OCSMesh: End-to-End Mesh Generation

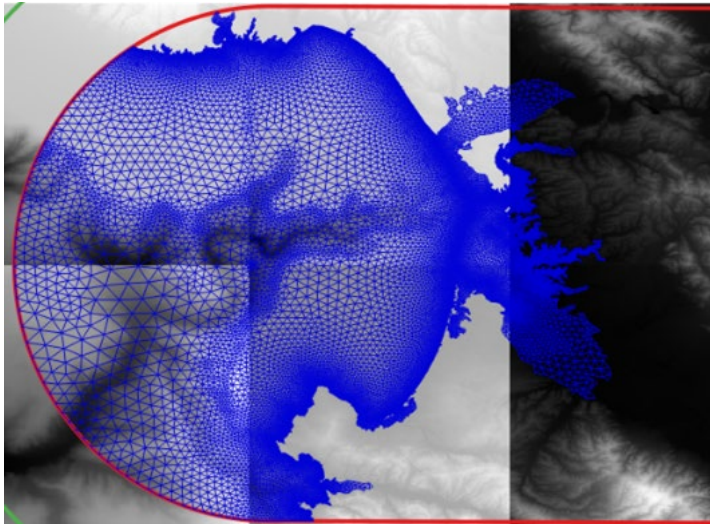
National Ocean Service

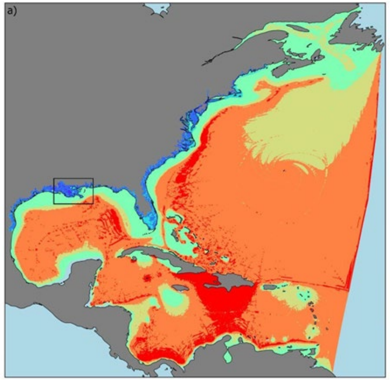
NOS Storm Surge Modeling Team (SSM)

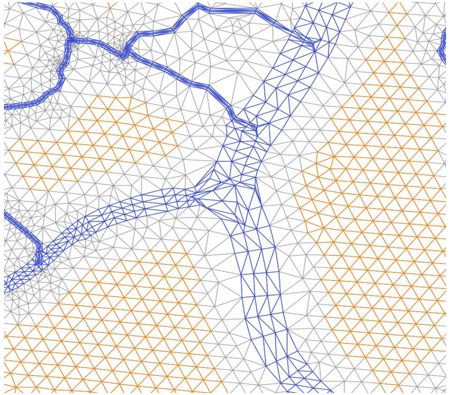
Coastal Marine Modeling Branch,

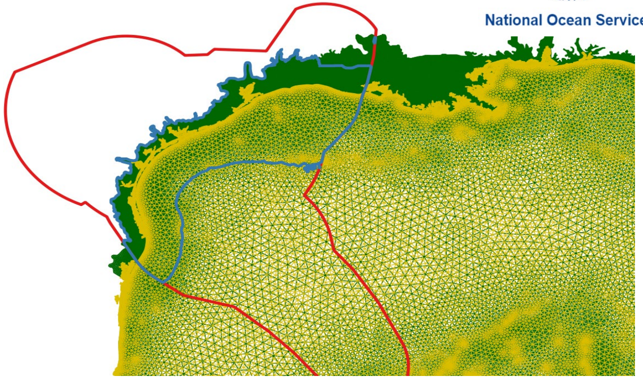
Coast Survey Development Laboratory,

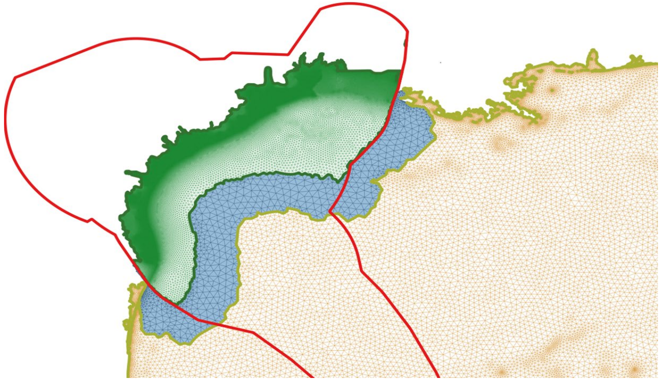
Office of Coast Survey, NOAA National Ocean Service











It takes a village to raise a child …

NOS/Office of Coast Survey Storm Surge Modeling Team

Felicio Cassalho, Soroosh Mani, Yuji Funakoshi, Lei Shi, Zizang Yang, Jack Reeves Eyre, Fariborz Daneshvar, Yunfang Sun, Atieh Alipour, Xu Chen, Edward Myers, Greg Seroka, Saeed Moghimi

Academic partners (>20 PIs, Scientists, Postdocs and PhD students)

- University of Notre Dame
- Virginia Institute of Marine Science
- Argonne National Laboratory
- National Center for Atmospheric Research
- Texas Advanced Computing Center
- Columbia River Inter-Tribal Fish Commission
- Louisiana State University
- Sandia National Laboratories
- University of Massachusetts – Dartmouth
- University of North Carolina at Chapel Hill
- Cooperative Institute for Great Lake Research
- Oregon State University

International partners

- European Commission Joint Research Centre, Belgium
- Helmholtz-Zentrum Hereon, Germany
- Laboratório Nacional de Engenharia Civil, Portugal
- International Hydrographic Organization
- United Nations

NOAA and agency partners

- NOS
  - IOOS, CO-OPS, NGS, OR&R
- NWS
  - National Hurricane Center
  - Office of Science and Technology Integration
  - Environmental Modeling Center
  - Office of Water Prediction
- OAR
  - Great Lakes Environmental Research Laboratory
  - Earth Prediction Innovation Center (EPIC)
- USGS
- EPA
- NSF
- DoS

Industrial partners

- Ocean Associates Inc
- NVIDIA
- Spatial Front Inc

NOS Storm Surge Modeling Team

Products and services

Operational

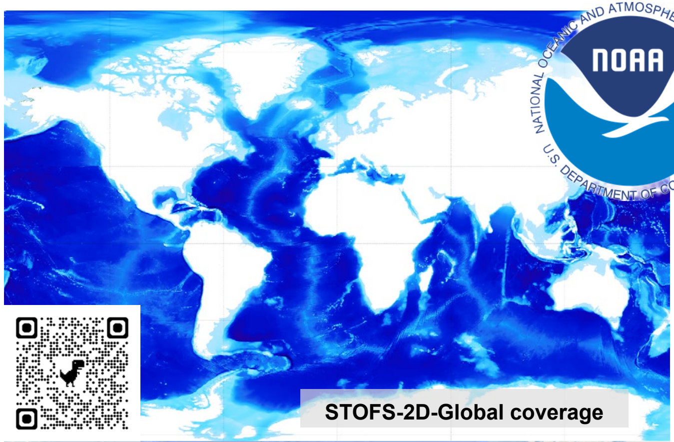
- **Surge & Tide Operational Forecast System (STOFS)**
  - Two-dimensional global ***(STOFS-2D-Global)***
  - Three-dimensional (density-layered) coastal storm surge including inland hydrology extremes ***(STOFS-3D-Atlantic)***

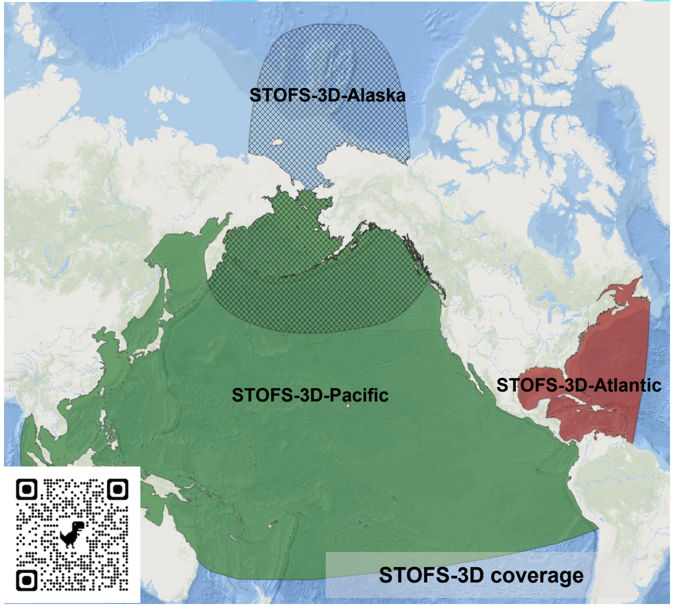
Pre-Operational

- **Surge & Tide Operational Forecast System (STOFS)**
  - Three-dimensional guidance system for Pacific Ocean *(STOFS-3D-Pacific)*

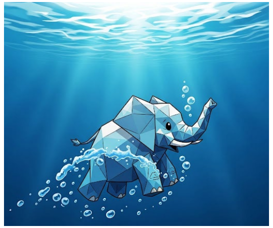
Research and development

- Three-dimensional guidance system for Alaska (STOFS-3D-Alaska)
- Development of the NOAA's Next-generation Coastal Ocean Model Coupling infrastructure (UFS-Coastal)
- Data-driven unstructured mesh generation (OCSMesh)
- On-demand probabilistic inland-coastal flooding prediction system (Storm surge / Wave / inland-hydrology)
- Deep learning based bias correction for STOFS-2D-Global
- Hazard Assessment for DoS Select At-Risk Diplomatic Posts (global support)
- Long term reanalysis (19 years STOFS-2D-Global and …)





OCSMesh

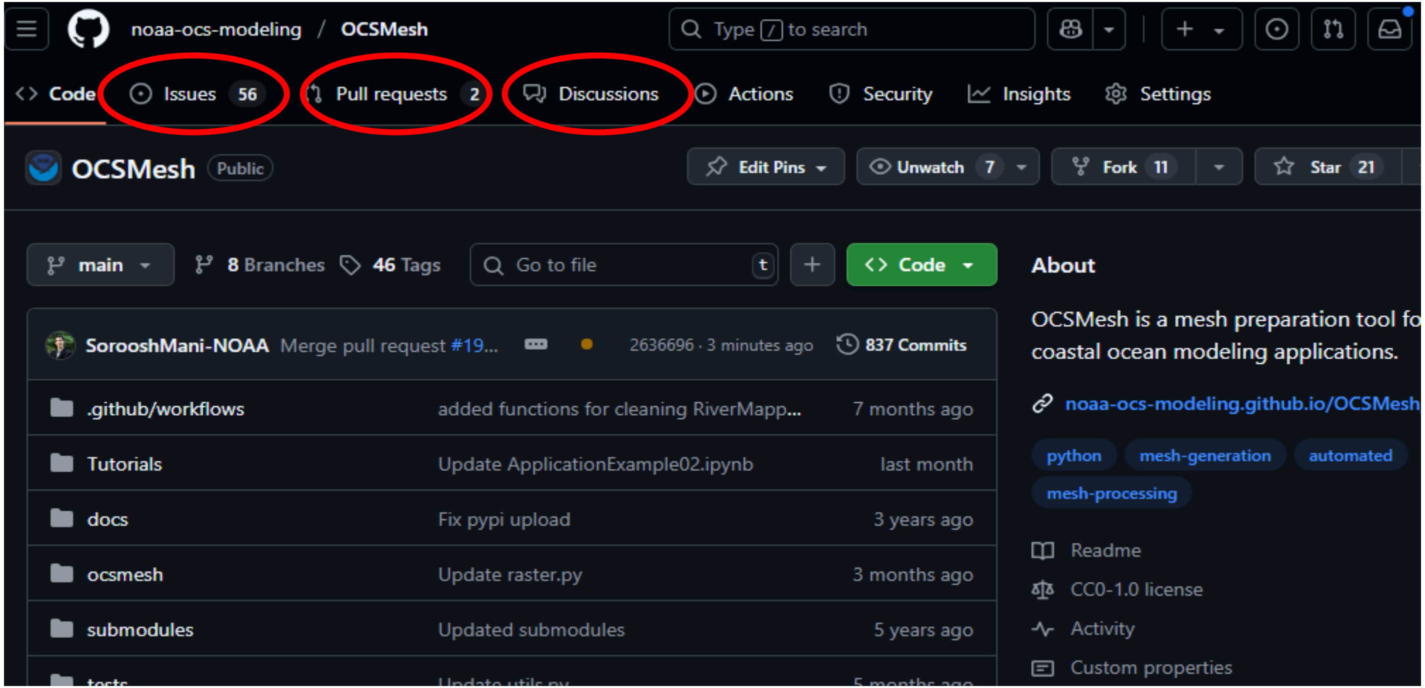


*Generated by Gemini

OCSMesh is an **open source**, **data-driven**, object-oriented **Python** toolkit that streamlines the creation of high-quality **unstructured** grids for coastal ocean models by generating new meshes **from scratch** and uniquely enabling the **subsetting** and **merging** of existing ones. The package is geared towards **SCHISM** applications, but can be utilized for **other models**.

- Based on **Jigsaw** mesh engine's Python wrapper, Jigsaw-Python (Engwirda 2014).
- Uses **Triangle** mesh engine (Shewchuk 2001) through some of its utility functions for custom workflows.
- Dedicated classes for definition of domain and element sizes based on different inputs as well as basic handling of rasters.
    - During a meshing two singletons, a final *Geom* and a final *Hfun* (sizing function), are used to define the meshing boundary and sizes
    - *Geom* represents the mesh domain that can be created via input raster, shape, or pre-existing mesh files.
    - The *Hfun* (size function) defines the element size in several ways: fixed global maximum and minimum, based on DEM slopes, and based on a detected or specified region or distance from it.

Development Driven by Community Needs*



https://github.com/noaa-ocs-modeling/OCSMesh

* Read “nerds”

To auto-mesh, or not to auto-mesh: that is the question…

Whether 'tis nobler in the mind to suffer the constraints and inaccuracies of a different sorts of grids, Or to make **simplices** against a sea of **data**, and by automating, end them. To commit; to push; No more; and by a push to say we end the **headache** and the thousand **manually defined** regions that meshing is heir to. 'Tis a consummation devoutly to be wish'd. To commit, to push; To push, perchance to simulate.

Ay, there's the rub: For in that piece of code what bugs may come, when we have shuffled off this **manual workflow**, must give us pause. There's the respect that makes the calamity of so **long iterations**. For who would bear the overlaps of shapes, the preprocessor's wrong, the proud license contumely, the pangs of dispriz'd grid, the UI's delay, the invalid ops,

And the spurns that **scientist**, of the coastlines takes, when he himself might **his domain mesh** make with a **bare Python script**?

Who would fardels bear to grunt and sweat under a weary **workflow**? But that the dread of something after automation, the undiscover'd process, from whose steps, all simulations return, puzzles the will, and makes us rather bear those ills we have than fly to others that we know not of?

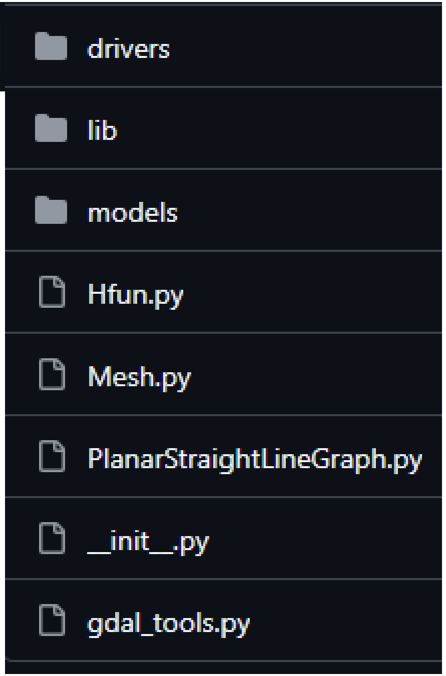
Thus **status quo** does make cowards of us all; and thus the native hue of resolution is sicklied o'er with the pale cast of **clicks**. And modeling efforts of **great skill and accuracy**, with this regard their boundaries turn awry, and lose the name of **simulation**.

Soft you now! The fair **community**! In thy unstructured meshes, be **OCSMesh** remember'd.

*Image and text generated by Gemini and then modified

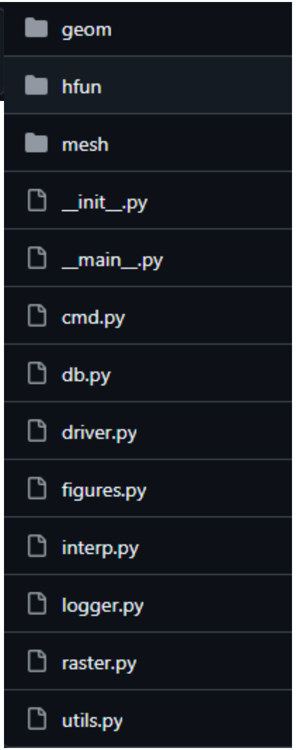
From the initial commit …

- General structure of the code established
- Handling of size functions and rasters implemented
- Handling of CRS and mesh cleanup implemented
- Different iterations and mesh engines tested



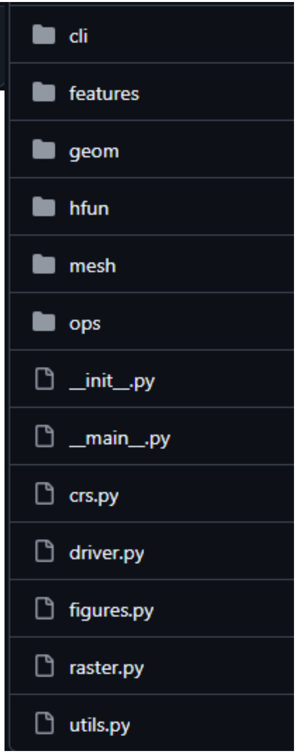
It evolved …

- Bug fixes and improvements
- First iteration of multi-DEM size function and geometry functionality
- First iteration of subsetting
- OCSMesh for large scale meshes
- OCSMesh for on-demand application

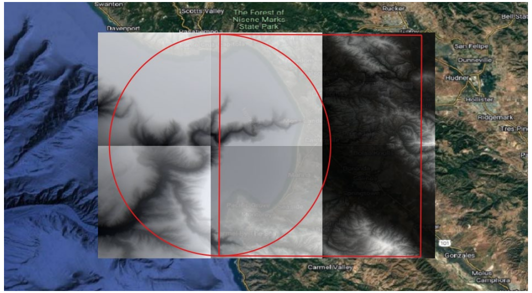


And it blossomed

- Further improvements, bugfixes and refinements
- Full fledged cross-scale capability implemented and tested
- Runtime optimizations implemented
- Quad generation
- Continuous Creek-to-Ocean meshing



Simple Workflow

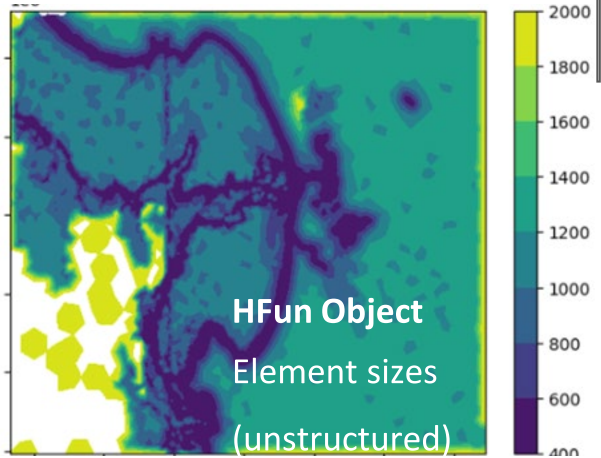


Digital Elevation Model

Data-driven

Geom Object

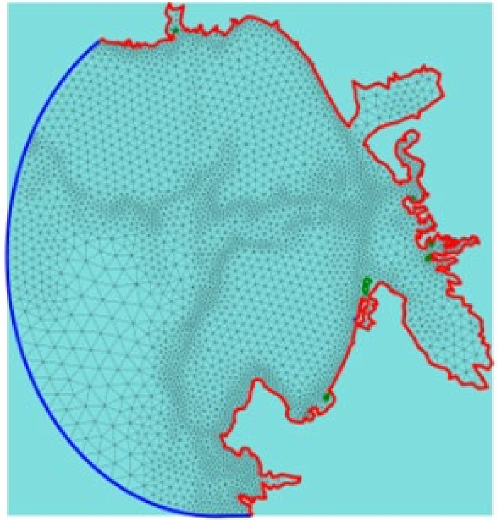
Mesh domain



HFun Object

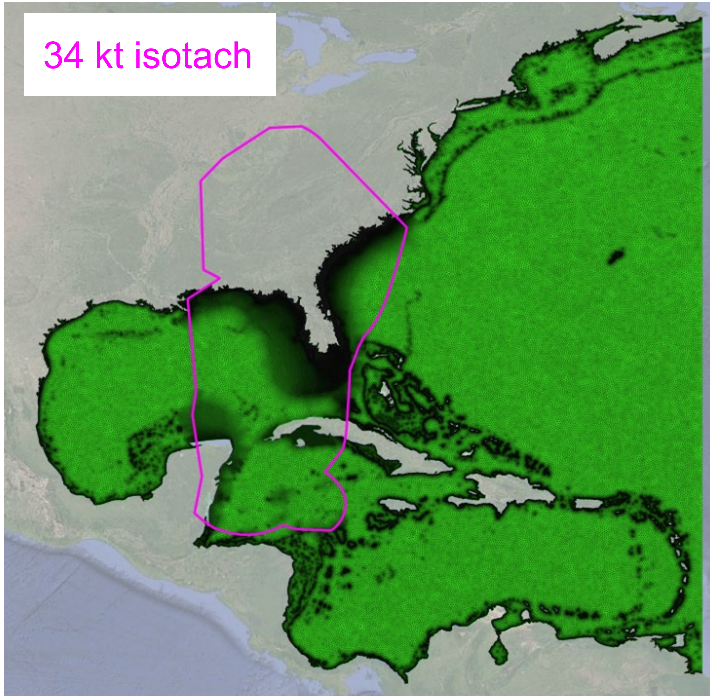
Element sizes

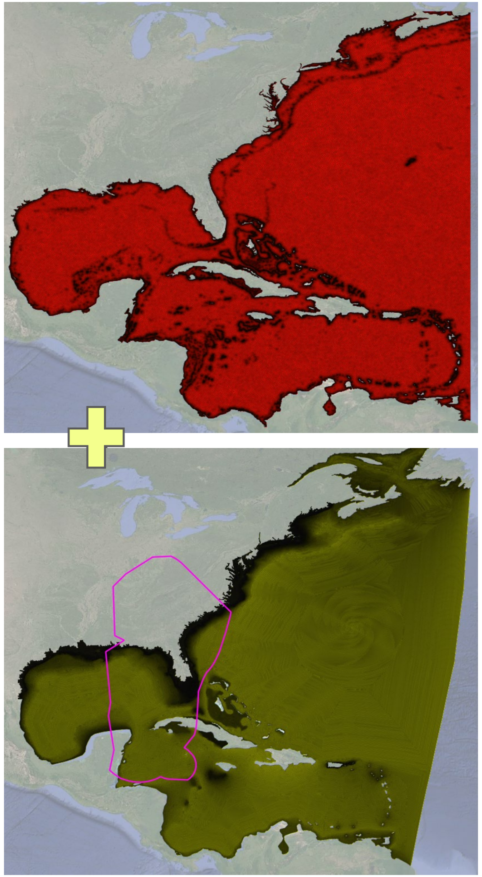
(unstructured)



Example for Helene, 2024

- On-demand subset





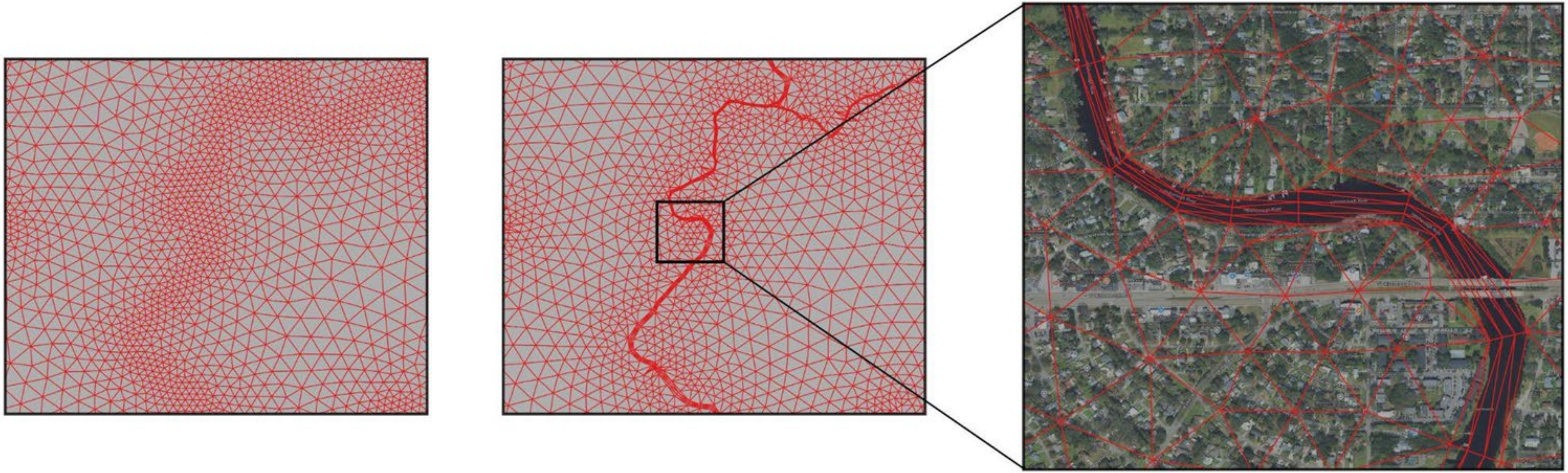
Coarse mesh
< 21 km

STOFS mesh
> 4.5 m

Why is OCSMesh Different?

Goals:

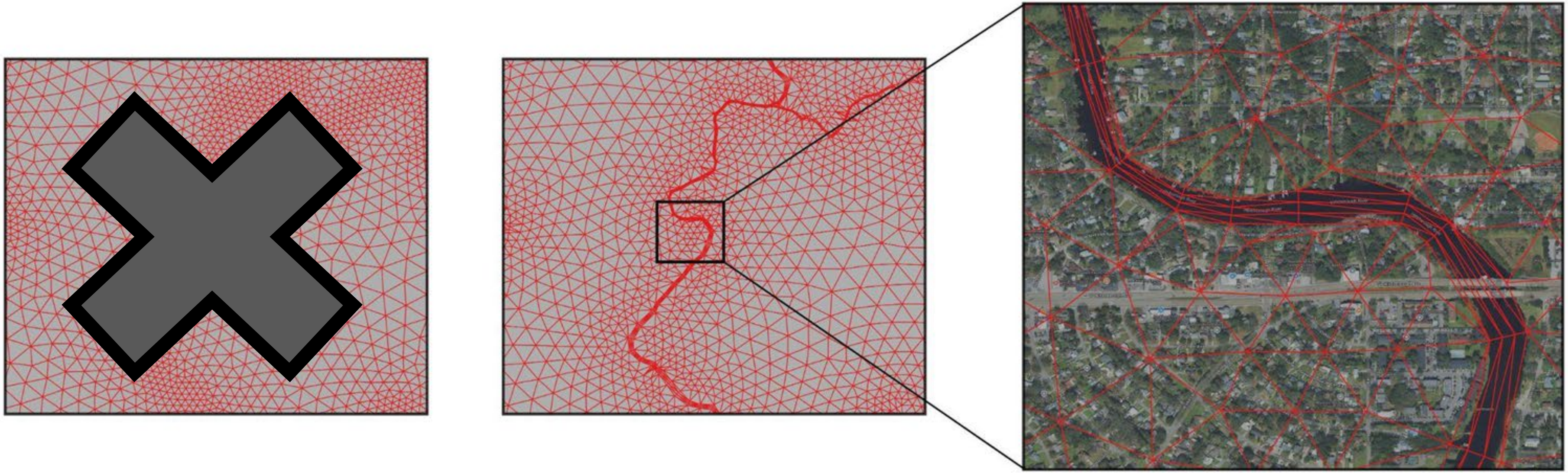
- To promote a mesh generation process that is objective, reproducible, automated, and defensible.
- To overcome the mesh generation trilemma (accuracy, computational cost, and 'quality').
- To relentlessly pursuit of fidelity to the underlying DEM (e.g., capable of resolving arbitrarily small channels with no restriction).
- Quick turn around!



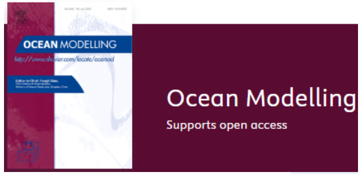
Why is OCSMesh Different?

Goals:

- To promote a mesh generation process that is objective, reproducible, automated, and defensible.
- To overcome the mesh generation trilemma (accuracy, computational cost, and ‘quality’).
- To relentlessly pursuit of fidelity to the underlying DEM (e.g., capable of resolving arbitrarily small channels with no restriction).
- Quick turn around!



OCSMesh’s first paper!



Action	Manuscript Number	Title	Initial Date Submitted	Status Date	Current Status
Action Links	OCEMOD-D-25-00093	OCSMesh and an end-to-end workflow for fully automatic mesh generation with application to compound flood studies	Apr 11, 2025	Jul 03, 2025	Under Review

Top Downloads For:
Hydrology & Ocean Sciences eJournal

Recent Top Papers (60 days) As of: 30 May 2025 - 29 Jul 2025

Rank	Paper	Downloads
1.	Ocsmesh and an End-to-End Workflow for Fully Automatic Mesh Generation with Application to Compound Flood Studies Felicio Cassalho, Soroosh Mani, Fei Ye, Yinglong Zhang and Saeed Moghimi *affiliation not provided to SSRN*, Government of the United States of America - National Oceanic & Atmospheric Administration (NOAA), William & Mary Virginia Institute of Marine Science, William & Mary Virginia Institute of Marine Science and Government of the United States of America - National Oceanic & Atmospheric Administration (NOAA) Date Posted: 22 Apr 2025 Last Revised: 22 Apr 2025	133

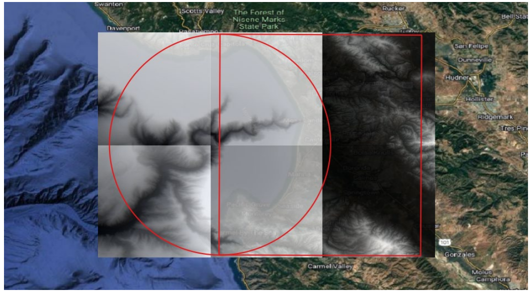
Top Downloads For:
Earth Science Research Network

Recent Top Papers (60 days) As of: 30 May 2025 - 29 Jul 2025

Rank	Paper	Downloads
1.	Seismological Features and Preliminary Damage Assessment of the Devastating March 28, 2025 Myanmar Earthquake: A Comprehensive Overview Chenna Rajaram, Jaya Prakash Vemuri and Shubham Singhal Rajeev Gandhi Memorial College Engineering & Technology, Mahindra University, and *affiliation not provided to SSRN* Date Posted: 16 Apr 2025 Last Revised: 17 Apr 2025	481
2.	Ocsmesh and an End-to-End Workflow for Fully Automatic Mesh Generation with Application to Compound Flood Studies Felicio Cassalho, Soroosh Mani, Fei Ye, Yinglong Zhang and Saeed Moghimi *affiliation not provided to SSRN*, Government of the United States of America - National Oceanic & Atmospheric Administration (NOAA), William & Mary Virginia Institute of Marine Science, William & Mary Virginia Institute of Marine Science and Government of the United States of America - National Oceanic & Atmospheric Administration (NOAA) Date Posted: 22 Apr 2025 Last Revised: 22 Apr 2025	133

Geom Object

- Defines the mesh domain:



Domain Shapefile:

```
domain = gpd.read_file("./domain1.shp")
```

List of Raster Objects:

```
geom_rast_list = [Raster(f) for f in dem_paths]
hfun_rast_list = [Raster(f) for f in dem_paths]
interp_rast_list = [Raster(f) for f in dem_paths]
```

Geom Object

```
geom = Geom(
    geom_rast_list,
    base_shape=domain.union_all(),
    base_shape_crs=domain.crs,
    zmax=15
    )
multipolygon = geom.get_multipolygon()
gpd.GeoSeries(multipolygon, crs=geom.crs).plot()
```

HFun Object

- Size function:

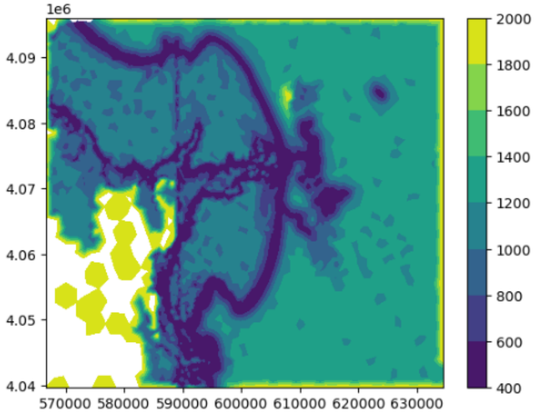
Hfun

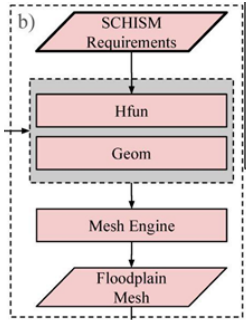
```
hfun = Hfun(
    hfun_rast_list,
    base_shape=domain.buffer(0.1).union_all(),
    base_shape_crs=geom.crs,
    hmin=400, hmax=2000,
    method='fast')
```

```
C:\Users\Felicio.Cassalho\AppData\Local\Temp\1\ipykernel_7932\1186571159.py:3: UserWar
```

```
# Constant values for floodplain:
hfun.add_constant_value(800, lower_bound=5, upper_bound=10)
hfun.add_constant_value(1200, lower_bound=10)
# Auto-identify channels:
hfun.add_channel(level=0, width=1500, target_size=200, expansion_rate=0.01)
# Slope-based resolution:
hfun.add_subtidal_flow_limiter(hmin=400, hmax=1000, lower_bound=-1000, upper_bound=0)
# High-resolution at the coast:
hfun.add_contour(0, .001, 400)
```

```
# Jigsaw size function
hfun_msh_t = hfun.msh_t()
plot_tri(hfun_msh_t)
```





Legend:

- Outside OCSMesh
- Floodplain Mesh Gen
- River Mesh Gen
- Mesh Combination

Jigsaw Driver

- Main engine for mesh generation:

Passing Geometry and Size Functions to the Mesh Driver

```
driver = JigsawDriver(geom, hfun)
mesh = driver.run()
```

Interpolating DEM into Mesh (we could use a different set of DEMs here)

```
mesh.interpolate(interp_rast_list)
```

Defining Mesh Boundaries

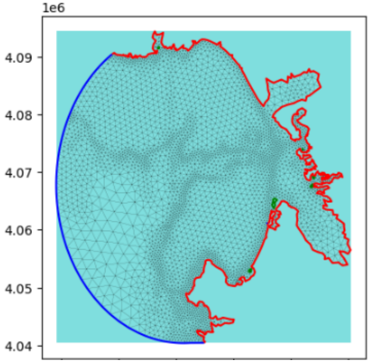
```
mesh.boundaries.auto_generate(-1)
```

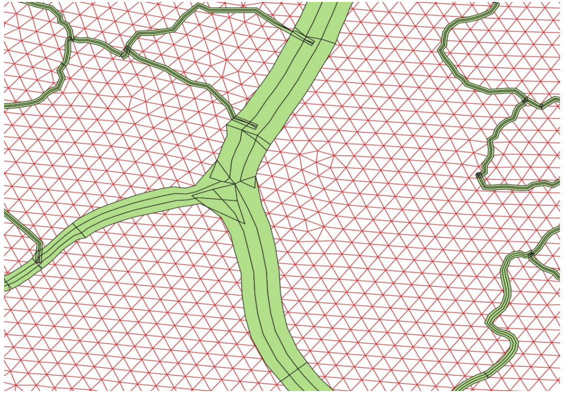
Visual Check

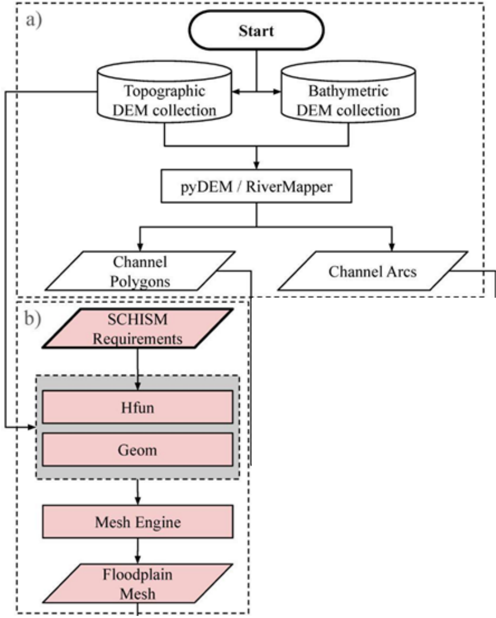
```
gs_region = gpd.GeoSeries(mesh.bbox, crs=mesh.crs)
ax = gs_region.plot(color='c', alpha=.5)
plot_mesh_edge(mesh.msh_t, ax=ax, lw=0.1, color='k')
mesh.boundaries.ocean().plot(ax=ax, color='b', label='Ocean')
mesh.boundaries.interior().plot(ax=ax, color='g', label='Islands')
mesh.boundaries.land().plot(ax=ax, color='r', label='Land')
```

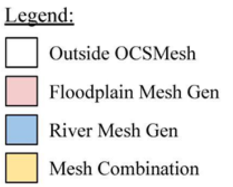
```
C:\Users\Felicio.Cassalho\AppData\Local\Temp\1\ipykernel_7932\4008164403.py:6: DeprecationWar
  mesh.boundaries.ocean().plot(ax=ax, color='b', label='Ocean')
```

```
<Axes: >
```









Jigsaw Driver

- Main engine for mesh generation:

Passing Geometry and Size Functions to the Mesh Driver

```
driver = JigsawDriver(geom, hfun)
mesh = driver.run()
```

Interpolating DEM into Mesh (we could use a different set of DEMs here)

```
mesh.interpolate(interp_rast_list)
```

Defining Mesh Boundaries

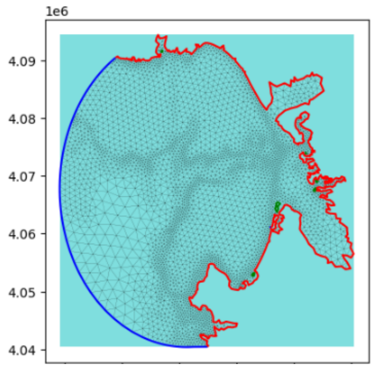
```
mesh.boundaries.auto_generate(-1)
```

Visual Check

```
gs_region = gpd.GeoSeries(mesh.bbox, crs=mesh.crs)
ax = gs_region.plot(color='c', alpha=.5)
plot_mesh_edge(mesh.msh_t, ax=ax, lw=0.1, color='k')
mesh.boundaries.ocean().plot(ax=ax, color='b', label='Ocean')
mesh.boundaries.interior().plot(ax=ax, color='g', label='Islands')
mesh.boundaries.land().plot(ax=ax, color='r', label='Land')
```

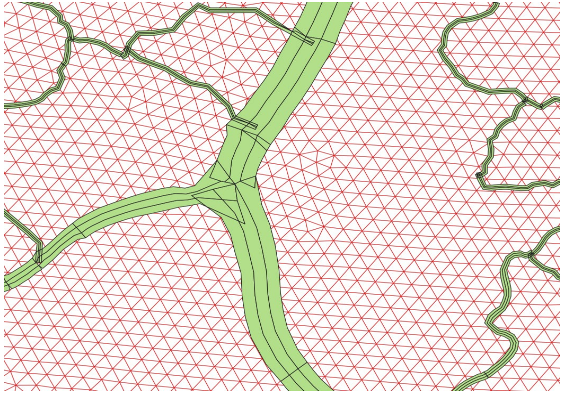
```
C:\Users\Felicio.Cassalho\AppData\Local\Temp\1\ipykernel_7932\4008164403.py:6: DeprecationWar
  mesh.boundaries.ocean().plot(ax=ax, color='b', label='Ocean')
```

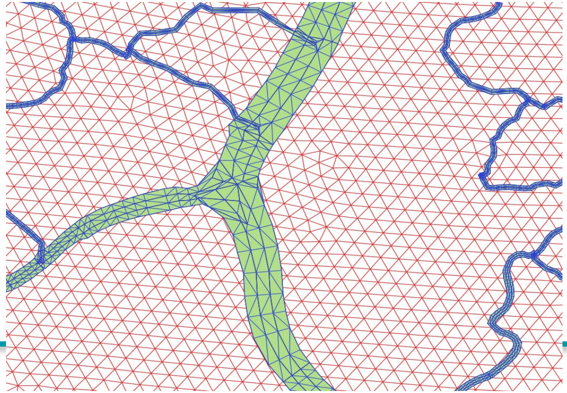
```
<Axes: >
```

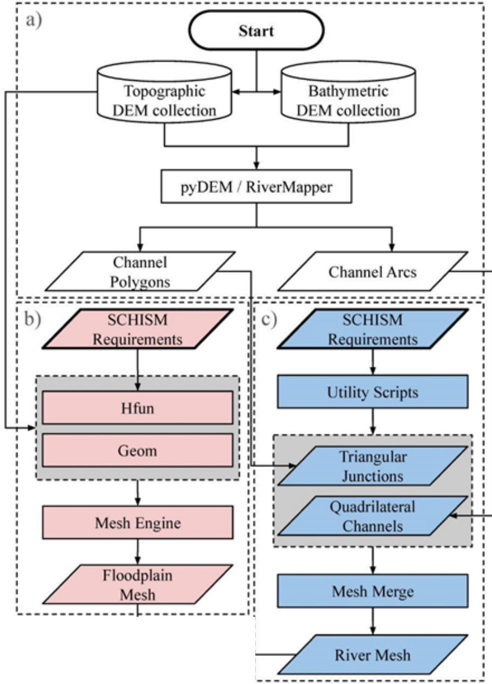


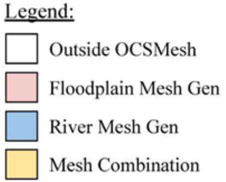
Shapefile Tri/Quad

```
rm_poly = gpd.read_file(path+"inputs/total_river_polys.shp")
river_tr = ocsmesh.utils.triangulate_rivermapper_poly(rm_poly)
river_tr = ocsmesh.utils.clip_mesh_by_shape(river_tr, gdf.union_a
ocsmesh.Mesh(river_tr).write(path+"outputs/river_tr.2dm", format=
```

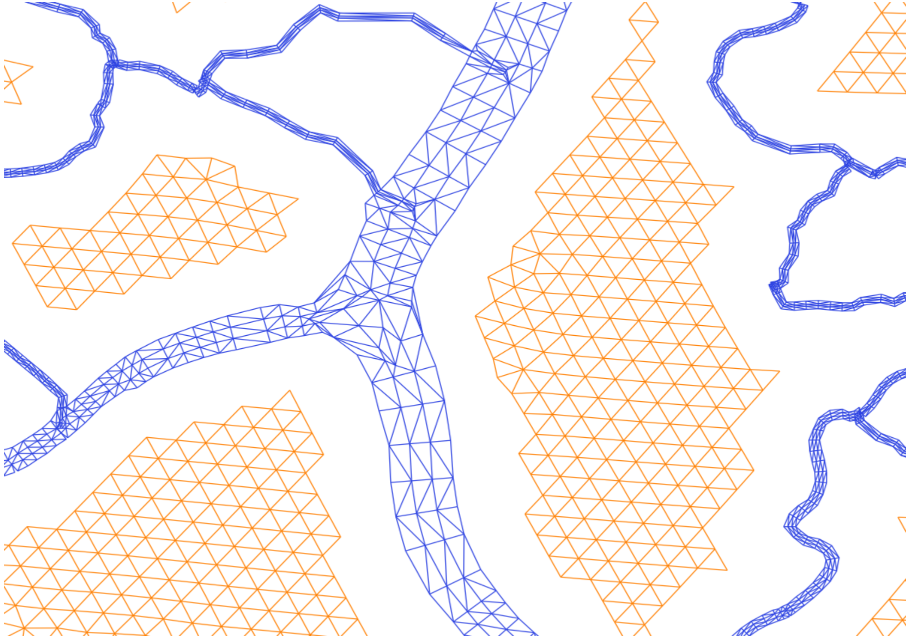


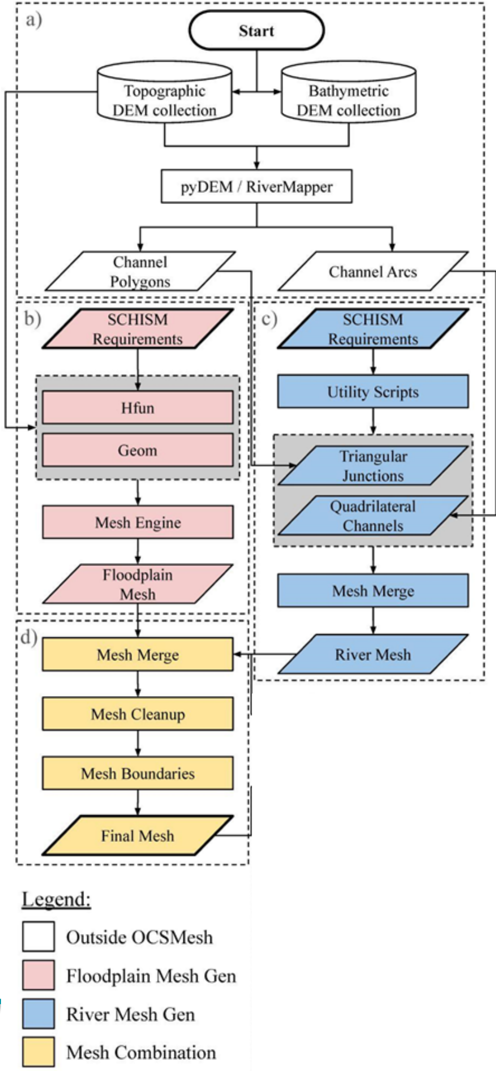






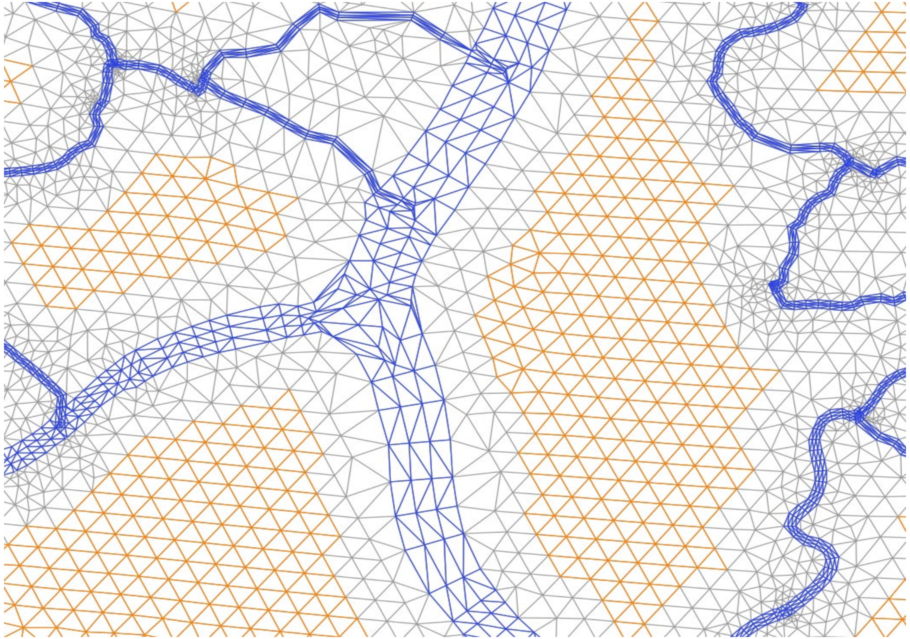
Mesh Merging

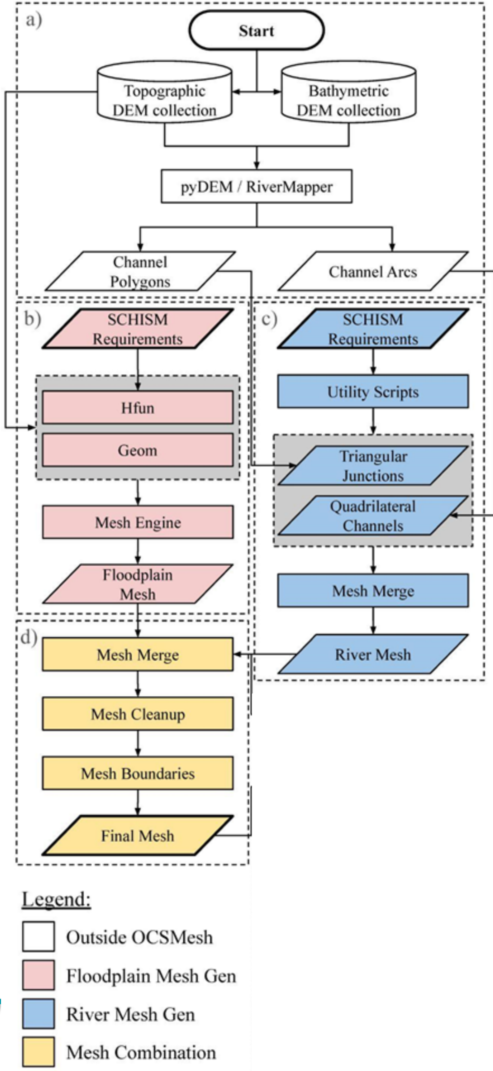




Mesh Merging

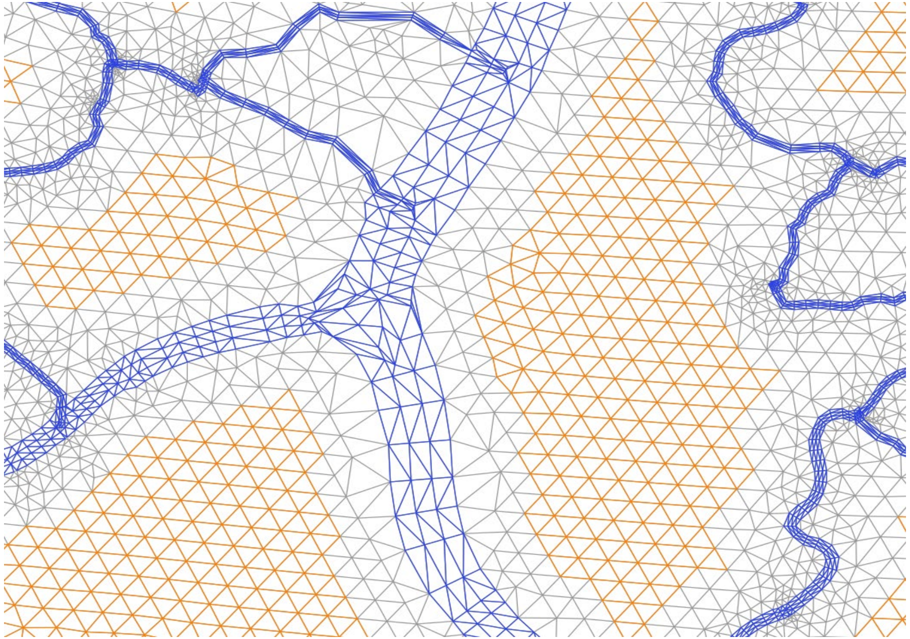
```
fp_r = ocsmesh.utils.merge_overlapping_meshes([fp_mesh.msh_t,river_mesh.msh_t])
```

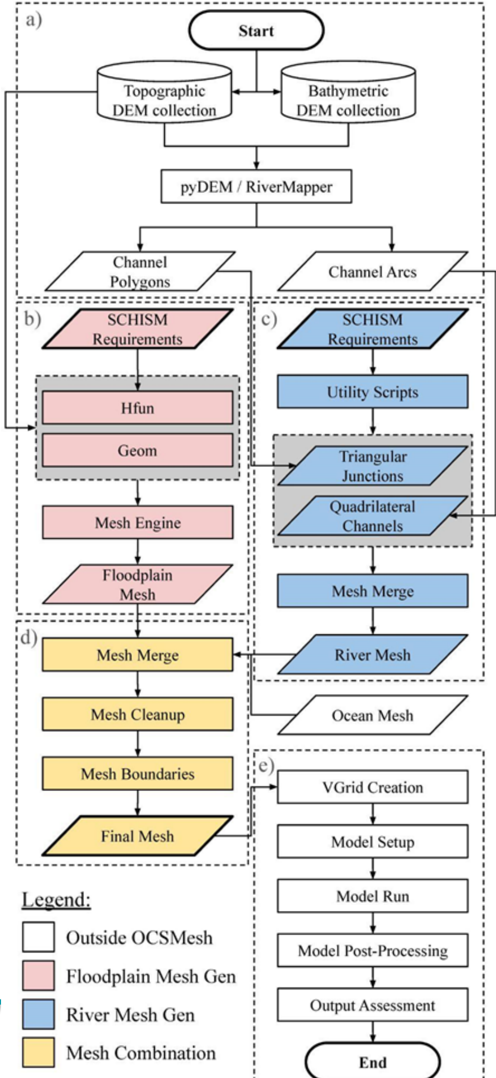




Mesh Merging

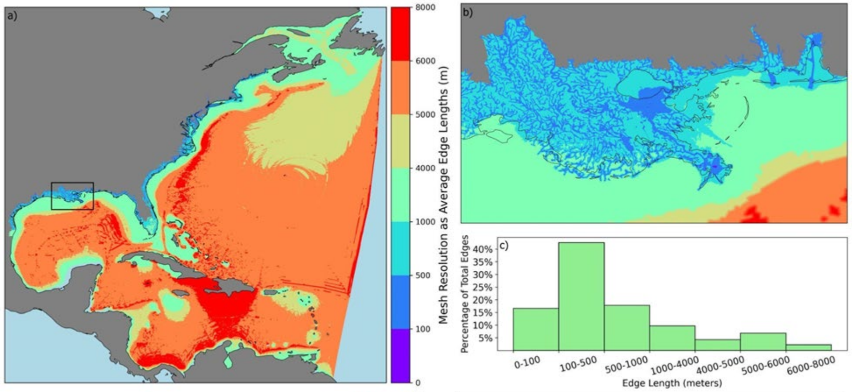
```
fp_r = ocsmesh.utils.merge_overlapping_meshes([fp_mesh.msh_t,river_mesh.msh_t])
```



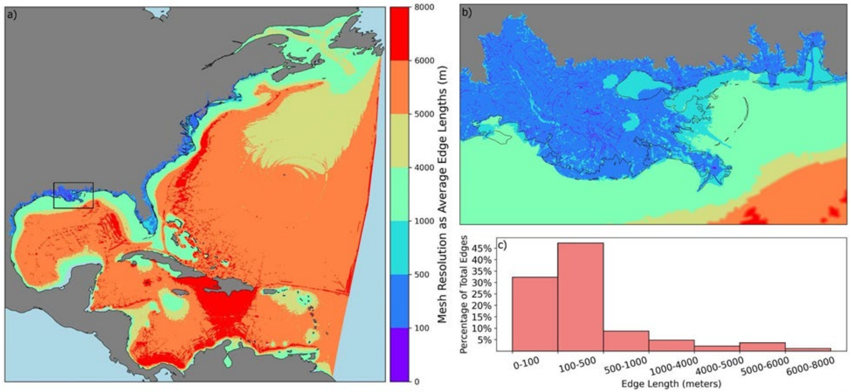


OCSMesh Mesh Reproducibility

<u>STOFS-3D</u>

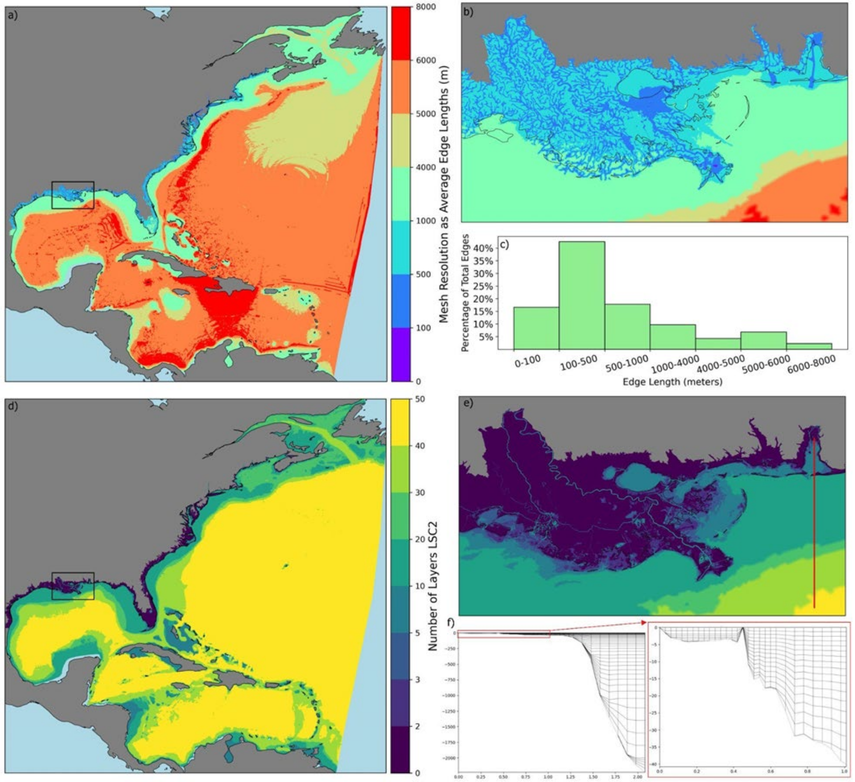


<u>OCSMesh (< 6hrs)</u>

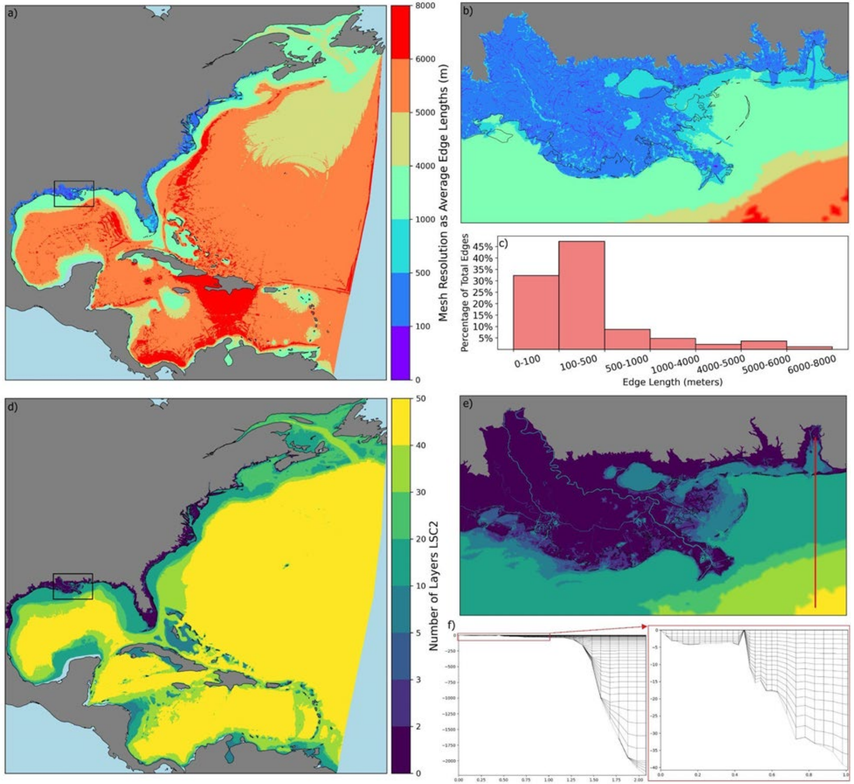


OCSMesh Mesh Reproducibility

STOFS-3D

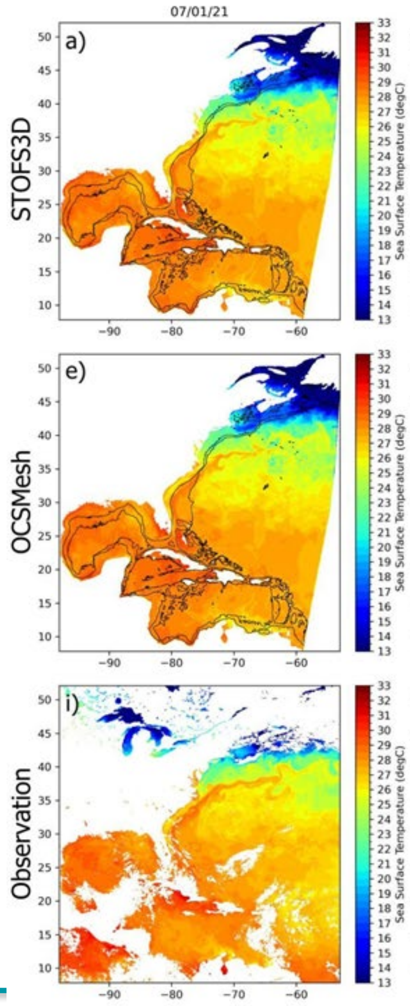


OCSMesh (< 6hrs)



Performance

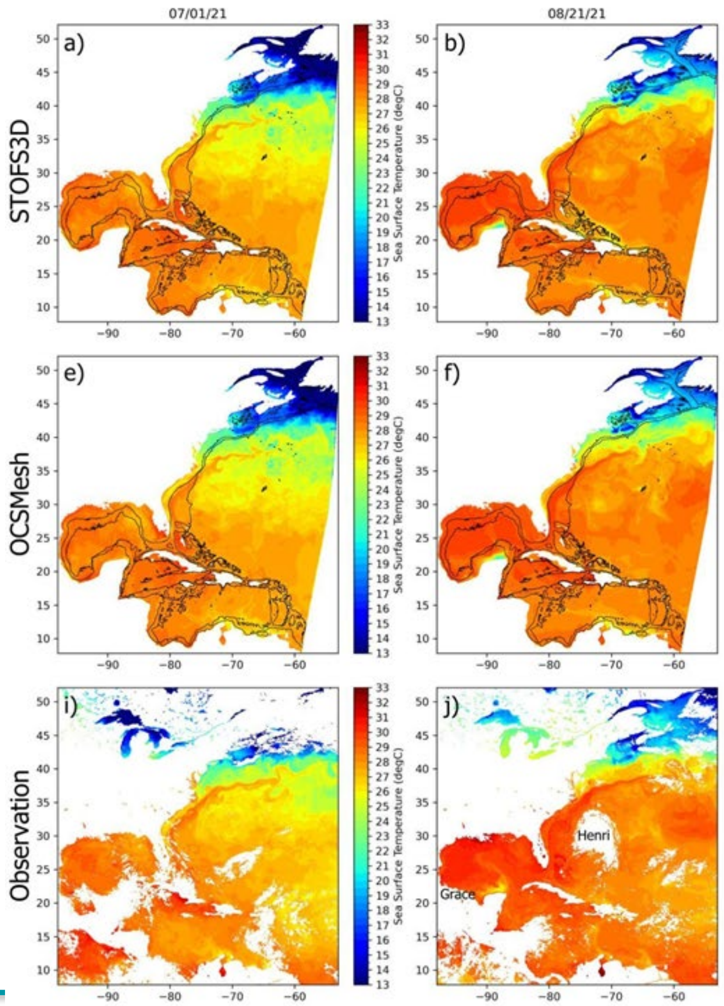
Reproducibility



Performance

Reproducibility

- Cold water wake (Hurricane Grace and Henri)

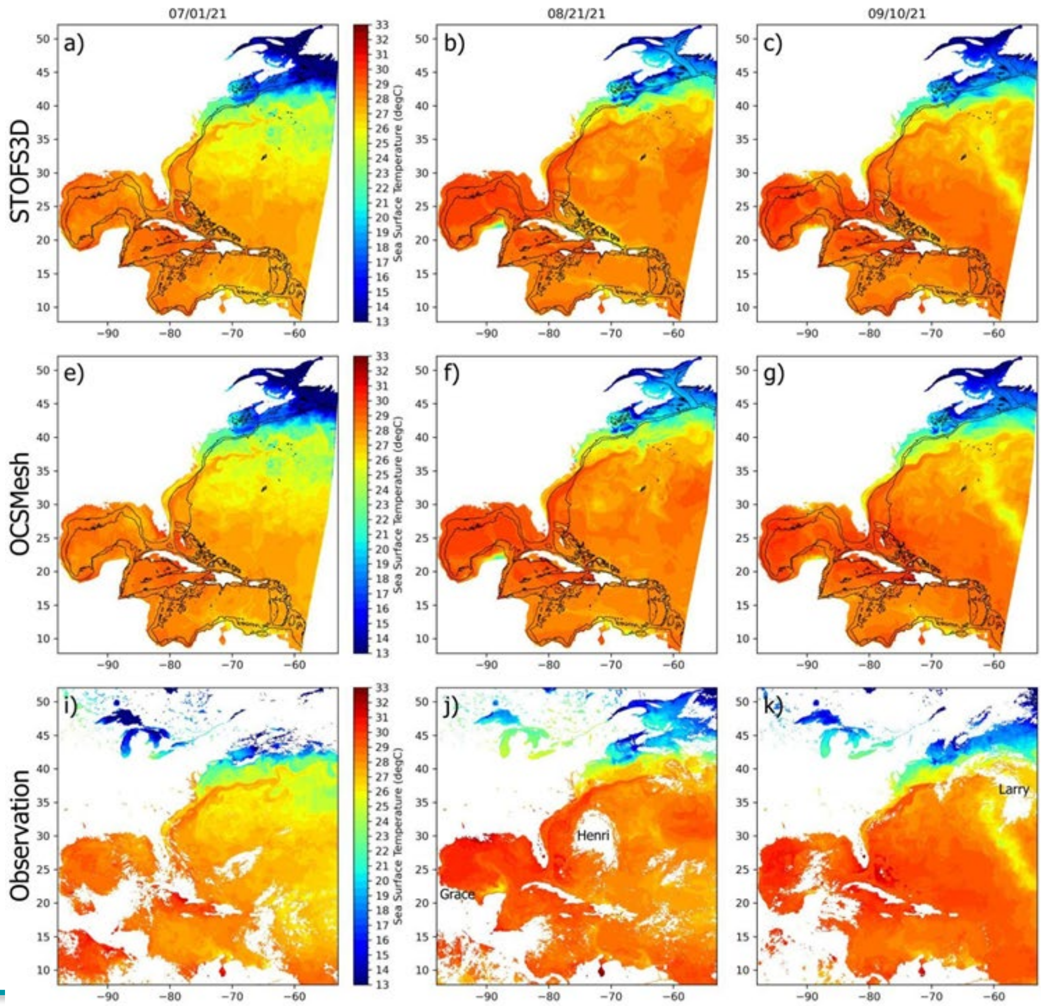


Performance

Reproducibility

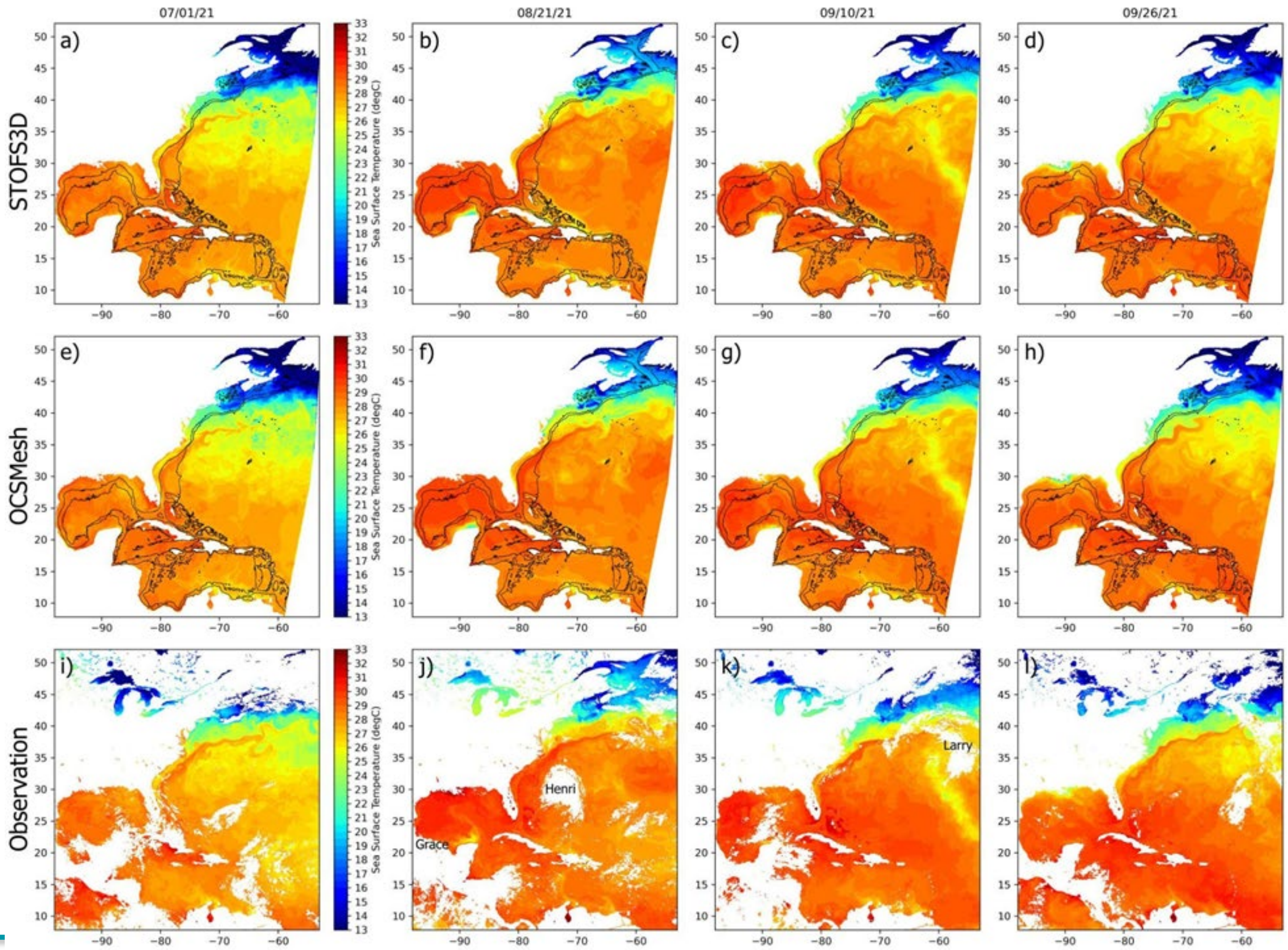
- Cold water wake (Hurricane Grace and Henri)

- Cold water wake along Hurricane Larry' track



Performance Reproducibility

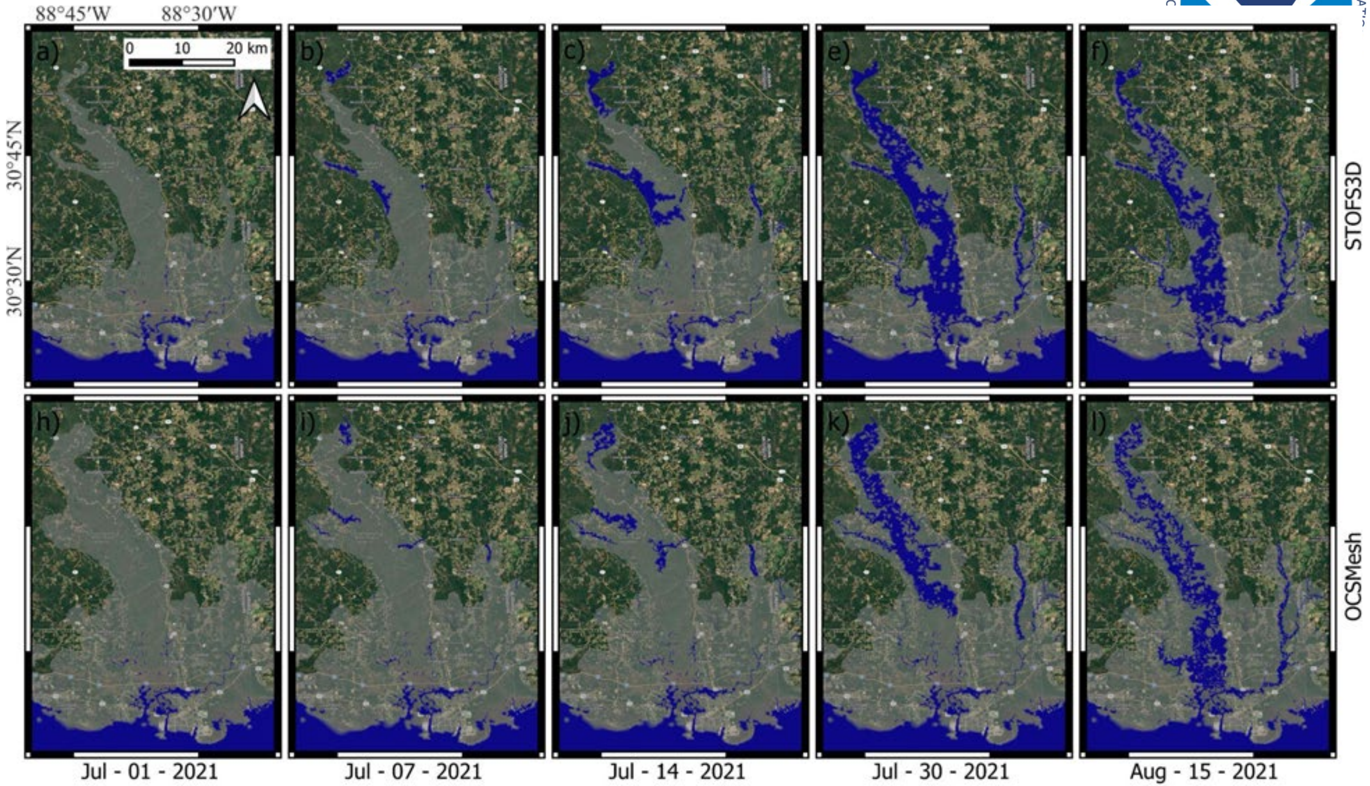
- Cold water wake (Hurricane Grace and Henri)
- Cold water wake along Hurricane Larry' track
- Cold water along the Mississippi Delta



NOAA/NOS' Office of Coast Survey

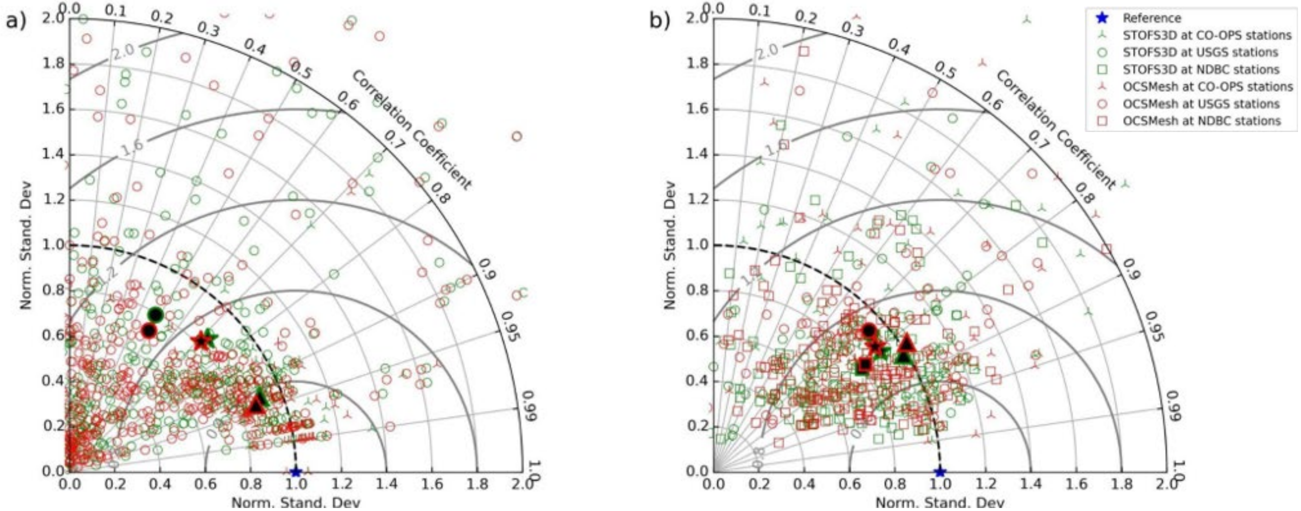
Performance Reproducibility

- Cold water wake (Hurricane Grace and Henri)
- Cold water wake along Hurricane Larry' track
- Cold water along the Mississippi Delta
- Downstream Runoff Propagation

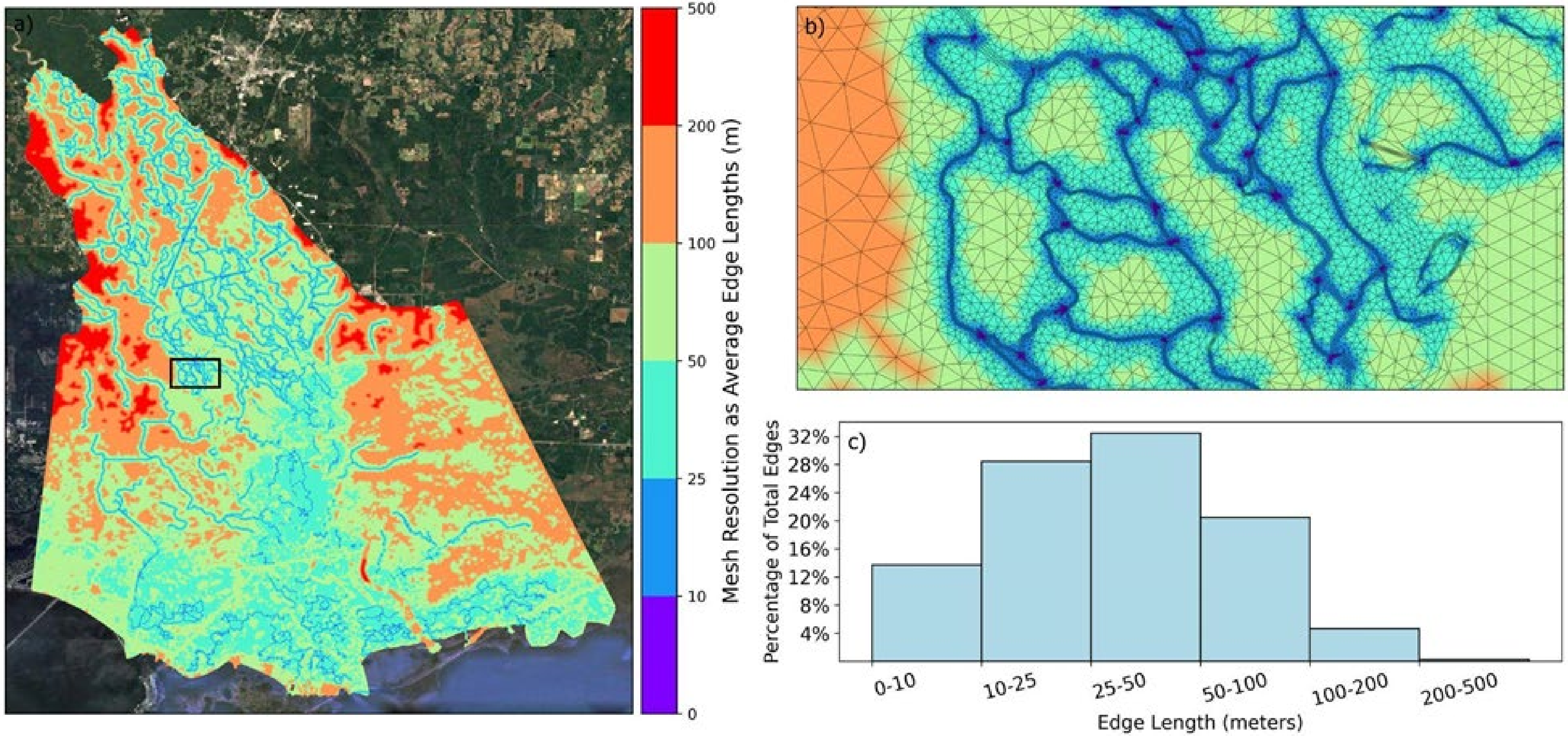


Performance Reproducibility

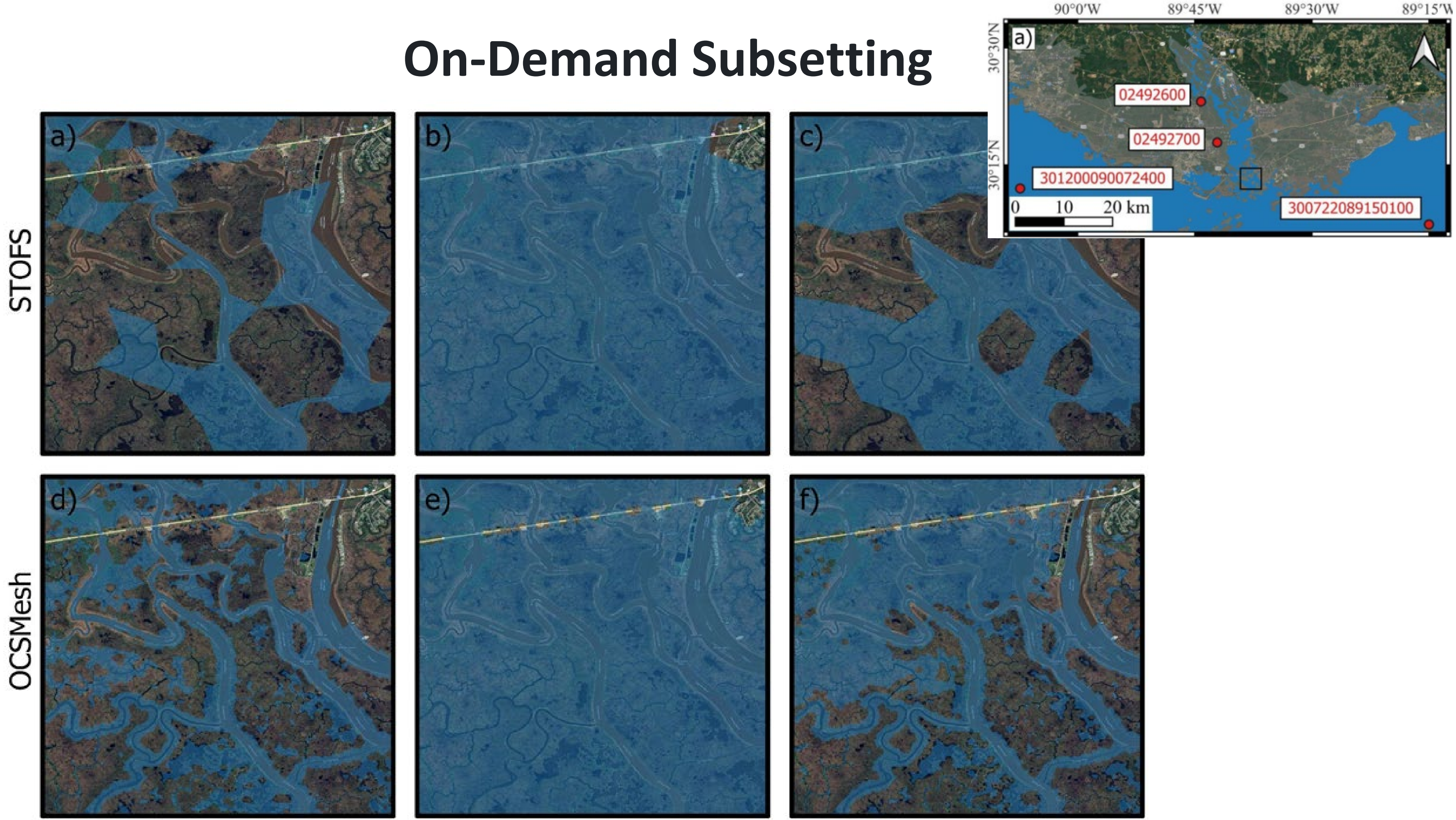
- Cold water wake (Hurricane Grace and Henri)
- Cold water wake along Hurricane Larry’ track
- Cold water along the Mississippi Delta
- Downstream Runoff Propagation
- Skill Assessment



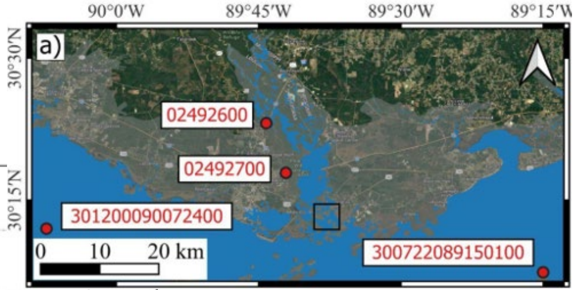
On-Demand Subsetting

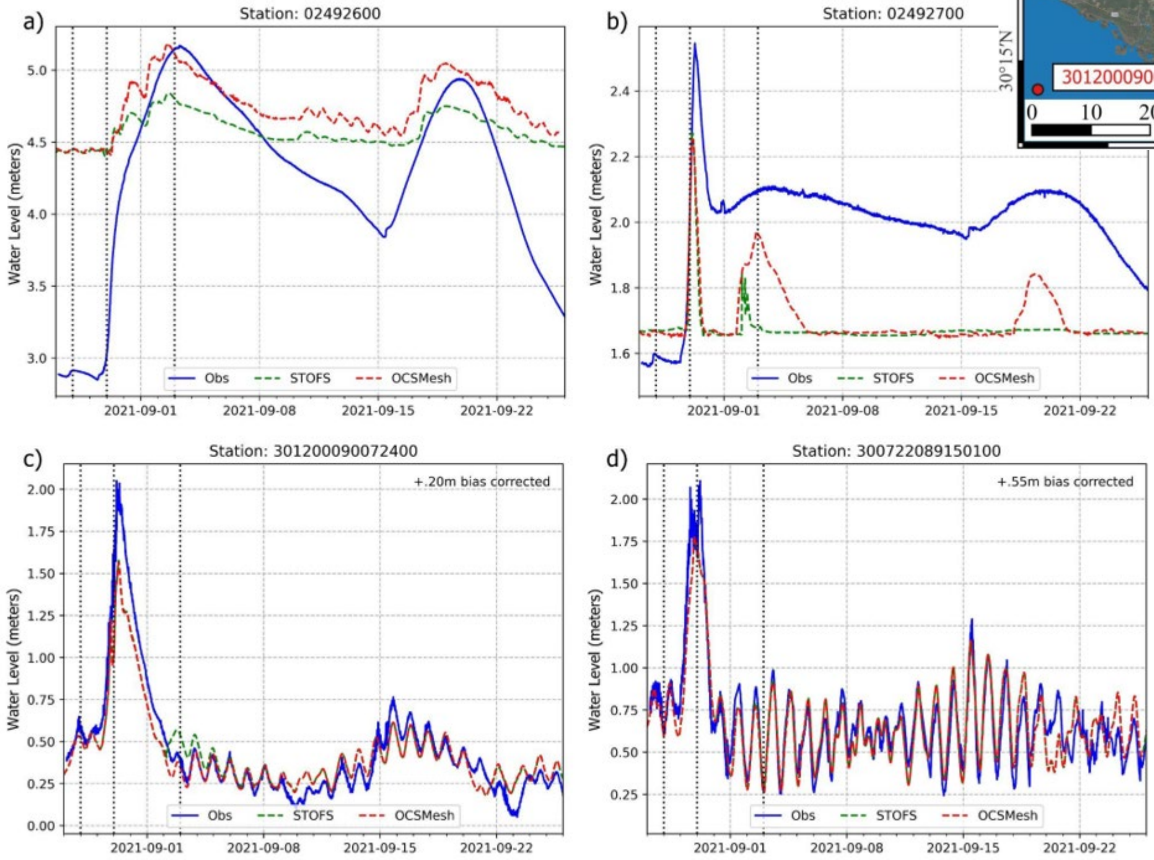


On-Demand Subsetting



On-Demand Subsetting



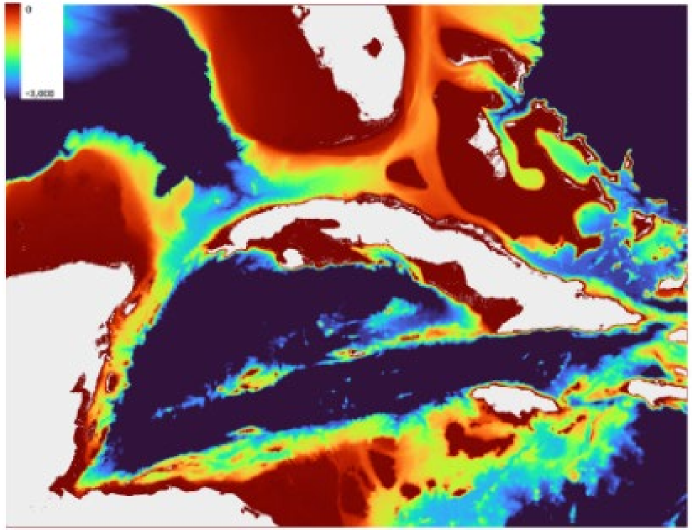


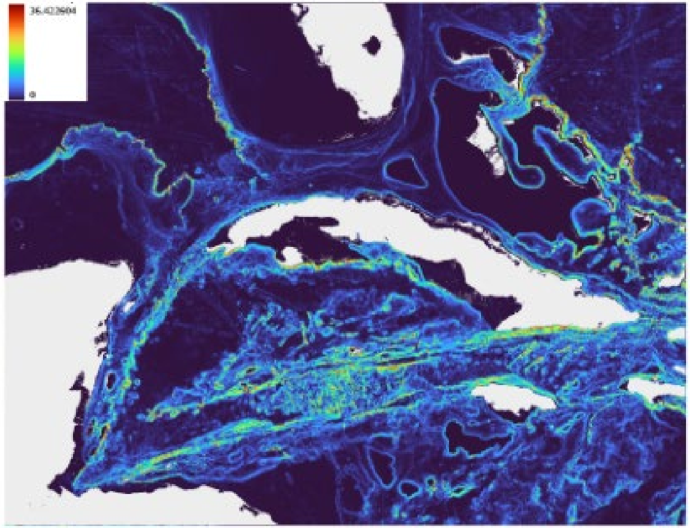
DEM-based Ocean Mesh Gen for 3D-SCHISM

1. **eddy-regime zones:** should be featureless with horizontal resolution >> depth
2. **non-eddy regime zones:** for meshing purposes it can be treated as overland (i.e., variable resolution as a function of depth/slope/channels/etc.)
3. **transition zones:** between eddy and non-eddy, usually high slope, usually narrow.

- The first goal is to be able to delineate these zones

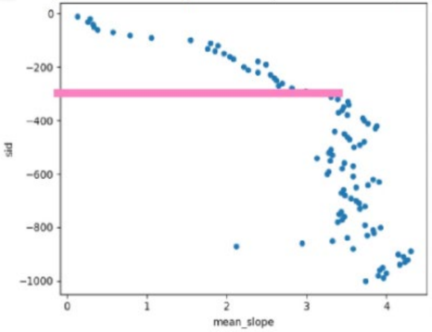
With this in mind I started to put together some simple functions based on the DEM/Slope:

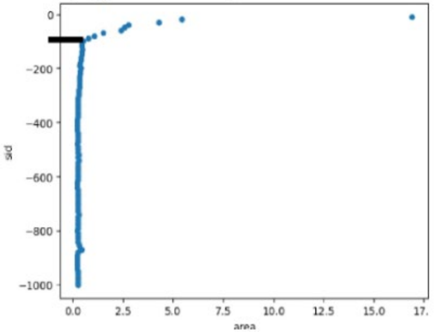




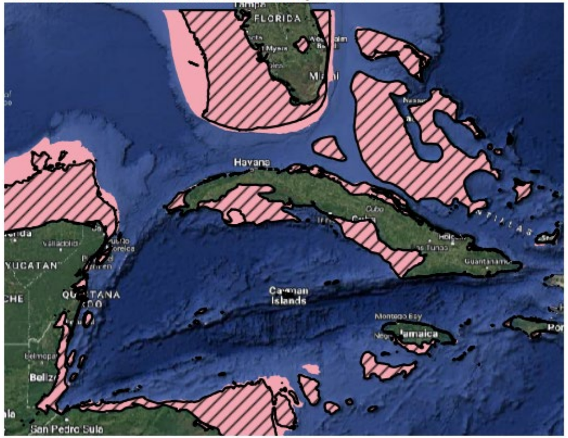
DEM-based Ocean Mesh Gen for 3D-SCHISM

- non-eddy zone = mean slope inflection point VS. non-eddy zone= contour polygon area inflection point:



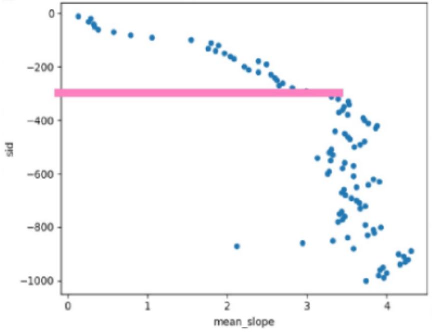


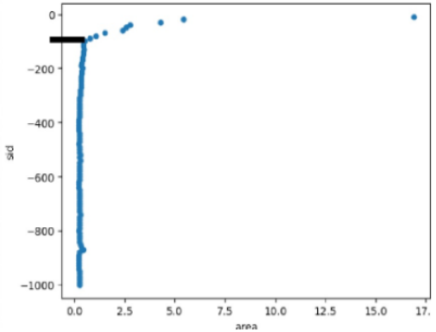
- These different criteria results in the non-eddy zones:



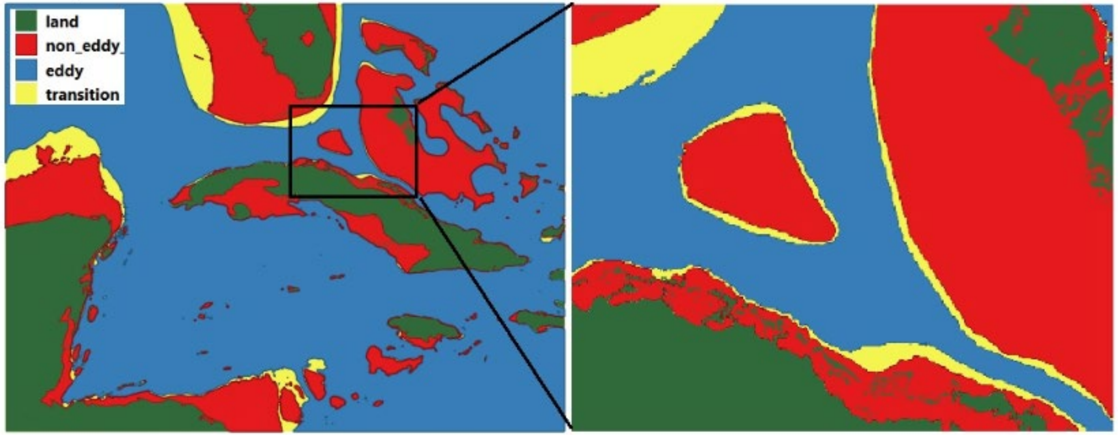
DEM-based Ocean Mesh Gen for 3D-SCHISM

- non-eddy zone = mean slope inflection point VS. non-eddy zone= contour polygon area inflection point:





The final product would be something like:



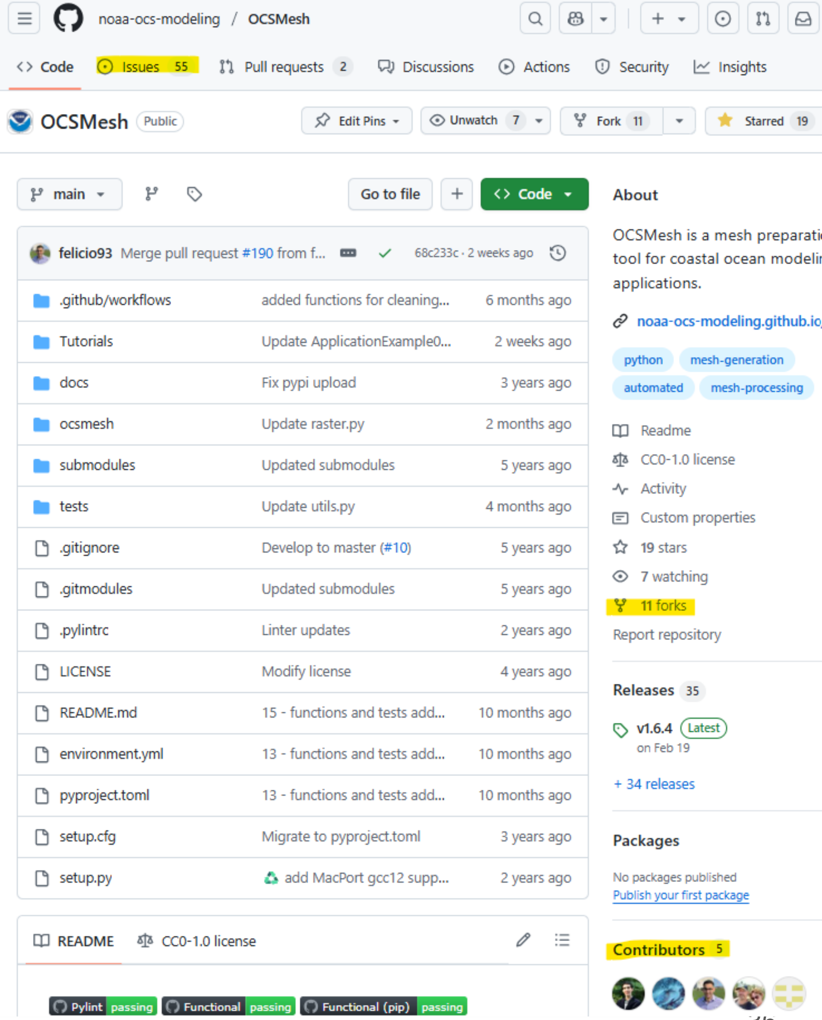
Parallelization of Hfun

- Creating the Hfun is computationally expensive.
- In its current form Hfun is not picklable and thus cannot be parallelized.
- GSoC proposal accepted and intern onboarded.
- Goals:
  - Parallelize Hfun Object.
  - Potentially parallelize the generation of other objects (e.g., Geom) and functions (e.g., interpolate).
  - *Reduce number of dependencies.*
  - *Improve efficiency.*

Google Summer of Code

Conclusions

- **Past development:**
  - Geom
  - Hfun
  - Collector
  - Jigsaw Mesh Engine
- **Recent updates:**
  - End-to-end Mesh Gen
  - Seamless integration of meshes
  - Creek-to-Ocean
- **Plan for the future:**
  - Code Parallelization
  - Automate Ocean Mesh Gen



Thank You!

Any Questions?