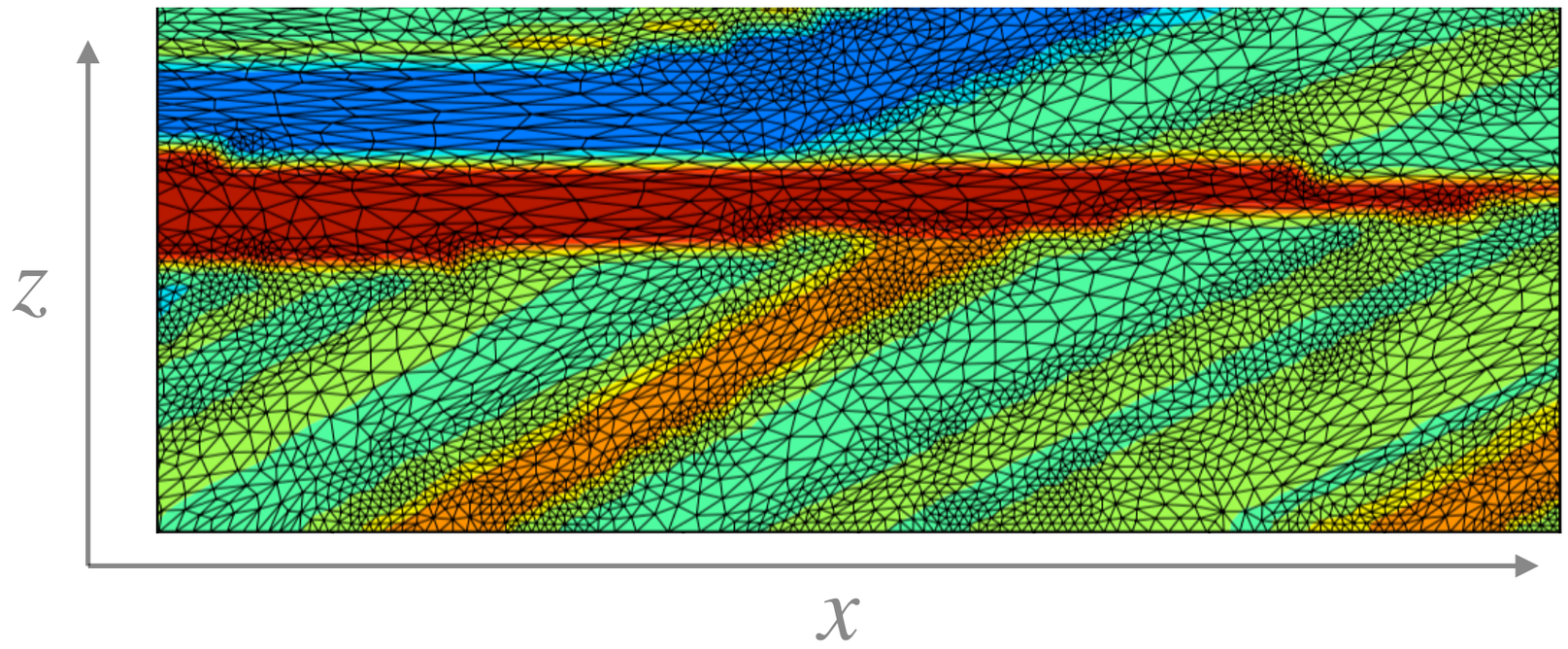
DistMesh for automatic unstructured mesh generation for subsurface imaging

Keith J. Roberts, Ph.D.
Advisor: Dr. Rafael Gioria, Dr. Edson Gomi
Universidade de São Paulo
RCGI-Research Centre for Gas Innovation
16-09-2019



Overview

- Introduction to seismic inversion.
- Introduction to mesh generation (in particular DistMesh)
- Proposed developments

- **Goal:** Automatically develop unstructured meshes for the seismic inversion problem solved using the Finite Element Method.

What is seismic inversion?

Acoustic wave equation. equation

$$\partial_t^2 p = c^2 \nabla^2 p + s$$

$c(\mathbf{x})$ = speed of sound

$s(\mathbf{x}, t)$ = pressure source field

$p(\mathbf{x}, t)$ = unkonwn pressure field

$x \in \mathbb{R}^N$ where $N = 2,3$

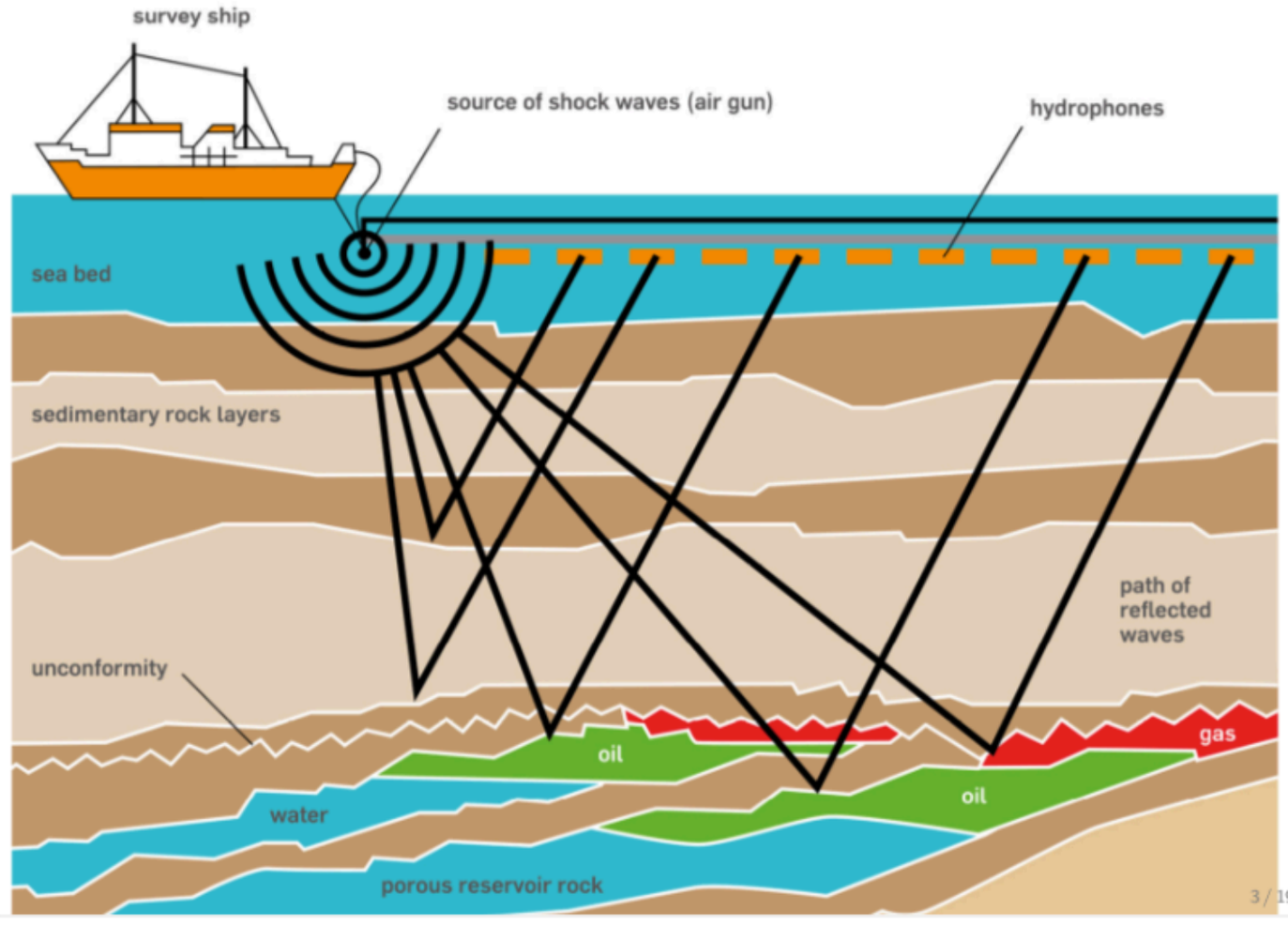


Image from: Devito Jupyter notebook tutorial 1

What is seismic inversion?

Create an "accurate" image that depicts the form and structure of the subsurface.

- We need to a velocity model to get an accurate image (recall c^2)!
- Improve velocity model using FWI.

RTM: Reverse Time Migration

FWI: Full Waveform Inversion

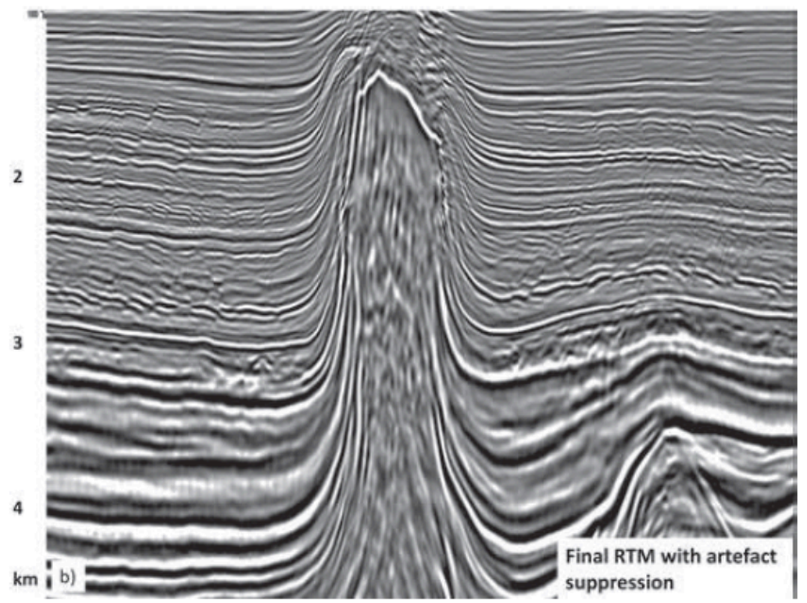
1. **Forward modeling** a synthetic shot (i.e. evaluating what the expanding wavefront looks like at incremental steps of a few milliseconds of propagation time).

2. **Back-propagating** the recorded data (i.e. evaluating what the recorded field data looked like for previous propagation times, working backwards from the final recording time to time zero).

3. **Imaging condition:** convolving extrapolated wavefields together at each corresponding propagation time-step.

4. **Storing**: intelligently summing the results for all propagation steps.

Example result of subsurface from RTM



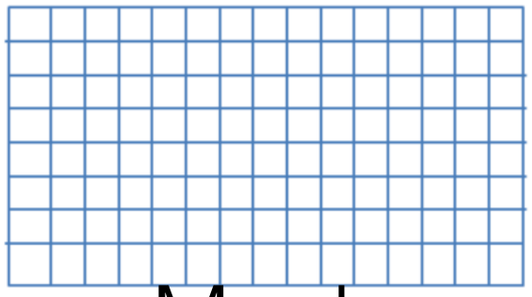
I.F. Jones. Tutorial: Migration imaging conditions. First Break , 32:45–55,12 2014.

Why could we use a mesh?

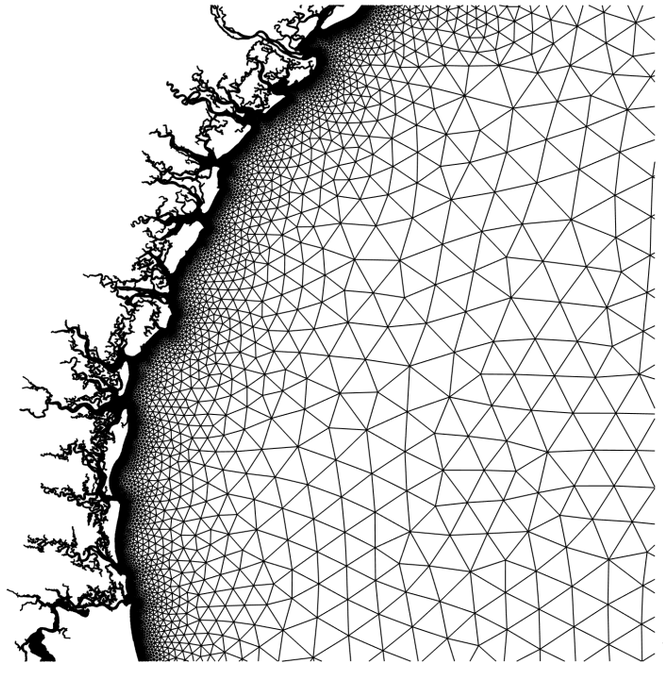
- Traditionally, seismic inversions have relied on Finite Difference Methods (**FDM**) instead of the Finite Element Method (**FEM**) to model the subsurface.
  - So what are some benefits for using unstructured meshes for this problem?
  - Fewer DoFs per domain with locally higher resolution.

Pros of **FDM**	Pros of **FEM**
Algorithmic simplicity of code	Geometric adaptivity (feature constraints)
Grid development is easy	Seamless cross-scale modeling
Numerical analysis is easier	Mesh adaptation

Grid

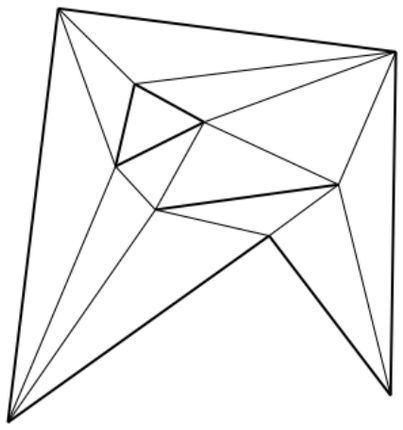


Mesh

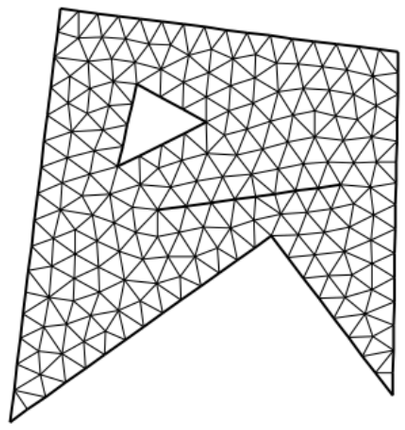


What defines a mesh?

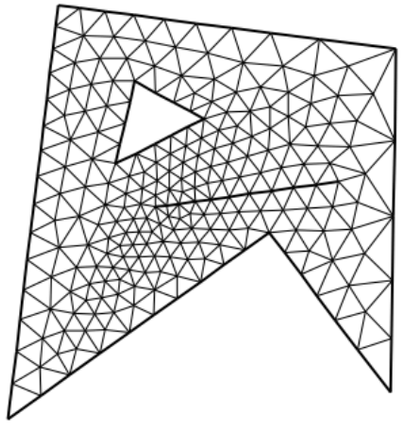
- A discrete representation of some domain of some spatial domain Ω
- Composed of polygonal elements (e.g., **triangles**, **tetrahedral**, quadrilaterals, mixed triangle/quads).



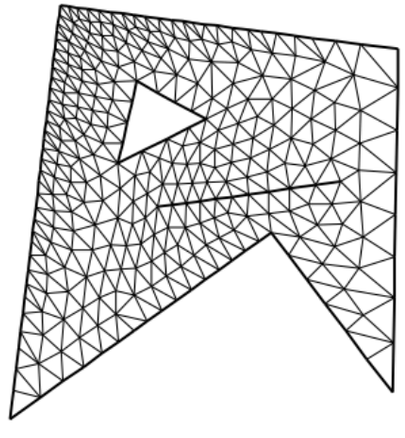
CDT



Uniform



Isotropic



Anisotropic

Frank J. Bossen and Paul Heckbert. A pliant method for anisotropic mesh generation Proceedings of the 5th International Meshing Roundtable, 10 1998

Mesh generators available?

- Lots of standard mesh generators available!
  - **General**: *CGAL*, *Qhull*, *gmsh*, *Triangle, DistMesh, Bubble packing, ParaView, TetGen…*

  - **Problem specific**: *MeshIT*, *RingMesh, JIGSAW-Geo*, *PRAGMATIC*, ***OceanMesh2D,*** ADMesh+, Shingle, SMS, GMS…

Functionality we need for seismic imaging…

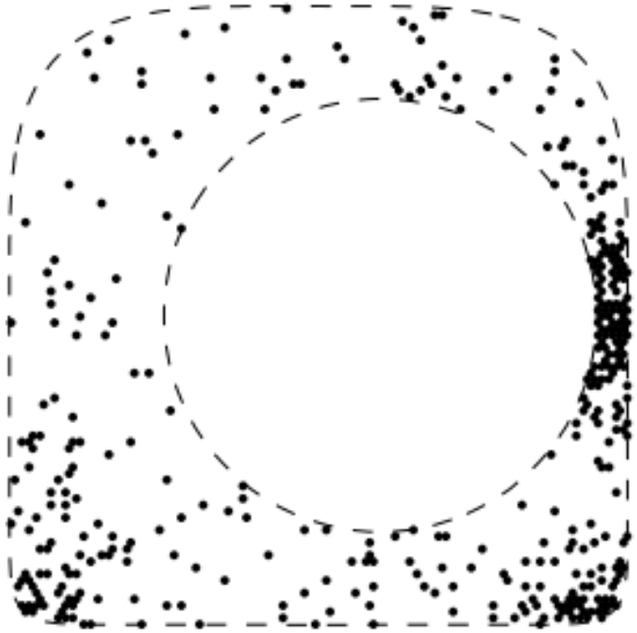
Support both 2D and 3D meshing.
Support distributed memory parallelism.
Contain mesh size functions/no resolution controls.
Support both mesh generation and adaptation.
Support anisotropic and isotropic shaped elements

Introduction to DistMesh

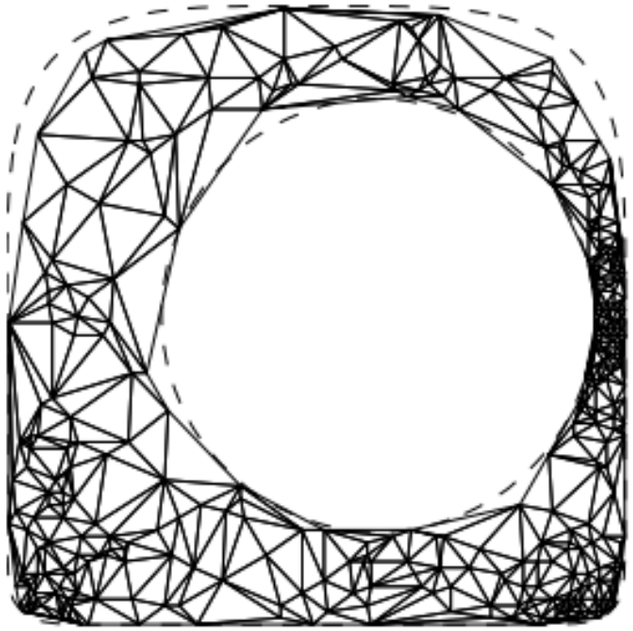
A simple algorithm that combines a physical principle of force equilibrium in a truss structure with a mathematical representation of the geometry using signed distance functions.

- Persson, P.-O. and Strang, G. 2004 A simple mesh generator in Matlab. SIAM Review.
  Download scripts at: http://persson.berkeley.edu/distmesh/

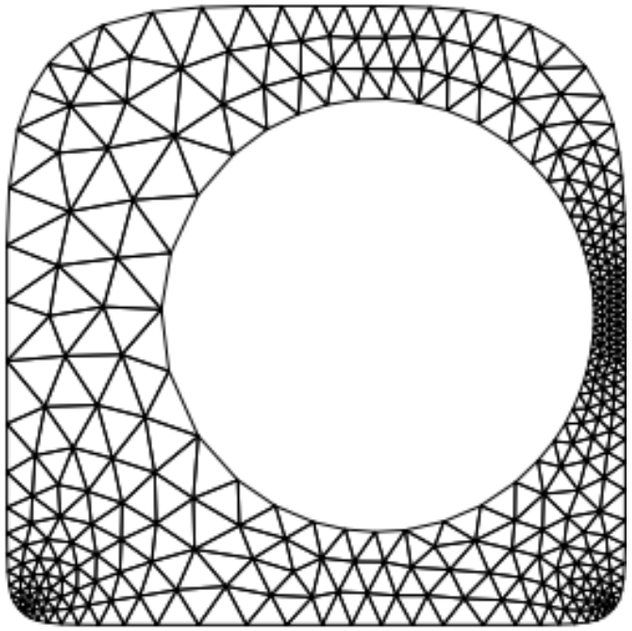
1–2: Distribute points



3: Triangulate



4–7: Force equilibrium



Why use DistMesh?

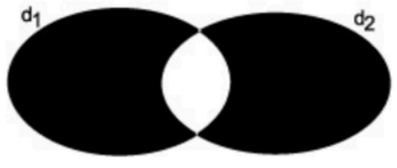
Pros of **DistMesh**	Cons of **DistMesh**
Algorithmic simplicity (“easy” to modify)	No guaranteed tria. quality
Mesh adaptation	Only implemented in high-level languages (MATLAB)
Support for N-dimensional meshing and anisotropic meshing	Not paralleized

Simple overview of DistMesh

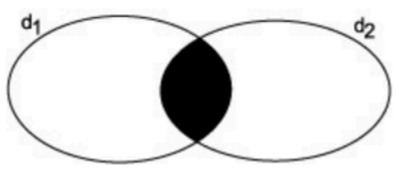
Step 1. Define a domain using signed distance functions.

Union: 

$\min(d_1(x), d_2(x))$

Difference: 

$\max(d_1(x), -d_2(x)) \cup$
$\max(-d_1(x), d_2(x))$

Intersection; 

$\max(d_1(x), d_2(x))$

Step 2. Distribute a set of vertices interior to the domain.

Step 3. Iteratively move vertices to obtain force equilibrium.

Step 4. Apply termination criterion when all vertices are fixed in space.

Signed distance functions

in-polygon test

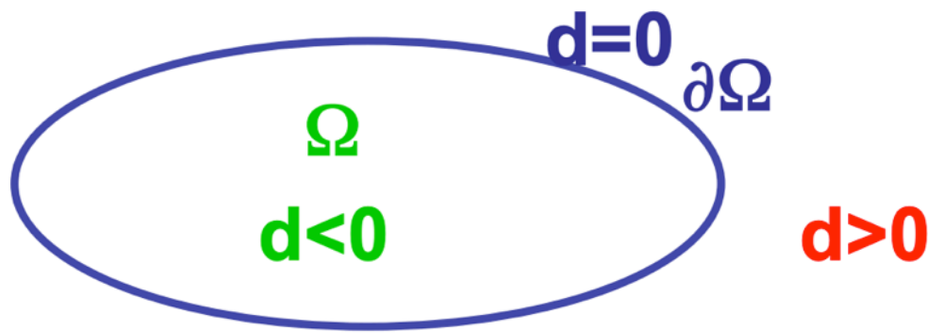
Fast marching method

$$d(\boldsymbol{x})_\Omega = s_\Omega(\boldsymbol{x}) \min_{\boldsymbol{y}\in\partial\Omega}(||\boldsymbol{x}-\boldsymbol{y}||)$$

sign function

nearest distance

$$s(\boldsymbol{x})_\Omega := \begin{cases} -1, & \text{if } \boldsymbol{x} \in \Omega. \\ +1, & \text{if } \boldsymbol{x} \in \mathbb{R}^2 \setminus \Omega. \end{cases}$$



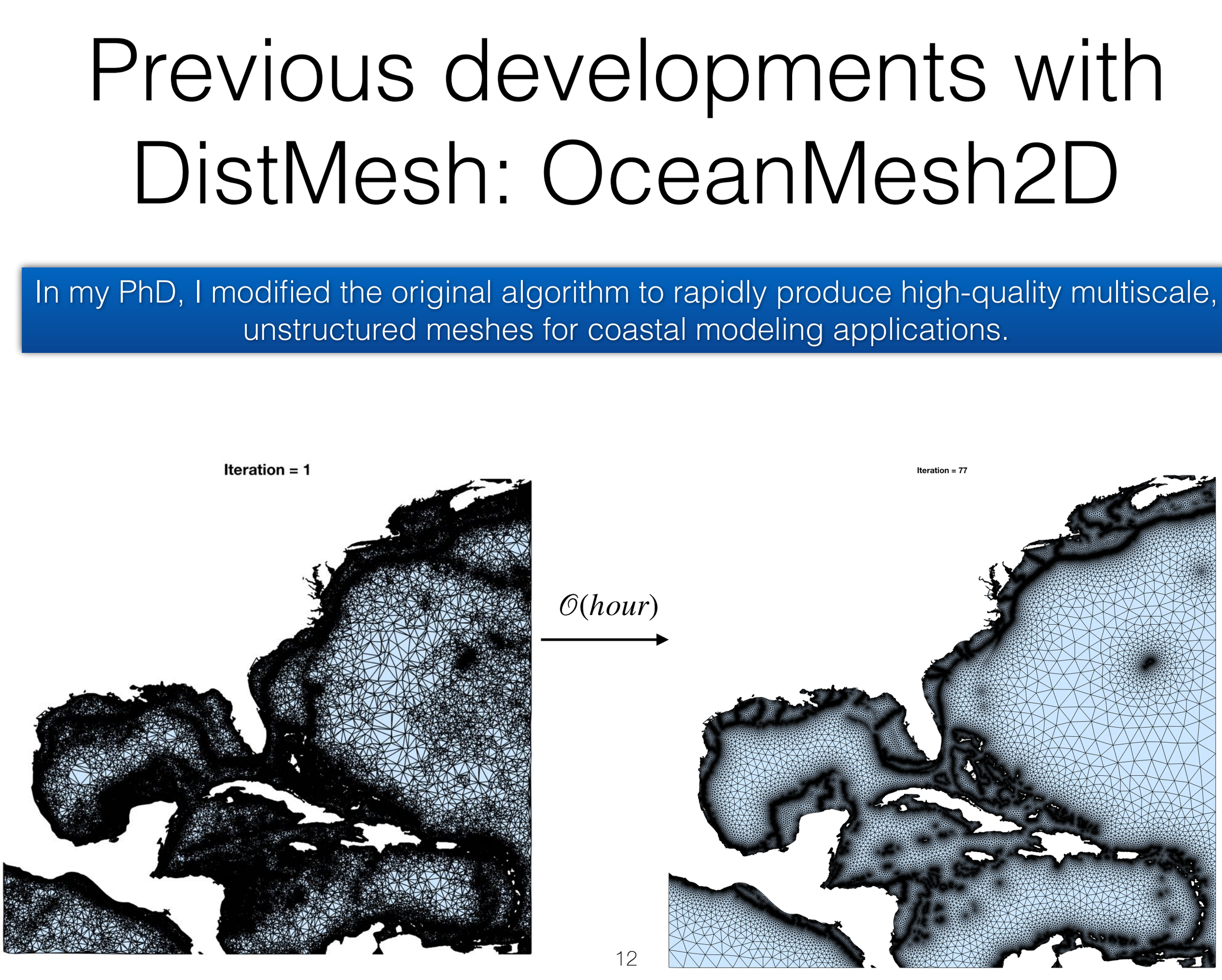
$s(\mathbf{x})$ = area/volume defined by polygon

$$\Omega := \left\{ \boldsymbol{x} \in \mathbb{R}^2 : d(\boldsymbol{x})_\Omega \leq 0 \right.$$

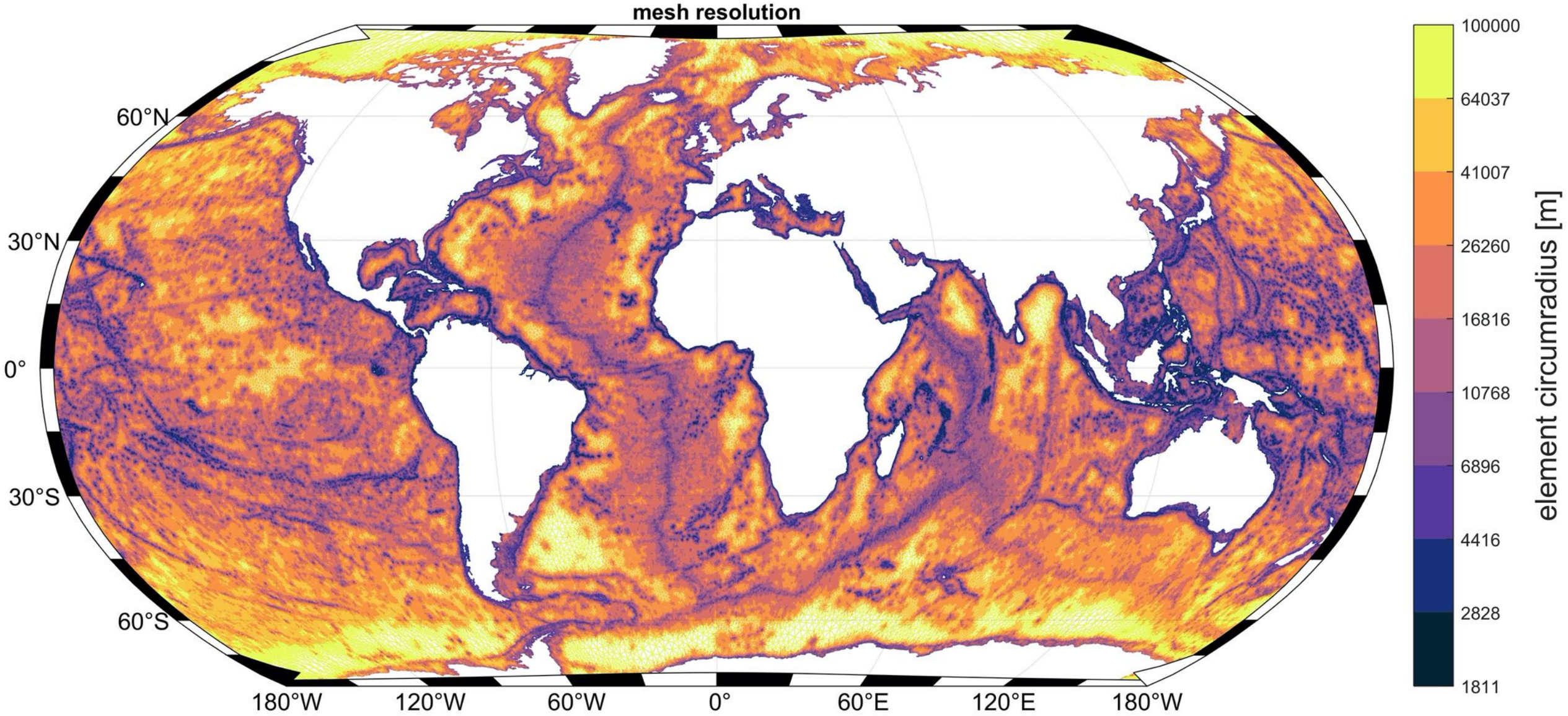
$$\partial\Omega := \left\{ \boldsymbol{x} \in \mathbb{R}^2 : d(\boldsymbol{x})_\Omega = 0 \right.$$

Previous developments with DistMesh: OceanMesh2D

In my PhD, I modified the original algorithm to rapidly produce high-quality multiscale, unstructured meshes for coastal modeling applications.

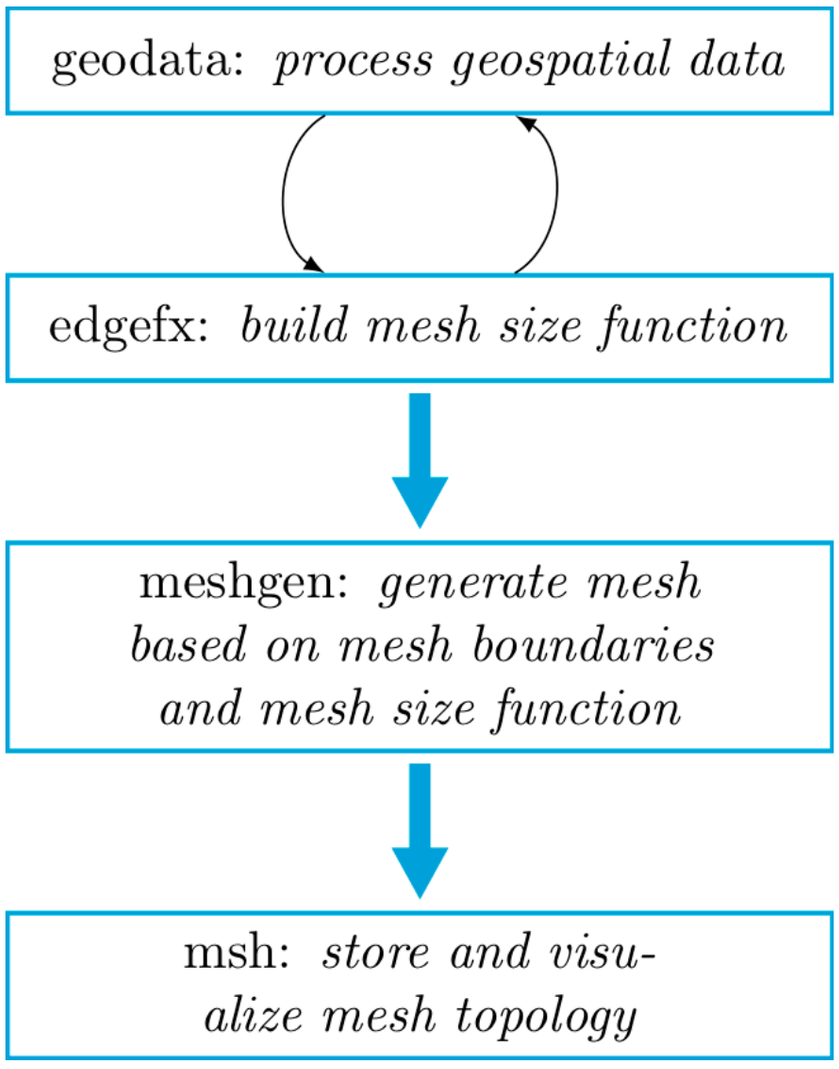


Previous developments with DistMesh: OceanMesh2D



Automatic mesh generation workflows

Workflows can be scripted at a high level



Software: https://github.com/CHLNDDEV/OceanMesh2D/tree/Projection

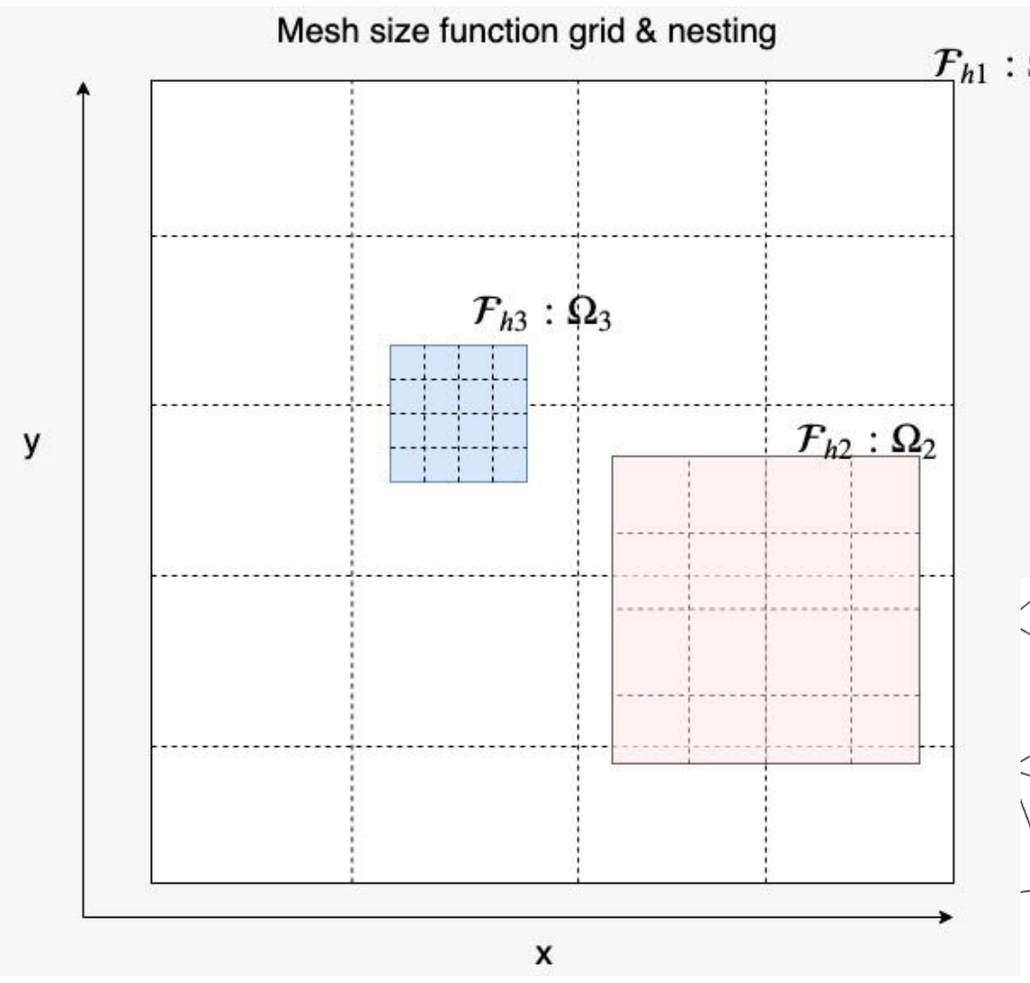
User-guide: Roberts, K. J., Pringle, W. J, 2018. OceanMesh2D: User guide - Precise distance-based two-dimensional automated mesh generation toolbox intended for coastal ocean/shallow water. https://doi.org/10.13140/RG.2.2.21840.61446/2.

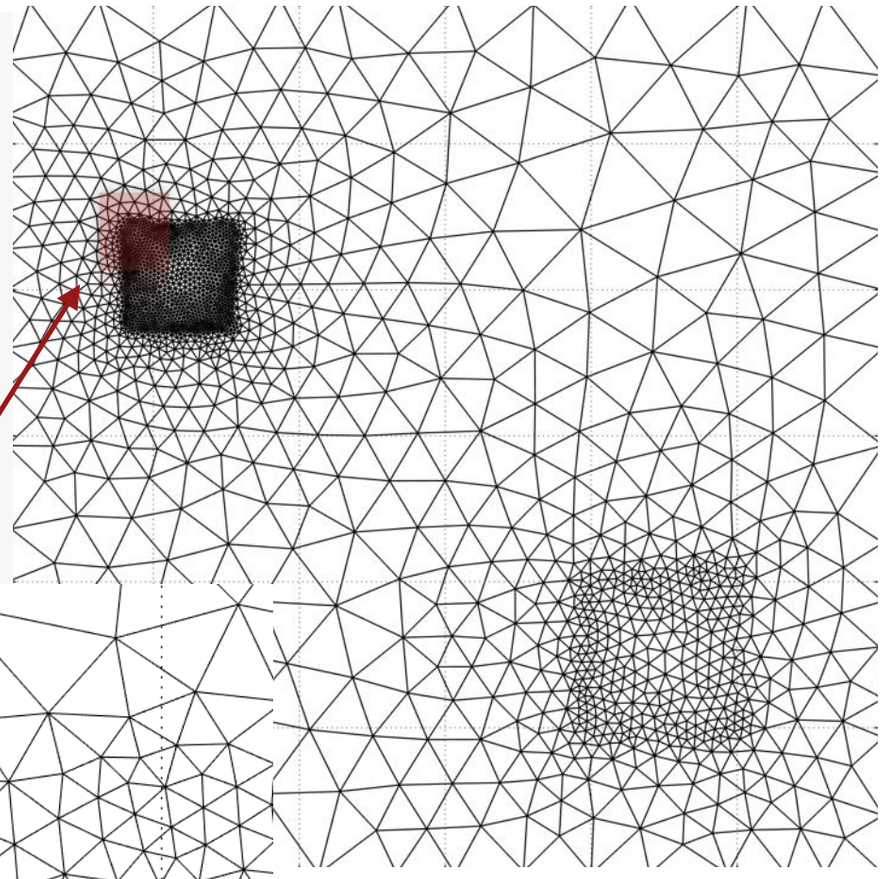
Paper: K. J. Roberts, W. J. Pringle, and J. J. Westerink. Oceanmesh2d 1.0. Matlab-based software for two-dimensional unstructured mesh generation in coastal ocean modeling. Geoscientific Model Development , 12(5):1847–1868, 2019.

Mesh size functions

Modifying size functions from OceanMesh2D

- Functions can be nested with different options.

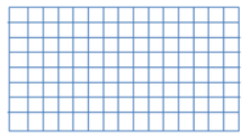


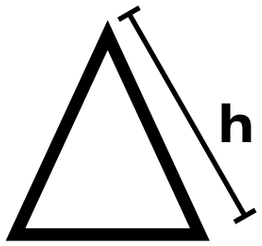


Meshes automatically “blend" into each other

Edgefx: Mesh sizes are controlled via sizing functions

$$h(x, y, z) \rightarrow h$$

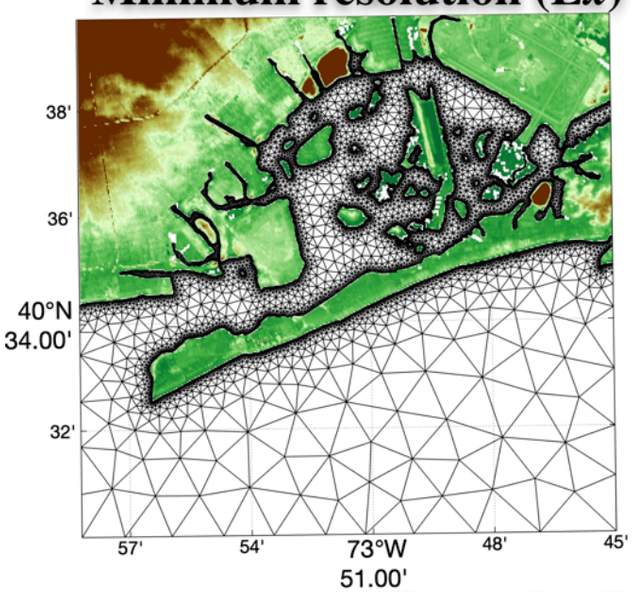




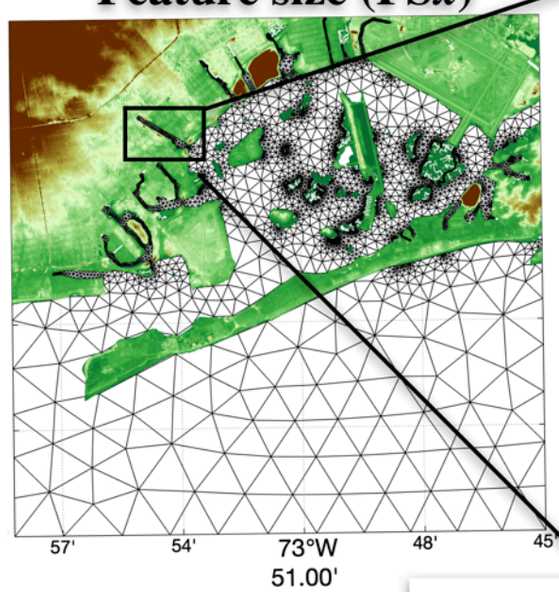
Combinations of mesh size functions are created

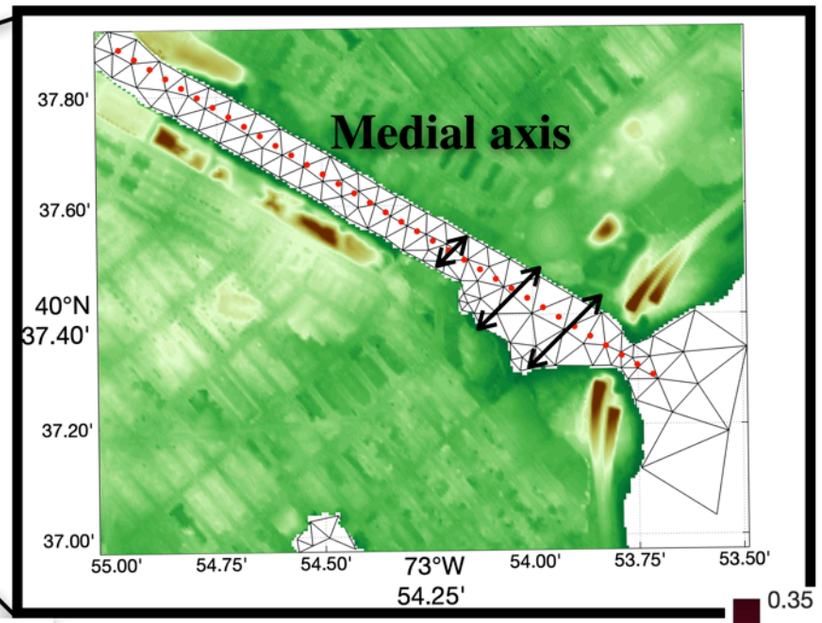
$$h = \min\left[(h_{Lx} \text{ or } h_{Fsx}), h_{Sx}, h_{Cx}\right] \text{ with } g \leq G$$

Minimum resolution (L*x*)

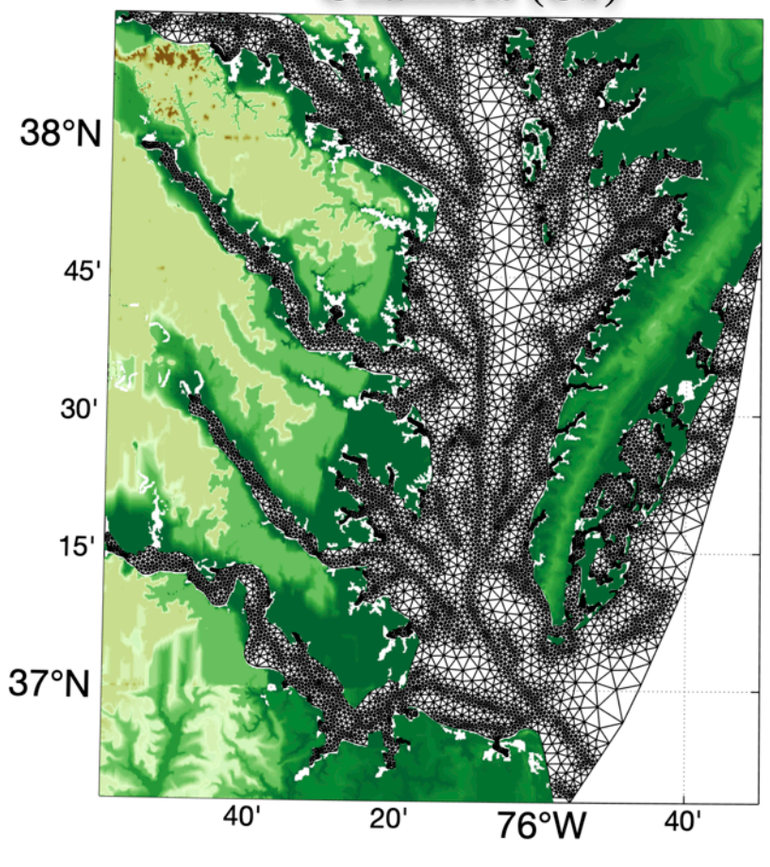


Feature size (FS*x*)



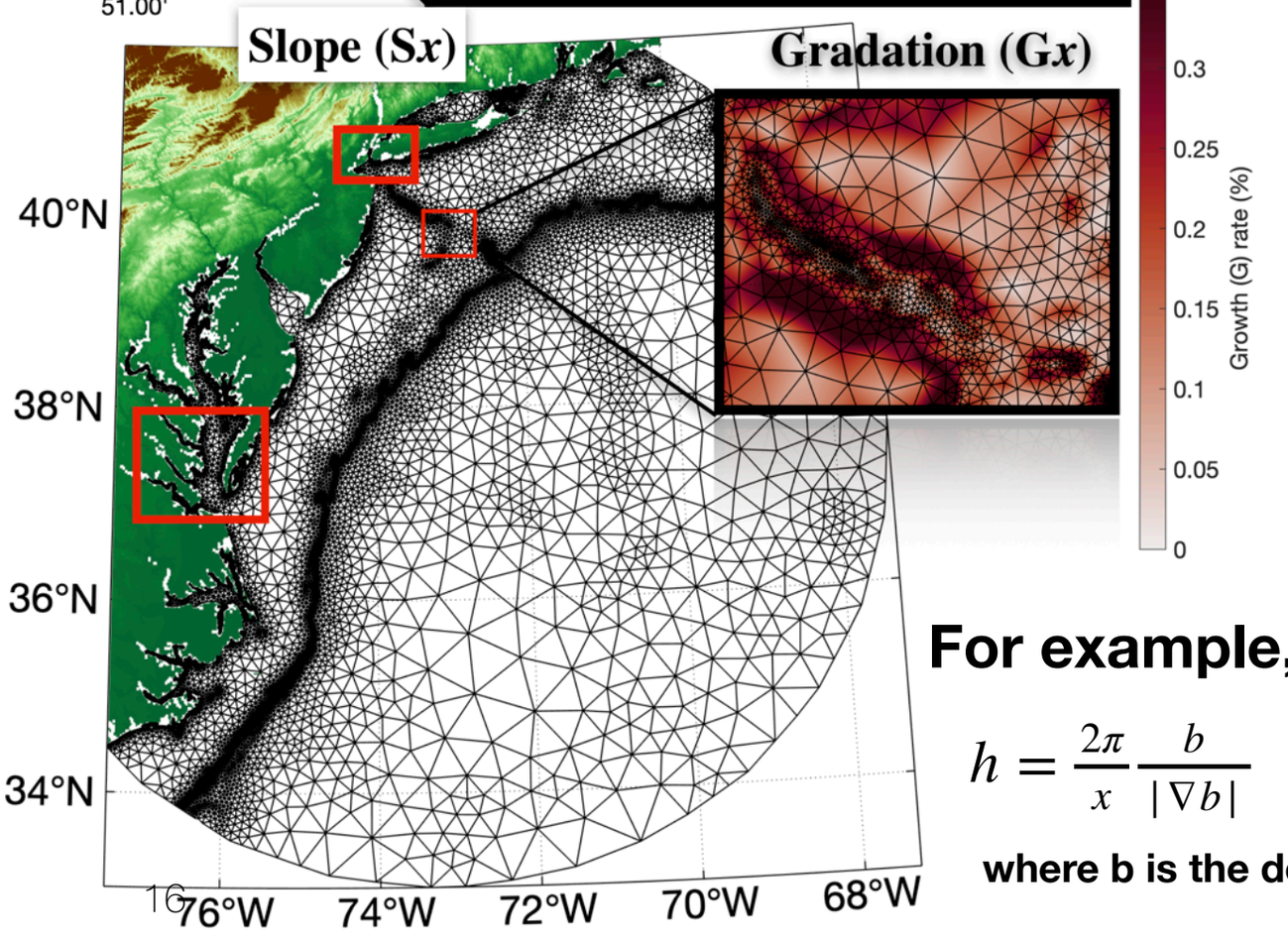


Channels (C*x*)



Slope (S*x*)

Gradation (G*x*)



For example,

$$h = \frac{2\pi}{x}\frac{b}{|\nabla b|}$$

where b is the depth

Size gradation control

Gradient-limiting mesh size functions based on the work of:

P.-O. Persson, 2006. Mesh size functions for implicit geometries and PDE-based gradient limiting. Engineering with Computers.

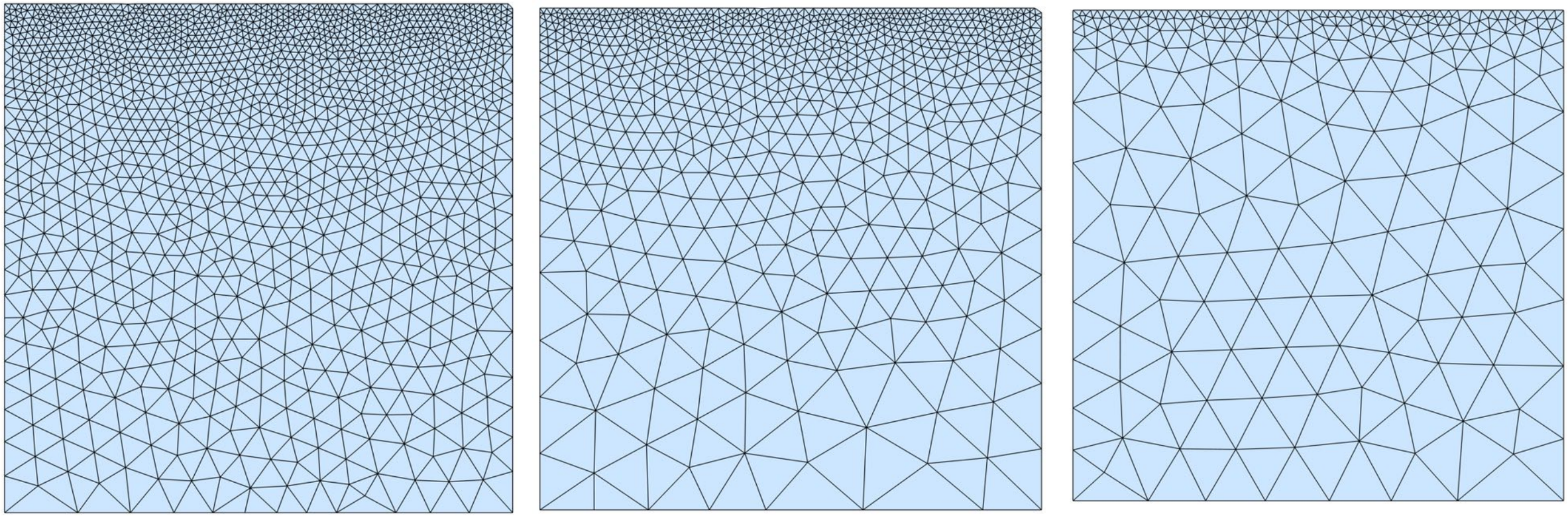
$$\frac{\partial h}{\partial t} + |\nabla h| = \min(|\nabla h|, g(x, y, z))$$

h length of triangle's edge

Note when $\nabla h \leq g$ **then** $\frac{\partial h}{\partial t} = 0$

Grade bound is enforced on mesh size function

Low gradation ⟶ Steep gradation



Mesh size functions for seismic imaging

Resolution coarsening in sponge-absorbing layer.

Adapting resolution to acquisition geometry.

Altering elemental sizes and grading rates based on FWI model updates, CFL considerations, and existing velocity models.

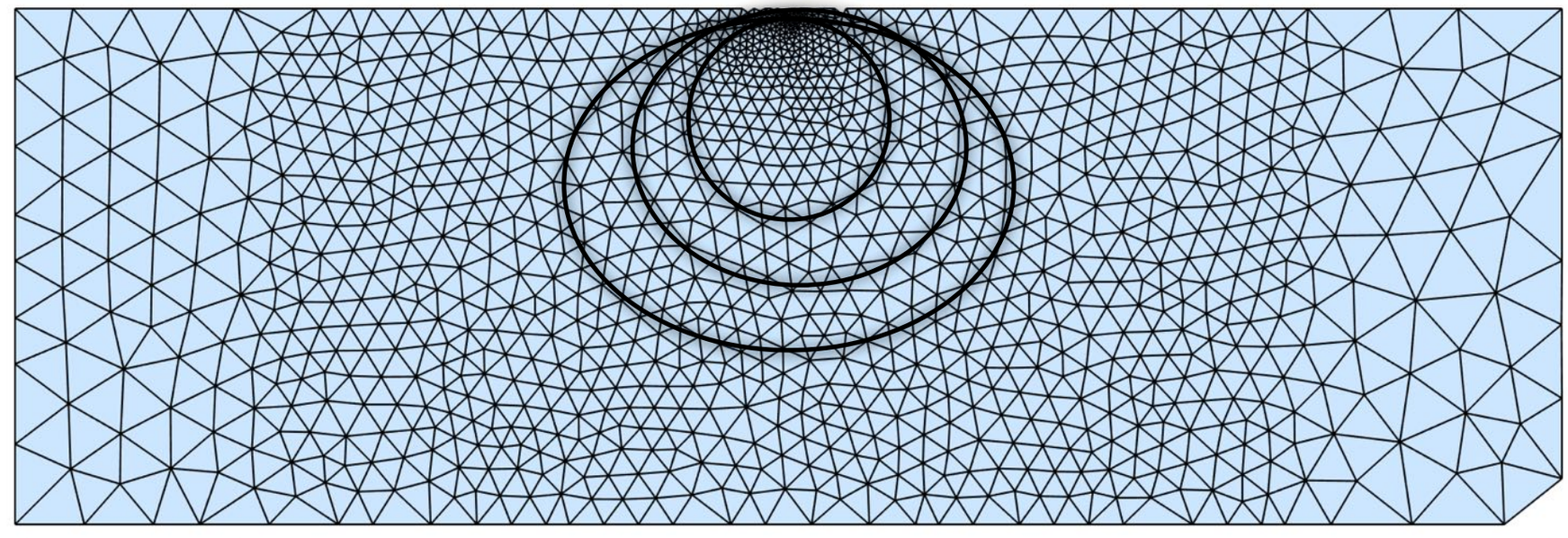
Support for anisotropic elements

Related work: Roberts, K. J., Pringle, W. J., Westerink, J. J., Contreras, M. T., & Wirasaet, D. (2019, May 10). On the automatic and a priori design of unstructured mesh resolution for coastal ocean circulation models. https://doi.org/10.31223/osf.io/nwde7

Mesh size functions

An adapted mesh for each shot

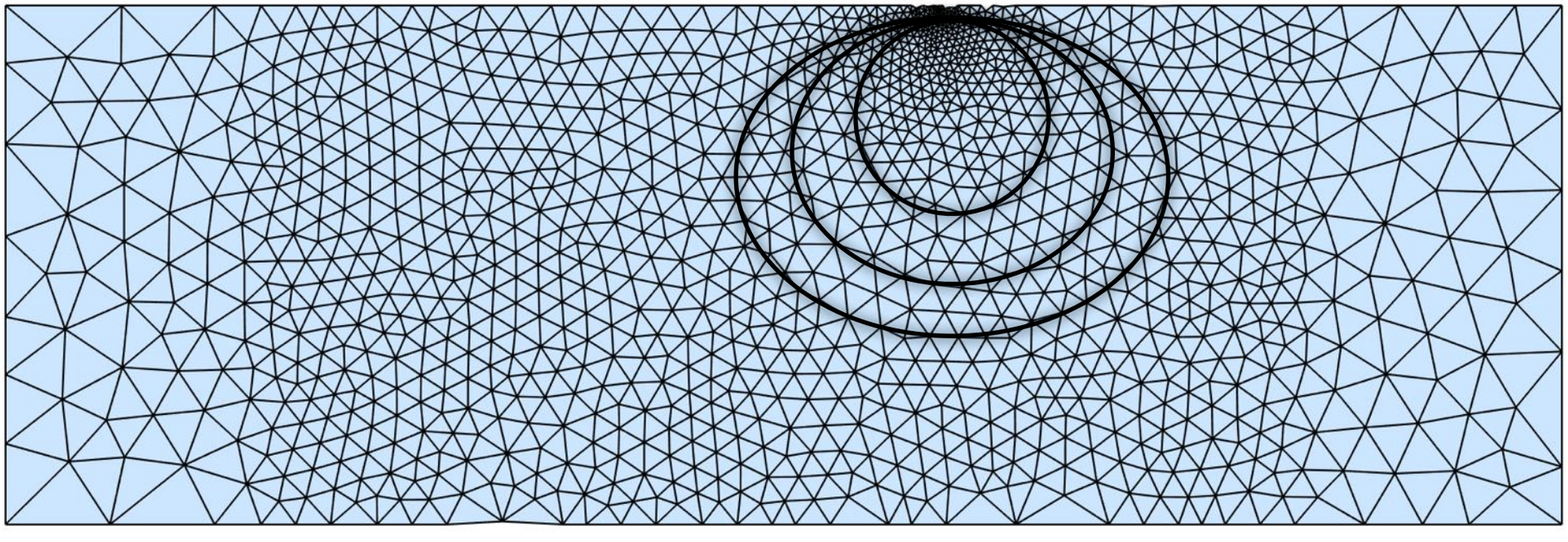
Shot #1



Sponge

Sponge

Shot #2



Adapting meshes

Finite elements for FWI present “offline” mesh adaptation opportunities.

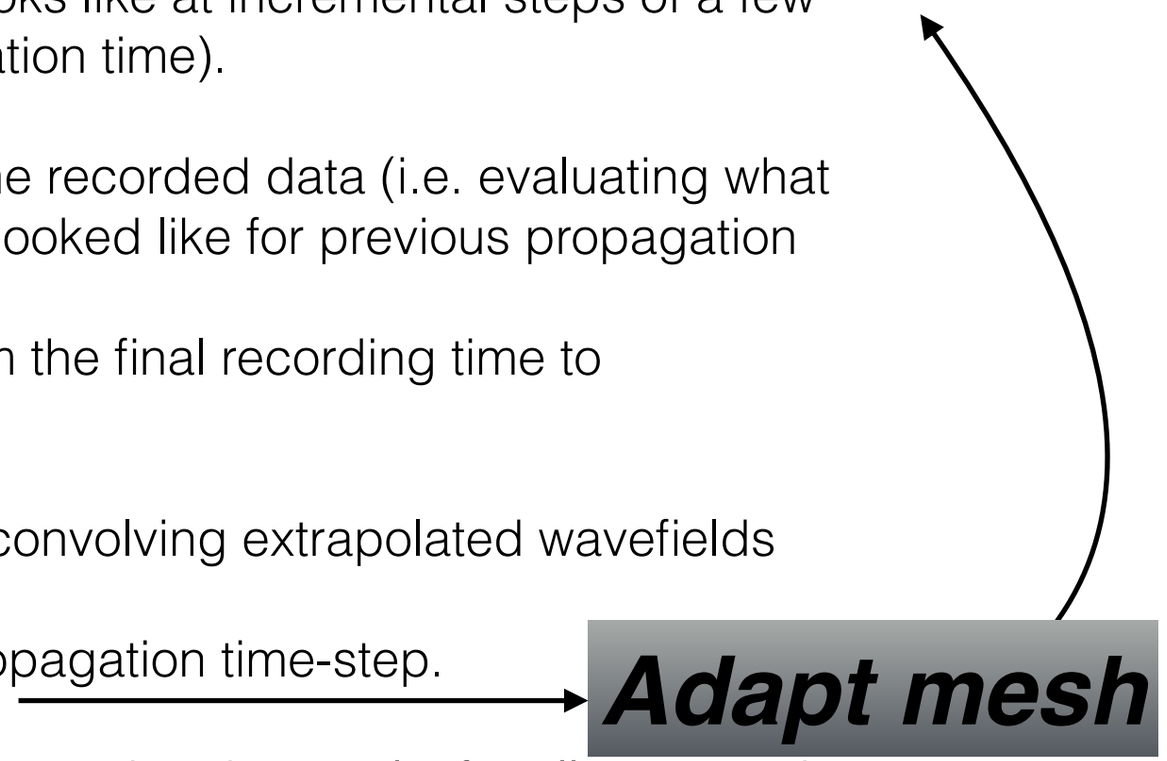
Recall,

1. **Forward modeling** a synthetic shot (i.e. evaluating what the expanding wavefront looks like at incremental steps of a few milliseconds of propagation time).

2. **Back-propagating** the recorded data (i.e. evaluating what the recorded field data looked like for previous propagation times,
working backwards from the final recording time to time zero).

3. **Imaging condition:** convolving extrapolated wavefields together at
each corresponding propagation time-step.

4. **Storing**: intelligently summing the results for all propagation steps.



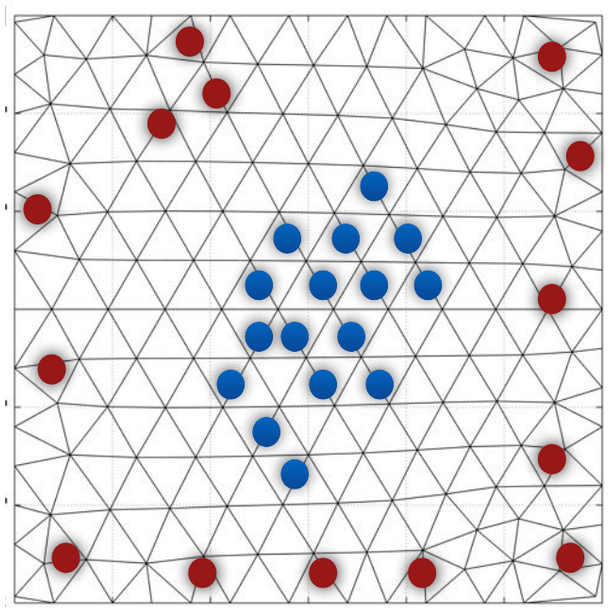
How does mesh adaptation in a realistic problem alter the subsurface imaging process?

Strategy to support mesh adaptation

Edge splits and collapses and valency reductions.

- Performed when model update becomes available

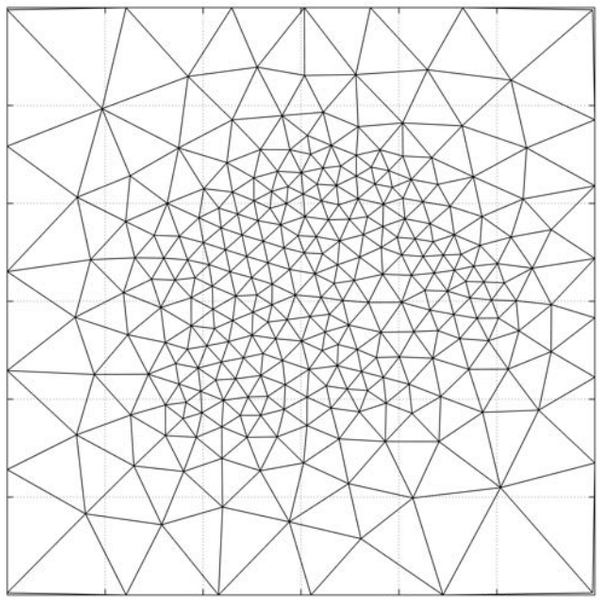
FWI Iter.=1



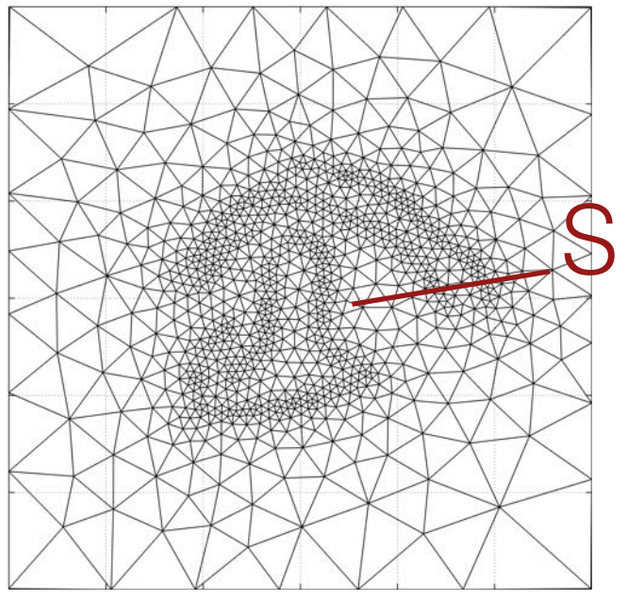
Edge collapses

Edge splits

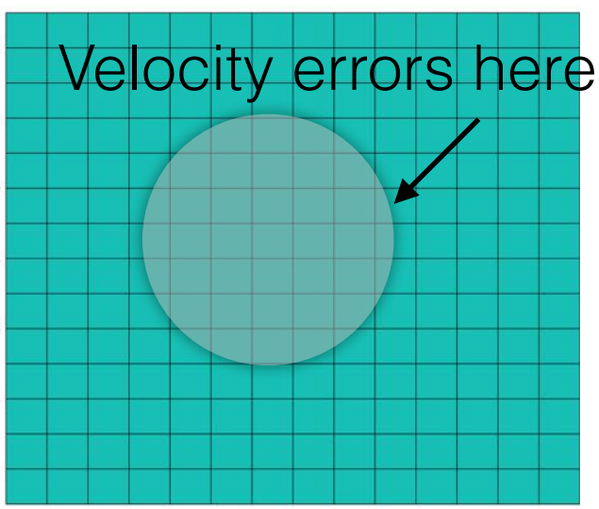
FWI Iter.=2

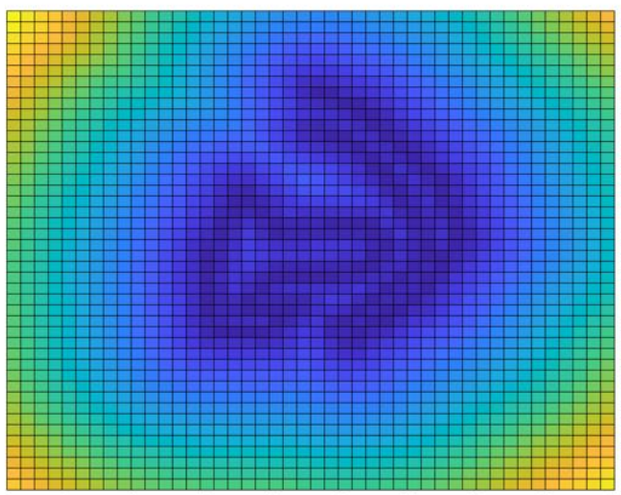


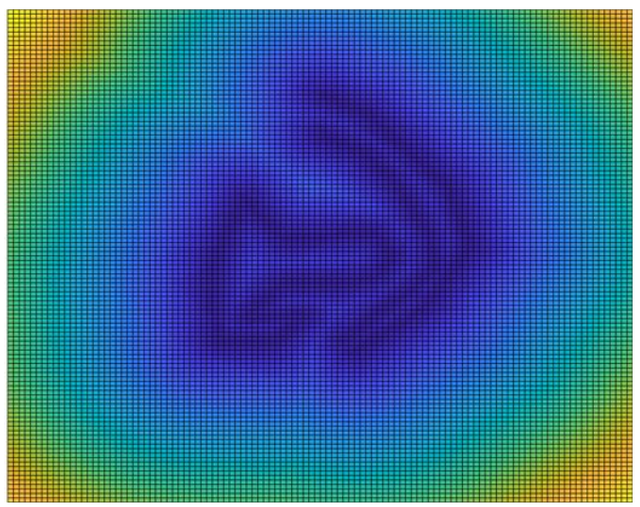
FWI Iter.=3



Mesh size functions







Strategy to adapt meshes

Recall, Step 2. Distribute a set of vertices interior to the domain.

If we start DistMesh from a “good” initial point set, then convergence will be rapid.

1. The mesh size function is edited where model updates are required (these model updates come out during FWI).

2. New points are injected where mesh size function has changed.

3. Perform a sequence of mesh improvement strategies on the **existing point distribution** and meshing iterations incrementally move vertices to achieve a new force equilibrium**.**

4. The strategies used in step 2) are repeated periodically until convergence criteria is met.

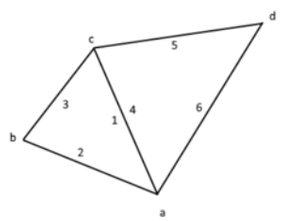
Anisotropy

Both size and direction of elements can be controlled

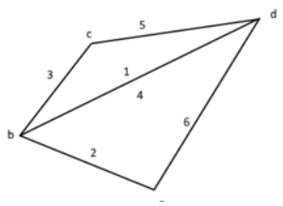
- Size functions becomes a tensor.
- Mesh size generation is accomplished with edge flips when Delaunay criterion is evaluated in metric space.
- Use PRAGMATIC.

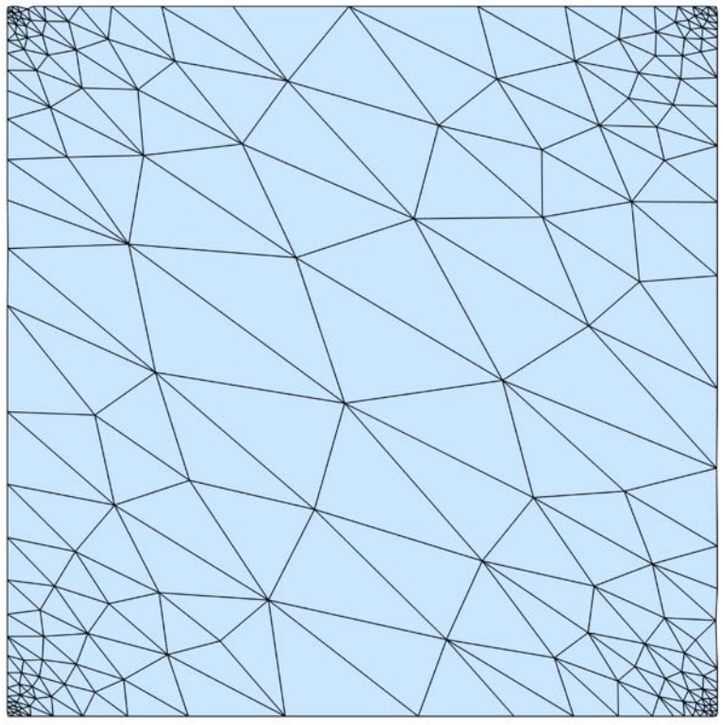
$$M = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

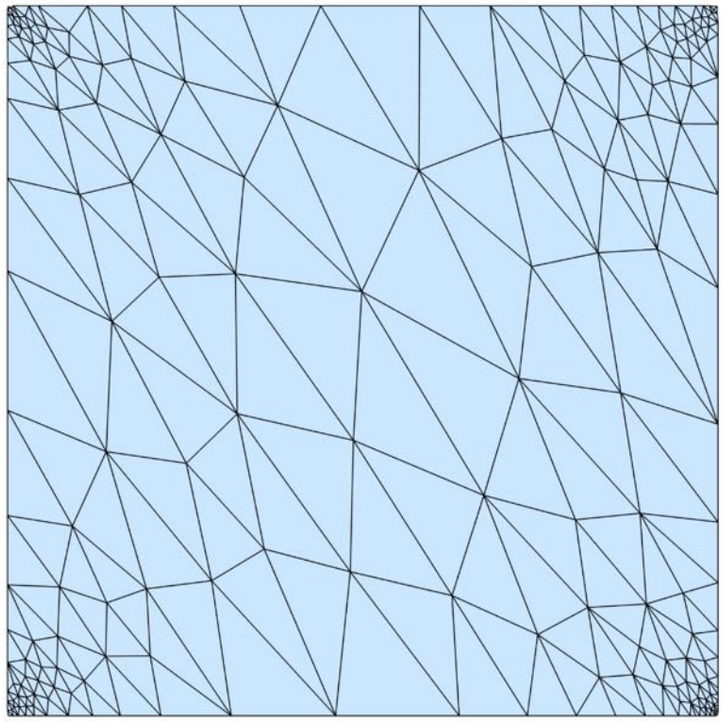
$$d(\mathbf{x},\mathbf{y}) = \sqrt{(\mathbf{x}-\mathbf{y})^T \mathbf{M}_{\text{avg}} (\mathbf{x}-\mathbf{y})}$$

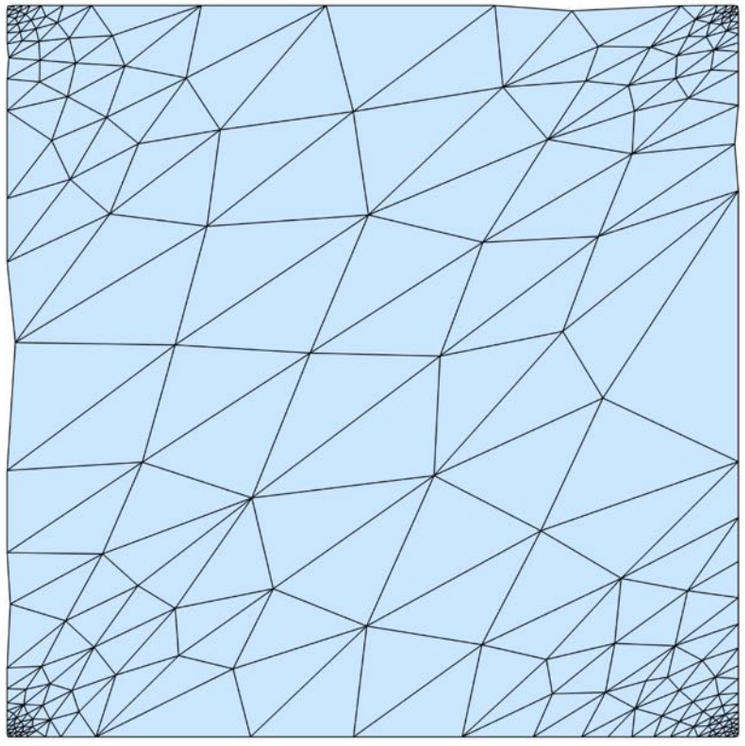


$$(a \times b)(c^T M\ d) + (a^T M\ b)(c \times d) > 0$$





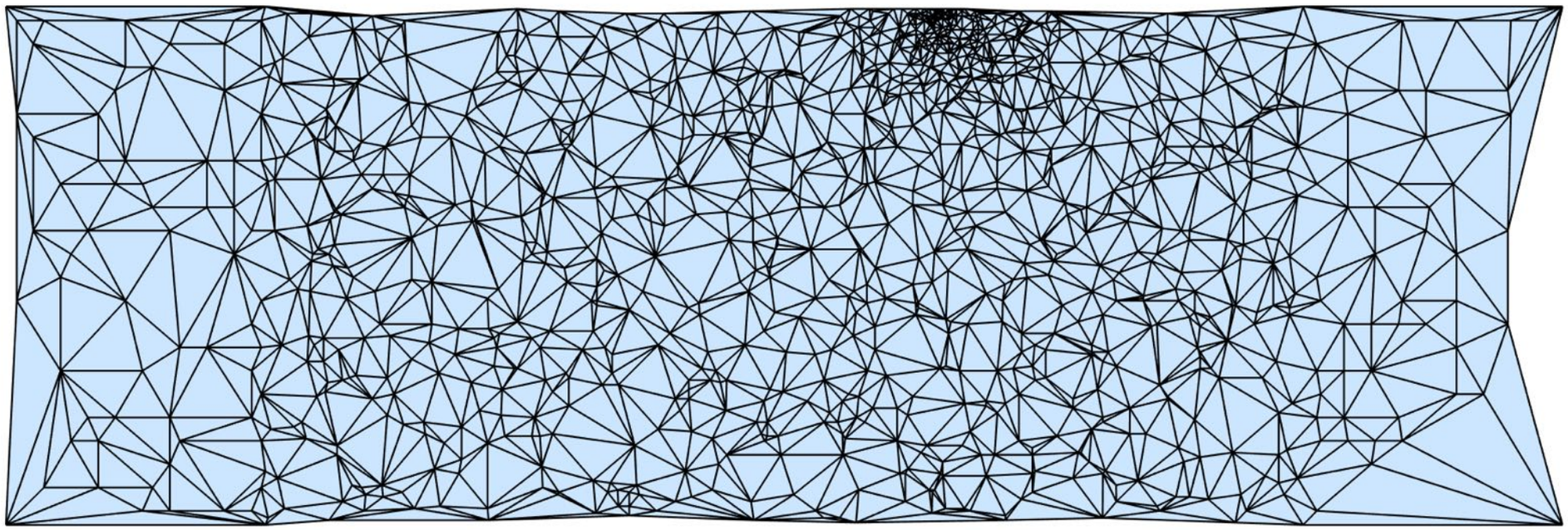




Parallelizing DistMesh

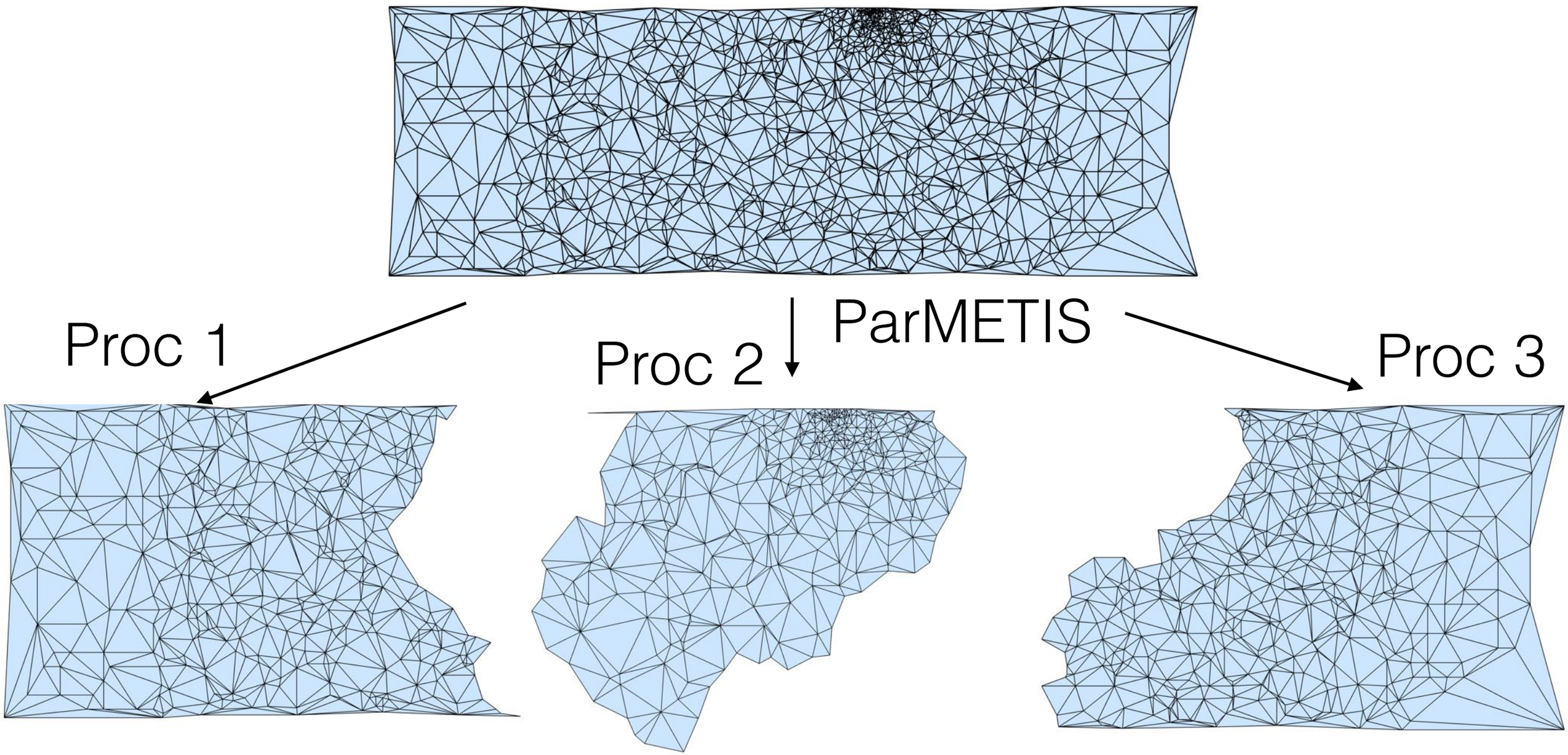
- Recall, Step 2. Distribute a set of vertices interior to the domain.

- This step can be done as a serial pre-processing step.



Parallelizing DistMesh

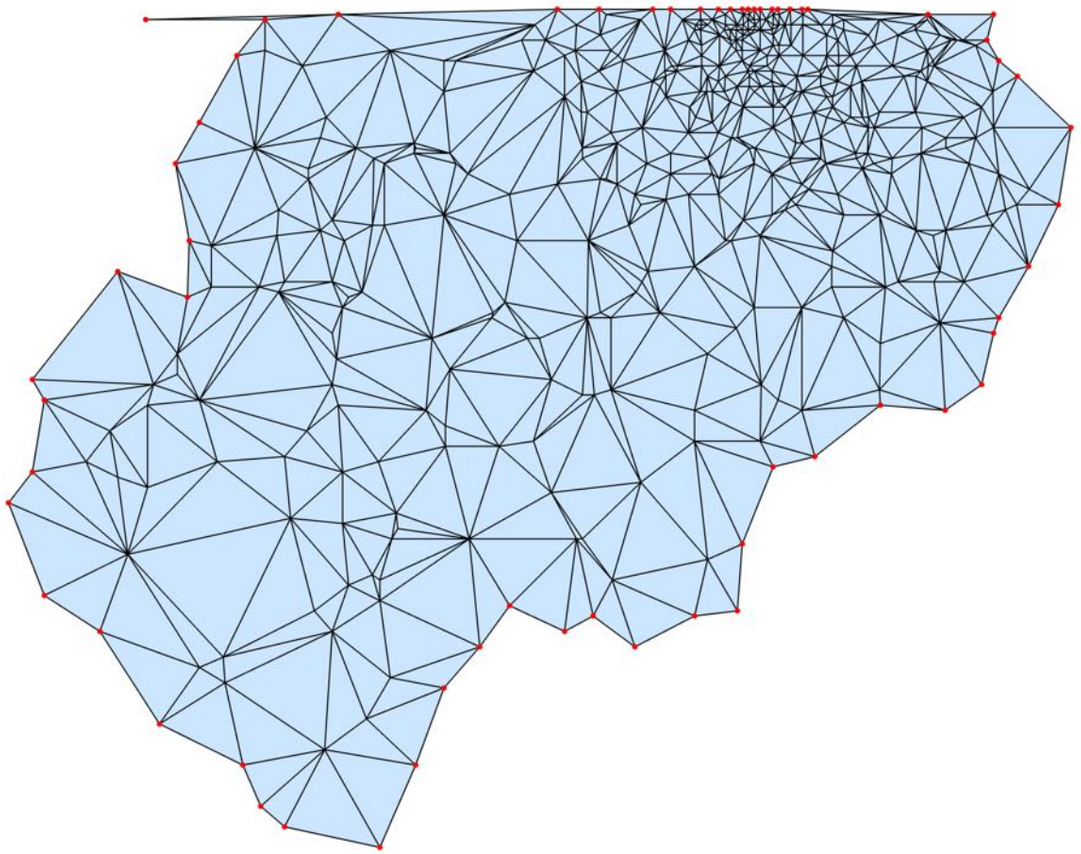
- Global triangulation is broken into subdomains.
  - This occurs as a pre-processing step
- Each subdomain executes its own problem.



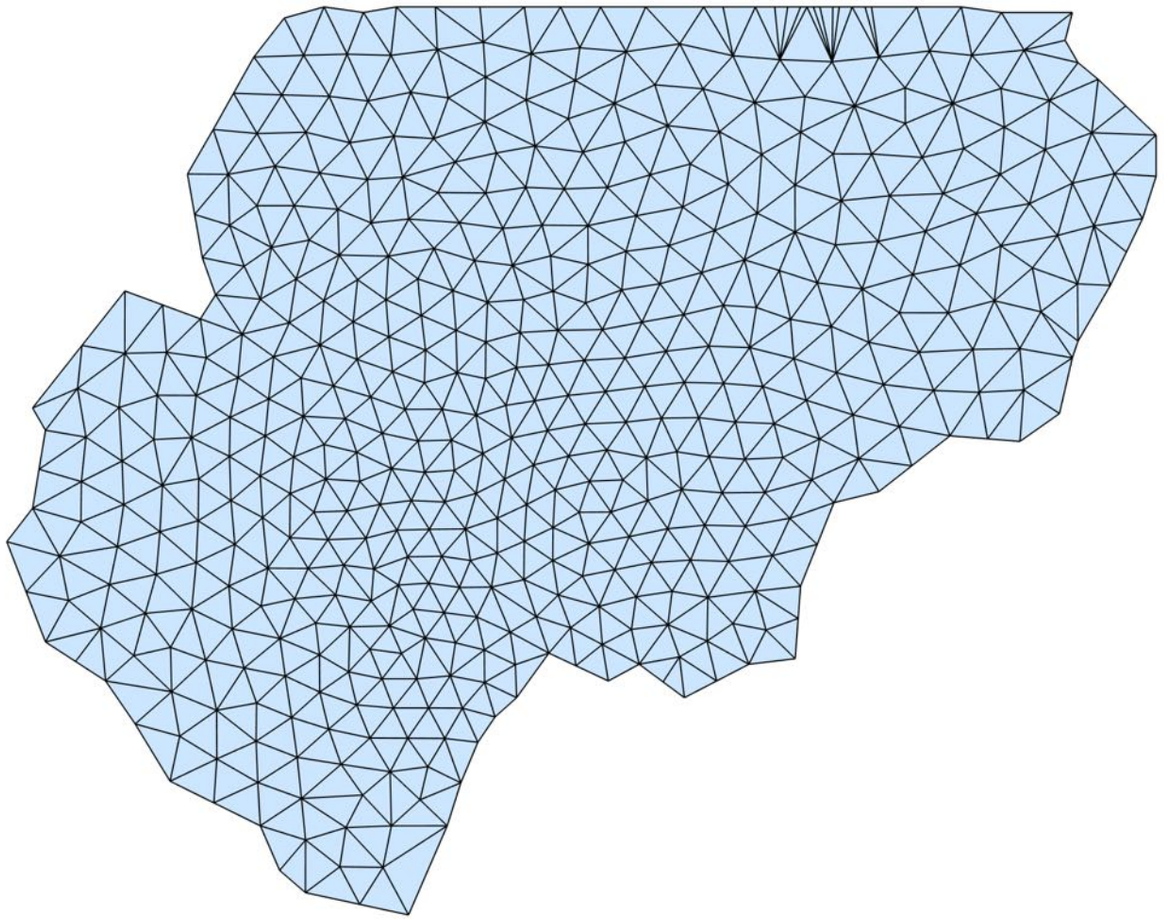
Parallelizing DistMesh

- Global triangulations are broken into subdomains
  - This occurs as a pre-processing step
- Subdomain boundaries/separators are “fixed” in space

Before



After



Merging subdomains entails: t = [t1; t1+length(p1)];
p = [p1; p2]; p=unique(p);

Parallelizing DistMesh

- This initial triangulation reflects the “true” vertex density.

  - Good load balancing.

- Full reuse of serial DistMesh algorithm.
  - Each subdomain operates as if the subdomain is independent of its neighbors.

    - Minimal inter-processor communication

      - Good scalability?

Summary of ideas

- **Port DistMesh into a HPC compatible environment.**
  - Serial algorithm is written in modern FORTRAN.
  - Algorithm is “simple” —> easier to port.
  - Parallelize this algorithm using fixed points with a graph decomp.

- Develop **mesh size functions** in the Julia or Python language to port the mesh size functionality from OceanMesh2D.
  - Adapt functionality accordingly for seismic inversion problems.

- **Investigate role of mesh size resolution on FWI convergence and velocity model updates.**

Thanks for listening!