type 17 - Format (I3,4XA60) <u>Variables to be Analyzed File</u>

This record (plus the terminator record) identifies the file from which the variable ID's are to be taken for which analyses are to be made. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 17 is read by subroutine RDSNAM.

<u>KFILP</u> -	Unit number for the variable ID's.
<u>PRENAM</u> -	Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 18 - Format (I3,4XA60) <u>Individual Variable Control File</u>

This record (plus the terminator record) identifies the file from which the individual variable's .CN control files reside. Records are read until the terminator KFILAN = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 18 is read by subroutine RDSNAM.

KFILAN - Unit number for the .CN files.

ANLNAM - Name of file corresponding to KFILAN. When KFILAN = KFILDI, ANLNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 19 - Format (I3,4XA60) Variable Constants File

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description, scaling for packing, and gridprinting information.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 19 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 20 - Deleted.

Record Type 20 - Format
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,1XAl7,I4,3I2) Variable List

This group of records contains the variable ID's for which analyses are to be made (the Ids of the output analysis gird). When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 21 is read by subroutine RDLVRB. Upon completion of reading this record type, N=1,NPRED.

ID(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only) (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

JP(J,N) - For each variable N, JP(1,N) indicates whether that variable can have a gridprint on Unit No. IP(22), and JP(2,N) indicates whether that variable can have a TDLPACK grid output to unit No. KFILOG. JP(3,N) indicates whether the disposable vector records will be written to Unit KFILOV (see the Data Output section). This is an override feature for the parameters for gridprinting and TDLPACKing in each variable's control file (e.g., 'U405ATMPMOS^^^.CN').

ANLTAB(N) - This is the ASCII name of the control file for the variable. It is limited to 17 characters. The first 5 characters must be U405A, U405B, or U405D; ANLTAB( ) determines which of these three modules is called to analyze a particular variable (only U405A is implemented). (CHARACTER*17)

INLTAB(N) - The unit number for ANLTAB(N). Needed only for IBM operations.

IWRITS(N) - 1 when the final analysis is to be written to the internal random access file for possible use by another variable. It is not clipped (to the NDFD grid) and is not TDLPACKed.
- 2 when the variable LTAG( ) is written to the internal random access file for possible use by another variable. This is a vector record and is not packed. For instance, if the U-wind for a station is tossed, that station should not be used for the V-wind or speed.
- 3 when both the grid [(1) above] and LTAG( ) [(2) above] are written.
- 0 otherwise.

IWRITA(N) - 1 when the data are to be written as ASCII for 'plot.sh' to file Unit No. KFILVO.
- 0 otherwise.

ICOMPT(N) - 1 when the variable is not actually analyzed but computed from other analyses (e.g., wind direction).
- 0 normally.

CONTROL FILE INPUT: (Name read from 'U155.CN') (Unit = KFILDT)

Record Type 1 - Format (7I10) Date List

When the dates are not provided in file 'U155.CN', this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU155), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous positive date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL (read in Record Type 1) to the first date/time in IDATE( ). Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from 'U155.CN') (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) Station List

When the station list is not provided on file 'U155.CN', this group of records identifies the stations (or locations) to be used in the analyses. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from KFILDI. It is also not needed when KFILD(1) = KFILD(2); for this case, both the station list and the associated information are read according to the format in the following control file description for Unit No. KFILD(2).

CCALL(K) - Call letters (or other 8-character location designators) of stations (or locations) which are to be used (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

CONTROL FILE INPUT: (Name read from 'U155.CN') (Unit = KFILD(2))

Record Type 1 - Format Station Locations
(A8,1XA8,1XA17,4XA2,1XI6,1XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) K (K=1,NSTA) which are to be used. It is needed unless KFILD(1) = KFILD(2) = KFILDI; for this case, both the station list and the associated information are read from the input control file KFILDI. When KFILD(1) = KFILD(2) ≠ KFILDI, this control file is the same as the one above and both the station list and the associated information are taken from this file. If both files are needed (KFILD(1) ≠ KFILD(2) ≠ KFILDI), the call letters here are matched with those in the station list and the appropriate information extracted. When this station directory is alphabetical and NALPH = 1 (see Record Type 3), the final list of stations will be alphabetical no matter the order read in. (However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) Most of this information is used only within subroutine RDSTQN or RDSTQA to give information about the stations. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see Record Type 3).

CCALLD(K,J) - Call letters (or other character location designators) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list unless KFILD(1) = KFILD(2), in which case these call letters comprise the station list and at the completion of reading this file. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,1XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used for the name. (CHARACTER*20)

IQUAL(K) - 6-digit integer in which the rightmost digit is a sea/land descriptor, and the next ones to the left represent data quality. (See U450A writeup, Record Type 1a, IQUALC.)

ELEV(K) - Elevation of station, read in feet by the readers RDSTQN or RDSTQA and converted to meters.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited, except when it also constitutes the list to be kept, in which case the number of entries is limited by ND1-1. No terminator is used.

CONTROL FILE INPUT: (Name read from 'U155.CN') (Unit = KFILP)

Record Type 1 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,1XA17,I4,3I2)
Variable List

When the variables to use are not in file 'U155.CN', this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted, and the variable list is taken from input file KFILDI. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDLVRB. Upon completion of reading this record type, N=1,NPRED. See Record Type 21, file 'U155.CN', unit KFILDI, for other details; the format is exactly the same.

CONTROL FILE INPUT: (Name read from 'U155.CN') (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3,F11.4,F10.4,F9.2,F8.2,2XA12)
Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all MOS-2000 users and is expected to be a separate file. Note that the format matches that for a file input to other MOS-2000 programs such as U600, U660, and U850.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALT - When IDTEMP(1) matches an ID(1,N), ISCALT is stored in ISCALD(N). This variable is the scaling constant (power of 10) to use when packing the interpolated data.

SMULTT - When IDTEMP(1) matches an ID(1,N), SMULTT is stored in SMULT(N). This variable is the multiplicative factor (power of 10) to use when gridprinting the gridpoint data.

SADDT - When IDTEMP(1) matches an ID(1,N), SADDT is stored in SADD(N). This variable is the additive factor to use when gridprinting the gridpoint data.

CONTT - When IDTEMP(1) matches an ID(1,N), CONTT is stored in CINT(N). This variable is the contour interval to use when gridprinting the data. It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT.

ORIGNT - When IDTEMP(1) matches an ID(1,N), ORIGNT is stored in ORIGIN(N). This variable is the contour origin to use when gridprinting the data It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT.

UNITST - When IDTEMP(1) matches an ID(1,N), UNITST is stored in UNITST(N). These characters define the units of the data after application of SMULTT and SADDT and are used in the visual inspection of the data in gridprinted maps. (CHARACTER*12)

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in SETPLN called by RDV155, much of it associated with the plain language description. See Chapter 4, Variable Identification, of "MOS-2000" TDL Office Note 00-1 for details.

DATA INPUT:

All gridpoint and vector data input to U155 (other than control files) will be in the TDLPACK format read from sequential or MOS-2000 random access files, the vector data being accompanied by one or more directory records. The unit numbers and file names are provided in KFILIN(J) and NAMIN(J) [J=1,NUMIN (max of ND6)], respectively for sequential data and KFILRA(J) and RACESS(J) [J=1,NUMRA (max of 6)], respectively, for random access data. Gridpoint random access data must be from Unit Nos. 42-44 and vector data on Units 45-49. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG.

A. SEQUENTIAL GRIDPOINT DATA

One or more sources of gridpoint data are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U155.CN', Record Type 6. Each file is closed when an EOF is reached.

The definition of the grid to which the data in each record pertain is contained in that same record. The interpolation routines use this information, and precompute station locations (convert latitude/longitude to I/J) so that this computation doesn't have to be done for each record. However, a check is always made to assure that the correct grid information is used. In fact, not only can the data sources have different grid definitions, the definition within a particular source can change once or many times. It might be normal for the grid definition to change during the archive period. Be aware, though, that each time the grid definition changes when reading records that are to be used, extra computation must be done. ND11 (see "Setting up the Driver DRU155") must be $\geq$ the number of input grid sources (unit numbers).

For gridpoint data, the unit number KFILIN( ) must be < 80, and the model number MODNUM( ) must match the model number DD in the ID's for which data are to be taken from that unit. The RR is operative for the lookback feature. Unit numbers 42 through 49 are reserved for the MOS-2000 External Random Access System.

B. SEQUENTIAL VECTOR DATA

One or more sources of vector data are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U155.CN', Record Type 6. Each file is closed when an EOF is reached.

Each source (file) must have a directory record at the beginning, and one or more other directory records can occur in the same file following a trailer record (see MOS-2000 Software Documentation, Chapter 6). Each file, even read on the same unit number, must have a directory record at the beginning. The occurrence of a new directory record causes some additional computation.

For vector data, the unit number KFILIN( ) must be $\geq$ 80 (Nos. 97 and 99 are reserved for other uses) and the model number MODNUM( ) (see Record Type 6) must be zero. The RR is (is not) operative when the DD in the variable ID is (is not) zero. In this way, hourly data, which will have DD = 0, will have RR operative.

C. RANDOM ACCESS GRIDPOINT DATA

A first guess for one or more of the data sets to be analyzed can be on a MOS-2000 random access file. In addition, the terrain and land/sea mask fields are expected to be there. Unit Nos. 43 through 44 are reserved for input gridpoint data. Note that while a date/time is in the record definition in TDLPACK, there is no way to choose a particular date/time from a random access file. Therefore, the date/time needed must be the only date/time on the file for a particular ID( ).

D. RANDOM ACCESS VECTOR DATA

Up to 5 sources of station-oriented random access vector data are accommodated. These data are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information.

E. DATA FROM VARIABLE CONSTANT FILE

Station latitude and longitude are available from Unit Number KFILD(2) and are stored permanently for use in STALAT(K) and STALON(K), respectively (K=1,NSTA). Also, station (location) elevations are in ELEV(K)in meters, WBAN numbers are in IWBAN(K), a sea/land station designation is in LNDSEA(K), and data quality flags are in IQUAL(K,J) (J=1,5) for two types of data. (The elevations in the constant file are in feet and are converted to meters by the readers.) For instance, J=1 might pertain to temperature data and J=2 to precipitation. The WBAN values are not currently used.

F. RADII OF INFLUENCE FOR CORRECTION AND ELEVATIONS FOR LAPSE RATE

Radii of influence by station and pass VRAD(K,L) (K=1,NSTA) (L=1,6) and the highest and lowest elevations of gridpoints that will be influenced by this station are read from Unit No. KFILSL (when $\neq$ 0) that are used instead of the radii read in U405A and for lapse rate calculation. Note that this pertains to the whole run and not by element, but control of its use is by element. That is, VRAD(,) could be used for temperature but not for PoP. This file is produced by U178 according to input variables, and calculated radii can be overridden by an input file. U178 calculates the largest radii (for pass 1) necessary (if not overridden) and then by a formula calculates the radii for other passes.

DATA OUTPUT

There are six forms of data output, besides the diagnostics and other information on Units KFILDO and IP( ). All gridpoint data written will have DD = corresponding to the DD in the ID(1). See the individual modules (e.g., U405A) for additional details.

A. Gridpoint ASCII FOR VIEWING OR PRINTING

Gridprinted maps (zebra charts) over the subset area (see Record Type 3 of control file U155.CN) of each variable for which JP(1,) = 1 will be written to unit No. **IP(22)**, provided IP(22) is not zero and provided the appropriate values of NPRT( ) and JPRT( ) indicate gridprinting (these values are in the individual analysis control files). These data are written by individual modules (e.g., U405A).

B. TDLPACK GRIDS FOR CHECKOUT OR QUALITY CONTROL

TDLPACK grids over the subset area (see Record Type 3) of each variable for which JP(2,) = 1 will be written to the dataset whose name has been provided to **OUTDIS** and unit number to **KFILOG** from the control file 'U155.CN', Record Type 10, provided KFILOG is not zero and provided the appropriate values of NTDL(J) and JTDL(J) indicate TDLPACKing (these values are in the individual analysis control files). These "disposable" grids are written by individual modules (e.g., U405A) and pertain to each analysis pass J. Continuous fields will have a mesh length of MESHL; discontinuous will have MESHD. **These disposable grids are not clipped to the NDGD area** (see NCLIP in Record Type 1 in U405A writeup).

The IDs for these TDLPACK disposable grids are:

ID(1) = CCCFFFBDD, the same as the analysis being computed.
ID(2) = Pass number*10000 for the analyses and **990000** for the first guess (the LLLL portion) + the level [UUUU = IDPARS(7)].
ID(3) = Variable ID(3).
ID(4) = Variable ID(4) with the ISG (last 3 digits) replaced with 000 (unsmoothed) or 010 (smoothed) however it was done.

C. TDLPACK VECTORS FOR CHECKOUT OR QUALITY CONTROL

TDLPACK vectors of observations for each variable for which JP(3,) = 1 will be written to the dataset whose name has been provided to **OUTVEC** and unit number to **KFILOV** from the control file 'U155.CN', Record Type 11, provided KFILOV is not zero. Each variable so designated will have two records written per pass--(a) a record with only tossed observations not set to 9999, and (b) a record with only questionable observations not set to 9999. "Questionable" is defined by a threshold by pass which is a fraction of the ER1 read in the individual module .CN files. An observation called "tossed" is not also called "questionable." These vectors allows a map to be plotted with only tossed (or questionable) obs. These data are written by individual modules BCD5 and BCDW5 in U405A and U405B, respectively (U405B not currently implemented). These records are not applicable to discontinuous variables dealt with by U405D (U405D not yet implemented). This file also contains the calculated lapse rates, when calculated; the units are the units of the analyzed variable per meter * 1000.

The IDs for these TDLPACK disposable vectors are:

```
ID(1) = CCCFFFBDD, same as ID(1) of the data being analyzed
        (from ITABLE( ,2, )).
ID(2) = pass number*10000 for the "tossed" data (the LLLL
        portion).
      = pass number*10000 + 100000 for the "questionable" data
        (the LLLL portion).
      = 980000 FOR lapse rates.
ID(3) = ID(3) of the variable being analyzed.
ID(4) = ID(4) of the variable being analyzed.
```

D. SEQUENTIAL TDLPACK GRIDPOINT ARCHIVE

TDLPACK grids are written to the dataset whose name has been provided to **OUTNAM** and unit number to **KFILIO** from the control file 'U155.CN', Record Type 8, provided KFILIO is not zero. These data are written by the individual modules as created and therefore if a run aborts some data may be available but not necessarily all. When KFILIO ≠ 0, the final output (last pass of analysis) will be written for each variable. The mesh length of the archived grids are MESHB from U155.CN, and would usually match the mesh length of the last pass in the individual variable control file. **These disposable grids can be clipped to the NDGD area** (see NCLIP in Record Type 1 in U405A writeup).

The IDs for these TDLPAK archive grids are read in the control file in U405A. The DD is added according to the analysis being done. These are the same data with the same IDs as in G. GRIDPOINT DATA ON EXTERNAL RANDOM ACCESS FILE below. This provides an alternative to save the final analysis on a sequential or random access file, or both.

E. SEQUENTIAL TDLPACK VECTOR ARCHIVE

TDLPACK vectors are written to the dataset whose name has been provided to **OUTQCV** and unit number to **KFILQC** from the control file 'U155.CN', Record Type 12, provided KFILQC is not zero. These data are written by the individual modules as created, and therefore if a run aborts, some data may be available but not all. When KFILQC ≠ 0, one record will be written for each case (date/time) for each continuous variable containing the only the "accepted" stations, values for the tossed stations being set to 9999. These quality controlled data are written only when error checking is done on the last pass, which would be normal.

The 4-word ID is the variable being analyzed from ITABLE(,2,) in U405A, but the units may have changed, so the FFFBDD is from the ID(,) read in Record Type 21, the output variable.

The IDs for these TDLPAK archive vectors are:

```
ID(1) = CCC from the variable being analyzed from ITABLE( ,2, )
        in U405A with replacement FFFB00 from the ID of the
        analysis being made from Record Type 20.
```

ID(2) = ID(2) of the data being analyzed.
ID(3) = ID(3) of the data being analyzed.
ID(4) = ID(4) of the data being analyzed.

F. GRIDPOINT DATA AS ASCII FOR GMOS_PLOT

The non-missing data are written for use in plotting by ‘plot.sh’. The values per station, one record per station, are:

1) Longitude (minus for West) (REAL)
2) Latitude (REAL)
3) Values to plot. (INTEGER) These can be scaled if necessary by a call in U504A for specific CCCFFFB. Whatever is written out is plotted. For instance, wind direction xx and speed yy can be xx/yy.

G. GRIDPOINT DATA ON EXTERNAL RANDOM ACCESS FILE

The final analyses are packed and written when KFILRA( ) = 42 is provided. These are the same data with the same IDs as in D. SEQUENTIAL TDLPACK GRIDPOINT ARCHIVE above. This provides an alternative to save the final analysis on a sequential or random access file, or both.

EXAMPLE CONTROL FILE: 'U155.CN'

An example exists as file 'U155.CN' in subdirectories (e.g., ‘temp’) in directory 'home21/mos2k/staging/u155_mos_alaska_10_20_08'’ on blizzard. The easiest way to set up a run is to take an existing control file and modify it.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U155.CN', Record Type 1 and the definition of KFILDO in the driver DRU155. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control without jeopardizing data.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Only Lambert, north polar stereographic, and Mercator grids are accommodated. Formats and other guidelines in MOS-2000 documents are followed.

It is assumed that the grid size NXL by NYL are such that NXL-1 and NYL-1 are evenly divisible by 4. Otherwise, inconsistencies may occur in changing from one grid mesh length to a larger one. This might not be a problem, and wouldn't be unless a mesh length in the analysis process is

> MESHB, but unless not reasonable should be adhered to. For safety, if this condition is not met, it is counted as a fatal error.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The mainframe IBM treats them as comments.

The TDLPACK format for gridpoint data provides for both a primary and a secondary missing data indicator. If a primary value is found during unpacking of gridpoint (not vector) data, this is flagged at U155 completion by a statement on the output KFILDO file. The output grids can have missing values; if a specific grid does, the TDLPACKed grid will so indicate.

Arrays FD1( ), FD2( ), ... FD7( ) have been provided for general work arrays.

It is necessary that each variable be present for Day 1 for that variable to be used on subsequent days

A unit number of zero should never be read in the .CN control file. If a file is not needed, just omit it from the control file; this will be interpreted in U155 and called routines as 0. The routine RDSDAM does not recognize a zero unit number..

The unit numbers for the random access files must be in the range 42 through 49; those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
42	400-499	Gridpoint data (read/write) Write only in U155
43	400-499	Gridpoint data (read/write)
44	400-499	Gridpoint data (read only)
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts (read only)
49	200-299	Forecasts (read/write)

U155 will write to Unit No. 42, but is excluded from reading in GTHRES. Note that the CCC range may need to be modified in some routines if gridded MOS is put onto a random access file.

COMMENTS

U155 should run on an HP platform or the IBM at NCEP.

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing data will usually not be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities.

When no variable in a run has a particular model number, the file containing that model will not be needed; therefore, the file is closed. Closing the file keeps the data from being read when not needed.

A count is kept of each time a missing value indicator is found when unpacking grids (not vector data). The total is printed at the end of the run when > 0.

A "missing" value of 9997 encountered in input vector data is set to PXMISS (see Record Type 3). This allows 9997 to remain intact (PXMISS = 9997), set to zero (PXMISS = 0), set to "normal" missing (PXMISS = 9999), or even some other value.

Although the primary purpose of U155 is to produce packed gridpoint data, writing of such data is not done if KFILIO = 0 (see Record Type 8). That is, no output unit number and file name have been provided. A secondary purpose of U155 is to produce packed quality controlled observational (vector) data, writing of such data is not done if KFILQC = 0 (see Record Type 12). That is, no output unit number and file name have been provided. This feature can be used for checkout runs.

The modules of U155 (e.g., U405A) operate on grids of variable dimension. The grid length is specified at each stage of the process. For instance, the first guess and first pass of a sea level pressure analysis may be at 80 km.--perhaps the mesh length of an NCEP grid available for a first guess. Subsequent passes of the analysis may use other mesh lengths. U155 interpolates either linearly or quadratically to obtain a denser grid, or will use alternate grid points to thin a grid. Note, however, that interpolation does not work well for discontinuous fields and must be avoided. It is important to note that the internal value of MESH is the mesh length of the active grid at that point and that XP( ) and YP( ) contain the IX and JY positions of the stations relative to that grid. That is, MESH and XP( ) and YP( ) follow the grid. Or you could think of the grid and XP( ) and YP( ) following MESH--same difference. Subroutines exist for these mesh length transformations which always have to be done in steps of a factor of two.

It is assumed that the grid that is to furnish a first guess (e.g., an NCEP GFS model or a MOS generalized gridpoint forecast) is large enough to encompass the analysis grid (see ALATL, ALONL, NXL, and NYL in Record Type 3). The analysis grid is defined for this run by a lower left latitude and longitude and a "basic" mesh length equal MESHB. This has, then, a defined NX by NY extent. The first step in obtaining a first guess (assuming it is from a grid and not an average of observations or a constant) is to position the analysis grid within the first guess grid at the first guess grid mesh length. The next step is to make the guess the desired mesh length.

Seven forms of output (besides diagnostics, etc.) are provided for (see the section Data Output). The primary output is for gridpoint archival and is written to unit number KFILIO (providing it is not zero) and/or KFILRA( ) = 42 (when it is provided). These will all be over the analysis grid area at mesh length MESHB and size NXL by NYL. Quite likely MESHB will be the same as the mesh length on the last pass of the analysis. Experience has shown that all analysis passes will likely be at MESHB mesh length.

The quality control of continuous variables, with U405A and U405B, is quite good for some variables (e.g., sea level pressure), and an archive of these quality controlled observations is possible on Unit No. KFILQC. This archive can be used in development, input for operations, verification, etc.

Because the analysis grid may be quite large, it is inconvenient to try to look at all of it at one time. Therefore, a subset of the analysis area is provided as either a gridprint (zebra chart) on unit number IP(22) or as a TDLPACK record on unit number KFILOG. For instance, the subset (see NXGMIN, etc., in 'U155.CN', Record Type 3) can be of such a size that the gridprint will exactly fill one page of print. The TDLPACK output can be used in GEMPACK-related software, U203, although the TDLPACK output on unit number KFILIO can also be used for that purpose. A feature of the subset is that every grid of "continuous" variables will be on a mesh length of MESHL, and therefore will overlay with grids of other variables. For this to happen, some grids may be thickened or thinned from their original mesh length. In a similar manner, every grid of "non-continuous" variables will be on a mesh length of MESHD, and will therefore overlay with grids of other such variables. When MESHD = MESHL, then all grids will overlay. **Note that thickening of discontinuous grids cannot be done; they must be created at, or less than, the mesh length to be displayed.**

The output analysis on KFILIO will always have the smoothing indicator S in the ID set to 0 even though smoothing may have done in the analysis process.

When sampling is done (see ISMPL, Record Type 3), the points are determined in subroutine POINTS. A random latitude and random longitude within the analysis area are determined and stored in STALAT( ) and STALON( ), respectively. The Call Letters in CCALL(,1) are given an eight-character designation starting with "Q," the first being "Q0000001;

the other columns in CCALL(,) are set to blanks. LNDSEA( ) is given the value of the closest gridpoint in SEALND(,). The location name in NAME( ) contains the latitude and longitude to one decimal place.

CHARACTERISTICS OF THE U155 BACKUP FEATURE

Backup is provided by saving grids according to the variables MINVEC (for vector data) and MINMOD (for gridpoint data) (see Record Type 3). Note that saving data may increase run time. The process works as follows:

RDSTR1 Runs to acquire data for Day 1, date/time = NDATE. All data with the date range determined by MINVEC and MINMOD (as well as RR in IDPARS(9,) for gridpoint data) are put into internal storage and indicated in LSTORE(,).

GFETCH Marks the data in LSTORE(11,) that were retrieved for DAY 1.

LMSTR4 (a) Fills MSTORE(,), sets MSTORE(6,) with the cycles needed based on INCCYL, and sets MSTORE(7,) to MINVEC or MINMOD depending on the type of data, vector or gridded, respectively.

(b) For gridded data, sets LSTORE(12,) with the date of the data in LSTORE(8,) + MINMOD, except the result is reduced by MINMOD when it is $\leq$ NDATE.

(c) For vector data, sets LSTORE(12,) to the date of the data in LSTORE(8,) + MAX(MINVEC-INCCYL,0). Normally, MINVEC = 0 unless more than one hour of data are needed in an analysis.

(d) It then marks for deletion (sets LSTORE(1,) = 0) when LSTORE(12,) is $\leq$ the date of the data in LSTORE(8,).

GCPAC Discards data not needed (when LSTORE(1,) = 0) and "compresses" LSTORE(,).

RDSTR7 Is called for all days after Day 1. It reads and stores in internal storage the data indicated in MSTORE(,) when the data read are in the date range determined from MINVEC and MINMOD (and RR in IDPARS(12,) for gridded data) as appropriate. When it stores the data, GSTORE sets LSTORE(12,) to the date of the data + MINVEC or MINMOD which are in MSTORE(7,).

LMSTR2 Is called after RDSTR7, except for the last day, and marks data for deletion by setting LSTORE(1,) = 0 when LSTORE(12,) < the next date in the list of dates.

GCPAC Discards data not needed, except for the last day, (when LSTORE(1,) = 0) and "compresses" LSTORE(,).

In called routines like U405a, IBACKN and IBACKL read from individual variable control files define whether backup data will or will not be used, and how far back to go. U155 must save sufficient data to satisfy all such needs by the analysis routines (see MINVEC and MINMOD, Record Type 3).

SETTING UP THE DRIVER DRU155

The preparation of the driver for a particular U155 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B - Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine.

NBLOCK - Set to 6400*10/L3264B.

MAXSTA - Maximum number of stations (or points) that can be dealt with (i.e., used in the analysis). Note that this need not be as large as the number of stations in the directory (read on Unit No. KFILD(2)) unless, of course, the station directory is to be used as the station list. ND1 is set as the maximum of NBLOCK and MAXSTA.

ND2 - ND2*ND3 is the maximum size of the analysis grid that can be dealt with. ND2 and ND3 are set separately to highlight the possible dimension of the grids. However, in the called routines, the size is only limited by the product, not each dimension individually.

ND3 - See ND2.

ND4 - The maximum number of variables in the variable list.

ND5 - Dimension of IPACK( ), IWORK( ), and DATA( ). ND2X3 and ND5 should be set in the driver DRU155:

ND2X3 = MAX(ND1,ND2*ND3)
ND5 = MAX(ND2X3, size of largest input grid including the terrain and land/sea grids).

ND6 - Maximum number of all sequential file input sources (e.g., models) that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc.

ND7 - The size of ISO( ), IS1( ), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of variables stored in the MOS-2000 Internal Storage System. Since all records are stored for Day 1, ND9 must be large enough to hold all records of the date/time of Day 1 on all input files, including any RR > 0.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. While too small a number might result in more disk accesses than necessary (although caching may alleviate that) and too large a number might result in wasted memory and possible excess paging, experience has shown that the size of ND10 is relatively unimportant.

ND11 - The maximum number of grid combinations that can be dealt with. For instance, if the GFS and LAMP data are both being used, and the grids are different, then ND11 would be $\geq$ 2.

ND12 - The maximum size of the terrain and land/sea mask grids on the external random access file. This may be ND2*ND3, but not necessarily. By defining ND12, these two grids can be large compared to the analysis area.

ND13 - The maximum total stations involved in the pairs list.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. Note that ND1 must be GE NBLOCK. None of these values should be $\leq$ 0; the minimum value would be 1.

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND2*ND3). Every effort has been made so that the variables will not be overflowed if values too small are used; however, some danger may still exist.

The "startup" control file 'U155.CN' is opened in DRU155, so that this aspect of U155 can be rather easily changed without recompiling or having separate version of the main (sub)routine U155. An example exists as file 'U155.CN' in directory 'home21/mos2k/staging/toibm/dru155 on blizzard.

NONSYSTEM ROUTINES USED

Use the load line in one of the subdirectories (e.g., ‘temp’) in file 'home21/mos2k/staging/u155_mos_alaska_10_20_08', dataset 'u155.com' on blizzard. The “alaska” just means this version works for Alaska as well as for the CONUS.

LANGUAGE: FORTRAN77 with some HP extensions.
LOCATION: u155lib. The driver is in one of the subdirectories (e.g., ‘temp’) in 'home21/mos2k/staging/u155_mos_alaska_10_20_08' on blizzard.

U405A

ANALYZES QUASI-CONTINUOUS VARIABLES

Harry R. Glahn
October 1, 2004

PURPOSE: To produce objective analyses of quasi-continuous variables, such as observed surface temperature or sea level pressure, or forecast temperature or probability of precipitation, on a variable-sized computational grid. Analyses are performed with an adaptation of the Bergthorssen-Cressman-Doos successive approximation technique starting with a first guess grid. Data to be analyzed are error checked as specified in a control file against the current analysis, the current analysis being the first guess for the first pass and for any subsequent pass, the analysis from the previous pass.

The considerable adaptations include (1) essentially a separate analysis for ocean, inland water bodies, and land, (2) a lapse rate computation on-the-fly and applied to the analysis, the method depending on whether or not the lapse rate is of the expected sign (3) error correction which employs a buddy check only when a datum is in serious question, (4) combining forecasts from more than one cycle (start time) in one analysis, (5) several options for smoothing, (6) a lookback feature if a previous analysis is used for the first guess, (7) an option of blending an input grid over the ocean with an average first guess over land, (8) the ability to withhold stations for withheld data tests, (9) the ability to sample a grid and to use those points in the analysis, (10) the ability to set, or to nudge, the closest gridpoint to the station value after the last pass, (11) the ability to apply station-specific quality control flags that can depend on the variable being analyzed (12) the restricted use of surface winds in sea level pressure analysis to better define lows and troughs, and (13) the ability to do preprocessing and/or postprocessing by called routines. The lapse rate can be calculated from the data being analyzed or on upper air data.

U405A is called from U155 and is controlled by a set of control files 'U405AXXXXXXXXX.CN', where XXXXXXXXX pertains to a specific variable (e.g., XXXXXXXXX = MOSTMP^^^ for MOS temperature; the locations ^^^ can be used to further delineate the control file). 'U405AXXXXXXXXX.CN' is explained below. An understanding of the 'U155.CN' control file will be needed to understand certain variables mentioned below (e.g., IP22).

The genesis of U405A is the LAMP U400A and its precursors. LAMP analyzes sea level pressure and saturation deficit, a moisture variable. These elements have not been implemented in U450A, but vestiges remain; sea level pressure can be rather easily incorporated if necessary.

<u>CONTROL FILE INPUT:</u> 'U405AXXXXXXXXX.CN' (Unit = KFILDI)

Record Type 1 - Format <u>Overall Control</u>
(8I4,F8.0,1X,A16,5I4) (for other than sea level pressure)

This record contains the number of analysis passes to perform and other information which is not pass specific.

<u>NPASS</u> - Number of analysis passes to make, maximum of 6. However, one or more of these can be omitted (e.g., the first) by setting a corresponding NTYPE(,) = 0 (see below).

<u>IFSTGS</u> - Controls gridprinting to unit number IP22 or TDLPACKing and writing of the first guess to unit number KFILOG of the subsetted area:
0 = first guess is neither gridprinted nor TDLPACKed,
1 = the first guess is to be gridprinted,
2 = first guess is to be TDLPACKed and written,
3 = first guess is to be both gridprinted and TDLPACKed and written.

<u>IGUESS</u>(J) - The type of first guess to use in priority order (J=1,4).
1 = A constant (see GUESS),
2 = The primary first guess grid (e.g., MOS forecast grid),
3 = An alternate first guess grid (e.g., a model forecast),
4 = The average of all data values to be used in the analysis.
For example, if IGUESS(1) = 2, then the first priority for first guess is the primary grid. If IGUESS(2) = 1, the second priority is the constant GUESS (see below). If one source is not available, the next priority is tried. The grids for IGUESS(2) and IGUESS(3) are read into ITABLE(, ,) from U405A.CN (see Variables Accommodated below).

<u>IBACKN</u> - The maximum number of cycles to go back from the "current" cycle for a primary forecast grid to be used as a first guess at 6-h intervals. For instance, if IBACKN = 2 and the current primary forecast grid is not available, U405A will look for 2 previous runs at 6-hr intervals. By setting IBACKN = 2, then either the 6-h or 12-h previous run could be used in that priority order. When a previous run is used, the projection(s) are adjusted accordingly. The "current run" of the primary forecast grid is defined to be the cycle of the analysis, mod 6. That is, if the analysis were for 03Z, the current forecast grid is defined as 00Z. No assumption is made as to when the model actually runs. If only the current run is desired (no lookback), set IBACKN = 0. If IBACKN < 0, an NCEP forecast is not used even when IGUESS( ) = 2.

<u>IBACKL</u> - The maximum number of cycles to go back for a backup grid to be used as a first guess at 6-h intervals. For instance, if IBACKL = 2, and the forecast from the desired run of the backup forecast model is not available, U405A will look for 1 more previous run. When a previous cycle is used, the projection(s) are adjusted accordingly.

GUESS - The constant value to use as a first guess when IGUESS( ) = 1.

TITLE - A 16-character title for this variable, the variable being defined by the 4-word ID read by U155. (CHARACTER*16)

NSMTYP - Type of smoothing of the analysis (see B(,), Record Type 5, below).
1 = Normal 5-point. Does not accommodate missings (9999.).
2 = Same as 1, except no change is made unless one or more of the points to contribute to the new value has been changed. Does not accommodate missings (9999.).
3 = 9-point smoothing used on last pass only, any other pass defaults to NSMTYP = 2. Does not accommodate missings (9999.).
4 = For passes GE 4 for sea level pressure (SLP) only, same as 2 except a point is not changed if it is lower than both points above and below it or if it lower than both side points. Diagonals are also considered. For other passes and other variables, defaults to 2. (Also see Comments section. Sea level pressure not currently implemented.)
5 = Special terrain-following smoother.
6 = Same as 5, except two passes over the grid.
7 = Same as 5, except three passes over the grid.
If NSMTYP is not one of these values, smoothing will not be done. Note that Nos. 1-3 do not account for missings (9999.), but 4-7 do.

However, note that a postprocessing routine can be called that does smoothing in addition to the above. Two currently available are OSMTH and ORSMTH (see comments).

I400ADG - 1 = Diagnostics about analysis will be written to KFILDO.
0 = Most diagnostics will not be written.

LAPFG - 0 = Lapse will be computed from the data. This is the normal setting for the CONUS
1 = Compute lapse from first guess. This might be used if LAMP forecasts were being analyzed and the data were too sparse to compute from the data, so a first guess MOS might be used.
2 = Compute lapse from upper air data with subroutine LAPSUA. This is normal for Alaska where the stations are too sparse to compute a lapse from.
3 = Compute lapse from upper air data and surface data with LAPSEU. This is alternative to 2) above for Alaska, and is probably preferable.

LIMITX - The number of times the control information for this variable (CCCFFF) will be printed. Multiple taus for a particular CCCFFF will be printed only if LIMITX has not been exceeded.

IVRAD - 0 = Use R(,) normally.
1 = Use VARD ( ) as read from the Record Type 20 in U155. This gives the option of using different radii for different stations, a necessity for Alaska. The file is prepared by U178.

Record Type 1a - Format <u>Overall Control</u>
(I4,4F6.2,4I3,2F6.0,I4,2I2,I4,F4.0,2I4,2X,6I1)

<u>IQUALC</u> - The column in QUALWT( ) where the data quality for a station resides for the variable being analyzed. Column corresponds to the digit in IQUAL( ) (see U155 control file input, Unit KFILD(2)). See Appendix I "Use of Quality Control Flags."

<u>QUALWT</u>(J) - The fractional weights to apply to the four qualities of data (J=1,4) (see IQUALC above).

<u>ISETP</u> - Flag to indicate whether a station's value will be used for the closest gridpoint value when it is not 9999 or has not been tossed out through quality control (see appendix).
0 = don't use this feature (normal setting),
1 = nudge the gridpoint toward the station value but do not cross an integer value,
2 = set the gridpoint to the station value (has not produced good results.

<u>ILS</u> - The land/sea grid use parameter:
1 = use it--treat land and water points separately (see LNDWAT(,), Record Type 9, and ILS in U155,
0 otherwise.

<u>IBKPN</u> - Flag to indicate whether to apply BK(,) to positive or negative lapse rates:
+1 = apply to positive lapse rates (positive is odd for temperature and dew point),
+2 = same as +1, except don't apply downward,
-1 = apply to negative lapse rates (negative is odd for snow amount and wind speed)
-2 = same as -1, except don't apply downward,
0 = don't use BK(,),
99= don't compute or use lapse.
See BK(,) and ELCORR(,) below.

<u>LPNO</u> - Maximum number of points to use in computing lapse rate. IPKPN NE 99 and LPNO = 0 are not compatible. Not used when LAPFG = 2 or 3.

<u>HGTTHA</u> - The elevation difference in meters between a station and a gridpoint which when exceeded, the station value will not influence that gridpoint.

<u>HGTTHB</u> - The elevation difference in meters between a station and any interpolated point between the station and a gridpoint on the elevation grid which when exceeded, the station value will not influence that gridpoint.

<u>NWITH</u> - The number of data values (e.g., stations) to withhold in denied data tests. This can apply in either WTHOL1 or WTHOL2. Set = 0 for normal use.

ISEED - A number from 1 to 10 to indicate which seed value in ISEEDT( ) to use to start the randomization process. ISEEDT(J) is set by DATA STATEMENT in U405A. Note that this initialization is done only once in a U155 run.

ITYPR - For randomization studies:
1 = Use subroutine WTHOL1–to compute error analysis.
2 = Use subroutine WTHOL2–to save data for U600 error analysis.

NBLEND - Defines how the first guess will be used when IGUESS = 2 or 3. When set to 1, the incoming first guess will be used only over water as defined by the land/sea mask, and the average of land data over land. Set = 0 to disable this blending.

CSTSM - The smoothing parameter "B" to use instead of "BQ" when any gridpoint is ocean water, but not all involved in the smoothing are water. Used only with SMOTHG with NSMTYP = 5, 6, or 7.

N4P - Operates when the datum and the surrounding points are not all of the same type:
4 = Indicates the 4 gridpoints surrounding the station will be checked when trying to find a gridpoint of the same type. Actually, this is always done, and does not require this exact value.
12 = Same as above, except an additional 12 points will be checked when none of the 4 points is of the same type (a 4 x 4 grid).
16 = 16 points checked (12 is possible for backward compatibility).
36 = 36 points checked. While it may not be very important, a 16 or 4 should be used instead of 36 when they are sufficient and do not give diagnostics.

NCLIP - =1 to clip the output grid to the NDGD size.
=0 otherwise.

NSHLN(J) - (J=1,6) provides 5 options for smoothing dependent on terrain. Set NSHLN(J) = 1 to do the following:

```
J = 1 – Smooth high terrain, high variable value,
  = 2 -   "      "     "   , low variable value,
  = 3 -   "      "     "   , no high or low variable value
  = 4 -   "     low terrain, high variable value,
  = 5 -   "      "     "   , low variable value,
  = 6 -   "      "     "   , no high or low value.
```

WTWTL - The fractional weight of water values to apply to land grid-points. Operates only when ILS = 1 and LNDWAT(,) = 1.

WTLTW - The fractional weight of land values to apply to water grid-points. Operates only when ILS = 1 and LNDWAT(,) = 1.

Record Type 1b - Format (I4,5I6,/(4X,5F6.0)) Overall Control

NORUNS - The number of cycles (or runs) to be combined in one analysis, (presumably) all forecasts verifying at the same time. This should be a number between 1 and 5, inclusive.

NHRRUN(J) - The hours prior to the run time in NDATE (transmitted from U155) to indicate the times of previous cycles (J=1,5). Only NORUNS are actually used. Note that the first would normally apply to NDATE, and would be 0.

WTRUNA(J) - The weights to apply to each of the runs used in the analysis (J=1,5). Only NORUNS are actually used. If NORUNS = 1, then WTRUNA(1) should = 1.

WTRUNL(J) - The weights to apply in computing the lapse rates for each of the runs used in the analysis (J=1,5). Only NORUNS are actually used. If NORUNS = 1, then WTRUNL(1) should = 1. (Not actually implemented; all runs (cycles) used are weighted equally in computing lapse rates.)

Record Type 1c - Format (I4,5I6,/(4X,5F6.0)) Overall Control

ITABLE(J,L) - 4-word ID information for this variable (J=1,4):

L = 1-The analysis to produce. The DD and tau can be 0, making this generic. The CCCFFFB must match the ID read in the U155.CN file. Word 4 must also match; this is necessary because of thresholds.

L = 2-The vector data needed to make the analysis. Must include DD. (For direction, the u-wind.)

L = 3-The grid normally needed for a first guess, if one is required for IGUESS = 2. Must include DD. (For direction, the v-wind.)

L = 4-The grid normally needed for a first guess, if one is required for IGUESS = 3. Must include DD.

L = 5-Vector data that may be needed to compute the lapse rate (e.g., dew point might be used with temperature).

L = 6-Same as L = 5, if needed. Must include DD. (For direction, the revised speed for output.)

L = 7-Vector or grid that can be used in pre- or post-processing, as needed. Must include DD.

Record Type 2 - Format (i) Mesh Lengths

MSHPAS(J,L) - Nominal mesh length in km for each pass (J=1,6) for each first guess option (L=1,4). Only NPASS values are read and used from each of 4 records. Since MESH cannot equal 0, if MSHPAS(J,) is zero, pass J is skipped.

Record Type 3 - Format (6F8.0) Error Criteria

ER1(J,L,M) - Error criteria in units of the data being analyzed for each pass (J=1,6), for each first guess option (L=1,4), and for each month of the year (M=1,12). Only NPASS values are read and used from each of 48 records. The value M for the month being processed will be used. When data from more than one month are analyzed, the appropriate switch is made.

Record Type 4 - Format (6I8) <u>Type of Correction</u>

<u>NTYPE</u>(J,L) - Type of correction to make for each pass (J=1,6) for each first guess option (L=1,4). Only NPASS values are read and used from each of 4 records. The values of NTYPE (,) are explained in the appendix.

Record Type 5 - Format (6F8.0) <u>Smoothing Parameter</u>

<u>B</u>(J,L) - Smoothing parameter for each pass (J=1,6) for each first guess option (L=1,4). Only NPASS values are read and used from each of 4 records. The smoothing procedure is explained in the appendix.

Record Type 6 - Format (6F8.0) <u>Radius of Influence</u>

<u>R</u>(J,L) - Radius of influence in MSHPAS(J,L) grid units for each pass (J=1,6) for each first guess option (L=1,4). Only NPASS values are read and used from each of 4 records. Generally speaking, a data value affects a gridpoint only when it is within R(,) grid units of the gridpoint. <u>Note that the actual distance depends on the gridlength being used on that pass.</u> This is further explained in the appendix. As a special feature for ocean and inland water data points, R is increased by a factor of RWATO for ocean and RWATI for inland water. This is because of sparsity of water data points.

The use of R(,) can be suspended and replaced by a specific set of radii, one for each station, read from "Variable Radius and Hi/Lo Elevations File" on Unit No. KFILSL (See U155 writeup). This file provides a specific radius, by pass, for each station. This file is read once per run, and its use is controlled by IVRAD by variable (see Record Type 1).

Record Type 7 - Format (6I8) <u>Type of Interpolation</u>

<u>ITRPLQ</u>(J,L) - The type of interpolation to be performed whenever interpolation is necessary for each pass (J=1,6) and each first guess option (L=1,4). Only NPASS values are read and used from each of 4 records. If one pass of an analysis is on a grid of, say, 10 km, and the next pass is on 5-km, then interpolation is necessary to arrive at the "guess" for the next pass. ITRPLQ(,) is also used as necessary to arrive at thickened grids for the subsetted (see Data Output section) area. The values are:

1 = bilinear,
2 = biquadratic. (This won't work if grid has missing values.)

This does not apply to the interpolation into a grid to get a value at a station; for this, the quadratic ITRP or special ITRPSL is used (see Comments section).

Record Type 8 - Format (6F8.0) R Star Modification of Radius of Influence

RSTAR(J,L) - The fraction of R(,) to use when the datum is outside the grid for each pass (J=1,6) and first guess option (L=1,4). Only NPASS values are read and used from each of 4 records. That is, R(,) may be 8 grid units, but since the interpolation into the grid to get the "analysis" at that time becomes extrapolation outside the grid, it may be desirable to cut the 8 to, say, 2. This can be done by using RSTAR(,) = .25.

Record Type 9 - Format (6F8.0) Flags for Use of Land/Sea Distinction

LNDWAT(J,L) - Flag to indicate how to use the land/sea grid mask and the land/sea digits in IQUAL( ) for each pass (J=1,6) and first guess option (L=1,4) When ILS = 0, these values are not used. ILS gives a quick way to control this feature, while LNDWAT(,) can vary by pass and first guess. Only NPASS values are read and used from each of 4 records. (See Comments section.)
0 = Don't use land/sea differences.
1 = Use land/sea differences.
2 = Use land/sea differences and don’t change the ocean values; leave the ocean as first guess.
3 = Use land/sea differences and don’t change the ocean or inland water values.

Record Type 10 - Format (6F8.0) Type of Correction Algorithm

IALGOR(J,L) - Flag to indicate how to use the terrain in the correction.
1 = Normal, unweighted correction.
2 = Weighted by distance. (Implemented, but not currently used.)

Record Type 11 - Format (6F8.0) Elevation Correction Weight

ELCORR(J,L) - The fraction of the elevation correction to apply for each pass (J=1,6) and first guess option (L=1,4). Only NPASS values are read and used from each of 4 records. The elevation correction is explained in the appendix under "Use of Lapse Rate Adjustment." Note that this may not apply for “unusual” lapse rates (see ELCORU(,) below).

Record Type 12 - Format (6F8.0) Maximum Distance to Apply Unusual Lapse Rates

BK(J,L) - The maximum distance in grid lengths over which to apply the positive (IBKPN = +) or negative (IBKPN = -) lapse rates for each pass (J=1,6) and first guess option (L=1,4). Only NPASS values are read and used from each of 4 records. This special correction for "unusual" lapse rates is explained in the appendix.

Record Type 13 - Format (6F8.0) Fraction of Elevation Correction for Unusual Lapse Rates

ELCORU(J,L) - The fraction of the lapse rate correction to apply to the "unusual" lapse rates for each pass (J=1,6) and first guess option (L=1,4). Only NPASS values are read and used from each of 4 records. This special correction for "unusual" lapse rates is explained in the appendix.

Record Type 14 - Format (6I8) Ocean Water R Factor

RWATO(J) - A factor to multiply by R(J,) to increase the radius of influence for ocean water points (J=1,6). Note ths does not vary by first guess value. Only NPASS values are read and used. Ocean points are usually much more sparse that land points, and a different radius of influence is required.

The use of R(,), and therefore, RWATO( ),can be suspended and replaced by a specific set of radii, one for each station, read from "Variable Radius and Hi/Lo Elevations File" on Unit No. KFILSL (See U155 writeup). This file provides a specific radius, by pass, for each station. This is used for Alaska; it may not be needed for CONUS.

Record Type 15 - Format (6I8) Inland Water R Factor

RWATI(J) - A factor to multiply by R(J,) to increase the radius of influence for inland water points (J=1,6). Note ths does not vary by first guess value. Only NPASS values are read and used. Inland water points are usually more sparse that land points, and a different radius of influence is required.

The use of R(,), and therefore, RWATI( ),can be suspended and replaced by a specific set of radii, one for each station, read from "Variable Radius and Hi/Lo Elevations File" on Unit No. KFILSL (See U155 writeup). This file provides a specific radius, by pass, for each station. This is used for Alaska; it may not be needed for CONUS.

Record Type 16 - Format (6I8) Sampled Points to Use

IPOINT(J,L) - The maximum number of sampled points to use in the analysis for each pass (J=1,6) and first guess option (L=1,4). With the current implementation, the maximum of NPASS values in IPOINT(,L) is used for each pass. That is, the number of sampled points does not vary by pass. This feature is not used with first guess options 1 and 4.

Record Type 17 - Format (6F8.0) Multiplicative Constant for Gridprinting

SMULT(J) - The multiplicative factor to use before gridprinting the data for each pass (J=1,6). Only NPASS values are read and used. That is, if visibility is in miles and fractions, it may be desirable to use SMULT( ) = 100 so that the quarter mile values can be seen.

Record Type 18 - Format (6I8) Additive Factor for Gridprinting

SADD(J) - The additive factor to use before gridprinting the data for each pass (J=1,6). Only NPASS values are read and used. Usually this is zero.

Record Type 19 - Format (6F8.0) Origin for Gridprinting

ORIGIN(J) - The origin when gridprinting the data for each pass (J=1,6). Only NPASS values are read and used. That is, the base contour will be at ORIGIN( ).

Record Type 20 - Format (6F8.0) Contour Interval for Gridprinting

CINT(J) - The contour interval when gridprinting the data for each pass (J=1,6). Only NPASS values are read and used. CINT( ) is in reference to the original units, not after the multiplication by SMULT( ).

Record Type 21 - Format (6I8) Gridprint Option Unsmoothed Grids

NPRT(J) - Gridprinting option for the unsmoothed analysis for each pass(J=1,6) for the subsetted area. Only NPASS values are read and used.
1 = gridprint,
0 = otherwise.

Record Type 22 - Format (6I8) Gridprint Option for Smoothed Grids

JPRT(J) - Gridprinting option for the smoothed analyses for each pass (J=1,6) for the subsetted area. Only NPASS values are read and used.
1 = gridprint,
0 = otherwise.

Record Type 23 - Format (6I8) TDLPACK Option for Unsmoothed Grids

NTDL(J) - TDLPACKing option for the unsmoothed analyses for each pass (J=1,6) for the subsetted area. Only NPASS values are read and used.
1 = TDLPACK and write,
0 = otherwise.

Record Type 24 - Format (6I8) TDLPACK Option for Smoothed Grids

JTDL(J) - TDLPACKing option for the smoothed analyses for each pass (J=1,6) for the subsetted area. Only NPASS values are read and used.
1 = TDLPACK and write,
0 = otherwise.

Record Types 25 - Format (A6,I3,1X,A60/2I8,F8.0,2I8,3F8.0) Preprocessing Routines

PREPRO(J) - The names of the preprocessing routines, file unit numbers (KFILPR(J), file names (PREPFL(J)(J=1,3)and 8 values that can be used in the routine's execution. Leave blank for none.

Record Types 26 - Format (A6/5F8.0,I8,F8.0) Postprocessing Routines for Archive Grids

POSTAR(J) - The names of the postprocessing routines for archive output (J=1,3) and 8 values that can be used in the routine's execution. Leave blank for none.

Record Types 27 - Format (A6/5F8.0,I8,F8.0) Postprocessing Routines for Disposable Grids

POSTDS(J) - The names of the postprocessing routines for archive output (J=1,3) and 8 values that can be used in the routine's execution. Leave blank for none.

Record Type 28 - Format (6I8) Weights for Wind Correction for SLP

WNDWT(J,L) - **For SLP only**, fractional weight to apply to correction estimated by the wind and geostrophic relationship for each pass (J=1,6) and first guess option (L=1,4). For instance, 1.0 means apply full weight; 0.5 means apply 50% of the full weight. Only NPASS values are read and used from each of 4 records, and only when CCCFFF = 001201. This is omitted except for SLP.

DATA INPUT:

Most gridpoint and vector data needed by U405A have been stored by U155 into the MOS-2000 Internal Storage System or in variables passed to U405A. GFETCH is used to retrieve the data. The External Random Access file system can be accessed if needed; for instance, MOS forecasts to be analyzed may be in an external random access file. U405A can also write grids to Internal Storage for use by other routines or for another entry to U405A (see IWRITS in U155, Record Type 20).

DATA OUTPUT

There are seven forms of data output, besides the diagnostics and other information on Units KFILDO and IP( ). These generally pertain to all analysis modules, and are explained in U155. Differences are that the discontinuous variables do not have quality controlled observational data written, because there is no effective error checking possible. See the Data Output section of U155. (The discontinuous module U405D is not implemented.)

EXAMPLE CONTROL FILE: 'U405AMOSTMP^^^.CN'

An example exists as file 'U405ATMPMOS^^^.CN' in directory 'home21/mos2k/staging/glahn_1_9_08/dru155' on blizzard. The easiest way to set up a run is to take an existing control file and modify it.

OUTPUT:

Output is generally explained in the U155 writeup.

RESTRICTIONS

Restrictions in U155 apply to U405A.

COMMENTS

Comments in U155 apply to U405A.

Not all options have been thoroughly checked out. U155/U405A were adapted from the LAMP analysis modules, and some options were carried along that may not be needed for U155. Some of these, while still present in some degree of completeness, may need work to behave like a U155 user will need (for instance, the backup procedure involving IBACKN and IBACKL in Record Type 1).

Although the primary purpose of U155 and U405A is to produce packed grid-point data, writing of such data is not done if KFILIO = 0 (see U155, Record Type 8). A secondary purpose is to write vector records of observations in which all but the tossed obs have been set to 9999; however, this file will not be written if KFILQC = 0. That is, no output unit number and file name have been provided. A similar record is written to the same file in which all but the "questionable" obs have been set to 9999. Questionable is defined as not meeting a threshold which is a fraction of the primary error criteria ER1 read in U405AXXXXXXXXX.CN. The fraction can vary by pass, but is hardwired in FRACT( ) in BCD5 to 0.6. If this seems to be questioning too many values, set it to a larger number. This feature can be used for checkout runs. Note that rendering the unit number = 0 is done by omitting the file in the input .CN file, not by leaving the file name there and setting the unit number = 0. The records so produced can be used by the U203/ugem combination to plot a map of the tossed and/or questionable observations. This file could be used to summarize the data tossed over a sample period to help identify bad stations.

The modules of U155 (e.g., U405A) operate on grids of variable dimension. The grid length is specified at each stage of the process (see Appendix II for a discussion of this capability). For instance, the first guess and first pass of the sea level pressure analysis may be at 80 km.--possibly the mesh length of an NCEP or MOS forecast grid available for a first guess. Subsequent passes of the analysis may use other mesh lengths. U405A interpolates either linearly or quadratically to obtain a denser grid, or will use alternate grid points to thin a grid. It is important to note that the value of MESH is the mesh length of the active grid at that point and that XP( ) and YP( ) contain the IX and JY positions of the stations relative to that grid. That is, MESH and XP( ) and YP( ) follow the grid. Or you could think of the grid and XP( ) and YP( ) following MESH--same difference. Subroutines exist for these mesh length transformations which always have to be done in steps of a factor of two. Note that thinning of a grid cannot be done if NX-1 or NY-1 is not divisible by the necessary factors of two to transfer from one mesh length to another. It should be possible to use a mesh length as small as nominal 1.25 km; checkout has been done at 5 km. (Although using a larger mesh length for the first pass than later passes is attractive for cp efficiency, it has not worked well in practice, primarily because of using a lapse rate.)

It is assumed that the grid that is to furnish a first guess (e.g., an NCEP GFS model or a MOS generalized gridpoint forecast) is large enough to encompass the analysis grid. The analysis grid is defined by a lower left latitude and longitude and extents NXL and NYL at a "basic" mesh length MESHB. The first step in obtaining a first guess (assuming it is from a grid and not an average of observations or a constant) is to position the analysis grid within the first guess grid at the first guess grid's mesh length. The next step is to make the guess the desired mesh length.

There are four interpolation schemes employed in U155/U405A. They are:

(1) Bilinear or biquadratic, as specified by ITRPLQ(,) read in Record Type 7, when it is necessary to interpolate to get a grid of smaller mesh length. For instance, the incoming first guess might be at a 10-km gridlength and the analysis is to be made at 5 km. Routines DENSER and DENSRQ are used for linear and quadratic interpolation, respectively.

(2) The value at a station (observation location) as implied by the current analysis grid is needed in order to:
   a) error check the observation with respect to the current analysis,
   b) on occasion, buddy check an observation about to be tossed,
   c) to verify an unsmoothed analysis on each pass (when verification is specified, and
   d) to verify a smoothed analysis after each pass (when verification is specified.
   e) determine the correction that observation is to cause to the gridpoints within its radius of influence on each pass.

   For each of these except e), a special bilinear routine ITRPSL is used. The observation locations are specified in the station dictionary as either land, ocean, inland water, or both land and inland water. Also, each gridpoint is specified as land, ocean, or inland water. The idea is that the land stations will only affect land gridpoints and the water stations will only affect water gridpoints. However, some stations are "close" to both land and water, and are designated as "both." When a station is designated as land (ocean, inland water) only land (ocean, inland water) gridpoints will be used in the interpolation. When the four surrounding gridpoints are not all of the necessary type, interpolation will not be done, but rather the closest gridpoint of the proper type will be used. If none of the four is of the correct type, a larger set of points may be searched (see N4P, Record Type 1a). If a gridpoint of the desired type is still not found, that station will not be used. When the station is designated as "both," then there is no discrimination as to the land or inland water gridpoints; they can be whatever mix occurs.

For e), a routine ITRPSX much like ITRPSL is used, except instead of linear interpolation, the value from the gridpoint nearest in elevation to the station, from the four surrounding gridpoints, is taken, and then the station's lapse rate is used to get a value at the station. This became necessary because a station in a valley would cause an increase in, for instance, temperature on the surrounding mountains, but because of the lapse rate would be at a cooler temperature. Then with ITRPSL, the interpolated value would be cooler than the station, and further warming would occur. ITRPSX is an attempt to get a value representative of the location of the station. It is offset in the horizontal up to about 3.5 km on a 5-km grid; this will be only about a mile maximum on a 2.5 km grid.

(3) Other that (1) and (2) above, the quadratic interpolation routine ITRP is used. For instance, this is used in determining whether the difference in station and gridpoint elevation is within specified bounds (see HGTTHA and HGTTHB in Record Type 1a).

Use of JP(,N), where N is the variable being analyzed:

```
JP(1,N) controls gridprinting (IP22 must be ≥ 1)
JP(2,N) controls writing to KFILOG:
        IFSTGS must be ≥ 2 for first guess to be tdlpacked;
        NTDL(LP) must be ≥ 1 for analysis after pass LP to be tdlpacked;
JP(2,N) must be ≥ 1 for analysis for variable N to be tdlpacked;
JP(3,N) must be ≥ 1 for printing (not tdlpacking) of vector data destined
        for KFILOV and KFILQC
```

Writing to any file (KFILIO, KFILOG, KFILOV, KFILQC, KFILVO) will be done only when the file unit number is ≥ 0. This is achieved by omitting the unit number and file name in the .CN file, leaving only the terminator.

U405A assumes all gridded data needed are available as input. If computations have to be done, they must be preprocessed or modules added to U405A and the names read into PREPRO (Record Type 25). FSTGS5 could be modified for first guess fields to handle entries in ITABLE(,4,) (see Record Type 1c) if necessary. Vector data can be processed through a module called by OPTX, the usual vector data switcher for MOS-2000. Post processing for output can be handled by routines read into POSTAR and/or POSTDS (Record Types 26 and 27).

ILS read in Record Type 1a applies to all passes and controls how the water and land points are treated relative to each other. When = 0, all points are treated the same, and this may be appropriate for PoP where there are no good data over the ocean, and the values need to be extended there. When = 1, ocean, inland water, and land points are treated separately according to LNDWAT(,) (see Record No. 9), essentially making three analyses merged into one.

VARIABLES ACCOMMODATED BY U405A

U155 calls U405A once per variable and date/time when the associated control file name starts with 'U405A' (see ANLTAB( ) in U155, Record Type 21). Furnished to U405A is a 4-word MOS-2000 variable identifier for which an analysis grid is to be produced.

U405A makes sure the ITABLE(,1) read in U405A.CN is appropriate for the ID provided by U155. For each variable processed (analyzed), the name of its specific ".CN" file needed is available from U155 (e.g., 'U405ATMPMOS^^^.CN' for MOS temperature). This control file was explained earlier under "CONTROL FILE INPUT."

NONSYSTEM ROUTINES USED

Use the load line in one of the subdirectories (e.g., 'temp') in file 'home21/mos2k/staging/u405a_mos_alaska_10_20_08', dataset 'u155.com' on blizzard. The "alaska" just means this version works for Alaska as well as for the CONUS.

LANGUAGE: FORTRAN77 with some HP extensions.
LOCATION: u155lib. The driver is in one of the subdirectories (e.g., 'temp') in 'home21/mos2k/staging/u405a_mos_alaska_10_20_08' on blizzard.

APPENDIX I TO U405A

ANALYSIS PROCEDURE

The values of NTYPE(,) (see Record Type 4) are explained below:

0 = Skip this pass. Not usually used.

1 = Correction applied to each gridpoint is the sum of the individual differences between observation and analysis grid values interpolated to stations divided by the number of stations within a radius R of the gridpoint being corrected.

$$C_1 = \frac{1}{n}\sum_{i=1}^{n} (O_{x,y} - A_{x,y})_i \qquad d_i \le R$$

R = Radius of influence in grid units.
n = Number of observations within radius R of gridpoint being corrected.
$O_{x,y}$ = Observed station value at point x, y.
$A_{x,y}$ = Analyzed value of the variable interpolated from gridpoints to the point x, y.
d_i = Distance in grid units between the observation x, y and the gridpoint.

2 = Correction applied to each gridpoint is the same as type 1 correction except the distance of each station from a grid-point is considered in the correction for that station. When a type 2 correction is being used, lots of distant stations, even with a few close stations, will result in small corrections being applied to a gridpoint.

$$C_2 = \frac{1}{n}\sum_{i=1}^{n} \frac{R^2 - d_i^2}{R^2 + d_i^2} (O_{x,y} - A_{x,y})_i \qquad d_i < R$$

3 = Correction applied to each gridpoint is the same as type 2 correction except the sum of the distances of each station from a gridpoint is also considered in the correction.

$$C_3 = \frac{\frac{1}{n}\sum_{i=1}^{n} \frac{R^2 - d_i^2}{R^2 + d_i^2} (O_{x,y} - A_{x,y})_i}{\frac{1}{n}\sum_{i=1}^{n} \frac{R^2 - d_i^2}{R^2 + d_i^2}} \qquad d_i < R$$

By having the sum of weights in the denominator, a larger correction is applied to a gridpoint than with a type 2 correction. This may not produce good results when n is very small and R is large. Therefore, when NTYPE(,) = 3 is being used, for those gridpoints where n = 1 and R > 3, a type 2 correction is made; however water gridpoints on pass 1 (only) are an exception to this restriction. In other words, a Type 3 correction can be made for any n when R < 3 and for any R when n > 1.

SMOOTHING PROCEDURES

The smoothing parameter B(,) (see Record Type 5) is explained below:

$$S_{i,j} = \frac{A_{i,j} + b\overline{A}_{i,j}}{1 + b}$$

where:
$S_{i,j}$ = Smoothed gridpoint values at point i, j.
$A_{i,j}$ = Value of gridpoint at point i, j before smoothing.
b = Smoothing parameter.
$\overline{A}_{i,j} = (1/4)\ (A_{i+1,j}+A_{i-1,j}+A_{i,j+1}+A_{i,j-1})$

Low values produce the least smoothing and higher values produce heavier smoothing. A value of zero indicates no smoothing on this pass.

Values of NSMTYP and their uses are:

1 = The "normal" smoother defined above.
2 = If NSMTYP = 1 is used successively on several passes, areas with no data will be smoothed considerably from the first guess, which may already be quite good over the ocean, for instance. In order to control this, NSMTYP = 2 can be used, which does not correct a point unless one of the values involved in the smoothing was changed by the analysis (subroutine SMOTHN).
3 = NSMTYP = 3 gives a 9-point smoothing on the last pass only; otherwise it is the same as NSMTYP = 2. This is Shuman's smoother.
4 = NSMTYP = 4 is specialized for sea level pressure, and attempts to keep lows and troughs from being smoothed. If the pressure is $\leq$ 1013 mb, the point is the lowest in a sequence of points in eight directions, and the gradient is $\geq$ 0.012 mb/km over 240 km, then the point is not smoothed. In addition, the neighboring point is smoothed only 0.15 of the value it would normally be smoothed.
5 = NSMTYP = 5 designates a special terrain-following smoother. In general, a 5- or 9-point smoother is used in relatively flat terrain; no smoothing is done for a maximum or minimum terrain point; and when a line of three gridpoints in any one of eight directions of the compass is found over which the terrain is relatively the same, smoothing is done only along that line. Any other point is not smoothed.
6 = Two passes with NSMTYP = 5.
7 = Three passes with NSMTYP = 5.

An ocean (not internal water) smoother is provided as a postprocessing subroutine OSMTH. This will smooth ocean gridpoints over a square, the size determined by the penultimate value read in associated with POSTAR or POSTDS and the increment specified by the ultimate value (usually 1) (an increment of > 1 allows for not all points along the axes to be used, which may be time-saving). A similar routine, OSMTH1, performs the same function as OSMTH except smoothing is limited to an inscribed circle (diameter of the circle is the same length as the side of the square.

An ocean "ray" smoother is provided as a postprocessing routine ORSMTH. This will smooth along 8 equally spaced rays emanating from the gridpoint being smoothed, such that when a land or missing gridpoint is encountered, the ray stops. This keeps small bays (e.g., Prince William Sound) from being affected strongly by the open ocean. Following the ray smoothing, 2 passes of a 9-point square is applied to reduce grid-length jigger. This is used for Alaska, but is not needed for the CONUS.

PRIMARY AND OPTIONAL FIRST GUESS

The ID in ITABLE(,3) is the ID of the primary first guess field, if one is needed. It is used when IGUESS( ) = 2. So that the U405A.CN can be generic, the same DD and tau is used as the analysis to produce (from the U155.CN file).

The ID in ITABLE(,4) is the ID of a secondary first guess to use if the primary is not available, and is used with IGUESS( ) = 3. For instance, if the max temperature grid were not available as a first guess, the temperature at the same tau and DD could be used.

Note that the purpose of specific entries in ITABLE(,) can be different for different variables.

USE OF TERRAIN TO INHIBIT INFLUENCE OF DATUM ON ANALYSIS

For many weather elements, there is high spatial correlation in the absence of such factors as terrain and land/water abutments. The analysis procedure relies on this correlation to "spread" the value of a datum to surrounding gridpoints within a specified radius of influence. However, differences in elevation can violently disturb this small scale correlation to a point that a temperature, say, in a valley may not be well correlated with the temperature in neighboring valley or on a neighboring mountain even when those points are within the radius of influence.

U405A can use the terrain to limit the effect of a datum on a gridpoint in two ways:

1) When the datum elevation and the gridpoint elevation differ by more than HGTTHA (see Record Type 1a), the datum will not influence the gridpoint.

2) When the datum and the gridpoint are separated by terrain exceeding HGTTHB (see Record Type 1a), the datum will not influence the gridpoint. The elevation between the datum location and the gridpoint is found by following a line from the station to the gridpoint, and at

intervals of MESHE (the terrain mesh length) from the datum, interpolating into the terrain grid.

When this feature is not desired, set HGTTHA and HGTTHB to some large value, like 20,000. So far, this feature has not been necessary. The use of HGTTHB would be quite cp intensive because a "line of sight" would have to be computed for every station to every gridpoint it might influence for each analysis pass, at least until the value HGTTHB was encountered.

USE OF LAPSE RATE ADJUSTMENT

A terrain adjustment procedure has been implemented. There are two options.

(1) The first option, LAPFG = 1 (see Record Type 1), used if the density of stations permits, is essentially, a lapse rate in units of the variable being analyzed per kilometer can be used when adjusting a gridpoint. These values, one per station, are computed by the subroutine LAPSE. (Note that this is not a "free air" lapse rate, but only a value determined by the data to be used in the terrain adjustment algorithm.) A program U175 exists to determine for each station in its directory a set of points around it that differs in elevation as much as possible at a minimum radius. U175 writes out the list of stations and its neighbors. This list is read by U155 (Record Type 15) and applies to all analyses of the U155 run. U405A, entered once for each variable being analyzed, uses the pairs list for each station to compute the average lapse rate around the station, when IBKPN NE 99 and file KFILLP has been provided.

A lapse rate to be used for each station is calculated by summing all the differences of the variable being analyzed for the pairs (defined by the U175 list) and dividing by the sum of the differences in terrain elevation between the pairs. Note that this is not the same as calculating each individual lapse rate between the pairs and averaging.

The lapse rates can be applied in any fractional amount by pass according to ELCORR(,) read in Record Type 11. There is also additional control for "unusual" lapse rates (see next section).

(2) The second option, LAPFG = 2 (see Record Type 1), is used in Alaska where the density of stations is too sparse to compute a value based on station values. Here, upper air data are used. For each station, values have been read from a file that define the lowest and highest elevation of any gridpoint that will be affected by the station. The lapse rate is computed as the difference between the value of the variable being analyzed (so far, this has only been used for analyzing temperature, dew point, max and min temperature, and wind), on the upper air grid closest above the station and the variable value on the upper air grid closest above the highest elevation to be affected by the station, divided by the elevation difference. This option differs from option (1) above in that this really is a computed free-air lapse rate.

(3) The second option, LAPFG = 3 is the same as LAPFG = 2 except the value of the variable being analyzed at the station is used at the lower level rather than the value from the upper air. This option seems preferable, although on the testing done there is not much difference between LAPFG = 2 and 3.

UNUSUAL LAPSE RATES

For each variable, the lapse rate will have an "expected" sign. For instance temperature usually decreases with elevation, but snow amount is more likely to increase with elevation. When the lapse rates are reversed to these expected signs, such a lapse rate may be quite localized and should not be used over a wide area. This is what has been found with temperature, for instance, along the California seacoast. The temperatures may increase inland, even though the elevation also increases.

It was necessary to limit the effect of unusual lapse rates, and to do so IBKPN, BK(,), and ELCORU(,) were added. IBKPN indicates, in effect, which sign of lapse rate is unusual, and BK(,) and ELCORU(,) indicates how to handle an unusual lapse rate.

ADJUSTMENT FOR LAND/SEA ABUTMENTS

When a datum is over land, either an observation or a forecast, then it may not have a lot of small scale correlation with a similar observation or forecast over adjacent water, even within the radius of influence, and vice versa. To try to overcome this discontinuity in correlation, the following procedure is used by U405A when ILS = 1 (see Record Type 1a) and LANDWAT(,) = 1 (see Record Type 9) for a specific first guess and analysis pass.

The control file read with Unit No. KFILD(2) (see U155 writeup) includes a parameter called a sea/land descriptor. When it is a "0," the location is ocean and the datum will affect only ocean gridpoints. When it is a "9," the location is definitely land and will affect only land gridpoints. When it is a "3," the location is inland water and will affect only inland water gridpoints. When the value is "6," the datum is very close to an inland water/land boundary, and the datum will be allowed to affect both land and inland water (currently not ocean) gridpoints.

The definition of land/water gridpoints is contained in the "land/sea mask" located in the same MOS-2000 external random access file as the terrain. This mask has a 0 for ocean water, a 9 for land, and a 6 of inland water. So, the gridpoints are definitely either land, ocean, or inland water, but the data locations can be either or "both" land and inland water.

The capability of the ocean values to fractionally “bleed” into the land and vice versa has been incorporated with WTWTL (weight water to land) and WTLTW (weight land to water) (see Record Type 1b), respectively. There is no bleeding when ILS = 1, LNDWAT(,) = 1, WTWTL = 0, and WTLTW = 0. When, for instance, WTWTL is set to some factional value, then the water values will affect land gridpoints by that fractional amount. The count of values affecting the gridpoint is also weighted by the same amount. As usual, the effect is within the radius of influence, distance and elevation weighted. So far, this capability looks too unstable for the temperature suite, but is used for wind.

USE OF DATA QUALITY FLAGS

Record Type 1a contains quality control parameters for four categories of data quality. The parameter IQUALC determines whether (=1) or not (=0) these qualities are to be used. The four values of QUALWT( ) are fractional weights to apply to the correction procedure for the four qualities of data. One would expect, the highest quality would have a 1.0 specified, and the others might be less even to being zero. This means that once the correction to be made at a gridpoint due to a particular datum is determined, that correction will be multiplied by QUALWT( ) for that data type.

The quality of a datum is specified in the Control file on Unit No. KFILD(2) (see U155) by station. Each station will have a value 1, 2, 3, or 4, if the quality is good enough to be used; a higher value says, in effect, "don't use me." These four values apply correspondingly to the four values in QUALWT( ).
In the station dictionary, the 6 digits usually reserved for the Block/Station Number are used for quality control flags (values 0-4) and the type of station [ocean (0), land (9), inland water (3) or inland water and land (6)]. The readers employed in INT155 (rdstqn or rdstqa) bring these back as IQUAL( ); most readers don’t return the values in Block/Station Number field. U155 works this way.

INT155 parses the six rightmost digits in the Block/Station Number field into six values, starting from the RIGHTmost:

LNDSEA(K)	indicates the type of station K (see above),
IQUAL(K,J)	a quality number pertaining to this station (J=1,5). This can be 0, or 1 through 4.

In the specific variable .CN file, the first value read in Record Type 1a is IQUALC. This is a value 1, 2, 3, or 4 and refers to the quality control flag column in the dictionary.

The next four values in Record Type 1a are weights (QUALWT(J), J = 1,4) that can be applied to a specific station’s data in the analysis of that specific variable. These four weights are keyed to the 4 possible values in IQUAL(K,M), where M = IQUALC (either 1, 2, 3, or 4). As a special case, if IQUAL(K,M) = 0, the weight is zero and does not depend on QUALWT(J).

So, for instance, station K could have IQUAL(K,1) = 1 in the dictionary. When running temperature, IQUALC could = 1, and the weight to apply to that station would be QUALWT(J) where J (1 through 4) is the value in IQUAL(K,1). Another station could have another value J in IQUAL(K,1) for a different weight for that station. Or, IQUAL(K,2) could be used with IQUALC = 2.

When running PoP, the values in QUALWT(J) can be different because the control file is different. IQUALC can be either 1, 2, 3, or 4.

This arrangement does not give unlimited flexibility, but probably provides all that is needed at this time–at least all the flexibility we can/want to describe.

USE GEOSTROPHIC WIND FOR SEA LEVEL PRESSURE ANALYSIS

For SLP only (not currently implemented), in order to try to get and preserve low pressure in tight lows and troughs, the surface wind at a station is used together with its pressure to make a correction according to the geostrophic relationship under certain circumstances. For such a correction to be applied, the values of WNDWT and WNDGRD must not be zero and the pressure at the gridpoint for which the correction is to be made must be < 1013 mb. In addition, the gridpoint value must be a minimum in either the X or Y direction and the gradient over 240 km must exceed .012 mb/km. This is calculated by finding the difference between the gridpoint and the minimum of the two end points, each 240 km away. This difference must be exceed .012 mb/km. If the gridpoint is to have a wind correction, the neighboring point in either direction also gets a weight of 75 percent of the full correction. A gridpoint is allowed a maximum of three wind corrections per pass.

USE OF A STATION VALUE TO NUDGE THE CLOSEST GRIDPOINT TOWARD THAT VALUE

When ISETP = 1, after the last pass, the closest gridpoint to each station is found and the gridpoint value is nudged toward the station value, but with the aim of not creating a discontinuity or bull's-eye in a graphic display. It is changed by most 0.9 units, and never crosses in integer values. Examples:

Station Value	Gridpoint Value	New gridpoint value (ISETP = 1)	New gridpoint value (ISETP = 2)
50.5	53.2	50.9	50.5
"	52.9	50.9	"
"	50.0	50.5	"
"	50.1	50.5	"
"	50.9	50.5	"
"	48.5	50.1	"
"	51.1	50.5	"

ERROR CHECKING PROCEDURE

Generally, a value (observation or forecast) is not used if it differs from the current analysis [defined to be the interpolated value from the grid (first by subroutine ITRPSL, the by ITRP if necessary), see Comments Section above] by ER1 units of the variable being analyzed for that pass and first guess option. However, if it meets 1.5*ER1 and one of its two nearest neighbors also differs from the analysis by ER1*F (where F is hardwired to 0.6) and in the same direction, then, the observation is accepted. The neighbor assisting the observation being checked will also not be discarded if it meets the 1.5*ER1 criterion. This process (a specialized "buddy check") requires nearest neighbors to be found (a very time-consuming process) only infrequently. For observations, this check is quite good for sea level pressure, and marginally so for temperature and dew point. MOS forecasts should not have any bad data unless the equations are unstable or a bad ob was input; however, bad values do occur.

Another check is made for sea level pressure only when the pressure is < 1013 mb and the observation is about to be tossed. Before a station is tossed, its pressure is estimated from each of its two closest neighbors by using the neighbor's pressure and surface wind. If the wind is < WNDTHR, the wind is not used and the check is not made. A factor of 2 is applied to the calculated geostrophic gradient to account for the reduction in surface wind due to friction. If the estimate from either station is within ER1 mb, the station is kept. This check does not affect the acceptance of the neighbors. A main purpose of this check is to attempt to keep good observations near hurricanes and other intense lows.

As an added feature, for sea level pressure only, if the observation is < 970 mb and the observation is 50 mb lower than the analysis, then 100 mb is added to the observation, and if the result is within ER1 mb, the observation is modified and accepted. Also, if the observation is > 1040 mb, and the observation is 50 mb higher than the analysis, then 100 mb is subtracted from the observation, and if the result is within ER1 mb, the observation is modified and accepted. Otherwise, the observation is left alone and not accepted. This check was important in the manual observation days when it was easy to tabulate an observation as, say, 940.5, instead of 1040.5, but may not be very important today.

As explained in the U155 writeup, a record can be written for each pass in which all except the tossed obs have been set to 9999. This allows GEMPAK plotting routines to isolate by color the tossed obs vs the good obs. In addition, a record can be written for each analysis pass in which all except "questionable" obs have been set to 9999. This allows GEMPAK plotting routines to isolate by color those observations which, while not tossed, are questionable. "Questionable" is defined by pass as a function of ER1, the fraction being hardwired in subroutine BCD5 (currently 0.6). The "ER1Q" thresholds depend on variable being analyzed and pass number, but have been hardwired to 0.6 of the ER1 criteria read in U405AXXXXXXXXXX.CN for the 6 possible passes. These records, with an ID that varies by pass number, are for tuning the analysis procedure. They may also be useful for quality control of data; MOS forecasts or observations frequently tossed or even questioned for a station may indicate a recurring problem with that station/element. A bad data value can potentially cause a bad lapse rate for several stations. A safety feature is to limit the value of the lapse rate to values between HMINUS and HPLUS (set in a data statement in LAPSE), currently - 0.1, and +0.1, respectively. If this range is exceeded, the lapse rate is set to zero, meaning the elevation will have no effect on the correction caused by this station. (It is recognized that legitimate lapse values can vary by weather element, but the range is not large, and this crude check is deemed sufficient, especially because the thresholds are very subjective.)

In addition, a single record can be written for each analyzed variable in which the obs tossed on <u>the last pass</u> have been set to 9999 (see KFILQC in U155). This record has an ID composed form the ITABLE(,2,) with the VVVBDD replaced from the ID of the variable being analyzed. This would allow for spatially quality controlled obs to be used in a downstream program.

<u>USE OF MULTIPLE CYCLES IN ONE ANALYSIS</u>

When comparing the analysis of MOS forecasts for a particular projection from a particular cycle to a similar analysis made for the previous cycle for forecasts verifying at the same time, one notes:

1) There is great overall, perhaps even surprising, consistency in the forecasts, at all projections. This augurs well for the accuracy of the models and the post-processing techniques.
2) There is some vacillation from cycle to cycle. This has been noted by forecasters for years, and may be due to differences in numerical model accuracy depending on start time or/and differences in the postprocessing specifics, including sample size and different predictors in the equations.
3) The stations for which there are forecasts are not the same from cycle to cycle. This leads to considerable vacillation at those particular spots from cycle to cycle.

It is not likely that exactly the same stations having forecasts can be arranged and maintained, so the vacillation caused by having a station at a location on one cycle and not the next will undoubtedly cause consternation when viewing and looping graphics. In order to combat this, U155/U450 will combine forecasts from more than one cycle in one analysis; up to five are accommodated, although it is expected for MOS only two will be used. Each set of forecasts (of the same variable and ID) can be designated (see NHRRUN( ), Record Type 1b) in terms of a number of hours previous to the current cycle time, and the projection will be adjusted to match. Each such forecast can be weighted so that the most current cycle could be weighted more heavily than the older one (see NHRRUA( ), Record Type 1b). [If needed, a weighing can be also be applied in computing the lapse rates (see NHRRUL( ), Record Type 1b); presently, all cycles used are weighted equally].

While it might be that combining an older forecast with a newer will degrade the analysis and hence the accuracy of the forecasts, it is also possible the combination will improve the overall analysis in the spirit of ensembles and averaging of forecasts of similar accuracy. Withheld data tests have shown that the overall accuracy for temperature in terms of MSE is about the same with two combined cycles as without, and combining greatly improves temporal consistency. It also doesn't seem to matter much whether the two cycles, 12 apart, are weighted 1 to 1 or 2 to 1 (current cycle weighted twice as much as the one 12 hours old).

AUGMENTING ONE SET OF FORECASTS WITH ANOTHER

There are fewer LAMP stations with forecasts than MOS. The analyses of some variables do not look reasonable with only the LAMP list, and there are too few stations in some areas to calculate a good lapse rate. The MOS stations are used to augment the LAMP ones in the following way. A preprocessor U179 is run to find for every MOS station up to MSTA (a control in U179) stations that can have both a MOS and a LAMP forecast. These are written to a file read by U155 with the preprocessor routine AUMGT1. When an analysis is made, a station with a LAMP forecast is, of course, left that way. A MOS station without a LAMP forecast is entered into the analysis by adding to the MOS value the average of the difference of the MOS and LAMP values at the MSTA stations. This is in effect determining whether the LAMP indicates the MOS value is too high or too low for that area, according to LAMP, then adjusts the MOS value accordingly. This seems to work OK and gives an acceptable pattern. LAMP has no water stations, so MOS values are used there. That means that the land/water distinction can be made, and that the analysis over water will be exactly the same as MOS, provided the relevant analysis parameters are the same. This procedure is justified; LAMP is an update to MOS.

ANALYSIS OF PROBABILITY FORECASTS

Station values of points on a CDF are being produced by a technique called EKDMOS, in which an equation developed on means of predictors from ensembles is applied to each ensemble member, and kernel density fitting applied. These are element values (e.g., temperature) and represent, for instance, the forecast value below which 10% of the observations are forecast to occur. These values can be analyzed for multiple percentage values. U155/U405 has been adapted so that multiple probability levels are essentially analyzed together, in that the lapse rate is calculated from them all and applied to each. Because the same stations will be present for each probability level (for a particular date and projection), the process should be linear, and the values from the CDF are non-decreasing, so the grids should exhibit that characteristic. To analyze the maps, the IDs must be in sequence; they are judged a sequence by the CCCFFFs matching, and the LLLL in Word 2 of the ID being between 0 and 100.

WITHHOLDING DATA

Data can be withheld by subroutine WTHOL1 for purposes of doing withheld data tests when NWITA > 0 and ITYPR = 1. Subroutine FITWTH is used to compute statistics for both the withheld and non-withheld stations.

Data can be withheld by subroutine WHTOL2 for outputting data for analysis by U600 in developing an equation for error estimation when NWITH > 0 and ITYPR = 2. The data are written to IP24 in TDLPACK. The differences between the data values withheld and the interpolated analysis vales are written for U600 predictand data. Computations are made at those same withheld data locations of such thing as terrain roughness (subroutines called by WHTOL2) to be used as predictor data. U600 then produces an equation that can be used at each gridpoint to get an estimate of the analysis error (note this does not include errors of the observational error (when analyzing observations) through postprocessing subroutine ER-EST. These errors are written to the archival file on Unit No. KFILIO.

PREPROCESSING ROUTINES:

Some input data for analysis are preprocessed. This might be a simple change of scale in order to operate better with U405A diagnostics or be more complicated to produce a new variable for analysis. Those in use are:

AUGMT1 – Augments one set of forecasts with another, specifically augmentation of LAMP forecasts with MOS.

SCALX – Scales the input by a power of 10. For instance, PoP is input as decimal and is scaled by 10**2.

SCALXI – Same function as SCALX, except additionally rounds to whole numbers.

SCLSNO – Six snow categorical forecasts and associated probabilities are used to produce a value to analyze in inches *10 (tenths of inches).

SCLQ06 – Six 6-h QPF categorical forecasts and associated probabilities are used to produce a value to analyze in inches *10**2 (hundredths of inches).

SCLQ12 – Seven 12-h QPF categorical forecasts and associated probabilities are used to produce a value to analyze in inches *10**2 (hundredths of inches).

Only one such preprocessing routine can be used per variable being analyzed. It is likely this will have to be modified so that AUGMT1 can be used in conjunction with another variable.

COMPUTATIONAL POSTPROCESSING ROUTINES WHEN ICOMPT = 1 (see U155):

Some variables are not actually analyzed but are computed from other analyses, in the spirit of U155/U405A being a general postprocessing program. ICOMPT = 1 signals this option, and considerable input in U405A.CN is not used, but the format of the file must be maintained. Those in use are:

DIRSPD – Processes u- and v- wind and wind speed:
1) Calculates direction from u- and v-wind.
2) Sets direction missing when both the u- and v-wind < 0.1 kt.
3) Sets speed < 0.1 kt to zero.
4) Sets direction to zero when speed $\leq$ 1 kt.

QPF6P6 – Computes 12-h QPF from two 6-h values. 12-h QPF can be analyzed, but it was decided to compute it from 6-h values.

GENERAL POSTPROCESSING ROUTINES

The ROUTINES below are used to process the analyzed grids for output:

POST – Scales a variable to a power of ten and truncates at both or one end, depending on input.

OSMTH – Smooths ocean values over a square, sise depending on input.

OSMTH1 – Smooths ocean values over a circle, size depending on input.

ORSMTH – Smooths ocean values on 16 rays outward from the point being smoothed. A ray stops when a land point is encountered.

HSMTH – Smooths all points over a square, size depending on input.

CKTDP – Checks dewpoint analysis with analyzed temperature. If dewpoint exceeds temperature, the dewpoint is set to the temperature.

CKPOP – Checks 12-h PoP with two 6-h PoPs covering the same period. The 12-h PoP is constrained to the sum of the two 6-h values and the larger of the two.

CKQPF – Checks two QPF fields, one analyzed with terrain correction and one without. Any value < TRUNC has to exist on both analyses, or it is set to zero. TRUNC is an input variable to CKQPF.

CKMAX – Checks daytime max temperature with all 3-h values covering the daytime period. If any 3-h value exceeds the max, the max is set to the 3-h value.

CKMIN – Checks nighttime min temperature with all 3-h values covering the nighttime period. If any 3-h value is less than the min, the min is set to the 3-h value.

CKWNDG – Checks wind gust grid (actually total wind) with wind speed grid. If the speed exceeds the gust, the gust is set to the speed.

CIGFT – Changes ceiling height in categories to hundredths of feet.

VISMI – Changes visibility in categories to miles.

SETCFT – Assures, for ceiling height, that the closest gridpoint to each analyzed point (station) value is within the forecast category at that station.

SETVMI – Assures, for visibility, that the closest gridpoint to each analyzed point (station) value is within the forecast category at that station.

SKYAMT – Changes cloud amount in categories to percent coverage.

<u>SCALX</u> – Scales the input by a power of 10. For instance, PoP is analyzed as decimal and is scaled by 10**-2 for output in fractions. Also used as a preprocessing routine.

<u>CONCPR</u> – Makes cumulative ceiling height probabilities non-decreasing for increasing thresholds.

<u>CONVPR</u> – Makes cumulative visibility height probabilities non-decreasing for increasing thresholds.

<u>CONEKD</u> – Makes EKDMOS grids non-decreasing for increasing levels of probability.

<u>EREST</u> – Returns an estimated error grid. Associated with hourly analyses.

APPENDIX II TO U450A

U155 Grid Size and Gridlength Analysis

U155/U405A can use several grid sizes and gridlengths in its various computations. The below attempts to explain some of the intricacies of the process.

MESHB, NXL, and NYL are read from the U155.CN file. This will be the size (NXL by NYL) of the analysis grid at "nominal" meshlength MESHB. MESHB is a moniker for the exact meshlength BMESH, which is found by subroutine ACTUAL called by subroutine INT155. For instance, a Lambert grid with a MESHB of 5 km will have a BMESH of 5.07941 km. This grid has a lower left corner point at latitude/longitude ALATL/ALONL, also read from the U155.CN file. Subroutine ACTUAL uses the map projection NPROG read in the U155.CN file.

MESHE, NXE, and NYE are read from the U155.CN file. This will be the size (NXE by NYE) of the constant grids (terrain, masks, etc.) at "nominal" meshlength MESHE. MESHE is a moniker for the exact meshlength EMESH, which is found by subroutine ACTUAL called by subroutine INT155.

MESHL is read from the U155.CN file. This is the "nominal" gridlength of any subsetted (quality control) grid output of a "continuous" field. MESHL is a moniker for the exact meshlength XMESHL, which is found by subroutine ACTUAL called by subroutine INT155. The subsetted area is specified by MXGMIN, NXGMAX, NYGMIN, and NYGMAX (in terms of grid columns and rows) read in control file U155.CN.

(MESHD is read from the U155.CN file. This is the "nominal" gridlength of any subsetted grid output of a "discontinuous" field. MESHD is a moniker for the exact meshlength DMESH, which is found by subroutine ACTUAL called by subroutine INT155. A routine for analyzing discontinuous fields has not been implemented.)

The positions of the stations in terms of the NXL by NYL grid, XPL( ) and YPL( ), are calculated from the latitudes and longitudes in STALAT( ) and STALON( ) by subroutine XYCOM1 called by U155 and are permanent.

Subroutine GTHRES reads the constant "high resolution" grids of size NXE by NYE and cuts them if necessary to the analysis grid area NXL by NYL. The gridlength read must match EMESH for the grid to be returned. The orientation, map projection, and latitude of "tangency" are checked to agree with those read from the U155.CN file to be used in the analysis. The gridlength of the constant grids is not changed to match BMESH. GTHRES will gridprint to unit number IP22 and output to unit number KFILGO a subsetted area if desired on the area read in U155.CN at nominal gridlength MESHL.

The positions of the stations in terms of the constant grid, XPE( ) and YPE( ), are calculated linearly and correspond to MESHE, NXE, and NYE. (I say "linearly" because the calculation does not require the expensive conversion from latitudes and longitudes.) They are permanent, as XPE, YPE, and MESHE do not change.

MESH is introduced in subroutine FSTGS5 called from U405A. It is the gridlength of the first guess grid returned (whatever it may be) and is the value of MSHPAS(1,MGUESS) read from the U405A.CN file., where MGUESS is the actual first guess option used. (I say "actual" because the guess option specified by the U405A.CN may not be available, and an alterative choice is used, as specified in the same control file.) As such, it is the mesh length to be used in the first pass of the analysis. This does not have to be equal to MESHB, but must be consistent to a power of two (for the actual grid length, not necessarily the nominal gridlength). FSTGS5 returns the first guess grid in P(NX,NY), where NX and NY correspond to MESH. The positions of the stations for this "current grid" are calculated from XPL( ) and YPL( ) linearly and correspond to MESH, NX, and NY. They "follow the grid" and are recalculated whenever (if ever) MESH, NX, and NY change because of a different value in MSHPAS(,).

MESH can change according to MSHPAS(LP,MGUESS) for each pass LP. If there is a change, NX, NY, XP( ), and YP( ) are recalculated in BCD5.

Gridprinting and writing to unit number KFILOG with subroutine PAWGTS is done at MESHL over the (possibly subsetted) area specified in the U155.CN file in terms of grid rows and columns. (The "subsetted" grid can be the whole grid.) This output can be unsmoothed, smoothed, or both.

Final grids can be packed and written by subroutine PAWLPM to internal storage according to the switch IWRITS (which can be different for different weather element analyses) at gridlength MESH. This is before postprocessing and clipping, but after the normal smoothing; these grids can be used in future U155 computations if needed. They are packed because the full set of metadata needs to be carried along.

The writing to the archive file unit number KFILIO, by PAWGTS (as a result of the last pass analysis) is at gridlength MESHB in U405A after all analysis passes in BCD5. If MESH ≠ MESHB, then the NX by NY grid is converted to MESHB by subroutine SZGRDM. This is after any postprocessing specified in U405A.CN and clipping to the output by mask CPNDFD by setting values likely to be incorrect to missing (9999).

The writing to RA file, unit number KFILRA( ) = 42 with subroutine PACKGR also is at gridlength MESHB.

In summary, MESHB in U155.CN is the desired, final gridlength for output. Other grid sizes and gridlengths are for internal use or for quality control output. BCD5 handles the quality control and subsetted grids. U405A does the final output. It is not likely MESH will change over the course of the analysis (this has been shown to be generally unproductive), but MESH could be < MESHE.

APPENDIX III TO U405A

Considerations in Using Preprocessor U174, U178, and U179

U174 – Finds Pairs for Lapse Rate Computation

U174 reads a station list and a station directory. The directory can be complete and universal. The purpose of U174 is to provide pairs of stations that can be used to calculate a "lapse rate" in order for U155/U405 to make an elevation correction. For every station in the station list, a list of $\leq$ 100 stations is generated; these are called pairs–the station with each of the stations in the list is a pair. For each station, these pairs are generated by making multiple passes over the meta data for combinations of distance between the pair and elevation difference between the pair. The first search is made for a small horizontal distance and a large elevation difference. The combination of distance and elevation difference are modified per pass by increasing the distance and decreasing the elevation. A total of 21 passes are currently used with hardwired combinations of distance and elevation difference multipliers.

The distance and elevation difference can be partially controlled by the inputs XDIST (in meters) and NDEMIN (in gridlengths), respectively. The first pass will be made with a distance of XDIST*0.5 and an elevation difference of NDEMIN*7.0; the last pass is with a distance of XDIST*2.25 and an elevation difference of NDEMIN*1.3. Normally, NDEMIN is set to 100, so the smallest vertical difference used between stations is 130 meters. XDIST can be set based on the number of items in the station list. For a large list, XDIST might 30, for a small list, 60. That is, the smaller the list, the greater the distance necessary to find pairs.

When a pass is completed, any station having 60 or more pairs is not searched for again. In the searches, if 100 pairs is reached, no further processing for that station is done, so not all pairs may be found for that pass. This is likely not a concern because the stations found on previous passes are saved for the next one. In that way, the most important pairs are always retained. A print after each pass showing the status of pairs is provided so that the quality of the searching can be judged. It is likely the NDEMIN = 100 should remain fixed as any vertical separation of less than 130 meters would not be meaningful.

Setting XDIST and the station list to use is more problematical. U174 generates for each station in the list up to 100 pairs. They are written in binary and keyed to the locations in the list. When U155 is run, these must be keyed to the station list used in U155. It was thought this should not be done for each analysis, but once per run of U155, so the list is read in and processed in U155. That means one list of pairs is used for all analyses performed in that run. If all such analyses had about the same amount of data, that would be no problem, but for MOS the station list for max and min temperature, for instance, is much larger (> 10,000) than for temperature (less than a third of that). When a station in the U155 list does not occur in the U174 list, the lapse rate for that station is not (cannot be) calcu-lated, and is set to zero. This argues for running U155 for temperature, dewpoint, etc. with a small number of stations separately from max and min.

However, he consistency checking in U155 depends on them being in the same run. Given that, the station list for U174 should be the max/min list. By so doing, the actual pairs for temperature and dew point will be reduced to about a third of the total list. This may be OK, the number of pairs being generally greater than 20 after the reduction. The alternative would be to use the external random access file (RA), rather than the internal one, to retrieve the needed data for checking. In operations, this should be OK, as output is to the RA file anyway. Another possible alternative would be to allow more that 100 pairs with a minimum of > the current 60, but this would increase running time. (It is not known whether RA can be used by multiple MOS-2000 programs simultaneously.)

U178 - Establishes Variable Radii by Station

U178 reads a station list and a station directory. The directory can be complete and universal. The purpose of U178 is to create for each station up to six radii to use in U405A in analyzing the data. This is the so called "variable radius" option. If it is not used, then the radii (constant for every station) is read from the U405A.CN control file. U178 creates the list in two steps. First, for every gridpoint, the MAXSTA stations nearest it within a radius of XDIST gridlengths (not necessarily the same value as in U174) are found and saved, along with the distances to the gridpoint and the station elevation. Then, these lists (one for every gridpoint within the NDFD area[1]) are searched for the stations in the station list and the largest distance becomes the 1st pass search radius for that station. This guarantees that every gridpoint has MAXSTA stations affecting it on the first pass, providing there are MAXSTA within the distance XDIST. Because each gridpoint is involved in the U178 searches, it takes considerably longer to run than U174.

The search radii in U405A are quite important, and should be calculated with a station list that approximates the one that will be used in the analyses. If a station being analyzed with data is not in the U178 list, the search radii default to the constant values in U405A.CN. This would not be a catastrophe, but is not optimum. If the station list used in U178 is much too large (larger than the one used in the analysis), then there is no guarantee that every gridpoint will be affected by MAXSTA stations. A list that is slightly too large will also not be a catastrophe.

U178 has been run with MAXSTA = 15 and XDIST = 175 for the CONUS. A value of XDIST that large is used to accommodate water points which are considered in the same run of U178. However, the lists generated separate land, ocean, and inland water points. The radii for a water station will likely be = XDIST, because there will almost certainly not be MAXSTA values found, and it needs to be large enough to find at least one water station for each water gridpoint.

U178 provides for an override to the calculated radii. This was necessary for Alaska, but has not been necessary for the CONUS. For Alaska, bogus stations had to be created, each a function of other specific stations' values, and the

[1] The constraint to the NDFD area was removed for hourly analyses.

bogus stations were used in the station list input to U178. Even so, some very specific radii had to be specified (for instance, to keep Bering Sea bogus values from affecting the Pacific Ocean south of the Aleutian peninsula and chain.

<u>U179</u> - Finds Pairs for Augmentation

U179 reads two station lists and a station directory. The directory can be universal and complete. The purpose of U179 is to identify small sets of stations, MAXSTA in number, around each station in the short list that are in both the short and long list. The sets of stations are separated by land, ocean, and inland water.

The purpose so far is to augment LAMP forecasts with bias corrected MOS forecasts. However, there could be other uses, for instance to insure a constant set of value to analyze. There are fewer dewpoint forecasts than temperature forecasts. Dewpoint values could be augmented at locations were there is no forecast by computing an "offset" between surrounding temperature and dew point stations and plugging in a "bogus" dew point value. Subroutine AUGMT1 could probably be used without change.

U179 is a fairly simple process. The value of MAXSTA should probably be in the range 3 to 6. For LAMP, I recommend 4.

U110

GENERATES GRID POINT ID’S AND THEIR LOCATIONS FOR A GRID SPECIFICATION

Mitchell Weiss
April 21, 2008

PURPOSE:

U110 generates a list of grid point IDs and their locations for a given grid specification. Two ASCII output files are generated; one in station table format and the other in GIS “text” format. Both files contain the grid point ID, and each grid point’s latitude and longitude. The station table file’s latitudes and longitudes are all listed as positive values with a one character hemispheric notation attachment. The GIS “text” file’s latitude and longitude are listed with their actual earth-oriented values. U110 generates files for Polar Stereographic, Lambert Conformal, and Mercator projections. The user has the option of either listing the grid specifications directly onto the control file (NVALL=0) or reading the grid specifications from an external file (NVALL=1), see below for the definition of NVALL. U110 is written to run on a 32-bit or a 64-bit machine.

Some familiarity with other MOS-2000 documents will be helpful for a full understanding of this write up.

CONTROL FILE INPUT: 'U110.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3)

This record contains unit numbers for the run, and can even control the default output file number. KFILDI is the input unit number as specified in DRU110.

IPINIT – Four characters, usually a user’s initials plus a run number to append to “U110" which also identifies a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U110 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = ‘HRG2' and IP(1) = 40, the file name for Unit No. 40 = ‘U110HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U110 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information. It may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). One value has been identified as indicated for values of J:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used un-changed, as specified in DRU110, where KFILDO is set to 12. Changing the default unit number allows multiple runs of U110 or other programs within the same directory without overwriting.
(2) = The input file name (DATNAM) containing the name of the file containing the requested grid. The file is requested only when NVALL=1. When there are errors, print will be unit KFILDO as well as to unit IP(2).
(3) = The name of the requested grid (AWGRID). The grid name is only requested when NVALL =1. When there are errors, print will be unit KFILDO as well as to unit IP(3).
(4) = The specifications of the requested grid. Output will include the value of NVALL, the map projection, numbers of rows and columns, the mesh length, lower left corner latitude and longitude, tangent latitude, orientation longitude (not required for Mercator Projections), and the value of L3264B. These values are printed for NVALL = 1 or 0. When there are errors, print will be unit KFILDO as well as to unit IP(4).
(5) = A printout of the time zone values and their corresponding counter position value within the grid. The counter position increments from the lower left hand corner of the grid and counts up (left to right) along each succeeding rows, until the last point (upper right hand corner) is reached. This output may produce a voluminous amount of printed records depending on the size of the grid.
(6) = A printout of the terrain height values and their corresponding counter position value within the grid. The counter position values increment the same as IP(5). This output may produce a voluminous amount of printed records depending on the size of the grid.
(23) = Information concerning the opening and closing of files.

Record Type 2 - Format (A72)

This record is used to identify the run.

RUNID -   72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (5(I10/)) **Run Control Parameters**

This record contains one control value for the run (NVALL) and eight grid specification values which immediately follow. Each value is on a separate line. Brief comments may be added to each line to assist the user in identifying each grid specification value.

NVALL -   When NVALL = 0, the requested grid specifications must be listed in the eight lines which follow. IF NVALL =1, the grid specifications are obtained from an external file, and the eight values listed below are **not** used. (I10)

MAPRJ -   Indicates the map projection of the grid; 3 for Polar Stereographic, 5 for Lambert Conformal, and 7 for Mercator. The value listed on this line is only used when NVALL = 0. (I10)

NUMX - Number of rows in the X coordinate of the grid. The value listed on this line is only used when NVALL = 0. (I10)

NUMY - Number of columns in the Y coordinate of the grid. The value listed on this line is only used when NVALL = 0. (I10)

XMESH - The mesh length of the grid (meters). The value listed on this line is only used when NVALL = 0. (F10.3)

XLATLL - The lower left corner latitude of the grid. The value listed on this line is only used when NVALL = 0. (F10.4)

XLONLL - The lower left corner longitude of the grid. The value listed on this line is only used when NVALL = 0. (F10.4)

XLAT - The tangential latitude, the latitude at which XMESH applies. The value listed on this line is only used when NVALL = 0. (F10.3)

ORIENT - The orientation longitude(not required for Mercator Projections), the longitude of the central meridian of the grid. The value listed on this line is only used when NVALL = 0. (F10.3)

Record Type 4 - Format (I3,4X,A60) **File Containing the Grid Name**

This record (plus the terminator record) identifies the name of the file from which the grid name is read. If there is more than one grid name in the file, the last grid name is selected. The terminator KFILDT( ) = 99 follows the file name. Record Type 4 is read by subroutine RDSNAM. NVALL =1 must be specified. If NVALL=0, this record is omitted.

KFILDT - Unit number of the file containing the input grid name.

DATNAM - Name of file where the grid name resides. When KFILDT = KFILDI, DATNAM is not used and the grid name can be read as “Default”, see Record Type 5. (CHARACTER*60)

Record Type 5 - Format (A20) **Grid Name**

This record (plus the terminator record: 99999999) identifies the name of the requested grid. If more than one grid name is listed, the last grid name is selected. If KFILDT (read in Record Type 4) is not equal to KFILDI, Record Type 5 is omitted. NVALL =1 must be specified. If NVALL=0, this record is omitted.

AWGRID - The grid name of the requested grid, which is used to obtain the specifications of this particular grid from an external file (see Record Type 6). Grid names are read until the terminator 99999999 is reached. Remember, if there is more than one grid name listed, the last one listed is chosen. (CHARACTER*20)

Record Type 6 - Format (I3,4X,A60) **Grid Specification File**

This record (plus the terminator record) identifies the external file, which contains the grid specifications for a list of grids. The terminator KFILSPEC( ) = 99 follows the file name. Within this file, records are read until the requested grid name (AWGRID) is reached. The grid specifications are then read and the program continues. If the requested grid name is not found in the external file, the program will stop with an error message. This Record Type 6 is read by subroutine RDSNAM. NVALL =1 must be specified. If NVALL=0, this record is omitted.

KFILSPEC - Unit number of the external file containing the list of grids and their specifications.

SPECNAM - Name of the external file of grid specifications.(CHARACTER*60)

Record Type 7 - Format (I3,4X,A60) **GIS "Text" Time Zone Gridded Input File**

This record (plus the terminator record) identifies the input file of time zone values for each grid point represented by the specified grid. The input text file is in a format used for display within ArcGIS. The values from this file are matched to their respective station grid ID and are written to both the station table and GIS "text" output files (see below). The user may choose to not write time zone data to the output files. The terminator KFILTMZN( ) = 99 follows the file name. Record Type 7 is read by subroutine RDSNAM.

KFILTMZN - Unit number for the GIS "text" time zone gridded input file.

TMZNNAM - Name of the ASCII GIS "text" input file. (CHARACTER*60)

Record Type 8 - Format (I3,4X,A60) **GIS "Text" Terrain Height Gridded Input File**

This record (plus the terminator record) identifies the input file of terrain height values for each grid point represented by the specified grid. This input text file is in a format used for display within ArcGIS. The values from this file are matched to their respective station grid ID and are written to both the station table and GIS "text" output files (see below). The user may choose to not write terrain height data to the output files. The terminator KFILTRHT( ) = 99 follows the file name. Record Type 8 is read by subroutine RDSNAM.

KFILTRHT - Unit number for the GIS "text" terrain height gridded input file.

TRHTNAM - Name of the ASCII GIS "text" input file. (CHARACTER*60)

Record Type 9 - Format (I3,4X,A60) **Station Table Output File**

This record (plus the terminator record) identifies the output file of grid point IDs, latitudes, and longitudes for the requested grid in the format of a MOS-2000 station table. Values of either time zone, terrain height, or both may also be included in the output file (see Record Types 7 and 8). The terminator KFILIO( ) = 99 follows the file name. Record Type 9 is read by subroutine RDSNAM.

KFILIO - Unit number for the ASCII station table output file.

OUTNAM - Name of the ASCII station table output file. (CHARACTER*60)

Record Type 10 - Format (I3,4X,A60) **GIS “Text” Output File**
This record (plus the terminator record) identifies the output file of grid point IDs, latitudes, and longitudes for the requested grid in the format of a GIS “text” file. Values of either time zone, terrain height, or both may also be included in the output file (see Record Types 7 and 8). The terminator KFILGIS( ) = 99 follows the file name. Record Type 10 is read by subroutine RDSNAM.

KFILGIS - Unit number for the ASCII GIS “text” output file.

GISNAM - Name of the ASCII GIS “text” output file. (CHARACTER*60)

EXAMPLE CONTROL FILE: 'U110.CN'

Examples exist for file 'U110.CN' in directory /mdl1/save/lmp/we21mw/u130/dru110 on the IBM. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

All diagnostic output is handled through IP( ) values, Record Type 1, as specified in the control file U110.CN. All errors will be written to IP(1) or the default output file if IP(1) is not used. Every effort has been made to notify the user of problems and potential problems and proceed under control. All diagnostic output is in ASCII format for viewing by the user.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a “D” in Column 1. The IBM does not allow this. It is best to replace the “D” with, for instance, “C****” in columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

DO NOT USE unit number in the range of 45 through 49, since they are restricted for use with random access files.

COMMENTS

When NVALL=1, the grid specification listed in the control file is not used. A file containing the requested grid name (Record Type 4) or a default listing of the grid name (Record Type 5), and the external file containing a list of grids and their specifications (Record Type 6), **must** be included in the control file. When NVALL =0, the grid specification in the control file is used and Record Types 4-6 **must** be omitted from the control file.

The user may choose to write either the time zone or terrain height or both input files to the station table and GIS text output files. The user may also choose to write neither of these input files.

The grid point ID identifies the location of a grid point on a grid. Starting from the lower left corner of the grid, the 8 character grid point ID allow for four digits for the X component and Y component respectively. The grid point IDs increment upward (northward) along succeeding columns until the upper right hand corner of the grid is reached. Examples can be found in /mdl1/save/lmp/we21mw/u130/dru110.

When using a GRIB grid specification, longitude values must be changed from earth-oriented values (west negative/east positive) to the MDL standard where west longitudes are positive and east longitudes are positive values greater than 180 degrees. The GRIB grid specification for latitude (north positive/south negative) is consistent with the MDL standard.

Most errors are output for printing starting with **** to IP(1) or the default diagnostic output file.

Example of a grid name in the grid name file:

```
AWIP 253 253
AWII 202 202
99999999
```

(The grid name AWII 202 202 is chosen)

SETTING UP THE DRIVER DRU110

The preparation of the driver for a particular U110 run is relatively painless. It consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine.
ND1 -	Maximum number of grid points that can be used.
ND2 -	X-dimension maximum size of the grid that can be used.
ND3 -	Y-dimension maximum size of the grid that can be used.
ND4 -	Not used.
ND5 -	Not used.
ND6 -	Maximum number of input data sets that can be used.

ND7 -	Not used.
ND8 -	Not used.
ND9 -	Not used.
ND10 -	Not used.
ND11 -	Not used.
ND12 -	Not used.
ND13 -	Not used.

Do not change the computation or value of the variable L3264W. The user can see from the DRU110 template what effect this values has on storage from where it occurs in the DIMENSION statements.

The user can see from the DRU110 template what effect each of these values has on storage from where it occurs in the DIMENSION statement. Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

NONSYSTEM ROUTINES USED

On the IBM, non-system MOS-2000 subroutines are used in /mdl1/save/mos/mos2k/moslib.

LANGUAGE: FORTRAN77 with some FORTRAN-90 compliant HP extensions.

LOCATION: The complete path for U110 code and U110-associated subroutines is /mdl1/save/lmp/we21mw/u130/dru110 on the IBM.

U138

CONVERTS SEQUENTIAL TDLPACK DATA INTO GRIB2

David E. Rudack
July 22, 2008

PURPOSE: U138 is designed to convert gridded sequential TDLPACK data into GRIB2. U138 performs the same functionality as U135 but differs with respect to the input file type. U138 processes TDLPACK gridded sequential file(s) while U135 processes a gridded external random access file. U138 supports the conversion of data on North polar stereographic, Lambert conformal, and Mercator map projections. U138 also accommodates a bitmap field as input in order to "mask" portions of the processed grid(s) prior to being packed into GRIB2. U138 outputs a GRIB2 file according to the grid specifications set forth in the grid definition file. While U138 currently supports specific Product Definition and Data Representation Templates, other templates may be added if necessary. U138 uses a driver DRU138 so that dimensions of variables may be tailored by PARAMETER statements by the user without requiring a separate copy of the program U138 for every application. U138 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. Some familiarity with GRIB2 Document FM 92 GRIB, and other MOS-2000 documents will be necessary for a full understanding of this write-up.

A full description of U138 is not included here because it would be essentially a copy of U135. Rather, only the differences are described.

Record Type 6 - Format (I3,4XA60) **TDLPACK Input Gridded Sequential File(s)**

This group of records identifies the TDLPACK data set from which the data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

KFILIN( ) - Unit number(s) for the gridded TDLPACK input sequential file(s). Any unit number may be assigned to KFILIN( ) as long it does not conflict with any other unit number being used elsewhere in U138.CN. Note that because each input file is processed separately, all input files may posses the same unit number.

NAMIN(J) - Name of file corresponding to file KFIIN( ). (CHARACTER*60)

DATA INPUT:

All input data to be processed by U138 must be in sequential gridded TDLPACK format. The input unit number(s) and file name(s) are provided in KFILIN( ) and NAMIN( ) (see Record Type 6). The definition of the grid to which the data in each record pertain is contained in that same record. If the user desires to mask a portion of the input data, a gridded random access file may be defined in U138.CN (on unit number KFILBM) (see Record Type 13 in the U135 write-up).

A. SEQUENTIAL GRIDPOINT DATA

One or more sources of sequential gridpoint data are accommodated, the data set names and unit numbers having been provided by NAMIN(J) and KFILIN(J), respectively from the control file 'U138.CN', Record Type 6. Each file is closed when EOF is reached.

B. RANDOM ACCESS GRIDDED BITMAP DATA

Only one bitmap (in TDLPACK format) is accommodated, the data set name and unit number having been provided to BMPNAM and KFILBM respectively, from the control file 'U138.CN', Record Type 13 (see the U135 write-up). This data type is used to mask a portion of the TDLPACK data field. The data on this file (generated by U361) **must** have the same output grid characteristics as those stipulated in the grid definition file. Otherwise, U138 gracefully terminates.

DATA OUTPUT:

There is one form of data output, besides the diagnostics and other information on Units KFILDO and IP( ). An output GRIB2 file must be defined in U138.CN; otherwise, U138 will not execute.

BINARY GRIB2 FORMAT

All gridded output data are packed in GRIB2 format. They are packed by using NCEP's GRIB2 encoding modules (GRIBCREATE, ADDGRID, ADDFIELD, and GRIBEND). The data are written to KFILGO using subroutine WRYTE. All data are written to the dataset whose name has been provided to OUTGRD and unit number to KFILGO from the control file 'U138.CN', Record Type 7 (see U135 write-up).

EXAMPLE CONTROL FILE: 'U138.CN'

An example exists as file 'U138.CN' in directory /mdl1/save/mos/mos2k/u13xlib/ on the IBM. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more files as described above under control file 'U138.CN', Record Type 1 (see the U135 write-up), and the definition of KFILDO in the driver DRU138. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control without jeopardizing data.

RESTRICTIONS:

Since U138 does not utilize the features of MOS-2000 linking data type to specific unit number type, the user may use any desired unit number (even values between 42-49, which are normally reserved for external random access files). This statement remains true as long as the input file's unit number(s) do not conflict with an IP( ) value or another other unit number for a different record type.

Since U138 does not utilize the MOS-2000 mass storage system:

1) All sequential input files **must** be placed in the control file in chronological order. In addition, all variables that will be processed for a specific date/time **must** be on the same file. They may not be spread over two or more files.

2) All variables that are processed in the element list file (Record Type 11 - see the U135 write-up) will be written to the GRIB2 file (KFILGO) in the order it is encountered on the input sequential file(s). Consequently, the order of the variables listed in the element list file may not match the order of the records written to the output GRIB2 file.

3) Each GRIB2 message can only contain one variable (field). Each variable (for Sections 4 and 5) must be delineated with a "777777" (see Record Type 9 in the U135 write-up for variable set-up). If a group contains more than one variable, U138 terminates with a diagnostic error message.

SETTING UP THE DRIVER DRU138:

The preparation of the driver for a particular U138 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the IBM when used with certain compiler options).
ND4 –	Maximum number of bitmap variables that can be defined in the bitmap variable list. (Maximum should be set to one)
ND5 –	Dimension of IPACK( ), IWORK( ), and DATA( ). Note that ND5 must be greater than or equal to NX*NY.
ND6 –	Maximum number of TDLPACK input gridded sequential files allowed in one run of U138.
ND7 –	Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). Should be greater than or equal to 54.
ND8 -	Maximum number of dates that can be dealt with in one run of U138. Should be set to a value of one.
ND10 –	Maximum number of fields allowed in any one specific group (GRIB2 message) in one run of U138.
ND11 –	Maximum number of output grid entries defined in the grid definition file corresponding to Section 3 of the GRIB2 message.
ND12 –	Maximum number of groups (or GRIB2 messages) that can be processed in one run of U138.
LCGRIB –	Maximum length (bytes) of the GRIB2 message (stored in character array CGRIB( )).
IGDSTMPLEN –	Maximum dimension of IGDSTMPL( ) located in the Grid Definition Section of the GRIB2 message (Section 3).
IPDSTMPLEN –	Maximum dimension of IPDSTMPL( ) located in the Product Definition Section of the GRIB2 message (Section 4).
IDRSTMPLEN –	Maximum dimension of IDRSTMPL( ) located in the Data Representation Section of the GRIB2 message (Section 5).
IDEFNUM –	The entries in array IDEFLIST( ); i.e., the number of rows (columns) for which optional gridpoints are defined. (Used if IGDS(3).NE.0)
NUMCOORD –	Number of values in array COORDLIST( ).

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND5). Every effort has been made so that the variables will not be overflowed if values too small are used. The user should **not** change the values of the last five parameters noted above. These values have been carefully chosen to work within the framework of U138.

The "startup" control file "U138.CN" is opened in DRU138, so that this aspect of U138 can be rather easily changed without recompiling or having separate versions of the main (sub)routine U138. This is different on the IBM where unit/file assignments are made differently and possibly when U135 is run in batch mode.

NONSYSTEM ROUTINES USED

U138 is essentially a wrapper program that utilizes NCEP routines that pack and write GRIB2 data. Consequently, U138 is intended to be run on the IBM. The following NCEP libraries are used to link with U135 and must be placed in the makefile:

```
/nwprod/lib/libg2_4.a
/nwprod/lib/libw3_4.a
/nwprod/lib/libbacio_4.a
/usrx/local/64bit/lib/libjasper.a
/usrx/local/64bit/lib/libpng.a
/usrx/local/64bit/lib/libz.a
```

LANGUAGE: FORTRAN

LOCATION: U138lib. The driver is in /mdl1/save/mos/mos2k/u13xlib/RUN/dru138.f

U201

PREPARES MOS-2000 PREDICTOR/PREDICTAND DATASET

Harry R. Glahn
January 1, 1995

PURPOSE: U201 is the first in a series of programs designed to develop statistical relationships, usually regression equations, between predictands and predictors from archived data. The primary input data, gridpoint fields from numerical models, are packed in a GRIB-like TDLPACK format. Because the packed data are self describing, the gridpoint data can come from any model or models, the number being essentially unlimited. Vector data, also in TDLPACK format, are accommodated. In addition, constant data can be provided in random access files; these data are also packed in the TDLPACK format. U201 uses a driver DRU201 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U201 (actually, subroutine) for every application. U201 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. The output of U201 is a TDLPACK dataset containing values at specific points, usually station locations, which is in the same format as the possible vector input. (Output can be on a grid, where gridpoints are treated as "station" locations.) The first record in each output dataset is the station directory (usually) containing station call letters. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U201.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU201.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U201" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U201 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U201HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U201 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. Values have been defined as indicated below for values of J:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU201 DATA statement = 12. Changing the default unit number allows multiple runs of U201 or other programs within the same directory without overwriting.

(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).

(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).

(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).

(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).

(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.

(7) = The variable ID list in summary form. If there are errors, the predictor list will be written to the default output file unit KFILDO as well as to unit IP(7).

(8) = The variable ID list in summary form after reordering low to high. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)

(9) = The variable ID list in summary form after reordering. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the predictor constant file on unit KFILCP (see below).

(10) = The variable ID's for the first day (Day 1) as read from the archive tapes. This is just a list of all the packed gridpoint fields on the input files.

(11) = The variable ID's of the archived fields actually needed, in order as they appear on the archive files for Day 1.

(12) = The I,J positions of the stations on the NGRID grids, together with the call letters and names. NGRID, at this point, is the number of different grid combinations extant in the input data. That is, NGM archived data and MRF archived data will have different grid definitions. After the data are read for the first day in the date list, the I,J positions are calculated and saved; IP(12) controls the printing of those values. If, during the U201 run, a different grid definition is encountered, it is accommodated automatically, but no print of the I,J positions is provided. In addition, when IP(12) ≠ 0, each directory record read from a vector input will be output on unit IP(12). For input of hourly data, this creates voluminous output.

(13) = Gridpoint fields. When the variable list indicates gridpoint values are to be written for viewing (JP(1,)>0), they will be written to unit IP(13).
(14) = Gridpoint fields. When the variable list indicates gridpoint values are to be contoured and written for viewing (JP(2,)>0), they will be written to unit IP(14).
(15) = Interpolated values. When the variable list indicates interpolated values are to be written for viewing (JP(3,)>0), they will be written to unit IP(15).
(16) = Diagnostics for constant and linearization routines (e.g., stations in threshold lists that are not being dealt with in this run).
(17) = The values of IFIND( ), ISTAV( ), and ITIME( ) after Day 1. (See Comments section for explanation.)
(18) = The variables saved in the MOS-2000 Internal Storage System after Day 1.
(23) = Information concerning opening and closing of sequential files.

For checkout, it may be advisable to set all these values to zero or the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) **Run Identification**

This record is used to identify the run.

<u>RUNID</u> - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (7(I10/),F10.0) **Control Parameters**

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

<u>KSKIP</u> - When nonzero, KSKIP indicates that the output file is to be moved forward until all data for date KSKIP have been skipped. KSKIP is input as either YYMMDDHH or YYYYMMDDHH and then used as YYYYMMDDHH.

<u>KWRITE</u> - The existing call letters record (directory) pertaining to the data for the date/time KSKIP (see above) will be checked with the one available for writing. If they match, the new one will not be written; if they don't match, the new one will be written when KWRITE = 1, but the program will halt with a diagnostic when KWRITE = 0. Note that this has no effect when KSKIP = 0.

<u>NSKIP</u> - The number of errors that will be tolerated on Day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results, except when <u>no</u> data are found for Day 1, the program halts after Day 1.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - The increment in hours between date/times that are put into IDATE( ) (see Record Type 5) as a result of date spanning in subroutine DATPRO. This variable is also used in determining what data are to be saved in the Internal MOS-2000 Storage System. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL to the first date/time in IDATE( ).

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

PXMISS - The value to substitute in the output for a secondary missing value of 9997. This allows the 9997 to be maintained if desired, set to zero in the case of the implied value of near zero for forecasts, or even set to some other value. Note that this does not apply to missing values other than 9997.

Record Type 4 - Format (I3,4XA60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

NAMDT - Name of file where this date list resides. When KFILDT = KFILDI, NAMDT is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) **Date List**

This group of records determines the date/times for which interpolated output is wanted. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input.

Terminator is 99999999. Maximum number of dates, sans terminator, is ND8. Data for the first date in the list <u>must</u> be available or U201 stops. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL (read in Record Type 1) to the first date/time in IDATE( ).

Record Type 6 - Format (2I3,1XA60) <u>Gridpoint and Vector Input Data Files</u>

This group of records identifies the sequential data sets from which the gridpoint and/or vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN. If all input data are from a constant file, only the terminator is necessary.

<u>KFILIN</u>(J) - Unit number for the input gridpoint or vector file. KFILIN( ) must be < 80 for gridpoint data and $\geq$ 80 for vector data. Also note that values 97, 99, and 44 through 49 are reserved for other uses.

<u>MODNUM</u>(J) - Source of gridpoint data that are on this file (e.g., 6 = NGM gridpoint data, 7 = eta model, 8 = aviation model). This number is used to match with the "DD" in the variable ID. For vector data, this value <u>must</u> be zero.

<u>NAMIN</u>(J) - Name of file where this model input resides. (CHARACTER*60)

Note that when data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number. Having the same unit number is not mandatory, but is highly recommended. Vector data can either precede, follow, or be mixed with gridpoint data sets in the list read; the only restriction is that those with the same unit number are in proper sequence.

Record Type 7 - Format (I3,4XA60) <u>Random Access Files</u>

This group of records identifies the random access data sets from which constant data are read. Records are read until the terminator KFILRA( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND12 (ND12 $\leq$ 6). This Record Type 7 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMRA. If no data are needed from a constant file, only the terminator is necessary.

<u>KFILRA</u>(J) - Unit number for the random access constant file (J=1,NUMRA). Unit numbers must be in the range 44 to 49; see "Restrictions" for more information.

<u>RACESS</u>(J) - Name of file corresponding to KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) <u>Vector Output File</u>

This record (plus the terminator record) identifies the output file. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

<u>KFILIO</u> - Unit number for the output file.

<u>OUTNAM</u> - Name of file where this output is to reside. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) <u>Station and Location Files</u>

This pair of records (plus the terminator record) identifies the file(s) from which the station (or location) information is obtained. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

<u>KFILD</u>(J) - Unit number for the file containing the station call letters for which interpolation is to be done (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the default input file number, KFILDI, in which case DIRNAM(1) is not used. Also, KFILD(1) can equal KFILD(2), in which case the call letters list will consist of all stations on the directory file; note that both a list of stations <u>and</u> a directory cannot exist on unit KFILD(1) = KFILD(2). (Normally, the directory would not reside on KFILDI, but is accommodated if KFILDI = KFILD(2). The input would follow Record Type 10.)

<u>DIRNAM</u>(J) - Name of file matching KFILD(J). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (7(A8,1X)) <u>Station List</u>

This group of records identifies the stations for which output is desired. If KFILD(1) ≠ KFILDI (the default input file), this group is omitted, and the information is taken from another source.

<u>CCALL</u>(K) - Call letters (or other 8-character location designator) of stations (or locations) for which output values are desired (K=1,NSTA). This list is read within subroutine RDSTAD or RDSTAL by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations in the list are kept, but a diagnostic is furnished on unit IP(5). For NALPH = 1, the stations are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory are put at the end of the list. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank (b) filled on the right, e.g., 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) <u>Variable File</u>

This record (plus the terminator record) identifies the file from which the variable ID's are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

<u>KFILP</u> - Unit number for the variable ID's.

<u>PRENAM</u> - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) <u>Variable Constants File</u>

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description and gridprinting information.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

<u>KFILCP</u> - Unit number for the constants.

<u>CONNAM</u> - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 13 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2) <u>Variable List</u>

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 13 is read by subroutine RDPRED. Upon completion of reading this record type, N=1,NPRED.

<u>ID</u>(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only) (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

<u>JP</u>(J,N) - For each variable N, JP(,N) indicates print or no print when ≠ 0 or = 0, respectively, for gridpoint values (J=1), "zebra" gridprint map (J=2), and interpolated values at the stations (J=3). These values combined with IP(13), IP(14), and IP(15) allow easy control of output. For instance, all variables could have JP(2,) ≠ 0 and IP(14) ≠ 0 for a diagnostic run. All such print could be turned off by setting IP(14) = 0 in this control file. Alternatively, IP(14) could be ≠ 0 and a selected gridprint obtained by changing JP(2,) from 0 to ≠ 0.

CONTROL FILE INPUT: (Name read from U201.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) **Date List**

When the dates are not provided in file U201.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU201), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U201.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided on file 'U201.CN', this group of records identifies the stations (or locations) for which interpolated output is desired. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from KFILDI. It is also not needed when KFILD(1) = KFILD(2); for this highly unusual case, both the station list and the associated information are read according to the format in the following control file description for Unit No. KFILD(2).

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which interpolated values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 9, Control File 'U201.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U201.CN) (Unit = KFILD(2))

Record Type 1 - Format **Station Locations**
(A8,1XA8,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. It is needed unless KFILD(1) = KFILD(2) = KFILDI; for this highly unusual case, both the station list and the associated information are read from the input control file KFILDI. When KFILD(1) = KFILD(2) ≠ KFILDI, this control file is the same as the one above and both the station list and the associated information are taken from this file. If both files are needed (the usual case; KFILD(1) ≠ KFILD(2) ≠ KFILDI), the call letters here are matched with those in the station list and the appropriate information extracted. When this station directory is alphabetical, the final list of stations will be alphabetical no matter the order read in. (Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run.) However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited, except when it also constitutes the list to be kept, in which case the number of entries is limited by ND1-1. No terminator is used. Most of this information is used only within subroutine RDSTAD or RDSTAL to give information about the stations. For example, there is no use of the elevation in the equation derivation. However, this information is available for a computation routine, if one is needed. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see Record Type 3).

<u>CCALLD</u>(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list unless KFILD(1) = KFILD(2), in which case these call letters comprise the station list and at the completion of reading this file, K=1,NSTA. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

<u>NAME</u>(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

<u>NELEV</u>(K) - Elevation of the station. Format is I5.

<u>SIGNLA</u> - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

<u>XLAT</u> - Latitude in degrees. Format is F7.4.

<u>SIGNLO</u> - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

<u>LONDD</u> - Longitude in degrees west. Format is F8.4.

<u>CCALLD</u>(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

<u>IWBAN</u>(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited, except when it also constitutes the list to be kept, in which case the number of entries is limited by ND1-1. No terminator is used.

CONTROL FILE INPUT: (Name read from U201.CN) (Unit = KFILP)

Record Type 1 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2) Variable List

When the variables to use are not in file 'U201.CN', this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted, and the variable list is taken from input file KFILDI. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDPRED. Upon completion of reading this record type, N=1,NPRED. See Record Type 13, file 'U201.CN', unit KFILDI, for other details; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U201.CN) (Unit = KFILCP)

Record Type 1 - Format
(3(I9,1X),I3,2XA32,I3,F11.4,F10.4,F9.2,F8.2,2XA12) Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all MOS-2000 users and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be very unusual for that to be the case. Note that the format matches that for a file input to other programs such as U600, U660, and U850.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are al-lowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 charac-ters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALT - When IDTEMP(1) matches an ID(1,N), ISCALT is stored in ISCALD(N). This variable is the scaling constant (power of 10) to use when packing the interpolated data.

SMULTT - When IDTEMP(1) matches an ID(1,N), SMULTT is stored in SMULT(N). This variable is the multiplicative factor (power of 10) to use when gridprinting the gridpoint data.

SADDT - When IDTEMP(1) matches an ID(1,N), SADDT is stored in SADD(N). This variable is the additive factor to use when gridprinting the gridpoint data.

CONTT - When IDTEMP(1) matches an ID(1,N), CONTT is stored in CINT(N). This variable is the contour interval to use when gridprinting the data. It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT.

ORIGNT - When IDTEMP(1) matches an ID(1,N), ORIGNT is stored in ORIGIN(N). This variable is the contour origin to use when gridprinting the data It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT.

UNITST - When IDTEMP(1) matches an ID(1,N), UNITST is stored in UNITST(N). These characters define the units of the data after application of SMULTT and SADDT and are used in the visual inspection of the data in gridprinted maps. (CHARACTER*12)

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDPRED, much of it associated with the plain language description. See Chapter 4, Variable Identification, for details.

DATA INPUT:

All archived model and vector data input to U201 will be in the MOS-2000 TDLPACK format read from sequential files, the vector data being accompanied by one or more directory records. The unit numbers and file names are provided in KFILIN(J) and NAMIN(J) (J=1,NUMIN), respectively. Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format, except for the call letters (directory) record. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG.

A. SEQUENTIAL GRIDPOINT DATA

One or more sources of gridpoint data are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U201.CN', Record Type 6. Each file is closed when an EOF is reached.

The definition of the grid to which the data in each record pertain is contained in that same record. The interpolation routines use this information, and precompute station locations (convert latitude/longitude to I/J) so that this computation doesn't have to be done for each record. However, a check is always made to assure that the correct grid information is used. In fact, not only can the data sources have different grid definitions, the definition within a particular source can change once or many times. It might be normal for the grid definition to change during the archive period. Be aware, though, that each time the grid definition changes when reading records that are to be used, extra computation must be done.

Normally, ND11 (see "Setting up the Driver DRU201") will be $\geq$ the number of input grid sources (unit numbers).

For gridpoint data, the unit number KFILIN( ) must be < 80 (Nos. 44 through 49 are reserved for random access data), and the model number MODNUM( ) must match the model number DD in the ID's for which data are to be taken from that unit. The RR is operative for the lookback feature.

B. SEQUENTIAL VECTOR DATA

One or more sources of vector data are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U201.CN', Record Type 6. Each file is closed when an EOF is reached.

Each source (file) must have a directory record at the beginning, and one or more other directory records can occur in the same file following a trailer record (see MOS-2000 Software Documentation, Chapter 6). Each file, even read on the same unit number, must have a directory record at the beginning. The occurrence of a new directory record causes some additional computation.

For vector data, the unit number KFILIN( ) must be $\geq$ 80 (Nos. 97 and 99 are reserved for other uses) and the model number MODNUM( ) (see Record Type 6) must be zero. The RR is (is not) operative when the DD in the variable ID is (is not) zero. In this way, hourly data, which will have DD = 0, will have RR operative, while output from a previous run of U201, which will (usually) have DD > 0, will not have an operative RR. (This feature is associated with the saving of data internally so that the data will be available when necessary. It will have to be watched carefully for possible unexpected results. It is believed that, for instance, the most output from U201 that will have DD = 0 will also have RR = 0, so it won't matter whether RR is operative or not. <u>However, output hourly data with RR > 0 run back through U201 may create a problem.</u>)

C. RANDOM ACCESS VECTOR DATA

Up to 5 sources station-oriented random access data are accommodated. These data exist in the MOS-2000 External Random Access File System. These data are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth work can contain a threshold with the "B" in the first word a zero.

In addition, station elevation, latitude, and longitude are available from Unit Number KFILD(2) and are stored permanently for use in NELEV( ), STALAT( ), and STALON( ), respectively.

D. RANDOM ACCESS GRIDPOINT DATA

Gridpoint data can be accessed from the external random access files with Unit No. 44. Normally this would be constant data such as terrain height.

DATA OUTPUT

There are two forms of data output, besides the diagnostics and other information on Units KFILDO and IP( ).

A. ASCII FOR VIEWING OR PRINTING

Gridprinted maps of each variable for which JP(1,) ≠ 0 without contours and/or for which JP(2,) ≠ 0 with contours are written to Units IP(13) and IP(14), respectively, when those units numbers ≠ 0. Interpolated values are written for each variable for which JP(3,) ≠ 0 to Unit IP(15) when IP(15) ≠ 0. This makes for easy control; all such print can be eliminated by changing the IP( ) values, and there is still control on each variable J through JP(,J).

B. SEQUENTIAL BINARY MOS-2000 FORMAT

All interpolated data from gridpoint sources and data from vector sources are packed in the MOS-2000 TDLPACK format. They are packed with subroutine PACK1D and its associated subroutines by calling PACKV. All data are written to the dataset whose name has been provided to NAMOUT and unit number to KFILIO from the control file 'U201.CN', Record Type 8, unless KFILIO = 0, in which case data are not output. The first record will consist of the "station directory" of call letters, 8 characters written in binary (in subroutine SKIPWR) so that they can be read into and manipulated in an INTEGER array. Each record designated in Record Type 13, control File U201.CN, will cause an "interpolated" record to be written for each date/time even though all values might be designated as missing. Follow-on programs will then always have an expected record even though the values might be "missing." This includes constant data taken from the MOS-2000 External Random Access File System as designated in Record Type 7. An exception is when a computational subroutine passes IER = -1 back through OPTION to PRED21 or PRED22; in that case, the record is not written. Finally, at the end of the run, a trailer record is written.

EXAMPLE CONTROL FILE: 'U201.CN'

An example exists as file 'U201.CN' in directory 'users.tdl.glahn.dr201' on the GDP. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U201.CN', Record Type 1 and the definition of KFILDO in the driver DRU201. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control without jeopardizing data.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Only north polar stereographic and Lambert gridpoint data are accommodated. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The CRAY does not allow this. It is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

The TDLPACK format for gridpoint data provides for both a primary and a secondary missing data indicator. If a primary value is found during unpacking of gridpoint (not vector) data, this is flagged at U201 completion by a statement on the output KFILDO file. However, the computations on grids do not provide for missing values, and the packed output records will not indicate that missing values may be present. It is not expected that gridpoint data will have missing values. If missing values become possible at some time, then U201 will have to be modified to handle the situation. However, if the missing gridpoints are in an area where no interpolation to stations is to be done (e.g., in the Pacific in the lower left of the grid), no adjustment would need be made. It's only when the missing data could be involved in the bilinear or biquadratic interpolation would there be a problem.

Arrays FD1( ), FD2( ), ... FD7( ) have been provided for general work arrays. Arrays FDVERT( ) and FDTIME( ) have been reserved for use by subroutines VERTP and TIMEP, respectively. FDSINS( ) and FDMS( ) have been reserved for use by routines dealing with the sine of the latitude and a map factor. L1DATA( ) and SDATA1( ) are reserved for use by linearization and constant routines. Some care is needed when writing new and complicated computational subroutines that the FD1( ), FD2 ( ), etc. arrays are not overwritten in other routines. For instance, if a routine were to call OPTN2 which needed to call VORTH, then VORTH would use arrays FD1( ), FD2( ), and FD3( ). Anything the calling program had in these arrays would be wiped out.

The binary indicator "B" in the first word of the variable ID must be only 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), or 5 (indicating a grid binary). Subroutine CKIDS, called from RDPRED, prints a diagnostic if a binary is not indicated as a 1 or 5, but the variable is not eliminated. Therefore, if a point binary is made from gridpoint data by U201, it is assumed it is for a predictor, not a predictand. Normally, binaries (except grid binaries) will not be made in U201 because follow-on programs can do this. In fact, in U600 for example, if a binary is used and the basic variable to make it from is available on the U201 output, the binary will be made in U600 even though the binary were actually on the U201 output; the binary on U201 output would be used only if the basic variable were not available there.

Since vector data are allowed as input, and even the output of a previous U201 run, it not known *a priori* whether the computational elements of the ID (e.g., "B" or "RR") have already been used in preparing the data associated with the ID or whether these elements need to be operative. U201 tries initially to find the full ID on the input (computational elements not to be operative), then if the data cannot be found, the computations are done on the more basic data which must be available. With this flexibility comes the restriction that the "T" (transformation) and "B" (binary) capabilities cannot be operative on input vector data. For instance, suppose the sine of the day of the year were input as a vector. It could not be made into a binary or transformed with the "T". However, if it were computed through OPTION, it could be transformed or made into a binary. While this is a restriction, it is thought to be unimportant, because the T is seldom used and the making of binaries should, in general, be done in follow-on programs, not in U201. In case of difficulties with this restriction, just do what is needed in a computational subroutine. (If it is necessary to have a binary of a variable that exists only as a vector, it can be computed, packed, and written in U660.)

When a runtime offset, RR, is not zero and MODNUM( ) is not zero, meaning that data from a previous model run are needed, then data are carried forward in the MOS-2000 Internal Storage System. In the storing of data, as the file is read from a previous cycle to the current one, some data from an intermediate cycle may be saved that are not needed. This does not cause an error, and comes into play rarely, because RR = 0 except possibly for developing medium-range forecast equations, in producing data for special studies, or in developing predictands from observations over a period of time (e.g., summing rainfall amounts or averaging temperatures). When the input is vector data (MODNUM( ) = 0), RR is used only when DD in the variable ID = 0. This means that for a (vector) variable input from a previous run of U201 in which model data were used, it is assumed that the computations have already been made.

Although a vector format (non-gridpoint) file can have multiple call letters records (directories), U201 will not prepare a file with multiple call letters records. When KSKIP = 0, the call letters record will be written as the first record in the file; otherwise, the record will not be written, but in that case the existing call letters record will be checked for agreement. Disagreement will cause the program to stop.

The unit numbers for the random access files must be in the range 45 through 49 (although U201 will not write to a random access file); those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
44	400-499	Gridpoint data
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

COMMENTS

U201 was written to produce "predictors," and that terminology is used in many places. However, the word "variables" is more appropriate, since the use of the data produced is not restricted to predictors.

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing data will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a field is to be computed in OPTION and a field needed for the computation cannot be found, a diagnostic will be provided (possibly "****PREDICTOR NOT IDENTIFIED IN OPTN2") <u>and</u> in addition "****ALL VALUES IN FIELD...ARE MISSING IN PKMS99". While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. This probably means a set of dates should be supplied to U201 for which it is known there are good data. Then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed. Note that an error may produce missing data values, and may be counted for both.

On Day 1, GFETCH is entered directly at least once for every variable (except when a field can be reused, such as vorticity and a binary from vorticity) because it is not yet known when OPTION must be entered. ("Day 1 is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) <u>is not</u> counted as an error in PRED21 for Day 1. It <u>is</u> counted as an error in PRED22, because GFETCH should not be entered directly when OPTION is needed.

It is possible that no gridpoint or vector data on sequential files be used, and all input furnished be on "constant" random access files.

A number of computational routines are written and available for U201. Also, a few have been written primarily for illustrative purposes. For instance, L1D50CFFF linearizes variables with a set of thresholds.

L2D601000 and L2D600101 each linearize a pair of variables with sets of thresholds. These are provided to show how the thresholds can be provided and accessed, and for a template for other linearization routines. The routine COMBIN is provided to show how one might combine output from two models into one variable. This kind of combination is not recommended, but is possible in U201. If the combination is not a simple one, as it is in COMBIN, then a new variable name will have to be coined.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1.

It is unlikely that when no variable in a run has a particular model number, that the file containing that model will be needed; therefore, the file is closed. Closing the file keeps the data from being read when not needed. A highly unlikely exception is for a subroutine like COMBIN that would use data from a model that was not indicated by the model number in the variable ID. If this is the case, include a variable with the (otherwise omitted) model number to force the file to remain open. This variable can be a dummy (see OPTION subroutine) to keep it from being written.

It may be that a computational subroutine will need data at a date/time previous to that needed by any variable read in Record Type 13 (e.g., computation of mean value composed of previous values). In order to force saving of data for all date/times needed, include a variable with the first word = 799000000 and an appropriate RR. This will cause return from OPTION with IER = -2; a record with that ID will not be written. This will not be counted toward errors or missing data.

It is generally <u>not</u> necessary that each variable be present for Day 1 for that variable to be used on subsequent days (see the following section "Characteristics of the U201 Lookback Feature").

A count is kept of each time a missing value indicator is found when unpacking <u>grids</u> (not vector data). This is unexpected, and at the end of a run, the value is printed if non-zero.

A "missing" value of 9997 encountered in input <u>vector</u> data is set to PXMISS (see Record Type 3). This allows 9997 to remain intact (PXMISS = 9997), set to zero (PXMISS = 0), set to "normal" missing (PXMISS = 9999), or even some other value.

Although the primary purpose of U201 is to produce packed vector data for other programs, writing of such data is not done if KFILIO = 0 (see Record Type 8). That is, no output unit number and file name have been provided. This feature can be used for checkout runs.

The variables IFIND( ), ISTAV( ), and ITIME( ) that can be printed for each variable in table form under IP(17) control are set after processing Day 1 data and have the following meanings:

IFIND = 1	When the variable can be found directly from the input and does not have to be computed through OPTION.
= 0	Variable has been identified in OPTION and will be computed there.
= 2	Variable has been identified in OPTION but data returned are missing. Therefore, it is not yet known whether this variable will be input directly or computed through OPTION.

ISTAV = 1	When the data are vector format either on initial input or on return from OPTION.
= 0	Variable computed from gridded data.
= 2	Variable has not been found yet and it is still not known whether its input will be vector or gridded.

ITIME = 1	When the time offset RR is operative.
= 0	Otherwise. This may (also) occur when RR = 0.

Only a very limited sample of data has been available for checkout; as more data are available, changes to U201 may be necessary.

<u>CHARACTERISTICS OF THE U201 LOOKBACK FEATURE</u>

U201 attempts to save data from previous cycles (date/times previous to NDATE, the date/time being processed). Such data are denoted by RR > 0 in ID(3). Consider the following conditions:

o   The variable needed <u>is in</u> the ID list input by the user in Record Type 13.

  - All data needed are found for Day 1

    RR = 0   No data are saved from one NDATE to the next.

    RR > 0   Data for the variable with RR > 0 are saved for the number of hours indicated by RR.

  - Some data needed are not found for Day 1

    RR = 0   Not a problem.

    RR > 0   Usually not a problem.

o   The variable needed <u>is not in</u> the ID list input by the user, but is needed by a computational subroutine.

  - All data needed are found for Day 1

    RR ≥ 0   Shouldn't be a problem, <u>provided the subroutine looks for all variables it will need for subsequent cycles</u>. This may require the subroutine to ask for data not needed for the particular cycle being processed (e.g., max temperature is not calculated for each cycle).

- Some data needed are not found for Day 1

RR $\geq$ 0	No recovery; U201 has no way of knowing that a subroutine tried to access a variable and was unsuccessful. An example is 1000-850 mb thickness, and either the 850-mb or 1000-mb height was not available for the tau specified. The needed heights will not be saved <u>unless</u> the height itself is a variable in the ID list and has the same tau as the thickness.

While every effort has been made to make the lookback feature robust, and to not require the user to specify (possibly in a separate list) all the variables that might be needed in all subroutines, not all situations could be handled. It is believed that all will be well when all data needed for Day 1 are present. It should be rare when one would need to make a run that would require data that were not available for the first date/time.

SETTING UP THE DRIVER DRU201

The preparation of the driver for a particular U201 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with (i.e., interpolation done for). Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) unless, of course, the station directory is to be used as the station list. ND1 must also be GE NBLOCK (see below).
ND2 -	ND2*ND3 is the maximum size of the grid that can be dealt with. ND2 and ND3 are set separately to highlight the possible dimension of the grids. However, in the called routines, the size is only limited by the product, not each dimension individually.
ND3 -	See ND2.
ND4 -	The maximum number of variables (variables) for which interpolated values can be provided.

ND5 -	Dimension of IPACK( ), IWORK( ), and DATA( ). ND2X3 and ND5 should be set in the driver DRU201:
	
	ND2X3 = MAX(ND1,ND2*ND3,ND12)
	ND5=ND2x3

ND6 -	Maximum number of all sequential file input sources (e.g., models) that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc.
ND7 -	The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 -	The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.
ND9 -	The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records of the date/time of Day 1 on all input files.
ND10 -	The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.
ND11 -	The maximum number of grid combinations that can be dealt with. For instance, if the NGM and AVN data are being used, and the grids are different, then ND11 would be $\geq$ 2.
ND12 -	Maximum number of random access files (MOS-2000 External File System) that can be used. Limit is 6 because 6 unit numbers have been reserved for random access files.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.) Note that ND1 must be GE NBLOCK.

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND2*ND3). Every effort has been made so that the variables will not be overflowed if values too small are used; however, some danger may still exist.

The "startup" control file "U201.CN" is opened in DRU201, so that this aspect of U201 can be rather easily changed without recompiling or having separate version of the main (sub)routine U201. This will be different on the CRAY where unit/file assignments are made differently, and possibly when U201 is run in batch mode.

TYPES OF COMPUTATIONS IN U201

It is believed that most preparation of predictor and predictand datasets, as well as those for verification and other purposes, can be done by U201. However, the handling of input gridpoint and vector data are markedly different.

A. GRIDPOINT DATA

U201 will perform most of the necessary basic functions, as is indi-cated in the table below ordered by processing indicator. It is expected that only polar stereographic and Lambert archives will be used, and those are the only arrangements made. There is no hard coding for these map projections, except that routines pertaining to other projections are not provided.

```
PROCESSING INDICATORS

B - Binary (Value < threshold = 0; Value ≥ threshold = 1)
    1 = point binary
        BINARY
    5 = Grid binary
        GRIDB

DD - Model number
    1 = Trajectory
    6 = NGM
    7 = Eta

V - Vertical Processing (Same field (CCCFFF) and projection (ttt),
    different levels (UUUU and LLLL)
    VERTP
        0 = None
        1 = Difference (UUUU-LLLL)
        2 = Sum
        3 = Mean

T - Transformation
    TRANS
        1 = Square
        2 = Square root

RR, O, HH - Time Operation on Fields (1) and (2)
    TIMEP
       Field (1) = Run time offset (NDATE - RR), projection tau + HH
       Field (2) = Run time offset (NDATE - RR), projection tau
            1 = Mean
            2 = Difference (1)-(2)
            3 = Maximum
            4 = Minimum
       Field (1) = Run time offset (NDATE - RR), projection tau + HH
       Field (2) = Current run time, projection tau
            5 = Mean
            6 = Difference (2)-(1)
            7 = Maximum
            8 = Minimum

I - Interpolation
    1 = Biquadratic
        INTRPA
    2 = Bilinear
        INTRPB
    3 = Precipitation
        INTRP
```

S - Smoothing
- 1 = 5-point
  SMTH5
- 2 = 9-point
  SMTH9
- 3 = 25-point
  SMTH25

The processing is always done in the following order:

V,
RR, O, HH,
S,
I,
T, and
B,

except that grid binaries (B = 5) are computed prior to S.

When OPTION is entered, the data returned can be either gridpoint (in which case S, I, T, and B are operative) or vector. When vector, only T and B (point binaries) are operative.

B. <u>VECTOR DATA</u>

The capability for vector data input has the following four primary purposes:

(1) To allow the computation of variables (probably predictors) that are a combination of observations and gridpoint data (e.g., pseudo stratification).
(2) To allow the production of variables (probably predictands) that combine observations from more than one hour.
(3) To transform a variable of otherwise prepare a variable for use in regression.
(4) To cull from a large number of variables those needed for a specific regression run (e.g., predictands needed for a regression run from hourly data).

Partly because this input can be from a previous run of U201, the processing indicators in the IDs may be there because the computations have already been made, not because they need to be made as in the case of gridpoint data. For this reason, all processing of vector data must be done in subroutines through OPTION. However, when vector data are returned from OPTION, the T and B operators are still operative, except for CCC in the range 300-399 (combination of models), in which case it is assumed all the processing has been done. (This concept may need some tailoring as specific uses are identified. However, use of T will probably be rare, and other programs using the output of U201 should have even more binary capability than U201. Binary capability in U201 is limited because the heavy use of U201 will be in developing predictors, which have restricted use of binary definitions.)

VARIABLES HANDLED IN U201

All basic fields can be processed by TIMEP and VERTP (see above) and interpolated, plus the following variables indicated below can be computed and interpolated. It is noted that TIMEP and VERTP can call (through OPTN2) other routines for processing, and TIMEP can call VERTP. In this way, a field on which time processing is desired can ask VERTP to do a vertical process on fields that have to be computed by other routines. The user need not arrange for the basic fields to be present; U201 will arrange that.

CCCRRR	Routine	Write-up No.	Description
003010	MIXRAT	2.54	Mixing ratio on an isobaric surface
003011	MIXRAT	2.54	Mixing ratio on a constant height surface
003016	MIXRAT	2.54	Mixing ratio on a sigma surface
003030	SPECHUM	2.55	Specific humidity on an isobaric surface
003031	SPECHUM	2.55	Specific humidity on a constant height surface
003036	SPECHUM	2.55	Specific humidity on a sigma surface
003100	DEWPT	2.50	Dew point on an isobaric surface
003101	DEWPT	2.50	Dew point on a constant height surface
003106	DEWPT	2.50	Dew point on a sigma surface
003200	KINDEX	2.53	K Index
003265	NONCNVP	2.52	3-h non-convective precip amount
003270	NONCNVP	2.52	6-h non-convective precip amount
003280	NONCNVP	2.52	12-h non-convective precip amount
004002	GWIND	2.26	Grid-oriented geostrophic u-wind from heights
004012	GWIND	2.26	Earth-oriented geostrophic u-wind from heights
004102	GWIND	2.26	Grid-oriented geostrophic v-wind from heights
004112	GWIND	2.26	Earth-oriented geostrophic v-wind from heights
004212	GWIND	2.26	Geostrophic wind speed from heights
004210	WSPEED	2.33	Wind speed from u- and v- components
004011	EOWND	2.30	Earth-oriented u-wind from u and v compo-nents
004111	EOWND	2.30	Earth-oriented v-wind from u and v compo-nents
006010	VORTW	2.22	Relative vorticity from u and v components
006020	VORTH	2.20	Geostrophic relative vorticity from heights
006110	DIVW	2.24	Divergence from u and v components
007210	TTOTALS	2.51	Total totals index
010201	FORIER	4.11	Sin DOY (basic date)
010202	FORIER	4.11	Sin 2*DOY (basic date)
010203	FORIER	4.11	Cos DOY (basic date)
010204	FORIER	4.11	Cos 2*DOY (basic date)
010201	FORIER	4.11	Sin DOY (basic date plus projection)
010202	FORIER	4.11	Sin 2*DOY (basic date plus projection)
010203	FORIER	4.11	Cos DOY (basic date plus projection)
010204	FORIER	4.11	Cos 2*DOY (basic date plus projection)

WRITING NEW COMPUTATIONAL SUBROUTINES

New computational routines can be written as needed. The existing routines, such as VORTH, can be used as templates. Generally, the steps to be followed are:

1) Include the routine in the u201lib directory, and see that it is included in the makefile.

2) Enter the routine name with its calling sequence in the switching routine OPTION and probably in OPTN2 with the proper IF tests. Generally, this switching would be on IDPARS(1) and/or IDPARS(2).

3) Use the work arrays FD1( ), FD2( ), etc. as needed.

4) The new routine can call OPTN2 (or even OPTION) if computed variables are needed for which routines are already written. If this is done, the complete OPTN2 calling sequence must be used for the new routine, so that it can be passed to OPTN2.

5) If OPTN2 is called or OPTION recalled, care must be taken to not overlap the use of the arrays FD1( ), FD2( ), etc. That is, if VORTH were to be called by OPTN2, it would use arrays FD1( ), FD2( ), and FD3( ). If these arrays were used by the new routine, they would be wiped out when OPTN2 and VORTH were called. However, data are never carried from variable to variable in FD1( ), etc. That is, each time a new variable is dealt with in OPTION, each of the FDi( ) series is available for use. It is only in the computation of a particular variable that the work space must be managed by the user. Arrays NELEV( ), STALAT( ), and STALON( ) hold the station elevations, latitudes, and longitudes, respectively, in the order of the stations in CCALL(,) that can be used in any subroutine; they shouldn't, of course, be modified. Other "constant" data must be obtained from the MOS-2000 External Direct Access Storage System by calling subroutine CONST. Elevations can also be obtained through CONST, if desired.

(6) When a computational routine returns vector data, it must set ISTAV = 1; when the data returned are gridpoint, it must set ISTAV = 0.

(7) As a special case, when data for a particular hour cannot be available (e.g., max/min temperature at more than one hour each per day), the computational routine can set IER = -1 to indicate a record is not to be written for that variable for that date/time.

(8) OPTION will produce a diagnostic as appropriate when a called routine does not return IER = 0. The computational routine probably does not need an additional diagnostic. Also, the routine should not return IER = 47 (indicating missing data), but something else coordinated with the error table for MOS-2000. OPTION, PRED21, and PRED22 will then be able to handle the situation.

SPECIAL CONSIDERATIONS FOR LINEARIZATION ROUTINES

One of the powerful elements of MOS is that even though the basic statistical model may be linear, the predictors can be very non-linear. Such possibilities are provided by cumulative binaries, grid binaries, and the "T" in the ID. Non-linear combinations of basic or even of derived fields can be provided as predictors. One-dimensional (1-d) and 2-dimensional (2-d) linearization techniques have been used in the past, but are somewhat difficult to implement in either the developmental or implementation systems. Provision has been made for such linearizations to be performed within U201 subroutines.

Basically, the technique is to divide the predictor to be linearized into discrete categories by using thresholds and to provide a "linearized" value for each category. This might well be the probability of that category occurring. These values must have been determined from previous developmental work. For two variables, two sets of thresholds--one for each variables--are needed, and the result will be "boxes" within a 2-d diagram or table, and each box must have a "linearized" value. This sounds relatively simple, until one realizes that the sets of thresholds may well vary, not only with the variable(s), but with station or region, time of year (season), projection, and run (cycle) time. Not only is the development of such thresholds a chore, but how does one arrange to implement this technique?

The writeups for subroutines L1D and L2D provide information on the implementation within U201 of the linearization techniques, including the file naming convention and record structure for the thresholds. Thresholds are read by L1D1 or L2D1 on first entry to them as appropriate on Unit No. 97, which is closed upon exit and, therefore, can be used for all such files. The thresholds are stored in the MOS-2000 Internal Storage System and accessed by L1D1 and L2D1 when needed. These threshold files must be available or L1D1 and L2D1 will try to open and read them for each day in the sample; that is, if the thresholds are not available in the MOS-2000 Internal Storage System, they will be read from a file of the appropriate name. This means that for a new season or cycle, a new set is read and stored, so all thresholds are not necessarily read on Day 1 (note that "Day 1" used to represent the first "case," and, therefore, only the first cycle).

OPTION switches to L1D or L2D based on (some portion of) CCC (and possibly FFF). L1D and L2D are themselves switchers into which must be build the calls to, for example, L1D50CFFF and L2D601000, respectively. To insert a new linearization technique, the following must be done:

1) Include the routine in the u201lib directory, and see that it is included in the makefile. Note that for 1-d linearizations, the existing L1D50CFFF will handle most needs, and a new routine won't be necessary.

2) Enter the routine name with its calling sequence in the switching routine L1D or L2D with the proper IF tests.

3) Since non-specific linearization routines are called, which can call other computational routines through OPTN2, include in the call

sequence all variables that are in the call sequences to those routines.

4) Guidelines for managing the arrays FD1( ), FD2( ), etc., must be followed as for other routines as explained in the previous section.

<u>NONSYSTEM ROUTINES USED</u>

Use the load line in file 'home21.tdllib.dru201', dataset 'u201.com', along with the libraries 'home21.tdllib.u201lib' and 'home21.tdllib.moslib', all on blizzard.

<u>LANGUAGE</u>: FORTRAN77 with some HP extensions.

<u>LOCATION</u>: u201lib. The driver is in home/tdllib/dru201.

DIRCMP

COMPUTES GRID LOCATIONS FOR POINTS

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To compute the XI and YJ positions of stations in terms of the grid specified. Only north polar stereographic and Lambert grids are provided for. DIRCMP is used in U201, but may have other uses.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
    CALL DIRCMP(KFILDO,IP12,CCALL,NAME,STALAT,STALON,DIR,NGRIDC,ND1,NSTA,IER)
```

<u>KFILDO</u> - Unit number of output (print) file. Diagnostics, including error conditions are written here. (INPUT)

<u>IP12</u> - Indicates whether (> 0) or not (= 0) the list of stations and their I,J positions on the grid will be printed to the file whose unit number is IP12. (INPUT)

<u>CCALL</u>(K) - Call letters of NSTA stations (or locations) (K=1,NSTA). Each has 8 characters provided for, left justified. Used for printing when IP12 > 0. (CHARACTER*8) (INPUT)

<u>NAME</u>(K) - Names of stations corresponding to CCALL(K) (K=1,NSTA). Each has 20 characters provided for, left justified. Used for printing when IP12 > 0. (CHARACTER*20) (INPUT)

<u>STALAT</u>(K) - Latitudes of stations in degrees corresponding to CCALL(K) (K=1,NSTA). (INPUT)

<u>STALON</u>(K) - Longitudes of stations in degrees corresponding to CCALL(K) (K=1,NSTA). (INPUT)

<u>DIR</u>(K,J) - The XI (J=1) and YJ (J=2) positions on the grid for station K (K=1,NSTA), computed from the information in NGRIDC( ). (OUTPUT)

<u>NGRIDC</u>(J) - The grid characteristics used to compute the XI and YJ positions.
J = 1-- Map projection number, 3 = Lambert and 5 = polar stereographic. These numbers correspond to WMO/NMC usage.
J = 2-- Grid length in millimeters at latitude specified in NGRIDC(3).
J = 3-- Latitude in degrees * 10000 at which grid length is correct. In the case of the Lambert projection, this is also the latitude of tangency.
J = 4-- Longitude in degrees * 10000 at which grid length is correct.

J = 5-- Latitude in degrees * 10000 of lower left corner of grid.
J = 6-- Longitude in degrees * 10000 of lower left corner of grid.
Note that the units for the variables for J=2,6 are those associated with a grid in the TDLPACK format.
(INPUT)

ND1 - Maximum number of stations that can be dealt with. Dimension of several variables. (INPUT)

NSTA - Number of values in CCALL( ), etc. (INPUT)

IER - Status return.
 0 = Good return.
60 = The map projection number is not 3 or 5.
(OUTPUT)

EXAMPLE

```
      PARAMETER (ND1=60)
      CHARACTER*8 CCALL(ND1)
      CHARACTER*20 NAME(ND1)
      DIMENSION STALAT(ND1),STALON(ND1),DIR(ND1,2)
      DIMENSION NGRIDC(6,3)
      DATA KFILDO/6/
      ...
      NSTA=20
      N=1
      IP12=4
      ...

      CALL DIRCMP(KFILDO,IP12,CCALL,NAME,STALAT,STALON,DIR,NGRIDC(1,N),ND1,
     1            NSTA,IER)
```

The unit number for diagnostics is 6. The list of stations and their latitudes, longitudes, and grid positions are to be written to Unit No. 4; IP12 could have been = KFILDO, in which case the list would have been to the default output file. The call letters and names, used only for the listings, must be in CCALL( ) and NAME( ), respectively. The station latitudes and longitudes in degrees must be in STALAT( ) and STALON( ), respectively. NGRIDC(,N) must contain the characteristics of the grid for which the XI and YJ positions of the stations are to be calculated. Note in the explanation of NGRIDC( ) above the units to use in this INTEGER array. ND1 is the dimension of several variables, and must be carried because it is the first dimension of the doubly subscripted variable DIR(,). NSTA is the actual number of stations in CCALL( ), etc. IER will be zero on return unless there is a problem, the only one being if the map projection is not one that is accommodated.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. In addition, when IP12 > 0, the list of stations and their locations will be written to Unit No. IP12.

RESTRICTIONS:

None other than indicated above. The number of stations, NSTA, is of course limited to ND1.

NONSYSTEM ROUTINES USED:

LMLLIJ and PSLLIJ

LANGUAGE: With U201

LOCATION: MOS-2000 Library

GRCOMB

STORES GRID CHARACTERISTICS

Harry R. Glahn
April 1, 1995

PURPOSE: To determine for U201 whether or not the grid characteristics for the field just unpacked are the same as are already stored in NGRID(,), and if not, they are stored. Also, in this case, the XI and YJ positions of the stations corresponding to the new grid characteristics are calculated by calling DIRCMP.

RESTRICTIONS:

The number of grid combinations that can be stored is indicated by an input variable. If this number is about to be exceeded, a diagnostic is printed and an error condition returned. The arrays will not be overflowed.

NONSYSTEM ROUTINES USED:

DIRCMP

LANGUAGE: FORTRAN 77

LOCATION: With U201

GSITB

SMOOTHS, INTERPOLATES, TRANSFORMS, AND MAKES GRID AND POINT BINARIES

Harry R. Glahn
December 1, 1996

PURPOSE: To perform the following functions in order:
(1) grid binaries
(2) smooths field
(3) interpolates into grid to points
(4) transforms, and
(5) makes point binaries
all according to the MOS-2000 ID. This package is provided for use in such routines as COMBIN and L12D2.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL GSITB(KFILDO,ID,IDPARS,THRESH,DIR,SDATA,NSTA,ND1,DATA,ND5,
           IS2,ND7,FD1,ND2X3,IER)
```

KFILDO	- Unit number of output (print) file. (INPUT)
ID(J)	- The 4-word MOS-2000 ID of the variable being dealt with (J=1,4). (INPUT)
IDPARS(J)	- The MOS-2000 parsed ID (J=1,15). (INPUT)
THRESH	- The binary threshold associated with IDPARS( ). (INPUT)
DIR(K,J)	- The IX (J=1) and JY (J=2) positions on the gird for each station K (K=1,NSTA). (INPUT)
SDATA(K)	- Returned data (K=1,NSTA). (OUTPUT)
NSTA	- Number of stations being dealt with. (INPUT)
ND1	- Dimension of SDATA( ) and first Dimension of DIR(,). (INPUT)
DATA(J)	- Input gridpoint data (J=1,ND5). This is really an NX by NY grid, which is referenced by DIR(,). This is the way data are usually transferred from routine to routine within U201. (INPUT)
ND5	- Dimension of DATA( ). (INPUT)
IS2(J)	- TDLPACK Section 2 ID's (J=1,12). IS2(3) and IS2(4) define the NX by NY array size in DATA( ). (INPUT)
ND7	- Dimension of IS2( ). (INPUT)
FD1(J)	- Work array for use in smoothing (J=1,ND2X3). This can be any of the FDx( ) arrays in U201 that is not otherwise in use. (INTERNAL)

ND2X3 - Dimension of FD1( ). Must be ≥ NX*NY (if smoothing is to be done). (INPUT)

IER - Status return.
0 = Good return.
Any other value originates in a called subroutine. (OUTPUT)

EXAMPLE

```
    PARAMETER (ND1=600),
   1           ND5=5000,
   2           ND7=54,
   3           ND2X3=5000)
C
    DIMENSION ID(4),IDPARS(15)
    DIMENSION SDATA(ND1),DIR(ND1,2)
    DIMENSION DATA(ND5)
    DIMENSION IS2(ND7)
    DIMENSION FD1(ND2X3)
C
    DATA KFILDO/6/
    DATA ID/001000506,000000850,200000012,0130004120/
    DATA THRESH/1300/
    ...
    NSTA=450
    ...
    (Furnish gridpoint data in DATA( ), station locations in DIR( , ), field
    identification in IS2( ), and IDPARS( ) to conform to ID( ).)

    CALL GSITB(KFILDO,ID,IDPARS,THRESH,DIR,SDATA,NSTA,ND1,DATA,ND5,
               IS2,ND7,FD1,ND2X3,IER)
```

The default output file is Unit No. 6. The MOS-2000 ID specified is for the 12-h projection of 850-mb height for model No. 6 that is to be made into a 9-point smoothed [IDPARS(14) = 2] grid binary [IDPARS(3) = 5] with the threshold of 1300 meters (THRESH = 1300), bi-quadratically interpolated [IDPARS(13) = 1], and transformed by taking the square root [IDPARS(8) = 2]. (Note that both point binary and grid binary transformations cannot be made, because only one digit is supplied for the binary capability.) The gridpoint data are furnished in DATA( ), the array FD1( ) is used as a work array for smoothing, interpolation is done for 450 stations (points) according to the locations specified in DIR(,), the square root is taken of the values, and those values are returned in SDATA( ).

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than mentioned above when used within the framework of U201.

COMMENTS:

GSTIB is not used in PRED12 or PRED22 because, for efficiency, this block of code is entered in various places and, therefore, is not conducive to a single subroutine. (Each of the functions performed is itself a subroutine.)

Generally, all of the variables in the call are available in any routine that would need to use GSITB.

NONSYSTEM ROUTINES USED:

GRIDB, SMTH5, SMTH9, SMTH25, INTRP, INTRPA, INTRPB, TRANS, BINARY, TIMPR

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDSTR2

STORES ALL DATA FOR DAY ONE FOR U201

Harry R. Glahn
April 1, 1995

PURPOSE: To read and store in the internal MOS-2000 file structure all data for a sequence of dates. It is used by U201, but could have a use in other programs. When U201 is initiated, it is not known what variables will be needed in the computations and interpolation. It can be determined the date/times that span the data needed for the first day (actually, the first date/time). All data on all input files for those date/times are read and stored packed in the MOS-2000 internal file system. This system provides a certain amount of core storage, and when it is exhausted, the rest of the information is written to disk.

RESTRICTIONS:

None other than the general MOS-2000 guidelines, etc..

NONSYSTEM ROUTINES USED:

UNPACK, GSTORE, UPDAT, GRCOMB

LANGUAGE: FORTRAN 77

LOCATION: u201lib

COMPID

COMPUTES IDS FOR DATA IN MOS-2000 EXTERNAL FILE SYSTEM

Harry R. Glahn
December 1, 1996

PURPOSE: To provide the IDs for extracting constant data from MOS-2000 external random access files. COMPID is entered when CCC in ID(1) [IDPARS(1)] = 4xx. Since these data are expected to be used as predictors in U600 and the developed equations, the tau in ID(3) [IDPARS(12)] is added to NDATE to get the date for which date are required. Two sets of IDs are returned, the 2nd set being necessary only when interpolation is to be done (e.g., max temperatures provided only every 5th day). In addition, the day of the year (DOY) is returned as well as an "interpolation factor," the latter being the fraction of the way the DOY is from the 1st of the two values being used in interpolation.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL COMPID(KFILDO,ID,IDPARS,NDATE,KD,LD,MDOY,R,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

ID(J) - The predictor ID's (J=1,4). (INPUT)

IDPARS(J) - The MOS-2000 parsed ID (J=1,15):
1 - CCC (class of variable)
2 - FFF (subclass of variable)
3 - B (binary indicator)
4 - DD (data source, model number)
5 - V (vertical application)
6 - LLLL (lower level, 0 if only 1 level)
7 - UUUU (upper level)
8 - T (transformation)
9 - RR (run time offset in hours, always + and back in time)
10 - O (time application)
11 - HH (time period in hours)
12 - τττ (projection in hours)
13 - I (interpolation type)
14 - S (smoothing indicator)
15 - G (grid indicator)
(INPUT)

NDATE - The date/time for which the constant data are needed in YYYYMMDDHH format. (INPUT)

KD(J) - The 1st 4-word ID (J=1,4) for the constant. The calculation follows the format in the chapter "Variable Identification" (e.g., for daily values, the location in ID(2) corresponding to LLLL will be the DOY.) (OUTPUT)

LD(J)	-	The 2nd 4-word ID (J=1,4) for the constant. For constants that need not be interpolated, this return is not needed and will be the same as KD( ). (OUTPUT)
MDOY	-	The day of the year as calculated from NDATE. (OUTPUT)
R	-	The interpolation factor, being the fraction of the way the DOY is from the constant defined by KD( ) to the constant defined by LD( ). When interpolation is not needed, R = 0. (OUTPUT)
IER	-	Status return. 0 = Good return. 161 = Could not identify a CCCFFF for which to compute KD(2). 162 = Computational error. (OUTPUT)

EXAMPLE:

```
DIMENSION ID(4),KD(4),LD(4),IDPARS(15)
DATA KFILDO/6/
DATA ID/422006000,0,0,0/
DATA NDATE/1995012200/
...
[fill IDPARS( ) corresponding to ID( )]

CALL COMPID(KFILDO,ID,IDPARS,NDATE,KD,LD,MDOY,R,IER)
```

The default output file number is 6. ID(1) specifies calendar day max temperatures which are to be provided every 5th day on the constant file. The DOY is computed as 22 and returned in MDOY. KD( ) is returned as 422006000, 200000, 0, 0, and LD( ) is returned as 422006000, 250000, 0, 0. R is returned as .40, because MDOY is 40% of the way between DOY 20 and DOY 25, those days being indicated in KD(2) and LD(2), respectively. Had the projection tau not been zero in IDPARS(12), the DOY would have indicated that.

OUTPUT:

Diagnostic messages will be to Unit No. KFILDO.

RESTRICTIONS:

COMPID accommodates yearly, 2-season (April-September, etc.), 4-season (March-May, etc.), monthly, daily, and 5-day (values being at DOY numbers evenly divisible by 5) values. It is assumed the monthly values are to be interpolated to DOY from the midpoints of the months involved.

NONSYSTEM ROUTINES CALLED:

UPDAT, DOY

LANGUAGE: FORTRAN 77

LOCATION: moslib

PRED21

COMPUTES PREDICTORS FOR U201 FOR DAY 1

Harry R. Glahn
April 1, 1995

PURPOSE: To obtain for U201 all predictors for Day 1. All data available in the input file(s) that are needed for Day 1 have been read and stored in the internal MOS-2000 Storage System. The predictors have been numerically ordered so that access to the Storage System is minimized. For each predictor ID encountered, it is determined whether or not it is already held in PRED21 from the previous computation. If not, the Storage System is accessed. If the field is found, processing of the field continues. If not, OPTION is entered, which attempts to compute the predictor, at which point processing continues. This processing includes, as indicated by the elements of the ID, making a grid binary, smoothing, gridprinting for quality control (which must be used sparingly or the output file will become very large), interpolation, mathematical transformations, and making a point binary variable. The predictor is then packed and written to the output file. The ID's for the fields read from the input files and stored in the Storage System that are actually used are stored and marked so that for the following days only those fields need be considered.

RESTRICTIONS:

None other than the general MOS-2000 guidelines, etc.

NONSYSTEM ROUTINES USED:

GFETCH, OPTION, GRIDB, SMTH5, SMTH9, SMTH25, INTRP, INTRPA, INTRPB, TRANS, BINARY, PREDX1, PREDX2

LANGUAGE: FORTRAN 77

LOCATION: u201lib

PRED22

COMPUTES PREDICTORS FOR U201 FOR DAYS AFTER DAY 1

Harry R. Glahn
April 1, 1995

PURPOSE: To obtain for U201 all predictors for all days after the first. PRED22 reads the input file(s) and from the ID's stored by PRED21 determines whether or not a particular field is needed. If not, it is skipped. If it is needed, the computation(s) that can be done on it without entering OPTION are done. Also, if it is to be needed by OPTION, it is stored in the internal MOS-2000 Storage System. After all data that may be needed for the day being processed have been read, PRED22 passes through the predictors (again) to see whether more processing is necessary. From here on, the operation of PRED22 is much like that of PRED21. For each predictor ID encountered, it is determined whether or not the predictor has already been computed, If not, it is determined whether or not it is already held in PRED22 from the previous computation. If not, the Storage System is accessed. The predictors have been numerically ordered so that access to the Storage System is minimized. If the field is found, processing of the field continues. If not, OPTION is entered, which attempts to compute the predictor, at which point processing continues. (Note that all non-OPTION computations for the specific day being processed have already been done. The access to the Storage System outside OPTION is made only for lagged predictors which have been saved there.) This processing includes, as indicated by the elements of the ID, making a grid binary, smoothing, gridprinting for quality control (which must be used sparingly or the output file will become very large), interpolation, mathematical transformations, and making a point binary variable. The predictor is then packed and written to the output file.

RESTRICTIONS:

None other than the general MOS-2000 guidelines, etc.

NONSYSTEM ROUTINES USED:

GFETCH, OPTION, GRIDB, SMTH5, SMTH9, SMTH25, INTRP, INTRPA, INTRPB, TRANS, BINARY, PREDX1, PREDX2, GRCOMB, GSTORE, UNPACK, UNPKBG, UPDAT

LANGUAGE: FORTRAN 77

LOCATION: u201lib

OPTION

SWITCHES TO COMPUTATIONAL SUBROUTINES

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To switch to a computational subroutine when it is necessary to process other than a basic variable. The necessary elements of the variable ID in IDPARS( ) are checked to effect the switching. As new computational subroutines are developed, the pattern in OPTION can be repeated to deal with the new routines. Gridpoint calculations that do not have to be made through OPTION are grid binaries and smoothing of either basic or computed fields. Point computations that do not have to be made through OPTION include those based on the transformation variable, T, in IDPARS(8); those based on the point binary transformation variable, B, in IDPARS(3); and the basic interpolation based on the variable, I, in IDPARS(13) (i.e., the linear, quadratic, or special precip interpolations). In U201, OPTION is entered through PRED1 and PRED2.

<u>RESTRICTIONS</u>:

Any predictor that needs to be computed for U201 should be able to be computed through OPTION.

<u>COMMENTS</u>:

The data needed for the computation are accessed through the internal MOS-2000 storage system. Work arrays are available for provision to the computation routine. The map factor and sine of the latitude are provided in arrays with names FDMS( ) and SINLAT( ), respectively, and should be used for no other purpose. Most bookkeeping functions are performed for the user by other routines in U201. When writing a new computation routine, the designer should use the pattern established by a similar one (e.g., DIVW is much like VORTW).

Some computational routines may need data that themselves need to be computed. For instance, in the vertical averaging or differencing of divergence (the average or difference of divergence at two levels), indicated by the processing variable, V, in IDPARS(5), first the divergence must be calculated, then the average or difference performed. This is done by first entering VERTP through OPTION, then VERTP, upon finding that the divergence is not available in the internal MOS-2000 storage system, calls OPTN2 twice, once for each level, which in turn calls DIVW. When IDPARS(5) is not zero, then VERTP rather that DIVW is entered from OPTION.

However, VERTP is not entered from OPTION when IDPARS(10) NE 0. This means that when both VERTP and TIMEP need to be used, TIMEP is entered first, then through OPTN2, VERTP is entered and in tern through calling OPTN2 DIVW is entered. Note that this sequence uses OPTN2 twice; this is permissible because the FORTRAN 77 extensions allow recursive routines (a particular compile option may be needed for the CRAY). OPTN2 is almost a carbon copy of OPTION, except that it does not necessarily need all the capability of OPTION (does not contain all the calls to other routines).

In particular, it does not (should not) contain TIMEP because if TIMEP is needed it is entered immediately in OPTION.

As a special feature, whenever the 4-word ID cannot be found to switch to a computational subroutine, IER is set to -1 (as well as all data values returned set to 9999). This is used in PRED21 and PRED22 to indicate the data record is not to be written. This circumstance should be due to either an error in input, in which case a rerun would likely have to be made and there is no need to write the "missing" data, or the variable is a "dummy" with an appropriate RR to cause the lookback feature to save past data on Day 1, in which case the "missing" data should not be written.

NONSYSTEM ROUTINES USED:

VORTH, VORTW, DIVW, for example.

LANGUAGE: FORTRAN 77

LOCATION: With U201

OPTN2

SWITCHES SECONDARILY TO COMPUTATIONAL SUBROUTINES

Harry R. Glahn
April 1, 1995

PURPOSE: To switch to a computational subroutine when it is necessary to make the switch from other computational subroutines. The necessary elements of the variable ID in IDPARS( ) are checked to effect the switching. As new computational subroutines are developed, the pattern in OPTN2 can be repeated to deal with the new routines. In U201, OPTN2 is not entered directly through PRED1 or PRED2, but is entered from other computational routines.

RESTRICTIONS:

Any predictor that needs to be computed for U201 should be able to be computed through OPTION and OPTN2.

COMMENTS:

The data needed for the computation are accessed through the internal MOS-2000 storage system. Work arrays are available for provision to the computation routine. The map factor and sine of the latitude are provided in arrays with names FDMS( ) and SINLAT( ), respectively, and should be used for no other purpose. Most bookkeeping functions are performed for the user by other routines in U201. When writing a new computation routine, the designer should use the pattern established by a similar one (e.g., DIVW is much like VORTW).

Some computational routines may need data that themselves need to be computed. For instance, in the vertical averaging or differencing of divergence (the average or difference of divergence at two levels), indicated by the processing variable, V, in IDPARS(5), first the divergence must be calculated, then the average or difference performed. This is done by first entering VERTP through OPTION, then VERTP, upon finding that the divergence is not available in the internal MOS-2000 storage system, calls OPTN2 twice, once for each level, which in turn calls DIVW. When IDPARS(5) is not zero, then VERTP rather that DIVW is entered from OPTION.

However, VERTP is not entered from OPTION when IDPARS(10) NE 0. This means that when both VERTP and TIMEP need to be used, TIMEP is entered first, then through OPTN2, VERTP is entered and in tern through calling OPTN2 DIVW is entered. Note that this sequence uses OPTN2 twice; this is permissible because the FORTRAN 77 extensions allow recursive routines (a particular compile option may be needed for the CRAY). OPTN2 is almost a carbon copy of OPTION, except that it does not necessarily need all the capability of OPTION (does not contain all the calls to other routines). In particular, it does not (should not) contain TIMEP because if TIMEP is needed it is entered immediately in OPTION.

NONSYSTEM ROUTINES USED:

VORTH, VORTW, DIVW, for example.

LANGUAGE: FORTRAN 77

LOCATION: With U201

PREDX1

PREPARES TITLE AND GRIDPRINTS FIELD

Harry R. Glahn
April 1, 1995

PURPOSE: To prepare a title for a gridprint and to gridprint the field with title via PRTGR. PREDX1 is used by PRED1 and PRED2 in U201, but may have uses wherever grids are being dealt with in the MOS-2000 system.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL PREDX1(KFILDO,IDPARS,THRESH,SMULT,SADD,ORIGIN,CINT,PLAIN,UNITS,
            NDATE,KHR,DATA,ND5,IS2,ND7,IP14,ISTOP,IER)
```

KFILDO	- Unit number of output (print) file. (INPUT)
IDPARS(J)	- The MOS-2000 parsed ID (J=1,15) (INPUT)
THRESH	- The binary threshold associated with IDPARS( ). (INPUT)
SMULT	- Multiplicative scaling factor for displayed data values. Must be set to 1 if data are not to be modified by this variable (see SADD below). (INPUT)
SADD	- Additive scaling factor for displayed data values. Must be set to 0 if data are not to be modified by this variable. Scaled value = original value * SMULT + SADD rounded. (INPUT)
ORIGIN	- Value to be used as a starting point for contouring. This pertains to the original values, not as modified by SMULT above. Not used if CINT = 0. (INPUT)
CINT	- When CINT ≠ 0, it is the value to be used for the contour interval. This pertains to the original values, not as modified by SMULT above. When CINT = 0, data will be displayed without contouring. (INPUT)
PLAIN	- Plain language description of the predictors. (CHARACTER*32) (INPUT)
UNITS	- Units of the data that apply after multiplying by SMULT and adding SADD. (CHARACTER*12) (INPUT)
NDATE	- Current date/time in form YYYYMMDDHH. (INPUT)
KHR	- Hours to offset (go back in time) date/time when printing. This would normally be -IDPARS(9). (INPUT)
DATA(J)	- Input data array (J=1,ND5). This is actually a grid with dimensions defined in IS2( ) (see below). Data are stored in NMC convention. Starting point GRID(1,1) on the map is lower

left with data stored left to right (the IX direction), then the next row up (the JY direction), etc. (INPUT)

ND5	-	Dimension of DATA( ). (INPUT)
IS2(J)	-	MOS-2000 TDLPACK Section 2 ID's (J=1,ND7). IS2(3) and IS2(4) are, respectively, the NX and NY dimensions of the grid in DATA( ). (INPUT)
ND7	-	Dimension of IS2( ). (INPUT)
IP14	-	The unit number on which to write the gridprinted field. (INPUT)
ISTOP	-	Incremented by 1 if an error occurs. (INPUT-OUTPUT)
IER	-	Status return. 0 = Good return. Other values can be generated by subroutine PRTGR. (OUTPUT)

EXAMPLE:

```
    PARAMETER (ND5=2000)
    PARAMETER (ND7=54)
    DIMENSION DATA(ND5)
    DIMENSION IS2(ND7)
    DATA KFILDO/6/
    CINT=3.
    ORIGIN=1000.
    SMULT=1.
    SADD=0.
    IP14=2
    ...

    CALL PREDX1(KFILDO,IDPARS,THRESH,SMULT,SADD,ORIGIN,CINT,
   1            PLAIN,UNITS,NDATE,-IDPARS(9),DATA,ND5,IS2,ND7,IP14,ISTOP,IER)
```

The default output file number is 6. The gridprinted field is to be written to Unit No. 2. An array of IS2(3) X IS2(4) values will be contoured and written for display with a contour interval of 3., and an origin of 1000., after multiplying by SMULT and adding SADD. A title prepared from PLAIN, UNITS, NDATE, and IDPARS( ) will written at the bottom of each page of the map. The title of the gridprint is fashioned in the following way:

Characters	Title
1-3	Tau, taken from IDPARS(12)
4-7	'-HR '
8-39	Main part of the title, taken from PLAIN(1:32)
40	' '
41-52	Units, taken from UNITS(1:12). If this is a grid binary, the units will be replaced with the value in THRESH.

PLAIN( ) will have already been prepared in RDPRED in the following manner:

Characters	Title
1-32	As read from the predictor constant file, overwritten as necessary as indicated below.
26	'T' when the variable is a (point) transformation.
27-28	'GB' when the variable is a grid binary.
28	'B' when the variable is a point binary.
30-32	'S5 ', 'S9 ', or 'S25' when the variable includes smoothing.

However, the 'B' and 'T' will not be in the title because these "point" or "vector" transformations are made after the gridprint (that is, they don't apply to grids).

OUTPUT:

A printed matrix of the data or a printed contoured display of the data with title as provided by PRTGR will be written to Unit No. IP14. Diagnostic messages will be to Unit No. KFILDO.

RESTRICTIONS:

None other than the general MOS-2000 guidelines, etc.

NONSYSTEM ROUTINES CALLED:

PRTGR, UPDAT

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

PACKV

PACKS AND WRITES 1-DIMENSIONAL DATA IN TDLPACK FORMAT TO A SEQUENTIAL FILE

Harry R. Glahn
April 1, 1995

PURPOSE: To pack in TDLPACK format a vector of data with PACK1D and to write the data with WRITEP to a **sequential** file. This is used for interpolated and other data in U201, collated data in U660, and may have other uses. It prepares from input the TDLPACK identification in IS0( ), IS1( ), IS2( ), and IS4( ). It also writes to another file for viewing, when desired, the interpolated values by point (station) at the resolution packed.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL PACKV(KFILDO,KFILIO,ID,IDPARS,JP,ISCALD,ISCALE,IPLAIN,PLAIN,
   1           NDATE,NYR,NMO,NDA,NHR,CCALL,ISDATA,SDATA,ND1,
   2           NSTA,IPACK,ND5,MINPK,IS0,IS1,IS2,IS4,ND7,XMISSP,XMISSS,
   3           IPX,NWORDS,NTOTBY,NTOTRC,
   4           L3264B,L3264W,ISTOP,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

KFILIO - Unit number of output file for packed data. (INPUT)

ID(J) - The MOS-2000 predictor ID (J=1,4). (INPUT)

IDPARS(J) - The MOS-2000 parsed ID (J=1,15). (INPUT)

JP(J) - Indicates whether ($\geq$ 0) or not (= 0) the data for the variable will be output for viewing (J=1,3).
J=Not used.
J=2 Not used.
J=3 Listing of vector values to the resolution packed.
(INPUT)

ISCALD - Decimal scaling constant to use when packing the data. See below. (INPUT)

ISCALE - Binary scaling constant to use by PACK1D when packing the data. Packed value = original value $*10^{ISCALD}*2^{ISCALE}$ rounded to INTEGER. Normally ISCALE is zero. (INPUT)

IPLAIN(L,J) - 32 characters (L=1,L3264W) (J=1,4) of plain language description of predictor. Note that this requires 8 32-bit words to hold the description, but only four 64-bit words. Equivalenced to PLAIN (see below). (INPUT)

PLAIN - 32 characters of plain language description of variable. Equivalenced to IPLAIN. (CHARACTER*32) (INPUT)

NDATE	- The date/time for which predictors are being dealt with. NDATE is in the format YYYYMMDDHH. (INPUT)
NYR	- Year, 4 digits. This is YYYY in NDATE. (INPUT)
NMO	- Month, 2 digits. This is MM in NDATE. (INPUT)
NDA	- Day, 2 digits. This is DD in NDATE. (INPUT)
NHR	- Hour, 2 digits. This is HH in NDATE. (INPUT)
CCALL(K)	- 8-character call letters (K=1,NSTA). Used for printout only. (CHARACTER*8) (INPUT)
ISDATA(K)	- Work array (K=1,NSTA). (INTERNAL)
SDATA(K)	- Data to be packed and written (K=1,NSTA). (INPUT)
ND1	- Dimension of CCALL( ), ISDATA( ), and SDATA( ). (INPUT)
NSTA	The number of call letters in CCALL( ) and of data values in SDATA( ) (K=1,NSTA). (INPUT)
IPACK(J)	- Work array (J=1,ND5). (INTERNAL)
ND5	- Dimension of IPACK( ). (INPUT)
MINPK	- Minimum group size when packing the data. (INPUT)
IS0(J)	- MOS-2000 TDLPACK section 0 ID's (J=1,ND7) (INTERNAL)
IS1(J)	- MOS-2000 TDLPACK section 1 ID's (J=1,ND7) (INTERNAL)
IS2(J)	- MOS-2000 TDLPACK section 2 ID's (J=1,ND7) (INTERNAL)
IS4(J)	- MOS-2000 TDLPACK section 4 ID's (J=1,ND7) (INTERNAL)
ND7	- Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). (INPUT)
XMISSP	- Primary missing value. If not zero (for no missing value), this is normally 9999. (INPUT)
XMISSS	- Secondary missing value. If not zero (for no missing value), this is normally 9997. (INPUT)
IPX	- Indicates whether ($\geq$ 0) or not (= 0) data values will be written to Unit No. IPX. For writing to occur, JP(3) must also be $\geq$ 0, indicating that the values are to be written. By having this dual control, JP(3) can be different for each variable. On the other hand, all such print can be turned on or off by changing a single variable IPX which applies to the whole run (all variables). IPX is IP15 in U201 but may be different in other programs. (INPUT)

<u>NWORDS</u>	- The total number of words of packed data written.
<u>NTOTBY</u>	- The total number of bytes in the file. The bytes are counted and NTOTBY updated as the data are written. (INPUT-OUTPUT)
<u>NTOTRC</u>	- The total number of records in the file. The records are counted and NTOTRC updated as the data are written. (INPUT-OUTPUT)
<u>L3264B</u>	- Integer word length of machine being used, either 32 or 64. (INPUT)
<u>L3264W</u>	- Number of words in 64 bits on the machine being used, either 1 or 2. (INPUT)
<u>ISTOP</u>	- ISTOP is incremented by 1 for each error encountered. (INPUT-OUTPUT)
<u>IER</u>	- Status return. 0 = Good return. 16 = ND7 not large enough. Other values can be generated by called subroutines, UNPKBG, PACK1D, and WRITEP. (OUTPUT)

<u>EXAMPLE</u>

```
    PARAMETER (ND1=500)
    PARAMETER (ND5=ND1)
    PARAMETER (ND7=54)
    PARAMETER (L3264B=32)
    PARAMETER (L3264W=64/L3264B)
C
    CHARACTER*32 PLAIN
    CHARACTER*8 CCALL(ND1)
C
    DIMENSION ISDATA(ND1),SDATA(ND1)
    DIMENSION IPLAIN(L3264W,ND1)
    DIMENSION IPACK(ND5)
    DIMENSION IS0(ND7),IS1(ND7),IS2(ND7),IS4(ND7)
    DIMENSION ID(4),IDPARS(15),JP(3)
C
    EQUIVALENCE (PLAIN,IPLAIN)

    DATA KFILDO/6/
    DATA KFILIO/20/
    ...
    NSTA=250
    ISCALD=2
    ISCALE=0
    MINPK=14
    XMISSP=9999.
    XMISSS=0.
    IPX=15
    NTOTBY=0
    NTOTRC=0
C
    CALL PACKV(KFILDO,KFILIO,ID,IDPARS,JP,ISCALD,ISCALE,IPLAIN,PLAIN,
   1           1995010100,1995,01,01,00,CCALL,ISDATA,SDATA,ND1,
   2           NSTA,IPACK,ND5,MINPK,IS0,IS1,IS2,IS4,ND7,XMISSP,XMISSS,
   3           IPX,NWORDS,NTOTBY,NTOTRC,
   4           L3264B,L3264W,ISTOP,IER)
```

The default output unit number is 6. The unit number for the output file is 20, the recommended number for this file for U201. The NSTA values of data must be furnished in SDATA( ), and if the data are to be printed (for diagnostics and quality control) the call letters must be in the CHARACTER variable CCALL( ). The plain language that is to be packed is in the INTEGER variable IPLAIN(,), which is equivalenced to the CHARACTER*32 PLAIN variable in such a way that supports either a 32-bit or 64-bit word length machine. The data are for January 1, 1995, at 0000 UTC. The data are to be scaled *10^2 before rounding and packing. Primary missing values of 9999. may be present, but no secondary missing values are indicated. The number of records and bytes written will be counted in NTOTRC and NTOTBY, respectively; also, the number of words (for either a 32- or 64-bit machine) written will be returned. (Note that FORTRAN writes an extra 8 bytes per record, so if you check NTOTBY with the size of the file, expect to find this FORTRAN overhead present.) The data are to be written for viewing on Unit No. 15; JP(3) must ≠ 0 for writing to occur. Note that this output is <u>to the resolution packed</u>. Normally, the calling program would be dealing with many variables, so the dimension of ID( ) would be ID(4,ND1), etc.

<u>OUTPUT</u>:

Diagnostic messages will be written to Unit No. KFILDO. When indicated by IPX and JP(3), the data along with the corresponding call letters will be written to Unit No. IPX.

<u>RESTRICTIONS</u>:

None other than stated above.

<u>COMMENTS</u>:

Name changed from PREDX2.

<u>NONSYSTEM ROUTINES USED</u>:

UNPKBG, PACK1D, WRITEP

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

MAPLAT

COMPUTES SINE OF LATITUDE AND MAP FACTOR

Harry R. Glahn
April 1, 1995

PURPOSE: To compute sine of the latitude and map factor for a grid for polar stereographic and Lambert map projections. These fields are calculated in either subroutine LMMAPF or PSMAPF on initial entry and saved in FDSINS(,) and FDMS(,), respectively. They are not recalculated unless the map characteristics change. This is efficient under the assumption that most uses will be for one model/grid. The archive could change at some point; that would be OK. A diagnostic is printed for only the first 10 recomputations to alert the user that this has happened. Recomputation is not treated as an error.

RESTRICTIONS:

Only polar stereographic and Lambert map projections are supported.

NONSYSTEM ROUTINES USED:

LMMAPF, PSMAPF

LANGUAGE: FORTRAN 77

LOCATION: With U201

PSMAPF

COMPUTES SINE OF LATITUDE AND MAP FACTOR FOR POLAR STEREOGRAPHIC MAP

Harry R. Glahn
April 1, 1995

PURPOSE: To compute sine of the latitude and map factor for a grid for the polar stereographic map projection. PSMAPF is called by MAPLAT.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

PSLLIJ

LANGUAGE: FORTRAN 77

LOCATION: With U201

LMMAPF

COMPUTES SINE OF LATITUDE AND MAP FACTOR FOR LAMBERT MAP

Harry R. Glahn
April 1, 1995

PURPOSE: To compute sine of the latitude and map factor for a grid for the Lambert map projection. LMMAPF is called by MAPLAT.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

LMIJLL

LANGUAGE: FORTRAN 77

LOCATION: With U201

CONST

RETRIEVES DATA FROM MOS-2000 EXTERNAL RANDOM ACCESS FILE

Harry R. Glahn
December 1, 1996

PURPOSE: To retrieve data from a MOS-2000 external random access file. CONST is called from U201's OPTION and vector programs' OPTX

RESTRICTIONS:

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

COMMENTS:

The call sequence is appropriate for both U201 and vector oriented routines (e.g., U600). All the facilities of the MOS-2000 internal storage system and external random access file system are used.

On the first use of a particular random access file, the "directory" record is read with ID(1) = 400001000, ID(2) = 0, ID(3) = 0, and ID(4) = 0. A correspondence table between this directory and the stations for which data are needed is formed and stored in the MOS-2000 internal storage system with IDs the same as above, except ID(2) = KFILX. On subsequent entries when the same file is needed, this correspondence table is used.

The subroutine COMPID is used to compute the IDs needed to fetch the data. For instance, the ID in the call sequence will be generic in the sense that the date/time of the constant on the file is not specified. When constants are furnished for each day (or month or season) the full ID must be computed. The structure shown in Chapter 4, Variable Identification, is followed.

IP(16) can be used to print certain information with the /D option.

See routine OPTION or OPTX for appropriate call sequence. Variables have usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH, GSTORE, RDTDLM, COMPID, UNPACK

LANGUAGE: FORTRAN 77

LOCATION: moslib

VORTH

COMPUTES GEOSTROPHIC VORTICITY

Harry R. Glahn
April 1, 1995

PURPOSE: To compute geostrophic vorticity on a grid from a grid of heights at any level for either a north polar stereographic or Lambert map projection. VORTH is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The actual vorticity computation is done by VORTH1, which uses the usual 5-point stencil, except for the outside row and column on each side. For these rows (and columns), linear extrapolation is used from the adjacent 2 rows (and columns). Given the heights in meters, the vorticity is returned x10^5 sec^{-1}. The MOS-2000 ID (CCCFFF) processed is:

006 020 - Geostrophic vorticity.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

GFETCH, MAPLAT, VORTH1, (and optionally TIMPR, and PRTGR)

LANGUAGE: FORTRAN 77

LOCATION: With U201

VORTH1

COMPUTES GEOSTROPHIC VORTICITY FOR VORTH

Harry R. Glahn
April 1, 1995

PURPOSE: To compute geostrophic vorticity on a grid from a grid of heights. VORTH1 is called by VORTH, but could be used for other purposes. However, the sine of the latitude and the map factor have to be furnished to VORTH1. The computation is done with the usual 5-point stencil, except for the outside row and column on each side. For these rows (and columns), linear extrapolation is used from the adjacent 2 rows (and columns). Given the heights in meters, the vorticity is returned x10^5 sec^{-1}.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

EXTRAP

LANGUAGE: FORTRAN 77

LOCATION: With U201

VORTW

COMPUTES VORTICITY

Harry R. Glahn
April 1, 1995

PURPOSE: To compute vorticity on a grid from grids of u- and v-winds at any level for either a north polar stereographic or Lambert map projection. VORTW is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The actual vorticity computation is done by VORTW1, which uses the usual 2-gridlength finite differencing technique in each direction, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed vorticity is used from the adjacent 2 rows (and columns). Given the winds in m/sec, the vorticity is returned x10^5 sec^{-1}. The MOS-2000 ID (CCCFFF) processed is:

006 010 - Vorticity.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

GFETCH, MAPLAT, VORTW1, (and optionally TIMPR, and PRTGR)

LANGUAGE: FORTRAN 77

LOCATION: With U201

VORTW1

COMPUTES VORTICITY FOR VORTW

Harry R. Glahn
April 1, 1995

PURPOSE: To compute vorticity on a grid from grids of u- and v-winds. VORTW1 is called by VORTW, but could be used for other purposes. However, the sine of the latitude and the map factor have to be furnished to VORTW1. The computation is done with the usual 2-gridlength finite differencing technique in each direction, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed vorticity is used from the adjacent 2 rows (and columns). Given the winds in m/sec, the vorticity is returned x10^5 sec^{-1}.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

EXTRAP

LANGUAGE: FORTRAN 77

LOCATION: With U201

DIVW

COMPUTES DIVERGENCE

Harry R. Glahn
April 1, 1995

PURPOSE: To compute divergence on a grid from grids of u- and v-winds at any level for either a north polar stereographic or Lambert map projection. DIVW is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The actual divergence computation is done by DIVW1, which uses the usual 2-gridlength finite differencing technique in each direction, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed divergence is used from the adjacent 2 rows (and columns). Given the winds in m/sec, the divergence is returned x10^5 sec^{-1}. The MOS-2000 ID's (CCCFFF) processed are:

006 110 - Divergence (isobaric surface)
006 111 - Divergence (constant height surface)
006 116 - Divergence (constant sigma surface)

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

GFETCH, MAPLAT, DIVW1

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system.

DIVW1

COMPUTES DIVERGENCE FOR DIVW

Harry R. Glahn
April 1, 1995

PURPOSE: To compute divergence on a grid from grids of u- and v-winds. DIVW1 is called by DIVW, but could be used for other purposes. However, the sine of the latitude and the map factor have to be furnished to DIVW1. The computation is done with the usual 2-gridlength finite differencing technique in each direction, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed divergence is used from the adjacent 2 rows (and columns). Given the winds in m/sec, the divergence is returned x10^5 sec^{-1}.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

EXTRAP

LANGUAGE: FORTRAN 77

LOCATION: With U201

GWIND

COMPUTES GEOSTROPHIC WIND

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To compute geostrophic grid-oriented or earth-oriented u- or v-wind or geostrophic wind speed on a grid from a grid of heights at any level for either a north polar stereographic or Lambert map projection. GWIND is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The actual wind computation is done by GUWIND, GVWIND, or GSWIND, which use the usual 2-gridlength finite differencing technique, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed field is used from the adjacent 2 rows (and columns). Given the heights in meters, the winds will be in m/sec. The MOS-2000 ID's (CCCFFF) processed are:

004 002	- Grid-oriented geostrophic u-wind.
004 012	- Earth-oriented geostrophic u-wind.
004 102	- Grid-oriented geostrophic v-wind.
004 112	- Earth-oriented geostrophic v-wind.
004 212	- Geostrophic wind speed.

<u>RESTRICTIONS</u>:

The grid must be at least 4 x 4 in size.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH, MAPLAT, GUWIND, GVWIND, GSWIND, LMIJLL, PSIJLL, (and optionally TIMPR, and PRTGR)

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: With U201

GUWIND

COMPUTES GEOSTROPHIC U-WIND

Harry R. Glahn
April 1, 1995

PURPOSE: To compute geostrophic grid-oriented or earth-oriented u-wind on a grid from a grid of heights. GUWIND is called from GWIND, but could be used for other purposes. However, the sine of the latitude and the map factor have to be furnished to GUWIND. GUWIND uses the usual 2-gridlength finite differencing technique, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed field is used from the adjacent 2 rows (and columns). Given the heights in meters, the winds will be in m/sec. The MOS-2000 ID's (CCCFFF) processed are:

004 002	Grid-oriented geostrophic u-wind.
004 012	Earth-oriented geostrophic u-wind.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

EXTRAP, EOUWND

LANGUAGE: FORTRAN 77

LOCATION: With U201

GVWIND

COMPUTES GEOSTROPHIC V-WIND

Harry R. Glahn
April 1, 1995

PURPOSE: To compute geostrophic grid-oriented or earth-oriented v-wind on a grid from a grid of heights. GVWIND is called from GWIND, but could be used for other purposes. However, the sine of the latitude and the map factor have to be furnished to GVWIND. GVWIND uses the usual 2-gridlength finite differencing technique, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed field is used from the adjacent 2 rows (and columns). Given the heights in meters, the winds will be in m/sec. The MOS-2000 ID's (CCCFFF) processed are:

004 102 - Grid-oriented geostrophic v-wind.
004 112 - Earth-oriented geostrophic v-wind.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

EXTRAP, EOVWND

LANGUAGE: FORTRAN 77

LOCATION: With U201

GSWIND

COMPUTES GEOSTROPHIC WIND SPEED

Harry R. Glahn
April 1, 1995

PURPOSE: To compute geostrophic wind speed from a grid of heights. GSWIND is called from GWIND, but could be used for other purposes. However, the sine of the latitude and the map factor have to be furnished to GSWIND. GSWIND uses the usual 2-gridlength finite differencing technique, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed field is used from the adjacent 2 rows (and columns). Given the heights in meters, the wind will be in m/sec. The MOS-2000 ID's (CCCFFF) processed are:

004 212 - Geostrophic wind speed.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

EXTRAP

LANGUAGE: FORTRAN 77

LOCATION: With U201

EOWND

COMPUTES EARTH-ORIENTED WIND

Harry R. Glahn
April 1, 1995

PURPOSE: To compute earth-oriented u- or v-wind on a grid from grids of grid-oriented u-and v-winds at any level for a north polar stereographic Lambert, or mercator map projection. EOWND is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The actual wind computation is done by EOUWND or EOVWND. The output units will be the same as the input units with two exceptions, namely, for ID = 004061 or 004161, input winds on a surface of constant height in m/sec are converted to knots within the EOWND subroutine (see below). The MOS-2000 ID's (CCCFFF) processed are:

004 010	Earth-oriented u-wind, m/sec, isobaric surface
004 110	Earth-oriented v-wind, m/sec, isobaric surface
004 011	Earth-oriented u-wind, m/sec, constant height surface
004 111	Earth-oriented v-wind, m/sec, constant height surface
004 066	Earth-oriented u-wind, m/sec, constant sigma surface
004 166	Earth-oriented v-wind, m/sec, constant sigma surface
004 061	Earth-oriented u-wind, kts, constant height surface
004 161	Earth-oriented v-wind, kts, constant height surface.

NONSYSTEM ROUTINES USED:

GFETCH, EOUWND, EOVWND, PSLLIJ, LMLLIJ

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system

EOUWND

COMPUTES EARTH-ORIENTED U-WIND

Harry R. Glahn
April 1, 1995

PURPOSE: To compute earth-oriented u-wind on a grid from grids of grid-oriented u- and v-wind. EOUWND is called by EOWND, but could be used for other purposes. However, the grid coordinates of the north pole position have to be furnished to EOUWND. The output units will be the same as the input units.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: With U201

EOVWND

COMPUTES EARTH-ORIENTED V-WIND

Harry R. Glahn
April 1, 1995

PURPOSE: To compute earth-oriented v-wind on a grid from grids of grid-oriented u- and v-wind. EOVWND is called by EOWND, but could be used for other purposes. However, the grid coordinates of the north pole position have to be furnished to EOVWND. The output units will be the same as the input units.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: With U201

WSPEED

COMPUTES WIND SPEED

Harry R. Glahn
April 1, 1995

PURPOSE: To compute wind speed on a grid from grids of u- and v-winds at any level. Normally, the wind speeds are in units of m/sec, but these units are not essential for the execution of the algorithm found in the subroutine. WSPEED is an almost exclusively for U201 and similar routines (e.g., U203), and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The output units will be the same as the input units. The MOS-2000 ID's (CCCFFF) processed at the time of this write-up are:

004210 - Wind speed (m/sec) - constant pressure surface
004211 - Wind speed (m/sec) - constant height surface
004216 - Wind speed (m/sec) - constant sigma level

NONSYSTEM ROUTINES USED:

GFETCH (and optionally TIMPR)

LANGUAGE: FORTRAN 77 with extensions.

LOCATION: /home21/tdllib/u201lib (HP)
/nfsuser/g06/mos2k/u201lib (IBM-SP)

EXTRAP

EXTRAPOLATES A VARIABLE ON A GRID

Harry R. Glahn
April 1, 1995

PURPOSE: To linearly extrapolate to the outermost row and column (one on each side) from the two adjacent rows or columns. EXTRAP is called from a number of U201 routines.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: With U201

GRIDB

MAKES A GRID BINARY VARIABLE

Harry R. Glahn
May 1, 1994

<u>PURPOSE</u>: A threshold is used to make a binary variable. GRIDB operates on a 2-dimensioned array and is a counterpart to BINARY which operates on a 1-dimensioned array. The input variable IDPARS controls how the threshold is used, and sections of code can be added to prepare variables which are not strictly binary.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
CALL GRIDB(KFILDO,ID,IDPARS,THRESH,P,FD,NX,NY,IER)
```

<u>KFILDO</u> - Unit number of output (print) file. (INPUT)

<u>ID</u>(J) - 4-word MOS-2000 ID (J=1,4). Used only for diagnostic print. (INPUT)

<u>IDPARS</u> - A single digit controlling how the "binary" is produced.
5 = P(,) becomes 1 when the input value P(,) GE THRESH, and 0 otherwise.
Sections of code can be added to devise other binary-like variables corresponding to other values of IDPARS (e.g., P(,) remains unchanged unless it is GE THRESH, then it becomes 0). Values of IDPARS available for use are 6, 7, 8, and 9. (INPUT)

<u>THRESH</u> - The threshold value to use in making the binary variable. (INPUT)

<u>P</u>(IX,JY) - The input variable to transform to a binary (IX=1,NX; JY=1,NY). Normally, these are values at gridpoints. On exit, P( ) contains the transformed variable. (INPUT-OUTPUT)

<u>FD</u>(IX,JY) - A work array (IX=1,NX; JY=1,NY). For IDPARS = 5, this is not needed, but is provided in case of future need. (INTERNAL)

<u>NX,NY</u> - The number of gridpoints in the IX and JY directions, respectively. (INPUT)

<u>IER</u> - Status return.
0 = Good return.
64 = Threshold has the missing value = 9999.
65 = Value of IDPARS not used in this routine.
66 = Reserved.
67 = Reserved.
When IER NE 0, P( ) is set to 9999. (OUTPUT)

EXAMPLE

```
    DIMENSION ID(4),P(NX,NY),FD(NX,NY)
    DATA KFILDO/6/
    ...

    CALL GRIDB(KFILDO,ID,1,50.,P,FD,NX,NY,IER)
```

The unit number for diagnostics is 6. The MOS-2000 4-word ID is in ID( ). The variable P(IX,JY) is given the value 1 when the input value P(IX,JY) is GE 50 and given the value 0 otherwise. Note that THRESH = 50 is REAL, and that IDPARS = 1 is INTEGER. A return on IER = 0 means GRIDB operated as expected; any other value for IER indicates all values in P(,) = 9999 (IX=1,NX; JY=1,NY). FD(,) is a work array for possible use in GRIDB.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: With U201

VERTP

COMPUTES VERTICAL VARIABLES

Harry R. Glahn
April 1, 1995

PURPOSE: To compute a variable which is a function of another variable at two different levels. For instance, one might want to compute the thickness between 1000 and 500 mb. The routine is accessed from OPTION when the vertical processing indicator, V, in IDPARS(5), > 0 and the time processing indicator, T, in IDPARS(10), = 0. Because of this sequence, when both vertical and time processing is to be done, the vertical processing is done first by entering TIMEP first (i.e., TIMEP will arrange for the vertical processing before taking the time difference). VERTP can also be entered from OPTN2, which occurs via TIMEP. The two levels dealt with are LLLL in IDPARS(6) and UUUU in IDPARS(7). VERTP is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation.

RESTRICTIONS:

Any predictor, either basic or computed, should be able to be processed by VERTP. However, with only the information carried in the ID, the CCCFFF's of the variables used in the computations are the same, only the levels are different (e.g., thickness can be computed because it is a function of two heights or the difference of two vorticities can be computed).

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, OPTN2 (and optionally TIMPR)

LANGUAGE: FORTRAN 77

LOCATION: With U201

TIMEP

COMPUTES VARIABLES AS A FUNCTION OF TIME

Harry R. Glahn
April 1, 1995

PURPOSE: To compute a variable which is a function of another variable at two different times. For instance, one might want to compute the difference in the thickness between 1000 and 500 mb at projections 6 hours and 12 hours from the same model run. The routine is accessed from OPTION when the time processing indicator, T, in IDPARS(10), > 0. TIMEP is not included in OPTN2. TIMEP is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation of exactly how the variables RR, O, HH, and $\tau\tau\tau$ are used to do the computation.

RESTRICTIONS:

Any predictor, either basic or computed, should be able to be processed by TIMEP. However, with only the information carried in the ID, the CCCFFF's of the variables used in the computations are the same, only the times, and possibly the levels, are different (e.g., a change of thickness over two projection times can be computed because the change is a function of two thicknesses, which in turn are each the difference between two heights. Also, only one model can be involved, although it can be for different run times.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, OPTN2

LANGUAGE: FORTRAN 77

LOCATION: With U201

L1D

SWITCHES TO 1-DIMENSIONAL LINEARIZATION SUBROUTINES

Harry R. Glahn
June 1, 1995

PURPOSE: To switch to a specific 1-dimensional linearization subroutine. L1D is called from U201's OPTION when IDPARS(1) $\geq$ 500 and $\leq$ 599; that is, CCC values between 500 and 599 inclusive are reserved for 1-d linearization techniques.

COMMENTS:

For linearizing model variables, the convention has been adopted that the names of the routines called by L1D are of the form 5xcfff, where

```
    5 = the 1-d linearization key,
    x = the particular combination of season number, projection number,
        and cycle number,
    c = the 00C of the model variable to be linearized, and
  fff = the FFF of the model variable to be linearized.
```

Following this convention, the routine L1D50CFFF pertains to any model variable, and the linearization thresholds are provided according to the particular combination of season number, projection number, and cycle number defined in L1D50CFFF. This is possible (partly) because the ccc of any model variable is only 1 digit. Linearization of other types of variables will have to be dealt with in a more specific manner (e.g., an unused "key" other that 5 could be defined to apply to observations or 59cfff could pertain to a specific set of variables where the cfff is not necessarily defined in IDPARS(1) and IDPARS(2)).

L1D5xcfff calls L1D1, which in turn calls L1D2 and L1D3. L1D1, L1D2, and L1D3 are "generic" in that they should apply to any linearized variable (1-dimensional) and any variable to be linearized. It is only L1D5xcfff that is specific. This makes it possible for implementation of a new linearization by doing only the following:

- o Enter the subroutine name in the IF-THEN-ELSE calling loop with the appropriate switching test.

- o Use an existing L1D5xcfff (e.g., L1D50CFFF) as a template, and provide the information and arrange for the computations as indicated below.

- o Include in the plain language file 'MOS2000PRED' the plain language for the linearized predictor when necessary. This should not be necessary for linearizing the model variables, because entries should already be made.

- o Provide the threshold files.

Several threshold files will normally be required for each variable to linearize. On the HP, this should not be a problem, and providing and quality controlling ASCII files of the format required by the linearizing routines should be relatively easy. However, in operations on the CRAY, the number and naming of files may be a problem. This will be addressed later.

The switching done by L1D could, of course, be put in OPTION, rather than having a separate switching routine. It was thought best to not encumber OPTION with what might be a large number of such linearization routines.

Several routines L1D5xcfff may exist. Instructions are included here as to how to build another one. Generally they operate in the following manner:

- o One or more sets of thresholds and associated values (usually probabilities) are read from a file with the name 'L1Dqqqqqqddxxyyccbbbbtttt.CN', where qqqqqq = the specific CCCFFF of the linearized variable, dd = data source, xx = season number, yy = projection number, cc = cycle, bbbb = the bottom level, and tttt = the top level. The format of this file is given below. The values that L1D5xcfff must provide to its called subroutines, in addition to IDPARS( ), are defined as:

  - QQQQQQ = CCCFFF = IDPARS(1)*1000 + IDPARS(2), IDPARS( ) being in the calling sequence.

  - DD = the data source from IDPARS(4), IDPARS( ) being in the calling sequence.

  - XX = a number computed from the date. This is done each time the routine is entered so that when the date changes, XX can also change. While any two-digit number will work, the following reservations to be followed:

1-12	= monthly values (1 = January, etc.)
13-16	= 3-month seasons (13 = Spring--March-May, etc.)
17-18	= 6-month seasons (17 = Summer--April-September, etc.)
19	= Year round
20	= special "freezing" season

    Other values can be added as needed. Note that these are the same numbers used in defining "constant" data.

  - YY = a number computed from the projection. This is done each time the routine is entered so that when the projection of the variable changes, YY can also change. While any two-digit number will work, the following reservations are suggested:

    0 = projections of 0-5, 24-29, etc.
    1 = projections of 6-11, 30-35, etc.
    2 = projections of 12-17, 36-41, etc.
    3 = projections of 18-23, 42-47, etc.
    4 = all projections

- CC = the run (or cycle) time (in hours), which is the last two digits in the date/time NDATE.

It is suggested that for both XX and CC, the date/time used be that of NDATE <u>after</u> the run time offset RR has been applied. It is also suggested that the date/time does not depend on projection for purposes of accessing thresholds.

Each set of thresholds is read with a list of stations to which the thresholds apply. All sets are read and stored in arrays dimensioned for them in L1D5xcfff (actually read and stored in L1D1). A record is prepared and stored in the MOS-2000 Internal Storage System that gives for each station information sufficient for L1D1 to determine which set of thresholds is needed for that station. When stored, the record is tagged in such a way that it will be retained throughout the run. When L1D1 is entered, it calculates the identification of the station information record and attempts to retrieve it from the storage system. If the record cannot be found, then a file is opened with the appropriate name and the required set of thresholds read. In this way, thresholds are read as needed, and they don't have to be read again in the same run. This arrangement allows the thresholds to be identified with the appropriate stations without storing the thresholds multiple times and without excessive reading. The array L1DATA( ) has been reserved for the variable to hold the relationship between the station and the threshold sets. It is not needed outside L1D, and can be reused for other transformations. Note that the file is ASCII, which makes for easy editing. Also, the thresholds and probabilities are segmented in the way they would usually be developed, and that there can be different regions of stations for different DD, XX, YY, CC combinations. While the number of files may grow to a moderate number, the amount of data in each ASCII file is relatively small and efficiency should not be a problem, especially since only a portion of the files for each variable would usually have to be read in any one run.

o The NOCAT-1 thresholds (read into THRES( )) and their associated NOCAT "linearization values" (read into PROB( )) operate on the value of the variable VAL in subroutine L1D3 in the following manner:

```
IF      VAL ≤ THRES(1), THEN VAL = PROB(1)
ELSE IF VAL ≤ THRES(2), THEN VAL = PROB(2)
ELSE IF VAL ≤ THRES(3), THEN VAL = PROB(3)
...
ELSE IF VAL ≤ THRES(NOCAT-1), THEN VAL = PROB(NOCAT-1)
ELSE VAL = PROB(NOCAT)
```

It is believed that any 1-d linearization can follow this pattern. If not, then other options must be added to the 1-d linearization routines.

o Normally all information in IDPARS( ) pertains to the <u>variable to be linearized</u> except a portion of IDPARS(1) and possibly IDPARS(2), which (partially) define the <u>linearized variable</u>. However, the variable to be linearized is defined in L1D5xcfff, so that <u>any</u> variable can be associated with the particular CCCFFF. Note that the information in the plain language file 'MOS2000PRED' must match cccfff for plain language identification of the variable to be linearized.

- The variable to linearize is obtained by L1D1. It first looks for it in the MOS-2000 Storage System, as would other routines such as PRED1. If it can't find it, a call is made to OPTION. Note that this is a return to OPTION even though L1D (which calls L1D5xcfff, which calls L1D1) has been called from OPTION. This is permissible, since both HP and CRAY FORTRAN 77 extensions support reentry. OPTION then routes the request to the appropriate subroutine, which may call other routines such as OPTN2. In this way, any variable that is available (or can be made available) in original form can be linearized (see comments in OPTION writeup 2.12). It is for this capability that the call sequence is as extensive as it is.

- The format of the "threshold" file read in L1D1 is:

Record Type 1 - Format (2I3,6I5)

This record contains the same information as the file name, which is used to verify that this is the correct file. File names can be so easily changed, that an error could easily occur. This verification should eliminate an accident. Each value read pertains to the variable to linearize.

IDCCC	-	The CCC, compare to IDPARS(1).
IDFFF	-	The FFF, compare to IDPARS(2).
ID	-	The DD, compare to IDPARS(4).
IDX	-	The season number, compare to XX.
IDY	-	The projection number, compare to YY.
IDC	-	The cycle time, compare to CC.
IDB	-	The bottom level, compare to IDPARS(6).
IDT	-	The top level, compare to IDPARS(7).

Record Type 2 - Format (I4)

The single value read determines how many values to read in "threshold" records.

NOCAT - The number of threshold values to read is NOCAT-1; the number of associated values to read is NOCAT.

Record Type 3 - Format (20F8.0)

The thresholds are read into a temporary array so that the terminator will not overflow the array.

THRESH(J), J=1,NOCAT-1 - The thresholds that are to be used in the linearization as defined above. Note that while NOCAT values of PROB( ) are necessary, only NOCAT-1 values of the thresholds are necessary. THRESH(NOCAT) is set by subroutine L1D1 to 99999999. which should be larger than any value that will occur for the variable to linearize. (Evidently 99999999 is larger than the single precision variable can handle; if it is printed in L1D1, it comes out

100000000.00.) This value THRESH(NOCAT) is not actually used in the current algorithm in L1D3.

Record Type 4 - Format (20F8.0)

The probabilities associated with the thresholds read in Record Type 3 are read into a temporary array so that the terminator will not overflow the array.

PROB(J), J=1,NOCAT - The probabilities PROB(J) that are to be used with the thresholds THRESH(J) in the linearization as defined above.

Record Type 5 - Format (7(A8,1X))

The left justified call letters of the stations to which the thresholds read above apply.

CCALLD(J), J=1,NSTA - The call letters of the stations to which the thresholds apply. The list is read with subroutine RDC and must end with the terminator '99999999'. That is, the number of values, NSTA, need not be known beforehand. Blanks in the list are ignored. If a duplicate call letters station designation is found, the thresholds and probabilities for the list in which it was first encountered are used. Duplicate stations and stations in the threshold file that are not in the station list cause diagnostics. Printing of such diagnostics can usually be controlled by the calling program with "IP( )" values.

Record Types 3, 4, and 5 are repeated in sequence until the terminator 9999 is found for THRESH(1). This ends the use of this file, and the file is closed. (Actually, an end of file will also terminate the process without the terminator 9999.)

NONSYSTEM ROUTINES USED:

L1D50CFFF

LANGUAGE: FORTRAN 77

LOCATION: u201lib

L1D50CFFF

LINEARIZES A VARIABLE, 2 SEASONS, 4 PROJECTIONS, BY CYCLE

Harry R. Glahn
June 1, 1995

<u>PURPOSE</u>: To arrange for the 1-dimensional (1-d) linearization of variables. It is called from L1D when IDPARS(1) = 50c. This routine is specific in that (1) XX, the season number, is computed as 17 or 18 for summer and winter, respectively, (2) YY, the projection number, is computed as 0, 1, 2, or 3, as suggested in the writeup for L1D, (3) CC as the cycle time, and (4) the rest of the information in IDPARS( ) pertains to the variable to linearize (e.g., the smoothing variable in IDPARS(14)). That is, with IDPARS(1) replaced with IDPARS(1)-500, the variable defined in IDPARS( ) will be retrieved or computed in the same way before linearization as it would be in PRED1 or PRED2.

<u>RESTRICTIONS</u>:

The linearization process must by that in L1D3. Certain values are set by PARAMETER statements that control space allocation (dynamic allocation is not supported by the CRAY FORTRAN 77 compiler). They are:

NREG -	The maximum number of regions that can be accommodated by all calls to L1D50CFFF in a particular run.
NTHRES -	The maximum number of thresholds in any call to L1D50CFFF; this limits the value of NOCAT (see writeup L1D).
L1DCMB -	The maximum number of linearization threshold combinations that can be accommodated in all calls to L1D50CFFF. The combinations can vary with CCC and FFF of the variable to linearize, data source, season number, projection number, cycle time, and bottom and top levels (see writeup L1D).

<u>COMMENTS</u>:

See the L1D writeup for more details, which apply to all such linearization routines with this format. When L1D50CFFF is entered, NDATE is the date/time for which the data are needed. However, the thresholds are accessed according to what date/time was being processed <u>after</u> application of RR in IDPARS(9). Also, the date/time for purposes of accessing thresholds does not depend on the projection. This means that all thresholds will be keyed to the same date/time, no matter what the projection or IDPARS(9). This is in agreement with the way that such thresholds would normally be developed.

Another routine, say L1D51CFFF, can be used for another combination of season number, projection number, and cycle description.

NONSYSTEM ROUTINES USED:

L1D1, UPDAT.

LANGUAGE: FORTRAN 77

LOCATION: u201lib

L1D1

SUPPORTS 1-DIMENSIONAL LINEARIZATION

Harry R. Glahn
June 1, 1995

<u>PURPOSE</u>: To support specific 1-d linearization routines (e.g., L1D50CFFF) by reading transformation data (thresholds and probabilities); computing, when it doesn't already exist, the index record and storing it in the MOS-2000 Storage System; calling L12D2 to get the variable to linearize; and calling L1D3 to do the linearization. These three routines, L1D1, L12D2, and L1D3, should not have to be changed when adding a new variable to linearize, unless the linearization algorithm (not just the actual threshold and probability values) is changed. By defining the variable to linearize in the calling routine allows L1D1 to be generic to all 1-d linearizations. The transformation data are read from a file the first time they are needed and stored in dimensioned variables. There can be different transformation values for different stations, usually by region, and the pointer to the correct transformation data for each station is kept in a key record stored in the MOS-2000 Storage System.

<u>RESTRICTIONS</u>:

The linearization process must by that in L1D3.

<u>COMMENTS</u>:

See the L1D writeup for more details, which apply to all such 1-d linearization routines, especially for the format of the threshold file.

<u>NONSYSTEM ROUTINES USED</u>:

L12D2, L1D3, GFETCH, GSTORE, RDC.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: u201lib

L12D2

PROVIDES VARIABLE TO LINEARIZE FOR L1D1 AND L2D1

Harry R. Glahn
June 1, 1995

<u>PURPOSE</u>: To support L1D1 and L1D2 by retrieving a variable to linearize and performing other processing when called for. An attempt is made to find it in the MOS-2000 Storage System. If it is not there, OPTION is called, and all the features available to PRED1 and PRED2 from OPTION are available. Processing is limited to smoothing options and interpolation options, the same as are in PRED1 and PRED2. Other processing (point transformations and grid and point binary computations) would probably not make sense in this context, and are not provided. If they are needed, then can easily be added.

<u>RESTRICTIONS</u>:

None other that stated elsewhere.

<u>COMMENTS</u>:

See the L1D writeup for more details, which apply to all such 1-d linearization routines. This routine operates the same for L2D1 and for L1D1, except it is called twice from L2D1 and only once from L2D1.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH, OPTION, SMTH5, SMTH9, SMTH25, INTRP, INTRPA, INTRPB

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: With U201

L1D3

COMPUTES 1-DIMENSIONAL LINEARIZED VARIABLE

Harry R. Glahn
June 1, 1995

<u>PURPOSE</u>: To support L1D1 by computing the linearized variable from information supplied by L1D1. It operates in the manner described in the L1D writeup.

<u>RESTRICTIONS</u>:

None other than stated elsewhere.

<u>COMMENTS</u>:

See the L1D writeup for more details, which apply to all such 1-d linearization routines.

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: With U201

L1D506020

LINEARIZES 850-MB GEOSTROPHIC VORTICITY

Harry R. Glahn
June 1, 1995

PURPOSE: To arrange for the linearization of 850-mb geostrophic vorticity. It is called from L1D when IDPARS(1) = 506 and IDPARS(2) = 020. This routine is specific in that (1) the variable to linearize is 850-mb geostrophic vorticity, (2) XX, the season number, is computed as 5 or 6 for summer and winter, respectively, (3) YY, the projection number, is computed as 0, 1, 2, or 3, as suggested in the writeup for L1D, (4) CC is the cycle time, and (5) the rest of the information in IDPARS( ) pertains to the variable to linearize (e.g., the smoothing variable in IDPARS(14)). That is, with IDPARS(1) replaced with 006, and IDPARS(2) replaced with 020, the variable defined in IDPARS( ) will be retrieved or computed in the same way before linearization as it would be in PRED1 or PRED2.

RESTRICTIONS:

The linearization process must by that in L1D3. Certain values are set by PARAMETER statements that control space allocation (dynamic allocation is not supported by the CRAY FORTRAN 77 compiler). They are:

NREG -	The maximum number of regions that can be accommodated.
NTHRES -	The maximum number of thresholds; this limits the value of NOCAT (see writeup L1D).
L1COMB -	The maximum number of linearization threshold combinations that can be accommodated. The combinations can vary with data source, season number, projection number, and run number (see writeup L1D).

COMMENTS:

See the L1D writeup for more details, which apply to all such linearization routines with this format. When L1D506020 is entered, NDATE is the date/time for which the data are needed. However, the thresholds are accessed according to what date/time was being processed after application of RR in IDPARS(9). Also, the date/time for purposes of accessing thresholds does not depend on the projection. This means that all thresholds will be keyed to the same date/time, no matter what the projection or IDPARS(9). This is in agreement with the way that such thresholds would normally be developed.

Another routine, nearly identical to this one, would be needed to linearize, say, 1000-mb geostrophic vorticity. Although this L1D506020 routine could be generic to the level, it is not reasonable to make it so. Because the threshold file names do not contain the level, it is not possible to distinguish thresholds except by cccfff. This routine will not allow cccfff = 506020 to be used with any level except 1000. However, it is assumed other information such as smoothing and even model number could use the same thresholds. If this is not the case, then just define another cccfff.

NONSYSTEM ROUTINES USED:

L1D1, UPDAT.

LANGUAGE: FORTRAN 77

LOCATION: u201lib

L2D

SWITCHES TO 2-DIMENSIONAL LINEARIZATION SUBROUTINES

Harry R. Glahn
June 1, 1995

PURPOSE: To switch to a specific 2-dimensional linearization subroutine. L2D is called from U201's OPTION when IDPARS(1) $\geq$ 600 and $\leq$ 699; that is, CCC values between 600 and 699 inclusive are reserved for 2-d linearization techniques.

RESTRICTIONS:

The CCCFFF of the linearized variable must be in the array LDPRED( ) for switching to occur. The dimension of LDPRED( ) can be increased if necessary. The subroutine switched to must have the name 'L2Dcccfff' where cccfff pertains to the actual variable (e.g., 601000), and this name must be entered in the switching loop (see COMMENTS below).

COMMENTS:

L2Dcccfff calls L2D1, which in turn calls L12D2 and L2D3. L2D1, L12D2, and L2D3 are "generic" in that they should apply to any linearized variable (2-dimensional) and any variable to be linearized. It is only L2Dcccfff that is specific (the cccfff is the actual value for the linearized variable). This makes it possible for implementation of a new linearization by doing only the following:

- o Put the CCCFFF of the new linearized variable in LDPRED( ) IN L2D.
- o Enter the subroutine name in the IF-THEN-ELSE calling loop starting at statement 130 in L2D.
- o Use an existing L2Dcccfff (e.g., L2D601000 or L2D600101) as a template, and provide the information and arrange for the computations as indicated below.
- o Include in the plain language file 'MOS2000PRED' the plain language for the predictor.

This switching could, of course, be put in OPTION, rather than having a separate switching routine. It was thought best to not encumber OPTION with what might be a large number of such linearization routines.

Several routines L2Dcccfff may exist. Instructions are included here as to how to build another one. Generally they operate in the following manner:

- o One or more sets of thresholds and associated values (usually probabilities) are read from a file with the name 'L2Dqqqqqqddxxyycc.CN', where qqqqqq = the specific CCCFFF of the linearized variable, dd = data source, xx = season number, yy = projection number, and cc = cycle. The format of this file is given below. The values that L2Dcccfff must provide to its called subroutines are defined as:

- QQQQQQ = CCCFFF = IDPARS(1)*1000 + IDPARS(2), IDPARS( ) being in the calling sequence.

- DD = the data source from IDPARS(4), IDPARS( ) being in the calling sequence.

- XX = a number computed from the date. This is done each time the routine is entered so that when the date changes, XX can also change. While any two-digit number will work, the following reservations to be followed:

1-12	= monthly values (1 = January, etc.)
13-16	= 3-month seasons (13 = Spring--March-May, etc.)
17-18	= 6-month seasons (17 = Summer--April-September, etc.)
19	= Year round
20	= special "freezing" season

  Other values can be added as needed. Note that these are the same numbers used in defining "constant" data.

- YY = a number computed from the projection. This is done each time the routine is entered so that when the projection of the variable changes, YY can also change. While any two-digit number will work, the following reservations are suggested:

  0 = projections of 0-5, 24-29, etc.
  1 = projections of 6-11, 30-35, etc.
  2 = projections of 12-17, 36-41, etc.
  3 = projections of 18-23, 42-47, etc.
  4 = all projections

- CC = the run (or cycle) time (in hours), which is the last two digits in the date/time.

It is suggested that for both XX and CC, the date/time used be that of NDATE <u>after</u> the run time offset RR has been applied. It is also suggested that the date/time does not depend on projection for purposes of accessing thresholds.

Each set of thresholds is read with a list of stations to which the thresholds apply. All sets are read and stored in arrays dimensioned for them in L2Dcccfff (actually read and stored in L2D1). A record is prepared and stored in the MOS-2000 Internal Storage System that gives for each station information sufficient for L2D1 to determine which set of thresholds is needed for that station. When stored, the record is tagged in such a way that it will be retained throughout the run. When L2D1 is entered, it calculates the identification of the station information record and attempts to retrieve it from the storage system. If the record cannot be found, then a file is opened with the appropriate name and the required set of thresholds read. In this way, thresholds are read as needed, and they don't have to be read again in the same run. This arrangement allows the thresholds to be identified with the appropriate stations without storing the thresholds multiple times and without excessive reading. The array L1DATA( ) has been reserved for the variable to hold the relationship between the station and the threshold sets. It is not needed outside L1D, and can be reused for other transformations. Note that the file is ASCII, which makes for easy editing. Also, the thresholds and probabilities are segmented in the way they would usually be developed, and that there can be different regions of stations for different DD, XX, YY, CC combinations. While the number of files may grow to a moderate number, the amount of data in each ASCII file is relatively small and efficiency should not be a problem, especially since only a portion of the files for each variable would usually have to be read in any one run.

o The NOCAT1-1 thresholds (read into THRES1( )) for the first variable of the pair and the NOCAT2-1 thresholds (read into THRES2( )) and their associated NOCAT1*NOCAT2 "linearization values" (read into PROB( )) operate on the two values VAL1 and VAL2 of the variables in subroutine L2D3 to obtain the linearized value VALUE in the following manner:

```
IF      VAL1 ≤ THRES1(1), THEN INDEX1 = 1
ELSE IF VAL1 ≤ THRES1(2), THEN INDEX1 = 2
ELSE IF VAL1 ≤ THRES1(3), THEN INDEX1 = 3
...
ELSE IF VAL1 ≤ THRES1(NOCAT1-1), THEN INDEX1 = NOCAT1-1)
ELSE INDEX1 = NOCAT1

IF      VAL2 ≤ THRES2(1), THEN INDEX2 = 1
ELSE IF VAL2 ≤ THRES2(2), THEN INDEX2 = 2
ELSE IF VAL2 ≤ THRES2(3), THEN INDEX2 = 3
...
ELSE IF VAL2 ≤ THRES2(NOCAT2-1), THEN INDEX2 = NOCAT2-1)
ELSE INDEX2 = NOCAT2

VALUE = PROB(INDEX1,INDEX2)
```

It is believed that any 2-d linearization can follow this pattern. If not, then other options must be added to the 2-d linearization routines.

o Normally, some information in IDPARS( ) pertains to the variables to be linearized except IDPARS(1) and IDPARS(2), which (partially) define the linearized variable. However, the variables to be linearized are completely defined in L2Dcccfff, so that any variable can be associated with the particular CCCFFF. Note that the information in the plain language file 'MOS2000PRED' must match cccfff for plain language identification of the variable to occur in the output.

o The variables to linearize are obtained by L2D1. It first looks for them in the MOS-2000 Storage System, as would other routines such as PRED1. If it can't find them, a call (for each, as necessary) is made to OPTION. Note that this is a return to OPTION even though L2D (which calls L2Dcccfff, which calls L2D1) has been called from OPTION. This is permissible, since both HP and CRAY FORTRAN 77 extensions support reentry (actually, even recursion). OPTION then routes the request to the appropriate subroutine, which may call other routines such as OPTN2. In this way, any variable that is available (or can be made available) in original form can be one of the linearized pair (see comments in OPTION writeup 2.12). It is for this capability that the call sequence is as extensive as it is.

o The format of the "threshold" file read in L2D1 is:

Record Type 1 - Format (2I3,4I4)

This record contains the same information as the file name, which is used to verify that this is the correct file. File names can be so easily changed, that an error could easily occur. This verification should eliminate an accident. Each value read pertains to the variable to linearize.

IDCCC	-	The CCC, compare to IDPARS(1).
IDFFF	-	The FFF, compare to IDPARS(2).
ID	-	The DD, compare to IDPARS(4).
IDX	-	The season number, compare to XX.
IDY	-	The projection number, compare to YY.
IDC	-	The cycle time, compare to CC.

Record Type 2 - Format (2I4)

The two values read determine how many values to read in the following records.

NOCAT1 - The number of threshold values to read for the first variable is NOCAT1-1.

NOCAT2 - The number of threshold values to read for the second variable is NOCAT2-1.

Record Type 3 - Format (20F8.0)

The thresholds for the first variable are read into a temporary array so that the terminator will not overflow the array.

THRES1(J), J=1,NOCAT1-1 - The thresholds for the first variable that are to be used in the linearization as defined above. THRES1(NOCAT1) is set by subroutine L2D1 to 99999999. which should be larger than any value that will occur for the variable to linearize. (99999999 is larger than the single precision variable can handle on the HP; if it is printed in L2D1, it comes out 100000000.00.) This value THRES1(NOCAT1) is actually used in the current algorithm in L2D3; if a value of the variable to be linearized exceeds this value, a diagnostic is written, and IER set = 115. This will cause all values for this date/time to be set to missing. If this happens, something must be wrong, and the trouble should be diagnosed.

Record Type 4 - Format (20F8.0)

The thresholds for the second variable are read into a temporary array so that the terminator will not overflow the array.

THRES2(J), J=1,NOCAT2-1 - The thresholds for the second variable that are to be used in the linearization as defined above. THRES2(NOCAT2) is set by subroutine L2D1 to 99999999. which should be larger than any value that will occur for the variable to linearize. (99999999 is larger than the single precision variable can handle on the HP; if it is printed in L2D1, it comes out 100000000.00.) This value THRES2(NOCAT2) is actually used in the current algorithm in L2D3; if a value of the variable to be linearized exceeds this value, a diagnostic is written, and IER set = 116. This will cause all values for this date/time to be set to missing. If this happens, something must be wrong, and the trouble should be diagnosed.

Record Type 5 - Format (20F8.0)

The probabilities associated with the thresholds read in Record Types 3 and 4 are read into a temporary array so that the terminator will not overflow the array.

J=1,NOCAT1 sets of values are read as below:

PROB(J,L), L=1,NOCAT2 - The probabilities PROB(J,L) that are to be used with the thresholds THRES1(J) and THRES2(L) in the linearization as defined above.

Record Type 6 - Format (7(A8,1X))

The left justified call letters of the stations to which the thresholds read above apply.

CCALLD(J), J=1,NSTA - The call letters of the stations to which the thresholds apply. The list is read with subroutine RDC and must end with the terminator '99999999'. That is, the number of values, NSTA, need not be known beforehand. Blanks in the list are ignored. If a duplicate call letters station designation is found, the thresholds and probabilities for the list in which it was first encountered are used. Duplicate stations and stations in the threshold file that are not in the station list cause diagnostics. Printing of such diagnostics can usually be controlled by the calling program with "IP( )" values.

Record Types 3, 4, 5, and 6 are repeated in sequence until the terminator 9999 is found for THRES1(1). This ends the use of this file, and the file is closed. (Actually, an end of file will also terminate the process without the terminator 9999.)

<u>NONSYSTEM ROUTINES USED</u>:

L2Dcccfff.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: u201lib

L2D601000

LINEARIZES 850-MB AND 1000-MB HEIGHT

Harry R. Glahn
June 1, 1995

PURPOSE: To arrange for the linearization of 850-mb and 1000-mb height. It is called from L2D when IDPARS(1) = 601 and IDPARS(2) = 000. This routine is specific in that (1) the variables to linearize are 850-mb height (first variable) and 1000-mb height (second variable), (2) XX, the season number, is computed as 5 or 6 for summer and winter, respectively, (3) YY, the projection number, is computed as 0, 1, 2, or 3, as suggested in the writeup for L2D, (4) CC is the cycle time, and (5) the model number, run time offset, time application and time period, projection, interpolation type, smoothing, and grid indicator are the same as the information in IDPARS( ). That is, with IDPARS(1) replaced with 001, IDPARS(2) replaced with 000, and IDPARS(6) replaced with 850, the first variable defined in IDPARS( ) will be retrieved in the same way as it would be in PRED1 or PRED2. With IDPARS(1) replaced with 001, IDPARS(2) replaced with 000, and IDPARS(6) replaced with 1000, the second variable can also be obtained. (Since there are 15 values in IDPARS( ) for each of the first and second variables, the surest way to tell exactly how they are defined is to look at the source code.)

RESTRICTIONS:

The linearization process must by that in L2D3. Certain values are set by PARAMETER statements that control space allocation (dynamic allocation is not supported by the CRAY FORTRAN 77 compiler). They are:

NREG -	The maximum number of regions that can be accommodated.
NTHRS1 -	The maximum number of thresholds for the first variable; this limits the value of NOCAT1 (see writeup L2D).
NTHRS2 -	The maximum number of thresholds for the second variable; this limits the value of NOCAT2 (see writeup L2D).
L2COMB -	The maximum number of linearization threshold combinations that can be accommodated. The combinations can vary with data source, season number, projection number, and run number (see writeup L2D).

COMMENTS:

See the L2D writeup for more details, which apply to all such linearization routines with this format. When L2D601000 is entered, NDATE is the date/time for which the data are needed. However, the thresholds are accessed according to what date/time was being processed after application of RR in IDPARS(9). Also, the date/time for purposes of accessing thresholds does not depend on the projection. This means that all thresholds will be keyed to the same date/time, no matter what the projection or IDPARS(9). This is in agreement with the way that such thresholds would normally be developed.

<u>NONSYSTEM ROUTINES USED</u>:

L2D1, UPDAT.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: u201lib

L2D1

SUPPORTS 2-DIMENSIONAL LINEARIZATION

Harry R. Glahn
June 1, 1995

PURPOSE: To support specific 2-d linearization routines (e.g., L2D601000) by reading transformation data (thresholds and probabilities); computing, when it doesn't already exist, the index record and storing it in the MOS-2000 Storage System; calling L12D2 twice to get the variables to linearize; and calling L2D3 to do the linearization. These three routines, L2D1, L12D2, and L2D3, should not have to be changed when adding a new variable to linearize, unless the linearization algorithm (not just the actual threshold and probability values) is changed. By defining the variable to linearize in the calling routine allows L2D1 to be generic to all 2-d linearizations. The transformation data are read from a file the first time they are needed and stored in dimensioned variables. There can be different transformation values for different stations, usually by region, and the pointer to the correct transformation data for each station is kept in a key record stored in the MOS-2000 Storage System.

RESTRICTIONS:

The linearization process must by that in L2D3.

COMMENTS:

See the L2D writeup for more details, which apply to all such 2-d linearization routines, especially for the format of the threshold file.

NONSYSTEM ROUTINES USED:

L12D2, L2D3, GFETCH, GSTORE, RDC.

LANGUAGE: FORTRAN 77

LOCATION: u201lib

L2D3

COMPUTES 2-DIMENSIONAL LINEARIZED VARIABLE

Harry R. Glahn
June 1, 1995

PURPOSE: To support L2D1 by computing the linearized variable from information supplied by L2D1. It operates in the manner described in the L2D writeup.

RESTRICTIONS:

None other than stated elsewhere.

COMMENTS:

See the L2D writeup for more details, which apply to all such 2-d linearization routines.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: With U201

L2D600101

LINEARIZES 850-MB GEOSTROPHIC VORTICITY AND 1000-MB HEIGHT

Harry R. Glahn
June 1, 1995

PURPOSE: To arrange for the linearization of 850-mb geostrophic vorticity and 1000-mb height. It is called from L2D when IDPARS(1) = 600 and IDPARS(2) = 101. This routine is specific in that (1) the variables to linearize are 850-mb geostrophic vorticity (first variable) and 1000-mb height (second variable), (2) XX, the season number, is computed as 5 or 6 for summer and winter, respectively, (3) YY, the projection number, is computed as 0, 1, 2, or 3, as suggested in the writeup for L2D, (4) CC is the cycle time, and (5) the model number, run time offset, time application and time period, projection, interpolation type, smoothing, and grid indicator are the same as the information in IDPARS( ). That is, with IDPARS(1) replaced with 006, IDPARS(2) replaced with 020, and IDPARS(6) replaced with 850, the first variable defined in IDPARS( ) will be computed in the same way as it would be in PRED1 or PRED2. With IDPARS(1) replaced with 001, IDPARS(2) replaced with 000, and IDPARS(6) replaced with 1000, the second variable can also be obtained. (Since there are 15 values in IDPARS( ) for each of the first and second variables, the surest way to tell exactly how they are defined is to look at the source code.)

RESTRICTIONS:

The linearization process must by that in L1D3. Certain values are set by PARAMETER statements that control space allocation (dynamic allocation is not supported by the CRAY FORTRAN 77 compiler). They are:

NREG - The maximum number of regions that can be accommodated.

NTHRS1 - The maximum number of thresholds for the first variable; this limits the value of NOCAT1 (see writeup L2D).

NTHRS2 - The maximum number of thresholds for the second variable; this limits the value of NOCAT2 (see writeup L2D).

L2COMB - The maximum number of linearization threshold combinations that can be accommodated. The combinations can vary with data source, season number, projection number, and run number (see writeup L2D).

COMMENTS:

See the L2D writeup for more details, which apply to all such linearization routines with this format. When L2D600101 is entered, NDATE is the date/time for which the data are needed. However, the thresholds are accessed according to what date/time was being processed after application of RR in IDPARS(9). Also, the date/time for purposes of accessing thresholds does not depend on the projection. This means that all thresholds will be keyed to the same date/time, no matter what the projection or IDPARS(9). This is in agreement with the way that such thresholds would normally be developed.

NONSYSTEM ROUTINES USED:

L2D1, UPDAT.

LANGUAGE: FORTRAN 77

LOCATION: u201lib

COMBIN

COMBINES DATA FROM TWO MODELS INTO ONE PREDICTOR

Harry R. Glahn
December 1, 1996

PURPOSE: To compute the mean, difference, max, or min of values interpolated from two different models. COMBIN is called from OPTION when the CCC is in the range 300-399. The two models for which the average is taken are hardwired in COMBIN, because the MOS-2000 ID does not provide for defining two models.

RESTRICTIONS:

The input variables are identical except for the model from which they come. DD is not actually used; the model numbers are hardwired in COMBIN. This is, at present, a demonstration program, and both model numbers are the same. Note that if the computation involves going back in time for the model run (IDPARS(9) NE 0), then the model data for that previous time must be available to GFETCH. To make this happen, make sure both models involved have some other variable with RR of the same or greater value so that the data will have been saved for Day 1.

COMMENTS:

This routine is primarily for illustrative purposes. At present, the switch from OPTION to COMBIN is made when CCC is in the range 300-399. COMBIN could be made into a switcher so that other combinations could be made. Then COMBIN would perform like L1D and L2D, switching to a unique routine that would arrange to retrieve and compute grids from OPTION in the normal way, and then to perform the desired combination.

NONSYSTEM ROUTINES USED:

TIMPR, GFETCH, OPTION, GSITB

LANGUAGE: FORTRAN 77

LOCATION: u201lib

DEWPT

COMPUTES DEW POINT

Christopher A. Fiebrich
Wei Yan
December 1, 1998

PURPOSE: To compute dew point on a grid from grids of (1) temperature, pressure, and relative humidity, or (2) temperature, pressure, and specific humidity at any level. If either of the above sets of meteorological variables is available, then the dew point temperature can be computed. The grid containing the pressure values is internally created by using the TDL ID if the dew point is to be calculated on an isobaric surface. DEWPT is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The dew point is returned in degrees Kelvin. The MOS-2000 ID's (CCCFFF) processed are:

003 100	=	Dew Point on an isobaric surface;
003 101	=	Dew Point on a surface of constant height;
003 106	=	Dew Point on a sigma surface.

RESTRICTIONS:

One of the above sets of meteorological data must be available. The temperature data must be in degrees Kelvin, and the relative humidity data must be in percent.

For computations on a constant height or sigma surface, the subroutine expects pressure data from the NGM to be in hecto-Pascals. All other model pressure data are expected to be in Pascals, and are converted to hecto-Pascals within the DEWPT, MIXRAT, or SPECHUM subroutines. These latter two subroutines are used to obtain the appropriate moisture variables in order to compute the dew point.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, MIXRAT, SPECHUM

LANGUAGE: FORTRAN 77 with HP extensions.

LOCATION: With U201 in the u201lib file system.

TTOTALS

COMPUTES TOTAL TOTALS INDEX

Benjamin Sfanos
December 1, 1998

PURPOSE: To compute the Total Totals Index. This is defined as the temperature at 850 mb plus the dew point at 850 mb minus twice the value of the temperature at 500 mb. TTOTALS is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The Total Totals index is returned in units of degrees Kelvin. The MOS-2000 ID (CCCFFF) processed is:

007 210 = Total Totals Index.

RESTRICTIONS:

To compute the total totals index, the temperature at 850 and 500 mb as well as the dew point at 850 mb must be available. Generally, the direct model output does not contain the dew point; consequently, TTOTALS must obtain the dew point by calling subroutine DEWPT. TTOTALS also assumes that the temperature and dew point variables are in degrees Kelvin.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, DEWPT

LANGUAGE: FORTRAN 77 with HP extensions.

LOCATION: File ttotals.f in the U201 library /home21/tdllib/u201lib on the HPs and /jdsk40/we/mos2000/u201lib on the CRAY.

NONCNVP

COMPUTES NON-CONVECTIVE PRECIPITATION AMOUNT

Benjamin Sfanos
December 1, 1998

<u>PURPOSE</u>: To compute the non-convective precipitation amount. This is defined as the difference between the model forecast of the total precipitation amount and the model forecast of the convective precipitation amount. In the runs of the Eta, Aviation, and Medium-Range Forecast Models, the total forecast precipitation amount is partitioned into convective and non-convective components. In the model archives, only the total precipitation amount and the convective amount are saved. This subroutine allows the calculation of the non-convective amount. Since the model precipitation may be valid for 3-, 6-, or 12-h accumulation periods, this subroutine must be able to do this computation for any of the three accumulation periods. NONCNVP is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The non-convective precipitation amount is returned in the units of precipitation amount, generally millimeters (mm). The MOS-2000 ID's (CCCFFF) processed are:

003 265	=	Non-convective precipitation amount (mm), 3-h period.
003 270	=	Non-convective precipitation amount (mm), 6-h period.
003 280	=	Non-convective precipitation amount (mm), 12-h period.

<u>RESTRICTIONS</u>:

To compute the non-convective precipitation amount, both the total precipitation amount and the convective precipitation amount must be available for the requested period.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH

<u>LANGUAGE</u>: FORTRAN 77 with HP extensions.

<u>LOCATION</u>: File noncnvp.f in the U201 library /home21/tdllib/u201lib on the HPs and /jdsk40/we/mos2000/u201lib on the CRAY.

KINDEX

COMPUTES THE K-INDEX

Kathryn K. Hughes
December 1, 1998

PURPOSE: To compute George's K stability index. Based on the formula given below, the K index is computed on a grid from grids of temperature and dew point. Because the basic model variables generally do not include dew point, the dew point is first computed from grids of pressure, temperature, and relative humidity. The grid containing the pressure values is internally created by using the TDL ID. KINDEX is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The K-INDEX is returned in degrees C. The MOS-2000 ID (CCCFFF) processed is:

007 200 = K-INDEX.

RESTRICTIONS:

Grids of temperature and dew point at the appropriate isobaric levels must be available. As noted above, the dew point generally must be first computed from pressure, temperature, and relative humidity variables. The subroutine expects pressure data from the NGM to be in hectopascals (millibars). All other model pressure data are expected to be in Pascals, and are converted to hectopascals within the DEWPT subroutines. Temperatures and dew points are assumed to be in degrees Kelvin.

NOTE TO USER:

K-INDEX = (850T - 500T) + 850Td - (700T - 700Td)

The K-INDEX is useful for predicting non-severe warm season convective activity. As the value of the K-INDEX increases, the probability of thunderstorm development also increases.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, DEWPT

LANGUAGE: FORTRAN 77 with HP extensions.

LOCATION: With U201 in the u201lib file system.

MIXRAT

COMPUTES MIXING RATIO

Christopher A. Fiebrich
Wei Yan
December 1, 1998

PURPOSE: To compute the mixing ratio from grids of temperature, pressure, and relative humidity at any level. The grid containing the pressure values is internally created by using the TDL ID if the mixing ratio is to be calculated on an isobaric surface. MIXRAT is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The mixing ratio is returned with units kg/kg. The MOS-2000 ID's (CCCFFF) processed are:

003 010 = Mixing ratio on an isobaric surface.
003 011 = Mixing ratio on a surface of constant height.
003 016 = Mixing ratio on a sigma surface.

RESTRICTIONS:

Temperature, relative humidity, and pressure data must be available. The temperature data must be in degrees Kelvin, and the relative humidity data must be in percent. For computations on a constant height or sigma surface, the subroutine expects pressure data from the NGM to be in hectopascals (millibars). All other model pressure data are expected to be in Pascals, and are converted to hectopascals within the MIXRAT subroutine.

The algorithm used to compute the mixing ratio is known as the Teten-Stackpole approximation and is described more completely within the internal documentation of MIXRAT.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77 with HP extensions.

LOCATION: With U201 in the u201lib file system.

SPECHUM

COMPUTES SPECIFIC HUMIDITY

Christopher A. Fiebrich
Wei Yan
December 1, 1998

PURPOSE: To compute the specific humidity from grids of temperature, pressure, and relative humidity at any level. The grid containing the pressure values is internally created by using the TDL ID if the specific humidity is to be calculated on an isobaric surface. SPECHUM is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The specific humidity is returned with units kg/kg. The MOS-2000 ID's (CCCFFF) processed are:

003 030 = Specific humidity on an isobaric surface.
003 031 = Specific humidity on a surface of constant height.
003 036 = Specific humidity on a sigma surface.

RESTRICTIONS:

Temperature, relative humidity, and pressure data must be available. The temperature data must be in degrees Kelvin, and the relative humidity data must be in percent. For computations on a constant height or sigma surface, the subroutine expects pressure data from the NGM to be in hectopascals (millibars). All other model pressure data are expected to be in Pascals, and are converted to hectopascals within the SPECHUM subroutine.

The algorithm used to compute the specific humidity is known as the Tetn-Stackpole approximation and is described more completely within the internal documentation of SPECHUM.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77 with HP extensions.

LOCATION: File spechum.f in the U201 library /home21/tdllib/u201lib on the HPs and /jdsk40/we/mos2000/u201lib on the CRAY.

U202

COMPUTES AND COPIES GRIDPOINT AND VECTOR DATA SETS

Harry R. Glahn
December 1, 2000

PURPOSE: U202 is derived from U201, and its operation and control parameters are almost identical. The purpose of U201 is to compute variables using either grids, vectors (station data), or both; to interpolate to a common set of station locations in the case of gridpoint calculations; and to write the vector data to a TDLPACK vector dataset. Much computational capability is available through a call to subroutine OPTION, governed by the variable ID, which acts as a switcher to computational subroutines. Input can include TDLPACK gridpoint data from one or more files, TDLPACK vector data from one or more files, and "constant" data in the MOS-2000 External Random Access System.

U202 differs only in that gridpoint data or computations done on gridpoint data are not interpolated to station locations, but are written as TDLPACK gridpoint data, the grid characteristics being whatever the input was or whatever was used in the computations. Therefore, U202 can be used for copying a subset of TDLPACK gridpoint data to another data set. In a similar manner, U202 will copy TDLPACK vector data to another data set, where the station list can be culled down from the input list. Even though U202 is mainly for copying data, all the computational capabilities of U201 are retained, except, of course, the interpolation to station locations and the in-line processing of such data (e.g., making binaries). Any computation must be done in subroutines.

The primary difference in input from U201 to U202 is that instead of one "interpolated" output, there is a "vector" and a "gridpoint" output. The vector output will have a directory record consisting of the stations culled from the input list(s). The outputs will be readable by any of the MOS-2000 programs, including U201 or U202 itself.

A full description of U202 is not included here because it would be essentially a copy of U201. Rather, the differences are described.

Record Type 3 - Format (7(I10/),F10.0) Control Parameters

KSKIP - KSKIP pertains to both output files

KWRITE - Pertains only to the vector output.

Record Type 8a - Format (I3,4XA60) Vector Output File

This record (plus the terminator record) identifies the output file. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

KFILIO - Unit number for the output file.

OUTNAM - Name of file where this output is to reside. (CHARACTER*60)

Record Type 8b - Format (I3,4XA60) Gridpoint Output File

This record (plus the terminator record) identifies the output file. Records are read until the terminator KFILGO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

KFILGO - Unit number for the output file.

OUTGRD - Name of file where this output is to reside. (CHARACTER*60)

DATA OUTPUT

There are three forms of data output, besides the diagnostics and other information on Units KFILDO and IP( ).

A. ASCII FOR VIEWING OR PRINTING

Gridprinted maps of each variable for which JP(1,) ≠ 0 without contours and/or for which JP(2,) ≠ 0 with contours are written to Units IP(13) and IP(14), respectively, when those units numbers ≠ 0. Vector values are written for each variable for which JP(3,) ≠ 0 to Unit IP(15) when IP(15) ≠ 0. This makes for easy control; all such print can be eliminated by changing the IP( ) values, and there is still control on each variable J through JP(,J). This is identical to U201.

B. SEQUENTIAL VECTOR MOS-2000 FORMAT

All vector data from the vector inputs or computed in computational routines for station locations are packed in the MOS-2000 TDLPACK format. They are packed with subroutine PACK1D and its associated subroutines by calling PACKV. All data are written to the dataset whose name has been provided to NAMOUT and unit number to KFILIO from the control file 'U202.CN', Record Type 8a, unless KFILIO = 0, in which case data are not output. The first record will consist of the "station directory" of call letters, 8 characters written in binary (in subroutine SKIPWR) so that they can be read into and manipulated in an INTEGER array. When data are not available to read and copy, a record is not written.

C. SEQUENTIAL GRIDPOINT MOS-2000 FORMAT

All gridpoint data from the gridpoint inputs or computed in computational routines on a grid are packed in the MOS-2000 TDLPACK format. They are packed with subroutine PACK2D and its associated subroutines by calling PACKG. All data are written to the dataset whose name has been provided to OUTGRD and unit number to KFILGO from the control file 'U202.CN', Record Type 8b, unless KFILGO = 0, in which case data are not output. When data are not available to read and copy, a record is not written.

RESTRICTIONS:

There are places in the U201 writeup where the reference is obviously only to vector data; just ignore these comments with reference to gridpoint data.

Most of the same routines are used in U202 as in U201. One of these checks the elements in the ID for consistency. Since interpolation is not done, the interpolation parameter is meaningless; however, the program may expect a "legitimate" value--that is a value that would be used in U201. If U202 complains, just comply. Remember, that NO computation is done on vector data (e.g., making binaries), unless it is within computational subroutines.

EXAMPLE CONTROL FILE: 'U202.CN'

An example exists as file 'U202.CN' in directory 'home21/tdllib/dru202' on the blizzard. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

NONSYSTEM ROUTINES USED

Use the load line in file 'home21.tdllib.dru202', dataset 'u202.com', along with the libraries 'home21.tdllib.u202lib', 'home21.tdllib.u201lib' and 'home21.tdllib.moslib', all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: u202lib. The driver is in home21/tdllib/dru202.

U203

PREPARES TLDPACK DATA FOR GEMPAK

Harry R. Glahn
January 1, 2001

PURPOSE: U203 is derived from U201, and its operation and control parameters are almost identical. The purpose of U201 is to compute variables using either grids, vectors (station data), or both; to interpolate to a common set of station locations in the case of gridpoint calculations; and to write the vector data to a TDLPACK vector dataset. Much computational capability is available through a call to subroutine OPTION, governed by the variable ID, which acts as a switcher to computational subroutines. Input can include TDLPACK gridpoint data from one or more files, TDLPACK vector data from one or more files, and "constant" data in the MOS-2000 External Random Access System. Since all follow-on MOS programs operate on vector data, all output is vector.

U203 differs only in that gridpoint data or computations done on gridpoint data are not necessarily interpolated to station locations, but can be written as gridpoint data, the grid characteristics being whatever the input was or whatever was used in the computations. The purpose of U203 is to prepare either gridpoint or vector data for input to GEMPAK for viewing. All the computational capabilities of U201 are retained; the interpolation to station locations and the in-line processing of such data (e.g., making binaries) is optional in U203.

The primary difference in input from U201 to U203 is that instead of providing one "interpolated" output, there is a "vector" and a "gridpoint" output. The outputs can be used by scripts to view and overlay the data with GEMPAK.

Even though the input to U203 is very similar to U201, a full description in included here because it is believed this will be a heavily used program.

CONTROL FILE INPUT: 'U203.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU203.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U203" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U203 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U203HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

<u>IP</u>(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U203 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. Values have been defined as indicated below for values of J:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU203 DATA statement = 12. Changing the default unit number allows multiple runs of U203 or other programs within the same directory without overwriting.

(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).

(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).

(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).

(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).

(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.

(7) = The variable ID list in summary form. If there are errors, the predictor list will be written to the default output file unit KFILDO as well as to unit IP(7).

(8) = The variable ID list in summary form including the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)

(9) = The variable ID list in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the predictor constant file on unit KFILCP (see below).

(10) = The variable ID's for the first day (Day 1) as read from the archive tapes. This is just a list of all the packed gridpoint fields on the input files.

(11) = The variable ID's of the archived fields actually needed, in order as they appear on the archive files for Day 1.

(12) = The I,J positions of the stations on the NGRID grids, together with the call letters and names. NGRID, at this point, is the number of different grid combinations extant in the input data. That is, NGM archived data and

MRF archived data will have different grid definitions. After the data are read for the first day in the date list, the I,J positions are calculated and saved; IP(12) controls the printing of those values. If, during the U203 run, a different grid definition is encountered, it is accommodated automatically, but no print of the I,J positions is provided. In addition, when IP(12) ≠ 0, each directory record read from a vector input will be output on unit IP(12). For input of hourly data, this creates voluminous output.

(13) = Gridpoint fields. When the variable list indicates gridpoint values are to be written for viewing (JP(1,)>0), they will be written to unit IP(13). This pertains only to gridpoint fields written to files defined in Record Type 10.

(14) = Gridpoint fields. When the variable list indicates gridpoint values are to be contoured and written for viewing (JP(2,)>0), they will be written to unit IP(14). This pertains only to gridpoint fields written to files defined in Record Type 10.

(15) = Vector values. When the variable list indicates vector values are to be written for viewing (JP(3,)>0), they will be written to unit IP(15). These values may be original vector values or be interpolated from gridpoint fields. Note that the same variables will not be written to IP(13) and/or IP(14) and also to IP(15); if that is needed, two different variables can be used, one to produce gridpoint output and the other to produce vector output.

(16) = Diagnostics for constant and linearization routines (e.g., stations in threshold lists that are not being dealt with in this run).

(17) = The values of IFIND( ), ISTAV( ), and ITIME( ) after Day 1. (See Comments section for explanation.)

(18) = The variables saved in the MOS-2000 Internal Storage System after Day 1.

(23) = Information concerning opening and closing of sequential files.

For checkout, it may be advisable to set all these values to zero or the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (11(I10/),F10.0) Control Parameters

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

NSKIP - The number of errors that will be tolerated on Day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results, except when no data are found for Day 1, the program halts after Day 1.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - The increment in hours between date/times that are put into IDATE( ) (see Record Type 5) as a result of date spanning in subroutine DATPRO. This variable is also used in determining what data are to be saved in the Internal MOS-2000 Storage System. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL to the first date/time in IDATE( ).

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

NXGMIN, NXGMAX, NYGMIN, NYGMAX - Define a subset area in terms of NXL by NYL of the input grid. By setting these variables to zero no subset area will be defined and the max grid dimensions of the input grid will be used. A diagnostic in the KFILDO file will detail the latitude and longitude of the grid with the dimensions chosen. There are restrictions put on the grid due to GEMPAK requiring less then 97000 total points on output grid. If the grid size of the input grid exceeds this, a diagnostic will appear in the KFILDO file.

PXMISS - The value to substitute in the output for a secondary missing value of 9997. This allows the 9997 to be maintained if desired, set to zero in the case of the implied value of near zero for forecasts, or even set to some other value. Note that this does not apply to missing values other than 9997.

Record Type 4 - Format (I3,4XA60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

NAMDT - Name of file where this date list resides. When KFILDT = KFILDI, NAMDT is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) **Date List**

This group of records determines the date/times for which output is wanted. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous (positive) date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL (read in Record Type 1) to the first date/time in IDATE( ).

Record Type 6 - Format (2I3,1XA60) **Gridpoint and Vector Input Data Files**

This group of records identifies the sequential data sets from which the gridpoint and/or vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN. If all input data are from a constant file, only the terminator is necessary.

KFILIN(J) - Unit number for the input gridpoint or vector file. KFILIN( ) must be < 80 for gridpoint data and $\geq$ 80 for vector data. Also note that values 97, 99, and 44 through 49 are reserved for other uses, and, at present, values 11-20 are reserved for GEMPAK.

MODNUM(J) - Source of gridpoint data that are on this file (e.g., 6 = NGM gridpoint data, 7 = eta model, 8 = aviation model). This number is used to match with the "DD" in the variable ID. For vector data, this value must be zero.

NAMIN(J) - Name of file where this model input resides. (CHARACTER*60)

Note that when data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number. Having the same unit number is not mandatory, but is highly recommended. Vector data can either precede, follow, or be mixed with gridpoint data sets in the list read; the only restriction is that those with the same unit number are in proper sequence.

Record Type 7 - Format (I3,4XA60) **Random Access Files**

This group of records identifies the random access data sets from which constant data are read. Records are read until the terminator KFILRA( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND12 (ND12 ≤ 6). This Record Type 7 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMRA. If no data are needed from a constant file, only the terminator is necessary.

KFILRA(J) - Unit number for the random access constant file (J=1,NUMRA). Unit numbers must be in the range 44 to 49; see "Restrictions" for more information.

RACESS(J) - Name of file corresponding to KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) **Flat File**

This record (plus the terminator record) identifies the so called "flat file." Records are read until the terminator KFILFF( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. It is expected that there be a file defined here.

KFILFF - Unit number for the flat file.

FLATFF - Name of flat file. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) **Vector Output File**

This record (plus the terminator record) identifies the vector output file. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 9 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If vector data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and vector data are not written.

KFILIO - Unit number for the output file.

OUTSTA - Name of file where this output is to reside. (CHARACTER*128)

Record Type 10 - Format (I3,4XA60) **Gridpoint Output File(s)**

These record(s) (plus the terminator record) identifies the gridpoint output file(s). Records are read until the terminator KFILGO( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND13. This Record Type 10 is read by subroutine RDSNAM. This file(s) will be opened as 'NEW'. If gridpoint data are not to be saved, only the terminator is necessary here; in that case, files are not opened and gridpoint data are not written.

KFILGO(J) - Unit number(s) for the output file(s) (J=1,NGOUT).

OUTGRD(J) - Names of files where this output is to reside (J=1,NGOUT). (CHARACTER*60)

Record Type 11 - Format (I3,4XA60) **Station and Location Files**

This pair of records (plus the terminator record) identifies the file(s) from which the station (or location) information is obtained. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 11 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which vector data are to be written (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the default input file number, KFILDI, in which case DIRNAM(1) is not used. Also, KFILD(1) can equal KFILD(2), in which case the call letters list will consist of all stations on the directory file; note that both a list of stations and a directory cannot exist on unit KFILD(1) = KFILD(2). (Normally, the directory would not reside on KFILDI, but is accommodated if KFILDI = KFILD(2). The input would follow Record Type 12.)

DIRNAM(J) - Name of file matching KFILD(J). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (7(A8,1X)) **Station List**

This group of records identifies the stations for which vector output is desired. If KFILD(1) ≠ KFILDI (the default input file), this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which vector output values are desired (K=1,NSTA). This list is read within subroutine RDSTAD or RDSTAL by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations are not permitted by GEMPAK and should not be included. For NALPH = 1, the stations are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory are put at the end of the list. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank (b) filled on the right, e.g., 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 13 - Format (I3,4XA60) Variable File

This record (plus the terminator record) identifies the file from which the variable ID's and associated information is to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 13 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 14 - Format (I3,4XA60) Variable Constants File

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description and gridprinting information.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 14 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 15 - Variable List

This group of records comes in trios. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of trios, sans the terminator record, = ND4. This Record Type 15 is read by subroutine RDU203. Upon completion of reading this record type, N=1,NPRED.

The first of the trio contains the variable ID's.

Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2)

ID(J,N) - The first 3 (J=1,3) words of the variable ID (N=1,NPRED) plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only) (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

JP(J,N) - For each variable N, JP(,N) (N=1,NPRED) indicates print or no print when ≠ 0 or = 0, respectively, for gridpoint values (J=1), "zebra" gridprint map (J=2), and interpolated values at the stations (J=3). These values combined with IP(13), IP(14), and IP(15) allow easy control of output. For instance, all variables could have JP(2,) ≠ 0 and IP(14) ≠ 0 for a diagnostic run. All such print could be turned off by setting IP(14) = 0 in this control file. Alternatively, IP(14) could be ≠ 0 and a selected gridprint obtained by changing JP(2,) from 0 to ≠ 0.

The second of the trio contains the information to put into the flat file.

Format (1X,A4,1X,I3,1X,A10,F3.1,1X,A4,I2,1X,A30)

PARMS(N) - The name of the variable as used by GEMPAK (N=1,NPRED). Pick your poison. (CHARACTER*4)

INCTRL(N) - GEMPAK control number (N=1,NPRED). These values will be used by scripts to control the GEMPAK functions. The values are not fully determined, but a "1" may be used to contour the data.

PAREA(N) - The area for GEMPAK to plot (N=1,NPRED). GEMPAK has certain areas defined (e.g., "TX" for Texas). For gridpoint data, "ALL" can be used, in which case the lower left and upper right latitudes and longitudes will be written to indicate the complete grid area. (CHARACTER*39)

FILTER(N) - GEMPAK "filter" value for vector data or "skip" value for gridpoint data (N=1,NPRED). For skip, 0 means to skip no gridpoints when plotting; # means to skip every # number of gridpoints. For filter, 0 means to filter no stations when plotting; other values, in the vicinity of 1, indicate some filtering (thinning).

OUTDV(N) - GEMPAK output device (N=1,NPRED). The current values are:

XW = image will display in x-window, which the user can interact with.
GF = Each image will be saved as a gif file in the current directory with names gempak_1.gif, gempak_2.gif, etc.
PSI = Postscript--Landscape/Letter/Monochrome output to current directory, with names gempak_1.ps, gempak_2.ps, etc.
PSC = Postscript--Landscape/Letter/32 colors output to current directory, with names gempak_1.ps, gempak_2.ps, etc.
PSP = Postscript--Portrait/Letter/20 gray colors output to current directory, with names gempak_1.ps, gempak_2.ps, etc.
blank = GEMPAK script stops to allow the option of running U203 to create binary files, but not running GEMPAK to display them.

Any files already existing with the names above will be overwritten. (CHARACTER*4)

NDIMGE(N) - End image flag for GEMPAK (N=1,NPRED). A "1" means to plot a field then end the current GEMPAK image; "0" means to continue plotting on top of (overlaid on) the previous plot. If there is a sequence of 3 IDs, and the first has an NDIMGE = 1, the second has a 0, and the third has a 1, it will create 2 images. The first two IDs will be overlaid on one image, and the 3rd will be plotted alone as an image. (CHARACTER*2)

GTITLE(N) - GEMPAK title of variable (N=1,NPRED). Pick it. (CHARACTER*30)

The third of the trio contains the optional GEMPAK file name.

Format (1X,A60)

RESTOR(N) - Optional file name for GEMPAK (N=1,NPRED). If this optional file name is not to be used, include this record as an "999999". Note that this is not the terminator for the trio of ID records, which is also "999999". This file is used if the user wants to "restore" his/her GEMPAK settings. (CHARACTER*60)

CONTROL FILE INPUT: (Name read from U203.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) **Date List**

When the dates are not provided in file U203.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU203), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous (positive) date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U203.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided on file 'U203.CN', this group of records identifies the stations (or locations) for which vector output is desired. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from KFILDI. It is also not needed when KFILD(1) = KFILD(2); for this highly unusual case, both the station list and the associated information are read according to the format in the following control file description for Unit No. KFILD(2). U203 will not accommodate duplicate stations; there would be no need for them, and GEMPAK will not accommodate them.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which interpolated values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 9, Control File 'U203.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U203.CN) (Unit = KFILD(2))

Record Type 1 - Format Station Locations
(A8,1XA8,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. It is needed unless KFILD(1) = KFILD(2) = KFILDI; for this highly unusual case, both the station list and the associated information are read from the input control file KFILDI. When KFILD(1) = KFILD(2) ≠ KFILDI, this control file is the same as the one above and both the station list and the associated information are taken from this file. If both files are needed (the usual case; KFILD(1) ≠ KFILD(2) ≠ KFILDI), the call letters here are matched with those in the station list and the appropriate information extracted. When this station directory is alphabetical, the final list of stations will be alphabetical no matter the order read in. (Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run.) However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited, except when it also constitutes the list to be kept, in which case the number of entries is limited by ND1-1. No terminator is used. Most of this information is used only within subroutine RDSTAD or RDSTAL to give information about the stations. For example, there is no use of the elevation in the equation derivation. However, this information is available for a computation routine, if one is needed. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see Record Type 3).

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list unless KFILD(1) = KFILD(2), in which case these call letters comprise the station list and at the completion of reading this file, K=1,NSTA. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,1XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited, except when it also constitutes the list to be kept, in which case the number of entries is limited by ND1-1. No terminator is used.

CONTROL FILE INPUT: (Name read from U203.CN) (Unit = KFILP)

Record Type 1 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2) **Variable List**

When the variables to use are not in file 'U203.CN', this group of records contains the variable ID's and associated information (trios of records). When KFILP = KFILDI, this file is omitted, and the variable list is taken from input file KFILDI. Trios of records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDPRED. Upon completion of reading this record type, N=1,NPRED. See Record Type 15, file 'U203.CN', unit KFILDI, for other details; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U203.CN) (Unit = KFILCP)

Record Type 1 - Format
(3(I9,1X),I3,2XA32,I3,F11.4,F10.4,F9.2,F8.2,2XA12) **Variable Constants**

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all MOS-2000 users and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be very unusual for that to be the case. Note that the format matches that for a file input to other programs such as U600, U660, and U850.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

<u>IDTEMP</u>(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

<u>PLAINT</u> - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

<u>ISCALT</u> - When IDTEMP(1) matches an ID(1,N), ISCALT is stored in ISCALD(N). This variable is the scaling constant (power of 10) to use when packing the interpolated data.

<u>SMULTT</u> - When IDTEMP(1) matches an ID(1,N), SMULTT is stored in SMULT(N). This variable is the multiplicative factor (power of 10) to use when gridprinting the gridpoint data.

<u>SADDT</u> - When IDTEMP(1) matches an ID(1,N), SADDT is stored in SADD(N). This variable is the additive factor to use when gridprinting the gridpoint data.

<u>CONTT</u> - When IDTEMP(1) matches an ID(1,N), CONTT is stored in CINT(N). This variable is the contour interval to use when gridprinting the data. It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT.

<u>ORIGNT</u> - When IDTEMP(1) matches an ID(1,N), ORIGNT is stored in ORIGIN(N). This variable is the contour origin to use when gridprinting the data It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT.

<u>UNITST</u> - When IDTEMP(1) matches an ID(1,N), UNITST is stored in UNITST(N). These characters define the units of the data after application of SMULTT and SADDT and are used in the visual inspection of the data in gridprinted maps. (CHARACTER*12)

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDPRED, much of it associated with the plain language description. See Chapter 4 Variable Identification, in "MOS-2000," <u>TDL Office Note</u> 00-1, for details.

<u>DATA INPUT</u>:

All gridpoint and vector (station) data input to U203 will be in the MOS-2000 TDLPACK format read from sequential files, the vector data being accompanied by one or more directory records. The unit numbers and file names are provided in KFILIN(J) and NAMIN(J) (J=1,NUMIN), respectively. Constant data are provided in the random access MOS-2000 External File

System; these data are also in TDLPACK format, except for the call letters (directory) record. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG.

A. SEQUENTIAL GRIDPOINT DATA

One or more sources of gridpoint data are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U203.CN', Record Type 6. Each file is closed when an EOF is reached.

The definition of the grid to which the data in each record pertain is contained in that same record. The interpolation routines (when used) use this information, and precompute station locations (convert latitude/longitude to I/J) so that this computation doesn't have to be done for each record. However, a check is always made to assure that the correct grid information is used. In fact, not only can the data sources have different grid definitions, the definition within a particular source can change once or many times. It might be normal for the grid definition to change during the archive period. Be aware, though, that each time the grid definition changes when reading records that are to be used, extra computation must be done. Normally, ND11 (see "Setting up the Driver DRU203") will be $\geq$ the number of input grid sources (unit numbers).

For gridpoint data, the unit number KFILIN( ) must be < 80 (Nos. 44 through 49 are reserved for random access data), and the model number MODNUM( ) must match the model number DD in the ID's for which data are to be taken from that unit. The RR is operative for the lookback feature.

B. SEQUENTIAL VECTOR DATA

One or more sources of vector data are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U203.CN', Record Type 6. Each file is closed when an EOF is reached.

Each source (file) must have a directory record at the beginning, and one or more other directory records can occur in the same file following a trailer record (see MOS-2000, TDL Office Note 00-1, Chapter 6). Each file, even read on the same unit number, must have a directory record at the beginning. The occurrence of a new directory record causes some additional computation.

For vector data, the unit number KFILIN( ) must be $\geq$ 80 (Nos. 97 and 99 are reserved for other uses) and the model number MODNUM( ) (see Record Type 6) must be zero. The RR is (is not) operative when the DD in the variable ID is (is not) zero. In this way, hourly data, which will have DD = 0, will have RR operative, while output from a previous run of U201, which will (usually) have DD > 0, will not have an operative RR. (This feature is associated with the saving of data internally so that the data will be available when necessary. It will have to be watched carefully for possible unexpected results. It is believed that, for instance, the most output from U203 that will have DD = 0 will also have RR = 0, so it won't matter whether RR is operative or not. <u>However, output hourly data with RR > 0 run through U203 after a U201 run may create a problem.</u>)

C. <u>RANDOM ACCESS VECTOR DATA</u>

Up to 5 sources station-oriented random access data are accommodated. These data exist in the MOS-2000 External Random Access File System. These data are accessed with prescribed unit numbers (45 through 49), depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

In addition, station elevation, latitude, and longitude are available from Unit Number KFILD(2) and are stored permanently for use in NELEV( ), STALAT( ), and STALON( ), respectively.

D. <u>RANDOM ACCESS GRIDPOINT DATA</u>

Gridpoint data can be accessed from the external random access files with Unit No. 44. Normally, this would be constant data such as terrain height.

<u>DATA OUTPUT</u>

There are two forms of data output, besides the diagnostics and other information on Units KFILDO and IP( ).

A. <u>ASCII FOR VIEWING OR PRINTING</u>

Gridprinted maps of each variable for which JP(1,) ≠ 0 without contours and/or for which JP(2,) ≠ 0 with contours are written to Units IP(13) and IP(14), respectively, when those units numbers ≠ 0. Vector values are written for each variable for which JP(3,) ≠ 0 to Unit IP(15) when IP(15) ≠ 0. This makes for easy control; all such print can be eliminated by changing the IP( ) values, and there is still control on each variable J through JP(,J).

B. <u>A FLAT FILE FOR CONTROL OF GEMPAK SCRIPTS</u>

Certain of the control information read by U203 in the U203.CN file is written to the flat file, which is used by scripts to control the operation of GEMPAK.

C. <u>BINARY FILE FOR VECTOR DATA</u>

One file is used for vector data for GEMPAK. The file number and file name are specified in Record Type 9.

D. <u>BINARY FILE(S) FOR GRIDPOINT DATA</u>

One or more files are used for gridpoint data for GEMPAK. The file numbers and names are specified in Record Type 10. All grids with the same characteristics (size, placement, etc.) are put on the same file, but each different set of characteristics requires a different file. For instance, an archived AVN and ETA grid would be put on separate files because of different sizes, etc.

<u>EXAMPLE CONTROL FILE</u>: 'U203.CN'

An example exists as file 'U203.CN' in directory 'users24.glahn.u203.dru203' on ice. The easiest way to set up a run is to take an existing control file and modify it; <u>DO NOT START FROM SCRATCH</u>.

<u>OUTPUT</u>:

Output that can be printed can be put onto one or more of several files as described above under control file 'U203.CN', Record Type 1 and the definition of KFILDO in the driver DRU203. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control without jeopardizing data.

<u>RESTRICTIONS</u>

GEMPAK has some features that impose restriction on U203, and more may be found, as the use of the GEMPAK subroutines called by U203 are not well understood. The restrictions are associated with GEMPAK file use. First and foremost, GEMPAK assigns its own unit numbers; U203 has no control over them, and GEMPAK has reserved numbers 11-20 for its use. This means that U203 must not use unit numbers 11-20 for its purposes. Since the "normal" default output KFILDO file is 12, some other number must be used. Also, it is standard practice to use the IP( ) numbers as either 12 or the same number as the IP( ) number. That is, IP(15) might be Unit No. 15. This will not now (reliably) work. It is recommended that all IP( ) numbers be given either the number 32 (KFILDO = 12 + 20) or the IP( ) number + 20 (e.g., IP(15) = 35). This then, will put the KFILDO default output to the file with a name ending in "32" as explained in Record Type 1.

In addition, the maximum number of output files seems to be set in an INCLUDE file to 3. This should not be a major difficulty (why would you need more than one vector and two gridpoint files?).

While the PARMS( ) variable can usually be whatever the user chooses, GEMPAK processing requires some specific PARMS. For instance, in order to plot wind barbs in knots (BRBK), one of the following pairs, as read by U203, must appear in the ASCII flat file: (SKNT, DRCT) or (UKNT, VKNT). For wind barbs in m/s (BRBM), use (SPED, DRCT) or (UWND, VWND).

Most other restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. However, GEMPAK routines called by U203 will not handle very large grids; this is a GEMPAK restriction. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. North Polar Stereographic, Lambert, and Mercator gridpoint data are accommodated. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The IBM treats these as comments, so they won't bother.

The TDLPACK format for gridpoint data provides for both a primary and a secondary missing data indicator. If a primary value is found during unpacking of gridpoint (not vector) data, this is flagged at U203 completion by a statement on the output KFILDO file. However, the computations on grids do not provide for missing values, and the packed output records will not indicate that missing values may be present. It is not expected that gridpoint data will have missing values. If missing values become possible at some time, then U203 will have to be modified to handle the situation. However, if the missing gridpoints are in an area where no interpolation to stations is to be done (e.g., in the Pacific in the lower left of the grid), no adjustment would need be made. It's only when the missing data could be involved in the bilinear or biquadratic interpolation would there be a problem.

Arrays FD1( ), FD2( ), ... FD7( ) have been provided for general work arrays. Arrays FDVERT( ) and FDTIME( ) have been reserved for use by subroutines VERTP and TIMEP, respectively. FDSINS( ) and FDMS( ) have been reserved for use by routines dealing with the sine of the latitude and a map factor. L1DATA( ) and SDATA1( ) are reserved for use by linearization and constant routines. Some care is needed when writing new and complicated computational subroutines that the FD1( ), FD2 ( ), etc. arrays are not overwritten in other routines. For instance, if a routine were to call OPTN2 which needed to call VORTH, then VORTH would use arrays FD1( ), FD2( ), and FD3( ). Anything the calling program had in these arrays would be wiped out.

The binary indicator "B" in the first word of the variable ID must be only 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), or 5 (indicating a grid binary). Subroutine CKIDS1, called from RDU203, prints a diagnostic if a binary is not indicated as a 0, 1, or 5, but the variable is not eliminated. The data will be missing, and will not be written because it is not known whether it was intended for the data to be gridpoint or vector.

Since vector data are allowed as input, and even the output of a previous U201 run, it not known *a priori* whether the computational elements of the ID (e.g., "B" or "RR") have already been used in preparing the data associated with the ID or whether these elements need to be operative. U203 tries initially to find the full ID on the input (computational elements not to be operative), then if the data cannot be found, the computations are done on the more basic data which must be available. <u>With this flexibility comes the restriction that the "T" (transformation) and "B" (binary) capabilities cannot be operative on input vector data.</u> For instance, suppose the sine of the day of the year were input as a vector. It <u>could not</u> be made into a binary or transformed with the "T". However, if it were computed through OPTION, it <u>could</u> be transformed or made into a binary. While this is a restriction, it is thought to be unimportant; in case of difficulties with this restriction, just do what is needed in a computational subroutine. (If it is necessary to have a binary of a variable that exists only as a vector, it can be computed, packed, and written in U660.)

When a runtime offset, RR, is not zero and MODNUM( ) is not zero, meaning that data from a previous model run are needed, then data are carried forward in the MOS-2000 Internal Storage System. In the storing of data, as the file is read from a previous cycle to the current one, some data from an intermediate cycle may be saved that are not needed. This does not cause an error, and comes into play rarely, because RR = 0 except possibly for developing medium-range forecast equations, in producing data for special studies, or in developing predictands from observations over a period of time (e.g., summing rainfall amounts or averaging temperatures). <u>When the input is vector data (MODNUM( ) = 0), RR is used only when DD in the variable ID = 0.</u> This means that for a (vector) variable input from a previous run of U201 in which model data were used, it is assumed that the computations have already been made.

The unit numbers for the random access files must be in the range 44 through 49 (although U203 will not write to a random access file); those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
44	400-499	Gridpoint data (not yet implemented)
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U203
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

<u>COMMENTS</u>

U203 is derived from U201 and except for the output is essentially the same code. U201 was written to produce "predictors," and that terminology is still used in some places. However, the word "variables" is more appropriate, since the use of the data produced is not restricted to predictors.

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing data will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a field is to be computed in OPTION and a field needed for the computation cannot be found, a diagnostic will be provided (possibly "****PREDICTOR NOT IDENTIFIED IN OPTN2") <u>and</u> in addition "****ALL VALUES IN FIELD...ARE MISSING IN PKMS99". While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. This probably means a set of dates should be supplied to U203 for which it is known there are good data. Then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed. Note that an error may produce missing data values, and may be counted for both.

On Day 1, GFETCH is entered directly at least once for every variable (except when a field can be reused, such as vorticity and a binary from vorticity) because it is not yet known when OPTION must be entered. ("Day 1 is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) <u>is not</u> counted as an error in PRED25 for Day 1. It <u>is</u> counted as an error in PRED26, because GFETCH should not be entered directly when OPTION is needed.

It is possible that no gridpoint or vector data on sequential files be used, and all input furnished be on "constant" random access files.

A number of computational routines have been written for U201 and are available for U203. Also, a few have been written primarily for illustrative purposes. For instance, L1D50CFFF linearizes variables with a set of thresholds. L2D601000 and L2D600101 each linearize a pair of variables with sets of thresholds. These are provided to show how the thresholds can be provided and accessed, and for a template for other linearization routines. The routine COMBIN is provided to show how one might combine output from two models into one variable. This kind of combination is not recommended, but is possible in U203. If the combination is not a simple one, as it is in COMBIN, then a new variable name will have to be coined.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1.

It is unlikely that when no variable in a run has a particular model number, that the file containing that model will be needed; therefore, the file is closed. Closing the file keeps the data from being read when not needed. A highly unlikely exception is for a subroutine like COMBIN that would use data from a model that was not indicated by the model number in the variable ID. If this is the case, include a variable with the (otherwise omitted) model number to force the file to remain open. This variable can be a dummy (see OPTION subroutine) to keep it from being written.

It may be that a computational subroutine will need data at a date/time previous to that needed by any variable read in Record Type 15 (e.g., computation of mean value composed of previous values). In order to force saving of data for all date/times needed, include a variable with the first word = 799000000 and an appropriate RR. This will cause return from OPTION with IER = -2; a record with that ID will not be written. This will not be counted toward errors or missing data.

It is generally <u>not</u> necessary that each variable be present for Day 1 for that variable to be used on subsequent days (see the following section "Characteristics of the U203 Lookback Feature").

A count is kept of each time a missing value indicator is found when unpacking <u>grids</u> (not vector data). This is unexpected, and at the end of a run, the value is printed if non-zero.

A "missing" value of 9997 encountered in input <u>vector</u> data is set to PXMISS (see Record Type 3). This allows 9997 to remain intact (PXMISS = 9997), set to zero (PXMISS = 0), set to "normal" missing (PXMISS = 9999), or even some other value.

The variables IFIND( ), ISTAV( ), and ITIME( ) that can be printed for each variable in table form under IP(17) control are set after processing Day 1 data and have the following meanings:

IFIND	= 1	When the variable can be found directly from the input and does not have to be computed through OPTION.
	= 0	Variable has been identified in OPTION and will be computed there.
	= 2	Variable has been identified in OPTION but data returned are missing. Therefore, it is not yet known whether this variable will be input directly or computed through OPTION. It is also not known whether the output was expected to be vector or gridpoint.
ISTAV	= 1	When the data are vector format either on initial input or on return from OPTION.
	= 0	Variable is gridpoint.
	= 2	Variable has not been found yet and it is still not known whether its input will be vector or gridded.
ITIME	= 1	When the time offset RR is operative.
	= 0	Otherwise. This may (also) occur when RR = 0.

<u>CHARACTERISTICS OF THE U203 LOOKBACK FEATURE</u>

U203 attempts to save data from previous cycles (date/times previous to NDATE, the date/time being processed). Such data are denoted by RR > 0 in ID(3). Consider the following conditions:

o The variable needed <u>is in</u> the ID list input by the user in Record Type 15.

- All data needed are found for Day 1

RR = 0	No data are saved from one NDATE to the next.
RR > 0	Data for the variable with RR > 0 are saved for the number of hours indicated by RR.

- Some data needed are not found for Day 1

RR = 0	Not a problem.
RR > 0	Usually not a problem.

o The variable needed <u>is not in</u> the ID list input by the user, but is needed by a computational subroutine.

- All data needed are found for Day 1

RR $\geq$ 0	Shouldn't be a problem, <u>provided the subroutine looks for all variables it will need for subsequent cycles</u>. This may require the subroutine to ask for data not needed for the particular cycle being processed (e.g., max temperature is not calculated for each cycle).

- Some data needed are not found for Day 1

RR $\geq$ 0	No recovery; U203 has no way of knowing that a subroutine tried to access a variable and was unsuccessful. An example is 1000-850 mb thickness, and either the 850-mb or 1000-mb height was not available for the tau specified. The needed heights will not be saved <u>unless</u> the height itself is a variable in the ID list and has the same tau as the thickness.

While every effort has been made to make the lookback feature robust, and to not require the user to specify (possibly in a separate list) all the variables that might be needed in all subroutines, not all situations could be handled. It is believed that all will be well when all data needed for Day 1 are present. It should be rare when one would need to make a run that would require data that were not available for the first date/time.

<u>SETTING UP THE DRIVER DRU203</u>

The preparation of the driver for a particular U203 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B - Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine.

ND1 - Maximum number of stations (or points) that can be dealt with (i.e., interpolation done for). Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) unless, of course, the station directory is to be used as the station list. ND1 must also be GE NBLOCK (see below).

ND2 - ND2*ND3 is the maximum size of the grid that can be dealt with. ND2 and ND3 are set separately to highlight the possible dimension of the grids. However, in the called routines, the size is only limited by the product, not each dimension individually.

ND3 - See ND2.

ND4 - The maximum number of variables (variables) for which output can be provided.

ND5 - Dimension of IPACK( ), IWORK( ), and DATA( ). ND2X3 and ND5 should be set in the driver DRU203:

```
ND2X3 = MAX(ND1,ND2*ND3,ND12)
ND5=ND2x3
```

ND6 - Maximum number of all sequential file input sources (e.g., models) that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records of the date/time of Day 1 on all input files.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

ND11 - The maximum number of grid combinations that can be dealt with. For instance, if the NGM and AVN data are being used, and the grids are different, then ND11 would be $\geq$ 2.

ND12 - Maximum number of random access files (MOS-2000 External File System) that can be used. Limit is 6 because 6 unit numbers have been reserved for random access files.

ND13 - The Maximum number of gridpoint output files.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.) Note that ND1 must be GE NBLOCK.

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND2*ND3). Every effort has been made so that the variables will not be overflowed if values too small are used; however, some danger may still exist.

TYPES OF COMPUTATIONS IN U203

It is believed that most preparation of predictor and predictand datasets, as well as those for verification and other purposes, can be done by U201. U203 can prepare data in a similar fashion and should be useful for viewing datasets. The handling of input gridpoint and vector data is markedly different.

A. GRIDPOINT DATA

U203 will perform most of the necessary basic functions, as is indicated in the table below ordered by processing indicator. It is expected that only polar stereographic and Lambert archives will be used, and those are the only arrangements made. There is no hard coding for these map projections, except that routines pertaining to other projections are not provided.

PROCESSING INDICATORS

B - Binary (Value < threshold = 0; Value $\geq$ threshold = 1)
- 1 = point binary
  BINARY
- 5 = Grid binary
  GRIDB

DD - Model number
- 1 = Trajectory
- 5 = Lamp
- 6 = NGM
- 7 = Eta
- 8 = Avn

V - Vertical Processing (Same field (CCCFFF) and projection (ttt), different levels (UUUU and LLLL)
VERTP
- 0 = None
- 1 = Difference (UUUU-LLLL)
- 2 = Sum
- 3 = Mean

```
  T - Transformation
      TRANS
          1 = Square
          2 = Square root

  RR, O, HH - Time Operation on Fields (1) and (2)
      TIMEP
         Field (1) = Run time offset (NDATE - RR), projection tau + HH
         Field (2) = Run time offset (NDATE - RR), projection tau
              1 = Mean
              2 = Difference (1)-(2)
              3 = Maximum
              4 = Minimum
         Field (1) = Run time offset (NDATE - RR), projection tau + HH
         Field (2) = Current run time, projection tau
              5 = Mean
              6 = Difference (2)-(1)
              7 = Maximum
              8 = Minimum

  I - Interpolation
      1 = Biquadratic
          INTRPA
      2 = Bilinear
          INTRPB
      3 = Precipitation
          INTRP

  S - Smoothing
      1 = 5-point
          SMTH5
      2 = 9-point
          SMTH9
      3 = 25-point
          SMTH25
      4 = 25-point applied twice
          SMTH2X
      5 = 25-point applied thrice
          SMTH3X
```

The processing is always done in the following order:

```
  V,
  RR, O, HH,
  S,
  I,
  T, and
  B,
```

except that grid binaries (B = 5) are computed prior to S.

When OPTION is entered, the data returned can be either gridpoint (in which case S, I, T, and B are operative) or vector. When vector, only T and B (point binaries) are operative.

B. VECTOR DATA

The computational capability for vector data input has the following three primary purposes:

(1) To allow the computation of variables (probably predictors) that are a combination of observations and gridpoint data (e.g., pseudo stratification).
(2) To allow the production of variables (probably predictands) that combine observations from more than one hour.
(3) To transform a variable or produce a binary when the data originated as gridpoint.

Partly because this input can be from a previous run of U201, the processing indicators in the IDs may be there because the computations have already been made, not because they need to be made as in the case of gridpoint data. For this reason, all processing of vector data must be done in subroutines through OPTION. However, when vector data are returned from OPTION, the T and B operators are still operative, except for CCC in the range 300-399 (combination of models), in which case it is assumed all the processing has been done. (This concept may need some tailoring as specific uses are identified. However, use of T will probably be rare.)

VARIABLES HANDLED IN U203

All basic fields can be processed by TIMEP and VERTP (see above) and interpolated, plus the following variables indicated below can be computed and interpolated. It is noted that TIMEP and VERTP can call (through OPTN2) other routines for processing, and TIMEP can call VERTP. In this way, a field on which time processing is desired can ask VERTP to do a vertical process on fields that have to be computed by other routines. The user need not arrange for the basic fields to be present; U203 will arrange that.

CCCFFF	Routine	Write-up No.	Description
003010	MIXRAT	2.54	Mixing ratio on an isobaric surface
003011	MIXRAT	2.54	Mixing ratio on a constant height surface
003016	MIXRAT	2.54	Mixing ratio on a sigma surface
003030	SPECHUM	2.55	Specific humidity on an isobaric surface
003031	SPECHUM	2.55	Specific humidity on a constant height surface
003036	SPECHUM	2.55	Specific humidity on a sigma surface
003100	DEWPT	2.50	Dew point on an isobaric surface
003101	DEWPT	2.50	Dew point on a constant height surface
003106	DEWPT	2.50	Dew point on a sigma surface
003200	KINDEX	2.53	K Index
003265	NONCNVP	2.52	3-h non-convective precip amount
003270	NONCNVP	2.52	6-h non-convective precip amount
003280	NONCNVP	2.52	12-h non-convective precip amount
004002	GWIND	2.26	Grid-oriented geostrophic u-wind from heights
004012	GWIND	2.26	Earth-oriented geostrophic u-wind from heights

004102	GWIND	2.26	Grid-oriented geostrophic v-wind from heights
004112	GWIND	2.26	Earth-oriented geostrophic v-wind from heights
004212	GWIND	2.26	Geostrophic wind speed from heights
004210	WSPEED	2.33	Wind speed from u- and v- components
004011	EOWND	2.30	Earth-oriented u-wind from u and v components
004111	EOWND	2.30	Earth-oriented v-wind from u and v components
006010	VORTW	2.22	Relative vorticity from u and v components
006020	VORTH	2.20	Geostrophic relative vorticity from heights
006110	DIVW	2.24	Divergence from u and v components
007210	TTOTALS	2.51	Total totals index
010201	FORIER	4.11	Sin DOY (basic date)
010202	FORIER	4.11	Sin 2*DOY (basic date)
010203	FORIER	4.11	Cos DOY (basic date)
010204	FORIER	4.11	Cos 2*DOY (basic date)
010201	FORIER	4.11	Sin DOY (basic date plus projection)
010202	FORIER	4.11	Sin 2*DOY (basic date plus projection)
010203	FORIER	4.11	Cos DOY (basic date plus projection)
010204	FORIER	4.11	Cos 2*DOY (basic date plus projection)

Check "Computer Programs for MOS-2000," TDL Office Note 00-2, for other routines available.

WRITING NEW COMPUTATIONAL SUBROUTINES

New computational routines can be written as needed. The existing routines, such as VORTH, can be used as templates. Generally, the steps to be followed are:

1) Include the routine in the u201lib or u203lib directory, and see that it is included in the makefile.

2) Enter the routine name with its calling sequence in the switching routine OPTION and probably in OPTN2 with the proper IF tests. Generally, this switching would be on IDPARS(1) and/or IDPARS(2).

3) Use the work arrays FD1( ), FD2( ), etc. as needed.

4) The new routine can call OPTN2 (or even OPTION) if computed variables are needed for which routines are already written. If this is done, the complete OPTN2 calling sequence must be used for the new routine, so that it can be passed to OPTN2.

5) If OPTN2 is called or OPTION recalled, care must be taken to not overlap the use of the arrays FD1( ), FD2( ), etc. That is, if VORTH were to be called by OPTN2, it would use arrays FD1( ), FD2( ), and FD3( ). If these arrays were used by the new routine, they would be wiped out when OPTN2 and VORTH were called. However, data are never carried from variable to variable in FD1( ), etc. That is, each time a new variable is dealt with in OPTION, each of the FDi( ) series is available for use. It is only in the computation of a particular

variable that the work space must be managed by the user. Arrays NELEV( ), STALAT( ), and STALON( ) hold the station elevations, latitudes, and longitudes, respectively, in the order of the stations in CCALL(,) that can be used in any subroutine; they shouldn't, of course, be modified. Other "constant" data must be obtained from the MOS-2000 External Direct Access Storage System by calling subroutine CONST. Elevations can also be obtained through CONST, if desired.

(6) When a computational routine returns vector data, it must set ISTAV = 1; when the data returned are gridpoint, it must set ISTAV = 0.

(7) As a special case, when data for a particular hour cannot be available (e.g., max/min temperature at more than one hour each per day), the computational routine can set IER = -1 to indicate a record is not to be written for that variable for that date/time.

(8) OPTION will produce a diagnostic as appropriate when a called routine does not return IER = 0. The computational routine probably does not need an additional diagnostic. Also, the routine should not return IER = 47 (indicating missing data), but something else coordinated with the error table for MOS-2000 contained in "MOS-2000," TDL Office Note 00-1. OPTION, PRED25, and PRED26 will then be able to handle the situation.

SPECIAL CONSIDERATIONS FOR LINEARIZATION ROUTINES

One of the powerful elements of MOS is that even though the basic statistical model may be linear, the predictors can be very non-linear. Such possibilities are provided by cumulative binaries, grid binaries, and the "T" in the ID. Non-linear combinations of basic or even of derived fields can be provided as predictors. One-dimensional (1-d) and 2-dimensional (2-d) linearization techniques have been used in the past, but are somewhat difficult to implement in either the developmental or implementation systems. Provision has been made for such linearizations to be performed within U203 subroutines.

Basically, the technique is to divide the predictor to be linearized into discrete categories by using thresholds and to provide a "linearized" value for each category. This might well be the probability of that category occurring. These values must have been determined from previous developmental work. For two variables, two sets of thresholds--one for each variables--are needed, and the result will be "boxes" within a 2-d diagram or table, and each box must have a "linearized" value. This sounds relatively simple, until one realizes that the sets of thresholds may well vary, not only with the variable(s), but with station or region, time of year (season), projection, and run (cycle) time. Not only is the development of such thresholds a chore, but how does one arrange to implement this technique?

The writeups for subroutines L1D and L2D provide information on the implementation within U203 of the linearization techniques, including the file naming convention and record structure for the thresholds. Thresholds are read by L1D1 or L2D1 on first entry to them as appropriate on Unit No. 97, which is closed upon exit and, therefore, can be used for all such files. The thresholds are stored in the MOS-2000 Internal Storage System and accessed by L1D1 and L2D1 when needed. These threshold files must be available or L1D1 and L2D1 will try to open and read them for each day in the sample; that is, if the thresholds are not available in the MOS-2000 Internal Storage System, they will be read from a file of the appropriate name. This means that for a new season or cycle, a new set is read and stored, so all thresholds are not necessarily read on Day 1 (note that "Day 1" used to represent the first "case," and, therefore, only the first cycle).

OPTION switches to L1D or L2D based on (some portion of) CCC (and possibly FFF). L1D and L2D are themselves switchers into which must be build the calls to, for example, L1D50CFFF and L2D601000, respectively. To insert a new linearization technique, the following must be done:

1) Include the routine in the u201lib directory, and see that it is included in the makefile. Note that for 1-d linearizations, the existing L1D50CFFF will handle most needs, and a new routine won't be necessary.

2) Enter the routine name with its calling sequence in the switching routine L1D or L2D with the proper IF tests.

3) Since non-specific linearization routines are called, which can call other computational routines through OPTN2, include in the call sequence all variables that are in the call sequences to those routines.

4) Guidelines for managing the arrays FD1( ), FD2( ), etc., must be followed as for other routines as explained in the previous section.

NONSYSTEM ROUTINES USED

Use the load line in file '/lamp2000/dru203', dataset 'u203.com' on ice. Note that the script u203.ksh is also necessary.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: /lamp2000/u203lib on ice. The driver is in /lamp2000/dru203 on ice.

TDPAVG

AVERAGES TEMPERATURE AND DEW POINT TO REMOVE NEGATIVE DEW POINT DEPRESSION

Feng Liang
May 23, 2003

PURPOSE: To compute the average of the temperature and dew point when the dew point is greater than the temperature. This routine is accessed for the temperature or dew point MOS-2000 IDs:

002 303 - consistency checked temperature

003 303 - consistency checked dew point

RESTRICTIONS:

None.

COMMENTS:

The gridded temperature and dew point forecasts are interpolated to stations before comparison. The data won't be changed unless the dew point is greater than the temperature. Missing data are replaced with 9999.

NONSYSTEM ROUTINES USED:

GFETCH, INTRPA, INTRPB

LANGUAGE: FORTRAN 77.

LOCATION: /home21/tdllib/u201lib (HP) and
/nfsuser/g06/u201lib (IBM-SP).

OBSPOPC

CLASSIFIES PRESENT WEATHER PRECIPITATION OBSERVATIONS INTO ONE OF THREE POSSIBLE CATEGORIES

David RUdack
July 19, 2000

PURPOSE: To classify present weather precipitation observations into one of three possible categories (drizzle, stratiform, or convective). If the present weather observation does not indicate that precipitation is occurring, this subroutine returns a value of 9999 as a flag for no precipitation. In this way, this subroutine can create the predictand for the development of the conditional probability of precipitation characteristics. This routine contains two separate algorithms in order to process the data depending upon the value of NDATE. OBSPOPC is almost exclusively used for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID for categorizing the precipitation characteristic is:

708605000 000000000 000000000 000

RESTRICTIONS:

Due to the conversion from SAO observations to METAR observations in 1996, the present weather codes differ in the hourly data archive before and after December 1, 1996. While this subroutine is equipped to handle both numbering conventions, the user must not try to process in a single U201 run dates that span this December 1, 1996, cutoff due to the configuration of the internal MOS-2000 mass storage system. Therefore, to correctly process data across this threshold, the user must do one run containing dates up to and including November 30, 1996, and then start a new run for December 1, 1996, and beyond.

If no data are available for the first processing date, the subroutine will not work.

COMMENTS:

In categorizing the precipitation characteristic at each reporting station, all forms of precipitation events (rain, snow, freezing precipitation, hail, etc.) are counted, including the occurrences of ice crystals. Note, too, that in assessing the precipitation characteristic, all three present weather groups available in the hourly archive after December 1, 1996, are considered. The following values are returned from the subroutine:

1 = drizzle;
2 = stratiform type of precipitation;
3 = convective or showery type of precipitation;
9999 = no case (either no precipitation, undetermined precipitation type, or missing data).

For automated stations that do not have a present weather sensor, the value returned from OBSPOPC will be 9999.

For more information on the present weather codes and the hourly archive, see TDL Office Note 00-01.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH

<u>LANGUAGE</u>: FORTRAN 90.

<u>LOCATION</u>: /home21/tdllib/u201lib (HP)
/nfsuser/g06/mos2k/u201lib (IBM-SP)

OBSPOPO

CLASSIFIES PRESENT WEATHER INTO
A YES/NO DESIGNATION ACCORDING TO
THE OCCURRENCE/NONOCCURRENCE OF PRECIPITATION

Rebecca Allen

May 13, 1999

PURPOSE: To classify station (vector) present weather reports into a yes (=1) or no (=0) designation according to whether precipitation is occurring or not at the station at that hour. OBSPOPO is an almost exclusively for U201 and similar routines (e.g., U203), and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID for the designation of precipitation occurrence is:

708504000 000000000 000000000 000

RESTRICTIONS:

Due to the conversion from SAO observations to METAR observations in 1996, the present weather codes differ in the hourly data archive before and after December 1, 1996. While this subroutine is equipped to handle both numbering conventions, the user must not try to process in a single U201 run dates that span this December 1, 1996, cutoff due to the configuration of the internal MOS-2000 mass storage system. Therefore, to correctly process data across this threshold, the user must do one run containing dates up to and including November 30, 1996, and then start a new run for December 1, 1996, and beyond.

If no data are available for the first processing date, the subroutine will not work.

COMMENTS:

In determining whether or not precipitation is occurring at the station, all forms of precipitation events (rain, snow, freezing precipitation, hail, etc.) are counted including the occurrences of ice crystals. Note, too, that in assessing the occurrence of precipitation, all three present weather groups available in the hourly archive after December 1, 1996, are considered.

For automated stations that do not have a present weather sensor, the value returned from OBSPOPO will be 9999.

For more information on the present weather codes and the hourly archive see TDL Office Note 00-01.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90.

LOCATION: /home21/tdllib/u201lib (HP)
/nfsuser/g06/mos2k/u201lib (IBM-SP)

OBSPOPO3

CLASSIFIES PRESENT WEATHER INTO A YES/NO DESIGNATION ACCORDING TO THE OCCURRENCE/NONOCCURRENCE OF PRECIPITATION OVER A 3-HOUR PERIOD

David Rudack

July 1, 2000

PURPOSE: To classify station (vector) present weather reports into a yes (=1) or no (=0) designation according to whether precipitation occurred or not at the station during a particular 3-h period. OBSPOPO3 is an almost exclusively for U201 and similar routines (e.g., U203), and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID for the designation of precipitation occurrence during a 3-h interval is:

```
708564000 000000000 000000000 000
```

RESTRICTIONS:

Due to the conversion from SAO observations to METAR observations in 1996, the present weather codes differ in the hourly data archive before and after December 1, 1996. While this subroutine is equipped to handle both numbering conventions, the user must not try to process in a single U201 run dates that span this December 1, 1996, cutoff due to the configuration of the internal MOS-2000 mass storage system. Therefore, to correctly process data across this threshold, the user must do one run containing dates up to and including November 30, 1996, and then start a new run for December 1, 1996, and beyond.

For this subroutine to work properly, an additional input variable record must be included in the U201.CN file. This special record (CCC=799) is needed to help initialize U201's internal storage system. The record is:

```
799000000 0000000000 003000000 000
```

The RR value of 03 in the third word enables U201 to access the 3 hours of data preceding the first process date; these extra hours of data are required to classify the present weather for the 3-h interval. Note that a record with this special variable ID is not written to the output data file.

If no data are available for the first processing date, the subroutine will not work.

COMMENTS:

In determining whether or not precipitation is occurring at the station, all forms of precipitation events (rain, snow, freezing precipitation, hail, etc.) are counted including the occurrences of ice crystals. Note, too, that in assessing the occurrence of precipitation, all three present weather groups available in the hourly archive after December 1, 1996, are considered. For a given 3-h interval ending at time t, the observations from t-3, t-2, t-1, and t hours are considered. Note, too, that measurable precipitation is not considered in this variable; only the occurrence of precipitation at any one of the four hourly reports is needed to indicate that precipitation has occurred.

For automated stations that do not have a present weather sensor, the value returned from OBSPOPO3 will be 9999.

For more information on the present weather codes and the hourly archive see TDL Office Note 00-01.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90.

LOCATION: /home21/tdllib/u201lib (HP)
/nfsuser/g06/mos2k/u201lib (IBM-SP)

OBSPRWXBIN

PRODUCES FREEZING/NO FREEZING, SNOW/NO SNOW,
RAIN/NO RAIN BINARIES FROM PRESENT WEATHER

Rebecca Allen

May 1, 2000

PURPOSE: To produce any of three binary variables that represent either freezing/no freezing, snow/no snow, or rain/no rain. First, the present weather reports are classified into their respective precipitation type categories (see Comments). Then, depending on which predictor is requested, a binary indicator is set if the precipitation type occurred. OBSPRWXBIN is an almost exclusively for U201 and similar routines (e.g., U203), and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 IDs for the present weather binaries are:

```
708551000 000000000 000000000 000 - freezing/no freezing
708552000 000000000 000000000 000 - snow/no snow
708553000 000000000 000000000 000 - rain/no rain
```

RESTRICTIONS:

Due to the conversion from SAO observations to METAR observations in 1996, the present weather codes differ in the hourly data archive before and after December 1, 1996. While this subroutine is equipped to handle both sets of codes, the user must not try to process dates that span this December 1, 1996 cutoff due to the configuration of the mass storage system. Therefore, to correctly process data across this threshold, the user must do one run containing dates up to and including November 30, 1996, and then start a new run for December 1, 1996, and beyond.

If there are no data available for the first processing date, the subroutine will not work properly.

COMMENTS:

The present weather categories are as follows:
- 0 = no present weather, or non-precipitation reports (i.e. fog)
- 1 = freezing precipitation or anything in combination with freezing precipitation
- 2 = ice pellets in combination with non-freezing precipitation
- 3 = pure ice pellets
- 4 = pure snow
- 5 = rain and snow mixed
- 6 = pure rain
- 7 = thunderstorms

This classification scheme is identical to that used in obsptype to create short-range precipitation type predictands except for one difference. In obsptype, non-precipitation events are assigned a value of 9999 so these cases will be thrown out in U600. In the case of obsprwxbin, non-precipitation events are assigned a precipitation type value of 0, which in turn becomes a 0 (non-occurrence) for the binary. For the freezing binary (708551), values of 1, 2, or 3 indicate an event occurrence, and the value of the variable is set to 1; otherwise, the variable is set to 0. For the snow binary (708552), a value of 4 indicates an event occurrence, and the value of the variable is set to 1; otherwise, the variable is set to 0. For the rain binary (708553), values of 5, 6, or 7 indicate an event occurrence, and the value of the variable is set to 1; otherwise, the variable is set to 0.

In the data post-December 1, 1996, there is no way to distinguish whether the precipitation associated with the thunderstorms in category 7 was rain or snow. Therefore, if a thunderstorm is reported along with any other precipitation group, it will be coded according to that other precipitation group. Otherwise, it is coded as a 7 and considered to be liquid precipitation.

Automated stations can report a weather descriptor UP, which designates Unknown Precipitation. In this subroutine, reports of hail, squalls, and UP are assigned a missing value because the precipitation type is not known for these reports.

For more information on the present weather codes and the hourly archive see TDL Office Note 00-01.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90.

LOCATION: /home21/tdllib/u201lib (HP)
/nfsuser/g06/mos2k/u201lib (IBM-SP)

ADVCTW

COMPUTES ADVECTION FROM WIND

Christopher A. Fiebrich
August 29, 1997

PURPOSE: To compute the advection of a quantity from the wind field on a grid by using grids of u- and v- winds and a grid of the requested quantity Q to be advected. The advection can be computed at any level for either a Northern Hemispheric polar stereographic or Northern Hemispheric Lambert map projection. ADVCTW is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The actual advection computation is done by ADVCTW1, which uses the usual 2-gridlength finite differencing technique in each direction, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed advection field is used from the adjacent 2 rows (and columns). Given the advecting winds in m/sec, the advection is returned as Q*sec^{-1}.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

GFETCH, MAPLAT, ADVCTW1

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system.

ADVCTW1

COMPUTES ADVECTION FOR ADVCTW

Christopher A. Fiebrich
August 29, 1997

PURPOSE: To compute the advection of a quantity on a grid from the u- and v-winds. ADVCTW1 is called by ADVCTW, but could be used for other purposes. However, the sine of the latitude and the map factor have to be furnished to ADVCTW1. The computation is done with the usual 2-gridlength finite differencing technique in each direction, except for the outside row and column on each side. For these rows (and columns), linear extrapolation of the computed advection field is used from the adjacent 2 rows (and columns). Given the advecting winds in m/sec, the advection is returned as Q*sec^{-1}.

RESTRICTIONS:

The grid must be at least 4 x 4 in size.

NONSYSTEM ROUTINES USED:

EXTRAP

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system.

CORDP

ESTIMATES DEW POINTS DURING EXTREMELY COLD CONDITIONS OR WHEN RELATIVE HUMIDITY IS REPORTED

Mark Shirey
Kevin L. Carroll
July 1, 2000

PURPOSE: To correct for missing dew points during extremely cold conditions or to obtain the dew point from a report of temperature and relative humidity. When the air temperature is -30 degrees F or less, dew points are often not reported in the hourly reports. To avoid having a biased developmental sample, this subroutine sets the dew point to the air temperature when the dew point is missing in the hourly report, and the air temperature is -30 degrees F or less.

For many of the non-traditional reporting networks (mesonets), the hourly reports contain temperature and relative humidity observations. Under these conditions, CORDP estimates the dew point from the temperature and relative humidity reports.

CORDP is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID for the designation of this estimated dew point value is:

```
703102000 000000000 000000000 000
```

RESTRICTIONS:

The call sequence is appropriate for U201 and other "gridded" programs. As with other U201 subroutines, if no data are available for the first processing date, the subroutine will not work.

COMMENTS:

For more information on the hourly archive, see TDL Office Note 00-01.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

DPTDPR

COMPUTES DEW POINT DEPRESSION

Wei Yan
December 1, 1999

PURPOSE: To compute the dew point depression on a grid from grids of temperature and dew point at any level. This field is defined as the difference between temperature and dew point at a specific level. DPTDPR is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The dew point depression is returned in units of degrees Kelvin. The MOS-2000 ID's (CCCFFF) processed are:

003 170	Dew point depression (isobaric surface)
003 171	Dew point depression (constant height surface)
003 176	Dew point depression (constant sigma surface)

RESTRICTIONS:

To compute the dew point depression, the temperature and the dew point must be available for the requested period, and both must be in degrees Kelvin.

NONSYSTEM ROUTINES USED:

GFETCH, DEWPT, PRSID1

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system.

DIR2UV

COMPUTES U- AND V-WIND COMPONENTS FROM WIND DIRECTION AND SPEED

Mark A. Shirey
December 1, 1998

PURPOSE: To compute the u- and v-wind components from station observations or MOS forecasts by using the observed or forecast wind direction and speed. If the observed wind direction is variable, both wind components are set to missing. Likewise, if either the wind direction or speed is missing, both components are set to missing. Subroutine DIR2UV is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The components returned have a speed in knots. The MOS-2000 ID's (CCCFFF) processed by this subroutine are:

```
704010 = u- component of hourly observational wind
704110 = v- component of hourly observational wind
204010 = u- component of MOS forecast
204110 = v- component of MOS forecast
```

RESTRICTIONS:

Both the wind direction and speed must be available and not missing (and for wind direction, not variable) in order to compute the u- and v-wind components.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

DEWPCORR

PROVIDES BIAS-CORRECTED DEW POINT FORECAST FROM MODEL OUTPUT

Kevin L. Carroll
February 1, 2000

PURPOSE: TO compute a bias corrected field for dew point temperature. The bias correction is based on a model run made RR hours previously and valid at the initial time of the current model run. The bias in the model forecast of dew point is estimated by the difference of the model forecast and the initial condition valid at the same time. The bias is then added back to the current model forecast of the dew point to produce a field of bias-corrected dew points. The bias correction is based on only one previous run of the model.

DEWPCORR is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID for the designation of the bias corrected dew point value is:

003190BDD - bias-corrected dew point on an isobaric surface
003191BDD - bias-corrected dew point on a constant height surface

RESTRICTIONS:

The value of RR in the MOS-2000 identifier must be such that a previous model run is available at the time specified by the date list in the U201 control file as modified by the RR. Like most U201 subroutines, if the bias correction can not be computed for the first date in the date list, then the bias correction field will be missing for all dates in the list.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, DEWPT

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

FRZLVL

COMPUTES FREEZING LEVEL PRESSURE AT GRIDPOINTS

Benjamin Sfanos
October 1, 1998

PURPOSE: To compute freezing level in mb (hPa) on a grid from temperature fields at the set of isobaric surfaces available in the model archive. FRZLVL is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. Different ID's are used to call FRZLVL because of the different isobaric levels saved in each model archive. In other words, FRZLVL and the ID structure were designed to access different isobaric levels available in an archive. This subroutine was modified in February 2004 to change the ID's processed within the code because of a previous error. The MOS-2000 ID's (CCCFFF) processed are:

```
002 046 - Freezing level (mb) - Old AVN/MRF archive (6 levels)
002 047 - Freezing level (mb) - 32-km Eta archive (11 levels)
002 048 - Freezing level (mb) - GFS archive (12 levels)
002 049 - Freezing level (mb) - NGM archive (9 levels)
```

RESTRICTIONS:

The ID used must be compatible with the isobaric temperatures saved in the model archive. For example, if the ID of 002046 is used, then temperatures at all 6 isobaric levels specified in FRZLVL must be available in the model archive.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system.

LINDEX

COMPUTES LIFTED INDEX

Kathryn K. Hughes
May 26, 2000

PURPOSE: To compute the lifted index on an isobaric surface. The lifted index is a predictor of latent instability used to forecast thunderstorms. It is defined as the difference in temperature for a parcel lifted adiabatically from a pressure layer (usually somewhat close to the surface), to 500 mb, This 500-mb temperature is compared to the temperature of the environment. It is computed on a grid from grid point values of temperature, the relative humidity, the pressure and temperature of the lifted condensation level, and the temperature at 500 mb. Functions located with the subroutine calculate the equivalent potential temperature for the grid points where the pressure of the lifted condensation level is below 500 mb. The pressure level from which the parcel is lifted is determined by the TDL ID (see below). LINDEX is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The lifted index is the difference between two temperatures so the value is returned in Kelvin (or degrees Celsius). The MOS-2000 ID (CCCFFF) processed is:

007 020 = Lifted Index (Kelvin)

According to a National Weather Service guide entitled "A Fingertip Guide to Key Upper Air Index Values Used in Evaluating Severe Weather and Flash Flood Potential" written by Luis Giordano, NWS Pittsburgh, PA, 1994, a lifted index below -2 indicates a moderate potential for severe weather, while a value below -6 indicates a high potential for severe weather.

RESTRICTIONS:

This predictor is only valid when calculated from a standard level of pressure archived for the given model. The user supplies the starting pressure level of the parcel of air when requesting the LINDEX predictor by indicating the pressure level in the second ID word (UUUU). Pressure levels near the surface (i.e. 1000 mb, 950 mb, or 925 mb) give results similar to the Best Lifted Index available from the numeric models.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, LCL

INTERNAL FUNCTIONS USED:

FSVP, FSMR, FMIXR, FTHETAE

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

LMMAPS

COMPUTES SINE OF LATITUDE AND MAP FACTOR FOR LAMBERT CONFORMAL GRID

Harry R. Glahn
July 1, 2000

PURPOSE: To compute sine of the latitude and map factor for a set of stations residing on a lambert conformal grid. The I and J coordinates of the stations relative to the grid are known, as is the grid orientation and resolution.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

LMIJLL

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system.

MAXMIN

COMPUTES DAYTIME MAXIMUM AND NIGHTTIME MINIMUM TEMPERATURES

Mitchell Weiss
December 8, 2003

PURPOSE: To compute either a daytime maximum temperature (daytime max) or nighttime minimum temperature (nighttime min) estimate from station observations. This subroutine is called by subroutines OBSDMAXT and OBSNMINT. The definition of daytime max and nighttime min and how these definitions are applied to different time zones can be found in the writeups of these two respective subroutines. Subroutine MAXMIN is called to generate the daytime max and nighttime min estimates for dates from December 1, 1996 to the present. Estimates of the max and min are generated by using hourly temperatures and 6-h maximum or minimum temperature observations. The cccfff of the identifiers processed by this subroutine are the following:

702 001	- daytime max (degrees F)
702 011	- nighttime min (degrees F)

RESTRICTIONS:

For each station, the missing value (9999.) is assigned if an estimate of the daytime max and nighttime min temperature can not be made due to excessive missing data. For the first date to be processed, a physical record for all required input variables must be present on the input data set. If all physical records are not present (for example, one hour of temperature data is not available), then the missing value (9999.) is assigned to every station for that first date and all subsequent dates.

Note that the first date to be used can be no earlier than 1996120106. For dates prior to December 1, 1996, 0600 UTC, subroutine MAXMINB is used to estimate the max and min.

For other restrictions, the user is referred to the documentation for subroutines OBSDMAXT and OBSDMINT.

NONSYSTEM ROUTINES USED:

CKFRMSG, CKABCD, RMXMN_FILL, MAXMIN_PART

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

MAXMINB

COMPUTES DAYTIME MAXIMUM AND NIGHTTIME MINIMUM TEMPERATURES

Mitchell Weiss
December 8, 2003

PURPOSE: To compute either a daytime maximum temperature (daytime max) or nighttime minimum temperature (nighttime min) estimate from station observations. This subroutine is called by subroutines OBSDMAXT and OBSNMINT. The definition of daytime max and nighttime min and how these definitions are applied to different time zones can be found in the writeups of these two respective subroutines. Subroutine MAXMINB is called to generate the daytime max and nighttime min estimates for dates prior to December 1, 1996. Estimates of the max and min are generated by using hourly temperatures, 12-h maximum or minimum temperature observations, and 24-h maximum or minimum temperature observations. The cccfff of the identifiers processed by this subroutine are the following:

702 001 - daytime max (degrees F)
702 011 - nighttime min (degrees F)

RESTRICTIONS:

For each station, the missing value (9999.) is assigned if an estimate of the daytime max and nighttime min temperature can not be made due to excessive missing data. For the first date to be processed, a physical record for all required input variables must be present on the input data set. If all physical records are not present (for example, one hour of temperature data is not available), then the missing value (9999.) is assigned to every station for that first date and all subsequent dates.

Note that all dates to be processed within a U201 run that calls MAXMINB must be prior to December 1, 1996, and that dates being processed can not overlap the date of December 1, 1996, 0600 UTC, For a date of, or subsequent to, December 1, 1996, 0600 UTC, subroutine MAXMIN is used to estimate the max and min and separate runs of U201 must be made.

For other restrictions, the user is referred to the documentation for subroutines OBSDMAXT and OBSDMINT.

NONSYSTEM ROUTINES USED:

CKABC, CKPRIOD

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

MCMAPF

COMPUTES SINE OF LATITUDE AND MAP FACTOR FOR MERCATOR MAP

Harry R. Glahn
June 1, 2002

PURPOSE: To compute sine of the latitude and map factor for a grid for the mercator map projection. MCMAPF is called by MAPLAT.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

MCIJLL

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system.

MCMAPS

COMPUTES SINE OF LATITUDE AND MAP FACTOR FOR MERCATOR GRID

Harry R. Glahn
July 1, 2002

<u>PURPOSE</u>: To compute sine of the latitude and map factor for a set of stations residing on a mercator grid. The I and J coordinates of the stations relative to the grid are known, as is the grid orientation and resolution.

<u>RESTRICTIONS</u>:

None.

<u>NONSYSTEM ROUTINES USED</u>:

MCIJLL

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: With U201 in the u201lib file system.

MDIV

COMPUTES MOISTURE DIVERGENCE

C. A. Fiebrich
Wei Yan (rev.)
November 1, 1998

PURPOSE: To compute moisture divergence on a grid by using grids of u- and v-winds and specific humidity at any level for a Northern Hemispheric polar stereographic, Northern Hemispheric Lambert conformal, or mercator map projection. MDIV is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The specific humidity is obtained by a call to SPECHUM. The divergence term of moisture divergence ($q*(\partial u/\partial x+\partial v/\partial y)$) is computed by calling DIVW. The returned quantity (divergence from the winds) is then multiplied by the specific humidity. The advection term of moisture divergence ($u*\partial q/\partial x+v*\partial q/\partial y$) is computed by a call to ADVCTW. Given the winds in m/sec and the specific humidity in kg/kg, the moisture divergence is returned as $[kg/kg]s^{-1}*10^{8}$. The MOS-2000 ID's (CCCFFF) processed are:

003 500 = Moisture divergence on an isobaric surface;
003 501 = Moisture divergence on a constant height surface;
003 506 = Moisture divergence on a sigma surface.

RESTRICTIONS:

The grid must be at least 4 x 4 in size. At this time, if moisture divergence is needed for a constant height surface, the computations can be done only at the 10-m level.

NONSYSTEM ROUTINES USED:

GFETCH, SPECHUM, PRSID1, ADVCTW

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system.

MEANRH

COMPUTES MEAN RELATIVE HUMIDITY BETWEEN TWO ISOBARIC LEVELS

James C. Su
January 1, 1999

<u>PURPOSE</u>: To compute the mean relative humidity for a user-specified layer bounded by two isobaric levels. The relative humidity is calculated in terms of percent. Multiple isobaric levels between the upper and lower bounds are used in the calculation, provided that the appropriate grids of relative humidity and specific humidity are available. A piece-wise trapezoidal rule (applicable to a situation when the distance between values along the abscissa, that is, the pressure levels, are variable) is used to integrate the equation for obtaining the mean value from the specific humidity and the saturated specific humidity. Since the available isobaric levels vary among model archives, the identifiers used to call MEANRH vary and specify which isobaric layers are to be included in the calculation. Depending on the fields available in the NWP model archive, different identifiers may be used to calculate the mean relative humidity between two levels. Subroutine MEANRH is designed for U201, and the call sequence is tailored for that use. Additional explanation is provided in the documentation of U201. The MOS-2000 ID's (CCCFFF) processed in this subroutine are:

003 040 =	Mean relative humidity, with isobaric levels of 1000, 950, 900, 850, 800, 750, 700, 500, and 300 mb available;
003 041 =	Mean relative humidity, with isobaric levels of 1000, 925, 850, 700, 500, and 300 mb available;
003 042 =	Mean relative humidity, with isobaric levels of 1000, 950, 900, 850, 800, 750, 700, 600, 500, 400, and 300 mb available;
003 043 =	Mean relative humidity, with isobaric levels of 1000, 975, 950, 925, 900, 850, 800, 750, 700, 600, 500, and 300 mb available.

The second ID word (VLLLLUUUU) must be set to 0LLLLUUUU, where LLLL and UUUU are the lower and upper bounds, respectively, of the layer. Note that both values are given in mb and that "lower" and "upper" in this context denote the position of the bounds relative to the earth’s surface.

<u>RESTRICTIONS</u>:

The values of relative and specific humidity must be available on all levels between the lower and upper bounds specified by the identifier (see above). Note that a value of LLLL=UUUU is not allowed. Note, too, that different ID’s can be used to obtain the mean relative humidity from the same model archive, provided that the requisite levels are available.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH, SPECHUM, PRSID1

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: With U201 in the u201lib file system.

MODELMXMN

COMPUTES 12-H MAX OR MIN TEMPERATURE FROM 6-H MODEL VALUES

Rebecca L. Cosgrove
July 1, 2002

PURPOSE: To compute the 12-h maximum or minimum temperature by using the model forecasts of the two 6-h max or min values comprising the 12-h period of interest. MODELMXMN is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The maximum or minimum temperatures are returned in units of degrees Kelvin. The MOS-2000 ID's (CCCFFF) processed are:

002 051 = Maximum temperature (Kelvin)
002 061 = Minimum temperature (Kelvin).

RESTRICTIONS:

The 12-h max/min can be only computed for forecast projections that are a multiple of 6. Note, too, that the 6-h max/min values must be available in the model archives in order for this subroutine to produce the 12-h values.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1

LANGUAGE: FORTRAN 77.

LOCATION: With U201 in the u201lib file system.

OBSCIGHT

COMPUTES CEILING HEIGHT

Mitchell Weiss
October 16, 1998

PURPOSE: To compute ceiling height from station observations by using the cloud height and amount from six possible levels. The ceiling height is estimated from the cloud height of the lowest layer of reported broken or overcast cloud cover (5/8 sky coverage or greater) or from the vertical visibility when a total obscuration of the sky is reported. If cloud coverage fails to reach 5/8 coverage or a partially obscured sky condition is reported, the sky coverage is not considered a ceiling, and, therefore, the ceiling height is considered unlimited. Clouds reported above the level of a partially obscured sky can still constitute a ceiling height by using the rules described above. Ceiling height is estimated if both cloud height and amount are available for each reported level. If missing values for either variable occur at a specific level, then the determination of the ceiling height is stopped at the previous level. OBSCIGHT is a routine designed for U201, and the call sequence is tailored for that use. Additional explanation is provided in the documentation of U201. The ceiling height is returned in hundreds of feet. The value 888. denotes unlimited ceiling. The MOS-2000 identifier for ceiling height is:

708000000 000000000 000000000 000

RESTRICTIONS:

For each station, both cloud height and amount must be available for the first level, or the missing value of 9999. is assigned to ceiling height. Once cloud amount and height have been observed for the first level, the algorithm assumes that missing values occurring for either variable at higher levels indicate that the observations of cloud amount and height have ended. The ceiling height is then determined from the cloud height and amount information of the previous level. If the actual record of either cloud height or amount is missing for any level, then the missing value of 9999. is returned for all stations for that particular hour. Ceiling height estimates from automated (ASOS) sites will be generated for clouds observed only up to 12,000 ft due to limitations of the ASOS cloud observing instrument.

For dates prior to December 1, 1996, 0000 UTC, the observation of ceiling height is obtained directly from the SAO reports in the hourly archives. From December 1, 1996, 0000 UTC to the present, the ceiling height is computed from METAR reports in the manner described above. As a result, a user who wishes to obtain ceiling height observations both before and after this date must run distinct U201's for the period before, and for the period after, this date.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

KINXRF

COMPUTES INTERACTIVE PREDICTOR K-INDEX * THUNDERSTORM RELATIVE FREQUENCY

Ronald Reap
October 8, 1998

PURPOSE: To compute the interactive predictor KF, defined by the product of the K stability index (K) and the thunderstorm relative frequency (F). The interactive predictor is computed from grids of the K index interpolated to stations and the relative frequencies of thunderstorms derived from lightning data at stations. The cccfff of the identifiers processed by this subroutine are the following:

007460	= KINDEX * previous 3 hr thunderstorm relative frequency
007465	= KINDEX * subsequent 3 hr thunderstorm relative frequency
007470	= KINDEX * previous 6 hr thunderstorm relative frequency
007475	= KINDEX * subsequent 6 hr thunderstorm relative frequency
007480	= KINDEX * previous 12 hr thunderstorm relative frequency
007485	= KINDEX * subsequent 12 hr thunderstorm relative frequency
007490	= KINDEX * previous 24 hr thunderstorm relative frequency

KINXRF is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation.

RESTRICTIONS:

Since the KF predictor relies on station-based relative frequency data and on the K-index interpolated to stations, the KF predictor can not be used as a grid-binary variable. In other words, the KF predictor is a station-oriented variable and **not** a grid-point field.

NONSYSTEM ROUTINES USED:

KINDEX, INTRPB, PRSID1, CONST

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

OBSDMAXT

COMPUTES DAYTIME MAXIMUM TEMPERATURE

Mitchell Weiss
January 21, 1999

PURPOSE: To compute daytime maximum temperature for station (vector) data. Daytime maximum temperature (daytime max) is defined as the highest observed temperature occurring for the period 7 a.m. through 7 p.m. local standard time. Estimates are determined from stations located over the following seven time zones: the Atlantic time zone, the four continental U.S. time zones, and the two Alaskan time zones, of which the westernmost includes Hawaii. Hourly temperatures for the 24 hours preceding each process date are used in the calculation. For process dates up through November 30, 1996, calculations are generated by using hourly temperatures and both 12- and 24-h maximum temperatures (old-data processing). For process dates falling on December 1, 1996, and later, estimates are generated from hourly temperature and 6-h maximum temperatures (new-data processing). Separate algorithms are used to estimate daytime max for each combination of input data. To ensure adequate temporal coverage of temperature for all time zones, the daytime max estimates are stored in the 06Z record of the day following the date for which the estimate is valid. For example, the process date 1996120506 will contain daytime max estimates for December 4, 1996. Special consideration is given for daytime max estimates for November 30, 1996, which require input data from December 1, 1996 (see restrictions below). OBSDMAXT is an almost exclusive routine of U201 and the call sequence is tailored for that use. Additional information about U201 can be found in the MOS-2000 users guide. The daytime max is returned in degrees Fahrenheit, with the CCCFFF portion of the MOS-2000 ID designated as 702001. To correctly generate the daytime max within U201, a special input variable record must be included. The special record ID along with the daytime max ID are shown as follows:

```
799000000 000000000 024000000 000 = Special ID
702001000 000000000 000000000 000 = Daytime max temperature
```

The special record (CCC = 799) is needed to help properly initialize U201's mass storage system. The RR value of 24 in the third word will enable U201 to save for access the 24 hours of data preceding the first process date. A record with this special variable ID **will not** be written to the output data file. Other requirements and restrictions involving daytime max within U201 are listed in the next section.

RESTRICTIONS:

For each station, the missing value 9999. is assigned if an estimate of the daytime max can not be made due to excessive missing data. When used within U201, the following conditions must also be followed:

1. Daytime max is only generated for process dates containing the hour of 06Z. Remember, the process (or stored) date will be one day later than the actual date of the estimate (see example above).

2. The starting(first) process date does not have be 06Z. However, the hour of the first process date must either be 00Z, 06Z, 12Z, or 18Z; otherwise, daytime max estimates for all subsequent process dates will be set to missing.

3. For the first process date, all required physical records **must** be present. For example, for a start date of 1996120300, all hourly temperature records and 6-h max temperature records must be present for the period 1996120200 - 1996120300. If all physical records are not present, the returned values of daytime max for all subsequent process dates will either be missing or will be incorrect.

4. INCCYL must either be 1,2,3,6,12, or 24. When processing daytime max and nighttime minimum temperatures (nighttime min) together, INCCYL must be 12 or less.

5. To properly process an entire calendar month (assuming that both nighttime min and daytime max temperatures are being generated), input data must be available from the last day of the previous month and the first day of the subsequent month.

6. The starting and ending process date **can not** overlap 1996120106. Periods covering old-data processing (up through 1996120106) or new-data processing (after 1996120106) **must** be done in separate runs.

7. **EXCEPTION:** If the first processing date is 1996120100 (assuming December 1996 processing), new-data processing will be assumed. This special condition (exclusive to daytime max) allows processing of December 1996 in conjunction with other parameters if so desired. Naturally, the value of the daytime max for 1996120106 should be ignored.

NONSYSTEM ROUTINES USED:

GFETCH, MAXMIN, MAXMINB

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

OBSDPTD

COMPUTES DEW POINT DEPRESSION FROM OBSERVATIONAL DATA

Mitchell Weiss
December 1, 2000

<u>PURPOSE</u>: To compute the observed dew point depression by subtracting the dew point temperature from the air temperature. When the air temperature is -30° F or less, dew points are often not reported in the hourly reports. For consistency with subroutine CORDP, OBSDPTD sets the dew point to the air temperature when the dew point is missing in the hourly report and the air temperature is -30° F or less. Thus, under these extremely cold conditions, the dew point depression is effectively set to 0. OBSDPTD is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 CCCFFF for the designation of the dew point depression is:

703 101 = Dew Point Depression

<u>RESTRICTIONS</u>:

As with other U201 subroutines, if data records (temperature or dew point) are missing for the first process date, then the subroutine will be unable to generate the dew point depressions for any of the requested dates.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH, MAXMIN, MAXMINB

<u>LANGUAGE</u>: FORTRAN 90

<u>LOCATION</u>: With U201 in the u201lib file system.

OBSMRCLD

COMPUTES MEAN TOTAL SKY COVER FOR A 12-H PERIOD

Kevin L. Carroll
June 1, 2000

<u>PURPOSE</u>: To compute the observed mean total sky cover over a 12-h time period ending at 0000 or 1200 UTC. For each of the 13 observations comprising the 12-h period, the total sky cover is estimated by calling subroutine OBSTCLD which uses the reported sky cover and the coincident satellite-based estimated sky cover to create a complemented observation. Subsequently, each of the 13 sky observations is multiplied by a weighting factor, ranging from 0.0 to 1.0, according to the following:

```
CLR = 0.0
FEW = 0.15
SCT = 0.38
BKN = 0.69
OVC = 1.00
OBSCURED = 1.00
```

For the final calculation of the mean over the 12-h period, the weighted value of the first and last observations of the period are multiplied by 0.5 and then summed with the weighted values of each of the other observations within the period. The sum of all these values is divided by the total number of observations less one. Thus, for a 12-h period containing observations for all hours, the denominator used in calculating the mean is 12. The final value represents the mean total sky cover amount over the 12-h period and ranges from 0.00 (clear at all hours) to 1.00 (overcast at all hours). OBSMRCLD is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. To correctly generate the mean sky cover within U201, a special input variable record must be included. The special record ID along with the mean total sky cover ID are shown as follows:

```
799000000 000000000 012000000 000 = Special Id
708315000 000000000 000000000 000 = Total Sky Cover
```

The special record (CCC = 799) is needed to help properly initialize U201's mass storage system. The RR value of 12 in the third word will enable U201 to save and access the 12 hours of data preceding the first process date. A record with this special variable ID **<u>will not</u>** be written to the output data file. Other restrictions involving the use of this subroutine within U201 are listed in the next section.

<u>RESTRICTIONS</u>:

As with other U201 subroutines, if data records (observed sky cover or the satellite estimate of cloud cover at any of the required 13 hours) are missing for the first process date, then the subroutine will be unable to use all of the appropriate hours for any of the requested dates. The user needs to look carefully at the U201 output messages.

<u>NONSYSTEM ROUTINES USED</u>:

OBSTCLD, UPDAT

<u>LANGUAGE</u>: FORTRAN 90

<u>LOCATION</u>: With U201 in the u201lib file system.

OBSMRPTYPE

COMPUTES PRECIPITATION TYPE FOR A 12-H PERIOD

Rebecca Cosgrove
October 6, 2000

PURPOSE: To classify the precipitation occurring during a 12-h period (ending at the valid time) into one of four mutually exclusive medium-range precipitation types based on the combination of present weather reported over those 12 hours (see Comments). If the present weather report is missing or if no precipitation is reported, the precipitation type value is set to 9999. OBSMRPTYPE is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. To correctly generate the prevailing precipitation type within U201, a special input variable record must be included. The special record ID along with the precipitation type ID are shown as follows:

```
799000000 000000000 012000000 000 = Special ID
708511000 000000000 000000000 000 = Precipitation Type
```

The special record (CCC = 799) is needed to help properly initialize U201's mass storage system. The RR value of 012 in the third word will enable U201 to save for access the 12 hours of data preceding the first process date. A record with this special variable ID **will not** be written to the output data file. Other restrictions involving the use of this subroutine within U201 are listed in the next section.

RESTRICTIONS:

Due to the conversion from SAO observations to METAR observations in 1996, the present weather codes differ in the hourly data archive before and after December 1, 1996. While this subroutine is equipped to handle both numbering conventions, the user must not try to process dates that span this December 1, 1996, cutoff due to the configuration of the mass storage system. Therefore, to correctly process data across this threshold, the user must make one run containing dates up to and including November 30, 1996, and then start a new run for December 1, 1996, and beyond. In addition, because of this restriction, and because the 12-h period ends with the 12 hours preceding the valid date, it is not possible to process any valid dates from December 1, 1996, at 00Z through December 1, 1996, at 11Z.

As with other U201 subroutines, if no data are available for the first processing date/time, the subroutine will not compute the precipitation type categories. The user needs to look carefully at the U201 output messages to verify that the precipitation type value is being estimated properly.

<u>COMMENTS</u>:

The 12-h medium-range precipitation types are categorized as follows:

1 = freezing precipitation or anything in combination with freezing precipitation
2 = pure snow
3 = snow mixed with rain
4 = pure rain

For the mixed categories of 1 and 3, we do not distinguish between mixed in time and mixed at a specific hour. In other words, a precipitation type of 1 would be assigned to both a 12-h period with RAPL at one hour and a 12-h period with RA at one hour and PL at a different hour.

In order to calculate the precipitation type observation over a 12-h period, the station in question must have reported at least seven observations out of a possible 13, and at least three of these must be occurrences of precipitation.

In the data available after December 1, 1996, the precipitation associated with thunderstorms is not distinguishable as rain or snow. Therefore, if a thunderstorm is reported along with any other precipitation, the precipitation type is coded according to that other precipitation group. If no other precipitation is reported, the thunderstorm is classified as rain.

Automated stations can report a weather descriptor UP for Unknown Precipitation. In this subroutine, we throw out reports of UP because the precipitation type is not known for these reports. We also throw out reports of hail and squalls.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH, UPDAT

<u>LANGUAGE</u>: FORTRAN 90

<u>LOCATION</u>: With U201 in the u201lib file system.

OBSMRWSP

COMPUTES OBSERVED MAXIMUM WIND SPEED FOR A 12-H PERIOD

Mary C. Erickson
January 1, 2002

PURPOSE: To compute the observed maximum wind speed over a 12-h time period ending at 0000 or 1200 UTC. For each of the 13 observations comprising the 12-h period, the wind speed observation is read and examined. The maximum value of the wind speed is then extracted from the available hourly values. A minimum of 7 observations is required in order for the maximum wind speed to be calculated. OBSMRWSP is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. To correctly generate within U201 the maximum wind speed for the 12-h period, a special input variable record must be included. The special record ID along with the maximum wind speed ID are shown as follows:

```
799000000 000000000 012000000 000 = Special ID
704420000 000000000 000000000 000 = Maximum Wind Speed
```

The special record (CCC = 799) is needed to help properly initialize U201's mass storage system. The RR value of 12 in the third word will enable U201 to save for access the 12 hours of data preceding the first process date. A record with this special variable ID **will not** be written to the output data file. Other restrictions involving the use of this subroutine within U201 are listed in the next section.

RESTRICTIONS:

As with other U201 subroutines, if data records (observed wind speed at any of the required 13 hours) are missing for the first process date, then the subroutine will be unable to use all of the appropriate hours for any of the requested dates. The user needs to look carefully at the U201 output messages.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

OBSNMINT

COMPUTES NIGHTTIME MINIMUM TEMPERATURE

Mitchell Weiss
January 21, 1999

PURPOSE: To compute nighttime minimum temperature for station (vector) data. Nighttime minimum temperature (nighttime min) is defined as the lowest observed temperature occurring for the period 7 p.m. through 8 a.m. local standard time. Estimates are determined from stations located over the following seven time zones: the Atlantic time zone, the four continental U.S. time zones, and the two Alaskan time zones, of which the westernmost includes Hawaii. Hourly temperatures for the 24 hours preceding each process date are used in the calculation. For process dates up through November 30, 1996, calculations are generated by using hourly temperatures and both 12- and 24-h minimum temperatures (old-data processing). For process dates falling on December 1, 1996, and later, estimates are generated from hourly temperature and 6-h minimum temperatures (new-data processing). Separate algorithms are used to estimate nighttime min for each combination of input data. To ensure adequate temporal coverage of temperature for all time zones, the nighttime min estimates are stored in the 18Z record of the day on which the ending hour of the nighttime period occurs. For example, the process date 1996120418 will contain nighttime min estimates for December 4, 1996. Special consideration is given for nighttime min estimates for November 30, 1996, which require input data from December 1, 1996 (see restrictions below). OBSNMINT is an almost exclusive routine of U201 and the call sequence is tailored for that use. Additional information about U201 can be found in the MOS-2000 users guide. The nighttime min is returned in degrees Fahrenheit, with the CCCFFF portion of the MOS-2000 ID designated as 702011. To correctly generate the nighttime min within U201, a special input variable record must be included. The special record ID along with the daytime max ID are shown as follows:

```
799000000 000000000 024000000 000 = Special Id
702011000 000000000 000000000 000 = Nighttime Min Temperature
```

The special record (CCC = 799) is needed to help properly initialize U201's mass storage system. The RR value of 24 in the third word will enable U201 to save for access the 24 hours of data preceding the first process date. A record with this special variable ID **will not** be written to the output data file. Other requirements and restrictions involving nighttime min within U201 are listed in the next section.

RESTRICTIONS:

For each station, the missing value 9999. is assigned if an estimate of the nighttime min can not be made due to excessive missing data. When used within U201, the following conditions must also be followed:

1. Nighttime min is only generated for process dates containing the hour of 18Z. Remember, the process (or stored) date will be the date of the hour ending the estimate (see example above).

2. The starting(first) process date does not have be 18Z. However, the hour of the first process date must either be 00Z, 06Z, 12Z, or 18Z; otherwise, nighttime min estimates for all subsequent process dates will be set to missing.

3. For the first process date, all required physical records **must** be present. For example, for a start date of 1996120300, all hourly temperature records and 6-h min temperature records must be present for the period 1996120200 - 1996120300. If all physical records are not present, the returned values of nighttime min for all subsequent process dates will either be missing or will be incorrect.

4. INCCYL must either be 1,2,3,6,12, or 24. When processing daytime max and nighttime min temperatures together, INCCYL must be 12 or less.

5. To properly process an entire calendar month (assuming that both nighttime min and daytime max temperatures are being generated), input data must be available from the last day of the previous month and the first day of the subsequent month.

6. The starting and ending process date **can not** overlap 1996113018. Periods covering old-data processing (up through 1996113018) or new-data processing (after 1996113018) **must** be done in separate runs.

7. **EXCEPTION:** If the first processing date is 1996120100 (assuming December 1996 processing), new-data processing will be assumed. This special condition allows processing of December 1996 in conjunction with other parameters if so desired.

NONSYSTEM ROUTINES USED:

GFETCH, MAXMIN, MAXMINB

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

OBSOBVIS

ASSIGNS PRESENT WEATHER REPORTS TO CATEGORIES OF NON-PRECIPITATING OBSTRUCTIONS TO VISION

Benjamin Sfanos
July 1, 2000

PURPOSE: To classify station (vector) present weather reports into various categories according to whether or not a non-precipitating obstruction to vision occurred at the station at that hour. In other words, OBSOBVIS examines the present weather groups to determine whether a reported reduction in visibility is due to a phenomenon other than precipitation. OBSOBVIS is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID (CCCFFF) processed is:

708 251 = Non-precipitating Obstructions to Vision

Categories and the associated numbers produced by this subroutine are as follows:

1 = None of the following
2 = Haze, smoke, or dust
3 = Mist (fog with visibility $\geq$ 5/8 mi)
4 = Fog (fog with visibility < 5/8 mi)
5 = Blowing phenomena (snow, sand, dust)

RESTRICTIONS:

Due to the conversion from SAO observations to METAR observations in 1996, the method by which the obstructions to vision were saved differ in the hourly data archive before and after December 1, 1996. While this subroutine is equipped to handle both conventions, the user must not try to process in a single U201 run dates that span this December 1, 1996, cutoff due to the configuration of the internal MOS-2000 mass storage system. Therefore, to correctly process data across this threshold, the user must do one run containing dates up to and including November 30, 1996, and then start a new run for December 1, 1996, and beyond.

As with other U201 subroutines, if no data are available for the first processing date, the subroutine will not work properly. The user should carefully check the messages in the U201 ASCII output files.

COMMENTS:

Note that in assessing the obstruction to vision all three present weather groups available in the hourly archive after December 1, 1996, are considered. Note, also, that a precipitation event, though causing reduced visibility, will be classified as a non-event (category 1) unless mist, fog, a blowing phenomenon, or other non-precipitating obstruction to vision is also reported.

For automated stations that do not have a present weather sensor, the value returned from OBSOBVIS will be set to 9999.

For more information on the present weather codes and the hourly archive see TDL Office Note 00-01.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

OBSTWET

COMPUTES THE AVERAGE OF THE OBSERVED TEMPERATURE AND DEW POINT

Rebecca Cosgrove
May 1, 2000

PURPOSE: To produce an approximation for the observed wet bulb temperature by computing the average of the observed temperature and observed dew point. OBSTWET is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID (CCCFFF) is:

703 103 = Average of Observed Temperature and Dew Point

RESTRICTIONS:

As with other U201 subroutines, if either of the data records (temperature or dew point) is unavailable for the first processing date, the subroutine will not work properly. The user should carefully check the U201 output messages.

COMMENTS:

If either the temperature or dew point observation record is missing, the observed wet bulb temperature will be set to missing. The corrected observed dew point is used rather than the raw observed dew point to safeguard against missing dew points during extreme cold events (see documentation for CORDP).

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, CORDP

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

PBLMIX

CALCULATES THE COMBINED INFLUENCE OF THE TEMPERATURE PROFILE,
AVERAGE WIND, AND RELATIVE HUMIDITY IN THE PLANETARY BOUNDARY LAYER

David Rudack
December 1, 2000

PURPOSE: To calculate the combined influence of the temperature profile, average wind, and relative humidity in the planetary boundary layer as defined by the numerical model. Blending these elements is useful in determining whether or not the environment is conducive for the development of radiation fog. The resultant value can then be used as a predictor in the development of visibility and obstruction to vision forecast equations. The predictor is defined as:

$$\text{Predictor} = (\text{TEMP}_{\text{925 mb}} * \text{RH}_{\text{mean PBL}}) / (\text{TEMP}_{\text{1000 mb}} * \text{WSPEED}_{\text{mean between 1000 mb \& 925 mb}})$$

In certain areas of the country (e.g., the western U.S.), where the elevation is considerably higher than anywhere else in the U.S., a predictor that uses variables from pressure levels other than those noted above may be more appropriate. In this instance, the predictor is defined as:

$$\text{Predictor} = (\text{TEMP}_{\text{700 mb}} * \text{RH}_{\text{mean between 850 mb \& 700 mb}}) / (\text{TEMP}_{\text{850 mb}} * \text{WSPEED}_{\text{mean between 850 mb \& 700 mb}})$$

PBLMIX is an exclusive routine for U201 and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The two MOS-2000 IDs (CCCFFF) processed:

007 500 = Predictor for lower terrain (e.g., eastern U.S.)
007 510 = Predictor for higher terrain (e.g., western U.S.)

RESTRICTIONS:

As with other U201 subroutines, if any of the data records needed in the calculation is unavailable for the first processing date, the subroutine will not work properly. The user should carefully check the U201 output messages.

COMMENTS:

This program was exclusively designed to retrieve data on a constant pressure surface. If working with a different type of surface is more desirable, PBLMIX must be modified.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, WSPEED, MEANRH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

PCPX6

COMPUTES OBSERVED PRECIPITATION AMOUNT

Mark Shirey
December 1, 1998

PURPOSE: To compute the observed precipitation amount over a specified period from 6-h reports. The amount will be calculated with the period ending on NDATE. If one of the 6-h periods is missing, then the total precipitation amount is set to missing. A trace is coded as .004. PCPX6 is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The precipitation amount is returned in units of inches, which are the units for the hourly observations. The MOS-2000 ID's (CCCFFF) processed are:

703 220 = 12-h precipitation total
703 230 = 18-h precipitation total
703 240 = 24-h precipitation total

RESTRICTIONS:

To compute the precipitation amount, all the 6-h precipitation reports needed for the total precipitation calculation must be available. Precipitation amounts can only be calculated when NDATE ends at 0000 UTC, 0600 UTC, 1200 UTC, or 1800 UTC. Because RR is used in this subroutine, a dummy ID record is required in the U201.CN file. This record must have IDPARS(1) set to 799, with RR equal to the time period in hours that the total precipitation amount will cover minus 6 hours. For example, the 24-precipitation total would have RR=18, with the variable id's in the U201.CN file (record 13 of the control file) equal to:

```
799000000 000000000 018000000
703240000 000000000 000000000
```

Special attention must be paid to the first processing date. Since this program looks back in time, data must be available before the first process date. For example, if the user is starting on December 2 at 0000 UTC, and wants the 24-h precipitation ending at that time, data must be available for each component 6-h period back to December 1 at 0600 UTC. If the 6-h precipitation amount data are missing from December 1 at 0600, 1200, or 1800 UTC or from December 2 at 0000 UTC, then the subroutine will not run properly. Please check the U201 output files and be sure that data are available at the appropriate time periods before the first process date, or the program will not produce the appropriate results.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

POTEMP

COMPUTES POTENTIAL TEMPERATURE

Wei Yan
December 9, 1999

PURPOSE: To compute the potential temperature on a grid from grids of temperature and pressure at any level. The grid containing the pressure values is internally created by using the TDL ID if the potential temperature is to be calculated on an isobaric surface. POTEMP is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for any additional explanation. The potential temperature is returned in units of degrees Kelvin. The MOS-2000 ID's (CCCFFF) processed are:

002 100 = Potential Temperature on an isobaric surface.
002 101 = Potential Temperature on a surface of constant height.
002 106 = Potential Temperature on a sigma surface.

RESTRICTIONS:

To compute the potential temperature, the temperature and the pressure data must be available for the requested period. POTEMP expects pressure data from the NGM to be in hectopascals (millibars) for computation on a constant height or sigma surface. All other model pressure data are expected to be in Pascals, and are converted to hectopascals within the POTEMP subroutine.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

PSMAPS

COMPUTES SINE OF LATITUDE AND MAP FACTOR FOR STATIONS PROJECTED ONTO A POLAR STEREOGRAPHIC MAP

Harry R. Glahn
July 1, 2002

<u>PURPOSE</u>: TO compute the sine of the latitude and the map factor for a set of NSTA points which are located on a polar stereographic grid projection. The X, Y coordinates of the points are in arrays XI( ) and YJ( ), respectively. PSMAPS is called by subroutines TSLCM and UPSLOP.

<u>RESTRICTIONS</u>:

None.

<u>NONSYSTEM ROUTINES USED</u>:

PSLLIJ

<u>LANGUAGE</u>: FORTRAN 90

<u>LOCATION</u>: With U201 in the u201lib file system.

QADV

COMPUTES ADVECTION OF SPECIFIC HUMIDITY

Wei Yan
February 1, 1998

<u>PURPOSE</u>: To compute specific humidity advection on a grid by using grids of u- and v-winds and specific humidity at any level for either a Northern Hemisphere polar stereographic or Northern Hemisphere Lambert map projection. QADV is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The specific humidity data are retrieved by a call to SPECHUM. The specific humidity advection, defined as $-(u*\partial q/\partial x+v*\partial q/\partial y)$, is then computed by a call to ADVCTW. Given the advecting winds in m/sec and the specific humidity in kg/kg, the specific humidity advection is returned as $[s^{-1}]*s^{-1}*10^8$. The MOS-2000 ID's (CCCFFF) processed are:

004 350 =	Specific humidity advection on an isobaric surface;
004 351 =	Specific humidity advection on a constant height surface;
004 356 =	Specific humidity advection on a sigma surface.

<u>RESTRICTIONS</u>:

The grid must be at least 4 x 4 in size. Note that this subroutine does not compute geostrophic specific humidity advection, but is designed only to compute specific humidity advection from the model winds.

<u>NONSYSTEM ROUTINES USED</u>:

ADVCTW, DIVW, PRSID1, SPECHUM

<u>LANGUAGE</u>: FORTRAN 90

<u>LOCATION</u>: With U201 in the u201lib file system.

RHVV

COMPUTES PRODUCT OF VERTICAL VELOCITY AND RELATIVE HUMIDITY

Ben Sfanos
October 1, 1998

PURPOSE: To compute the product of the vertical velocity and the relative humidity at any isobaric level. RHVV is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The vertical velocity and the relative humidity fields are obtained by a call to GFETCH which will read these variables from the direct model output. Given the vertical velocity in Pa/sec and the relative humidity in percent, the product is returned as %-Pa/sec. The MOS-2000 ID (CCCFFF) processed by this subroutine is:

007 420 = product on an isobaric surface specified by the second ID word.

RESTRICTIONS:

The grid must be at least 4 x 4 in size. Note that this subroutine does not compute relative humidity, but is designed only to read the vertical velocity and the relative humidity at the isobaric level specified by the second ID word. For this subroutine to work properly, both the vertical velocity and the relative humidity must be available.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

SAMEGR

COMPARES GRID NUMBER AND GRID EXTENT OF TWO GRIDS

Bob Glahn
July 1, 2003

<u>PURPOSE</u>: To compare the grid (NSLAB) number and the x and y extent of two gridded MOS-2000 records processed within the MOS-2000 system. SAMEGR is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation.

<u>NONSYSTEM ROUTINES USED</u>:

NONE

<u>LANGUAGE</u>: FORTRAN 90

<u>LOCATION</u>: With U201 in the u201lib file system.

SATLEVNUM

COMPUTES FRACTION OF SATURATED LEVELS IN VERTICAL SOUNDING

Mitchell Weiss
December 17, 2003

PURPOSE: To compute the fraction of saturated levels in a vertical sounding extracted from a model profile. This estimated fraction is computed on a grid, from grids of constant pressure for relative humidity and height, and the terrain height. All the input variables except terrain height are model variables. The terrain height is a constant and matches the grid resolution of the respective model. The fraction of saturated levels is estimated by determining the fraction of pressure levels above the surface where the relative humidity exceeds or matches the given relative humidity threshold. The relative humidity thresholds available to use are 85, 90, and 95 percent. The requested relative humidity threshold is designated by the FFF portion of the first word of the MOS-2000 ID. The MOS-2000 ID will also depend on the vertical profile requested, and the date and model requested. The list of assigned MOS-2000 ID's are shown below. For dates prior to October 1, 1998, AVN model estimates use a 5-level vertical profile and are therefore assigned their own set of MOS-2000 ID's. For dates following October 1, 1998, AVN(GFS) model estimates may use an 11-level vertical profile along with a different set of MOS-2000 ID's. For Eta model estimates, the 9-level vertical profile is standard for all time periods; therefore, the assigned MOS-2000 ID's are valid for all dates. The algorithm determines which pressure levels are saturated starting at the lowest pressure level, and proceeding upward. Pressure level relative humidity values either exceeding or matching the relative humidity threshold must occur at a height above the surface (terrain) elevation or they will be excluded. If the model relative humidity fails to reach the relative humidity threshold for all qualifying pressure levels, a value of zero is returned. SATLEVNUM is an almost exclusive routine of U201, therefore the sequence of calling arguments is compatible with this code. Additional information about U201 can be found in the MOS-2000 system documentation. SATLEVNUM estimates are returned as fractions ranging from 0.00 to 1.00. The MOS-2000 ID's (CCCFFF) processed are summarized below.

```
003 341 = AVN ( 5-level); RH threshold = 85
003 351 = AVN ( 5-level); RH threshold = 90
003 361 = AVN ( 5-level); RH threshold = 95
003 342 = Eta ( 9-level); RH threshold = 85
003 352 = Eta ( 9-level); RH threshold = 90
003 362 = Eta ( 9-level); RH threshold = 95
003 343 = GFS (11-level); RH threshold = 85
003 353 = GFS (11-level); RH threshold = 90
003 363 = GFS (11-level); RH threshold = 95
```

RESTRICTIONS:

Grids of relative humidity and height must be available for all pressure levels. The grid of the terrain height must also be available. SATLEVNUM will exclude pressure levels determined to be below the terrain height. The relative humidity thresholds are restricted to the values of 85, 90, and 95 percent.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH, CONST1

<u>LANGUAGE</u>: FORTRAN 90

<u>LOCATION</u>: With U201 in the u201lib file system.

SATLEVRH

COMPUTES HEIGHT OF SATURATION LEVEL IN VERTICAL SOUNDING

Mitchell Weiss
December 17, 2003

PURPOSE: To compute the height of the saturation level in a vertical sounding extracted from a model profile. This estimated height is computed on a grid, from grids of constant pressure for relative humidity and height, and the terrain height. All the input variables except terrain height are model variables. The terrain height is a constant and matches the grid resolution of the respective model. The saturation level height is estimated by determining the lowest height above the surface where the relative humidity exceeds or matches the given relative humidity threshold. The relative humidity thresholds available to use are 85, 90, and 95 percent. The requested relative humidity threshold is designated by the FFF portion of the first word of the MOS-2000 ID. The MOS-2000 ID will also depend on the vertical profile requested, and the date and model requested. The list of assigned MOS-2000 ID's are shown below. For dates prior to October 1, 1998, AVN model estimates use a 5-level vertical profile and are therefore assigned their own set of MOS-2000 ID's. For dates following October 1, 1998, AVN(GFS) model estimates may use an 11-level vertical profile along with a different set of MOS-2000 ID's. For Eta model estimates, the 9-level vertical profile is standard for all time periods; therefore, the assigned MOS-2000 ID's are valid for all dates. The algorithm estimates the saturation height starting at the lowest pressure level, and proceeds upward. Pressure level relative humidity values either exceeding or matching the relative humidity threshold must occur at a height above the surface (terrain) elevation or they will be skipped. If the model relative humidity fails to reach the relative humidity threshold at any available pressure level, a value of 35000 feet is returned. SATLEVRH is an almost exclusive routine of U201, therefore the sequence of calling arguments is compatible with this code. Additional information about U201 can be found in the MOS-2000 system documentation. SATLEVRH estimates are returned in the units of hundreds of feet. The MOS-2000 ID's (CCCFFF) processed are summarized below.

```
003 311 = AVN ( 5-level); threshold = 85
003 321 = AVN ( 5-level); threshold = 90
003 331 = AVN ( 5-level); threshold = 95
003 312 = Eta ( 9-level); threshold = 85
003 322 = Eta ( 9-level); threshold = 90
003 332 = Eta ( 9-level); threshold = 95
003 313 = GFS (11-level); threshold = 85
003 323 = GFS (11-level); threshold = 90
003 333 = GFS (11-level); threshold = 95
```

RESTRICTIONS:

Grids of relative humidity and height must be available for all pressure levels. The grid of the terrain height must also be available. SATLEVRH will not estimate heights that are determined to be below the terrain elevation. The relative humidity thresholds are restricted to the values of 85, 90, and 95 percent.

NONSYSTEM ROUTINES USED:

GFETCH, CONST1

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

SNOWFL

COMPUTES OBSERVED SNOWFALL AMOUNT

Mark Shirey
February 19, 1999

PURPOSE: To compute the observed snowfall amount over a specified period from 6-h reports. The amount will be calculated with the period ending on NDATE. If one of the 6-h periods is missing, then the total calculation will be set to missing. A trace is coded as .04. SNOWFL is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The snowfall amount is returned in units of inches, which are the units for the observations. The MOS-2000 ID's (CCCFFF) processed are:

708 402	= 12-hr snowfall total
708 405	= 24-hr snowfall total

RESTRICTIONS:

To compute the snowfall amount, all the 6-h snowfall reports needed for the total snowfall calculation must be available in the input vector (station-oriented) dataset. Snowfall amounts can only be calculated when NDATE ends at 0000 UTC, 0600 UTC, 1200 UTC, or 1800 UTC. Because RR is used in this subroutine, a dummy ID record is required U201.CN file. This record must have IDPARS(1) set to 799, with RR equal to the time period in hours that the total snowfall amount will cover minus 6 hours. For example, the 24-h snowfall total would have RR=18, with the variable id's in the U201.CN file (record 13 of the control file) equal to:

```
799000000 000000000 018000000
708405000 000000000 000000000
```

Special attention must be paid to the first processing date. Since this program looks back in time, data must be available before the first process date. For example, if the user is starting on December 2 at 0000 UTC, and wants the 24-h snowfall ending at that time, data must be available for each component 6-h period back to December 1 at 0600 UTC. If the 6-h snowfall amount is missing from December 1 at 0600, 1200, or 1800 UTC or from December 2 at 0000 UTC, then the subroutine will not run properly. Please check the U201 output files and be sure that data are available at the appropriate time periods before the first process date, or the program will not work properly.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

VORTADV

COMPUTES ABSOLUTE VORTICITY ADVECTION

Christopher A. Fiebrich
December 1, 1997

PURPOSE: To compute absolute vorticity advection on a grid by using grids of u- and v-winds and relative vorticity at any level for either a Northern Hemisphere polar stereographic or Northern Hemisphere Lambert map projection. VORTADV is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The relative vorticity data are retrieved by a call to VORTW, and then the absolute vorticity is computed internally. The absolute vorticity advection, defined as $-(u*\partial(\xi+f)/\partial x+v*\partial(\xi+f)/\partial y)$, is then computed by a call to ADVCTW. Given the advecting winds in m/sec and the relative vorticity in s^{-1}, the absolute vorticity advection is returned as $[s^{-1}]*s^{-1}*10^{10}$. The MOS-2000 ID's (CCCFFF) processed are:

004 300	=	Absolute Vorticity Advection on an isobaric surface;
004 301	=	Absolute Vorticity Advection on an isohyetal surface;
004 306	=	Absolute Vorticity Advection on a sigma surface.

RESTRICTIONS:

The grid must be at least 4 x 4 in size. Note that this subroutine does not compute geostrophic vorticity advection, but is designed only to compute vorticity advection from the model winds.

NONSYSTEM ROUTINES USED:

VORTW, ADVCTW

LANGUAGE: FORTRAN 77

LOCATION: With U201

SNOWEQ

COMPUTES ESTIMATED 24-H SNOWFALL AMOUNT PREDICTION

Benjamin Sfanos
February 1, 2003

PURPOSE: To compute the 24-h snowfall amount predicted by an NWP model. The estimate is based on an average temperature over the 24-h period as well as the predicted precipitation amount during that period. This predictor is computed by estimating the average temperature over the 24-h period from 2-m temperature forecasts valid every 3 hours (12 hours in the case of the original GFS extended-range model). According to the average temperature, the precipitation amount predicted by the model over the same 24-h period is then converted to a snowfall equivalent by using multiplicative factors provided by NCDC. SNOWEQ is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID's (CCCFFF) processed in SNOWEQ allow for different model temporal resolution in both the temperature forecasts and the precipitation accumulations. The ID’s allowed are:

003 640 =	Assumes model temperatures and precipitation values are available at 3-h increments
003 645 =	Assumes model temperatures and precipitation values are available at 12-h increments only

RESTRICTIONS:

As the subroutine is currently written, the 24-h snowfall amount can be calculated from either the GFS or Eta models. The algorithm is designed so that the forecast projection in U201 for the 24-h snowfall amount predictor has to be 24 hours or greater.

As with other U201 subroutines, if no data are available for the first processing date, the subroutine will not work. Also, according to the specified MOS-2000 ID, the computation requires that all the necessary temperature and precipitation variables needed to estimate the snowfall amount be available in the model archives. If any of these variables are missing, the predictor will be returned with a value of 9999.

Note that ID=003645 should only be used for older, extended-range archives of GFS data or for predictors beyond 192 hours, when the temporal resolution of the model output is every 12 hours. Under these circumstances, only the 24-h snowfall amounts ending at 0000 UTC or 1200 UTC can be calculated; the user is restricted to predictors ending at these times. For the short-range GFS archive, the extended-range (from 87 to 192 hours) GFS archive from April 2002 to the present, and the Eta archive, the ID of 003640 should be used. The projection time of the 24-h snowfall amount is then restricted to be greater than or equal to 24 hours and less than or equal to the last projection time available in the archive.

COMMENTS:

There are two known deficiencies in this subroutine that can be addressed in future versions of the software. First, the average temperature is computed over the entire 24-h period, and the snowfall equivalent factor estimated by this average is applied to the precipitation predicted over the 24-h period. A more accurate snowfall amount can be provided by calculating the estimate for the individual 3-h periods and then summing the individual 3-h snowfall estimates to generate a 24-h estimate. Secondly, in the current algorithm, the average temperature over the 24-h period is done by only using temperatures at the end of

3-h (or 12-h) periods and calculating a simple average. A more correct estimate of the average is based upon using the temperatures at the beginning and end of the appropriate periods and calculating a weighted average.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, TPCP3, TPCP12

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

SSR

COMPUTES SINGLE STATION REGRESSION (SSR) PREDICTORS

Rebecca Cosgrove
May 1, 2000

PURPOSE: To produce any of 4 SSR predictors using the appropriate model field and the associated equation coefficients to arrive at a probability that snow will occur at a given station. Data returned from this subroutine is valid at stations (vector data). SSR is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID's (CCCFFF) for the SSR predictors are:

001700 = 1000-850 mb thickness, grid binary, cutoff of 1300 m
002700 = 2-m temperature, grid binary, cutoff of 275 K
002710 = 850-mb temperature, grid binary, cutoff of 273 K
002715 = 850-mb temperature, continuous

RESTRICTIONS:

In order to run this subroutine, the user must supply in the U201.CN file a constant file that contains the SSR coefficients. These coefficients must have been computed previously and stored in the constant file. As with other U201 subroutines, if no data are available for the first processing date, the subroutine will not work properly. Please check the U201 output files and be sure that data are available for the first process date or the u201 program will not produce the desired results.

COMMENTS:

The SSR predictor is computed by evaluating a linear regression equation consisting of a model field and two coefficients. The table below indicates the id's of the coefficients according to the id of the SSR variable.

SSR ID	B_0 Coefficient	B_1 Coefficient
001700	461700	461701
002700	462700	462701
002710	462710	462711
002715	462715	462716

In this subroutine, the appropriate model field is fetched, and then processed before being inserted into the regression equation. Processing includes calculating a grid binary of the original field based on a preset cutoff (for all but 002715), smoothing the grid binary field as directed in the smoothing digit of the id, and interpolating the field to stations using the bilinear interpolation scheme. Again, the user can specify what type of smoothing should be done on the grid binary field, but the grid binary cutoffs and interpolation are preset in ssr.f.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, CONST1, SMTH5, SMTH9, SMTH25, SMTH2X, SMTH3X, INTRPB

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

SVRVEC

CREATES THUNDERSTORM AND SEVERE WEATHER PREDICTAND DATA

Kathryn K. Hughes
July 17, 2000

PURPOSE: To create predictand data valid over an interval of time from hourly reports of thunderstorm occurrence or severe weather events. In this particular subroutine, the hourly reports have been previously created in U523 which processes and encodes lightning data or severe weather reports into TDLPACK. These reports are stored in vector format at a set of regularly spaced points (i.e., a grid). SVRVEC retrieves the observed vector predictand data in TDLPACK and then aggregates the reports for a user-specified period of time. Examples of the observed vector data are lightning flashes, reports of tornadoes, large hail, damaging winds, and pilot reports of clear-air-turbulence and icing. SVRVEC sums hourly reports or determines the maximum values of reports for each grid block for an interval of hours designated by the predictand ID. Intervals normally used are 3, 6, 12, or 24 hours. Summed/max values are then returned to the calling program (OPTION). SVRVEC is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID's (CCCFFF) processed are:

707 200 = Occurrence of a thunderstorm
707 300 = Unconditional occurrence of severe weather
707 310 = Conditional occurrence of severe weather
707 320 = Number of tornadoes in the grid block
707 330 = Largest F-scale value in the grid block
707 340 = Number of hail reports in the grid block
707 350 = Largest hail size (inches) in grid block
707 360 = Number of wind reports in grid block
707 370 = Highest wind speed (knots) in grid block
707 380 = Total C-G lightning flashes in grid block
707 390 = C-G negative flashes in grid block
707 400 = C-G positive flashes in grid block
707 410 = Maximum signal strength (KA) in grid block
707 420 = Highest number of strokes per flash

The predictand ID's for specific hourly intervals (1, 3, 6, 12, or 24-hr) to process the data are designated in IDPARS(2)(the FFF of the first ID word) where:

FFF = XX0 for 1-hr interval
" = XX1 for 3-hr interval
" = XX2 for 6-hr interval
" = XX3 for 12-hr interval
" = XX4 for 24-hr interval

An example of hail reports summed over a 3-hr period:

CCCFFF = 707341

<u>RESTRICTIONS</u>:

Lightning records need to be present as input data in the U201.CN control file in order to calculate unconditional severe weather. The model number for the lightning data should be set to zero.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH, UPDAT

<u>LANGUAGE</u>: FORTRAN 90

<u>LOCATION</u>: With U201 in the u201lib file system.

SWEATI

COMPUTES SWEAT INDEX

Benjamin Sfanos
December 9, 1999

<u>PURPOSE</u>: To compute the SWEAT index. Based on the formula below, the SWEAT index is computed on a grid from dew point, total totals index, wind speed, and wind direction. Since the direct model output variables do not include dew point, total totals index, wind speed, and wind direction, these variables must first be computed within this subroutine. SWEATI is am almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for any additional explanation. The MOS-2000 ID processed is:

007 220 = SWEAT Index

<u>RESTRICTIONS</u>:

To compute the SWEAT index, the dew point at 850 mb must be available. The wind speed and wind direction at 850 and 500 mb, as well as the total totals index, must be computed. After the computation, wind speed is converted from m/s to knots for this subroutine.

<u>COMMENTS</u>:

SWEAT Index = 12 $*850T_d$ + 20*(TT-49) + $2V_{850}$ + V_{500} + 125*(S + 0.2), where

$850T_d$ = the 850-mb dew point,
TT = the total totals index,
V_{850} = the 850-mb wind speed,
V_{500} = the 500-mb wind speed, and
S = the sine of (500-mb wind direction - 850-mb wind direction).

NOTE: If TT < 49, 20*(TT-49) = 0. If $850T_d < 0$, $12*850T_d = 0$. Also, the shear term, that is, 125+(S+0.2) is set to zero when any of the following conditions are not met:

(1) 850-mb wind direction is in the range 130-250 degrees,
(2) 500-mb wind direction is in the range 210-310 degrees,
(3) (500-mb wind direction - 850-mb wind direction) is greater than zero, and
(4) both the 850-mb and 500-mb wind speeds are greater than or equal to 15 knots.

<u>NONSYSTEM ROUTINES USED</u>:

DEWPT, GFETCH, PRSID1, TTOTALS, WSPEED, WINDDR

<u>LANGUAGE</u>: FORTRAN 90

<u>LOCATION</u>: With U201 in the u201lib file system.

SWTXRF

COMPUTES PRODUCT OF SWEAT INDEX AND SEVERE THUNDERSTORM RELATIVE FREQUENCY

Kathryn K. Hughes
July 17, 2002

PURPOSE: To compute the interactive predictor SWTXRF which is the product of the SWEAT index and the relative frequency of severe thunderstorms. This predictor is computed on a grid from grids of the SWEAT index and the relative frequencies of severe thunderstorms. SWTXRF is an exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 IDs (CCCFFF) processed are:

007360 = SWEATI * previous 3 hr Severe Weather Relative Frequency
007365 = SWEATI * subsequent 3 hr Severe Weather Relative Frequency
007370 = SWEATI * previous 6 hr Severe Weather Relative Frequency
007375 = SWEATI * subsequent 6 hr Severe Weather Relative Frequency
007380 = SWEATI * previous 12 hr Severe Weather Relative Frequency
007365 = SWEATI * subsequent 12 hr Severe Weather Rel. Frequency
007370 = SWEATI * previous 24 hr Severe Weather Relative Frequency

COMMENTS:

The SWTXRF is useful for predicting severe thunderstorm activity. The higher the SWTXRF, the greater the probability of severe thunderstorm development. Since the relative frequencies of severe weather are rather noisy, and the values of the SWEAT index may get very high (well over 400), the user should use this variable as some type of point binary predictor. Possible cutoffs as a binary predictor may be 10, 25, 50, and 100. Since the SWTXRF variable relies on station-based relative frequency data, it is not possible to treat these fields as grid binaries.

NONSYSTEM ROUTINES USED:

PRSID1, SWEATI, CONST, INTRPB, SMTH5, SMTH9, SMTH25

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

TDPAVG

COMPUTES AVERAGE OF LAMP TEMPERATURE AND DEWPOINT FORECASTS

Bob Glahn
April 1, 2003

PURPOSE: To compute the average of LAMP temperature and dewpoint forecasts when the dewpoint exceeds the temperature. This subroutine is entered when either a temperature or dewpoint identifier is supplied in the control file. Temperature or dewpoint values are returned in the SDATA array. Note that the consistency check between the temperature and dewpoint is performed at stations. TDPAVG is almost exclusively for use in U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 IDs (CCCFFF) processed are:

002 303 = temperature converted from 002 301
003 303 = dewpoint converted from 003 301

NONSYSTEM ROUTINES USED:

GFETCH, INTRPA, INTRPB

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

TIMGRD

INTERPOLATES MODEL GRID FIELDS TO INTERMEDIATE HOURS

Bob Glahn
September 1, 2003

PURPOSE: To interpolate from model grids valid at 3-h intervals to intermediate hours. For instance, output from the GFS model at 3-h intervals can be interpolated to hourly values for use in LAMP development. TIMGRD is almost exclusively for use in U201 or its derivatives, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 IDs (CCCFFF) processed, that is, IDPARS(1) and IDPARS(2) are:

00X XXX =	Model forecasts, when the projection in ID(3) is not evenly divisible by 3, and the model DD in ID(1) is not 5 (i.e., is not the LAMP model)

COMMENTS:
This subroutine is used exclusively for interpolation of model data.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

TIMTRP

INTERPOLATES FORECASTS TO INTERMEDIATE HOURS

Bob Glahn
August 1, 2003

PURPOSE: To interpolate from forecast model grids or MOS forecasts valid at 3-h intervals to intermediate hours. For instance, output from the GFS model at 3-h intervals or MOS 3-h forecasts can be interpolated to hourly values for use in LAMP development. TIMTRP is almost exclusively for use in U201 or its derivatives, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 IDs (CCCFFF) processed, that is, IDPARS(1) and IDPARS(2) are:

00X XXX - Model forecasts, when the projection in ID(3) is not evenly divisible by 3, and the model DD in ID(1) is not 5 (i.e., is not the LAMP model)

2XX XXX - Statistical forecasts, when the projection in ID(3) is not evenly divisible by 3.

COMMENTS:

This subroutine can be used for interpolation of either vector (station) or gridpoint data. Data with identifiers other than the ones given above can be accommodated by revising the CALL statement in OPTION and by revising the criteria in TIMTRP.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, OPTN2

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

TMPADV

COMPUTES TEMPERATURE ADVECTION

Christopher A. Fiebrich
Wei Yan(rev.)
November 30, 1997

PURPOSE: To compute temperature advection on a grid by using grids of u- and v-winds and temperature at any level for either a Northern Hemisphere polar stereographic or Northern Hemisphere Lambert conformaal map projection. TMPADV is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The temperature data are retrieved by a call to GFETCH. The temperature advection -(u*∂T/∂x+v*∂T/∂y) is then computed by a call to ADVCTW. Given the winds in m/sec and the temperature in K, the temperature advection is returned as [K]*s^{-1}*10^5. The MOS-2000 ID (CCCFFF) processed are:

004 320 = Temperature advection on an isobaric surface,
004 321 = Temperature advection on a constant height surface,
004 326 = Temperature advection on a sigma surface.

RESTRICTIONS:

The grid must be at least 4 x 4 in size. Note that this subroutine does not compute geostrophic temperature advection, but is only designed to use model winds in the temperature advection computations.

NONSYSTEM ROUTINES USED:

GFETCH, ADVCTW, PRSID1

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

TPCP3

COMPUTES 3-H TOTAL OR CONVECTIVE PRECIPITATION AMOUNT

Benjamin Sfanos
Joseph Maloney(rev.)

December 8, 1999

PURPOSE: To compute the total or convective 3-h precipitation amount. The total forecast precipitation amount output by the numerical prediction model is divided into convective and non-convective components in both the Global Spectral and Eta models. In the MDL model archives, only the total precipitation and convective amounts are saved. Moreover, the interval of time for which precipitation is output (that is, over a 3-, 6-, 9-, or 12-h period) varies from model to model. This subroutine allows for the calculation of either the total or convective 3-h precipitation amount and is able to do the computation for only a 3-h accumulation period. TPCP3 is an almost exclusive routine of U201, and the calling sequence is accommodated for that use. Any additional information pertaining to U201 can be located in the MOS-2000 documentation guide. The total or convective precipitation amount is returned in units of precipitation amount, which is usually mm. The MOS-2000 ID's processed are:

003 205 = Total precipitation amount (mm)
003 235 = Convective precipitation amount (mm)

RESTRICTIONS:

The grid must be at least 4 x 4 in size. To compute the total or convective precipitation amount, the convective precipitation as well as the total precipitation must be available for the requested period.

COMMENTS:

When the requested tau is greater than 3, the 6-h precipitation fields are accumulated first. A 3-h precipitation amount with tau = tau - 3 is then read and subtracted from the 6-h values. Note that subroutine NONCNVP is used when computing the non-convective precipitation.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 `in the u201lib file system.`

TPCP6

COMPUTES 6-H TOTAL OR CONVECTIVE PRECIPITATION AMOUNT

Benjamin Sfanos
Joseph Maloney(rev.)

December 8, 1999

PURPOSE: To compute the total or convective 6-h precipitation amount. The total forecast precipitation amount output by the numerical prediction model is divided into convective and non-convective components in both the Global Spectral and Eta models. In the MDL model archives, only the total precipitation and convective amounts are saved. Moreover, the interval of time for which precipitation is output (that is, over a 3-, 6-, 9-, or 12-h period) varies from model to model. This subroutine allows for the calculation of either the total or convective 6-h precipitation amount and is able to do the computation for only a 6-h accumulation period. TPCP6 is an almost exclusive routine of U201, and the calling sequence is accommodated for that use. Any additional information pertaining to U201 can be located in the MOS-2000 documentation guide. The total or convective precipitation amount is returned in units of precipitation amount, which is usually mm. The MOS-2000 ID's processed are:

003 210 = Total precipitation amount (mm)
003 240 = Convective precipitation amount (mm)

RESTRICTIONS:

The grid must be at least 4 x 4 in size. To compute the total or convective precipitation amount, the convective precipitation as well as the total precipitation must be available for the requested period.

COMMENTS:

The algorithm used to compute the 6-h precipitation amount depends on the model. In some cases, two 3-h values can be summed to get the 6-h amount. In other cases, a 6-h value is subtracted from a 12-h value or a 3-h value is subtracted from a 9-h value to get the appropriate 6-h value. Note that subroutine NONCNVP is used when computing the non-convective precipitation.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 `in the u201lib file system.`

TPCP12

COMPUTES 12-H TOTAL OR CONVECTIVE PRECIPITATION AMOUNT

Benjamin Sfanos
Joseph Maloney(rev.)

December 8, 1999

PURPOSE: To compute the total or convective 12-h precipitation amount. The total forecast precipitation amount output by the numerical prediction model is divided into convective and non-convective components in both the Global Spectral and Eta models. In the MDL model archives, only the total precipitation and convective amounts are saved. Moreover, the interval of time for which precipitation is output (that is, over a 3-, 6-, 9-, or 12-h period) varies from model to model. This subroutine allows for the calculation of either the total or convective 12-h precipitation amount and is able to do the computation for only a 12-h accumulation period. TPCP12 is an almost exclusive routine of U201, and the calling sequence is accommodated for that use. Any additional information pertaining to U201 can be located in the MOS-2000 documentation guide. The total or convective precipitation amount is returned in units of precipitation amount, which is usually mm. The MOS-2000 ID's processed are:

003 220 = Total precipitation amount (mm)
003 250 = Convective precipitation amount (mm)

RESTRICTIONS:

The grid must be at least 4 x 4 in size. To compute the total or convective precipitation amount, the convective precipitation as well as the total precipitation must be available for the requested period.

COMMENTS:

The algorithm used to compute the 12-h precipitation amount depends on the model. In some cases, two 6-h values can be summed to get the 12-h amount. In other cases, the 12-h amount is summed from a combination of 3- and 6-h amounts. Note that subroutine NONCNVP is used when computing the non-convective precipitation.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

TPCP24

COMPUTES 24-H TOTAL OR CONVECTIVE PRECIPITATION AMOUNT

Benjamin Sfanos
Joseph Maloney(rev.)

December 8, 1999

PURPOSE: To compute the total or convective 24-h precipitation amount. The total forecast precipitation amount output by the numerical prediction model is divided into convective and non-convective components in both the Global Spectral and Eta models. In the MDL model archives, only the total precipitation and convective amounts are saved. Moreover, the interval of time for which precipitation is output (that is, over a 3-, 6-, 9-, or 12-h period) varies from model to model. This subroutine allows for the calculation of either the total or convective 24-h precipitation amount and is able to do the computation for only a 24-h accumulation period. TPCP24 is an almost exclusive routine of U201, and the calling sequence is accommodated for that use. Any additional information pertaining to U201 can be located in the MOS-2000 documentation guide. The total or convective precipitation amount is returned in units of precipitation amount, which is usually mm. The MOS-2000 ID's processed are:

003 226 = Total precipitation amount (mm)
003 256 = Convective precipitation amount (mm)

RESTRICTIONS:

The grid must be at least 4 x 4 in size. To compute the total or convective precipitation amount, the convective precipitation as well as the total precipitation must be available for the requested period.

COMMENTS:

The algorithm used to compute the 24-h precipitation amount depends on the model. Various combinations of 3-, 6-, and 12-h precipitation amounts have been allowed in the computations. Note that subroutine NONCNVP is used when computing the non-convective precipitation.

At one time, this subroutine used MOS-2000 ID's of 003225 (total precipitation) and 003255 (convective precipitation) to identify the variable being calculated. An error was discovered in that version of the subroutine, and these original ID's were removed from TPCP24. In addition, at one time, TPCP24 calculated the non-convective precipitation, but that logic was also removed.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 90

LOCATION: With U201 `in the u201lib file system.`

TRPD

DOES CALCULATIONS FOR INTERPOLATION SUBROUTINE INTRPD

Harry R. Glahn
July 1, 2003

PURPOSE: To do computations for the interpolation subroutine INTRPD. Subroutine TRPD furnishes a value at each of NSTA stations from the closest gridpoint to that station within the 4-gridpoint square in which the station is located such that the gridpoint value:

1) is not missing (=9999.);
2) is not defined as unusable by an external grid mask; and
3) is not more than ECONST feet different in elevation from the station.

RESTRICTIONS:

None

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

TSLCM

CALCULATES UPSLOPE IN TERRAIN ELEVATION

Harry R. Glahn
July 1, 2002

PURPOSE: To do computations for subroutine TSLOP. Subroutine TSLCM calculates the upslope in the terrain elevation. Upslope in the west or south direction are defined to be positive. The slope in both the x- and y-directions on the grid are used to calculate this value after being turned to be oriented in the north-south and east-west direction relative to the earth.

RESTRICTIONS:

None

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

UPSLCM

CALCULATES UPSLOPE WINDS FROM U AND V WIND COMPONENTS

Harry R. Glahn
July 1, 2002

PURPOSE: To do computations for subroutine UPSLOP. Subroutine UPSLCM calculates the upslope winds caused by variations in the terrain elevation. Upslope winds in the west or south direction are defined to be positive. The u- and v-winds, as well as the slope in the terrain elevation, are oriented with respect to the north-south and east-west direction on the earth.

RESTRICTIONS:

None

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 90

LOCATION: With U201 `in the u201lib file system.`

WETBULBT

COMPUTES WET BULB TEMPERATURE

Wei Yan
December 9, 1999

PURPOSE: To compute the wet bulb temperature on a grid from grids of temperature, dew point, and pressure at any level. The grid containing the pressure values is internally created by using the MOS-2000 identifier if the wet bulb temperature is to be calculated on an isobaric surface. WETBULBT is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for any additional explanation. The wet bulb temperature is returned in units of degrees Kelvin. The MOS-2000 ID's (CCCFFF) processed are:

003 110 = wet bulb temperature on an isobaric surface.
003 111 = wet bulb temperature on a surface of constant height.
003 116 = wet bulb temperature on a sigma surface.

RESTRICTIONS:

To compute the wet bulb temperature, the temperature and the dew point data must be available for the requested period, and both must be in degrees Kelvin.

WETBULBT expects pressure data from the NGM to be in hectopascals (millibars) for computation on a constant height or sigma surface. All other model pressure data are expected to be in Pascals, and are converted to hectopascals within the DEWPT subroutine.

NONSYSTEM ROUTINES USED:

PRSID1, GFETCH, DEWPT

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

WINDDR

COMPUTES WIND DIRECTION IN DEGREES

Benjamin Sfanos
December 9, 1999

PURPOSE: To compute the wind direction in degrees at any level. If the earth-oriented u- or v-wind is missing, the wind direction is set to missing. WINDDR is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for any additional explanation. The wind direction is returned in units of degrees. The MOS-2000 ID's (CCCFFF) processed are:

004 200 = Wind direction on an isobaric surface.
004 201 = Wind direction on a surface of constant height.
004 206 = Wind direction on a sigma surface.

RESTRICTIONS:

To compute the wind direction, the u- and v- components of the wind must be available for the requested projection. These wind components are oriented with respect to the earth coordinate system, so that the computed wind direction is relative to the north-south and east-west earth orientation. Note that if the wind speed at the grid point is less than 0.05 m/sec, than the wind speed is assumed to be zero, and the wind direction is set to zero, i.e., calm.

COMMENTS:

While this subroutine is used in U201 to compute the wind direction on the model grid and so provides an estimate of wind direction at a grid point, interpolation of the wind direction to station locations is not recommended, and should not be done. Because the wind is a vector quantity, estimation of the wind direction at a station location should be done by interpolating the u- and v-components of the wind to the site, and then calculating the direction. Subroutine DIRFUV, which can be called by MOS-2000 codes (for example, U660) used process station-oriented data, is useful for obtaining the proper wind direction from interpolated wind components.

NONSYSTEM ROUTINES USED:

EOWND, PRSID1

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

ZRPRED

COMPUTES VERTICAL PROFILE INDEX INDICATING OCCURRENCE OF FREEZING RAIN

Rebecca Allen
May 1, 2000

PURPOSE: To produce a vertical sounding index (also known as the zr predictor), which indicates whether the vertical wet bulb temperature profile, in conjunction with the surface temperature, is conducive to the occurrence of freezing precipitation. This predictor is computed by using model wet bulb temperatures, surface pressure, and the 3- or 12-h accumulated precipitation. The data returned from ZRPRED are valid at stations (vector data), and have values ranging from 0 to 2. A value of 0 indicates that the vertical sounding of the wet bulb temperature and the surface temperature are not conducive to the occurrence of freezing precipitation. ZRPRED is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID's (CCCFFF) processed in ZRPRED allow for different model vertical resolutions and precipitation accumulations. The ID's allowed are:

003 152 = Eta (currently not active)
003 153 = GFS (on or after 10/1/98)
003 154 = GFS (prior to 10/1/98; uses 12-h precipitation values)
003 156 = GFS (prior to 10/1/98; uses 3-h precipitation values)

RESTRICTIONS:

As with some other U201 subroutines, if no data are available for the first processing date, the subroutine will not work. Also, the computation requires a minimum of five wet bulb temperatures at various vertical levels, the 2-m wet bulb temperature, the 3-h (or 12-h) accumulated precipitation, and the surface pressure. If any of these fields are missing, the zr predictor will be returned with a value of 9999.

Note that an ID = 003154 is only applicable to older, extended-range archives of GFS data. Note, too, that the zr predictor can not be computed from the Eta model archives as ZRPRED is currently configured.

COMMENTS:

There are three components to the zr predictor. First, the 2-meter wet bulb temperature is checked to see if it is cold enough for freezing rain to occur. A sliding scale is used to allow for the greatest weight to be given to temperatures below 273K, and lesser weight given to values between 273K and 276K. Next, the vertical wet bulb temperature profile is checked to see if there is a warm inversion aloft. If both of these conditions are met, the zr predictor will receive a value greater than zero. Finally, if it was determined that freezing precipitation could occur, a check is done of the 3-h (or 12-h) model accumulated precipitation. If this value is greater than 0, an additional value of 1 is added to the value of the zr predictor computed in the first two steps.

A value for the zr predictor is computed at each grid point, and then those final values are interpolated to stations.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, TPCP3, TPCP12, WETBULBT

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

OBSADJSPD

ADJUSTS OBSERVED WIND SPEEDS TO A 10 METER SURFACE

David E. Rudack
June 20, 2008

PURPOSE:

To adjust observed wind speeds to a 10 meter surface for those anemometers that are either above or below 10 meters. Wind speed observations taken at the 10 meter level remain unchanged by this subroutine. The following equation is used to modify the wind speed observations not taken at 10 meters:

SPD_{adj} = Adjusted wind speed (knots)
SPD_{orig} = Original Wind speed (knots)
Height = Anemometer height (meters)

$$\mathbf{SPD_{adj} = SPD_{orig} * (10.0/Height) ** 0.11}$$

The following MOS-2000 ID (CCCFFF) may be used:

704330

RESTRICTIONS:

This subroutine is designed to process observations only.

COMMENTS:

For data input, this subroutine requires a constant file that contains each station's anemometer height. See subroutine OPTION for the appropriate call sequence.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: u201lib

OBSADJDIR

SETS THE OBSERVED WIND DIRECTION TO ZERO (CALM) WHEN THE ADJUSTED WIND SPEED EQUALS ZERO

David E. Rudack
June 20, 2008

PURPOSE:

To set the observed wind direction to a value of zero (calm) when the observed adjusted wind speed equals zero. All wind directions with adjusted wind speeds of 1 knot or greater remain unchanged by this subroutine.

The following MOS-2000 ID (CCCFFF) may be used:

704230

RESTRICTIONS:

This subroutine is designed to process observations only.

COMMENTS:

For data input, this subroutine requires the observed adjusted wind speed (704320000). See subroutine OPTION for the appropriate call sequence.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: u201lib

OBSADJUV

ADJUSTS THE OBSERVED U- AND V- WIND COMPONENTS TO A 10 METER SURFACE

David E. Rudack
June 20, 2008

PURPOSE:

To adjust the observed U- and V- wind components to a 10 meter surface (for those stations with anemometers above or below 10 meters). The observed U- and V- wind components for those stations with an anemometer height of 10 meters, remain unchanged.

The following MOS-2000 IDs (CCCFFF) may be used:

704020	– Adjusted U-component
704120	– Adjusted V-component

RESTRICTIONS:

This subroutine is designed to process observations only.

COMMENTS:

This subroutine requires the adjusted wind speed (704320000) and the adjusted wind direction (704220000) as data input. See subroutine OPTION for the appropriate call sequence.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, CONST

LANGUAGE: FORTRAN 77

LOCATION: u201lib

VARG

COMPUTES VARIANCE AT POINTS ON A GRID

Harry R. Glahn
March 15, 2009

PURPOSE: To compute the variance at each point on a grid of the surrounding points within a prescribed radius R indicated by IDPARS(8) and a table ITABLE( ) within the subroutine. Any CCCFFF is accommodated; the key is IDPARS(8) must ≥ 7. Actually, the computation is not over a circle, but a square 2*R on a side. When R = 1, 9 points will be involved.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

This routine is called from OPTION or OPTN2 from U201. The variable for which the variance is computed can be a computed variable, and even over time. OPTION is entered, and if IDPARS(8) is > 7, VARG is entered. If the variable is available in internal storage, it is retrieved and the variance computed. If not, then OPTN2 is entered and necessary computations are made, with a return to VARG for the variance to be computed.

The values in ITABLE are:
Days 1-5 = 1
Days 6-13 = 2
Days 13-16 = 3
When IDPARS(8) = 9, the radius R = the value in ITABLE( ) for the forecast day.
When IDPARS(8) = 8, the radius R = the value in ITABLE( ) for the forecast day times 2.
When IDPARS(8) = 7, the radius R = the value in ITABLE( ) for the forecast day times 4.
This gives some flexibility as to R which varies by projection–the longer the projection, the larger the radius.

If it turns out, these radii should be something else, then ITABLE can be changed. If that is done, then previous calculations of, or resulting from, VARG will likely have to be discarded.

Note that the actual area covered by the variance calculation depends on the gridlength. The GFS archive has two different resolutions. This will not cause a problem, as long as the application of equations is made on the same resolution as the archive, but one should be aware of the change when diagnosing results.

NONSYSTEM ROUTINES USED:

GFETCH, VARG1

LANGUAGE: FORTRAN 90

LOCATION: u201lib

PCPEFF

COMPUTES PRECIPITATION EFFICIENCY

Joseph Maloney
August 19, 2008

PURPOSE: To compute the precipitation efficiency, which is the product of mean relative humidity in a layer and the precipitable water. This is based on research from the Peachtree City forecast office by James Noel and Jeffrey C. Dobur entitled “A Pilot Study Examining Model Derived Precipitation Efficiency for use in Precipitation Forecasting in the Eastern United States.”

PCPEFF is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The precipitable water is obtained by a call to GFETCH which reads the variable from the direct model output. The mean relative humidity is obtained by a call to MEANRH which calculates the mean relative humidity in the desired layer. Note that ID(2) is used to determine the layer used for the mean RH calculation. The user should see the documentation for subroutine MEANRH for additional explanation of the proper setting of ID(2).

In the final calculation, precipitable water is converted from mm to inches, and the mean relative humidity is converted from percent to decimal. Thus, precipitation efficiency is returned in inches, following the methodology of the paper. The MOS-2000 ID (CCCFFF) processed by this subroutine is:

003 370 =	precipitation efficiency for an isobaric layer specified by the second ID word.

RESTRICTIONS:

The grid must be at least 4 x 4 in size. For this subroutine to work properly, precipitable water and ALL relative humidity fields required by subroutine MEANRH must be available.

NONSYSTEM ROUTINES USED:

GFETCH, MEANRH

LANGUAGE: FORTRAN 90

LOCATION: With U201 in the u201lib file system.

REFERENCE:

Noel, James, and Jeffrey C. Dobur, "A Pilot Study Examining Model-Derived Precipitation Efficiency for use in Precipitation Forecasting in the Eastern United States", *Natl. Wea. Digest*, December 2002, 26:3, 3-8.

U350

CREATES MOS-2000 RANDOM ACCESS FILES

Harry R. Glahn
December, 1996

<u>PURPOSE</u>: U350 establishes one or more random access files for the MOS-2000 external file system. The master key record is written as well as a blank key record. U350 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by a PARAMETER statement in which the variable L3264B is set to 32 or 64. Each record read from the control file U350.CN will cause one file to be created.

<u>CONTROL FILE INPUT:</u> 'U350.CN' (Unit = KFILDI)

Record Type 1 - Format (2I8,1XA60)

This record contains the information about the file to create. Records are read until the terminator 9999 is encountered when reading MAXENT, and for each record read, a file will be created. KFILDI is the input unit number as specified in a DATA statement.

<u>MAXENT</u> - The maximum number of key entries in each key record in the file to be created. <u>Use ≤ 300 to be consistent with random access file routines, else NW in those routines will have to changed; 300 is strongly recommended.</u>

<u>NBYTES</u> - The physical record size in bytes of the file to be created. This is specified in bytes because most of the data to be written to the file system will be in TDLPACK format, and therefore the record size will be nearly constant whether the CRAY or the HP workstations are being used.

<u>CFILX</u> - The name of the file to create, maximum of 60 characters. This can be only the file name itself, or can include the path in case the file is to be created outside the directory where U350 is running. (CHARACTER*60)

<u>EXAMPLE CONTROL FILE</u>: 'U350.CN'

An example exists as file 'U350.CN' in directory 'home21/tdllib/u350' on blizzard.

<u>OUTPUT</u>:

All output is to the default print file Unit No. KFILDO. It consists of the information about the file opened.

<u>RESTRICTIONS</u>

A file with the same path and name as the one to be created cannot already exist.

COMMENTS

Note that there is no dimension or restriction as to logical record size.

The attachment indicates some of the capabilities and characteristics of the programs to manage the random access files--U351, U352, U353, and U361.

SETTING UP U350

Normally, the existing U350 will not have to be changed. Two PARAMETER variables are used:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
NW -	Set to some number such that NW $\geq$ NBYTES (read from the control file U350.CN) *6. An array with dimension NW is used to write the blank key record.

NONSYSTEM ROUTINES USED

WRKEYM, IERX, TIMPR

LANGUAGE: FORTRAN 77

LOCATION: home21/tdllib/dru350 on blizzard

ATTACHMENT TO U350

CAPABILITIES OF U351, U352, AND U353

U351	U352	U353	U361
Reads a station list for the RA file station directory. Blanks in the list are deleted.	Same as U351.	Same as U351.	Not needed.
If this is a new file, writes the station list to the RA station directory <u>alphabetized</u>. If the file already read has a station directory, it must match exactly the one read after being alphabetized.	Same as U351	Writes the station list to the RA station directory <u>alphabetized</u>. The station directory(s) of the input RA file(s) do <u>not</u> have to match the station directory of the output. This allows deletion and/or addition of stations. When adding stations, the corresponding data values will be set to 9999.	Not needed.
Reads one or more sets of data to write to the RA file from the ASCII .CN file. Reading is done with RDF which deletes zeros and blanks. Omitting zeros could be a problem; may have to replace with 9999 before reading.	Reads one or more data sets to write to the RA file from one or more sequential TDLPACK files.	Reads one or more data sets to write to the RA file from one or more RA files.	Reads one or more data sets from file KFILP (name read from .CN file) preceded for each data set by data ID and grid information.

U351	U352	U353	U361
IDs of data to write are contained with the data read. All data read will be written.	IDs of data to write are read in U352; only the data sets read on input that match the IDs read are written to the RA file. That is, not all data on the sequential input files need be written to the RA file.	IDs of data to not copy from the input RA file(s) are read with the name(s) of the input file(s).	Same as U351.
Data are written with replacement.	Same as U351.	Data are written with replacement. This means duplicate IDs on one or more of the input RA files will have only one corresponding ID on the output. If there are duplicates on the input, the one on the output will be the last one read, unless it is indicated as a record to delete.	Same as U351
Writes to a new or an existing RA file.	Same as U351.	Writes only to a new RA file. This allows the new file, which might be identical to the old file, to be inventoried. Empty input records, that might have occurred when writing with replacement, are omitted.	Same as U351
The stations in the RA station directory are printed in the ftn12 file.	Same as U351.	The stations in the output RA station directory are printed in the ftn12 file.	Not needed.

U351	U352	U353	U361
IDs of records written are printed in the ftn12 file.	Same as U351.	IDs of records deleted from the input files are printed along with the IDs of records written are printed in the ftn12 file.	Same as U351, plus grid information. Data are written with /D compile.
Normal program stop is "STOP 351". If an error stop occurs, the output RA master directory is not updated (effectively, no data have been added).	Normal program stop is "STOP 352". If an error stop occurs, the output RA master directory is not updated (effectively, no data have been added).	Normal program stop is "STOP 353". If an error stop occurs, the output RA master directory is not written (effectively, no data have been written).	Same as U351, except normal program stop is "STOP U361".

U351

PUTS DATA INTO MOS-2000 RANDOM ACCESS FILES

Harry R. Glahn
December 1, 1996

PURPOSE: U351 is a template for how data can be put into a MOS-2000 external random access file. It writes the directory call letters record, if one is not already present, and optionally station WBAN numbers, elevations, latitudes, and longitudes, as well as data supplied to it. If a call letters record is already present, U351 checks to make sure it is the same as the one that would have been written; if not, U351 stops. The call letters are always the "New" station identifiers; no provision is made for "old" call letters. U351 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by a PARAMETER statement in which the variable L3264B is set to 32 or 64. Multiple input files are accommodated as well as multiple data records and their identification on a file.

CONTROL FILE INPUT: 'U351.CN' (Unit = KFILDI)

Record Type 1 - Format (A72) **Run Identification**

This record is used to identify the run.

Record Type 2 - Format (4I4) **Control Parameters**

IWRELE - Indicates whether (1) or not (0) the station elevations are to be written.

IWRWBN - Indicates whether (1) or not (0) the station WBAN numbers are to be written.

IWRLAT - Indicates whether (1) or not (0) the station latitudes are to be written.

IWRLON - Indicates whether (1) or not (0) the station longitudes are to be written.

Record Type 3 - Format (I3,4XA60) **Station and Location Files**

This pair of records (plus the terminator record) identifies the data sets from which the station (or location) information is obtained. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 3 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters which are to be written (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the default input file number, KFILDI = 5, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 4 - Format (I3,4XA60) **Random Access Output File**

This record (plus the terminator record) identifies the MOS-2000 random access output file. Records are read until the terminator KFILX( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILX - Unit number for the random access output file.

CFILX - Name of the file corresponding to KFILX. (CHARACTER*60)

Record Type 5 - Format (7(A8,1X)) **Station List**

This group of records identifies the stations for the station directory to be written to the file. If KFILD(1) ≠ KFILDI (the default input file), this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) (K=1,NSTA). This list is read within subroutine RDSTAD by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations in the list are not tolerated; a diagnostic is furnished on unit IP(5) = 12. The stations are placed in alphabetical order providing the directory is in alphabetical order. Stations not in the directory are put at the end of the list. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank (b) filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. These call letters can be "new" station identifiers or "old" call letters; however, the matching new station identifiers will be the ones written to the output file. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1.

Record Type 6 - Format (I3,4XA60) **Input Data File**

This pair of records (plus the terminator record) identifies the data sets from which the station (or location) information is obtained for the data to read. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILP(J) - Unit number for the file containing the station call letters which match the ASCII data to read (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the default input file number, KFILDI = 5, in which case CFILP(1) is not used.

CFILP(J) - Name of file matching KFILP(J). When KFILP( ) = KFILDI, CFILP( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

This Record Type 6 can be read in quadruples with Record Types 7, 8, and 9. That is, files are defined in Record Type 6, then station locations are read from Record Type 7, then data identification and data are read multiple times as indicated in Record Types 8 and 9. KFILP(1) can be the same on repeated readings; KFILP(2) would normally always be the same.

Record Type 7 - Format (7(A8,1X)) **Stations on Input File**

This group of records identifies the stations on the input file.

CCALLD(K) - Call letters (or other 8-character location designator) of stations (or locations) (K=1,NSTA). This list is read within subroutine RDSTAL by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations in the list are not tolerated; a diagnostic is furnished on unit IP(5) = 12. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALLD( ) will be blank (b) filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. These call letters can be "new" station identifiers or "old" call letters; however, the matching new station identifiers will be the ones written to the output file. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1.

Record Type 8 - Format (I9,3I10,2I4,1XA32) **Variable ID**

Each entry in this group of records identifies a variable to be read from Unit No. KFILP(1) and written to Unit No. KFILX(1). Reading of Record Types 8 and 9 continues alternately until the terminator ID(1) = 999999 is found. At that point, the run terminates.

ID(J) - The 4-word MOS-2000 ID of the data to write (J=1,4).

ISCALE - The scaling parameter. The data packed and written will be the original values read in *10**ISCALE, rounded.

NCHAR - Number of characters of data provided in PLAIN( ) to be packed with the data, maximum of 32.

PLAIN - Up to 32 characters of data to pack describing the data. (CHARACTER*32)

Record Type 9 - Format (7F9.0) **Data**

RECORD(J) - The data to pack (J=1,NSTA). These data are read with subroutine RDF, which eliminates blanks and zeros, and stops reading when it finds the terminator 999999. The values must be in the same order as the station list read in Record Type 5 or its alternate.

CONTROL FILE INPUT: (Name read from U351.CN) (Unit = KFILD(1) or KFILP(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided on file 'U351.CN', this group of records identifies the stations (or locations) for the station directory to be written to the file. If KFILD(1) read in Record Type 5 = KFILDI (the default input unit number), this file.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) (K=1,NSTA). This list is read within subroutine RDSTAD by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations in the list are not tolerated; a diagnostic is furnished on unit IP(5) = 12. The stations are placed in alphabetical order providing the directory is in alphabetical order. Stations not in the directory are put at the end of the list. RDSTAD also supplies the station elevations, WBAN numbers, latitudes, and longitudes in case they are to be written (see Record Type 2). The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank (b) filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. These call letters can be "new" station identifiers or "old" call letters; however, the matching new station identifiers will be the ones written to the output file. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1.

CONTROL FILE INPUT: (Name read from U351.CN) (Unit = KFILD(2) or KFILP(2))

Record Type 1 - Format **Station Locations**
(A8,1XA8,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. It is needed unless KFILD(1) = KFILD(2) = KFILDI; for this highly unusual case, both the station list and the associated information are read from the input control file KFILDI. When KFILD(1) = KFILD(2) ≠ KFILDI, this control file is the same as the one above and both the station list and the associated information are taken from this file. If both files are needed (the usual case; KFILD(1) ≠ KFILD(2) ≠ KFILDI), the call letters here are matched with those in the station list and the appropriate information extracted. When this station directory is alphabetical, the final list of stations will be alphabetical no matter the order read in. No terminator is used. Most of this information is used only within subroutine RDSTAD to give information about the stations.

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list unless KFILD(1) = KFILD(2), in which case these call letters comprise the station list and at the completion of reading this file, K=1,NSTA. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited, except when it also constitutes the list to be kept, in which case the number of entries is limited by ND1-1. No terminator is used.

EXAMPLE CONTROL FILE: 'U351.CN'

An example exists as file 'U351.CN' in directory 'home21/tdllib/.u351' on blizzard.

OUTPUT:

All output is to the default print file Unit No. KFILDO which is set by DATA statement to 12.

RESTRICTIONS

The random access file must have been created with U350. The subroutine RDF used to data read into RECORD( ) to pack and write eliminates zeros and blanks.

U351 obtains information from the station directory with the subroutine RDSTAL. A variable furnished to RDSTAL is "NEW" which in U351 is set to 1 to signify that the "new" station identifiers are to be used; no provision is made for putting "old" call letters on the output file.

A diagnostic indicating "INCREASE NW FROM xxx TO yyy..." will occur if the PARAMETER in the file access routines does not match MAXENT used in U350 to set up the file. This can probably be cleared up by recreating the file with U350 and using MAXENT = yyy (probably 300).

COMMENTS

Constant data may be put into the MOS-2000 external random access files from a variety of sources. U351 will accept thresholds produced in ASCII by program U830. U351 uses the routine PAWRA, which in turn uses UNPKBG, PACK1D, and WRTDLM.

Normally, the station directory files DIRNAM(2), CFILX(2), and CFILP(2) will all be the same, but wouldn't have to be.

The directory on the input need not be the same as the one on the random access file. The input data are put into the proper order (corresponding to the directory read in Record Type 5), packed, and written with replacement; if a duplicate is found, it is written in place, provided no more physical records are required than were originally used. If more physical records are needed, the old key is "zeroed out" and a new key and record written. This will cause fragmentation, but should be of no concern for a few replacements. Even with wholesale replacement, fragmentation may not be a problem. This can be checked by comparing the sizes of the old and new files.

Duplicate stations in Record Types 5 or 7 are not permissible. If a duplicate is found, a diagnostic is printed and U351 stops.

Once all records that need to be processed from an input file have been read and written to the random access output, another input file name is read with Record Type 6, followed by another station directory and set of variables. This allows processing of multiple input files in one U351 run. Note that this requires a double "99" terminator after the last file name, one for the last file, and one for the empty set.

Memory should be no problem, so ND1, ND4, and ND5 can be reasonably large so that they won't have to be changed frequently.

Note that while the station list on the ASCII input need not be the same as an existing directory on the random access output, the input directory read in Record Type 5 <u>must</u> match the output directory (if one exists). This is for safety to a assure a "standard" list is being dealt with.

If there are stations on the input ASCII packed file that are not on the random access file, that station is just, of course, omitted. If there are stations in the random access station directory not on the sequential packed input file, that station will be given the "missing" value of 9999.

The attachment to the writeup for U350 gives characteristics of programs U351, U352, and U353.

Any stop other than STOP U351 and a time stamp in the ftn12 file is an error and the master directory is not updated. That is, no data have been added to the output random access file.

<u>SETTING UP U351</u>

Five PARAMETER variables are used:

ND1 -	The maximum number of stations or locations that can be dealt with on this run. This has little effect on storage, so it can be kept at some reasonably large number.
ND5 -	Set to ND1.
ND7 -	Dimension of variables to use in the packing of the data. Keep this at 54.
L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
L3264W -	The number of "words" on the machine being used in 64 bits. Set to 64/L3264B.

<u>NONSYSTEM ROUTINES USED</u>

RDSNAM, RDSTAL, RDTDLM, WRTDLM, PAWRA, RDF, CLFILM, TIMPR

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: home21/tdllib/u351

PAWRA

PREPARES, PACKS, AND WRITES A RECORD TO A MOS-2000 RANDOM ACCESS FILE

Harry R. Glahn
December 1, 1996

PURPOSE: To prepare, pack, and write a record to a MOS-2000 random access file. It was written to be called from U351 or similar program. It does little except put the 4-word MOS ID and the plain language into IS1( ). Then it calls PACK1D and WRTDLM.

RESTRICTIONS:

PAWRA is to be used only with the MOS-2000 external file system.

COMMENTS:

While the call sequence is fairly long, the use of PAWRA can more easily be deduced by looking at U351 than at an explanation of each variable.

NONSYSTEM ROUTINES USED:

UNPKBG, PACK1D, WRTDLM

LANGUAGE: FORTRAN 77

LOCATION: moslib

U352

PUTS PACKED DATA INTO A MOS-2000 RANDOM ACCESS FILE

Harry R. Glahn
March 20, 1998

PURPOSE: U352 is used for putting TDLPACK data into a MOS-2000 external random access file. It writes the directory call letters record, if one is not already present, and optionally station WBAN numbers, elevations, latitudes, and longitudes taken from the station directory. If a call letters record is already present, U352 checks to make sure it is the same as the one that would have been written; if not, U352 stops. The call letters are always the "New" station identifiers; no provision is made for "old" call letters. The directory on the packed input need not be the same as that to be written. U352 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by a PARAMETER statement in which the variable L3264B is set to 32 or 64. Multiple input files are accommodated, as well as multiple records on a file.

CONTROL FILE INPUT: 'U352.CN' (Unit = KFILDI)

Record Type 1 - Format (A72) Run Identification

This record is used to identify the run.

Record Type 2 - Format (4I4) Control Parameters

IWRELE - Indicates whether (1) or not (0) the station elevations are to be written.

IWRWBN - Indicates whether (1) or not (0) the station WBAN numbers are to be written.

IWRLAT - Indicates whether (1) or not (0) the station latitudes are to be written.

IWRLON - Indicates whether (1) or not (0) the station longitudes are to be written.

Record Type 3 - Format (I3,4XA60) Station and Location Files

This pair of records (plus the terminator record) identifies the data sets from which the station (or location) information is obtained. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 3 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters which are to be written (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the default input file number, KFILDI = 5, in which case DIRNAM(1) is not used.

<u>DIRNAM</u>(J) - Name of file matching KFILD(J). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 4 - Format (I3,4XA60) <u>Random Access Output File</u>

This record (plus the terminator record) identifies the MOS-2000 random access output file. Records are read until the terminator KFILX( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

<u>KFILX</u> - Unit number for the random access output file.

<u>CFILX</u> - Name of the file corresponding to KFILX. (CHARACTER*60)

Record Type 5 - Format (7(A8,1X)) <u>Station List</u>

This group of records identifies the stations for the station directory to be written to the file. If KFILD(1) ≠ KFILDI (the default input file), this group is omitted, and the information is taken from another source.

<u>CCALL</u>(K) - Call letters (or other 8-character location designator) of stations (or locations) (K=1,NSTA). This list is read within subroutine RDSTAD by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations in the list are kept, but a diagnostic is furnished on unit IP(5) = KFILDO. The stations are placed in alphabetical order providing the directory is in alphabetical order. Stations not in the directory are put at the end of the list. RDSTAD also supplies the station elevations, WBAN numbers, latitudes, and longitudes in case they are to be written (see Record Type 2). The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank (b) filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. These call letters can be "new" station identifiers or "old" call letters; however, the matching new station identifiers will be the ones written to the output file. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1.

The following two record types occur in groups. Record Type 6 defines the unit number and file name for input, and Record Type 7 defines the records on that file that are to be put into the random access file. (Note that not all records on the file are necessarily written to the output file.)

Record Type 6 - Format (I3,4XA60) <u>Input Data File</u>

This record (plus the terminator record) identifies the MOS-2000 TDLPACK sequential input file. Records are read until the terminator KFILX( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 6 is read by subroutine RDSNAM.

<u>KFILP</u> - Unit number for the TDLPACK input file.

<u>PAKNAM</u> - Name of the file corresponding to KFILP. (CHARACTER*60)

Record Type 7 - Format (I9,3I10) Variable Ids

Each entry in this group of records identifies a variable to be input from file PAKNAM and written to file CFILX. Reading continues until the terminator ID(1,) = 999999 is found.

ID(J,L) - The 4-word MOS-2000 ID of the data to write (J=1,4) (L=1,NVRBL). Note that this must match exactly the IDs to process, including word 4.

CONTROL FILE INPUT: (Name read from U352.CN) (Unit = Kfild(1)

Record Type 1 - Format (7(A8,1X)) Station List

When the station list is not provided on file 'U352.CN', this group of records identifies the stations (or locations) for the station directory to be written to the file. If KFILD(1) read in Record Type 5 = KFILDI (the default input unit number), this file.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) (K=1,NSTA). This list is read within subroutine RDSTAD by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations in the list are kept, but a diagnostic is furnished on unit IP(5) = KFILDO. The stations are placed in alphabetical order providing the directory is in alphabetical order. Stations not in the directory are put at the end of the list. RDSTAD also supplies the station elevations, WBAN numbers, latitudes, and longitudes in case they are to be written (see Record Type 2). The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank (b) filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. These call letters can be "new" station identifiers or "old" call letters; however, the matching new station identifiers will be the ones written to the output file. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1.

CONTROL FILE INPUT: (Name read from U352.CN) (Unit = KFILD(2))

Record Type 1 - Format Station Locations
(A8,1XA8,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. It is needed unless KFILD(1) = KFILD(2) = KFILDI; for this highly unusual case, both the station list and the associated information are read from the input control file KFILDI. When KFILD(1) = KFILD(2) ≠ KFILDI, this control file is the same as the one above and both the station list and the associated information are taken from this file. If both files are needed (the usual case; KFILD(1) ≠ KFILD(2) ≠ KFILDI), the call letters here are matched with those in the station list and the appropriate information extracted. When this station directory is alphabetical, the final list of stations will be alphabetical no matter the order read in. No terminator is used. Most of this information is used only within subroutine RDSTAD to give information about the stations.

<u>CCALLD</u>(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list unless KFILD(1) = KFILD(2), in which case these call letters comprise the station list and at the completion of reading this file, K=1,NSTA. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

<u>NAME</u>(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

<u>NELEV</u>(K) - Elevation of the station. Format is I5.

<u>SIGNLA</u> - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

<u>XLAT</u> - Latitude in degrees. Format is F7.4.

<u>SIGNLO</u> - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

<u>LONDD</u> - Longitude in degrees west. Format is F8.4.

<u>CCALLD</u>(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

<u>IWBAN</u>(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited, except when it also constitutes the list to be kept, in which case the number of entries is limited by ND1-1. No terminator is used.

<u>EXAMPLE CONTROL FILE</u>: 'U352.CN'

An example exists as file 'U352.CN' in directory 'home21.tdllib.u352' on blizzard.

<u>OUTPUT</u>:

All output is to the default print file Unit No. KFILDO which is set by DATA statement to 12.

<u>RESTRICTIONS</u>

The random access file must have been created with U350. Note that no input flag is used to determine whether or not the call letters record is to be written. If it is to be written, it must be to a newly created file.

U352 obtains information from the station directory with the subroutine RDSTAD. A variable furnished to RDSTAD is "NEW" which in U352 is set to 1 to signify that the "new" station identifiers are to be used; no provision is made for putting "old" call letters on the output file. RDSTAD alphabetizes the stations before writing; this should help the user in getting a match between the input list in CCALL( ) and the directory already on the file.

A diagnostic indicating "INCREASE NW FROM xxx TO yyy..." will occur if the PARAMETER in the file access routines does not match MAXENT used in U350 to set up the file. This can probably be cleared up by recreating the file with U350 and using MAXENT = yyy (probably 300).

COMMENTS

Constant data may be put into the MOS-2000 external random access files from a variety of sources. Program U351 exists to serve as a template for how the process can work and expects ASCII input data. This U352 is likely the program that will be used most for the purpose. U352 uses the routine PAWRA, which in turn uses UNPKBG, PACK1D, and WRTDLM.

WBAN numbers, elevations, latitudes, and longitudes are expected to be from the MOS-2000 directory, and are packed before writing. Writing is done with replacement. This allows a change to be made without completely recreating and rewriting the file.

All other data input records are expected to be packed. The directory on this input need not be the same as the one on the constant file. The input records are unpacked, put into the proper order, packed, and written with replacement; if a duplicate is found, it is written in place, provided no more physical records are required than were originally used. If more physical records are needed, the old key is "zeroed out" and a new key and record written. This will cause fragmentation, but should be of no concern for a few replacements. Even with wholesale replacement, fragmentation may not be a problem. This can be checked by comparing the sizes of the old and new files.

Once all records that need to be processed from an input file have been read and written to the random access output, another input file name is read with Record Type 6, followed by another set of variables. This allows processing of multiple input files in one U352 run. Note that this requires a double "99" terminator after the last file name, one for the last file, and one for the empty set.

Memory should be no problem, so ND1, ND4, and ND5 can be reasonably large so that they won't have to be changed frequently.

Any stop other than STOP U352 and a time stamp in the ftn12 file is an error and the master directory is not updated. That is, no data have been added to the output random access file.

Note that while the directory on the sequential input need not be the same as an existing directory on the random access output, the input directory read in Record Type 5 <u>must</u> match the output directory (if one exists). This is for safety to a assure a "standard" list is being dealt with.

If there are stations on the input sequential packed file that are not on the random access file, that station is just, of course, omitted. If there are stations in the random access station directory not on the sequential packed input file, that station will be given the "missing" value of 9999.

The attachment to the writeup for U350 gives characteristics of programs U351, U352, and U353.

<u>SETTING UP U352</u>

Six PARAMETER variables are used:

ND1 -	The maximum number of stations or locations that can be dealt with on this run.
ND4 -	The maximum number of data records to read and write, plus 1.
ND5 -	Set to ND1.
ND7 -	Dimension of variables to use in the packing of the data. Keep this at 54.
L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
L3264W -	The number of "words" on the machine being used in 64 bits. Set to 64/L3264B.

<u>NONSYSTEM ROUTINES USED</u>

RDSNAM, RDSTAD, RDTDLM, WRTDLM, PAWRA, CLFILM, TIMPR, UNPACK, IERX, PACK1D, UNPKBG

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: home21.tdllib.u352 on blizzard

U353

PUTS PACKED DATA INTO A MOS-2000 RANDOM ACCESS FILE

Harry R. Glahn
February 15, 2000

PURPOSE: U353 is used for transferring data from one or more MOS-2000 random access files onto another MOS-2000 random access file. It writes the directory call letters record and optionally station WBAN numbers, elevations, latitudes, and longitudes taken from the station directory. The call letters are always the "New" station identifiers; no provision is made for "old" call letters. The directory on the packed inputs need not be the same as that to be written; the station directory to be written is read from the U353.CN control file or another file designated by the control file. U353 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by a PARAMETER statement in which the variable L3264B is set to 32 or 64.

CONTROL FILE INPUT: 'U353.CN' (Unit = KFILDI)

Record Type 1 - Format (A72) **Run Identification**

This record is used to identify the run.

Record Type 2 - Format (4I4) **Control Parameters**

IWRELE - Indicates whether (1) or not (0) the station elevations are to be written.

IWRWBN - Indicates whether (1) or not (0) the station WBAN numbers are to be written.

IWRLAT - Indicates whether (1) or not (0) the station latitudes are to be written.

IWRLON - Indicates whether (1) or not (0) the station longitudes are to be written.

These variables pertain only to taking the data from the station constant file and writing it; they do not pertain to copying data from one random access file to another.

Record Type 3 - Format (I3,4XA60) **Station and Location Files**

This pair of records (plus the terminator record) identifies the data sets from which the station (or location) information is obtained. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 3 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters which are to be written (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the default input file number, KFILDI = 5, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 4 - Format (I3,4XA60) **Random Access Output File**

This record (plus the terminator record) identifies the MOS-2000 random access output file. Records are read until the terminator KFILX( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILX - Unit number for the random access output file.

CFILX - Name of the file corresponding to KFILX. (CHARACTER*60)

Record Type 5 - Format (7(A8,1X)) **Station List**

This group of records identifies the stations for the station directory to be written to the file. If KFILD(1) ≠ KFILDI (the default input file), this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) (K=1,NSTA). This list is read within subroutine RDSTAD by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations in the list are kept, but a diagnostic is furnished on unit IP(5) = KFILDO. The stations are placed in alphabetical order providing the directory is in alphabetical order. Stations not in the directory are put at the end of the list. RDSTAD also supplies the station elevations, WBAN numbers, latitudes, and longitudes in case they are to be written (see Record Type 2). The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank (b) filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. These call letters can be "new" station identifiers or "old" call letters; however, the matching new station identifiers will be the ones written to the output file. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1.

The following two record types occur in groups. Record Type 6 defines the unit number and file name for input, and Record Type 7 defines the records on that file that are not to be put onto the random access file.

Record Type 6 - Format (I3,4XA60) **Input Data File**

This record (plus the terminator record) identifies an input MOS-2000 random access file. Records are read until the terminator KFILP( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 6 is read by subroutine RDSNAM. When only the terminator is found, U353 halts; this is the normal program termination.

KFILP - Unit number for the random access input file.

CFILP - Name of the file corresponding to KFILP. (CHARACTER*60)

Record Type 7 - Format (I9,3I10) Variable Ids

Each entry in this group of records identifies a variable not to be copied from file CFILP and written to file CFILX. Reading continues until the terminator ID(1,) = 999999 is found.

ID(J,L) - The 4-word MOS-2000 ID of the data to omit (J=1,4) (L=1,NVRBL). Note that this must match exactly the IDs to process, including word 4.

CONTROL FILE INPUT: (Name read from U353.CN) (Unit = KFILD(1)

When the station list is not provided on file 'U353.CN', this group of records identifies the stations (or locations) for the station directory to be written to the file. If KFILD(1) read in Record Type 5 = KFILDI (the default input unit number), this separate file is not used.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) (K=1,NSTA). This list is read within subroutine RDSTAD by subroutine RDC, which eliminates any blanks found in the input. Duplicate stations in the list are kept, but a diagnostic is furnished on unit IP(5) = KFILDO. The stations are placed in alphabetical order providing the directory is in alphabetical order. Stations not in the directory are put at the end of the list. RDSTAD also supplies the station elevations, WBAN numbers, latitudes, and longitudes in case they are to be written (see Record Type 2). The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank (b) filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. These call letters can be "new" station identifiers or "old" call letters; however, the matching new station identifiers will be the ones written to the output file. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1.

CONTROL FILE INPUT: (Name read from U353.CN) (Unit = KFILD(2))

Record Type 1 - Format Station Locations
(A8,1XA8,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. It is needed unless KFILD(1) = KFILD(2) = KFILDI; for this highly unusual case, both the station list and the associated information are read from the input control file KFILDI. When KFILD(1) = KFILD(2) ≠ KFILDI, this control file is the same as the one above and both the station list and the associated information are taken from this file. If both files are needed (the usual case; KFILD(1) ≠ KFILD(2) ≠ KFILDI), the call letters here are matched with those in the station list and the appropriate information extracted. When this station directory is alphabetical, the final list of stations will be alphabetical no matter the order read in. No terminator is used.

Most of this information is used only within subroutine RDSTAD to give information about the stations.

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list unless KFILD(1) = KFILD(2), in which case these call letters comprise the station list and at the completion of reading this file, K=1,NSTA. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited, except when it also constitutes the list to be kept, in which case the number of entries is limited by ND1-1. No terminator is used.

EXAMPLE CONTROL FILE: 'U353.CN'

An example exists as file 'U353.CN' in directory 'home21.tdllib.u353' on blizzard.

OUTPUT:

All output is to the default print file Unit No. KFILDO which is set by DATA statement to 12.

RESTRICTIONS

The random access file must have been created with U350. It must be new; no station directory is on it. If a station directory is found, an error is assumed and the program stops.

U353 obtains information from the station directory with the subroutine RDSTAD. A variable furnished to RDSTAD is "NEW" which in U353 is set to 1 to signify that the "new" station identifiers are to be used; no provision is made for putting "old" call letters on the output file. RDSTAD alphabetizes the stations before writing.

A diagnostic indicating "INCREASE NW FROM xxx TO yyy..." will occur if the PARAMETER in the file access routines does not match MAXENT used in U350 to set up the file. This can probably be cleared up by recreating the file with U350 and using MAXENT = yyy (probably 300).

COMMENTS

Constant data may be put into the MOS-2000 external random access files from a variety of sources. Program U351 exists to serve as a template for how the process can work and expects ASCII input data; U352 is used for taking data from a MOS-2000 TDLPACK sequential file and writing one or more of those records to the random access file.

WBAN numbers, elevations, latitudes, and longitudes are expected to be from the MOS-2000 directory, and are packed before writing. Writing is done with replacement. This means that if data are copied from a file that has, for example, latitudes on it, and stations are to be added, then to get the latitudes of the added stations, IWRLAT must equal 1 in Record Type 2, <u>and</u> this record must <u>not</u> be copied from the existing file else it would overwrite the one with the added stations.

All other data input records are expected to be packed in one or more MOS-2000 random access files. The directory (ies) on this (these) input(s) need not be the same as the one read in Record Type 5 (or from file KFILD(2). If the input station directory from a particular file is the same as the one to be written to the output, the desired records are copied with replacement without unpacking. If the input station directory from a particular file is <u>not</u> the same as the one to be written to the output, the input records are unpacked, put into the proper order, stations values omitted or added as necessary, packed, and written with replacement.

Once all records that need to be processed from an input file have been read and written to the random access output, another input file name is read with Record Type 6, followed by another set of variables in Record Type 7. This allows processing of multiple input files in one U353 run. Note that this requires a double "99" terminator after the last file name, one for the last file, and one for the empty set.

Memory should be no problem, so ND1, ND4, and ND5 can be reasonably large so that they won't have to be changed frequently.

Any stop other than STOP U353 and a time stamp in the ftn12 file is an error and the master directory is not updated. That is, no data have been written to the output random access file.

Note that the directory on a random access input need not be the same as the directory to be written to the random access output.

If there are stations on the input file that are not on the output file, data for those stations are just, of course, omitted. If there are stations in the station directory to be written that are not on input file, those stations' data will be given the "missing" value of 9999.

U353 can be used to inventory a file by copying it to another file unchanged.

The attachment to the writeup for U350 gives characteristics of programs U351, U353, and U353.

SETTING UP U353

Six PARAMETER variables are used:

ND1 -	The maximum number of stations or locations that can be dealt with on this run.
ND4 -	The maximum number of data records to read and write, plus 1.
ND5 -	Set to ND1.
ND7 -	Dimension of variables to use in the packing of the data. Keep this at 54.
L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
L3264W -	The number of "words" on the machine being used in 64 bits. Set to 64/L3264B.

NONSYSTEM ROUTINES USED

RDSNAM, RDSTAD, RDTDLM, WRTDLM, PAWRA, CLFILM, TIMPR, UNPACK, IERX, PACK1D, UNPKBG

LANGUAGE: FORTRAN 77

LOCATION: home21.tdllib.u353 on blizzard

U354

INVENTORIES A MOS-2000 RANDOM ACCESS FILE

Harry R. Glahn
May 25, 2000

PURPOSE: U354 is used to list the MOS IDs and other information residing on a MOS-2000 external random access file. U354 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by a PARAMETER statement in which the variable L3264B is set to 32 or 64.

CONTROL FILE INPUT: 'U354.CN' (Unit = KFILDI)

Record Type 1 - Format (A72) **Run Identification**

This record is used to identify the run.

Record Type 2 - Format (I4) **Control Parameter**

IPCALL - Indicates whether not the station identifiers (call letters) on the input file are to be written for printing.
0 = Do no write.
1 = Write with Format(10A8)--10 identifiers per line.
2 = Write with Format(A8)--one identifier per line.

Record Type 3 - Format (I3,4XA60) **Input Random Access File**

This record (plus the terminator record) identifies an input MOS-2000 random access file. Records are read until the terminator KFILP( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 3 is read by subroutine RDSNAM. When only the terminator is found, U354 halts.

KFILP - Unit number for the random access input file.

CFILP - Name of the file corresponding to KFILP. (CHARACTER*60)

EXAMPLE CONTROL FILE: 'U354.CN'

An example exists as file 'U354.CN' in directory 'home21.tdllib.u354' on blizzard.

OUTPUT:

All output is to the default print file Unit No. KFILDO which is set by DATA statement to 12.

RESTRICTIONS

The random access file must have been created with U350.

A diagnostic indicating "INCREASE NW FROM xxx TO yyy..." will occur if the PARAMETER in the file access routines does not match MAXENT used in U350 to set up the file. This can probably be cleared up by recreating the file with U350 and using MAXENT = yyy (probably 300).

COMMENTS

None.

SETTING UP U354

Five PARAMETER variables are used:

ND1 -	The maximum number of stations or locations that can be dealt with on this run.
ND5 -	Set to ND1.
ND7 -	Dimension of variables to use in the packing of the data. Keep this at 54.
L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
L3264W -	The number of "words" on the machine being used in 64 bits. Set to 64/L3264B.

NONSYSTEM ROUTINES USED

RDSNAM, RDTDLM, TIMPR, UNPACK, IERX, UNPKBG

LANGUAGE: FORTRAN 77

LOCATION: home21.tdllib.u354 on blizzard

U361

PUTS PACKED GRIDPOINT DATA INTO A MOS-2000 RANDOM ACCESS FILE

Harry R. Glahn
March 15, 2002

PURPOSE: U361 is used for putting TDLPACK gridpoint data into a MOS-2000 external random access file. U361 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by a PARAMETER statement in which the variable L3264B is set to 32 or 64. Multiple grids on an input file are accommodated, but only one input file and one output file can be used per run. Input data are ASCII.

CONTROL FILE INPUT: 'U361.CN' (Unit = KFILDI)

Record Type 1 - Format (A72) **Run Identification**

This record is used to identify the run.

Record Type 2 - Format (I3,4XA60) **Random Access Output File**

This record (plus the terminator record) identifies the MOS-2000 random access output file. Records are read until the terminator KFILX( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 2 is read by subroutine RDSNAM.

KFILX - Unit number for the random access output file.

CFILX - Name of the file corresponding to KFILX. (CHARACTER*60)

Record Type 3 - Format (I3,4XA60) **Input Data File**

This record (plus the terminator record) identifies the ASCII input file. Records are read until the terminator KFILP( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 3 is read by subroutine RDSNAM.

KFILP - Unit number for the ASCII input file.

CFILP - Name of the file corresponding to KFILP. (CHARACTER*60)

CONTROL AND DATA FILE INPUT: (Name read from 'U361.CN') (Unit = KFILP)

Record Types 1, 2, and 3 occur in trios. Record Type 1 identifies the data, Record Type 2 contains information for packing, and Record Type 3 contains the ASCII data.

Record Type 1 - Format (I9,3I10) **Variable Ids**

ID(J) - The 4-word MOS-2000 ID of the data to write (J=1,4). ID(1) = 0 signifies program termination.

Record Type 2 - Format (4(I10/),5(F10.0/),I10/4(F10.0/),I10/A32) Variable Ids

ISCALE - The scaling parameter for packing. The data will be packed to the resolution 10**ISCALE, rounded.

NPROJ - The number of the map projection to which the data refer (5 = polar stenographic). Only polar stereographic, Lambert, and Mercator are accommodated.

NX - The X extent of the grid. This is the number of values to read per record in Record Type 3.

NY - The Y extent of the grid. This is the number of records to read per grid in Record Type 3.

ALAT - The North latitude of the lower left corner of the grid in degrees. Use 3 (not more unless zeros) decimal places. South latitude would be negative.

ALON - The West longitude of the lower left corner of the grid in degrees. Use 3 (not more unless zeros) decimal places.

XMESH - The mesh length of the grid (95.25 = 1/4 Bedient).

ORIENT - The grid orientation in degrees West longitude (e.g., 105 for the AVN archive). Use zero for Mercator.

XLAT - The latitude where MESH applies (60 for the AVN archive).

IPRINT - 1 if the grid is to be gridprinted; 0 otherwise.

CINT - The contour interval for gridprinting. Not used when CINT = 0.

ORIGIN - The contour origin for gridprinting. Not used when CINT = 0.

SMULT - Multiplicative factor for gridprinting. Not used when CINT = 0.

SADD - Additive factor for gridprinting. Not used when CINT = 0.

NCHAR - Number of characters in PLAIN for packing with the grid.

PLAIN - Plain language to pack with the grid, NCHAR characters, maximum of 32.

Record Type 3 - Format (10F12.0) Data

FD1(J) - The data read row by row, NX*NY values, NX values per record.

EXAMPLE CONTROL FILE: 'U361.CN'

An example exists as file 'U361.CN' in directory home21/glahn/gmos_204/u350 on blizzard.

OUTPUT:

All diagnostic output is to the default print file Unit No. KFILDO which is set by DATA statement to 12.

RESTRICTIONS

The random access file must have been created with U350.

A diagnostic indicating "INCREASE NW FROM xxx TO yyy..." will occur if the PARAMETER in the file access routines does not match MAXENT used in U350 to set up the file. This can probably be cleared up by recreating the file with U350 and using MAXENT = yyy (probably 300).

COMMENTS

Constant data may be put into the MOS-2000 external random access files from a variety of sources. Program U351 exists to serve as a template for how the process can work for vector data and expects ASCII input data. U352 is likely the program that will be used most for vector data; it uses the routine PAWRA, which in turn uses UNPKBG, PACK1D, and WRTDLM. U361 is for writing gridpoint data and uses PACK2D, PAWRAG, and WRTDLM.

The records are packed and written with replacement; if a duplicate is found, it is written in place, provided no more physical records are required than were originally used. If more physical records are needed, the old key is "zeroed out" and a new key and record written. This will cause fragmentation, but should be of no concern for a few replacements. Even with wholesale replacement, fragmentation may not be a problem unless records re very large (e.g., 5 km data over the U.S.). This can be checked by comparing the sizes of the old and new files.

Any stop other than STOP U361 and a time stamp in the ftn12 file is an error and the master directory is not updated. That is, no data have been added to the output random access file.

SETTING UP U361

Four independent PARAMETER variables are used:

ND2X3 -	The maximum size of the grids on this run. Individual dimensions do not matter as long as the grid is $\le$ ND2X3.
ND5 -	Set = ND2X3.
ND7 -	Dimension of variables to use in the packing of the data. Keep this at 54.
L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).

NONSYSTEM ROUTINES USED

RDSNAM, WRTDLM, PAWRAG, CLFILM, TIMPR, IERX, PACK2D, PRTGR

LANGUAGE: FORTRAN 77

LOCATION: home21/glahn/gmos_204/u350 on blizzard

PAWRAG

PREPARES, PACKS, AND WRITES A GRIDPOINT RECORD TO A MOS-2000 RANDOM ACCESS FILE

Harry R. Glahn
March 15, 2002

PURPOSE: To prepare, pack, and write a gridpoint record to a MOS-2000 random access file. It was written to be called from U361 or similar program. It puts the metadata into the form for TDLPACK. Then it calls PACK2D and WRTDLM.

RESTRICTIONS:

PAWRAG is to be used only with the MOS-2000 external file system.

COMMENTS:

While the call sequence is fairly long, the use of PAWRAG can more easily be deduced by looking at U361 than at an explanation of each variable.

NONSYSTEM ROUTINES USED:

UNPKBG, PACK2D, WRTDLM

LANGUAGE: FORTRAN 77

LOCATION: moslib

U365

INTERPOLATES GRIDDED DATA FROM ONE GRID TO ANOTHER

David E. Rudack
Paul J. Dallavalle
Harry R. Glahn
September 1, 2002

PURPOSE: U365 is designed to interpolate data from an input grid to an output grid. The output grid usually possesses different grid characteristics than the input grid. U365 supports North polar stereographic, Lambert conformal and Mercator projections. The input grid's map projection need not match the output map projection grid. The input data are gridded (e.g. numerical models, satellite and radar) and are packed in a GRIB-like TDLPACK format. Because the packed data are self describing, the gridded data may originate from any data source, the number being essentially unlimited. U365 uses a driver DRU365 so that dimensions of variables may be tailored by PARAMETER statements by the user without requiring a separate copy of the program U365 for every application. U365 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. The output of U365 is a TDLPACK dataset file containing values at gridpoints on the user defined output grid. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U365.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU365.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U365" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U365 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'DER2' and IP(2) = 40, the file name for Unit No. 40 = 'U365DER240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U365 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. Values have been defined as indicated below for values of J:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is

read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU365 DATA statement = 12. Changing the default unit number allows multiple runs of U365 or other programs within the same directory without overwriting.

(2) =	The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) =	The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(6) =	The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.
(7) =	The variable ID list in summary form. If there are errors, the predictor list will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) =	The variable ID list in summary form after reordering low to high. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)
(9) =	The variable ID list in summary form after reordering. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the predictor constant file on unit KFILCP (see below).
(13) =	Gridded fields. When the variable list indicates gridded values are to be written for viewing (JP(1,)>0), they will be written to unit IP(13).
(16) =	Statements indicating the writing of data to a sequential file.

For checkout, it may be advisable to set all these values to zero or the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (5(I10/),2(F10.4/),F10.5/,2(F10.4/) Control Parameters

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

JSTOP - The total number of errors that will be tolerated before the program halts (See comments section).

<u>INCCYL</u> - The increment in hours between date/times that are put into IDATE( ) (see Record Type 5) as a result of date spanning in subroutine DATPRO. This variable is also used in determining what data are to be saved in the Internal MOS-2000 Storage System. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL to the first date/time in IDATE( ).

<u>MPROJ</u> - The map projection number of the output grid. (3 for Northern hemispheric Lambert, 5 for North Polar stereo-graphic and 7 for Mercator projection)

<u>NX</u> - The number of gridpoints in the x-direction on the output grid.

<u>NY</u> - The number of gridpoints in the y-direction on the output grid.

<u>ALAT</u> - The latitude of the lower left corner gridpoint of the output grid. The range is 0 to 90 degrees in the northern hemisphere.

<u>ALON</u> - The longitude of the lower left corner gridpoint of the output grid. West longitude is 0 through 180 and east longitude is 180 through 359.9999.

<u>MESH</u> - Mesh length at XLAT of the output grid expressed in kilometers.

<u>ORIENT</u> - Grid orientation (i.e. the vertical longitude) of the output grid in degrees. West longitude is 0 through 180 and east longitude is 180 through 359.9999.

<u>XLAT</u> - Latitude at which the grid length (MESH) applies. The range is 0 to 90 degrees in the northern hemisphere. This is also the latitude of tangency in case of the Lambert map projection.

Record Type 4 - Format (I3,4XA60) **<u>Date List File</u>**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

<u>KFILDT</u> - Unit number for the file containing the input date list.

<u>NAMDT</u> - Name of file where this date list resides. When KFILDT = KFILDI, NAMDT is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) **<u>Date List</u>**

This group of records determines the date/times for which interpolated output is wanted. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

<u>IDATE</u>(J)- Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of

hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8. Data for the first date in the list must be available or U365 stops. The date/times put into Record Type 5 must be such that all could be arrived at by successively adding INCCYL (read in Record Type 1) to the first date/time in IDATE( ).

Record Type 6 - Format (2I3,1XA60) **Gridded Input Data Files**

This group of records identifies the sequential data sets from which the gridpoint data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

KFILIN(J) - Unit number for the input gridpoint file(s). KFILIN( )is not restricted to the conventional values used for gridded data in the MOS-2000 system. Only values of 97, 99, and those conflicting with an IP( ) value or another file unit number other than KFILIN( ) in the .CN file may not be used.

NAMIN(J) - Name of file where this model input resides. (CHARACTER*60)

Record Type 7 - Format (I3,4XA60) **Gridded TDLPACK Output File**

This record (plus the terminator record) identifies the gridded output file. Records are read until the terminator KFILGO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 7 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

KFILGO - Unit number for the output file.

OUTGRD - Name of file where this output is to reside. (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) **Variable File**

This record (plus the terminator record) identifies the file from which the variable ID's are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) Variable Constants File

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description and gridprinting information, the latter of which is not operable in U365.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 9 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 10 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2) Variable List

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 10 is read by subroutine RDPRED. Upon completion of reading this record type, N=1,NVRBLS.

ID(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only) (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

JP(J,N) - For each variable N, JP(,N) indicates print or no print when ≠ 0 or = 0, respectively, for gridpoint values (J=1). JP(2,N) controls the interpolation type to be performed on the output grid. Values range from 1 to 4. (See comments for details.) JP(3,N) is not used.

CONTROL FILE INPUT: (Name read from U365.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) Date List

When the dates are not provided in file U365.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU365), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U365.CN) (Unit = KFILP)

Record Type 1 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2) **Variable List**

When the variables to use are not in file 'U365.CN', this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted, and the variable list is taken from input file KFILDI. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDPRED. Upon completion of reading this record type, N=1,NVRBLS. See Record Type 13, file 'U365.CN', unit KFILDI, for other details; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U365.CN) (Unit = KFILCP)

Record Type 1 - Format
(3(I9,1X),I3,2XA32,I3,F11.4,F10.4,F9.2,F8.2,2XA12) **Variable Constants**

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all MOS-2000 users and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be very unusual for that to be the case. Note that the format matches that for a file input to other programs such as U600, U660, and U850.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALT - When IDTEMP(1) matches an ID(1,N), ISCALT is stored in ISCALD(N). This variable is the scaling constant (power of 10) to use when packing the interpolated gridded data.

SMULTT - When IDTEMP(1) matches an ID(1,N), SMULTT is stored in SMULT(N). This variable is the multiplicative factor (power of 10) to use when gridprinting the gridpoint data. **(Not used)**

SADDT - When IDTEMP(1) matches an ID(1,N), SADDT is stored in SADD(N). This variable is the additive factor to use when gridprinting the gridpoint data. **(Not used)**

CONTT - When IDTEMP(1) matches an ID(1,N), CONTT is stored in CINT(N). This variable is the contour interval to use when gridprinting the data. It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT. **(Not used)**

ORIGNT - When IDTEMP(1) matches an ID(1,N), ORIGNT is stored in ORIGIN(N). This variable is the contour origin to use when gridprinting the data It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT. **(Not used)**

UNITST - When IDTEMP(1) matches an ID(1,N), UNITST is stored in UNITST(N). These characters define the units of the data after application of SMULTT and SADDT and are used in the visual inspection of the data in gridprinted maps. (CHARACTER*12) **(Not used)**

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDPRED, much of it associated with the plain language description. See Chapter 4, Variable Identification, for details.

DATA INPUT:

All archived model data input to U365 will be in the MOS-2000 TDLPACK format read from sequential files. The unit numbers and file names are provided in KFILIN(J) and NAMIN(J) (J=1,NUMIN), respectively. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG.

SEQUENTIAL GRIDPOINT DATA

One or more sources of gridpoint data are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U365.CN', Record Type 6. Each file is closed when an EOF is reached.

The definition of the grid to which the data in each record pertain is contained in that same record. The input grid characteristics of a particular variable must remain constant throughout the run of U365. Normally, ND11 (see "Setting up the driver DRU365") will be $\geq$ the number of input grid sources (unit numbers).

DATA OUTPUT

There is one form of data output, besides the diagnostics and other information on Units KFILDO and IP( ).

SEQUENTIAL BINARY MOS-2000 FORMAT

All data interpolated to the output grid are packed in the MOS-2000 TDLPACK format. They are packed with subroutine PACKG and all its associated subroutines. All data are written to the dataset whose name has been provided to OUTGRD and unit number to KFILGO from the control file 'U365.CN', Record Type 7, unless KFILGO = 0, in which case data are not output.

<u>EXAMPLE CONTROL FILE</u>: 'U365.CN'

An example exists as file 'U365.CN' in directory '/home21/tdllib/dru365' on the GDP. The easiest way to set up a run is to take an existing control file and modify it; <u>DO NOT START FROM SCRATCH</u>.

<u>OUTPUT</u>:

Output that can be printed can be put onto one or more of several files as described above under control file 'U365.CN', Record Type 1 and the definition of KFILDO in the driver DRU365. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control without jeopardizing data.

<u>RESTRICTIONS</u>

Since U365 does not follow the normal procedure when opening sequential input files, there is no unit number restriction imposed upon the user. The user may use any desired unit number (even values between 44-49, which are normally reserved for external random access files). This statement remains true as long as the input file’s unit number(s) do not conflict with an IP( ) value or another other unit number for a different record type.

All sequential input files **must** be placed in the control file in chronological order. In addition, all variables that are to be processed for a specific date/time **must** be on the same file. They may not be spread over two or more files.

The grid characteristics of the input and output grids need not be identical. For example, the input grid may have a Lambert projection with a true mesh length of 80 km at 60 degrees north while the output grid may be North polar stereographic having a true mesh of 10 km at 40 degrees north. A host of different combinations are provided for, all of which are set at the top of the U365.CN file.

In the situation where a gridpoint on the output grid lies outside the boundaries of the input grid, the datum (on the output grid) is assigned a value of 9999.

The options such as the look back feature, the creation of binaries, the smoothing of data and the computation of other fields that are found in other MOS-2000 programs, are not present in U365. Removing these options from U365 was done to reduce its functionality to a simple grid-to-grid interpolation program.

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. North polar stereographic, Lambert and Mercator projection gridpoint data are accommodated. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. It is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

COMMENTS

U365 creates an output grid in a two-step process. First, the coordinates of the output grid are calculated relative to the input grid coordinates. Second, the data is the then interpolated from the input grid to the output grid using one of four possible types of interpolating schemes. The user controls which type of interpolation process to invoke by setting JP(“2",N) to the appropriate value.

JP(2,N) = 1 = Biquadratic interpolation uses subroutine **INTRPA**
JP(2,N) = 2 = Bilinear interpolation uses subroutine **INTRPB**
JP(2,N) = 3 = Precipitation amount interpolation uses subroutine **INTRP**
JP(2,N) = 4 = Nearest neighbor interpolation uses subroutine **INTRPC**

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) JSTOP (see Record Type 3). It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities.

SETTING UP THE DRIVER DRU365

The preparation of the driver for a particular U365 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the IBM).
ND2 -	ND2*ND3 is the maximum size of the grid that can be dealt with. ND2 and ND3 are set separately to highlight the possible dimension of the grids. However, in the called routines, the size is only limited by the product, not each dimension individually.
ND3 -	See ND2.
ND4 -	The maximum number of variables used in a run of U365.
ND5 -	Dimension of IPACK( ), IWORK( ), and DATA( ). ND2X3 and ND5 should be set in the driver DRU365:

ND6 - Maximum number of all sequential file input sources (e.g., models) that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND11 - The maximum number of **input** grids that can be dealt with. Recall that for each run of U365 one and only one **output** grid is generated.

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND2*ND3). Every effort has been made so that the variables will not be overflowed if values too small are used; however, some danger may still exist.

The "startup" control file "U365.CN" is opened in DRU365, so that this aspect of U365 can be rather easily changed without recompiling or having separate version of the main (sub)routine U365. This will be different on the IBM where unit/file assignments are made differently, and possibly when U365 is run in batch mode.

NONSYSTEM ROUTINES USED

Use the load line in file '/home21/tdllib/dru365', dataset 'u365.com', along with the libraries '/home21/tdllib/u365lib' and '/home21/tdllib/moslib', all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: U365lib. The driver is in /home21/tdllib/dru365.

CKIDS

CHECKS INTERNAL CONSISTENCY OF MOS-2000 ID

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To check for consistency certain information within a predictor ID. The following items are checked and the status return IER indicates the problem, unless IDPARS(4) = 80, 81, or 82:

- o When NTYPVR = 1, IDPARS(3) must = 0, 1, or 5 (legitimate binary predictor indicator). Else, IER = 45.
- o When NTYPVR ≠ 1, IDPARS(3) must = 0, 1, 2, 3, or 5 (legitimate binary predictor or predictand indicator). Else, IER = 45.
- o When IDPARS(1) = 003, and when IDPARS(2) = 205, 210, 220, 235, 240, or 250, then IDPARS(13) must = 3 (precipitation amount fields should have an interpolation value of 3). Else, IER = 43.
- o When IDPARS(1) ≠ 003, IDPARS(13) must not = 3 (non-precipitation fields shouldn't have precipitation interpolation). Else, IER = 44.
- o When IDPARS(13) ≠ 1, 2, or 3, then IDPARS(1) must be 010 (trig functions), in the range 400-499 (constants), or in the range 700-799 (observations). Else, IER = 46.
- o When IDPARS(3) = 5 (grid binary), IDPARS(13) must = 0, 1, or 2. Else, IER = 45.

As other values of the parameters in IDPARS are added (e.g., another interpolation routine in which IDPARS(13) may have a value other than 1, 2, or 3), then this routine will need to be updated. An effort is made to find all inconsistencies, but if there is more than one, IER can only indicate the last one found. When IDPARS(4) = 80, 81, or 82 (AEV data), no checks are made.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

CALL CKIDS(KFILDO,N,ID,IDPARS,NTYPVR,IER)

<u>KFILDO</u>	- Unit number of output (print) file. Diagnostics, including error conditions are written here. (INPUT)
<u>N</u>	- Predictor number. Used for diagnostic print only. (INPUT)
<u>ID</u>(J)	- The MOS-2000 predictor ID (J=1,4). Used for diagnostic print only. (INPUT)
<u>IDPARS</u>(J)	- The parsed, individual components of the predictor ID (J=1,15). It is on these values that the checking is done. (INPUT)

<u>NTYPVR</u>	- Type of variable. 1 = Predictor. 2 = Predictand. (INPUT)
<u>IER</u>	- Status return. 0 = Good return. For other values, see Purpose (OUTPUT)

<u>EXAMPLE</u>

```
PARAMETER (ND4=40)
DIMENSION ID(4,ND4)
DIMENSION IDPARS(15,ND4)
DATA KFILDO/6/
...
N=1
...

CALL CKIDS(KFILDO,N,ID(1,N),IDPARS(1,N),1,IER)
```

The unit number for diagnostics is 6. The MOS-2000 predictors (NTYPVR = 1) must be in an array ID(,) and the corresponding array IDPARS(,). In this example, ND4 = 40 predictors are possible. The second value in the call sequence, N, indicates the first predictor in the list will be checked (i.e., IDPARS(,1)). IER = 0 unless there is a possible error found.

<u>OUTPUT</u>:

Diagnostic messages will be written to Unit No. KFILDO.

<u>RESTRICTIONS</u>:

The checking applies to the capabilities of U201 and U600 as of the header date. Other programs may have other checking needs and restrictions. CKIDS is not appropriate for AEV data, where IDPARS(4) is 80, 81, or 82.

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

BINARY

MAKES A BINARY VARIABLE

Harry R. Glahn
May 1, 1994

PURPOSE: A threshold is used to make a binary variable. BINARY operates on a 1-dimensioned array and is a counterpart to GRIDB which operates on a 2-dimensioned array. The input variable IDPARS controls how the threshold is used, and sections of code can be added to prepare variables which are not strictly binary.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL BINARY(KFILDO,ID,IDPARS,THRESH,SDATA,NSTA,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

ID(J) - 4-word MOS-2000 ID (J=1,4). Used only for diagnostic print. (INPUT)

IDPARS - A single digit controlling how the "binary" is produced.
- 1 = SDATA( ) becomes 1 when the input value SDATA( ) GE THRESH, and 0 otherwise.
- 6 = SDATA( ) becomes -9999 when the input value SDATA( ) GT THRESH, and 0 otherwise.
- 7 = SDATA( ) becomes -9999 when the input value SDATA( ) LT THRESH, and 0 otherwise.
- 8 = SDATA( ) becomes 9999 when the input value SDATA( ) GT THRESH, and 0 otherwise.
- 9 = SDATA( ) becomes 9999 when the input value SDATA( ) LT THRESH, and 0 otherwise.

1 is used for the normal binary predictor. 8 and 9 are for "matching" variables in U850 verification, and may be useful in other programs for stratification. 6 and 7 are used for "matching or" variables in U850 verification. Sections of code can be added to devise other binary-like variables corresponding to other values of IDPARS (e.g., SDATA( ) remains unchanged unless it is GE THRESH, then it becomes 0). (INPUT)

THRESH - The threshold value to use in making the binary variable. (INPUT)

SDATA(K) - The input variable to transform to a binary (K=1,NSTA). Normally, these are values at stations. On exit, SDATA( ) contains the transformed variable. (INPUT-OUTPUT)

NSTA - The number of values in SDATA( ), normally the number of stations. (INPUT)

```
    IER       - Status return.
                 0 = Good return.
                60 = Threshold has the missing value = 9999.
                61 = Value of IDPARS not used in this routine.
                62 = Reserved.
                63 = Reserved.
                When IER NE 0, SDATA( ) is set to 9999.  (OUTPUT)
```

EXAMPLE

```
    DIMENSION ID(4),SDATA(NSTA)
    DATA KFILDO/6/
    ...

    CALL BINARY(KFILDO,ID,1,50.,SDATA,NSTA,IER)
```

The unit number for diagnostics is 6. The MOS-2000 4-word ID is in ID( ). The variable SDATA( ) is given the value 1 when the input value is GE 50 and given the value 0 otherwise. Note that THRESH = 50 is REAL, and that IDPARS = 1 is INTEGER. A return on IER = 0 means BINARY operated as expected; any other value for IER indicates all values in SDATA( ) = 9999 (J=1,NSTA).

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

SMTH5

SMOOTHS A FIELD BY A 5-POINT SMOOTHER

Harry R. Glahn
May 1, 1994

PURPOSE: A 2-dimensional array is smoothed with a 5-point smoother. Each resulting smoothed point is the average of itself and the closest four points. A linear gradient is assumed for the outer two rows and two columns. No error checking is done and a status return is not provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL SMTH5(KFILDO,P,FD,NX,NY)
```

KFILDO	- Unit number of output (print) file. (INPUT)
P(IX,JY)	- The input variable to smooth (IX=1,NX; JY=1,NY). Normally, these are values at gridpoints. On exit, P( ) contains the smoothed variable. (INPUT-OUTPUT)
FD(IX,JY)	- A work array (IX=1,NX; JY=1,NY). (INTERNAL)
NX,NY	- The number of gridpoints in the IX and JY directions, respectively. The dimensions of P(,) and FD(,). (INPUT)

EXAMPLE

```
DIMENSION P(NX,NY),FD(NX,NY)
DATA KFILDO/6/
...

CALL SMTH5(KFILDO,P,FD,NX,NY)
```

The unit number for diagnostics is 6. The variable P(,) is smoothed with the simple 5-point smoother. FD(,) must be provided as a work array.

OUTPUT:

None. KFILDO is provided in case output is added.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

SMTH9

SMOOTHS A FIELD WITH A 9-POINT SMOOTHER

Harry R. Glahn
May 1, 1994

PURPOSE: A 2-dimensional array is smoothed with a 9-point smoother. Each resulting smoothed point is the average of itself and the closest eight points. A linear gradient is assumed for the outer two rows and two columns. This assumption results in smoothing only along an edge, because the extrapolated value outside the grid balances the value just inside the grid. The corner points are unchanged. No error checking is done and a status return is not provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL SMTH9(KFILDO,P,FD,NX,NY)
```

KFILDO - Unit number of output (print) file. (INPUT)

P(IX,JY) - The input variable to smooth (IX=1,NX; JY=1,NY). Normally, these are values at gridpoints. On exit, P( ) contains the smoothed variable. (INPUT-OUTPUT)

FD(IX,JY) - A work array (IX=1,NX; JY=1,NY). (INTERNAL)

NX,NY - The number of gridpoints in the IX and JY directions, respectively. The dimensions of P(,), and FD(,). (INPUT)

EXAMPLE

```
DIMENSION P(NX,NY),FD(NX,NY)
DATA KFILDO/6/
...

CALL SMTH9(KFILDO,P,FD,NX,NY)
```

The unit number for diagnostics is 6. The variable P(,) is smoothed with the simple 9-point smoother. FD(,) is provided as a work array.

OUTPUT:

None. KFILDO is provided in case output is added.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

SMTH25

SMOOTHS A FIELD WITH A 25-POINT SMOOTHER

Harry R. Glahn
May 1, 1994

PURPOSE: A 2-dimensional array is smoothed with a 25-point smoother. Each resulting smoothed point is the average of itself and the 24 points in a square box around it (i.e., all points within a square 5 points on a side, with the point to be changed in the center, are averaged. A linear gradient is assumed for the outer rows and columns (two on each side). Some accommodation is made along the edges. The corner points are unchanged. No error checking is done and a status return is not provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL SMTH25(KFILDO,P,FD,NX,NY)
```

KFILDO	- Unit number of output (print) file. (INPUT)
P(IX,JY)	- The input variable to smooth (IX=1,NX; JY=1,NY). Normally, these are values at gridpoints. On exit, P( ) contains the smoothed variable. (INPUT-OUTPUT)
FD(IX,JY)	- A work array (IX=1,NX; JY=1,NY). (INTERNAL)
NX,NY	- The number of gridpoints in the IX and JY directions, respectively. The dimensions of P(,) and FD(,). (INPUT)

EXAMPLE

```
DIMENSION P(NX,NY),FD(NX,NY)
DATA KFILDO/6/
...

CALL SMTH25(KFILDO,P,FD,NX,NY)
```

The unit number for diagnostics is 6. The variable P(,) is smoothed with the simple 25-point smoother. FD(,) is provided as a work array.

OUTPUT:

None. KFILDO is provided in case output is added.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

INTRPA

INTERPOLATES QUADRATICALLY INTO A GRID TO POINT LOCATIONS

Harry R. Glahn
May 1, 1994

PURPOSE: Interpolates to points, usually station locations, from a 2-dimensional grid. Bi-quadratic interpolation is used where possible, bi-linear interpolation is used in outside grid interval, and linear extrapolation is used if the point is outside the grid. No error checking is done and a status return is not provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL INTRPA(KFILDO,P,NX,NY,DIR,ND1,NSTA,SDATA)
```

KFILDO	- Unit number of output (print) file. (INPUT)
P(IX,JY)	- The input grid to interpolate into (IX=1,NX; JY=1,NY). Normally, these are values at gridpoints. (INPUT)
NX,NY	- The number of gridpoints in the IX and JY directions, respectively. The dimensions of P(,). (INPUT)
DIR(K,J)	- The IX (J=1) and JY (J=2) positions on the grid for station K (K=1,NSTA). (INPUT)
ND1	- The maximum number of stations whose locations are in DIR(,). This is the first dimension of DIR(,). (INPUT)
NSTA	- The number of stations whose locations are in DIR(,) and whose interpolated values are to be returned. It is used as the dimension of SDATA( ). (INPUT)
SDATA(K)	- The interpolated values for all stations are returned in SDATA( ) (K=1,NSTA). (OUTPUT)

EXAMPLE

```
DIMENSION P(NX,NY)
DIMENSION DIR(ND1,2),SDATA(ND1)
DATA KFILDO/6/
...

CALL INTRPA(KFILDO,P,NX,NY,DIR,ND1,NSTA,SDATA)
```

The unit number for diagnostics is 6. NSTA values are to be returned in SDATA( ) interpolated from the grid in P(,). The dimensions of the grid are NX and NY in the IX and JY directions, respectively. The values returned in SDATA( ) correspond in order to the station locations given in DIR(K,J) in terms of the IX (J=1) and JY (J=2) locations.

OUTPUT:

None. KFILDO is provided in case output is added.

RESTRICTIONS:

NSTA must be LE ND1. Each value in DIR(,) should, of course, indicate a position within the grid or reasonably close (one or two gridlenghts) to it or a poor "interpolated" value may result.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

INTRPB

INTERPOLATES LINEARLY INTO A GRID TO POINT LOCATIONS

Harry R. Glahn
May 1, 1994

<u>PURPOSE</u>: Interpolates to points, usually station locations, from a 2-dimensional grid. Bi-linear interpolation is used, and linear extrapolation is used if the point is outside the grid. If any one of the 4 points needed for the interpolation is missing (9999), the returned value is set to missing.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
CALL INTRPB(KFILDO,P,NX,NY,DIR,ND1,NSTA,SDATA)
```

<u>KFILDO</u> - Unit number of output (print) file. (INPUT)

<u>P</u>(IX,JY) - The input grid to interpolate into (IX=1,NX; JY=1,NY). Normally, these are values at gridpoints. (INPUT)

<u>NX,NY</u> - The number of gridpoints in the IX and JY directions, respectively. The dimensions of P(,). (INPUT)

<u>DIR</u>(K,J) - The IX (J=1) and JY (J=2) positions on the grid for station K (K=1,NSTA). (INPUT)

<u>ND1</u> - The maximum number of stations whose locations are in DIR(,). This is the first dimension of DIR(,). (INPUT)

<u>NSTA</u> - The number of stations whose locations are in DIR(,) and whose interpolated values are to be returned. It is used as the dimension of SDATA( ). (INPUT)

<u>SDATA</u>(K) - The interpolated values for all stations are returned in SDATA( ) (K=1,NSTA). (OUTPUT)

<u>EXAMPLE</u>

```
DIMENSION P(NX,NY)
DIMENSION DIR(ND1,2),SDATA(ND1)
DATA KFILDO/6/
...

CALL INTRPB(KFILDO,P,NX,NY,DIR,ND1,NSTA,SDATA)
```

The unit number for diagnostics is 6. NSTA values are to be returned in SDATA( ) interpolated from the grid in P(,). The dimensions of the grid are NX and NY in the IX and JY directions, respectively. The values returned in SDATA( ) correspond in order to the station locations given in DIR(K,J) in terms of the IX (J=1) and JY (J=2) locations.

<u>OUTPUT</u>:

None. KFILDO is provided in case output is added.

<u>RESTRICTIONS</u>:

NSTA must be LE ND1. Each value in DIR(,) should, of course, indicate a position within the grid or reasonably close (one or two gridlengths) to it or a poor "interpolated" value may result.

<u>COMMENTS</u>:

All 4 points needed for the interpolation must be non-missing, or a missing value is returned.

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: MOS-2000 Library

INTRP

INTERPOLATES INTO A PRECIPITATION AMOUNT FIELD TO POINT LOCATIONS

Harry R. Glahn
May 1, 1994

PURPOSE: Interpolates to points, usually station locations, from a 2-dimensional grid containing precipitation amounts or some similar variable. Bi-linear interpolation is used, and linear extrapolation is used if the point is outside the grid. However, before interpolation the field is modified so that the zero line is preserved at approximately half way between a point with precipitation (a non-zero value) and a point with no precipitation (a zero value). No error checking is done and a status return is not provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL INTRP(KFILDO,P,FD,NX,NY,DIR,ND1,NSTA,SDATA)
```

KFILDO	- Unit number of output (print) file. (INPUT)
P(IX,JY)	- The input grid to interpolate into (IX=1,NX; JY=1,NY). Normally, these are values at gridpoints. (INPUT)
FD(IX,JY)	- A work array (IX=1,NY; JY=1,NY). (INTERNAL)
NX,NY	- The number of gridpoints in the IX and JY directions, respectively. The dimensions of P(,) and FD(,). (INPUT)
DIR(K,J)	- The IX (J=1) and JY (J=2) positions on the grid for station K (K=1,NSTA). (INPUT)
ND1	- The maximum number of stations whose locations are in DIR(,). This is the first dimension of DIR(,). (INPUT)
NSTA	- The number of stations whose locations are in DIR(,) and whose interpolated values are to be returned. It is used as the dimension of SDATA( ). (INPUT)
SDATA(K)	- The interpolated values for all stations are returned in SDATA( ) (K=1,NSTA). (OUTPUT)

EXAMPLE

```
DIMENSION P(NX,NY),FD(NX,NY)
DIMENSION DIR(ND1,2),SDATA(ND1)
DATA KFILDO/6/
...

CALL INTRP(KFILDO,P,FD,NX,NY,DIR,ND1,NSTA,SDATA)
```

The unit number for diagnostics is 6. NSTA values are to be returned in SDATA( ) interpolated from the grid in P(,). The dimensions of the grid are NX and NY in the IX and JY directions, respectively. The values returned in SDATA( ) correspond in order to the station locations given in DIR(K,J) in terms of the IX (J=1) and JY (J=2) locations.

OUTPUT:

None. KFILDO is provided in case output is added.

RESTRICTIONS:

NSTA must be LE ND1. Each value in DIR(,) should, of course, indicate a position within the grid or reasonably close (one or two gridlenghts) to it or a poor "interpolated" value may result.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

TRANS

TRANSFORMS A VARIABLE

Harry R. Glahn
May 1, 1994

PURPOSE: To transform a variable into another one. Up to 9 transformations can be defined. At present, square and square root have been defined. Others can be added as needed. The single digit input variable IDPARS controls how the transformation is made, and sections of code can be added.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL TRANS(KFILDO,ID,IDPARS,SDATA,NSTA,FD,ND2X3,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

ID(J) - 4-word MOS-2000 ID (J=1,4). Used only for diagnostic print. (INPUT)

IDPARS - A single digit controlling how the transformed variable is produced.
1 = SDATA( ) becomes the square of SDATA( ).
2 = SDATA( ) becomes the square root of SDATA( ). Negative values, if any, are set to the missing indicator 9999. (INPUT)

SDATA(K) - The input variable to transform (K=1,NSTA). These values can be at gridpoints. Since no spatial relations are required, the calculations can be performed in a linear array. On exit, SDATA( ) contains the transformed variable. (INPUT-OUTPUT)

NSTA - The number of values in SDATA( ). This must be LE ND2X3. (INPUT)

FD(K) - A work array (K=1,ND2X3). For IDPARS = 1 and 2, this is not needed, but is provided in case of future need. (INTERNAL)

ND2X3 - The dimension of FD( ). (INPUT)

IER - Status return.
0 = Good return.
63 = Value of IDPARS not used in this routine.
When IER NE 0, SDATA( ) is set to 9999. (OUTPUT)

EXAMPLE

```
DIMENSION ID(4),SDATA(ND2X3),FD(ND2X3)
DATA KFILDO/6/
...

CALL TRANS(KFILDO,ID,1,SDATA,50,FD,ND2X3,IER)
```

The unit number for diagnostics is 6. The MOS-2000 4-word ID is in ID( ). The 50 values in SDATA( ) are given the values SDATA( )**2. A return on IER = 0 means TRANS operated as expected; any other value for IER indicates all values in SDATA( ) = 9999 (K=1,ND2X3).

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than stated above. For IDPARS = 2, negative values in SDATA( ) will be set to 9999.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

DATGEN

GENERATES DATE/TIMES BETWEEN TWO DATE/TIMES

Harry R. Glahn
December 1, 1994

<u>PURPOSE</u>: To generate all date/times between two input date/times at a specified interval.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

CALL DATGEN(KFILDO,IP,IDATE,JDATE,INCCYL,NWORK,LD,ND8,IER)

<u>KFILDO</u>	- Unit number of output (print) file. Diagnostics, including error conditions are written here. (INPUT)
<u>IP</u>	- Unit number for output (print) file. The generated dates are put here, unless IP = 0. IP can equal KFILDO. (INPUT)
<u>IDATE</u>	- Starting date/time in format YYYYMMDDHH. (INPUT)
<u>JDATE</u>	- Ending date/time in format YYYYMMDDHH. All date/times between IDATE and JDATE inclusive at an interval of INCCYL are put into NWORK( ) starting with NWORK(LD). (INPUT)
<u>INCCYL</u>	- Increment in hours (see JDATE). (INPUT)
<u>NWORK</u>(J)	- The generated date/times are put into NWORK( ), starting with NWORK(LD). (INPUT-OUTPUT)
<u>LD</u>	- On entry, LD is the location in NWORK( ) where IDATE is to go. On exit, LD is the location in NWORK( ) of JDATE. (INPUT-OUTPUT)
<u>ND8</u>	- Dimension of NWORK( ). (INPUT)
<u>IER</u>	- Status return. 0 = Good return. 26 = Ending date not after beginning date or INCCYL is not $\geq$ 0. NWORK( ) and LD are not modified. 27 = Value of LD is negative. NWORK( ) and LD are not modified. 28 = NWORK( ) too small to hold dates. NWORK( ) has been filled to capacity and LD updated accordingly to ND8. (OUTPUT)

EXAMPLE

```
PARAMETER (ND8=100)
DIMENSION NWORK(ND8)
DATA KFILDO/6/
DATA LD/1/
...

CALL DATGEN(KFILDO,6,1994120100,1994123112,12,NWORK,LD,ND8,IER)
```

The unit number for diagnostics is 6. Also, the generated dates are written for printing to the same unit number. All dates for December 1994 at an interval of 12 hours are generated and put into NWORK( ) starting with NWORK(1). LD is returned with the location in NWORK( ) where 1994123112 resides.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. The generated dates will be written to Unit No. IP.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

FORIER

COMPUTES HARMONICS OF DAY OF YEAR

Harry R. Glahn
April 1, 1995

PURPOSE: To compute sine or cosine of the day of the year (first harmonic) or of twice the day of the year (second harmonic). The day can be either that in the date furnished or can consider the forecast projection, tau.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL FORIER(KFILDO,IDPARS,JD,NDATE,SINCOS,NSTA,ISTAV,IER)

    KFILDO    - Unit number of output (print) file.  (INPUT)

    IDPARS(J) - The MOS-2000 variable identification parsed into its
                15 components (J=1,15).  The portions used in this routine are
                indicated below.  First the day for which the sine or cosine
                is wanted is taken as just the day of the year in NDATE (DAY)
                or the DAY + tau in days (call this DAY+tau).  Then the values
                of IDPARS( ) used are as indicated:
                (1) = 010--Indicates geoclimatic data.
                (2) = 201--Sine DAY.
                    = 202--Cosine DAY.
                    = 203--Sine 2*DAY.
                    = 204--Cosine 2*DAY.
                    = 205--Sine DAY+tau.
                    = 206--Cosine DAY+tau.
                    = 207--Sine 2*(DAY+tau).
                    = 208--Cosine 2*(DAY+tau).
                (7) = Used to shift phase of harmonic IDPARS(7) days
                (12)= Tau used in the calculations as indicated above and in
                      NDATE below.
                (INPUT)

    JD(J)     - 4-word MOS-2000 ID (J=1,4) sans processing indicators.  Used
                only for diagnostic print.  (INPUT)

    NDATE     - The date in form YYYYMMDDHH for which the harmonic is to be
                computed, possibly with the addition of the projection, tau.
                (INPUT)

    SINCOS(J) - The array into which the harmonic value is to be put
                (J=1,NSTA).  (OUTPUT)

    NSTA      - Used as the dimension of SINCOS( ).  The computed harmonic
                value will be put into each location of SINCOS(J) (J=1,NSTA).
                (INPUT)

    ISTAV     - Since these are "station" (not gridpoint) values, ISTAV is set
                = 1.  (OUTPUT)
```

IER - Status return.
  0 = Good return.
  103 = IDPARS(1) NE 010 or IDPARS(2) NE one of the 8 values shown above.
  When IER NE 0, all NSTA values of SINCOS( ) are set to 9999. (OUTPUT)

EXAMPLE

```
    PARAMETER (ND1=700)
    DIMENSION JD(4)
    DIMENSION IDPARS(15)
    DIMENSION SINCOS(ND1)
    DATA KFILDO/6/
    ...

    IDPARS(1)=010
    IDPARS(2)=205
    IDPARS(12)=120
    NDATE=1995030300
    NSTA=50
    ...

    CALL TRANS(KFILDO,IDPARS,JD,NDATE,SINCOS,NSTA,ISTAV,IER)
```

The unit number for diagnostics is 6. The sine is to be calculated for NDATE + 5 days (the projection tau of 120 hours = 5 days) and placed in SINCOS( ). The same value will be put into the first 50 locations of SINCOS( ). Upon entry to TRANS, JD( ) should contain the variable ID for possible diagnostic printing. ISTAV is returned = 1. Do not use the values in SINCOS( ) unless IER = 0.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

RDI

READS FORMATTED RECORDS INTO AN INTEGER ARRAY

Harry R. Glahn
April 1, 1995

PURPOSE: To read data into an INTEGER array according to a given format. Records are read one at a time until a terminator is found. Blanks and zeros are eliminated from the data read.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

CALL RDI(KFILDO,IP,KFIL,IDATA,ND,ITEMP,NT,FMT,NVAL,ITERM,IER)

KFILDO	- Unit number of output (print) file. Diagnostics, including error conditions, are written here. (INPUT)
IP	- Unit number of optional output (print) file. Diagnostics, including error conditions, are written here, in addition to Unit No. KFILDO, when IP > 0 and ≠ KFILDO.
KFIL	- Unit number of the data set from which to read data. (INPUT)
IDATA(J)	- Array in which the data are returned (J=1,NVAL). (OUTPUT)
ND	- Size of array IDATA( ). (INPUT)
ITEMP( )	- Temporary array that must be of at least size NT. (INTERNAL)
NT	- Number of values to read per record indicated in FMT( ). (INPUT)
FMT( )	- Format to be used to read a record of data. It must pertain to REAL data. (CHARACTER*(*)) (INPUT)
NVAL	- The number of values returned in IDATA( ). (OUTPUT)
ITERM	- Terminator for the data to read. Records are read one by one according to the format in FMT( ) until the terminator is found. (INPUT)
IER	- Status return. 0 = Good return. 20 = Error reading or end of file on unit KFIL. 21 = Too many non-blank and non-zero data elements in records being read on unit KFIL before the terminator ITERM is found. (OUTPUT)

EXAMPLE

```
    PARAMETER (ND1=100)
    DIMENSION IDATA(ND1)
    DIMENSION ITEMP(50)
    DATA KFILDO/6/
    ...

    CALL RDI(KFILDO,20,IDATA,ND1,ITEMP,50,'(12I6)',NVAL,9999,IER)
    ...
```

The unit number for diagnostics is 6. The unit number to read the data from is 20. The data are read according to FORMAT(12I6) until the terminator 9999 is found. At that point, the data are returned in the array IDATA(J), J=1,NVAL. Each record of data is read into ITEMP( ), which being dimensioned 50 will hold the 12 values indicated in the format. After reading each record, zeros and blanks read as zeros are eliminated, and then the data transferred to IDATA( ) up to the terminator 9999. In this way, the dimension of IDATA( ) need not be greater that the number of "good" values read. The array IDATA( ) will not be overflowed; if too many values are present, a return with IER = 21 will be made. With that condition, ITEMP( ) will contain the ND1 values read up to that point.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

RDF

READS FORMATTED RECORDS INTO A REAL ARRAY

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>:

To read data into a REAL array according to a given format. Records are read one at a time until a terminator is found. Zeros and blanks are eliminated from the data read.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

CALL RDF(KFILDO,IP,KFIL,DATA,ND,TEMP,NT,FMT,NVAL,TERM,IER)

<u>KFILDO</u>	- Default unit number of output (print) file. Diagnostics, including error conditions, are written here. (INPUT)
<u>IP</u>	- Unit number of optional output (print) file. Diagnostics, including error conditions, are written here, in addition to Unit No. KFILDO, when IP > 0 and ≠ KFILDO.
<u>KFIL</u>	- Unit number of the data set from which to read data. (INPUT)
<u>DATA</u>(J)	- Array in which the data are returned (J=1,NVAL). (OUTPUT)
<u>ND</u>	- Size of array DATA( ). (INPUT)
<u>TEMP</u>( )	- Temporary array that must be of at least size NT. (INTERNAL)
<u>NT</u>	- Number of values to read per record indicated in FMT( ). (INPUT)
<u>FMT</u>( )	- Format to be used to read a record of data. It must pertain to REAL data. (CHARACTER*(*)) (INPUT)
<u>NVAL</u>	- The number of values returned in DATA( ). (OUTPUT)
<u>TERM</u>	- Terminator for the data to read. Records are read one by one according to the format in FMT( ) until the terminator is found. (INPUT)
<u>IER</u>	- Status return. 0 = Good return. 20 = Error reading or end of file on unit KFIL. 21 = Too many non-blank and non-zero data elements in records being read on unit KFIL before the terminator TERM is found. (OUTPUT)

EXAMPLE

```
    PARAMETER (ND1=100)
    DIMENSION DATA(ND1)
    DIMENSION TEMP(50)
    DATA KFILDO/6/
    DATA IP/6/
    ...

    CALL RDF(KFILDO,IP,20,DATA,ND1,TEMP,50,'(12F6.0)',NVAL,9999.,IER)
    ...
```

The unit number for diagnostics is 6. The unit number to read the data from is 20. The data are read according to FORMAT(12F6.0) until the terminator 9999. is found. Note that the terminator is REAL. At that point, the data are returned in the array DATA(J), J=1,NVAL. Each record of data is read into TEMP( ), which being dimensioned 50 will hold the 12 values indicated in the format. After reading each record, zeros and blanks read as zeros are eliminated, and then the data transferred to DATA( ) up to the terminator 9999. In this way, the dimension of DATA( ) need not be greater that the number of "good" values read. The array DATA( ) will not be overflowed; if too many values are present, a return with IER = 21 will be made. With that condition, TEMP( ) will contain the ND1 values read up to that point.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. When IP > 0 and ≠ KFILDO, the diagnostic messages will also be written to Unit No. IP.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

RDC

READS FORMATTED RECORDS INTO A CHARACTER ARRAY

Harry R. Glahn
April 1, 1995

PURPOSE: To read data into a CHARACTER array according to a given format. Records are read one at a time until a terminator is found. Blanks are eliminated from the data read.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

CALL RDC(KFILDO,IP,KFIL,CDATA,ND,CTEMP,NT,FMT,NVAL,CTERM,IER)

KFILDO	- Default unit number of output (print) file. Diagnostics, including error conditions, are written here. (INPUT)
IP	- Unit number of optional output (print) file. Diagnostics, including error conditions, are written here, in addition to Unit No. KFILDO, when IP > 0 and ≠ KFILDO.
KFIL	- Unit number of the data set from which to read data. (INPUT)
CDATA(J)	- Array in which the data are returned (J=1,NVAL). (CHARACTER*8) (OUTPUT)
ND	- Size of array CDATA( ). (INPUT)
CTEMP( )	- Temporary array that must be of at least size NT. (CHARACTER*8) (INTERNAL)
NT	- Number of values to read per record indicated in FMT( ). (INPUT)
FMT( )	- Format to be used to read a record of data. It must pertain to character data. (CHARACTER*(*)) (INPUT)
NVAL	- The number of values returned in CDATA( ). (OUTPUT)
CTERM	- Terminator for the data to read. Records are read one by one according to the format in FMT( ) until the terminator is found. (CHARACTER*8) (INPUT)
IER	- Status return. 0 = Good return. 20 = Error reading or end of file on unit KFIL. 21 = Too many non-blank and non-zero data elements in records being read on unit KFIL before the terminator CTERM is found. (OUTPUT)

EXAMPLE

```
    PARAMETER (ND1=100)
    CHARACTER CDATA(ND1)
    CHARACTER CTEMP(50)
    DATA KFILDO/6/
    DATA IP/6/
    ...

    CALL RDC(KFILDO,IP,20,CDATA,ND1,CTEMP,50,'(4A8)',NVAL,'9999',IER)
    ...
```

The unit number for diagnostics is 6. The unit number to read the data from is 20. The data are read according to FORMAT(4A8) until the terminator '9999' is found. Note that the terminator is CHARACTER. At that point, the data are returned in the array CDATA(J), J=1,NVAL. Each record of data is read into CTEMP( ), which being dimensioned 50 will hold the four values indicated in the format. After reading each record, blanks are eliminated, and then the data transferred to CDATA( ) up to the terminator '9999'. In this way, the dimension of CDATA( ) need not be greater that the number of "good" values read. The array CDATA( ) will not be overflowed; if too many values are present, a return with IER = 21 will be made. With that condition, CDATA( ) will contain the ND1 values read up to that point.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. When IP > 0 and NE KFILDO, the diagnostic messages will also be written to Unit No. IP.

RESTRICTIONS:

This routine was written to read 8-character strings, and the variables CDATA( ), CTEMP( ), and CTERM are CHARACTER*8 in RDC. With care, this routine may be applicable to strings of less than 8 characters, but has not been tested for that circumstance.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

DATPRO

PROCESSES A DATE/TIME LIST

Harry R. Glahn
April 1, 1995

PURPOSE: To accept a date/time list, which may contain negative values indicating a date span from that date backward thru the previous date; modify the dates from YYMMDDHH to YYYYMMDDHH; insure that all date/times are in ascending order; and create the dates in the indicated date spans.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL DATPRO(KFILDO,IDATE,NWORK,ND8,INCCYL,NDATES,IP2,IP3,IER)
```

KFILDO - Unit number of output (print) file. Diagnostics, including error conditions, are written here. (INPUT)

IDATE(J) - Initial date/time list (J=1,NDATES), which may contain negative values indicating a date span. On output, IDATE(J) contains the complete date list with the dates in the spans filled in (J=1,NDATES). Date/times are input as YYMMDDHH and output as YYYYMMDDHH. Zeros in the list are treated as errors. (INPUT-OUTPUT)

NWORK(J) - Work array (J=1,ND8). (INTERNAL)

ND8 Dimension of IDATE( ) and NWORK( ). (INPUT)

INCCYL - Increment in hours between the date/times that are to be created in the date spanning process. (INPUT)

NDATES - On input, the number of values input in IDATE( ). On output, the number of values output in IDATE( ). (INPUT-OUTPUT)

IP2 - Indicates whether (> 0) or not (= 0) the input dates in IDATE( ) will be printed. When there are no errors, the print will be to Unit No. IP2; when there are errors, print will be to Unit No. KFILDO, as well as to Unit No. IP2 when IP2 ≠ KFILDO. (INPUT)

IP3 - Indicates whether (> 0) or not (= 0) the output dates in IDATE( ) will be printed. When there are no errors, the print will be to Unit No. IP3; when there are errors, print will be to Unit No. KFILDO, as well as to Unit No. IP3 when IP3 ≠ KFILDO. (INPUT)

IER - Status return.
 0 = Good return.
25 = Error in the list.
Other errors may be returned from DATGEN.
(OUTPUT)

EXAMPLE

```
PARAMETER (ND8=100)
DIMENSION IDATE(ND8),NWORK(ND8)
DATA KFILDO/6/
DATA IP2/7/
DATA IP3/8/
...

CALL DATPRO(KFILDO,IDATE,NWORK,ND8,12,NDATES,IP2,IP3,IER)
```

The unit number for diagnostics is 6. The input dates are to be written to Unit No. 7, and if there are errors detected, the input dates will also be written to Unit No. 6. The output dates are to be written to Unit No. 8, and if there are errors detected, the output dates will also be written to Unit No. 6. The NDATES input dates must be in IDATE( ) and the dates including those created by date spanning will be returned in IDATE( ); NDATES will have been changed if necessary to include the number of output dates in IDATE( ). The number of hours between date/times when date/times are created is 12 hours. On input, NDATES must be $\leq$ ND8. When creating date/times, IDATE( ) will not be overflowed. When IER ≠ 0, a diagnostic will indicate the problem. The diagnostic will always be to Unit No. KFILDO, and to IP2 unless IP2 = 0 or = KFILDO.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO and also to Unit No. IP2 when IP2 ≠ 0 and ≠ KFILDO. Input dates will be written to Unit No. IP2 when IP2 ≠ 0 and also to Unit No. KFILDO when IP2 ≠ KFILDO. Output dates will be written to Unit No. IP3 when IP3 ≠ 0 and also to Unit No. KFILDO when IP3 ≠ KFILDO.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

DATGEN

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

RDSNAM

READS UNIT NUMBERS AND OPENS ASSOCIATED FILES

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To read unit numbers and data set names and the models to which the data apply for one or more data sets, and to open the files. The files will be opened as 'NEW', 'OLD', or 'SCRATCH' as indicated by STATUS and as 'FORMATTED' or 'UNFORMATTED' as indicated by FORMX. IF a file is opened as 'OLD' and an error occurs indicating the file is not present, a try will be made to open it as a new file. When a file is opened as 'NEW', and the file already exists, a diagnostic will occur and RDSNAM will halt. If a unit number is read that is the same as the default input KFILDI, it is not opened. As an added feature, a check is made to insure that no unit number is equal to one of the IP( ) numbers; if that occurs, the offending IP( ) number(s) is set to the default output KFILDO.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
    CALL RDSNAM(KFILDI,KFILDO,INUNIT,NAM,MODNUM,JFOPEN,ND6,NUMIN,STATUX,FORMX,
   1            IP,IER)
```

<u>KFILDI</u> - Unit number for input file from which to read the file information. (INPUT)

<u>KFILDO</u> - Unit number of output (print) file. Diagnostics, including error conditions, are written here. (INPUT)

<u>INUNIT</u>(J) - Unit numbers for data sets as read by RDSNAM (J=1,NUMIN). Values are read until a terminator 99 is found. (OUTPUT)

<u>NAM</u>(J) - Data set names corresponding to the unit numbers in INUNIT( ) (J=1,NUMIN). The name is limited to 60 characters. (CHARACTER*(*)) (OUTPUT)

<u>MODNUM</u>(J) - Model number corresponding to the unit numbers in INUNIT( ) (J=1,NUMIN). This is the model furnishing the data on Unit No. INUNIT(J), and in some uses may not be meaningful; if so, just leave the input blank and don't use it. (OUTPUT)

<u>JFOPEN</u>(J) - For each file corresponding to INUNIT(J), JFOPEN(J) is set to 1 if the file is open or an open has been attempted, and set to 2 otherwise (J=1,NUMIN). (OUTPUT)

<u>ND6</u> - The dimension of INUNIT( ), NAM( ), JFOPEN( ), and MODNUM( ). (INPUT)

<u>NUMIN</u> - The number of values read into INUNIT( ), NAM( ), and MODNUM( ). Maximum of ND6. (OUTPUT)

STATUX	- Holds the character string 'NEW', 'OLD', 'SCRATCH' or 'NOT'. Except for "NOT", this is used in the OPEN statement for the files whose numbers are in INUNIT( ). When STATUX = 'NOT', the file(s) is not opened. Note that STATUX pertains to all files in this call to RDSNAM. (CHARACTER*(*)) (INPUT)
FORMX	- Holds the character string 'UNFORMATTED' or 'FORMATTED'. This is used in the OPEN statement for the files whose numbers are in INUNIT( ). Note that FORMX pertains to all files in this call to RDSNAM. (CHARACTER*(*)) (INPUT)
IP(J)	- Unit numbers that are used for output in the calling program (J=1,25). If a value is read into INUNIT( ) that is the same as any value in IP( ), that IP( ) number is set to KFILDO, a diagnostic is printed, and processing continues. This situation is considered as an error in the input data. Note that 25 values are expected. In addition, IP(23) indicates whether (>0) or not (=0) a statement that the file has been opened is written to Unit IP(23). (INPUT)
IER	- Status return. 0 = Good return. 29 = Too many input data sets indicated. The maximum is ND6. 30 = Cannot find the 99 terminator. This would occur only after the condition leading to IER = 29. 31 = Trouble opening a file, or the file unit number = one of the IP( ) numbers. (OUTPUT)

EXAMPLE

```
      PARAMETER (ND6=5)
      CHARACTER*60 NAM(ND6)
      DIMENSION INUNIT(ND6),MODNUM(ND6)
      DIMENSION IP(25)
      DATA IP/25*0/
      DATA KFILDI/5/
      DATA KFILDO/6/
      ...

      CALL RDSNAM(KFILDI,KFILDO,INUNIT,NAM,MODNUM,ND6,NUMIN,'OLD','FORMATTED',
     1            IP,IER)
```

The unit number for diagnostics is 6. The unit number to read the data from is 5. Three values are read, in order INUNIT( ), MODNUM( ), and NAM( ), from each record according to FORMAT(2I3,1X,A20) until the terminator 99 is found in the first field (i.e., read into INUNIT). An attempt is made to open each file as 'OLD' and 'FORMATTED'. The arrays will not be overflowed, the maximum number of values being ND6 = 5 (besides the terminator). Since the 25 IP( ) values are all zero, no conflict should occur in unit numbers. Since IP(23) = 0, no indication that the file has been opened will be written to unit No. IP(23). The number of file numbers read is returned in NUMIN. Note that the file names can contain up to 60 characters.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than stated above, except HP UNIX error code 908 is used to denote a missing file that was attempted to be opened as 'OLD'. If the unit number is the same as an IP( ) number, that IP( ) number is set to KFILDO, and the file is allowed to be opened.

A file cannot be opened as 'NEW' that already exists; if such an attempt is made, RDSNAM halts.

NONSYSTEM ROUTINES USED:

IERX

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

IPOPEN

OPENS FILES USED FOR OUTPUT

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To open for output files for non-zero unit numbers in IP( ), except when IP( ) = KFILDO which is considered to be the default print file. The file names will be made up of (1) 4 characters furnished by the calling program in CALING, (2) then 4 characters furnished by the calling program in IPINIT, and (3) then 2 characters from IP( ). The files are opened as 'NEW', and any existing file with the same name is deleted. As an added feature, when IP(1) ≠ KFILDO, KFILDO is set equal to IP(1) and the file opened. This is so the default file (with name ftn12 on the HP workstations) will not be overwritten by other programs. Only the portion of the print on KFILDO written previous to the call to IPOPEN would be lost. Also, a check is made to insure that no unit number in IP( ) is equal to 45 through 49, 97, and 99, which are reserved for other uses in MOS-2000 programs.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

CALL IPOPEN(KFILDO,CALING,IPINIT,IP,IER)

<u>KFILDO</u>	- Unit number of output (print) file. Diagnostics, including error conditions, are written here. When IP(1) ≠ KFILDO, KFILDO will be set equal to IP(1). (INPUT-OUTPUT)
<u>CALING</u>	- 4 characters representing the calling program name. (CHARACTER*4) (INPUT)
<u>IPINIT</u>	- 4 characters to help define output file name. These 4 characters are appended to the 4 characters in CALING. (CHARACTER*4) (INPUT)
<u>IP</u>(J)	- Unit numbers that are used for output in the calling program (J=1,25). When IP(J) > 0 and IP(J) ≠ KFILDO, the two digits will be appended to the 8-character name already formed from CALING and IPINIT, and the file on Unit No. IP(J) with the name just formed will be opened as "NEW'. Any existing file with the same name will be deleted. Note that 25 values are expected in IP( ). Values of 45-49, 97, and 99 are not permissible. (INPUT)
<u>IER</u>	- Status return. 0 = Good return. 31 = Trouble opening a file. (OUTPUT)

EXAMPLE 1

```
    DIMENSION IP(25)
    DATA IP/6,10,23*0/
    DATA KFILDO/6/
    ...

    CALL IPOPEN(KFILDO,'U201','HRG1',IP,IER)
```

The default unit number for output is 6. Since the number in IP(1) = KFILDO, a file on Unit No. IP(1) is not opened. However, the file 'U201HRG110' will be opened on Unit No. 10. Note that the naming convention allows the program and person (initials) as well as the particular output to generate unique names. In this way, multiple runs of, say, U201 can be made by the same person without overwriting the files by using slightly different "initials." If all values in IP( ) = KFILDO, then no file would be opened and all output would be to the default file KFILDO.

EXAMPLE 2

```
    DIMENSION IP(25)
    DATA IP/15,10,23*0/
    DATA KFILDO/6/
    ...

    CALL IPOPEN(KFILDO,'U201','HRG1',IP,IER)
```

The only difference between this example and the last one is that the default output file unit number KFILDO will be changed from 6 to 15 and two files opened instead of one.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO, as well as the names of files opened.

RESTRICTIONS:

None other than stated above. Unit numbers of 45 through 49, 97, and 99 are reserved for other uses in MOS-2000 programs; if one of these numbers is read, a diagnostic is provided, and the offending value is set to the default value of KFILDO.

NONSYSTEM ROUTINES USED:

IERX

LANGUAGE: FORTRAN 77

LOCATION: home21/moslib on blizzard

IERX

WRITES MESSAGE FOR SYSTEM ERRORS

Harry R. Glahn
April 1, 1995

PURPOSE: To write to the output (print) file an error message indicating what routine the error occurred in, the location within that routine, and the system error number. Optionally, the same message is written to another file. IERX is used after an I/O operation on an error condition.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL IERX(KFILDO,IPN,IOS,RUTINE,STATMT)
```

KFILDO	- Default unit number of output (print) file. A statement indicating the system error number is written to this file. (INPUT)
IPN	- Unit number for optional print file. When IPN ≠ KFILDO, the message written to Unit No. KFILDO will also be written to Unit No. IPN. (INPUT)
IOS	- The error number furnished by the system. This number is printed in the message. (INPUT)
RUTINE	- 6 characters of the name of the calling program. (CHARACTER*(*)) (INPUT)
STATMT	- Statement number in calling program near the call to IERX. Maximum of 4 characters printed. (CHARACTER*(*)) (INPUT)

EXAMPLE

```
    DATA KFILDO/6/
    DATA IPN/10/
    ...

101 CALL IERX(KFILDO,IPN,IOS,'U201','101 ')
```

The default unit number for output is 6. Since the number in IPN ≠ KFILDO, the diagnostic message will be written to both Unit Nos. 6 and 10. The program calling IERX is U201, and the call is near statement 101. IOS has been returned by a system I/O statement. Note that both of the last two arguments have 4 characters each.

OUTPUT:

A diagnostic message will be written to Unit No. KFILDO and optionally to Unit No. IPN.

<u>RESTRICTIONS</u>:

None other than stated above.

<u>NONSYSTEM ROUTINES USED</u>:

None

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: MOS-2000 Library

RDPRED

READS AND ORDERS PREDICTOR LIST

Harry R. Glahn
April 1, 1995

PURPOSE: To read predictor list and associated information from a file on Unit No. KFILP. KFILP can be the default input file or can be a separate file. Also, predictor names and other information from the predictor constant file are retrieved from the file read on Unit No. KFILCP and matched with the predictors. Each predictor ID in ID(,) is duplicated in JD(,) except that the processing portions are omitted; the result is called the "basic" ID. The predictor list is then reordered with the fourth word of the ID (i.e., JD(4,)), which was zero, replaced with processing information. This replacement is only for ordering, and the zero is then restored to JD(4,). The result is the predictor list being in an order for optimum use for processing grid fields in U201.

RESTRICTIONS:

It is assumed the files on unit numbers KFILP and KFILCP have been opened.

NONSYSTEM ROUTINES USED:

PRSID, BASICP, SORT16, CKIDS, XCHANG, FSORT

LANGUAGE: FORTRAN 77

LOCATION: moslib

UPDAT

ADDS HOURS TO A DATE/TIME

Harry R. Glahn
April 1, 1995

PURPOSE: To add KHR hours to date/time in JDATE, and to provide the updated date/time in MDATE. The date/times are in the form YYYYMMDDHH.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL UPDAT(JDATE,KHR,MDATE)
```

JDATE	- The date/time in format YYYYMMDDHH to which KHR hours are to be added. (INPUT)
KHR	- The number of hours to add to JDATE. This can be negative as well as positive. (INPUT)
MDATE	- The updated date/time in the same format as JDATE. MDATE can be the same as JDATE. (OUTPUT)

EXAMPLE

```
CALL UPDAT(1995123100,36,MDATE)
```

The date December 31, 1995, and time 0000 will have 36 hours added to it, to become January 1, 1996, at 1200. MDATE will be returned 1996010112.

OUTPUT:

None.

RESTRICTIONS:

None, provided neither JDATE or MDATE is after Feb. 28, 2400.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

PACK1D

PACKS A 1-DIMENSIONAL ARRAY

Harry R. Glahn
April 1, 1995

PURPOSE: To pack a one-dimensional array of data in TDLPACK format at units resolution after multiplying by a given factor. The smallest value is subtracted to make all values positive. Additional values are taken out at nonuniform steps. Both a primary and a secondary "missing" value can be accommodated. 2nd order (spatial) differences are not calculated or considered. Subroutine PACK is called to do the actual packing once the data are scaled and put into INTEGER format.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL PACK1D(KFILDO,A,IC,NSTA,IS0,IS1,IS2,IS4,ND7,XMISSP,XMISSS,
   1            IPACK,ND5,MINPK,LX,IOCTET,L3264B,IER)
```

KFILDO	- Unit number of output (print) file. (INPUT)
A(J)	- Array holding the original (unpacked) data (J=1,NSTA). (INPUT)
IC(J)	- Array for scaled and rounded data (J=1,NSTA). (INTERNAL)
NSTA	- The number of values in A( ). Also treated as the dimension of A( ) and IC( ). (INPUT)
IS0(L)	- Holds the values to furnish for TDLPACK Section 0 (L=1,ND7). (INPUT)
IS1(L)	- Holds the values to furnish for TDLPACK Section 1 (L=1,ND7). (INPUT)
IS2(L)	- Holds the values to furnish for TDLPACK Section 2 (L=1,ND7). (INPUT)
IS4(L)	- Holds the values to furnish for TDLPACK Section 4 (L=1,ND7). (INPUT)
ND7	- Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). With the practical limit of 32 bytes of plain language in IS1( ), ND7 = 54 is sufficient. (INPUT)
XMISSP	- When primary missing values can be present in the data, they will have the value XMISSP. When XMISSP = 0, it signifies no primary (or secondary) missing values are present. While XMISSP is REAL, it is converted to INTEGER, so it should be a whole number. Normally, this will be 9999 when not zero. (INPUT)

<u>XMISSS</u> - When secondary missing values can be present in the data, they will have the value XMISSS. When XMISSS = 0, it signifies no secondary missing values are present. While XMISSS is REAL, it is converted to INTEGER, so it should be a whole number. Normally, this will be 9997 when not zero to accommodate the value sometimes produced by a forecast equation and may exist in the MOS forecast archive. It should not be non-zero unless XMISSP is also non-zero. (INPUT)

<u>IPACK</u>(J) - The array to hold the actual packed data (J=1,ND5) (OUTPUT)

<u>ND5</u> - Dimension of IPACK( ). Since the full array is zeroed on entry, ND5 should not be unrealistically large. IPACK( ) will not be overflowed. (INPUT)

<u>MINPK</u> - Values are packed in groups of minimum size MINPK. Only when the number of bits to handle a group changes will a new group be formed. (INPUT)

<u>LX</u> - The number of groups (the number of 2nd order minima). While LX is needed only in subroutine PACK, it is output in case the user wants to know it. (OUTPUT)

<u>IOCTET</u> - The total packed data array size in octets. This is not necessarily the value returned in IS0(2). Rather, it has been increased from IS0(2) if necessary to make it evenly divisible by 64. This makes writing possible in 64-bit chunks as required when writing in whole words on the CRAY. (OUTPUT)

<u>L3264B</u> - Must have the value of 32 or 64, indicating the word length of the machine being used.

<u>IER</u> - Status return.
  0 = Good return.
134 = MISSP = 0 and MISSS NE 0. MISSS is treated as zero.
Other values can be generated by called subroutines.
(OUTPUT)

<u>EXAMPLE</u>

```
    PARAMETER (ND1=500)
    PARAMETER (ND5=ND1)
    PARAMETER (ND7=54)
    DIMENSION A(ND1),IC(ND1)
    DIMENSION IS0(ND7),IS1(ND7),IS2(ND7),IS4(ND7)
    DIMENSION IPACK(ND5)
    DATA KFILDO/6/
    DATA L3264B/32/
    ...
    NSTA=450
    ...
    CALL PACK1D(KFILDO,A,IC,NSTA,IS0,IS1,IS2,IS4,ND7,9999.,9997.,
   1            IPACK,ND5,15,LX,IOCTET,L3264B,IER)
```

The default output unit number is 6. A( ) must contain 450 values to pack. The description of the contents of IS0( ), IS1( ), IS2( ), and IS4( ) is contained in Chapter 5 "Data Record Structure" in TDL Office Note "MOS-2000." As indicated there, some of the values are filled in by the packing routines. The maximum size needed for these arrays is 54 words. The data can have primary missing values of 9999. and secondary missing values of 9997.; note that these are REAL. The packed data are returned in IPACK( ) of maximum size 500 words. The dimension for this array should not be extremely large, and will not be overflowed. However, if it is about to be exceeded, an error return will be generated, and the data should not be used. Unless ND1 were small, ND5 = ND1 should be safe; otherwise, the packing would not have accomplished any space reduction. The minimum group size is to be 14, a reasonable value. (Actually, adjustment between two groups may make a group size as small as one. This happens when a "spike" appears in the data.) The number of groups used in packing is returned in LX, and the number of octets in the packed data is returned in IOCTET. L3264B is set to 32 to indicate the program is running on a 32-bit machine. IER values can be set by the called subroutines.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than stated above.

COMMENTS:

The primary and secondary missing values need not be specific values (e.g., 9999 and 9997) for packing purposes, but certain programs using the data expect specific values. The secondary "missing" value can be a legitimate value that may occur many times in the data, and may pack more efficiently as such. For instance, one such use is the value 888 for observed cloud height.

PACK1D also furnishes the capabilities:

- o If MISSS = 888, say, and no 888 exists in the data, no secondary missing value will be furnished to the packer. That is, the local value of MISSS is set = 0.

- o If MISSP = 9999 and no such value exists in the data, the local value of MISSP is set to 0, provided MISSS = 0.

- o If MISSP = 0 and MISSS ≠ 0 (a no-no), the local value of MISSS is set = 0.

NONSYSTEM ROUTINES USED:

PACK

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

PACK2D

PACKS A 2-DIMENSIONAL ARRAY

Harry R. Glahn
April 1, 1995

PURPOSE: To pack a two-dimensional array of data in TDLPACK format at units resolution after multiplying by a given factor. The smallest value is subtracted to make all values positive. Additional values are taken out at nonuniform steps. Both a primary and a secondary "missing" value can be accommodated. 2nd order (spatial) differences are packed rather than the values themselves if calculations show them to be useful; however, spatial differences are not used if there are missing values. The normal situation for 2-dimensional (gridpoint) data should be for no missing values. Subroutine PACK is called to do the actual packing once the data are scaled and put into INTEGER format.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL PACK2D(KFILDO,A,IA,IC,NX,NY,IS0,IS1,IS2,IS4,ND7,XMISSP,XMISSS,
   1            IPACK,ND5,MINPK,LX,IOCTET,L3264B,IER)
```

KFILDO	- Unit number of output (print) file. (INPUT)
A(IX,JY)	- Array holding the original (unpacked) gridpoint data (IX=1,NX) (JY=1,NY). (INPUT)
IA(IX,JY)	- Array for scaled and rounded data (IX=1,NX) (JY=1,NY). (INTERNAL)
IC(IX,JY)	- Work array (IX=1,NX) (JY=1,NY). (INTERNAL)
NX	- First dimension of A(,), IA(,), and IC(,). This should be the size of the grid in the horizontal (west-east) direction. (INPUT)
NY	- Second dimension of A(,), IA(,), and IC(,). This should be the size of the grid in the vertical (south-north) direction. (INPUT)
IS0(L)	- Holds the values to furnish for TDLPACK Section 0 (L=1,ND7). (INPUT)
IS1(L)	- Holds the values to furnish for TDLPACK Section 1 (L=1,ND7). (INPUT)
IS2(L)	- Holds the values to furnish for TDLPACK Section 2 (L=1,ND7). (INPUT)
IS4(L)	- Holds the values to furnish for TDLPACK Section 4 (L=1,ND7). (INPUT)

ND7	- Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). With the practical limit of 32 bytes of plain language in IS1( ), ND7 = 54 is sufficient. (INPUT)
XMISSP	- When primary missing values can be present in the data, they will have the value XMISSP. When XMISSP = 0, it signifies no primary (or secondary) missing values are present. While XMISSP is REAL, it is converted to INTEGER, so it should be a whole number. It would be unusual for this to be other than zero for gridpoint data. When non-zero, it would normally be 9999. (INPUT)
XMISSS	- When secondary missing values can be present in the data, they will have the value XMISSS. When XMISSS = 0, it signifies no secondary missing values are present. While XMISSS is REAL, it is converted to INTEGER, so it should be a whole number. It would be very unusual for this to be other than zero for gridpoint data. When non-zero, it would normally be 9997. (INPUT)
IPACK(J)	- The array to hold the actual packed data (J=1,ND5) (OUTPUT)
ND5	- Dimension of IPACK( ). Since the full array is zeroed on entry, ND5 should not be unrealistically large. IPACK( ) will not be overflowed. (INPUT)
MINPK	- Values are packed in groups of minimum size MINPK. Only when the number of bits to handle a group changes will a new group be formed. (INPUT)
LX	- The number of groups (the number of 2nd order minima). While LX is needed only in subroutine PACK, it is output in case the user wants to know it. (OUTPUT)
IOCTET	- The total packed data array size in octets. This is not necessarily the value returned in IS0(2). Rather, it has been increased from IS0(2) if necessary to make it evenly divisible by 64. This makes writing possible in 64-bit chunks as required when writing in whole words on the CRAY. (OUTPUT)
L3264B	- Must have the value of 32 or 64, indicating the word length of the machine being used.
IER	- Status return. 0 = Good return. 134 = MISSP = 0 and MISSS NE 0. MISSS treated as zero. Other values can be generated by called subroutines. (OUTPUT)

EXAMPLE

```
    PARAMETER (ND2=50)
   1           ND3=40)
```

```
    PARAMETER (ND2X3=ND2*ND3)
    PARAMETER (ND5=ND2*ND3)
    PARAMETER (ND7=54)
    DIMENSION A(ND2X3),IA(ND2X3),IC(ND2X3)
    DIMENSION IS0(ND7),IS1(ND7),IS2(ND7),IS4(ND7)
    DIMENSION IPACK(ND5)
    DATA KFILDO/6/
    DATA L3264B/32/
    ...
    NX=45
    NY=35
    ...

    CALL PACK2D(KFILDO,A,IA,IC,NX,NY,IS0,IS1,IS2,IS4,ND7,0.,0.,
   1            IPACK,ND5,14,LX,IOCTET,L3264B,IER)
```

The default output unit number is 6. A(,) must contain a grid of values 45 X 35 to pack. These actual grid dimensions do not have to match ND2 and ND3; the only requirement is that NX*NY ≤ ND2*ND3. The description of the contents of IS0( ), IS1( ), IS2( ), and IS4( ) is contained in Chapter 5 "Data Record Structure" in TDL Office Note "MOS-2000." As indicated there, some of the values are filled in by the packing routines. The maximum size needed for these arrays is 54 words. The data do not have missing values; note that the zeros are REAL. The packed data are returned in IPACK( ) of maximum size 2000 words. The dimension for this array should not be extremely large, and will not be overflowed. However, if it is about to be exceeded, an error return will be generated, and the data should not be used. Unless ND2*ND3 were small, ND5 = ND2*ND3 should be very safe; otherwise, the packing would not have accomplished any space reduction. Even ND2*ND3/2 should be safe. The minimum group size is to be 14, a reasonable value. (Actually, adjustment between two groups may make a group size as small as one. This happens when a "spike" appears in the data.) The number of groups used in packing is returned in LX, and the number of octets in the packed data is returned in IOCTET. L3264B is set to 32 to indicate the program is running on a 32-bit machine. IER values can be set by the called subroutines. Subroutine PACKXX is used to estimate whether spatial differences will be useful.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than stated above.

COMMENTS:

The primary and secondary missing values need not be specific values (e.g., 9999 and 9997) for packing purposes, but certain programs using the data expect specific values. The secondary "missing" value can be a legitimate value that may occur many times in the data, and may pack more efficiently as such. For instance, one such use is the value 888 for observed cloud height.

PACK2D also furnishes the capabilities:

- If MISSS = 888, say, and no 888 exists in the data, no secondary missing value will be furnished to the packer. That is, the local value of MISSS is set = 0.

- If MISSP = 9999 and no such value exists in the data, the local value of MISSP is set to 0, provided MISSS = 0.

- If MISSP = 0 and MISSS ≠ 0 (a no-no), the local value of MISSS is set = 0.

NONSYSTEM ROUTINES USED:

PACK, PACKXX

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

PACKXX

DETERMINES WHETHER TO PACK SPATIAL DIFFERENCES WITH NO MISSING VALUES

Harry R. Glahn
April 1, 1995

PURPOSE: To be used by PACK2D when there are no missing values to estimate whether or not it will be beneficial to pack spatial differences. Original values are indicated when the average range of consecutive groups of size MINPK of the second order differences is larger than the average range of consecutive groups of size MINPK of the original values. This routine gives the correct answer when the choice is clear. When it isn't, it doesn't matter much which choice is made.

RESTRICTIONS:

Will not accommodate missing values.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

PACK

PACKS DATA IN TDLPACK FORMAT

Harry R. Glahn
April 1, 1995

PURPOSE: To be used by PACK1D or PACK2D to pack the data in the TDLPACK format, as described in Chapter 5 "Data Record Structure" in TDL Office Note "MOS-2000." PACK1D AND PACK2D prepare the data for packing, and in particular if spatial differences are to be packed, to calculate those differences. The smallest value is subtracted to make all values positive. Additional values are taken out at nonuniform steps with a minimum group size of MINPK. Variables LBIT( ), JMAX( ), JMIN( ), NOV( ), and their dimensions NDG are not carried as arguments to make for ease of use. NDG is hardwired to 500, which is the maximum number of groups that can be used in the packing. If 500 is not enough, PACK (through PKMS99 or PKMS00 and PACKGP) automatically increases a working copy of MINPK by 50 percent until the number of groups required falls below 500. In this case, a diagnostic is written to the default output file to let the user know that this has happened. However, the packed data are all right. Subroutine PKMS99 or PKMS00 is used to determine the grouping to use, the minimum values, etc., depending, respectively, on whether or not missing data can be present.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

PKBG, PKMS00, PKMS99, PKC4LX, PKS4LX.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

PACK00

ASSISTS PACKING WHEN THERE CANNOT BE MISSING VALUES

Harry R. Glahn
April 1, 1995

PURPOSE: To be used by PACK to determine, through PACKGP, the groups to use in packing in TDLPACK format, their minima, etc. when there cannot be missing values, and to remove the overall minimum and local minimum from the each of the groups.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

PACKGP

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

PKMS99

ASSISTS PACKING WHEN THERE CAN BE MISSING VALUES

Harry R. Glahn
April 1, 1995

PURPOSE: To be used by PACK to determine, through PACKGP, the groups to use in packing in TDLPACK format, their minima, etc. when there can be missing values, and to remove the overall minimum and local minimum from the each of the groups. PKMS99 calls PKMS97 when there can be secondary missing values.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

PACKGP, PKMS97

LANGUAGE: FORTRAN 77

LOCATION: moslib

PACKGP

DETERMINES GROUPS FOR PACKING

Harry R. Glahn
April 1, 1995

PURPOSE: To be used by PACK00, PACK99, or PACK97 to determine the groups to use in packing in TDLPACK format, their minima, etc. PACKGP is designed to determine the groups such that a small number of bits is necessary to pack the data without excessive computations. If all values in a group (after removing the local minimum) are zero, the number of bits to use in packing these values is defined as zero. Even though the groups are initially of size MINPK or larger, an adjustment between two groups (the lookback procedure) may make a group smaller than MINPK, and even as small as 1. The control on group size is (generally) that the sum of the sizes of the two consecutive groups, each of size MINPK or larger, remains the same. When determining the number of bits necessary for packing, the largest value that can be accommodated in, say, MBITS, is 2**MBITS-1; this largest value is reserved for the missing value indicator (only) when missing values can be present. In addition, the value 2**MBITS-2 is reserved for a "secondary" missing value (only) when such can be present.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

PKBG

PACKS BITS INTO WORDS

Harry R. Glahn
April 1, 1995

PURPOSE: To pack bits into words in an array. PKBG will work on either 32- or 64-bit words, but only a maximum of 32 bits can be packed on one call. On input, the word and bit pointers indicate where in the array to start packing (leftmost bit); on output, they indicate the bit and word just after the bits packed into. Packing will not be done if the packed array would overflow. The bits are "inserted" rather than "added." That is, the bits to the right of the packed bits are retained, as well as the bits to the left. This means that the array to be packed into should be zeroed before being packed, and any area packed into must have zero bits on entry for proper packing to occur.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL PKBG(KFILDO,IPACK,ND5,LOC,IPOS,NVALUE,NBIT,L3264B,IER,*)
```

KFILDO	- Unit number of output (print) file. Only on the /D compile option is this file written to. (INPUT)
IPACK(J)	- Array to pack into (J=1,ND5). (INPUT-OUTPUT)
ND5	- Dimension of IPACK( ). (INPUT)
LOC	- Number of the word in IPACK( ) to start packing. This is updated as necessary after packing. It must be in the range 1 to ND5 on input. If the last bit in IPACK( ) has been packed, with no more bits to pack on this call, LOC is returned as ND5+1 with no error indicated. (INPUT-OUTPUT)
IPOS	- Bit position (counting the leftmost bit in the word as 1) to start packing. IPOS must be $\geq$ 1 and $\leq$ L3264B. This is updated as necessary after packing. (INPUT-OUTPUT)
NVALUE	- The rightmost NBIT bits in NVALUE are packed into IPACK( ). (INPUT)
NBIT	- The rightmost NBIT bits in NVALUE are packed into IPACK( ). NBIT must be $\geq$ 0 and $\leq$ 32. (INPUT)
L3264B	- Integer word length of machine being used. This has been tested for 32 and 64. (INPUT)

IER - Status return.
0 = Good return.
1 = Packing would overflow IPACK( ). That is, LOC is outside the range 1 to ND5 with more bits to pack.
2 = IPOS not in the range 1 to L3264B.
3 = NBIT not in the range 0 to 32.
4 = NBIT = 0, but NVALUE ≠ 0.
(OUTPUT)

* - Alternate return when IER ≠ 0. Since this routine may be called many times, using an alternate return eliminates a separate check of IER after each call.

EXAMPLE

```
PARAMETER (ND5=1000)
DIMENSION IPACK(ND5)
DATA KFILDO/6/
DATA L3264B/32/
...
LOC=1
IPOS=1

CALL PKBG(KFILDO,IPACK,ND5,LOC,IPOS,50,16,L3264B,IER,250)
```

The unit number for optional diagnostics is 6. The call is made on a 32-bit machine. The value 50 is placed into the left half of IPACK(1). Upon return, LOC is still 1 and IPOS has been updated to 17. If another identical call to the above were made, a value of 50 would be put into the right half of IPACK(1), and LOC would be updated to 2 and IPOS would be 1. If IER is not zero, the return is made to FORTRAN statement number 250.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO only if PKBG is compiled with the /D option on the HP workstation.

RESTRICTIONS:

None other than stated above. Note that this routine acts as "insertion" rather than "addition." That is, the bits to the right of the packed bits are retained, as well as the bits to the left. This means that the array to be packed into should be zeroed before being packed, and any area packed into must have zero bits on entry for proper packing to occur. No accommodation is made for negative values; PKBG operates on the rightmost bits of NVALUE.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

UNPACK

UNPACKS DATA FROM TDLPACK FORMAT

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To unpack a one- or two-dimensional array of data in TDLPACK format packed by routine PACK as called by PACK1D or PACK2D. It handles (1) 2nd order difference removal called complex packing by GRIB, (2) 2nd order spatial differences and alternate row reversal for gridpoint data, and (3) both primary and secondary missing values.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
    CALL UNPACK(KFILDO,IPACK,IWORK,DATA,ND5,IS0,IS1,IS2,IS4,ND7,MISSP,MISSS,
   1            IGIVE,L3264B,IER)
```

<u>KFILDO</u>	- Unit number of output (print) file. (INPUT)
<u>IPACK</u>(J)	- Array holding the packed data (J=1,ND5). (INPUT)
<u>IWORK</u>(J)	- Work array (J=1,ND5). (INTERNAL)
<u>DATA</u>(J)	- Unpacked data returned (J=1,ND5). (OUTPUT)
<u>ND5</u>	- Dimension of IPACK( ), IWORK( ), and DATA( ). (INPUT)
<u>IS0</u>(L)	- Holds the values from TDLPACK Section 0 (L=1,ND7). (OUTPUT)
<u>IS1</u>(L)	- Holds the values from TDLPACK Section 1 (L=1,ND7). (OUTPUT)
<u>IS2</u>(L)	- Holds the values from TDLPACK Section 2 (L=1,ND7). (OUTPUT)
<u>IS4</u>(L)	- Holds the values from TDLPACK Section 4 (L=1,ND7). (OUTPUT)
<u>ND7</u>	- Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). With the practical limit of 32 bytes of plain language in IS1( ), ND7 = 54 is sufficient. (INPUT)
<u>MISSP</u>	- When primary missing values are present in the data, they have the value MISSP. When MISSP = 0, it signifies no primary (or secondary) values are present. Normally, this will be 9999 when non-zero. Note that MISSP is INTEGER. (OUTPUT)
<u>MISSS</u>	- When secondary missing values are present in the data, they have the value MISSS. When MISSS = 0, it signifies no or secondary values are present. MISSS will not be non-zero unless MISSP is also non-zero. Normally, this will be 9997 when non-zero. Note that MISSS is INTEGER. (OUTPUT)

<u>IGIVE</u> - Indicates how much output is wanted.
1 = ID's only--fills IS0( ), IS1( ), IS2( ), and IS4( ).
Otherwise, both ID's and data. (INPUT)

<u>L3264B</u> - Must have the value of 32 or 64, indicating the word length of the machine being used.

<u>IER</u> - Status return.
0 = Good return.
10 = Can't find beginning of message.
11 = Can't find end of message.
12 = Edition number not what's expected. This is hardwired in UNPACK to 0.
13 = Bit map indicated in record, but not supported in this edition.
14 = Too many groups indicated. Maximum of 500 has been set by PARAMETER statement. This should match the corresponding value in PACK.
15 = Gridpoint data and missing values are indicated. This is not supported for convenience and efficiency, but also because missing values in gridpoint data are not, in general, checked in downstream programs and allowing them here would likely result in errors that might not be detected later.
16 = ND7 $\leq$ 22 or IS1( ) about to be overflowed. The latter may indicate an error in Section 1 data.
17 = Dimension ND5 not large enough to accommodate data.
18 = IS4( ) indicates simple packing which is not supported.
Other values can be generated by called subroutines.
(OUTPUT)

<u>EXAMPLE</u>

```
      PARAMETER (ND2=50)
     1           ND3=40)
      PARAMETER (ND5=ND2*ND3)
      PARAMETER (ND7=54)
      DIMENSION IPACK(ND5),IWORK(ND5),DATA(ND5)
      DIMENSION IS0(ND7),IS1(ND7),IS2(ND7),IS4(ND7)
      DATA KFILDO/6/
      DATA L3264B/32/
      ...

      CALL UNPACK(KFILDO,IPACK,IWORK,DATA,ND5,IS0,IS1,IS2,IS4,ND7,MISSP,MISSS,
     1            2,L3264B,IER)
```

The default output unit number is 6. Both ID's and data are to be returned. Upon return, DATA( ) contains the unpacked values in a linear array. The description of the data is in the contents of IS0( ), IS1( ), IS2( ), and IS4( ) which are described in Chapter 5 "Data Record Structure" in TDL Office Note "MOS-2000." The maximum size needed for these arrays is 54 words. If the data have missing values, those missing values are indicated by the values MISSP and MISSS for primary and secondary, respectively; a zero indicates no missing values. L3264B is set to 32 to indicate the program is running on a 32-bit machine. IER values can be set by the called subroutine, UNPKBG. If the data returned are gridpoint, the NX and NY dimensions of the grid are in IS2(3) and IS2(4), respectively. The grid of data can then be dealt with as a doubly subscripted variable DATA(NX,NY).

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

UNPKBG, UNPKLX, UNPKOO, UNPKPO, UNPKPS

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

UNPKBG

UNPACKS BITS FROM WORDS

Harry R. Glahn
April 1, 1995

PURPOSE: To extract bits from words. UNPKBG will work on either 32- or 64-bit words, but only a maximum of 32 bits can be unpacked on one call. On input, the word and bit pointers indicate where to start unpacking (leftmost bit); on output, they indicate the bit and word just after the bits unpacked. Unpacking will not be done outside the array.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL UNPKBG(KFILDO,IPACK,ND5,LOC,IPOS,NVALUE,NBIT,L3264B,IER,*)
```

KFILDO - Unit number of output (print) file. Only on the /D compile option is this file written to. (INPUT)

IPACK(J) - Array to unpack from (J=1,ND5). (INPUT)

ND5 - Dimension of IPACK( ). (INPUT)

LOC - Number of the word in IPACK( ) to start unpacking. This is updated as necessary after unpacking. This must be in the range 1 to ND5 on input. If the last bit in IPACK( ) has been unpacked, with no more bits to unpack on this call, LOC is returned as ND5+1 with no error indicated. (INPUT-OUTPUT)

IPOS - Bit position (counting the leftmost bit in the word as 1) to start unpacking. IPOS must be $\geq$ 1 and $\leq$ L3264B. This is updated as necessary after unpacking. (INPUT-OUTPUT)

NVALUE - The NBIT bits in IPACK( ) indicated by LOC and IPOS are returned in the rightmost NBIT bits of NVALUE. (OUTPUT)

NBIT - The NBIT bits in IPACK( ) indicated by LOC and IPOS are returned in the rightmost NBIT bits of NVALUE. (OUTPUT)

L3264B - Integer word length of machine being used. This has been tested for 32 and 64. (INPUT)

IER - Status return.
 0 = Good return.
 6 = Packing would be outside IPACK( ). That is, LOC is outside the range 1 to ND5 with more bits to unpack.
 7 = IPOS not in the range 1 to L3264B.
 8 = NBIT not in the range 0 to 32.
(OUTPUT)

*	- Alternate return when IER ≠ 0. Since this routine may be called many times, using an alternate return eliminates a separate check of IER after each call.

EXAMPLE

```
PARAMETER (ND5=1000)
DIMENSION IPACK(ND5)
DATA KFILDO/6/
DATA L3264B/32/
...
LOC=1
IPOS=1

CALL UNPKBG(KFILDO,IPACK,ND5,LOC,IPOS,NVALUE,16,L3264B,IER,250)
```

The unit number for optional diagnostics is 6. The call is made on a 32-bit machine. The 16 bits in the left half of IPACK(1) are returned in NVALUE. Upon return, LOC is still 1 and IPOS has been updated to 17. If another identical call to the above were made, the 16 bits in the right half of IPACK(1) would be returned in NVALUE, and LOC would be updated to 2 and IPOS would be 1. If IER is not zero, the return is made to FORTRAN statement number 250.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO only if UNPKBG is compiled with the /D option on the HP workstation.

RESTRICTIONS:

None other than stated above. Note that UNPKBG operates only on bits; no accommodation is made for negative numbers. Also note that on input, the word and bit pointers indicate where to start unpacking (leftmost bit); on output, they indicate the bit and word just after the bits unpacked. Therefore, LOC and IPOS must be variables, not constants.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

SKIPWR

SKIPS RECORDS AND WRITES STATION LIST

Harry R. Glahn
April 1, 1995

PURPOSE: To skip records on an output file in the TDLPACK format that has, or is to have, a station directory (call letters list). First, if data are to be skipped, an attempt is made to read the station directory record. If the record is read correctly, the directory is checked with the one provided to SKIPWR, but only when KCHECK ≠ 0. If they are the same, or are not checked because KCHECK = 0, the sequential file is read until all data up to and including an input date in KSKIP have been passed. If data are not to be skipped, it is assumed the file is empty, and the station directory is written. The directory record read or written consists of consecutive strings of 32 bits, each string being 8 characters. The record is written as a number of bytes evenly divisible by 64 to accommodate reading on the CRAY. When KWRITE ≠ 0, after any skipping is done, a trailer record and then the call letters record is written. SKIPWR is used by U201, but should have use for any program that writes a "vector" type file. The multiple directory record feature was added to accommodate hourly data archives. Note that the trailer record signals that the next record is a directory record; it does not signal that this is the end of the data records. That is, a trailer should not be written unless a directory record follows.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL SKIPWR(KFILDO,KFILIO,KSKIP,KWRITE,KCHECK,CCALL,ND1,NSTA,
   1            CCALLD,NDX,IPACK,ND5,NTOTBY,NTOTRC,L3264B,L3264W,IER)
```

KFILDO	-	Unit number of output (print) file. (INPUT)
KFILIO	-	Unit number of file to read and skip records. (INPUT)
KSKIP	-	When non-zero, indicates that the output file is to be moved forward until all data for date KSKIP have been skipped or an EOF has been found. KSKIP is in the format YYYYMMDDHH. (INPUT)
KWRITE	-	When non-zero, the call letters record will be written (but see the table below KCHECK), following the data to be skipped, and after writing a trailer record.
KCHECK	-	When non-zero, the call letters record last read, if any, will be checked with the one provided in CCALL( ), and an error returned when they do not match. For the actions given possible values of KSKIP, KWRITE, and KCHECK, see the table below:

KSKIP	KCHECK	KWRITE	ACTION
0	0	0	Write directory.
0	0	1	Write directory.
0	1	0	Write directory.
0	1	1	Write directory.
Date	0	0	Not a reasonable combination. Treat as KCHECK = 1, KWRITE = 0.
Date	0	1	Write trailer and directory after skipping
Date	1	0	Check last directory read while skip-ping, error if no match.
Date	1	1	Check last directory read while skip-ping, write trailer and directory only if no match.

<u>CCALL</u>(K) - 8-byte station call letters (K=1,NSTA) which comprise the directory record to write. (CHARACTER*8) (INPUT)

<u>ND1</u> - The dimension of CCALL( ). (INPUT)

<u>NSTA</u> - Number of call letters in CCALL( ), maximum of ND1. (INPUT)

<u>CCALLD</u>(K) - Work array. When 8-byte station call letters records are read, they are read into CCALLD( ) (K=1,NDX). (CHARACTER*8) (INTERNAL)

<u>NDX</u> - Maximum number of stations that can be in an existing direc-tory record to be read on the output file. Dimension of CCALLD( ). (INPUT)

<u>IPACK</u>(J) - Work array (J=1,ND5). ND5 must be $\geq$ 192/L3264B. (INTERNAL)

<u>ND5</u> - Dimension of IPACK( ). (INPUT)

<u>NTOTBY</u> - The total number of bytes on the file. It is initialized to zero, and the bytes counted when skipping records and writing the trailer and directory records. (OUTPUT)

<u>NTOTRC</u> - The total number of records on the file. It is initialized to zero, and the records counted when skipping records and writing the trailer and directory records. (OUTPUT)

<u>L3264B</u> - Integer word length in bits of machine being used, either 32 or 64. (INPUT)

<u>L3264W</u> - Number of INTEGER machine words in 64 bits on the machine being used, either 1 or 2. (INPUT)

<u>IER</u> - Status return.
  0 = Good return.
140 = Error reading call letters record.
141 = CCALLD( ) not large enough.

142 = Call letters read from the file do not match those in CCALL( ).
143 = Error when skipping records.
144 = Error writing call letters record.
Other values can be generated by called subroutines UNPKBG and TRAIL. (OUTPUT)

EXAMPLE

```
    PARAMETER (ND1=500)
    PARAMETER (ND5=ND1)
    PARAMETER (NDX=ND1)
    PARAMETER (L3264B=32)
    PARAMETER (L3264W=64/L3264B)
C
    CHARACTER*8 CCALL(ND1),CCALLD(NDX)
C
    DIMENSION IPACK(ND5)
C
    DATA KFILDO/6/
    DATA KFILIO/20/
    DATA KSKIP/1994123112/
    ...
    NSTA=250

    CALL SKIPWR(KFILDO,KFILIO,KSKIP,0,1,CCALL,ND1,NSTA,
   1            CCALLD,NDX,IPACK,ND5,NTOTBY,NTOTRC,L3264B,L3264W,IER)
```

The default output unit number is 6. The unit number for the output file is 20, the recommended number for this file for U201. Since KSKIP ≠ 0 and KCHECK ≠ 0, the file is spaced over all data including those for December 31, 1994, 1200 GMT, and the last directory record read is checked with the contents of CCALL( ) which contains 250 8-character call letters. If there is a match, all is well; if there is not a match, return is made with IER = 142 because KWRITE = 0. (Had KWRITE been ≠ 0, a trailer and the directory in CCALLD( ) would have been written with IER = 0.) L3264B, set by parameter, would be 64 when running on the CRAY. The total number of bytes (records) skipped and written will be returned in NTOTBY (NTOTRC).

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO, as well as a summary of the file operations performed.

RESTRICTIONS:

None other than stated above. Note that the station directory itself is not packed in the TDLPACK format, although all other data are. One or more such directories can exist on a file. Because every precaution must be taken to not overwrite good data, the user should always check the IER status return.

<u>NONSYSTEM ROUTINES USED</u>:

UNPKBG, TRAIL

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

WRITEP

WRITES RECORDS AND SWITCHES FILES

Harry R. Glahn
April 1, 1995

PURPOSE: To write records on an output file in the TDLPACK format. The number of bytes and records written are counted.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL WRITEP(KFILDO,KFILIO,IPACK,NWORDS,NTOTBY,NTOTRC,L3264B,IER)
```

KFILDO	- Unit number of output (print) file. (INPUT)
KFILIO	- Unit number of file to write. (INPUT)
IPACK(J)	- The (packed) data to write (J=1,NWORDS). (INPUT)
NWORDS	- The number of words to write from IPACK( ). Used as the dimension of IPACK( ). (INPUT)
NTOTBY	- The total number of bytes on the file are counted. (INPUT-OUTPUT)
NTOTRC	- The total number of records on the file are counted. (INPUT-OUTPUT)
L3264B	- Integer word length of machine being used, either 32 or 64. (INPUT)
IER	- Status return. 0 = Good return. 70 = Error writing record. (OUTPUT)

EXAMPLE

```
PARAMETER (ND5=500)
PARAMETER (L3264B=32)
DIMENSION IPACK(ND5)
DATA KFILDO/6/
DATA KFILIO/20/
...

NWORDS=250

CALL WRITEP(KFILDO,KFILIO,IPACK,NWORDS,NTOTBY,NTOTRC,L3264B,IER)
```

The default output unit number is 6. The unit number for the output file is 20, the recommended number for this file for U201. 250 words will be written, which is in terms of the machine being used. L3264B, set by parameter, would be 64 when running on the CRAY. It is intended that a running total of the number of bytes (records) written will be returned in NTOTBY (NTOTRC); to be so, the calling program must not modify NTOTBY (NTOTRC).

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO, as well as a summary of the tape operations performed when the compile /D option is used.

RESTRICTIONS:

None other than stated above. Note that this routine is not the one to use for writing a station directory; rather use SKIPWR. The tape changing facility is not yet provided; it can be added when the CRAY situation becomes know.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

PRTGR

CONTOURS TWO-DIMENSIONAL ARRAYS

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: Writes to a data set for printing a two-dimensional array or a subset of an array with or without contouring. A double quadratic interpolation using Bessel's Central Difference Formula is used to construct contours. As an option, bilinear interpolation can be done.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS</u>:

```
    CALL PRTGR(KFILDO,GRID,NX,NY,CINT,ORIGIN,SMULT,SADD,IOPT,TITLE,IER)
```

<u>KFILDO</u> - Unit number of output (print) file. (INPUT)

<u>GRID</u>(IX,JY)- Input data array (IX=1,NY) (JY=1,NY). Data are stored in NMC convention. Starting point GRID(1,1) on the map is lower left with data stored left to right (the IX direction), then the next row up (the JY direction), etc. (INPUT)

<u>NX</u> - First dimension of GRID(,). (INPUT)

<u>NY</u> - Second dimension of GRID(,). (INPUT)

<u>CINT</u> - When CINT $\neq$ 0, it is the value to be used for the contour interval. This pertains to the original values, not as modified by SMULT below. When CINT = 0, data will be displayed without contouring. (INPUT)

<u>ORIGIN</u> - Value to be used as a starting point for contouring. This pertains to the original values, not as modified by SMULT below. Not used if CINT = 0. (INPUT)

<u>SMULT</u> - Multiplicative scaling factor for displayed data values. Must be set to 1 if data are not to be modified by this variable (see SADD below). (INPUT)

<u>SADD</u> - Additive scaling factor for displayed data values. Must be set to 0 if data are not to be modified by this variable. Scaled value = original value * SMULT + SADD rounded. (INPUT)

<u>IOPT</u>( ) - Additional options table:
Note: All the values in the option table must be initialized if the table is used (IOPT(1) $\neq$ 0). When IOPT(1) = 0, default values are used. (INPUT)

(1) - 0 = Option table will not be used. Otherwise, option table will be used.
(2) - Minimum IX value to print. Default = 1. Also, default = 1 if IOPT(2) $\leq$ 0.

```
        (3) - Maximum IX value to print.  Default = NX.  Also, default = NX
              if IOPT(3) > NX.
        (4) - Minimum JY value to print.  Default = 1.  Also, default = 1 if
              IOPT(4) ≤ 0.
        (5) - Maximum JY value to print.  Default = NY.  Also, default = NY
              if IOPT(5) > NY.
        (6) - 1 if all interpolation is to be bilinear.  Otherwise,
              biquadratic interpolation will be done where possible.
              Default is biquadratic interpolation.
        (7) - Not used.
        (8) - Page width in gridpoints.  When IOPT(8) = 0, default of 24 is
              used.  Normally, 24 is the maximum a printer can handle.
```

<u>TITLE</u> - Title to be printed at the bottom of each page. Maximum of 74 characters. (CHARACTER*(*))

<u>IER</u> - Status return.
 0 = Good return.
90 = Overflow in scaling.

<u>EXAMPLE 1</u> (Minimal Call):

```
    PARAMETER (NX=8)
    PARAMETER (NY=10)
    DIMENSION GRID(NX,NY)
    DATA KFILDO/6/
    ...

    CALL PRTGR(KFILDO,GRID,NX,NY,3.,1000.,1.,0.,0,
   1           '              THIS IS A PRTGR TEST.     ',IER)
```

The output file number is 6. An array of 8 x 10 values will be contoured and written for display with a contour interval of 3, an origin of 1000, and no scaling. Only one page will be needed, biquadratic interpolation will be done, and a title will be printed across the bottom of each page.

<u>EXAMPLE 2</u>:

```
    CHARACTER*40 TITLE/'THIS IS A NO CONTOUR TEST               '/
    DIMENSION GRID(35,30),TITLE(10),IOPT(8)
    KFILDO/6/
    IOPT(1)=1
    IOPT(2)=10
    IOPT(3)=20
    IOPT(4)=10
    IOPT(5)=20
    IOPT(6)=0
    IOPT(8)=0
    NX=35
    NY=30
    CINT=0.
    ORIGIN=0.
    SMULT=10.
    SADD=0.
    ...
```

```
CALL PRTGR(KFILDO,GRID,NX,NY,CINT,ORIGIN,SMULT,SADD,IOPT,TITLE,IER)
```

The output file number is 6. An 11 x 11 array will be extracted from a 35 x 30 array and without contouring. The data will be scaled to display tenths. The minimum IX and JY values are each 10.

OUTPUT:

A printed matrix of the data or a printed contoured display of the data and diagnostic messages.

RESTRICTIONS:

None other than listed previously.

COMMENTS:

This routine performs primarily the same function as the NMC routine GRDPRT. The TDL routine PRTGRD in the W4LIB also performs the same function, but the data for PRTGRD are stored in TDL rather than NMC convention. The contouring algorithm is the same in both PRTGRD and PRTGR, although some rearrangement of code was made for PRTGR. The code was adapted from the DG Eclipse routine PRTGR and the VAX routine of the same name.

NONSYSTEM ROUTINES CALLED:

BESEL, SETUP.

LANGUAGE: FORTRAN 77.

LOCATION: MOS-2000 Library

BESEL

INTERPOLATES FOR GRIDPRINTING

Harry R. Glahn
April 1, 1995

PURPOSE: To perform the Bessel interpolation where possible for the gridprinting routine PRTGR. As an option, linear interpolation can be done.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

SETUP

SETS OPTIONS FOR PRTGR

Harry R. Glahn
April 1, 1995

PURPOSE: To set up certain information for the gridprinting routine PRTGR.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

PRSID

CONSTRUCTS MOS-2000 ID AND PARSES IT INTO COMPONENT PARTS

Harry R. Glahn
April 1, 1995

PURPOSE: To construct the 4-word MOS-2000 ID and its component parts from information supplied, according to Chapter 4 "Variable Description" in TDL Office Note "MOS-2000." The information supplied is in a format that is used for reading predictor MOS ID's, such as in RDPRED for U201. The routine PRSID1 can be used to parse the 4-word ID into its 15 component parts.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL PRSID(KFILDO,ITEMP,TEMP,ID,IDPARS,THRESH,ISTOP)

    KFILDO    - Unit number of output (print) file.  (INPUT)

    ITEMP(J)  - The information, along with TEMP below, that a MOS-2000
                program such as RDPRED would read to define variables (J=1,6):
                 1 - First ID = ID(1)
                 2 - Second ID = ID(2)
                 3 - Third ID = ID(3)
                 4 - Last portion of ID = ID(4)
                 5 - Fractional part of THRESH
                 6 - Tens exponent of ITEMP(5)
                (INPUT)

    TEMP      - Sign of threshold THRESH.  (CHARACTER*1)  (INPUT)

    ID(J)     - The predictor ID's (J=1,4).  (OUTPUT)

    IDPARS(J) - The MOS-2000 parsed ID (J=1,15):
                 1 - CCC (class of variable)
                 2 - FFF (subclass of variable)
                 3 - B (binary indicator)
                 4 - DD (data source, model number)
                 5 - V (vertical application)
                 6 - LLLL (lower level, 0 if only 1 level)
                 7 - UUUU (upper level)
                 8 - T (transformation)
                 9 - RR (run time offset in hours, always + and back in time)
                10 - O (time application)
                11 - HH (time period in hours)
                12 - τττ (projection in hours)
                13 - I (interpolation type)
                14 - S (smoothing indicator)
                15 - G (grid indicator)
                (OUTPUT)
```

THRESH	- The binary threshold associated with ID( ) and IDPARS( ). When the variable is neither binary nor a constant (CCC in the range 400-499), THRESH is set to zero and a diagnostic provided. (OUTPUT)
ISTOP	- Incremented by one each time an error is encountered. This happens only if a threshold THRESH is not zero and the variable is not a binary (IDPARS(3) = 0) nor a constant (CCC = 400-499). In that case, THRESH is set = 0, a diagnostic produced, and ISTOP incremented. (INPUT/OUTPUT)

EXAMPLE:

```
    CHARACTER*1 TEMP
    DIMENSION ID(4),IDPARS(15),JP(3)
    DATA KFILDO/6/
    DATA KFILP/5/
    ...
    READ(KFILP,103)(ITEMP(J),J=1,4),TEMP,ITEMP(5),ITEMP(6),(JP(J),J=1,3)
103 FORMAT(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2)
    CALL PRSID(KFILDO,ITEMP,TEMP,ID,IDPARS,THRESH,ISTOP)
```

The default output file number is 6. The variable information has been read into ITEMP( ), TEMP, and JP( ) on Unit No. 5. ITEMP( ) and TEMP are furnished to PRSID. JP( ) is already in the format needed. The information is processed into ID( ), IDPARS( ), and THRESH.

OUTPUT:

Diagnostic messages will be to Unit No. KFILDO only with the /D compile option.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES CALLED:

None

LANGUAGE: FORTRAN 77

LOCATION: moslib

BASICP

COMPUTES BASIC MOS-2000 PREDICTOR ID

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To construct a basic 4-word MOS-2000 ID by omitting certain processing information from the full MOS-2000 ID.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS</u>:

```
    CALL BASICP(KFILDO,IDPARS,JD)

    KFILDO    - Unit number of output (print) file.  (INPUT)

    IDPARS(J) - The MOS-2000 parsed ID (J=1,15):
                 1 - CCC (class of variable)
                 2 - FFF (subclass of variable)
                 3 - B (binary indicator)
                 4 - DD (data source, model number)
                 5 - V (vertical application)
                 6 - LLLL (lower level, 0 if only 1 level)
                 7 - UUUU (upper level)
                 8 - T (transformation)
                 9 - RR (run time offset in hours, always + and back in time)
                10 - O (time application)
                11 - HH (time period in hours)
                12 - τττ (projection in hours)
                13 - I (interpolation type)
                14 - S (smoothing indicator)
                15 - G (grid indicator)
                (INPUT)

    JD(J)     - The basic predictor ID's (J=1,4).  This is the same as ID( )
                which corresponds to IDPARS( ), except for the omitted parts:
                 3 - B (binary indicator)
                 8 - T (transformation)
                13 - I (interpolation type)
                14 - S (smoothing indicator)
                15 - G (grid indicator)
                (OUTPUT)
```

<u>EXAMPLE</u>:

```
    DIMENSION IDPARS(15),JD(3)
    DATA KFILDO/6/
    ...
    CALL BASICP(KFILDO,IDPARS,JD)
```

The default output file number is 6. The basic variable ID is put into JD( ) from IDPARS( ).

OUTPUT:

None.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES CALLED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

SORT16

SORTS VALUES IN AN ARRAY

Harry R. Glahn
June 20, 1995

PURPOSE: To sort low to high a list, each value composed of 128 bits. This would characteristically be a list of MOS-2000 variables. An associated variable is returned with the values indicating where the variables were relocated to. This variable can then be used by subroutine XCHANG to exchange another variable so that its location will match the reordered values. SORT16 is used in RDPRED in U201 and RDVRBL in U600, and may have other uses.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL SORT16(JD,INDEX,NVRBL)
```

JD(I,J)	- List to sort (I=1,4) (J=1,NVRBL). (INPUT-OUTPUT)
INDEX(J)	- Work array that upon return will indicate for each input value JD(,J) where it was relocated to in JD(,) (J=1,NVRBL). (OUTPUT)
NVRBL	- Number of values in JD(,) and INDEX( ). (INPUT)

EXAMPLE:

```
DIMENSION ID(4,NVRBL),INDEX(NVRBL)
...
CALL SORT16(JD,INDEX,NVRBL)
```

The basic variable JD(,) list (which in U201 and U600 is the full MOS-2000 variable ID sans certain processing information) is furnished for sorting. It is dimensioned so that when it is received in SORT16 as REAL*16 on the HP workstation, sorting can be done as a vector.

OUTPUT:

None.

RESTRICTIONS:

This routine is different on the HP 32-bit workstation than on the 64-bit CRAY to the extent that the variable receiving JD(,), is REAL*16 on the HP workstation which accommodates the 128 bits, but since the word length on the CRAY is 64 bits, there are actually 256 bits passed. This is dealt with by having two DOUBLE PRECISION REAL words per variable in SORT16 on the CRAY.

NONSYSTEM ROUTINES CALLED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

TIMPR

TIME STAMPS OUTPUT

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To print a 20-character string along with the current date/time.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
CALL TIMPR(KFILDS,KFILDO,ITITLE)
```

<u>KFILDS</u>	- Default unit number of output (screen) file. (INPUT)
<u>KFILDO</u>	- Default unit number of output (print) file. (INPUT)
<u>ITITLE</u>	- 20-character string to identify the date/time being written. (CHARACTER*20) (INPUT)

<u>EXAMPLE</u>

```
DATA KFILDS/1/
DATA KFILDO/6/
...

CALL TIMPR(KFILDS,KFILDO,'THIS IS A TEST      ')
```

The unit number for printing the message "THIS IS A TEST   " and date/time is 6. When TIMPR is compiled with the /D option, the same information is written to Unit No. 1, which can be the current console screen.

<u>OUTPUT</u>:

Message and data/time will be written to Unit No. KFILDO, and if TIMPR is compiled with the /D option the same information will be written to Unit No. KFILDS.

<u>RESTRICTIONS</u>:

None. The system routine FDATE is used to get the current date/time.

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: MOS-2000 Library

PSLLIJ

CONVERTS LATITUDE/LONGITUDE TO GRID COORDINATES FOR POLAR STEREOGRAPHIC MAP

Harry R. Glahn
April 1, 1995

PURPOSE: To convert a point expressed in latitude and longitude to grid coordinates on a polar stereographic map projection. Latitude and Longitude are in the MOS-2000 convention--plus for north latitude, minus for south latitude, and plus for west longitude with a wrap to 360 across 180 degrees.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL PSLLIJ(KFILDO,ALAT,ALON,XMESHL,ORIENT,XLAT,XLATLL,XLONLL,XI,YJ)
```

KFILDO	- Unit number of output (print) file. (INPUT)
ALAT	- North latitude in degrees of the point for which the grid coordinates are wanted. Minus for south latitude. (INPUT)
ALON	- West longitude in degrees of the point for which the grid coordinates are wanted. This can vary from 0 through 360. (INPUT)
XMESHL	- Mesh length in meters at XLAT degrees N. latitude. (INPUT)
ORIENT	- Orientation of grid in degrees west longitude. This is the longitude at which the columns for the grid are vertical. (INPUT)
XLAT	- North latitude in degrees at which XMESHL applies. (INPUT)
XLATLL	- North latitude of lower left (1,1) corner point of the grid. (INPUT)
XLONLL	- West longitude of lower left (1,1) corner point of the grid. (INPUT)
XI	- IX (left to right) gridpoint number of the point ALAT, ALON, considering the lower left point to be (1,1). (OUTPUT)
YJ	- JY (bottom to top) gridpoint number of the point ALAT, ALON, considering the lower left point to be (1,1). (OUTPUT)

EXAMPLE

```
    DATA KFILDO/6/
    DATA ALAT=80.
   1     ALON=120.
   2     XMESHL=40000.
   3     ORIENT=90.
   4     XLAT=60.
   5     XLATLL=10.
   6     XLONLL=140.
    ...

    CALL PSLLIJ(KFILDO,ALAT,ALON,XMESHL,ORIENT,XLAT,XLATLL,XLONLL,XI,YJ)
```

The default output unit number is 6. The left to right and bottom to top gridpoint positions will be returned in XI and YJ, respectively, for the point 80 degrees N. and 120 degrees W. The gridlength is 40 km at 60 degrees N. The orientation of the grid is 90 degrees W. The lower left corner point of the grid is at 10 degrees N., 140 degrees W.

OUTPUT:

None except diagnostics with the /D compile option.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

LMLLIJ

CONVERTS LATITUDE/LONGITUDE TO GRID COORDINATES FOR LAMBERT MAP

Harry R. Glahn
April 1, 1995

PURPOSE: To convert a point expressed in latitude and longitude to grid coordinates on a Lambert conformal map projection. Latitude and Longitude are in the MOS-2000 convention--plus for north latitude, minus for south latitude, and plus for west longitude with a wrap to 360 across 180 degrees.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL LMLLIJ(KFILDO,ALAT,ALON,XMESHL,ORIENT,XLAT,XLATLL,XLONLL,XI,YJ)
```

KFILDO	- Unit number of output (print) file. (INPUT)
ALAT	- North latitude in degrees of the point for which the grid coordinates are wanted. Minus for south latitude. (INPUT)
ALON	- West longitude in degrees of the point for which the grid coordinates are wanted. This can vary from 0 through 360. (INPUT)
XMESHL	- Mesh length in meters at XLAT degrees N. latitude. (INPUT)
ORIENT	- Orientation of grid in degrees west longitude. This is the longitude at which the columns for the grid are vertical. (INPUT)
XLAT	- North latitude in degrees at which XMESHL applies. (INPUT)
XLATLL	- North latitude of lower left (1,1) corner point of the grid. (INPUT)
XLONLL	- West longitude of lower left (1,1) corner point of the grid. (INPUT)
XI	- IX (left to right) gridpoint number of the point ALAT, ALON, considering the lower left point to be (1,1). (OUTPUT)
YJ	- JY (bottom to top) gridpoint number of the point ALAT, ALON, considering the lower left point to be (1,1). (OUTPUT)

EXAMPLE

```
    DATA KFILDO/6/
    DATA ALAT=80.
   1     ALON=120.
   2     XMESHL=40000.
   3     ORIENT=90.
   4     XLAT=60.
   5     XLATLL=10.
   6     XLONLL=140.
    ...

    CALL LMLLIJ(KFILDO,ALAT,ALON,XMESHL,ORIENT,XLAT,XLATLL,XLONLL,XI,YJ)
```

The default output unit number is 6. The left to right and bottom to top gridpoint positions will be returned in XI and YJ, respectively, for the point 80 degrees N. and 120 degrees W. The gridlength is 40 km at 60 degrees N. The orientation of the grid is 90 degrees W. The lower left corner point of the grid is at 10 degrees N., 140 degrees W.

OUTPUT:

None except diagnostics with the /D compile option.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

LMIJLL

CONVERTS GRID COORDINATES TO LATITUDE/LONGITUDE FOR LAMBERT MAP

Harry R. Glahn
April 1, 1995

PURPOSE: To convert a point expressed in grid coordinates on a Lambert conformal map projection to latitude and longitude. Latitude and Longitude are in the MOS-2000 convention--plus for north latitude, minus for south latitude, and plus for west longitude with a wrap to 360 across 180 degrees.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL LMIJLL(KFILDO,XI,YJ,XMESHL,ORIENT,XLAT,XLATLL,XLONLL,ALAT,ALON,IER)
```

KFILDO	- Unit number of output (print) file. (INPUT)
XI	- IX (left to right) gridpoint number of the point for which the latitude and longitude are wanted, considering the lower left point to be (1,1). (INPUT)
YJ	- JY (bottom to top) gridpoint number of the point for which the latitude and longitude are wanted, considering the lower left point to be (1,1). (INPUT)
XMESHL	- Mesh length in meters at XLAT degrees N. latitude. (INPUT)
ORIENT	- Orientation of grid in degrees west longitude. This is the longitude at which the columns for the grid are vertical. (INPUT)
XLAT	- North latitude in degrees at which XMESHL applies. (INPUT)
XLATLL	- North latitude of lower left (1,1) corner point of the grid. (INPUT)
XLONLL	- West longitude of lower left (1,1) corner point of the grid. (INPUT)
ALAT	- North latitude in degrees of the point XI,YJ. Minus for south latitude. (OUTPUT)
ALON	- West longitude in degrees of the point XI,YJ. This can vary from 0 through 360. (OUTPUT)
IER	- Status return. 0 = good return. 105 = The requested XI,YJ point is in the forbidden zone, i.e., off the Lambert map in the open space where the cone is cut. In this case, ALAT and ALON are returned = 9999.

EXAMPLE

```
     DATA KFILDO/6/
     DATA XI=10.
    1     YJ=12.
    2     XMESHL=40000.
    3     ORIENT=90.
    4     XLAT=60.
    5     XLATLL=10.
    6     XLONLL=140.
     ...

     CALL LMIJLL(KFILDO,XI,YJ,XMESHL,ORIENT,XLAT,XLATLL,XLONLL,ALAT,ALON,IER)
```

The default output unit number is 6. The latitude and longitude of the point in grid coordinates XI = 10 (left to right) and YJ = 12 (bottom to top) will be returned in ALAT and ALON, respectively. The gridlength is 40 km at 60 degrees N. The orientation of the grid is 90 degrees W. The lower left corner point of the grid is at 10 degrees N., 140 degrees W.

OUTPUT:

None except diagnostics with the /D compile option. Note the IER = 105 error return.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

SKIPR

SKIPS RECORDS

Harry R. Glahn
April 1, 1995

PURPOSE: To skip records on a file in the TDLPACK format until a specified date/time has been passed. SKIPR should have use for any program that writes a gridpoint type file. SKIPWR performs similar, as well as additional, functions for vector data.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL SKIPR(KFILDO,KFILIO,KSKIP,NTOTBY,NTOTRC,L3264B,IER)
```

KFILDO	- Unit number of output (print) file. (INPUT)
KFILIO	- Unit number of file to read and skip records. (INPUT)
KSKIP	- When non-zero, indicates that the output file is to be moved forward until all data for date KSKIP have been skipped. KSKIP is in the format YYYYMMDDHH. (INPUT)
NTOTBY	- The total number of bytes on the file. It is initialized to zero, and the bytes counted when skipping records. (OUTPUT)
NTOTRC	- The total number of records on the file. It is initialized to zero, and the records counted when skipping records. (OUTPUT)
L3264B	- Integer word length of machine being used, either 32 or 64. (INPUT)
IER	- Status return. 0 = Good return. 143 = Error when skipping records. Other values can be generated by called subroutine, UNPKBG. (OUTPUT)

EXAMPLE

```
PARAMETER (L3264B=32)
DATA KFILDO/6/
DATA KFILIO/20/
...
CALL SKIPR(KFILDO,KFILIO,1994123112,NTOTBY,NTOTRC,L3264B,IER)
```

The default output unit number is 6. The unit number for the output file is 20. Since KSKIP ≠ 0, the file is moved forward until all data including those for December 31, 1994, 1200 UTC, have been passed. L3264B, set by parameter, would be 64 when running on the CRAY. The total number of bytes (records) skipped will be returned in NTOTBY (NTOTRC).

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO, as well as a summary of the tape operations performed.

RESTRICTIONS:

None other than stated above. Because every precaution must be taken to not overwrite good data, the user should always check the IER status return.

NONSYSTEM ROUTINES USED:

UNPKBG

LANGUAGE: FORTRAN 77

LOCATION: moslib

PRSID1

PARSES MOS-2000 ID INTO COMPONENT PARTS

Harry R. Glahn
June 1, 1995

PURPOSE: To parse the 4-word MOS-2000 ID into its 15 component parts. The difference between this routine and PRSID is that PRSID uses information as read from a file, and composes ID( ) as well as the component parts in IDPARS( ). This routine only parses ID( ) into IDPARS( ).

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL PRSID(KFILDO,ID,IDPARS)

    KFILDO    - Unit number of output (print) file.  (INPUT)

    ID(J)     - The predictor ID's (J=1,4).  (INPUT)

    IDPARS(J) - The MOS-2000 parsed ID (J=1,15):
                 1 - CCC (class of variable)
                 2 - FFF (subclass of variable)
                 3 - B (binary indicator)
                 4 - DD (data source, model number)
                 5 - V (vertical application)
                 6 - LLLL (lower level, 0 if only 1 level)
                 7 - UUUU (upper level)
                 8 - T (transformation)
                 9 - RR (run time offset in hours, always + and back in time)
                10 - O (time application)
                11 - HH (time period in hours)
                12 - τττ (projection in hours)
                13 - I (interpolation type)
                14 - S (smoothing indicator)
                15 - G (grid indicator)
                (OUTPUT)
```

EXAMPLE:

```
    DIMENSION ID(4),IDPARS(15)
    DATA KFILDO/6/
    CALL PRSID(KFILDO,ID,IDPARS)
```

The default output file number is 6. The 4-word MOS 2000 ID is in ID( ), and the 15 component parts are returned in IDPARS( ).

OUTPUT:

No printed output is provided.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES CALLED:

None

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

RDSTAD

READS AND ALPHABETIZES A STATION LIST AND DIRECTORY

Harry R. Glahn
April 1, 1995

PURPOSE: To read a list of station call letters and associated information from a station directory. The list, composed of up to 8 characters per station can (1) be taken from the directory, in which case all stations in the directory will be in the list or (2) be on a separate file in which case the list is ended with the terminator '99999999'. The stations in the list returned will be in alphabetical order provided the directory is alphabetical. Any duplicate stations found in the separate list will be kept, and a diagnostic provided. Stations in the list that are not in the directory will be put at the end of the list.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL RDSTAD(KFILDO,IP4,IP5,KFILD,NEW,CCALL,CCALLD,NAME,NELEV,IWBAN,
            STALAT,STALON,ITIMEZ,IFOUND,ND1,NSTA,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

IP4 - When IP4 > 0, the station list (call letters only) will be written to Unit No. IP4. If there are input errors, the station list will be written to the default output file Unit No. KFILDO as well as to IP4. (INPUT)

IP5 - When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5. (INPUT)

KFILD(J) - Unit number from which to read the station list (J=1) and station directory (J=2). It is assumed the files have been opened. If all stations are to be used from the directory, KFILD(1) = KFILD(2). (INPUT)

NEW - 1 when new ICAO station identifiers (call letters) are being used (columns 1-8 in the station directory);
0 when the old station call letters are being used (columns 10-17 in the station directory. (INPUT)

CCALL(K,J)- Array in which up to 6 sets (J=1,6) of 8 call letters per station are returned (K=1,NSTA). The list will be in alphabetical order provided the directory is that way. Stations not in the directory will be put at the end of the list. CCALL(,1) will contain the call letters to be used as station or location identifiers in the calling program; the other 5 sets (J=2,6) are taken from the directory. When KFILD(1) ≠ KFILD(2), the station list returned in CCALL(,1) will be from the file on Unit No. KFILD(1), and will be in the order found on the directory, normally alphabetized by ICAO station identifiers. When KFILD(1) = KFILD(2), the station list returned in CCALL(,1) will be from the station directory and will be in the order found on the directory; otherwise, the station list is read by subroutine RDC from unit No. KFILD(1). The call letters returned will be the ICAO station identifiers (columns 1-8) when NEW = 1, and will be the old call letters (columns 10-17) when NEW = 0. The other lists in CCALL(,J) always come from the directory, and for J=3,6 come from columns 83-90, 92-99, 101-108, and 110-117 in that order. The list in CCALL(,2) also comes from the directory, being the ICAO call letters (columns 1-8) when NEW = 0 and the old call letters (columns 10-17) when NEW = 1. That is, it is intended that when CCALL(,1) contains the ICAO station identifiers, CCALL(,2) will contain the old call letters, and vice versa. (CHARACTER*8) (OUTPUT)

<u>CCALLD</u>(K)	-	Call letters as read from the directory (K=1,ND1). Note that this dimension has to be only as large as the number of stations to be kept, not the number of stations in the directory (unless, of course, KFILD(1) = KFILD(2)). (CHARACTER*8) (INTERNAL)
<u>NAME</u>(K)	-	Names as read from the directory ordered according to CCALL(K,1) (K=1,NSTA). (CHARACTER*20) (OUTPUT)
<u>NELEV</u>(K)	-	Elevations of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)
<u>STALAT</u>(K)	-	Latitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)
<u>STALON</u>(K)	-	Longitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)
<u>TIMEZ</u>(K)	-	Time zone of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). This is the number of hours the station is different from (earlier than) UTC. (OUTPUT)
<u>IWBAN</u>(K)	-	WBAN numbers of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)
<u>IFOUND</u>(K)	-	Used to keep track of the stations found in the directory. (K=1,ND1). (INTERNAL)
<u>ND1</u>	-	First dimension of CCALL(,), and dimension of CCALLD( ), NELEV( ), IWBAN( ), STALAT( ), STALON( ), and IFOUND( ). (INPUT)
<u>NSTA</u>	-	The number of elements (stations) returned in CCALL(,) and the associated arrays. (OUTPUT)

```
    IER       - Status return.
                 0 = Good return.
                20 = Error or EOF reading KFILD(1) by RDC.  Aborts.
                21 = Too many stations for dimension ND1 when reading by RDC.
                     Aborts.
                33 = Error reading station directory on Unit No. KFILD(2).
                     Aborts.
                34 = List to be kept too long for dimension ND1.  Aborts with
                     print.
                35 = One or more stations not found in the directory.  Normal
                     return.
                36 = One or more stations are duplicates.  Normal return.
                37 = Both IER = 35 and 36 above have occurred.  Normal return.
                (OUTPUT)
```

EXAMPLE

```
    PARAMETER (ND1=500)
    CHARACTER*8 CCALL(ND1,6),CCALLD(ND1)
    CHARACTER*20 NAME(ND1)
    DIMENSION NELEV(ND1),IWBAN(ND1),STALAT(ND1),STALON(ND1),IFOUND(ND1)
    DIMENSION KFILD(2),IP(25)
    DATA IP/25*0/
    DATA KFILDO/12/
    DATA KFILD/28,29/,
   1     NEW/1/
    ...

    CALL RDSTAD(KFILDO,IP(4),IP(5),KFILD,NEW,CCALL,CCALLD,NAME,NELEV,
   1            IWBAN,STALAT,STALON,ITIMEZ,IFOUND,ND1,NSTA,IER)
```

The default output unit number is 12. The unit number to read the station list from is 28, and the station directory is on Unit No. 29. Since IP(4) and IP(5) = 0, the station list will not be printed, unless there are fatal errors, in which case the list and the error found will be on Unit No. 12. The NSTA ICAO identifiers are returned in CCALL(,1), along with the associated information in other columns of CCALL(,) and in the arrays as indicated above. Since the station list will be the ICAO station identifiers (NEW = 1), CCALL(,2) will contain the old call letters. The maximum number of stations to be returned is ND1. The station list is read with FORMAT(7(A8,1X)) until the terminator '99999999' is found. Normally, each record would contain only one station just for ease of user manipulation, but up to 7 stations could be in one record. A description of the station dictionary can be found elsewhere.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. When IP4 > 0, the station list (call letters only) will be written to Unit No. IP4. If there are station list input errors, the station list will be written to the default output file Unit No. KFILDO as well as to IP4. When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are directory input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5.

RESTRICTIONS:

None other than stated above.

COMMENTS:

This routine performs essentially the same function as RDSTAL, except that the stations are alphabetized in RDSTAD, provided the directory is alphabetized, and they are not in RDSTAL. If a station cannot be found in the directory, the last 5 columns of CCALL(,) will be blank. Note that all six columns of the directory are searched. That is, suppose a record in the directory contains 'KDEN   ', 'DEN   ', and 'DVX   ' in columns 1-8, 10-17, and 83-90, respectively. Either 'KDEN', 'DEN', or 'DVX' could be read by subroutine RDC, and KDEN would be returned in CCALL(,1). If KDEN changed to, say, 'KDUM' and the directory is updated, then 'KDUM' would be returned in CCALL(,1) and would be used in the calling program to identify output. This means that the directory record might be different on different data sets, but the MOS-2000 software should handle it, provided 'KDEN' is added to the entry in columns 92-99.

This routine is used when only one "group" of stations is being dealt with (e.g., in U201). For multiple groups, RDSTGA can be used.

When KFILD(1) ≠ KFILD(2), the input station list can be the default input file for the calling (main) program, or be a separate file.

It would be unusual to use the feature of KFILD(1) = KFILD(2) to return the complete station directory, unless one were to prepare a special (smaller) directory for some special purpose. However, if one does need the complete directory, RDSTAL will return it. Since there is no reordering of this directory, RDSTAL and RDSTAD operate the same way for this feature.

There are six routines which read a station list and return a list taken from either the first or second column of the directory, depending on the value of NEW; all six operate in the same way in this regard. Two routines deal with only a single list of stations and return the time zone. Four routines deal with multiple lists (groups) of stations. Three alphabetize the station list (by group) and three return the list in the order read. A group name is returned by two routines; this name is read immediately following the group list of stations with an A30 format. Their different characteristics are:

Routine	Single Station List	Multiple Station List	Group Name	Time Zone	Alphabetize
RDSTAL	X			X	
RDSTAD	X			X	X
RDSTGN		X			
RDSTGA		X			X
RDSTNL		X	X		
RDSTNA		X	X		X

RDSTAD is used in U201, U700, U900, U351, and U150.

Many changes were made in October 2000 to assure that the station wanted from the directory was returned. Any directory entry can have up to 6 station identifiers, the leftmost being called the ICAO identifier, and next column being called the SAO call letters, although that nomenclature does not have to be followed. The rule followed is that the leftmost identifier that matches the identifier in the input list is the one used. That is, if a link I0V1 was in a particular record in a column 2-6 and was also in the first column, then the station in the first column is the one returned. The older version of RDSTAD did not assure this, and the result depended on the order in which the records were encountered.

Diagnostics were also improved, so that not only duplicate and missing stations are identified, but duplicates that may be caused by using a link for one of the stations and when a station is kept because of a link and the column (2 through 6) where the identifier occurred. In this context, SAO call letters are in Col. 2 when ICAO identifiers are being used (NEW = 1) and ICAO identifiers are in Col. 2 when SAO call letters are being used (NEW = 0).

In addition, station directory information is not printed to KFILDO except when IP5 = KFILDO or the error return is fatal. The table below shows the print structure as a function of the values of IER and IP5.

		IER =	
	0	33,34	35,36,37
IP5 = 0	---	KFILDO	---
IP5 = KFILDO	KFILDO	KFILDO	KFILDO
IP5 = IP5	IP5	KFILDO IP5	IP5

<u>NONSYSTEM ROUTINES USED</u>:

RDC

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: MOS-2000 Library

XCHANG

ORDERS VALUES IN AN ARRAY

Harry R. Glahn
June 20, 1995

PURPOSE: To order the values in an array according to values in an index array provided. The array of values to order would characteristically be an array associated with MOS-2000 IDs which were ordered by subroutine SORT16, and the index array would have been produced by SORT16. SORT16 and XCHANG are used in RDPRED in U201 and RDVRBL in U600, and may have other uses.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL XCHANG(IVAR,NUMROW,INDEX,IWORK,NVRBL)
```

IVAR(I,J)	- Variable to order (I=1,NUMROW) (J=1,NVRBL). Each "row" I is ordered. (INPUT-OUTPUT)
NUMROW	- The number of "rows" in variable IVAR(,). (INPUT)
INDEX(J)	- The index variable that can be provided by SORT16. It contains the locations (J=1,NUMROW) in IVAR(,) that the original values in IVAR(,J) went to. INDEX( ) is unchanged by XCHANG.
IWORK(J)	- Work array (J=1,NUMROW). (INTERNAL)
NVRBL	- Number of values in IVAR(,). Used as dimension of IVAR(,), INDEX( ), and IWORK( ). (INPUT)

EXAMPLE:

```
DIMENSION JD(4,NVRBL),ID(4,NVRBL),INDEX(NVRBL),IWORK(NVRBL)
...
CALL SORT16(JD,INDEX,NVRBL)
CALL XCHANG(ID,4,INDEX,IWORK,NVRBL)
```

The basic variable JD(,) list, which in U201 and U600 is the full MOS-2000 variable ID sans certain processing information, has been sorted by SORT16, and the array ID(I,J) (I=1,4) (J=1,NVRBL) is ordered in the same way, so that ID(,J) corresponds to JD(, J). INDEX is unchanged and can be used in another call to XCHANG.

OUTPUT:

None.

RESTRICTIONS:

None other than those stated above.

NONSYSTEM ROUTINES CALLED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

GSTORE

STORES DATA INTO INTERNAL MOS-2000 STORAGE SYSTEM

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To store data into the internal MOS-2000 storage structure. All accounting data are taken care of so that the data can be retrieved by GFETCH. A certain amount of core is provided for storage and when that is filled, the data are stored on disk. The user need not worry about the internals of this system. GSTORE is used by U201 and other programs.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
    CALL GSTORE(KFILDO,KFIL,ID,NSLAB,LSTORE,ND9,LITEMS,DATA,NWORDS,NPACK,
   1            NRR,NDATE,CORE,ND10,LASTL,NBLOCK,LASTD,NSTORE,L3264B,IER)
```

<u>KFILDO</u> - Unit number of output (print) file. (INPUT)

<u>KFIL</u> - Unit number for writing the data in DATA( ) to disk when CORE( ) is full. The file is opened as scratch on the first entry to GSTORE. (INPUT)

<u>ID</u>(J) - The 4-word ID of the variable to store (J=1,4). This is the way the data will be identified and recalled by GFETCH. (INPUT)

<u>NSLAB</u> - This is used for different purposes for different programs and is a value to store in LSTORE(10,). For U201, it is the number of the grid combination. Vector programs may use it for the number of the input data set the data being stored came from. Use "1" for routines for which this is not meaningful. (INPUT)

<u>LSTORE</u>(L,J) - The array to hold information about the data stored (L=1,12) (J=1,LITEMS). This is initialized to zero on the first entry to GSTORE. (INPUT-OUTPUT)

<u>ND9</u> - The second dimension of LSTORE(,). (INPUT)

<u>LITEMS</u> - The number of items (columns) in LSTORE(,) that are filled, maximum of ND9. This is initialized to zero as needed on the first entry to GSTORE. LITEMS is modified if compaction is done by GCPAC. (INPUT-OUTPUT).

<u>DATA</u>(J) - The data to store (J=1,NWORDS). (INPUT)

<u>NWORDS</u> - The number of words in DATA( ) to store. NWORDS is stored in LSTORE(6,). (INPUT)

<u>NPACK</u> - 2 when the data in DATA( ) are in the TDLPACK format. 1 when data are not packed. NPACK is stored in LSTORE(7,). (INPUT)

NRR - The last date/time for which these data will be needed. When that has been passed, GCPAC will arrange for the data to be discarded. For most programs, NRR can be 0; for U201, when the lookback feature is used, NRR must be calculated and provided. NRR is stored in LSTORE(12,). (INPUT)

NDATE - Date/time of the data to be stored in format YYYYMMDDHH. This is stored in LSTORE(8,). (INPUT)

CORE(J) - The linear array where the data are to be stored when space is available (J=1,ND10). Normally, this is used only within GSTORE, GFETCH, LMSTR1, LMSTR2, and GCPAC. (INPUT-OUTPUT)

ND10 - Dimension of CORE( ). (INPUT)

LASTL - The last location in CORE( ) used. Initialized to zero on first entry to GSTORE. (OUTPUT)

NBLOCK - The block size in words of the random disk file on which to store the data when CORE( ) is full. If the size of the records is known, then NBLOCK should be large enough to hold the data. If NBLOCK is large and the NWORDS is small, then space will be wasted on the disk and the I/O may be larger than needed. (INPUT)

LASTD - The total number of physical records on disk. Initialized to zero on first entry to GSTORE. (OUTPUT)

NSTORE - Incremented by one each time GSTORE is entered. It is a running count from the beginning of the program. This count is maintained internally and furnished to the user in case it is needed. (OUTPUT)

L3264B - Must have the value of 32 or 64, indicating the word length of the machine being used. (INPUT)

IER - Status return.
 0 = Good return.
 50 = Data cannot be stored. No space is available in LSTORE(,).
 9xx= A system error number will be returned from subroutine WRDISK when the file can't be opened or data written to the file.
 (OUTPUT)

EXAMPLE

```
PARAMETER (ND10=10000)
PARAMETER (ND9=100)
DIMENSION CORE(ND10)
DIMENSION DATA(500)
DIMENSION LSTORE(12,ND9)
DIMENSION ID(4)
DATA KFILDO/6/,
  1     KFIL/10/
   DATA L3264B/32/
   DATA NSLAB/1/
   DATA NDATE/1995010100/
   ...

   CALL GSTORE(KFILDO,KFIL,ID,NSLAB,LSTORE,ND9,LITEMS,DATA,200,1,0,
  1            NDATE,CORE,ND10,LASTL,300,LASTD,NSTORE,L3264B,IER)
```

The default output unit number is 6, and the unit number for the MOS-2000 disk storage is 10. ID( ) is dimensioned 4 and holds the ID of the data to store. NSLAB may be meaningless; for U201 it is the grid combination; for other programs, it may have another value to store and retrieve in LSTORE(6,). For instance, in U600, it is the sequence number of the input data sets. LSTORE must be set up with dimensions, but is used only internally by GSTORE, GFETCH, LMSTR1, LMSTR2, and GCPAC. ND9 is the second dimension of LSTORE, and upon return from GSTORE, LITEMS will hold the number of items in LSTORE(,), maximum of ND9. The data to be stored are in DATA( ), and 200 values are to be stored. The data are unpacked (NPACK = 1), and need not be kept past the current date (NRR = 0). The current date, the date of the data, is NDATE = 1995010100. ND10 = 10000 locations are available for storage of data in CORE( ); when full, the disk will be used, which has a block size of 300 words. The number of physical records on disk upon return from GSTORE is LASTL. Also upon return, LASTD is the last location in CORE( ) used, and NSTORE is the running count of how many times GSTORE has been called. The subroutine is called from a 32-bit machine. IER will be zero upon return unless a problem has been encountered.

<u>OUTPUT</u>:

Diagnostic messages will be written to Unit No. KFILDO. The scratch file on Unit No. KFIL is opened even though it may not be needed. A diagnostic is provided if it can't be opened. This is not fatal at this point, but if the file is needed, a system error return will be provided.

<u>RESTRICTIONS</u>:

LSTORE(,) is for the exclusive use of GSTORE, GFETCH, LMSTR1, LMSTR2, and GCPAC. While information could be obtained from it, it should not be modified, and any information actually needed should be returned in the call sequence. Care must be taken in setting the dimensions of the variables involved, as well as the block size for the MOS-2000 storage system disk.

Surrogates of LASTL, LASTD, and NSTORE are kept in COMMON BLOCK SVLAST. GSTORE and GCPAC use this common block; other programs should not use it.

<u>NONSYSTEM ROUTINES USED</u>:

WRDISK

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

GFETCH

FETCHES DATA FROM INTERNAL MOS-2000 STORAGE SYSTEM

Harry R. Glahn
April 1, 1995

PURPOSE: To fetch data from the internal MOS-2000 storage structure. The data are returned unpacked, whether or not they are packed according to the MOS-2000 TDLPACK format in the file system. All accounting data are taken care of so that other data can be stored by GSTORE. A certain amount of core is provided for storage and when that is filled, the data are stored on disk. The user need not worry about the internals of this system. GFETCH is used by U201 and other programs.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL GFETCH(KFILDO,KFIL,ID,LOCPRD,LSTORE,ND9,LITEMS,IS0,IS1,IS2,IS4,ND7,
   1            IPACK,IWORK,DATA,NDX,NWORDS,NPACK,NDATE,NTIMES,
   2            CORE,ND10,NBLOCK,NFETCH,NSLAB,MISSP,MISSS,L3264B,ITIME,IER)
```

KFILDO	- Unit number of output (print) file. (INPUT)
KFIL	- Unit number for reading the data from disk when not in CORE( ). (INPUT)
ID(J)	- The 4-word ID of the variable to fetch (J=1,4). (INPUT)
LOCPRD	- The location of the variable ID( ) in the variable list when GFETCH is accessed from PRED1. When GFETCH is accessed by U201 from OPTION or computation routines called from OPTION, LOCPRD must be 7777. For other uses, set LOCPRD = 7777. This may be stored in LSTORE(11,). (INPUT)
LSTORE(L,J)	- The array to hold information about the data stored (L=1,12) (J=1,LITEMS). (INPUT-OUTPUT)
ND9	- The second dimension of LSTORE(,). (INPUT)
LITEMS	- The number of items (columns) in LSTORE(,) that are filled, maximum of ND9. LITEMS is modified if compaction is done by GCPAC. (OUTPUT).
IS0(J)	- The Section 0 decoded information from TDLPACK. (OUTPUT)
IS1(J)	- The Section 1 decoded information from TDLPACK. (OUTPUT)
IS2(J)	- The Section 2 decoded information from TDLPACK. (OUTPUT)
IS4(J)	- The Section 4 decoded information from TDLPACK, except for the data themselves. (OUTPUT)
ND7	- Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). A value of 54 is sufficient.

IPACK(J) - Work array (J=1,NDX). (INTERNAL)

IWORK(J) - Work array (J=1,NDX). (INTERNAL)

DATA(J) - The returned data (J=1,NWORDS). (OUTPUT)

NDX - Dimension of DATA( ). Dimension of IPACK( ) and IWORK( ) must be at least this large. (INPUT)

NWORDS - The number of words in returned DATA( ). (OUTPUT)

NPACK - 2 when the data in DATA( ) were in the TDLPACK format. 1 when data were not packed. (OUTPUT)

NDATE - Date/time of the data to be fetched in format YYYYMMDDHH. (INPUT)

NTIMES - The number of times, including this one, that these particular data have been fetched. (OUTPUT)

CORE(J) - The linear array where the data are stored when space is available (J=1,ND10). Normally, this is used only within GSTORE and GFETCH. (INPUT-OUTPUT)

ND10 - Dimension of CORE( ). (INPUT)

NBLOCK - The block size in words of MOS-2000 random disk file on which the data are stored when CORE( ) is full. This must match the value provided to GSTORE, and must not change during the program run.

NFETCH - Incremented by one each time GFETCH is entered. It is a running count from the beginning of the program. This count is maintained internally and furnished to the user in case it is needed. (OUTPUT)

NSLAB - For U201, the number of the slab in DIR(, ,) and in NGRID(,) defining the characteristics of this grid. For other programs, this number can mean something else. In U600, it is the "model number" from whence the data came. It is stored and retrieved in LSTORE(10,) without change. (OUTPUT)

MISSP - The primary missing data value indicator as taken from TDLPACK. When MISSP = 0, it signifies no primary (or secondary) values are present. Normally, this will be 9999 when non-zero. Note that MISSP is an integer. (OUTPUT)

MISSS - The secondary missing data value indicator as taken from TDLPACK. When MISSS = 0, it signifies no secondary values are present. MISSS will not be non-zero unless MISSP is also non-zero. Normally, this will be 9999 when non-zero. Note that MISSP is an integer. (OUTPUT)

<u>L3264B</u> - Must have the value of 32 or 64, indicating the word length of the machine being used. (INPUT)

<u>ITIME</u> - Serves two functions. Equals zero when an adjustment in time using RR is not to be made; ≠ otherwise. Most programs will probably use ITIME = 0. However, U201 routines require ITIME ≠ 0 because data for previous date/times may be needed. However, for other programs, such as U600, the variable is present for the time in NDATE. When ITIME < 0, this value is to be stored in LSTORE(12,). This is used in VRBL61 and allows a maximum tau to be calculated for predictands in LMSTR6. (INPUT)

<u>IER</u> - Status return.
- 0 = Good return.
- 47 = Data cannot be found.
- 48 = Dimension NDX is not large enough for DATA( ) to hold the data.
- 49 = Number of words returned from UNPACK not what's expected.
- 9xx= A system error number will be returned from subroutine RDDISK when the file can't be read correctly.

Other returns may come from subroutine UNPACK. (OUTPUT)

<u>EXAMPLE</u>

```
      PARAMETER (NDX=1000)
      PARAMETER (ND7=54)
      PARAMETER (ND9=100)
      PARAMETER (ND10=10000)
C
      DIMENSION DATA(NDX)
      DIMENSION IS0(ND7),IS1(ND7),IS2(ND7),IS4(ND7)
      DIMENSION LSTORE(12,ND9)
      DIMENSION CORE(ND10)
      DIMENSION ID(4)
C
      DATA KFILDO/6/,
     1     KFIL/10/
      DATA L3264B/32/
      DATA NDATE/1995010100/
      DATA LOCPRD/7777/
      DATA ITIME/0/
      ...

      CALL GFETCH(KFILDO,KFIL,ID,LOCPRD,LSTORE,ND9,LITEMS,IS0,IS1,IS2,IS4,ND7,
     1            IPACK,IWORK,DATA,NDX,NWORDS,NPACK,NDATE,NTIMES,
     2            CORE,ND10,NBLOCK,NFETCH,NSLAB,MISSP,MISSS,L3264B,ITIME,IER)
```

The default output unit number is 6, and the unit number for the MOS-2000 disk storage is 10. ID( ) is dimensioned 4 and holds the ID of the data to fetch. LOCPRD is usually 7777. LSTORE must be set up with dimensions, but is used only internally by GSTORE, GFETCH, LMSTR1, LMSTR2, and GCPAC.

ND9 is the second dimension of LSTORE, and upon return from GSTORE, LITEMS will hold the number of items in LSTORE(,), maximum of ND9. The full identification of the data, when they have been packed in TDLPACK format, will be returned in IS0( ), IS1( ), IS2( ), and IS4( ) as appropriate. The data fetched are in DATA( ), and NWORDS values are returned. The data returned were stored packed or unpacked, as indicated by NPACK, but are returned unpacked. The date for which the data are to be fetched is NDATE = 1995010100. ND10 = 10000 locations are available for storage of data in CORE( ); when full, the disk will be used, which has a block size of 300 words. Upon return, NFETCH is the running count of how many times GFETCH has been called. NSLAB is used by U201, but probably has no meaning for other programs. The missing value indicators, when taken from TDLPACK, are MISSP and MISSS for primary and secondary, respectively; while this is INTEGER, the actual values in DATA( ) are REAL. The subroutine is called from a 32-bit machine. It is not called from U201, and ITIME = 0. IER will be zero upon return unless a problem has been encountered.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

LSTORE(,) is for the exclusive use of GSTORE, GFETCH, LMSTR1, LMSTR2, and GCPAC. While information could be obtained from it, it should not be modified, and any information actually needed should be returned in the call sequence. Care must be taken in setting the dimensions of the variables involved, as well as the block size for the MOS-2000 storage system disk.

NONSYSTEM ROUTINES USED:

RDDISK, UNPACK

LANGUAGE: FORTRAN 77

LOCATION: moslib

WRDISK

WRITES DATA INTO INTERNAL MOS-2000 DISK FILE

Harry R. Glahn
April 1, 1995

PURPOSE: To write for GSTORE the data needed to be saved on disk in the internal MOS-2000 file structure. All accounting data are taken care of so that the data can be retrieved by GFETCH and new data stored by GSTORE. A certain amount of core is provided for storage and when that is filled, the data are stored on disk, blocked according to input parameters so that the data can be retrieved as random access. Blocking is used for space efficiency, because the records are of differing sizes, unknown beforehand. The user need not worry about the internals of this system. WRDISK is used by U201 and other programs.

RESTRICTIONS:

None other than the general MOS-2000 guidelines, etc.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

RDDISK

READS DATA FROM INTERNAL MOS-2000 DISK FILE

Harry R. Glahn
April 1, 1995

<u>PURPOSE</u>: To read for GFETCH the data needed from disk in the internal MOS-2000 file structure. All accounting data are taken care of. A certain amount of core is provided for storage and when that is filled, the data are stored on disk, blocked according to input parameters so that the data can be retrieved as random access. Blocking is used for space efficiency, because the records are of differing sizes, unknown beforehand. The user need not worry about the internals of this system. RDDISK is used by U201 and other programs.

<u>RESTRICTIONS</u>:

None other than the general MOS-2000 guidelines, etc.

<u>NONSYSTEM ROUTINES USED</u>:

None

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: MOS-2000 Library

LMSTR1

INITIALIZES MSTORE AND PREPARES LSTORE FOR GCPAC AFTER DAY 1

Harry R. Glahn
June 1, 1995

PURPOSE: To initialize MSTORE(,) after day 1, and to prepare LSTORE(,) for elimination of unneeded entries by GCPAC. MSTORE(,) contains the variable identifiers that need to be saved in the Internal MOS 2000 Storage System after day 1.

RESTRICTIONS:

The information in LSTORE(,), must, of course, be correct. Other routines, such as those U201 or U600, store the data and the attributes so that LMSTR1 will work properly.

COMMENTS:

Since it is not known beforehand just which variables will be needed, all possible needed ones are stored for day 1, and a record is kept of those used. Thereafter, only those needed are stored in the Internal MOS Storage System. (This is not quite true. When time spanning is done, variables used in the time spanning from a model run time (cycle) other than the one being processed may be saved.)

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: With U201

LMSTR2

PREPARES LSTORE FOR GCPAC

Harry R. Glahn
June 1, 1995

<u>PURPOSE</u>: To prepare LSTORE(,) for elimination of unneeded entries by GCPAC.

<u>RESTRICTIONS</u>:

The information in LSTORE(,), must, of course, be correct. Other routines, such as those of U201 or U600, store the data and the attributes so that LMSTR2 will work properly.

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: With U201

GCPAC

COMPRESSES DATA IN INTERNAL MOS-2000 STORAGE SYSTEM

Harry R. Glahn
April 1, 1995

PURPOSE: To eliminate the data in the internal MOS-2000 storage structure that have been marked as not being needed again and to compress the remainder. All accounting data are taken care of so that the data can be retrieved by GFETCH and new data stored by GSTORE. A certain amount of core is provided for storage and when that is filled, the data are stored on disk. The user need not worry about the internals of this system. GCPAC is used by U201, but may have uses in other programs.

RESTRICTIONS:

None other than the general MOS-2000 guidelines, etc.

NONSYSTEM ROUTINES USED:

RDDISK, WRDISK

LANGUAGE: FORTRAN 77

LOCATION: With U201

RDSTRX

STORES ALL DATA FOR DAY ONE

Harry R. Glahn
July 25, 1996

PURPOSE: To read and store in the internal MOS-2000 file structure all data for a sequence of dates. It is used by U600, but could have a use in other programs. When U600 is initiated, it is not known what variables will be needed in the computations and interpolation. It can be determined the date/times that span the data needed for the first day (actually, the first date/time). All data on all input files for those date/times are read and stored packed in the MOS-2000 internal file system. This system provides a certain amount of core storage, and when it is exhausted, the rest of the information is written to disk.

RESTRICTIONS:

None other than the general MOS-2000 guidelines, etc..

NONSYSTEM ROUTINES USED:

UNPACK, GSTORE, UPDAT, GRCOMB

LANGUAGE: FORTRAN 77

LOCATION: moslib

WRTDLM

WRITES DATA INTO MOS-2000 EXTERNAL FILE SYSTEM

Harry R. Glahn
December 1, 1996

<u>PURPOSE</u>: To write a record of data to the external MOS-2000 file system. Data can be written with or without replacement. When writing without replacement, checking for duplicates can be done.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
CALL WRTDLM(KFILDO,KFILX,CFILX,ID,RECORD,NSIZE,NREPLA,NCHECK,L3264B,IER)
```

<u>KFILDO</u> - Unit number of output (print) file. (INPUT)

<u>KFILX</u> - Unit number for writing the data. (INPUT)

<u>CFILX</u> - Name of the file where the data are to be located. (INPUT)

<u>ID</u>(J) - A 4-word ID of the variable to write (J=1,4). This should be the MOS-2000 ID extended to include constant data. (INPUT)

<u>RECORD</u>(J) - The data to be written (J=1,NSIZE). (INPUT)

<u>NSIZE</u> - The number of values to write from RECORD( ). (INPUT)

<u>NREPLA</u> - Replacement flag.
0 = Not replacing record.
1 = Replacing record. If an existing record with the same IDs is not found, the record is not written and return is made with IER set = 156.
2 = Replacing record. If an existing record with the same IDs is not found, a new record is written.
(INPUT)

<u>NCHECK</u> - Identification checking flag.
0 = Not checking for duplicates.
1 = Checking for duplicates. If an existing record with the same IDs is found, the data are not written and return is made with IER set = 157.
(INPUT)

<u>L3264B</u> - Must have the value of 32 or 64 indicating the word length of the machine being used. (INPUT)

<u>IER</u> - Status return. These values may originate in called subroutines. Any value not listed below would indicate an error in the file system software.
0 = Good return.
156 = Replacing record, and no record found to replace (NREPLA = 1).
157 = Not replacing record, but match found (NCHECK = 1)
9xx= A system error number will be returned when a problem occurs on opening, reading, or writing a file. (OUTPUT)

EXAMPLE

```
    PARAMETER (ND5=1000)
C
    CHARACTER*60 CONST
C
    DIMENSION RECORD(ND5)
    DIMENSION ID(4)
C
    DATA KFILDO/6/,
   1     KFILX/30/
    DATA L3264B/32/
    DATA CFILX/' '/
    DATA CONST/'CONSTANTDATA'/
    DATA ID/001000,850,0,0/
    ...
    NSIZE=450
    CFILX(1:12)=CONST(1:12)
    ...
    CALL WRTDLM(KFILDO,KFILX,CFILX,ID,RECORD,NSIZE,0,0,L3264B,IER)
```

The default output file is Unit No. 6. The unit number for the random access file is 30, and the file name is 'CONSTANTDATA'. The 450 values of 850-mb heights to write are in RECORD( ). Neither replacement nor checking for duplicates is to be done. The program is running on a 32-bit machine. IER should be checked for zero for a good write.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

The MOS-2000 External File System uses a COMMON block ARGC. There are three parameters that are set in the five routines in which ARGC occurs. The only real limit this imposes on the user is that the PARAMETER NW is the largest possible number of keys in a key record of the file being read or written. A value for NW should be established and used for (nearly) all MOS-2000 programs. Note that this is not a limit on the number of keys in the file.

COMMENTS:

The MOS-2000 external file system is described elsewhere. Any error producing IER ≠ 0, will give a diagnostic on Unit No. KFILDO.

NONSYSTEM ROUTINES USED:

FLOPNM, WRTM

LANGUAGE: FORTRAN 77

LOCATION: moslib

WRTM

WRITES DATA FOR WRLM

Harry R. Glahn
December 1, 1996

PURPOSE: To write a record of data to the external MOS-2000 file system. It is called by WRTDLM, which is called by the user for writing data. It takes care of the necessary bookkeeping and calls RDKEYM, WRKEYM and WRDATA as necessary.

RESTRICTIONS:

WRTM is to be used only with the MOS-2000 external file system.

NONSYSTEM ROUTINES USED:

RDKEYM, WRKEYM, WRDATA

LANGUAGE: FORTRAN 77

LOCATION: moslib

WRKEYM

WRITES A KEY RECORD

Harry R. Glahn
December 1, 1996

PURPOSE: To write a key record to the external MOS-2000 file system. It is called by FLOPNM, RDTDLM, and WRTDLM, the latter two of which are called by the user for reading and writing data.

RESTRICTIONS:

WRKEYM is to be used only with the MOS-2000 external file system.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: moslib

WRDATM

WRITES A DATA RECORD

Harry R. Glahn
December 1, 1996

<u>PURPOSE</u>: To write a data record to the external MOS-2000 file system. It is called by WRTDLM, which is called by the user for writing data.

<u>RESTRICTIONS</u>:

WRDATM is to be used only with the MOS-2000 external file system.

<u>NONSYSTEM ROUTINES USED</u>:

None

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

RDTDLM

READS DATA FROM MOS-2000 EXTERNAL FILE SYSTEM

Harry R. Glahn
December 1, 1996

PURPOSE: To read a record of data from the external MOS-2000 file system.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

CALL RDTDLM(KFILDO,KFILX,CFILX,ID,RECORD,NSIZE,NVALUE,L3264B,IER)

KFILDO - Unit number of output (print) file. (INPUT)

KFILX - Unit number for reading the data. (INPUT)

CFILX - Name of the file where the data are located. (INPUT)

ID(J) - A 4-word ID of the variable to read (J=1,4). This should be the MOS-2000 ID extended to include constant data. As an option, when ID(1) = 9999, the next record in the file is read. This is useful when the calling program needs to know all the records in the file. On the first call, the first data record (not the key record, which is handled in the usual manner) is read. This would usually be the call letters (directory) record. When there is no next record, a return is made with IER ≠ 0. (INPUT)

RECORD(J) - The array into which to read the data (J=1,NSIZE). (OUTPUT)

NSIZE - The size of the array RECORD( ). (INPUT)

NVALUE - The number of values returned in RECORD( ). (OUTPUT)

L3264B - Must have the value of 32 or 64 indicating the word length of the machine being used. (INPUT)

IER - Status return. These values may originate in called subroutines. Any value not listed below would indicate an error in the file system software.

0 =	Good return.
153 =	No next record to read when reading sequentially.
154 =	NSIZE not large enough to hold the data record.
155 =	Can't find the IDs of the record to read.
9xx=	A system error number will be returned when a problem occurs on opening, reading, or writing a file. (OUTPUT)

EXAMPLE

```
    PARAMETER (ND5=1000)
C
    CHARACTER*60  CONST
C
    DIMENSION RECORD(ND5)
    DIMENSION ID(4)
C
    DATA KFILDO/6/,
   1     KFILX/30/
    DATA L3264B/32/
    DATA CFILX/' '/
    DATA CONST/'CONSTANTDATA'/
    DATA ID/001000,850,0,0/
    ...
    CFILX(1:12)=CONST(1:12)
    ...
    CALL WRTDLM(KFILDO,KFILX,CFILX,ID,RECORD,ND5,NVALUE,L3264B,IER)
```

The default output file is Unit No. 6. The unit number for the random access file is 30, and the file name is 'CONSTANTDATA'. The values in the record read are for 850-mb height and are returned in RECORD( ), the number of values being returned as NVALUE. The maximum number of values that could be read and returned is 1000. The program is running on a 32-bit machine. IER should be checked for zero for a good read.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

The MOS-2000 external file system uses a COMMON block ARGC. There are three parameters that are set in the five routines in which ARGC occurs. The only real limit this imposes on the user is that the PARAMETER NW is the largest possible number of keys in a key record of the file being read or written. A value for NW should be established and used for (nearly) all MOS-2000 programs. Note that this is not a limit on the number of keys in the file.

COMMENTS:

The MOS-2000 External File System is described elsewhere. Any error producing IER ≠ 0, will give a diagnostic on Unit No. KFILDO, and each element of RECORD( ) will contain 9999.

NONSYSTEM ROUTINES USED:

FLOPNM, RDTM

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDTM

READS DATA FOR RDTDLM

Harry R. Glahn
December 1, 1996

PURPOSE: To read a record of data from the external MOS-2000 file system. It is called by RDTDLM, which is called by the user for reading data. It takes care of the necessary bookkeeping and calls RDKEYM, WRKEYM and RDDATM as necessary.

RESTRICTIONS:

RDTM is to be used only with the MOS-2000 external file system.

NONSYSTEM ROUTINES USED:

RDKEYM, WRKEYM, RDDATM

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDKEYM

READS A KEY RECORD

Harry R. Glahn
December 1, 1996

PURPOSE: To read a key record from the external MOS-2000 file system. It is called by FLOPNM, RDTDLM, and WRTDLM, the latter two of which are called by the user for reading and writing data.

RESTRICTIONS:

RDKEYM is to be used only with the MOS-2000 external file system.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDDATM

READS A DATA RECORD FOR RDTM

Harry R. Glahn
December 1, 1996

PURPOSE: To read a data record from the external MOS-2000 file system. It is called by RDTDLM, which is called by the user for writing data.

RESTRICTIONS:

RDDATM is to be used only with the MOS-2000 external file system.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: moslib

FLOPNM

OPENS A MOS-2000 FILE

Harry R. Glahn
December 1, 1996

PURPOSE: To prepare for reading or writing a MOS-2000 external direct access file. This includes, when necessary, opening the file and reading the master key record and a key record. It also includes, when necessary, closing a file when the maximum number allowed to be open are open. Closing a file may require writing a key record and master key record. The file system uses a COMMON block ARGC. The first time FLOPNM is called, it initializes this COMMON block by calling ARINIT.

RESTRICTIONS:

FLOPNM is to be used only with the MOS-2000 external file system.

NONSYSTEM ROUTINES USED:

ARINIT, CLFM, RDKEYM

LANGUAGE: FORTRAN 77

LOCATION: moslib

CLFILM

CLOSES A FILE IN THE MOS-2000 EXTERNAL FILE SYSTEM

Harry R. Glahn
December 1, 1996

PURPOSE: To close a file in the external MOS-2000 file system.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL CLFILM(KFILDO,KFILX,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

KFILX - Unit number for the file to close. (INPUT)

IER - Status return.
  0 = Good return.
  9xx= A system error number will be returned when a problem occurs on writing the file. (OUTPUT)

EXAMPLE

```
C
    DATA KFILDO/6/,
   1     KFILX/30/
    ...
    CALL CLFILM(KFILDO,KFILX,IER)
```

The default output file is Unit No. 6. The unit number for the random access file to close is 30. IER should be checked for zero for a good read.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

CLFILM is to be used only with the MOS-2000 external file system.

COMMENTS:

The MOS-2000 external file system is described elsewhere. The key and master key records are written as necessary before closing the file. Any error producing IER ≠ 0, will give a diagnostic on Unit No. KFILDO.

NONSYSTEM ROUTINES USED:

CLFM

LANGUAGE: FORTRAN 77

LOCATION: moslib

CLFM

CLOSES A FILE FOR CLFILM

Harry R. Glahn
December 1, 1996

PURPOSE: To close a file in the external MOS-2000 file system. It is called by CLFILM, which is used by the file system and can be called by a user. It takes care of the necessary bookkeeping before closing the file, and writes the key and master key records if necessary.

RESTRICTIONS:

CLFM is to be used only with the MOS-2000 external file system.

NONSYSTEM ROUTINES USED:

WRKEYM, TDLPRM (\D Only)

LANGUAGE: FORTRAN 77

LOCATION: moslib

ARINIT

INITIALIZES COMMON BLOCK ARGC

Harry R. Glahn
December 1, 1996

PURPOSE: To initialize for FLOPNM the COMMON block ARGC.

RESTRICTIONS:

ARINIT is to be used only with the MOS-2000 external file system.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: moslib

TDLPRM

PRINTS CONTENTS OF COMMON BLOCK ARGC IN THE MOS-2000 FILE SYSTEM

Harry R. Glahn
December 1, 1996

PURPOSE: To provide a printout of the contents of the COMMON block ARGC. It can be called by any of the routines in the file system software. The call is actually in the code in various places, either commented out or as a /D option that can be easily activated. TDLPRM can be used occasionally to monitor the operation of the file system.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL TDLPRM(KFILDO,FROM)
```

KFILDO	- Unit number of output (print) file. (INPUT)
FROM	- Up to 8 characters indicating from where TDLPRM was called. This becomes part of the print. (CHARACTER*8) (INPUT)

EXAMPLE

```
DATA KFILDO/6/
...
CALL TDLPRM(KFILDO,'FLOPNM')
```

The default output file is Unit No. 6. The routine calling TDLPRM is FLOPNM.

OUTPUT:

Output will be written to Unit No. KFILDO.

RESTRICTIONS:

TDLPRM is to be used only with the MOS-2000 external file system.

COMMENTS:

Considerable print will occur if many records of data are being dealt with.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: moslib

DOY

COMPUTES THE DAY OF THE YEAR

Harry R. Glahn
December 1, 1996

PURPOSE: To compute the day of the year as well as the year, month, day, and hour from a date/time in YYYYMMDDHH format.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL DOY(NDATE,JYR,JMO,JDA,JHR,MDOY)
```

NDATE	- The date/time for which the day of the year (DOY) is needed in YYYYMMDDHH format. (INPUT)
JYR	- The year (4 digits) corresponding to NDATE. (OUTPUT)
JMO	- The number of the month corresponding to NDATE. (OUTPUT)
JDA	- The day of the month corresponding to NDATE. (OUTPUT)
JHR	- The hour (2 digits) corresponding to NDATE. (OUTPUT)
MDOY	- The day of the year from 1 through 365 or 366 in case of leap year. Note that the year 2000 is not a leap year. (OUTPUT)

EXAMPLE:

```
DATA NDATE/1995012200/
...
CALL DOY(NDATE,JYR,JMO,JDA,JHR,MDOY)
```

The DOY corresponding to January 22, 1995 is returned in MDOY. Also, the year (1995), month (1), day (22), and hour (0) are returned in JHR, JMO, JDA, JHR, respectively.

OUTPUT:

None.

RESTRICTIONS:

None. (Does not handle non-leap years of 1900, etc.)

NONSYSTEM ROUTINES CALLED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDDIR

READS DIRECTORY RECORD

Harry R. Glahn
December 1, 1996

PURPOSE: To read the directory (or call letters) record from a vector-type TDLPACK file and to initialize INDEXC( ) with the locations of the stations in the directory of the stations in CCALL(,).

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL RDDIR(KFILDO,KFILIN,IP12,NAMIN,NDATE,CCALL,INDEXC,ND1,NSTA,
   1           CCALLD,ND5,MSTA,L3264B,L3264W,IER)
```

KFILDO - Unit number of output (print) file. Diagnostics, including error conditions, are written here. (INPUT)

KFILIN - Unit number from which to read the directory record. (INPUT)

IP12 - Indicates whether (>0) or not (=0) the list of stations on the input file will be printed to the file whose unit number is IP12. (INPUT)

NAMIN - Data set name corresponding to the unit number KFILIN. The name is limited to 60 characters. (CHARACTER*60) (INPUT)

NDATE - Date being processed. Not actually used. (INPUT)

CCALL(K,J) - The 8-character call letters of the stations for which data are wanted (J=1) and up to 5 substitute stations (J=2,6) (K=1,NSTA). (INPUT)

INDEXC(K) - The location of each station in CCALL(K,1) in the directory record is put into INDEXC(K). If a station in CCALL(,1) cannot be found, then the other columns in CCALL(,J) are searched in the order J=2,6 until either the substitute station is found in the directory or a blank is found in CCALL(,J), at which point the search is discontinued. Once the primary call letters (J=1) or substitute call letters are found, INDEXC( ) is filled with the corresponding value. If no station in CCALL(K,J) (J=1,6) is found in the directory, INDEXC(K) is set = 9999. (OUTPUT)

ND1 - The dimension of INDEXC( ) and first dimension of CCALL(,). (INPUT)

NSTA - Number of stations with call letters in CCALL(,). (INPUT)

CCALLD(K) - Array into which to read the directory record. (INTERNAL)

ND5 - Dimension of CCALLD( ). (INPUT)

<u>MSTA</u> - Number of stations whose call letters are in the input directory record. (OUTPUT)

<u>L3264B</u> - Either 32 or 64 indicating the word length of the machine being used. (INPUT)

<u>L3264W</u> - The number of machine words in 64 bits. Not actually used. (INPUT)

<u>IER</u> - Status return.

```
  0 = Good return.
120 = One or more stations in CCALL( , ) can't be found in the
      directory record as read into CCALLD( ).
140 = Error reading directory record.
145 = Size of directory record exceeds array size CCALLD(ND5).
146 = End of file found.
(OUTPUT)
```

<u>EXAMPLE</u>

```
    PARAMETER (ND1=200)
    PARAMETER (ND5=500)

    CHARACTER*8 (CCALL(ND1,6))
    CHARACTER*8 (CCALLD(ND5))
    CHARACTER*60 NAMIN/'U201OUTHRG'/

    DIMENSION INDEXC(ND1)

    DATA IP12/12/
    DATA KFILIN/5/
    DATA KFILDO/12/
    DATA NSTA/100/
    DATA L3264B/32/,
   1     L3264W/2/
    ...
    [fill CCALL( , )]
    ...

    CALL RDDIR(KFILDO,KFILIN,IP12,NAMIN,CCALL,INDEXC,ND1,NSTA,
   1           CCALLD,ND5,MSTA,L3264B,L3264W,IER)
```

The unit number for diagnostics is 12. The unit number to read the data from is 5 and the corresponding file name is 'U201OUTHRG'; the name is used only for a possible diagnostic. The stations on the file read will be printed to Unit No. 12. The number of call letters in CCALL(,) is 100, the maximum being 200. The call letters record read can be as long as 500; if it exceeds that, a diagnostic is produced and an error return IER = 145 is made; however, the array CCALLD( ) will not be overflowed. The run is being made on a 32-bit machine, there being 2 "words" in 64 bits.

<u>OUTPUT</u>:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None other than already mentioned.

NONSYSTEM ROUTINES USED:

FINDST

LANGUAGE: FORTRAN 77

LOCATION: moslib

TRAIL

WRITES TRAILER RECORD

Harry R. Glahn
March 1, 1997

PURPOSE: To write a trailer record that is to precede a directory (call letters) record on a vector TDLPACK file. It is not expected that this routine be called by a user directly.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL TRAIL(KFILDO,KFILIO,L3264B,L3264W,NTOTBY,NTOTRC,IER)
```

KFILDO	- Unit number of output (print) file. (INPUT)
KFILIO	- Unit number of file to read and skip records. (INPUT)
L3264B	- Integer word length in bits of machine being used, either 32 or 64. (INPUT)
L3264W	- Number of INTEGER machine words in 64 bits on the machine being used, either 1 or 2. (INPUT)
NTOTBY	- The total number of bytes on the file. It is increased in TRAIL by 24. (INPUT/OUTPUT)
NTOTRC	- The total number of records on the file. It is increased in TRAIL by 1. (INPUT/OUTPUT)
IER	- Status return. 0 = Good return. Other values can be generated by called subroutine PKBG, or can be set to IOS because of a writing error. (OUTPUT)

EXAMPLE

```
    PARAMETER (L3264B=32)
    PARAMETER (L3264W=64/L3264B)
C
    DATA KFILDO/6/
    DATA KFILIO/20/
    DATA NTOTBY/0/
    DATA NTOTRC/0/
C
    CALL TRAIL(KFILDO,KFILIO,L3264B,L3264W,NTOTBY,NTOTRC,IER)
```

The default output unit number is 6. The unit number for the output file is 20. NTOTBY will be increased by 24 and NTOTRC by 1.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

None.

COMMENTS:

It is not expected that this routine be called by a user directly. A trailer record is to be used only directly prior to a directory record on a vector TDLPACK file. When such a file is to be written, the routine SKIPWR should be used, which will call TRAIL as necessary.

NONSYSTEM ROUTINES USED:

PKBG

LANGUAGE: FORTRAN 77

LOCATION: moslib

CONSTX

RETRIEVES DATA FROM MOS-2000 EXTERNAL RANDOM ACCESS FILE FOR VECTOR-ORIENTED PROGRAMS

Harry R. Glahn
December 1, 1996

PURPOSE: To retrieve data from a MOS-2000 external random access file for vector-oriented programs. CONSTX is called from OPTX when IDPARS(1) $\geq$ 400 and $\leq$ 499; that is, CCC values between 400 and 499 inclusive are reserved for data in these random access files.

RESTRICTIONS:

The call sequence is appropriate for U600 and other "vector" programs. All the facilities of the MOS-2000 internal storage system and external random access file system are used. The file unit number will be 95, and the file name is in KFILRA( ).

COMMENTS:

On the first use of a particular random access file, the "directory" record is read with ID(1) = 400001000, ID(2) = 0, ID(3) = 0, and ID(4) = 0. A correspondence table between this directory and the stations for which data are needed is formed and stored in the MOS-2000 internal storage system with IDs the same as above, except ID(2) = KFILX. On subsequent entries when the same file is needed, this correspondence table is used.

The subroutine COMPID is used to compute the IDs needed to fetch the data. For instance, the ID in the call sequence will be generic in the sense that the date/time of the constant on the file is not specified. When constants are furnished for each day (or month or season) the full ID must be computed. The structure shown in Chapter 4, Variable Identification, is followed. The tau in the furnished constant ID is added to NDATE (the date being processed) to determine the date for which the constants are desired.

NONSYSTEM ROUTINES USED:

GFETCH, GSTORE, RDTDLM, COMPID, UNPACK

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDVRBL

READS AND ORDERS VARIABLE LIST

Harry R. Glahn
August 25, 1996

<u>PURPOSE</u>: To read a combined predictor and predictand list and associated information from a file on Unit No. KFILP. KFILP can be the default input file or can be a separate file. Also, variable names and other information from the variable constant file are retrieved from the file read on Unit No. KFILCP and matched with the variables; some insertions are made for computed predictors. Each predictor ID in ID(,) is duplicated in JD(,) except that some portions are omitted; the result is called the "basic" ID. (This basic list is not defined the same way as in U201.) The variable list is then reordered in such a way that the predictors, sans the binary variable indicator "B," are in order low to high, including thresholds, then all the predictands, low to high. All binaries of the same variable are together. From the single threshold read, an upper and lower are set by subroutine THSET. The model number "DD" is set to zero for variables in the range CCC = 400-499.

<u>RESTRICTIONS</u>:

It is assumed the files on unit numbers KFILP and KFILCP have been opened.

<u>NONSYSTEM ROUTINES USED</u>:

PRSID, SORT16, FSORT, XCHANG, CKIDS

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

OPTX

SWITCHES TO COMPUTATIONAL SUBROUTINES FOR VECTOR-ORIENTED PROGRAMS

Harry R. Glahn
July 25, 1996

PURPOSE: To switch to a computational subroutine when it is necessary to process other than a basic or binary variable for vector-oriented programs such as U600. It is expected that this routine will not be needed much; most computations should be done in U201. In U600, OPTX is entered through VRBL61 AND VRBL62.

RESTRICTIONS:

None other than the general MOS-2000 guidelines, etc. Since this routine is in moslib, any routine called by it must be in moslib, if the standard load line is used.

COMMENTS:

The data needed for the computation are accessed through the internal MOS-2000 storage system. Work arrays are available for provision to the computation routine.

NONSYSTEM ROUTINES USED:

CONST	-	Fetches "constant" data from a MOS-2000 random access file.
TIMEPV	-	Calculates time difference.
VERTX	-	Vertical processing.
DIFFV	-	Difference of two variables.
DIRFUV	-	Wind direction from U and V.
MPSKTS	-	Converts kts to m/s (for wind speed).
ZRONEG	-	Sets negative values to zero (for wind speed).
POPDIF	-	Calculates absolute value of difference of two pop forecasts.
FORIER	-	First or second harmonic of day of year.

LANGUAGE: FORTRAN 77

LOCATION: moslib

PACKYY

DETERMINES WHETHER TO PACK SPATIAL DIFFERENCES WITH PRIMARY MISSING VALUES

Harry R. Glahn
June 1, 1997

PURPOSE: To be used by PACK2D when there are primary missing values to estimate whether or not it will be beneficial to pack spatial differences. Spatial differences are not used when a secondary missing value is indicated to PACK2D. Original values are indicated when the average range of consecutive groups of size MINPK of the second order differences is larger than the average range of consecutive groups of size MINPK of the original values. This routine gives the correct answer when the choice is clear. When it isn't, it doesn't matter much which choice is made.

RESTRICTIONS:

Will not accommodate secondary missing values.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

PKMS97

ASSISTS PACKING WHEN THERE CAN BE SECONDARY MISSING VALUES

Harry R. Glahn
June 1, 1997

PURPOSE: To be used by PKMS99 (called by PACK) to determine, through PACKGP, the groups to use in packing in TDLPACK format, their minima, etc. when there can be secondary missing values, and to remove the overall minimum and local minimum from the each of the groups.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

PACKGP

LANGUAGE: FORTRAN 77

LOCATION: moslib

PKS4LX

ASSISTS SUBROUTINE PACK IN SIMPLE PACKING

Harry R. Glahn
June 1, 1997

PURPOSE: To be used by PACK in packing a string of values that are not divided into groups. PKS4LX was created for efficiency by putting the bit stuffing instructions in-line, rather than calling PKBG within the loop.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

PKC4LX

ASSISTS SUBROUTINE PACK IN COMPLEX PACKING

Harry R. Glahn
June 1, 1997

PURPOSE: To be used by PACK in packing a one or more strings of values. PKC4LX was created for efficiency by putting the bit stuffing instructions in-line, rather than calling PKBG within the loop.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

UNPKOO

ASSISTS UNPACKING WHEN THERE ARE NO MISSING VALUES

Harry R. Glahn
June 1, 1997

PURPOSE: To be used by UNPACK when there are no missing values present. Scaling is not used, but the reference value is. UNPKOO was created for efficiency by putting the bit unstuffing instructions in-line, rather than calling UNPKBG within the loop.

RESTRICTIONS:

Not to be used when missing values are present.

NONSYSTEM ROUTINES USED:

UNPACK.

LANGUAGE: FORTRAN 77

LOCATION: moslib

UNPKPO

ASSISTS UNPACKING WHEN THERE ARE PRIMARY MISSING VALUES

Harry R. Glahn
June 1, 1997

PURPOSE: To be used by UNPACK when there are primary missing values present. Scaling is not used, but the reference value is. UNPKPO was created for efficiency by putting the bit unstuffing instructions in-line, rather than calling UNPKBG within the loop.

RESTRICTIONS:

Not to be used when secondary missing values are present.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

UNPKPS

ASSISTS UNPACKING WHEN THERE ARE PRIMARY AND SECONDARY MISSING VALUES

Harry R. Glahn
June 1, 1997

PURPOSE: To be used by UNPACK when there are primary and secondary missing values present. Scaling is not used, but the reference value is. UNPKPO was created for efficiency by putting the bit unstuffing instructions in-line, rather than calling UNPKBG within the loop.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

UNPKLX

ASSISTS IN UNPACKING OF GROUP INFORMATION

Harry R. Glahn
June 1, 1997

PURPOSE: To be used by UNPACK to unpack the group minima, sizes, etc. Scaling is not used and neither is a reference value. UNPKLX was created for efficiency by putting the bit unstuffing instructions in-line, rather than calling UNPKBG within the loop.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

FSORT

COMPLETES SORTING VALUES IN AN ARRAY

Harry R. Glahn
J. Paul Dallavalle
July 15, 1997

<u>PURPOSE</u>: To complete sorting MOS-2000 variable ID's previously processed by the routine SORT16. SORT16 does not necessarily end with the thresholds associated with the variables in order low to high. FSORT does that when more than one threshold exists for a variable with otherwise the same ID. An associated variable is provided and is returned updated with the values indicating where the variables were relocated to. This variable can then be used by subroutine XCHANG to exchange another variable so that its location will match the reordered values. FSORT is used in RDPRED in U201 and RDVRBL in U600, and may have other uses.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS</u>:

```
CALL FSORT(KFILDO,JD,INDEX,THRESH,NVRBL)
```

<u>KFILDO</u> - Unit number of output (print) file. Normally not used, but is available in case diagnostics need be inserted. (INPUT)

<u>JD</u>(I,J) - List to sort (I=1,4) (J=1,NVRBL), according to THRESH( ). (INPUT-OUTPUT)

<u>INDEX</u>(J) - Array that upon entry indicates the location from where the ID came before sorting by SORT16. It is updated on return and indicates for each input value JD(,J) where it came from in the original list in JD(,) (J=1,NVRBL). (INPUT-OUTPUT)

<u>THRESH</u>(J) - The thresholds associated with JD(,J) (J=1,NVRBL). These thresholds are what the sorting in FSORT is done on. Only when two variables in JD(,) are the same is any rearrangement necessary. Upon entry, the thresholds associated with an ID may not be in order, as SORT16 did not assure that. (INPUT-OUTPUT)

<u>NVRBL</u> - Number of values in JD(,), THRESH( ), and INDEX( ). (INPUT)

<u>EXAMPLE</u>:

```
DIMENSION ID(4,NVRBL),THRESH(NVRBL),INDEX(NVRBL)
...
CALL FSORT(KFILDO,JD,INDEX,THRESH,NVRBL)
```

The basic variable JD(,) list (which in U201 and U600 is the full MOS-2000 variable ID sans certain processing information and has already been processed by SORT16) is furnished for the completion of sorting. It is dimensioned so that when it is received in FSORT as REAL*16 on the HP workstation, sorting can be done as a vector.

OUTPUT:

None.

RESTRICTIONS:

This routine is different on the HP 32-bit workstation than on the 64-bit CRAY to the extent that the variable in FSORT receiving JD(,), is REAL*16 on the HP workstation which accommodates the 128 bits, but since the word length on the CRAY is 64 bits, there are actually 256 bits passed. This is dealt with by having two DOUBLE PRECISION REAL words per variable in FSORT on the CRAY.

NONSYSTEM ROUTINES CALLED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

RDSTAL

READS A STATION LIST AND DIRECTORY

Harry R. Glahn
August 1, 1997

<u>PURPOSE</u>: To read a list of station call letters and associated information from a station directory. The list, composed of up to 8 characters per station can (1) be taken from the directory, in which case all stations in the directory will be in the list or (2) be on a separate file, in which case the list is ended with the terminator '99999999'. The stations in the list returned <u>will be in the order read</u>. Any duplicate stations found in the separate list will be kept, and a diagnostic provided.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
CALL RDSTAL(KFILDO,IP4,IP5,KFILD,NEW,CCALL,NAME,NELEV,IWBAN,
            STALAT,STALON,ITIMEZ,IFOUND,ND1,NSTA,IER)
```

<u>KFILDO</u> - Unit number of output (print) file. (INPUT)

<u>IP4</u> - When IP4 > 0, the station list (call letters only) will be written to Unit No. IP4. If there are input errors, the station list will be written to the default output file Unit No. KFILDO as well as to IP4. (INPUT)

<u>IP5</u> - When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5. (INPUT)

<u>KFILD</u>(J) - Unit number from which to read the station list (J=1) and station directory (J=2). It is assumed the files have been opened. If all stations are to be used from the directory, KFILD(1) = KFILD(2). (INPUT)

<u>NEW</u> - 1 when new ICAO station identifiers (call letters) are being used (columns 1-8 in the station directory);
0 when the old station call letters are being used (columns 10-17 in the station directory. (INPUT)

<u>CCALL</u>(K,J)- Array in which up to 6 sets (J=1,6) of 8 call letters per station are returned (K=1,NSTA). CCALL(,1) will contain the call letters to be used as station or location identifiers in the calling program; the other 5 sets (J=2,6) are taken from the directory. When KFILD(1) ≠ KFILD(2), the station list returned in CCALL(,1) will be from the file on Unit No. KFILD(1), and <u>will be in the order read</u>; when KFILD(1) = KFILD(2), the station list returned in CCALL(,1) will be from the station directory and will be in the order found on the directory. The call letters returned in CCALL(,1) will be

the ICAO station identifiers (columns 1-8) when NEW = 1, and will be the old call letters (columns 10-17) when NEW = 0. The other lists in CCALL(,J) always come from the directory, and for J=3,6 come from columns 83-90, 92-99, 101-108, and 110-117 in that order. The list in CCALL(,2) also comes from the directory, being the ICAO call letters (columns 1-8) when NEW = 0 and the old call letters (columns 10-17) when NEW = 1. That is, it is intended that when CCALL(,1) contains the ICAO station identifiers, CCALL(,2) will contain the old call letters, and vice versa. (CHARACTER*8) (OUTPUT)

NAME(K) - Names as read from the directory ordered according to CCALL(K,1) (K=1,NSTA). (CHARACTER*20) (OUTPUT)

NELEV(K) - Elevations of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

STALAT(K) - Latitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

STALON(K) - Longitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

ITMEZ(K) - Time zone of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). This is the number of hours the station is different from (earlier than) UTC. (OUTPUT)

IWBAN(K) - WBAN numbers of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

IFOUND(K) - Used to keep track of the stations found in the directory. (K=1,ND1). (INTERNAL)

ND1 - First dimension of CCALL(,), and dimension of NELEV( ), IWBAN( ), STALAT( ), STALON( ), and IFOUND( ). (INPUT)

NSTA - The number of elements (stations) returned in CCALL(,) and the associated arrays. (OUTPUT)

IER - Status return.
0 = Good return.
20 = Error or EOF reading KFILD(1) by RDC. Aborts.
21 = Too many stations for dimension ND1 when reading by RDC. Aborts.
33 = Error reading station directory on Unit No. KFILD(2). Aborts.
34 = List to be kept too long for dimension ND1. Aborts with print.
35 = One or more stations not found in the directory. Normal return.
36 = One or more stations are duplicates. Normal return.
37 = Both IER = 35 and 36 above have occurred. Normal return.
(OUTPUT)

EXAMPLE

```
    PARAMETER (ND1=500)
    CHARACTER*8 CCALL(ND1,6)
    CHARACTER*20 NAME(ND1)
    DIMENSION NELEV(ND1),IWBAN(ND1),STALAT(ND1),STALON(ND1),IFOUND(ND1)
    DIMENSION KFILD(2),IP(25)
    DATA IP/25*0/
    DATA KFILDO/12/
    DATA KFILD/28,29/,
   1     NEW/1/
    ...

    CALL RDSTAL(KFILDO,IP(4),IP(5),KFILD,NEW,CCALL,NAME,NELEV,
   1            IWBAN,STALAT,STALON,ITIMEZ,IFOUND,ND1,NSTA,IER)
```

The default output unit number is 12. The unit number to read the station list from is 28, and the station directory is on Unit No. 29. Since IP(4) and IP(5) = 0, the station list will not be printed, unless there are fatal errors, in which case the list and the error found will be on Unit No. 12. The NSTA ICAO identifiers are returned in CCALL(,1), along with the associated information in other columns of CCALL(,) and in the arrays as indicated above. Since the station list will be the ICAO station identifiers (NEW = 1), CCALL(,2) will contain the old call letters. The maximum number of stations to be returned is ND1. The station list is read by subroutine RDC with FORMAT(7(A8,1X)) until the terminator '99999999' is found. Normally, each record would contain only one station just for ease of user manipulation, but up to 7 stations could be in one record. A description of the station dictionary can be found elsewhere.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. When IP4 > 0, the station list (call letters only) will be written to Unit No. IP4. If there are station list input errors, the station list will be written to the default output file Unit No. KFILDO as well as to IP4. When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are directory input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5.

RESTRICTIONS:

None other than stated above.

COMMENTS:

This routine performs essentially the same function as RDSTAD, except that the stations are not alphabetized in RDSTAL and they are in RDSTAD, provided the station directory is alphabetized. If a station cannot be found in the directory, the last 5 columns of CCALL(,) will be blank. Note that all six columns of the directory are searched. That is, suppose a record in the directory contains 'KDEN   ', 'DEN   ', and 'DVX   ' in columns 1-8, 10-17, and 83-90, respectively. Either 'KDEN', 'DEN', or

'DVX' could be read by subroutine RDC, and KDEN would be returned in CCALL(,1). If KDEN changed to, say, 'KDUM' and the directory is updated, then 'KDUM' would be returned in CCALL(,1) and would be used in the calling program to identify output. This means that the directory record might be different on different data sets, but the MOS-2000 software should handle it, provided 'KDEN' is added to the entry in columns 92-99.

This routine is used when only one "group" of stations is being dealt with (e.g., U201). For multiple groups, RDSTGN can be used.

When KFILD(1) ≠ KFILD(2), the input station list can be the default input file for the calling (main) program, or be a separate file.

It would be unusual to use the feature of KFILD(1) = KFILD(2) to return the complete station directory, unless one were to prepare a special (smaller) directory for some special purpose. However, if one does need the complete directory, RDSTAL will return it. Since there is no reordering of this directory, RDSTAL and RDSTAD operate the same way for this feature.

There are six routines which read a station list and return a list taken from either the first or second column of the directory, depending on the value of NEW; all six operate in the same way in this regard. Two routines deal with only a single list of stations and return the time zone. Four routines deal with multiple lists (groups) of stations. Three alphabetize the station list (by group) and three return the list in the order read. A group name is returned by two routines; this name is read immediately following the group list of stations with an A30 format. Their different characteristics are:

Routine	Single Station List	Multiple Station List	Group Name	Time Zone	Alphabetize
RDSTAL	X			X	
RDSTAD	X			X	X
RDSTGN		X			
RDSTGA		X			X
RDSTNL		X	X		
RDSTNA		X	X		X

RDSTAL is used in U201, U700, U900, U351, and U150.

Many changes were made in October 2000 to assure that the station wanted from the directory was returned. Any directory entry can have up to 6 station identifiers, the leftmost being called the ICAO identifier, and next column being called the SAO call letters, although that nomenclature does not have to be followed. The rule followed is that the leftmost identifier that matches the identifier in the input list is the one used. That is, if a link I0V1 was in a particular record in a column 2-6 and was also in the first column, then the station in the first column is the one returned. The older version of RDSTAL did not assure this, and the result depended on the order in which the records were encountered.

Diagnostics were also improved, so that not only duplicate and missing stations are identified, but duplicates that may be caused by using a link for one of the stations and when a station is kept because of a link and the column (2 through 6) where the identifier occurred. In this context, SAO call letters are in Col. 2 when ICAO identifiers are being used (NEW = 1) and ICAO identifiers are in Col. 2 when SAO call letters are being used (NEW = 0).

In addition, station directory information is not printed to KFILDO except when IP5 = KFILDO or the error return is fatal. The table below shows the print structure as a function of the values of IER and IP5.

		IER =	
	0	33,34	35,36,37
IP5 = 0	---	KFILDO	---
IP5 = KFILDO	KFILDO	KFILDO	KFILDO
IP5 = IP5	IP5	KFILDO IP5	IP5

<u>NONSYSTEM ROUTINES USED</u>:

RDC

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: MOS-2000 Library

FINDST

INITIALIZES ARRAY WITH STATION LOCATIONS

Harry R. Glahn
August 1, 1997

PURPOSE: To initialize INDEXC( ) with the locations of the stations in a directory record of the stations in CCALL(,). The location of each station in CCALL(K,1) in the directory record is put into INDEXC(K). If a station in CCALL(,1) cannot be found, then the other columns in CCALL(,J) are searched in the order J=2,6 until either the substitute station is found in the directory or a blank is found in CCALL(,J), at which point the search is discontinued. Once the primary call letters (J=1) or substitute call letters are found, INDEXC( ) is filled with the corresponding value. If no station in CCALL(K,J) (J=1,6) is found in the directory, INDEXC(K) is set = 99999999. Called from RDDIR. (OUTPUT)

The order of search was modified in October 2000 to better accommodate the links in the station dictionary that are put into CCALL(,J) (J=2,6) by routines like RDSTAL. All entries in CCALL(,1) are searched for in the directory record before any other columns are searched. Before, the entries in CCALL(K,J) (J=1,6) were searched before another value of K was used. This can cause problems when the same call letters appear more than once in the station dictionary. The order of precedence now is that the leftmost occurrence of a particular call letter combination will be used. That is, the combination in J=2 will be used before it will be used in J=3, should such a thing happen. Note that if the same call letter combination would appear in the same column for two or more different stations (e.g., J=3), results will be unpredictable, depending on the order of the stations in the list (the first one in the list CCALL(K,) will be used).

Sometimes the user may want to know when stations in CCALL(,) are not found in the directory record. However, this can cause voluminous print on some jobs. To try to accommodate this, for a normal compile, when one or more stations are not found in the station directory, the list is written to the default output file and IER = 120 on return. When compiled with the /D option on the HPs, this print will not occur, but rather the statement "STATIONS MISSING IN THE DIRECTORIES WILL NOT BE PRINTED OR AN ERROR INDICATED." will appear once on the default output file, and IER = 0 is returned.

RESTRICTIONS:

None other than already mentioned.

NONSYSTEM ROUTINES USED:

NONE

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDSTGN

READS ONE OR MORE STATION LISTS AND DIRECTORY INFORMATION

Harry R. Glahn
August 1, 1997

PURPOSE: To read one or more lists of station call letters and associated information from a station directory. The lists read are composed of up to 8 characters per station. Each list is ended by the terminator '99999999' and reading is ended with an empty set. The stations in the lists returned will be in the order read. Any duplicate stations found will be kept, and a diagnostic provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL RDSTGN(KFILDO,IP4,IP5,KFILD,NEW,CCALL,NAME,NELEV,IWBAN,
                STALAT,STALON,IFOUND,NGP,ND1,KGP,NSTA,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

IP4 - When IP4 > 0, the station list (call letters only) will be written to Unit No. IP4. If there are input errors, the station list will be written to the default output file Unit No. KFILDO as well as to IP4. (INPUT)

IP5 - When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5. (INPUT)

KFILD(J) - Unit number from which to read the station list (J=1) and station directory (J=2). It is assumed the files have been opened. KFILD(1) should not equal KFILD(2). (INPUT)

NEW - 1 when new ICAO station identifiers (call letters) are being used (columns 1-8 in the station directory);
0 when the old station call letters are being used (columns 10-13 in the station directory. (INPUT)

CCALL(K,J)- Array in which up to 6 sets (J=1,6) of 8 call letters per station are returned (K=1,NSTA). CCALL(,1) will contain the call letters to be used as station or location identifiers in the calling program; the other 5 sets (J=2,6) are taken from the directory. The station list returned in CCALL(,1) will be in the order read. The call letters returned in CCALL(,1) will be the ICAO station identifiers (columns 1-8) when NEW = 1, and will be the old call letters (columns 10-17) when NEW = 0. The other lists in CCALL(,J) always come from the directory, and for J=3,6 come from columns 83-90, 92-99, 101-108, and 110-117 in that order. The list in CCALL(,2) also comes from the directory, being the ICAO call letters (columns 1-8) when NEW = 0 and the old call letters

```
               (columns 10-17) when NEW = 1.  That is, it is intended that
               when CCALL( ,1) contains the ICAO station identifiers,
               CCALL( ,2) will contain the old call letters, and vice versa.
               (CHARACTER*8)  (OUTPUT)

NAME(K)      - Names as read from the directory ordered according to
               CCALL(K,1) (K=1,NSTA).  (CHARACTER*20)  (OUTPUT)

NELEV(K)     - Elevations of stations from the directory ordered according to
               CCALL(K,1) (K=1,NSTA).  (OUTPUT)

STALAT(K)    - Latitudes of stations from the directory ordered according to
               CCALL(K,1) (K=1,NSTA).  (OUTPUT)

STALON(K)    - Longitudes of stations from the directory ordered according to
               CCALL(K,1) (K=1,NSTA).  (OUTPUT)

IWBAN(K)     - WBAN numbers of stations from the directory ordered according
               to CCALL(K,1) (K=1,NSTA).  (OUTPUT)

IFOUND(K)    - Used to keep track of the stations found in the directory.
               (K=1,ND1).  (INTERNAL)

NGP(K)       - The sizes of KGP groups of stations (K=1,KGP), max of ND1.
               (OUTPUT)

ND1          - First dimension of CCALL( , ), and dimension of NAME( ),
               NELEV( ), IWBAN( ), STALAT( ), STALON( ), IFOUND( ), and
               NGP( ).  (INPUT)

KGP          - The number of groups of stations read, and the number of
               values returned in NGP( ), max of ND1.  (OUTPUT)

NSTA         - The total number of elements (stations) returned in CCALL( , )
               and the associated arrays.  (OUTPUT)

IER          - Status return.
                0 = Good return.
               20 = Error or EOF reading KFILD(1) by RDC.  Aborts.
               21 = Too many stations for dimension ND1 when reading by RDC.
                    Aborts.
               33 = Error reading station directory on Unit No. KFILD(2).
                    Aborts.
               34 = List to be kept too long for dimension ND1.  Aborts with
                    print.
               35 = One or more stations not found in the directory.  Normal
                    return.
               36 = One or more stations are duplicates.  Normal return.
               37 = Both IER = 35 and 36 above have occurred.  Normal return.
               (OUTPUT)
```

EXAMPLE

```
    PARAMETER (ND1=500)
    CHARACTER*8 CCALL(ND1,6)
    CHARACTER*20 NAME(ND1)
    DIMENSION NELEV(ND1),IWBAN(ND1),STALAT(ND1),STALON(ND1),IFOUND(ND1),
   1          NGP(ND1)
    DIMENSION KFILD(2),IP(25)
    DATA IP/25*0/
    DATA KFILDO/12/
    DATA KFILD/28,29/,
   1     NEW/1/
    ...

    CALL RDSTGN(KFILDO,IP(4),IP(5),KFILD,NEW,CCALL,NAME,NELEV,
   1            IWBAN,STALAT,STALON,IFOUND,NGP,ND1,KGP,NSTA,IER)
```

The default output unit number is 12. The unit number to read the station list from is 28, and the station directory is on Unit No. 29. Since IP(4) and IP(5) = 0, the station lists will not be printed, unless there are fatal errors, in which case the lists and the errors found will be on Unit No. 12. The NSTA ICAO identifiers are returned in CCALL(,1), along with the associated information in other columns of CCALL(,) and in the arrays as indicated above. The first NGP(1) values will be for group 1, the second NGP(2) values will be for group 2, etc. for all KGP groups. Since the station lists will be the ICAO station identifiers (NEW = 1), CCALL(,2) will contain the old call letters. The maximum number of stations to be returned is ND1. The station lists are read by subroutine RDC with FORMAT(7(A8,1X)) until the terminator '99999999' is found. An empty set will terminate the reading. Note that the end of all groups must have two terminators, one for the last non-empty group and one for the empty group. Normally, each record would contain only one station just for ease of user manipulation, but up to 7 stations could be in one record. A description of the station dictionary can be found elsewhere.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. When IP4 > 0, the station lists (call letters only) will be written to Unit No. IP4. If there are station list input errors, the station lists will be written to the default output file Unit No. KFILDO as well as to IP4. When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are directory input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5.

RESTRICTIONS:

None other than stated above.

COMMENTS:

This routine performs essentially the same function as RDSTGA, except that the stations are not alphabetized in RDSTGN and they are in RDSTGA, provided the station directory is alphabetized. If a station cannot be found in the directory, the last 5 columns of CCALL(,) will be blank. Note that all six columns of the directory are searched. That is, suppose a record in the directory contains 'KDEN   ', 'DEN   ', and 'DVX   ' in columns 1-8, 10-17, and 83-90, respectively. Either 'KDEN', 'DEN', or 'DVX' could be read by subroutine RDC, and KDEN would be returned in CCALL(,1). If KDEN changed to, say, 'KDUM' and the directory is updated, then 'KDUM' would be returned in CCALL(,1) and would be used in the calling program to identify output. This means that the directory record might be different on different data sets, but the MOS-2000 software should handle it, provided 'KDEN' is added to the entry in columns 92-99.

This routine can be used when more than one "group" of stations is being dealt with. For a single group, RDSTAL can be used. When RDSTAL is used, it is likely only one terminator is needed rather than two in the case of RDSTGN.

There are six routines which read a station list and return a list taken from either the first or second column of the directory, depending on the value of NEW; all six operate in the same way in this regard. Two routines deal with only a single list of stations and return the time zone. Four routines deal with multiple lists (groups) of stations. Three alphabetize the station list (by group) and three return the list in the order read. A group name is returned by two routines; this name is read immediately following the group list of stations with an A30 format. Their different characteristics are:

Routine	Single Station List	Multiple Station List	Group Name	Time Zone	Alphabetize
RDSTAL	X			X	
RDSTAD	X			X	X
RDSTGN		X			
RDSTGA		X			X
RDSTNL		X	X		
RDSTNA		X	X		X

RDSTGN is used in U600, U625, U660, U661, U710, U910, U830, and U602.

Many changes were made in October 2000 to assure that the station wanted from the directory was returned. Any directory entry can have up to 6 station identifiers, the leftmost being called the ICAO identifier, and next column being called the SAO call letters, although that nomenclature does not have to be followed. The rule followed is that the leftmost identifier that matches the identifier in the input list is the one used. That is, if a link I0V1 was in a particular record in a column 2-6 and was also in the first column, then the station in the first column is the one returned. The older version of RDSTGN did not assure this, and the result depended on the order in which the records were encountered.

Diagnostics were also improved, so that not only duplicate and missing stations are identified, but duplicates that may be caused by using a link for one of the stations and when a station is kept because of a link and the column (2 through 6) where the identifier occurred. In this context, SAO call letters are in Col. 2 when ICAO identifiers are being used (NEW = 1) and ICAO identifiers are in Col. 2 when SAO call letters are being used (NEW = 0).

In addition, station directory information is not printed to KFILDO except when IP5 = KFILDO or the error return is fatal. The table below shows the print structure as a function of the values of IER and IP5.

	IER =		
	0	33,34	35,36,37
IP5 = 0	---	KFILDO	---
IP5 = KFILDO	KFILDO	KFILDO	KFILDO
IP5 = IP5	IP5	KFILDO IP5	IP5

<u>NONSYSTEM ROUTINES USED</u>:

RDC

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

RDSTGA

READS AND ALPHABETIZES ONE OR MORE STATION LISTS AND DIRECTORY INFORMATION

Harry R. Glahn
August 1, 1997

<u>PURPOSE</u>: To read and alphabetize one or more lists of station call letters and associated information from a station directory. The lists read are composed of up to 8 characters per station. Each list is ended by the terminator '99999999' and reading is ended with an empty set. The stations in the list returned <u>will be alphabetized</u> by group provided the station directory is alphabetical. Any duplicate stations found will be kept, and a diagnostic provided.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
    CALL RDSTGA(KFILDO,IP4,IP5,KFILD,NEW,CCALL,CCALLD,NAME,NELEV,IWBAN,
                STALAT,STALON,IFOUND,NGP,NGPRUN,ND1,KGP,NSTA,IER)
```

<u>KFILDO</u> - Unit number of output (print) file. (INPUT)

<u>IP4</u> - When IP4 > 0, the station list (call letters only) will be written to Unit No. IP4. If there are input errors, the station list will be written to the default output file Unit No. KFILDO as well as to IP4. (INPUT)

<u>IP5</u> - When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5. (INPUT)

<u>KFILD</u>(J) - Unit number from which to read the station list (J=1) and station directory (J=2). It is assumed the files have been opened. KFILD(1) should not equal KFILD(2). (INPUT)

<u>NEW</u> - 1 when new ICAO station identifiers (call letters) are being used (columns 1-8 in the station directory);
0 when the old station call letters are being used (columns 10-17 in the station directory. (INPUT)

<u>CCALL</u>(K,J)- Array in which up to 6 sets (J=1,6) of 8 call letters per station are returned (K=1,NSTA). The lists of stations will be in alphabetical order <u>within groups</u>, given that the directory is in alphabetical order. CCALL(,1) will contain the call letters to be used as station or location identifiers in the calling program; the other 5 sets (J=2,6) are taken from the directory. The call letters returned in CCALL(,1) will be the ICAO station identifiers (columns 1-8) when NEW = 1, and will be the old call letters (columns 10-17) when NEW = 0. The other lists in CCALL(,J) always come from the directory, and for J=3,6 come from columns 83-90, 92-99, 101-108, and 110-117 in that order. The list in CCALL(,2) also comes from

the directory, being the ICAO call letters (columns 1-8) when NEW = 0 and the old call letters (columns 10-17) when NEW = 1. That is, it is intended that when CCALL(,1) contains the ICAO station identifiers, CCALL(,2) will contain the old call letters, and vice versa. (CHARACTER*8) (OUTPUT)

<u>CCALLD</u>(K) - Scratch array for holding unalphabetized list of stations (K=ND1). (CHARACTER*8) (INTERNAL)

<u>NAME</u>(K) - Names as read from the directory ordered according to CCALL(K,1) (K=1,NSTA). (CHARACTER*20) (OUTPUT)

<u>NELEV</u>(K) - Elevations of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>STALAT</u>(K) - Latitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>STALON</u>(K) - Longitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>IWBAN</u>(K) - WBAN numbers of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>IFOUND</u>(K) - Used to keep track of the stations found in the directory. (K=1,ND1). (INTERNAL)

<u>NGP</u>(K) - The sizes of KGP groups of stations (K=1,KGP), max of ND1. (OUTPUT)

<u>NGPRUN</u>(K) - Scratch array (K=1,ND1). (INTERNAL)

<u>ND1</u> - First dimension of CCALL(,), and dimension of NAME( ), NELEV( ), IWBAN( ), STALAT( ), STALON( ), IFOUND( ), and NGP( ). (INPUT)

<u>KGP</u> - The number of groups of stations read, and the number of values returned in NGP( ), max of ND1. (OUTPUT)

<u>NSTA</u> - The total number of elements (stations) returned in CCALL(,) and the associated arrays. (OUTPUT)

<u>IER</u> - Status return.
 0 = Good return.
20 = Error or EOF reading KFILD(1) by RDC. Aborts.
21 = Too many stations for dimension ND1 when reading by RDC. Aborts.
33 = Error reading station directory on Unit No. KFILD(2). Aborts.
34 = List to be kept too long for dimension ND1. Aborts with print.
35 = One or more stations not found in the directory. Normal return.
36 = One or more stations are duplicates. Normal return.
37 = Both IER = 35 and 36 above have occurred. Normal return.
(OUTPUT)

EXAMPLE

```
    PARAMETER (ND1=500)
    CHARACTER*8 CCALL(ND1,6),CCALLD(ND1)
    CHARACTER*20 NAME(ND1)
    DIMENSION NELEV(ND1),IWBAN(ND1),STALAT(ND1),STALON(ND1),IFOUND(ND1),
   1          NGP(ND1),NGPRUN(ND1)
    DIMENSION KFILD(2),IP(25)
    DATA IP/25*0/
    DATA KFILDO/12/
    DATA KFILD/28,29/,
   1     NEW/1/
    ...

    CALL RDSTGA(KFILDO,IP(4),IP(5),KFILD,NEW,CCALL,CCALLD,NAME,NELEV,
   1            IWBAN,STALAT,STALON,IFOUND,NGP,NGPRUN,ND1,KGP,NSTA,IER)
```

The default output unit number is 12. The unit number to read the station list from is 28, and the station directory is on Unit No. 29. Since IP(4) and IP(5) = 0, the station lists will not be printed, unless there are fatal errors, in which case the lists and the errors found will be on Unit No. 12. The NSTA ICAO identifiers are returned in CCALL(,1), along with the associated information in other columns of CCALL(,) and in the arrays as indicated above. CCALL(,1) will be in alphabetical order by group, provided the directory is alphabetical. The first NGP(1) values will be for group 1, the second NGP(2) values will be for group 2, etc. for all KGP groups. Since the station lists will be the ICAO station identifiers (NEW = 1), CCALL(,2) will contain the old call letters. The maximum number of stations to be returned is ND1. The station lists are read by subroutine RDC with FORMAT(7(A8,1X)) until the terminator '99999999' is found. An empty set will terminate the reading. Note that the end of all groups must have two terminators, one for the last non-empty group and one for the empty group. Normally, each record would contain only one station just for ease of user manipulation, but up to 7 stations could be in one record. A description of the station dictionary can be found elsewhere.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. When IP4 > 0, the station lists (call letters only) will be written to Unit No. IP4. If there are station list input errors, the station lists will be written to the default output file Unit No. KFILDO as well as to IP4. When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are directory input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5.

RESTRICTIONS:

None other than stated above.

COMMENTS:

This routine performs essentially the same function as RDSTGN, except that the stations are alphabetized in RDSTGA and they are not in RDSTGN. If a station cannot be found in the directory, the last 5 columns of CCALL(,) will be blank. Note that all six columns of the directory are searched. That is, suppose a record in the directory contains 'KDEN   ', 'DEN   ', and 'DVX   ' in columns 1-8, 10-17, and 83-90, respectively. Either 'KDEN', 'DEN', or 'DVX' could be read by subroutine RDC, and KDEN would be returned in CCALL(,1). If KDEN changed to, say, 'KDUM' and the directory is updated, then 'KDUM' would be returned in CCALL(,1) and would be used in the calling program to identify output. This means that the directory record might be different on different data sets, but the MOS-2000 software should handle it, provided 'KDEN' is added to the entry in columns 92-99.

This routine can be used when more than one "group" of stations is being dealt with. For a single group, RDSTAD can be used. When RDSTAD is used, it is likely only one terminator is needed rather than two in the case of RDSTGA.

There are six routines which read a station list and return a list taken from either the first or second column of the directory, depending on the value of NEW; all six operate in the same way in this regard. Two routines deal with only a single list of stations and return the time zone. Four routines deal with multiple lists (groups) of stations. Three alphabetize the station list (by group) and three return the list in the order read. A group name is returned by two routines; this name is read immediately following the group list of stations with an A30 format. Their different characteristics are:

Routine	Single Station List	Multiple Station List	Group Name	Time Zone	Alphabetize
RDSTAL	X			X	
RDSTAD	X			X	X
RDSTGN		X			
RDSTGA		X			X
RDSTNL		X	X		
RDSTNA		X	X		X

RDSTGA is used in U600, U625, U660, U661, U710, U910, U830, and U602.

Many changes were made in October 2000 to assure that the station wanted from the directory was returned. Any directory entry can have up to 6 station identifiers, the leftmost being called the ICAO identifier, and next column being called the SAO call letters, although that nomenclature does not have to be followed. The rule followed is that the leftmost identifier that matches the identifier in the input list is the one used. That is, if a link I0V1 was in a particular record in a column 2-6 and was also in the first column, then the station in the first column is the one returned. The older version of RDSTGA did not assure this, and the result depended on the order in which the records were encountered.

Diagnostics were also improved, so that not only duplicate and missing stations are identified, but duplicates that may be caused by using a link for one of the stations and when a station is kept because of a link and the column (2 through 6) where the identifier occurred. In this context, SAO call letters are in Col. 2 when ICAO identifiers are being used (NEW = 1) and ICAO identifiers are in Col. 2 when SAO call letters are being used (NEW = 0).

In addition, station directory information is not printed to KFILDO except when IP5 = KFILDO or the error return is fatal. The table below shows the print structure as a function of the values of IER and IP5.

	IER =		
	0	33,34	35,36,37
IP5 = 0	---	KFILDO	---
IP5 = KFILDO	KFILDO	KFILDO	KFILDO
IP5 = IP5	IP5	KFILDO IP5	IP5

<u>NONSYSTEM ROUTINES USED</u>:

RDC

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

COMPID

COMPUTES IDS FOR DATA IN MOS-2000 EXTERNAL FILE SYSTEM

Harry R. Glahn
December 1, 1996

PURPOSE: To provide the IDs for extracting constant data from MOS-2000 external random access files. COMPID is entered when CCC in ID(1) [IDPARS(1)] = 4xx. Since these data are expected to be used as predictors in U600 and the developed equations, the tau in ID(3) [IDPARS(12)] is added to NDATE to get the date for which date are required. Two sets of IDs are returned, the 2nd set being necessary only when interpolation is to be done (e.g., max temperatures provided only every 5th day). In addition, the day of the year (DOY) is returned as well as an "interpolation factor," the latter being the fraction of the way the DOY is from the 1st of the two values being used in interpolation.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL COMPID(KFILDO,ID,IDPARS,NDATE,KD,LD,MDOY,R,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

ID(J) - The predictor ID's (J=1,4). (INPUT)

IDPARS(J) - The MOS-2000 parsed ID (J=1,15):
 1 - CCC (class of variable)
 2 - FFF (subclass of variable)
 3 - B (binary indicator)
 4 - DD (data source, model number)
 5 - V (vertical application)
 6 - LLLL (lower level, 0 if only 1 level)
 7 - UUUU (upper level)
 8 - T (transformation)
 9 - RR (run time offset in hours, always + and back in time)
 10 - O (time application)
 11 - HH (time period in hours)
 12 - τττ (projection in hours)
 13 - I (interpolation type)
 14 - S (smoothing indicator)
 15 - G (grid indicator)
 (INPUT)

NDATE - The date/time for which the constant data are needed in YYYYMMDDHH format. (INPUT)

KD(J) - The 1st 4-word ID (J=1,4) for the constant. The calculation follows the format in the chapter "Variable Identification" (e.g., for daily values, the location in ID(2) corresponding to LLLL will be the DOY.) (OUTPUT)

LD(J)	-	The 2nd 4-word ID (J=1,4) for the constant. For constants that need not be interpolated, this return is not needed and will be the same as KD( ). (OUTPUT)
MDOY	-	The day of the year as calculated from NDATE. (OUTPUT)
R	-	The interpolation factor, being the fraction of the way the DOY is from the constant defined by KD( ) to the constant defined by LD( ). When interpolation is not needed, R = 0. (OUTPUT)
IER	-	Status return. 0 = Good return. 161 = Could not identify a CCCFFF for which to compute KD(2). 162 = Computational error. (OUTPUT)

EXAMPLE:

```
DIMENSION ID(4),KD(4),LD(4),IDPARS(15)
DATA KFILDO/6/
DATA ID/422006000,0,0,0/
DATA NDATE/1995012200/
...
[fill IDPARS( ) corresponding to ID( )]

CALL COMPID(KFILDO,ID,IDPARS,NDATE,KD,LD,MDOY,R,IER)
```

The default output file number is 6. ID(1) specifies calendar day max temperatures which are to be provided every 5th day on the constant file. The DOY is computed as 22 and returned in MDOY. KD( ) is returned as 422006000, 200000, 0, 0, and LD( ) is returned as 422006000, 250000, 0, 0. R is returned as .40, because MDOY is 40% of the way between DOY 20 and DOY 25, those days being indicated in KD(2) and LD(2), respectively. Had the projection tau not been zero in IDPARS(12), the DOY would have indicated that.

OUTPUT:

Diagnostic messages will be to Unit No. KFILDO.

RESTRICTIONS:

COMPID accommodates yearly, 2-season (April-September, etc.), 4-season (March-May, etc.), monthly, daily, and 5-day (values being at DOY numbers evenly divisible by 5) values. It is assumed the monthly values are to be interpolated to DOY from the midpoints of the months involved.

NONSYSTEM ROUTINES CALLED:

UPDAT, DOY

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDONEV

READS AND RETURNS A SPECIFIC VARIABLE FROM A TDLPACK VECTOR FILE

Harry R. Glahn
December 1, 1997

PURPOSE: To read a TDLPACK vector file on Unit No. KFILIN(IN), and return to the calling program the unpacked data when the variable read is in ID( ). When a terminator is found, indicating a following directory record, the directory record is read. When an EOF is found, a new file is opened if it exists. RDONEV is used by routines that need only one variable per date/time per input file. Therefore, internal storage and associated bookkeeping are not required. Only one variable is returned per call.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL RDONEV(KFILDO,KFILIN,NAMIN,JFOPEN,NUMIN,
   1            MSDATE,IN,ID,IDPARS,
   2            NDATE,CCALL,XDATA,INDEXC,ND1,NSTA,
   3            CCALLD,IPACK,IWORK,DATA,ND5,
   4            IS0,IS1,IS2,IS4,ND7,
   5            IP12,IP14,IP23,L3264B,L3264W,ISTOP,IER)
```

KFILDO - Unit number of the output (print) file. (INPUT)

KFILIN(J) - Unit numbers for input TDLPACK vector data (J=1,NUMIN). (INPUT)

NAMIN(J) - Names of files corresponding to KFILIN(J) (J=1,NUMIN). The file to be accessed on this call is in NAMIN(IN). (INPUT)

JFOPEN(J) - For each file identified in NAMIN( ), JFOPEN( ) takes the following values:
0 = file has already been used,
1 = file is open, and
2 = file has not yet been opened.
(INPUT/OUTPUT)

NUMIN - The number of values in KFILIN( ), NAMIN( ), and JFOPEN( ). Also treated as their dimensions.

MSDATE - Keeps track of whether any data are available for a particular date on an input file. MSDATE should be initialized to zero by the calling program for each date/time. It is set to 1 by RDONEV when a date/time has been found on that file. Used for diagnostic print when IP14 ≠ 0. (INPUT/OUTPUT)

IN - The index of KFILIN( ), NAMIN( ), and JFOPEN( ) designating the specific file in the list to be accessed in this call. If a trailer record is found on file (NAMIN(IN), file NAMIN(IN+1) is opened and its directory record read when NAMIN((N) =

NAMIN(IN+1); JFOPEN(IN) and JFOPEN(IN+1) are set to 0 and 1, respectively. (INPUT)

<u>ID</u>(J)	- The 4-word ID of the variable to return. (INPUT)
<u>IDPARS</u>(J)	- The parsed, individual components of the variable in ID( ) (J=1,15). (INPUT)
<u>NDATE</u>	- The date/time in format YYYYMMDDHH for which the variable is to be returned. (INPUT)
<u>CCALL</u>(K,J)	- 8-character station (or gridpoint location) call letters to provide data for (J=1) and 5 possible other station call letters (J=2,6) that can be used instead if the primary (J=1) station cannot be found in an input directory (K=1,NSTA). All station data returned are keyed to this list. (CHARACTER*8) (INPUT)
<u>XDATA</u>(K)	- The data returned for the variable identified in ID( ), (K=1,NSTA). These data are in the order specified by CCALL(,). Missing values are 9999 and secondary missing values of 9997 are returned as zero. (OUTPUT)
<u>INDEXC</u>(K,J)	- Locations of the NSTA stations (K=1,NSTA) for each of the input files (J=1,NUMIN) in the list CCALL(K,) in reference to the station call letters record on the input data set being dealt with. This is updated if another call letters record is encountered. The values in INDEXC(,) are used until this new call letters record is found, then the new values in INDEXC(,) put there by subroutine RDDIR are used. This array must be initialized for file KFILIN(IN) before RDONEV is entered. (INPUT/OUTPUT)
<u>ND1</u>	- The maximum number of stations for which data can be returned. (INPUT)
<u>NSTA</u>	- The number of stations in CCALL(,). (INPUT)
<u>CCALLD</u>(K)	- 8-character station call letters as read from an input file (K=1,ND5). Note that this may require ND5 to be greater than ND1. (INTERNAL)
<u>IPACK(J)</u>	- Holds the TDLPACK record (J=1,ND5). (INTERNAL)
<u>IWORK(J)</u>	- Work array (J=1,ND5). (INTERNAL)
<u>DATA(J)</u>	- Work array (J=1,ND5). (INTERNAL)
<u>ND5</u>	- Dimension of IPACK( ), IWORK( ), and DATA( ). This must be $\geq$ the maximum directory record size on any file accessed. (INPUT)

<u>IS0</u>(J)	-	The Section 0 decoded information from TDLPACK (J=1,ND7). (OUTPUT)
<u>IS1</u>(J)	-	The Section 1 decoded information from TDLPACK (J=1,ND7). (OUTPUT)
<u>IS2</u>(J)	-	The Section 2 decoded information from TDLPACK (J=1,ND7). (OUTPUT)
<u>IS4</u>(J)	-	The Section 4 decoded information from TDLPACK, except for the data themselves (J=1,ND7). (OUTPUT)
<u>ND7</u>	-	Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). A value of 54 is sufficient.
<u>IP12</u>	-	Indicates whether (>1) or not(=0) the list of stations on the input files will be provided on the file whose unit number is IP12. (INPUT)
<u>IP14</u>	-	Indicates whether (>1) or not(=0) a diagnostic will be provided on Unit IP14 when there are no data for a desired date/time on a particular input file. (With the switching of data sets, this is not of much use, and may be misleading.) (INPUT)
<u>IP23</u>	-	Indicates whether (>1) or not(=0) statements about EOF and file openings and closing will be output for printing on Unit IP23. (INPUT)
<u>L3264B</u>	-	Must have the value of 32 or 64, indicating the word length in bits of the machine being used. (INPUT)
<u>L3264W</u>	-	Must have the value of 1 or 2, indicating how many words on the machine being used there are in 64 bits. (INPUT)
<u>ISTOP</u>	-	Incremented by one each time an error is encountered. (INPUT/OUTPUT)
<u>IER</u>	-	Status return. 0 = Good return. 31 = Trouble opening file or switching files. 38 = Dimension IPACK( ) too small to hold the data being read. Note that this is not in reference to the number of stations being requested in CCALL(,), but the number on the input file. 137 = No more data available on the file being accessed. Not usually an error. 138 = 500 reading errors have occurred. 146 = Error reading directory record when switching files. Other returns may come from other subroutines. (OUTPUT)

EXAMPLE

```
    PARAMETER (L3264B=32)
    PARAMETER (L3264W=64/L3264B)
    PARAMETER (ND1=150)
    PARAMETER (ND5=1000)
    PARAMETER (ND7=54)
C
    CHARACTER*8 CCALL(ND1,6)
    CHARACTER*8 CCALLD(ND5)
    CHARACTER*20 NAMIN(NUMIN)

    DIMENSION XDATA(ND1)
    DIMENSION INDEX(ND1,NUMIN)
    DIMENSION IPACK(ND5),IWORK(ND5),DATA(ND5)
    DIMENSION IS0(ND7),IS1(ND7),IS2(ND7),IS4(ND7)
    DIMENSION KFILIN(NUMIN),JFOPEN(NUMIN)
    DIMENSION ID(4),IDPARS(15)
C
    DATA KFILDO/6/
    DATA NDATE/1995010100/
    DATA MSDATE/0/
    ...

    CALL RDONEV(KFILDO,KFILIN,NAMIN,JFOPEN,NUMIN,
   1            MSDATE,IN,ID,IDPARS,
   2            NDATE,CCALL,XDATA,INDEXC,ND1,NSTA,
   3            CCALLD,IPACK,IWORK,DATA,ND5,
   4            IS0,IS1,IS2,IS4,ND7,
   5            IP12,IP14,IP23,L3264B,L3264W,ISTOP,IER)
```

The default output unit number is 6. ID( ) is dimensioned 4 and holds the IDs of the variable desired. IDPARS( ) is dimensioned 15 and holds the 15 elements of ID( ). KFILIN( ), and NAMIN( ) must have been filled with NUMIN values and JFOPEN( ) set appropriately. CCALL(,) must have been filled with the call letters of the station locations for which data are wanted. The full identification of the data will be returned in IS0( ), IS1( ), IS2( ), and IS4( ) as appropriate. The data are fetched unpacked in XDATA( ) in the order specified in CCALL(,). Primary missing values of 9999 are returned as such, but secondary missing values of 9997 are returned as zero. The date for which the data are to be fetched is NDATE = 1995010100. The subroutine is called from a 32-bit machine. IER will be zero upon return unless a problem has been encountered.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. Other diagnostic information may be written to Unit Nos. IP12, IP14, and/or IP23, when these values are not zero.

RESTRICTIONS:

Care must be taken concerning JFOPEN( ), CCALL(,), and INDEXC(,); once initialized, RDONEV should take care of them.

COMMENTS:

RDONEV was adapted from RDVECT and is a somewhat simplified version, not having the lookahead capability and not using the MOS-2000 Internal Storage System. It is used by routines that need only one variable per date/time per input file. It switches files when necessary, and performs the other functions of RDVECT. Normally, one variable ID( ) would be wanted from each different value of KFILIN(J); the same value of KFILIN(J) in sequence means the same variable is wanted from each file and the file may have to be switched. If all values of KFILIN(J) are not equal, then the user will need to know which variable will come from each file.

In using RDONEV, CCALL(,) must be initialized, which can be done by calling RDSTGA, RDSTGN, RDSTAD, or RDSTAL, depending on whether the stations are to be in one list or more and whether the stations are to be alphabetized or not. KFILIN( ) and NAMIN( ) should be filled by calling RDSNAM, which will open the first file in a sequence of identical numbers in KFILIN( ) and set JFOPEN( ) appropriately for all file names read. This gives the total number of files in the list, NUMIN. Before calling RDONEV, the directory record must be read on the open files. This can be done by calling RDDIR, which will initialize INDEXC(,IN). This can be done with something like:

```
     Do 225 IN=1,NUMIN
     IF(JFOPEN(IN).NE.1)GO TO 225
     CALL RDDIR(...,KFILIN(IN),...,NAMIN(IN),...,INDEXC(1,IN),...,IER)
     ...
     Treat error returns IER
225  CONTINUE
```

NONSYSTEM ROUTINES USED:

UNPACK, UNPKBG, RDDIR

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDVECT

READS AND RETURNS A VARIABLE FROM A LIST FROM A TDLPACK VECTOR FILE

Harry R. Glahn
December 1, 1997

PURPOSE: To read a TDLPACK vector file on Unit No. KFILIN(IN), and return to the calling program the unpacked data when the variable read is in MSTORE(,J), J=1,LITEMS. When a terminator is found, indicating a following directory record (or an end of file), the directory record is read. RDVECT is used by VRBL62, which is called by U600 and U660, and must use the lookahead feature; that is, the data may not be needed in the order they exist on the input file. Therefore, internal storage and associated bookkeeping are required. Only one variable is returned per call. On each call, data are read until either a needed variable is found or until the date NDATE updated by MAXTAU is passed. When the data to be used are stored in the MOS-2000 Internal Storage System, GTVECT is later used to return the stored variable.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL RDVECT(KFILDO,KFIL10,KFILIN,NAMIN,JFOPEN,MDATE,NUMIN,
   1            MSDATE,MAXTAU,IN,IPRINT,PXMISS,
   2            ID,IDPARS,JD,LD,NWHERE,NVRBL,N,ISTAB,
   3            NDATE,CCALL,SDATA,XDATA,INDEXC,ND1,NSTA,
   4            CCALLD,IPACK,IWORK,DATA,ND5,
   5            LSTORE,MSTORE,ND9,LITEMS,MITEMS,CORE,ND10,
   6            LASTL,LASTD,NBLOCK,NSTORE,
   7            IS0,IS1,IS2,IS4,ND7,
   8            IP12,IP14,IP23,L3264B,L3264W,ISTOP,IER)
```

KFILDO	- Unit number of the output (print) file. (INPUT)
KFIL10	- Unit number of the MOS-2000 Internal File System access. (INPUT)
KFILIN(J)	- Unit numbers for input TDLPACK vector data (J=1,NUMIN). (INPUT)
NAMIN(J)	- Names of files corresponding to KFILIN(J) (J=1,NUMIN). The file to be accessed on this call is in NAMIN(IN). (INPUT)
JFOPEN(J)	- For each file identified in NAMIN( ), JFOPEN( ) takes the following values: 0 = file has already been used, 1 = file is open, and 2 = file has not yet been opened. (INPUT/OUTPUT)

MDATE(J) - For each file identified in NAMIN( ), MDATE( ) is the last date read by RDSTRX for Day 1. It is used to keep a diagnostic from printing in RDVECT

NUMIN - The number of values in KFILIN( ), NAMIN( ), and JFOPEN( ). Also treated as their dimensions. (INPUT)

MSDATE - Keeps track of whether any data are available for a particular date on an input file. MSDATE should be initialized to zero by the calling program for each date/time. It is set to 1 by RDVECT when a desired date/time has been found on that file. Used for diagnostic print when IP14 ≠ 0. (INPUT/OUTPUT)

MAXTAU - The maximum tau (projection) of any variable needed from file IN. On each call, data are read until either a needed variable is found or until the date NDATE updated by MAXTAU is passed. (INPUT)

IN - The index of KFILIN( ), NAMIN( ), and JFOPEN( ) designating the specific file in the list to be accessed in this call. If a trailer record is found on file NAMIN(IN), file NAMIN(IN+1) is opened and its directory record read when NAMIN(IN) = NAMIN(IN+1); JFOPEN(IN) and JFOPEN(IN+1) are set to 0 and 1, respectively. (INPUT)

IPRINT - Can be used by the calling program to keep a diagnostic from being printed for each missing variable when the entire day is missing. RDVECT sets IPRINT = 0 when an EOF is reached and MSDATE = 0. (INPUT/OUTPUT)

PXMISS - The value to substitute for a secondary missing value of 9997. This allows 9997 to be maintained if desired, set to zero for implied values of near zero, set to the normal missing value of 9999, or even set to some other value. RDVECT makes this substitution when the data are read and before return.

ID(J,N) - The 4-word ID of the variables to return, one at a time (J=1,4) (N=1,NVRBL). (INPUT)

IDPARS(J,N) - The parsed, individual components of the variable in ID(J,N) (J=1,15) (N=1,NVRBL). (INPUT)

JD(J,N) - The basic 4-word ID of the variables to return, one at a time (J=1,4) (N=1,NVRBL). This is the same as ID(J,N), except the following portions are omitted;
B = IDPARS(3,),
T = IDPARS(8,),
I = IDPARS(13,),
S = IDPARS(14,),
G = IDPARS(15,), and
THRESH( ).
(INPUT)

<u>LD</u>(J) - Holds the 3 ID words of the data saved in SDATA( ), and associated information (J=1,6). (OUTPUT)

<u>NWHERE</u>(N) - Indicates where the variable is to come from (N=1,NVRBL):
0 = Undetermined,
1 = from the input file,
2 = a binary from a basic variable, and
3 = computed in OPTX.
Generally, the first variable that requires the use of a variable directly from the input will get a value of 1. Other variables, when immediately following, that can use the same input variable will get a value of 2. (INPUT)

<u>NVRBL</u> - The number of variables identified in ID(,), IDPARS(,), JD(,), and NWHERE( ). Also used as their dimensions. (INPUT)

<u>N</u> - The location in the list ID(,N) of the variable returned. (OUTPUT)

<u>ISTAB</u> - 1 when the variable returned is binary; 0 otherwise. Note that it is not sufficient for IDPARS(3,) to be other than zero for the variable to be returned as binary. (OUTPUT)

<u>NDATE</u> - The date/time in format YYYYMMDDHH for which a variable is to be returned. (INPUT)

<u>CCALL</u>(K,J) - 8-character station (or gridpoint location) call letters to provide data for (J=1) and 5 possible other station call letters (J=2,6) that can be used instead if the primary (J=1) station cannot be found in an input directory (K=1,NSTA). All station data returned are keyed to this list. (CHARACTER*8) (INPUT)

<u>SDATA</u>(K) - Array in which the data returned in XDATA( ) may be saved and returned for possible reuse in the calling program (J=1,NSTA). These data are identified by LD( ) and are in the order specified by CCALL(,). This is an immediately accessible array outside the MOS-2000 Internal Storage System. Data are stored here only when the list in ID(,), JD(,), and IDPARS(,) indicates they may be needed. This option works well only when the variables in ID(,) are ordered (as they are in U600, but not necessarily as they are in U660). (OUTPUT)

<u>XDATA</u>(K) - The data returned for the variable identified in ID(,N), (K=1,NSTA). These data are in the order specified by CCALL(,). (OUTPUT)

<u>INDEXC</u>(K,J) - Locations of the NSTA stations (K=1,NSTA) for each of the input files (J=1,NUMIN) in the list CCALL(K,) in reference to the station call letters record on the input data set being dealt with. This is updated if another call letters record is encountered. The values in INDEXC(,) are used until this

new call letters record is found, then the new values in INDEXC(,) put there by subroutine RDDIR are used. This array must be initialized for file KFILIN(IN) before RDVECT is entered. (INPUT/OUTPUT)

<u>ND1</u>	- The maximum number of stations for which data can be returned. (INPUT)
<u>NSTA</u>	- The number of stations in CCALL(,). (INPUT)
<u>CCALLD</u>(K)	- 8-character station call letters as read from an input file (K=1,ND5). Note that this may require ND5 to be greater than ND1. (INTERNAL)
<u>IPACK(J)</u>	- Holds the TDLPACK record (J=1,ND5). (INTERNAL)
<u>IWORK(J)</u>	- Work array (J=1,ND5). (INTERNAL)
<u>DATA(J)</u>	- Work array (J=1,ND5). (INTERNAL)
<u>ND5</u>	- Dimension of IPACK( ), IWORK( ), and DATA( ). This must be $\geq$ the maximum directory record size on any file accessed. (INPUT)
<u>LSTORE</u>(L,J)	- Holds information about the data stored in the MOS-2000 Internal Storage System (L=1,12) (J=1,LITEMS). This is for communication between MOS-2000 infrastructure routines, and the user should not modify it. (INPUT/OUTPUT)
<u>MSTORE</u>(L,J)	- Holds the IDs and associated information about the data needed (L=1,8) (J=1,MITEMS). This is for communication between MOS-2000 infrastructure routines, and the user should not modify it. (INPUT)
<u>ND9</u>	- The second dimension of LSTORE(,) and MSTORE(,). Must be large enough so that LSTORE(,) can hold the IDs of all records stored in the MOS-2000 Internal Storage System at one time. (INPUT)
<u>LITEMS</u>	- The number of items (columns) in LSTORE(,) that are filled and being used at a particular time. This may be increased in RDVECT by subroutine GSTORE. (INPUT/OUTPUT)
<u>MITEMS</u>	- The number of items (columns) in MSTORE(,). (INPUT)
<u>CORE</u>(J)	- The array to store or retrieve the data identified in LSTORE(,) (J=1,ND10). When CORE( ) is full, the data are stored on disk. (INPUT/OUTPUT)
<u>ND10</u>	- The dimension of CORE( ). (INPUT)
<u>LASTL</u>	- The last location in CORE( ) used. (INPUT/OUTPUT)

<u>LASTD</u>	- The total number of physical records on disk in the MOS-2000 Internal Storage System. (INPUT/OUTPUT)
<u>NBLOCK</u>	- The block size in words of the random access disk file of the MOS-2000 Internal Storage System. (INPUT)
<u>NSTORE</u>	- Running count of number of times data are stored by GSTORE. Initialized to 0 the first time GSTORE is called and the value retained in GSTORE. (OUTPUT).
<u>IS0</u>(J)	- The Section 0 decoded information from TDLPACK (J=1,ND7). (INTERNAL)
<u>IS1</u>(J)	- The Section 1 decoded information from TDLPACK (J=1,ND7). (INTERNAL)
<u>IS2</u>(J)	- The Section 2 decoded information from TDLPACK (J=1,ND7). (INTERNAL)
<u>IS4</u>(J)	- The Section 4 decoded information from TDLPACK, except for the data themselves (J=1,ND7). (INTERNAL)
<u>ND7</u>	- Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). A value of 54 is sufficient.
<u>IP12</u>	- Indicates whether (>1) or not(=0) the list of stations on the input file(s) will be provided on the file whose unit number is IP12. (INPUT)
<u>IP14</u>	- Indicates whether (>1) or not(=0) a diagnostic will be provided on Unit IP14 when there are no data for a desired date/time on a particular input file. (With the switching of data sets, this is not of much use, and may be misleading.) (INPUT)
<u>IP23</u>	- Indicates whether (>1) or not(=0) statements about EOF and file openings and closing will be output for printing on Unit IP23. (INPUT)
<u>L3264B</u>	- Must have the value of 32 or 64, indicating the word length in bits of the machine being used. (INPUT)
<u>L3264W</u>	- Must have the value of 1 or 2, indicating how many words on the machine being used there are in 64 bits. (INPUT)
<u>ISTOP</u>	- Incremented by one each time an error is encountered. (INPUT/OUTPUT)
<u>IER</u>	- Status return. 0 = Good return. 31 = Trouble opening file or switching files. 38 = Dimension IPACK( ) too small to hold the data being read. Note that this is not in reference to the number

```
                      of stations being requested in CCALL( , ), but the
                      number on the input file.
                137 = No more data available on the file being accessed.  Not
                      usually an error.
                138 = 500 reading errors have occurred.  The number 500 is
                      adjustable in RDVECT as variable LSTOP.
                146 = Error reading first directory record when switching
                      files.
                Other returns may come from other subroutines.
                (OUTPUT)
```

<u>EXAMPLE</u>

```
    PARAMETER (L3264B=32)
    PARAMETER (L3264W=64/L3264B)
    PARAMETER (ND1=150)
    PARAMETER (ND4=50)
    PARAMETER (ND5=1000)
    PARAMETER (ND7=54)
    PARAMETER (ND9=100)
    PARAMETER (ND10=10000)
C
    CHARACTER*8 CCALL(ND1,6)
    CHARACTER*8 CCALLD(ND5)
    CHARACTER*20 NAMIN(NUMIN)

    DIMENSION SDATA(ND1),XDATA(ND1)
    DIMENSION INDEX(ND1,NUMIN)
    DIMENSION ID(4,ND4),IDPARS(15,ND4),JD(4,ND4),NWHERE(ND4)
    DIMENSION IPACK(ND5),IWORK(ND5),DATA(ND5)
    DIMENSION IS0(ND7),IS1(ND7),IS2(ND7),IS4(ND7)
    DIMENSION LSTORE(12,ND9),MSTORE(8,ND9)
    DIMENSION CORE(ND10)
    DIMENSION KFILIN(NUMIN),JFOPEN(NUMIN),MDATE(NUMIN)
C
    DATA KFILDO/6/,
   1     KFILIN/20/
    DATA NDATE/1995010100/
    DATA NBLOCK/300/
    DATA MSDATE/0/
    DATA PXMISS/0./
    ...

    CALL RDVECT(KFILDO,KFIL10,KFILIN,NAMIN,JFOPEN,MDATE,NUMIN,
   1            MSDATE,MAXTAU,IN,IPRINT,PXMISS,
   2            ID,IDPARS,JD,LD,NWHERE,NVRBL,N,
   3            NDATE,CCALL,SDATA,XDATA,INDEXC,ND1,NSTA,
   4            CCALLD,IPACK,IWORK,DATA,ND5,
   5            LSTORE,MSTORE,ND9,LITEMS,MITEMS,CORE,ND10,
   6            LASTL,LASTD,NBLOCK,NSTORE,
   7            IS0,IS1,IS2,IS4,ND7,
   8            IP12,IP14,IP23,L3264B,L3264W,ISTOP,IER)
```

The default output unit number is 6. ID(,) is dimensioned 4 by ND1 and holds the IDs of the NVRBL variables desired; when one of them is encountered in reading, it is returned. IDPARS(,) is dimensioned 15 by ND1 and holds the 15 elements of ID(,). LSTORE must be set up with dimensions, but is used only internally by GSTORE. ND9 is the second dimension of LSTORE, and upon return from GSTORE, LITEMS will be the number of items in LSTORE(,), maximum of ND9. KFILIN( ), NAMIN( ) and MDATE( ) must have been filled with NUMIN values and JFOPEN( ) and NWHERE( ) set appropriately. MSTORE(,MITEMS) must have been filled with the MITEMS needed to be returned. CCALL(,) must have been filled with the call letters of the station locations for which data are wanted. The full identification of the data, when they have been packed in TDLPACK format, will be returned in IS0( ), IS1( ), IS2( ), and IS4( ) as appropriate. The data fetched are in XDATA( ) in the order specified in CCALL(,), and also in SDATA( ) when the list in MSTORE(,) indicates the data may be needed again. The ID of the data returned is ID(,N). The data are returned unpacked; any value of 9997 found is set to the PXMISS value of zero. The date for which the data are to be fetched is NDATE = 1995010100. ND10 = 10000 locations are available for storage of data in CORE( ); when full, the disk will be used, which has a block size of 300 words. The subroutine is called from a 32-bit machine. IER will be zero upon return unless a problem has been encountered.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. Other diagnostic information may be written to Unit Nos. IP12, IP14, and/or IP23, when these values are not zero.

RESTRICTIONS:

LSTORE(,) is for the exclusive use of GSTORE, GFETCH, RDVECT, LMSTR1, LMSTR2, and GCPAC. While information could be obtained from it, it should not be modified, and any information actually needed should be returned in the call sequence. Care must be taken in setting the dimensions of the variables involved, as well as the block size for the MOS-2000 Internal Storage System disk. Care must be taken concerning JFOPEN( ), CCALL(,), and INDEXC(,); once initialized, RDVECT should take care of them.

COMMENTS:

RDVECT and its associate GTVECT may be difficult to use in new programs unless the MOS-2000 infrastructure is well understood. However, several programs (will) need the same functions as performed by RDVECT and GTVECT, so they should be used if practical. Quite likely, the best approach to using RDVECT is to use, say, VRBL62 as a template, rather than starting from scratch with this writeup; use the writeup to supplement the information in the template. For more information on the MOS-2000 Internal Storage System, see Chapter 8 in the TDL office note titled MOS-2000.

Since the data being returned are vector, they have already been unpacked and keyed to the call letters. Therefore, IS0( ), IS1( ), IS2( ), and IS4( ) are not needed. However, they may be needed as work arrays in CONSTX through OPTX.

NONSYSTEM ROUTINES USED:

UNPACK, UPDAT, UNPKBG, RDDIR, GSTORE

LANGUAGE: FORTRAN 77

LOCATION: moslib

RDNAME

READS PLAIN LANGUAGE AND SCALING PARAMETERS FROM VARIABLE CONSTANT FILE

Harry R. Glahn
December 1, 1997

PURPOSE: To read the plain language and scaling parameters for a set of variables from the Variable Constant File. The plain language in the file is augmented appropriately. For instance, a "GB" is placed in columns 28 and 29 of the plain language when the variable is a grid binary.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL RDNAME(KFILDO,KFILCP,IP9,ID,IDPARS,JD,ISCALD,PLAIN,ND4,NVRBL,
   1            ISTOP,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

KFILCP - Unit number from which to read the Variable Constant File. (INPUT)

IP9 - Indicates whether (>0) or not (=0) the variable list will be written to Unit IP9 after the plain language is attached. (INPUT)

ID(J,N) - The 4-word (J=1,4) MOS-2000 predictor IDs for NVRBL variables for which data are to be acquired from the Variable Constant File (N=1,NVRBL). (INPUT)

IDPARS(J,N) - The MOS-2000 parsed IDs corresponding to ID(,N) (J=1,15) (N=1,NVRBL). (INPUT)

JD(J,N) - The basic 4-word (J=1,4) MOS-2000 predictor IDs for NVRBL variables for which data are to be acquired from the Variable Constant File (N=1,NVRBL). JD(J,N) is the same as ID(J,N), except that the portions pertaining to processing are omitted:
B = IDPARS(3,N),
T = IDPARS(8,N),
I = IDPARS(13,N),
S = IDPARS(14,N),
G = IDPARS(15,N), and
THRESH( ). (INPUT)

ISCALD(N) - Scaling constants to use when packing the data (N=1,NVRBL). Packed value = original value *10^{ISCALD} rounded to INTEGER. (OUTPUT)

PLAIN(N) - 32 characters of plain language description of the predictors (N=1,NVRBL). (CHARACTER*32) (OUTPUT)

ND4 - The dimension of variable ISCALD( ), and the second dimension of ID(,), IDPARS(,), AND JD(,). (INPUT)

NVRBL - The number of variables identified in ID(,), etc., maximum of ND4. (INPUT)

ISTOP - ISTOP is incremented by 1 if a reading error is encountered. (INPUT-OUTPUT)

IER - Status return. (OUTPUT)
 0 = Good return.
 Other returns from the Fortran read are possible.

EXAMPLE

```
    PARAMETER (ND4=50)

    CHARACTER*32 PLAIN(ND4)

    DIMENSION ID(4,ND4),IDPARS(15,ND4),JD(4,ND4),ISCALD(ND4)

    DATA KFILDO/6/
    DATA IP9/30/
    ...
    CALL RDNAME(KFILDO,KFILCP,IP9,ID,IDPARS,JD,ISCALD,PLAIN,ND4,NVRBL,
   1            ISTOP,IER)
```

The default output unit number is 6. The unit number for reading the constant file has been defined, probably by RDSNAM, in KFILCP. The NVRBL variables for which the information is to be taken from the Variable Constant File must have been defined in ID(,), JD(,), and IDPARS(,). ISTOP will be incremented by one is there is an error; the only errors possible are those associated with a read, in which case IER will be the IOS reading error.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. When IP9 ≠ 0, the variable list along with the plain language and scaling parameters will be written to Unit IP9.

RESTRICTIONS:

None other than stated above.

COMMENTS:

A reading error will cause an immediate return.

NONSYSTEM ROUTINES USED:

None

LANGUAGE: FORTRAN 77

LOCATION: moslib

GTVECT

RETURNS A VECTOR VARIABLE FROM MOS-2000 INTERNAL STORAGE

Harry R. Glahn
January 1, 1998

PURPOSE: To retrieve a vector record from MOS-2000 Internal Storage or a MOS-2000 External Random Access file and return to the calling program the unpacked data requested. GTVECT is used by VRBL62, which is called by U600 and other "vector" routines that use the lookahead feature when a "predictand" variable is requested. GTVECT is a companion to RDVECT. RDVECT reads data, unpacks them, and stores (only) the values for the stations indicated in CCALL( ) in the MOS-2000 Internal Storage system when indicated by MSTORE(,); GTVECT will then retrieve such data. If the "raw" variable is not available, OPTX is called to compute the required variable.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL GTVECT(KFILDO,KFIL10,RACESS,IPRINT,
   1            ID,IDPARS,TRESHL,JD,ITAU,NWHERE,NVRBL,N,ISTAB,
   2            NDATE,CCALL,ISDATA,SDATA,XDATA,ND1,NSTA,
   3            ICALLD,CCALLD,IPACK,IWORK,DATA,ND5,
   4            LSTORE,ND9,LITEMS,CORE,ND10,
   5            LASTL,LASTD,NBLOCK,NSTORE,NFETCH,
   6            IS0,IS1,IS2,IS4,ND7,
   7            L3264B,L3264W,ISTOP,IER)
```

KFILDO	- Unit number of the output (print) file. (INPUT)
KFIL10	- Unit number of the MOS-2000 Internal File System access. (INPUT)
RACESS	- File name for the MOS-2000 External Random Access file. (CHARACTER*60) (INPUT)
IPRINT	- Used to keep a diagnostic from being printed (IPRINT = 0) for each missing variable when the entire day is missing. (INPUT)
ID(J,N)	- The 4-word ID's of the variables, one of which is to be returned (J=1,4) (N=1,NVRBL). (INPUT)
IDPARS(J,N)	- The parsed, individual components of the variable in ID(J,N) (J=1,15) (N=1,NVRBL). (INPUT)
TRESHL(N)	- The lower binary threshold associated with IDPARS(,N) (N=1,NVRBL). (INPUT)
JD(J,N)	- The basic 4-word ID of the variable N to return (J=1,4) (N=1,NVRBL). This is the same as ID(J,N), except the following portions are omitted:

```
                B = IDPARS(3, ),
                T = IDPARS(8, ),
                I = IDPARS(13, ),
                S = IDPARS(14, ),
                G = IDPARS(15, ), and
                THRESH( ).
                JD( , ) is used to retrieve a basic field, before processing
                has been done.
                (INPUT)
```

<u>ITAU</u>(N) - The number of hours to add to NDATE to get the date/time for which to return the data. (INPUT)

<u>NWHERE</u>(N) - Indicates where the variable is to come from (N=1,NVRBL):
0 = Undetermined,
1 = from the input file,
2 = a binary from a basic variable, and
3 = computed in OPTX.
Generally, the first variable that requires the retrieval directly from MOS-2000 Internal Storage will get a value of 1. Other variables, when immediately following, that can use the same retrieved variable will get a value of 2. (INPUT)

<u>NVRBL</u> - The number of variables identified in ID(,), IDPARS(,), JD(,), ITAU( ), and NWHERE( ). Also used as their dimensions. (INPUT)

<u>N</u> - The location in the list ID(,N) of the variable to be returned. (INPUT)

<u>ISTAB</u> - 1 when the variable returned is binary; 0 otherwise. (OUTPUT)

<u>NDATE</u> - The date/time in format YYYYMMDDHH that is being processed. ITAU is added to NDATE to get the date/time for which to return data. Note that the tau in IDPARS(12,N) may also come into play. (INPUT)

<u>CCALL</u>(K,J) - 8-character station (or gridpoint location) call letters to provide data for (J=1) and 5 possible other station call letters (J=2,6) that can be used instead if the primary (J=1) station cannot be found in an input directory (K=1,NSTA). All station data in the MOS-2000 Internal Storage System have been previously keyed to this list by RDVECT or will be in GTVECT from a MOS-2000 External Random Access file through OPTX and CONSTX. (CHARACTER*8) (INPUT)

<u>ISDATA</u>(K) - Work array (K=1,ND1). (INTERNAL)

<u>SDATA</u>(K) - Array in which the data returned in XDATA( ) may be saved and reused in a subsequent call to GTVECT (J=1,NSTA). These data are internally identified by LD( ) and are in the order specified by CCALL(,). This is an immediately accessible array outside the MOS-2000 Internal Storage System. Data are stored here only when the list in ID(,), JD(,), and

IDPARS(,) indicates they may be needed. This option works well only when the variables in ID(,) are ordered (as they are in U600, but not necessarily as they are in U660). (INTERNAL)

XDATA(K)	- The data returned for the variable identified in ID(,N), (K=1,NSTA). These data are in the order specified by CCALL(,). (OUTPUT)
ND1	- The maximum number of stations for which data can be returned. (INPUT)
NSTA	- The number of stations in CCALL(,). (INPUT)
ICALLD(L,K)	- 8-character station call letters as characters in an integer variable (L=1,L3264W) (K=1,ND5). Equivalenced in calling program to CCALLD( ). (INTERNAL)
CCALLD(K)	- 8-character station call letters (K=1,ND5). Note that this may require ND5 to be greater than ND1. (CHARACTER*8) (INTERNAL)
IPACK(J)	- Work array (J=1,ND5). (INTERNAL)
IWORK(J)	- Work array (J=1,ND5). (INTERNAL)
DATA(J)	- Work array (J=1,ND5). (INTERNAL)
ND5	- Dimension of IPACK( ), IWORK( ), and DATA( ). This must be $\geq$ the maximum directory record size on any file accessed. (INPUT)
LSTORE(L,J)	- Holds information about the data stored in the MOS-2000 Internal Storage System (L=1,12) (J=1,LITEMS). This is for communication between MOS-2000 infrastructure routines, and the user should not modify it. (INPUT/OUTPUT)
ND9	- The second dimension of LSTORE(,). Must be large enough so that LSTORE(,) can hold the IDs of all records stored in the MOS-2000 Internal Storage System at one time. (INPUT)
LITEMS	- The number of items (columns) in LSTORE(,) that are filled and being used at a particular time. (INPUT/OUTPUT)
CORE(J)	- The array to retrieve the data from identified in LSTORE(,) (J=1,ND10). When CORE( ) is full, the data are stored on disk. (INPUT/OUTPUT)
ND10	- The dimension of CORE( ). (INPUT)
LASTL	- The last location in CORE( ) used. (INPUT/OUTPUT)
LASTD	- The total number of physical records on disk in the MOS-2000 Internal Storage System. (INPUT/OUTPUT)

NBLOCK	- The block size in words of the random access disk file of the MOS-2000 Internal Storage System. (INPUT)
NSTORE	- Running count of number of times data are stored by GSTORE. Initialized to 0 the first time GSTORE is called and the value retained in GSTORE. (OUTPUT).
IS0(J)	- Work array (J=1,ND7). (INTERNAL)
IS1(J)	- Word array (J=1,ND7). (INTERNAL)
IS2(J)	- Work array (J=1,ND7). (INTERNAL)
IS4(J)	- Work array (J=1,ND7). (INTERNAL)
ND7	- Dimension of IS0( ), IS1( ), IS2( ), and IS4( ). A value of 54 is sufficient.
L3264B	- Must have the value of 32 or 64, indicating the word length in bits of the machine being used. (INPUT)
L3264W	- Must have the value of 1 or 2, indicating how many words on the machine being used there are in 64 bits. (INPUT)
ISTOP	- Incremented by one each time an error is encountered. (INPUT/OUTPUT)
IER	- Status return. 0 = Good return. 139 = Missing data for the variable N being requested. Other returns may come from other subroutines. (OUTPUT)

EXAMPLE

```
      PARAMETER (L3264B=32)
      PARAMETER (L3264W=64/L3264B)
      PARAMETER (ND1=150)
      PARAMETER (ND4=50)
      PARAMETER (ND5=1000)
      PARAMETER (ND7=54)
      PARAMETER (ND9=100)
      PARAMETER (ND10=10000)
C
      CHARACTER*8 CCALL(ND1,6)
      CHARACTER*8 CCALLD(ND5)
      CHARACTER*60) RACESS

      DIMENSION ISDATA(ND1),SDATA(ND1),XDATA(ND1)
      DIMENSION ID(4,ND4),IDPARS(15,ND4),TRESHL(ND4),JD(4,ND4),
     1          ITAU(ND4),NWHERE(ND4)
      DIMENSION ICALLD(L3264W,ND5),IPACK(ND5),IWORK(ND5),DATA(ND5)
      DIMENSION IS0(ND7),IS1(ND7),IS2(ND7),IS4(ND7)
      DIMENSION LSTORE(12,ND9)
      DIMENSION CORE(ND10)
C
```

```
      DATA KFILDO/6/
      DATA NDATE/1995010100/
      DATA NBLOCK/300/
      ...

      CALL GTVECT(KFILDO,KFIL10,RACESS,IPRINT,
     1            ID,IDPARS,TRESHL,JD,ITAU,NWHERE,NVRBL,N,ISTAB,
     2            NDATE,CCALL,ISDATA,SDATA,XDATA,ND1,NSTA,
     3            ICALLD,CCALLD,IPACK,IWORK,DATA,ND5,
     4            LSTORE,ND9,LITEMS,CORE,ND10,
     5            LASTL,LASTD,NBLOCK,NSTORE,NFETCH,
     6            IS0,IS1,IS2,IS4,ND7,
     7            L3264B,L3264W,ISTOP,IER)
```

The default output unit number is 6. ID(,) is dimensioned 4 by ND1 and holds the IDs of the NVRBL variables being dealt with; the one being asked for is in ID(,N). IDPARS(,) is dimensioned 15 by ND1 and holds the 15 elements of ID(,); TRESHL( ) holds the associated lower threshold of each variable. LSTORE must be set up with dimensions, but is used only internally by GSTORE. ND9 is the second dimension of LSTORE, and LITEMS is be the number of items in LSTORE(,), maximum of ND9. CCALL(,) must have been filled with the call letters of the station locations for which data are wanted. The data fetched are in XDATA( ) in the order specified in CCALL(,), and also in SDATA( ) when the list in ID(,) indicates the data may be needed again. The data are returned unpacked. The date for which the data are to be fetched is NDATE = 1995010100 modified appropriately by ITAU(N) and IDPARS(12,N). ND10 = 10000 locations are available for storage of data in CORE( ); when full, the disk will be used, which has a block size of 300 words. The subroutine is called from a 32-bit machine. IER will be zero upon return unless a problem has been encountered.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO.

RESTRICTIONS:

LSTORE(,) is for the exclusive use of GSTORE, GFETCH, RDVECT, LMSTR1, LMSTR2, and GCPAC. While information could be obtained from it, it should not be modified, and any information actually needed should be returned in the call sequence. Care must be taken in setting the dimensions of the variables involved, as well as the block size for the MOS-2000 Internal Storage System disk.

COMMENTS:

GTVECT and its associate RDVECT may be difficult to use in new programs unless the MOS-2000 infrastructure is well understood. However, several programs (will) need the same functions as performed by RDVECT and GTVECT, so they should be used if practical. Quite likely, the best approach to using GTVECT is to use, say, VRBL62 as a template, rather than starting from scratch with this writeup; use the writeup to supplement the information in the template. For more information on the MOS-2000 Internal Storage System, see Chapter 8 in the TDL office note titled MOS-2000.

Since the data being returned are vector, they have already been unpacked and keyed to the call letters. Therefore, IS0( ), IS1( ), IS2( ), and IS4( ) are not needed. However, they may be needed as work arrays in CONSTX through OPTX.

When a variable is to be used as a predictor and may be entered into an equation, then the ITAU( ) will not be available and the information must be carried in IDPARS(12,N). This is true of "constants" (CCC = 4xx) and observations (CCC = 7xx) and certain other variables such as trigonometric function (CCC = 010).

<u>NONSYSTEM ROUTINES USED</u>:

OPTX, TIMPR

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

SETPLN

PLACES PLAIN LANGUAGE FOR VARIABLES

Harry R. Glahn
May 1, 1998

PURPOSE: To read the variable constant file (see MOS documentation, TDL ON 98-xx, Chapter 11), match the entries with a list of variables, and provide to the calling program the constant data for those variables.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
    CALL SETPLN(KFILDO,KFILCP,ID,IDPARS,JD,ISCALD,SMULT,SADD,ORIGIN,
   1            CINT,PLAIN,UNITS,ND4,NVRBL,ISTOP,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

KFILCP - Unit number of the variable constant file. (INPUT)

ID(J,N) - The list (N=1,NVRBL) of MOS 4-word IDs (J=1,4) for which constants are desired. (INPUT)

IDPARS(J,N) - The MOS-2000 parsed ID (J=1,15) (N=1,NVRBL):
1 - CCC (class of variable)
2 - FFF (subclass of variable)
3 - B (binary indicator)
4 - DD (data source, model number)
5 - V (vertical application)
6 - LLLL (lower level, 0 if only 1 level)
7 - UUUU (upper level)
8 - T (transformation)
9 - RR (run time offset in hours, always + and back in time)
10 - O (time application)
11 - HH (time period in hours)
12 - $\tau\tau\tau$ (projection in hours)
13 - I (interpolation type)
14 - S (smoothing indicator)
15 - G (grid indicator)
(INPUT)

JD(J,N) - The basic variable IDs (J=1,4) (N=1,NVRBL). These are the same as the IDs, except that B, T, I, S, and G (see MOS documentation) are omitted. (INPUT)

ISCALD(N) - The decimal scale factor associated with ID(,N) taken from the variable constant file. (OUTPUT)

SMULT(N) - The factor by which the values associated with ID(,N) are to be multiplied before gridprinting. Taken from the variable constant file. (OUTPUT)

SADD(N) - The factor to be added to the values associated with ID(,N), after multiplying by SMULT(,N) before gridprinting. (OUTPUT)

ORIGIN(N) - The origin for contouring values for the variable ID(, N) in terms of the units in UNITS(N) (N=1,NVRBL). (OUTPUT)

CINT(N) - The contour interval for contouring in terms of units in UNITS(N) (N=1,NVRBL). (OUTPUT)

PLAIN(N) - The plain language of the variable. (CHARACTER*32) (OUTPUT)

UNITS(N) - The units of the data associated with ID(, N). (CHARACTER*12) (OUTPUT)

ND4 - The maximum value of NVRBL. Dimension of ISCALD( ), SMULT( ), SADD( ), ORIGIN( ), and CINT( ), and second dimension of ID(,), IDPARS(,), and JD(,). (INPUT)

NVRBL - The number of values in ID(,N), etc. (INPUT)

ISTOP - Incremented by one when there is a system (reading) error. (INPUT/OUTPUT)

IER - Status return. The only return provided is a good return, IER = 0. This does not mean that every variable has been located in the input file. (OUTPUT)

EXAMPLE:

```
      PARAMETER (ND4=100)
C
      CHARACTER*12 UNITS(ND4)
      CHARACTER*32 PLAIN(ND4)
C
      DIMENSION ID(4,ND4),IDPARS(15,ND4),JD(4,ND4)
      DIMENSION ISCALD(ND4),SMULT(ND4),SADD(ND4),ORIGIN(ND4),
     1          CINT(ND4)
      ...
      DATA KFILDO/12/,
     1     KFILCP/23/
      DATA ISTOP/0/

      [fill ID( , ), IDPARS( , ), and JD( , ) with N=NVRBL values.]

      CALL SETPLN(KFILDO,KFILCP,ID,IDPARS,JD,ISCALD,SMULT,SADD,ORIGIN,
     1            CINT,PLAIN,UNITS,ND4,NVRBL,ISTOP,IER)
```

The output file number is 12. The unit number for the variable constant file is 23. The data from the constant file for the NVRBL variables in ID(,N) comes back in ISCALE(N), etc.

OUTPUT:

Diagnostic messages will be to Unit No. KFILDO.

RESTRICTIONS:

As more specific IDs are entered into the variable constant file, SETPLN may have to be modified. Generally, the more specific ID in the file should precede the more generic. Whenever a new entry is made in the file, the output of SETPLN should be checked. This can usually be easily done, because such programs as U201, U600, and U660 list the variables with the plain language, etc. The critical thing here is whether it finds the correct variable in the list.

COMMENTS:

Other processing occurs with the reading of these records, much of it associated with the plain language description. This is done so that these records can be somewhat generic and apply to many variables as defined in ID(,).

When only one level is involved ("V" = 0) the level UUUU in ID(2,N) is inserted into PLAIN(N)(1:4) so that each level for an otherwise similar variable will not have to be included in this file.

When "V" = 1 in ID word 2, the plain language is tailored so that PLAIN(N)(1:10) contains the two levels UUUU and LLLL in ID(2,N) separated by a "-" (indicating a layer difference). Also, ORIGIN(N) and SADD(N) are set = 0, those values being more appropriate for a difference field.

When "V" = 2 in ID word 2, the plain language is tailored so that PLAIN(N)(1:10) contains the two levels UUUU and LLLL in ID(2,N) separated by a "+" (indicating a layer sum). Also, ORIGIN(N) and SADD(N) are doubled, those values being more appropriate for a difference field.

When "V" = 3 in ID word 2, the plain language is tailored so that PLAIN(N)(1:15) contains the two levels UUUU and LLLL in ID(2,N) separated by an "A" (indicating a layer average).

When this is a time lag variable (O ≠ 0 in ID word 3), an "LX" is inserted into PLAIN(N)(24:25) where "X" is the digit O (this is the value of "OH", not a "zero"). In this case, the full information cannot be carried in the plain language. The tau will be as indicated in IDPARS(12) and the date will be the basic date updated with -IDPARS(9).

When this is a grid binary variable ("B" = 5 in ID word 1), the following substitutions are made: SMULT(N) = 100; SADD(N) = 0; PLAIN(N)(28:29) = "GB"; UNITS(N)(1:12) = " "; ORIGIN(N) = .5; CINT(N) = .25, and ISCALD(N) = 2.

When this is a point binary variable ("B" = 1 in ID word 1), the following substitutions are made: SMULT(N) = 1, SADD(N) = 0; PLAIN(29) = "B"; UNITS(N) = " "; ORIGIN(N) = .5; CINT(N) = 1; and ISCALD(N) = 0.

When "S" in ID word 4 is 1, 2, or 3, a smoothing indicator "S5 ", "S9 ", or "S25", respectively, is inserted into PLAIN(N)(30:32).

When a transformation is to be made (on the interpolated values) ("T" ≠ 0 in ID word 3), "T" is inserted into PLAIN(N)(27) and UNITS(N)(1:12) is set to " ".

When a 1-dimensional linearization is done, "D1" is inserted into PLAIN(N)(26:27).

When a 2-dimensional linearization is done, "D2" is inserted into PLAIN(N)(26:27).

When a variable has a CCC in the range 400-499 ("constant" data), a "C" is inserted into PLAIN(26).

If IDPARS(4,N) = 80, "MOS" is inserted into PLAIN(N)(6:8).
If IDPARS(4,N) = 81, "LCL" is inserted into PLAIN(N)(6:8).
If IDPARS(4,N) = 82, "OBS" is inserted into PLAIN(N)(6:8).
If the model number in the entry in the variable constant file = 82, "OBS" is inserted into PLAIN(N)(6:8).

Additions will be needed to this file as new variables are defined. Note that entries can be rather generic. That is, one entry can apply to all models (DD = 0 in the entry). If a specific model is to have a different value of, say, ISCALD( ), than other models, then include the specific entry (DD ≠ 0) for that model and be sure it precedes the generic definition for the same CCCFFF. While the definitions so far only depend on CCCFFFBFF (in ID(1)), later entries may have to depend on other words in the ID.

<u>NONSYSTEM ROUTINES CALLED</u>:

None

<u>LANGUAGE:</u> FORTRAN 77

<u>LOCATION</u>: moslib

SMTH2X

SMOOTHS A FIELD WITH A 25-POINT SMOOTHER TWICE

Harry R. Glahn
May 1, 1999

PURPOSE: A 2-dimensional array is smoothed with a 25-point smoother. Each resulting smoothed point is the average of itself and the 24 points in a square box around it (i.e., all points within a square 5 points on a side, with the point to be changed in the center, are averaged. A linear gradient is assumed for the outer rows and columns (two on each side). Some accommodation is made along the edges. The corner points are unchanged. Then the smoothed array is smoothed again in the same manner. This, then, gives a value at a gridpoint influence over a 9 X 9 area. No error checking is done and a status return is not provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL SMTH2X(KFILDO,P,FD,NX,NY)
```

KFILDO	- Unit number of output (print) file. (INPUT)
P(IX,JY)	- The input variable to smooth (IX=1,NX; JY=1,NY). Normally, these are values at gridpoints. On exit, P(,) contains the smoothed variable. (INPUT-OUTPUT)
FD(IX,JY)	- A work array (IX=1,NX; JY=1,NY). (INTERNAL)
NX,NY	- The number of gridpoints in the IX and JY directions, respectively. The dimensions of P(,) and FD(,). (INPUT)

EXAMPLE

```
DIMENSION P(NX,NY),FD(NX,NY)
DATA KFILDO/6/
...

CALL SMTH2X(KFILDO,P,FD,NX,NY)
```

The unit number for diagnostics is 6. The variable P(,) is smoothed twice with the simple 25-point smoother. FD(,) is provided as a work array.

OUTPUT:

None. KFILDO is provided in case output is added.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

SMTH25.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: MOS-2000 Library

SMTH3X

SMOOTHS A FIELD WITH A 25-POINT SMOOTHER THRICE

Harry R. Glahn
May 1, 1999

PURPOSE: A 2-dimensional array is smoothed with a 25-point smoother. Each resulting smoothed point is the average of itself and the 24 points in a square box around it (i.e., all points within a square 5 points on a side, with the point to be changed in the center, are averaged. A linear gradient is assumed for the outer rows and columns (two on each side). Some accommodation is made along the edges. The corner points are unchanged. Then the smoothed array is smoothed twice again in the same manner. This, then, gives a value at a gridpoint influence over a 13 X 13 area. No error checking is done and a status return is not provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL SMTH3X(KFILDO,P,FD,NX,NY)
```

KFILDO	- Unit number of output (print) file. (INPUT)
P(IX,JY)	- The input variable to smooth (IX=1,NX; JY=1,NY). Normally, these are values at gridpoints. On exit, P(,) contains the smoothed variable. (INPUT-OUTPUT)
FD(IX,JY)	- A work array (IX=1,NX; JY=1,NY). (INTERNAL)
NX,NY	- The number of gridpoints in the IX and JY directions, respectively. The dimensions of P(,) and FD(,). (INPUT)

EXAMPLE

```
DIMENSION P(NX,NY),FD(NX,NY)
DATA KFILDO/6/
...

CALL SMTH3X(KFILDO,P,FD,NX,NY)
```

The unit number for diagnostics is 6. The variable P(,) is smoothed twice with the simple 25-point smoother. FD(,) is provided as a work array.

OUTPUT:

None. KFILDO is provided in case output is added.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

SMTH25.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

RDSTNA

READS AND ALPHABETIZES ONE OR MORE STATION LISTS WITH GROUP NAMES

Harry R. Glahn
May 1, 1999

<u>PURPOSE</u>: To read and alphabetize one or more lists of station call letters with group names and associated information from a station directory. The lists read are composed of up to 8 characters per station. Each list is ended by the terminator '99999999' and reading is ended with an empty set. The stations in the lists returned <u>will be alphabetized</u> by group provided the station directory is alphabetical. Any duplicate stations found will be kept, and a diagnostic provided. Stations in a group list that are not in the directory will be put at the end of the group list.

<u>CALL AND EXPLANATION OF FORMAL PARAMETERS:</u>

```
CALL RDSTNA(KFILDO,IP4,IP5,KFILD,NEW,CCALL,CCALLD,NAME,NELEV,IWBAN,
            STALAT,STALON,IFOUND,NGP,GPNAME,NGPRUN,ND1,KGP,NSTA,IER)
```

<u>KFILDO</u> - Unit number of output (print) file. (INPUT)

<u>IP4</u> - When IP4 > 0, the station list (call letters only) will be written to Unit No. IP4. If there are input errors, the station list will be written to the default output file Unit No. KFILDO as well as to IP4. (INPUT)

<u>IP5</u> - When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5. (INPUT)

<u>KFILD</u>(J) - Unit number from which to read the station list (J=1) and station directory (J=2). It is assumed the files have been opened. KFILD(1) should not equal KFILD(2). (INPUT)

<u>NEW</u> - 1 when new ICAO station identifiers (call letters) are being used (columns 1-8 in the station directory);
0 when the old station call letters are being used (columns 10-17 in the station directory. (INPUT)

<u>CCALL</u>(K,J)- Array in which up to 6 sets (J=1,6) of 8 call letters per station are returned (K=1,NSTA). The lists of stations will be in alphabetical order <u>within groups</u>, provided the directory is in alphabetical order. CCALL(,1) will contain the call letters to be used as station or location identifiers in the calling program; the other 5 sets (J=2,6) are taken from the directory. The call letters returned in CCALL(,1) will be the ICAO station identifiers (columns 1-8) when NEW = 1, and will be the old call letters (columns 10-17) when NEW = 0. The other lists in CCALL(,J) always come from the directory,

and for J=3,6 come from columns 83-90, 92-99, 101-108, and 110-117 in that order. The list in CCALL(,2) also comes from the directory, being the ICAO call letters (columns 1-8) when NEW = 0 and the old call letters (columns 10-17) when NEW = 1. That is, it is intended that when CCALL(,1) contains the ICAO station identifiers, CCALL(,2) will contain the old call letters, and vice versa. (CHARACTER*8) (OUTPUT)

<u>CCALLD</u>(K) - Scratch array for holding unalphabetized list of stations (K=ND1). (CHARACTER*8) (INTERNAL)

<u>NAME</u>(K) - Names as read from the directory ordered according to CCALL(K,1) (K=1,NSTA). (CHARACTER*20) (OUTPUT)

<u>NELEV</u>(K) - Elevations of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>STALAT</u>(K) - Latitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>STALON</u>(K) - Longitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>IWBAN</u>(K) - WBAN numbers of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>IFOUND</u>(K) - Used to keep track of the stations found in the directory. (K=1,ND1). (INTERNAL)

<u>NGP</u>(K) - The sizes of KGP groups of stations (K=1,KGP), max of ND1. (OUTPUT)

<u>GPNAME</u>(K) - The names of the KGP groups (K=1,KGP), max of ND1. (CHARACTER*30) (OUTPUT)

<u>NGPRUN</u>(K) - Scratch array (K=1,ND1). (INTERNAL)

<u>ND1</u> - First dimension of CCALL(,), and dimension of NAME( ), NELEV( ), IWBAN( ), STALAT( ), STALON( ), IFOUND( ), and NGP( ). (INPUT)

<u>KGP</u> - The number of groups of stations read, and the number of values returned in NGP( ), max of ND1. (OUTPUT)

<u>NSTA</u> - The total number of elements (stations) returned in CCALL(,) and the associated arrays. (OUTPUT)

```
    IER       - Status return.
                 0 = Good return.
                20 = Error or EOF reading KFILD(1) by RDC.  Aborts.
                21 = Too many stations for dimension ND1 when reading by RDC.
                     Aborts.
                33 = Error reading station directory on Unit No. KFILD(2).
                     Aborts.
                34 = List to be kept too long for dimension ND1.  Aborts with
                     print.
                35 = One or more stations not found in the directory.  Normal
                     return.
                36 = One or more stations are duplicates.  Normal return.
                37 = Both IER = 35 and 36 above have occurred.  Normal return.
                (OUTPUT)
```

EXAMPLE

```
    PARAMETER (ND1=500)
    CHARACTER*8 CCALL(ND1,6),CCALLD(ND1)
    CHARACTER*20 NAME(ND1)
    CHARACTER*30 GPNAME(ND1)
    DIMENSION NELEV(ND1),IWBAN(ND1),STALAT(ND1),STALON(ND1),IFOUND(ND1),
   1          NGP(ND1),NGPRUN(ND1)
    DIMENSION KFILD(2),IP(25)
    DATA IP/25*0/
    DATA KFILDO/12/
    DATA KFILD/28,29/,
   1     NEW/1/
    ...

    CALL RDSTNA(KFILDO,IP(4),IP(5),KFILD,NEW,CCALL,CCALLD,NAME,NELEV,
   1            IWBAN,STALAT,STALON,IFOUND,NGP,GPNAME,NGPRUN,ND1,KGP,NSTA,IER)
```

The default output unit number is 12. The unit number to read the station list from is 28, and the station directory is on Unit No. 29. Since IP(4) and IP(5) = 0, the station lists will not be printed, unless there are fatal errors, in which case the lists and the errors found will be on Unit No. 12. The NSTA ICAO identifiers are returned in CCALL(,1), along with the associated information in other columns of CCALL(,) and in the arrays as indicated above. CCALL(,1) will be in alphabetical order by group, provided the directory is alphabetical. The first NGP(1) values will be for group 1, the second NGP(2) values will be for group 2, etc. for all KGP groups. In a similar manner, GPNAME( ) will contain the group names. Since the station lists will be the ICAO station identifiers (NEW = 1), CCALL(,2) will contain the old call letters. The maximum number of stations to be returned is ND1. The station lists are read by subroutine RDC with FORMAT(7(A8,1X)) until the terminator '99999999' is found. An empty set will terminate the reading. Note that the end of all groups must have two terminators, one for the last non-empty group and one for the empty group. Normally, each record would contain only one station just for ease of user manipulation, but up to 7 stations could be in one record. A description of the station dictionary can be found elsewhere.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. When IP4 > 0, the station lists (call letters only) will be written to Unit No. IP4. If there are station list input errors, the station lists will be written to the default output file Unit No. KFILDO as well as to IP4. When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are directory input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5.

RESTRICTIONS:

None other than stated above.

COMMENTS:

This routine performs essentially the same function as RDSTNL, except that the stations are alphabetized in RDSTNA and they are not in RDSTNL. If a station cannot be found in the directory, the last 5 columns of CCALL(,) will be blank. Note that all six columns of the directory are searched. That is, suppose a record in the directory contains 'KDEN   ', 'DEN   ', and 'DVX   ' in columns 1-8, 10-17, and 83-90, respectively. Either 'KDEN', 'DEN', or 'DVX' could be read by subroutine RDC, and KDEN would be returned in CCALL(,1). If KDEN changed to, say, 'KDUM' and the directory is updated, then 'KDUM' would be returned in CCALL(,1) and would be used in the calling program to identify output. This means that the directory record might be different on different data sets, but the MOS-2000 software should handle it, provided 'KDEN' is added to the entry in columns 92-99.

There are six routines which read a station list and return a list taken from either the first or second column of the directory, depending on the value of NEW; all six operate in the same way in this regard. Two routines deal with only a single list of stations and return the time zone. Four routines deal with multiple lists (groups) of stations. Three alphabetize the station list (by group) and three return the list in the order read. A group name is returned by two routines; this name is read immediately following the group list of stations with an A30 format. Their different characteristics are:

Routine	Single Station List	Multiple Station List	Group Name	Time Zone	Alphabetize
RDSTAL	X			X	
RDSTAD	X			X	X
RDSTGN		X			
RDSTGA		X			X
RDSTNL		X	X		
RDSTNA		X	X		X

RDSTNA is used in U850.

Many changes were made in October 2000 to assure that the station wanted from the directory was returned. Any directory entry can have up to 6 station identifiers, the leftmost being called the ICAO identifier, and next column being called the SAO call letters, although that nomenclature does not have to be followed. The rule followed is that the leftmost identifier that matches the identifier in the input list is the one used. That is, if a link I0V1 was in a particular record in a column 2-6 and was also in the first column, then the station in the first column is the one returned. The older version of RDSTNA did not assure this, and the result depended on the order in which the records were encountered.

Diagnostics were also improved, so that not only duplicate and missing stations are identified, but duplicates that may be caused by using a link for one of the stations and when a station is kept because of a link and the column (2 through 6) where the identifier occurred. In this context, SAO call letters are in Col. 2 when ICAO identifiers are being used (NEW = 1) and ICAO identifiers are in Col. 2 when SAO call letters are being used (NEW = 0).

In addition, station directory information is not printed to KFILDO except when IP5 = KFILDO or the error return is fatal. The table below shows the print structure as a function of the values of IER and IP5.

	IER =		
	0	33,34	35,36,37
IP5 = 0	---	KFILDO	---
IP5 = KFILDO	KFILDO	KFILDO	KFILDO
IP5 = IP5	IP5	KFILDO IP5	IP5

<u>NONSYSTEM ROUTINES USED</u>:

RDC

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

RDSTNL

READS ONE OR MORE STATION LISTS WITH GROUP NAMES

Harry R. Glahn
May 1, 1999

PURPOSE: To read one or more lists of station call letters with group names and associated information from a station directory. The lists read are composed of up to 8 characters per station. Each list is ended by the terminator '99999999' and reading is ended with an empty set. The stations in the lists returned will be in the order read. Any duplicate stations found will be kept, and a diagnostic provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL RDSTNL(KFILDO,IP4,IP5,KFILD,NEW,CCALL,NAME,NELEV,IWBAN,
            STALAT,STALON,IFOUND,NGP,GPNAME,ND1,KGP,NSTA,IER)
```

KFILDO - Unit number of output (print) file. (INPUT)

IP4 - When IP4 > 0, the station list (call letters only) will be written to Unit No. IP4. If there are input errors, the station list will be written to the default output file Unit No. KFILDO as well as to IP4. (INPUT)

IP5 - When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5. (INPUT)

KFILD(J) - Unit number from which to read the station list (J=1) and station directory (J=2). It is assumed the files have been opened. KFILD(1) should not equal KFILD(2). (INPUT)

NEW - 1 when new ICAO station identifiers (call letters) are being used (columns 1-8 in the station directory);
0 when the old station call letters are being used (columns 10-17 in the station directory. (INPUT)

CCALL(K,J)- Array in which up to 6 sets (J=1,6) of 8 call letters per station are returned (K=1,NSTA). CCALL(,1) will contain the call letters to be used as station or location identifiers in the calling program; the other 5 sets (J=2,6) are taken from the directory. The station list returned in CCALL(,1) will be in the order read. The call letters returned in CCALL(,1) will be the ICAO station identifiers (columns 1-8) when NEW = 1, and will be the old call letters (columns 10-17) when NEW = 0. The other lists in CCALL(,J) always come from the directory, and for J=3,6 come from columns 83-90, 92-99, 101-108, and 110-117 in that order. The list in CCALL(,2) also comes from the directory, being the ICAO call letters (columns 1-8) when NEW = 0 and the old call letters (columns

10-17) when NEW = 1. That is, it is intended that when CCALL(,1) contains the ICAO station identifiers, CCALL(,2) will contain the old call letters, and vice versa. (CHARACTER*8) (OUTPUT)

<u>NAME</u>(K) - Names as read from the directory ordered according to CCALL(K,1) (K=1,NSTA). (CHARACTER*20) (OUTPUT)

<u>NELEV</u>(K) - Elevations of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>STALAT</u>(K) - Latitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>STALON</u>(K) - Longitudes of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>IWBAN</u>(K) - WBAN numbers of stations from the directory ordered according to CCALL(K,1) (K=1,NSTA). (OUTPUT)

<u>IFOUND</u>(K) - Used to keep track of the stations found in the directory. (K=1,ND1). (INTERNAL)

<u>NGP</u>(K) - The sizes of KGP groups of stations (K=1,KGP), max of ND1. (OUTPUT)

<u>GPNAME</u>(K) - The names of the KGP groups (K=1,KGP), max of ND1. (CHARACTER*30) (OUTPUT)

<u>ND1</u> - First dimension of CCALL(,), and dimension of NAME( ), NELEV( ), IWBAN( ), STALAT( ), STALON( ), IFOUND( ), and NGP( ). (INPUT)

<u>KGP</u> - The number of groups of stations read, and the number of values returned in NGP( ), max of ND1. (OUTPUT)

<u>NSTA</u> - The total number of elements (stations) returned in CCALL(,) and the associated arrays. (OUTPUT)

<u>IER</u> - Status return.
 0 = Good return.
20 = Error or EOF reading KFILD(1) by RDC. Aborts.
21 = Too many stations for dimension ND1 when reading by RDC. Aborts.
33 = Error reading station directory on Unit No. KFILD(2). Aborts.
34 = List to be kept too long for dimension ND1. Aborts with print.
35 = One or more stations not found in the directory. Normal return.
36 = One or more stations are duplicates. Normal return.
37 = Both IER = 35 and 36 above have occurred. Normal return.
(OUTPUT)

EXAMPLE

```
    PARAMETER (ND1=500)
    CHARACTER*8 CCALL(ND1,6),CCALLD(ND1)
    CHARACTER*20 NAME(ND1)
    CHARACTER*30 GPNAME(ND1)
    DIMENSION NELEV(ND1),IWBAN(ND1),STALAT(ND1),STALON(ND1),IFOUND(ND1),
   1          NGP(ND1)
    DIMENSION KFILD(2),IP(25)
    DATA IP/25*0/
    DATA KFILDO/12/
    DATA KFILD/28,29/,
   1     NEW/1/
    ...

    CALL RDSTNL(KFILDO,IP(4),IP(5),KFILD,NEW,CCALL,NAME,NELEV,
   1            IWBAN,STALAT,STALON,IFOUND,NGP,GPNAME,ND1,KGP,NSTA,IER)
```

The default output unit number is 12. The unit number to read the station list from is 28, and the station directory is on Unit No. 29. Since IP(4) and IP(5) = 0, the station lists will not be printed, unless there are fatal errors, in which case the lists and the errors found will be on Unit No. 12. The NSTA ICAO identifiers are returned in CCALL(,1), along with the associated information in other columns of CCALL(,) and in the arrays as indicated above. The first NGP(1) values will be for group 1, the second NGP(2) values will be for group 2, etc. for all KGP groups. In a similar manner, GPNAME( ) will contain the group names. Since the station lists will be the ICAO station identifiers (NEW = 1), CCALL(,2) will contain the old call letters. The maximum number of stations to be returned is ND1. The station lists are read by subroutine RDC with FORMAT(7(A8,1X)) until the terminator '99999999' is found. An empty set will terminate the reading. Note that the end of all groups must have two terminators, one for the last non-empty group and one for the empty group. Normally, each record would contain only one station just for ease of user manipulation, but up to 7 stations could be in one record. A description of the station dictionary can be found elsewhere.

OUTPUT:

Diagnostic messages will be written to Unit No. KFILDO. When IP4 > 0, the station lists (call letters only) will be written to Unit No. IP4. If there are station list input errors, the station lists will be written to the default output file Unit No. KFILDO as well as to IP4. When IP5 > 0, the station directory information will be written to Unit No. IP5. If there are directory input errors, the same information will be written to the default output file Unit No. KFILDO as well as to IP5.

RESTRICTIONS:

None other than stated above.

COMMENTS:

This routine performs essentially the same function as RDSTNA, except that the stations are alphabetized in RDSTNA and they are not in RDSTNL. If a station cannot be found in the directory, the last 5 columns of CCALL(,) will be blank. Note that all six columns of the directory are searched. That is, suppose a record in the directory contains 'KDEN   ', 'DEN   ', and 'DVX   ' in columns 1-8, 10-17, and 83-90, respectively. Either 'KDEN', 'DEN', or 'DVX' could be read by subroutine RDC, and KDEN would be returned in CCALL(,1). If KDEN changed to, say, 'KDUM' and the directory is updated, then 'KDUM' would be returned in CCALL(,1) and would be used in the calling program to identify output. This means that the directory record might be different on different data sets, but the MOS-2000 software should handle it, provided 'KDEN' is added to the entry in columns 92-99.

There are six routines which read a station list and return a list taken from either the first or second column of the directory, depending on the value of NEW; all six operate in the same way in this regard. Two routines deal with only a single list of stations and return the time zone. Four routines deal with multiple lists (groups) of stations. Three alphabetize the station list (by group) and three return the list in the order read. A group name is returned by two routines; this name is read immediately following the group list of stations with an A30 format. Their different characteristics are:

Routine	Single Station List	Multiple Station List	Group Name	Time Zone	Alphabetize
RDSTAL	X			X	
RDSTAD	X			X	X
RDSTGN		X			
RDSTGA		X			X
RDSTNL		X	X		
RDSTNA		X	X		X

RDSTNL is used in U850.

Many changes were made in October 2000 to assure that the station wanted from the directory was returned. Any directory entry can have up to 6 station identifiers, the leftmost being called the ICAO identifier, and next column being called the SAO call letters, although that nomenclature does not have to be followed. The rule followed is that the leftmost identifier that matches the identifier in the input list is the one used. That is, if a link I0V1 was in a particular record in a column 2-6 and was also in the first column, then the station in the first column is the one returned. The older version of RDSTNL did not assure this, and the result depended on the order in which the records were encountered.

Diagnostics were also improved, so that not only duplicate and missing stations are identified, but duplicates that may be caused by using a link for one of the stations and when a station is kept because of a link and the column (2 through 6) where the identifier occurred. In this context, SAO call letters are in Col. 2 when ICAO identifiers are being used (NEW = 1) and ICAO identifiers are in Col. 2 when SAO call letters are being used (NEW = 0).

In addition, station directory information is not printed to KFILDO except when IP5 = KFILDO or the error return is fatal. The table below shows the print structure as a function of the values of IER and IP5.

	IER =		
	0	33,34	35,36,37
IP5 = 0	---	KFILDO	---
IP5 = KFILDO	KFILDO	KFILDO	KFILDO
IP5 = IP5	IP5	KFILDO IP5	IP5

<u>NONSYSTEM ROUTINES USED</u>:

RDC

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

CONST

RETRIEVES DATA FROM MOS-2000 EXTERNAL RANDOM ACCESS FILE

Harry R. Glahn
December 1, 1996

PURPOSE: To retrieve data from a MOS-2000 external random access file. CONST is called from U201's OPTION and vector programs' OPTX

RESTRICTIONS:

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

COMMENTS:

The call sequence is appropriate for both U201 and vector oriented routines (e.g., U600). All the facilities of the MOS-2000 internal storage system and external random access file system are used.

On the first use of a particular random access file, the "directory" record is read with ID(1) = 400001000, ID(2) = 0, ID(3) = 0, and ID(4) = 0. A correspondence table between this directory and the stations for which data are needed is formed and stored in the MOS-2000 internal storage system with IDs the same as above, except ID(2) = KFILX. On subsequent entries when the same file is needed, this correspondence table is used.

The subroutine COMPID is used to compute the IDs needed to fetch the data. For instance, the ID in the call sequence will be generic in the sense that the date/time of the constant on the file is not specified. When constants are furnished for each day (or month or season) the full ID must be computed. The structure shown in Chapter 4, Variable Identification, is followed.

IP(16) can be used to print certain information with the /D option.

See routine OPTION or OPTX for appropriate call sequence. Variables have usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH, GSTORE, RDTDLM, COMPID, UNPACK

LANGUAGE: FORTRAN 77

LOCATION: moslib

NORWND

MODIFIES WIND DIRECTION

Harry R. Glahn
May 1, 1999

PURPOSE: To retrieve from a MOS-2000 internal random access file a wind direction, or if it is not available, the u- and v-components necessary for its computation, and to set wind directions > 245 degrees to 360 - 345 degrees. A purpose for this is in the computation of a contingency table in U850 for wind direction, with one category being centered on north; setting winds just west of north to negative makes this possible. NORWND is called from OPTX.

RESTRICTIONS:

The call sequence is appropriate for vector oriented routines (e.g., U850). All the facilities of the MOS-2000 internal storage system are used.

The following CCC FFF for earth-oriented wind direction are accommodated:

004 203	Constant pressure
004 204	Constant height
004 205	Geostrophic
704 203	Observed
204 203	AEV wind
204 205	AEV significant wind

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have usual MOS-2000 definitions.

OPTX is called if direction needs to be computed from u and v components. For instance, CCCFFF = 004203 is provided to NORWND. If that wind direction is not available, OPTX will be called to compute it with subroutine DIRFUV.

NONSYSTEM ROUTINES USED:

GFETCH, OPTX

LANGUAGE: FORTRAN 77

LOCATION: moslib

TIMEPV

COMPUTES TIME DIFFERENCE, MEAN, MAX, OR MIN OF TWO VARIABLES

Harry R. Glahn
May 1, 1999

<u>PURPOSE</u>: To retrieve from the MOS-2000 internal random access file two variables of the same basic definition and compute their difference, mean, or the maximum or minimum value of the two. The operation is controlled by IDPARS(10) as follows:

1 = Mean ((1)+(2))/2
2 = Difference (1)-(2)
3 = Absolute difference (1)-(2)
4 = Max ((1),(2))
5 = Min ((1),(2))

The first varialbe (1) and the second variable (2) have the same definition, except the dates and projections (taus) are defined:

AEV		Forecasts	Observations
1st Variable	MDATE	= NDATE+ITAU	(Same)
	TAU	= IDPARS(12)	(Same)
2nd Variable	MDATE	= NDATE+ITAU	(Same)
	TAU	= IDPARS(12)-IDPARS(11)	(Same)

Non-AEV		Forecasts	Observations
1st Variable	MDATE	= NDATE+ITAU	(Same)
	TAU	= IDPARS(12)	(Same)
2nd Variable	MDATE	= NDATE+ITAU	MDATE = NDATE+ITAU-IDPARS(11)
	TAU	= IDPARS(12)-IDPARS(11)	TAU = IDPARS(12)

where MDATE is the date of the data, NDATE is the date being processed, ITAU has the usual definition (for lookahead), and TAU is the projection to use in IDPARS(12) when fetching data.

<u>RESTRICTIONS</u>:

None other than discussed above.

<u>COMMENTS</u>:

See routine OPTX for appropriate call sequence. Variables have usual MOS-2000 definitions.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH

<u>LANGUAGE</u>: FORTRAN 77

LOCATION: moslib

POPDIF

COMPUTES DIFFERENCE OF TWO POP FORECASTS

Harry R. Glahn
May 1, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file two PoP forecasts and compute their absolute difference. The resulting variable can be used as a "matching" variable in U850 to compute scores for only those cases when two PoP forecasts differed by some threshold. The following CCC FFF are accommodated:

CCC FFF	Difference of Variables
203 506	203 505 0 81 - 203 505 0 80 (local and AEV MOS)
203 507	203 505 0 81 - 203 505 0 06 (local and NGM MOS)
203 508	203 505 0 81 - 203 505 0 08 (local and AVN MOS)
203 509	203 505 0 81 - 203 505 0 07 (local and ETA MOS)

Other combinations can be added; POPDIF is hardcoded for the above computations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

ZRONEG

SETS NEGATIVE VALUES TO ZERO

Harry R. Glahn
May 1, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file a variable and to set the negative values to zero. This is appropriate for MOS wind speed forecasts. The following CCC FFF are accommodated:

CCC FFF	CCC FFF with negative values
204 212	204 210 (MRF mean speed)
204 213	204 211 (inflated speed)

Other combinations can be added; ZRONEG is hardcoded for the above computations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

MPSKTS

CONVERTS METERS PER SECOND TO KNOTS

Harry R. Glahn
May 1, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file a variable whose units are m/s and convert it to kt. This is appropriate for wind speed. The following CCC FFF are accommodated:

CCC FFF	CCC FFF of variable in m/s
004 030	004 010 (constant pressure u-wind component)
004 130	004 110 (constant pressure v-wind component)
004 230	004 210 (constant pressure wind speed)
004 031	004 011 (constant height u-wind component)
004 131	004 111 (constant height v-wind component)
004 231	004 211 (constant height wind speed)
004 032	004 012 (geostrophic u-wind component)
004 132	004 112 (geostrophic v-wind component)
004 232	004 212 (geostrophic wind speed)

Other combinations can be added; MPSKTS is hardcoded for the above computations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions. The conversion factor used is kt = 1.9424 m/s. Observations and MOS forecasts are likely already in kt; the model winds that can be used for comparison are in m/s.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

DIRFUV

COMPUTES WIND DIRECTION FROM COMPONENTS

Harry R. Glahn
May 1, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file two wind components and to compute the wind direction in degrees. The following CCC FFF are accommodated:

CCC FFF	CCC FFF
004 200	004 010 and 004 110 (constant pressure)
004 201	004 011 and 004 111 (constant height)
004 202	004 012 and 004 112 (geostrophic)
704 200	704 010 and 704 110 (observed)
204 200	204 010 and 204 110 (forecasts)

Other combinations can be added; DIRFUV is hardcoded for the above computations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions. The observed direction should be available on the archive, but only the components might be on the U201 output (for predictands), and the direction is needed. In this case, the computed direction may not match exactly the observed direction because of roundoff in packing. This routine is useful in U850.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

VERTX

COMPUTES VERTICAL DIFFERENCE, SUM, OR MEAN OF TWO VARIABLES

Harry R. Glahn
May 1, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file two variables of the same basic definition and compute their difference, sum or mean. The same variable definition (JD(1) the same), the same tau (IDPARS(12)), and the same run offset (IDPARS(9)) are used; the different levels are given by IDPARS(6) and IDPARS(7). The operation is controlled by IDPARS(5) as follows:

IDPARS(5)	Computation at Levels
1 = Difference	(IDPARS(7) - (IDPARS(6))
2 = Sum	(IDPARS(7) + (IDPARS(6))
3 = Mean	[(IDPARS(7) + (IDPARS(6))]/2

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

DIFFV

COMPUTES DIFFERENCE OF A FORECAST AND AN OBSERVATION

Harry R. Glahn
May 1, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file a forecast and an observation and compute their algebraic or absolute difference. The input variable CCC must be 28X or 29X. Then, the minuend (1) has a CCC = 20X and the subtrahend (2) CCC=70X. This capability can be used via U850 to compute scores only when the differences are greater than a threshold. Using a lookup table and the forecast projection and itau, the id for the observation is built by DIFFV. Although the id is built internally, the id provided in the observation section of U850 score group 6 serves to insure the variable is stored in the internal random access file and is available when needed. Therefore, the user must always provide the correct observation id. First, DIFFV builds the id for a non-AEV observation and attempts to retrieve the data. If successful, the algebraic or absolute value is computed and the subroutine is exited. If the "primary" non-AEV observation is not successfully retrieved, a new variable id is built for the corresponding "secondary" AEV observation, if one exists, and another attempt is made to retrieve data. No warning messages are written to KFILDO if the primary, non-AEV observations are retrieved successfully and used in the computation. Warning messages are written if secondary, AEV observations are used. Please follow the guidelines listed below under Comments to insure the correct ob is used and note the other restrictions.

The date and tau depend on whether the data are forecasts or observations. The date/time is computed for the forecast and observation as follows from information contained in the forecast id:

(1) JDATE = NDATE -RR + ITAU
(2) JDATE = NDATE -RR + ITAU + IDPARS(12)

where JDATE is the computed date/hour of the data (the forecast cycle for forecasts or forecast valid time for observations), NDATE is the basic date being processed, RR has the usual definition (for lookback), ITAU has the usual definition (for look-ahead), and IDPARS(12) is the forecast projection. For max/min temperature, the observation date/time may be adjusted by 6 hours to account for the difference between the AEV and non-AEV data labeling schemes.

<u>RESTRICTIONS</u>:

Unit conversion (e.g., Kelvin to Fahrenheit) and rounding of forecasts and/or observations must be done before the data are provided to U850 because subroutine diffv does not perform these actions. The subroutine only works for variables contained in the internal table. Gridded MOS forecasts interpolated to points are not handled in the current design.

<u>COMMENTS</u>:

The following situations are handled:

1) Non-AEV or AEV forecast compared to non-AEV observation: In this situation, the user should include the id for the non-AEV observation in the observation section of the variable list. The user should not include ids for AEV observations anywhere in the variable list. Under certain circumstances, such as when the desired non-AEV observation (the “primary” ob) is unavailable and the corresponding AEV (“secondary”) observation is available, subroutine DIFFV will attempt to retrieve the AEV observation. The user would know this occurred by a warning message printed to KFILDO indicating that DIFFV attempted to retrieve the secondary AEV observation. As long as the AEV observation id doesn’t appear in the variable list, the secondary AEV observation won’t be retrieved successfully. It will be impossible to inadvertently use the AEV observation when non-AEV obs were desired. The user will always know that an attempt was made to retrieve the AEV obs by a printed warning message. Only entire data records are used, and data are not substituted on a station-by-station basis for a given date.

2) AEV or non-AEV forecast compared to AEV observation: The id for the appropriate AEV observation should be provided in the observation section of the variable list to insure it is correctly stored in the internal random access file. Non-AEV observations should NEVER be provided on any input file and ids for non-AEV obs should NEVER be included in the variable list, even in the matching variable section. Doing so will result in the use of non-AEV obs in the computations and NO MESSAGE WILL BE WRITTEN TO KFILDO indicating the use of non-AEV obs. Note that DIFFV writes a warning message for each day/cycle, which can result in many messages written for large samples. This is considered necessary to insure that users know which observation was used in the computation.

See routine OPTX for appropriate call sequence. Variables have usual MOS-2000 definitions.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH, UPDAT

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

TRUNCP

TRUNCATES A SET OF PROBABILITIES TO 0 AND 1

David Rudack
June 15, 1999

PURPOSE: To truncate a set of probability values to the range 0 to 1. This could be a single probability (such as PoP) or a set of NCAT values (such as for cloud amount categories). The following CCC FFF are accommodated:

CCC FFF desired	CCC FFF computed from
208330	208450
208310	208440

Other combinations can be added. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None, other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions. This routine is useful in U710 and U910; see the U710 writeup for a discussion of a "series" of variables and the value NCAT.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

NMLPRB

NORMALIZES A SET OF PROBABILITIES TO LIE IN THE RANGE 0 TO 1

David Rudack
June 15, 1999

PURPOSE: To normalize a set of probability values to the range 0 to 1. The input would be a set of NCAT values (such as for cloud amount categories). The process is as follows: First, any negative values are set = 0. Then the probabilities in the set are totaled. And finally, the modified probabilities in the set are divided by the total. The computed set will then add to unity, and no value will be < 0 or > 1. The following CCC FFF are accommodated:

CCC FFF desired	CCC FFF computed from
208340	208313
208310	208314

Other combinations can be added. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions. This routine is useful in U710 and U910; see U710 writeup for a discussion of a "series" of variables and the value NCAT.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

INTOMM

CONVERTS INCHES TO MILLIMETERS

David E. Rudack
July 11, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file a variable whose units are inches and convert it to millimeters. This is appropriate for precipitation amount. The following CCC FFF are accommodated:

CCC FFF desired	CCC FFF computed from
203 244	203 240 (6-h precip amt)
203 344	203 340 (12-h precip amt)
203 444	203 440 (24-h precip amt)

Other combinations can be added; INTOMM is hardcoded for the above computations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions. The conversion factor used is mm = 25.41 in.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

FTOKEL

CONVERTS FAHRENHEIT TO KELVIN

David E. Rudack
July 11, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file a variable whose units are fahrenheit and convert it to Kelvin. This is appropriate for temperature. As examples only, the following CCC FFF are accommodated:

CCC FFF desired	CCC FFF computed from
202 004	202 000 (temp from equations)
202 009	202 005 (alternate temp from equations)
202 024	202 020 (temp checked with dew point)

Other combinations have been added and more can be added; FTOKEL is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions. The conversion factor used is
$K = (F-32)(5/9) + 273.15$ F.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

KWHTOJ

CONVERTS KILOWATT HOURS TO JOULES

David E. Rudack
July 11, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file a variable whose units are kilowatt hours and convert it to Joules. The following CCC FFF is accommodated:

CCC FFF desired	CCC FFF computed from
209 304	209 300

Other combinations can be added; KWHTOJ is hardcoded for the above computations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions. The conversion factor used is J = 3.6E+6 kwh.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

TMPCMP

CONSISTENCY CHECKS TEMPERATURE AND DEW POINT

David E. Rudack
July 11, 1999

<u>PURPOSE</u>: To retrieve from the MOS-2000 internal random access file a temperature and a dew point variable, both for the same projection; compare them; and set the returned value equal to the average of the two when the dew point is greater the temperature. The following CCC FFF are accommodated:

CCC FFF desired	CCC FFF computed from
202 020	202 000 and 203 000 (temperature)
203 020	203 000 and 202 000 (dew point)

Other combinations can be added; TMPCMP is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed.

<u>RESTRICTIONS</u>:

None other than discussed above.

<u>COMMENTS</u>:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

TEMPAV

AVERAGES TWO VARIABLES WITH THE SAME PROJECTION

David E. Rudack
July 11, 1999

PURPOSE: To retrieve from the MOS-2000 internal random access file two variables for the same projection and average them. The following CCC FFF are accommodated:

CCC FFF desired	CCC FFF computed from
202 010	202 000 and 202 005 (temperature)
203 010	203 000 and 203 005 (dew point)

Other combinations can be added; TEMPAV is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed. This routine is for projections where two forecasts are made for the same variable, and it is desired that the "operational" value be the average of the two.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

OBSPTYPE

CLASSIFIES PRESENT WEATHER INTO PRECIPITATION TYPE CATEGORIES

Rebecca Allen
April 2, 1999

PURPOSE: To classify station present weather reports into their respective precipitation type categories (see Comments). If the present weather report is missing or if no precipitation is reported, the precipitation type value is set to 9999. OBSPTYPE is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The MOS-2000 ID for precipitation type is:

```
708501000 000000000 000000000 000
```

RESTRICTIONS:

Due to the conversion from SAO observations to METAR observations in 1996, the present weather codes differ in the hourly data archive before and after December 1, 1996. While this subroutine is equipped to handle both numbering conventions, the user must not try to process dates that span this December 1, 1996, cutoff due to the configuration of the mass storage system. Therefore, to correctly process data across this threshold, the user must make one run containing dates up to and including November 30, 1996, and then start a new run for December 1, 1996, and beyond.

As with other U201 subroutines, if no data are available for the first processing date/time, the subroutine will not compute the precipitation type categories.

COMMENTS:

The present weather categories are as follows:

1 = freezing precipitation or anything in combination with freezing precipitation
2 = ice pellets in combination with non-freezing precipitation
3 = pure ice pellets
4 = pure snow
5 = rain and snow mixed or in combination
6 = pure rain
7 = thunderstorms

In the data available after December 1, 1996, the precipitation associated with the thunderstorms in category 7 is not distinguishable as rain or snow. Therefore, if a thunderstorm is reported along with any other precipitation group, the precipitation type is coded according to that other precipitation group.

Automated stations can report a weather descriptor UP for Unknown Precipitation. In this subroutine, we do not allow for UP because the precipitation type is not known for these reports. Therefore, if the user is using this subroutine as a method to determine a "yes or no, is it precipitating?" variable, any cases of UP will not be included.

For more information on the present weather codes and the hourly archive, see TDL Office Note 00-01.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77 with HP extensions.

LOCATION: File obsptype.f is in the U201 library /home21/tdllib/u201lib on the HP’s and /jdsk40/we/mos2000/u201lib on the CRAY.

OBSTCLD

COMPUTES TOTAL CLOUD COVERAGE

Mitchell Weiss
February 9, 1999

PURPOSE: To compute the total cloud coverage for station data based on surface observations and, in some, cases the GOES Satellite Cloud Product (see Comments). OBSTCLD is an almost exclusive routine of U201; therefore the call sequence is tailored for that use. Additional information about U201 can be found in the MOS-2000 software documentation guide. The values of total cloud are coded as shown in column 3 of Table 1. The MOS-2000 ID for total cloud coverage is:

```
708312000 000000000 000000000 000
```

RESTRICTIONS:

For period 1 processing (date/times 1995083123 and earlier; see Comments), the starting and ending processing date must precede September 1, 1995, 00Z.

For period 2 processing (date/times 1995090100 - 1996113023, the starting and ending processing dates must be within the period September 1, 1995, 00Z - November 30, 1996, 23Z.

For period 3 processing (date/times 1996120100 and later), the following restrictions apply:

1. The processing start date must be December 1, 1996, 00Z or later.

2. The record for station type must be available to properly categorize each station. If this record is missing, the value 9999. is returned.

3. For all station types, cloud coverage must be available for the first level, or the value 9999. is returned. Once cloud coverage has been observed for the first level, missing values occurring at higher levels will be assumed to be the limit of cloud coverage observations.

COMMENTS:

Total cloud coverage is estimated by determining the highest amount of observed cumulative cloud coverage for each station. Estimates of total cloud coverage are different for the following three time periods:

Period 1 (1995083123 and earlier)
Period 2 (1995090100 - 1996113023)
Period 3 (1996120100 and later)

For periods 1 and 2, previously calculated values of total cloud coverage are carried over from the older TDL archive described in Office Note 74-14 and converted to METAR coded values shown in column 3 of Table 1. Within OBSTCLD, period 2 values are complemented with GOES Satellite Cloud Product (SCP) cloud coverage. Complementing is necessary due to the introduction of ASOS observing systems at many stations during this period. Since the ASOS cloud observing instrument is incapable of observing clouds above 12,000 feet, SCP complementing is necessary. Due to the absence of “station type” information during this period, complementing is applied to all stations. Since SCP sampling was limited during period 2, if the SCP value for a given station is missing, the original (uncomplemented) total cloud coverage is returned. For period 3, estimates of total cloud coverage are generated by using multilayer cloud amount observations from the METAR reports, the SCP data, and the "station type" information from the METAR report. The technique of estimating cloud coverage is consistent with the method used for periods 1 and 2; however, enhancements to the technique were made because of the availability of the station type information. Moreover, if SCP data are unavailable during period 3, the total cloud coverage is set to missing.

The amount of total cloud coverage versus sky coverage is given in Table 1. A totally obscured sky is assumed to be completely overcast, but for a partially obscured sky (POB), only clouds observed above the POB layer are included in the calculation. If, for example, no clouds are observed above the POB layer, total cloud coverage equals the value for clear.

Estimates of total cloud coverage depend on the method of cloud observation for a given station. Essentially, cloud observations fall into three categories as follows:

1. Manual estimates using only observed cloud coverage from human observers.
2. ASOS estimates complemented with manual observations.
3. ASOS estimates complemented with SCP cloud coverage.

For all three categories, total cloud estimates start with surface observations. Manual estimates and ASOS estimates use up to six and three possible cloud layers, respectively. The highest cumulative cloud coverage for any level equals the total cloud coverage. If missing values occur at a given level, the cloud coverage from the previous lower level becomes the estimate. For category 1, the total cloud coverage is based only on surface observations and no other steps are taken. For categories 2 and 3, complementing ASOS cloud observations with additional cloud information is necessary because the ASOS cloud observing instrument cannot observe clouds above 12,000 feet. For category 2, estimates of cloud coverage are determined from ASOS cloud coverage observations which may or may not be augmented with manual observations. For category 3, the algorithm selects the highest ASOS and SCP cloud coverages, but only if both ASOS and SCP cloud coverage are available. Unlike period 2 complementing discussed earlier, if either the ASOS or SCP estimate is missing, the value of the cloud coverage is also set to missing.

Table 1. Relationship between sky coverage, cloud coverage, and coded value.

SKY COVERAGE	CLOUD COVERAGE	CODED VALUE
CLEAR (ASOS)	= 0	0
CLEAR (MANUAL)	= 0	1 CHANGED TO 0
FEW	> 0 - <= 2/8	2
SCATTERED	>= 3/8 - <= 4/8	3
BROKEN	>= 5/8 - <= 8/8	6
OVERCAST	= 8/8	8
TOTAL OBSCURATION	ASSUME 8/8	10

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77 with HP extensions.

LOCATION: File obstcld.f is in the U201 library /home21/tdllib/u201lib on the HP’s and /jdsk40/we/mos2000/u201lib on the CRAY.

SFCTCLD

COMPUTES TOTAL CLOUD COVER

Mitchell Weiss
February 9, 1999

PURPOSE: To compute the total cloud coverage for station data based only on surface observations. Total cloud coverage is estimated by determining the highest amount of observed cumulative cloud coverage for each station. SFCTCLD is an almost exclusive routine of U201; therefore, the call sequence is tailored for that use. Additional information about U201 can be found in the MOS-2000 software documentation. The values of total cloud are coded as shown in column 3 of Table 1. The MOS-2000 ID for this estimate of the total cloud coverage is:

```
708311000 000000000 000000000 000
```

RESTRICTIONS:

For period 1 processing (date/times 1996113023 and earlier), the starting and ending process date must precede November 30, 1996, 23Z.

For period 2 processing (date/times 1996120100 and later), the following restrictions apply:

1. The start date must be December 1, 1996, 00Z or later.

2. For all station types, cloud coverage must be available for the first level, or the value 9999. is returned. Once cloud coverage has been observed for the first level, missing values occurring at higher levels will be assumed to be the limit of cloud coverage observations.

COMMENTS:

Estimates of (surface) total cloud coverage are different for the following two periods:

Period 1 (1996113023 and earlier)
Period 2 (1996120100 and later).

For period 1, previously calculated values of total cloud coverage are carried over from the older TDL archive described in Office Note 74-14 and converted to METAR coded values shown in column 3 of Table 1. For period 2, estimates of surface total cloud coverage are generated by using multilayer cloud amounts provided in the METAR reports. The technique is consistent with the method used for period 1. The amount of total cloud coverage versus sky coverage is given in Table 1. A totally obscured sky is assumed to be completely overcast, but for a partially obscured sky (POB), only clouds observed above the POB layer are included in the calculation. If, for example, no clouds are observed above the POB layer, surface total cloud coverage is assumed to be clear. Estimates determined from stations with manual or ASOS observations will use up to six and three possible cloud layers, respectively. The highest cumulative cloud coverage for any level equals the surface total cloud coverage. If missing values occur at a given level, the cloud coverage from the previous lower level becomes the estimate.

Table 1. Relationship between sky coverage, cloud coverage, and coded value.

SKY COVERAGE	CLOUD COVERAGE	CODED VALUE
CLEAR (ASOS)	= 0	0
CLEAR (MANUAL)	= 0	1 CHANGED TO 0
FEW	> 0 - <= 2/8	2
SCATTERED	>= 3/8 - <= 4/8	3
BROKEN	>= 5/8 - <= 8/8	6
OVERCAST	= 8/8	8
TOTAL OBSCURATION	ASSUME 8/8	10

<u>NONSYSTEM ROUTINES USED:</u>

GFETCH

<u>LANGUAGE:</u> FORTRAN 77 with HP extensions.

<u>LOCATION:</u> File sfctcld.f is in the U201 library /home21/tdllib/u201lib on the HP’s and /jdsk40/we/mos2000/u201lib on the CRAY.

LCL

COMPUTES THE PRESSURE AND TEMPERATURE OF THE LIFTED CONDENSATION LEVEL

Kathryn K. Hughes
November 6, 1998

PURPOSE: To compute the pressure (PLCL) or temperature (TLCL) of the lifted condensation level on a grid from grids of (1) temperature, pressure, and dew point at any level. The grid containing the pressure values is internally created by using the TDL ID if the LCL is to be calculated on an isobaric surface. LCL is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The PLCL is returned in Pascals (mb *100). The TLCL is returned in Kelvin. The MOS-2000 ID's (CCCFFF) processed are:

003 160 = Pressure of the lifted condensation level;
003 161 = Temperature of the lifted condensation level;

RESTRICTIONS:

For computations on isohyetal or sigma surfaces, the subroutine expects pressure data from the NGM to be in hecto-Pascals. All other model pressure data are expected to be in Pascals, and are converted to hecto-Pascals within the DEWPT or SPECMIX (called if specific humidity is not available directly from the model) subroutines.

COMMENTS:

The calculation to compute the temperature and the pressure of the lifted condensation level was based on the formula presented by Bolton (MWR 1980, Volume 108). This is also the same computation used by GEMPAK.

```
TLCL = [1/(1/DEWPT-56)-LN(TMP/DEWPT)/800)]+56

PLCL = PRESS*(TLCL/TMP))**(1/KAPPA)
```

All temperatures are expressed in Kelvin and the initial pressure is given in millibars.

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, DEWPT, SPECMIX

LANGUAGE: FORTRAN 77 with HP extensions.

LOCATION: File lcl.f is in the U201 library /home21/tdllib/u201lib on the HP’s and /jdsk40/we/mos2000/u201lib on the CRAY.

THETAE

COMPUTES EQUIVALENT POTENTIAL TEMPERATURE

Kathryn K. Hughes
April 16, 1999

PURPOSE: To compute the equivalent potential temperature on an isobaric surface. The equivalent potential temperature is defined as the temperature which a parcel of air attains when it is lifted dry adiabatically to its lifted condensation level, then lifted psuedo-adiabatically until the parcel contains no moisture, then brought down adiabatically to 1000 mb. It is computed on a grid from grid point values of temperature, mixing ratio, and the temperature of the lifted condensation level. The pressure level from which the parcel is lifted is determined by the TDL ID (see below). THETAE is an almost exclusive routine for U201, and the call sequence is tailored for that use. The user should see the documentation for U201 for additional explanation. The equivalent potential temperature is returned in Kelvin. The MOS-2000 ID (CCCFFF) processed is:

003 130 = Equivalent Potential Temperature (Kelvin)

RESTRICTIONS:

This predictor is only valid when calculated from a standard level of pressure archived for the given model. The user supplies the starting pressure level of the parcel of air when requesting the THETAE predictor by indicating the pressure level in the second ID word (UUUU).

NONSYSTEM ROUTINES USED:

GFETCH, PRSID1, LCL, MIXRAT

LANGUAGE: FORTRAN 77 with HP extensions.

LOCATION: File thetae.f is in the U201 library /home21/tdllib/u201lib on the HP’s and /jdsk40/we/mos2000/u201lib on the CRAY.

APPTMP

CALCULATES APPARENT TEMPERATURE

David E. Rudack
October 1, 1999

PURPOSE: To calculate the apparent temperature (heat index) from the temperature and dew point in Fahrenheit. The following CCC FFF is accommodated:

CCC FFF desired	CCC FFF computed from
202 030	202 000 and 203 000

Other combinations can be added; APPTMP is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

DPDPRS

CALCULATES DEW POINT DEPRESSION

David E. Rudack
October 1, 1999

PURPOSE: To calculate the dew point depression from the temperature and dew point in Fahrenheit. The following CCC FFF is accommodated:

CCC FFF desired	CCC FFF computed from
202 040	202 000 and 203 000

Other combinations can be added; DPDPRS is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

MAXTEST

ENSURES CONSISTENCY BETWEEN 3-H TEMPERATURES AND MAX OR MIN FORECASTS

David E. Rudack
October 1, 1999

PURPOSE: To ensure consistency between the 3-h temperature forecasts and the maximum or minimum temperature forecasts for that period. The following CCC FFF are accommodated:

CCC FFF desired	CCC FFF computed from
202 120 (max)	202 000 and 202 001
202 220 (min)	202 000 and 202 011

Other combinations can be added; MAXTEST is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

MRFTEST

ENSURES CONSISTENCY BETWEEN SUCCESSIVE MAX OR MIN FORECAST TEMPERATURES

David E. Rudack
October 1, 1999

PURPOSE: To ensure that the maximum (minimum) temperature is consistent with the previous minimum (maximum) and the following minimum (maximum) temperatures. The following CCC FFF are accommodated:

CCC FFF desired	CCC FFF computed from
202 140 (max)	202 001 and 202 011
202 240 (min)	202 011 and 202 001

Other combinations can be added; MRFTEST is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

OPFCST

MAKES OPERATIONAL FORECASTS FROM PRIMARY AND BACKUP FORECASTS

David E. Rudack
October 1, 1999

<u>PURPOSE</u>: To determine operational forecasts from primary and backup forecasts. The following CCC FFF B DD is accommodated:

CCC FFF B DD desired	CCC FFF computed from
2XX XXX X 0X	2XX XXX X 1X and 2XX XXX X 2X

Another level of backup could be added to accommodate 2XX XXX X 3X

<u>RESTRICTIONS</u>:

None other than discussed above.

<u>COMMENTS</u>:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

RHUMID

CALCULATES RELATIVE HUMIDITY

David E. Rudack
October 1, 1999

PURPOSE: To calculate relative humidity from temperature and dew point in Fahrenheit. The following CCC FFF is accommodated:

CCC FFF desired	CCC FFF computed from
203 060	202 000 and 203 000

Other combinations can be added; RHUMID is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions. The calculation is RH = vapor pressure/saturation vapor pressure.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

SUNFCT

CALCULATES FRACTIONAL AMOUNT OF SUNSHINE

David E. Rudack
October 1, 1999

PURPOSE: To calculate the fractional amount of sunshine from forecast probabilities of clear, scattered, broken, and overcast sky conditions. The following CCC FFF is accommodated:

CCC FFF desired	CCC FFF computed from
209 300	208 340

RESTRICTIONS:

The latitude and longitude for each station must be accessible.

COMMENTS:

See routine OPT for appropriate call sequence. Variables have the usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

FETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

SOLENG

COMPUTES THE FRACTION OF POSSIBLE SOLAR ENERGY

David E. Rudack
October 1, 1999

PURPOSE: To calculate the fraction of possible solar energy from dew point and cloud probability forecasts. The following CCC FFF is accommodated:

CCC FFF desired	CCC FFF computed from
209 200	203 100 and 208 300

Other combinations can be added; SOLENG is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

The latitude and longitude for each station must be accessible.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

SOLAMT

COMPUTES SOLAR ENERGY

David E. Rudack
October 1, 1999

PURPOSE: To calculate the solar energy in KWH/M^2. The following CCC FFF is accommodated:

CCC FFF desired	CCC FFF computed from
209 210	209 200

RESTRICTIONS:

The fraction of solar energy (209 200) and the latitude for each station must be accessible; solar energy can be calculated by SOLENG.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

SUNAMT

COMPUTES AMOUNT OF SUNSHINE

David E. Rudack
October 1, 1999

PURPOSE: To calculate the amount of sunshine in hours. The following CCC FFF is accommodated:

CCC FFF desired	CCC FFF computed from
209 310	209 366

RESTRICTIONS:

The fraction of sunshine (209 366) and the latitude for each station must be accessible.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions. Sunshine amount is calculated by multiplying the fraction of sunshine by the astronomical hours of sun.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

SUNHRS

CALCULATES HOURS OF SUNSHINE AND EXTRATERRESTRIAL RADIATION

David E. Rudack
October 1, 1999

PURPOSE: To calculate the hours of astronomical sunshine and extraterrestrial radiation.

RESTRICTIONS:

The latitude of each station must be accessible.

COMMENTS:

Not called directly from OPTX.

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

WNDCHL

CALCULATES WIND CHILL

David E. Rudack
October 1, 1999

<u>PURPOSE</u>: To calculate the wind chill given the temperature in Fahrenheit and wind speed in knots. The following CCC FFF is accommodated:

CCC FFF desired	CCC FFF computed from
202 040	202 000 and 201 211

Other combinations can be added; WNDCHL is hardcoded for certain combinations. If combinations are added, the switching call in OPTX may have to be changed.

<u>RESTRICTIONS</u>:

None other than discussed above.

<u>COMMENTS</u>:

See routine OPTX for appropriate call sequence. Variables have the usual MOS-2000 definitions.

<u>NONSYSTEM ROUTINES USED</u>:

GFETCH

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: moslib

FCSTDF

COMPUTES DIFFERENCE BETWEEN LOCAL AND MOS FORECASTS

Harry R. Glahn
March 1, 2000

PURPOSE: To retrieve from the MOS-2000 internal random access file a local AEV forecast and a matching MOS forecast and compute the absolute or algebraic difference. The resulting variable can be used as a "matching" variable in U850 to compute scores for only those cases when two forecasts differ by some threshold. The following CCC FFF are accommodated:

CCC FFF	Difference of Variables
202 801 ^ DD	202 001 0 81 - 202 001 0 DD (daytime max temp)
202 811 ^ DD	202 011 0 81 - 202 011 0 DD (nighttime min temp)
203 805 ^ DD	203 505 0 81 - 203 505 0 DD (12-h pop)
204 810 ^ DD	204 210 0 81 - 204 211 0 DD (wind speed, MOS)
202 901 ^ DD	202 001 0 81 - 202 001 0 DD (daytime max temp)
202 911 ^ DD	202 011 0 81 - 202 011 0 DD (nighttime min temp)
203 905 ^ DD	203 505 0 81 - 203 505 0 DD (12-h pop)
204 900 ^ DD	204 200 0 81 - 204 200 0 DD (wind direction)
204 910 ^ DD	204 210 0 81 - 204 211 0 DD (wind speed, MOS)

Other combinations can be added if appropriate; FCSTDF is hardcoded for the above computations. If combinations are added, the switching call in OPTX may have to be changed.

RESTRICTIONS:

None other than discussed above.

COMMENTS:

See routine OPTX for appropriate call sequence. Variables have usual MOS-2000 definitions.

The matching variable with B = 9 can be used with the absolute differences to compute scores with U850 where the absolute differences between the local and MOS forecasts differ by $\geq$ the amount indicated by a positive threshold in ID(4). For instance,

```
ID(1) = 202901980
ID(2) = 0
ID(3) = 24
ID(4) = 399001000
```

can be used to compute scores for 24-h max temperature when the absolute difference between the local and NGM MOS is $\geq$ 4 deg.

The matching variable with B = 9 can be used with the algebraic differences to compute scores with U850 where the positive differences between the local and MOS forecasts differ by $\geq$ the amount indicated by a positive threshold in ID(4). For instance,

```
ID(1) = 202801980
ID(2) = 0
ID(3) = 24
ID(4) = 399001000
```

can be used to compute scores for 24-h max temperature when the difference between the local and NGM MOS is $\geq$ +4 deg.

The matching variable with B = 8 can be used with the algebraic differences to compute scores with U850 where the negative differences between the local and MOS forecasts differ by $\geq$ the amount indicated by a negative threshold in ID(4). For instance,

```
ID(1) = 202801880
ID(2) = 0
ID(3) = 24
ID(4) = -399001000
```

can be used to compute scores for 24-h max temperature when the difference between the local and NGM MOS is $\geq$ -4 deg.

As a special feature, the difference between wind directions is always made $\leq$ 180 deg,

NONSYSTEM ROUTINES USED:

GFETCH

LANGUAGE: FORTRAN 77

LOCATION: moslib

INTRPC

FINDS VALUE IN GRID CLOSEST TO POINT LOCATIONS

Harry R. Glahn
June 10, 2002

PURPOSE: Finds and returns the value at a gridpoint closest to point locations. The point must be within 1/2 gridlength outside the grid boundaries to have a non-missing value. No error checking is done and a status return is not provided.

CALL AND EXPLANATION OF FORMAL PARAMETERS:

```
CALL INTRPC(KFILDO,P,NX,NY,DIR,ND1,NSTA,SDATA)
```

KFILDO	- Unit number of output (print) file. (INPUT)
P(IX,JY)	- The input grid to get a value from (IX=1,NX; JY=1,NY). (INPUT)
NX,NY	- The number of gridpoints in the IX and JY directions, respectively. The dimensions of P(,). (INPUT)
DIR(K,J)	- The IX (J=1) and JY (J=2) positions on the grid for station K (K=1,NSTA). (INPUT)
ND1	- The maximum number of stations whose locations are in DIR(,). This is the first dimension of DIR(,). (INPUT)
NSTA	- The number of stations whose locations are in DIR(,) and whose closest values are to be returned. It is used as the dimension of SDATA( ). (INPUT)
SDATA(K)	- The closest values for all stations are returned in SDATA( ) (K=1,NSTA). (OUTPUT)

EXAMPLE

```
DIMENSION P(NX,NY)
DIMENSION DIR(ND1,2),SDATA(ND1)
DATA KFILDO/6/
...

CALL INTRPC(KFILDO,P,NX,NY,DIR,ND1,NSTA,SDATA)
```

The unit number for diagnostics is 6. NSTA values are to be returned in SDATA( ) from the grid in P(,). The dimensions of the grid are NX and NY in the IX and JY directions, respectively. The values returned in SDATA( ) correspond in order to the station locations given in DIR(K,J) in terms of the IX (J=1) and JY (J=2) locations.

OUTPUT:

None. KFILDO is provided in case output is added.

RESTRICTIONS:

NSTA must be LE ND1. Each value in DIR(,) should, of course, indicate a position within the grid or reasonably close (one half gridlength) to it or a "missing" value of 9999 will be returned.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: MOS-2000 Library

TSLOP

CALCULATES WEST OR SOUTH TERRAIN SLOPE

Harry R. Glahn
July 20, 2002

PURPOSE: To compute the west or south terrain slope. Given a terrain grid on either a polar stereographic, Lambert, or Mercator map projection, the earth-oriented slope components are computed at station (point) locations. The ID CCCFFFs are:

409xyz

where

x represents the map projection
- 3 = Lambert,
- 5 = polar stereographic, and
- 7 = Mercator,

y represents the grid length

```
    0 = 1/4 Bedient
    1 = 1/8    "
    2 = 1/16   "
    3 = 1/32   "
    4 = 1/64   "
    5 = 1/128  "
    6 = 1/256  "
```

z designates the slope component
- 1 = west (positive)
- 2 = south (positive)

The y is used only to differentiate the grids for access and the plain language for printing and packing; the actual grid length in the packed record is what is used for computations. These designators can actually be correct for a polar stereographic map at 60 degrees N, but will be approximate for other situations.

RESTRICTIONS:

A large value of ND5 will probably be required for the terrain grid.

It is assumed TSLOP will be used with U201, and PRED21 and PRED22 have been modified to assist its use (see comments below). It should work with programs similar to U201, and the operation will be the same for Day 1. However, the computations will also have to be made for each following day.

COMMENTS:

The terrain grid is accessed through CONSTG and is fully defined in TDLPACK.

Since a 5-km grid over a large area requires a large array, ND5 in the driver must be set large enough to accommodate the grid, but ND2X3 need not necessarily be that large. The computations are all made at stations, so the "computational" arrays can be smaller. This also reduces the computations, because the number of stations is usually much smaller than the number of gridpoints. Even so, reading a 1 to 5 meg grid takes time. The terrain slope does not vary by date/time, so it can be made only once per run. To accomplish this, PRED21 stores the slope components (by station) when computed into the Internal Storage System, then just before PRED22 calls OPTION a check is made of whether the CCC = 409 and if it is the record is retrieved from internal storage. In this way, OPTION and TSLOP are not entered except on the first date/time through PRED21.

The slope components can be accessed by MOS-2000 vector-type programs (e.g., U600) by either packing them for each date/time in U201, or putting them into the External Random Access Storage in vector form. While writing "constants" to a file for each date/time may seem distasteful, it probably doesn't matter that much, and would eliminate the step of putting them into the External Storage System for every station that may be used in a, say, U600, run.

The smoothing and interpolation options in the 4th ID word (I and S) are available for use. That is, one can smooth the terrain before computation, and then interpolate as desired. Note that smoothing pertains to the terrain field, not to components after computation. This is all done within TSLOP. The call sequence is appropriate for grid-oriented programs such as U201, and is much like that used for other routines called from OPTION.

NONSYSTEM ROUTINES USED:

CONSTG, GRCOMB, SMTH5, SMTH9, SMTH25, SMTH2X, SMTH3X, TSLCM

LANGUAGE: FORTRAN 77

LOCATION: u201lib

UPSLOP

CALCULATES UPSLOPE MOTION

Harry R. Glahn
July 20, 2002

<u>PURPOSE</u>: To use a terrain field and a wind field to compute upslope motion. The terrain field and U- and V-wind components can be on a polar stereographic, Lambert conformal, or Mercator map projection, and do not have to be with the same mesh length or cover the same area. However, the map projection and orientation must be the same. The ID CCCFFFs are:

```
    005xyz

where

x represents the map projection
    3 = Lambert,
    5 = polar stereographic, and
    7 = Mercator,

y represents the grid length
    0 = 1/4 Bedient
    1 = 1/8    "
    2 = 1/16   "
    3 = 1/32   "
    4 = 1/64   "
    5 = 1/128  "
    6 = 1/256  "

z designates the type of wind surface (isobaric or constant height)
    and the smoothing to be done on the wind components before
    computation
    0 = isobaric wind, no smoothing
    1 =      "       , 5-point smoothing
    2 =      "       , 9-point smoothing
    3 =      "       , 25-point smoothing
    4 =      "       , 25-point smoothing twice
    5 = constant height wind, no smoothing
    6 =         "           , 5-point smoothing
    7 =         "           , 9-point smoothing
    8 =         "           , 25-point smoothing
    9 =         "           , 25-point smoothing twice
```

The y is used only to differentiate the grids for access and the plain language for printing and packing; the actual grid length in the packed record is what is used for computations. These designators can actually be correct for a polar stereographic map at 60 degrees N, but will be approximate for other situations.

RESTRICTIONS:

A large value of ND5 will probably be required for the terrain grid.

The terrain and wind grids must pertain to the same map projection and orientation.

COMMENTS:

The terrain grid is accessed through CONSTG and is fully defined in TDLPACK.

Since a 5-km grid over a large area requires a large array, ND5 in the driver must be set large enough to accommodate the grid, but ND2X3 need not necessarily be that large. The computations are all made at stations, so the "computational" arrays can be smaller. This also reduces the computations, because the number of stations is usually much smaller than the number of gridpoints. Even so, reading a 1 to 5 meg grid takes time. The terrain slope does not vary by date/time, so it can be computed only once per run. To accomplish this, UPSLOP stores the slope magnitude and direction (by station) when computed into the Internal Storage System. Before UPSLOP computes the slope components, a check is made of whether they already exist in storage and if they do, they are retrieved and not recomputed. In this way, the terrain need be read only once for any given advecting field. The magnitude and direction of the slope are stored in internal storage with the same ID as the upslope ID, except the "G" (last digit in the 4th word) is used as "1" for the magnitude and as "2" for the direction. This means the components will have to be computed for each variable ID, but only once. Note that the smoothing parameter "S" (penultimate digit in the 4th ID word) can be different for different variables and pertains to smoothing the terrain.

The upslope component can be accessed by MOS-2000 vector-type programs (e.g., U600) by either packing it for each date/time in U201, or putting it into the External Random Access Storage System in vector form. While writing "constants" to a file for each date/time may seem distasteful, it probably doesn't matter that much, and would eliminate the step of putting them into the External Storage System for every station that may be used in a, say, U600, run.

The smoothing and interpolation options in the 4th ID word (I and S) are available for use. That is, one can smooth the terrain before computation, and then interpolate as desired. Note that smoothing pertains to the terrain field, not to components after computation. But also note that the interpolation parameter "I" pertains to interpolation into the wind fields; interpolation into the terrain field is with a special linear interpolation over a single grid length. This is all done within UPSLOP. While the smoothing parameter "S" pertains to the terrain field, the wind components can also be smoothed and that is controlled by the z in FFz as outlined in the Purpose above.

It should not matter what map projection these computations are made on. That is, the computations could be made on a polar stereographic projection for development, and a Lambert used for operations. However, for the full resolution terrain, these computations will not be identical. Testing will show how much different they may be.

The call sequence is appropriate for grid-oriented programs such as U201, and is much like that used for other routines called from OPTION.

NONSYSTEM ROUTINES USED:

CONSTG, GRCOMB, SMTH5, SMTH9, SMTH25, SMTH2X, SMTH3X, UPSLCM, GFETCH, GSTORE, INTRP, INTRPA, INTRPB, MCMPAS, PSMAPS, LMMAPS

LANGUAGE: FORTRAN 77

LOCATION: u201lib

STRATVAR

CALCULATES A STRATIFICATION VARIABLE

David E. Rudack
February 1, 2004

<u>PURPOSE</u>: To calculate a stratification variable that was defined in U602. This subroutine was written so that the user can generate forecasts for stations or regions that contain one or more stratified predictors in the regression equation. STRATVAR is called from OPTX.

<u>RESTRICTIONS:</u>

The call sequence is appropriate for vector oriented routines (e.g., U700). All the facilities of the MOS-2000 internal storage system are used.

The user must first enter the stratification four-word MOS-2000 ID in the matrix table, "**ITAB( ),**" defined in STRATVAR. The next entry in **ITAB( )** is the "base" variable which may or may not be a binary value. The last entry rows in the matrix table **ITAB( )** consist of the "auxiliary" MOS-2000 IDs which must be binaries. The maximum number of entries for the auxiliary variables is four. The user has control over the total number of **ITAB( )** matrix tables by setting the parameter "NDIM" in STRATVAR.

The final entry in STRATVAR is the matrix table named "**THRESHOLD( )".** This matrix table contains the floating point values for each fourth word threshold value in the matrix table **ITAB( ).** The user has control over the total number of **THRESHOLD( )** matrix tables by setting the parameter "NDIM" in STRATVAR. For a more detailed discussion of the definitions of the base and auxiliary variables please see the U602 writeup.

<u>COMMENTS:</u>

See subroutine OPTX for the appropriate call sequence. If a particular base or auxiliary variable needs to be calculated, STRATVAR will call OPTX to calculate the variable needed.

<u>NONSYSTEM ROUTINES USED:</u>

GFETCH, PRSID2, BASICP, OPTX, TIMPR, BINARY, UPDAT

<u>LANGUAGE:</u> FORTRAN 77

<u>LOCATION:</u> moslib

CONST1

RETRIEVES DATA FROM MOS-2000 EXTERNAL RANDOM ACCESS FILE

Harry R. Glahn
December 1, 1996

PURPOSE: To retrieve data from a MOS-2000 external random access file. CONST1 is called from U201's OPTION and other U201-type programs which process both gridded as well as vector data. CONST1 is used to provide the MOS-2000 programs with constants, forecasts, or other data stored in a MOS-2000 external random access file. The data read from the constant file can be valid over a grid or at stations. The CCC in the MOS-2000 identifier determines the unit number:

CCC Range	Unit No.	Use	Type of Data
400-499	44	INPUT	GRIDPOINT
400-499	45	INPUT	VECTOR
500-699	46	INPUT	VECTOR
800-899	47	INPUT	VECTOR
200-299	48	INPUT	VECTOR
200-299	49	OUTPUT/INPUT	VECTOR

Subroutine CONST is called, in lieu of CONST1, by programs that process only vector data. For constants in the CCC = 4xx series, interpolation may be done to get the data wanted.

RESTRICTIONS:

The unit numbers for the random access files must be in the range 44 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
44	400-499	"True" gridded constants (terrain ht., etc.)
45	400-499	"True" station constants (rel. freq., etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

COMMENTS:

The call sequence is appropriate for U201 and related programs. All the facilities of the MOS-2000 internal storage system and external random access file system are used.

On the first use of a particular vector-oriented random access file, the "directory" record is read with ID(1) = 400001000, ID(2) = 0, ID(3) = 0, and ID(4) = 0. A correspondence table between this directory and the stations for which data are needed is formed and stored in the MOS-2000 internal storage system with IDs the same as above, except ID(2) = KFILX.

On subsequent entries when the same file is needed, this correspondence table is used.

The subroutine COMPID is used to compute the IDs needed to fetch the data. For instance, the ID in the call sequence will be generic in the sense that the date/time of the constant on the file is not specified. When constants are furnished for each day (or month or season), the full ID must be computed. The structure shown in Chapter 4, Variable Identification, is followed.

IP(16) can be used to print certain information with the /D option.

See routine OPTION for appropriate call sequence. Variables have usual MOS-2000 definitions.

NONSYSTEM ROUTINES USED:

CONSTG, CONST, GRCOMB

LANGUAGE: FORTRAN 77

LOCATION: With u201 in the u201lib file system.

CONSTG

RETRIEVES GRIDDED CONSTANT DATA FROM MOS-2000 EXTERNAL RANDOM ACCESS FILE

Harry R. Glahn
September 1, 2000

PURPOSE: To retrieve data from the gridded MOS-2000 external random access file for use in grid-oriented programs. CONSTG is called by CONST1 in U201-type programs when IDPARS(1) $\geq$ 400 and $\leq$ 499 and when the input random access file number is set to 44. Note that CCC values between 400 and 499 inclusive are reserved for data in these random access files.

RESTRICTIONS:

The call sequence is appropriate for U201 and other "gridded" programs. In these programs, the file unit number will be 44, and the file name is in KFILRA( ).

COMMENTS:

Unlike CONSTX, which is used to read constants appropriate for vector data, at this time CONSTG can not distinguish between gridded constant records valid for different dates.

NONSYSTEM ROUTINES USED:

RDTDLM, UNPACK

LANGUAGE: FORTRAN 77

LOCATION: With U201 in the u201lib file system.

INTRPD

INTERPOLATES TO STATION BY USING VALUE OF CLOSEST GRIDPOINT

Harry R. Glahn
July 1, 2003

PURPOSE: Returns a value at each station from the closest gridpoint to that station within the 4-gridpoint box surrounding the station. The gridpoint is such that: 1) the value is not missing (that is, not equal to 9999.); 2) the value is not defined as unusable by a grid mask; and 3) the terrain elevation of the gridpoint is within a specified distance of the terrain elevation of the station. This limit is fixed within INTRPD by a DATA statement, and is currently set to 500 feet for each variable. This subroutine is for U201 for use in the NDFD verification task and is restricted accordingly.

RESTRICTIONS:

The terrain and mask grids are read by subroutine CONST1 from an external random access file with unit number 44. The cccfff of the terrain identifier is defined internally in INTRPD. In like manner, the cccfff of the mask identifier is also defined internally in subroutine INTRPD. The "y" in the terrain and mask fff's (that is, the second f in the fff string) designates the grid mesh in fractions of bedients. This value is determined by subroutine ACTNOM and is inserted into the identifier before reading the random access file. A value of 4 represents a nominal 5-km grid. If the id being processed by INTRPD is not found in the list of recognized id’s encoded in INTRPD, then a "generic" set of terrain and mask grids is used.

The "acceptable" difference between the elevation of the gridpoint to be used and the station terrain is also set internally within INTRPD. This value can be different, according to the variables being processed. The generic value of 500 feet is currently used.

INTRPD is currently set up to process the following variables:

202 050	NDFD surface temp in degrees F
202 130	NDFD daytime max in degrees F
202 230	NDFD nighttime min in degrees F
203 030	NDFD dewpoint in degrees F
204 230	NDFD surface wind direction in degrees
204 330	NDFD surface wind speed in kt
203 520	NDFD POP

NONSYSTEM ROUTINES USED:

CONST1, ACTNOM, SAMEGR.

LANGUAGE: FORTRAN 77

LOCATION: With u201 in the u201lib.

TGINTRP

TEMPORALLY INTERPOLATES GRIDDED MODEL DATA TO STATIONS USING QUADRATIC OR LINEAR INTERPOLATION

David E. Rudack
January 1, 2007

<u>PURPOSE</u>:

To temporally interpolate gridded model data to stations at “off time” projections (e.g., a 7- or 8-h forecast) using a quadratic or linear interpolation. When performing a temporal interpolation for discontinuous variables (e.g., precipitation amount), a linear interpolation is used. TGINTRP is currently designed to use model data at 6-h hour increments.

The algorithm can be summarized as follows: An “on time” (e.g., a 12- or 18-h forecast) gridded record is retrieved or computed and is then spatially interpolated (usually using a bilinear interpolation) to stations. If a discontinuous variable is processed, a different spatial interpolation is used (intrp.f). This process is repeated three additional times for continuous variables and one additional time for discontinuous variables. Once the data have been spatially interpolated to stations, a quadratic interpolation (for continuous variables) or a linear interpolation (for discontinuous variables) is performed. For quadratic interpolation, two projections on either side of the desired projection are needed. If only one projection on one side is available, linear interpolation is done.

The following MOS-2000 ID (CCCFFF and ISG) can be used:

00XXXX – 9XX

<u>RESTRICTIONS:</u>

Currently, TGINTRP is designed to temporally interpolate model data available at 6-h increments to a finer temporal scale. If the user would like to use data available at increments other than 6 hours, a modification of the code will be necessary.

The only discontinuous spatial interpolation used by TGINTRP is for precipitation amount. Consequently, if the user would like to temporally interpolate a discontinuous variable that requires an alternate type of interpolation (such as nearest neighbor (intrpc.f)), TGINTRP would need to be modified.

<u>COMMENTS:</u>

See subroutine OPTION for the appropriate call sequence.

NONSYSTEM ROUTINES USED:

GFETCH, OPTN2, PRSID1, INTRP, AND INTRPB

LANGUAGE: FORTRAN 77

LOCATION: u201lib

U520

Quality Controls Hourly Observations

Rebecca L. Allen
February 3, 1999

<u>PURPOSE</u>: U520 reads the hourly observation data from the hourly surface observation tables, quality controls the data, and then packs up the quality controlled data in TDLPACK format for archiving. The quality control portion of the program serves to eliminate or correct a wide range of errors in the data. Subroutines of U520 check for errors in temperatures, dewpoints, cloud observations, precipitation amounts, atmospheric pressure, wind data, visibility and present weather reports. The input to U520 is a file containing any number of the hourly surface tables concatenated together, a list of all the stations that the user wishes to process hourly observations for, the station dictionary, and the dates of the data to be processed. The output of U520 consists of a file containing all of the quality controlled data in TDLPACK format. In addition, this program also produces a list of all instances where an observation was changed, either to a correct value, or to insert a missing value, and a file containing all of the dates for which observations were quality controlled.

<u>CONTROL FILE INPUT</u>: 'U520.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,12I3) <u>**Output Identification**</u>

This record is here to be consistent with other MOS-2000 programs, and only serves to identify the run on the default print unit.

<u>IPINIT</u> - 4 characters, usually a user's initials plus a run number. This is written as the first record on the default output file. The run number allows multiple runs of U520 to be uniquely identified. (Character*4)

<u>IP</u>(J) - These values are here to be consistent with other MOS-2000 programs, but serve no other purpose (J=1,12).

Record Type 2 - Format (I10/I10) <u>**Dates to Process**</u>

This record contains the dates and hours for which quality controlled data are requested. Note that each value is on a separate line.

<u>LDATE</u> - The date and hour at which to start the quality control process (see Comments).

<u>MDATE</u> - The last date and hour that will be quality controlled (see Comments).

Record Type 3 - Format (I3,4X,A60) <u>**Input Hourly Observations**</u>

This record (plus terminator record) identifies the data set that contains the hourly surface observations that will be quality controlled.

<u>NUNIT</u> - Unit number for the file containing the hourly surface observation tables.

<u>HRYNAM</u> - Name of file where the hourly tables reside. (CHARACTER*60)

Record Type 4 - Format (I3,4X,A60) <u>**Station and Location Files**</u>

This pair of records (plus the terminator record) identifies the files from which the station information is obtained. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record is 2.

<u>KFILD</u>(J) - Unit number for the file containing the list of stations for which data will be quality controlled (J=1), and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2).

<u>DIRNAM</u>(J) - Name of file matching KFILD(J). (CHARACTER*60)

Record Type 5 - Format (I3,4X,A60) <u>**Output Hourly Observations**</u>

This record identifies the output file that will contain the TDLPACKed hourly observations. Records are read until the terminator, 99, is reached. Maximum number of records sans the terminator = 1. This file will be opened as 'NEW'.

<u>IPUNIT</u> - Unit number for the output file containing the TDLPACK hourly observations.

<u>PKNAM</u> - Name of the file where this output resides. (CHARACTER*60)

Record Type 6 - Format (I3,4X,A60) <u>**Error Listing**</u>

This record identifies the output file that will contain a listing of all the errors and corrections made to the hourly data by the quality control subroutines in U520. Records are read until the terminator, 99, is reached. Maximum number of records sans the terminator = 1. This file will be opened as 'NEW'.

<u>IEUNIT</u> - Unit number for the output file containing the error listing.

<u>QCERRNAM</u> - Name of the file where this output resides. (CHARACTER*60)

Record Type 7 - Format (I3,4X,A60) <u>**Hours Processed**</u>

This record identifies the output file that will contain a list of each hour that was processed by the program. The statement will tell whether or not data were available for each hour. Records are read until the terminator, 99, is reached. Maximum number of records sans the terminator = 1. This file will be opened as 'NEW'.

<u>IFUNIT</u> - Unit number for the output file containing the dates that have been qc'd.

<u>QCDNAM</u> - Name of the file where this output resides. (CHARACTER*60)

CONTROL FILE INPUT: (Name read from U520.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) Station List

This group of records identifies the stations for which quality controlled hourly observations are desired. These stations must be in alphabetical order.

CCALL(K) - ICAO station identifiers designators) of stations for which quality controlled observations are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. The terminator is '99999999'. The maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U520.CN) (Unit = KFILD(2))

Record Type 1 - Format Station Directory
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which quality controlled observations are desired. The ICAO station identifiers read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. The directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTAD to give information about the stations.

CCALLD(K,J) - Call letters (or other character location designator) of stations (J=1). As stated above, these call letters are matched with those in the station list.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited. No terminator is used.

DATA INPUT:

The file on unit NUNIT (see Record Type 3) contains the hourly surface observation tables. The data for each hour are contained in a separate table with the first two records being the table heading with the date, and the column titles, and the last line being a trailer record. The user must concatenate the hourly files for all hours needed for processing (see Comments).

The data are read in the following manner: Once the code has read the first two records of the table, containing the table title and column headers, each record containing an observation is read as a character string 256 characters long. After a line is read in, it is then broken up into the various weather elements. The program will continue in this manner until the trailer record for that hour is read. Then it will move on to the next hour if it is within the requested time frame.

DATA OUTPUT:

There are three output files generated by U520. Each is labeled with the name of the file, followed by the dates that are processed as an extension. For example, the packed data for the month of January, 1998, would be 'pkdata.1998010100_1998013123'.

A. QUALITY CONTROLLED HOURLY DATA (pkdata.xxxxx) (Unit = IPUNIT)

After the observations have been quality controlled, they are put into TDLPACK format. This one file will contain all of the observations for the period of time that the user specified be processed.

B. LIST OF ERRORS (qcerr.xxxxx) (Unit = IEUNIT)

Every time a piece of data is changed, an error message is written to this output file. Each line has the following structure: station ID, date of observation, type of station, error number, incorrect data value, corrected data value, and a plain language explanation of what the error was. For a complete description of the errors and associated error codes, see Chapter 13, part B, or TDL Office Note 00-01.

C. LIST OF DATES PROCESSED (qcd.xxxxx) (Unit = IFUNIT)

This file contains a listing of all the dates for which data were quality controlled, and whether or not data were available for each hour.

EXAMPLE CONTROL FILE: 'U520.CN'

An example exists as file 'U520.CN' in directory /user/g06/u520/run on the IBM. The easiest way to set up a run is to take an existing control file and modify it.

<u>RESTRICTIONS</u>:

Some array sizes are set through PARAMETER statements in the driver, DRU520. Formats and other guidelines in other MOS-2000 documents are followed.

There are two time periods that the user must keep in mind when running U520. The first is the time period for which quality controlled data are desired. This time period must be 24 hours, or a multiple of 24 hours, and that period MUST start at 00Z and end at 23Z. These times must be set in the control file, U520.CN, as the variables LDATE and MDATE.

The second time period of importance is the period of hourly data that must be supplied as input for the program. See the comments section to determine what hourly tables are required.

Both the input station list on Unit KFILD(1) and the stations in the station directory on Unit KFILD(2) must be in alphabetical order.

<u>COMMENTS</u>:

U520 is structured so that it quality controls the data in synoptic chunks of 6 hours within a 24-h 0000 to 2300 period. This is necessary to make sure that variables such as 6-hr max/min and precipitation totals are consistent with the observations through that period. U520 reads in data that it will need to process the first day, quality controls those data 6 hours at a time, and then packs up the data for that entire day. Then it moves on to the next day. Because of this set-up, and because of the need to quality control 24-h chunks, there must be in the input data tables hourly observations from 25 hours before the period to process, and 2 hours after the end of the period to process. This is, an extra 27 hours outside of the desired period is re-quired. For example, if the user wants to quality control the data for the month of June 1998, he/she must include in the input file all of the hourly observation tables from 2300 UTC on May 30, 1998 to 0100 UTC on July 1, 1998. The user will set the beginning and ending dates to be processed in the control file as LDATE and MDATE respectively. The program will then compute the actual hours for which data is needed in order to process that specified time. A message will be printed to KFILDO reminding the user of the exact hourly tables that must be included in the input file NUNIT.

The three output files are opened in RDSNAM with the status ‘NEW’. It is therefore assumed that they do not already exist. If they do, the program will not overwrite the old files, but will instead create files using just the unit numbers from the control file (on the HP’s it will be ftn.XX, on the IBM it will be fort.XX)

For a more complete description of the hourly data or the quality control checks performed in U520, refer to Chapter 13 of TDL Office Note 00-01.

<u>SETTING UP THE DRIVER DRU520</u>

The preparation of the driver for U520 should be relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's and IBM) or 64 for a 64-bit machine (e.g., the CRAY)
ND1 -	Maximum number of stations in the station list. At this time, given the current number of observing sites, 3500 is a reasonable value.
ND5 -	Dimension of IPACK( ), IWORK( ), and DATA( ).
ND7 -	The size of IS0( ), IS1( ), IS2( ), and IS4( ). Use 54 or greater.
ND8 -	The maximum number of date/times that can be used. Keep in mind that for a full month of data (31 days), this value must be at least 771.

<u>NON SYSTEM ROUTINES USED</u>:

INT520, UPDAT, DATGEN, RDSNAM, RDSTAD, RDC, PACK1D, PACK, PKBG, PKMS00, PACKGP, TRAIL, WRITEP, PUTCHAR

<u>LANGUAGE</u>: Fortran 90.

<u>LOCATION</u>: /user/g06/u520 on the IBM.y

U530

DEVELOPS DICTIONARY FOR OBJECTIVE ANALYSIS OF HOURLY DATA

Harry R. Glahn
Jung-Sun Im
January 18, 2011

PURPOSE: U530 reads one or more files of hourly data, with or without an initial dictionary, and prepares a dictionary for use by U155. The input hourly data are in the format prepared by a modification to HRLYTBL (modification necessary to include the elevation). The dictionary format is explained in TDL ON 00-1, Chapter 10. U530 is controlled by a control file 'U530.CN'.

Whenever the station dictionary is updated, U155 preprocessors such as U174, U178, U179, etc. should be run. In GOBS T/Td/Wind tasks, U178 requires manually created override radius of influence (R) (U178's preprocessor) for water stations. To create the override R, the distance to the land grid from water observation point is helpful. An ASCII file which consists of station identifier, station type, latitude, longitude, elevations, land/water flag, and distance to the closest land grid point from a water station is output.

In keeping with other MOS-2000 programs, U530 is written as a subroutine and uses a driver DRU530 so that dimensions of many variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U530 (actually, subroutine) for every application; these parameters are called NDx and will be referred to in this writeup. U530 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver.

CONTROL FILE INPUT: 'U530.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU530.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U530" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U530 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U201HRG240'. Do not use a blank for one of the characters. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U530 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). If another file unit number is designated the same as an IP( ) value, that IP( ) is set to KFILDO. This is ASCII output, generally for diagnostic purposes. Values have been defined as indicated below for values of J:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU530 data statement = 12. Changing the default unit number allows multiple runs of U530 or other programs within the same directory without overwriting.

(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).

(3) = The dates in IDATE( ). These will be the same as in (2) above. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3). (This is not useful information; see Record Type 4.)

(4) = The list of stations that are (1) to be retained from the input dictionary (if one exists) without modification (labeled "PERMANENT LIST," and (2) stations that are not to be in the output dictionary (labeled "DISCARD LIST").

(5) = The input station dictionary information. Stations will be in alphabetical order. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5). This includes 8-character station identifiers, station name, WBAN number, data quality flags and report type, latitude, longitude, elevation, time zone, and substitute station(s). When IFIRST = 0 (see Record Type 3), this file will be blank except for the time stamp.

(7) = A listing of all stations in the input dictionary in order (if one exists), then all stations found on the hourly files not in the dictionary, including stations with the same call letters but different elevation, latitude, or longitude when these metadata are outside the tolerances (see Record Type 3, XLTOL and XELTOL). The listing includes the initial dictionary date/time or the first date/time a new station was encountered, the last time the station was encountered with the same metadata, the number of matching metadata reports, the number of non-matching metadata reports (labeled "BAD") the number of times since the metadata were changed, and the location in the list of a link if it exists (a link is a station with the same call letters, but different metadata). Link will be labeled with the ICAO identifier, but with the 8th character being a "1" or "2" indicating its link status. Station names can exist only from the input dictionary.

(8) = Much the same data as in IP(7) but includes the station type explicitly (NTYPE, see Record Type 3) and the disposition of the station for the output dictionary. Some diagnostics may not be correct if this is not activated. Also, at the beginning of the file, a line is printed each time a link is replaced with another; this indicates the volatility of the station metadata.

(9) = A listing of call letters, name, latitude, longitude, link location in the list, and KPTOSS( ), the latter being the disposition to be made of the station:
0 = normal, keep,
1 = in permanent list, keep (see Record Type 8),
2 = in discard list, discard (see Record Type 9),
3 = old station, not kept because of DEL (see Record Type 3),
4 = new station, not added because of NADD (see Record Type 3).

(15) = A listing of grid IX, JY positions of all stations, including links, before making final dictionary. These positions conform to the grid read from the External Random Access file (see Record Type 6 and Record Type 3).
(16) = Results of defining the type of station (0, 3, 9) by using the grid from the External Random Access file.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format(4(I10/),2(F10.0/),(2(I10/),I10) Control Parameters

This record contains several control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing what the value is for which variable.

IFIRST -	Indicates whether (=1) or not (=0) an initial dictionary is provided. When IFIRST = 0, a dictionary need not be provided in Record Type 7; it doesn’t hurt if it is, but it will not be used.
JSTOP -	The total number of errors that will be tolerated before the program halts.
NDEL -	The number of hourly reports necessary to retain a station in the input dictionary. When NDEL = 0, a station with no reports will be kept; when NDEL = -1, a station with no reports will be tossed.
NADD -	The number of hourly reports for a particular station necessary to add it into the dictionary, if it is not in the incoming dictionary.
XLTOL -	The tolerance of the latitude or longitude (in degrees) before the station is considered inconsistent with one with the same call letters.
XELTOL -	The tolerance of the elevation (in feet) before the elevation of the station is considered inconsistent with one with the same call letters.
LLCHG -	The number of consistent latitudes and longitudes necessary to change an initial dictionary entry.
NECHG -	The number of consistent elevations necessary to change an initial dictionary entry.
NPROJ -	The map projection used in this run (3 = Lambert, 5 = polar stereographic, 7 = Mercator).
ORIENT -	The orientation of the map–the “vertical” longitude.
XLAT -	The latitude where the nominal grid length of the grid used is correct.
NX -	The size of the grid in the IX (east-west) direction.
NY -	The size of the grid in the JY (north-south) direction..

ALAT - The latitude of the lower left corner point of the grid.

ALON - The longitude of the lower left corner point of the grid.

MESH - The nominal grid length of the grid.

L3264W - The number of bits in a word- normally 32.

Record Type 4 - Format (I3,4XA60) Date List File

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM. The minimum number read is one, indicating only one date to be processed for checkout. The maximum number is two, the last being the last date/time to be processed. The negative spanning function is not used. All date/times between the two read here will be processed if the hourly data are available (see Record Type 12).

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI (=5), DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) Date List

This group of records determines the beginning and ending date/times for which input is to be read. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list. Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is 2 even though ND8 may be greater than that. On completion of reading this record type, J=1,NUMIN.

Record Type 6 - Format (I3,4XA60) Random Access Files

This group of records identifies the random access data sets from which constant or other data are read or written. Records are read until the terminator KFILRA( ) = 99 is reached. Maximum number of records, sans the terminator record, = 6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMRA. If no data are needed involving a constant file, only the terminator is necessary. If the output is to a random access file, use a KFILRA( ) = 42. (U530 uses land/water and distance grids from this file.)

KFILRA(J) - Unit number for the random access constant file (J=1,NUMRA). Unit numbers must be in the range 42 to 49; see "Restrictions" for more information.

RACESS(J) - Name of file corresponding to KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 7 - Format (I3,4XA60) Station and Location Files

This pair of records (plus the terminator record) identifies the file(s) from which the station (or location) information is obtained (the input dictionary). Records are read until the terminator KFILD( ) = 99 is reached. Expected number of records, sans the terminator record, = 2. This Record Type 7 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

<u>KFILD</u>(J) - Unit number for the file containing the station call letters which are to be used in U530 and the station dictionary which holds the latitudes, longitudes, WBAN numbers, elevations, quality control flags (in the block/station number field), and names for each possible station (J=2). This dictionary file should be entered twice with the same unit number (e.g., KFILD(1) = KFILD(2)).

<u>DIRNAM</u>(J) - Name of file matching KFILD(J). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) <u>Permanent Stations File</u>

This record (plus the terminator record) identifies the file from which the stations that are not to be changed from the input dictionary is taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM.

<u>KFILP</u> - Unit number for the permanent stations file.

<u>FLNAMP</u> - Name of file matching KFILP. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) <u>Discard Stations File</u>

This record (plus the terminator record) identifies the file from which the stations that are not to be included in the output dictionary is taken. Records are read until the terminator KFILT = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 9 is read by subroutine RDSNAM.

<u>KFILT</u> - Unit number for the discard stations file.

<u>FLNAMT</u> - Name of file matching KFILT. (CHARACTER*60)

Record Type 10 - Format (I3,4XA60) <u>Output Station Call Letters File</u>

This record (plus the terminator record) identifies the file to which the stations that are in the completed dictionary will be written. Records are read until the terminator KFILOL = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 10 is read by subroutine RDSNAM.

<u>KFILOL</u> - Unit number for the output station list.

<u>FLNAMOL</u> - Name of file matching KFILOL. (CHARACTER*60)

Record Type 11 - Format (I3,4XA60) <u>Output Station Dictionary File</u>

This record (plus the terminator record) identifies the file to which the completed dictionary will be written. Records are read until the terminator KFILOD = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

<u>KFILOD</u> - Unit number for the output station dictionary.

<u>FLNAMOD</u> - Name of file matching KFILOD. (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) Output Water Station Distance File

This record (plus the terminator record) identifies the file to which the distances for water stations will be written (supplementary ASCII output file to be used for computing U178's override R). Records are read until the terminator KFILOS = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILOS - Unit number for the output water station distance.

FLNAMOS - Name of file matching KFILOS. (CHARACTER*60)

Record Type 13 - Format (I3,4XA60) Hourly Data Input

This record (plus the terminator record) identifies the first file from which the hourly data will be read. Records are read until the terminator KFILAS = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILAS - Unit number for the input hourly data.

FLNAMA - Name of file matching KFILAS. (CHARACTER*60)

CONTROL FILE INPUT: (Name read from 'U530.CN') (Unit = KFILDT)

Record Type 1 - Format (7I10) Date List

When the dates are not provided in file 'U530.CN' in Record Type 5, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU530), this file is omitted.

IDATE(J) - Initial date list. Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Expected number of dates, sans terminator, is 2 even though ND8 may be greater than that. On completion of reading this record type, J=1,NUMIN.

DATA INPUT:

A. HOURLY DATA

The input data are hourly data files produced by a modification of HRLYTBL, the primary reason for the modification was to capture the elevation which was not in the original code. This list must be in alphabetical order, which is the way the program prepares it.

B. PERMANENT STATION LIST

A list of stations that are to be kept without change, no matter what the hourly data indicate, is read on Unit No. KFILP and file name KFNAMP (See Record Type 8). A station in this list will always be included in the output station dictionary, provided it is in the input dictionary. This list must be in alphabetical order. This list can be empty.

C. DISCARD STATION LIST

A list of stations that are to be discarded, no matter what the hourly data indicate, is read on Unit No. KFILT with file name KFNAMT (See Record Type 9). This list applies to both the stations in the incoming dictionary and new stations found in the hourly data. These stations will not be put into the output dictionary. If a station occurs in this list that is also in the permanent station list (see B above), the permanent status trumps the discard status. This list must be in alphabetical order. This list can be empty.

D. LAND/WATER GRID

A grid meeting the specifications of the grid characteristics in the U155.CN file, Record Type 3. Each gridpoint has a value of 0 for ocean, 3 for inland water, or 9 for land.

E. DISTANCE GRID

A grid meeting the specifications of the grid characteristics in the U155.CN file, Record Type 3. Each water gridpoint has a value of the distance to the closest land gridpoint (unit is Km). For land gridpoints, 9999 is assigned.

DATA OUTPUT:

Besides the diagnostics on Units KFILDO and IP( ), there are three forms of output. Those are the (modified) dictionary, a station list of the stations in the dictionary, and the distances to the closest land grid from water stations in the dictionary.

A. STATION LIST

This is an ASCII file of the ICAO identifiers (also called call letters) that are in the output dictionary. The format is A8 with two terminators “99999999" consistent with other station lists. The file name is contained in Data Type 10.

B. STATION DICTIONARY

The dictionary for the station list in A above in the format described in TLD ON 00 1, Chapter 10. The file name is defined in Data Type 11.

C. WATER STATION DISTANCE

This is an ASCII file which consists of station identifier, station type, latitude, longitude, elevations, land/water flag, and distance to the closest land gridpoint from a water station (unit is grid length). Whenever the station dictionary is updated, U155 preprocessors such as U174, U178, U179, etc. should be run. In GOBS T/Td/Wind tasks, U178 requires manually created override R (U178’s preprocessor) for water stations. To create the override R, the distance to the land gridpoint from a water point is helpful. The file name is defined in Data Type 12.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file U530.CN, Record Type 1 and the definition of KFILDO in the driver DRU530. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control without jeopardizing data. Some errors cause a hard stop when proceeding is useless.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes.

IBM workstations accommodate optionally compiled statements with a "D" in Column 1.

A unit number of zero should never be read in the .CN control file. If a file is not needed, just omit it from the control file; this will be interpreted in U530 and called routines as 0. The routine RDSDAM does not recognize a zero unit number.

The unit number from the random access files should be 44 for U530, and is used only for reading (42 and 43 should work, also).

The two files to provide the incoming dictionary and station list (IFRST = 1) must be the same file on the same unit number.

Each hour of the hourly data files must be on a separate file with specific header information, as placed there by the preprocessing program. In addition, a terminator 'ZZZZZZZZ' must follow the last station ICAO number, with only blanks following it (see Appendix I for more detail).

Testing has been only on a 32-bit IBM workstation.

If there is an incoming dictionary, the last station in it is always kept.

COMMENTS

U530 runs on the IBM workstation or at NCEP.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with JSTOP (see Record Type 3). At the end of the run, the number of "errors" is printed. Note that it is not always known what is an error or what might be sometimes expected. Some errors are such that proceeding would not be productive, and U530 stops with a diagnostic.

Not all information for the dictionary is known for new stations (those not in the incoming dictionary). For new stations, NTYPE( ) is set to 3, indicating an automated station; this may or may not be correct. For all stations, new and for old that do not have a non-zero time zone, one is calculated based on the longitude of the station. This is approximate, because the time zone boundaries are not always along a longitude line. The western extremities of the zones in degrees W. longitude are as follows, and are consistent with the analysis subroutine U405A (only CONUS is accommodated):

Atlantic	67.0
Eastern	86.5
Central	104.0
Mountain	115.0

A few characteristics of other MOS programs are carried over in order to use subroutines unchanged. For instance, the last 5 columns in CCALL(,) are not used.

Appendix II explains the assignment of types within the dictionary field "NBLOCK."

SETTING UP THE DRIVER DRU530

The preparation of the driver for a particular U530 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

ND1 - The maximum number of stations that will be dealt with, including stations in the input dictionary and all different ones found in the hourly data.

ND2 - ND2*ND3 is the maximum size of the grid that can be dealt with. ND2 and ND3 are set separately to highlight the possible dimension of the grids. However, in the called routines, the size is only limited by the product, not each dimension individually.

ND3 - See ND2.

ND5 - Dimension of IPACK( ), IWORK( ), and DATA( ). Set to ND2*ND3.

ND7 - The size of ISO( ), IS1( ), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be read. A value of 2 is sufficient.

ND12 - The maximum number of External Random Access file units and names that can be read.

The "startup" control file 'U530.CN' is opened in DRU530, so that this aspect of U530 can be rather easily changed without recompiling or having separate version of the main (sub)routine U530. An example exists as file 'U530.CN' in directory '/mdl/data/lamp2k/staging/u530' on halo.

NONSYSTEM ROUTINES USED

Use the load line in '/mdl/data/lamp2k/staging/u530' on halo.

LANGUAGE: FORTRAN 90 with some IBM extensions.

LOCATION: U530lib. The driver is in '/mdl/data/lamp2k/staging/u530' on halo.

APPENDIX I

A station dictionary contains information about a set of stations; the dictionary is described in TDL ON 00-1, Chapter 10. The stations are identified by ICAO identifiers, may times called call letters in keeping with past identifiers. A dictionary has two primary purposes:

(1) The information in the dictionary, such as station name, location, etc., can be matched with a list of stations being used with a MOS-2000 program. This means that only a list of stations may have to be furnished to a program (such as U201), and all the other station information can come from a somewhat universal dictionary. The latitudes and longitudes are relied on to be correct in case they are needed by the program.

(2) When real-time data are being dealt with, some metadata may be different from what is expected. A report can be in error, or a station location can have been changed and report with the same call letters, but with different metadata. It is usually not known (in real-time) whether the unexpected metadata are indeed correct or not. In such a case, the incoming metadata can be matched with the dictionary information, and if the station is not included, or the metadata do not match within specified tolerances, then the data cannot be safely used.

Preparation of a dictionary takes much painstaking work, verifying locations, etc. Different uses (e.g., MOS, EKDMOS, and LAMP) may require slightly different dictionaries, but uniformity should be the rule, and departures taken only if the "standard" is not usable unchanged. It does not matter if more stations are included in the dictionary than the stations used in the particular program being run.

Analysis of real time observations requires about double the number of stations needed in most MOS/LAMP programs. The dictionary is not long-term or stable- stations come, go, and their metadata change. (Metadata are defined here to be latitude, longitude, and elevation.) So, to keep pace with the mildly volatile station set, a dictionary has to be prepared at some interval; too frequent incurs too much hand work in quality control, too infrequent means the dictionary gets out of date, and considerable data may not be used in the analyses.

In real-time hourly data files, a set of call letters may come in fairly regularly, but with different metadata. Quite likely, in this case, there are really two stations in different locations, but reporting with the same call letters. MOS-2000 programs can=t deal with two stations with the same call letters but at different locations. In this case, the dictionary can contain the call letters and one set of metadata. This will constitute a validated location for that station, and any incoming hourly data whose metadata do (don't) match the dictionary will (will not) be used. This will, in effect, use one of the duplicate stations, but not the other.

U530 would normally start with an initial dictionary wherein the latitudes and longitudes may be to four decimal places. Hourly data are read, report by report, as prepared by a slightly modified version of HRLYTBL. (The modification was necessary because the original did not extract the station elevation from the hourly data and write it out.) U530 requires that the station dictionary be alphabetically ordered, and also that the hourly data on each hourly file be ordered. Both of these requirements are satisfied by the upstream programs. It is expected that each hourly file will be a separate file with the ZZZZZZZZ terminator. However, it is also expected that nothing follow the ZZZZZZZZ, unless the format matches the expected observation. This is because the records are read with FORTRAN reads with a specific format, and if the format does not match that expected, an error will occur. This is in distinction to U520 which reads the report in ASCII and parses the parts internally; U530 lets FORTRAN reads do the work. In earlier days when errors might occasionally occur in format, the ASCII reading might be necessary, but it seems an error of this kind would be rare. If an error is detected, reading that particular file is just discontinued, with an error report. The hourly files differ only by the date/time, and U530 expects a file each hour and will open and close files as necessary. Missing files cause no trouble.

As hourly data are read, a tolerance for latitudes and longitudes (in degrees) and another for elevation (in feet) are used to determine whether the metadata match an existing station. If they do, statistics such as the number of reports with those metadata are updated. If they do not, the station with the new metadata is saved, and linked back to the original station. Two such stations are saved when the metadata of the second one added did not match either the one in the dictionary or the linked one. If different metadata are encountered that do not match either of the three now saved, a decision is made to replace one of the links with the newer report. This decision is based on the last reporting dates. In the end, the station metadata put into the output file will depend on how recently the metadata were reported and the number of reports required to change the metadata (NADD, LLCHG, NECHG from Data Type 3). There may be a few situations when the station seems unstable (has come in with perhaps several different locations) where the decision is made to toss it.

New stations are inserted into the original list so that the stations are output in alphabetical order, as is required for a dictionary. (It is noted that lower case letters are higher in order than capitals.) The date/time in the dictionary will be the original if no change has been made, or will be the date of the first report that matches the output if the metadata were changed; this conforms to the intent of the date/time in the dictionary. The incoming dictionary may have blanks for the date, indicating the station has been in existence for a long time; they will remain such unless the metadata are changed.

Many of the stations are associated with MOS/LAMP forecasts, and the locations cannot be changed. Some of these are to 4 decimal places in precision. The tolerance for location, XLTOL, should be set large enough so that the incoming data reported to 2 decimal places will not cause U530 to think the metadata have changed. A value of 0.011 will account for rounding to two decimal places (and is only a 1.1 km difference in either direction), but will not account for one place accuracy which some stations, notably moored buoys, report. However, a set of stations can be provided (see Data Type 8) that will not be changed no matter what the hourly data indicate. These considerations retain the 4-place accuracy when justified. (Perhaps XLTOL should be hardwired at 0.011 for most stations and 0.51 for buoys and other stations that report to one place accuracy. Actually, this is already done for moored buoys.)

The variable NDEL (see Data Type 3) can be used to discard stations if less than NDEL reports are found in the incoming data set. Also, a set of stations can be provided (see Data Type 9) that will indicate the stations to be discarded whether or not reports for that station are found. This applies to either the initial dictionary or newly found stations. That is, it may be known that a particular station is unreliable, and it can be discarded. If NDEL = 0, a station with no reports will be kept; if NDEL = -1, a station with no reports will be tossed. When stations have no reports, they will be labeled in IP(8) "NO REPORTS." When not zero, but less than NDEL, they will be labeled "FEW REPORTS."

A station in the incoming dictionary has a type 0, 3, 6, or 9 associated with it, indicating, respectively, ocean, inland water, both inland water and land, and land. An hourly station type does not indicate unambiguously which type of station this is, although MBUO should indicate water (but does not differentiate between ocean and inland water). The grid read from the External Random Access file indicates the gridpoints are of type 0, 3, or 9. If the station has been designated as "permanent" (see Data Input, Type B), the type is left as is. Otherwise, the land/water file is used to designate the station type from the closest gridpoint. These designations may not always be the best for use in the U155 analysis, and some hand work may be necessary close to a land/water boundary. Appendix II explains the assignment of types within the dictionary field "NBLOCK."

Just before U530 calls the subroutine (dctout) that prepares the output dictionary, the file prepared with stations and links, contains just that: each station has an entry and possibly one or two links. The original entry has been modified as necessary to now contain the information to go into the dictionary. The output on IP(8) indicates the station's status and its links.

The output ASCII dictionary is on file name FLNAMOD read in Data Type No. 11. A list of stations in the usual A8 format with two terminators is on a file with name read as FLNAMOL read in Data Type No. 10.

U530 will operate if no initial dictionary is used. This seems to not be the best choice, however. With the incoming data in alphabetical order and a substantial incoming dictionary, Halo processes 6 months of data in about 15 minutes; with no input dictionary, the run took 1 hour, 16 minutes; this substantial difference in time is because the incoming dictionary entries are in order, but the "new" stations are not, so more searching time is required for each station read. In any case, the processing time is satisfactory.

Decisions are made in U530 to require as little hand work as possible. Many decisions are clear cut (given the rules established), and IP(8) shows what the decision was in each case. There are a few times when the metadata are not changed, but questioned. These could be looked at for possible change if the differences in elevations are substantial, but probably not worth much effort, especially for the elevation.

My recommendation is that a dictionary of hourly data be prepared once a year, and that dictionary be archived and labeled as to the time period so that a rerun of historical data can be made with the same dictionary that was used in real time. With experience, and confidence is gained, a 6-month update might be considered. I believe, with the safety built in that an incoming real-time report will not be used unless it matches the dictionary, that hand work will be limited to a day or two per update, or a week at most. If that is not the case, we should determine why so much work is required.

APPENDIX II

HANDLING OF STATION TYPE

A field in the MOS-2000 dictionary called NBLOCK in the TDL Office Note ON 00-1, consisting of 6 digits, was originally for the purpose or holding the block/station number of a station. The use of the field for this purpose faded out, and it became used for (1) defining with the four values 0, 3, 6, and 9 the station type as either ocean, inland water, land or inland water, and land, respectively (the last and 6th digit), and (2) five digits to represent quality flags for different weather variables. Later, the first, leftmost digit, was taken to also describe the station type, but in a different way. It was given a value 2 for mesonet data, 3 for COOP stations, 4 for RFC, 5 for synoptic, and 1 for all other types.

In preparing a new dictionary with U530, certain decisions had to be made. This appendix attempts to describe how this field was prepared. In preparing the dictionary, some stations are in the input dictionary, and some are new. Below, I call these "old" and "new," respectively.

Buoys with the incoming designator MBUO should be water, but that does not determine ocean versus inland water. Also, some stations not designated as BUOY should be designated as water. A 2.5-km land/water grid was used to try to designate as many stations as possible correctly. However, some stations are outside the grid, and have to be handled differently.

Entries that match old stations

Stations outside the land/water grid

The complete field is used unchanged. Because the incoming dictionary should have been reasonably correct, it was left intact.

Stations inside the land/water grid

Leftmost digit–left intact. Note that the hourly data do not contain COOP reports, so any such stations in the incoming dictionary will likely be discarded.

Center 4 digits–left intact.

Rightmost digit–corresponds to land/water grid–0, 3, or 9. Note that a 6 is not used, as there is no way to determine it.

Entries consisting of new stations

Stations outside the land/water grid

Leftmost digit–corresponds to hourly data type, 2 for mesonet (MSxx), 5 for synoptic (SYNO), 6 for moored buoys (MBUO), and 1 for all other.

Center 4 digits–generic "1111".

Rightmost digit–corresponds to hourly data type–0, 3, or 9.

Stations inside the land/water grid

Leftmost digit–corresponds to hourly data type– 2, 5, 6, or 1.

Center 4 digits–generic "1111".

Rightmost digit–corresponds to land/water grid–0, 3, or 9.

It is not possible to categorize every station correctly; some hand work will be necessary. With the leftmost digit being 6 for MBUO stations, it should be relatively easy to plot and remove, for instance, stations in Canada that are likely in lakes that are not recognized by MOS-2000 software. Or, in certain instances, they could just be changed to land.

U600

PERFORMS SCREENING REGRESSION ANALYSIS

Harry R. Glahn
January 1, 1996

PURPOSE: U600 is one in a series of programs designed to develop statistical relationships, usually regression equations, between predictands and predictors from archived data. The primary input data, point data from observations, interpolated from model fields by U201, etc., are packed in a GRIB-like TDLPACK format. Because the packed data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in a random access file; these data are also packed in the TDLPACK format. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U600 uses a driver DRU600 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U600 (actually, subroutine) for every application. U600 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. The default input and output unit numbers are set to 5 and 12, respectively, in dru600. The output of U600 is a set of regression equations prepared for evaluation by other programs. The equations are also output in ASCII for viewing or printing. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U600.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) **Output Control**

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU600.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U600" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U600 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U600HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U600 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). Values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU600 DATA statement = 12. Changing the default unit number allows multiple runs of U600 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = The variable IDs and the forced predictor IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.
(7) = The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) = The variable ID list in summary form after reordering low to high. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)
(9) = The variable list in summary form after reordering. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).
(10) = The variable ID's for the first day (day 1) as read from the archive datasets. This is just a list of all the variables on the input files.
(11) = The variable ID's of the archived data actually needed [entries in MSTORE(,)].
(12) = The list of stations on the input file(s).
(13) = Missing data diagnostics (see NOMISS in Record Type 3).
(14) = A diagnostic will be provided when there are no data for a particular input file.
(15) = All data values in the AA(,) matrix. For each case (cycle or day), after all input data have been read, the

AA(,) matrix holds all variable values for all stations. The print is by station, then the variable values are printed in the order dealt with in U600 (see IP(9)). Except for a very few days, this would produce voluminous files.

(16) = All or a portion of the data values in the AA(,) matrix. For each case (cycle or day), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The print is by variable controlled by JP( ) (see Record Type 13 below), the station values being printed in the order dealt with in U600 (see IP(5)). Except for a very few days or a few variables, this would produce voluminous files.

(17) = The cross product matrices. Except for a very few variables, this would be a lot of data. Also, the print is formatted F12.2, so many cases or large variable values may cause the value to not be printable. (Subroutine PRCROS could be modified for another format.)

(18) = The predictor means, standard deviations, and correlations of predictands with predictors.

(19) = The predictand means and standard deviations.

(20) = All equations, as the predictors are being selected, will be saved on unit IP(20) for viewing. When IP(20) ≠ KFILDO, only the last equation (the one with all the predictors) will be put on unit KFILDO (unless IP(20) = KFILDO). See the Comments section for more explicit information.

(21) = Summary reductions of variance and standard error of estimate.

(22) = A summary of the number of times each predictor is used in the equations.

(23) = Information concerning opening and closing of files.

(24) = Information as to where the variables are to be found--directly on input, binary computed from a previous variable, or (at least attempted to be) computed through subroutine OPTX.

For checkout, it may be advisable to set all these values to zero or the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files. If a file of the same name exists, it will be deleted when IP( ) ≠ 0 except for KFILDO. Do not use 45 through 49, 97, or 99 as unit numbers; they are reserved for other purposes.

Record Type 2 - Format (A72) **Run Identification**

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (9(I8/),7(F8.4/),3(I8/),F8.0/I8) **Control Parameters**

This group of records contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

<u>NSKIP</u> - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

<u>JSTOP</u> - The total number of errors that will be tolerated before the program halts (see Comments section).

<u>INCCYL</u> - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. Because of the lookahead feature required for predictands, and to maintain efficiency, date/times should not be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship.

<u>NST</u> - The (maximum) number of predictors to select.

<u>MFORCE</u> - The (maximum) number of predictors to force into the equation. The order read in (see Record Type 14) is the order to be forced. Various variance and covariance thresholds as described below apply.

<u>NSELT</u> - Selection method; there are two implemented. When NSELT = 1, the next predictor selected is the one that gives the highest additional reduction of variance (RV) with ANY ONE of the predictands. When NSELT = 2, the next predictor selected is the one that gives the highest AVERAGE additional RV over ALL of the predictands. However, since an average RV is harder to achieve than an RV for a single predictand, when CUTOFF (see below) has been reached for NSELT = 2, selection reverts to NSELT = 1 for that equation set.

<u>NSMETH</u> - Stopping method. The only one implemented is using CUTOFF, FORCUT, COLN, COLNB, and COLNGB (see below). Set = 1.

<u>NECVRB</u> - The number of values of an individual variable that is required for that variable to be used in an equation. A zero means that any number will be accepted. When NECVRB $\neq$ 0, the AA( ) matrices are saved and no cross products are calculated on pass 1 through the data. This can be 0 and the thresholds below will still apply. The primary purpose of NECVRB is to eliminate a predictand (or predictor) for a particular station (or group) when data are (largely) unavailable (e.g., one or more projections when doing simultaneous development). However, it can also apply to predictors, for example, observations; that would mean that the equation (for that station or group) could be developed, but not with that predictor. This keeps a variable which is largely missing for a particular station or group from deleting development for that station or group.

NECCAS -	The number of cases that is required for an equation to be developed. This applies to single station equations or generalized operator equations. Use 0 to disable, but not a good idea.
CUTOFF -	For NSELT = 1 (see above), the additional fraction of the variance of one or more of the predictands that has to be explained for another predictor to be included in the equation when screening predictors. For NSELT = 2, the additional average fraction of the variance of all of the predictands that has to be explained for another predictor to be included in the equation. Use 0 to disable, but not a good idea.
FORCUT -	The additional fraction of the variance of one or more of the predictands that has to be explained for another predictor to be included in the equation when forcing predictors. Use 0 to force all predictors, but not a good idea. Note that when FORCUT is > CUTOFF, the "forced" predictor could be rejected, but accepted by the normal screening process. Also note that this applies to individual predictand reductions of variance, not an average regardless of the value of NSELT.
VARNB =	The variance of a point binary predictor necessary for the variable to be used. Use 0 to disable.
VARNGB =	The variance of a grid binary predictor necessary for the variable to be used. Use 0 to disable.
COLN -	The acceptance threshold for collinearity of other than binary predictors. That is, a predictor will not be allowed into the equation if (1-COLN) of its variance has been explained by the predictors already selected. Use 0 to disable.
COLNB -	The acceptance threshold for collinearity of cumulative or discrete (point) binary predictors. That is, a point binary predictor will not be allowed into the equation if (1-COLNB) of its variance has been explained by the predictors already selected. Use 0 to disable.
COLNGB -	The acceptance threshold for colinearity of grid binary predictors. That is, a grid binary predictor will not be allowed into the equation if (1-COLNGB) of its variance has been explained by the predictors already selected. Use 0 to disable.
NOMISS -	Indicates whether (>0) or not (=0) missing data will be identified on file KFILDO [IP(13) = 0] or IP(13) [IP(13) > 0]. (See Comments section.)
NEW -	Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 may not be useful.

PXMISS - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value. In U600, this must be 0 or 9999. A value of 9999 will cause a 9997 datum to be treated as missing; a value of zero will be used as such. A value of 9997 might occur in the input data if forecasts were being used--an unusual circumstance.

MODRUN - A value to be used as the identifying "model/run" number for identifying the equations (the predictands). This 2-digit value is inserted as "DD" into the first word of the ID of each predictand just before the equations are output on file Unit No. KFILEQ (see Record Type 8).

Record Type 4 - Format (I3,4XA60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) **Date List**

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U600 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

Record Type 6 - Format (I3,4XA60) <u>Vector Data Input Files</u>

This group of records identifies the data sets from which the point data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

<u>KFILIN</u>(J) - Unit numbers for the data files (J=1,NUMIN).

<u>NAMIN</u>(J) - Names of files corresponding to KFILIN(J) where these data reside (J=1,NUMIN). (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. (A complication is the necessity for the lookahead feature for predictands.) When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4XA60) <u>Random Access Files</u>

This group of records identifies the MOS-2000 random access data sets from which constant (and possibly other) data are read. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. If no data are needed from a random access file, only the terminator is necessary.

<u>KFILRA</u>(J) - Unit numbers for the random access constant files (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

<u>RACESS</u>(J) - Names of files of constant data matching KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) <u>Equation Output File</u>

This record (plus the terminator record) identifies the equation output file. Note that this is in addition to the print provided through IP( ) values. Records are read until the terminator KFILEQ( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If this is an empty set, the equations will not be output.

<u>KFILEQ</u> - Unit number for the output equation file.

<u>OUTNAM</u> - Name of file where this output is to reside. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) **Station and Location Files**

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which equation development is to be done (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used. When KFILD(1) = KFILD(2) and DIRNAM(1) = DIRMAM(2), all stations in the directory are used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (14(A8,1X)) **Station List**

This group of records identifies the (groups of) stations for which equation development is desired. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read within subroutine RDSTAD by RDC, which eliminates any blanks found in the input. Duplicate stations in the list for a group are kept, but a diagnostic is furnished on unit IP(5). Note that this diagnostic applies only to duplicates within a group, not from group to group. For NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty set terminates the station input. That is, the last group and its terminator must be followed by another terminator signifying an empty set. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) **Variable File**

This record (plus the terminator record) identifies the file from which the variable ID's (both predictors and predictands) are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI (set in DRU600), PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) **Variable Constants File**

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 13 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,8XI2,I3,1XI6) **Variable List**

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 13 is read by subroutine RDVRBL. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading predictors in U201.

ID(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

JP(N) - For each variable N, JP(N) indicates print or no print when ≠ 0 or = 0, respectively, for data values at the stations. These values combined with IP(15) and IP(16) allow easy control of output. For instance, all variables could have JP( ) ≠ 0 and IP(15) ≠ 0 (or IP(16) ≠ 0) for a short diagnostic run. All such print could be turned off by setting IP(15) = 0 (or IP(16) = 0) in this control file. Alternatively, IP(15) (or IP(16)) could be ≠ 0 and selected data obtained by changing JP( ) from 0 to ≠ 0.

NTYPVR(J) - The type of variable (J=1,NVRBL):
1 = predictor.
2 = predictand.
NTYPVR determines how the tau is used in the variable ID. For predictors, the variable must be present just as specified including the tau. For predictands, the lookahead feature must be used, so tau is added to the current date NDATE to get the

date for which the variable is needed at tau = 0. As a special feature, if a predictor is an observation (CCC between 700 and 799 inclusive) and not from the AEV data archive (DD = 82), then tau is treated as lookahead as it is with predictands.

<u>LIMIT</u>(J) - The number of zeros or of ones necessary for a binary predictand to have for an equation developed for it. This means that both AVG(N)*SMPL and (1-AVG(N))*SMPL must be $\geq$ LIMIT(N) for predictand N, where AVG(N) is the binary predictand average and SMPL is the sample size.

Record Type 14 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3) <u>Forced Predictors</u>

This group of records contains the variable ID's of the forced predictors. Records are read until the terminator IDFORC(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 14 is read by subroutine RDFORC. Upon completion of reading this record type, N=1,NFORCE. Note that the format and information for IDFORC(,) is exactly the same as for ID(,) above, up to the point needed.

<u>IDFORC</u>(J,N) - The first 3 (J=1,3) words of the predictor ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed. This is the exact format for variable ID's as for Record Type 12.

Note that if the number of (useable) predictors read for forcing does not match MFORCE read in Record Type 3, the number actually read will be used. This is flagged as an error to give the user feedback and positive control on the (number of) forced predictors.

<u>CONTROL FILE INPUT</u>: (Name read from U600.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) <u>Date List</u>

<u>When the dates are not provided in file U600.CN</u>, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU600), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input.

Terminator is 99999999. Data for the first date in the list must be available or U600 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U600.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided in file U600.CN, this group of records identifies the stations (or locations) to be included in the regression analysis. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which interpolated values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 9, control file 'U600.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U600.CN) (Unit = KFILD(2))

Record Type 1 - Format **Station Locations**
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. The call letters read from KFILD(2) (see Record Type 9) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical no matter the order read in (see NALPH in Record Type 3). (Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run.) However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. For example, there is no use of the elevation in the equation derivation. However, this information is available for a computation routine, if one is needed. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3).

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited, except when it also constitutes the list to be kept (see Record Type 9), in which case the number of entries is limited by ND1-1. No terminator is used.

CONTROL FILE INPUT: (Name read from U600.CN) (Unit = KFILP)

Record Type 1 - Format
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,8XI2,I3,1XI6) **Variable List**

When the variables to use in the run are not in file U600.CN, this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDVRBL. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading predictors in U201. See Control File Input for U600.CN, Record Type 13, unit KFILDI above for more detailed information; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U600.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32) **Variable Constants**

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U600 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

Other processing occurs with the reading of these records in RDVRBL, much of it associated with the plain language description. See Chapter 4 in TDL Office Note 00-1, Variable Definition, for details.

DATA INPUT:

All data interpolated from models and produced by U201 for input to U600 and all other vector data will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note 00-1 MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format, except for the call letters (directory) record. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U600.CN in Record Types 6 and 7.

A. SEQUENTIAL VECTOR DATA

One or more sources of vector data, each probably prepared by U201, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U600.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated. Observation records are expected to have the date of the observations and a tau of 0.

B. RANDOM ACCESS VECTOR DATA

Up to 5 sources of random access vector data are accommodated; however, it is unlikely more than 1 or 2 will be needed. These data exist in the MOS-2000 External Random Access File System. These data are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

DATA OUTPUT

All equations are placed in an ASCII file, on Unit No. KFILEQ (see Record Type 8), that can be easily edited, especially for the stations to apply the equations to, and used directly by other programs such as U700 and U900 (see "MOS Equations" in TDL Office Note MOS-2000). Note that the predictand IDs have been modified so that the 2-digit value MODRUN (see Record Type 3) has been inserted as "DD" into the first word. This allows identification of the model or models on which the equations were based, or even a different development run on the same model (e.g., for primary and backup equations). Observations (CCC in the range 700-799) will have a tau [IDPARS(12)] representing how far ahead to get the observation. That is, tau is added to NDATE, then the record read with a tau = 0. (In U700 or U900, the "predictand" IDs will be modified to "forecast" IDs.)

EXAMPLE CONTROL FILE: 'U600.CN'

An example exists as file 'U600.CN' in directory home21.glahn.dru600 on blizzard. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U600.CN', Record Type 1 and the definition of KFILDO in the driver DRU600. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The CRAY does not allow this. It is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

The binary indicator "B" in the first word of a predictor ID must be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), or 5 (indicating a grid binary). For a predictand, "B" can also take the value of 2 (indicating a cumulative variable from below) or 3 (indicating a discrete variable). For the latter, the upper threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID's. If there is no lower one, the lower threshold is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999. Subroutine CKIDS called from RDVRBL prints a diagnostic if B for a predictor is not 0, 1, or 5 or for a predictand is not 0, 1, 2, 3, or 5. However, the variable is not eliminated.

Especially because of the lookahead feature necessary for predictands (and possibly for observations as predictors), all data needed for Day 1 should actually be available (e.g., there be a data record for all predictands, no matter how far ahead, for Day 1) when the variable is to be computed through subroutine OPTX. (There may also be cases when this restriction is also true for variables not computed through OPTX.) This does not mean that there cannot be a station with "missing" data. Also, each variable should be accessed on the same unit number. That is, if two seasons of data are on different files, they must have the same unit number.

It is not necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles, unless the variable is computed from other variables not otherwise used as a "raw" variable. That is, a raw (not computed) variable need not be present for Day 1, but one used only in computations must be, because U600 has no way of knowing it will be needed for future cycles.

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

U600 might use Unit No. 45 or even 48 or 49.

COMMENTS

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to

5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U600 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event is interpreted as having the value of PXMISS (see Record Type 3). Normally, PXMISS would be zero. Otherwise, to interpret the value as 9999, the case would have to be thrown out. Presumably, the only time 9997 will occur is when operational forecasts from U900 or U910 or test forecasts from U700 or U710 are input. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such. These values will be used the same way no matter how they are packed.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired (see IP(13)). This probably means a set of dates should be supplied to U600 for which it is known there are good data; then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

On Day 1, GFETCH is entered directly for every predictor (except a point binary) because it is not yet known when OPTX must be entered. ("Day 1" is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) <u>is not</u> counted as an error in VRBL1 for Day 1. It <u>is</u> counted as an error in VRBL2 (through OPTX), because GFETCH should not be entered directly when OPTX is needed.

When a point binary is needed, it will be computed in U600 if possible (i.e., the "basic" variable is available) even though it may exist on an input file. However, other computed predictors, such as sine and cosine functions are used from the input file if they exist, even though they might be computable in U600. For strictly mathematical variables like sine and cosine, it is likely more efficient to do the computation in U600.

The "interpolated" data input(s) can contain constant data taken from the MOS-2000 External Random Access File System by U201. Alternatively, U600 can take constant data directly from MOS-2000 External Random Access files. It is possible that no interpolated data on sequential files be used, and all input furnished be on "constant" random access files.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1.

It is unlikely that when no predictor in a run has a particular model number, that the file containing that model will be needed; therefore, the diagnostic to this effect should be heeded, and that model's file(s) should not be used because it (they) will be read during the run even though not needed. An exception is for a subroutine that would use data from a model that was not indicated by the model number in the predictor ID. Even though the possibility is remote, that is the reason that the file is not automatically closed.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 10 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

Equations can be output on either the default output unit number KFILDO or the unit number specified in IP(20), or both. An option is available for writing all the equations as each individual predictor is selected or just the final equation with all predictors. The table below shows the options.

Default Output Unit No.	Value of IP(20)	Equations on Default Output	Equations on IP(20)
12	12	All	(Same as default)
12	0	Last Equation	None
12	20 (say)	Last Equation	All

<u>SETTING UP THE DRIVER DRU600</u>

The preparation of the driver for a particular U600 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B - Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).

ND1 - Maximum number of stations (or points) that can be dealt with in the regression analysis. Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) unless, of course, the station directory is to be used as the station list.

ND2 - The size of the work area Q( ). This holds the sample size(s), sums, and sums of cross-products (in the non-redundant portion of the matrix). A test run of a day or so will provide guidance on the value of ND2 that would be appropriate for a similar run on many days. If more than one set of equations is to be developed (e.g., more than one "region" or single station), Q( ) must be large enough to handle one such set. If it is not large enough to handle all such sets at one time, more than one pass through the data will be required and the run will be much slower.

ND3 - Not used.

ND4 - The maximum number of variables (predictors and predictands) that can be dealt with.

ND5 - Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.

ND6 - Maximum number of all sequential file input sources (e.g., models) that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

WRITING NEW SUBROUTINES

It is not expected that computational routines will be used to any great extent in U600. Rather, computations, except for point binaries, are expected to be done in U201. Any computations that would be done in U600 would have to be done on point data (not gridpoint data) because gridpoint data are not accommodated in U600. Even so, an "option" subroutine, OPTX, is provided for possible use. (In the case of development for gridpoints, the gridpoints are given identifications in the same way as other locations, such as stations.)

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib,dru600, dataset 'u600.com', along with the libraries 'home21.tdllib.moslib' and 'home21.tdllib.u600lib', all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: home21/tdllib/u600lib. The driver is in dru600.

INT600

INITIALIZES FROM CONTROL FILE FOR U600

Harry R. Glahn
January 1, 1998

PURPOSE: Performs most of the initialization for U600. It reads the control file U600.CN, and other input files. It also writes the header to the output equation file if such a file is to be used. This is just an in-line subroutine pulled out of U600 basically to shorten the long U600.

RESTRICTIONS:

Is useful only for U600.

NONSYSTEM ROUTINES USED:

IOPEN, TIMPR, RDSNAM, RDI, DATPRO, RDSTGN, RDSTGA, RDVRBL, RDFORC, WRHEAD, and IERX

LANGUAGE: FORTRAN 77

LOCATION: u600lib

VRBL61

FURNISHES VARIABLES TO U600 FOR DAY 1

Harry R. Glahn
July 25, 1996

PURPOSE: To obtain for U600 all variables for Day 1. All data available in the input file(s) that are needed for Day 1 have been read and stored in the internal MOS-2000 Storage System. The predictors have been numerically ordered so that access to the Storage System is minimized. For each predictor ID encountered, it is determined whether or not it is already held in VRBL61. If the field is found, processing of the field continues. If not, OPT600 is entered, which attempts to compute the predictor, at which point processing continues. The ID's for the fields read from the input files and stored in the Storage System that are actually used are stored and marked so that for the following days only those fields need be considered.

RESTRICTIONS:

U600 operates on point data; gridpoint calculations are not supported, but rather are done in U201.

NONSYSTEM ROUTINES USED:

GFETCH, OPT600, BINFUL, VRBLX1

LANGUAGE: FORTRAN 77

LOCATION: u600lib

VRBL62

FURNISHES VARIABLES TO U600 FOR DAYS AFTER DAY 1

Harry R. Glahn
July 25, 1996

PURPOSE: To obtain for U600 all variables for all days after the first. VRBL62 reads the input file(s) and from the ID's stored by VRBL61 determines whether or not a particular field is needed. If not, it is skipped. If it is needed, the computation(s) that can be done on it without entering OPT600 are done. Also, if it is to be needed by OPT600, it is stored in the internal MOS-2000 Storage System. After all data that may be needed for the day being processed have been read, VRBL62 passes through the predictors (again) to see whether more processing is necessary. From here on, the operation of VRBL62 is much like that of VRBL61. For each predictor ID encountered, it is determined whether or not the predictor has already been computed, If not, it is determined whether or not it is already held in VRBL62 from the previous computation. If not, the Storage System is accessed. If the field is found, processing of the field continues. If not, OPT600 is entered, which attempts to compute the predictor, at which point processing continues.

RESTRICTIONS:

U600 operates on point data; gridpoint calculations are not supported, but rather are done in U201.

NONSYSTEM ROUTINES USED:

GSTORE, OPT600, BINFUL, VRBLX1, UNPACK, UNPKBG

LANGUAGE: FORTRAN 77

LOCATION: u600lib

VRBLX1

PRINTS AA MATRIX FOR U600

Harry R. Glahn
July 25, 1996

<u>PURPOSE</u>: To print the AA(,) matrix for U600 under control of IP(15) and IP(16) by variable or station, respectively.

<u>RESTRICTIONS</u>:

None, except the data are printed under the F format with 2 decimal places of accuracy.

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: u600lib

WRAA

WRITES AA MATRIX FOR U600

Harry R. Glahn
July 25, 1996

<u>PURPOSE</u>: To write the AA( ) matrix for U600 when needed. If all equations can be developed on one pass through the data, the AA( ) matrix need not be saved; otherwise, it does. That is, when multiple passes are needed, the input for all except the first is from the file written by WRAA. Even though AA( ) is a doubly dimensioned variable in some routines, the data are arranged contiguously so that writing can be done here from the single dimensioned variable A( ). A vector MISING( ) associated with AA( ) is also written.

<u>RESTRICTIONS</u>:

The first time WRAA is entered, the file is opened. That is, WRAA operates differently the first time entered from all other times. The name of the file written consists of the 4 characters from IPINIT (which the user has available to identify this specific run) plus the characters "AA". This should uniquely identify this scratch file. The file name is provided to the calling routine U600, which deletes the file before stopping (assuming a run to completion).

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: u600lib

RDAA

READS AA MATRIX FOR U600

Harry R. Glahn
July 25, 1996

PURPOSE: To read the AA( ) matrix for U600 that has been written by WRAA when needed. If all equations can be developed on one pass through the data, the AA( ) matrix is not saved; otherwise, it is. That is, when multiple passes are needed, the input for all except the first is from the file written by WRAA. Even though AA( ) is a doubly dimensioned variable in some routines, the data are arranged contiguously so that reading can be done here into the single dimensioned variable A( ). A vector MISING( ) associated with AA( ) is also read.

RESTRICTIONS:

The name of the file read consists of the 4 characters from IPINIT (which the user has available to identify this specific run) plus the characters "AA". This should uniquely identify this scratch file. The file name is provided to the calling routine U600, which deletes the file before stopping (assuming a run to completion).

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: u600lib

SAVREC

SAVES INFORMATION FOR U600 FOR RESTORING

Harry R. Glahn
July 25, 1996

<u>PURPOSE</u>: To save certain information internally for U600 when IP(20) NE 0 for possible restoring. Restoring is done with routine RESTOR.

<u>RESTRICTIONS</u>:

None.

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: u600lib

RESTOR

RESTORES INFORMATION FOR U600 SAVED BY SAVREC

Harry R. Glahn
July 25, 1996

<u>PURPOSE</u>: To restore certain information to U600 that has been saved by SAVREC.

<u>RESTRICTIONS</u>:

None.

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: u600lib

SUMRY

PROVIDES SUMMARIES FOR U600

Harry R. Glahn
July 25, 1996

PURPOSE: To furnish for U600 summary statistics. The data have been saved to unit KFIL08 from which SUMRY reads. There are two kinds of summary. First, the average reduction of variance and standard error of estimate are provided for each predictand on unit IP(22). Second, the number of times each possible predictor is selected in each of the equation positions 1 through 10, all other positions grouped, and the total for the predictor are provided on unit IP(23). If the reduction of variance for a particular predictand is near zero (LT .000001), it is assumed that no equation was developed for that predictand, and the zero is not included in the average.

RESTRICTIONS:

None other than the I and F formats of the output on unit KFILDO.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: u600lib

RDFORC

READS A FORCED PREDICTOR LIST

Harry R. Glahn
August 25, 1996

PURPOSE: To read a list of predictors to force for U600. Discards any predictors read that are not in the total predictor list.

RESTRICTIONS:

It is assumed the file on which the forced predictors reside has been opened.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: u600lib

XFER

TRANSFERS CROSS-PRODUCTS TO TWO-DIMENSIONAL ARRAY

Harry R. Glahn
August 25, 1996

PURPOSE: To transfer sample sizes, sums, and sums of cross products of variables from a single-dimensioned array Q( ) to SAMPL, SUM( ), and P(,), respectively. For storage efficiency, this information, including "half" of the symmetric cross-product matrix, is stored in a single dimensioned-array until needed.

RESTRICTIONS:

The order of information is, of course, prescribed.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: u600lib

XPROD

COMPUTES SAMPLE SIZES, SUMS, AND SUMS OF CROSS-PRODUCTS

Harry R. Glahn
August 25, 1996

PURPOSE: To compute sample sizes, sums, and "half" the symmetric matrix of sums of cross products from a matrix of data in AA(,) for U600. All results are put into a linear array. Several groups of stations can be done, each group consisting of a different number of stations.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: u600lib

PRCROS

PRINTS SAMPLE SIZE, SUMS, AND SUMS OF CROSS-PRODUCTS

Harry R. Glahn
August 25, 1996

PURPOSE: To print sample size, sums, and "half" of the symmetric sums of cross products for U600.

RESTRICTIONS:

None.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: u6OOlib

VRBVEC

COMPUTES A USEABILITY VECTOR FOR VARIABLES FOR U600

Harry R. Glahn
August 25, 1996

PURPOSE: To compute a vector for each station or group to indicate which variables can be used. This is done by group so that the minimum number needed for an equation applies to a group as well as to single stations. All groups/stations are done in one pass, even though equations for all groups/stations may not be able to be done together. In U600, when a variable can't be used for a station (or all stations in a group), the variance will be zero, and the variable will not be used. This is done primarily to cull out predictands at a particular projection so that equations will be developed for other projections. However, it can also apply to observations as predictors; that would mean that the equation (for that station or group of stations) could be developed, but not with that predictor.

RESTRICTIONS:

The data are read from a file that is assumed to be open.

NONSYSTEM ROUTINES USED:

RDAA

LANGUAGE: FORTRAN 77

LOCATION: u600lib

WREQ

WRITES EQUATIONS FOR PRINTING

Harry R. Glahn
August 25, 1996

PURPOSE: To write regression equations for printing, together with the reduction of variance, multiple correlation, and standard error associated with each equation/predictand.

RESTRICTIONS:

It is assumed the file on which the equations are written has been opened.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: u600lib

WRHEAD

WRITES HEADER RECORDS FOR OPERATIONAL EQUATIONS

Harry R. Glahn
August 25, 1996

<u>PURPOSE</u>: To write the first few records to a file which is to contain equations for operational use.

<u>RESTRICTIONS</u>:

It is assumed that the formatted file on which the header records are to be written has been opened.

<u>NONSYSTEM ROUTINES USED</u>:

None.

<u>LANGUAGE</u>: FORTRAN 77

<u>LOCATION</u>: u600lib

WROPEQ

WRITES EQUATIONS FOR OPERATIONAL USE

Harry R. Glahn
August 25, 1996

PURPOSE: To write equations for operational use for U600. The first few records should have been written by subroutine WRHEAD.

RESTRICTIONS:

It is assumed that the formatted file on which the equations are to be written has been opened.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: u600lib

THSET

SETS THRESHOLDS

Harry R. Glahn
August 25, 1996

PURPOSE: To use the single threshold provided in TRESHL( ) to set a lower and upper threshold in TRESHL( ) and TRESHU( ), respectively. For values of the binary indicator "B" in the first word ID of the following values, the indicated actions are taken:

(0) For a non-binary variable, TRESHL( ) and TRESHU( ) are set to zero.
(1) For a binary cumulative from above, the upper threshold is set = 99999. This value of B is almost mandatory for predictors. However, the routine does not stop, but provides an error code IER = 126 and a diagnostic to the default output file.
(2) For a binary cumulative from below, the lower threshold is set -99999. It is intended that B = 0 or 1 for predictors for operational equations for ease of understanding. However, B = 2 could be accommodated.
(3) For a discrete binary, the threshold provided is the upper one, and the lower one is set to the upper one of the next lower discrete binary with the same ID's. In case this is the uppermost discrete binary with the same ID's, the threshold provided should be 99999 if all remaining cases are to be included. [Actually, only 4 places of accuracy can be provided, so the threshold should be 9999 as it would be in U201; it will be set in THSET to 99999 for symmetry. (Both .9999E+04 = 9999 and .9999E+05 = 99990 will be treated the same way.)] In the case this is the lowermost discrete binary with the same ID's, the lower threshold is set = -99999. If a discrete binary is indicated and there is only one with the same ID's, then the resulting binary will perform the same as B = 2.
(4) Not used or checked.
(5) Indicates grid binary. The lower threshold is not changed and the upper is set to 0.
(6) Indicates a "matching" variable. Set upper threshold = 0.
(7) Indicates a "matching" variable. Set upper threshold = 0.
(8) Indicates a "matching or" variable. Set upper threshold = 0.
(9) Indicates a "matching or" variable. Set upper threshold = 0.

RESTRICTIONS:

None other than stated above.

NONSYSTEM ROUTINES USED:

None.

LANGUAGE: FORTRAN 77

LOCATION: moslib

U660

COLLATES, PRINTS, AND PACKS DATA

Harry R. Glahn
January 1, 1998

PURPOSE: U660 collates data from a variety of MOS-2000 vector inputs, and writes for printing and packs and writes vector output for the stations and variables designated by control files. Because the packed data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in a random access file; these data are also packed in the TDLPACK format. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U660 uses a driver DRU660 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U660 (actually, subroutine) for every application. U660 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. It is possible to access data for variables at a date/time after the date/time being processed; this is accomplished with the variable "ITAU( )" and is called the "lookahead" feature. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U660.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU660.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U660" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U660 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U660HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U660 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are

different (except for IP(1)). 18 values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU660 DATA statement = 12. Changing the default unit number allows multiple runs of U660 or other programs within the same directory without overwriting.

(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).

(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).

(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).

(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).

(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.

(7) = The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).

(8) = The variable ID list in summary form. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)

(9) = The variable list in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).

(10) = The variable ID's for the first day (day 1) as read from the archive tapes. This is just a list of all the variables on the input files.

(11) = The variable ID's of the archived data actually needed, in order as they appear on the archive files for day 1.

(12) = The list(s) of stations on the input file(s).

(13) = Not used.

(14) = A diagnostic will be provided when there are no data for a particular input file. With tape switching, this may not be of much use, and could be misleading.

(15) = Data written in the order packed for each variable indicated by JP(3,) > 0. This is separate from the optional writing associated with JP(2,). Except for a very few days, this would produce voluminous files.

(16) = All or a portion of the data values in the AA(,) matrix. For each case (cycle), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The print is by variable controlled by JP(2,) (see Record Type 12 below), then the station values are printed in the order dealt with in U660 (see IP(5)). Except for a very few days or a few variables and stations, this would produce voluminous files.

(17) = All or a portion of the data values in the AA(,) matrix are written to a delimited text file for each case (cycle) up to and including NPRINT cases. The print is by variable controlled by JP(2,) (see Record Type 12 below), then the station values are printed in the order dealt with in U660 (see IP(5)). Except for a very few days or a few variables and stations, this would produce voluminous files.

(23) = Information concerning opening and closing of files.

(24) = Information as to where the variables are to be found--directly on input, binary computed from a previous variable, or (at least attempted to be) computed through subroutine OPTX.

For checkout, it may be advisable to set all these values to the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (7(I10/),F10.0/(I10)) Control Parameters

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

KSKIP - When nonzero, indicates the output file on unit KFILIO (see Record Type 8) is to be moved forward until all data for date/time KSKIP have been skipped. KSKIP is input as either YYYYMMDDHH or YYMMDDHH, and then used as YYYYMMDDHH. Note that when KSKIP > 0, there must be data with a directory record on the output file. Otherwise, the first read will be attempted and the program judges non-matching stations in the directory. Also, KSKIP must be < IDATE(1) (see Record Type 5).

KWRITE - The existing call letters record (directory) pertaining to the data for the date/time KSKIP (see above) will be checked with the one available for writing. If they match, the new one will not be written; if they don't match, the new one will be written when KWRITE = 1, but the program will halt with a diagnostic when KWRITE = 0. Note that this has no effect when KSKIP = 0.

<u>NSKIP</u> - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

<u>JSTOP</u> - The total number of errors that will be tolerated before the program halts (see Comments section).

<u>INCCYL</u> - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. Because of the lookahead feature required for predictands, and to maintain efficiency, date/times should not be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship.

<u>NEW</u> - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

<u>NALPH</u> - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

<u>PXMISS</u> - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value.

<u>NPRINT</u> - The number of cycles of data to write for printing to units IP(16) and/or IP(17) under the format control and JP(2,) provided with each variable (see Record Type 13). Both units can be used simultaneously; however, this would usually only be done for testing because NPRINT is the maximum number of dates written to the output file(s). NPRINT is normally used to limit printing to IP(16) to a small subset of the total number of dates. When writing to IP(17), NPRINT must be large enough to write the entire sample to IP(17).

<u>ICHARS</u> - The number of characters of call letters to print when printing is indicated by JP(2,). This is constrained to be between 4 and 8 inclusive.

<u>LNGTH</u> - The line length for printing to unit IP(16). For a line printer, 132 is appropriate; for a laser printer, 80 may be desirable.

Record Type 4 - Format (I3,4XA60) <u>Date List File</u>

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

<u>KFILDT</u> - Unit number for the file containing the input date list.

<u>DATNAM</u> - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) <u>Date List</u>

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

<u>IDATE</u>(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list <u>must</u> be available or U660 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be > KSKIP (see Record Type 3).

Record Type 6 - Format (I3,4XA60) <u>Vector Input Data Files</u>

This group of records identifies the data sets from which the vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN. If all input data are from a constant file, only the terminator is necessary.

<u>KFILIN</u>(J) - Unit number for the data file.

<u>NAMIN</u>(J) - Name of file where these data reside. (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. (A complication is the necessity for the lookahead feature for predictands.) When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if

2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4XA60) <u>**Random Access Files**</u>

This group of records identifies the MOS-2000 random access data sets from which constant (and possibly other) data are read. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. If no data are needed from a random access file, only the terminator is necessary.

<u>KFILRA</u>(J) - Unit number for the random access constant files (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

<u>RACESS</u>(J) - Names of files of random access data sets matching KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) <u>**Vector Output File**</u>

This record (plus the terminator record) identifies the packed vector output file. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

<u>KFILIO</u> - Unit number for the vector output file.

<u>OUTNAM</u> - Name of file where this output is to reside. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) <u>**Station and Location Files**</u>

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

<u>KFILD</u>(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

<u>DIRNAM</u>(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (7(A8,1X)) <u>Station List</u>

This group of records identifies the (groups of) stations for which data are to be processed. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

<u>CCALL</u>(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read within subroutine RDSTAD by RDC, which eliminates any blanks found in the input. Duplicate stations in the list for a group are kept, but a diagnostic is furnished on unit IP(5). Note that this diagnostic applies only to duplicates within a group, not from group to group. For NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty set terminates the station input. That is, the last group and its terminator must be followed by another terminator signifying an empty set. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) <u>Variable File</u>

This record (plus the terminator record) identifies the file from which the variable ID's are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

<u>KFILP</u> - Unit number for the variable ID's.

<u>PRENAM</u> - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) <u>Variable Constants File</u>

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. [Variable constants include plain language description and the decimal scaling factor for packing, when writing binary output to unit KFILIO (see record Type 8)]. Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

<u>KFILCP</u> - Unit number for the constants.

<u>CONNAM</u> - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 13 - Format <u>Variable List</u>
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,I3,8XA1,I2,1XI1,1X30A1)

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 13 is read by subroutine RDVR66. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201.

<u>ID</u>(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

<u>JP</u>(J,N) - JP(J,N) indicates for J=1,3:
1 = Whether (>0) or not (=0) variable N will be packed and written to unit KFILIO.
2 = Whether (>0) or not (=0) variable N will be written to units IP(16) and/or IP(17) with the format provided. IP(16) is the primary output for viewing or printing; IP(17) is the delimited text output. Also see NPRINT in Record Type 3 and CFMT below.
3 = Whether (>0) or not (=0) variable N will be written to unit IP(15) to the resolution packed. This is for checkout and quality control of data being written. Since data are packed only when JP(1,N) > 0, both JP(1,N) and JP(3,N) must be > 0 for IP(15) output.

<u>ITAU</u>(N) - The number of hours to add to NDATE (the date being processed, see Record Type 5) to get the variable N. That is, the tau in the ID is left intact, and ITAU is used to "look ahead."

<u>CFMT</u>(N) - The format descriptor for writing on units IP(16) and IP(17) when JP(2,N) > 0. For IP(16), it must be either "F" or "I"; for IP(17), only "F" is allowed.

<u>IWDTH</u>(N)- The field width for writing on units IP(16) and IP(17) when JP(2,N) > 0. Must be $\leq$ 30.

<u>IPREC</u>(N)- The precision for writing on units IP(16) and IP(17) when IP(2,N) > 0. For an "F" format, this is the number of digits after the decimal point. For an "I" format, it is the number of digits always written. Note that for a "zero" to be printed, IPREC( ) must be > 0; otherwise, zero will be printed as a blank.

HEAD(J,N) - The column heading for writing to units IP(16) and IP(17) when JP(2,N) > 0. For IP(16), limited to J=1,30 characters; for IP(17), limited to J=1,11 characters, with character 1 containing a user specified delimiter (colon, semicolon, or comma) for separating data values. In all cases, only IWDTH(N) characters are read. When writing to IP(16), HEAD(,) is right justified; when writing to IP(17), HEAD is left justified and padded by blanks if necessary to fill the J=1,11 characters.

CONTROL FILE INPUT: (Name read from U660.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) **Date List**

When the dates are not provided in file U660.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU660), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U660 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U660.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided in file U660.CN, this group of records identifies the stations (or locations) to be included in the regression analysis. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which interpolated values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 9, control file 'U660.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U660.CN) (Unit = KFILD(2))

Record Type 1 - Format Station Locations
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical within each group no matter the order read in (see NALPH in Record Type 3). [Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. Note that alphabetization is not overall, but by group (see Record Type 10)]. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3).

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited. No terminator is used.

CONTROL FILE INPUT: (Name read from U660.CN) (Unit = KFILP)

Record Type 1 - Format Variable List
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,I3,8XA1,I2,1XI1,1X30A1)

When the variables to use in the run are not in file U660.CN, this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDVR66. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201. See Control File Input for U660.CN, Record Type 12, unit KFILDI above for more detailed information; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U660.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U660 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the data for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer.

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDVR66, much of it associated with the plain language description. See Chapter 4, Variable Identification, for details.

<u>DATA INPUT</u>:

All data input to U660 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format. All such vector data must be preceded by a call letters (directory) record. The sequential files can have multiple directory records, each applying to the data until an end of file or another directory is encountered; the random access files have only one directory record each, which applies to all data in the file. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U660.CN in Record Types 6 and 7.

A. <u>SEQUENTIAL VECTOR DATA</u>

One or more sources of vector data, each probably prepared by U201, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U660.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

B. <u>RANDOM ACCESS VECTOR DATA</u>

Up to 5 sources of random access vector data are accommodated. These data exist in the MOS-2000 External Random Access File System. These data are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants can be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

<u>DATA OUTPUT</u>

There are two forms of data output, besides the diagnostics and other information on units KFILDO and IP( ).

A. <u>ASCII FOR VIEWING OR PRINTING</u>

A maximum of NPRINT (see Record Type 3) <u>cycles</u> of data will be written to the file on unit IP(16) under control of the format provided with each variable N, the variables written controlled by JP(2,N) (see

Record Type 13). The data columns are headed by HEAD(,N) (see Record Type 13). Listing is by station group. If a line length for printing is not sufficient for all variables, then more than one line is used. Line length is specified as LNGTH characters in Record Type 3.

When JP(3,N) > 0, the data will also be written to unit IP(17) to the resolution packed; this is for quality control and would not be used for large samples of data.

B. BINARY MOS-2000 FORMAT

Data for each variable N for which JP(1,N) > 0 (see Record Type 13) for all NDATES cycles will be packed in TDLPACK format and written to unit number KFILIO to the dataset whose name has been provided to OUTNAM (see Record Type 8), unless KFILIO = 0, in which case data are not output. The data are packed with subroutine PACK1D and its associated subroutines.

C. ASCII DELIMITED TEXT FOR INGESTING INTO SPREADSHEET OR OTHER SOFTWARE

A maximum of NPRINT (see Record Type 3) cycles of data will be written to the file on unit IP(17) under control of the format provided with each variable N (printing is limited to "F" format only, see Record Type 13). The data columns are headed by HEAD(,N) where Character 1 of HEAD(,N) contains the delimiter character and Characters 2-11 contain the variable description (see Record Type 13). All variables for a given station and cycle are written to a single line whose limit is determined by the physical characteristics of the hardware.

EXAMPLE CONTROL FILE: 'U660.CN'

An example exists as file 'U660.CN' in directory home21.tdllib.dru660 on blizzard. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U660.CN', Record Type 1 and the definition of KFILDO in the driver DRU660. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The CRAY does not allow this. It is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

The binary indicator "B" in the first word of an ID can be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), 5 (indicating a grid binary), 2 (indicating a cumulative variable from below), or 3 (indicating a discrete variable). For the latter, the upper threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID's. If there is no lower one, the lower threshold is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999.

Since the variables are not ordered (as they are in U600), if a discrete binary is desired (one with both upper and lower thresholds), these variable ID's must be input in sequence and in the correct order (lower threshold first). To make sure all is well, check the thresholds on unit IP(8) or IP(9). Normally, this function should not be needed in U660.

It is not necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles, <u>unless</u> the variable is computed from other variables not otherwise used as a "raw" variable. That is, a raw (not computed) variable need not be present for Day 1, but one used <u>only</u> in computations must be, because U660 has no way of knowing it will be needed for future cycles.

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

U660 would likely not need Unit No. 46. Since U660 does not write to a random access file, either 48 or 49 could be used, and the forecasts would not be compromised.

<u>COMMENTS</u>

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to

5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U660 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 3). Presumably, the only time 9997 will occur is when operational forecasts are input or U700 produces test forecasts. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. This probably means a set of dates should be supplied to U660 for which it is known there are good data. Then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

On Day 1, GFETCH is entered directly for every variable (except a point binary) because it is not yet known when OPTX must be entered. ("Day 1 is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) <u>is not</u> counted as an error in VRBL61 for Day 1. It <u>is</u> counted as an error in VRBL62 (through OPTX), because GFETCH should not be entered directly when OPTX is needed.

When a point binary is needed, it will be computed in U660 if possible (i.e., the "basic" variable is available) even though it may exist on an input file. However, other computed variables, such as sine and cosine functions are used from the input file if they exist, even though they might be computable in U660.

The vector (or "interpolated") data input(s) can contain constant data taken from the MOS-2000 External Random Access File System by U201. Alternatively, U660 can take constant data directly from a MOS-2000 External Random Access files. It is possible that no interpolated data on sequential files be used, and all input furnished be on "constant" random access files.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 10 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

Although a primary purpose of U660 is to provide packed vector data for other programs, writing of such data is not done if KFILIO = 0 (see Record Type 8). That is, no output unit number and file name have been provided. This feature can be used for checkout or if the user wants just a few days for printing.

<u>SETTING UP THE DRIVER DRU660</u>

The preparation of the driver for a particular U660 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the directory (read on unit KFILD(2)) unless, of course, the station directory is to be used as the station list (see ND5).
ND2 -	Not used.
ND3 -	Not used.
ND4 -	The maximum number of variables that can be dealt with.
ND5 -	Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.
ND6 -	Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc., even though the same unit number might be used.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU660 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

WRITING NEW SUBROUTINES

It is not expected that computational routines will be used to any great extent in U660. Rather, computations, except for point binaries, are expected to be done in U201. Any computations that would be done in U660 would have to be done on point data (not gridpoint data) because gridpoint data are not accommodated in U660. Even so, an "option" subroutine, OPTX, is provided for possible use. (In the case of development for gridpoints, the gridpoints are given identifications in the same way as other locations, such as stations.)

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib.dru660, dataset 'u660.com', which includes the libraries 'home21.tdllib.u660lib' and 'home21.tdllib.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: u660lib. The driver is in dru660.

PRU660

PREPARES FORMATS AND WRITES FOR PRINTING FOR U660

Harry R. Glahn
January 1, 1998

PURPOSE: To prepare the FORMATs for the column headings and the columns of data for U660, and to actually write (a portion of) the data in the input matrix AA(,) as specified by the control parameters to Unit IP16. The columns are the variables, and the rows are the stations, by group if more than one group is specified. If the formats, which are supplied for each variable in terms of field width, precision (digits after the decimal point), and either F or I format, require more than one line, as many lines as are necessary are written with appropriate headings. For the I format, the data are rounded, and the "precision" is the number of digits that will always be written. (Note that with the I format, if "precision" is zero, then a zero will not be printed; that is, to get a zero printed, use "precision" of 1 (or greater). Printing of the variables is under control of JP(2,N), where 1 (0) means to print (not print) the variable N. Line length is controlled by an input variable, LNGTH.

RESTRICTIONS:

It is assumed the file on unit number IP16 has been opened.

NONSYSTEM ROUTINES USED:

PRSID, CKIDS, THSET

LANGUAGE: FORTRAN 77

LOCATION: u660lib

RDVR66

READS VARIABLE LIST FOR U660

Harry R. Glahn
January 1, 1998

PURPOSE: To read a combined predictor and predictand list and associated information from a file on Unit No. KFILP. The terms predictor and predictand are not entirely appropriate here, but have been retained from their use in U600. Predictor and predictand are the two "types" of variable in U660, and distinguish between how the tau (projection) is used. For predictands (type 2), the tau is used to "look ahead" to a future date/time and the variable is retrieved at that date/time with a tau of zero. Lookahead is not done for predictors (type 1). KFILP can be the default input file or can be a separate file. Also, variable names and other information from the variable constant file are retrieved from the file read on Unit No. KFILCP and matched with the variables; some insertions are made for computed predictors. Each predictor ID in ID(,) is duplicated in JD(,) except that some portions are omitted; the result is called the "basic" ID. (This basic list is not defined the same way as in U201.) From the single threshold read, an upper and lower are set by subroutine THSET, provided the variables are properly ordered as input. It is noted that the predictors and predictands are not reordered, but are used as they are input. This is so that the output for printing can be done (more easily) in the order desired.

RESTRICTIONS:

It is assumed the files on unit numbers KFILP and KFILCP have been opened. RDVR66 performs basically the same function for U660 as RDVRBL does for U600, and is useful only for U660.

NONSYSTEM ROUTINES USED:

PRSID, CKIDS, THSET

LANGUAGE: FORTRAN 77

LOCATION: u660lib

INT660

INITIALIZES FROM CONTROL FILE FOR U660

Harry R. Glahn
January 1, 1998

PURPOSE: Performs most of the initialization for U660. It reads the control file U660.CN, and other input files. It also skips up to the appropriate date on the output packed file as necessary. This is just an in-line subroutine for U660 that performs basically the same function for U660 that INT600 does for U600.

RESTRICTIONS:

Is useful only for U660.

NONSYSTEM ROUTINES USED:

IOPEN, TIMPR, RDSNAM, RDI, DATPRO, RDSTGN, RDSTGA, RDVR66, SKIPWR, and IERX

LANGUAGE: FORTRAN 77

LOCATION: u660lib

U602

PERFORMS SCREENING REGRESSION ANALYSIS FOR MULTIPLE PROJECTIONS

Harry R. Glahn
March 1, 2000

PURPOSE: U602 is one in a series of programs designed to develop statistical relationships, usually regression equations, between predictands and predictors from archived data. The primary input data, point data from observations, interpolated from model fields by U201, etc., are packed in a GRIB-like TDLPACK format. Because the packed data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in a random access file; these data are also packed in the TDLPACK format. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U602 uses a driver DRU602 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U602 (actually, subroutine) for every application. U602 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. The default input and output unit numbers are set to 5 and 12, respectively, in DRU602. The output of U602 is one or more sets of regression equations prepared for evaluation by other programs. The equations are also output in ASCII for viewing or printing. The primary difference between U600 and U602 is that the predictor taus in the 4-word IDs of U602 (other than observations and geophysical variables such as terrain height) pertain to hours ± the predictand taus, while in U600 the predictor taus are those to be applied to all predictands. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U602.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU602.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U602" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U602 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U602HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U602 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). Values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU602 DATA statement = 12. Changing the default unit number allows multiple runs of U602 or other programs within the same directory without overwriting.

(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).

(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).

(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).

(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).

(6) = The variable IDs and the forced predictor IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.

(7) = The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).

(8) = The variable ID list in summary form after reordering low to high. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)

(9) = The variable list in summary form after reordering. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).

(10) = The variable ID's for the first day (day 1) as read from the archive datasets. This is just a list of all the variables on the input files.

(11) = The variable ID's of the archived data actually needed [entries in MSTORE(,)].

(12) = The list of stations on the input file(s).

(13) = Missing data diagnostics (see NOMISS in Record Type 3), diagnostics regarding elimination of predictors and predictands on basis of variance left, and forced predictors not meeting FORCUT criterion (see Record Type 3).

(14) = A diagnostic will be provided when there are no data for a particular input file. This may be misleading.

(15) = All data values in the AA(,) matrix. For each case (cycle or day), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The print is by station, then the variable values are printed in the order dealt with in U602 (see IP(9)). Except for a very few days, this would produce voluminous files.

(16) = All or a portion of the data values in the AA(,) matrix. For each case (cycle or day), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The print is by variable controlled by JP( ) (see Record Type 13 below), the station values being printed in the order dealt with in U602 (see IP(5)). Except for a very few days or a few variables, this would produce voluminous files.

(17) = Sums, standard deviations, and cross product matrices. Except for a very few variables, this would be a lot of data. Also, the print is formatted F12.2, so many cases or large variable values may cause the value to not be printable. (Subroutine PRCRO6 could be modified for another format.)
(18) = The predictor means, standard deviations, and correlations of predictands with predictors.
(19) = The predictand means and standard deviations.
(20) = All equations, and important selection diagnostics, as the predictors are being selected, will be saved on unit IP(20) for viewing. When IP(20) ≠ KFILDO, only the last equation (the one with all the predictors) will be put on unit KFILDO (unless IP(20) = KFILDO).
(21) = Summary reductions of variance and standard error of estimate.
(22) = A summary of the number of times each predictor is used in the equations.
(23) = Information concerning opening and closing of files.
(24) = Information as to where the variables are to be found--directly on input, binary computed from a previous variable, or (at least attempted to be) computed through subroutine OPTX.
(25) = The first and second max and first and second min of each variable by station group and projection, along with the associated station identifiers and dates.

For checkout, it may be advisable to set all these values to zero or the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files. If a file of the same name exists, it will be deleted when IP( ) ≠ 0 except for KFILDO. **Do not use 8, 9, 11, 45 through 49, 97, or 99 as unit numbers; they are reserved for other purposes.**

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (9(I8/),7(F8.4/),I8/F8.4/3(I8/),F8.0/I8)
Control Parameters

This group of records contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

NSKIP - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. Because of the lookahead feature required for predictands, and to maintain efficiency, date/times should not be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship.

<u>NST</u> - The (maximum) number of predictors to select.

<u>MFORCE</u> - The (maximum) number of predictors to force into the equation. The order read in (see Record Type 14) is the order to be forced. Various variance and covariance thresholds as described below apply.

<u>NSELT</u> - Selection method; there are two implemented. When NSELT = 1, the next predictor selected is the one that gives the highest additional reduction of variance (RV) with <u>any one</u> of the predictands. When NSELT = 2, the next predictor selected is the one that gives the highest <u>average</u> additional RV over ALL of the predictands. However, since an average RV is harder to achieve than an RV for a single predictand, when CUTOFF (see below) has been reached for NSELT = 2, selection reverts to NSELT = 1 for that equation development.

<u>NSMETH</u> - Stopping method. The only one implemented is using CUTOFF, FORCUT, COLN, COLNB, and COLNGB (see below). Set = 1.

<u>NECVRB</u> - The number of values of a **predictand** variable that is required for that variable to be used in an equation. A zero means that any number will be accepted. When NECVRB ≠ 0, the AA( ) matrices are saved and no cross products are calculated on pass 1 through the data. This can be 0 and the thresholds below will still apply. The purpose of NECVRB is to eliminate a predictand for a particular station (or group) when data are (largely) unavailable (e.g., one or more projections when doing simultaneous development). This keeps a variable which is largely missing for a particular station or group from deleting development for that station or group. Note that for generalized operator development, NECVRB applies to the group, not individual stations in the group. See Appendix III for more information. (See LIMIT( ), Record Type 13, for a similar control on individual **predictors**.)

<u>NECCAS</u> - The number of cases that is required for an equation to be developed. This applies to single station equations or generalized operator equations. Use 0 to disable, but not a good idea.

<u>CUTOFF</u> - For NSELT = 1 (see above), the additional fraction of the variance of one or more of the predictands that has to be explained for another predictor to be included in the equation when screening predictors. For NSELT = 2, the additional <u>average</u> fraction of the variance of all of the predictands that has to be explained for another predictor to be included in the equation. Use 0 to disable, but not a good idea. (See NSELT above)

<u>FORCUT</u> - The additional fraction of the variance of one or more of the predictands that has to be explained for another predictor to be included in the equation when forcing predictors. Use 0 to force all predictors, but not a good idea. Note that when FORCUT is > CUTOFF, the "forced" predictor could be rejected, but accepted by the normal screening process. Also note that this applies to individual predictand reductions of variance, not an average regardless of the value of NSELT. FORCUT needs to be met for only one predictand at one projection for the forced predictor to be used, unless the remaining variance is zero or the appropriate colinearity criterion is not met. Selection for any one

predictand/projection will let the variable be selected for all predictands and projections unless the variance is zero.

<u>VARNB</u> = The variance of a point binary predictor necessary for the variable to be used. Use 0 to disable. For a specific sample size, this can be directly related to the minimum number of 0's or 1's that will be tolerated. Note that MOS probabilities (CCC = 2xx) are labeled as binary (B ≠ 0), even though they are really continuous. See IDSABL( ), Record Type 13, to disable this test. See Appendix III for more information.

<u>VARNGB</u> = The variance of a grid binary predictor necessary for the variable to be used. Use 0 to disable. A grid binary is a hybrid, being not fully binary nor fully continuous, and although in some cases, 0's and 1's will dominate, the relationship between the minimum number of 0's or 1's that will be tolerated is not as directly related to VARNGB as it is for VARNB. See Appendix III for more information.

<u>COLN</u> - The acceptance threshold for colinearity of other than binary predictors. That is, a predictor will not be allowed into the equation if (1-COLN) of its variance has been explained by the predictors already selected. Use 0 to disable.

<u>COLNB</u> - The acceptance threshold for colinearity of cumulative or discrete (point) binary predictors. That is, a point binary predictor will not be allowed into the equation if (1-COLNB) of its variance has been explained by the predictors already selected. Use 0 to disable.

<u>COLNGB</u> - The acceptance threshold for colinearity of grid binary predictors. That is, a grid binary predictor will not be allowed into the equation if (1-COLNGB) of its variance has been explained by the predictors already selected. Use 0 to disable.

<u>NTOSS</u> - Elimination method for a predictor (for all predictands) for one or more of the NPROJ projections L when the predictor does not meet its covariance criterion (COLN, COLNB, or COLNGB, depending on the type of variable), which is called CRITER below. (see FTOSS below):
10 = Always toss for L and only L.
11 = Toss for L only when
- (a) it doesn't meet FTOSS*CRITER, or
- (b) it meets FTOSS*CRITER and either L-1 or L+1 does not meet CRITER.

20 = Toss for all L (L=1,NPROJ) only when
- (a) L doesn't meet FTOSS*CRITER, or
- (b) L meets FTOSS*CRITER and either L-1 or L+1 does not meet CRITER.

30 = Toss for some L only when
- (a) L doesn't meet FTOSS*CRITER
  - (1) All L (L=L,NPROJ) when an observation (CCC=7xx),
  - (2) All L (L=1,L) when MOS (CCC=2xx), or
  - (3) All L, either L=1,L or L=L,NPROJ depending on the least number of L's to be removed
- (b) L meets FTOSS*CRITER and either L-1 or L+1 does not meet CRITER, then
  - (1) All L (L=L,NPROJ) when observation (CCC=7xx), or
  - (2) All L (L=1,L) when MOS (CCC=2xx), or

(3) All L, either L=1,L or L=L,NPROJ depending on the least number of L's to be removed

Note that **for the predictor to not be tossed for projection L, its remaining variance for both** neighboring projections must meet CRITER, even if it meets FTOSS*CRITER for projection L. See Appendix IV for more details.

<u>FTOSS</u> - The fraction of CRITER (see NTOSS) that has to be met for a predictor to be kept when its projection neighbors meet the criterion CRITER.

<u>NOMISS</u> - Indicates whether (>0) or not (=0) missing data will be identified on file KFILDO [IP(13) = 0] or IP(13) [IP(13) > 0]. (See Comments section.)

<u>NEW</u> - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

<u>NALPH</u> - Indicates whether (=1) or not (=0) the call letters will be alphabetized <u>by group</u> according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 may not be useful.

<u>PXMISS</u> - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value. In U602, this must be 0 or 9999. A value of 9999 will cause a 9997 datum to be treated as missing; a value of zero will be used as such. A value of 9997 might occur in the input data if MOS forecasts were being used.

<u>MODRUN</u> - A value to be used as the identifying "model/run" number for identifying the equations (the predictands). This 2-digit value is inserted as "DD" into the first word of the ID of each predictand just before the equations are output on file Unit No. KFILEQ (see Record Type 8).

Record Type 4 - Format (I3,4XA60) <u>Date List File</u>

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

<u>KFILDT</u> - Unit number for the file containing the input date list.

<u>DATNAM</u> - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) <u>Date List</u>

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

<u>IDATE</u>(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This

input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U602 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

Record Type 6 - Format (I3,4XA60) Vector Data Input Files

This group of records identifies the data sets from which the point data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

KFILIN(J) - Unit numbers for the data files (J=1,NUMIN).

NAMIN(J) - Names of files corresponding to KFILIN(J) where these data reside (J=1,NUMIN). (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. (A complication is the necessity for the lookahead feature for predictands.) When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4XA60) Random Access Files

This group of records identifies the MOS-2000 random access data sets from which constant (and possibly other) data are read. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. If no data are needed from a random access file, only the terminator is necessary.

KFILRA(J) - Unit numbers for the random access constant files (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

RACESS(J) - Names of files of constant data matching KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) Equation Output Files

This group of records (plus the terminator record) identifies the equation output files, one per projection. Note that this is in addition to the print provided through IP( ) values. Records are read until the terminator KFILEQ( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND12. This Record Type 8 is read by subroutine RDSNAM. These files will be opened as 'NEW'. If this is an empty set, the equations will not be output.

KFILEQ(J) - Unit numbers for the output equation files (J=1,NOEQFC).

OUTNAM(J) - Names of files where this output is to reside (J=1,NOEQFC). (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) Station and Location Files

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which equation development is to be done (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used. When KFILD(1) = KFILD(2) and DIRNAM(1) = DIRMAM(2), all stations in the directory are used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (14(A8,1X)) Station List

This group of records identifies the (groups of) stations for which equation development is desired. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read within subroutine RDSTGN or RDSTGA by RDC, which eliminate any blanks found in the input. Duplicate stations in the list are kept, but a diagnostic is furnished on unit IP(5); this is not counted as an error. Note that this diagnostic applies to the total list, not just duplicates within a group. For NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty set terminates the station input. That is, the last group and its terminator must be followed by another terminator signifying an empty set. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) Variable File

This record (plus the terminator record) identifies the file from which the variable ID's (both predictors and predictands) are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI (set in DRU602), PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) Variable Constants File

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 13 - Format
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,6XI2,I2,I3,1XI6,I3)
Variable List

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 13 is read by subroutine RDVRBX. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading predictors in U201.

ID(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

JP(N) - For each variable N, JP(N) indicates print or no print when ≠ 0 or = 0, respectively, for data values at the stations. These values combined with IP(15) and IP(16) allow easy control of output. For instance, all variables could have JP( ) ≠ 0 and IP(15) ≠ 0 (or IP(16) ≠ 0) for a short diagnostic run. All such print could be turned off by setting IP(15) = 0 (or IP(16) = 0) in this control file. Alternatively, IP(15) (or IP(16)) could be ≠ 0 and selected data obtained by changing JP( ) from 0 to ≠ 0. (See LTAU( ) below.)

IDSABL(N) - Indicates for each variable N whether (=1) or not (=0) the VARNB, VARNGB, COLNB, and COLNGB are disabled and the variable (designated as binary by B ≠ 0) is treated as continuous. This is important for MOS probabilities that are labeled as binary but are really continuous and have a very small variance.

NTYPVR(N) - The type of variable:
1 = predictor.
2 = predictand.
NTYPVR determines how the tau in LTAU( ) (see below) is used in the variable ID. For predictors other than observations, the tau is added to each individual predictand tau to get a predictor tau. That is, the specified predictor

tau is $\pm$ that for the predictands. Note that if more than one predictand tau is specified, more than one predictor variable will be created. Then, that predictor ID (including the computed tau) must be available. For predictands, the lookahead feature must be used, so tau is added to the current date NDATE to get the date for which the variable is needed at tau = 0. As a special feature, if a predictor is an observation (CCC between 700 and 799 inclusive) and not from the AEV data archive (DD = 82), then tau is treated as lookahead as it is with predictands. This allows an observation to be used at a time after, but not before, the model cycle run time.

LIMIT(N,L) - The number of **zeros or of ones** necessary for a binary **predictand** to have for an equation developed for it. This means that both AVG(N,L)*SMPL and (1-AVG(N,L))*SMPL must be $\geq$ LIMIT(N,L) for predictand N, where AVG(N,L) is the binary predictand average and SMPL is the sample size. This can vary by projection L. LIMIT(,) is also used to test the **total** number of **predictor** cases, but does not vary by projection. Note that NECVRB (see Record Type 3) must be > 0 for LIMIT( ) to apply to predictors. See Appendix III for more information.

LTAU(N) - The $\pm$ tau that is to be used for this predictor in relation to each predictand tau (see NTYPVR), unless the predictor is an observation. When the variable is an observation or a predictand, LTAU( ) is not used. As a special feature, to allow results compatible with U600, when LTAU( ) = 999, the tau in the predictor ID is left intact. Note that an observation prior to run time is not allowed, but after run time is allowed.

Record Type 14 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3) Forced Predictors

This group of records contains the variable ID's of the forced predictors. Records are read until the terminator IDFORC(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 14 is read by subroutine RDFORC. Upon completion of reading this record type, N=1,NFORCE. Note that the format and information for IDFORC(,) is exactly the same as for ID(,) above, up to the point needed. Since some predictors may be the same except for tau, use the tau for the 1st projection of the predictand; this forces for all predictands.

IDFORC(J,N) - The first 3 (J=1,3) words of the predictor ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed. This is the exact format for variable ID's as for Record Type 13.

Note that if the number of (useable) predictors read for forcing does not match MFORCE read in Record Type 3, the number actually read (and not tossed for some reason) will be used. This is flagged as an error to give the user feedback and positive control on the (number of) forced predictors.

Record Type 15 - Format (I9,1X,I9,1X,I9,1X,I3,1X,A1,1X,I4,1X,I3)
Stratification Predictors

This group of records contains the ID's of the stratification predictors and the associated variables. Records are read until the terminator IDSTRT(1, ,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND11. This Record Type 15 is read by subroutine RDSTRT. Upon completion of reading this record type, N=1,NSTRAT. Note that the format and information for IDSTRT(, ,) is exactly the same as for ID(,), Record Types 13 and 14, up to the point needed.

IDSTRT(J,M,N) - For M = 1, the first 3 (J=1,3) words of the predictor ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed. This is the exact format for variable ID's as for Record Type 12. IDSTRT(,1,) must appear in the variable list ID(,).

For M=2,max of 6, the associated variables needed to stratify the variable IDSTRT(,1,). The format is that same as above. The variable for M=2 is the "base" variable (i.e., the variable that is to be stratified), and the variables M=3 and above are the binary variables used for stratification; they can have values of B = 1 or 2. The list of associated variables is terminated by 999999. Note that B = 2 is not allowed for a predictor, but is allowed for an associated variable.

The group of stratification variables and associated variables as described above can be repeated up to ND11 times. A final terminator is needed. That is, this Record Type 15 must be terminated by two terminators = 999999, one to signify the end of the particular stratification and associated variables and one to signify no more stratification variables.

The stratification variable (M = 1 above) must match one of the variables in the extended predictor list, which is the list read in Record Type 13 with projections added. Therefore, a projection must be in IDSTRT(,1,) that matches one of the projections to be used. Usually a "1" will suffice.

See Appendix II for more information on the use of stratification variables.

CONTROL FILE INPUT: (Name read from U602.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) Date List

When the dates are not provided in file U602.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU602), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U602 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U602.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) Station List

When the station list is not provided in file U602.CN, this group of records identifies the stations (or locations) to be included in the regression analysis. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Types 9 and 10, control file 'U602.CN', for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U602.CN) (Unit = KFILD(2))

Record Type 1 - Format Station Locations
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which regression analysis is to be done. The call letters read from KFILD(2) (see Record Type 9) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical no matter the order read in (see NALPH in Record Type 3). (Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run.) However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. For example, there is no use of the elevation in the equation derivation. However, this information is available for a computation routine, if one is needed. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3).

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited, except when it also constitutes the list to be kept (see Record Type 9), in which case the number of entries is limited by ND1-1. No terminator is used.

CONTROL FILE INPUT: (Name read from U602.CN) (Unit = KFILP)

Record Type 1 - Format
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,6XI2,I2,I3,1XI6,I3)
Variable List

When the variables to use in the run are not in file U602.CN, this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDVRBX. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading predictors in U201. See Control File Input for U602.CN, Record Type 13, unit KFILDI above for more detailed information; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U602.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32) Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U602 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

Other processing occurs with the reading of these records in RDVRBX, much of it associated with the plain language description. See Chapter 4 in TDL Office Note 00-1, Variable Definition, for details. As new variables are defined, this file may need updating.

DATA INPUT:

All data interpolated from models and produced by U201 for input to U602 and all other vector data will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note 00-1 MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format, except for the call letters (directory) record. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U602.CN in Record Types 6 and 7.

A. SEQUENTIAL VECTOR DATA

One or more sources of vector data, each probably prepared by U201, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U602.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records within a file, as might exist on an hourly data archive file, are accommodated. Observation records are expected to have the date of the observations and a tau of 0.

B. RANDOM ACCESS VECTOR DATA

Up to 5 sources of random access vector data are accommodated; however, it is unlikely more than 1 or 2 will be needed. These data exist in the MOS-2000 External Random Access File System. They are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

DATA OUTPUT

All equations are placed in one or more ASCII files, on Unit No. KFILEQ (see Record Type 8), separated by projection, that can be easily edited, especially for the stations to apply the equations to, and used directly by other programs such as U700 and U900 (see "MOS Equations" in TDL Office Note 00-1 MOS-2000). Note that the predictand IDs have been modified so that the 2-digit value MODRUN (see Record Type 3) has been inserted as "DD" into the first word. This allows identification of the model or models on which the equations were based, or even a different development run on the same model (e.g., for primary and backup equations). Observations (CCC in the range 700-799) will have a tau [IDPARS(12)] representing how far ahead to get the observation. That is, tau is added to

NDATE, then the record read with a tau = 0. (In U700 or U900, the "predictand" IDs will be modified to "forecast" IDs.)

Each projection for which development is done will comprise a separate "set" of equations for U700 and will be put on a separate file. That is, if projections 1 through 12 are being developed in one run, there will be 12 output equation files on unit numbers and names as read in Record Type 8.

EXAMPLE CONTROL FILE: 'U602.CN'

An example exists as file 'U602.CN' in directory home21.glahn.dru602 on blizzard. The easiest way to set up a run is to take an existing control file and modify it.

OUTPUT:

Output that can be printed can be put onto one or more files as described above under control file 'U602.CN', Record Type 1 and the definition of KFILDO in the driver DRU602. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The IBM treats them as Comments.

The binary indicator "B" in the first word of a predictor ID must be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), or 5 (indicating a grid binary). For a predictand, "B" can also take the value of 2 (indicating a cumulative variable from below) or 3 (indicating a discrete variable). For the latter, the upper threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID's. If there is no lower one, the lower threshold is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999. Subroutine CKIDS called from RDVRBX prints a diagnostic if B for a predictor is not 0, 1, or 5 or for a predictand is not 0, 1, 2, 3, or 5. However, the variable is not eliminated.

Especially because of the lookahead feature necessary for predictands (and possibly for observations as predictors), all data needed for Day 1 should actually be available (e.g., there be a data record for all predictands, no matter how far ahead, for Day 1) **when the variable is to be computed through subroutine OPTX**. (There may also be cases when this restriction is also true for variables not computed through OPTX.) This does not mean that there cannot be a station with "missing" data. Also, each variable should be accessed on the same unit number. That is, if two seasons of data are on different files, they must have the same unit number.

Generally, it is not necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles, unless the variable is computed through OPTX from other variables not otherwise used as a "raw" variable. That is, a raw (not computed) variable need not be present for Day 1, but one used only in computations must be, because U602 has no way of knowing it will be needed for future cycles. If a variable is not available for Day 1, more storing of data on each subsequent cycle may occur.

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use (Read only except No. 49)
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

U602 might use Unit No. 45 or even 48 or 49.

COMMENTS

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U602 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular variable category. In this latter case, the event is interpreted as having the value of PXMISS (see Record Type 3). Normally, PXMISS would be zero. Otherwise, to interpret the value as 9999, the case would have to be thrown out. Presumably, the only time 9997 will occur is when operational forecasts from U900 or U910 or test forecasts from U700 or U710 are input. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such. These values will be used the same way no matter how they are packed.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually not be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired (see IP(13)). This probably means a set of dates should be supplied to U602 for which it is known there are good data; then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

On Day 1, GFETCH is entered directly for every predictor (except possibly a point binary) because it is not yet known when OPTX must be entered. ("Day 1" is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) is not counted as an error in VRBL63 for Day 1. It is counted as an error in VRBL64 (through OPTX), because GFETCH should not be entered directly when OPTX is needed.

When a point binary is needed, it will be computed in U602 if possible (i.e., the "basic" variable is available) even though it may exist on an input file. For a second generation MOS, the input variable may be a forecast of a binary variable (CCC = 2xx; B = 1, 2, or 3). In this case, the basic variable will not be available, and the variable is obtained directly from input. Even though designated as binary, this is really a continuous variable, being a probability of a binary event; it will not be further "binaried." However, other computed predictors, such as sine and cosine functions are used from the input file if they exist, even though they might be computable in U602. For strictly mathematical variables like sine and cosine, it is likely more efficient to do the computation in U602. Only a point binary can be computed in U602 without entering either OPTX or STRAT.

The "interpolated" data input(s) can contain constant data taken from the MOS-2000 External Random Access File System by U201. Alternatively, U602 can take constant data directly from MOS-2000 External Random Access files. It is possible that no interpolated data on sequential files be used, and all input furnished be on "constant" random access files.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first few (currently hardwired to 5) days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, more disk access may be required than necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage after Day 1. (With modern system swapping and cashing techniques, this seems to not be very important.)

It is unlikely that when no predictor in a run has a particular model number, that the file containing that model will be needed; therefore, the diagnostic to this effect should be heeded, and that model's file(s) should not be used because it (they) will be read during the run even though not needed. An exception is for a subroutine that would use data from a model that was not indicated by the model number in the predictor ID. Even though the possibility is remote, that is the reason that the file is not automatically closed.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers or (old) call letters are furnished in the Record Type 10 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

Equations can be output on either the default output unit number KFILDO or the unit number specified in IP(20), or both. An option is available for writing all the equations as each individual predictor is selected or just the final equation with all predictors. The table below shows the options. The plain language is written as each predictor is selected. The plain language is also written with the IDs in the equation, except for the first which would just be a repeat of the "selected" predictor.

Default Output Unit No.	Value of IP(20)	Equations on Default Output	Equations on IP(20)
12	12	All	(Same as default)
12	0	Last Equation	None
12	20 (say)	Last Equation	All

The two maxs and two mins of each variable are found in subroutine XMXMN6, which uses dynamically allocated storage. This storage can be quite large, and there could be difficulty in allocating space. This should not abort the program; the maxs and mins would just not be found and printed, but rather a diagnostic output to KFILDO.

To recap about how the projections are used. This can be a little confusing. For predictands (NTYPVR = 2), the tau in the 3rd word of the ID is used as a lookahead feature, just as with other MOS vector programs. This is also true for observations (CCC = 7xx), and allows an observation to be used future to the model cycle time. LTAU( ) is not used for predictands or observations as predictors. However, for (other) predictors, the LTAU( ) read (the last value in the record with the ID) is used to indicate the number of hours plus or minus each predictand projection time. For instance, if a predictand tau is 24 hours, a predictor LTAU( ) would likely be zero to indicate the model predictor would have a projection of 24 hours. However, if a -3 were used for LTAU( ), then the model predictor would have a projection of 21 hours. Any number of these can be used together, and they all apply to each predictand projection.

See Appendix I for more information on the use of predictors.

SETTING UP THE DRIVER DRU602

The preparation of the driver for a particular U602 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with in the regression analysis. Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) unless, of course, the station directory is to be used as the station list.
ND2 -	The size of the work area Q( ). This holds the sample size(s), sums, and sums of cross-products (in the non-redundant portion of the matrix). A test run of a day or so will provide guidance on the value of ND2 that would be appropriate for a similar run on many days. If more than one set of equations is to be developed (e.g., more than one "region" or single station), Q( ) must be large enough to handle one such set. If it is not large enough to handle all such sets at one time, more than one pass through the data will be required and the run will be much slower.

ND3 - Maximum number of projections.

ND4 - The maximum number of variables (predictors and predictands) per projection that can be dealt with. If different projections have different numbers of predictands, this is the maximum.

ND5 - Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.

ND6 - Maximum number of all sequential file input sources (e.g., models) that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records read for the date/time of Day 1 on all input files, including the predictands at future date/times.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number may result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

ND11 - The maximum number of stratification variables that can be used.

ND12 - The maximum number of output equation files that can be used.

In no case, should an NDx be < 1.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

WRITING NEW SUBROUTINES

It is not expected that computational routines will be used to any great extent in U602. Rather, computations, except for point binaries, are expected to be done in U201. Any computations that would be done in U602 would have to be done on point data (not gridpoint data) because gridpoint data are not accommodated in U602. Even so, an "option" subroutine, OPTX, is provided for possible use. (In the case of development for gridpoints, the gridpoints are given identifications in the same way as other locations, such as stations.)

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib.dru602, dataset 'u602.com', along with the libraries 'home21.tdllib.u602lib', 'home21.tdllib.u600lib', and 'home21.tdllib.moslib', all on blizzard, and loaded in that order.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: home21/tdllib/u602lib. The driver is in dru602.

APPENDIX I

Use of Variables in U602

The matching of variables, predictors with predictands, in U602 has to be done carefully, partly because there are several types of predictors:

1) Observations,
2) LAMP output,
3) MOS forecasts,
4) Constants, and
5) Model data (e.g., AVN, RUC)

This is also a bit more complicated because the predictors "march" with the predictand(s) as appropriate.

Only those predictors that start as gridded data (e.g., model data, LAMP gridded data, and possibly constants) plus the MOS forecasts need be run through U201 first. That is, LAMP <u>gridded</u> output and other model data must be interpolated to stations and put into vector form. If the constants (e.g., terrain) exist as grids, then this must be done for them also. The MOS forecasts are initially in vector form, but the lookback feature must be used in U201 to get a model run prior to the LAMP run time. Also, MOS forecasts are generally at 3-hourly projections, and if forecsts at other hours are needed, time interpolation must be done. In the table below, "RR" is the value of RR in U201. For a second generation LAMP (e.g., LAMP MOS temperature forecasts input for a ceiling predictand), this should not be necessary if the LAMP run times are the same (i.e., the run producing the temperature and ceiling forecasts are from the same start time). For MOS, or for any variable, U201 will output the date/time specified in the date/time list, even if the forecasts came from a previous date/time (RR > 0); the taus will not be modified.

Observations and vector constants do not have to go through U201. Also, quality controlled obs from LAMP can be used directly.

The discussion below refers to U602 input from any source.

Start with the predictand. You will have one or more weather variables in a U602 run (e.g., wind speed, u, v; temp, dp; clouds; etc.) and each will have multiple projections (taus). NTYPVR = 2. For these, the tau (IDPARS(12)) is the projection from the run time (run time is start time; the "persistence" ob time). If you are doing 2 weather variables, and projections 1 to 15 hours, you would have 30 variables with NTYPVR = 2. LTAU is not used.

Now consider the initial observation (CCC = 7xx, DD = 0) as a predictor. The tau will indicate the time of the observation <u>relative to the start time</u>. LTAU is not used. So, tau would normally be 0. (If one wanted for a test to see what would happen if we used the LAMP run at, say, 05Z, and an observation at 06Z, then tau would be 1. I don't see why this would be useful except to satisfy curiosity.)

Now consider a LAMP output variable as a predictor (CCC = 0xx, DD = 05). Use tau = 0 and LTAU = 0 for the LAMP forecasts to march and be valid at the same time as the predictand. This should be our first cut. If you wanted the LAMP forecasts to be valid 1 hour previous to the predictand, use LTAU = -1. In this case, for tau = 1, the LAMP predictor would be at start time. If LTAU = -2, then the predictor tau would be negative, which is not allowed; U602 will adjust it to 0. More than one LTAU can be used with the otherwise same predictor. This might be important if the LAMP was generally too fast or too slow. However, if multiple LTAUs are used and selected, they will tend to "water down" the timing aspect (essentially smoothing); we want to avoid this as much as possible. Also, predictors with consecutive taus will be highly correlated with each other and multiple taus will tend to not get selected, or if they are, may lead to instability. Don't let a very small increase in RV determine the inclusion of more than one LTAU. As a special feature, if LTAU = 999, the predictor does not march, but tau is used as a constant. Then with tau = 0, the initial LAMP variable could be used for all predictand projections (like an initial ob).

Now consider a model forecast (CCC = 0xx, DD = 08 for AVN). It works the same way, except the model run time has to be considered. This is indicated below as RR. Subtracting RR from the LAMP run time gives the model run time. The direct input from a model is to be severely minimized so as to make this an <u>update</u> system, so as to not "compete" with MOS at the longest projection. Note that for each LAMP start time, U201 will have to be run with the appropriate RR, and U602 will have a different tau to march in lockstep with the LAMP projection. For instance, if 00Z AVN is used for LAMP 05Z, then for U201 RR = 5, and for U602, tau = 5.

Now consider the MOS forecasts (CCC = 2xx, DD = 00). This works in the same way as model forecasts above. The RR for MOS and model forecasts would normally be the same. For second generation LAMP MOS forecasts, the same process works, with RR normally being 0, and U201 not being needed.

Now consider a constant as a predictor (CCC = 4xx, DD = 0). This is <u>really</u> a constant, and tau = 0 and LTAU = 0.

Now consider the harmonics of the day of the year (subroutine FORIER) as a special case of constants (CCC = 010, DD = 0). LTAU is not used. In order to give the harmonic the best chance of being selected, and also of not needing both components of the same harmonic, the offset "Y" in days can be used; this is UUUU in IDPARS(7) of ID(2). For instance, for Y = 20, the minimum of the cosine is at 20 days instead of 0 days. This seems to make sense for temperature, and maybe for other variables. Getting <u>exactly</u> the right value is not important and cannot be known anyway; it is unlikely it would vary by predictand. I expect only values of Y = approximately 20 or 0 will be needed, <u>and not both</u> for the same predictand. For LAMP where the projections are only a few hours from the start time, tau = IDPARS(12) would be 0 and FFF = 201-204, although Y can be used in MOS with FFF = 205-208 for larger projections to key the harmonic to the date of the predictand rather than the start time. Note, however, that in U602 a different variable will be computed for each tau, so that is more computation than necessary. If the projection is large, then possibly two values of Y could be used, say 20 and 30. (For U600, this capability does not pose the "marching" problem and too much computation.)

A summary follows, where:

Y = IDPARS(7)
RR = IDPARS(9)
h = IDPARS(12)

```
Variable ID NTYPVR tau LTAU  Run Date/Time  Variable        tau
CCCFFF   DD        (+) (±)   Date    Cycle  Date/time    (relative to ddd)


7xxxxx   00   2    ddd (00)  yyyymmddcc     yyyymmddcc+ddd   0
  obs
7xxxxx   00   1    ttt (00)  yyyymmddcc     yyyymmddcc+ttt   0
  obs
```

0xxxxx	05	1	(0)	hh	yyyymmddcc	yyyymmddcc	hh	
LAMP								
0xxxxx	05	1	h	999	yyyymmddcc	yyyymmddcc	h	(no march)
LAMP								
0xxxxx	09	1	(0)	hh	yyyymmddcc	yyyymmddcc-**RR**	hh+**RR**	
NCEP								
0xxxxx	09	1	h	999	yyyymmddcc	yyyymmddcc-**RR**	h+**RR**	(no march)
NCEP								
2xxxxx	00	1	(0)	hh	yyyymmddcc	yyyymmddcc-**RR**	hh+**RR**	
MOS								
2xxxxx	00	1	h	999	yyyymmddcc	yyyymmddcc-**RR**	h+**RR**	(no march)
MOS								
4xxxxx	00	1	(0)	hh	yyyymmddcc	yyyymmddcc	hh	(constant)
GEO								
4xxxxx	00	1	(0)	999	yyyymmddcc	yyyymmddcc	0	(no march)
GEO								
010xxx	00	1	h	n/a	yyyymmddcc	yyyymmddcc	0	(DOY)
Sin/Cos						xxx = 201-204		
010xxx	00	1	h	n/a	yyyymmddcc	yyyymmddcc+**Y**	0	(DOY + **Y**)
Sin/Cos						xxx = 201-204		
010xxx	00	1	h	n/a	yyyymmddcc	yyyymmddcc	h	(DOY + h)
Sin/Cos						xxx = 205-208		
010xxx	00	1	h	n/a	yyyymmddcc	yyyymmddcc+**Y**	h	(DOY + **Y** + h)
Sin/Cos						xxx = 205-208		

MOS IDs will have RR in them as used in U201.

Use Model No. = 0 for MOS forecast input file (for vector data), even though the 2xx IDs will have DD = 8 for AVN forecasts.

Full use of 4xxxxx constants requires Option 2 definition in ON 00-1, Chapter 14.

APPENDIX II

USE OF STRATIFICATION VARIABLES

STRAT is the subroutine that computes the stratification variables. A stratification variable is indicated by CCC=9xx and is in the ID(,) list. Since that variable will likely not be available on input, it must be computed. Before OPTX is entered, a check is made, and when CCC=9xx, STRAT is entered instead. A variable in IDSTRT(,1,N) must match the variable in the ID(,) list. That variable is found, and the variable to stratify is IDSTRT(,2,N). The variables indicated by IDSTRT(,M,N) (M > 2) are binary variables with B = 1 or 2 used to stratify IDSTRT(,2,N). Essentially, IDSTRT(,1,N) = IDSTRT(,2,N)*IDSTRT(,M,N), where M goes from 3 through a maximum of 6. By using M = 3 and 4 with appropriate thresholds, a stratification variable can be created with a desired range.

For instance, let T(L) = a + b*T(M) + c*T(O), where
T(L) is the predicted temperature at projection L,
T(M) is the MOS temperature at projection L, and
T(O) is the temperature observation.

Then form a stratification variable P(O) from the observation indicating precipitation [P(O)=1] or not [P(O)=0)], and develop the equation:

$$T(L) = a + b*T(M) + c*T(O) + d*P(O) + e*P(O)*T(M) + f*P(O)*T(O)$$

Then when P(O) = 1 (precip case), the equation becomes:

$$T(L) = (a + d) + (b + e)*T(M) + (c + f)*T(O)$$

and when P(O) = 0 (the no precip case), the equation becomes:

$$T(L) = a + b*T(M) + c*T(O)$$

For "full" stratification, each "normal" predictor in the equation can be multiplied by the stratification predictor. For several normal predictors, this will generate a lot of terms. It may be "partial" stratification (not all terms multiplied by the stratification variable) is sufficient to capture most of the information.

A basic MOS predictor (e.g., CCCFFF = 202020) with appropriate projections will "march" correctly.

The use of stratification variables can be tricky. To recap, here are some guidelines.

Consider the "stratification" to consist of:

1) A "stratification variable" used to identify the predictor,

2) A "base variable" which is the variable to be stratified, and

3) A "stratifying variable" which is a binary used to stratify the base variable.

There can be more than one stratifying variable (up to 4). Each will be a binary and normally the binary would be made within U602, but could come from the input. If there is more than one, they operate as "AND". That is, each must be a "1" for the base variable to retain its value in the stratification variable; otherwise the stratification variable is zero.

The stratification variable must match a variable in the input variable list. The only tricky thing here is that the projection must be one that will occur in the expanded predicator list. Therefore, the variable will normally have a tau = 1, although the variable in the predictor list will have a projection of zero. (This is the same as with forced predictors.) However, for MOS forecasts, the projection will probably be RR + 1 (see the example below).

The ID of the stratification variable is somewhat arbitrary, but must start with a 9. Consider the CCCFFF of the base variable to be CCC = C1 C2 C3, FFF = F1 F2 F3, then the convention for the stratification variable is CCC = 9 C1 C2, FFF = C3 F1 F2 to the extent possible.

The base variable can be an ob (e.g., 702000) or some other constant, or it can be a marching forecast (e.g., 202020008 0 007000008 0 for MOS temperature from the 0000 GMT GSM model valid 1 hour after 0700 GMT).

The stratifying variable will have a threshold that is used to make the stratifying binary. It can be a constant (e.g., an observation, tau = 0) or a marching forecast (e.g., 008311105 for LAMP clouds). The tau would be zero, and is adjusted in U602 to match the predictand tau.

An example of the base variable observed temperature stratified by LAMP clouds is:

```
970200000 000000000 000000001       000  TEMPERATURE            X
702000000 000000000 007000008       000  TEMPERATURE
008311105 000000000 000000000  550001400  UNCOMP TOTAL SKY          B
```

Another example where the base variable is the MOS temperature stratified by LAMP clouds is:

```
920202000 000000000 007000008       000  AVN MOS SFC TEMP (CHK)    X
202020008 000000000 007000008       000  AVN MOS SFC TEMP (CHK)
008311105 000000000 000000000  550001400  UNCOMP TOTAL SKY          B
```

The theory of stratification is that the base variable will have a different effect on the predictand when the binary is on than when it is off. In the above example, the observed temperature and the MOS temperature may be weighted differently when LAMP forecasts cloudy vs not cloudy. It is expected this effect would be more operative at short projections. Instead of the stratifying variable being LAMP clouds, it could be observed clouds.

CAUTION:

The stratification capability was written for LAMP and it is assumed in RDSTRT that the maximum projection is 30. Therefore, it may not work for the general MOS development.

VERIFICATION:

U602 uses the subroutine STRAT in u602lib to do the stratification. U700 uses the subroutine STRATVAR in moslib through OPTX. STRATVAR has to be set up specific to a particular stratification. Great care must be taken to make sure this is done correctly. For instance, GFETCH can be compiled with the /D option and the ftn12 viewed to make sure the correct variables are being fetched.

OPERATIONS:

It is recommended that if stratification variables are to be used in other than an experimental mode, they be computed though U201 subroutine(s) so that U602, U700, and U900 will all be assured of having the data computed in the same way. The stratification feature was put into

U602 to make experimentation relatively easy; it is more difficult to make the forecasts through U700.

APPENDIX III

Explanation of Throwout Criteria

VARNB	Used in U602 to test the number of point binary **predictor** 0's and 1's. A predictor will not be used if it does not have a minimum number of 0's and 1's calculated from the sample size and the average of the variable.
VARNGB	Used in U602 to test number of grid binary **predictor** 0's and 1's. A predictor will not be used if it does not have a minimum number of 0's and 1's calculated from the sample size and the average of the variable. This is not an exact calculation, because a grid binary is a hybrid. If most values are 0 or 1, the approximation would be good. Otherwise, the calculation will err on the side of keeping the predictor.
NECVRB	Used in U602 to retain all matrices before calculating the cross products. The raw data are necessary to compute the number of non missing cases.
	Used in VRBVE6 to test the number of **total predictand** cases for each projection. The total number of non-missing cases must be $\geq$ NECVRB for it to have an equation developed for it.
LIMIT(,)	Used in U602 to test the number of 0's and 1's of a point or grid binary **predictand**. The calculation for a grid binary is approximate.
	Used in VRBVE6 to test the **total number of predictor** cases. The number of non-missing cases, by variable M and by projection L, must be $\geq$ LIMIT(M,L) for predictor M to be used for projection L.

APPENDIX IV

Use of NTOSS and FTOSS

U625

INVENTORIES EQUATION FILES READ BY U700 AND U900

Darren Drewry
January 28, 2000

PURPOSE: U625 inventories equation files created by U600 or U602 and later used by U700 or U900. Optional output includes (1) a complete listing of the stations, by call letters, contained in the equation file(s), (2) a listing of those call letters with their corresponding station names, (3) a listing of stations by group (region) as they appear in the equation file, (4) a listing of the stations, by call letters, which are duplicated in the equation file, (5) a listing of the stations which appear in the Master Station List (if one is provided) and not in the equation file, as well as a list of the stations appearing in the equation file which are not found in the Master Station List, (6) a list of the stations and predictands for which there are no equations, (7) a list of the unique predictors in the equation files, (8) a list in U201 input form of those unique predictors that are not in an input list, and (9) a copy of the input equation file(s) sans the duplicate stations found in the input equation file(s) (assuming duplicate stations were present). Through values set by the user in the control file, a choice can be made between ICAO call letters and those found in the equation file(s). The user can also choose to have the station lists sorted alphabetically by call letters by specifying this in the control file. U625 uses a driver DRU625 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U625 for every application. U625 is written to run on a 32-bit or a 64-bit word-length machine, which is also controlled by PARAMETER statements in the driver.

Many of the design features, subroutines, and variable names used in U625 were adapted from program U660. Some familiarity with other MOS-2000 documents, especially those pertaining to U660, will be helpful for a full understanding of this writeup.

CONTROL FILE INPUT: 'U625.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the “default” output file number. KFILDI is the input unit number as specified in DRU625.

IPINIT - 4 characters, usually a user’s initials plus a run number, to append to “U625" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U625 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = ‘HRG2' and IP(5) = 40, the file name for Unit No. 40 = ‘U625HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U625 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This capability essentially allows separation of diagnostic and other types of information. Diagnostics may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). Nine values have been identified as indicated for values of J below:

(1) =	All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU625 DATA statement = 12. Changing the default unit number allows multiple runs of U625 or other programs within the same directory without overwriting.
(4) =	A complete listing of all of the stations, by call letters, appearing in the equation file. The call letters will appear 10 per line.
(5) =	A listing of the station call letters and their associated names, one per line.
(6) =	A regionalized listing of the stations by call letters. Each region is numbered.
(7) =	A listing of the stations, by call letters, that were duplicated in the equation file.
(8) =	Listings of the stations which were found in the master station list (if one was provided), but not found in the equation file. Also, a listing of the stations which were found in the equation file, but which were missing from the master station list. If no master station list is input, then IP(8) can not be output.
(9) =	The stations and predictands for which there are no equations.
(10) =	The variables read from Unit No. KFILM and the unique predictors in the equations for which there is not a matching variable in the list read from Unit KFILM.
(15) =	List of unique predictors.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (I10/I10/I10) Control Parameters

This record contains three control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

NALPH - When 1, all station list output will be alphabetized, including those stations written out to KFILEO(J)(see Record Type 5). When 0, all station list output will appear in the order in which it appeared in the equation file.

NEW - When 1, all station call letter output will be done by using the new ICAO call letters. When 0, all station call letter output will be done by using those call letters which appear in the equation file.

<u>NDATE</u> - For the case of equation files to be used in U900, NDATE specifies the date/time being processed. In this case, NDATE must be input as YYMMDDHH. This is a date/time for which the equations are valid. For equation files to be used in U700, NDATE must be '9999'.

Record Type 4 - Format (I3,4X,A60) <u>Input Equation File(s)</u>

These records (plus the terminator record) identify the equation files which will be used as input. Records are read until the terminator KFILEQ( ) = 99 is reached. Maximum number of records, sans the terminator record, is ND11 (ND11 is set in the driver DRU625). This Record Type 4 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

<u>KFILEQ(J)</u> - Unit number(s) for the input equation file(s).

<u>EQNNAM(J)</u> - Name(s) of these equation file(s). (CHARACTER*60)

Record Type 5 - Format (I3,4X,A60) <u>Modified Output Equation File(s)</u>

These records (plus the terminator record) identify the output modified equation files that contain the original equations sans the duplicate stations. Records are read until the terminator KFILEO( ) = 99 is reached. Maximum number of records, sans the terminator record, is ND11 (ND11 is set in the driver DRU625). This Record Type 5 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMOUT.

<u>KFILEO(J)</u> - Unit number(s) for the modified output equation file(s).

<u>EQNNAMO(J)</u> -Name(s) of these equation file(s). (CHARACTER*60)

Record Type 6 - Format (I3,4X,A60) <u>Station and Location Files</u>

This pair of records (plus the terminator records) identifies the file(s) which hold station identification and location information. Each record is read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 6 is read by two calls to subroutine RDSNAM. Upon completion of reading this record type, J=1,2 (see KFILD( ) below). <u>Note that a terminator is necessary for each of the call letters and location files</u>; this is different from other MOS-2000 programs because, in this case, KFILD(1) is optional.

<u>KFILD</u>(J) - Unit number for the file containing the station call letters for which equations are to be matched (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used. The station list read on KFILD(1) is optional; if it is not to be used, just use the terminator.

<u>DIRNAM</u>(J) - Name of file matching KFILD(J) (J=1,2). When KFILD(1) = KFILDI, DIRNAM(1) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 7 - Format (I3,4X,A60) <u>Predictor Constants File</u>

This record identifies the file which holds the predictor constant list used in the call to ALIST which determines the unique predictors. Records are read until the terminator = 99 is reached. Maximum number of records, sans the terminator record, = 1. Record Type 7 is read by subroutine RDSNAM.

KFILCP - Unit number of the predictor constant file.

CONNAM - Name of the predictor constant file. (CHARACTER*60)

Record Type 8 - Format (I3,4X,A60) U201 Predictor List

This record identifies the file which holds a predictor list in U201 format. This list is used to determine which of the unique predictors in the equations are not to be written to Unit No. IP(10). Records are read until the terminator = 99 is reached. Maximum number of records, sans the terminator record, = 1. Record Type 8 is read by subroutine RDSNAM.

KFILM - Unit number of the U201 predictor file.

CMNAM - Name of the U201 predictor file. (CHARACTER*60)

CONTROL FILE INPUT: (Name read from U625.CN) (Unit = KFILEQ(J))

Record Type 1 Input Equation File(s)

This group of records contains the input equations. The maximum number of files (sets of forecasts) is ND11 (see Record Type 4). The format of each set is exactly in the format output by U600 or U602. Each set must end with the terminator 99999999. See TDL Office Note 00-1 "MOS-2000," Chapter 15, for more information.

CONTROL FILE INPUT: (Name read from U625.CN) (Unit = KFILEO(J))

Record Type 1 Modified Output Equation File(s)

This group of records contains the modified equations sans the duplicate stations. The maximum number of files (sets of forecasts) is ND11 (see Record Type 5). The format of each set is exactly in the format output by U600 or U602. Each set must end with the terminator 99999999. See TDL Office Note 00-1 "MOS-2000," Chapter 15, for more information.

CONTROL FILE INPUT: (Name read from U625.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) Station List

This group of records identifies the stations (or locations) to be matched with the stations identified in the equations. Note that KFILD(1) can be the same as KFILDI; in this case, the call letters are read from the default input, KFILDI. Also, KFILD(1) can equal 0 (only the terminator read in Record Type 5), in which case this list is not read or used.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) to match with stations in the equation file (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is

ND1. (See Record Type 6, control file 'U625.CN', unit KFILDI for additional information.) (CHARACTER*8)

<u>CONTROL FILE INPUT</u>: (Name read from U625.CN) (Unit = KFILD(2))

Record Type 1 - Format <u>Station Locations</u>

(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) whose call letters were read from KFILD(1) when KFILD(1) is not equal to 0. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical no matter the order read in (see NALPH in Record Type 3). (Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3). This file is needed even when KFILD(1) = 0.

<u>CCALLD</u>(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

<u>NAME</u>(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

<u>NELEV</u>(K) - Elevation of the station. Format is I5.

<u>SIGNLA</u> - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

<u>XLAT</u> - Latitude in degrees. Format is F7.4.

<u>SIGNLO</u> - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

<u>LONDD</u> - Longitude in degrees west. Format is F8.4.

<u>CCALLD</u>(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

<u>IWBAN</u>(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited. No terminator is used.

CONTROL FILE INPUT: (Name read from U625.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32) Predictor Constants

This group of records contains information about the predictors in the equations. This file should be universally useable by all U625 users, and is expected to be a separate file. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

Other processing occurs with the reading of these records in RDVRBL, much of it associated with the plain language description. See Chapter 4 in TDL Office Note 00-1, Variable Definition, for details.

CONTROL FILE INPUT: (Name read from U625.CN) (Unit = KFILM)

Record Type 1 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3) Matching Variables

This group of records is a list of variables in the same format as read by U201. It is used to match with the unique predictors in the equations. The list read is printed to IP(10) and the unique predictors in the equations that are NOT in the U201-like list are output.

JD(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only) (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

DATA INPUT

All data input to U625, other than control information, will be in the form of equation files created for use by U700 or U900 (see TDL Office Note 00-1 "MOS-2000" for a description of these two slightly different formats). Multiple equation files, of one single type (U700 or U900), are accommodated, up to the maximum number of ND11, as set in the driver file DRU625. Only one master station list may be provided, if one is provided at all. Its form will be the same as that used in producing the equation files from U600 or U602. The above mentioned equation files and master station list are in ASCII format. Reading is done with standard Fortran reads.

DATA OUTPUT

Most output is handled through the IP( ) values, Record Type 1, as specified in the Control File ‘U625.CN’. Each IP value may be given a separate unit number so that each type of information can be viewed in a separate file. In this way, each type of output information for all input equation files would be viewed in one output file. If the user would like a modified version of the input equation file(s) sans the duplicate stations (if any are found), use Record Type 5. If the number of modified output equation files specified in Record Type 5 is not equal to the number of input equation files specified in Record Type 4, U625 terminates execution. All errors will be written to the default output file. Every effort has been made to notify the user of problems and potential problems and to proceed under user control. All output is in ASCII format for viewing by the user.

EXAMPLE CONTROL FILE: 'U625.CN'

An example exists as file 'U625.CN' in directory /nfsuser/g06/mos2k/eqnlib/dru625 on the IBM-SP. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

Only one type of input equation file, either for use by U700 or U900, can be used for any single run of U625, due to how RDEQN reads the files. The value of NDATE input in the control file must be ‘9999' if U700 equation files are to be used. Otherwise, NDATE must be a date/time for which the equation files are valid.

COMMENTS

When NEW = 1, the identifiers output in the output files denoted by IP(4) through IP(9) will be the new ICAO station identifiers whether or not the (new) ICAO identifiers or (old) call letters are furnished in the input equation files. When NEW = 0, the call letters furnished in the equation files will be used. For each station read from the equation file, provision is made for up to 5 substitute stations. This means that any of the possible identifiers contained in /nfsuser/g06/table/station.tbl will be recognized, and from these the ICAO identifiers can be found, if NEW=1.

The purpose of the predictor list read on Unit No. KFILM is to determine which of the predictors may be needed to add to the U201 control file for an operational run. That is, when the predictor list used by the operational run of U201 is input, then the output on IP(10) will show which predictors to add to the list to accommodate the equations in this run. Most errors are output for printing starting with **** to KFILDO.

SETTING UP THE DRIVER DRU625

The preparation of the driver for a particular U625 run is relatively painless. It consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine.
ND1 -	Maximum number of stations that can be used. Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)).
ND2 -	Maximum number of terms, or predictors, in any equation.
ND3 -	Maximum number of predictands in any equation.
ND4 -	Maximum number of unique predictors in the equations of all files used by this run of U625.
ND11 -	Maximum number of input and modified output equation files that can be used.
ND13 -	Maximum number of different equations (regions) for an equation file.

Do not change the value of L3264W.

The user can see from the DRU625 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect while others have considerable effect. Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

NONSYSTEM ROUTINES USED

Use makefile.u625 which points to the ‘moslib’ and all other source codes that are needed to create the u625 exectuable.

LANGUAGE: FORTRAN77 with some FORTRAN-90 compliant HP extensions.

LOCATION: /nfsuser/g06/mos2k/eqnlib. The driver is also located in the same directory.

U630

EDITS STATION LISTS IN EQUATION FILES READ BY U700 AND U900.

Darren Drewry
April 17, 2000

PURPOSE: U630 makes certain changes to specified regions in equation files created by U600 and later used by U700 or U900. There are three changes which U630 can make. The first is to add a station or group of stations to a specified region in an equation file. The second is to delete a station or group of stations from a specified region in an equation file. The third change, global deletion, is to delete all equations for the specified region from the equation file. In each case, the region to be deleted from, or added to, is specified by the call letters of the first station appearing in that region on the equation file. Some familiarity with other MOS-2000 documents will be necessary for a full understanding of this writeup.

CONTROL FILE INPUT: 'U630.CN' (Unit = KFILDI) Output Control

Record Type 1 - Format (A4,25I3)

This record contains unit numbers for the run, and can even control the “default” output file number. KFILDI is the input unit number as specified in DRU630.

IPINIT - 4 characters, usually a user’s initials plus a run number, to append to “U630" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U630 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = ‘HRG2' and IP(1) = 40, the file name for Unit No. 40 = ‘U630HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. Although 25 values are read for consistency with other programs and for possible expansion, only the first is used. This value should not be the same as any other unit numbers used in U630 except possibly KFILDO (the default output file). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information. The IP(1) value used is:

(1) = All error diagnostics plus other information not directed specifically to the default output file. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in

DRU630 DATA statement = 12. Changing the default unit number and IPINIT allows multiple runs of U630 or other programs within the same directory without overwriting.

Record Type 2 - Format (A72) **Run Identification**

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (I10,/,I10) **Run Control Parameters**

This record contains two control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

NEW - When 1, all station call letter output will be done by using the new ICAO call letters. When 0, all station call letter output will be done by using those call letters which appear in the equation file.

NDATE - For the case of equation files to be used in U900, NDATE specifies the date/time for which the equations are valid. In this case, NDATE must be input as YYMMDDHH. This date should represent the forecast cycle and a month and day that fall within the season for which the equations are valid. For equation files to be used in U700, NDATE must be 9999.

Record Type 4 - Format (I3,4X,A60) **Input Equation File(s)**

This record (plus the terminator record) identifies the equation files which will be used as input. Records are read until the terminator KFILEQ( ) = 99 is reached. Maximum number of records, sans the terminator record, is ND11 (ND11 is set in the driver DRU630). This record type 4 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

KFILEQ(J) - Unit number(s) for the input equation file(s).

EQNNAM(J) - Name(s) of these equation file(s). (CHARACTER*60)

Record Type 5 - Format (I3,4X,A60) **Output Equation File(s)**

This record (plus the terminator record) identifies the equation files which will be used as output after the specified changes have been made. Records are read until the terminator KFILEQ( ) = 99 is reached. The number of output equation files **must be equal** to the number of input equation files read in Record Type 4 above and the input and output will conform one-to-one (i.e., KFILEQ(J) input corresponds to KFILEO(J) output). This Record Type 5 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMOUT.

KFILEO(J) - Unit number(s) for the output equation file(s).

EQNNAMO(J) - Name(s) of these equation file(s). (CHARACTER*60)

Record Type 6 - Format (I3,4X,A60) **Station Dictionary File**

This record (plus the terminator record) identifies the file which holds the station dictionary. Maximum number of records, sans the terminator record, = 1. This Record Type 6 is read by subroutine RDSNAM.

KFILD - Unit number for the file containing the station dictionary which holds information such as the station names and station call letters for every station.

DIRNAM - Name of the file KFILD. (CHARACTER*60)

Record Type 7 - Formats given below **Change List**

These records (plus the terminator record) identify the additions and deletions to be made to the station call letter lists of the input equation file(s). Note that with a "Global Delete," all equations for the specified region will be deleted from the equation file(s). Each change is made to every input equation file. Records are read until the terminator NOPS( ) = 99 is reached. The format of the change list changes slightly depending on what type of change is to be made:

Additions - Format (I2,/,A8,/,(7(A8,1X)))

NOPS(I) - Integer identifying the type of change to be made. For the case of an addition, this integer is "1".

FRSTAS(I) - The call letters of the first station in the region in which the change is to be made. This is how the region is identified in the equation file.

STALST(I) - The list of station call letters which is to be added to the specified region in the equation file. Up to seven stations, including the terminator, may be put on one line. This call letter list is terminated with 99999999.

Deletions - Format (I2,/,A8,/,(7(A8,1X)))

NOPS(I) - Integer identifying the type of change to be made. For the case of a deletion, this integer is "-1".

FRSTAS(I) - The call letters of the first station in the region in which the change is to be made. This is how the region is identified in the equation file.

STALST(I) - The list of station call letters which is to be deleted from the specified region in the equation file. Up to seven stations, including the terminator, may be put on one line. This call letter list is terminated with 99999999.

Global Deletions - Format (I2,/,A8)

<u>NOPS(I)</u> - Integer identifying what type of change is to be made. For the case of a global deletion, this integer is "-2".

<u>FRSTAS(I)</u> - The call letters of the first station in the region which is to be deleted. This is how the region is identified in the equation file. Note that all equations for this region will be deleted from the equation file(s).

<u>CONTROL FILE INPUT</u>: (Name read from U630.CN) (Unit = KFILEQ(J))

Record Type 1 <u>Equations</u>

This group of records contains the equations to modify. The maximum number of files (sets of forecasts) is ND11 (see Record Type 4). The format of each set is exactly in the format output by U600. Each set must end with the terminator 99999999. See <u>TDL Office Note</u> 00-1 "MOS-2000," Chapter 15, for more information.

<u>CONTROL FILE INPUT</u>: (Name read from U625.CN) (Unit = KFILD)

Record Type 1 - Format <u>Station Locations</u>
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or loca-tions). The call letters read from KFILD are matched with those in the equations, and the appropriate information extracted by subroutine RDSTAL or RDSTAD. When this station directory is alphabetical, the final list of stations can be alphabetical no matter the order read in (see NALPH in Record Type 3). The number of stations in this directory is not limited. No terminator is used. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3). Only the station names and their substitute stations are used from this file. See Chapter 10, Station Directory, in <u>TDL Office Note</u> 00-1 for more information.

<u>DATA INPUT</u>

All equation file input to U630 will be in the form of equation files created by U600 for use by U700 or U900. Multiple equation files, of one single type (for use by either U700 or U900), are accommodated, up to the maximum number of ND11 as set in the driver file DRU630. The equation files are in ASCII format. Reading is done with standard Fortran reads.

<u>DATA OUTPUT</u>

The specified changes are made to the input equation file(s) and then written to the output equation file(s) specified in Record Type 5 above. If the number of output equation files specified in Record Type 5 is not equal to the number of input equation files specified in Record Type 4, U630 terminates execution, and an error message is printed to KFILDO. All diagnostic output is handled through the IP(1) value, Record Type 1, as specified in the Control File ‘U630.CN’. All errors will be written to IP(1) or the default output file when IP(1) = 0. Every effort has been made to notify the user of problems and potential problems and to proceed under user control. All diagnostic output is in ASCII format for viewing by the user.

EXAMPLE CONTROL FILE: 'U630.CN'

An example exists as file 'U630.CN' in directory /home21/tdllib/dru630 on Blizzard. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

COMMENTS

When NEW = 1, the station call letters output in the output equation file(s) will be the new ICAO station identifiers whether or not the (new) ICAO identifiers or (old) call letters are furnished in the input equation files. When NEW = 0, the station call letters furnished in the input equation file(s) will be used in the output equation file(s). For each station read from the equation file, provision is made for up to five substitute stations. This means that any of the possible identifiers contained the file read on Unit No. KFILD will be recognized, and from these the ICAO identifiers can be found, when NEW = 1.

Only one type of input equation file, either those for use by U700 or U900, can be used for any single run of U630, due to how RDEQN reads the files. The value of NDATE input in the control file must be 9999 if U700 equation files are to be used. For U900 equation files, NDATE must represent the forecast cycle and a month and day that fall within the season for which the equations are valid (see Record Type 3).

Most errors are output for printing starting with **** to IP(1) or the default output file.

SETTING UP THE DRIVER DRU630

The preparation of the driver for a particular U630 run is relatively painless. It consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B - Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine.

ND1 -	Maximum number of distinct stations that can appear in an equation file. Note that this does not include the number of stations in the directory (read on Unit No. KFILD).
ND2 -	Maximum number of terms, or predictors, in any equation.
ND3 -	Maximum number of predictands in any equation.
ND4 -	Not used.
ND5 -	Not used.
ND6 -	Not used.
ND7 -	Not used.
ND8 -	Not used.
ND9 -	Not used.
ND10 -	Not used.
ND11 -	Maximum number of input equation files.
ND12 -	Not used.
ND13 -	Maximum number of different equations (regions) for an equation file. For single station developments, this must be ≥ ND1.

Do not change the computation or value of the variables L3264W, ADD, DEL, GLODEL.

The user can see from the DRU630 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect, while others have considerable effect (e.g., ND11). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib.dru630, dataset 'u630.com', which includes the libraries 'home21.tdllib.u630lib' and 'home21.tdllib.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some FORTRAN-90 compliant HP extensions.

LOCATION: /home21/tdllib/u630lib. The driver is in /home21/tdllib/dru630.

U661

COLLATES, PRINTS, AND PACKS DATA

Harry R. Glahn
July 1, 2000

PURPOSE: U661 collates data from a variety of MOS-2000 vector inputs, and writes for printing and packs and writes vector output for the stations and variables designated by control files. Because the packed data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in a random access file; these data are also packed in the TDLPACK format. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U661 uses a driver DRU661 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U661 (actually, subroutine) for every application. U661 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. It is possible to access data for variables at a date/time after the date/time being processed; this is accomplished with the variable "ITAU( )" and is called the "lookahead" feature. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

U661 is a clone of U660, the only difference being that U660 has the full OPTX capability of making computations but U661 has a special version of OPTX residing in u661lib that does not call computation routines. It was found too confusing in some circumstances to not know, when using U660, whether the data were already "computed" and resided on the input files or whether they were computed by U660 from other available data. Except for the name of the driver (DRU661 instead of DRU660) and the load line (see below), U661 is used exactly as U660, even the .CN is U660.CN; see U660 writeup for details.

COMMENTS:

The print on the default ftn12 will indicate U661; however, the print on other IP( ) output will show "START U660", except for the "END 661".

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib.dru661, dataset 'u661.com', which includes the libraries 'home21.tdllib.u660lib', 'home21.tdllib.u661lib', and 'home21.tdllib.moslib' in that order, all on blizzard.

LOCATION: The driver is in dru661.

U662

MERGES, COLLATES, PRINTS, AND PACKS VECTOR DATA

Harry R. Glahn
July 20, 2002

PURPOSE: U662 collates data from a variety of MOS-2000 vector inputs, and writes for printing and packs and writes vector output for the stations and variables designated by control files. Because the packed data are self describing, the data can come from various sources, the number being essentially unlimited. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U662 uses a driver DRU662 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U662 (actually, subroutine) for every application. U662 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. It is possible to access data for variables at a date/time after the date/time being processed; this is accomplished with the variable "ITAU( )" and is called the "lookahead" feature. Data for a particular variable ID can come from different sources (input files) for different stations, provided the inputs are on different unit numbers. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U662.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU662.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U662" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U662 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U662HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U662 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). 17 values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU662 DATA statement = 12. Changing the default unit number allows multiple runs of U662 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.
(7) = The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) = The variable ID list in summary form. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)
(9) = The variable list in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).
(10) = The variable ID's for the first day (day 1) as read from the archive tapes. This is just a list of all the variables on the input files.
(11) = The variable ID's of the archived data actually needed, in order as they appear on the archive files for day 1.
(12) = The list(s) of stations on the input file(s).
(13) = A diagnostic will be provided when non missing data for a particular variable ID are on more than one input file for the same station (location).
(14) = A diagnostic will be provided when there are no data for a particular input file. With tape switching, this may not be of much use, and could be misleading.

(15) = Data written in the order packed for each variable indicated by JP(3,) > 0. This is separate from the optional writing associated with JP(2,). Except for a very few days, this would produce voluminous files.

(16) = All or a portion of the data values in the AA(,) matrix. For each case (cycle), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The print is by variable controlled by JP(2,) (see Record Type 12 below), then the station values are printed in the order dealt with in U662 (see IP(5)). Except for a very few days or a few variables and stations, this would produce voluminous files.

(23) = Information concerning opening and closing of files.

(24) = Information as to where the variables are to be found. This has little utility in U662, because all variables must come directly from the source (no computations done).

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (7(I10/),F10.0/(I10)) Control Parameters

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

KSKIP - When nonzero, indicates the output file on unit KFILIO (see Record Type 8) is to be moved forward until all data for date/time KSKIP have been skipped. KSKIP is input as either YYYYMMDDHH or YYMMDDHH, and then used as YYYYMMDDHH. Note that when KSKIP > 0, there must be data with a directory record on the output file. Otherwise, the first read will be attempted and the program judges non-matching stations in the directory. Also, KSKIP must be < IDATE(1) (see Record Type 5).

KWRITE - The existing call letters record (directory) pertaining to the data for the date/time KSKIP (see above) will be checked with the one available for writing. If they match, the new one will not be written; if they don't match, the new one will be written when KWRITE = 1, but the program will halt with a diagnostic when KWRITE = 0. Note that this has no effect when KSKIP = 0.

NSKIP - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. Because of the lookahead feature required for predictands, and to maintain efficiency, date/times should not be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship.

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

PXMISS - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value.

IWMISS - Indicates whether (=1) or not (=0) a record will be written to Unit KFILDO (See Record Type 8) when it consists of only missing data.

NPRINT - The number of cycles of data to write for printing to unit IP(16) under the format control and JP(2,) provided with each variable (see Record Type 13).

ICHARS - The number of characters of call letters to print when printing is indicated by JP(2,). This is constrained to be between 4 and 8 inclusive.

LNGTH - The line length for printing to unit IP(16). For a line printer, 132 is appropriate; for a laser printer, 80 may be desirable.

Record Type 4 - Format (I3,4XA60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) <u>Date List</u>

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

<u>IDATE</u>(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list <u>must</u> be available or U662 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be > KSKIP (see Record Type 3).

Record Type 6 - Format (I3,4XA60) <u>Vector Input Data Files</u>

This group of records identifies the data sets from which the vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

<u>KFILIN</u>(J) - Unit number for the data file.

<u>NAMIN</u>(J) - Name of file where these data reside. (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. (A complication is the lookahead feature.) When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4XA60) <u>Random Access Files</u>

<u>Data from the External Random Access File System are not used by U662.</u> However, Record Type 7 is left so that there will not be a difference between the U662.CN and U660.CN control files in this regard. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. Normally, only the terminator would be here; any files indicated would not be used.

KFILRA(J) - Unit number for the random access constant files (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

RACESS(J) - Names of files of random access data sets matching KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) **Vector Output File**

This record (plus the terminator record) identifies the packed vector output file. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

KFILIO - Unit number for the vector output file.

OUTNAM - Name of file where this output is to reside. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) **Station and Location Files**

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (14(A8,1X)) **Station List**

This group of records identifies the (groups of) stations for which data are to be processed. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read within subroutine RDSTAD by RDC, which eliminates any blanks found in the input. Duplicate stations in the list for a group are kept, but a diagnostic is furnished on unit IP(5). Note that this diagnostic applies only to duplicates within a group, not from group to group. For NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty set terminates the station input. That is, the last group and its terminator must be followed by another terminator signifying an empty set. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) **Variable File**

This record (plus the terminator record) identifies the file from which the variable ID's are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) **Variable Constants File**

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. [Variable constants include plain language description and the decimal scaling factor for packing, when writing binary output to unit KFILIO (see record Type 8)]. Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 13 - Format **Variable List**
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,I3,8XA1,I2,1XI1,1X30A1)

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 13 is read by subroutine RDVR66. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201.

ID(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

<u>JP</u>(J,N) - JP(J,N) indicates for J=1,3:
1 = Whether (>0) or not (=0) variable N will be packed and written to unit KFILIO.
2 = Whether (>0) or not (=0) variable N will be written to unit IP(16) with the format provided. This is the primary output for viewing or printing. Also see NPRINT in Record Type 3.
3 = Whether (>0) or not (=0) variable N will be written to unit IP(15) to the resolution packed. This is for checkout and quality control of data being written. Since data are packed only when JP(1,N) > 0, both JP(1,N) and JP(3,N) must be > 0 for IP(15) output.

<u>ITAU</u>(N) - The number of hours to add to NDATE (the date being processed, see Record Type 5) to get the variable N. That is, the tau in the ID is left intact, and ITAU is used to "look ahead."

<u>CFMT</u>(N) - The format descriptor for writing on unit IP(16) when JP(2,N) > 0. Must be either "F" or "I".

<u>IWDTH</u>(N)- The field width for writing on unit IP(16) when JP(2,N) > 0. Must be $\leq$ 30.

<u>IPREC</u>(N)- The precision for writing on unit IP(16) when IP(2,N) > 0. For an "F" format, this is the number of digits after the decimal point. For an "I" format, it is the number of digits always written. Note that for a "zero" to be printed, IPREC( ) must be > 0; otherwise, zero will be printed as a blank.

<u>HEAD</u>(J,N) - The column heading for writing to Unit IP(16) when JP(2,N) > 0. Limited to J=1,30 characters. Only IWDTH(N) characters are read. HEAD(,) is right justified when writing.

<u>CONTROL FILE INPUT</u>: (Name read from U662.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) <u>Date List</u>

<u>When the dates are not provided in file U662.CN</u>, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU662), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input.

Terminator is 99999999. Data for the first date in the list must be available or U662 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U662.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided in file U662.CN, this group of records identifies the stations (or locations) to be included in the regression analysis. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which interpolated values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 10, control file 'U662.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U662.CN) (Unit = KFILD(2))

Record Type 1 - Format **Station Locations**
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical within each group no matter the order read in (see NALPH in Record Type 3). [Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. Note that alphabetization is not overall, but by group (see Record Type 10)]. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3).

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).
IWBAN(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited. No terminator is used.

CONTROL FILE INPUT: (Name read from U662.CN) (Unit = KFILP)

Record Type 1 - Format **Variable List**
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,I3,8XA1,I2,1XI1,1X30A1)

When the variables to use in the run are not in file U662.CN, this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDVR66. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201. See Control File Input for U662.CN, Record Type 13, unit KFILDI above for more detailed information; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U662.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) **Variable Constants**

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U662 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the data for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer.

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDVR66, much of it associated with the plain language description. See Chapter 4, Variable Identification, for details.

DATA INPUT:

All data input to U662 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000). All such vector data must be preceded by a call letters (directory) record. The sequential files can have multiple directory records, each applying to the data until an end of file or another directory is encountered. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U662.CN in Record Type 6.

A. SEQUENTIAL VECTOR DATA

One or more sources of vector data (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U662.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

DATA OUTPUT

There are two forms of data output, besides the diagnostics and other information on units KFILDO and IP( ).

A. ASCII FOR VIEWING OR PRINTING

A maximum of NPRINT (see Record Type 3) or NDATES (see Record Type 5) cycles of data will be written to the file on unit IP(16) under control of the format provided with each variable N, the variables written controlled by JP(2,N) (see Record Type 13). The data columns are headed by HEAD(,N) (see Record Type 13). Listing is by station group. If a line length for printing is not sufficient for all variables, then more than one line is used. Line length is specified as LNGTH characters in Record Type 3.

When JP(3,N) > 0, the data will also be written to unit IP(17) to the resolution packed; this is for quality control and would not be used for large samples of data.

B. BINARY MOS-2000 FORMAT

Data for each variable N for which JP(1,N) > 0 (see Record Type 13) for all NDATES cycles will be packed in TDLPACK format and written to unit number KFILIO to the dataset whose name has been provided to OUTNAM (see Record Type 8), unless KFILIO = 0, in which case data are not output. The data are packed with subroutine PACK1D and its associated subroutines. The scaling and plain language are taken from the "Variable Constants" table read from Unit Number KFILCP (See Record Type 12). All stations are put into each record even though the stations may have been read in groups and printed that way. This grouping would likely not be useful for U662, unless only a listing is desired.

EXAMPLE CONTROL FILE: 'U662.CN'

An example exists as file 'U662.CN' in directory home21.tdllib.dru662 on blizzard. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U662.CN', Record Type 1 and the definition of KFILDO in the driver DRU662. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

It is probably not necessary for each variable needed to actually be available for Day 1 for each unit number for the variable to be used on subsequent cycles for that same input unit number. However, some pathological case might occur to render this statement incorrect.

COMMENTS

The primary purpose of U662, and the reason it would be used instead of U660 (or U661), is that is will merge data for a particular ID from more than one source (sequential input file), provided the records are on different unit numbers. For instance, temperatures from METAR data from a directory of stations may be on one file and co-op or meso-data obs of the same variable (temperature) for a different set of stations might be on another file. These can both be input (on different unit numbers), and the stations will all be put onto the output according to the list of stations read in Record Type 10. If the same station exists on both inputs, the first one encountered, reading the input files in the sequence provided in U662.CN, will be used; in this case, a diagnostic will be provided on IP(13) when not zero.

To provide this merging capability, U662 will not take data from a random access file. Also, no computational capabilities exist; binaries are not made (although binaries on an input are accommodated) and OPTX is not called. However, the lookahead feature is retained. <u>Extreme caution should be exercised when using lookahead. This essentially relabels data with a different date/time.</u> It could be, for instance, a max or min temperature now labeled (stored) at one time (on one source) might need to be changed to another time; this feature should allow that.

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U662 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 3). Presumably, the only time 9997 will occur is when operational forecasts are input or U700 produces test forecasts. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. This probably means a set of dates should be supplied to U662 for which it is known there are good data. Then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

On Day 1, GFETCH1 is entered directly for every variable for every source (input unit number). ("Day 1 is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH1) <u>is not</u> counted as an error in VRBL68 for Day 1. It is also not counted as an error in VRBL69 for subsequent days unless the variable has been found sometime in the past on that unit, then it is counted as an error. Note that it is not known until some processing is done whether or not all input files contain all variables.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 10 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

Although a primary purpose of U662 is to provide packed vector data for other programs, writing of such data is not done if a unit number and name are not read in Record Type 8 (note that the terminator must be there).

This feature can be used for checkout or if the user wants just a few days for printing.

The capability of reading the station list (e.g., Record Type 10) in groups may be useful for printout (listing of data by station and by group). However, the alphabetization is done by group, and the writing to the binary output is done by a single group composed of all the groups read in sequence. That is, reading by group does not guarantee (and usually would not produce) output ordered alphabetically by station.

SETTING UP THE DRIVER DRU662

The preparation of the driver for a particular U662 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B - Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).

ND1 - Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the directory (read on unit KFILD(2)) unless, of course, the station directory is to be used as the station list (see ND5).

ND2 - Not used.

ND3 - Not used.

ND4 - The maximum number of variables that can be dealt with.

ND5 - Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.

ND6 - Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc., even though the same unit number might be used.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System.

The user can see from the DRU662 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib.dru662, dataset 'u662.com', which includes the libraries 'home21.tdllib.u662lib', 'home21.tdllib.u660lib' and 'home21.tdllib.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: u662lib. The driver is in dru662.

U605

PERFORMS SCREENING REGRESSION ANALYSIS ON ENSEMBLE DATA

Harry R. Glahn
January 1, 2007

PURPOSE: U605 is one in a series of programs designed to develop statistical relationships, usually regression equations, between predictands and predictors from archived data. The primary input data, point data from observations, interpolated from model fields by U201, etc., are packed in a GRIB-like TDLPACK format. Because the packed data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in a random access file; these data are also packed in the TDLPACK format. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U605 uses a driver DRU605 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U605 (actually, subroutine) for every application. U605 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. The default input and output unit numbers are set to 5 and 12, respectively, in drU605. The primary output of U605 is a set of regression equations and a matrix prepared for evaluation by other programs. The equations are also output in ASCII for viewing or printing. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup. See the COMMENTS section for differences between U600 and U605.

CONTROL FILE INPUT: 'U605.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU605.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U605" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U605 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U605HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U605 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). Values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU605 DATA statement = 12. Changing the default unit number allows multiple runs of U605 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = The variable IDs and the forced predictor IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.
(7) = The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) = The variable ID list in summary form after reordering low to high. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)
(9) = The variable list in summary form after reordering. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).
(10) = The variable ID's for the first day (day 1) as read from the archive datasets. This is just a list of all the variables on the input files.
(11) = The variable ID's of the archived data actually needed [entries in MSTORE(,)].
(12) = The list of stations on the input file(s).
(13) = Missing data diagnostics (see NOMISS in Record Type 3).
(14) = A diagnostic will be provided when there are no data for a particular input file.

(15) = All data values in the AA(,) matrix. For each case (cycle or day), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The <u>print is by station</u>, then the variable values are printed in the order dealt with in U605 (see IP(9)). Except for a very few days, this would produce voluminous files.

(16) = All or a portion of the data values in the AA(,) matrix. For each case (cycle or day), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The <u>print is by variable</u> controlled by JP( ) (see Record Type 13 below), the station values being printed in the order dealt with in U605 (see IP(5)). Except for a very few days or a few variables, this would produce voluminous files.

(17) = The cross product matrices. Except for a very few variables, this would be a lot of data. Also, the print is formatted F12.2, so many cases or large variable values may cause the value to not be printable. (Subroutine PRCROS could be modified for another format.)

(18) = The predictor means, standard deviations, and correlations of predictands with predictors.

(19) = The predictand means and standard deviations.

(20) = All equations, as the predictors are being selected, will be saved on unit IP(20) for viewing. When IP(20) ≠ KFILDO, only the last equation (the one with all the predictors) will be put on unit KFILDO. See the Comments section for more explicit information.

(21) = Summary reductions of variance and standard error of estimate.

(22) = A summary of the number of times each predictor is used in the equations.

(23) = Information concerning opening and closing of files.

(24) = Information as to where the variables are to be found--directly on input, binary computed from a previous variable, or (at least attempted to be) computed through subroutine OPTX.

For checkout, it may be advisable to set all these values to zero or the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files. If a file of the same unit number exists, it will be deleted when IP( ) ≠ 0 except for KFILDO. Do not use 5, 42 through 49, 97, 98, or 99 as unit numbers; they are reserved for other purposes.

Record Type 2 - Format (A72) <u>Run Identification</u>

This record is used to identify the run.

<u>RUNID</u> - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (9(I8/),7(F8.4/),3(I8/),F8.0/I8) <u>Control Parameters</u>

This group of records contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

<u>NSKIP</u> - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

<u>JSTOP</u> - The total number of errors that will be tolerated before the program halts (see Comments section).

<u>INCCYL</u> - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. Because of the lookahead feature required for predictands, and to maintain efficiency, date/times should not be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship.

<u>NST</u> - The (maximum) number of predictors to select.

<u>MFORCE</u> - The (maximum) number of predictors to force into the equation. The order read in (see Record Type 14) is the order to be forced. Various variance and covariance thresholds as described below apply.

<u>NSELT</u> - Selection method; there are two implemented. When NSELT = 1, the next predictor selected is the one that gives the highest additional reduction of variance (RV) with ANY ONE of the predictands. When NSELT = 2, the next predictor selected is the one that gives the highest AVERAGE additional RV over ALL of the predictands. However, since an average RV is harder to achieve than an RV for a single predictand, when CUTOFF (see below) has been reached for NSELT = 2, selection reverts to NSELT = 1 for that equation set.

<u>NSMETH</u> - Stopping method. The only one implemented is using CUTOFF, FORCUT, COLN, COLNB, and COLNGB (see below). Set = 1.

<u>NECVRB</u> - The number of values of an individual variable that is required for that variable to be used in an equation. A zero means that any number will be accepted. When NECVRB ≠ 0, the AA( ) matri-ces are saved and no cross products are calculated on pass 1 through the data. This can be 0 and the thresholds below will still apply. The primary purpose of NECVRB is to eliminate a predictand for a particular station (or group) when data are (largely) unavailable (e.g., one or more projections when doing simultaneous development). However, it can also apply to predictors, for example, observations; that would mean that the equation (for that station or group) could be developed, but not with that predictor. This keeps a variable which is largely

missing for a particular station or group from deleting development for that station or group.

NECCAS - The number of cases that is required for an equation to be developed. This applies to single station equations or generalized operator equations. Use 0 to disable, but not a good idea.

CUTOFF - For NSELT = 1 (see above), the additional fraction of the variance of one or more of the predictands that has to be explained for another predictor to be included in the equation when screening predictors. For NSELT = 2, the additional average fraction of the variance of all of the predictands that has to be explained for another predictor to be included in the equation. Use 0 to disable, but not a good idea.

FORCUT - The additional fraction of the variance of one or more of the predictands that has to be explained for another predictor to be included in the equation when forcing predictors. Use 0 to force all predictors, but not a good idea. Note that when FORCUT is > CUTOFF, the "forced" predictor could be rejected, but accepted by the normal screening process. Also note that this applies to individual predictand reductions of variance, not an average regardless of the value of NSELT.

VARNB = The variance of a point binary predictor necessary for the variable to be used. Use 0 to disable.

VARNGB = The variance of a grid binary predictor necessary for the variable to be used. Use 0 to disable.

COLN - The acceptance threshold for colinearity of other than binary predictors. That is, a predictor will not be allowed into the equation if (1-COLN) of its variance has been explained by the predictors already selected. Use 0 to disable.

COLNB - The acceptance threshold for collinearity of cumulative or discrete (point) binary predictors. That is, a point binary predictor will not be allowed into the equation if (1-COLNB) of its variance has been explained by the predictors already selected. Use 0 to disable.

COLNGB - The acceptance threshold for colinearity of grid binary predictors. That is, a grid binary predictor will not be allowed into the equation if (1-COLNGB) of its variance has been explained by the predictors already selected. Use 0 to disable.

NOMISS - Indicates whether (>0) or not (=0) missing data will be identified on file KFILDO [IP(13) = 0] or IP(13) [IP(13) > 0]. (See Comments section.)

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 may not be useful.

PXMISS - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value. In U605, this must be 0 or 9999. A value of 9999 will cause a 9997 datum to be treated as missing; a value of zero will be used as such. A value of 9997 might occur in the input data if forecasts were being used--an unusual circumstance.

MODRUN - A value to be used as the identifying "model/run" number for identifying the equations (the predictands). This 2-digit value is inserted as "DD" into the first word of the ID of each predictand just before the equations are output on file Unit No. KFILEQ(1) (see Record Type 8).

MENSDD - A value to be inserted as DD into the first word of the ID to write out the first output equation file (see Record Type No. 8) when IWSDD = 1 (see below). This would be the DD of the model the equations are to be evaluated on.

IWSDD - 0 = MENSDD is not used (see above).
1 = MENSDD is inserted as the DD for predictands and for the model predictors.

IWENS - 0 = Multiple output files are not to be written. Only the first (if more than one) provided in Record Type No. 8 will be written, and then only if IWSDD $\geq$ 1.
1 = As many output files are written as different DD's of the predictor DD's and files provided in Record Type 8 (plus one when IWSDD $\geq$ 1). Each output file will have predictand, as well as predictor, DD's equal to the DD of an ensemble used in development.

Record Type 4 - Format (I3,4XA60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) Date List

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U605 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates (after date spanning), sans terminator, is ND8.

Record Type 6 - Format (I3,4XA60) Vector Data Input Files

This group of records identifies the data sets from which the point data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

KFILIN(J) - Unit numbers for the data files (J=1,NUMIN).

NAMIN(J) - Names of files corresponding to KFILIN(J) where these data reside (J=1,NUMIN). (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. (A complication is the necessity for the lookahead feature for predictands.) When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4XA60) Random Access Files

This group of records identifies the MOS-2000 random access data sets from which constant (and possibly other) data are read. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. If no data are needed from a random access file, only the terminator is necessary.

KFILRA(J) - Unit numbers for the random access constant files (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

RACESS(J) - Names of files of constant data matching KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) **Equation Output Files**

These records (plus the terminator record) identify the equation output files. Note that this is in addition to the print provided through IP( ) values. Records are read until the terminator KFILEQ( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND11. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If this is an empty set, the equations will not be output.

KFILEQ(J) - Unit numbers for the output equation files (J=1,NOEQFL).

OUTNAM(J) - Names of files where this output is to reside (J=1,NOEQFL). (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) **Station and Location Files**

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which equation development is to be done (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used. When KFILD(1) = KFILD(2) and DIRNAM(1) = DIRMAM(2), all stations in the directory are used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (14(A8,1X)) **Station List**

This group of records identifies the (groups of) stations for which equation development is desired. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read within subroutine RDSTAD by RDC, which eliminates any blanks found in the input. Duplicate stations in the list for a group are kept, but a diagnostic is furnished on unit IP(5). Note that this diagnostic applies only to duplicates within a group, not from group to group. For

NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty set terminates the station input. That is, the last group and its terminator must be followed by another terminator signifying an empty set. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) **Variable File**

This record (plus the terminator record) identifies the file from which the variable ID's (both predictors and predictands) are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI (set in DRU605), PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) **Variable Constants File**

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 13 - Format
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,8XI2,I3,1XI6) **Variable List**

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source on Unit No. KFILP. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 13 is read by subroutine RDVRBE. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading predictors in U201.

ID(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

<u>JP</u>(N) - For each variable N, JP(N) indicates print or no print when ≠ 0 or = 0, respectively, for data values at the stations. These values combined with IP(15) and IP(16) allow easy control of output. For instance, all variables could have JP( ) ≠ 0 and IP(15) ≠ 0 (or IP(16) ≠ 0) for a short diagnostic run. All such print could be turned off by setting IP(15) = 0 (or IP(16) = 0) in this control file. Alternatively, IP(16) could be ≠ 0 and selected data obtained by changing JP( ) from 0 to ≠ 0.

<u>NTYPVR</u>(J) - The type of variable (J=1,NVRBL):
1 = predictor.
2 = predictand.
NTYPVR determines how the tau is used in the variable ID. For predictors, the variable must be present just as specified including the tau. For predictands, the lookahead feature must be used, so tau is added to the current date NDATE to get the date for which the variable is needed at tau = 0. As a special feature, if a predictor is an observation (CCC between 700 and 799 inclusive) and not from the AEV data archive (DD = 82), then tau is treated as lookahead as it is with predictands. This allows an observation to be used as a predictor a few hours after the model time.

<u>LIMIT</u>(J) - The number of zeros or of ones necessary for a binary predictand to have for an equation developed for it. This means that both AVG(N)*SMPL and (1-AVG(N))*SMPL must be ≥ LIMIT(N) for predictand N, where AVG(N) is the binary predictand average and SMPL is the sample size.

Note that the DD's for the predictors can pertain to ensemble output; we have designated these numbers to be 40 to 79 inclusive. If more than one ensemble is to be used, then the DD's will specify which ensembles are to be combined. Each ensemble predictor with a specific DD must be repeated for all ensembles. That is, if three ensemble predictors are to be included with a DD of 40, and it is desired to use other ensembles, then exactly the same predictors (except for the DD's) must be included for all ensembles to be used.

Record Type 14 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3) <u>Forced Predictors</u>

This group of records contains the variable ID's of the forced predictors. Records are read until the terminator IDFORC(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 14 is read by subroutine RDFORC. Upon completion of reading this record type, N=1,NFORCE. Note that the format and information for IDFORC(,) is exactly the same as for ID(,) above, up to the point needed.

IDFORC(J,N) - The first 3 (J=1,3) words of the predictor ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed. This is the exact format for variable ID's as for Record Type 12.

Note that if the number of (useable) predictors read for forcing does not match MFORCE read in Record Type 3, the number actually read, and not tossed for some reason like being a duplicate, will be used. This is flagged as an error to give the user feedback and positive control on the (number of) forced predictors.

Even though more than one model, designated by different DD's, can be included as predictors, the forced predictor(s) do not have to be duplicated, but the DD for the forced model predictors must have a DD = MENSDD (see Record Type 3).

CONTROL FILE INPUT: (Name read from U605.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) **Date List**

When the dates are not provided in file U605.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU605), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U605 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U605.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided in file U605.CN, this group of records identifies the stations (or locations) to be included in the regression analysis. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 9, control file 'U605.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U605.CN) (Unit = KFILD(2))

Record Type 1 - Format Station Locations
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) to be used. The call letters read from KFILD(2) (see Record Type 9) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical no matter the order read in (see NALPH in Record Type 3). (Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run.) However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. For example, there is no use of the elevation in the equation derivation. However, this information is available for a computation routine, if one is needed. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3).

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited, except when it also constitutes the list to be kept (see Record Type 9), in which case the number of entries is limited by ND1-1. No terminator is used.

CONTROL FILE INPUT: (Name read from U605.CN) (Unit = KFILP)

Record Type 1 - Format (I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,8XI2,I3,1XI6) **Variable List**

When the variables to use in the run are not in file U605.CN, this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDVRBE. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading predictors in U201. See Control File Input for U605.CN, Record Type 13, unit KFILDI above for more detailed information; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U605.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32) **Variable Constants**

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U605 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

Other processing occurs with the reading of these records in RDVRBE, much of it associated with the plain language description. See Chapter 4 in TDL Office Note 00-1, Variable Definition, for details.

If a “specific” ID that includes a “B” or “DD” is in the list along with a more generic one without the “B” or “DD,” the specific one must precede the generic one for the specific one to be used.

DATA INPUT:

All data interpolated from models and produced by U201 for input to U605 and all other vector data will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note 00-1 MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format, except for the call letters (directory) record. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U605.CN in Record Types 6 and 7.

A. SEQUENTIAL VECTOR DATA

One or more sources of vector data, observations or prepared by U201, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U605.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated. Observation records are expected to have the date of the observations and a tau of 0.

B. RANDOM ACCESS VECTOR DATA

Up to 5 sources of random access vector data are accommodated; however, it is unlikely more than 1 or 2 will be needed. These data exist in the MOS-2000 External Random Access File System. These data are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

<u>DATA OUTPUT</u>

All equations are placed in an ASCII file, on Unit Nos. KFILEQ( ) (see Record Type 8), that can be easily edited, especially for the stations to apply the equations to, and used directly by other programs such as U705 and U905 (see "MOS Equations" in TDL Office Note MOS-2000). Note that the predictand IDs have been modified so that the 2-digit value MODRUN (see Record Type 3) or the DD of an ensemble has been inserted as "DD" into the first word. This allows identification of the model or models on which the equations were based, or even a different development run on the same model (e.g., for primary and backup equations). Observations (CCC in the range 700-799) will have a tau [IDPARS(12)] representing how far ahead to get the observation. That is, tau is added to NDATE, then the record read with a tau = 0. (In U705 or U905, the "predictand" IDs will be modified to "forecast" IDs.)

<u>EXAMPLE CONTROL FILE</u>: 'U605.CN'

An example exists as file 'U605.CN' in directory home21.glahn.drU605 on blizzard. The easiest way to set up a run is to take an existing control file and modify it.

<u>OUTPUT</u>:

Output that can be printed can be put onto one or more of several files as described above under control file 'U605.CN', Record Type 1 and the definition of KFILDO in the driver DRU605. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

<u>RESTRICTIONS</u>

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

Some compilers accommodate optionally compiled statements with a "D" in Column 1. If a particular compiler does not allow this, please just replace the "D" with "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily. Do <u>not</u> use <u>only</u> a “C” in column 1 because it is too hard for the eye to distinguish.

The binary indicator "B" in the first word of a <u>predictor</u> ID must be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), or 5 (indicating a grid binary). For a <u>predictand</u>, "B" can also take the value of 2 (indicating a cumulative variable from below) or 3 (indicating a discrete variable). For the latter, the upper threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID's. If there is no lower one, the lower threshold is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999. Subroutine CKIDS called from RDVRBE prints a diagnostic if B for a predictor is not 0, 1, or 5 or for a predictand is not 0, 1, 2, 3, or 5. However, the variable is not eliminated.

Especially because of the lookahead feature necessary for predictands (and possibly for observations as predictors), all data needed for Day 1 should actually be available (e.g., there be a data record for all predictands, no matter how far ahead, for Day 1) when the variable is to be computed through subroutine OPTX. (There may also be cases when this restriction is also true for variables not computed through OPTX.) This does not mean that there cannot be a station with "missing" data. Also, a particular variable should be accessed on the same unit number. That is, if two seasons of data are on different files, they <u>must</u> have the same unit number. Different variables can be on different files.

It is not necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles, <u>unless</u> the variable is computed from other variables not otherwise used as a "raw" variable. That is, a raw (not computed) variable need not be present for Day 1, but one used <u>only</u> in computations must be, because U605 has no way of knowing it will be needed for future cycles.

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

U605 might use Unit No. 45 or even 48 or 49.

<u>COMMENTS</u>

U605 is different from U600 in two major respects:

(1) Multiple models can be combined in U605 to effectively increase the sample size and include the influence of multiple models in one equation set. This does not pick predictors from different models like U600 can do. (For instance, U600 can be used to include predictors from a particular model cycle and the same, or other, predictors from a previous cycle. However, to my knowledge, it has never been used for this purpose except in early checkout.) This allows an equation set developed from multiple models to be used on more than one model in operations. For instance, an equation set could be applied to several or all of the ensembles used in its development. Also, it could be used on the

“operational, non-ensemble) model output, provided the predictors to be used were available from that model.

(2) The inverse of the matrix of cross products used in developing the equation is written to the output equation files for use in U705 and U905 to provide an error estimate. (The variable sums are also written, but are needed only for checkout and are not used by U705 or U905.) See “Probabilistic Forecasts from Regression Estimates” by Bob Glahn dated Nov. 12, 2006, for additional explanation of the purpose and process.

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U605 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event is interpreted as having the value of PXMISS (see Record Type 3). Normally, PXMISS would be zero. Otherwise, to interpret the value as 9999, the case would have to be thrown out. Presumably, the only time 9997 will occur is when operational forecasts from U905 or U910 or test forecasts from U705 or U710 are input. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such. These values will be used the same way no matter how they are packed.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired (see IP(13)). This probably means a set of dates should be supplied to U605 for which it is known there are good data; then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

On Day 1, GFETCH is entered directly for every predictor (except a point binary) because it is not yet known when OPTX must be entered. ("Day 1" is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) <u>is not</u> counted as an error in VRBL1 for Day 1. It <u>is</u> counted as an error in VRBL2 (through OPTX), because GFETCH should not be entered directly when OPTX is needed.

When a point binary is needed, it will be computed in U605 if possible (i.e., the "basic" variable is available) even though it may exist on an input file. However, other computed predictors, such as sine and cosine functions are used from the input file if they exist, even though they might be computable in U605. For strictly mathematical variables like sine and cosine, it is likely more efficient to do the computation in U605.

The "interpolated" data input(s) can contain constant data taken from the MOS-2000 External Random Access File System by U201. Alternatively, U605 can take constant data directly from MOS-2000 External Random Access files. It is possible that no interpolated data on sequential files be used, and all input furnished be on "constant" random access files.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 5 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1.

U600, U602, and U605 call subroutine LMSTR1 which determines whether data from a particular file has been used on Day 1. If data from a particular file has not been used, it is unlikely that file will be needed in the run and is closed. This may keep the program from reading much unnecessary data.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 10 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

Plain language will be written as each predictor is selected and for each predictor on the last equation. If less than NST predictors are selected, the last equation will be repeated when IP( ) $\neq$ 0. This plain language comes from the Variable Constants file located on Unit No. KFILCP. If the output indicates no match is found (no valid plain language), the user should investigate why, because there may be an error besides just not finding plain language. For new variables, the Variable Constants file must be updated to identify the new variable.

Equations can be output on either the default output unit number KFILDO or the unit number specified in IP(20), or both. An option is available for writing all the equations as each individual predictor is selected or just the final equation with all predictors. The table below shows the options.

Default Output Unit No.	Value of IP(20)	Equations on Default Output	Equations on IP(20)
12	12	All	(Same as default)
12	0	Last Equation	None
12	20 (say)	Last Equation	All

SETTING UP THE DRIVER DRU605

The preparation of the driver for a particular U605 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B - Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine.

ND1 - Maximum number of stations (or points) that can be dealt with in the regression analysis. Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) unless, of course, the station directory is to be used as the station list.

ND2 - The size of the work area Q( ). This holds the sample size(s), sums, and sums of cross-products (in the non-redundant portion of the matrix). A test run of a day or so will provide guidance on the value of ND2 that would be appropriate for a similar run on many days. If more than one set of equations is to be developed (e.g., more than one "region" or single station), Q( ) must be large enough to handle one such set. If it is not large enough to handle all such sets at one time, more than one pass through the data will be required and the run will be much slower.

ND3 - Not used.

ND4 - The maximum number of variables (predictors and predictands) that can be dealt with.

ND5 - Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.

ND6 - Maximum number of all sequential file input sources (e.g., models) that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

ND11 - The maximum number of models (ensembles) to use.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

<u>WRITING NEW SUBROUTINES</u>

It is not expected that computational routines will be used to any great extent in U605. Rather, computations, except for point binaries, are expected to be done in U201. Any computations that would be done in U605 would have to be done on point data (not gridpoint data) because gridpoint data are not accommodated in U605. Even so, an "option" subroutine, OPTX, is provided for possible use. (In the case of development for gridpoints, the gridpoints are given identifications in the same way as other locations, such as stations.)

If model variables are used in computation, the DD’s must be MENSDD (see Record Type 3). This is because all ensemble variables are carried internally with a DD of MENSDD; this way, the same predictors from different ensembles can be merged.

<u>NONSYSTEM ROUTINES USED</u>

Use the load line in home21.glahn,dru605, dataset 'u605.com', along with the libraries,'home21.glahn.u605lib' and 'home21.glahn.u600lib', 'home21.glahn.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: home21/glahn/u605lib. The driver is in dru605.

U700

MAKES FORECASTS FROM REGRESSION EQUATIONS

Harry R. Glahn
October 1, 1998

PURPOSE: U700 uses one or more sets of regression equations to make forecasts from a variety of MOS-2000 vector inputs, and writes for printing and packs and writes vector output for the stations and variables designated by control files and the regression equations. Because the packed input data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in random access files; these data are also packed in the TDLPACK format. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U700 uses a driver DRU700 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U700 (actually, subroutine) for every application. U700 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U700.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU700.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U700" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U700 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U700HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U700 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are

different (except for IP(1)). Values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU700 DATA statement = 12. Changing the default unit number allows multiple runs of U700 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = Not used.
(7) = The list of <u>unique</u> predictors as determined from the equations, including thresholds and parsed ID's in IDPARS(,). Also, the ID's of the variables to make forecasts for, determined from the predictand ID's with the equations. These predictand IDs are the observations used in development.
(8) = The list of <u>unique</u> predictors as determined from the equations. This list does not include the parsed ID's, but does include the plain language as determined from the variable constant file on unit KFILCP (see below).
(9) = The list of variables, including the plain language, to make forecasts for as determined from the predictand/ forecast file and the variable constant file on unit KFILCP (see below). It also includes the packing constant for each variable, taken from the variable constant file, that will be used in packing the forecasts in TDLPACK format.
(10) = The variable ID's for the first day (day 1) as read from the archive tapes. This is just a list of all the variables on the input files.
(11) = The variable ID's of the archived data actually needed.
(12) = The list(s) of stations on the input file(s). Note that any station list encountered will be output; especially for hourly data, this would be voluminous.
(13) = Lists the stations for which forecasts are desired that do not have an equation.

(14) = A diagnostic will be provided when there are no data for a particular input file. With tape switching, this may not be of much use, and could be misleading.
(15) = Lists the stations with the equations that are not in the list to make forecasts for. If no input is provided on unit KFILD (see Record Type 9), then all stations are used, and there would be no print here.
(16) = The input data for each predictor in the equations for each date/time. This would generate voluminous output except for a checkout run of only a few date/times and sets of equations.
(17) = The forecast for each predictand for each date/time. This would generate voluminous output except for a check-out run of only a few date/times and sets of equations.
(18) = The forecast for each predictand for each date/time <u>to the accuracy packed</u>. This would generate voluminous output except for a checkout run of only a few date/times and sets of equations.
(19) = Multiple correlation coefficient for each predictand for each equation.
(20) = Means for each predictand for each equation.
(21) = --The predictands as read from the equations.
--The stations with the equations.
--The stations to make forecasts for with the following equations.
--The equations (the ID's and coefficients).
(22) = Not used.
(23) = Information concerning opening and closing of files.
(24) = Information as to where the variables are to be found--directly on input, binary computed from a previous variable, or (at least attempted to be) computed through subroutine OPTX.

For checkout, it may be advisable to set all these values to the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) <u>Run Identification</u>

This record is used to identify the run.

<u>RUNID</u> - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (9(I10/),I10) <u>Control Parameters</u>

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

<u>KSKIP</u> - When nonzero, indicates the output file on Unit KFILFC (see Record Type 8) is to be moved forward until all data for date/time KSKIP have been skipped. KSKIP is input as either

YYYYMMDDHH or YYMMDDHH, and then used as YYYYMMDDHH. Note that when KSKIP > 0, there must be data with a directory record on the output file. Otherwise, the first read will be attempted and the program judges non-matching stations in the directory. Also, KSKIP must be < IDATE(1) (see Record Type 5). KSKIP is of no use in the present implementation because the output file is opened as 'NEW'. This eliminates possible user error of writing over good data (forecasts). If this were to be reinstated, just modify U700 to open file 'FORNAM' on Unit No. KFILFC as 'OLD'.

KWRITE - The existing call letters record (directory) pertaining to the data for the date/time KSKIP (see above) will be checked with the one available for writing. If they match, the new one will not be written; if they don't match, the new one will be written when KWRITE = 1, but the program will halt with a diagnostic when KWRITE = 0. Note that this has no effect when KSKIP = 0.

NSKIP - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO.

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

PXMISS - The value to be used instead of 9997, if a 9997 is encountered in the input data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value. If the equations produce a 9997. it is packed as such.

Record Type 4 - Format (I3,4XA60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) Date List

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U700 stops. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be > KSKIP (see Record Type 3).

Record Type 6 - Format (I3,4XA60) Vector Input Data Files

This group of records identifies the data sets from which the vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN. If all input data are from a random access constant file (see record Type 7), only the terminator is necessary.

KFILIN(J) - Unit number for the data file (J=1,NUMIN).

NAMIN(J) - Name of file where these data reside (J=1,NUMIN). (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model as described by "DD" in the variable ID, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4XA60) Random Access Files

This group of records identifies the MOS-2000 random access data sets from which constant (and possibly other) data are read. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. If no data are needed from a random access file, only the terminator is necessary.

KFILRA(J) - Unit numbers for the random access constant files (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

RACESS(J) - Names of files of constant data (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) **Forecast Output File**

This record (plus the terminator record) identifies the packed forecast output file. Records are read until the terminator KFILFC( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

KFILFC - Unit number for the output forecast file.

FORNAM - Name of file where this output is to reside. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) **Station and Location Files**

This pair of records (plus the terminator record) identifies the files which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (14(A8,1X)) **Station List**

This record identifies the stations for which forecasts are desired. If KFILD(1) ≠ KFILDI, this record is omitted, and the information is taken from another source. If this data set is empty (only the terminator exists), then forecasts will be made for all stations stored with the equations. If there is more than one set of equations, the stations with the first set are used as the station list in lieu of the list that can be provided here. If this option is used, any diagnostics should be carefully considered in case an error is made in the station list with the equations (e.g., a station occurring more than once with a different equation). If a list is provided, the output order can be controlled. Note that whatever the unit number in KFILD(1), at least the terminator must be present.

When forecasts are to be made for all stations with the equations, then a particular station could be included for more than one equation in a set (because the stations are stored with the equations) and forecasts could be made for that station from each equation. However, if the list of stations is provided here and a station exists with the equations more than once in a set, it is not known which equation to use. In this case, the first equation read that has that station with it is used.

Note that there is only one list of stations carried in U700 to make forecasts for; it can be provided from an input list or can be an amalgamation of the stations with the first set of equations.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which forecasts are desired (K=1,NSTA). This list is read within subroutine RDSTAD or RDSTAL by RDC, which eliminates any blanks found in the input. Duplicate stations in the list are kept, but a diagnostic is furnished on unit IP(5). For NALPH = 1, the stations are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) **Equation Files**

These records (plus the terminator record) identify the files from which the regression equations are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = ND11. This Record Type 11 is read by subroutine RDSNAM. Upon completion of reading J=1,KGP.

KFILEQ(J) - Unit number for a set of equations (J=1,KGP).

EQNNAM(J) - Name of file corresponding to KFILP (J=1,KGP). When KFILP = KFILDI, EQNNAM is not used and can be read as "DEFAULT"; this would be a very unusual circumstance. (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) **Predictand/Forecast Table File**

This record (plus the terminator record) identifies the file from which the "forecast" ID for each "predictand" ID is read. Records are read until the terminator KFILPF = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILPF - Unit number for the predictand/forecast correspondence file.

PFCORR - Name of file corresponding to KFILPF. (CHARACTER*60)

Record Type 13 - Format (I3,4XA60) Variable Constants File

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 13 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 14 Equations

These records contain the equations in the exact form output by U600 and U602 and described in TDL Office Note MOS-2000, Chapter 15. The first record contains the file name in ASCII. As a safety feature, this name must match the actual file name, EQNNAM. The equations are arranged:

(1) File name - Format (' 'A60)
(2) NTAND, number of predictands in this set - Format (' 'I4)
(3) ID(J,N) (J=1,4), ICAT(N), (N=1,NTAND), the predictand ID's - Format (' 'I9.9,2I10,I11,I8)
(4) CCALL(K), stations (call letters) to which the equations following apply - Format (14(' 'A8))
(5) The equations (See TDL Office Note MOS-2000, Chapter 15).

This file will contain one "set" of equations (or actually more, see below), set being defined as one or more equations having the same predictands. The file name, number of predictands, and predictand ID's will occur only once on the file [(1) and (2) above]; the sequence of equations and stations can be repeated until a blank call letters record is read [(4) above]. For this to occur correctly, the file must end with '99999999'; that is, a call letters record with only the terminator present. U600 and U602 produce a "blank" call letters record for this purpose. So, unless the file name is to be changed, the output of U600 and U602 can be directly input into U700 with no change.

Another file can be read for another "set" of equations. Another physical file must have a different unit number KFILEQ( ) and file name EQNNAM( ). However, since U700 does not rewind the file after reading, the second set can actually be on the same file, but the complete sequence of input must be repeated, including the file name [(1) above]. Note that the file name imbedded in each "set" of equations must match the actual file name EQNNAM( ). It is recommended for consistency that each set of equations be on a separate file (the normal way they would be developed and output) and that the unit numbers start with 50 and continue sequentially through 73 as necessary.

Note that when there is no station list read in Record Type 10, forecasts will be made for all stations read with the first set of equations in this Record Type 14.

CONTROL FILE INPUT: (Name read from U700.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) **Date List**

When the dates are not provided in file U700.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU700), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U700 stops. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be > KSKIP (see Record Type 3).

CONTROL FILE INPUT: (Name read from U700.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided in file U700.CN, this record identifies the stations (or locations) for which forecasts are desired. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI, or alternatively, for an empty set, forecasts are made for all stations with the equations. (Also see discussion for Record Type 10 above.)

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which forecasts are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 10, control file 'U700.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U700.CN) (Unit = KFILD(2))

Record Type 1 - Format **Station Locations**
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted by subroutine RDSTAL or RDSTAD. When this station directory is alphabetical, the final list of stations can be alphabetical no matter the order read in (see NALPH in Record Type 3). [Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3). Only the station names and their substitute stations are used from this file. See Chapter 10, Station Directory, in TDL Office Note MOS-2000 for more information.

CONTROL FILE INPUT: (Name read from U700.CN) (Unit = KFILEQ)

Record Type 1 Equations

When the equations to use in the run are not in file U700.CN, this group of records contains the equations from which to make forecasts. The maximum number of files (sets of forecasts) is ND11. The format of each set is exactly in the format output by U600 and U602. Each set must end with the terminator '99999999'. See Record Type 14 above and TDL Office Note MOS-2000, Chapter 15, for more information.

CONTROL FILE INPUT: (Name read from U700.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U700 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables in the equations for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the forecasts for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer.

Other processing occurs with the reading of these records in SETPLN, much of it associated with the plain language description. See Chapter 4, Variable Identification, for details. If an IDTEMP( ) is not found that matches a particular ID(,), then the default for PLAIN( ) is blanks and for ISCALD is zero, except for a binary, ISCALD is set = 3. However, see MCCCFFF below.

CONTROL FILE INPUT: (Name read from U700.CN) (Unit = KFILPF)

Record Type 1 - Format (I6,2XI6,I4) Predictand/Forecast Correspondence

This group of records contains the correspondence between the predictand ID (the data on which the equations were developed) and the forecast ID. That is, the forecast for, say, temperature has a different ID than does the temperature observation. Only the CCCFFF need be changed.

ICCCFFF - The CCCFFF of the predictand ID.

MCCCFFF - The CCCFFF of the corresponding forecast ID.

ICAT - Postprocessing indicator for ICCCFFF. For instance, ICAT = 5 would switch to postprocessor subroutine CAT5. this is primarily used for inflation (ICAT = 1) which has to be done in U700 if at all because to compute it requires the predictand mean and correlation, which are not carried forward into other programs. When ICAT = 0, no postprocessing is done in U700.

Note that when the MCCCFFF is known, the plain language and packing constant are taken from the variable constants file (unit KFILCP) for MCCCFFF rather than the predictand ID in ID(,).

DATA INPUT:

All data input to U700 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format. All such vector data must be preceded by a call letters (directory) record. The sequential files can have multiple directory records, each applying to the data until an end of file or another directory is encountered; the random access files have only one directory record each of which applies to all data in the file. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U700.CN in Record Types 6 and 7.

A. SEQUENTIAL VECTOR DATA

One or more sources of vector data on sequential files, each probably prepared by U201, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U700.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each source (file) has a directory record at the beginning, and each data value in each record applies to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

B. <u>RANDOM ACCESS VECTOR DATA</u>

Up to 5 sources of station constant data are accommodated; however, it is unlikely more than 1 or 2 will be needed. These data exist in the MOS-2000 External Random Access File System. These data are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

<u>DATA OUTPUT</u>

There are two forms of data output, besides the diagnostics and other information on Units KFILDO and IP( ).

A. <u>ASCII FOR VIEWING OR PRINTING</u>

Input data for each date/time for the variables in the equations will be written to the file on Unit IP(16) when IP(16) > 0. Forecasts for each date/time will be written to the file on unit IP(17) when IP(17) > 0. Forecasts for each date/time <u>to the accuracy packed</u> (see B. below) will be written to the file on unit IP(18) when IP(18) > 0.

Note that the volume of output will be such that this print capability should only be used for a small sample.

B. <u>SEQUENTIAL BINARY MOS-2000 FORMAT</u>

Forecasts for all variables for all NDATES cycles will be packed in TDLPACK format and written to unit number KFILFC to the dataset whose name has been provided to FORNAM (see Record Type 8), unless KFILFC = 0, in which case data are not output. The data are packed with subroutine PACK1D and its associated subroutines.

<u>EXAMPLE CONTROL FILE</u>: 'U700.CN'

An example exists as file 'U700.CN' in directory home21.tdllib.dru700 on blizzard. The easiest way to set up a run is to take an existing control file and modify it; <u>DO NOT START FROM SCRATCH</u>.

<u>OUTPUT</u>:

Output that can be printed can be put onto one or more of several files as described above under control file 'U700.CN', Record Type 1 and the definition of KFILDO in the driver DRU700. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The CRAY does not allow this. It is best to insert before the "D" in columns 1-5 the characters, "C****". This way, the possible optional statements can be spotted and be made operative very easily.

The binary indicator "B" in the first word of an predictor ID can be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), or 5 (indicating a grid binary). Note that a predictor "B" cannot be 2 (indicating a cumulative variable from below) or 3 (indicating a discrete variable); however, these are possible for predictand definition. For the latter, the upper threshold is the one provided with the variable and the lower threshold must be assumed or determined by the previous equation. These predictand binary definitions play no role in U700, but are needed for definition of the forecasts in the binary output.

It is not, in general, necessary for each variable needed to actually be available for Day 1 for the variable to be used on subsequent cycles, <u>unless</u> the variable is computed from other variables not otherwise used as "raw" variables. That is, a raw (not computed) variable need not be present for Day 1, but one used <u>only</u> in computations must be, because U700 has no way of knowing it will be needed for future cycles. As an added caveat, if <u>none</u> of the variables on a particular file is found and used for Day 1, the file will be closed and data on it will not be available for future days; this could easily be changed if it becomes a problem.

As indicated in the description for input Record Type 10, multiple forecasts for a station can be made for a set of equations if the station list with the equations is used, but only one forecast can be made if the station list is furnished in Record Type 10 or from a separate station list file.

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

U700 would likely not need Unit No. 46. Since U700 does not write to a random access file, either 48 or 49 could be used, and the forecasts would not be compromised.

COMMENTS

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U700 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 3). Presumably, the only time 9997 will occur on input is when forecasts are input. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. This probably means a set of dates should be supplied to U700 for which it is known there are good data. Then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

On Day 1, GFETCH is entered directly for every predictor (except a point binary) because it is not yet known when OPTX must be entered. ("Day 1 is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) <u>is not</u> counted as an error in FCST71 for Day 1.

It <u>is</u> counted as an error in FCST72 (through OPTX), because GFETCH should not be entered directly when OPTX is needed.

When a point binary is needed, it will be computed in U700 if possible (i.e., the "basic" variable is available) even though it may exist on an input file. However, other computed predictors, such as sine and cosine functions are used from the input file if they exist, even though they might be computable in U700.

The sequential data input(s) can contain constant data taken from the MOS-2000 External Random Access File System by U201. U201 can access up to 5 such data sets if necessary. Alternatively, U700 can take constant data directly from the MOS-2000 External Random Access files; all such access is through the subroutine CONST. It is possible that the IDs of such data have "B" = 0 with a non-zero threshold, indicating these are relative frequencies computed with that threshold. It is possible, but would be highly unlikely, that no data on sequential files be used, and all input furnished be on random access files. See "Restrictions" for unit numbers and associated uses.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1.

Because of the way RDSTAL and RDSTAD operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 10 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

Although a primary purpose of U700 is to provide packed vector forecasts for other programs, writing of such data is not done if KFILFC = 0 (see Record Type 8). That is, no output unit number and file name have been provided. This feature can be used for checkout or if the user wants just a few days for printing.

Note that the variable KWRITE enables the user to control whether or not a new station directory record is written to unit KFILFC, if the one existing on the output just prior to the data to be written does not match the one available to be written. If the user expects them to be the same, then KWRITE = 0 can be used to halt the program with a listing of the directory records. If the user is willing to accept a different directory, then KWRITE = 1 can be used. Also note that KWRITE has no effect unless KSKIP > 0; when KSKIP = 0, the call letters record is always written. <u>With the current implementation, KSKIP > 0 is incompatible with opening KFILFC as "NEW."</u>

When equations are developed, the predictands are usually identified as hourly observations or some variable calculated from them. The forecasts of these variables must have a different ID. The file whose unit number and name are read in Record Type 12 contains this correspondence. The plain language, etc. for those "forecast" IDs are on the file whose unit number and name are read in Record Type 13.

SETTING UP THE DRIVER DRU700

The preparation of the driver for a particular U700 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) (see ND5). It also has to be $\geq$ BLOCK, the size of a physical record size in the MOS-2000 Internal Storage System; DRU700 assures this.
ND2 -	Maximum number of terms in an equation.
ND3 -	Maximum number of predictands per equation set.
ND4 -	The maximum number of <u>unique</u> predictors that can be dealt with. That is, there might be 500 equations, each with 10 terms, but only 25 unique predictors.
ND5 -	Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1. Also, must be large enough to hold the largest packed record.
ND6 -	Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc., even though the same unit number might be used.
ND7 -	The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.
ND8 -	The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.
ND9 -	The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records for the date/time of Day 1 on all input files.
ND10 -	The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although operating system

caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

ND11 - The maximum number of sets of equations. This is the same as the maximum number of files containing equations.

ND13 - The maximum number of equations per set. For single station equations, this would be the same as ND1.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU700 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND3). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious, and if a problem occurs that is traced to a too small ND value, notify your supervisor or the U700 author.

WRITING NEW SUBROUTINES

It is not expected that computational routines will be used to any great extent in U700. Rather, computations, except for point binaries, are expected to be done in U201, or even U660. This is reasonable, because, for the development data, these same computations would be made for U600 or U602. Any computations that would be done in U700 would have to be done on point data (not gridpoint data) because gridpoint data are not accommodated in U700. Even so, an "option" subroutine, OPTX, is provided for possible use. It is also through OPTX that the "constant" data are accessed from the external random access files. (In the case of development for gridpoints, the gridpoints are given identifications in the same way as other locations, such as stations, and the data are treated as "vector" after leaving U201.)

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib.dru700, dataset 'u700.com', which includes the libraries 'home21.tdllib.u700lib' and 'home21.tdllib.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: u700lib. The driver is in dru700.

U710

POSTPROCESSES FORECASTS ON SEQUENTIAL FILES

Harry R. Glahn
June 15, 1999

PURPOSE: The primary purpose of U710 is to postprocess MOS-2000 forecasts residing on sequential file(s), usually produced by U700. U710 is derived from U660 and will also fulfill most (if not all) of the functions possible in U660. The postprocessing done is specified by the variable ID, and U710 switches to a postprocessing subroutine via OPTX to perform the calculations. U710 can collate data from a variety of MOS-2000 vector inputs, and writes for printing and packs and writes a sequential vector output file for the stations and variables designated by control files. Because the packed data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in random access files; these data are also packed in the TDLPACK format. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U710 uses a driver DRU710 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U710 (actually, subroutine) for every application. U710 is written to run on a 32-bit or a 64-bit word-length machine. This is done through PARAMETER statements in the driver. It is possible to access data for variables at a date/time after the date/time being processed; this is accomplished with the variable "ITAU( )" and is called the "lookahead" feature. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U710.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) **Output Control**

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU710.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U710" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U710 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U710HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U710 except possibly KFILDO (the default output file), although a value of one IP( )

can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). 17 values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU710 DATA statement = 12. Changing the default unit number allows multiple runs of U710 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.
(7) = The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) = The variable ID list in summary form. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)
(9) = The variable list in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).
(10) = The variable ID's for the first day (day 1) as read from the archive tapes. This is just a list of all the variables on the input files.
(11) = The variable ID's of the archived data actually needed, in order as they appear on the archive files for day 1.
(12) = The list(s) of stations on the input file(s).
(13) = Not used.

(14) = A diagnostic will be provided when there are no data for a particular input file. With tape switching, this may not be of much use, and could be misleading.
(15) = Data written in the order packed for each variable indicated by JP(3,) > 0. This is separate from the optional writing associated with JP(2,). Except for a very few days, this would produce voluminous files.
(16) = All or a portion of the data values in the AA(,) matrix. For each case (cycle), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The <u>print is by variable</u> controlled by JP(2,) (see Record Type 12 below), then the station values are printed in the order dealt with in U710 (see IP(4)). Except for a very few days or a few variables and stations, this would produce voluminous files.
(23) = Information concerning opening and closing of files.
(24) = Information as to where the variables are to be found--directly on input, binary computed from a previous variable, or (at least attempted to be) computed through subroutine OPTX.

For checkout, it may be advisable to set all these values to the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) **<u>Run Identification</u>**

This record is used to identify the run.

<u>RUNID</u> - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (7(I10/),F10.0/(I10)) **<u>Control Parameters</u>**

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

<u>KSKIP</u> - When nonzero, indicates the output file on unit KFILIO (see Record Type 8) is to be moved forward until all data for date/time KSKIP have been skipped. KSKIP is input as either YYYYMMDDHH or YYMMDDHH, and then used as YYYYMMDDHH. Note that when KSKIP > 0, there <u>must</u> be data with a directory record on the output file. Otherwise, the first read will be attempted and the program judges non-matching stations in the directory. Also, KSKIP must be < IDATE(1) (see Record Type 5).

<u>KWRITE</u> - The existing call letters record (directory) pertaining to the data for the date/time KSKIP (see above) will be checked with the one available for writing. If they match, the new one will not be written; if they don't match, the new one will be written when KWRITE = 1, but the program will halt with a diagnostic when KWRITE = 0. Note that this has no effect when KSKIP = 0.

NSKIP - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. Because of the lookahead feature required for predictands, and to maintain efficiency, date/times should not be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship.

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

PXMISS - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value.

NPRINT - The number of cycles of data to write for printing to unit IP(16) under the format control and JP(2,) provided with each variable (see Record Type 13).

ICHARS - The number of characters of call letters to print when printing is indicated by JP(2,). This is constrained to be between 4 and 8 inclusive.

LNGTH - The line length for printing to unit IP(16). For a line printer, 132 is appropriate; for a laser printer, 80 may be desirable.

Record Type 4 - Format (I3,4XA60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) **Date List**

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U710 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be > KSKIP (see Record Type 3).

Record Type 6 - Format (I3,4XA60) **Vector Input Data Files**

This group of records identifies the data sets from which the point data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN. If all input data are from a constant file, only the terminator is necessary.

KFILIN(J) - Unit number for the data file (J=1,NUMIN).

NAMIN(J) - Name of file where these data reside (J=1,NUMIN). (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. (A complication is the necessity for the lookahead feature for predictands.) When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4XA60) **Random Access Files**

This group of records identifies the random access data sets from which constant (and possibly other) data are read. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM.

If no data are needed from a random access file, only the terminator is necessary.

<u>KFILRA</u>(J) - Unit number for the random access constant file (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

<u>RACESS</u>(J) - Name of file of random access data corresponding to KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) **<u>Vector Output File</u>**

This record (plus the terminator record) identifies the packed vector output file. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written. (See JP(1,), Record Type 13.)

<u>KFILIO</u> - Unit number for the vector output file.

<u>OUTNAM</u> - Name of file where this output is to reside. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) **<u>Station and Location Information</u>**

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

<u>KFILD</u>(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

<u>DIRNAM</u>(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (14(A8,1X)) **<u>Station List</u>**

This group of records identifies the (groups of) stations for which data are to be processed. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

<u>CCALL</u>(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read within subroutine RDSTAD by RDC, which eliminates any blanks found in the input. Duplicate

stations in the list for a group are kept, but a diagnostic is furnished on unit IP(5). Note that this diagnostic applies only to duplicates within a group, not from group to group. For NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty set terminates the station input. That is, the last group and its terminator must be followed by another terminator signifying an empty set. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) **Variable File**

This record (plus the terminator record) identifies the file from which the variable ID's are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) **Variable Constants File**

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. [Variable constants include plain language description and the decimal scaling factor for packing, when writing binary output to unit KFILIO (see record Type 8)]. Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 13 - Format **Variable List**
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,I3,8XA1,I2,1XI1,1X30A1)

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 13 is read by subroutine RDVR79. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201.

ID(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

<u>JP</u>(J,N) - JP(J,N) indicates for J=1,3:
1 = Whether (>0) or not (=0) variable N will be packed and written to unit KFILIO. For writing to occur, a file number and name must be provided in Record Type 8.
2 = Whether (>0) or not (=0) variable N will be written to unit IP(16) with the format provided. This is the primary output for viewing or printing (see NPRINT, Record Type 3).
3 = Whether (>0) or not (=0) variable N will be written to unit IP(15) to the resolution packed. This is primarily for checkout and quality control of data being written.

<u>ITAU</u>(N) - The number of hours to add to NDATE (the date being processed, see Record Type 5) to get the variable N. That is, the tau in the ID is left intact, and ITAU is used to "look ahead."

<u>CFMT</u>(N) - The format descriptor for writing on unit IP(16) when JP(2,N) > 0. Must be either "F" or "I".

<u>IWDTH</u>(N)- The field width for writing on unit IP(16) when JP(2,N) > 0. If IWDTH( ) is > 30, it will be set to 30.

<u>IPREC</u>(N)- The precision for writing on unit IP(16) when IP(2,N) > 0. For an "F" format, this is the number of digits after the decimal point. For an "I" format, it is the number of digits always written. Note that for a "zero" to be printed, IPREC( ) must be > 0; otherwise, zero will be printed as a blank.

<u>HEAD</u>(J,N) - The column heading for writing to Unit IP(16) when JP(2,N) > 0. Limited to J=1,30 characters. Only IWDTH(N) characters are read. HEAD(,) is right justified when writing.

<u>CONTROL FILE INPUT</u>: (Name read from U710.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) **<u>Date List</u>**

<u>When the dates are not provided in file U710.CN</u>, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU710), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as

YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U710 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U710.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided in file U710.CN, this group of records identifies the stations (or locations) to be included in the regression analysis. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which interpolated values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 10, control file 'U710.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U710.CN) (Unit = KFILD(2))

Record Type 1 - Format **Station Locations**
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical within each group no matter the order read in (see NALPH in Record Type 3). [Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. Note that alphabetization is not overall, but by group (see Record Type 10)]. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3).

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

<u>NAME</u>(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

<u>NELEV</u>(K) - Elevation of the station. Format is I5.

<u>SIGNLA</u> - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

<u>XLAT</u> - Latitude in degrees. Format is F7.4.

<u>SIGNLO</u> - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

<u>LONDD</u> - Longitude in degrees west. Format is F8.4.

<u>CCALLD</u>(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).
<u>IWBAN</u>(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited. No terminator is used.

<u>CONTROL FILE INPUT</u>: (Name read from U710.CN) (Unit = KFILP)

Record Type 1 - Format **<u>Variable List</u>**
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X4I2,9XA1,I2,1XI1,1X30A1)

<u>When the variables to use in the run are not in file U710.CN</u>, this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDVR79. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201. See Control File Input for U710.CN, Record Type 13, unit KFILDI above for more detailed information; the format is exactly the same.

<u>CONTROL FILE INPUT</u>: (Name read from U710.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) **<u>Variable Constants</u>**

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U710 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the data for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer. If a variable being processed cannot be found in this file, the default SCALE = 0 is used.

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDVR79, much of it associated with the plain language description. See Chapter 4, Variable Identification, for details. If an IDTEMP( ) is not found that matches a particular ID(,), then the default for PLAIN( ) is blanks and ISCALD is set = 0, except for a binary forecast, ISCALD is set = 3.

DATA INPUT:

All data input to U710 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format, except for the call letters (directory) record. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U710.CN in Record Types 6 and 7.

A. SEQUENTIAL VECTOR DATA

One or more sources of vector data on sequential files, each probably prepared by U201, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U710.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

B. RANDOM ACCESS VECTOR DATA

Up to 5 sources of station constant data are accommodated; however, it is unlikely more than 1 or 2 will be needed. These data exist in the MOS-2000 External Random Access File System. These data are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

DATA OUTPUT

There are two forms of data output, besides the diagnostics and other information on units KFILDO and IP( ).

A. ASCII FOR VIEWING OR PRINTING

A maximum of NPRINT (see Record Type 3) or NDATES (see Record Type 5) cycles of data will be written to the file on unit IP(16) under control of the format provided with each variable N, the variables written controlled by JP(2,N) (see Record Type 13). The data columns are headed by HEAD(,N) (see Record Type 13). Listing is by station group. If a line length for printing is not sufficient for all variables, then more than one line is used. Line length is specified as LNGTH characters in Record Type 3.

When JP(3,N) > 0, the data will also be written to unit IP(17) to the resolution packed; this is for quality control and would not be used for large samples of data.

B. BINARY MOS-2000 FORMAT

Data for each variable N for which JP(1,N) > 0 (see Record Type 13) for all NDATES cycles will be packed in TDLPACK format and written to unit number KFILIO to the dataset whose name has been provided to OUTNAM (see Record Type 8), unless KFILIO = 0, in which case data are not output. The data are packed with subroutine PACK1D and its associated subroutines.

EXAMPLE CONTROL FILE: 'U710.CN'

An example exists as file 'U710.CN' in directory home21.tdllib.dru710 on blizzard. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U710.CN', Record Type 1 and the definition of KFILDO in the driver DRU710. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The CRAY does not allow this. It is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

The binary indicator "B" in the first word of an ID can be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), 5 (indicating a grid binary), 2 (indicating a cumulative variable from below), or 3 (indicating a discrete variable). For the latter, the upper threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID's. If there is no lower one, the lower threshold is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999.

Since the variables are not ordered (as they are in U600), if a discrete binary is desired (one with both upper and lower thresholds), these variable ID's must be input in sequence and in the correct order (lower threshold first). To make sure all is well, check the thresholds on unit IP(8) or IP(9). Ordering of variables is important in U710 when a "series" of variables is needed for postprocessing (e.g., normalizing a set of probabilities).

It is not necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles, <u>unless</u> the variable is computed from other variables not otherwise used as a "raw" variable. That is, a raw (not computed) variable need not be present for Day 1, but one used <u>only</u> in computations must be, because U710 has no way of knowing it will be needed for future cycles. Practically, data needed should be present for Day 1, because the primary purpose of U710 is to perform computations.

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use

45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

U710 would likely not need Unit No. 46. Since U710 does not write to a random access file, either 48 or 49 could be used, and the forecasts would not be compromised.

COMMENTS

The .CN control file is of exactly the same format as for U660, from which U710 was derived. The only difference in the driver, DRU710, is that U710 uses ND2 and U660 does not. In some cases U710 needs to deal with and return a set (or series) of variables (usually, if not always, a set of probabilities). For instance, a set of probabilities are to be normalized. ND2 is the maximum number of variables in such a series. For instance, for four cloud amount categories, ND2 must be $\geq$ 4.

Although a primary purpose of U710 is to provide postprocessed and packed vector data for other programs, writing of such data is not done if KFILIO = 0 (see Record Type 8); that is, no output unit number and file name have been provided. This feature can be used for checkout or if the user wants just a few days for printing. Also, the binary output of each variable is controlled by JP(1,) (see Record Type 13). Because of this, the user can print for viewing the variables used in computation, and both print and write the postprocessed variables.

If a variable is identified in Record Type 13 that is available on the input sequential file, then the data for that variable will be "copied" (for only the stations in the U710 station list) to the output file and/or printed. If the variable is not available on the input file, it will be computed in a routine through subroutine OPTX, provided an appropriate subroutine is available to be called by OPTX and the switching is provided in OPTX. Because these computation routines are used by both U710 (which has sequential input) and U910 (which has random access input), the data are returned in the computational subroutines through a subroutine RETVEC. RETVEC first looks in the internal random access file with GFETCH; if it is not there, it looks in the external random access file with CONST.

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to

4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U710 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 3). Presumably, the only time 9997 will occur is when operational forecasts are input or U700 produces test forecasts. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. This probably means a set of dates should be supplied to U710 for which it is known there are good data. Then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

On Day 1, GFETCH is entered directly for every variable because it is not yet known when OPTX must be entered. ("Day 1 is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) <u>is not</u> counted as an error in VRBL71 for Day 1. It <u>is</u> counted as an error in VRBL72 (through OPTX), because GFETCH should not be entered directly when OPTX is needed.

When a point binary is needed, it will be computed in U710 if possible (i.e., the "basic" variable is available) even though it may exist on an input file. However, other computed variables, such as sine and cosine functions are used from the input file if they exist, even though they might be computable in U710.

The sequential data input(s) can contain constant data taken from the MOS-2000 External Random Access File System by U201. U201 can access up to 5 such data sets if necessary. Alternatively, U700 can take constant data directly from the MOS-2000 External Random Access files; all such access is through the subroutine CONST. It is possible that the IDs of such data have "B" = 0 with a non-zero threshold, indicating these are relative frequencies computed with that threshold. It is possible, but would be highly unlikely, that no data on sequential files be used, and all input furnished be on random access files. See "Restrictions" for unit numbers and associated uses.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 10 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

<u>SETTING UP THE DRIVER DRU710</u>

The preparation of the driver for a particular U710 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the directory (read on unit KFILD(2)) unless, of course, the station directory is to be used as the station list (see ND5).
ND2 -	Maximum number of variables that can be dealt with in a series. This is mainly for probability forecasts.
ND3 -	Not used.
ND4 -	The maximum number of variables that can be dealt with.
ND5 -	Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.
ND6 -	Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc., even though the same unit number might be used.
ND7 -	The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.
ND8 -	The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU710 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

WRITING NEW SUBROUTINES

Computational routines are used to do the postprocessing. An "option" subroutine, OPTX, is provided for switching to the appropriate routine. Each postprocessing routine will characteristically have the following structure:

1. A table listing the IDs of the postprocessed variables that can be handled in the routine, together with the corresponding IDs of the variable(s) needed for the processing.

2. Composition of the IDs of the variable(s) needed for the particular variable being processed in the call, and the fetching of the data from either the Internal or External Storage System by subroutine RETVEC.

3. Computation of the variable(s).

Routines NMLPRB and TRUNCP are available in the U710 library for patterns; they contain error checking, etc. NCAT is not used unless a series of variables is to be returned by the routine.

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib.dru710, dataset 'u710.com', which includes the libraries 'home21.tdllib.u710lib' and 'home21.tdllib.moslib' in that order, all on blizzard.

<u>LANGUAGE</u>: FORTRAN77 with some HP extensions.

LOCATION: u660lib. The driver is in dru710.

U850

VERIFIES FORECASTS

Harry R. Glahn
April 15, 1998

PURPOSE: U850 collates data from a variety of MOS-2000 vector inputs packed in TDLPACK format, and "verifies" one set of data with another. Usually, the data verified are called herein "forecasts" but have some characteristics of "predictors" in other programs. The verifying data are usually called "observations" but have some characteristics of "predictands" in other programs. Seven verification packages are provided and each may have more than one output package associated with it. Different output packages gives the convenience of printing verification "scores" in different formats. There is a capability to write scores in ASCII in a standard format to IP(17). Because the packed input data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in a random access file; these data are also packed in the TDLPACK format. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U850 uses a driver DRU850 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U850 (actually, subroutine) for every application. U850 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. A variable ITAU( ) associated with each variable ID is used for the "lookahead" feature generally necessary for verifying observations (or predictands). An explanation of the scores and output packages is given in Appendices I and II. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U850.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU850.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U850" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U850 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U850HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U850 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). The values that have been defined are indicated for values of J below:

(1) =All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU850 DATA statement = 12. Changing the default unit number allows multiple runs of U850 or other programs within the same directory without overwriting.

(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).

(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).

(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).

(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).

(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.

(7) = The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).

(8) = The variable ID list in summary form. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)

(9) = The variable list in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below). Note that a full list of the IDs will be given even if IP(6-9) =0.

(10) = The variable ID's for the first day (day 1) as read from the archive tapes. This is just a list of all the variables on the input files.

(11) = The variable ID's of the archived data actually needed, in order as they appear on the archive files for day 1.

(12) = The list(s) of stations on the input file(s).

(13) = The taus as read and the list of variables needed.

(14) = A diagnostic will be provided when there are no data for a particular input file for a desired date/time. With tape switching, this may not be of much use, and could be misleading.

(15) = Total variable list, including NTYPVR, the "type" of variable and NWHERE, where the variable is to be found.

(16) = All or a portion of the data values in the AA(,) matrix. For each case (cycle), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The <u>print is by variable</u> controlled by

JP(,) (see Record Type 16 below), then the station values are printed in the order dealt with in U850 (see IP(5)). Except for a very few days or a few variables and stations, this would produce voluminous files.

(17) = The scores that are designated for computing and printing to KFILDO are also written an a stardard ASCII format (see Appendix V).

(23) = Information concerning opening and closing of files.

(25) = IDs for variables in LSTORE(,) after GCPAC discards those not needed for the next cycle when the number of cycles is $\leq$ 5. Also, with the /D compile option, lists the variables in LSTORE(,) before the discard process carried out by LMSTR6 and GCPAC, not contingent on the number of cycles.

For checkout, it may be advisable to set all these values to the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (7(I10/),F10.0/(I10)) Control Parameters

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

KSKIP - When nonzero, indicates the output file on Unit KFILIO (see Record Type 8) is to be moved forward until all data for date/time KSKIP have been skipped. KSKIP is input as either YYYYMMDDHH or YYMMDDHH, and then used as YYYYMMDDHH. Note that when KSKIP > 0, there must be data with a directory record on the output file. Otherwise, the first read will be attempted and the program judges non-matching stations in the directory. Also, KSKIP must be < IDATE(1) (see Record Type 5). Not yet implemented.

NSKIP - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

NSTIND - Indicates whether (=1) or not (=0) scores for individual stations are to be computed. Overall scores and scores for groups (if there is more than one) will always be computed and output.

INCCYL - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. Because of the lookahead feature required for observations, and to maintain efficiency, no date/time should be used that cannot be arrived at by successively adding INCCYL to the first date read.

That is, if the first date/time is Jan. 1, 1996, 00 UTC and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship.

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

PXMISS - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value.

IPRINT - The number of cycles of data to write for printing to unit IP(16) under the format control and JP(,) provided with each variable (see Record Types 16, 17, and 18). IPRINT is also furnished through subroutine FILLAA to GTVECT, and when > 0 (=0) will print (will not print) a diagnostic when a variable cannot be found. For this purpose, the value if IPRINT acts as a binary switch and IP(16) and JD(,) are not used. This is an unfortunate double use of IPRINT.

ICHARS - The number of characters of call letters to print when printing is indicated by JP(,). This is constrained to be between 4 and 8 inclusive.

LNGTH - The line length for printing to Unit IP(16). For a line printer, 132 is appropriate; for a laser printer, 80 may be desirable. For some output packages, if the line length is less than what would be required for all variables, no print occurs.)

NROUND - Rounding parameter.
0 = No rounding.
1 = Rounding to the nearest hundredth.
2 = Rounding to the nearest tenth.
3 = Rounding to the nearest whole number.
4 = Rounding to the nearest ten.
Note that NROUND applies to all scores for this run, as documented in Appendix I for the individual output packages.

Record Type 4 - Format (I3,4XA60) Date List File

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) Date List

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U850 stops. Date/times should not be closer together than INCCYL and must always be derivable by successively incrementing IDATE(1) by INCCYL. For example, if the first date/time is Jan. 1, 1996, 00 UTC and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be > KSKIP (see Record Type 3).

Record Type 6 - Format (I3,4XA60) Vector Input Data Files

This group of records identifies the data sets from which the point data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

KFILIN(J) - Unit numbers for the data files.

NAMIN(J) - Names of files where these data reside. (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. (A complication is the necessity for the lookahead feature for observations.) When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4XA60) Random Access Files

This group of records identifies the MOS-2000 random access data sets from which constant (and possibly other) data are read. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. If no data are needed from a random access file, only the terminator is necessary.

KFILRA(J) - Unit numbers for the random access constant files (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

RACESS(J) - Names of files of constant data (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) Binary Output File

This record (plus the terminator record) identifies the binary output data set. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. The scores computed are packed and written in vector form as are all other MOS-2000 vector records. (The actual writing has not been implemented, but this record type must be in the input, and the file, if one is designated, is opened as 'NEW'.)

KFILIO - Unit number for the binary output file.

OUTNAM - Name of file of binary output. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) Station and Location Files

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Types 10A and 10B - Formats(7(A8,1X)) and (A30) Station Lists and Names

This group of records identifies the (groups of) stations and group names for which forecasts are to be verified. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which forecasts are to be verified (K=1,NSTA). This list (**Record Type 10A)** is read within subroutine RDSTNA or RDSTNL by RDC, which eliminates any blanks found in the input. Duplicate stations in the list for a group are kept, but a diagnostic is furnished on unit IP(5). Note that this diagnostic applies only to duplicates within a group, not from group to group. For NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. Following a group terminator is the group name (**Record Type 10B**) consisting of up to 30 characters; this is to identify the group on printout. An empty set terminates the station input. That is, the last group and its terminator and name must be followed by another terminator signifying an empty set. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) Variable ID File

This record (plus the terminator record) identifies the file from which the variable ID's (forecasts, observations, and matching data and projections for each) are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's and projections.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) Variable Constants File

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

The following Record Types occur in groups, and define the score groups to compute, the scores to compute within those groups, the variables to use, and the output packages for the groups. Reading of Record Types 13 through 18 continues until the terminator SCOREG = '999999' (instead of SCOREG = 'SGROUP') is reached in Record Type 13.

In addition to the possibility of the sequence of Record Types 13 through 18 being repeated, the pairs of Record Types 14 and 15 can be repeated within that series. That is, for the score group number defined in Record Type 13 and the variables defined in Record Types 16 through 18, there can be more than one output package and scores defined for each by the Record Types 14 and 15. Pairs of Record Types 14 and 15 are read until the terminator SCOREN = '999999' (instead of SCOREN = 'OUTPUT') is reached.

Also, each of Record Types 16, 17, and 18 can have subgroups of records, each ending with the terminator 888888. The number of subgroups in records Types 16 and 17 must be equal, and the number of subgroups in Record Type 18 must be that same number or be zero.

The score group numbers, the output packages available, and the scores for each group are explained in Appendix 1. The scores are defined in Appendix 2.

Record Type 13 - Format (A6,I4) Score Group Number

This record identifies the score group number.

SCOREG - The characters 'SGROUP' to define this record.

ISSGP - The number of the score group (see Appendix I for definitions).

Note that Record Types 14 and 15 can be repeated, and must be terminated with '999999'.

Note that Record Types 14 and 15 can be repeated.

Record Type 14 - Format (A6,I4) Output Package Number

This record identifies the output package number to use for the score group number just read.

SCOREN - The characters 'OUTPUT' to define this record.

IDSOUT - The number of the output package for this group (see Appendix I for definitions).

When '999999' is encountered instead of 'OUTPUT', the reading of Record Types 14 and 15 is terminated for this score group.

Record Type 15 - Format (A6,30I4) Score Numbers

This record identifies the score numbers to output for the score group number and output package just read.

SCORES - The characters 'SCORES' to define this record.

IDSCOR - The numbers of the scores to output for this group and output package (see Appendix I for definitions).

To reapeat, Record Types 14 and 15 can be repeated, and must be terminated with '999999'.

Record Type 16 - Format Forecasts to Verify
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4XI2,4XI3,8XA1,I2,1XI1,1X30A1)

In this Record Type 16, each group of records ending with the terminator 888888 contains the ID's of the forecasts to verify for a particular projection I. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1, ,1,I) = 888888 or 999999 is reached. An 888888 indicates this is the end of the forecast IDs for this particular projection I; IDs for another projection can follow. A 999999 indicates this is the end of the forecasts. Maximum number of records, sans the terminator record, = ND4. This Record Type 16 is read by subroutine RD85 called by RDVR85. Upon completion of reading this record type, N=1,NVRBL(1). Note that the format and information for ID(, , ,) are exactly the same as for reading predictors in U201. Upon finding the terminator 999999, I = 1, NPROJ. Some score groups use only one variable to be verified by the observation per forecast system (e.g., for Score Group 1, each variable read here is a forecast system to be verified for a particular projection). However, some score groups may use more than one variable per forecast system. See Appendix I for more details on each score group.

ID(J,N,1,I) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

JP(N,1) - JP(N,1) indicates whether (>0) or not (=0) variable N will be written to Unit IP(16) with the format provided. Note that JP(,) does not vary by projection; the values read for projection 1 control the print.

ITAU(N,1,I) - The number of hours to add to NDATE (the date being processed, see Record Type 5) to get the variable N. That is, the tau in the ID is left intact, and ITAU is used to "look ahead," when necessary.

CFMT(N,1) - The format descriptor for writing on Unit IP(16) when JP(N,1) > 0. Must be either "F" or "I". Note that CFMT(,) does not vary by projection; the values read for projection 1 control the print.

IWDTH(N,1) - The field width for writing on Unit IP(16) when JP(N,1) > 0. Must be $\leq$ 30. When $\geq$ 10, the ID(,N,1,I) will be printed as a heading in addition to HEAD(,N,1,I) (see below). IWDTH(N,1) also controls the print width for some output packages for Score Groups 1 and 7 (see Appendix I). Note that IWDTH(,) does not vary by projection; the values read for projection 1 control the print.

IPREC(N,1) - The precision for writing on Unit IP(16) when JD(N,1) > 0. For an "F" format, this is the number of digits after the decimal point. For an "I" format, it is the number of digits always written. Note that for a "zero" to be printed, IPREC(,1) must be > 0; otherwise, zero will be printed as a blank. Note that IPREC(,) does not vary by projection; the values read for projection 1 control the print. With some compilers, it may not be possible to mix I and F formats in a socre group.

HEAD(J,N,1,I) - The column heading for writing to Unit IP(16) when JP(N,1) > 0. Limited to J=1,30 characters. All 30 characters are read, but only IWDTH(N,1) characters are printed to Unit IP(16). HEAD(,N,1) is right justified when writing to Unit IP(16). For printing of scores, HEAD(,N,1,I) is left justified, and in most instances no more than 11 to 13 characters can be accommodated when printing "IMP OVER".

Record Type 17 - Format Verifying Observations
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4XI2,4XI3,8XA1,I2,1XI1,1X30A1)

In this Record Type 17, each group of records ending with the terminator 888888 contains the ID's of the verifying observation(s) for a particular projection I. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1, ,2,I) = 888888 or 999999 is reached. An 888888 indicates this is the end of the observation IDs for this particular projection I; IDs for another projection can follow. A 999999 indicates this is the end of the observations. Maximum number of records, sans the terminator record, = ND4. This Record Type 17 is read by subroutine RD85 called by RDVR85. Upon completion of reading this record type, N=1,NVRBL(2). The information read is explained in Record Type 16 and is not repeated here. Some score groups use only one variable for the verifying "observation" per projection. However, some score groups (e.g., Score Group 2) may use more than one. See Appendix I for more details per score group. The number of projections to verify is the number of subgroups, each ending with an 888888, and must be the same in Record Types 16 and 17. That is, the first subgroup in Record Type 16 is verified by the first subgroup in Record Type 17, etc. Note that there may be multiple "systems" in Record Type 16, but all are verified by the one set of observations in the corresponding subgroup in Record Type 17.

ID(J,N,2,I) - See Record Type 16.

JP(N,2) - See Record Type 16.

ITAU(N,2,I) - See Record Type 16.

CFMT(N,2) - See Record Type 16.

IWDTH(N,2) - See Record Type 16.

IPREC(N,2) - See Record Type 16.

HEAD(3,N,2,I) - See Record Type 16.

Record Type 18 - Format Matching Variables
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4XI2,4XI3,8XA1,I2,1XI1,1X30A1)

In this Record Type 18, each group of records ending with the terminator 888888 contains the ID's of variables that must have non-missing values for the forecasts to be verified for a particular projection. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1, ,3,I) = 888888 or 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 18 is read by subroutine RD85 called by RDVR85. Upon completion of reading this record type, N=1,NVRBL(3). The information read is explained in Record Type 16 and is not repeated here. It may be that there are no IDs in this Record Type 18, and the only record contains the terminator 999999. This means that none of the projections to verify have matching variables. However, if there are entries here, then the number of subgroups (number of projections) must be the same as in Record Types 16 and 17. The binary variable B = IDPARS(3) can be used for stratification (see Comments section).

ID(J,N,3,I) - See Record Type 16.

JP(N,3) - See Record Type 16.

ITAU(N,3,I) - See Record Type 16.

CFMT(N,3) - See Record Type 16.

IWDTH(N,3) - See Record Type 16.

IPREC(N,3) - See Record Type 16.

HEAD(J,N,3,I) - See Record Type 16.

Note: U602 may produce an equation with zero coefficients for which no values are available and it can't be evaluated by U700. Therefore, U602 may have more cases that U700. To get U700/U850 results to match U602, stratification variables to include all U602 predictands may be necessary.

CONTROL FILE INPUT: (Name read from U850.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) Date List

When the dates are not provided in file U850.CN, this group of records determines the date/times for which data are to be input and processed.

If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU850), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list <u>must</u> be available or U850 stops. Date/times should not be closer together than INCCYL and must be derivable by successively incrementing IDATE(1) by INCCYL. For example, if the first date/time is Jan. 1, 1996, 00 UTC and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

<u>CONTROL FILE INPUT</u>: (Name read from U850.CN) (Unit = KFILD(1))

Record Types 1A and 1B - Formats (7(A8,1X)) and (A30) <u>Station Lists and Names</u>

<u>When the station list is not provided in file U850.CN</u>, this group of records identifies the stations (or locations) and group names to be included in the regression analysis. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

<u>CCALL</u>(K) - Call letters (or other 8-character location designator) of stations (or locations) for which interpolated values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 10A, control file 'U850.CN', unit KFILDI for additional information.) (CHARACTER*8)

<u>CONTROL FILE INPUT</u>: (Name read from U850.CN) (Unit = KFILD(2))

Record Type 1 - Format <u>Station Locations</u>
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations). The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical <u>within each group</u> no matter the order read in (see NALPH in Record Type 3). [Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. Note that alphabetization is not overall, but by group (see Record Type 10)]. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which eliminates the possibility of alphabetizing if the old 3-letter call letters were used. The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3). See Chapter 10, Station Directory, in <u>TDL Office Note</u> 00-1 for more information.

<u>CCALLD</u>(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call

letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.
SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited; no terminator is used.

CONTROL FILE INPUT: (Name read from U850.CN) (Unit = KFILP)

Record Type 1 - Format Forecasts, Observations, and Matching Variables
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X4I2,9XA1,I2,1XI1,1X30A1)

When the variables to use in the run are not in file U850.CN, this group of records is read in trios and contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1, ,1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RD85 called by RDVR85. Note that the format and information for ID(, , ,) are exactly the same as for reading predictors in U201. See Control File Input for U850.CN, Record Type 16, unit KFILDI above for more detailed information; the format is exactly the same.

Note that this Record Type 1 is repeated twice (three times total) for each score group defined (see Record Types 16, 17, and 18).

CONTROL FILE INPUT: (Name read from U850.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U850 users, and is expected to be a separate file.

Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

<u>IDTEMP</u>(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

<u>IDTEMP</u>(J) - These 3 words (J=2,4) are currently not used, but are meant to correspond to the ID words 2-4.

<u>PLAINT</u> - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. For forecasts and observations, the first 4 characters can indicate the model (e.g., ETA) from which the forecasts were made, or for the AEV data, MOS or LCL can be used. (CHARACTER*32)

<u>ISCALD</u> - This is the decimal scale factor to use when packing the data for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer.

Other processing occurs with the reading of these records in RDVR85, much of it associated with the plain language description. This is done so that these records can be somewhat generic and apply to many variables as defined in ID(, ,1). See subroutine SETPLN for more information on the use of plain language.

<u>DATA INPUT</u>:

All data input to U850 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format, except for the call letters (directory) record. The sequential files can have multiple directory records, each applying to the data until an end of file or another directory is encountered; the random access files have only one directory record, each of which applies to all data in the file. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U850.CN in Record Types 6 and 7.

A. <u>SEQUENTIAL VECTOR DATA</u>

One or more sources of vector data, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U850.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

B. RANDOM ACCESS VECTOR DATA

Up to 5 sources of random access data are accommodated; however, it is unlikely more than 1 or 2 will be needed. These data exist in the MOS-2000 External Random Access File System. These data are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

DATA OUTPUT

There are two forms of data output, besides the diagnostics and other information on Units KFILDO and IP( ).

A. ASCII FOR VIEWING OR PRINTING DATA

A maximum of NPRINT (see Record Type 3) or NDATES (see Record Type 5) cycles of data will be written to the file on Unit IP(16) under control of the format provided with each variable N, the variables written controlled by JP(N,) (see Record Type 16). The data columns are headed by HEAD(,N, ,) (see Record Type 16). Also, provided the column width specified by IWDTH(N,) (see Record Type 16) is $\geq$ 10 characters, the 4-word ID will be written. Listing is by station group. If a line length for printing is not sufficient for all variables, printing may not be done. Line length is specified as LNGTH characters in Record Type 3.

B. ASCII FOR VIEWING OR PRINTING SCORES

The scores designated in Record Type 15 and the output packages designated in Record Type 14 are written for viewing and/or printing on KFILDO.

C. ASCII FOR COMPUTER READING OF SCORES

The same scores printed in B. above will be written to IP(17) (when non zero) in a standard format. See Appendix V for a defintion of the format and scores.

D. BINARY MOS-2000 FORMAT (Not yet implemented)

The scores will be written to a binary file according to a designated output package.

EXAMPLE CONTROL FILE: 'U850.CN'

An example exists as file 'U850.CN' in directory home21.tldlib.dru850 on blizzard and an example is in Appendix III. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U850.CN', Record Type 1 and the definition of KFILDO in the driver DRU850. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. To disable, it is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

There is no check in U850 for a legitimate value of "B" in the first word of the ID. For a value of 3, indicating a discrete binary, the upper threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID's. If there is no lower one, the lower threshold is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999.

Since the variables are not ordered by the program (as they are in U600), if a discrete binary is desired (one with both upper and lower thresholds), these variable ID's must be input in sequence and in the correct order (lower threshold first). To make sure all is well, check the thresholds on Unit IP(8) or IP(9). Since scores are identified and variables read in groups, the order of the variables within a score group will usually be important and will be dictated by the subroutines computing the accumulants and scores (see the appendix for further explanation).

It is necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles.

Record Types 16, 17, and 18 contain the IDs for the forecasts, verifying observations, and "matching" variables, respectively. Each of these types can contain more than one set of IDs ending with 888888, with the ending of the type signaled by 999999. Each of the sets ending with 888888 pertains to a particular "projection," although there can actually be different taus within that projection. However, the thresholds for the scores requiring them are not carried for each projection; rather the last set read are those used. That is, if two sets (projections) were to be verified and the forecasts (or observations) had different thresholds in word 4 of the ID, the first set would be ignored, and the last set used. That is, whenever more than one projection is to be verified, it is expected that the thresholds will be the same (presumably the other identifying information would be the same also, except for the taus).

Because of the way GTVECT and GFETCH work, a variable with an RR [from pervious computation (in U201)], cannot be used with ITAU > 0.

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)

46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

U850 would likely need only Unit No. 45, but one day (cycle) of forecasts on Units 48 or 49 could be verified.

COMMENTS

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U850 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 3). Presumably, the only time 9997 will occur is when operational forecasts are input or U700 produces test forecasts. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such. These values will be used the same way no matter how they are packed.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. This probably means a set of dates should be supplied to U850 for which it is known there are good data; then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

On Day 1, a routine SETWHR is used to determine which of the variables are available without computation through OPTX.

When a point binary is needed, it will be computed in U850 if possible (i.e., the "basic" variable is available) even though it may exist on an input file. However, other computed variables, such as sine and cosine functions are used from the input file if they exist, even though they might be computable in U850.

The sequential data input(s) can contain constant data taken from the MOS-2000 External Random Access File System by U201. U201 can access up to 5 such data sets if necessary. Alternatively, U850 can take constant data directly from the MOS-2000 External Random Access files; all such access is through the subroutine CONST. It is possible that the IDs of such data have "B" = 0 with a non-zero threshold, indicating these are relative frequencies computed with that threshold. It is possible, but would be highly unlikely, that no data on sequential files be used, and all input furnished be on random access files. See "Restrictions" for unit numbers and associated uses.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage <u>after</u> Day 1.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 10 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

As a special feature for U850, B = IDPARS(3) in ID(1) can be 8 or 9 in addition to the usual values of 0, 1, 2, 3, or 5. This is used for stratification. When a matching variable has B = 8, all non missing values of that variable greater than the threshold in ID(4) will be set to 9999 (missing) and all other values will be set to 0. (Setting to zero is not important in U850, but it is possible this capability can be used in U600, and there the zeros would be important.) When a matching variable has B = 9, all non missing values of that variable less than the threshold in ID(4) will be set to 9999 and all other values will be set to 0.

As an additional special feature, B = IDPARS(3) in ID(1) can be 6 or 7 to accommodate "OR" matching variables. B = 6 or 7 operates essentially like B = 8 or 9, respectively, except that logic in U850 will use that case if <u>either</u> of the variables is "non-missing." For B = 8 or 9, <u>all</u> such variables have to be non-missing for the case to be used.

The ASCII output on IP(17) is in a standard format to provide a better source of scores for reading by computer than the printing to KFILDO (see Appendix V).

<u>SETTING UP THE DRIVER DRU850</u>

The preparation of the driver for a particular U850 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine.
ND1 -	Maximum number of stations (or points) that can be dealt with.

Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) unless, of course, the station directory is to be used as the station list (see ND5).

ND2 -	Maximum number of projections that can be dealt with for forecasts, observations, or matching data. Note that this is for each, not the total of all.
ND3 -	Not used.
ND4 -	The maximum number of forecasts, observations, or matching variables that can be dealt with.
ND5 -	Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1 and for safety should be $\geq$ ND12.
ND6 -	Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc., even though the same unit number might be used.
ND7 -	The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.
ND8 -	The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.
ND9 -	The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times.
ND10 -	The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.
ND11 -	The maximum number of score groups.
ND12 -	The size of the area to hold accumulants for a score group.
ND13 -	The maximum number of scores (numbers) to output for each of the score groups and output packages.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU850 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

PROGRAM STRUCTURE

See Appendix IV.

WRITING NEW SUBROUTINES

It is not expected that computational routines will be used to any great extent in U850. Rather, computations, except for point binaries, are expected to be done in U201. Any computations that would be done in U850 would have to be done on point data (not gridpoint data) because gridpoint data are not accommodated in U850. Even so, an "option" subroutine, OPTX, is provided for possible use. For instance, in verifying wind direction, the direction might not be available directly, but need to be computed from u and v components through OPTX. (In the case of development for gridpoints, the gridpoints are given identifications in the same way as other locations, such as stations.)

If a new score is needed, it may be possible to add it to an existing score group. However, be aware that a difference in print output may affect scripts prepared for reading the ASCII file. It may be an entirely new score group will be needed. This will likely involve changes to SCOPTA, SCOPTB, SCOPTC, and SCOPTD as well as new routines SCRXXA, SCRXXB, and SCRXXC.

NONSYSTEM ROUTINES USED

Use the load line in home21.tldlib.dru850, dataset 'u850.com', which includes the libraries 'home21.tdllib.u850lib' and 'home21.tdllib.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.
LOCATION: /mdl/mos2k/mdllib/u850lib. The driver is in
/mdl/mos2k/mdllib/u855lib/RUN/dru850.

APPENDIX I

SCORE GROUPS, SCORES, AND OUTPUT PACKAGES

The scores are produced and output in groups. Each score group has one or more output packages, giving flexibility in the form of the output. The score group numbers are read in Record Type 13; the associated scores to be output are read in Record Type 15. The output package to be used is read in the associated Record Type 14.

When more than one "system" is verified (more than one set of forecasts) with the same score group, for some scores an "improvement" over the first "system" (set of forecasts) is calculated and (can be) output. For instance, if the first set of forecasts were climatological relative frequencies and the second set were probability forecasts, an improvement over climatology would be calculated.

Score Group numbers provided are 1, 2, 3, 4, 5, 6, and 7. Each output package provides for printing overall scores (which is always done) scores by groups of stations (when there is more than one group--see Record Type 10), and by station (when desired--see Record Type 1, NSTIND) for each projection desired (see Record Type 16).

It is possible to verify a forecast valid at a particular date/time with an observation taken at (valid for) another date/time. Also, the AEV data are labeled and stored somewhat differently from other data. For instance, a 9-h MOS forecast of ceiling height produced at a particular date/time is normally written and archived at that same date/time, and the verifying observation would be found with the lookahead feature with a tau in the observation ID of zero [IDPARS(12, ,2,) = 0] and the corresponding ITAU( ) (see Record Type 16) = 9. However, the same AEV forecast would also be archived at the date/time of production, but the verifying observation, already collated on the AEV archive, is stored at a date/time of plus 6 hours (because that is where the local forecasts valid at the same time are archived) and with a tau of 3 (because that is the projection of the "matching" local forecast) [IDPARS(12, ,2,) = 3]. This observation would be accessed with that tau in the ID, and with ITAU( ) = 0. This all calls into question how the projections should be labeled in the output.

The method adopted is to output the forecast projection as JJJ in the verifying observation ID [i.e., IDPARS(12,)] plus ITAU( ) of the verifying observation minus ITAU( ) of the forecast. This seems to work for all "normal" verifications including for AEV data. If one verifies a forecast valid at a particular time with an observation other than for the valid time, care must be taken to interpret the results correctly, especially for the AEV data.

The NROUND parameter (See Record Type 3) is relevant and/or applied differently for different score groups. NROUND applies after any computation, including the making of binaries, is done to produce the data for the production of scores. That is, the data are readied for computation of accumulants for scores, and rounding, if any, is done then. See the individual score groups in this appendix for additional information.

Score Group 1

This score group is generally for continuous forecasts, for example temperature or wind speed. Usually, these forecasts and observations would not have thresholds associated with them. Each forecast variable read (Record Type 16) for a particular projection is a system to be verified. So, the number of variables read in Record Type 16 (and in Record Type 17) is the number of systems to be verified. The single corresponding variable read in Record Type 17 is the matching observation for each such system for that projection.

NROUND (See Record Type 3) applies to both forecasts and observations.

Score No.	Explanation
1	Sample size (always output whether or not specified).
2	Mean of observations.
3	Variance of observations.
4	Standard deviation of observations.
5	Mean of forecasts.
6	Variance of forecasts.
7	Standard deviation of forecasts.
8	Bias of forecasts (fcst-obs).
9	Mean absolute error (MAE) of forecasts.
10	Mean square error of forecasts.
11	Root mean square error (RMSE) of forecasts.
12	Log score.
13	Fractional improvement of log score.
14	Ratio of variance of forecasts to observations.
15	Fractional improvement of MAE.
16	Fractional improvement of RMSE.
17	Correlation coefficient of forecasts with observations.

Output Package	
01	Prints individual scores (e.g., mean for forecasts) for all projections on the same line if there is room, the maximum line length being determined by LNGTH (see Record Type 1). If there is not room on the line, this output package defaults to output package No. 02.
02	Prints scores one projection per line. That is, all scores for the first projection are printed, then all scores for the second projection, etc. This allows for quick comparison among groups for a particular projection.
03	Prints all scores on one line by projection. That is, for a particular projection, all the scores (1 through 17) specified (see Record Type 15) for all systems are put on one line. This may make for a very long line, and LNGTH is allowed to be exceeded; if this is the case, a diagnostic is printed, and output to a printer may not give pleasing results. For this output package, a particular score number may trigger print for related scores, according to the following table:

Score No.	Explanation
1	Sample size (always output whether or not specified).
2 or 5	Mean of observations and forecasts.
3 or 6	Variance of observations and forecasts.
4 or 7	Standard deviation of observations and forecasts.
8	Mean of observations and forecasts and bias of forecasts (fcst-obs).
9	Mean absolute error (MAE) of forecasts.
10	Mean square error of forecasts.
11	Root mean square error (RMSE) of forecasts.
12	Log score.
13	Log score and fractional improvement of log score.
14	Variance of observations and forecasts and ratio of variance of forecasts to observations.
15	MAE of forecasts and fractional improvement of MAE.
16	RMSE of forecasts and fractional improvement of RMSE.
17	Correlation coefficient of forecasts with observations.

04 Prints scores one projection per line. However, this is different from Output Package 02 in that the projections are grouped. That is, output package 04 prints all projections for overall, then all projections for each forecast group (when there is more than one), etc. This allows for quick comparison of projections for a particular group or station.

Note:

- o MPSKTS can be used to convert m/s to kt for wind speed with the appropriate CCCFFF.
- o ZRONEG can be used to set negative wind speeds to zero with the appropriate CCCFFF.

Score Group 2

This score group is generally for continuous forecasts (including probability forecasts) that are put into categorical form, for example wind speed or probabilities of ceiling height. These variables would usually have a threshold associated with them; each variable would be input for each threshold needed, and would be input in ascending order of thresholds. The discrete categories necessary for forming a contingency table are determined by the series of inputs. The upper threshold of a category is the one in the associated ID; the lower threshold of that category for that variable is the threshold of the previously read variable with the same ID (except, of course, for the threshold). If there is only one threshold for a variable, or if it is the first threshold for that variable, the lower threshold is automatically set at a large negative value. The last threshold read for a variable with more than one threshold should normally be 9999. A set of forecast variables read (Record type 16) for a particular projection, each of which is the same except for the threshold, comprise the forecast system to be verified. When more than one system is to be verified for a projection, one set immediately follows another. The single set of observation variables read (Record Type 17) is used to verify each system for the corresponding projection. It is expected that the number of categories for the observations is the same as the number of categories for the forecasts.

NROUND (See Record Type 3) is not relevant because the variables as specified in Record Types 16 and 17 should be binary.

The forecast and observation thresholds for each system verified are checked with those for each other system, and the thresholds for forecasts are checked with those for the verifying observations. If there are differences, a **** diagnostic is written. This may not be an error, but is a safeguard. For instance, the MOS ceiling forecasts are in category numbers, while the local ceiling forecasts are in hundreds of feet.

Score No.	Explanation
21	Contingency table of forecasts and observations.
22	Percent correct and improvement.
23	Heidke skill score and improvement.
24	Threat score and improvement, Probability of Detection (POD), and False Alarm Ratio (FAR) for category 1 (lowest category).
25	Same as 24 except for categories 1 and 2 combined.
26	Same as 24 except for the upper category.
27	Same as 24 except for the upper two categories combined.
28	Same as 24 except for the upper three categories combined.
29	Log score and fractional improvement.

Output Package	
21	Prints scores for all projections together. That is, overall scores for all projections are output together, then groups of stations for all projections, etc.
22	Prints all scores grouped by projection. That is, all scores are printed for the first projection, then for the second projection, etc.

Note:

o Subroutine NORWND can be used to calculate a contingency table and skill score for wind direction with the appropriate CCCFFF. Currently the values of CCCFFF accommodated are:

```
004 203
004 204
004 205
704 203
204 203
204 205
204 230
204 235
```

Score Group 3

This score group is for wind direction forecasts. Each pair of input forecasts read (Record Type 16) for a particular projection is a system, a direction followed by a speed. When more than one system is to be verified, one pair immediately follows another. Each verifying observation for the corresponding projection is also composed of a direction followed and a speed. So, the number of systems to be verified for a projection is the number of pairs of variables in Record Type 16. Thresholds are not used; rather the errors in direction are put into 9 categories defined internally.

NROUND (See Record Type 3) applies to wind direction only.

Score No.	Explanation
31	The relative frequencies (RF), cumulative relative frequencies (CRF), improvement of the system over the first system in terms of CRF, and frequencies of direction errors in 9 categories, the upper inclusive category boundaries being 0, 10, 20, 30, 60, 90, 120, and 150 degrees.
32	The same as Score No. 31, except only for cases when the observed speed is $\geq$ 10 kt.
33	The same as Score No. 31, except only for cases when both the observed and forecast speed is $\geq$ 10 kt. Note that this can create an unmatched sample for the different systems, because each is based on its own speed forecast.
34	MAE for wind direction.
35	Same as 34, except only when observed wind speed $\geq$ 10 kt.
36	Same as 34, except only when both observed and forecast wind speed $\geq$ 10 kt. Note that this will likely give unmatched samples when more than one system is being verified.

Output Package	
31	Prints scores for all projections together. That is, overall scores for all projections are output together, then groups of stations for all projections, etc.
32	Prints all scores grouped by projection. That is, all scores are printed for the first projection, then for the second projection, etc.

Score Group 4

This score group is for probability forecasts of a single event. Each forecast variable read in Record Type 16 for a particular projection is a system, and the corresponding observation variable for that projection read in Record Type 17 is the verifying observation for each system. So, the number of variables read in Record Type 16 is the number of systems to be verified for that projection. No threshold is used with the forecast; however, a threshold may be needed for the observation to define the event being forecast. That is, the observation might be precipitation, for which the threshold might be 0.01 inches, but could be some other value. If the observation is already binary, no threshold is needed.

NROUND (See Record Type 3) applies to forecasts only because the observations are binary.

Score No.	Explanation
41	P-score and fractional improvement.
42	Brier score and fractional improvement.
43	Event relative frequencies in 10 percent increments.
44	Same as 41, except forecasts constrained to 0 to 1 range.
45	Same as 42, except forecasts constrained to 0 to 1 range.
46	Same as 43, except forecasts constrained to 0 to 1 range.

Output Package	
41	Prints scores for all projections together. That is, overall scores for all projections are output together, then groups of stations for all projections, etc.
42	Prints all scores grouped by projection. That is, all scores are printed for the first projection, then for the second projection, etc.

Note:

- o A matching variable can be used with subroutine POPDIF to calculate scores for only those cases when the difference between MOS and local forecasts is $\geq$ some desired threshold.

Score Group 5

This score group is for probability forecasts of a set of NCAT ranked events (e.g., discrete categories of ceiling height). For verification of "old" forecasts, the NCAT probability forecasts would not have a threshold in the ID, but the category number would be in ID(2) and IDPARS(6). A system for a particular projection is determined by a sequence of variables being the same except for IDPARS(6). IDPARS(6) would range upward from 1 in increments of 1. For verification of MOS-2000 forecasts, the NCAT probability forecasts would normally have a threshold in the ID, and there would be no category number in ID(2) or IDPARS(6). The thresholds would be in an upward numerical sequence. The verifying observation for that projection would normally have the upper category limits in the IDs for it to be made into discrete binaries; the variable IDs would be identical except for the thresholds. The number of forecasts to be verified and the discrete verifying binaries must be the same. Old and new forecasts can be mixed in one run; the improvement of MOS-2000 forecasts over old forecasts can be calculated. Relative frequencies with CCC = 4XX can also be verified.

NROUND (See Record Type 3) applies to forecasts only.

The forecast and observation thresholds for each system verified are checked with those for each other system (the forecast thresholds should all be zero). If there are differences, a **** diagnostic is written. This may not be an error, but is a safeguard.

Score No.	Explanation
51	P-score and fractional improvement.
52	Same as 51, except that forecasts are constrained to 0 to 1 range.
53	Reliability of forecasts; the mean forecast, relative frequency of observations, and number of cases in 10 percent increments of the forecasts.
54	Ranked probability score, with the scores constrained to 0 to 1 range, and the cumulative frequency adjusted when necessary.

Output Package	
51	Prints scores for all projections together. That is, overall scores for all systems and then for all projections are output together, then groups of stations for all projections, etc.

Score Group 6

This score group is used to provide the frequencies and relative frequencies of errors in the forecasts. For instance, for the continuous variable max temperature, the algebraic or absolute difference of the forecast minus the observation is computed and the result put into discrete categories specified by the thresholds provided by the otherwise duplicative forecast IDs. This is accomplished through use of a subroutine DIFFV called by OPTX. In order to not have routines specific to one forecast and observation ID, an assumption was made that the forecast ID CCCFFF is 20XXXX and the verifying ID is 70XXXX. Then the IDs of the difference to input with the thresholds is either 78XXXX or 79XXXX, the 8 denoting the algebraic difference and the 9 an absolute difference. This assumption works for temperature, dew point, and wind speed, possibly the only variables needing this score group.

NROUND (See Record Type 3) is not used. The differences are put into categories, so rounding is not needed. (Rounding could be hardwired into DIFFV if rounding is needed before the differences are computed.)

The thresholds for the errors for each system verified are checked with those for each other system. If there are differences, a **** diagnostic is written. This is likely an error.

Score No.	Explanation
61	Algebraic differences are computed.
62	Same as 61, except that the differences are absolute.

Output Package	
61	Prints scores for all projections together. That is, overall scores for all systems and then for all projections are output together, then groups of stations for all projections, etc.
62	Prints all scores grouped by projection. That is, all scores are printed for the first projection, then for the second projection, etc.

Note: Only limited testing has been done, and only on AEV data.

<u>Score Group 7</u>

This score group is to provide one or more measures of the temporal consistency of one or more sets of forecasts all valid at the same time but made at different times (with, of course, different projections) (e.g., daytime max temperature forecasts valid on Dec. 10, 2004 made on 5 previous days). By definition, there is only one "projection," and for each system the projection printed will be the maximum tau for that system.

For each system, the first variable ID in Record Type 16 will have the shortest tau, and each succeeding ID for that same system will have a larger tau. Each record will have an ITAU which together with the tau will point to the score ITAU of the single verifying observation (Record Type 17). The number of forecasts per system will be determined when the taus cease to increase, or the 888888 terminator is reached if there is a single system. Then all the rest of the IDs in Record Type 16 must be in groups of that same size. For instance, if the first system (e.g., MOS) has 10 IDs, then the remaining IDs (before the 88888) in Record Type 16 must be evenly divisible by 10.

NROUND (See Record Type 3) applies to forecasts and observations.

Score No.	Explanation
71 =	FCS Score 1 with the first threshold T,
72 =	FCS Score 1 with the second threshold T,
73 =	FCS Score 1 with the third threshold T,
74 =	FCS Score 2 with the first threshold T,
75 =	FCS Score 2 with the second threshold T,
76 =	FCS Score 2 with the third threshold T,
77 =	CS

Output Package

71	Prints scores for all sets of forecasts being verified on one line.

APPENDIX II

DEFINITIONS OF SCORES

The scores computed by U855 are computed in groups. The groups were determined by similarity of computations. For instance, one group is concerned with contingency tables and the scores computed from them. All scores in a group are computed when the group is specified, but only the scores asked for are output.

The definitions and terminology follow, insofar as possible, the National Verification Plan published by the National Weather Service in June 1982 and still used as the primary basis of the National Verification Program. In the definitions, the following terminology is used:

N = sample size (number of cases)

f_i = the ith forecast,

o_i = the ith (matching) observation, and

$\sum = \sum_{i=1}^{N}$ summation over all N cases, except where indicated otherwise.

"Improvement" of a score means the improvement of the kth "system" or set of forecasts over the first. For instance, if the first set of forecasts was climatological means and the second set was MOS forecasts, an improvement of MOS over climatology would be calculated.

Score Group 1

This score group is generally for continuous forecasts, for example temperature or wind speed. Generally, these forecasts and observations would not have thresholds associated with them.

Score No.	Explanation
1	N = sample size.
2	$\overline{o} = \frac{1}{N} \sum o_i$ = mean of observations.
3	$\sigma_o^2 = \frac{1}{N} \sum (o_i - \overline{o})^2$ = variance of observations.
4	$\sigma_o = (\sigma_o^2)^{1/2}$ = standard deviation of observations.
5	$\overline{f} = \frac{1}{N} \sum f_i$ = mean of forecasts.
6	$\sigma_f^2 = \frac{1}{N} \sum (f_i - \overline{f})^2$ = variance of forecasts.
7	$\sigma_f = (\sigma_f^2)^{1/2}$ = standard deviation of forecasts.
8	$\overline{f} - \overline{o}$ = bias of forecasts.
9	$\text{MAE} = \frac{1}{N} \sum \lvert f_i - o_i \rvert$ = mean absolute error of forecasts.
10	$\text{MSE} = \frac{1}{N} \sum (f_i - o_i)^2$ = mean square error of forecasts.
11	$\text{RMSE} = (\text{MSE})^{1/2}$ = root mean square error of forecasts.
12	$\text{LS} = \frac{50}{N} \sum \log_{10} \frac{f_i}{o_i}$ = log score.
13	$\frac{\text{LS}_1 - \text{LS}_k}{\text{LS}_1}$ = fractional improvement of log score.
14	$\frac{\sigma_f^2}{\sigma_o^2}$ = ratio of variance of forecasts to observations.
15	$\frac{\text{MSE}_1 - \text{MSE}_k}{\text{MSE}_1}$ = fractional improvement of MSE.
16	$\frac{\text{RMSE}_1 - \text{RMSE}_k}{\text{RMSE}_1}$ = fractional improvement of RMSE.
17	$\text{CORR} = \frac{\overline{fo} - \overline{f}\,\overline{o}}{\sigma_f \sigma_o}$ = correlation, where $\overline{fo} = \frac{1}{N} \sum (f_i o_i)$

Score Group 2

This score group is generally for continuous forecasts, for example temperature or wind speed, that are to be put into m categories for entering into a contingency table. These forecasts and observations are expected to have the full IDs including thresholds. That is, a series of forecasts is read in which have the same IDs except for the thresholds; these define the categories. The same is true for the observations.

Score No.	Explanation
21	Contingency Table--not a score as such, but

(a) Contains information on which all the scores in this group are computed. The element X_{ij} in the table is the number of times the forecast was in the jth category and the observation was in the ith category. The row and column totals are also shown here with the subscript p.

Observed	Forecast Category				
Category	1	2	...	m	Total
1	X_{11}	X_{12}	...	X_{1m}	X_{1p}
2	X_{21}	X_{22}	...	X_{2m}	X_{2p}
.	.	.		.	.
.	.	.		.	.
.	.	.		.	.
m	X_{m1}	X_{m2}	...	X_{mm}	X_{mp}
p	X_{p1}	X_{p2}	...	X_{pm}	X_{pp}

(b) $\text{BIAS}_i = \dfrac{X_{pi}}{X_{ip}}$ = bias by category.

22 (a) $\text{FC} = \dfrac{\sum_{i=1}^{m} X_{ii}}{X_{pp}}$ = fraction correct.

(b) $\dfrac{\text{FC}_k - \text{FC}_1}{\text{F}_1}$ = fractional improvement in FC of forecast System k over that of system 1

23 (a) $SS = \frac{NC - E}{T - E}$ = Heidke skill score where

$$\text{number correct (NC)} = \sum_{i=1}^{m} X_{ii}$$

$$T = X_{pp}$$

$$E = \sum_{i=1}^{m} (X_{ip} X_{pi}) / T$$

(b) $\frac{SS_k - SS_1}{SS_1}$ = fractional improvement in SS of forecast system k over that of system 1.

The following scores are based on a contingency table of only 2 categories collapsed (if necessary) from the one shown above. For instance, for score No. 24, category 1 remains and all other categories are put into category 2.

24 (a) $TS = \frac{X_{11}}{X_{11} + X_{12} + X_{21}}$

$= \frac{X_{11}}{X_{pp} - X_{22}}$

$= \frac{X_{11}}{X_{p1} + X_{1p} - X_{11}}$ = threat score for category 1 (also called Critical Success Index, CSI).

(b) $\frac{TS_k - TS_1}{TS_1}$ = fractional improvement in TS of forecast system k over that of system 1.

(c) $\frac{X_{11}}{X_{1p}} = \frac{X_{11}}{X_{11} - X_{12}}$ = probability of detection of category 1.

(d) $FAR = \frac{X_{21}}{X_{p1}} = 1 - \frac{X_{11}}{X_{p1}} = \frac{X_{11}}{X_{11} - X_{21}}$ = false alarm ratio of category 1.

25 Same as for No. 24 except computed on lower 2 categories.

26 Same as for No. 24 except computed on upper category.

27 Same as for No. 24 except computed on upper 2 categories.

28 Same as for No. 24 except computed on upper 3 categories.

29 See scores 12 and 13 in Score Group 1.

Any time the table cannot be collapsed, the score will not be computed. For instance, score 28 cannot be computed on a three-category table.

Score Group 3

This score group is for wind direction. Each "system" has two forecast inputs, a direction in degrees and a speed (assumed to be kt) in that order, and there is a verifying observation consisting of a direction and speed in that order.

Score No.	Explanation
31	$F_i = \sum n_i$ = count of direction errors in each of 9 categories. $RF_i = F_i/N$ = relative frequency of errors in each of 9 categories. $CRF_i = \sum_{j=1}^{i} RF_{ij}$ = cumulative relative frequency of errors in each of 9 categories. $IMP_i = \frac{CRF_i - CRF_1}{CRF_1}$ = improvement in CRF over 1st system for for each of 9 categories.
32	Same as Score No. 31, except only for cases when the observed speed is $\geq$ 10 kt.
33	Same as Score No. 31, except only for cases when both the observed and forecast speed in $\geq$ 10 kt. Note that this can create an unmatched sample for the different systems, because each is based on its own speed forecast.

Score Group 4

This score group is for a probability forecast of a single event. The observation "verifying" the forecast can be continuous, and will then have a threshold associated with it which defines the "event" to which the probability relates. Or it could be a binary event, taking the values of only 0 and 1, in which case no threshold would be required.

Score No.	Explanation
41	(a) $P = \frac{2}{N}\sum(f_i - o_i)^2$ = P-score.
	(b) $\frac{P_1 - P_k}{P_1}$ = fractional improvement of P-score.
42	(a) $BS = \frac{P}{2}$ = Brier Score.
	(b) $\frac{BS_1 - BS_k}{BS_1}$ = fractional improvement of Brier-score.
43	(a) $\frac{1}{N}\sum o_i$ = Event relative frequency.
	(b) $\frac{1}{N}\sum f_i$ = Mean forecast.
	(c) Frequency of event in 10% forecast probability categories.
	(d) Relative frequency of event in 10% forecast probability categories.
	(e) Mean forecast probability in 10% forecast probability categories.
44	Same as 41 except forecasts truncated to 0 to 1 range.
45	Same as 42 except forecasts truncated to 0 to 1 range.
46	Same as 43 except forecasts truncated to 0 to 1 range.

Score Group 5

This score group is for probability forecasts of a set of m "ordered" events. The observation "verifying" the forecasts can be continuous, and will then have thresholds associated with it which defines the "events" to which the probabilities relate.

Score No.	Explanation
51	(a) $P = \frac{1}{N} \sum \sum_{j=1}^{m} (f_{ij} - o_{ij})^2$ = P-score.
	(b) $\frac{P_1 - P_k}{P_1}$ = fractional improvement of P-score.
52	Same as 51 except forecasts truncated to 0 to 1 range.
53	(a) Relative frequency of observations in each of m categories.
	(b) Mean probability forecast in each of m categories.
	(c) Frequency of observations in 10% forecast probability intervals for each of m categories.
	(d) Same as (c) except relative frequency.
	(e) Same as (c) except mean forecast probability.
54	(a) $RPS = \frac{1}{N} \sum \sum_{k=1}^{m} (\sum_{j=1}^{k} f_{ij} - \sum_{j=1}^{k} o_{ij})^2$
	(f_{ij} are truncated to range 0 to 1 and the next category in sequence adjusted by the truncated amount. As an example, if f_{i1}= -.05 and f_{i2} = .15, f_{i1} is used as 0 and f_{i2} is used as .10.)
	(b) $\frac{RPS_1 - RPS_k}{RPS_1}$ = fractional improvement of RPS.

<u>Score Group 6</u>

This score group is for the frequencies of <u>differences</u> between forecasts and observations. Each forecast system has m ordered discrete categories specified by thresholds in the IDs; the matching (single) observation is continuous. That is, the forecast IDs specify the categories which are applied after the differences are computed.

Score No.	Explanation
61	Frequencies and relative frequencies of algebraic errors.
62	Frequencies and relative frequencies of absolute errors.

Score Group 7

This score group is to provide one or more measures of the temporal consistency of one or more sets of forecasts all valid at the same time but made at different times (with, of course, different projections) (e.g., daytime max temperature forecasts valid on Dec. 10, 2004 made on 5 previous days). By definition, there is only one "projection," and for each system the projection printed will be the maximum tau for that system.

For each system, the first variable ID in Record Type 16 will have the shortest tau, and each succeeding ID for that same system will have a larger tau. Each record will have an ITAU which together with the tau will point to the score ITAU of the single verifying observation (Record Type 17). The number of forecasts per system will be determined when the taus cease to increase, or the 888888 terminator is reached if there is a single system. Then all the rest of the IDs in Record Type 16 must be in groups of that same size. For instance, if the first system (e.g., MOS) has 10 IDs, then the remaining IDs (before the 88888) in Record Type 16 must be evenly divisible by 10.

There are three scores in Score Group 7. The first two are similar and measure the convergence of the forecasts toward the observation. For the first score, a sequence of forecasts, all valid at the same time, is used with the formula:

$$\mathrm{FCS} = \frac{A + \frac{B}{T}}{N - 1 + \frac{C}{T}}$$

where:

A = A threshold "T" is used, and "A" is the number of times a forecast was changed "insignificantly" (defined to be $\leq$ T units) OR the changed forecast "moved closer" to the next forecast (or the observation for the last forecast) than the current one was. (Consider three forecasts, the current one F1, the changed one F2, and the next one F3. Then for F2 to "move closer" to F3,*F3-F2* < *F3-F1*. When F2 is the last forecast issued, the observation is substituted for F1.) The maximum value of A is N - 1.
B = The absolute value of the difference between first and last forecast.
T = The threshold value.
N = The number of forecasts used in computing F1 (the number of differences is N - 1).
C = The sum of the absolute values of the differences between the adjoining forecasts in the sequence. The observation is not used in this calculation.

Note that the observation is used only in determining whether or not the last forecast moved closer to the observation. Three possible values of T can be used. These are specific to the variable, and are conjoined to the MOS ID CCCFFF in subroutine SETCTH. If a new variable need be added, or the thresholds are to be changed, the change need be made only in one place--subroutine SETCTH.

The second score is like the first, except The observation is not used in the sequence in computing A. Therefore, the maximum value of A is one less and the denominator includes N - 2 instead of N - 1.

The Consistency Score is much simpler and is:

$$CS = \frac{D}{E}$$

where:

D = the absolute difference between the longest range forecast and the verifying observation, and

E = the sum of the absolute values of the differences between the adjoining forecasts in the sequence, including the difference between the last forecast and the verifying observation.

There is only output package, 71, and it is much like package 02 for Score Group 1. The Scores are defined:

1 = FCS Score 1 with the first threshold T,
2 = FCS Score 1 with the second threshold T,
3 = FCS Score 1 with the third threshold T,
4 = FCS Score 2 with the first threshold T,
5 = FCS Score 2 with the second threshold T,
6 = FCS Score 2 with the third threshold T,
7 = CS

Finally, the fractional improvement of each system over the first can be computed with:

8 = The fractional improvement for all other scores designated of each system over the first (e.g., if score 3 is not printed, neither will be its improvement).

APPENDIX III

EXAMPLE CONTROL FILE

An example .CN control file is shown on the following page. Note that a brief explanation of the record (line) can be given, provided it is beyond where the data will be read. The first several values can be interpreted by referring to the documentation above.

The date list is on the default input file, and dates for January 1, 1998, 0000 UTC through January 2, 1200 UTC at 12-hr intervals (INCYCL = 12) will be used.

The input data will be on two files with Unit Nos. 22 and 26. Constant data, if needed, will be on file 'MOSCONSTANT4'. The output (once it is implemented) would be on Unit No. 27.

The station list is on the default input file, and the directory is on Unit No. 28. The stations are in two groups with group names "EASTERN US" and "WESTERN US". The first group contains 2 stations, and the second one, three. Each station group ends with the terminator 99999999; an empty set signals the end of the station list(s).

The variable list is on the default input file, and the plain language IDs are on Unit No. 29.

One score group, Score Group No. 1, is to be used on this run, with scores 2, 3, and 4 being printed by output package 3.

Four forecasts for each of two projections are being verified, and each projection requires one verifying observation. No matching data are needed. Note that both 888888 and 999999 terminators are present for the matching data sets; the 888888 could have been omitted. The IDs indicate that all forecasts being verified are NGM MOS; those in the first "projection" are for projections 6, 12, 18, and 24 hours, and those in the second "projection" are for projections 12, 18, 24, and 30 hours. The verifying observation for the first projection is 12 hours ahead of the date being processed and for the second projection is 24 hours ahead. So, this is verifying NGM MOS forecasts valid at 6, 12, 18, and 24 hours after the cycle time with the 12-observation. Also, the NGM MOS forecasts valid at 12, 18, 24, and 30 hours after cycle time are verified with the 24-hour observation. In this way, one can determine whether the forecast valid at the observation time is really better than the forecast valid before or after that time.

The final 999999 indicates that no more score groups are present.

```
HRG1  0 12 12 12 12 12 12 12 12  0 12  0 12 12 12 12 12 12 12 12 12 12 12 12 12
THIS IS A MAY 1998 TEST OF U850
         0  XSKIP --SKIP PAST THIS DATE ON OUTPUT
        19  NSKIP --NUMBER OF ERRORS TO TOLERATE ON DAY 1 BEFORE STOPPING
       200  JSTOP --NUMBER OF ERRORS TO TOLERATE BEFORE STOPPING RUN
         1  NSTIND--1 FOR SCORES FOR INDIVIDUAL STATIONS, 0 = NOT
        12  INCCYL--HOURS BETWEEN CASES
         1  NEW   --1 FOR NEW ICAO CALL LETTERS, 0 = NOT
         1  NALPH --1 FOR ALPHABETIZATION OF CALL LETTERS, 0 = NOT
         0  PXMISS--VALUE TO INSERT FOR 9997
         2  NPRINT--NUMBER OF CYCLES OF DATA TO PRINT UNDER JP(2, ) AND JP(3, ) CONTROL
         4  CHARST--NUMBER OF CHARACTERS ALLOWED FOR CALL LETTERS WHEN PRINTING DATA
       200  LNGTH --LINE LENGTH FOR PRINTING TO IP(16)
         0  NROUND--ROUNDING PARAMETER
 05     DEFAULT                                                         **DATE LIST ON DEFAULT INPUT.
 99
  98010100
 -98010212
  99999999
 22     /mos5/mos/mosarc/199801                                         **INPUT
 26     /mos5/mos/hourly/199801                                         **INPUT
 99
 95     MOSCONSTANT4                                                    **CONSTANT DATA
 99
 27     /home21/glahn/mos2000data/U850OUTHRGmar14                       **OUTPUT
 99
 05     DEFAULT                                                         **STATION LIST ON DEFAULT INPUT.
 28     /home21/tdllib/table/station.tbl                                **STATION DIRECTORY.
 99
DCA
BOS       99999999
EASTERN US
TUL       BRO
SEA       99999999
WESTERN US
99999999
 05     DEFAULT                                                         **VARIABLE LIST ON DEFAULT INPUT.
 99
 29     /home21/glahn/mos2000data/MOS2000PRED                           **PREDICTOR CONSTANTS.
 99
SGROUP    1
OUTPUT   03
SCORES    2   3   4
999999
202000006 000000000 000000006 000   .0000E+00     1 1 0  0       F10.1 NGM
202000006 000000000 000000012 000   .0000E+00     1 1 0  0       F10.1 NGM
202000006 000000000 000000018 000   .0000E+00     1 1 0  0       F10.1 NGM
202000006 000000000 000000024 000   .0000E+00     1 1 0  0       F10.1 NGM
   888888
202000006 000000000 000000012 000   .0000E+00     1 1 0  0       F10.1 NGM
202000006 000000000 000000018 000   .0000E+00     1 1 0  0       F10.1 NGM
202000006 000000000 000000024 000   .0000E+00     1 1 0  0       F10.1 NGM
202000006 000000000 000000030 000   .0000E+00     1 1 0  0       F10.1 NGM
   888888
   999999
702000000 000000000 000000000 000 +.0000E+00     1 1 0 12       F10.1 TMP OB+12
   888888
702000000 000000000 000000000 000 +.0000E+00     1 1 0 24       F10.1 TMP OB+24
   888888
   999999
   888888
   999999
999999
```

APPENDIX IV

PROGRAM STRUCTURE

Many of the main programs in the MOS-2000 system have similar structure. However, subtle differences in the access and use of data make some subroutines unique, although very similar to others. For U850, separate subroutine writeups are provided only for those routines that may have a use outside U850. Rather, the structure is shown below with a brief explanation of the routines.

```
DRU850 - Driver.
   U850 - Main program.
      INT850 - Initialization.
         IPOPEN - Opens the IP( ) files as necessary.
         TIMPR  - Labels start of run on all IP( ) files.
         RDSNAM - Called several times.  Reads unit number and file name for
                  input/output, and opens file according to call sequence.
         RDI    - Reads date list.
         DATPRO - Prepares full date list by accommodating date spanning.
            UPDAT - Adds hours to date/time to get a new date/time.
         RDSTNL - Reads station list when alphabetization is not desired.
            RDC   - Reads the station list as characters.
         RDSTNA - Reads station list when alphabetization is desired.
            RDC   - Reads the station list as characters.
         RDVR85 - Reads the definition of scores, output packages, and varia-
                  bles to be used.
            RDSCR - Reads score group numbers, output package numbers, and
                    score numbers to output.
            RD85  - Reads the variable IDs.
               PRSID  - Forms the 4-word ID( ), its parsed parts in IDPARS( ),
                        and thresholds from the "raw" input.
               THSET  - Sets upper and lower thresholds from the sequence of
                        thresholds read with the variables, according to the
                        value of B = IDPARS(3).
               CALCAT - Calculates the number of categories in a set of IDs.
                        Checks for same number of categories for each system.
               CNTCAT - Same as CALCAT, except for Score Group 5 which does not
                        have thresholds, but rather a category number in ID-
                        PARS(6).  Checks for same number of categories for each
                        system.
         IERX   - Error routine used only if there is an I/O type error.  Will
                  define the statement number where the error occurred.
      RDSTRX - Reads vector data from all sources for the needed date/times for
               the first case, unpacks the data for these date/times, orders
               the data according to the input station list, and stores them
               into the MOS-2000 internal storage system.
         RDDIR  - Reads the call letters record from the vector input and
                  prepares indices for the vector input records to follow for
                  the station list.
            FINDST - Provides the indices for the station list.
         UNPKBG - Unpacks a string of bits into an integer word.
         UPDAT  - Adds hours to date/time to get a new date/time.
         UNPACK - The general MOS-2000 unpacking routine.  All data stored in
                  the MOS-2000 storage system have to be unpacked and matched
                  with the station list, because the station list may change on
                  a particular input.
            UNPKBG - Unpacks a string of bits into an integer word.
            UNPKLX - Unpacks a portion of the packed data.
            PNPKOO - Unpacks data when there are no missing values.
            UNPKPO -Unpacks data when there can be only primary missing
                    values.
            UNPKPS -Unpacks data when where can be both primary and secondary
                    missing values.
         GSTORE - Stores the data into the MOS-2000 internal storage system.
            WRDISK -Writes disk records as necessary.
      SETWHR - Sets indices to indicate where each variable is to come from.
               In U850, this is limited to 1, meaning directly from input, and
               3, meaning from OPTX.
         UPDAT  -Adds hours to date/time to get a new date/time.
      SCOPTA - Switcher to accumulant routines.  Also reads and stores the
               accumulants in the MOS-2000 internal storage system.
         GFETCH -Fetches accumulants from the MOS-2000 internal storage
                  system.  (This will be unsuccessful for the 1st case.)
            UPDAT  - Adds hours to date/time to get a new date/time.
            RDDISK - Reads the disk records in the MOS-2000 internal storage
                     when necessary.
            UNPACK - Should not be called here, because data are already stored
                     unpacked.
         FILLAA -Fills the AA matrix with all needed data for the score group
                  being processed.
            GTVECT - Gets a vector data record.
               UPDAT  -Adds hours to date/time to get a new date/time.
               GFETCH -Fetches data from the MOS-2000 internal storage system.
                       (See above for calls from GFETCH.)
               OPTX   -Calls computation routines.
                  CONSTX - Retrieves constant data from the external MOS-2000
                        storage system.
                  TIMEV  - Computes a time difference.  This is used in U850
                        when matching variables are in terms of time changes.
                  DIFFV  - Computes the difference of two variables for U850,
                        when the differences are to be put into categories.
                        This is for Score Group 6.
            BINFUL -Makes categorical binaries.
            BINARY -Makes cumulative binaries.
         SCR01A - Computes accumulants for the case being processed for Score
                  Group 1.
            FIXIT -Rounds as necessary.
         SCR21A - Computes accumulants for the case being processed for Score
                  Group 2.
         SCR31A - Computes accumulants for the case being processed for Score
                  Group 3.
            FIXIT -Rounds as necessary.
            BKCAT -Puts variable into discrete categories and assigns a
                    category number to each category.
         SCR41A - Computes accumulants for the case being processed for Score
                  Group 4.
            FIXIT -Rounds as necessary.
            BKCAT -Puts variable into discrete categories and assigns a
                    category number to each category.
         SCR51A - Computes accumulants for the case being processed for Score
                  Group 5.
            FIXIT -Rounds as necessary.
            BKCAT -Puts variable into discrete categories and assigns a
                    category number to each category.
         SCR61A - Computes accumulants for the case being processed for Score
                  Group 6.
         SCR71A - Computes accumulants for the case being processed for Score
                  Group 7.
            FIXIT -Rounds as necessary.
         GSTORE - Stores the accumulants into the MOS-2000 internal storage
                  system.  This should happen only once per score group for the
                  first case; the rest of the time replacement occurs (see
                  RSTORE).
         RSTORE - Replaces the accumulants into the MOS-2000 internal storage
                  system.
      LMSTR6 - Prepares list of variables needed for subsequent dates.
         UPDAT  - Adds hours to date/time to get a new date/time.
      GCPAC  - Compresses entries to be kept in the MOS-2000 internal storage
               system.
         RDDISK - Reads the disk records from the MOS-2000 internal storage
                  system when necessary.
         WRDISK - Writes disk records into the MOS-2000 internal storage system
                  as necessary.
      RDSTR8 - To read and store all variables needed for this case in the MOS-
               2000 internal storage system.  RDSTR8, SCOPTA, LMSTR2, and
               GCPAC below are repeated for each case.
         UPDAT  - Adds hours to date/time to get a new date/time.
         RDVECT - Reads a vector data record and stores in into the internal
                  MOS-2000 storage system.  Note that for U850, all data have
                  to be stored, which is different that most MOS-2000 programs,
                  because the data are not all needed at the same time, but are
                  accessed for different score groups.
            UPDAT  -Adds hours to date/time to get a new date/time.
            UNPKBG -Unpacks a string of bits into an integer word.
            RDDIR  -Reads the call letters record from the vector input and
                    prepares indices for the vector input records to follow
                    for the station list.
               FINDST -Provides the indices for the station list.
            UNPACK - The general MOS-2000 unpacking routine.  (See above for
                     calls from UNPACK.)
            GSTORE -Stores the data into the MOS-2000 internal storage system.
      SCOPTA - See above.  This entry is used for all cases after the first.
      LMSTR2 - Zeros out all entries in the MOS-2000 internal storage system
               that will not be needed on the next case.
      GCPAC  - Compresses entries to be kept in the MOS-2000 internal storage
               system.
      SCOPTB - Switcher for computing scores from accumulants.
         GFETCH - Fetches accumulants from the MOS-2000 internal storage
                  system.  (See above for calls from GFETCH.)
         SCR01B - Computes scores for the case being processed for Score
                  Group 1.
         SCR21B - Computes scores for the case being processed for Score
                  Group 2.
         SCR31B - Computes scores for the case being processed for Score
                  Group 3.
         SCR41B - Computes scores for the case being processed for Score
                  Group 4.
         SCR51B - Computes scores for the case being processed for Score
                  Group 5.
         SCR61B - Computes scores for the case being processed for Score
                  Group 6.
         SCR71B - Computes scores for the case being processed for Score
                  Group 7.
         GSTORE - Stores the scores into the MOS-2000 internal storage system.
      SCOPTC - Switcher for printing scores.  Sometimes, the accumulants are
               also printed.
         GFETCH - Fetches accumulants from the MOS-2000 internal storage
                  system.  (See above for calls from GFETCH.)
         GFETCH - Fetches scores from the MOS-2000 internal storage system.
                   (See above for calls from GFETCH.)
         SCR01C - Prints scores for output package 01.
            SCR02C - Prints scores for output package 02.  This is called from
                     SCR01C when a line cannot hold all scores.
         SCR02C - Prints scores for output package 02.
            SCR01X - Assists in forming formats for printing.
            SCR02X - Assists in forming formats for printing.
         SCR03C - Prints scores for output package 03.
         SCR04C - Prints scores for output package 04.
         SCR21C - Prints scores for output package 21.
            SCR25C - Assists in printing.
         SCR22C - Prints scores for output package 22.
            SCR25C - Assists in printing.
         SCR31C - Prints scores for output package 31.
         SCR32C - Prints scores for output package 32.
         SCR41C - Prints scores for output package 41.
         SCR42C - Prints scores for output package 42.
         SCR51C - Prints scores for output package 51.
         SCR61C - Prints scores for output package 61.
         SCR62C - Prints scores for output package 62.
         SCR71C - Prints scores for output package 71.
      SCOPTD - Switcher for writing scores to IP(17).
         GFETCH - Fetches accumulants from the MOS-2000 internal storage
                  system.  (See above for calls from GFETCH.)
         GFETCH - Fetches scores from the MOS-2000 internal storage system.
                  (See above for calls from GFETCH.)
         SCR01D - Writes scores for output package 01.
            WXELEM - Determines a portion of the score ID for writing.
            STRATID - Determines whether there are stratification variables to
                      modify score number.
            ROUNDER - Rounds scores to 2 decimal places.
         SCR21D - Writes scores for output package 02.
            WXELEM - Determines a portion of the score ID for writing.
            STRATID - Determines whether there are stratification variables to
                      modify score number.
            NUMVNTL - Computes denominator for scores from contingency table.
            NUMVNTU - Computes denominator for scores from contingency table.
         SCR31D - Prints scores for output package 03.
            WXELEM - Determines a portion of the score ID for writing.
            STRATID - Determines whether there are stratification variables to
                      modify score number.
         SCR41D - Prints scores for output package 04.
            WXELEM - Determines a portion of the score ID for writing.
            STRATID - Determines whether there are stratification variables to
                      modify score number.
            ROUNDER - Rounds P-scores to 4 decimal places and other scores to 2
                      decimal places.
         SCR61D - Prints scores for output package 61.
            WXELEM - Determines a portion of the score ID for writing.
            STRATID - Determines whether there are stratification variables to
                      modify score number.
         SCR71D - Prints scores for output package 71.
            WXELEM - Determines a portion of the score ID for writing.
            STRATID - Determines whether there are stratification variables to
                      modify score number.
            ROUNDER - Rounds scores to 3 decimal places.
```

APPENDIX V

DEFINITION OF FORMAT AND SCORE IDS WRITTEN TO IP(17)

To be added later.

U830

CALCULATES THRESHOLDS FOR MAKING CATEGORICAL FORECASTS

David E. Rudack
Harry R. Glahn
J. Paul Dallavalle
January 1, 2000

PURPOSE: U830 calculates thresholds used for determining categorical forecasts from probabilities. Various options based on a specified bias or maximizing the threat score within specified bias limits can be used. "Continuous" probabilities from above or from below and "discrete" probabilities are accommodated; see the appendix for a definition of the algorithms. These thresholds can be calculated for individual stations or groups of stations. Due to practical constraints imposed in the interest of code simplicity and relative ease of use, thresholds from only one "basic" MOS-variable can be computed on a single run. However, thresholds for different categories of the same variable can be computed in one run. U830 is derived from U660 and can perform many of the U660 functions. Input data must be in TDLPACK format. The first record in an input data-set is the "station" (or location) directory (usually) containing station call letters. U830 uses a driver DRU830 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U830 (actually, subroutine) for every application. U830 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U830.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU830.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U830" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U830 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U830HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U830 except possibly

KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). Values for J have been defined as indicated below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as non-zero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU830 DATA statement = 12. Changing the default unit number allows multiple runs of U830 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.
(7) = The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) = The variable ID list in summary form. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)
(9) = The variable list in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).
(10) = The variable ID's for the first day (day 1) as read from the archive tapes. This is just a list of all the variables on the input files.
(11) = The variable ID's of the archived data actually needed, in order as they appear on the archive files for day 1.
(12) = The list(s) of stations on the input file(s).

(13) = Not used.
(14) = A diagnostic will be provided when there are no data for a particular input file. With tape switching, this may not be of much use, and could be misleading.
(15) = Data written in the order packed for each variable indicated by JP(3,) > 0. This is separate from the optional writing associated with JP(2,). Except for a very few days, this would produce voluminous files.
(16) = All or a portion of the data values in the AA(,) matrix. For each case (cycle), after all input data have been read, the AA(,) matrix holds all variable values for all stations. The <u>print is by variable</u> controlled by JP(2,) (see Record Type 12 below), then the station values are printed in the order dealt with in U830 (see IP(5)). Except for a very few days or a few variables and stations, this would produce voluminous files.
(23) = Information concerning opening and closing of files.
(24) = Information as to where the variables are to be found--directly on input, binary computed from a previous variable, or (at least attempted to be) computed through subroutine OPTX.

For checkout, it may be advisable to set all these values to the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) <u>Run Identification</u>

This record is used to identify the run.

<u>RUNID</u> - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (7(I10/),F10.0/(I10)) <u>Control Parameters</u>

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

<u>NSKIP</u> - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

<u>JSTOP</u> - The total number of errors that will be tolerated before the program halts (see Comments section).

<u>INCCYL</u> - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. Because of the lookahead feature required for predictands, and to maintain efficiency, date/times should not be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996,

and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship.

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

NSCORE - 1 = Thresholds are determined based on attaining a specified bias.
2 = Thresholds are determined based on attaining the maximum threat score, within specified bias limits.

IDISCT - 0 = Probabilities are cumulative from above or below.
1 = Probabilities are discrete and the threshold is determined by coming from above.
2 = Probabilities are discrete and the threshold is determined by coming from below.

PXMISS - The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value.

NPRINT - The number of cycles of data to write for printing to unit IP(16) under the format control and JP(2,) provided with each variable (see Record Type 13).

ICHARS - The number of characters of call letters to print when printing is indicated by JP(2,). This is constrained to be between 4 and 8 inclusive.

LNGTH - The line length for printing to unit IP(16). For a line printer, 132 is appropriate; for a laser printer, 80 may be desirable.

Record Type 4 - Format (I3,4XA60) Date List File

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) Date List

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U830 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be > KSKIP (see Record Type 3).

Record Type 6 - Format (I3,4XA60) Input Vector Data Files

This group of records identifies the data sets from which the vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN. If all input data are from a constant file, only the terminator is necessary.

KFILIN(J) - Unit number for the data file.

NAMIN(J) - Name of file where these data reside. (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. (A complication is the necessity for the lookahead feature for predictands.) When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4X,A60) ASCII Output File

This record (plus the terminator record) identifies the data set which will contain a tabular printout of output constants, with stations grouped as specified by the station list (see Record Type 10) and with or without intra-group alphabetization as specified by the parameter NALPH in Record Type 2. Records are read until the terminator KFILAS = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record

Type 7 is read by subroutine RDSNAM and will be opened as 'NEW'. This output file is optional; if tabular ASCII output is not desired, only the terminator need be present.

KFILAS - Unit number for the ASCII output file.

ASCIFL - Name of file for tabular ASCII output of constants. (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) Vector Output File (Not yet implemented)

This record (plus the terminator record) identifies the packed threshold output file. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If thresholds are not to be saved, only the terminator is necessary here; in that case, a file is not opened and data are not written.

KFILIO - Unit number for the vector output file.

OUTNAM - Name of file where this output is to reside. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) Station and Location Files

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (14(A8,1X)) Station List

This group of records identifies the (groups of) stations for which data are to be processed. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read within subroutine RDSTAD by RDC, which eliminates any blanks found in the input. Duplicate stations in the list for a group are kept, but a diagnostic is furnished on unit IP(5). Note that this diagnostic applies only to duplicates within a group, not from group to group. For NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty set terminates the station input. That is, the last group and its terminator must be followed by another terminator signifying an empty set. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) Variable File

This record (plus the terminator record) identifies the file from which the variable ID's are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) Variable Constants File

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. This file is used for plain language description. Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 13 - Format Variable List
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,I3,8XA1,I2,1XI1,1X30A1)

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 13 is read by subroutine RDVR66. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201.

ID(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

JP(J,N) - JP(J,N) indicates for J=1,3:
1 = Whether (>0) or not (=0) variable N will be packed and written to unit KFILIO.
2 = Whether (>0) or not (=0) variable N will be written to unit IP(16) with the format provided. This is the primary output for viewing or printing.
3 = Whether (>0) or not (=0) variable N will be written to unit IP(15) to the resolution packed. This is primarily for checkout and quality control of data being written.

ITAU(N) - The number of hours to add to NDATE (the date being processed, see Record Type 5) to get the variable N. That is, the tau in the ID is left intact, and ITAU is used to "look ahead."

CFMT(N) - The format descriptor for writing on unit IP(16) when JP(2,N) > 0. Must be either "F" or "I".

IWDTH(N)- The field width for writing on unit IP(16) when JP(2,N) > 0. Must be $\leq$ 30.

IPREC(N)- The precision for writing on unit IP(16) when IP(2,N) > 0. For an "F" format, this is the number of digits after the decimal point. For an "I" format, it is the number of digits always written. Note that for a "zero" to be printed, IPREC( ) must be > 0; otherwise, zero will be printed as a blank.

HEAD(J,N) - The column heading for writing to Unit IP(16) when JP(2,N) > 0. Limited to J=1,30 characters. Only IWDTH(N) characters are read. HEAD(,) is right justified when writing.

For a U830 run, there will be one "observational" ID, and this must be the first in the list. Then, there must be as many probability forecast IDs as necessary for computing the thresholds, and the thresholds with those probability IDs must be in order (see the appendix for the computational algorithms).

Record Type 14 - Format (5F10.0,I9) Threshold Information

Each record in this group contains in order (1) the initial (or guess) threshold, (2) the lower bound of acceptable bias, (3) the upper bound of acceptable bias, and (4) the delta by which to (initially) modify the initial threshold to calculate a better one. There is one record here for each probability forecast for each group of stations, the first grouping being by station group. That is, there are five values for each group for the first variable, then this repeats for each variable. A terminator '^^^^999999' must be present.

<u>BKI</u>(L,N) - The initial breakpoint (L=1,KGP) (N=1,NVRBL-1). If the user does not wish to give a value, a 9999 may be used, in which case the program will compute one from the probabilities.

<u>BILO</u>(L,N) - The lower bound of acceptable bias (L=1,KGP) (N=1,NVRBL-1).

<u>BIHO</u>(L,N) - The upper bound of acceptable bias (L=1,KGP) (N=1,NVRBL-1).

<u>DELTA</u>(L,N)- The initial amount by which BKI(L,N) is changed in the computational algorithm (L=1,KGP) (N=1,NVRBL-1).

<u>EDELTA</u>(L,N)- The probability increment by which thresholds are calculated for printout of contingency tables and associated statistics (see Record Type 6) (L=1,KGP) (N=1,NVRBL-1).

<u>NEVENTS</u>(L,N)- The minimum number of matched observed-forecast events for which U830 will generate breakpoints for each group-category. (L=1,KGP) (N=1,NVRBL-1). Note: When the actual number of matched observed-forecast events is less than the desired number of events for a particular group-category the threshold value is set to .999 for that group-category.

Record Type 15 - Format (2(I9,1X)) <u>Output Variable ID</u>

This record contains the first two of the output variable ID's. ID(3) and ID(4) are taken from the IDs of the input probabilities. No terminator is needed.

<u>ID</u>(1) - CCCFFFBDD = 8xyyyyBDD, where x = 0 if the forecast probabilities are for cumulative categories; x = 1 if the forecast probabilities are for discrete categories and the user wants to calculate thresholds cumulatively from above; x = 2 if the forecast probabilities are for discrete categories and the user wants to calculate thresholds cumulatively from below. yyyy is the ID assigned to the threshold to be calculated.

<u>ID</u>(2) - VLLLLUUUU = 0LLMM0000, where LL = the cycle for which the thresholds are valid (e.g., 00 = 00Z, 06 = 06Z, etc.), and MM = coded value indicating the season for which the thresholds are valid (1 = January 1- January 31; 12 = December 1-December 31; 13 = March 1-May 31; 14 = June 1-August 31; 15 = September 1-November 30; 16 = December 1-February 29; 17 = April 1-September 30; 18 = October 1- March 31; 19 = January 1- December 31, etc.)

<u>CONTROL FILE INPUT</u>: (Name read from U830.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) <u>Date List</u>

<u>When the dates are not provided in file U830.CN</u>, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU830), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list <u>must</u> be available or U830 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

<u>CONTROL FILE INPUT</u>: (Name read from U830.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) <u>Station List</u>

<u>When the station list is not provided in file U830.CN</u>, this group of records identifies the stations (or locations) to be included in the regression analysis. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

<u>CCALL</u>(K) - Call letters (or other 8-character location designator) of stations (or locations) for which interpolated values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 9, control file 'U830.CN', unit KFILDI for additional information.) (CHARACTER*8)

<u>CONTROL FILE INPUT</u>: (Name read from U830.CN) (Unit = KFILD(2))

Record Type 1 - Format <u>Station Locations</u>
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical <u>within each group</u> no matter the order read in (see NALPH in Record Type 3). [Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. Note that alphabetization is not overall, but by group (see Record Type 10)]. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3).

CCALLD(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

NAME(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

NELEV(K) - Elevation of the station. Format is I5.

SIGNLA - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

XLAT - Latitude in degrees. Format is F7.4.

SIGNLO - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

LONDD - Longitude in degrees west. Format is F8.4.

CCALLD(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

IWBAN(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited. No terminator is used.

CONTROL FILE INPUT: (Name read from U830.CN) (Unit = KFILP)

Record Type 1 - Format Variable List
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,I3,8XA1,I2,1XI1,1X30A1)

When the variables to use in the run are not in file U830.CN, this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDVR66. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201. See Control File Input for U830.CN, Record Type 13, unit KFILDI above for more detailed information; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U830.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U830 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the data for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer.

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDVR66, much of it associated with the plain language description. See Chapter 4, Variable Identification in TDL Office Note MOS-2000, for details.

CONTROL FILE INPUT: (Name read from U830.CN) (Unit = KFILIO)

Record Type 1 - Format (5F10.0,I9) Threshold Information

When the variables to use in the run are not in file U830.CN, each record in this group contains in order (1) the initial (or guess) threshold, (2) the lower bound of acceptable bias, (3) the upper bound of acceptable bias, and (4) the delta by which to (initially) modify the initial threshold to calculate a better one. There is one record here for each probability forecast for each group of stations, the first grouping being by station group. That is, there are five values for each group for the first variable, then this repeats for each variable. A terminator '^^^^999999' must be present.

BKI(L,N) - The initial breakpoint (L=1,KGP) (N=1,NVRBL-1)

BILO(L,N) - The lower bound of acceptable bias (L=1,KGP) (N=1,NVRBL-1)

BIHO(L,N) - The upper bound of acceptable bias (L=1,KGP) (N=1,NVRBL-1)

DELTA(L,N)- The initial amount by which BKI(L,N) is changed in the computational algorithm (L=1,KGP) (N=1,NVRBL-1)

EDELTA(L,N)- The probability increment by which thresholds are calculated for printout of contingency tables and associated statistics (see Record Type 6) (L=1,KGP) (N=1,NVRBL-1).

NEVENTS(L,N)- The minimum number of matched observed-forecast events for which U830 will generate breakpoints for each group-category. (L=1,KGP) (N=1,NVRBL-1). Note: When the actual number of matched observed-forecast events is less than the desired number of events for a particular group-category the threshold value is set to .999 for that group-category.

DATA INPUT:

All data input to U830 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000). U830 does not access or need data from the random access MOS-2000 External File System. All such vector data must be preceded by a call letters (directory) record. The sequential files can have multiple directory records, each applying to the data until an end of file or another directory is encountered. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U830.CN in Record Type 6.

A. SEQUENTIAL VECTOR DATA

One or more sources of vector data, each probably prepared by U700 or U201, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U830.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

DATA OUTPUT

There are three forms of data output, besides the diagnostics and other information on units KFILDO and IP( ).

A. ASCII FOR VIEWING OR PRINTING INPUT DATA

A maximum of NPRINT (see Record Type 3) or NDATES (see Record Type 5) cycles of data will be written to the file on unit IP(16) under control of the format provided with each variable N, the variables written controlled by JP(2,N) (see Record Type 13). The data columns are headed by HEAD(,N) (see Record Type 13). Listing is by station group. If a line length for printing is not sufficient for all variables, then more than one line is used. Line length is specified as LNGTH characters in Record Type 3.

B. ASCII FOR STATIONS, THRESHOLD IDS, AND THRESHOLDS

When KFILAS ≠ 0, information for input to U351 is written to Unit No. KFILAS (see Record Type 7). This consists of:

(a) Station list - Format (7(A8,1X))

This is the same list of stations read in Record Type 10, CALL(K), K=1,NSTA, followed by a terminator '99999999'.

(b) Four threshold IDs, ISCALE, ICHAR, character definition Format - (I9,3I10.9,2I4,A32)

See Record Type 15 for a description of the IDs. ISCALE = 3 for the thresholds, ICHAR = 32, and character definition is either:
'CUMULATIVE FROM ABOVE THRESHOLDS'
'CUMULATIVE FROM BELOW THRESHOLDS'
'DISCRETE FROM ABOVE THRESHOLDS'
'DISCRETE FROM BELOW THRESHOLDS'

(C) Thresholds - Format (7F9.3)

Thresholds for each station for one probability category, followed by a terminator '999999'.

(b) and (c) then repeat for all IDs and thresholds, followed by a final terminator '999999' (two terminators at the end).

C. BINARY MOS-2000 FORMAT (Not yet implemented)

Each of the thresholds calculated will be packed in TDLPACK format and written to unit number KFILIO to the dataset whose name has been provided to OUTNAM (see Record Type 8), unless KFILIO = 0, in which case data are not output. Thresholds are calculated by station group, but a value is written for each station used in the run; each station in a group is given the group value. The data are packed with subroutine PACK1D and its associated subroutines.

EXAMPLE CONTROL FILE: 'U830.CN'

An example exists as file 'U830.CN' in directory home21.tdllib.dru830 on blizzard. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U830.CN', Record Type 1 and the definition of KFILDO in the driver DRU830. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The CRAY does not allow this. It is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

The binary indicator "B" in the first word of an ID can be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), 5 (indicating a grid binary), 2 (indicating a cumulative variable from below), or 3 (indicating a discrete variable). For the latter, the upper threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID's. If there is no lower one, the lower threshold is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999.

Since the variables are not ordered (as they are in U600), the probability forecasts must be input in sequence and in the correct order (lower threshold first). To make sure all is well, check the thresholds on unit IP(8) or IP(9).

It is not necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles, unless the variable is computed from other variables not otherwise used as a "raw" variable. That is, a raw (not computed) variable need not be present for Day 1, but one used only in computations must be, because U830 has no way of knowing it will be needed for future cycles.

Because of an INQUIRE statement in subroutine RDSNAM (that is needed in operations), do not use Unit Numbers 5, 6, or 7, except 5 is the default input file.

COMMENTS

The algorithms for computing the thresholds are explained in the appendix.

Each source of vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U830 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 3). Presumably, the only time 9997 will occur is when operational forecasts are input or U700 produces test forecasts; it is expected that U830 will be using forecasts produced by U700, so 9997 must be dealt with. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. This probably means a set of dates should be supplied to U830 for which it is known there are good data. Then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

On Day 1, GFETCH is entered directly for every variable (except a point binary) because it is not yet known when OPTX must be entered. ("Day 1 is a term that has come to be used for the "first case." It is actually the first cycle of the first day.) A return of IER = 47 (which means data were not found in GFETCH) <u>is not</u> counted as an error in VRBL83 for Day 1. It <u>is</u> counted as an error in VRBL84 (through OPTX), because GFETCH should not be entered directly when OPTX is needed.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage after Day 1.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 10 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

Although the primary purpose of U830 is to provide packed thresholds, writing of such data is not done if KFILIO = 0 (see Record Type 8). That is, no output unit number and file name have been provided. This feature can be used for checkout.

SETTING UP THE DRIVER DRU830

The preparation of the driver for a particular U830 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B - Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).

ND1 - Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the directory (read on unit KFILD(2)) unless, of course, the station directory is to be used as the station list (see ND5).

ND2 - Not used.

ND3 - Not used.

ND4 - The maximum number of variables that can be dealt with.

ND5 - Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.

ND6 - Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc., even though the same unit number might be used.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU830 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

WRITING NEW SUBROUTINES

It is not expected that computational routines will be used to any great extent in U830. Any computations that would be done in U830 would have to be done on point data (not gridpoint data) because gridpoint data are not accommodated in U830. Even so, an "option" subroutine, OPTX, is provided for possible use. (In the case of development for gridpoints, the gridpoints are given identifications in the same way as other locations, such as stations.)

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib.dru830, dataset 'u830.com', which includes the libraries 'home21.tdllib.u830lib' and 'home21.tdllib.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: /mdl1/save/mos/mos2k/u830lib. The driver, dru830.f, can be found in /mdl1/save/mos/mos2k/u830lib/RUN.

APPENDIX

ALGORITHMS FOR DETERMINING THRESHOLDS

BFABVE:

Subroutine BFABVE is designed to calculate threshold probabilities for probability forecasts in such a way that a user-specified categorical bias is obtained. The categorical probability values are either cumulative from above or discrete and the user wishes to obtain the categorical thresholds by starting at the upper end of the categorical breakpoints. An example of the former case may include the probability of precipitation amount (≥ .01 inches, ≥ .10 inches, etc.) while the latter may involve predicting the obstruction to vision. By defining the occurrence of obstruction of vision from the rarest (e.g. blowing snow) event to the most common event (no obstruction), one virtually ensures that the lower threshold values are assigned to the rarest event. As a consequence of the subroutines later used to compute the categorical forecasts, the most common event becomes the default value when none of the thresholds are exceeded. The total number of thresholds calculated by BFABVE is dependent upon the type of forecast probabilities. For example, if the user is dealing with cumulative from above probabilities and defines five breakpoint values, the output consists of five threshold values. However, in the instance where discrete probabilities are used and the user is starting at the upper end of categorical breakpoints, the output consists of only *four* threshold values, though five breakpoint values are required for user input. Here, the first (or lowest) probability breakpoint category will not have any corresponding threshold value. In either case, the number of 2x2 contingency tables generated in the output corresponds to the total number of output threshold values.

The algorithm can be divided into the following steps. First, the program either ingests an initial threshold guess as a starting point for the algorithm or calculates a first guess by averaging the forecast probabilities for the first category. Quality data control checks within the program ensure that missing forecasts and/or observations are not used in developing threshold values. Once an initial probability threshold guess has been established, 2x2 contingency tables are created by comparing the forecast probability values to the current threshold guess. If the forecast probability is greater than or equal to the threshold guess and the observation indicates the event has occurred, a "*yes*" forecast and "*yes*" observed are noted in the table. Other combinations of "*yes-no*", "*no-yes*" and "*no-no*" are recorded in the table as well. Once the forecasts and corresponding observations for each group of stations have been exhausted for all dates, the corresponding bias and threat score are calculated. If the bias falls within the user-specified range, the program terminates and the corresponding threshold is selected as the final threshold. If the bias value remains greater than or less than the bias range, the current threshold value is either increased or decreased, respectively, by delta (a small incremental amount specified by the user). If for some reason the bias value never falls within the bias range when using the same value of delta, delta is halved and then added to the current threshold guess. In either case, a new 2x2 contingency table is built by using this new threshold guess. This process is reiterated until a 2x2 table is created which yields a bias value within the bias range or until the algorithm determines that a bias value cannot be obtained within the range. In the latter case, the threshold value corresponding to the bias which is closest to the bias range is chosen as the final threshold. In the same manner, the algorithm proceeds stepwise through each category until all categories are exhausted. When evaluating discrete probabilities, the probabilities are summed from above in order to make them cumulative from above, before proceeding through the algorithm.

If the entire set of observations (spanning the input dates) for a particular group of stations is missing, the threshold values, 2x2 contingency tables and all associated skill scores are set to missing (9999). A quality control check at the end of the subroutine ensures that all threshold values lie within the range of .001 and .999.

BFBLOW:

Subroutine BFBLOW is designed to accomplish the same task as BFABVE. The only major distinction between the two subroutines is that the former is designed to accommodate cumulative from below forecast probabilities (forecasting the likelihood of visibility $\leq$ .25 miles, $\leq$ .50 miles, $\leq$ 1.0 mile, etc.) or discrete probabilities and the user wishes to obtain the categorical thresholds by starting at the lower end of the categorical breakpoints (for example, sky cover amount). Consequently, the only change made to the algorithm involves the criteria for building the 2x2 contingency tables. The criterion for an observed event in BFBLOW requires that the observation is *less* than the threshold specifying the categorical definition. This is different from BFABVE where an observed event corresponds to the observation being greater than or equal to the threshold specifying the categorical definition.

TSABVE:

Subroutine TSABVE is designed to provide the user with threshold values that correspond to the greatest threat score within a specified bias range. The probabilities used in this program are cumulative from above or discrete and the user wishes to obtain categorical forecasts by starting at the upper end of the categorical breakpoints. The algorithm for TSABVE can be divided into two major parts: (1) entering the bias range and (2) searching within the bias range for the greatest threat score value and its corresponding threshold value. The first portion of this task is accomplished in the same exact manner as in BFABVE. The second task begins by using the bias value found in part (1) for each group of stations. (Note: In the instance where a bias value within the range cannot be calculated for a particular group of stations, the final threshold value will correspond to the bias value closest to the range which displays the greatest threat score.) TSABVE uses the value of “edelta” specified by the user as the threshold increment probability by which 2x2 contingency tables and threat scores are generated for multiple values within the bias range determined in part (1). Usually, edelta is set to a very small number (~.005) to establish with some certainty that the maximum threat score value is indeed obtained. Bias and threat score values for the entire range are calculated by iteratively *adding* edelta to the initial threshold (i.e., the first threshold value within the bias range) until the bias value for each group of stations is less than the lower end of the bias range. Once this occurs, edelta is once again *subtracted* from the initial threshold until the bias value for the group of stations exceeds the upper end of the bias range. For each iteration, the current threat score and previous threat score values are compared. The maximum of the two values is saved and then compared with the threat score obtained in the subsequent iteration. When all iterations are complete, the threshold within the bias range that corresponds to the maximum threat score is chosen as the final threshold value. In the same manner, the algorithm proceeds stepwise through each category until all categories are exhausted. As noted above, if discrete probabilities are used, (N-1) thresholds are generated, where N is the total number of categorical forecasts. In other words, all categorical forecasts will have a corresponding threshold value excluding the lower-most category. A quality control check ensures that the final threshold values lie within in the range of .001 and .999.

TSBLOW:

The algorithm of TSBLOW is virtually a clone of TSABVE except for one notable exception. The former subroutine exclusively deals with cumulative from below probabilities or discrete probabilities and the user wishes to obtain the categorical thresholds by starting at the lower end of the categorical breakpoints. In order to accommodate this change, the generation of 2x2 contingency tables and corresponding bias values and threat scores follows the algorithm noted in BFBLOW. Thus, an observed event is said to have occurred when the observation is *less* than the value of the variable threshold. Once again, it is also important to note that if discrete probabilities are being used as input, (N-1) categorical thresholds will be generated, i.e., all categorical forecasts will have a corresponding threshold value excluding the upper-most category. Finally, a quality control check ensures that the final threshold values lie within in the range of .001 and .999.

U855

VERIFIES GRIDDED FORECASTS

Harry R. Glahn
January 1, 2003

PURPOSE: U855 collates data from a variety of MOS-2000 TDLPACK grid inputs and "verifies" one or more sets of gridded forecasts with gridded observations. Seven verification packages are provided and each may have more than one output package associated with it. Different output packages give the convenience of printing verification "scores" in different formats. Because the packed input data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in a random access file; these data are also in TDLPACK format. U855 uses a driver DRU855 so that dimensions of variables can be tailored by PARAMETER statements to user needs without requiring a separate copy of the main program U855 (actually, subroutine) for every application. U855 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. An explanation of the scores and output packages is given in Appendices I and II. Some familiarity with MOS-2000 documents will be necessary for full understanding of this write-up.

CONTROL FILE INPUT: 'U855.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU855.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U855" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U855 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U855HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U855 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). Values have been defined as indicated for values of J below:

(1) =	All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU855 DATA statement = 12. Changing the default unit number allows multiple runs of U855 or other programs within the same directory without overwriting.
(2) =	The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) =	The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) =	The WFO (or area) list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) =	The WFO (or area) directory information and WFO subcenter numbers. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) =	The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.
(7) =	The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) =	The variable ID list in summary form. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)
(9) =	The variable list in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).
(10) =	The variable ID's for the first day (day 1) as read from the input files. This is just a list of all the variables on the input files.
(11) =	The variable ID's of the data actually needed.
(12) =	Not used.
(13) =	Not used.
(14) =	Gridprint output.
(15) =	Total variable list needed, including where the variable is to be obtained (input directly or computed in a subroutine).
(16) =	For printing data under the control of JP(,1). (Not implemented.)

(17) = For printing gridpoint values under control of JP(,2). (Not implemented.)
(23) = Information concerning opening and closing of files.
(25) = IDs for variables in LSTORE(,) after GCPAC discards those not needed for the next cycle when the number of cycles is $\leq$ 5. Also, with the /D compile option, lists the variables in LSTORE(,) before the discard process before and after LMSTR8 and LSMST3, not contingent on the number of cycles.

Record Type 2 - Format (A72) Run Identification

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (7(I10/),F10.0/(I10)) Control Parameters

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

NSKIP - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

JSTOP - The total number of errors that will be tolerated before the program halts.

NSTIND - Indicates whether (=1) or not (=0) scores for individual WFOs or areas are to be computed. Overall scores and scores for groups (if there is more than one) will always be computed and output.

INCCYL - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. No date/time should be used that cannot be arrived at by successively adding INCCYL to the first date read. That is, if the first date/time is Jan. 1, 1996, 00 UTC and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship.

MPROJ - The map projection number (i.e., 5 = polar stereographic).

ORIENT - The map orientation; the vertical longitude.

XLAT - The latitude where the gridlength is specified.

ALAT - The latitude of the lower left corner of the grid being used.

ALON - The longitude of the lower left corner of the grid being used.

NX -	The X-extent of the grid in number of gridpoints.
NY -	The Y-extent of the grid in number of gridpoints.
NXGMIN -	The minimum (first) gridpoint in the X-direction of the "disposable" grid to print. A zero means a disposable grid will not be used. (Not implemented.)
NXGMAX -	The maximum (last) gridpoint in the X-direction of the grid to print (not implemented).
NYGMIN -	The minimum (first) gridpoint in the Y-direction of the grid to print (not implemented).
NYGMAX -	The maximum (last) gridpoint in the Y-direction of the grid to print (not implemented).
DMESH -	The gridlength of the "disposable" grid printed with the boundaries NXGMIN, NXGMAX, NYGMIN, and NYGMAX specified above (not implemented).
PXMISS -	The value to be used instead of 9997, if a 9997 is encountered in the data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value.
JPRINT -	The number of cycles of data to write for printing to unit IP(16) under the format control and JP(,) provided with each variable (see Record Types 17, 18, and 19). (Not implemented)
ICHARS -	The number of characters of call letters to print when printing is indicated by JP(,). This is constrained to be between 4 and 8 inclusive. (Not implemented)
LNGTH -	The line length for printing to Unit IP(16). For a line printer, 132 is appropriate; for a laser printer, 80 may be desirable. For some output packages, if the line length is less than what would be required for all variables, no print occurs.) (Not implemented)
NROUND -	Rounding parameter. 0 = No rounding. 1 = Rounding to the nearest hundredth. 2 = Rounding to the nearest tenth. 3 = Rounding to the nearest whole number. 4 = Rounding to the nearest ten. See the individual output packages in Appendix I for additional information.

Record Type 4 - Format (I3,4XA60) Date List File

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator

KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

<ins>KFILDT</ins> - Unit number for the file containing the input date list.

<ins>DATNAM</ins> - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) <ins>Date List</ins>

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

<ins>IDATE</ins>(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Date/times should not be closer together than INCCYL and must always be derivable by successively incrementing IDATE(1) by INCCYL. For example, if the first date/time is Jan. 1, 1996, 00 UTC and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

Record Type 6 - Format (I3,4XA60) <ins>Gridpoint Input Data Files</ins>

This group of records identifies the data sets from which the gridpoint data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN.

<ins>KFILIN</ins>(J) - Unit numbers for the data files.

<ins>NAMIN</ins>(J) - Names of files where these data reside. (CHARACTER*60)

An input could be from more than one model (or source), or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard. When data sets are to be used in sequence, they should be read in the proper order, be in sequence, and have the same unit number. That is, if 2 years of data are to be used and one year is on one data set and the other year on another, then the first should immediately precede the second in the list and both should have the same unit number.

Record Type 7 - Format (I3,4XA60) <ins>Random Access Files</ins>

This group of records identifies the MOS-2000 random access data sets from which constant (and possibly other) data are read. Records are read until the terminator KFILRA( ) = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. If no data are needed from a random access file, only the terminator is necessary. Although 5 is the limit, only one with KFILRA( ) = 44 will be used. (Not implemented.)

KFILRA(J) - Unit numbers for the random access constant files (J=1,NUMRA). Unit number must be 44; see "Restrictions" for more information.

RACESS(J) - Names of files of constant data (J=1,NUMRA). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) Binary Area Scores Output File

This record (plus the terminator record) identifies the vector binary output data set. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. The scores computed are packed and written in vector form as are all other MOS-2000 vector records. (The actual writing has not been implemented, but this record type must be in the input, and the file, if one is designated, is opened as 'NEW'.)

KFILOV - Unit number for the vector binary output file.

OUTVEC - Name of file of vector binary output. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) Binary Gridpoint Output File

This record (plus the terminator record) identifies the gridpoint binary output data set. Records are read until the terminator KFILOG = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 9 is read by subroutine RDSNAM. The scores computed are packed and written in gridpoint form as are all other MOS-2000 gridpoint records. This Record Type applies only to Score Group 7.

KFILOG - Unit number for the gridpoint binary output file.

OUTGRD - Name of file of gridpoint binary output. (CHARACTER*60)

Record Type 10 - Format (I3,4XA60) Station and Location Files

This pair of records (plus the terminator record) identifies the file(s) which hold WFO information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 10 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the WFO call letters for which data are to be processed (J=1) and the station directory which holds the call letters, subcenter numbers, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD(J) = KFILDI, DIRNAM(J) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Types 11A and 11B - Formats(7(A8,1X)) and (A30) Station Lists and Names

This group of records identifies the (groups of) WFOs and group names for which forecasts are to be verified. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of areas (e.g., WFOs) for which forecasts are to be verified (K=1,NSTA). This list (**Record Type 11A)** is read within subroutine RDAREA by RDC, which eliminates any blanks found in the input. Duplicate areas in the list are kept, but a diagnostic is furnished on unit IP(5). The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. Following a group terminator is the group name (**Record Type 11B**) consisting of up to 30 characters; this is to identify the group on printout. An empty set terminates the station input. That is, the last group and its terminator and name must be followed by another terminator signifying an empty set. Maximum number of areas, sans terminator, is ND1. (CHARACTER*8)

Record No. 12 - Format (I3,4XA60) Variable ID File

This record (plus the terminator record) identifies the file from which the variable ID's (forecasts, observations, and matching data and projections for each) are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record No. 12 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's and projections.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record No. 13 - Format (I3,4XA60) Variable Constants File

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record No. 13 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

The following Record Types occur in groups, and define the score groups to compute, the scores to compute within those groups, the variables to use, and the output packages for the groups. Reading of Record Types 14 through 19 continues until the terminator SCOREG = '999999' (instead of SCOREG = 'SGROUP') is reached in Record Type 14.
In addition to the possibility of the sequence of Record Types 14 through 19 being repeated, the pairs of Record Types 15 and 16 can be repeated within that series. That is, for the score group number defined in Record Type 14 and the variables defined in Record Types 17 through 19, there can be more than one output package and scores defined for each by the Record Types 15 and 16. Pairs of Record Types 15 and 16 are read until the terminator SCOREN = '999999' (instead of SCOREN = 'OUTPUT') is reached.

Also, each of Record Types 17, 18, and 19 can have subgroups of records, each ending with the terminator 888888. The number of subgroups in records Types 17 and 18 must be equal, and the number of subgroups in Record Type 19 must be that same number or be zero.

The score group numbers, the output packages available, and the scores for each group are explained in Appendix 1. The scores are defined in Appendix 2.

Record Type 14 - Format (A6,I4) Score Group Number

This record identifies the score group number.

SCOREG - The characters 'SGROUP' to define this record.

ISSGP - The number of the score group (see Appendix I for definitions).

Record Type 15 - Format (A6,I4) Output Package Number

This record identifies the output package number to use for the score group number just read.

SCOREN - The characters 'OUTPUT' to define this record.

IDSOUT - The number of the output package for this group (see Appendix I for definitions).

When '999999' is encountered instead of 'OUTPUT', the reading of Record Types 15 and 16 is terminated for this score group.

Record No. 16 - Format (A6,30I4) Score Numbers

This record identifies the score numbers to output for the score group number and output package just read.

SCORES - The characters 'SCORES' to define this record.

IDSCOR - The numbers of the scores to output for this group and output package (see Appendix I for definitions).

Note that Record Types 15 and 16 can be repeated, and must be terminated with '999999'.

Record Type 17 - Format <u>Forecasts to Verify</u>
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4XI2,15XA1,I2,1XI1,1X30A1)

In this Record Type 17, each group of records ending with the terminator 888888 contains the ID's of the forecasts to verify. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1, ,1) = 888888 or 999999 is reached. An 888888 indicates this is the end of the forecast IDs. A 999999 indicates this is the end of the forecasts. Maximum number of records, sans the terminator record, = ND4. This Record Type 17 is read by subroutine RD86 called by RDVR86. Upon completion of reading this record type, N=1,NVRBL(1). Note that the format and information for ID(, ,) are exactly the same as for reading predictors in U201. Some score groups use only one variable to be verified by the observation per forecast system (e.g., for Score Group 1, each variable read here is a forecast system to be verified for a particular projection). However, some score groups may use more than one variable per forecast system. See Appendix I for more details on each score group.

<u>ID</u>(J,N,1) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

<u>JP</u>(N,J) - JP(N,J) indicates whether (>0) or not (=0) variable N will be written to Unit IP(16) with the format provided (J=1) or to IP(17) (J=2). <u>(Not implemented)</u>

<u>CFMT</u>(N,1) - The format descriptor for writing on Unit IP(16) when JP(N,1) > 0. Must be either "F" or "I".

<u>IWDTH</u>(N,1) - The field width for writing on Unit IP(16) when JP(N,1) > 0. Must be ≤ 30. When ≥ 10, the ID(,N,1) will be printed as a heading in addition to HEAD(,N,1) (see below). IWDTH(N,1) also controls the print width for some output packages for Score Group 1 (see Appendix I).

<u>IPREC</u>(N,1) - The precision for writing on Unit IP(16) when JD(N,1) > 0. For an "F" format, this is the number of digits after the decimal point. For an "I" format, it is the number of digits always written. Note that for a "zero" to be printed, IPREC(,1) must be > 0; otherwise, zero will be printed as a blank.

<u>HEAD</u>(J,N,1) - The column heading for writing to Unit IP(16) when

JP(N,1) > 0. Limited to J=1,30 characters. All 30 characters are read, but only IWDTH(N,1) characters are written to Unit IP(16). HEAD(,N,1) is right justified when writing to Unit IP(16). For printing of scores, HEAD(,N,1) is left justified, and in most instances no more than 11 to 13 characters can be accommodated when printing "IMP OVER".

Record No. 18 - Format <u>Verifying Observational Grid</u>
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4XI2,15XA1,I2,1XI1,1X30A1)

In this Record No. 18, each group of records ending with the terminator 888888 contains the ID's of the verifying grid. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1, ,2) = 888888 or 999999 is reached. A 999999 indicates this is the end of the verifying grids. Maximum number of records, sans the terminator record, = ND4. This Record Type 18 is read by subroutine RD86 called by RDVR86. Upon completion of reading this record type, N=1,NVRBL(2). The information read is explained in Record Type 17 and is not repeated here. Some score groups use only one variable for the verifying grid. However, some score groups (e.g., Score Group 2) may use more than one. See Appendix I for more details per score group. Note that there may be multiple "systems" in Record Type 16, but all are verified by the one set of observations in the corresponding subgroup in Record No. 18. The projection must = 0.

<u>ID</u>(J,N,2) - See Record Type 17.

<u>JP</u>(N,J) - See Record Type 17.

<u>CFMT</u>(N,2) - See Record Type 17.

<u>IWDTH</u>(N,2) - See Record Type 17.

<u>IPREC</u>(N,2) - See Record Type 17.

<u>HEAD</u>(3,N,2) - See Record Type 17.

Record Type 19 - Format <u>Matching Variables</u>
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4XI2,15XA1,I2,1XI1,1X30A1)

In this Record Type 19, each group of records ending with the terminator 888888 contains the ID's of variables that must have non-missing values for the forecasts to be verified. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1, ,3) = 888888 or 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 19 is read by subroutine RD86 called by RDVR86. Upon completion of reading this record type, N=1,NVRBL(3). The information read is explained in Record Type 17 and is not repeated here. It may be that there are no IDs in this Record Type 19, and the only record contains the terminator 999999. The binary variable B = IDPARS(3) can be used for stratification (see Comments section). The projection must = 0.

ID(J,N,3) - See Record Type 17.

JP(N,3) - See Record Type 17.

CFMT(N,3) - See Record Type 17.

IWDTH(N,3) - See Record Type 17.

IPREC(N,3) - See Record Type 17.

HEAD(J,N,3) - See Record Type 17.

CONTROL FILE INPUT: (Name read from U855.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) Date List

When the dates are not provided in file U855.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU855), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Date/times should not be closer together than INCCYL and must always be derivable by successively incrementing IDATE(1) by INCCYL. For example, if the first date/time is Jan. 1, 1996, 00 UTC and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U855.CN) (Unit = KFILD(1))

Record Types 1A and 1B - Formats (7(A8,1X)) and (A30) Station Lists and Names

When the station list is not provided in file U855.CN, this group of records identifies the areas or WFOs and group names to be verified. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of areas (e.g., WFOs) for which forecasts are to be verified (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of areas, sans terminator, is ND1. (See Record Type 11A, control file 'U855.CN', unit KFILDI for additional information.) (CHARACTER*8)

<u>CONTROL FILE INPUT</u>: (Name read from U855.CN) (Unit = KFILD(2))

Record Type 1 - Format <u>Station Locations</u> (A8,1X,I8,1XA17,4XA2)

This group of records provides information about the WFOs (or areas). The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDAREA to give information about the WFOs.

<u>CCALLD</u>(K) - Call letters (or other character location designator) of WFOs (or areas). As stated above, these call letters are matched with those in the station list.

<u>NSUBNO</u>(K) - Subcenter number of WFO matching CCALLD(K). This is used to define where on the grid this area is located.

<u>STNAME</u>(K) - 20-character name of area or WFO. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

The number of stations in this directory is not limited. No terminator is used.

<u>CONTROL FILE INPUT</u>: (Name read from U855.CN) (Unit = KFILP)

Record Type 1 - Format <u>Forecasts, Verifying grids, and Matching Variables</u>
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4XI2,15XA1,I2,1XI1,1X30A1)

<u>When the variables to use in the run are not in file U855.CN</u>, this group of records is read in trios and contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1, ,1) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RD86 called by RDVR86. Note that the format and information for ID(, ,) are exactly the same as for reading predictors in U201. See Control File Input for U855.CN, Record Type 17, unit KFILDI above for more detailed information; the format is exactly the same.

Note that this Record Type 1 is repeated twice (three times total) for each score group defined (see Record Types 17, 18, and 19).

<u>CONTROL FILE INPUT</u>: (Name read from U855.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) <u>Variable Constants</u>

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U855 users, and is expected to be a separate file.

Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=2,4) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. For forecasts and observations, the first 4 characters can indicate the model (e.g., ETA) from which the forecasts were made, or for the AEV data, MOS or LCL can be used. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the data for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer.

Other processing occurs with the reading of these records in RDVR86, much of it associated with the plain language description. This is done so that these records can be somewhat generic and apply to many variables as defined in ID(, ,1). See subroutine SETPLN for more information on the use of plain language.

DATA INPUT:

All data input to U855 will be gridded and be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U855.CN in Record Types 6 and 7.

A. SEQUENTIAL GRIDPOINT DATA

One or more sources of gridpoint data, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U855.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened.

B. RANDOM ACCESS VECTOR DATA

One source of random access data is accommodated on Unit No. 44 in the MOS-2000 External Random Access File System. See "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

DATA OUTPUT

There are two forms of data output, besides the diagnostics and other information on Units KFILDO and IP( ).

A. ASCII FOR VIEWING OR PRINTING DATA

A maximum of NPRINT (see Record Type 3) or NDATES (see Record Type 5) cycles of data will be written to the file on Unit IP(16) under control of the format provided with each variable N, the variables written controlled by JP(N,) (see Record Type 17). The data columns are headed by HEAD(,N,) (see Record Type 17). Also, provided the column width specified by IWDTH(N) (see Record Type 17) is $\geq$ 10 characters, the 4-word ID will be written. Listing is by station group. If a line length for printing is not sufficient for all variables, printing may not be done. Line length is specified as LNGTH characters in Record Type 3.

B. ASCII FOR VIEWING OR PRINTING SCORES

The scores designated in Record Type 16 and the output packages designated in Record Type 15 are written for viewing and/or printing.

C. BINARY Vector MOS-2000 FORMAT (Not yet implemented)

The scores will be written to a binary file according to a designated output package.

D. BINARY Gridpoint MOS-2000 FORMAT

The scores will be written to a binary file according to Output Package 71 for Score Group 7 (only). The records written can be put onto a map, possibly with ugem.ksh, via U203.

EXAMPLE CONTROL FILE: 'U855.CN'

Examples exist as file 'U855.CN' in directory home21/glahn/ndfd/dru855 on blizzard. There is an example of the .CN file and the resulting ftn12 output for each of the score groups implemented, namely, 1, 2, 3, 6, and 7. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U855.CN', Record Type 1 and the definition of KFILDO in the driver DRU855. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. For machines that do not allow this, it is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

There is no check in U855 for a legitimate value of "B" in the first word of the ID. For a value of 3, indicating a discrete binary, the upper threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID's. If there is no lower one, the lower threshold is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999.

Since the variables are not ordered by the program (as they are in U600), if a discrete binary is desired (one with both upper and lower thresholds), these variable ID's must be input in sequence and in the correct order (lower threshold first). To make sure all is well, check the thresholds on Unit IP(8) or IP(9). Since scores are identified and variables read in groups, the order of the variables within a score group will usually be important and will be dictated by the subroutines computing the accumulants and scores (see the appendix for further explanation).

It may be necessary for each variable needed for Day 1 to actually be available for Day 1 for the variable to be used on subsequent cycles. If the variable needed does not have to be obtained through the "option" subroutine OPTGRD, then U855 can recover even if the data are not there for day 1. However, even a "units" change may keep U855 from proceeding correctly. This restriction is due to efficiency considerations.

The unit number for the random access file must be 44.

COMMENTS

U855 was derived from U850; some of the code is identical, and other is very similar. Even so, there are major differences. The input data are TDLPACKed gridded, not TDLPACKed vector. It is desirable to be able to:

1. Compute some scores at individual gridpoints and to be able to display them as a grid,

2. verify gridpoints grouped by WFO CWA,

3. verify areas composed of groups of WFO CWAs (e.g., an NWS Region), and

4. verify areas not necessarily related to WFO CWAs.

For purposes of transmission of GRIB2 messages, a WMO "subcenter" number is assigned to each WFO. A grid exists (see Record Type 10) on which each gridpoint has the value of the subcenter number of the WFO responsible for that CWA. This grid is used to group the data into "vector" format for further processing and save in the MOS-2000 Internal Random Access System.

So, the data being dealt with exist in the MOS-2000 Internal Storage System in both gridded and vector format, with identical IDs except the "G" in ID(4) is "1" for gridded and "0" for vector. Especially when processing only a portion of the grid, this reduces the computation, which has become important with such large grids.

The terminology of "area" and "WFO" may be confusing. Although a map currently exists only for relating subcenter numbers for WFOs to gridpoints, if a grouping of gridpoints were wanted that was not a combination of WFO CWAs (for instance an RFC area of responsibility), a "subcenter" map could be designed to do that (dummy subcenter numbers could be assigned, although they already exist for RFCs). Then, the individual scores would not be for "WFOs" but be for "areas."

Several possible capabilities of U855 have not been implemented, although "hooks" have been left from U850 or otherwise been built in. These possible capabilities include:

1. Score Groups 4 and 5 have not been implemented; they are for probability forecasts, and no verifying analysis (yes or no for ground truth) exist for checkout. These will need to be implemented.

2. The capability to output a portion of the grid would use the variables in Record Type 2 MXGMIN, MXGMAX, MYGMIN, MYGMAX, and DMESH -- a so-called "disposable" grid.

3. Although the writing of the gridpoint binary file for Score Group 7 has been implemented, scores for individual WFOs or groups could be written for further processing and display for other score groups.

4. The use of "matching variables" in Record Type 19 has not been checked out, although the capability is there and it may work.

5. There may need to be additional computational subroutines written and implemented.

6. U855 is built primarily for matched samples, and that is what has been checked. It may be that it will work for unmatched samples, but this has not been tried and the input format may not accommodate it. It seems that most or all uses would want matched samples, and if unmatched samples were desired, separate runs could be made.

7. The values of B = 6, 7, 8, and 9 have not been checked out.

8. The use of random access files has not been implemented.

In the MOS-2000 ID, only 2 digits are allowed for "RR," the lookback feature. For U855, it will be desirable to verify forecasts made at least 7 days previously, so RR would have to be 168 as well as tau. U855 uses the "T" immediately before RR in ID(3) to be part of RR; this eliminates the use of the "T" for its intended purpose, but it would not be needed anyway.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U855 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 3). Presumably, the only time 9997 will occur is when operational or test MOS forecasts are input. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such. These values will be used the same way no matter how they are packed.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. This probably means a set of dates should be supplied to U855 for which it is known there are good data; then any diagnostic is really unexpected. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities. For instance, subroutine GCPAC prints compression information for the first 3 days. This will let the user determine whether the CORE( ) space provided for storage (i.e., ND10 words, see section below) is far larger, or smaller, than needed, etc. If CORE( ) is small, much disk access will be necessary. If it is large, then other users or swapping space for the current run may be impacted. Generally, it is hoped CORE( ) can be about the size to hold the intermediate storage after Day 1. Some systems may handle caching so efficiently that the size of ND10 does not matter much; also system setup (how much space available for swapping, etc.) may be a factor.

As a special feature for U855, B = IDPARS(3) in ID(1) can be 8 or 9 in addition to the usual values of 0, 1, 2, 3, or 5. This is used for stratification. When a matching variable has B = 8, all non missing values of that variable greater than the threshold in ID(4) will be set to 9999 (missing) and all other values will be set to 0. (Setting to zero is not important in U855, but it is possible this capability can be used in U600, and there the zeros would be important.) When a matching variable has B = 9, all non missing values of that variable less than the threshold in ID(4) will be set to 9999 and all other values will be set to 0. Since verification is on a matched sample, missing values will eliminate the case from the computations.

As an additional special feature, B = IDPARS(3) in ID(1) can be 6 or 7 to accommodate "OR" matching variables. B = 6 or 7 operates essentially like B = 8 or 9, respectively, except that logic in U855 will use that case if either of the variables is "non-missing." For B = 8 or 9, all such variables have to be non-missing for the case to be used.

SETTING UP THE DRIVER DRU855

The preparation of the driver for a particular U855 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B - Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).

ND1 - Maximum number of areas (or WFOs) that can be dealt with. Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)).

ND4 - The maximum number of forecasts, observations, and matching variables that can be dealt with (total of all).

ND6 - Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc., even though the same unit number might be used.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times. When verifying forecasts made several cycles before the verification valid time, ND9 may have to be several tens of thousands.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

ND11 - The maximum number of score groups.

ND13 - The maximum number of scores (numbers) to output for each of the score groups and output packages.

ND14 - The maximum number of grid characteristics that can be dealt with.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System.

The user can see from the DRU855 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

PROGRAM STRUCTURE

See Appendix III.

WRITING NEW SUBROUTINES

It is not expected that computational routines will be used to any great extent in U855. An "option" subroutine, OPTGRD, is provided for possible use. For instance, in verifying wind direction, the direction might not be available directly, but need to be computed from u and v components through OPTGRD. Also, unit conversions may be necessary.

NONSYSTEM ROUTINES USED

Use the load line in home21.glahn,ndfd.dru855, dataset 'u855.com', which includes the libraries 'home21.glahn,ndfd.u855lib', 'home21.glahn,ndfd.u850lib' and 'home21.tdllib.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: /mdl/mos2k/mdllib/u855lib. The driver is in /mdl/mos2k/mdllib/u855lib/RUN/dru855.

APPENDIX I

SCORE GROUPS, SCORES, AND OUTPUT PACKAGES

The scores are produced and output in groups. Each score group has one or more output packages, giving flexibility in the form of the output. The score group numbers are read in Record Type 14; the associated scores to be output are read in Record No. 16. The output package to be used is read in the associated Record Type 15.

When more than one "system" is verified (more than one set of forecasts) with the same score group, for some scores an "improvement" over the first "system" (set of forecasts) is calculated and (can be) output. For instance, if the first set of forecasts were climatological relative frequencies and the second set were probability forecasts, an improvement over climatology would be calculated.

Score Group numbers provided are 1, 2, 3, 4, 5, 6, and 7. Each output package provides for printing overall scores (which is always done) scores by groups of stations (when there is more than one group--see Record Type 10), and by station (when desired--see Record Type 1, NSTIND) for each projection desired (see Record Type 17).

It is possible to verify a forecast grid valid at a particular date/time with an "observation" grid taken at (valid for) another date/time.

The NROUND parameter (See Record Type 3) is relevant and/or applied differently for different score groups. NROUND applies after any computation, including the making of binaries, is done to produce the data for the production of scores. That is, the data are ready for computation of accumulants for scores, and rounding, if any, is done then. See the individual score groups in this appendix for additional information.

Score Group 1

This score group is generally for continuous forecasts, for example temperature or wind speed. Usually, these forecasts and observations would not have thresholds associated with them. Each forecast variable read (Record Type 16) for a particular projection is a system to be verified. So, the number of variables read in Record Type 17 is the number of "systems" to be verified, where a system can be a forecast from the same source (e.g., the NDFD) made earlier, or actually from a different source (e.g., gridded MOS or direct model output). The single corresponding variable read in Record No. 18 is the matching grid for each such system for that projection.

NROUND (See Record Type 3) applies to both forecasts and observations.

Score No.	Explanation
1	Sample size (always output whether or not specified).
2	Mean of observations.
3	Variance of observations.
4	Standard deviation of observations.
5	Mean of forecasts.
6	Variance of forecasts.
7	Standard deviation of forecasts.
8	Bias of forecasts (fcst-obs).
9	Mean absolute error (MAE) of forecasts.
10	Mean square error of forecasts.
11	Root mean square error (RMSE) of forecasts.
12	Log score.
13	Fractional improvement of log score.
14	Ratio of variance of forecasts to observations.
15	Fractional improvement of MAE.
16	Fractional improvement of RMSE.
17	Correlation coefficient of forecasts with observations.

Output Package

01 Prints scores (e.g., mean for forecasts) for all systems on the same line if there is room, the maximum line length being determined by LNGTH (see Record Type 1).

02 Essentially the same as Output Package No. 01.

03 Prints all scores on one line by projection. That is, for a particular projection, all the scores (1 through 17) specified (see Record No. 16) for all systems are put on one line. This may make for a very long line, and LNGTH is allowed to be exceeded; if this is the case, a diagnostic is printed, and output to a printer may not give pleasing results. For this output package, a particular score number may trigger print for related scores, according to the following table:

Score No.	Explanation
1	Sample size (always output whether or not specified).
2 or 5	Mean of observations and forecasts.
3 or 6	Variance of observations and forecasts.
4 or 7	Standard deviation of observations and forecasts.
8	Mean of observations and forecasts and bias of forecasts (fcst-obs).
9	Mean absolute error (MAE) of forecasts.
10	Mean square error of forecasts.
11	Root mean square error (RMSE) of forecasts.
12	Log score.
13	Log score and fractional improvement of log score.
14	Variance of observations and forecasts and ratio of variance of forecasts to observations.
15	MAE of forecasts and fractional improvement of MAE.
16	RMSE of forecasts and fractional improvement of RMSE.
17	Correlation coefficient of forecasts with observations.

04 The same as Output Package No. 02.

Score Group 2

This score group is generally for continuous forecasts (including probability forecasts) that are put into categorical form, for example wind speed or probabilities of ceiling height. These variables would usually have a threshold associated with them; each variable would be input for each threshold needed, and would be input in ascending order of thresholds. The discrete categories necessary for forming a contingency table are determined by the series of inputs. The upper threshold of a category is the one in the associated ID; the lower threshold of that category for that variable is the threshold of the previously read variable with the same ID (except, of course, for the threshold). If there is only one threshold for a variable, or if it is the first threshold for that variable, the lower threshold is automatically set at a large negative value. The last threshold read for a variable with more than one threshold should normally be 9999. A set of forecast variables read (Record type 17) for a particular projection, each of which is the same except for the threshold, comprise the forecast system to be verified. When more than one system is to be verified for a projection, one set immediately follows another. The single set of observation variables read (Record Type 18) is used to verify each system for the corresponding projection. It is expected that the number of categories for the observations is the same as the number of categories for the forecasts.

NROUND (See Record Type 3) is not relevant because the variables as specified in Record Types 17 and 18 should be binary.

The forecast and observation thresholds for each system verified are checked with those for each other system, and the thresholds for forecasts are checked with those for the verifying observations. If there are differences, a **** diagnostic is written. This may not be an error, but is a safeguard.

Score No.	Explanation
21	Contingency table of forecasts and observations.
22	Percent correct and improvement.
23	Heidke skill score and improvement.
24	Threat score and improvement, Probability of Detection (POD), and False Alarm Ratio (FAR) for category 1 (lowest category).
25	Same as 24 except for categories 1 and 2 combined.
26	Same as 24 except for the upper category.
27	Same as 24 except for the upper two categories combined.
28	Same as 24 except for the upper three categories combined.
29	Log score and fractional improvement.

Output Package

21	Prints scores for all systems together. That is, overall scores for all systems are output together, then groups of stations for all projections, etc.
22	The same as Output Package No. 21

Score Group 3

This score group is for wind direction forecasts. Each pair of input forecasts read (Record Type 17) for a particular projection is a system, a direction followed by a speed. When more than one system is to be verified, one pair immediately follows another. Each verifying observation for the corresponding projection is also composed of a direction followed and a speed. So, the number of systems to be verified is the number of pairs of variables in Record Type 17. Thresholds are not used; rather the errors in direction are put into 9 categories defined internally.

NROUND (See Record Type 3) applies to wind direction only.

Score No.	Explanation
31	The relative frequencies (RF), cumulative relative frequencies (CRF), improvement of the system over the first system in terms of CRF, and frequencies of direction errors in 9 categories, the upper category boundaries being 5, 15, 25, 35, 65, 95, 125, 155, and 180 degrees.
32	The same as Score No. 31, except only for cases when the observed speed is $\geq$ 10 kt.
33	The same as Score No. 31, except only for cases when both the observed and forecast speed is $\geq$ 10 kt. Note that this can create an unmatched sample for the different systems, because each is based on its own speed forecast.
34	MAE for wind direction.
35	Same as 34, except only when observed wind speed $\geq$ 10 kt.
36	Same as 34, except only when both observed and forecast wind speed $\geq$ 10 kt. Note that this will likely give unmatched samples when more than one system is being verified.

Output Package	
31	Prints scores for all systems together. That is, overall scores for all systems are output together, then groups of stations, etc.
32	Same as Output package No. 31.

Score Group 4 (Not Implemented)

This score group is for probability forecasts of a single event. Each forecast variable read in Record Type 17 for a particular projection is a system, and the corresponding observation variable for that projection read in Record No. 18 is the verifying observation for each system. So, the number of variables read in however, a threshold may be needed for the observation to define the event being forecast. That is, the observation might be precipitation, for which the threshold might be 0.01 inches, but could be some other value. If the observation is already binary, no threshold is needed.

NROUND (See Record Type 3) applies to forecasts only because the observations are binary.

Score No.	Explanation
41	P-score and fractional improvement.
42	Brier score and fractional improvement.
43	Event relative frequencies in 10 percent increments.
44	Same as 41, except forecasts constrained to 0 to 1 range.
45	Same as 42, except forecasts constrained to 0 to 1 range.
46	Same as 43, except forecasts constrained to 0 to 1 range.

Output Package

41	Prints scores for all projections together. That is, overall scores for all projections are output together, then groups of stations for all projections, etc.
42	Prints all scores grouped by projection. That is, all scores are printed for the first projection, then for the second projection, etc.

Note:

o A matching variable can be used with subroutine POPDIF to calculate scores for only those cases when the difference between MOS and local forecasts is $\geq$ some desired threshold.

Score Group 5 (Not Implemented)

This score group is for probability forecasts of a set of NCAT ranked events (e.g., discrete categories of ceiling height). For verification of "old" forecasts, the NCAT probability forecasts would not have a threshold in the ID, but the category number would be in ID(2) and IDPARS(6). A system for a particular projection is determined by a sequence of variables being the same except for IDPARS(6). IDPARS(6) would range upward from 1 in increments of 1. For verification of MOS-2000 forecasts, the NCAT probability forecasts would normally have a threshold in the ID, and there would be no category number in ID(2) or IDPARS(6). The thresholds would be in an upward numerical sequence. The verifying observation for that projection would normally have the upper category limits in the IDs for it to be made into discrete binaries; the variable IDs would be identical except for the thresholds. The number of forecasts to be verified and the discrete verifying binaries must be the same. Old and new forecasts can be mixed in one run; the improvement of MOS-2000 forecasts over old forecasts can be calculated. Relative frequencies with CCC = 4XX can also be verified.

NROUND (See Record Type 3) applies to forecasts only.

The forecast and observation thresholds for each system verified are checked with those for each other system (the forecast thresholds should all be zero). If there are differences, a **** diagnostic is written. This may not be an error, but is a safeguard.

Score No.	Explanation
51	P-score and fractional improvement.
52	Same as 51, except that forecasts are constrained to 0 to 1 range.
53	Reliability of forecasts; the mean forecast, relative frequency of observations, and number of cases in 10 percent increments of the forecasts.
54	Ranked probability score, with the scores constrained to 0 to 1 range, and the cumulative frequency adjusted when necessary.

Output Package

51	Prints scores for all projections together. That is, overall scores for all systems and then for all projections are output together, then groups of stations for all projections, etc.

Score Group 6

This score group is used to provide the frequencies and relative frequencies of errors in the forecasts. For instance, for the continuous variable max temperature, the algebraic or absolute difference of the forecast minus the observation is computed and the result put into discrete categories specified by the thresholds provided by the otherwise duplicative forecast IDs. This is accomplished through use of a subroutine GDIFF called by OPTGRD. In order to not have routines specific to one forecast and observation ID, an assumption was made that in the forecast ID(1) C8C means algebraic differences and C9C means absolute differences. This assumption works for temperature, dew point, and wind speed, possibly the only variables needing this score group.

NROUND (See Record Type 3) is not used. The differences are put into categories, so rounding is not needed. (Rounding could be hardwired into GDIFF if rounding is needed before the differences are computed.)

The thresholds for the errors for each system verified are checked with those for each other system. If there are differences, a **** diagnostic is written. This is likely an error.

Score No.	Explanation
61	Algebraic of differences are computed, depending on X = 8 or 9, respectively, in CXC.

Output Package

61	Prints overall, then grouped, then individual station scores.

Score Group 7

This score group is generally for continuous forecasts, for example temperature or wind speed; the scores computed are exactly the same as those in Score Group 1, except they are at individual gridpoints rather than being grouped by area (WFO). Usually, these forecasts and observations would not have thresholds associated with them. Each forecast variable read (Record Type 16) for a particular projection is a system to be verified. So, the number of variables read in Record Type 17 is the number of "systems" to be verified, where a system can be a forecast from the same source (e.g., the NDFD) made earlier, or actually from a different source (e.g., gridded MOS or direct model output). The single corresponding variable read in Record No. 18 is the matching grid for each such system for that projection. It is likely that only a few of the possible scores will be wanted at individual gridpoints for mapping, such as MAE.

NROUND (See Record Type 3) applies to both forecasts and observations.

Score No.	Explanation
1	Sample size (always output whether or not specified).
2	Mean of observations.
3	Variance of observations.
4	Standard deviation of observations.
5	Mean of forecasts.
6	Variance of forecasts.
7	Standard deviation of forecasts.
8	Bias of forecasts (fcst-obs).
9	Mean absolute error (MAE) of forecasts.
10	Mean square error of forecasts.
11	Root mean square error (RMSE) of forecasts.
12	Log score.
13	Fractional improvement of log score.
14	Ratio of variance of forecasts to observations.
15	Fractional improvement of MAE.
16	Fractional improvement of RMSE.
17	Correlation coefficient of forecasts with observations.

Output Package

71	Writes forecasts of the scores designated in TDLPACK format to Unit No. KFILOG. These can be mapped, perhaps through use of U203 and ugem.ksh.

APPENDIX II

DEFINITIONS OF SCORES

The scores computed by U855 are computed in groups. The groups were determined by similarity of computations. For instance, one group is concerned with contingency tables and the scores computed from them. All scores in a group are computed when the group is specified, but only the scores asked for are output.

The definitions and terminology follow, insofar as possible, the National Verification Plan published by the National Weather Service in June 1982 and still used as the primary basis of the National Verification Program. In the definitions, the following terminology is used:

N = sample size (number of cases)

f_i = the ith forecast,

o_i = the ith (matching) observation, and

$\Sigma = \sum_{i=1}^{N}$ summation over all N cases, except where indicated otherwise.

"Improvement" of a score means the improvement of the kth "system" or set of forecasts over the first. For instance, if the first set of forecasts was climatological means and the second set was MOS forecasts, an improvement of MOS over climatology would be calculated.

Score Group 1

This score group is generally for continuous forecasts, for example temperature or wind speed. Generally, these forecasts and observations would not have thresholds associated with them.

Score No.	Explanation
1	N = sample size.
2	$\bar{o} = \frac{1}{N} \sum o_i$ = mean of observations.
3	$\sigma_o^2 = \frac{1}{N} \sum (o_i - \bar{o})^2$ = variance of observations.
4	$\sigma_o = (\sigma_o^2)^{1/2}$ = standard deviation of observations.
5	$\bar{f} = \frac{1}{N} \sum f_i$ = mean of forecasts.
6	$\sigma_f^2 = \frac{1}{N} \sum (f_i - \bar{f})^2$ = variance of forecasts.
7	$\sigma_f = (\sigma_f^2)^{1/2}$ = standard deviation of forecasts.
8	$\bar{f} - \bar{o}$ = bias of forecasts.
9	$MAE = \frac{1}{N} \sum \lvert f_i - o_i \rvert$ = mean absolute error of forecasts.
10	$MSE = \frac{1}{N} \sum (f_i - o_i)^2$ = mean square error of forecasts.
11	$RMSE = (MSE)^{1/2}$ = root mean square error of forecasts.
12	$LS = \frac{50}{N} \sum \log_{10} \frac{f_i}{o_i}$ = log score.
13	$\frac{LS_1 - LS_k}{LS_1}$ = fractional improvement of log score.
14	$\frac{\sigma_f^2}{\sigma_o^2}$ = ratio of variance of forecasts to observations.
15	$\frac{MSE_1 - MSE_k}{MSE_1}$ = fractional improvement of MSE.
16	$\frac{RMSE_1 - RMSE_k}{RMSE_1}$ = fractional improvement of RMSE.
17	$CORR = \frac{\overline{fo} - \bar{f}\bar{o}}{\sigma_f \sigma_o}$ = correlation, where $\overline{fo} = \frac{1}{N} \sum (f_i o_i)$

Score Group 2

This score group is generally for continuous forecasts, for example temperature or wind speed, that are to be put into m categories for entering into a contingency table. These forecasts and observations are expected to have the full IDs including thresholds. That is, a series of forecasts is read in which have the same IDs except for the thresholds; these define the categories. The same is true for the observations.

Score No.	Explanation
21	Contingency Table--not a score as such, but

(a) Contains information on which all the scores in this group are computed. The element X_{ij} in the table is the number of times the forecast was in the jth category and the observation was in the ith category. The row and column totals are also shown here with the subscript p.

Observed	Forecast Category				
Category	1	2	...	m	Total
1	X_{11}	X_{12}	...	X_{1m}	X_{1p}
2	X_{21}	X_{22}	...	X_{2m}	X_{2p}
.	.	.		.	.
.	.	.		.	.
.	.	.		.	.
m	X_{m1}	X_{m2}	...	X_{mm}	X_{mp}
p	X_{p1}	X_{p2}	...	X_{pm}	X_{pp}

(b) $\text{BIAS}_i = \dfrac{X_{pi}}{X_{ip}}$ = bias by category.

22 (a) $\text{FC} = \dfrac{\sum_{i=1}^{m} X_{ii}}{X_{pp}}$ = fraction correct.

(b) $\dfrac{FC_k - FC_1}{F_1}$ = fractional improvement in FC of forecast System k over that of system 1

23 (a) $SS = \frac{NC - E}{T - E}$ = Heidke skill score where

number correct (NC) = $\sum_{i=1}^{m} X_{ii}$

$T = X_{pp}$

$E = \sum_{i=1}^{m} (X_{ip}X_{pi}) / T$

(b) $\frac{SS_k - SS_1}{SS_1}$ = fractional improvement in SS of forecast system k over that of system 1.

The following scores are based on a contingency table of only 2 categories collapsed (if necessary) from the one shown above. For instance, for score No. 24, category 1 remains and all other categories are put into category 2.

24 (a) $TS = \frac{X_{11}}{X_{11} + X_{12} + X_{21}}$

$= \frac{X_{11}}{X_{pp} - X_{22}}$

$= \frac{X_{11}}{X_{p1} + X_{1p} - X_{11}}$ = threat score for category 1 (also called Critical Success Index, CSI).

(b) $\frac{TS_k - TS_1}{TS_1}$ = fractional improvement in TS of forecast system k over that of system 1.

(c) $\frac{X_{11}}{X_{1p}} = \frac{X_{11}}{X_{11} - X_{12}}$ = probability of detection of category 1.

(d) $FAR = \frac{X_{21}}{X_{p1}} = 1 - \frac{X_{11}}{X_{p1}} = \frac{X_{11}}{X_{11} - X_{21}}$ = false alarm ratio of category 1.

25 Same as for No. 24 except computed on lower 2 categories.

26 Same as for No. 24 except computed on upper category.

27 Same as for No. 24 except computed on upper 2 categories.

28 Same as for No. 24 except computed on upper 3 categories.

29 See scores 12 and 13 in Score Group 1.

Any time the table cannot be collapsed, the score will not be computed. For instance, score 28 cannot be computed on a three-category table.

Score Group 3

This score group is for wind direction. Each "system" has two forecast inputs, a direction in degrees and a speed in that order, and there is a verifying grid consisting of a direction and speed in that order. The units to be specified on in Record Types 17, 18, and 19 would normally be in kt, although the inputs are not necessarily in those units. U855 will convert from m/s to kts with the correct IDs.

Score No.	Explanation
31	$F_i = \sum n_i$ = count of direction errors in each of 9 categories. $RF_i = F_i/N$ = relative frequency of errors in each of 9 categories. $CRF_i = \sum_{j=1}^{i} RF_{ij}$ = cumulative relative frequency of errors in each of 9 categories. $IMP_i = \frac{CRF_i - CRF_1}{CRF_1}$ = improvement in CRF over 1st system for for each of 9 categories.
32	Same as Score No. 31, except only for cases when the observed speed in $\geq$ 10 kt.
33	Same as Score No. 31, except only for cases when both the observed and forecast speed in $\geq$ 10 kt. Note that this can create an unmatched sample for the different systems, because each is based on its own speed forecast.

Score Group 4

This score group is for a probability forecast of a single event. The observation "verifying" the forecast can be continuous, and will then have a threshold associated with it which defines the "event" to which the probability relates. Or it could be a binary event, taking the values of only 0 and 1, in which case no threshold would be required.

Score No.	Explanation
41	(a) $P = \frac{2}{N}\sum(f_i - o_i)^2$ = P-score.
	(b) $\frac{P_1 - P_k}{P_1}$ = fractional improvement of P-score.
42	(a) $BS = \frac{P}{2}$ = Brier Score.
	(b) $\frac{BS_1 - BS_k}{BS_1}$ = fractional improvement of Brier-score.
43	(a) $\frac{1}{N}\sum o_i$ = Event relative frequency.
	(b) $\frac{1}{N}\sum f_i$ = Mean forecast.
	(c) Frequency of event in 10% forecast probability categories.
	(d) Relative frequency of event in 10% forecast probability categories.
	(e) Mean forecast probability in 10% forecast probability categories.
44	Same as 41 except forecasts truncated to 0 to 1 range.
45	Same as 42 except forecasts truncated to 0 to 1 range.
46	Same as 43 except forecasts truncated to 0 to 1 range.

Score Group 5

This score group is for probability forecasts of a set of m "ordered" events. The observation "verifying" the forecasts can be continuous, and will then have thresholds associated with it which defines the "events" to which the probabilities relate.

Score No.	Explanation
51	(a) $P = \frac{1}{N} \sum \sum_{j=1}^{m} (f_{ij} - o_{ij})^2$ = P-score.
	(b) $\frac{P_1 - P_k}{P_1}$ = fractional improvement of P-score.
52	Same as 51 except forecasts truncated to 0 to 1 range.
53	(a) Relative frequency of observations in each of m categories.
	(b) Mean probability forecast in each of m categories.
	(c) Frequency of observations in 10% forecast probability intervals for each of m categories.
	(d) Same as (c) except relative frequency.
	(e) Same as (c) except mean forecast probability.
54	(a) $RPS = \frac{1}{N} \sum \sum_{k=1}^{m} (\sum_{j=1}^{k} f_{ij} - \sum_{j=1}^{k} o_{ij})^2$
	(f_{ij} are truncated to range 0 to 1 and the next category in sequence adjusted by the truncated amount. As an example, if f_{i1}= -.05 and f_{i2} = .15, f_{i1} is used as 0 and f_{i2} is used as .10.)
	(b) $\frac{RPS_1 - RPS_k}{RPS_1}$ = fractional improvement of RPS.

Score Group 6

This score group is for the frequencies of differences between forecasts and observations. Each forecast system has m ordered discrete categories specified by thresholds in the IDs; the matching (single) observation is continuous. That is, the forecast IDs specify the categories which are applied after the differences are computed.

Score No.	Explanation
61	Frequencies and/or relative frequencies of algebraic errors.

Score Group 7

See Score Group 1.

APPENDIX III

Program Structure

Many of the main programs in the MOS-2000 system have similar structure. However, subtle differences in the access and use of data make some subroutines unique, although very similar to others. For U855, separate subroutine writeups are provided only for those routines that may have a use outside U855. Rather, the structure is shown below with a brief explanation of the routines.

```
DRU855 - Driver.
   U855 - Main program.
      INT855 -  Initialization.
         IPOPEN -  Opens the IP( ) files as necessary.
         IPRINT -  Prints IP( ) numbers.
         TIMPR  -  Labels start of run on all IP( ) files.
         RDSNAM -  Called several times.  Reads unit number and file name for
                   input/output, and opens file according to call sequence.
         RDI    -  Reads date list.
         DATPRO -  Prepares full date list by accommodating date spanning.
            UPDAT - Adds hours to date/time to get a new date/time.
         RDAREA -  Reads area (WFO) list.
            RDC   - Reads the station list as characters.
         RDVR86 -   Reads the definition of scores, output packages, and
                    variables to be used.
            RDSCR - Reads score group numbers, output package numbers, and
                    score numbers to output.
            RD86  - Reads the variable IDs.
            PRSID - Forms the 4-word ID( ), its parsed parts in IDPARS( ), and
                     thresholds from the "raw" input.
            THSET - Sets upper and lower thresholds from the sequence of
                     thresholds read with the variables, according to the value
                     of B = IDPARS(3).
            CALCAT - Calculates the number of categories in a set of IDs.
                      Checks for same number of categories for each system.
         IERX   -  Error routine used only if there is an I/O type error.  Will
                   define the statement number where the error occurred.
      RDAMAP -  Reads area map (correspondence between WFO and WMO Subcenter
                numbers.
         FREAD -   System routine to read non-Fortran file.
      GVPREP -  Prepares to put gridded data into vector format (grouped by
                WFO) for verification
      RDST55 -  Reads gridpoint data from all sources for the needed date/times
                for the first case, unpacks the data for these date/times, and
                stores them into the MOS-2000 internal storage system.
         UPDAT  - Adds hours to date/time to get a new date/time.
         UNPKBG - Unpacks a string of bits into an integer word.
         SWITCH - Switches files (closes one, opens another)
         UNPACK - The general MOS-2000 unpacking routine.  All data stored in
                  the MOS-2000 storage system have to be unpacked and matched
                  with the station list, because the station list may change
                  on a particular input.
```

```
          UNPKBG -  Unpacks a string of bits into an integer word.
          UNPKLX -  Unpacks a portion of the packed data.
          PNPKOO -  Unpacks data when there are no missing values.
          UNPKPO -  Unpacks data when there can be only primary missing
                   values.
          UNPKPS -  Unpacks data when where can be both primary and secondary
                   missing values.
       GRCOMD - Saves and monitors grid characteristics in NGRIDC( , )
       GSTORE - Stores the data into the MOS-2000 internal storage system.
          WRDISK -  Writes disk records as necessary.
    VRBL55 -  Prepares gridpoint data for verification.
       GLOOK - Determines whether or not a variable is in internal storage
               without returning the data
       GFETCH - Fetches data from internal storage
       OPTGRD - Switcher for calling subroutines
          GKTOF  - Changes units from Kelvin to fahrenheit
          GMTKT  - Changes units from m/s to kt
          GUVTDR - Computes wind direction from u and v
          GDIFF  - Computes difference of two variables
       UPDAT -  Adds hours to date/time to get a new date/time.
       GRIDB -  Makes a grid binary
       SMTH5 -  Smooths over 5 gridpoints
       SMTH9 -  Smooths over 9 gridpoints
       SMTH25 - Smooths over 25 gridpoints
       SMTH2X - Smooths over 25 gridpoints twice
       SMTH3X - Smooths over 25 gridpoints 3 times
       PREDX1 - Gridprints field
       GRDVEC - Puts gridpoint data into vector format
       GSTORE - Stores data into internal storage.
    SCOPXA -  Switcher to accumulant routines.  Also reads and stores the
              accumulants in a scratch file.
       FILLXA - Fills the AA matrix with all needed data for the score group
                being processed.
          BINFUL -  Makes categorical binaries.
          BINARY -  Makes cumulative binaries.
      SCX01A -  Computes accumulants for the case being processed for Score
                Group 1.
          FIXIT -  Rounds as necessary.
      SCX21A -  Computes accumulants for the case being processed for Score
                Group 2.
      SCX31A -  Computes accumulants for the case being processed for Score
                Group 3.
         FIXIT - Rounds as necessary.
         BKCAT - Puts variable into discrete categories and assigns a
                 category number to each category.
       SCX41A - Computes accumulants for the case being processed for Score
                Group 4.
          FIXIT - Rounds as necessary.
          BKCAT - Puts variable into discrete categories and assigns a
                  category number to each category.
       SCX51A - Computes accumulants for the case being processed for Score
                Group 5.
          FIXIT - Rounds as necessary.
      BKCAT - Puts variable into discrete categories and assigns a
              category number to each category.
   SCX61A - Computes accumulants for the case being processed for Score
            Group 6.
   SCX71A - Computes accumulants for the case being processed for Score
            Group 7.
LMSTR8 - Prepares list of variables needed for subsequent dates.
   UPDAT  - Adds hours to date/time to get a new date/time.
GCPAC  - Compresses entries to be kept in the MOS-2000 internal storage
         system.
   RDDISK - Reads the disk records from the MOS-2000 internal storage
            system when necessary.
   WRDISK - Writes disk records into the MOS-2000 internal storage
            system as necessary.
RDST56 - To read and store all variables needed for each case after the
         first in the MOS-2000 internal storage system.
   UPDAT  - Adds hours to date/time to get a new date/time.
   UNPKBG - Unpacks a string of bits into an integer word.
   SWITCH - Switches files (closes one, opens another)
   UNPACK - Unpacks data.
   GRCOMD - Stores and monitors grid characteristics in NGRIDC( , )
   GSTORE - Stores data into internal storage
VRBL56 - Prepares gridpoint data for verification for every case after
         the first
   GLOOK - Determines whether or not a variable is in internal storage
           without returning the data
   GFETCH - Fetches data from internal storage
   OPTGRD - Switcher for calling subroutines
      GKTOF  - Changes units from Kelvin to fahrenheit
      GMSTKT - Changes units from m/s to kt
      GUVTDR - Computes wind direction from u and v
      GDIFF  - Computes difference of two variables
   GRIDB  - Makes a grid binary
   SMTH5  - Smooths over 5 gridpoints
   SMTH9  - Smooths over 9 gridpoints
   SMTH25 - Smooths over 25 gridpoints
   SMTH2X - Smooths over 25 gridpoints twice
   SMTH3X - Smooths over 25 gridpoints 3 times
   PREDX1 - Gridprints field
   GRDVEC - Puts gridpoint data into vector format
   UPDAT - Adds hours to date/time to get a new date/time.
   GSTORE - Stores data into internal storage.
SCOPXA - See above.  This entry is used for all cases after the first.
LMSTR3 - Zeros out all entries in the MOS-2000 internal storage system
         that will not be needed on the next case.
GCPAC  - Compresses entries to be kept in the MOS-2000 internal storage
         system.
SCOPXB - Switcher for computing scores from accumulants.
   SCX01B - Computes scores for the case being processed for Score
            Groups 1 and 7.
   SCX21B - Computes scores for the case being processed for Score
            Group 2.
   SCX31B -  Computes scores for the case being processed for Score
             Group 3.
   SCX41B -  Computes scores for the case being processed for Score
             Group 4.
   SCX51B -  Computes scores for the case being processed for Score
             Group 5.
   SCX61B -  Computes scores for the case being processed for Score
             Group 6.
SCOPXC -  Switcher for printing scores.  Sometimes, the accumulants are
          also printed.
   SCR01C -  Prints scores for output package 01.
      SCR02C - Prints scores for output package 02.  This is called from
               SCR01C when a line cannot hold all scores.
   SCR02C -  Prints scores for output package 02.
   SCR03C -  Prints scores for output package 03.
      SCR01X - Assists in forming formats for printing.
   SCR04C -  Prints scores for output package 04.
   SCR21C -  Prints scores for output package 21.
      SCR25C - Assists in printing.
   SCR22C - Prints scores for output package 22.
      SCR25C - Assists in printing.
   SCR31C -  Prints scores for output package 31.
   SCR32C -  Prints scores for output package 32.
   SCR41C -  Prints scores for output package 41.
   SCR42C -  Prints scores for output package 42.
   SCR51C -  Prints scores for output package 51.
   SCR61C -  Prints scores for output package 61.
   SCR62C -  Prints scores for output package 62.
   SCX71C -  Writes scores for output package 71.
      PRSID1 - Parses 4-word into 15 component parts.  Ignores pro-
               cessing information in Word 4.
      PACKG - Packs and writes a grid
         UNPKBG -Unpacks a string of bits into an integer word
         PACK2D -Packs grid
            PACKXX - Helps in packing
            PACKYY - Helps in packing
            PACK - Does the heavy packing
               PKMS00 - Packs when there are no missing data
                  PACKGP - Determines groups for packing
               PKMS99 - Packs when there may be missing data
                  PKMS97 - Packs when there are both primary and
                           Secondary missing values
                     PACKGP - Determines groups for packing
                  PACKGP - Determines groups for packing
               PKBG - Packs a string of bits into an integer word
               PKS4LX - Part of the packing
               PKC4LX - Part of the packing
         WRITEP - Writes grid
```

U900

MAKES OPERATIONAL FORECASTS FROM REGRESSION EQUATIONS

Harry R. Glahn
July 1, 1999

PURPOSE: U900 uses one or more sets of regression equations to make forecasts from a variety of MOS-2000 vector inputs, and writes for printing and packs and writes vector output for the stations and variables designated by control files and the regression equations. Because the packed input data are self describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in random access files; these data are also packed in the TDLPACK format. The first record in each input dataset is the "station" (or location) directory (usually) containing station call letters. U900 uses a driver DRU900 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U900 (actually, subroutine) for every application. U900 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. U900 performs about the same functions as U700 but is designed for operational use. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U900.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) Output Control

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU900.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U900" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U900 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U900HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U900 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are

different (except for IP(1)). Values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU900 DATA statement = 12. Changing the default unit number allows multiple runs of U900 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = Not used.
(7) = The predictand list, including the parsed IDs, but not including the plain language.
(8) = Not used.
(9) = The list of predictands, including the plain language as determined from the variable constant file on unit KFILCP (see below). It also includes the packing constant for each variable, taken from the variable constant file, that will be used in packing the forecasts in TDLPACK format.
(10) = The variable ID's for each day/cycle as read from the sequential input. This is just a list of all the variables on the input files.
(11) = Not used.
(12) = The list(s) of stations on the input file(s). Note that any station list encountered will be output; especially for hourly data, this would be voluminous.
(13) = The stations for which forecasts are to be made that do not have equations.
(14) = A diagnostic when an equation set is being read for which the cycle time and the beginning and ending date/times do not match the date being processed.
(15) = Lists the stations with the equations that are not in the list to make forecasts for. If no input is provided on unit KFILD (see Record Type 9), then all stations are used, and there would be no print here.
(16) = The input data for each predictor in the equations for each date/time. This would generate voluminous output

except for a checkout run of only a few date/times and sets of equations.

(17) = The forecast for each predictand for each date/time. This would generate voluminous output except for a checkout run of only a few date/times and sets of equations.

(18) = The forecast for each predictand for each date/time <u>to the accuracy packed</u>. This would generate voluminous output except for a checkout run of only a few date/times.

(19) = Multiple correlation coefficient for each predictand for each equation.

(20) = Means for each predictand for each equation.

(21) = --The predictands as read from the equations.
--The stations with the equations.
--The stations to make forecasts for with the following equations.
--The equations (the ID's and coefficients).

(22) = Not used.

(23) = Information concerning opening and closing of files.

For checkout, it may be advisable to set all these values to the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) <u>Run Identification</u>

This record is used to identify the run.

<u>RUNID</u> - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (9(I10/),I10) <u>Control Parameters</u>

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

<u>KSKIP</u> - When nonzero, indicates the sequential output file on Unit KFILFC (see Record Type 8) is to be moved forward until all data for date/time KSKIP have been skipped. KSKIP is input as either YYYYMMDDHH or YYMMDDHH, and then used as YYYYMMDDHH. Note that when KSKIP > 0, there <u>must</u> be data with a directory record on the output file. Otherwise, the first read will be attempted and the program judges non-matching stations in the directory. Also, KSKIP must be $\leq$ IDATE(1) (see Record Type 5).

<u>KWRITE</u> - The existing call letters record (directory) pertaining to the data for the date/time KSKIP (see above) will be checked with the one available for writing. If they match, the new one will not be written; if they don't match, the new one will be written when KWRITE = 1, but the program will halt with a diagnostic when KWRITE = 0. Note that this has no effect when KSKIP = 0.

<u>NSKIP</u> - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

<u>JSTOP</u> - The total number of errors that will be tolerated before the program halts (see Comments section).

<u>INCCYL</u> - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO.

<u>NEW</u> - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

<u>NALPH</u> - Indicates whether (=1) or not (=0) the call letters will be alphabetized according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

<u>LOOKAH</u> - The number of hours to read ahead of the date/time being processed. This accommodates hourly data as predictors.

<u>NREPLA</u> - Record replacement flag for writing random access output:
0 = Not replacing record.
1 = Replacing record, error if record not found to replace and record is not written.
2 = Replacing record, write a new record if record not found to replace.

<u>NCHECK</u> - Identification checking flag for writing random access file:
0 = Don't check for duplicates.
1 = Check for duplicates, error if duplicate found.

<u>PXMISS</u> - The value to be used instead of 9997, if a 9997 is encountered in the input data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value. If the equations produce a 9997. it is packed as such.

Record Type 4 - Format (I3,4XA60) **<u>Date List File</u>**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

<u>KFILDT</u> - Unit number for the file containing the input date list.

<u>DATNAM</u> - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) <u>Date List</u>

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

<u>IDATE</u>(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list <u>must</u> be available or U900 stops. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be ≥ KSKIP (see Record Type 3). For operational running, it is expected that only the current date/cycle will be here.

Record Type 6 - Format (I3,4XA60) <u>Vector Input Data Files</u>

This group of records identifies the data sets from which the vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN. If all input data are from a random access constant file (see Record Type 7), only the terminator is necessary.

<u>KFILIN</u>(J) - Unit number for the data file (J=1,NUMIN).

<u>NAMIN</u>(J) - Name of file corresponding to KFILIN(J) where these data reside (J=1,NUMIN). (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model as described by "DD" in the variable ID, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard.

Record Type 7 - Format (I3,4XA60) <u>Random Access Files</u>

This group of records identifies the MOS-2000 random access data sets from which constant (and possibly other) data are read and to which the forecasts are written. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. If no data are needed from a random access file and the forecasts are not be written to a random access file, only the terminator is necessary.

<u>KFILRA</u>(J) - Unit number for the random access constant file (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

<u>RACESS</u>(J) - Name of file of corresponding to KFILRA(J). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) <u>**Sequential Forecast Output File**</u>

This record (plus the terminator record) identifies the packed forecast sequential output file. Records are read until the terminator KFILFC( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 8 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved on a sequential file, only the terminator is necessary here; in that case, a file is not opened and data are not written.

<u>KFILFC</u> - Unit number for the output forecast file.

<u>FORNAM</u> - Name of file where this output is to reside. (CHARACTER*60)

Record Type 9 - Format (I3,4XA60) <u>**Station and Location Files**</u>

This pair of records (plus the terminator record) identifies the files which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

<u>KFILD</u>(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

<u>DIRNAM</u>(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 10 - Format (14(A8,1X)) <u>**Station List**</u>

This record identifies the stations for which forecasts are desired. If KFILD(1) ≠ KFILDI, this record is omitted, and the information is taken from another source. If this data set is empty (only the terminator exists), then forecasts will be made for all stations stored with the equations. <u>Not providing a list is not usually a feasible option in U900.</u> If a list is provided, the output order can be controlled. Note that whatever the unit number in KFILD(1), at least the terminator must be present.

When forecasts are to be made in all stations with the equations, then a particular station could be included in more than one equation in a set (because the stations are stored with the equations) and forecasts could be made for that station from each equation. However, if the list of stations is provided here <u>and</u> a station exists with the equations more than once in a set, it is not known which equation to use. In this case, the first equation read that has that station with it is used.

Note that there is only one list of stations carried in U900 to make forecasts for; it can be provided from an input list or can be an amalgamation of the stations with the equations. In the latter case, there can be only one set of equations, which is equivalent to there being only one data set provided in Record Type 11 below.

<u>CCALL</u>(K) - Call letters (or other 8-character location designator) of stations (or locations) for which forecasts are desired (K=1,NSTA). This list is read within subroutine RDSTAD or RDSTAL by RDC, which eliminates any blanks found in the input. Duplicate stations in the list are kept, but a diagnostic is furnished on unit IP(5). For NALPH = 1, the stations are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

Record Type 11 - Format (I3,4XA60) <u>**Equation Files**</u>

These records (plus the terminator record) identify the files from which the regression equations are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = ND11. This Record Type 11 is read by subroutine RDSNAM. Upon completion of reading J=1,KGP.

<u>KFILEQ</u>(J) - Unit number for a set of equations (J=1,KGP).

<u>EQNNAM</u>(J) - Name of file corresponding to KFILP (J=1,KGP). When KFILP = KFILDI, EQNNAM is not used and can be read as "DEFAULT"; this would be a very unusual circumstance. (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) <u>**Predictand/Forecast Table File**</u>

This record (plus the terminator record) identifies the file from which the "forecast" ID for each "predictand" ID is read. Records are read until the terminator KFILPF = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

<u>KFILPF</u> - Unit number for the predictand/forecast correspondence file.

<u>PFCORR</u> - Name of file corresponding to KFILPF. (CHARACTER*60)

Record Type 13 - Format (I3,4XA60) <u>**Variable Constants File**</u>

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 13 is read by subroutine RDSNAM.

<u>KFILCP</u> - Unit number for the constants.

<u>CONNAM</u> - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 14 <u>Equations</u>

These records contain the equations in the exact form output by U600 and described in TDL Office Note MOS-2000, Chapter 15, except the file name has been replaced with the cycle time and the beginning and ending date/times to which the equations apply (see Chapter 15). The equations are arranged:

(1) Cycle and beginning and ending dates - Format (1XI4,5XI4,1XI4)
(2) NTAND, number of predictands in this set - Format (' 'I4)
(3) ID(J,N) (J=1,4), ICAT(N), (N=1,NTAND), the predictand ID's - Format (' 'I9.9,2I10,I11,I8)
(4) CCALL(K), stations (call letters) to which the equations following apply - Format (14(' 'A8))
(5) The equations (See TDL Office Note MOS-2000, Chapter 15).

This file will contain one "set" of equations (or actually more, see below), set being defined as one or more equations having the same predictands. The file name, number of predictands, and predictand ID's will occur only once on the file; the sequence of stations and equations can be repeated until a blank call letters record is read [(4) above]. For this to occur correctly, the file must end with '99999999'. That is, a call letters record with only the terminator present. U600 produces a "blank" call letters record for this purpose. So, the output of U600 can be directly input into U900 with no change, except for the file name (see above).

Another file can be read for another "set" of equations. Another physical file can have a different unit number KFILEQ( ) and file name EQNNAM( ). However, since U900 does not rewind the file after reading, the second set can actually be on the same file, but the complete sequence of input must be repeated, including the cycle and date/times [(1) above]. Note that the unit number should be the same for the multiple sets of equations on the same file. If more than one entry has the same unit number, they must be sequential.

<u>CONTROL FILE INPUT</u>: (Name read from U900.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) <u>Date List</u>

<u>When the dates are not provided in file U900.CN</u>, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU900), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U900 stops. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be ≥ KSKIP (see Record Type 3). For operational running, it is expected that only the current date/cycle will be here.

CONTROL FILE INPUT: (Name read from U900.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided in file U900.CN, this record identifies the stations (or locations) for which forecasts are desired. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI, or alternatively, for an empty set, forecasts are made for all stations with the equations. (Also see discussion for Record Type 10 above.)

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which forecasts are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 10, control file 'U900.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U900.CN) (Unit = KFILD(2))

Record Type 1 - Format **Station Locations**
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted by subroutine RDSTAL or RDSTAD. When this station directory is alphabetical, the final list of stations can be alphabetical no matter the order read in (see NALPH in Record Type 3). (Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3). Only the station names and their substitute stations are used from this file. See Chapter 10, Station Directory, in TDL Office Note MOS-2000 for more information.

CONTROL FILE INPUT: (Name read from U900.CN) (Unit = KFILEQ)

Record Type 1 Equations

When the equations to use in the run are not in file U900.CN, this group of records contains the equations from which to make forecasts. The maximum number of files (sets of forecasts) is ND11. The format of each set is exactly in the format output by U600 except for file name. Each set must end with the terminator '99999999'. See Record Type 14 above and TDL Office Note MOS-2000, Chapter 15, for more information.

CONTROL FILE INPUT: (Name read from U900.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) Variable Constants

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U900 users, and must be a separate file. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables in the equations for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the forecasts for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer.

Other processing occurs with the reading of these records in SETPLN, much of it associated with the plain language description. See Chapter 4, Variable Identification, for details. If an IDTEMP( ) is not found that matches a particular ID(,), then the default for PLAIN( ) is blanks and ISCALD is set = 0, except for a binary, ISCALD is set = 3. However, see MCCCFFF below.

<u>CONTROL FILE INPUT</u>: (Name read from U700.CN) (Unit = KFILPF)

Record Type 1 - Format (I6,2XI6,I4) <u>Predictand/Forecast Correspondence</u>

This group of records contains the correspondence between the predictand ID (the data on which the equations were developed) and the forecast ID. That is, the forecast for, say, temperature has a different ID than does the temperature observation. Only the CCCFFF need be changed.

ICCCFFF - The CCCFFF of the predictand ID.

MCCCFFF - The CCCFFF of the corresponding forecast ID.

ICAT - Postprocessing indicator for ICCCFFF. For instance, ICAT = 5 would switch to postprocessor subroutine CAT5. This is primarily used for inflation (ICAT = 1) which has to be done in U900 (in operations or U700 in development) if at all because to compute it requires the predictand mean and correlation, which are not carried forward into other programs. When ICAT = 0, no postprocessing is done in U700.

Note that when the MCCCFFF is known, the plain language and packing constant are taken from the variable constants file (unit KFILCP) for MCCCFFF rather than the predictand ID in ID(,). If a match is not found, the default ISCALE = 0 is used, except for a binary, ISCALE is set = 3.

<u>DATA INPUT</u>:

All data input to U900 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format. All such vector data must be preceded by a call letters (directory) record. The sequential files can have multiple directory records, each applying to the data until an end of file or another directory is encountered. Each random access file has only one directory record which applies to all data in the file. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U900.CN in Record Types 6 and 7.

A. <u>SEQUENTIAL VECTOR DATA</u>

One or more sources of vector data, each probably prepared by U201, (J=1,NUMIN) are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U900.CN', Record Type 6. Each file is closed when an EOF is reached, and if the next data set, as read in, has the same unit number, it is opened. Note that operational running of U900 will not allow multiple files with the same unit number; this should not pose a problem because normally only one date/time will be used on a run.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

B. RANDOM ACCESS VECTOR DATA

Up to 5 sources of data on random access files are accommodated; however, it unlikely more than 2 or 3 will be needed. These data exist or are written to the MOS-2000 External Random Access File System. They are accessed or written with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

DATA OUTPUT

There are three forms of data output, besides the diagnostics and other information on Units KFILDO and IP( ).

A. ASCII FOR VIEWING OR PRINTING

Input data for each date/time for the variables in the equations will be written to the file on Unit IP(16) when IP(16) > 0. Forecasts for each date/time will be written to the file on unit IP(17) when IP(17) > 0. Forecasts for each date/time to the accuracy packed (see B. below) will be written to the file on unit IP(18) when IP(18) > 0.

Note that the volume of output will be such that this print capability should only be used for a small sample.

B. SEQUENTIAL BINARY MOS-2000 FORMAT

Forecasts for all variables for all NDATES cycles will be packed in TDLPACK format and written sequentially to unit number KFILFC to the dataset whose name has been provided to FORNAM (see Record Type 8), unless KFILFC = 0, in which case data are not output. The data are packed with subroutine PACK1D and its associated subroutines.

C. RANDOM ACCESS BINARY MOS-2000 FORMAT

Forecasts for all variables for all NDATES cycles will be packed in TDLPACK format and written to random access unit number KFILX = 49 to the dataset whose name has been provided to CFILX (see Record Type 7). If a unit number 49 is not provided, the data are not output. The data are packed with subroutine PACK1D and its associated subroutines.

EXAMPLE CONTROL FILE: 'U900.CN'

An example exists as file 'U900.CN' in directory home21.glahn.dru900 on blizzard. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U900.CN', Record Type 1 and the definition of KFILDO in the driver DRU900. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The CRAY does not allow this. It is best to insert before the "D" in columns 1-5 the characters, "C****". This way, the possible optional statements can be spotted and be made operative very easily.

The binary indicator "B" in the first word of an predictor ID can be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), or 5 (indicating a grid binary). Note that a predictor "B" cannot be 2 (indicating a cumulative variable from below) or 3 (indicating a discrete variable); however, these are possible for predictand definition. For the latter, the upper threshold is the one provided with the variable and the lower threshold must be assumed or determined by the previous equation. These predictand binary definitions play no role in U900, but are needed for definition of the forecasts in the binary output.

It not necessary for each variable needed to actually be available for Day 1 for the variable to be used on subsequent cycles. However, in U900 it would be rare to be running for multiple cycles.

As indicated in the description for input Record Type 10, multiple forecasts for a station can be made for a set of equations if the station list with the equations is used, but only one forecast can be made if the station list is furnished in Record Type 10 or from a separate station list file. However, a list of stations to make forecasts for is mandated in operational running of U900, because records are written as equations are read and evaluated. If a list were not provided, the lists of stations with the equations would not match, and the run would abort.

Each equation set will start with a cycle and a beginning and ending date/time to which the equations apply. Once such a cycle and times are encountered that do not bracket the date/time being processed, that file will not be further used for that date, even though it might contain other equations.

While U900 will operate on more than one date/time, it is quite inefficient, because all equations will be read again for each subsequent date as processed. Also, the random access file to which the forecasts would be written has no way of identifying, outside of the data record itself, the date/time for which the forecasts are valid, so writing data for more than one day to random access is not practical. However, sequential output could be produced that would be usable.

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

U900 would likely not need Unit No. 46. Since U900 can write to a random access file, 49 could be used. Input forecasts could be accessed on Unit No. 48 or 49. Note that U900 could be used for copying certain forecasts from Unit No. 48 to Unit No. 49.

COMMENTS

U900 was written for operational use on the NCEP IBM computer, so it must conform to NCEP mandates. U700 performs very similar functions in a development environment, and could run on the NCEP IBM or an HP workstation. U700 was written to deal with a few equations for many days/cycles, whereas U900 was written to deal with many equations for a few (usually one) cycles. Therefore, the internal structure of U700 and U900 are quite different, although the user input is very similar. It is expected that U900 will be preceded by at least one U201 run that will provide the predictors for U900, although OPTX is available for use for switching to computational routines.

Each source of sequential vector data has a directory record associated with it which pertains to the vectors following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U900 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 3). Presumably, the only time 9997 will occur on input is when forecasts are input (it would be unusual for forecasts to be input to U900, but it is possible). Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. At the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

When a point binary is needed, it will be taken from the input if possible; if not provided, it will be computed from the "basic" variable.

The sequential data input(s) can contain constant data taken from the MOS-2000 External Random Access File System by U201. U201 can access several constant data sets if necessary. Alternatively, U900 can take constant data directly from MOS-2000 External Random Access files; all such access is through the subroutine CONST. It is possible, but would be highly unlikely, that no data on sequential files be used, and all input furnished be on random access files. It is possible that the IDs of such data have "B" = 0 with a non-zero threshold, indicating these are relative frequencies computed with that threshold. See "Restrictions" for unit numbers and associated uses.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities.

Because of the way RDSTAL and RDSTAD operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers of (old) call letters are furnished in the Record Type 9 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

Although a primary purpose of U900 is to provide packed vector forecasts for other programs in a random access file, writing of such data is not done if a unit number = 49 is not provided (see Record Type 8). That is, no output unit number and file name have been provided. This feature can be used for checkout.

Note that the variable KWRITE enables the user to control whether or not a new station directory record is written to the sequential file, if the one existing on the output just prior to the data to be written does not match the one available to be written. If the user expects them to be the same, then KWRITE = 0 can be used to halt the program with a listing of the directory records. If the user is willing to accept a different directory, then KWRITE = 1 can be used. Also note that KWRITE has no effect unless KSKIP > 0; when KSKIP = 0, the call letters record is always written.

When KFILX = 49 (random access output desired), the station list will be checked with the directory on the file if one exists. If they do not match, forecasts will not be written. (The random access file does not accommodate multiple directory records.) If no directory exists, it will be written.

When equations are developed, the predictands are usually identified as hourly observations or some variable calculated from them. The forecasts of these variables must have a different ID. The file whose unit number and name are read in Record Type 12 contains this correspondence. The plain language, etc. for those "forecast" IDs are on the file whose unit number and name are read in Record Type 13.

SETTING UP THE DRIVER DRU900

The preparation of the driver for a particular U900 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) (see ND5). It also has to be $\geq$ BLOCK, the size of a physical record size in the MOS-2000 Internal Storage System; DRU900 assures this.
ND2 -	Maximum number of terms in an equation.
ND3 -	Maximum number of predictands per equation set.
ND5 -	Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.
ND6 -	Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc.
ND7 -	The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for each day/cycle, ND9 must be large enough to hold all records for each date/time on all input files.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although operating system caching may alleviate that); too large a number will result in wasted memory and possible excess paging.

ND11 - The maximum number of sets of equations. This is the same as the maximum number of files containing equations.

ND13 - The maximum number of equations per set. For single station equations, this would be the same as ND1.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU900 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND3). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

WRITING NEW SUBROUTINES

It is not expected that computational routines will be used to any great extent in U900. Rather, computations, except possibly for point binaries, are expected to be done in U201 and in a postprocessing program U910. Any computations that would be done in U900 would have to be done on point data (not gridpoint data) because gridpoint data are not accommodated in U900. Even so, an "option" subroutine, OPTX, is provided for possible use. It is also through OPTX that the "constant" data are accessed from the external random access file. (In the case of development for gridpoints, the gridpoints are given identifications in the same way as other locations, such as stations, and the data are treated as "vector" after leaving U201.)

NONSYSTEM ROUTINES USED

For the present, use the load line in home21.glahn.dru900, dataset 'u900.com', which includes the libraries 'home21.glahn.u900lib' and 'home21.glahn.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

<u>LOCATION:</u> u900lib. The driver is in dru900.

U910

POSTPROCESSES FORECASTS ON RANDOM ACCESS FILES

Harry R. Glahn
August 10, 1999

PURPOSE: The primary purpose of U910 is to postprocess MOS-2000 forecasts residing on random access file(s), usually produced by U900. The postprocessing done is specified by the variable ID, and U910 switches to a postprocessing subroutine via OPTX to perform the calculations. Constant and other data can also be provided in random access files; these data are also packed in the TDLPACK format. The first record in each random access dataset is the "station" (or location) directory (usually) containing station call letters. U910 uses a driver DRU910 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U910 (actually, subroutine) for every application. U910 is written to run on a 32-bit or a 64-bit word-length machine. This is done through PARAMETER statements in the driver. It is possible to access data for variables at a date/time after the date/time being processed; this is accomplished with the variable "ITAU( )" and is called the "lookahead" feature. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U910.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) **Output Control**

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU910.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U910" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U910 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'HRG2' and IP(2) = 40, the file name for Unit No. 40 = 'U910HRG240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information will be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U910 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are

different (except for IP(1)). 12 values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU910 DATA statement = 12. Changing the default unit number allows multiple runs of U910 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = The variable IDs as they are being read in. This is good for checkout; for routine operation, IP(7), IP(8), and/or IP(9), may be better.
(7) = The variable ID list in summary form. If there are errors, the variable list will be written to the default output file unit KFILDO as well as to unit IP(7).
(8) = The variable ID list in summary form. This list includes the parsed ID's in IDPARS(,). (IDPARS(,) contains the 15 components of each ID.)
(9) = The variable list in summary form. This differs from the print in IP(8) in that IP(9) does not include the parsed ID's in IDPARS(,), but rather includes the information taken from the variable constant file on unit KFILCP (see below).
(10) = Not used.
(11) = Not used.
(12) = The list(s) of stations on the input file(s).
(13) = Not used.
(14) = Not used.
(15) = Data written in the order packed for each variable indicated by JP(3,) > 0. This is separate from the optional writing associated with JP(2,).
(16) = The data will be written for those variables for which JP(2,) > 0 (see NPRINT in Record Type 3 and Record Type 12 below) in the order dealt with in U910 (see IP(4)).

For checkout, it may be advisable to set all these values to the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) **Run Identification**

This record is used to identify the run.

RUNID - 72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (7(I10/),F10.0/(I10)) **Control Parameters**

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

KSKIP - When nonzero, indicates the output file on unit KFILIO (see Record Type 7) is to be moved forward until all data for date/time KSKIP have been skipped. KSKIP is input as either YYYYMMDDHH or YYMMDDHH, and then used as YYYYMMDDHH. Note that when KSKIP > 0, there must be data with a directory record on the output file. Otherwise, the first read will be attempted and the program judges non-matching stations in the directory. Also, KSKIP must be < IDATE(1) (see Record Type 5).

KWRITE - The existing call letters record (directory) pertaining to the data for the date/time KSKIP (see above) will be checked with the one available for writing. If they match, the new one will not be written; if they don't match, the new one will be written when KWRITE = 1, but the program will halt with a diagnostic when KWRITE = 0. Note that this has no effect when KSKIP = 0.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO. Because of the lookahead feature required for predictands, and to maintain efficiency, date/times should not be closer together than INCCYL. That is, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. This should pose no hardship; normally there will be only one date provided.

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized by group according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

NREPLA - Indicates whether (=0) the record written to the random access file is not replacing an existing record, or (=1) the record is to replace an existing one, and the record is not written if one cannot be found to replace, or (=2) the record is to replace an existing one if it exists, but is written in any case.

NPRINT - The number of cycles of data to write for printing to unit IP(16) under the format control and JP(2,) provided with each variable (see Record Type 12).

ICHARS - The number of characters of call letters to print when printing is indicated by JP(2,). This is constrained to be between 4 and 8 inclusive.

LNGTH - The line length for printing to unit IP(16). For a line printer, 132 is appropriate; for a laser printer, 80 may be desirable.

Record Type 4 - Format (I3,4XA60) **Date List File**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

KFILDT - Unit number for the file containing the input date list.

DATNAM - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) **Date List**

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U910 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8. IDATE(1) must be > KSKIP (see Record Type 3).

It is likely there will be only one date here

Record Type 6 - Format (I3,4XA60) **Random Access Files**

This group of records identifies the random access data sets from (to) which forecasts and constants are read (written). Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 6 is read by subroutine RDSNAM.

KFILRA(J) - Unit number for the random access constant file (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

RACESS(J) - Name of file of random access data corresponding to KFILRA(J) (J=1,NUMRA). (CHARACTER*60)

Record Type 7 - Format (I3,4XA60) **Sequential Vector Output File**

This record (plus the terminator record) identifies the packed sequential vector output file. Records are read until the terminator KFILIO( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 7 is read by subroutine RDSNAM. This file will be opened as 'NEW'. If packed data are not to be saved on a sequential file, only the terminator is necessary here; in that case, a file is not opened and data are not written. (See JP(1,), Record Type 12.)

KFILIO - Unit number for the vector output file.

OUTNAM - Name of file where this output is to reside. (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) **Station and Location Information**

This pair of records (plus the terminator record) identifies the file(s) which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. This Record Type 8 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2). KFILD(1) can be the input file number, KFILDI, in which case DIRNAM(1) is not used.

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). When KFILD( ) = KFILDI, DIRNAM( ) is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 9 - Format (14(A8,1X)) **Station List**

This group of records identifies the (groups of) stations for which data are to be processed. If KFILD(1) ≠ KFILDI, this group is omitted, and the information is taken from another source.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which equations are desired (K=1,NSTA). This list is read within subroutine RDSTAD by RDC, which eliminates any blanks found in the input. Duplicate stations in the list for a group are kept, but a diagnostic is furnished on unit IP(5). Note that this diagnostic applies only to duplicates within a group, not from group to group. For NALPH = 1, the stations in each group are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list in each group. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right. That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator of a group of stations (perhaps comprising a "region") is '99999999'. An empty set terminates the station input. That is, the last group and its terminator must be followed by another terminator signifying an empty set. Maximum number of stations, sans terminator, is ND1. (CHARACTER*8)

It would be rare for U910 to use more than one group of stations.

Record Type 10 - Format (I3,4XA60) **Variable File**

This record (plus the terminator record) identifies the file from which the variable ID's are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 10 is read by subroutine RDSNAM.

KFILP - Unit number for the variable ID's.

PRENAM - Name of file corresponding to KFILP. When KFILP = KFILDI, PRENAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 11 - Format (I3,4XA60) **Variable Constants File**

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. [Variable constants include plain language description and the decimal scaling factor for packing, when writing binary output (see record Types 6 and 7)]. Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

Record Type 12 - Format **Variable List**
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X3I2,I3,8XA1,I2,1XI1,1X30A1)

This group of records contains the variable ID's. When KFILP ≠ KFILDI, this group is omitted, and the variable list is taken from another source. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 12 is read by subroutine RDVR79. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201.

<u>ID</u>(J,N) - The first 3 (J=1,3) words of the variable ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only), (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

<u>JP</u>(J,N) - JP(J,N) indicates for J=1,3:
1 = Whether (>0) or not (=0) variable N will be packed and written to unit KFILIO and the random access file. For sequential writing to occur, a file number and name must be provided in Record Type 7. For random access writing to occur, a unit number of 49 must have been provided in Record Type 6.
2 = Whether (>0) or not (=0) variable N will be written to unit IP(16) with the format provided. This is the primary output for viewing or printing (see NPRINT, Record Type 3).
3 = Whether (>0) or not (=0) variable N will be written to unit IP(15) to the resolution packed. This is primarily for checkout and quality control of data being written.

<u>ITAU</u>(N) - The number of hours to add to NDATE (the date being processed, see Record Type 5) to get the variable N. That is, the tau in the ID is left intact, and ITAU is used to "look ahead."

<u>CFMT</u>(N) - The format descriptor for writing on unit IP(16) when JP(2,N) > 0. Must be either "F" or "I".

<u>IWDTH</u>(N)- The field width for writing on unit IP(16) when JP(2,N) > 0. If IWDTH( ) is > 30, it will be set to 30.

<u>IPREC</u>(N)- The precision for writing on unit IP(16) when IP(2,N) > 0. For an "F" format, this is the number of digits after the decimal point. For an "I" format, it is the number of digits always written. Note that for a "zero" to be printed, IPREC( ) must be > 0; otherwise, zero will be printed as a blank.

<u>HEAD</u>(J,N) - The column heading for writing to Unit IP(16) when JP(2,N) > 0. Limited to J=1,30 characters. Only IWDTH(N) characters are read. HEAD(,) is right justified when writing.

CONTROL FILE INPUT: (Name read from U910.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) **Date List**

When the dates are not provided in file U910.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU910), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list must be available or U910 stops. Date/times should not be closer together than INCCYL. For example, if the first date/time is Jan. 1, 1996, and INCCYL = 12, a time of 06 UTC should never be indicated. Maximum number of dates, sans terminator, is ND8.

Normally, there would be only one date here.

CONTROL FILE INPUT: (Name read from U910.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List**

When the station list is not provided in file U910.CN, this group of records identifies the stations (or locations) to be included in the regression analysis. It is not needed when KFILD(1) = KFILDI; in this case, the call letters are read from the KFILDI.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which interpolated values are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 9, control file 'U910.CN', unit KFILDI for additional information.) (CHARACTER*8)

CONTROL FILE INPUT: (Name read from U910.CN) (Unit = KFILD(2))

Record Type 1 - Format **Station Locations**
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which interpolated output is desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted. When this station directory is alphabetical, the final list of stations can be alphabetical within each group no matter the order read in (see NALPH in Record

Type 3). [Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. Note that alphabetization is not overall, but by group (see Record Type 9)]. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Most of this information is used only within subroutine RDSTGA or RDSTGN to give information about the stations. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3).

<u>CCALLD</u>(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

<u>NAME</u>(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

<u>NELEV</u>(K) - Elevation of the station. Format is I5.

<u>SIGNLA</u> - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

<u>XLAT</u> - Latitude in degrees. Format is F7.4.

<u>SIGNLO</u> - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

<u>LONDD</u> - Longitude in degrees west. Format is F8.4.

<u>CCALLD</u>(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).
<u>IWBAN</u>(K) - The WBAN number of the station. Format is I5)

The number of stations in this directory is not limited. No terminator is used.

CONTROL FILE INPUT: (Name read from U910.CN) (Unit = KFILP)

Record Type 1 - Format **Variable List**
(I9,1XI9,1XI9,1XI3,1XA1,1XI4,1XI3,4X4I2,9XA1,I2,1XI1,1X30A1)

When the variables to use in the run are not in file U910.CN, this group of records contains the variable ID's. When KFILP = KFILDI, this file is omitted. Records are read until the terminator ID(1,) = 999999 is reached. Maximum number of records, sans the terminator record, = ND4. This Record Type 1 is read by subroutine RDVR79. Upon completion of reading this record type, N=1,NVRBL. Note that the format and information for ID(,) are exactly the same as for reading variables in U201. See Control File Input for U910.CN, Record Type 12, unit KFILDI above for more detailed information; the format is exactly the same.

CONTROL FILE INPUT: (Name read from U910.CN) (Unit = KFILCP)

Record Type 1 - Format (3(I9,1X),I3,2XA32,I3) **Variable Constants**

This group of records contains information about the variables, as defined by ID(J,) (read previously) and IDTEMP(J). This file should be universally useable by all U910 users, and is expected to be a separate file; that is, while KFILD(2) could = KFILDI, it would be unusual for that to be the case. Note that the format matches that for a file input to U201; U201 uses additional information in the records in the file.

IDTEMP(1) - First word of variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=1,3) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALD - This is the decimal scale factor to use when packing the data for writing. That is, each datum is multiplied by 10^{ISCALD}, then rounded for packing the value as an integer. If a variable being processed cannot be found in this file, the default SCALE = 0 is used.

Even when a match is found, the rest of the ID's in ID(1,) are checked because there might be more than one match.

Other processing occurs with the reading of these records in RDVR79, much of it associated with the plain language description. See Chapter 4, Variable Identification, for details. If an IDTEMP( ) is not found that matches a particular ID(,), then the default for PLAIN( ) is blanks and ISCALD is set = 0, except for a binary forecast, ISCALD is set = 3.

<u>DATA INPUT</u>:

All data input to U910 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000) from the random access MOS-2000 External File System. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U910.CN in Record Type 6.

A. <u>RANDOM ACCESS VECTOR DATA</u>

Up to 5 sources of data are accommodated; however, it is unlikely more than 1 or 2 will be needed. These data exist in the MOS-2000 External Random Access File System. These data are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

<u>DATA OUTPUT</u>

There are three forms of data output, besides the diagnostics and other information on units KFILDO and IP( ).

A. <u>ASCII FOR VIEWING OR PRINTING</u>

A maximum of NPRINT (see Record Type 3) or NDATES (see Record Type 5) <u>cycles</u> of data will be written to the file on unit IP(16) under control of the format provided with each variable N, the variables written controlled by JP(2,N) (see Record Type 12). The data columns are headed by HEAD(,N) (see Record Type 12). Listing is by station group. If a line length for printing is not sufficient for all variables, then more than one line is used. Line length is specified as LNGTH characters in Record Type 3.

When JP(3,N) > 0, the data will also be written to unit IP(17) <u>to the resolution packed</u>; this is for quality control and would not be used for large samples of data.

B. <u>SEQUENTIAL BINARY MOS-2000 FORMAT</u>

Data for each variable N for which JP(1,N) > 0 (see Record Type 12) for all NDATES cycles will be packed in TDLPACK format and written to unit number KFILIO to the dataset whose name has been provided to OUTNAM (see Record Type 7), unless KFILIO = 0, in which case data are not output. The data are packed with subroutine PACK1D and its associated subroutines.

c. RANDOM ACCESS BINARY MOS-2000 FORMAT

Data for each variable N for which JP(1,N) > 0 (see Record Type 12) for all NDATES cycles will be packed in TDLPACK format and written to unit number KFILX = 49, (only) if a file with that number has been provided in Record Type 6. The data are packed with subroutine PACK1D and its associated subroutines.

EXAMPLE CONTROL FILE: 'U910.CN'

An example exists as file 'U910.CN' in directory home21.tdllib.dru910 on blizzard. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U910.CN', Record Type 1 and the definition of KFILDO in the driver DRU910. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

HP workstations accommodate optionally compiled statements with a "D" in Column 1. The CRAY does not allow this. It is best to replace the "D" with, for instance, "C****" in Columns 1-5. This way, the possible optional statements can be spotted and be made operative very easily.

The binary indicator "B" in the first word of an ID can be either 0 (indicating a continuous variable), 1 (indicating a cumulative binary from above), 5 (indicating a grid binary), 2 (indicating a cumulative variable from below), or 3 (indicating a discrete variable). For the latter, the upper threshold is the one provided with the variable and the lower threshold is set to the upper value of the next lower discrete binary with the same ID's. If there is no lower one, the lower threshold is automatically set to -99999. Also, when this is the upper discrete binary, and the threshold is input as either 9999 (i.e., .9999E04) or 99990 (i.e., .9999E05; only 4 significant places are provided for), it is automatically set to 99999.

Since the variables are not ordered (as they are in U600), if a discrete binary is desired (one with both upper and lower thresholds), these variable ID's must be input in sequence and in the correct order (lower threshold first). To make sure all is well, check the thresholds on unit

IP(8) or IP(9). Ordering of variables is important in U910 when a "series" of variables is needed for postprocessing (e.g., normalizing a set of probabilities).

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write

U910 would likely not need Unit No. 46. For U910 to write to a random access file, a unit number 49 must be provided. If it is desired to not write to the same file as input, use Unit No. 48 for reading.

COMMENTS

The .CN control file is of nearly the same format as that for U660, from which U910 was derived. The only difference in the driver, DRU910, is that U910 uses ND2 and U660 does not. In some cases U910 needs to deal with and return a set (or series) of variables (usually, if not always, a set of probabilities). For instance, a set of probabilities are to be normalized. ND2 is the maximum number of variables in such a series. For instance, for four cloud amount categories, ND2 must be $\geq$ 4.

Although the primary purpose of U910 is to provide postprocessed and packed vector data on a random access file for use by other programs, writing of such data is not done unless a unit number 49 is provided (see Record Type 6). Also, no sequential output is provided when KFILIO = 0 (see Record Type 7). That is, no output unit number and file name have been provided. This feature can be used for checkout. Also, the binary output of each variable is controlled by JP(1,) (see Record Type 12). Because of this, the user can print for viewing the variables used in computation, and both print and write the postprocessed variables.

If a variable is identified in Record Type 12 that is available on an input random access file, then the data for that variable will be "copied" (for only the stations in the U910 station list) to the output file and/or printed. If the variable is not available on the input file, it will be computed in a routine through subroutine OPTX, provided an appropriate subroutine is available to be called by OPTX and the switching is provided in OPTX. Because these computation routines are used by both U710 (which has sequential input) and U910 (which has random access input), the data are returned in the computational subroutines through a subroutine RETVEC. RETVEC first looks in the internal random access file with GFETCH; if it is not there, it looks in the external random access file with CONST.

Each source of vector data has a directory record associated with it. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW ≠ 1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U910 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 arises out of the evaluation of a set of forecast equations where there was insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted in each computational routine as having the value of (very near) zero, 9999, or (however unlikely) some other value. Presumably, the only time 9997 will occur is when operational forecasts are input from u900 or U700 produces test forecasts. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such.

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. At the end of Day 1 and at the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

All computations are expected to be done in subroutines, even to making a binary; that is, there is no binary capability in the basic U910 program.

U910 can take constant data directly from the MOS-2000 External Random Access files; all such access is through the subroutine CONST. It is possible that the IDs of such data have "B" = 0 with a non-zero threshold, indicating these are relative frequencies computed with that threshold.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities.

Because of the way RDSTGA and RDSTGN operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers or (old) call letters are furnished in the Record Type 9 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

While groups of stations can be used (see Record Type 9), it is expected that only one group will be used. Retaining multiple group capability from other MOS-2000 programs allowed certain subroutines to be reused. The list of stations used must match the list on Unit No. 49 if writing is being done and one exists. It is expected that all forecasts need to be put onto one file, so reading and writing would both be from Unit No. 49. The station list for U910 can be from the same source as used for the creation of the random access file (e.g., U350 and U352) and/or for making the forecasts (U900).

Note that if forecasts for a subset of stations were needed, possibly for a subset of forecast variables, on a random access file or/and a sequential file, U910 could be used to "copy" those forecasts. Just use the appropriate station list and the variables to copy.

Note also, that JP(1,) (see Record Type 12) controls both the writing to sequential and random access files. However, the file unit number must be appropriately provided for writing to occur.

While the capability to write a sequential file has been retained from other similar MOS-2000 programs, U910 will not process but one date because the input random access records do not distinguish dates. If <u>only</u> constant data were to be input (e.g., relative frequencies), then multiple dates could be processed. This would allow "copying" relative frequencies for a subset of stations to another random access file.

<u>SETTING UP THE DRIVER DRU910</u>

The preparation of the driver for a particular U910 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the directory (read on unit KFILD(2)) unless, of course, the station directory is to be used as the station list (see ND5).
ND2 -	Maximum number of variables that can be dealt with in a series. This is mainly for probability forecasts.
ND3 -	Not used.
ND4 -	The maximum number of variables that can be dealt with.
ND5 -	Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.
ND6 -	Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc., even though the same unit number might be used.

ND7 - The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.

ND8 - The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.

ND9 - The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records needed for the date/time of Day 1 on all input files, including the predictands at future date/times.

ND10 - The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. This can be quite small in U910, because it is used only to hold the indices for the station locations for the random access file(s).

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU910 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND4). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

WRITING NEW SUBROUTINES

Computational routines are used to do the postprocessing. An "option" subroutine, OPTX, is provided for switching to the appropriate routine. Each postprocessing routine will characteristically have the following structure:

1. A table listing the IDs of the postprocessed variables that can be handled in the routine, together with the corresponding IDs of the variable(s) needed for the processing.

2. Composition of the IDs of the variable(s) needed for the particular variable being processed in the call, and the fetching of the data from either the Internal or External Storage System by subroutine RETVEC.

3. Computation of the variable(s).

Routines NMLPRB and TRUNCP are available in the U910 library for patterns; they contain error checking, etc. NCAT is not used unless a series of variables is to be returned by the routine.

NONSYSTEM ROUTINES USED

Use the load line in home21.tdllib.dru910, dataset 'u910.com', which includes the libraries 'home21.tdllib.u910lib' and 'home21.tdllib.moslib' in that order, all on blizzard.

LANGUAGE: FORTRAN77 with some HP extensions.

LOCATION: u910lib. The driver is in dru910.

U911

INVENTORIES OPERATIONAL REGRESSION EQUATION FILES

David E. Rudack
August 15, 2005

PURPOSE: U911 inventories operational regression equation files and outputs user-specified equations and forecasts for viewing. U911 will also output all the unique predictor values associated with a specified equation. In addition, the user has the option of sending to output each equation's term-by-term forecast calculation for a user-specified date. This allows the user to view the term-by-term convergence of each equation to its corresponding forecast value for that date. The user may also set an acceptable tolerance range for a predictor's coefficient value. This is useful in identifying possible problematic predictors. U911 has the capability of processing station-based or regionalized equation files. The input data sets that contain predictor information are derived from a variety of MOS-2000 vector inputs. Because the packed input data are self-describing, the data can come from various sources, the number being essentially unlimited. Constant data can be provided in random access files. U911 uses a driver DRU911 so that dimensions of variables can be tailored by PARAMETER statements to user need without requiring a separate copy of the main program U911 (actually, subroutine) for every application. U911 is written to run on a 32-bit or a 64-bit word-length machine. This is accomplished by PARAMETER statements in the driver. Some familiarity with other MOS-2000 documents will be necessary for full understanding of this writeup.

CONTROL FILE INPUT: 'U911.CN' (Unit = KFILDI)

Record Type 1 - Format (A4,25I3) **Output Control**

This record contains unit numbers for the run, and can even control the "default" output file number. KFILDI is the input unit number as specified in DRU911.

IPINIT - 4 characters, usually a user's initials plus a run number, to append to "U911" to identify a particular segment of output indicated by a suffix IP(J) (see below). The run number allows multiple runs of U911 and writing of uniquely named files, provided the user uses a different run number for each run. For example, with IPINIT = 'DER2' and IP(2) = 40, the file name for Unit No. 40 = 'U911DER240'. DO NOT USE A BLANK FOR ONE OF THE CHARACTERS. (CHARACTER*4)

IP(J) - Each value (J=1,25) indicates whether (>0) or not (=0) certain information is to be written. When IP( ) > 0, the value indicates the unit number for output. These values should not be the same as any other unit numbers used in U911 except possibly KFILDO (the default output file), although a value of one IP( ) can be the same as the value of another IP( ). This is ASCII output, generally for diagnostic purposes. This capability essentially allows separation of diagnostic and other information in almost any way desired. However, to help assure that the user sees important diagnostic information, it may be output on the default output file in addition to IP( ) when they are different (except for IP(1)). Values have been defined as indicated for values of J below:

(1) = All error diagnostics plus other information not specifically identified with other IP( ) numbers. When IP(1) is read as nonzero, KFILDO, the default output file unit number, will be set to IP(1). When IP(1) is read as zero, KFILDO will be used unchanged, as specified in DRU911 DATA statement = 12. Changing the default unit number allows multiple runs of U911 or other programs within the same directory without overwriting.
(2) = The input dates in IDATE( ). These are the dates as actually read in. When there are errors, print will be to the default output file unit KFILDO as well as to unit IP(2).
(3) = The output dates in IDATE( ). These are the dates extended by date spanning. When there are errors, output will be to the default output file unit KFILDO as well as to unit IP(3).
(4) = The station list (call letters only). If there are input errors, the station list will be written to the default output file unit KFILDO as well as to unit IP(4).
(5) = The station directory information. If there are input errors in this list, the station list will be written to the default output file unit KFILDO as well as to unit IP(5).
(6) = Not used.
(7) = The predictand input list, including the parsed IDs, but not including the plain language.
(8) = Not used.
(9) = The list of input predictands, including the plain language as determined from the variable constant file on unit KFILCP (see below).
(10) = The variable ID's for each day/cycle as read from the sequential input. This is just a list of all the variables on the input files.
(11) = Not used.
(12) = The list(s) of stations on the input file(s). Note that any station list encountered will be output; especially for hourly data, this would be voluminous.
(13) = Not used.
(14) = A diagnostic when an equation set is being read for which the cycle time and the beginning and ending date/times do not match the date being processed.
(15) = Lists the stations that are found on the equation file for which equations and/or forecasts are not desired.
(16) = The predictor values in each equation for each date/time.
(17) = The forecast for each input predictand for each date/time.

(18) = Not used.
(19) = Indicates whether (>0) or not (=0) the equation for each input predictand and station will be printed for viewing. Also, indicates that a predictor's coefficient value lies outside the range stipulated by the user when KFILPR ≠ 0 (see Record Type 11).
(20) = Indicates whether (>0) or not (=0) the forecast summation for each input predictand and station will be printed for viewing.
(21) = Indicates whether (>0) or not (=0) the complete list of stations in the equation set will be written for viewing.
(22) = Not used.
(23) = Information concerning opening and closing of files.

For checkout, it may be advisable to set all these values to the default output file number. Later, others can be zero, and other output, if wanted, can be directed to other files.

Record Type 2 - Format (A72) **Run Identification**

This record is used to identify the run.

RUNID -72 characters of information to identify the run. (CHARACTER*72)

Record Type 3 - Format (7(I10/),F10.0) **Control Parameters**

This record contains a variety of control values for the run. Note that each value is on a separate line. A brief explanation can be put on the same line with the variable to assist the user in knowing which value is for which variable.

NSKIP - The number of errors that will be tolerated on day 1 before halting. Day 3 is usually completed before the stop actually occurs so that the user can see more results.

JSTOP - The total number of errors that will be tolerated before the program halts (see Comments section).

INCCYL - The increment in hours between date/times that are put into IDATE( ) as a result of date spanning in subroutine DATPRO.

NEW - Indicates whether (=1) the new ICAO call letters are to be used or whether (=0) the old 3-letter call letters are to be used. The directory used in MOS-2000 contains both. In either case, one is the substitute for the other, and there are up to 4 other substitute stations in the directory (see Comments section).

NALPH - Indicates whether (=1) or not (=0) the call letters will be alphabetized according to the station directory. Since the MOS-2000 directory is alphabetized by the new ICAO call letters, using NEW = 0 and NALPH = 1 doesn't make much sense.

LOOKAH - The number of hours to read ahead of the date/time being processed. This accommodates hourly data as predictors.

<u>JFCST</u> – Indicates whether (=1) or not (=0) forecasts are to be printed to IP(17).

<u>PXMISS</u> - The value to be used instead of 9997, if a 9997 is encountered in the input data. This allows maintaining a 9997, treating it as 0, as 9999, or some other value.

Record Type 4 - Format (I3,4XA60) **<u>Date List File</u>**

This record (plus the terminator record) identifies the data set from which the date list is read. Records are read until the terminator KFILDT( ) = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 4 is read by subroutine RDSNAM.

<u>KFILDT</u> - Unit number for the file containing the input date list.

<u>DATNAM</u> - Name of file where this date list resides. When KFILDT = KFILDI, DATNAM is not used and can be read as "DEFAULT". (CHARACTER*60)

Record Type 5 - Format (7I10) **<u>Date List</u>**

This group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) ≠ KFILDI, this Record Type 5 is omitted.

<u>IDATE</u>(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. This input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Data for the first date in the list <u>must</u> be available or U911 stops. Maximum number of dates, sans terminator, is ND8 which is set in the driver DRU911.

Record Type 6 - Format (I3,4XA60) **<u>Vector Input Data File(s)</u>**

This group of records identifies the data sets from which the vector data are read. Records are read until the terminator KFILIN( ) = 99 is reached. Maximum number of records, sans the terminator record, = ND6. This Record Type 6 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,NUMIN. If all input data are from a random access constant file (see Record Type 7), only the terminator is necessary.

<u>KFILIN</u>(J) - Unit number for the data file (J=1,NUMIN).

<u>NAMIN</u>(J) - Name of file corresponding to KFILIN(J) where these data reside (J=1,NUMIN). (CHARACTER*60)

Note that the model number is not used as it is in U201, but the format of the input data in Record Type 6 is maintained. An input could be from more than one model as described by "DD" in the variable ID, or the same model's data could exist on more than one data set; there is almost complete flexibility in this regard.

Record Type 7 - Format (I3,4XA60) **Random Access Input Data File(s)**

This group of records identifies the MOS-2000 random access data sets from which constant (and possibly other) data are read. Records are read until the terminator KFILRA = 99 is reached. Maximum number of records, sans the terminator record, = 5. This Record Type 7 is read by subroutine RDSNAM. If no data are needed from a random access file, only the terminator is necessary. Only random access files for point data are accommodated in u911.

KFILRA(J) - Unit number for the random access constant file (J=1,NUMRA). Unit numbers must be in the range 45 to 49; see "Restrictions" for more information.

RACESS(J) - Name of file corresponding to KFILRA(J). (CHARACTER*60)

Record Type 8 - Format (I3,4XA60) **Station List File and Station Location File**

This pair of records (plus the terminator record) identifies the files which hold station location information. Records are read until the terminator KFILD( ) = 99 is reached. Maximum number of records, sans the terminator record, = 2. KFILD(1) contains the stations for which equations/forecasts are desired. If the user desires, he/she may use the Station Location File (i.e, the Station directory) in place of a standard input station list (i.e., KFILD(1) = KFILD(2)). In this instance, the output would consist of every station in the station table. This Record Type 8 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,2.

KFILD(J) - Unit number for the file containing the station call letters for which data are to be processed (J=1) and the station directory which holds the latitudes, longitudes, WBAN numbers, elevations, and names for each possible station (J=2).

DIRNAM(J) - Name of file matching KFILD(J) (J=1,2). (CHARACTER*60)

Note that the input call letters (or other 8-character location designator of stations (or locations) for which equations/forecasts are desired are read within subroutine RDSTAD or RDSTAL by RDC, which eliminates any blanks found in the input. A diagnostic notifying the user of duplicate stations is furnished on unit IP(5). Before processing, however, duplicate stations are removed. For NALPH = 1, the stations are placed in alphabetical order providing the directory is in alphabetical order; stations not in the directory will be put at the end of the list. The call letters should (normally) be left justified, and if a full 8 characters are not present, CCALL( ) will be blank filled on the right.

That is 'OKCbbbbb' could be 'OKC' or 'OKCb'. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1.

Record Type 9 - Format (I3,4XA60) **Operational Equation File(s)**

These records (plus the terminator record) identify the files from which the operational regression equations are to be taken. Records are read until the terminator KFILP = 99 is reached. Maximum number of records, sans the terminator record, = ND11. This Record Type 9 is read by subroutine RDSNAM. Upon completion of reading this record type, J=1,KGP.

KFILEQ(J) - Unit number for a set of equations (J=1,KGP).

EQNNAM(J) - Name of file corresponding to KFILP (J=1,KGP). (CHARACTER*60)

Record Type 10 - Format (I3,4XA60) **Predictand Variable Input File**

This record (plus the terminator record) identifies the file from which the input MOS-2000 predictand ID is read. This ID identifies the equation set in which the desired equation resides. Duplicate IDs are removed prior to processing. Records are read until the terminator KFILPF = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 10 is read by subroutine RDSNAM.

KFILP - Unit number for the predictand input file.

PRENAM - Name of file corresponding to KFILP. (CHARACTER*60)

Record Type 11 - Format (I3,4XA60) **Predictor Variable Input File**

This record (plus the terminator record) identifies the file from which the input MOS-2000 equation predictor ID's are read. This Record Type 11 identifies equation predictors (in those equations indicated by Record Type 10) that are to be checked to determine if the corresponding equation coefficients lie within a user-defined stipulated range. While this Record Type 11 is optional, a terminator value of "99" must be present in the U911.CN file if this record is omitted. Duplicate IDs are removed prior to processing. Records are read until the terminator KFILPR = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 11 is read by subroutine RDSNAM.

KFILPR - Unit number for the predictor input file.

PRENAMR - Name of file corresponding to KFILPR. (CHARACTER*60)

Record Type 12 - Format (I3,4XA60) **MOS-2000 Variable Constant File**

This record (plus the terminator record) identifies the file from which the variable constants are to be taken. (Variable constants include plain language description.) Records are read until the terminator KFILCP = 99 is reached. Maximum number of records, sans the terminator record, = 1. This Record Type 12 is read by subroutine RDSNAM.

KFILCP - Unit number for the constants.

CONNAM - Name of file corresponding to KFILCP. (CHARACTER*60)

CONTROL FILE INPUT: (Name read from U911.CN) (Unit = KFILDT)

Record Type 1 - Format (7I10) **Date List**

When dates are not provided in file U911.CN, this group of records determines the date/times for which data are to be input and processed. If KFILDT (read in Record Type 4) = KFILDI (the input unit number as specified in DRU911), this file is omitted.

IDATE(J) - Initial date list, which may contain negative values indicating date spans. When a negative occurs, all dates between this value and the previous date are filled in at the increment of hours specified in INCCYL. The input date list is modified in subroutine DATPRO to contain the complete date list with the dates in the spans filled in (J=1,NDATES). Dates are input as YYMMDDHH and modified to YYYYMMDDHH. This list is read by subroutine RDI which eliminates any zeros found in the input. Terminator is 99999999. Maximum number of dates, sans terminator, is ND8.

CONTROL FILE INPUT: (Name read from U911.CN) (Unit = KFILD(1))

Record Type 1 - Format (7(A8,1X)) **Station List File**

This record identifies the stations (or locations) for which equations/forecasts are desired. Recall that the user may also use Record Type 1 (below) in place of the Station List File. If the user chooses the latter option, the format is described in the subsequent record type.

CCALL(K) - Call letters (or other 8-character location designator) of stations (or locations) for which forecasts are desired (K=1,NSTA). This list is read with subroutine RDC, which eliminates any blanks found in the input. Terminator is '99999999'. Maximum number of stations, sans terminator, is ND1. (See Record Type 8, control file 'U911.CN', unit KFILDI for additional information.)

CONTROL FILE INPUT: (Name read from U911.CN) (Unit = KFILD(2))

Record Type 1 - Format **Station Location File**
(A8,1XA4,1XA17,4XA2,8XI5,1XA1,F7.4,1XA1,F8.4,9XA8,3(1XA8),12XI5)

This group of records provides information about the stations (or locations) for which equations/forecasts are desired. The call letters read from KFILD(2) are matched with those in the station list read from KFILD(1) and the appropriate information extracted by subroutine RDSTAL or RDSTAD. When this station directory is alphabetical, the final list of stations can be alphabetical no matter the order read in (see NALPH in Record Type 3). (Although alphabetical arrangement is not essential at this step, it is highly recommended to make the output more compatible from run to run. However, the directory used in MOS-2000 is alphabetized by the new ICAO call letters, which would eliminate the possibility of alphabetizing if the old 3-letter call letters were used.) The number of stations in this directory is not limited. No terminator is used. Either the new ICAO or old 3-letter call letters can be used according to the value of NEW (see NEW in Record Type 3). Only the station names and their substitute stations are used from this file. See Chapter 10, Station Directory, in TDL Office Note MOS-2000 for more information.

<u>CCALLD</u>(K,J) - Call letters (or other character location designator) of stations (or locations) (J=1). As stated above, these call letters are matched with those in the station list. When NEW = 1, CCALLD(K,1) is read from the first field (A8) and CCALLD(K,2) is read from the second field (A4). When NEW ≠ 1, CCALLD(K,1) is read from the second field and CCALLD(K,2) is read from the first field.

<u>NAME</u>(K) - 20-character name of station. This is used for visual identification of the station in certain output. Format is A17,4XA2; this provides for a 17-character name, a blank, and a 2-character state abbreviation. Note that the last three characters in the "name" field in the directory are not used. (CHARACTER*20)

<u>NELEV</u>(K) - Elevation of the station. Format is I5.

<u>SIGNLA</u> - Sign of the latitude of the station, read as either "S" or "N". When read as "S", the latitude is set negative indicating South latitude. Format is A1.

<u>XLAT</u> - Latitude in degrees. Format is F7.4.

<u>SIGNLO</u> - Sign of the longitude of the station, read as either "E" or "W". When read as "E", the longitude is modified to make all longitudes West. That is, longitude will range from 0 through 360 and be in degrees West over the United States. Format is A1.

<u>LONDD</u> - Longitude in degrees west. Format is F8.4.

<u>CCALLD</u>(K,J) - Call letters of substitute stations (or locations) (J=3,6) (K=1,NSTA).

<u>IWBAN</u>(K) - The WBAN number of the station. Format is I5.

The number of stations in this directory is not limited, except when it also constitutes the list to be kept, in which case the number of entries is limited by ND1-1. No terminator is used.

<u>CONTROL FILE INPUT</u>: (Name read from U911.CN) (Unit = KFILEQ)

Record Type 1 **<u>Operational Equation File(s)</u>**

This group of records contains the operational equation set(s). The maximum number of files is ND11. The format of each set is exactly in the format output by U600 and U602 (see below) except for file name. Each set must end with the terminator '99999999'.

Equation File Format:

(1) Cycle and beginning and ending dates - Format (1XI4,5XI4,1XI4)
(2) NTAND, number of predictands in this set - Format (' 'I4)
(3) ID(J,N) (J=1,4), ICAT(N), (N=1,NTAND), the predictand ID's - Format (' 'I9.9,2I10,I11,I8)
(4) CCALL(K), stations (call letters) to which the equations following apply - Format (14(' 'A8))
(5) The equations (See TDL Office Note MOS-2000, Chapter 15).

<u>CONTROL FILE INPUT</u>: (Name read from U911.CN) (Unit = KFILP)

Record Type 1 - Format (3(I9,1X),I3,1X,A1,1X,I4,1XI3,4X3I2) **<u>Predictand Variable Input File</u>**

This group of records contains information about the predictands for which equations/forecasts are to be found. Records are read until the terminator ID(1,) = 999999 is reached. Records in this file (sequential by Record Type 10) are read by subroutine RDPRED. Maximum number of records, sans the terminator record, = ND4 which is set in the driver DRU911.

<u>ID</u>(J,N) - The first 3 (J=1,3) (N=1,NVRBLS) words of the input predictand ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign (either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only) (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

<u>JP</u>(J,N) - **Not used.**

<u>CONTROL FILE INPUT</u>: (Name read from U911.CN) (Unit = KFILPR)

Record Type 1 - Format(3(I9,1X),I3,1X,A1,1X,I4,1XI3,4X3I2,1X,F12.10,1X,F12.10) **<u>Predictor Variable Input File</u>**

This group of records contains information that identifies the predictor and its associated coefficient value tolerance range (for those equations desired by Record Type 10). Records are read until the terminator IDP(1,) = 999999 is reached. Records in this file (sequential by Record Type 11) are read by subroutine RDPRED9. Maximum number of records, sans the terminator record, = ND15 which is set in the driver DRU911.

<u>IDP</u>(J,N) - The first 3 (J=1,3) (N=1,NVRBLSP) words of the input predictor ID plus the last 3 digits of the 4th word, followed in order by the components of a threshold value consisting of (1) sign

(either minus, or plus or blank for plus, read as A1), (2) 4 digits to follow a decimal point, and (3) 3 digits representing the power of 10 by which to multiply the decimal value just read. For easy reading (only) (1) and (2) above can be separated by a decimal point and (2) and (3) separated by an "E". From these values, the 4th ID word (J=4) is composed.

JPP(J,N) - **Not used.**

COEFMIN(N) – Minimum predictor coefficient acceptance value. (N=1,NVRBLSP)

COEFMAX(N) – Maximum predictor coefficient acceptance value. (N=1,NVRBLSP)

CONTROL FILE INPUT: (Name read from U911.CN) (Unit = KFILCP)

Record Type 1 - Format(3(I9,1X),I3,2X,A32,I3,F11.4,F10.4,F9.2,F8.2,2XA12) **MOS-2000 Variable Constant File**

This group of records contains information about the predictand(s) and predictor variable(s) which are processed by U911. This file should be universally useable by all MOS-2000 users and is expected to be a separate file.

IDTEMP(1) - First word of the variable ID, either with or without the "B" and "DD". This is matched with all variables read for this run, both with and without the "B" and "DD". When there is a match, the constant information with IDTEMP(1) is stored as indicated below.

IDTEMP(J) - These 3 words (J=2,4) are currently not used, but are meant to correspond to the ID words 2-4.

PLAINT - When IDTEMP(1) matches an ID(1,N), PLAINT is stored in PLAIN(N). These 32 characters are used for visual identification of variables in certain output. Although 32 characters are allowed, the first 5 are reserved for a height indicator (e.g., 1000-), and those after character 23 may be overwritten for vertical or time processing. This generally leaves 18 characters besides height, smoothing, and other processing indicators. (CHARACTER*32)

ISCALT - When IDTEMP(1) matches an ID(1,N), ISCALT is stored in ISCALD(N).

SMULTT - When IDTEMP(1) matches an ID(1,N), SMULTT is stored in SMULT(N). This variable is the multiplicative factor (power of 10) to use when gridprinting the gridpoint data. **(Not used)**

SADDT - When IDTEMP(1) matches an ID(1,N), SADDT is stored SADD(N). This variable is the additive factor to use when gridprinting the gridpoint data. **(Not used)**

CONTT - When IDTEMP(1) matches an ID(1,N), CONTT is stored in CINT(N). This variable is the contour interval to use when gridprinting the data. It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT. **(Not used)**

ORIGNT - When IDTEMP(1) matches an ID(1,N), ORIGNT is stored in ORIGIN(N). This variable is the contour origin to use when gridprinting the data It applies to the units in which the data exist on the packed input tapes, not after manipulation by SMULTT and SADDT. **(Not used)**

UNITST - When IDTEMP(1) matches an ID(1,N), UNITST is stored in UNITST(N). These characters define the units of the data after application of SMULTT and SADDT and are used in the visual inspection of the data in gridprinted maps. **(Not used)**

DATA INPUT:

All data input to U911 will be in the MOS-2000 TDLPACK format (see "Data Record Structure" in TDL Office Note MOS-2000). Constant data are provided in the random access MOS-2000 External File System; these data are also in TDLPACK format. All such vector data must be preceded by a call letters (directory) record. The sequential files can have multiple directory records, each applying to the data until an end of file or another directory is encountered. Each random access file has only one directory record which applies to all data in the file. Reading is done with standard FORTRAN binary reads, and unpacking is done with subroutine UNPACK and its associated subroutine UNPKBG. The files for these data are specified in Control File U911.CN in Record Types 6 and 7.

A. SEQUENTIAL VECTOR DATA

One or more sources of vector data (J=1,NUMIN), each probably prepared by U201, are accommodated, the dataset names and unit numbers having been provided to NAMIN(J) and KFILIN(J), respectively, from the control file 'U911.CN', Record Type 6. Each file is closed when an EOF is reached. Note that U911 will not allow multiple files with the same unit number.

Each source (file) has a directory record at the beginning, and the data values in each record apply to the corresponding station in the directory. Multiple directory records, as might exist on an hourly data archive file, are accommodated.

B. RANDOM ACCESS VECTOR DATA

Up to 5 sources of data on random access files are accommodated; however, it unlikely more than 2 or 3 will be needed. These data exist on the MOS-2000 External Random Access Files. They are accessed with prescribed unit numbers, depending on the type of data; see "Restrictions" for more information. Since some constants may be relative frequencies, the fourth word can contain a threshold with the "B" in the first word a zero.

DATA OUTPUT:

While U911 does not furnish a binary output file, relevant information is printed to IP(16), IP(17), IP(19), and IP(20). These four IP files should supply the user with sufficient equation inventory information. Below, is an example of what the user would see if these IP( ) values were turned on.

IP(16) Output:

```
PREDICTOR DATA FOR PREDICTAND 703100015 000000000 000000017        000
----------------------------------------------------------------------

DATA FOR PREDICTOR  702000000 000000000 000000000        000 FOR DATE  2005063009
KDCA            70.00000

DATA FOR PREDICTOR  703100000 000000000 000000000        000 FOR DATE  2005063009
KDCA            67.00000

DATA FOR PREDICTOR  202020008 000000000 009000026        000 FOR DATE  2005063009
KDCA            79.30000

DATA FOR PREDICTOR  203020008 000000000 009000026        000 FOR DATE  2005063009
KDCA            72.90000

DATA FOR PREDICTOR  003301005 000000002 000000017        200 FOR DATE  2005063009
KDCA            67.20000

DATA FOR PREDICTOR  008311105 000000000 000000017  650001400 FOR DATE  2005063009
KDCA             1.00000

DATA FOR PREDICTOR  010202000 000000000 000000000        000 FOR DATE  2005063009
KDCA            -1.00000

DATA FOR PREDICTOR  004112005 000001000 000000017        200 FOR DATE  2005063009
KDCA            -5.40000

DATA FOR PREDICTOR  004221005 000000010 000000000        200 FOR DATE  2005063009
KDCA             1.50000

DATA FOR PREDICTOR  008311105 000000000 000000000  650001400 FOR DATE  2005063009
KDCA             1.00000

DATA FOR PREDICTOR  003132005 000000002 000000017        220 FOR DATE  2005063009
KDCA           332.10001

DATA FOR PREDICTOR  010203000 000000000 000000000        000 FOR DATE  2005063009
KDCA             -.05200
```

IP(17) Output:

```
 FORECASTS FOR DATE 2005063009 FOR PREDICTAND  703100015 000000000 000000017        000
 KDCA            72.30291
```

If any of the predictor values are missing, the forecast value is set to missing (9999). In the instance where an equation could not be developed because not enough cases were found for the desired predictand, the forecast value is set to a 9997.

IP(19) Output:

```
 THE FOLLOWING EQUATION IS VALID FOR THE 0900 UTC CYCLE AND SPANS THE DATES OF 0401 THROUGH 0930.

  REGRESSION EQUATION FOR  17 HOUR FORECAST OF      DEW POINT

 FOLLOWING EQUATION(S) APPLY TO THE FOLLOWING     1 STATIONS
    KDCA

 REGRESSION EQUATION                                              PREDICTAND MEAN =     57.08421
                                                          CORRELATION COEFFICIENT =       .96659
                                                                EQUATION CONSTANT =     18.38284

 VARIABLE ID                             PLAIN LANGUAGE                PROJ.(HR)  COEFFICIENT

 702000000 000000000 000000000        000        TEMPERATURE                  0      -.13140E+00
 703100000 000000000 000000000        000        DEW POINT                    0       .14461E+00
 202020008 000000000 009000026        000    AVN MOS SFC TEMP(CHK)           26       .12488E+00
 203020008 000000000 009000026        000    AVN MOS DEWPOINT(CHK)           26       .79779E+00
 003301005 000000002 000000017        200      2 DEWPOINT(Z)                 17       .20015E+00
 008311105 000000000 000000017  650001400        TOTAL SKY COVER        B     17       .48942E+00
 010202000 000000000 000000000        000        COS DAY OF YEAR              0       .14322E+01
 004112005 000001000 000000017        200   1000 ERTH GEO V-WIND(P)          17       .46545E-01
 004221005 000000010 000000000        200     10 WIND SPEED(Z)                0      -.24698E-01
 008311105 000000000 000000000  650001400        TOTAL SKY COVER        B      0      -.26865E+00
 003132005 000000002 000000017        220      2 EQUIV. POT TEMP SURF        17      -.80053E-01
 010203000 000000000 000000000        000        SIN 2*DOY                    0      -.48536E-01
```

IP(20) Output:

```
 FORECAST SUMMATION FOR PREDICTAND 703100015 000000000 000000017        000 FOR STATION: KDCA

 (NOTE THAT THE FIRST VALUE IN THE FORECAST SUMMATION IS THE PRODUCT OF THE
  FIRST COEFFCIENT VALUE AND ITS PREDICTOR VALUE ADDED TO THE EQUATION CONSTANT
  VALUE OF     18.38284.)

PREDICTOR DESCRIPTION          COEFFICIENT VALUE      PREDICTOR VALUE      RUNNING FORECAST TOTAL
---------------------          -----------------      ---------------      ----------------------
     TEMPERATURE                     -.13140             70.00000              9.18494
     DEW POINT                        .14461             67.00000             18.87351
 AVN MOS SFC TEMP(CHK)                .12488             79.30000             28.77681
 AVN MOS DEWPOINT(CHK)                .79779             72.90000             86.93537
   2 DEWPOINT(Z)                      .20015             67.20000            100.38576
     TOTAL SKY COVER        B         .48942              1.00000            100.87518
     COS DAY OF YEAR                 1.43218             -1.00000             99.44300
1000 ERTH GEO V-WIND(P)               .04654             -5.40000             99.19166
  10 WIND SPEED(Z)                   -.02470              1.50000             99.15461
     TOTAL SKY COVER        B        -.26865              1.00000             98.88596
   2 EQUIV. POT TEMP SURF            -.08005            332.10001             72.30038
```

SIN 2*DOY	-.04854	-.05200	72.30291

If one or more of the predictors is missing or an equation could not be generated because not enough cases were for the desired predictand, the forecast summation table will not be printed. U911 prints an error diagnostic notifying the user of these situations.

EXAMPLE CONTROL FILE: 'U911.CN'

An example exists as file 'U911.CN' in directory /nfsuser/g06/mos2k/u911lib/RUN on the IBM-SP. The easiest way to set up a run is to take an existing control file and modify it; DO NOT START FROM SCRATCH.

OUTPUT:

Output that can be printed can be put onto one or more of several files as described above under control file 'U911.CN', Record Type 1 and the definition of KFILDO in the driver DRU911. All errors will be to the default output file (KFILDO as possibly modified by IP(1)) as well as possibly to other files as defined by IP( ). Every effort has been made to notify the user of problems and potential problems and to proceed under user control.

RESTRICTIONS:

The information presented here as well as in the subsequent “comments” section is similar and in some cases redundant to that which is found in the U900 write-up. However, it is important to reiterate these points in the context of U911.

Most restrictions are only associated with variables in PARAMETER statements in the driver which control array sizes. In some places, machine word length is a factor, and 32-bit and 64-bit machines have been provided for by the use of PARAMETER statements, and the setting of L3264B to either 32 or 64. Formats and other guidelines in other MOS-2000 documents are followed.

When generating forecasts for more than one date/time, all input tdlpack files sharing the same type of data must have different unit numbers. This is different than other MOS-2000 runs where similar data must share the same unit number (e.g., U201).

Each equation set will start with a cycle and a beginning and ending date/time to which the equations apply. Once a cycle and time are encountered that do not bracket the date/time being processed, that file will not be further used for that date, even though it might contain other equations. The input date in the date list must overlap a date that spans the dates for which the desired equation is valid, otherwise, U911 terminates.

While U911 operates on more than one date/time, it is quite inefficient, because all equations will be read again for each subsequent date as processed.

The unit numbers for the random access files must be in the range 45 through 49, and those unit numbers are used for the following purposes related to values of CCC in ID(1):

Unit No.	CCC Range	Use
45	400-499	"True" constants (rel. freq., means, etc.)
46	500-599	1-d and 2-d constants, probably for U201
47	800-899	Thresholds for best category forecasts, etc.
48	200-299	Forecasts read only
49	200-299	Forecasts read/write (U911 will not write to an external random access file.)

COMMENTS

Each source of sequential vector data has a directory record associated with it which pertains to the vector data records following it up until another directory record is encountered. The directory indicates, in terms of station identifiers, where the datum in each record is to be found for a particular station's identifier (usually call letters). However, because station identifiers sometimes are changed and it is desired to mix data from the same location regardless of the particular identifier, provision is made for up to 5 substitute stations. When the new ICAO identifiers are being used (NEW = 1), the first substitute station is the old call letters taken from the second field in the station directory. When the old call letters are being used (NEW≠1), the first substitute station is the ICAO identifiers from the first field in the directory. The directory also contains up to 4 other stations (see the station directory documentation) that can be substituted for the station identifiers being used. Subroutine RDDIR gives additional details.

The sequential data input(s) can contain constant data taken from the MOS-2000 External Random Access File System by U201. Alternatively, U911 can take constant data directly from MOS-2000 External Random Access files; all such access is through the subroutine CONST. It is possible, but would be highly unlikely, that no data on sequential files be used, and all input furnished be on random access files. It is possible that the IDs of such data have "B" = 0 with a non-zero threshold, indicating these are relative frequencies computed with that threshold. See "Restrictions" for unit numbers and associated uses.

The input TDLPACK data can contain a primary missing value and a secondary missing value. U911 assumes that the former is the value 9999 and the latter is 9997. The 9999 indicates truly missing data, whereas the 9997 occurs in a set of forecast equations where there were insufficient data to derive an equation for a particular category element. In this latter case, the event can be interpreted as having the value of (very near) zero, 9999, or some other value as designated by PXMISS (see Record Type 3). Presumably, the only time 9997 will occur on input is when forecasts are input. Other values, such as "888" for cloud heights, can be packed as "missing" and will, of course, be returned by the unpacker as such.

The operational equation file generally contains one "set" of equations (or possibly more), set being defined as one or more equations having the same predictands. The number of predictands, and predictand ID's will occur only once on the file; the sequence of stations and equations can be repeated until a blank call letters record is read [(4) above]. For this to occur correctly, the file must end with '99999999'. That is, a call letters record with only the terminator present.

If the user desires diagnostic print for a station's equation and forecast, JFCST (Record Type 3) must be set to a value of one. If no data are found for that date, the equation and forecast will not be printed. In this instance, however, the user may still print the equation by setting JFCST to a value of zero.

If data are found for the processed date, but one or more predictor values are missing, an equation will still be output to IP(19) (even if JFCST=1). In this case, forecasts will be set to missing (9999) on IP(17) and the running forecast total on IP(20) will not be printed.

U911 does not perform any type of inflation on wind speed forecasts. The wind speed forecasts printed by U911 are directly calculated from the wind speed equation. If inflated forecasts are needed, they can be calculated by using the station wind speed forecast value, the equation's multiple correlation coefficient, and the predictand mean. These values are output on IP(17) and IP(19). The algorithm for this computation can be found in subroutine icat.f (in /nfsuser/g06/mos2k/moslib).

Most errors are output for printing starting with **** to the file with number designated by IP( ) (see Record Type 1) and a count is kept for matching with NSKIP and JSTOP (see Record Type 3). Missing variables will usually <u>not</u> be counted as an error, but a diagnostic is provided. It is not always obvious what is an error as opposed to something that might be expected to happen occasionally; therefore, the count can't be considered as absolute. When a variable cannot be found, a diagnostic will be provided (possibly "****CANNOT OBTAIN VARIABLE"). While these diagnostics could be eliminated, it is thought best to keep a watchful eye for errors. These can mostly be put on a separate file, if desired. At the end of the run, the number of "errors" and the number of missing variables (data records) are printed.

When a point binary is needed, it will be taken from the input if possible; if not provided, it will be computed from the "basic" variable.

Some information is provided for printout on each run that may seem repetitive. However, it is believed that the user should monitor this information and investigate seeming abnormalities.

Because of the way RDSTAL and RDSTAD operate, when NEW = 1, the station list used will be ICAO station identifiers whether or not the (new) ICAO identifiers or (old) call letters are furnished in the Record Type 9 list. When NEW = 0, the old call letters will be used even if ICAO identifiers are furnished in the list.

SETTING UP THE DRIVER DRU911

The preparation of the driver for a particular U911 run is relatively painless; it consists of using a template driver and modifying as necessary certain PARAMETER statements. These statements set values of:

L3264B -	Set to 32 for a 32-bit machine (e.g., the HP's) or 64 for a 64-bit machine (e.g., the CRAY).
ND1 -	Maximum number of stations (or points) that can be dealt with. Note that this does not include the number of stations in the directory (read on Unit No. KFILD(2)) (see ND5). It also has to be $\geq$ BLOCK, the size of a physical record size in the MOS-2000 Internal Storage System; DRU911 assures this.
ND2 -	Maximum number of terms in an equation.
ND3 -	Maximum number of predictands per equation set.
ND4 -	Maximum number of predictands that may be placed in KFILP for which equations will be found.
ND5 -	Maximum number of stations that can be in the directory of any input. Must be $\geq$ ND1.
ND6 -	Maximum number of all sequential file input sources that can be dealt with. If data from a model is on two files, then this would be counted as two, not one, etc.
ND7 -	The size of ISO( ), IS1(), IS2( ), and IS4( ). This would normally be 54.
ND8 -	The maximum number of date/times that can be used. This is the "extended" date list, not just the values read in.
ND9 -	The maximum number of fields (variables) stored in the MOS-2000 Internal Storage System. Since all fields are stored for Day 1, ND9 must be large enough to hold all records of the date/time of Day 1 on all input files.
ND10 -	The number of words of storage provided in the variable CORE( ) for the MOS-2000 Internal Storage System. When this is filled, a scratch disk file is used. Too small a number will result in more disk accesses than necessary (although operating system caching may alleviate that); too large a number will result in wasted memory and possible excess paging.
ND11 -	The maximum number of input equations files to be processed by U911 in one run.
ND13 -	The maximum number of equations per set. For single station equations, this would be the same as ND1.

ND14 - The maximum value of either ND3 or ND4. This is used to set the array size of those arrays passed into subroutines “rdpred.f” and “setpln.f”.

ND15 - The maximum number of predictors for which an acceptable coefficient tolerance range may be processed in one run of U911.

Do not change the computation for the variable L3264W, and probably not for NBLOCK. NBLOCK is the block size in words for disk records for the MOS-2000 Internal Storage System. (Experimentation may determine that a larger value is desirable.)

The user can see from the DRU911 template what effect each of these values has on storage from where it occurs in the DIMENSION statements. Some have relatively little effect (e.g., ND7), while others have considerable effect (e.g., ND3). Every effort has been made so that the variables will not be overflowed if values too small are used; however, be cautious.

WRITING NEW SUBROUTINES

It is not expected that computational routines will be used to any great extent in U911. Rather, computations, except possibly for point binaries, are expected to be done in U201 and in a post processing program U910. Any computations that would be done in U911 would have to be done on point data (not gridpoint data) because gridpoint data are not accommodated in U911. Even so, an "option" subroutine, OPTX, is provided for possible use. It is also through OPTX that the "constant" data are accessed from the external random access file. (In the case of development for grid-points, the gridpoints are given identifications in the same way as other locations, such as stations, and the data are treated as "vector" after leaving U201.)

NONSYSTEM ROUTINES USED

The following libraries are used to link with u911 and must be placed in the makefile:

```
/nfsuser/g06/mos2k/u201lib/libu201lib_4_blue.a
/nfsuser/g06/mos2k/moslib/libmoslib_4_blue.a
```

LANGUAGE: FORTRAN 90

LOCATION: u9xxlib. The driver is in /nfsuser/g06/mos2k/u9xxlib/RUN/dru911.