




UXarray: Extending Xarray for Enhanced Support of Unstructured Grids

John Clyne[1], Hongyu Chen[5], Philip Chmielowiec[1], Orhan Eroglu[1], Cecile Hannay[2], Robert Jacob[4], Rajeev Jain[4], Brian Medeiros[2], Paul Ullrich[5], Colin Zarzycki[3]

(and many, many more!)

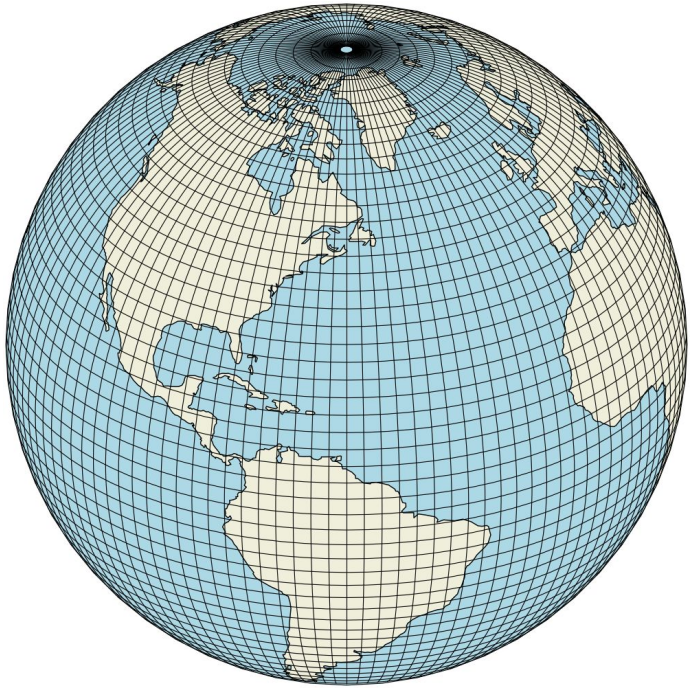
[1]NSF NCAR CISL [2]NSF NCAR CGD [3]Pennsylvania State University [4]Argonne National Laboratory [5]University of California, Davis

May 2, 2025

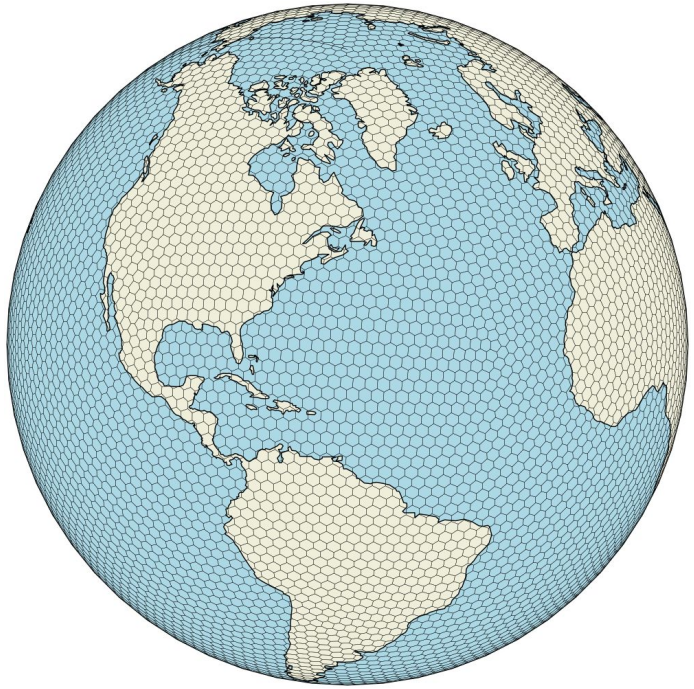
This material is based upon work supported by the National Science Foundation under Grants No.2126458 and 2126459

This material is based upon work supported by the National Center for Atmospheric Research, which is a major facility sponsored by the National Science Foundation under Cooperative Agreement No. 1852977.

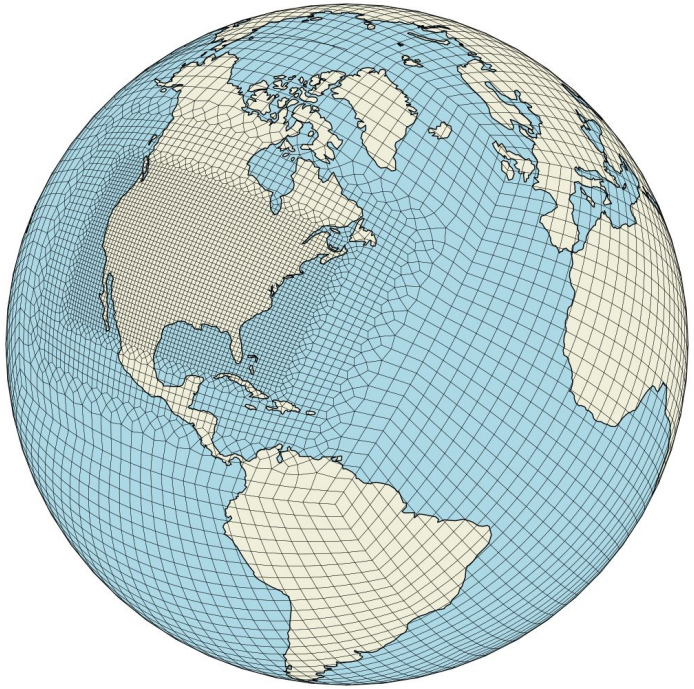
Motivation: Arrival of kilometer-scale global models for weather and climate



Structured “lat-lon” grid



Voronoi grid



Variable resolution, cube-sphere

Next generation, kilometer-scale climate and global weather models

After nearly two decades of development and evaluation, the climate and global weather modeling communities are transitioning from simple structured grids to more complex, but more scalable **unstructured grids** upon which governing equations of state are solved.

Challenges

- No widely used convention for the storage of unstructured grid data
  - Though, UGRID is being adopted by some: https://ugrid-conventions.github.io/ugrid-conventions/
- Few analysis tools are capable of working directly with unstructured mesh data
  - Most, if not all, developed for a particular model's output only
  - Resampling to structured grids is common practice, but has myriad pitfalls
- Global storm resolving resolution models can generate LOTS of data
  - This further exacerbates problems with limited set of existing tools that operate directly on unstructured meshes
- Data operators that have trivial and efficient implementations with structured grids can become complex and computationally expensive with unstructured meshes
  - E.g. Efficiently finding the grid cell containing a point

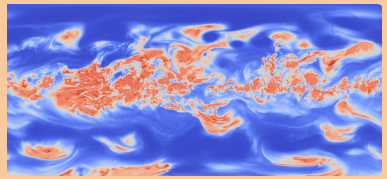
Goals

1. Develop **extensible**, **scalable**, open source software for analysis and visualization of unstructured grid data from ESS models

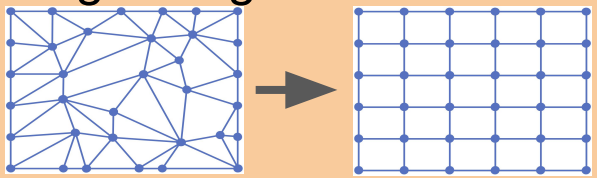
Computational operators

$\mathcal{F}(x)$ $\iint$ $\frac{\partial x}{\partial y}$ $\mathcal{L}(x)$? ?

Plotting



Regridding



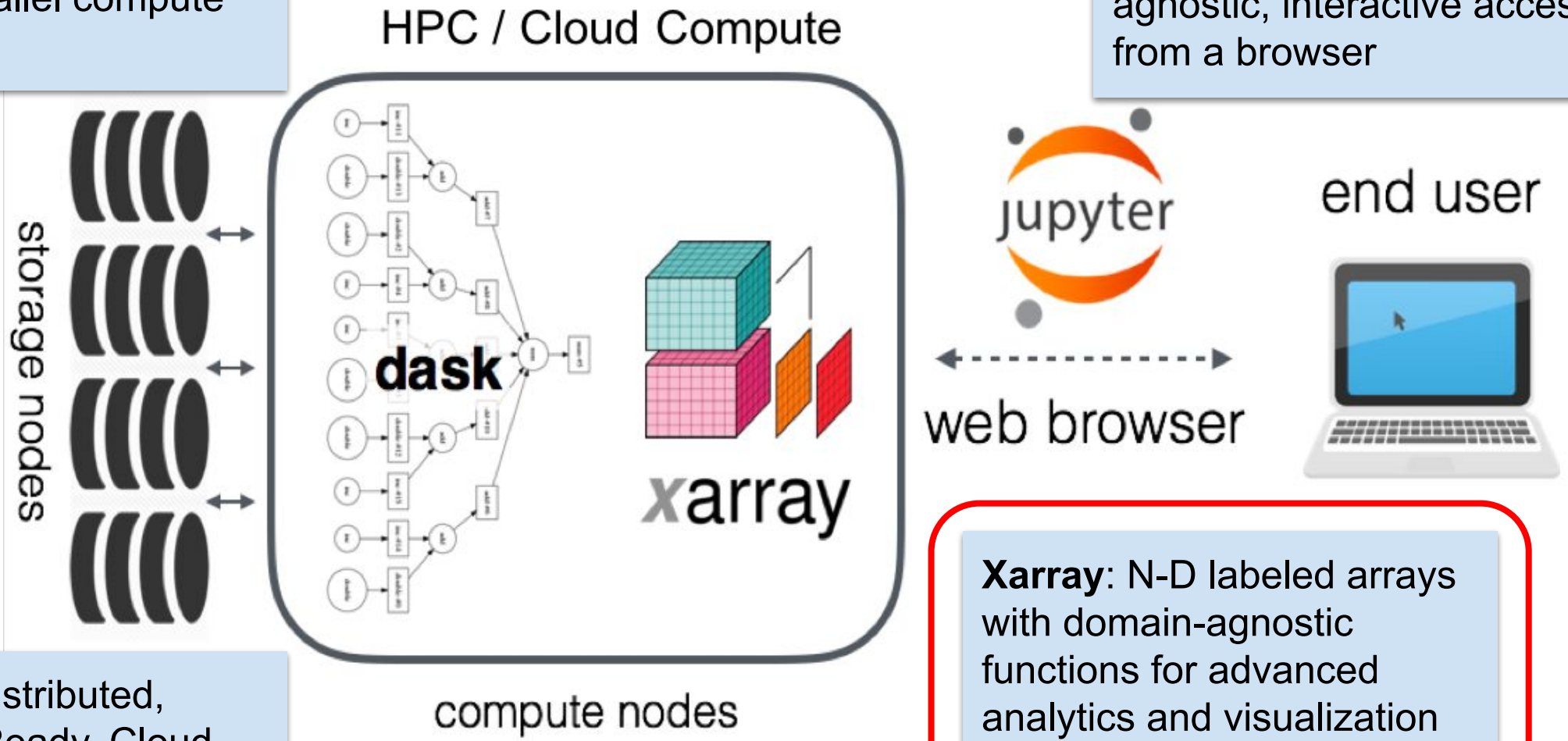
2. **Sustainable** and **community** owned

GitHub

Pangeo: A community platform for Big Data geoscience

Dask: parallel compute for Python

JupyterHub: platform agnostic, interactive access from a browser



Globally distributed, Analysis Ready, Cloud Optimized (**ARCO**) data

Xarray: N-D labeled arrays with domain-agnostic functions for advanced analytics and visualization

image source: https://pangeo.io/architecture.html

UXarray: Xarray + unstructured grid awareness

Supported Files Formats & Model Outputs

Unstructured grid file formats: UGRID, SCRIP, EXODUS

Unstructured grid model outputs: MPAS, CAM-SE, **ICON**, ESMF, GEOS, HEALPix, **FESOM2**

… also: Lat-Lon, Points

uxarray

UXarray

xarray

Xarray

xarray.Dataset

xarray.DataArray

inherits

inherits

uxarray.Grid

Unstructured grid topology in UGRID format
- Dimensions
- coordinates
- variables
- grid-specific functions

uxarray.UxDataset

Unstructured grid-aware dataset
- attributes
- operators

uxarray.UxDataArray

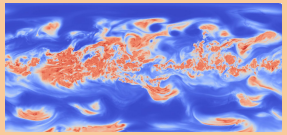
Unstructured grid-aware array
- attributes
- operators

has a

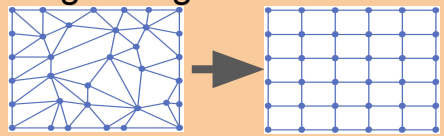
Computational operators

$\mathcal{F}(x)$ $\iint$ $\frac{\partial x}{\partial y}$ $\mathcal{L}(x)$

Plotting



Regridding



UXarray operates directly on Unstructured Grids (without regridding needed)

A sampling of UXarray's public class methods

```
# Open a dataset and analyze sea surface temperature
import uxarray as ux
uxds = ux.open_dataset(grid_file, data_file…)
grid = ux.open_grid(grid_file)
sst = uxds[`sst`]
```

Operators inherited as is from Xarray (grid info not needed!)

```
sst.argmin()
sst.mean()
sst.where()
```

Operators overridden from Xarray

```
sst.integrate()
sst.to_netcdf()
sst.plot()
```

<u>New</u> operators added by UXarray

```
grid.plot()
sst.plot.points()
sst.gradient()
sst.topological_mean()
```

Already using Xarray? UXarray should look pretty familiar!

Building the UXarray community

- **Everything on *public* GitHub**
- Open Source (Apache 2.0)
- **Built on Pangeo community AND Pangeo software stack**
- **Detailed contributors guide** + code of conduct
- Open discussion on all topics (GitHub Issues and Discussion forum)
    - Socialize major design decisions and roadmap
- **CI/CD infrastructure**
- Advocacy (conferences, discussion forums, events)
- A common programming language understood by scientists and software developers: **Python**
- **Lots of examples!**

UXarray Repository

Current status - Functionality (as of release 2025.4.0)

Supported model outputs / file formats:

- MPAS, CAM-SE, **ICON**, ESMF, GEOS, **FESOM2**, **HEALPix** models
- UGRID, SCRIP, EXODUS file formats
- Also structured grids and point-cloud data

Grid manipulation

- Metadata inspection
- Subsetting (box, circle, NN)
- Remapping (NN, inverse distance weighted)
- Cross sections (Constant latitudes and longitudes)

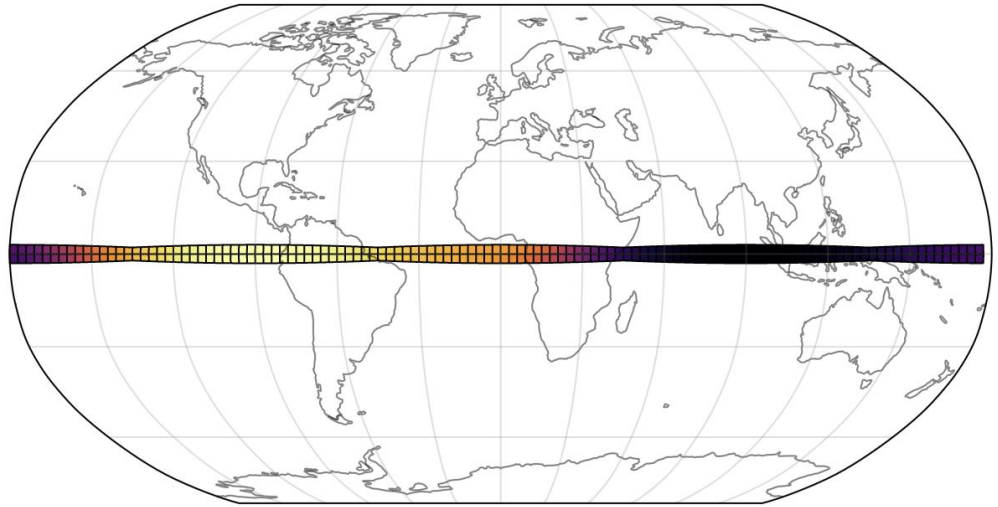
Computational operators

- Non-conservative zonal means
- Global weighted averaging
- Integration
- Gradients
- Difference

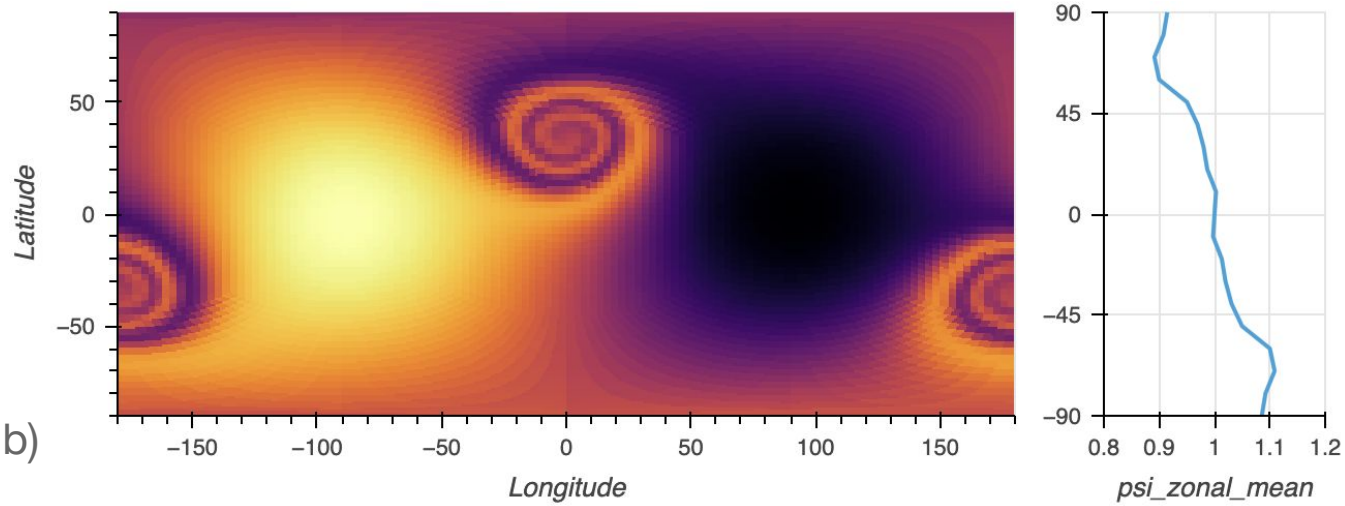
Plotting

- Several functions for plotting mesh and data
- Interactive plots with Holoviz (Bokeh and Matplotlib)
- Outputs for native Matplotlib plots

Cross-sections

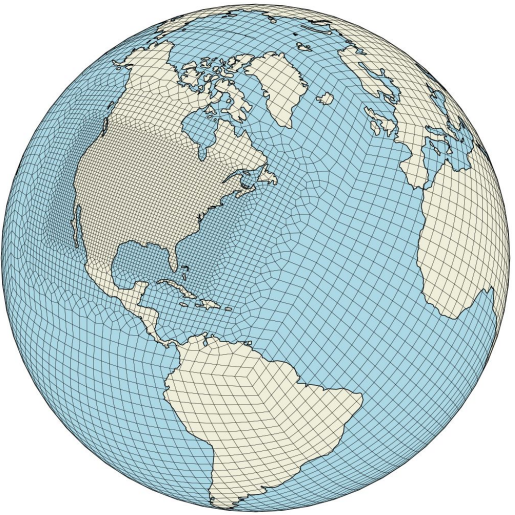


Non-conservative zonal means

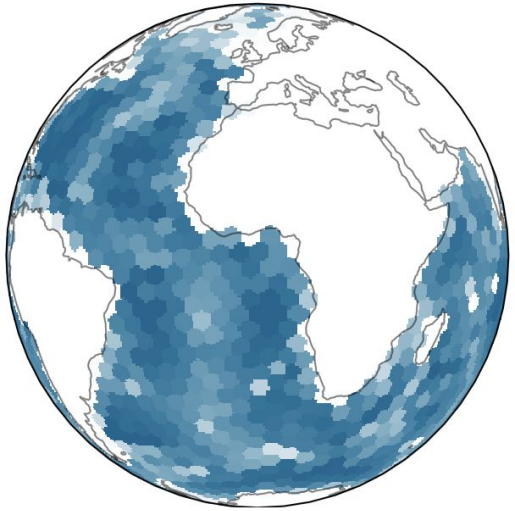


Current status - Visualization

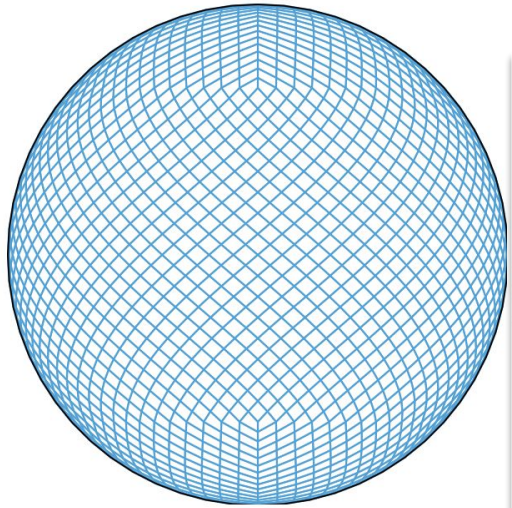
CAM-SE

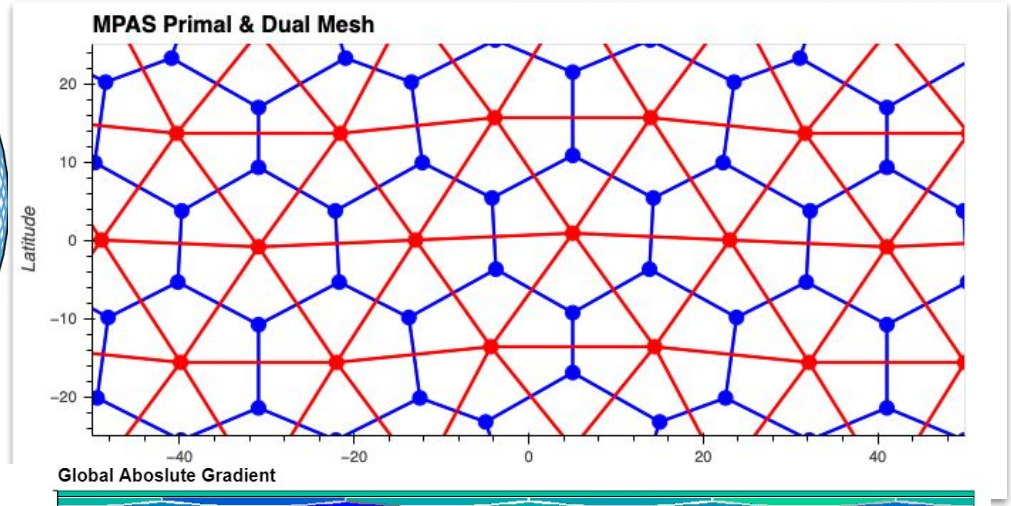


MPAS

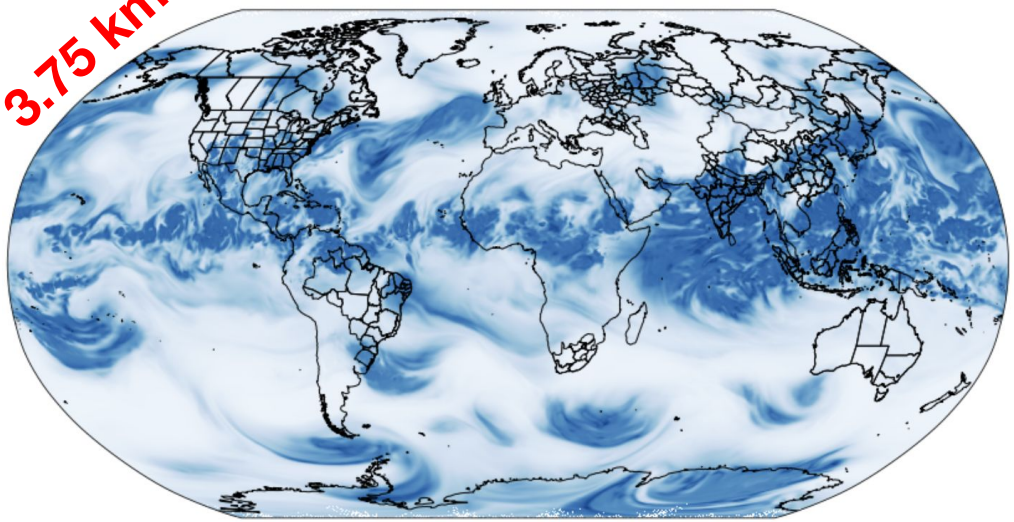


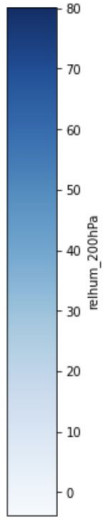
HEALPix

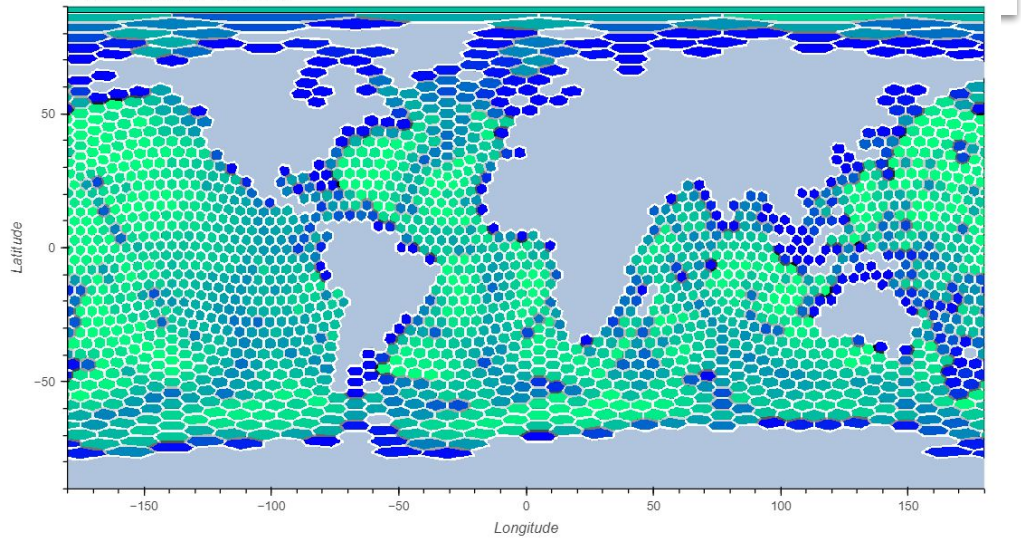




3.75 km

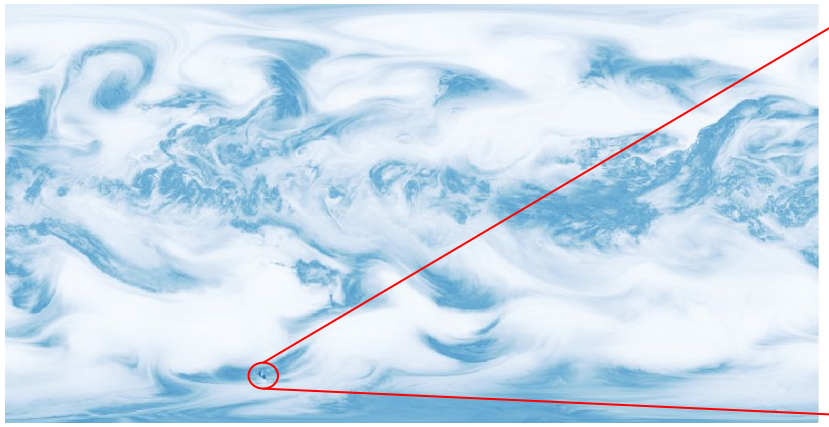




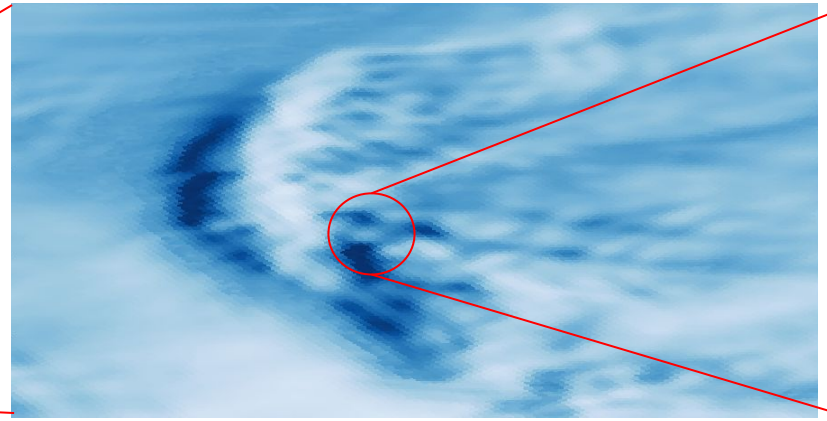


Current status - Visualization

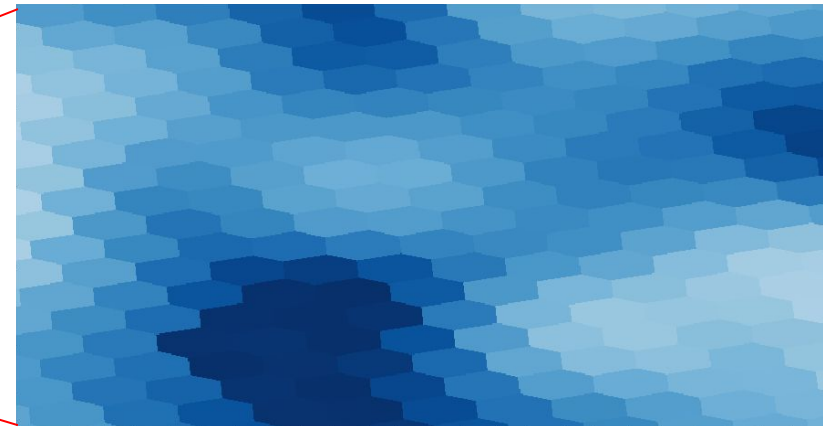
High-Resolution Example: 3.75km MPAS Grid



Global Polygon Raster
~42 million individual polygons



Zoomed Raster
Begin to observe the underlying grid topology (hexagons)



Further Zoomed Raster
After zooming far enough, the original hexagon grid becomes extremely clear

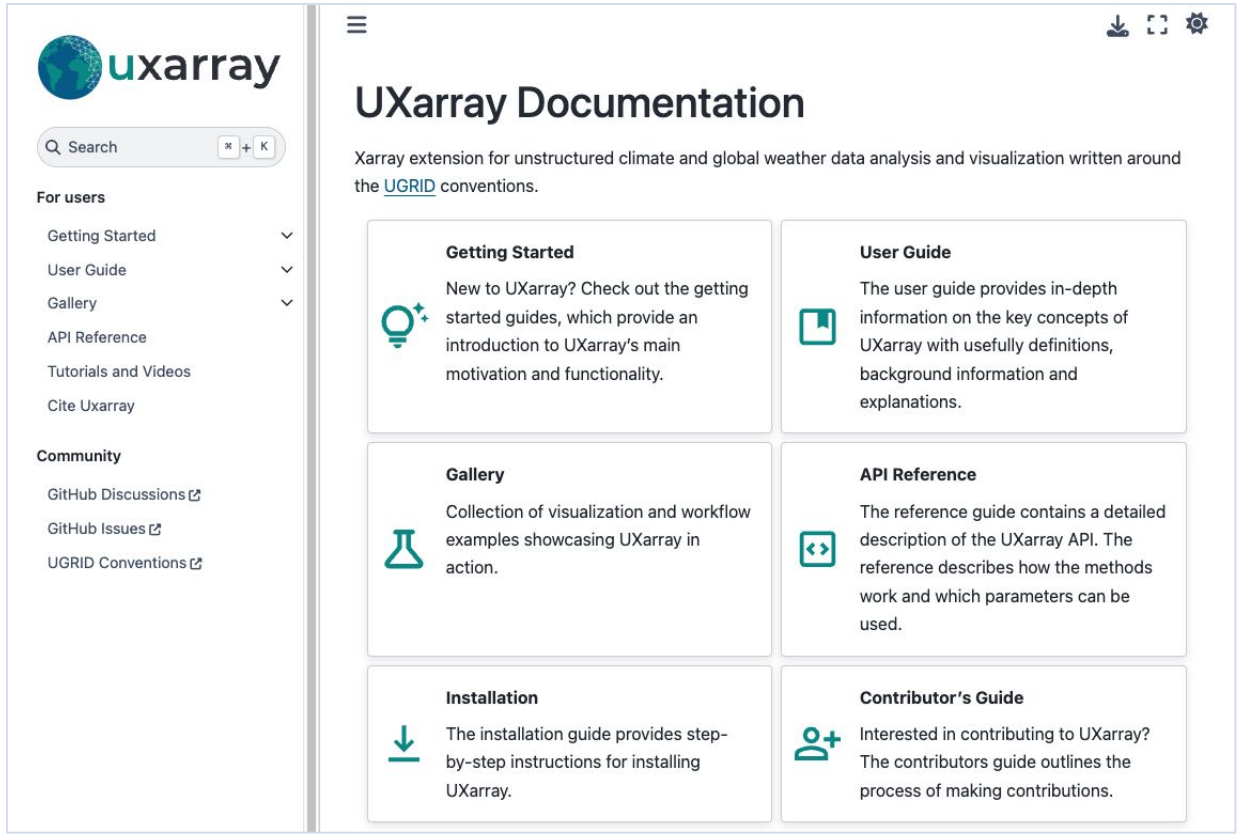
Current status - Continuous delivery

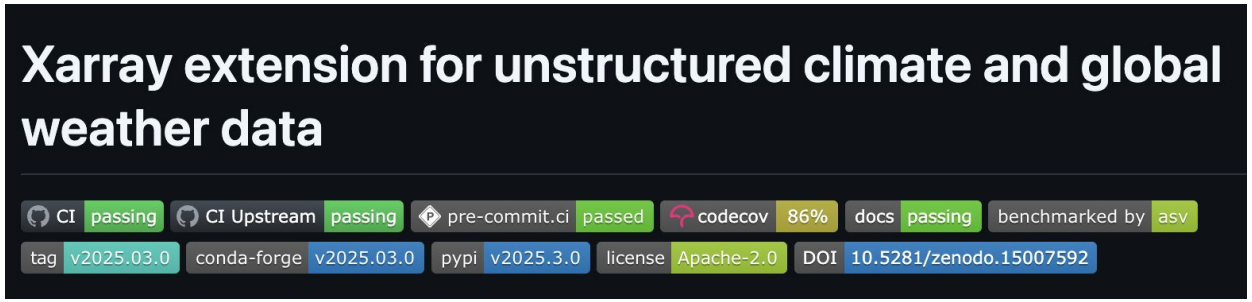
- Public facing website
  - https://uxarray.readthedocs.io/

- **Comprehensive user documentation**
  - Getting started
  - User guide
  - Contributor's guide
  - Gallery
  - API reference

- **Monthly releases on conda and PyPi**

- Detailed examples (Jupyter Notebooks)





Future work

Coming soon!

- Derivatives (advanced capabilities)
- Conservative zonal means
- Bilinear remapping

Longer term

- NSF CSSI proposal under review “Collaborative Research: Frameworks: UXarray - A sustainable tool to meet next generation earth system models’ analysis needs”
  - Appeal to other science domains (across the NSF GEO Directorate)
  - Become the go-to tool for working with unstructured grid earth sciences data
- New computational operators (div, curl, etc.)
- Plotting (Contours, vectors, etc.)
- More regridding as needed

uxarray

Real time roadmap

Lessons learned (from our scientific community)

- Git, GitHub and Python package management are barriers to many in the scientific community
- Jupyter Notebooks and JupyterHub are awesome!
- Dask is a technology for the 1% (at most)
- Socializing design decisions and feature requests can be super helpful
- Building a contributor community is really, really hard!

WCRP
Digital Earth
Lighthouse Activity

The World Climate Research Programme Global km-scale Hackathon

When: 12–17 May 2025

Where: 11 regional nodes worldwide

Our partners:

WCRP World Climate Research Programme
ESMO Earth System Modelling and Observations
next GEMS
WARM WORLD
EERIE European Eddy-Rich ESMs
Destination Earth

For more information

Visit our website

Summary

UXarray is an **extensible**, **scalable**, open source Python package for analysis and visualization of unstructured grid data from ESS models.

Builds on top of a software ecosystem widely used by the ESS community; i.e. extends the **Xarray** package, supports **Dask** parallelization, documents with **Jupyter**, etc.

Supports a growing number of file formats & model outputs

Community driven and developed, with an increasing amount of global community contributions

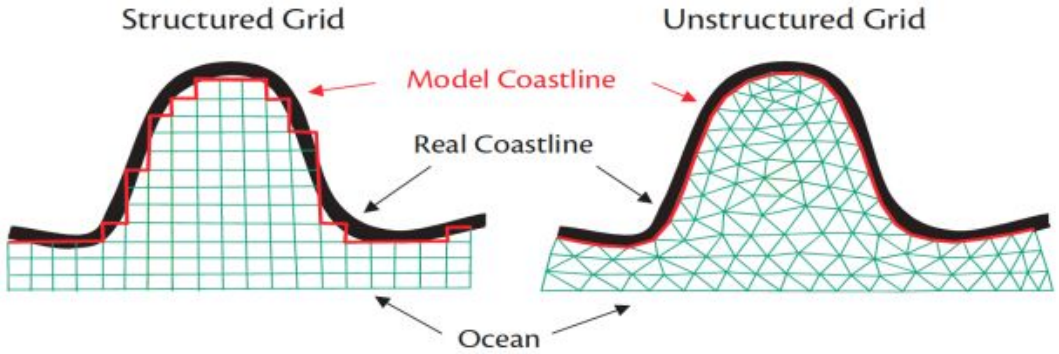
uxarray

https://uxarray.readthedocs.io

BACKUP

Unstructured Grid Geometry

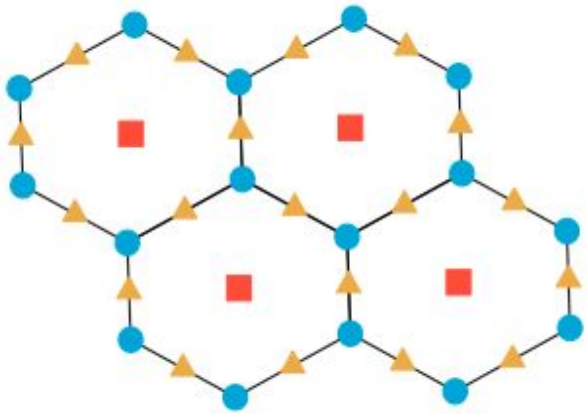
Adaptibility to complex geometries

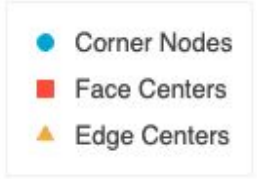


Geometric adaptability (example from FVCOM)

Local refinement

Variable res. MPAS Voronoi grid





Explicitly defined **connectivity** information for each geometric element; e.g.:

Connectivity	Dimension	Description
Face-Node	#Faces x #MaxPolygonNodes	Nodes' indices making up each face