

Generation of Grid-mesh with Multi-refinements for Using MPAS to Predict Severe Weather over Multiple Regions

Shuxia Zhang¹ and Richard Carpenter²

¹Metropolitan State University, St. Paul, MN
(Shuxia.Zhang@gmail.com)

²DTN, Burnsville, MN
(Richard.Carpenter@dtm.com)



Introduction

- **Warn-on-Forecasting System (WoFS)** tasked for increasing lead time of accurate severe weather warnings.
- **The Rise of MPAS (Model for Prediction Across Scales)**
 - Efficient resource allocation through variable resolution
 - Versatile tool for operational meteorology and long-term climate research
 - Able to deliver accurate weather forecasts from local to global scales
- **Optimal grid-mesh essential** to increase the longer **lead time**
 - Enabling variable resolution in multiple regions
 - Efficient forecasts of the severe weathers
 - Better harness the benefit of MPAS cross-scale capability
- **3 km global MPAS runs – exciting, but computationally expensive**

JIGSAW (GEO) - A Powerful Mesh Generator

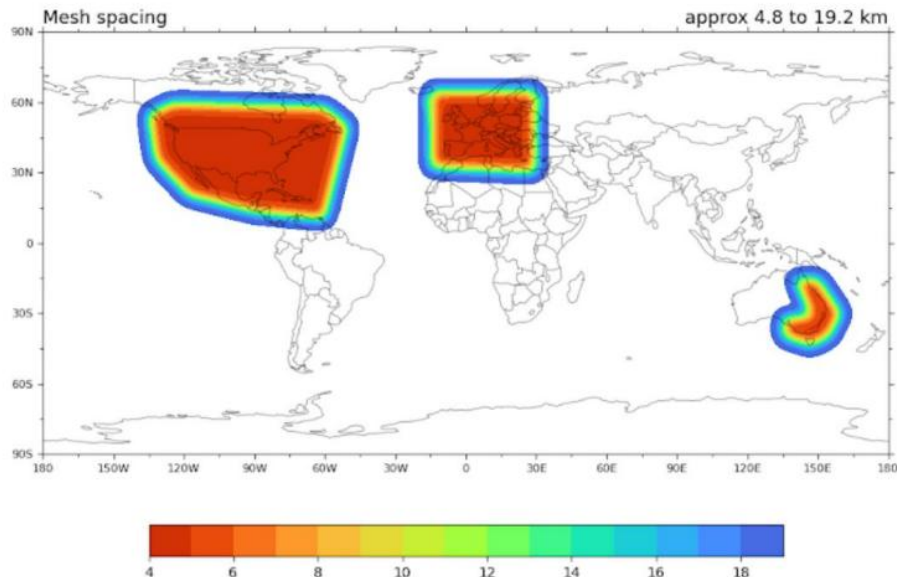
Code Author: **Darren Engwirda**

- Jigsaw – <https://github.com/dengwirda/jigsaw>
 - C++ Meshing library, version 1.0.0
 - Support of multiple threading through OpenMP
 - A fresh install is recommended over the conda installation
- Jigsaw-python - <https://github.com/dengwirda/jigsaw-geo-python>
 - Python interface for users to use Jigsaw library
 - Allow multiple-refinements of arbitrary shape
- The generated mesh consists of many triangular cells.
 - Output – VTK or GMSH, reformat is needed for MPAS
 - Mesh quality measures – **extremely important!**
 - TRISCR - *skewness and aspect-ratio of cells*
 - PWRSCR - *power-weighted centricity*
 - OBTUSE - *its largest interior angle*
 - TOPOL - *number of cells at a vertex*

Engwirda, D. (2017). *JIGSAW-GEO (1.0): Locally orthogonal staggered unstructured grid generation for general circulation modelling on the sphere*. Geoscientific Model Development, 10, 2117–2140.

Generation of customized grid mesh using JIGSAW

- The grid meshes provided by the MPAS team
 - Refinements are limited to simple circles and/or ellipses
- DTN grid mesh configurations using jigsaw (GEO)
 - 5km resolution in US, Europe and part of Australia,
 - 20km resolution elsewhere
 - Based upon case_4_.py - **Many thanks** to Darren Engwirda for some hints.



Modify struct `hmat` in case_4_.py

.....

`hmat.xgrid` - longitude

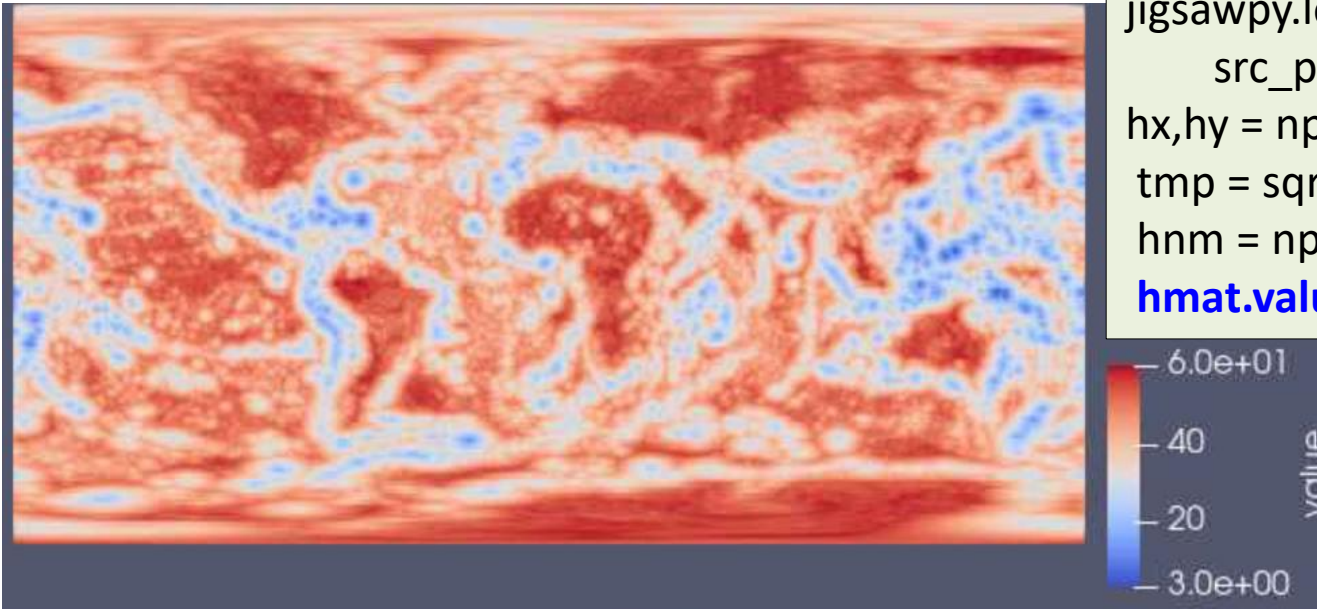
`hmat.ygrid` - latitude

`hmat.value` - resolution

Caveat: A refinement can be irregular shape, but must be mathematically continuous.

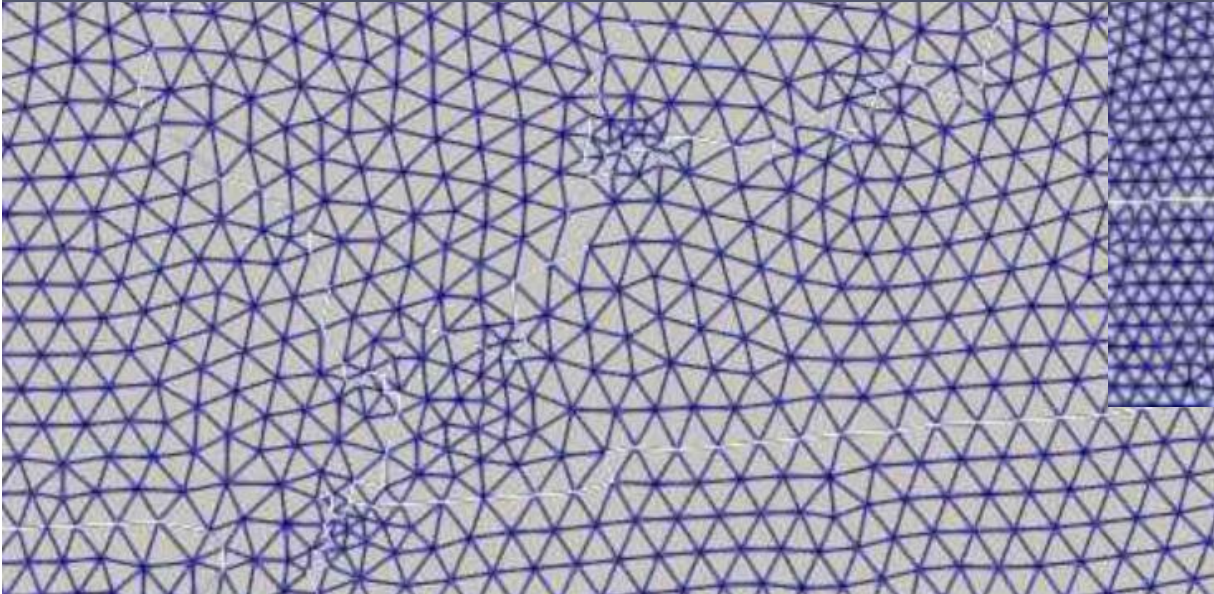
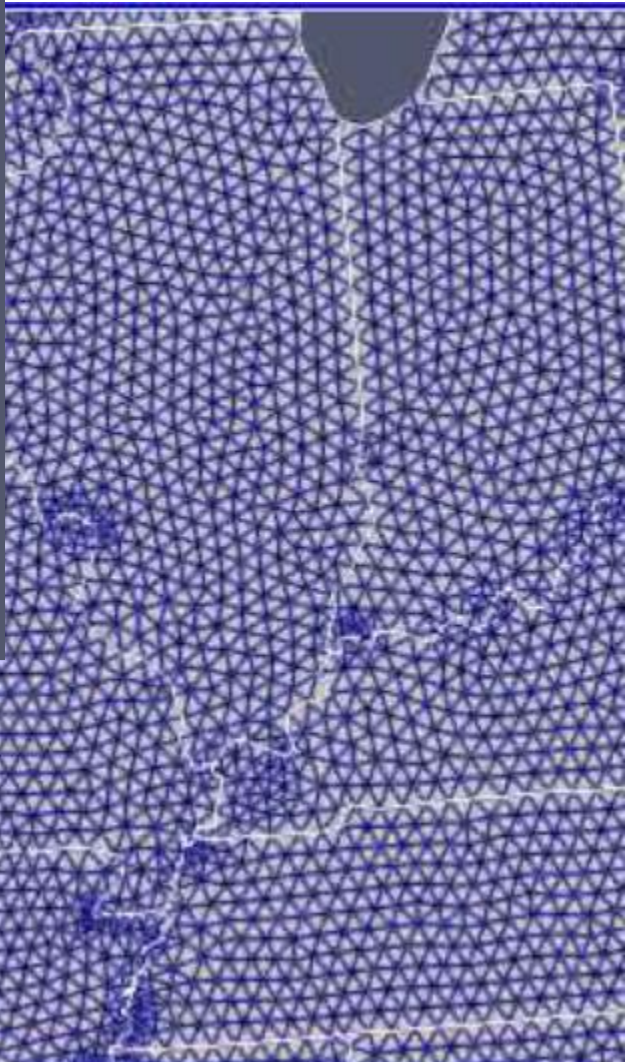
A Demo Case - Complex Grid Mesh

Grid resolution varies with local topographic gradient
(60 km, flat area), (3km, Steepest)



```
Modify hmat in case_4_.py  
.....  
jigsawpy.loadmsh(os.path.join(  
    src_path, "topo.msh"), topo)  
hx,hy = np.gradient(topo.value)  
tmp = sqrt(abs(hx)+abs(hy))  
hnm = np.max(tmp)  
hmat.value = 60-tmp*57/hnm
```

Metrix	Value	Implication
BISECT	7	Times of recursively splitting
TRISCR	0.91 ~ 0.98	Aspect ratio for all cells
PWRSCR	0.94 ~ 0.99	Size and shape well balanced
OBTUSE	0	All angles < 90 degree
TOPOL.	0.998	Ideal element's connectivity at a vertex



UIFCW25

A UFS Collaboration Powered by EPIC

Conversion of the JIGSAW mesh for MPAS Valid

- **Installation** of the Supportive Tools
 - Jigsaw, netcdf, mesh_conversion_tools, metis, paraview, xarray
- **Jigsaw_mesh_netcdf** - convert `mesh.msh` to `mesh_triangles.nc`
- Convert from triangles to MPAS valid mesh
Converter.x mesh_triangles.nc Voronoi_mesh.nc
- Decomposition of **Voronoi_mesh.nc** for parallel run of MPAS
**python processor_decompositions/make_partition_files.py **
-f Voronoi_mesh.nc -m gpmetis -b B
 - graph.info.part.B will be generated -
 - For scalability tests, we can run the partition for different B (512, 1024, 2048, ...)
- Then test/use the customized grid-mesh for MPAS
 - Following [MPAS Tutorial — Practice Session Guide](#).
 - Replace x1.10242.grid.nc with Voronoi_mesh.nc and the corresponding graph participation file

Conclusion

- **The presentation shared the key concept and procedure to generate customized grid-mesh for MPAS**
 - Objective – to help the community better harness the benefit of MPAS capability.
- **For [WoFS](#), this approach can be applied for generating regional grid:**
 - More accurately capture local geographic features
 - Dynamically refine as weather conditions evolve.
- **Potential Extensions:**
 - Incorporate real-time radar and satellite data
 - Leverage high-resolution elevation models
 - Use for Weather AI and Machine Learning
- **Acknowledgment:**
 - Many thanks to **Darren Engwirda** for Jigsaw (GEO) and answering questions via emails.
 - Many thanks to [Minnesota Supercomputing Institute](#) for the computing facilities used in this work.



Elapsed Time and Memory Consumption MS Copilot Projection

Here's a back-of-the-envelope comparison for a 28-day MPAS-Atmosphere run on Cheyenne (36 864 cores), contrasting a uniform 3 km global mesh with a variable-resolution mesh that nests 3 km over a limited region and uses 15 km elsewhere.

Configuration	3 km Area Fraction	Total Cells (rel.)	Core-hours	Wall-clock Time	Speedup vs Uniform
Uniform 3 km	100 %	1.000	2.70 M	74 h	1 ×
Regional 3 km (10 %) + 15 km (90 %)	10 %	0.137	0.37 M	10 h	7.4 ×
Regional 3 km (20 %) + 15 km (80 %)	20 %	0.235	0.63 M	17 h	4.3 ×
Regional 3 km (30 %) + 15 km (70 %)	30 %	0.324	0.88 M	27 h	2.7 ×

$$\frac{N_{\text{var}}}{N_{\text{uni}}} = \frac{f/3^2 + (1-f)/15^2}{1/3^2}$$

High-res fraction f	Total cells ratio	Memory ratio	Reduction vs uniform
100 %	1.000	100 %	0 %
10 %	0.136	13.6 %	86.4 %
20 %	0.232	23.2 %	76.8 %
30 %	0.328	32.8 %	67.2 %

$$\text{Memory}_{\text{regional}} = \left[f + (1-f) \times \left(\frac{3}{15} \right)^2 \right] \times \text{Memory}_{\text{uniform}}$$

