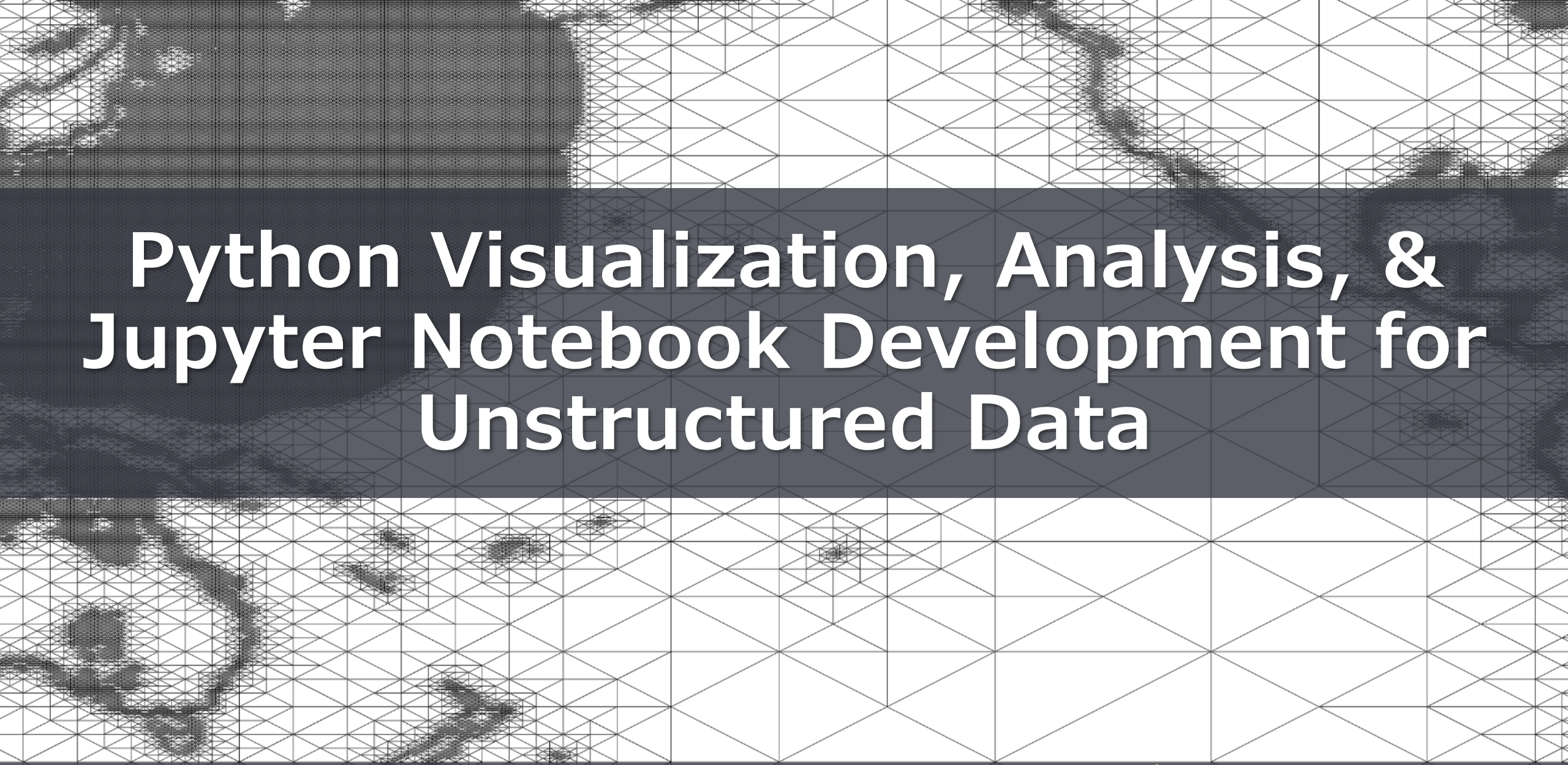


Python Visualization, Analysis, & Jupyter Notebook Development for Unstructured Data

NCAR | NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

Philip Chmielowiec
University of Illinois, SIParCS

July 27, 2022

NCAR | GeoCAT

PROJECT raijin

SIParCS Summer Internships in Parallel Computational Science BOULDER · CO

Overview

GeoCAT

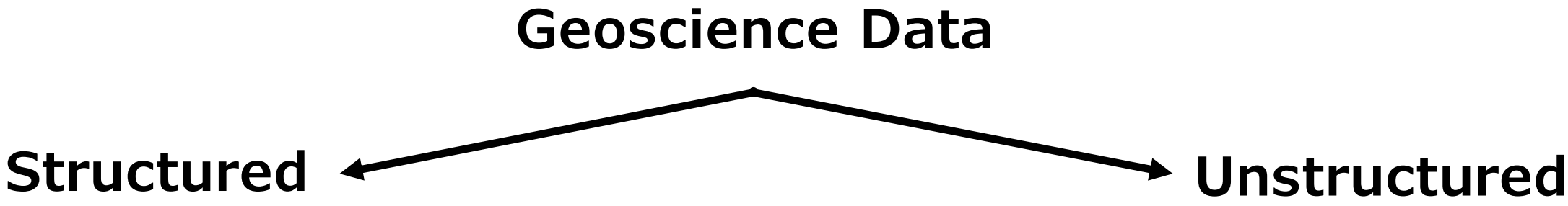
- **Geo**science **C**ommunity **A**nalysis **T**oolkit
- Data Analysis and Visualization Tools
  - GeoCAT-viz
  - GeoCAT-comp
  - GeoCAT-datafiles
  - GeoCAT-f2py
  - GeoCAT-examples

Project Raijin

- NSF EarthCube funded effort
- Enhance open-source development for unstructured data
  - Analysis
  - Visualization

Background

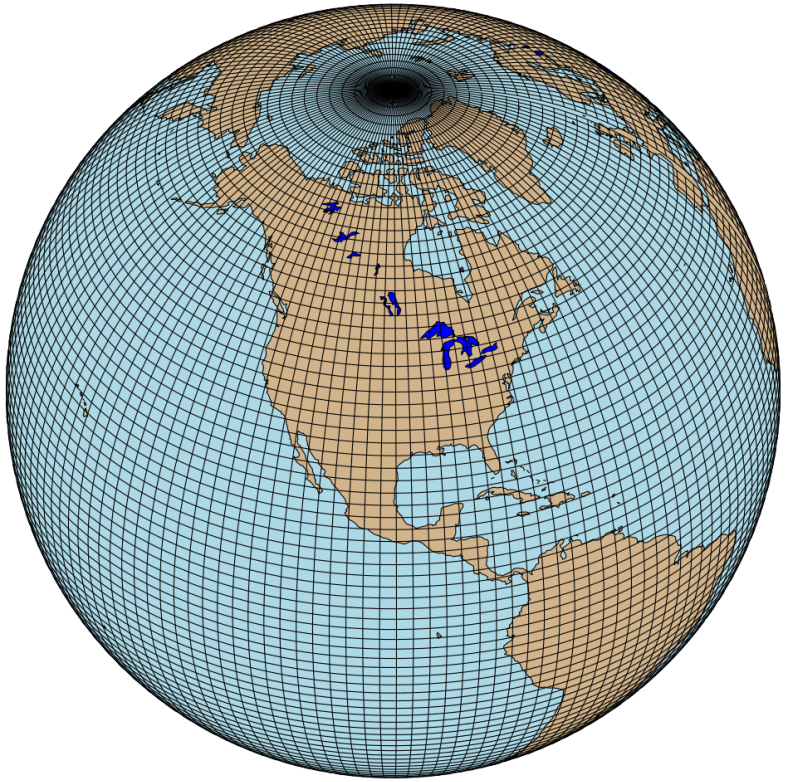
Grids



Grids

Geoscience Data

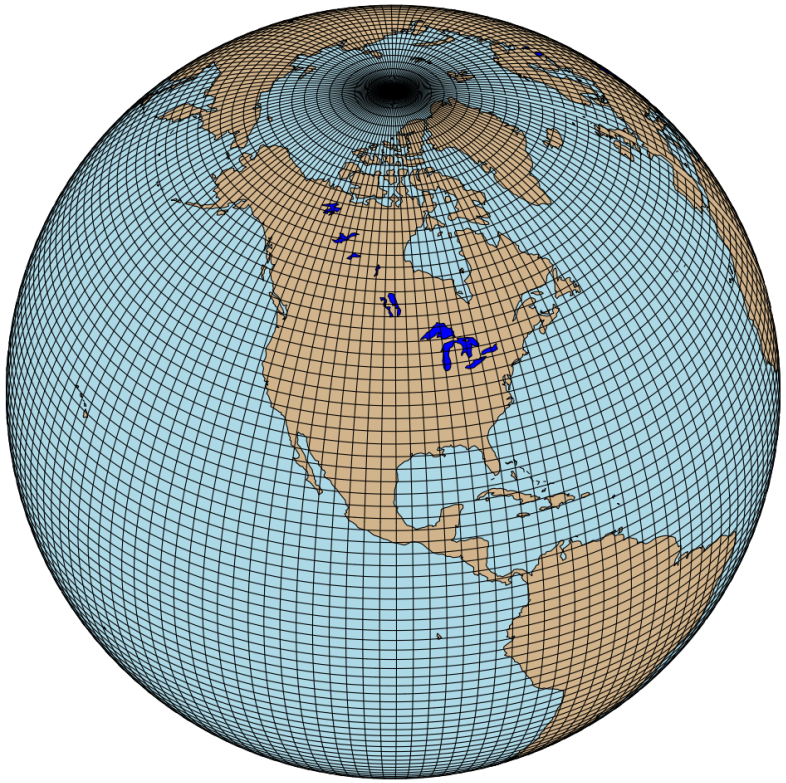
Structured ← → **Unstructured**



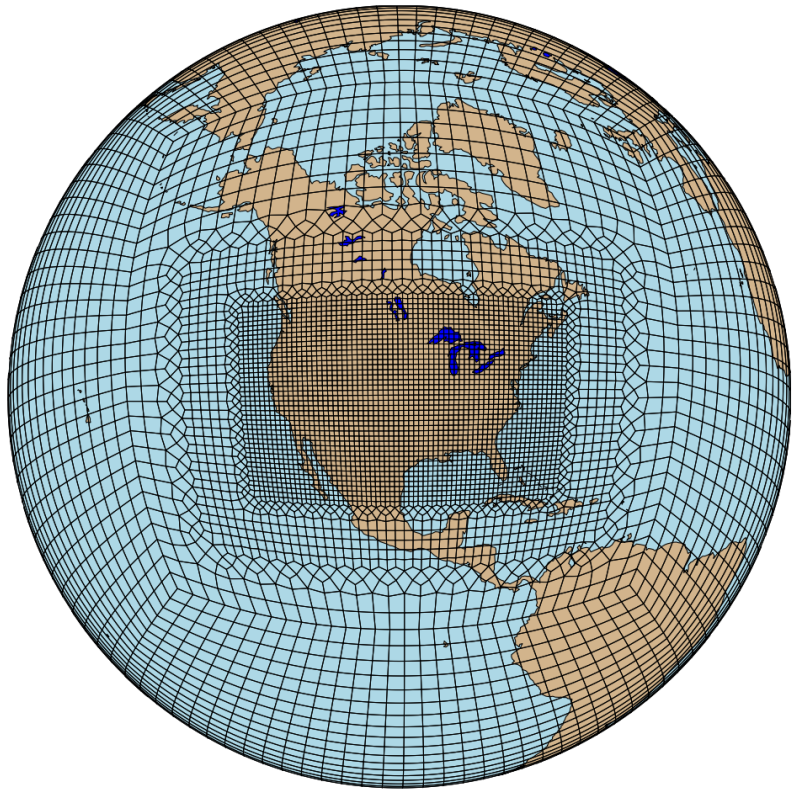
Grids

Geoscience Data

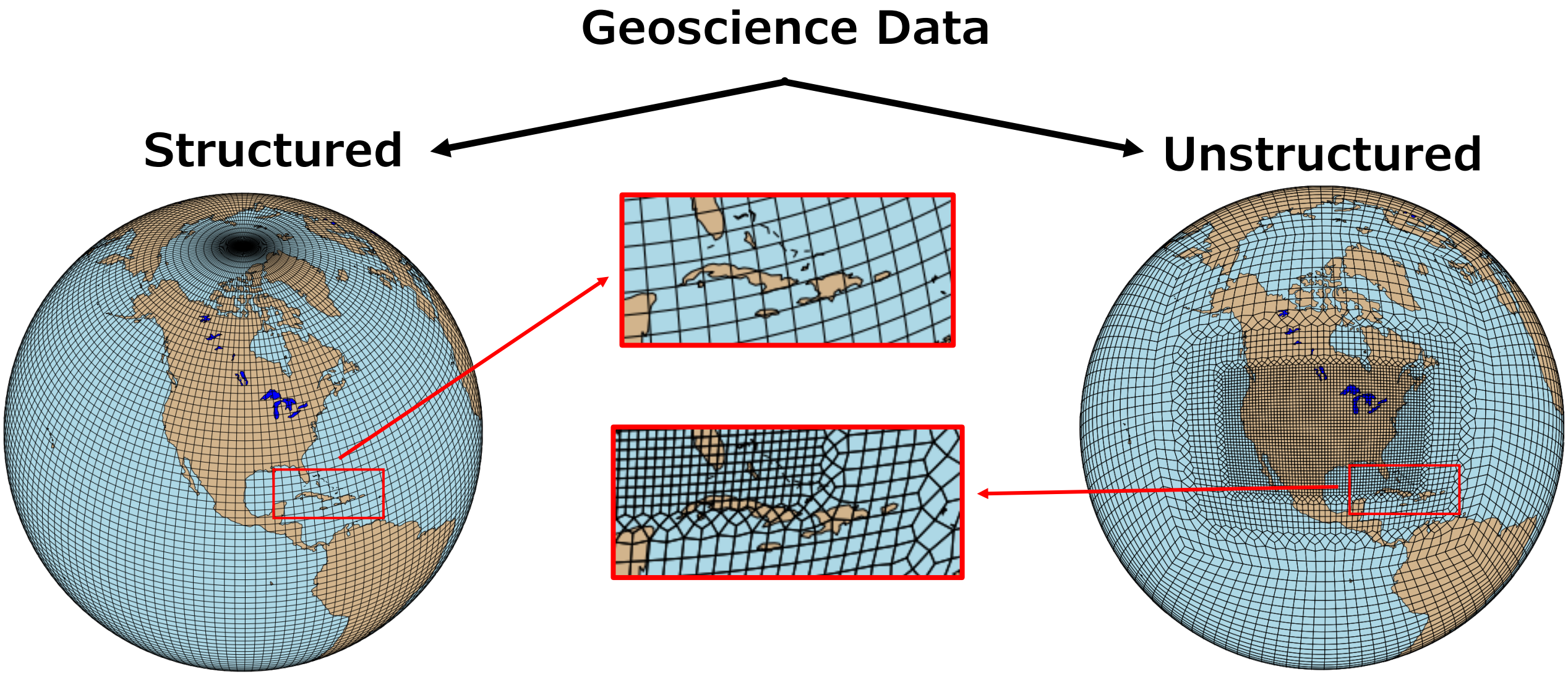
Structured



Unstructured



Grids



Coordinates & Data

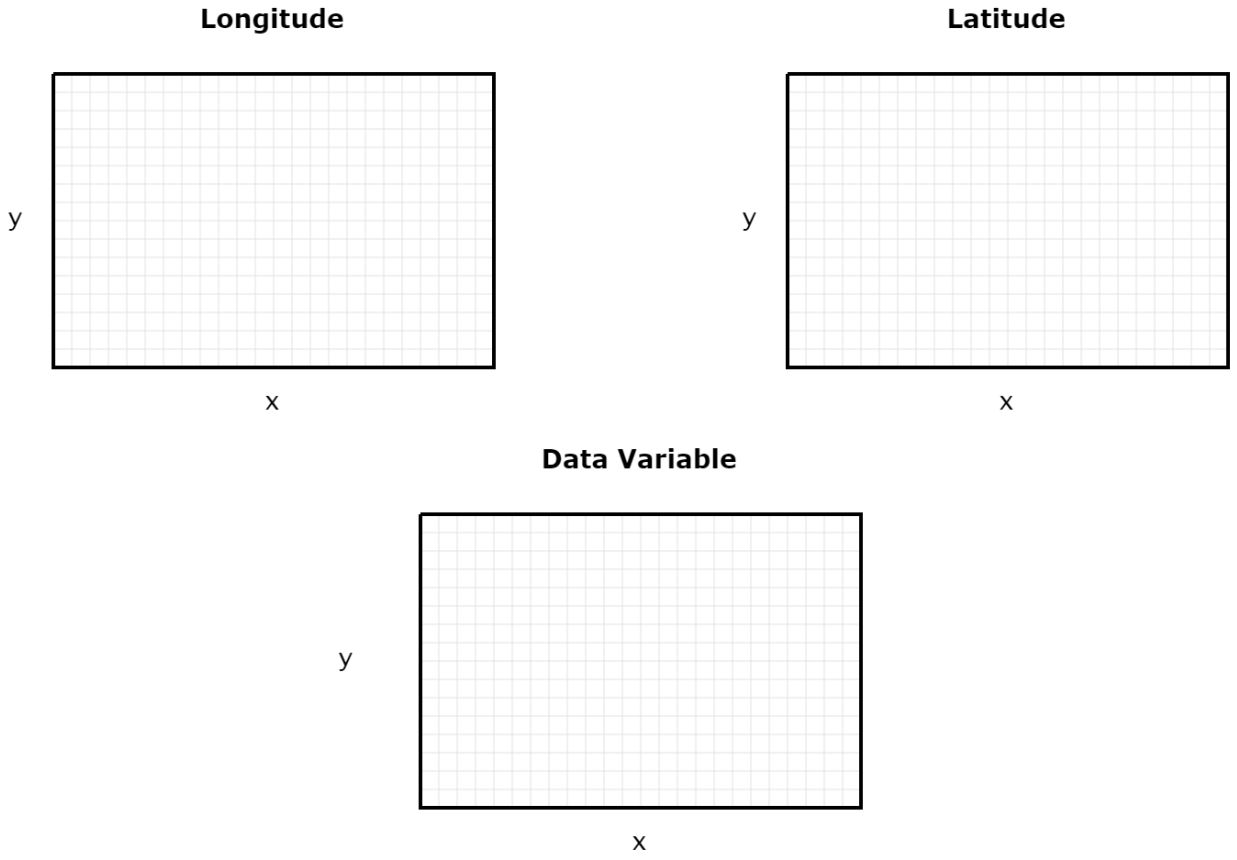
Structured

Unstructured

Coordinates & Data

Structured

$[n_{lat} \times n_{lon}]$ Grid
$[lat^{o} \times lon^{o}]$ Resolution

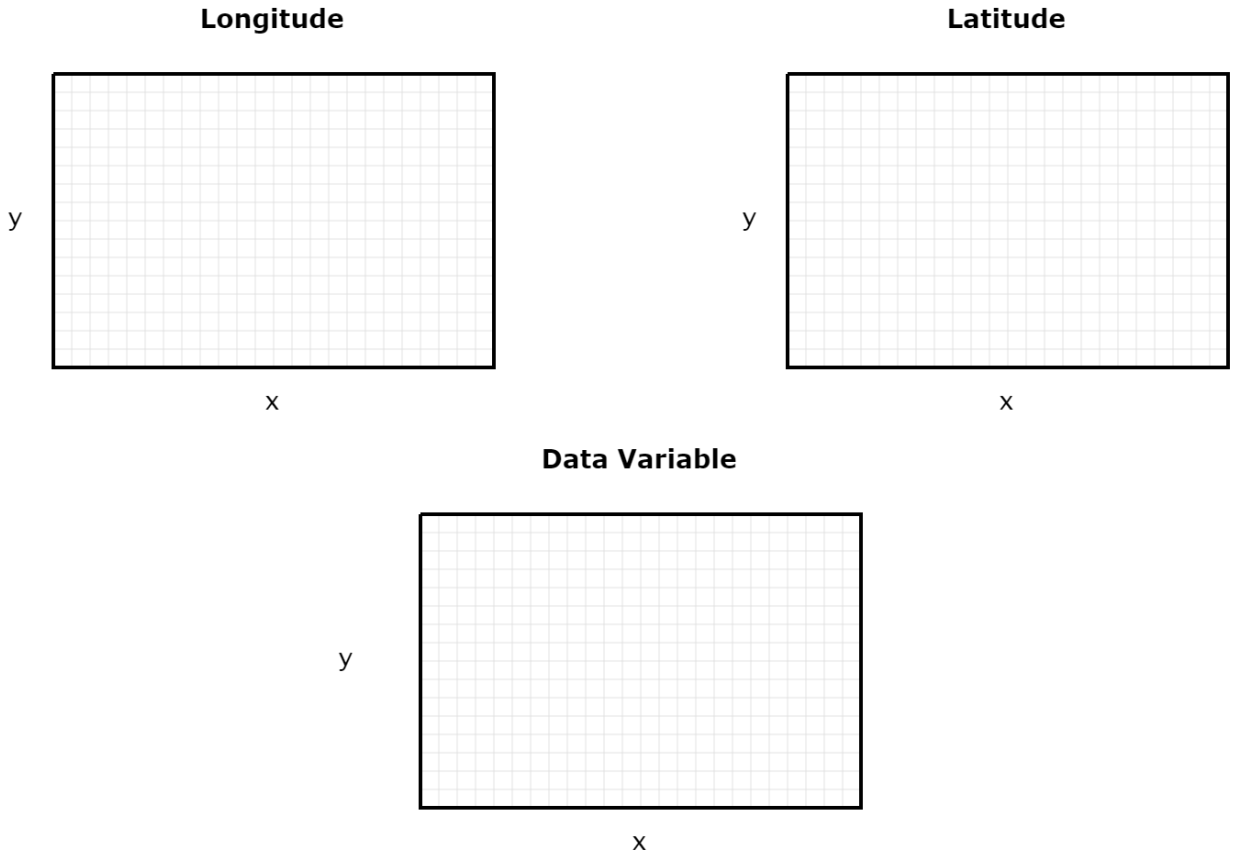


Unstructured

Coordinates & Data

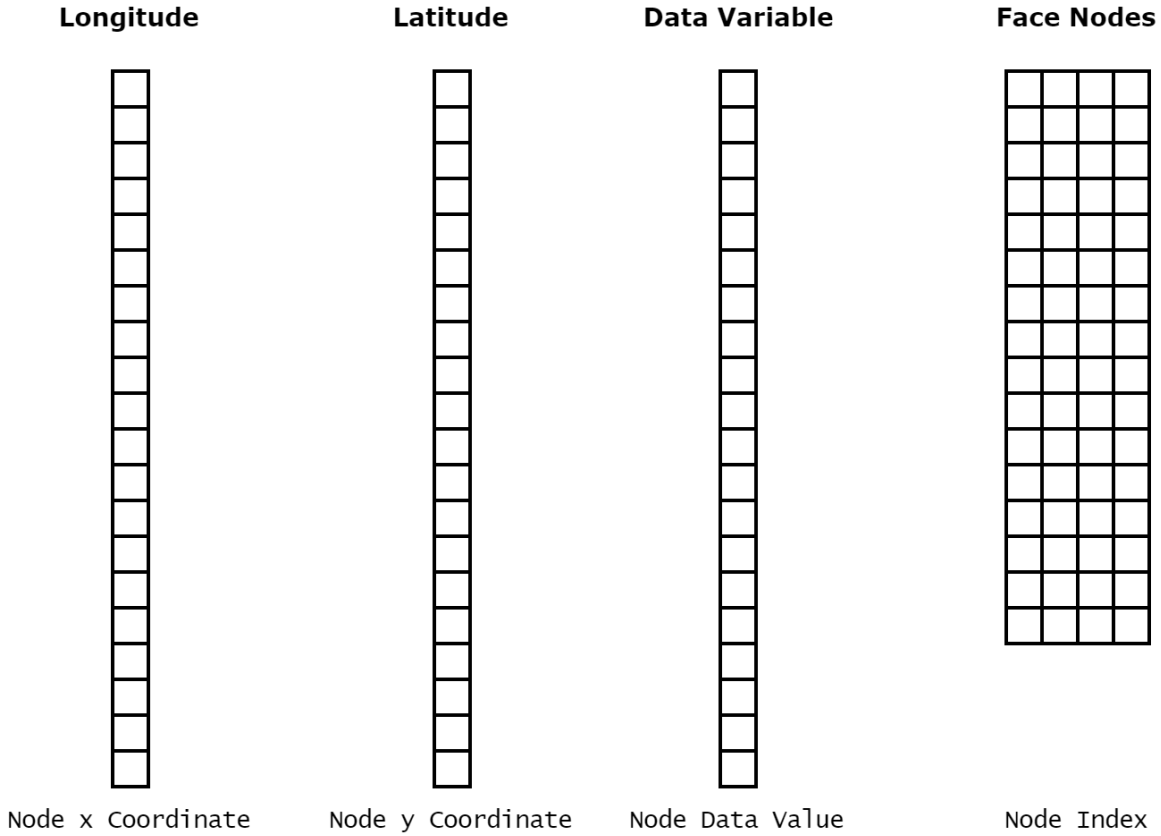
Structured

$[n_{lat} \times n_{lon}]$ Grid
$[lat^{o} \times lon^{o}]$ Resolution

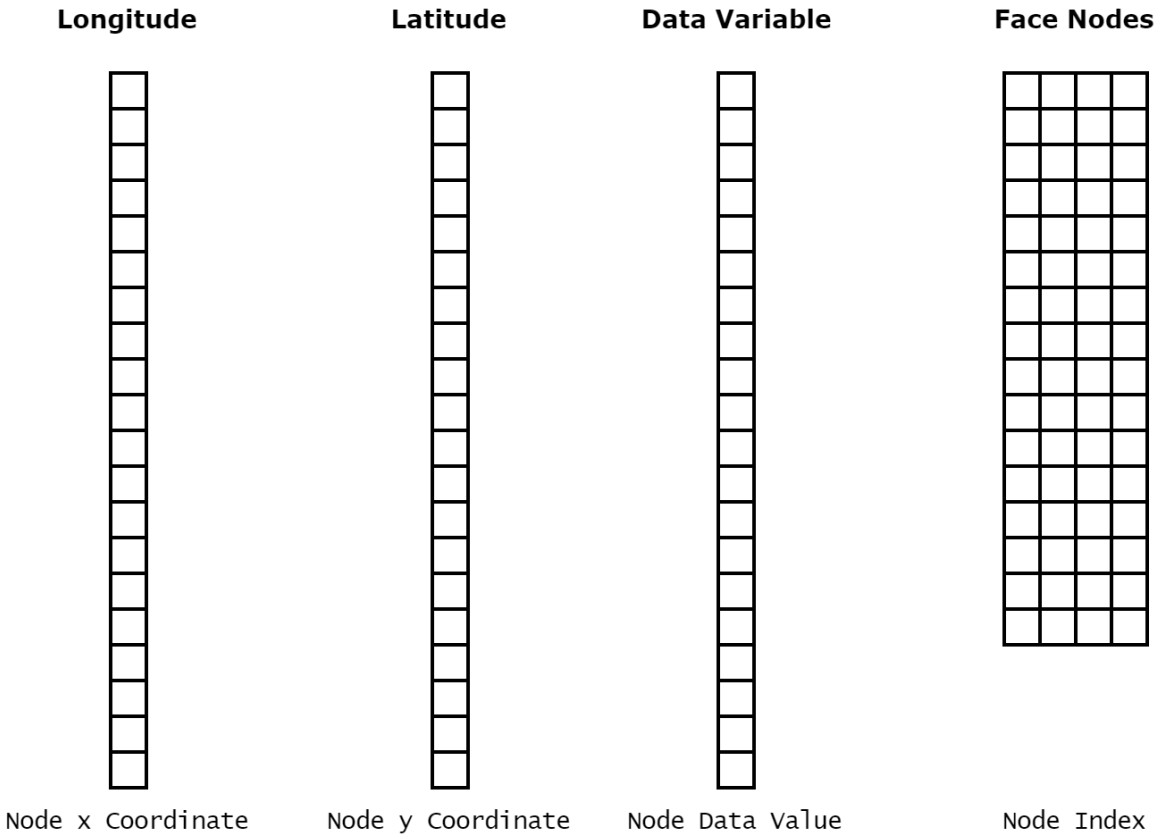


Unstructured

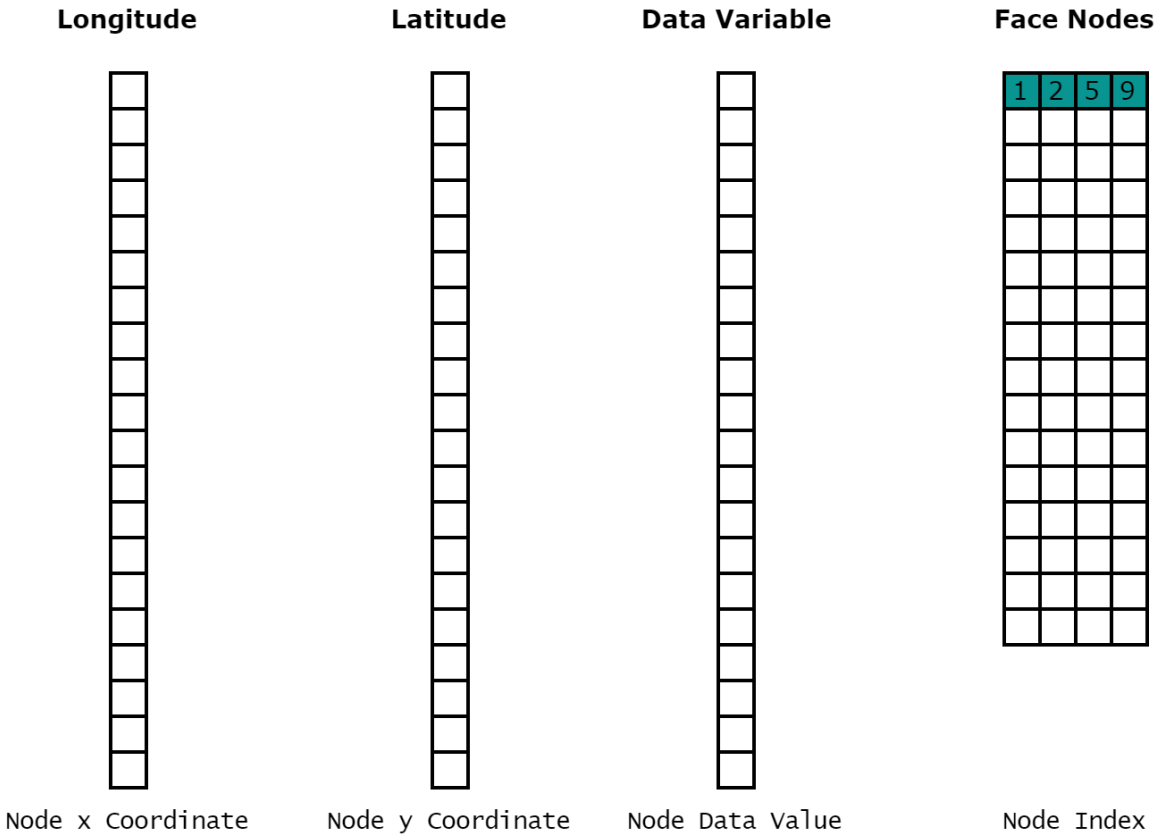
$[n_{face} \times n_{node}]$ Mesh
Variable Resolution



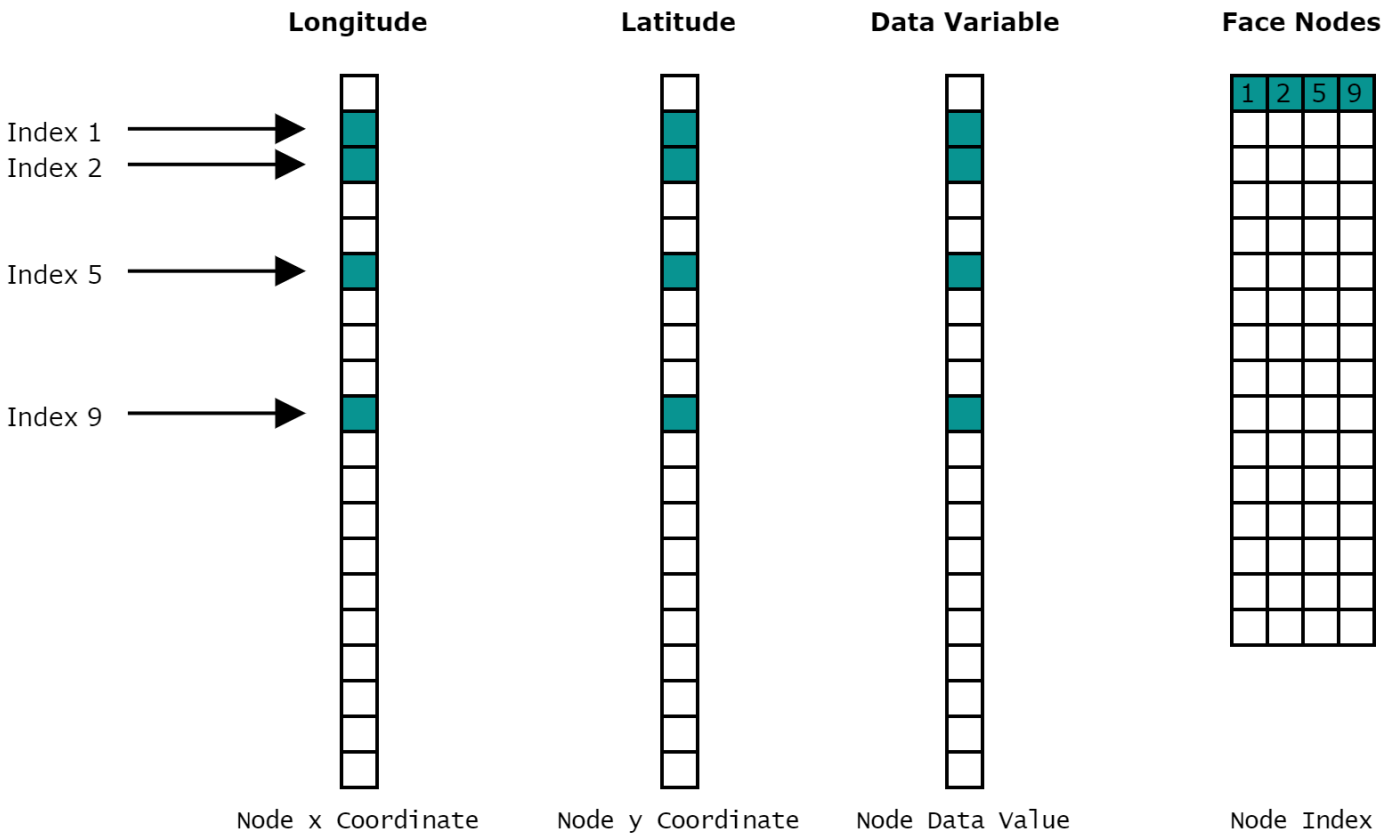
Face Construction



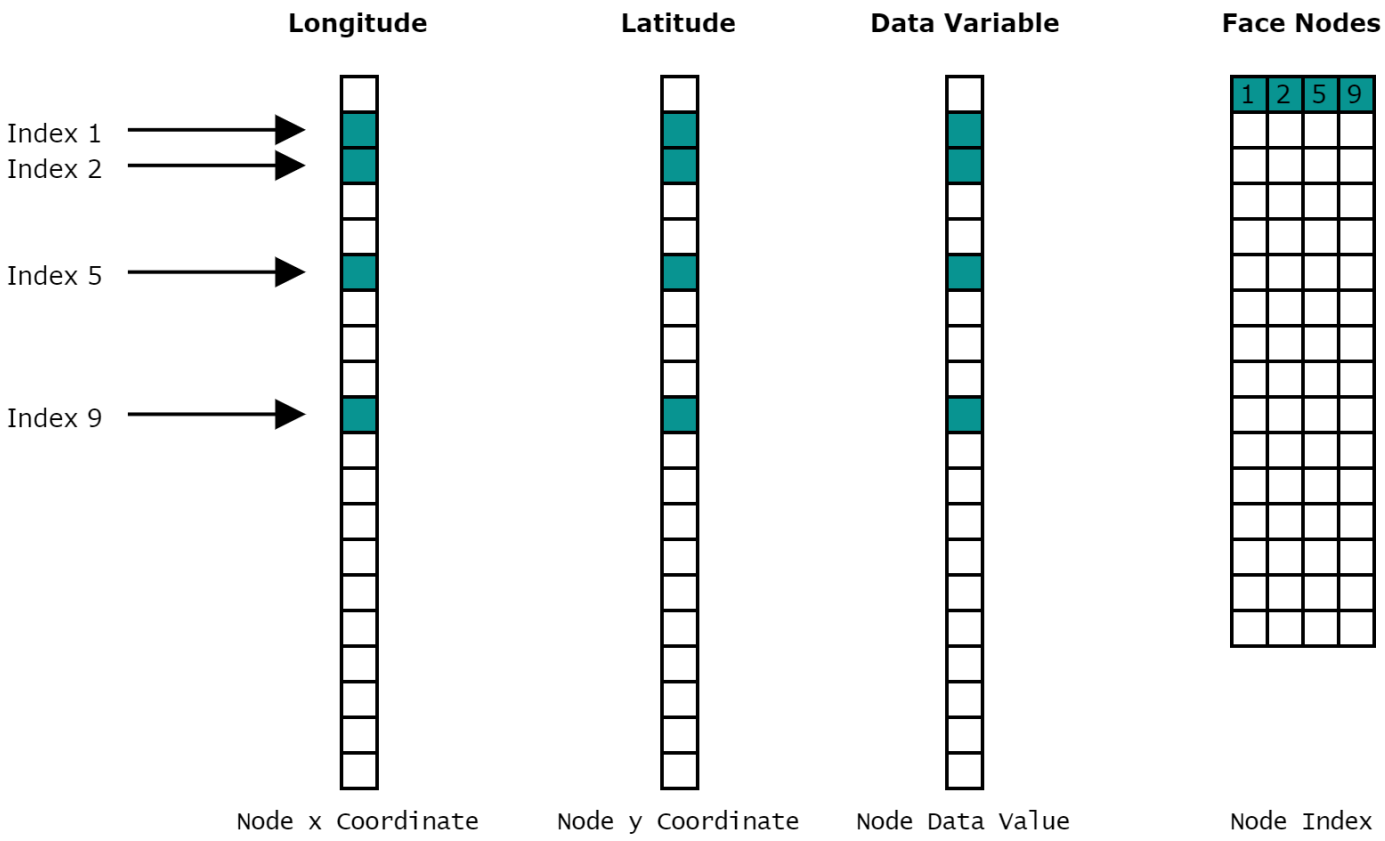
Face Construction



Face Construction



Face Construction



Face Nodes: [1, 2, 5, 9]

Longitude: $[x_1, x_2, x_5, x_9]$

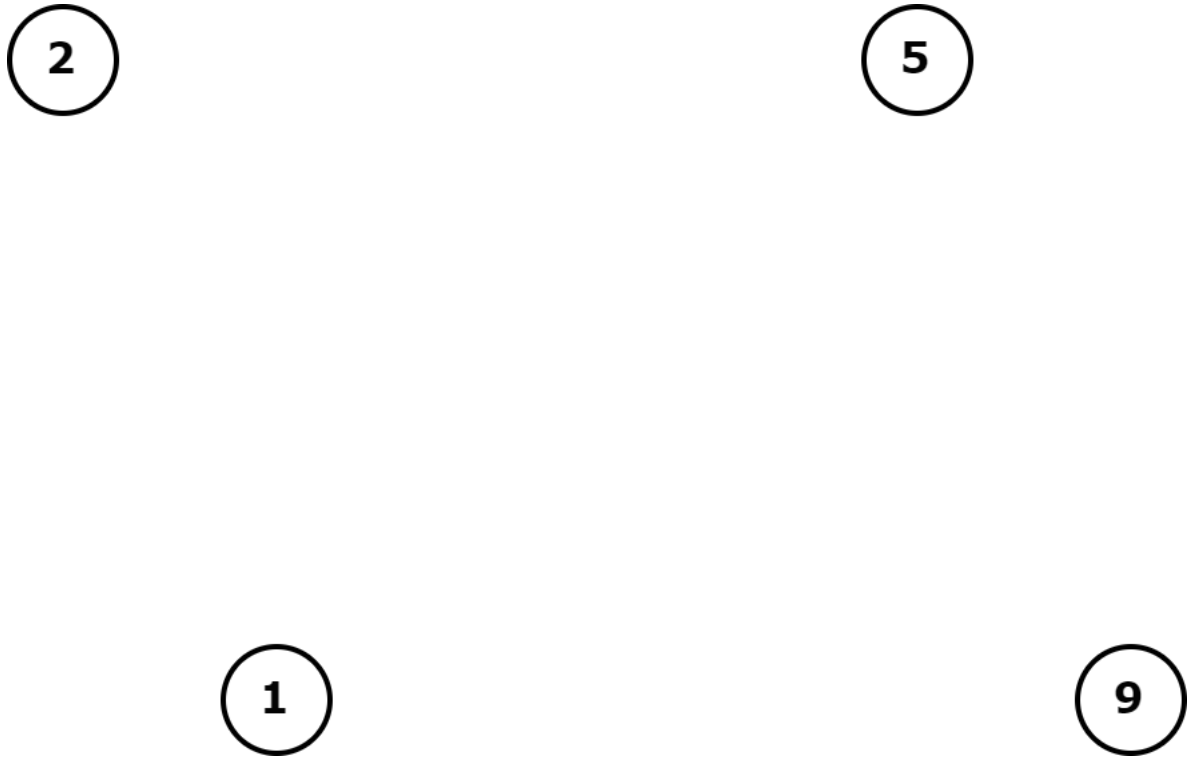
Latitude: $[y_1, y_2, y_5, y_9]$

Data: $[d_1, d_2, d_5, d_9]$

Face Construction

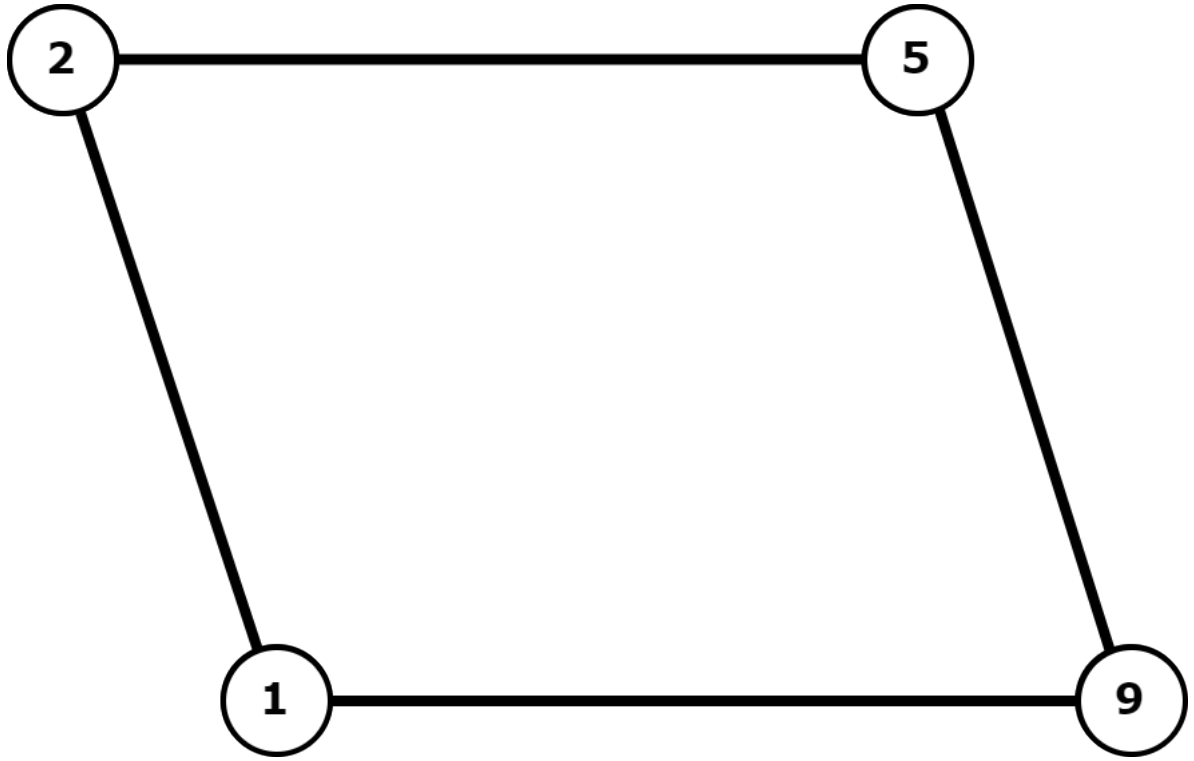
Face Nodes:	[1, 2, 5, 9]
Longitude:	[x_1, x_2, x_5, x_9]
Latitude:	[y_1, y_2, y_5, y_9]
Data:	[d_1, d_2, d_5, d_9]

Face Construction



Face Nodes:	[1, 2, 5, 9]
Longitude:	[x_1, x_2, x_5, x_9]
Latitude:	[y_1, y_2, y_5, y_9]
Data:	[d_1, d_2, d_5, d_9]

Face Construction



Face Nodes: [1, 2, 5, 9]
Longitude: $[x_1, x_2, x_5, x_9]$
Latitude: $[y_1, y_2, y_5, y_9]$
Data: $[d_1, d_2, d_5, d_9]$

Mesh Construction

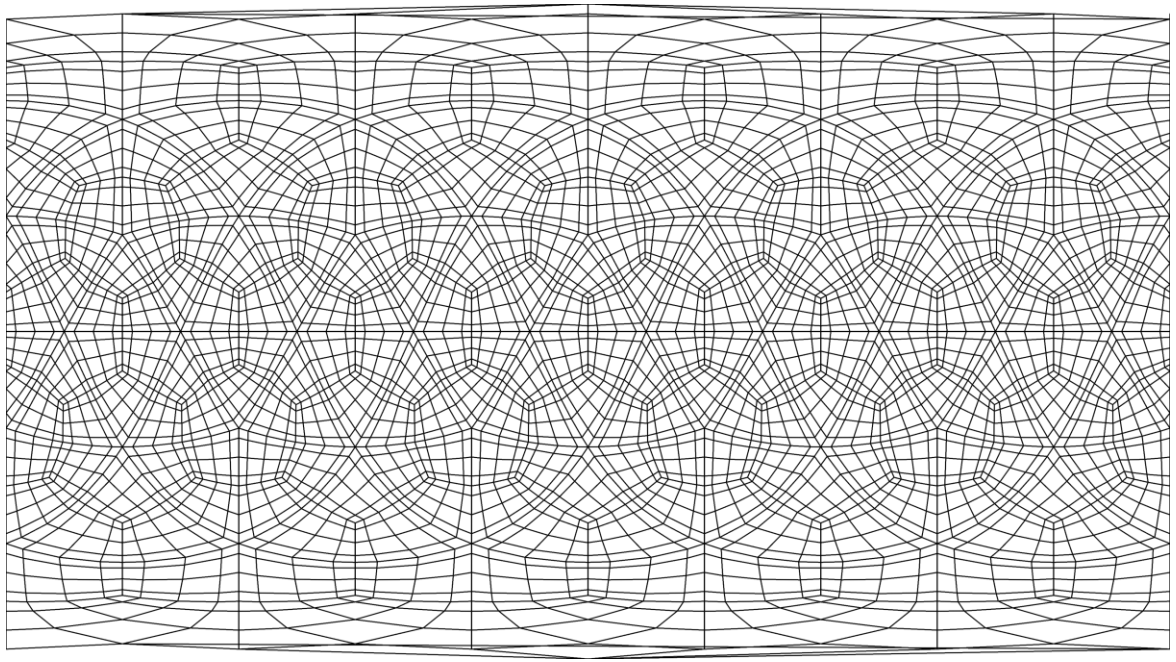
Direct Reconstruction

Delaunay Triangulation

Mesh Construction

Direct Reconstruction

Given a set of unstructured points (x, y) and their connectivity information (face nodes), reconstructs each face to create a 2D mesh of geometries defined by the dataset

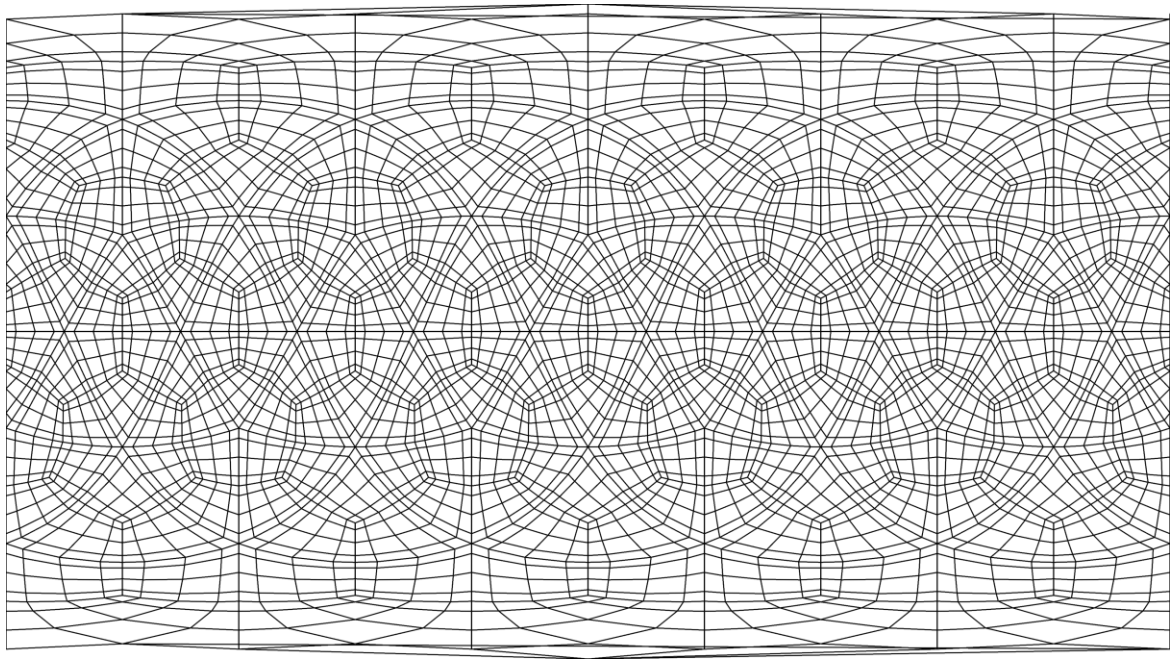


Delaunay Triangulation

Mesh Construction

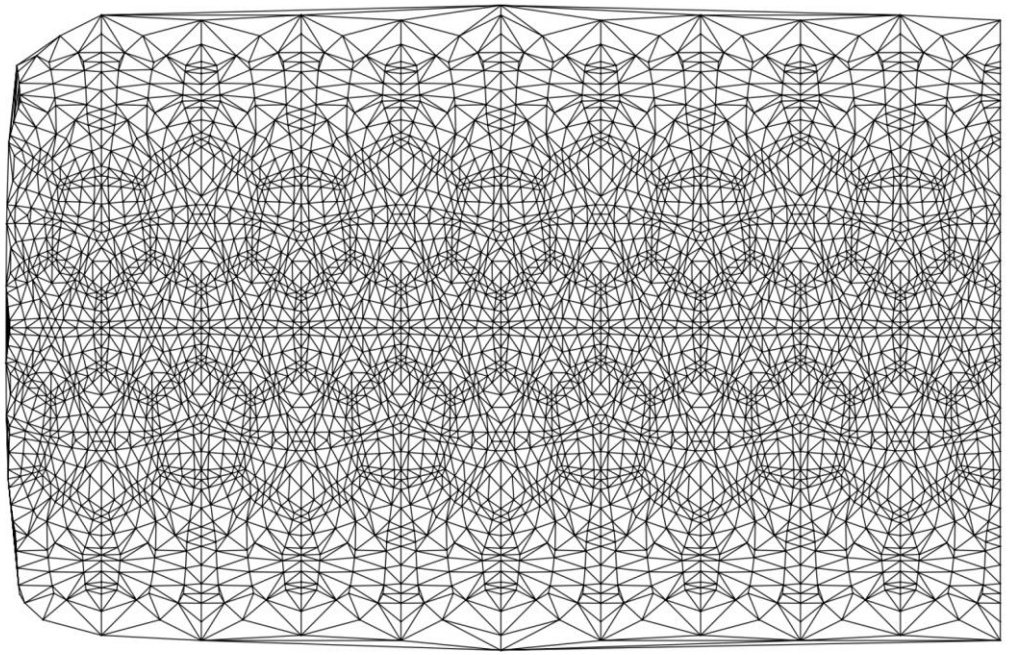
Direct Reconstruction

Given a set of unstructured points (x, y) and their connectivity information (face nodes), reconstructs each face to create a 2D mesh of geometries defined by the dataset



Delaunay Triangulation

Given a set of unstructured points (x, y), constructs a triangular mesh with the property that no vertex in the interior of the circumcircle of any triangle is in the triangulation



Project

Overview

GeoCAT-viz
GeoCAT-comp
GeoCAT-datafiles
GeoCAT-f2py
GeoCAT-examples

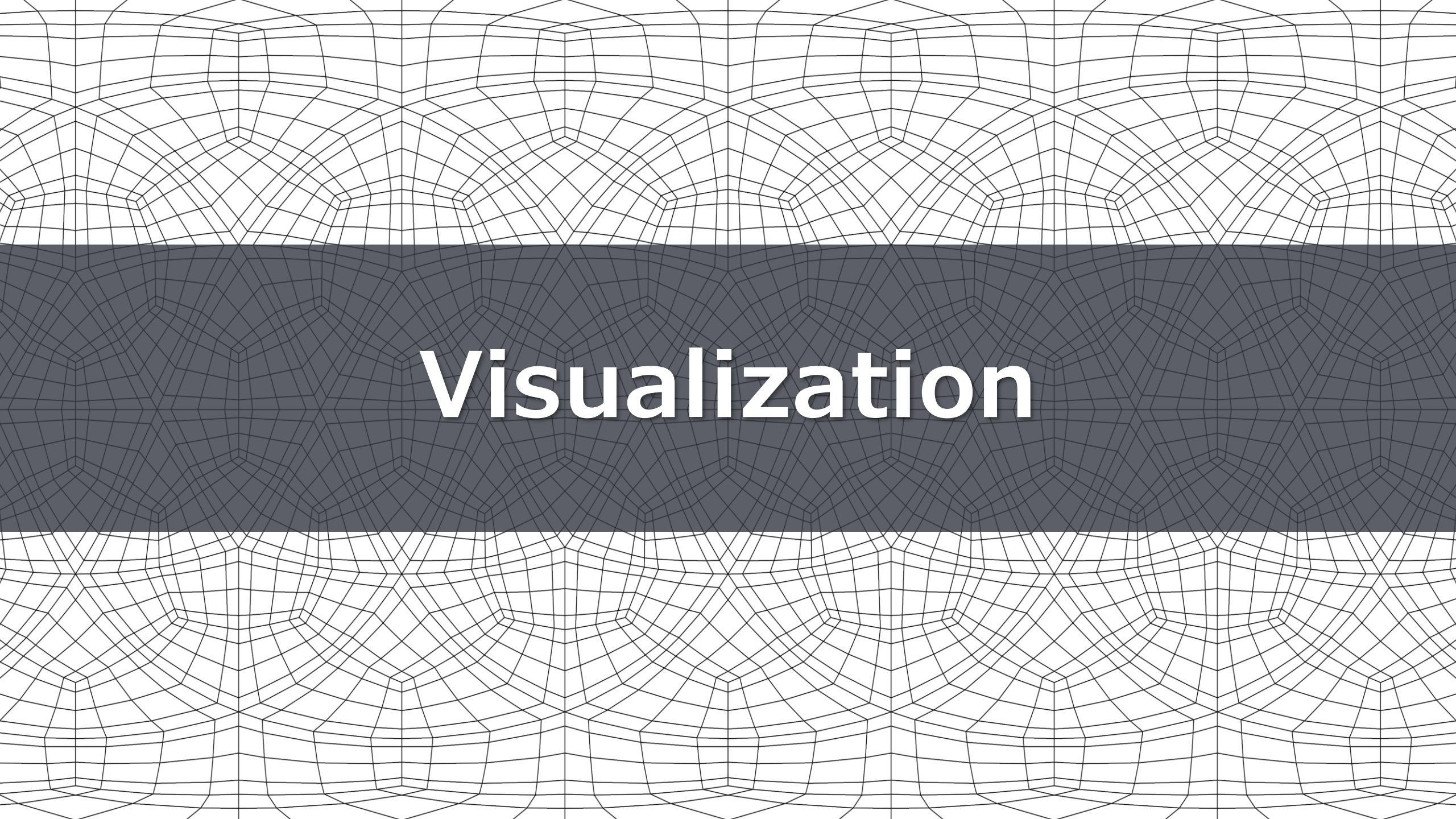
Analysis
Visualization

Overview

GeoCAT-viz
GeoCAT-comp
GeoCAT-datafiles
GeoCAT-f2py
1,2 GeoCAT-examples

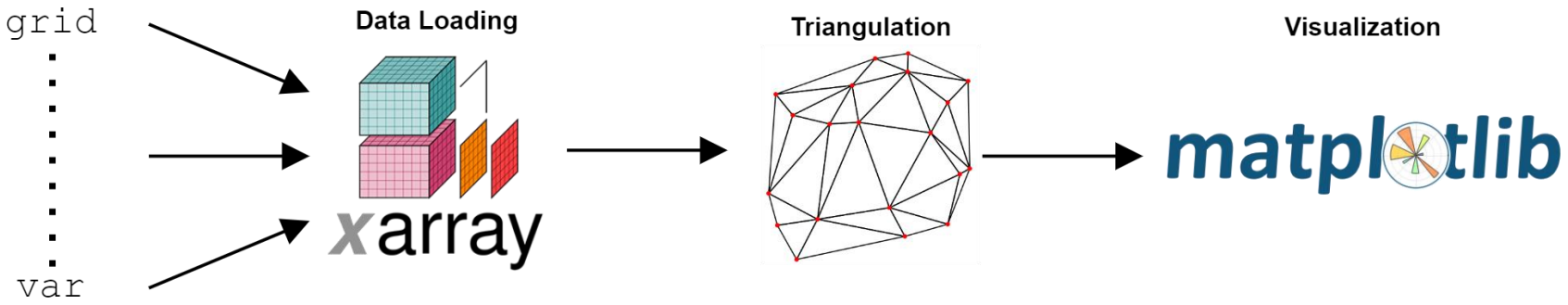
2,3 Analysis
1 Visualization

1. Research methods for visualizing unstructured data without resampling
2. Contribute to the Uxarray Python package
3. Develop usage examples for working with unstructured data

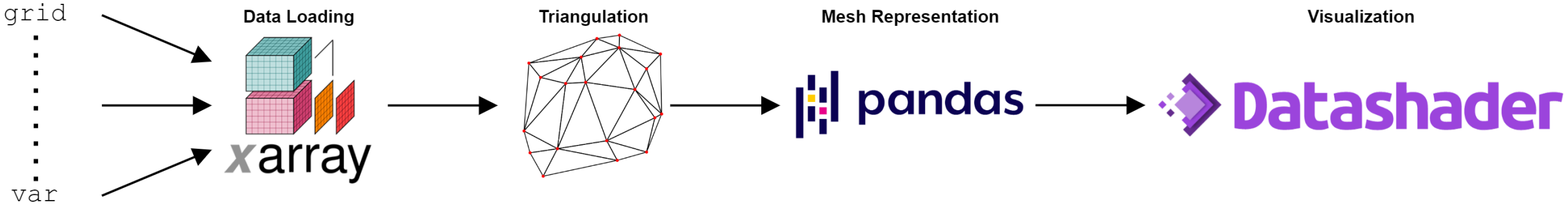

Visualization

Workflow

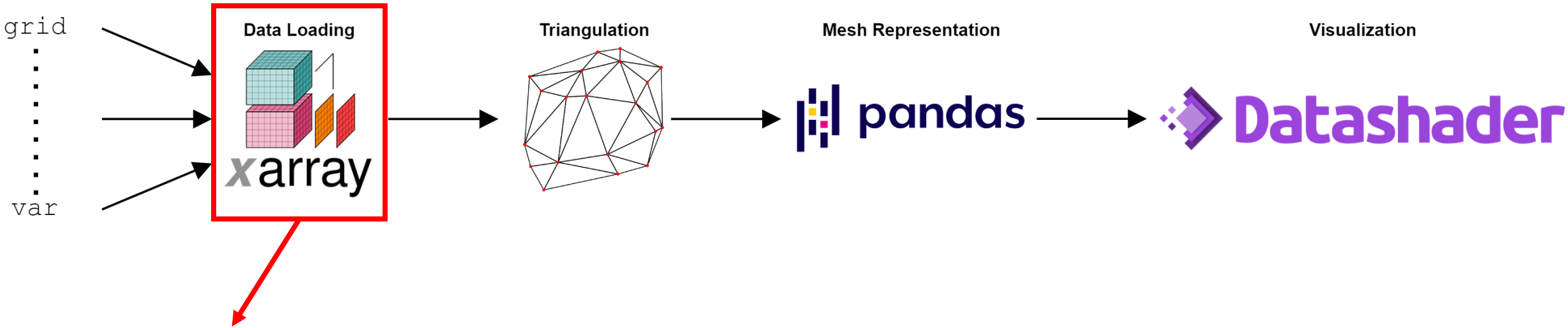
Small Datasets



Large Datasets

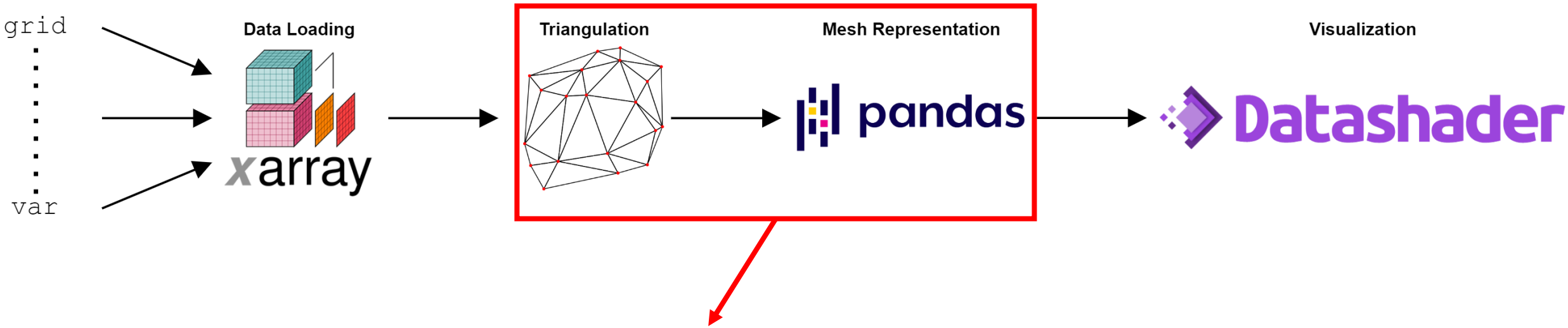


Issues



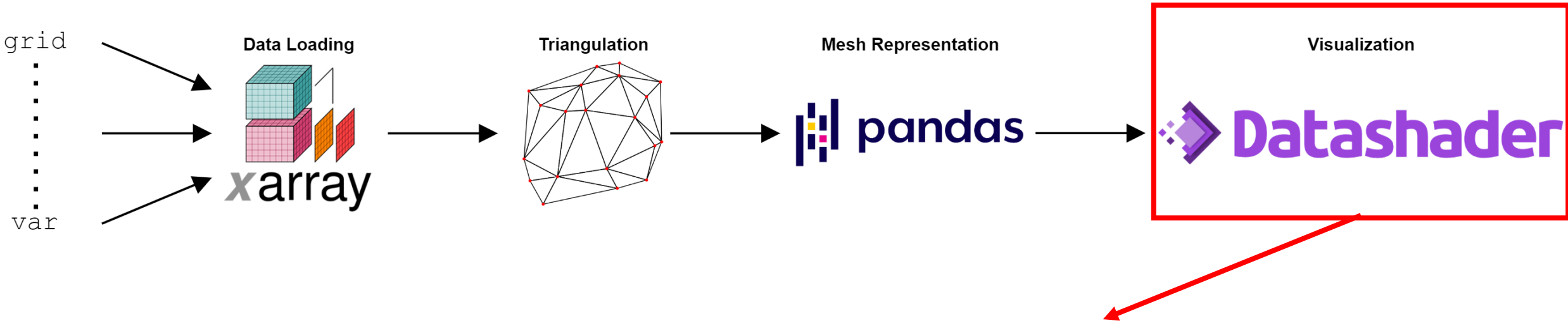
- Lacking support for handling unstructured coordinates
- No convenient way to load data and grid files together

Issues



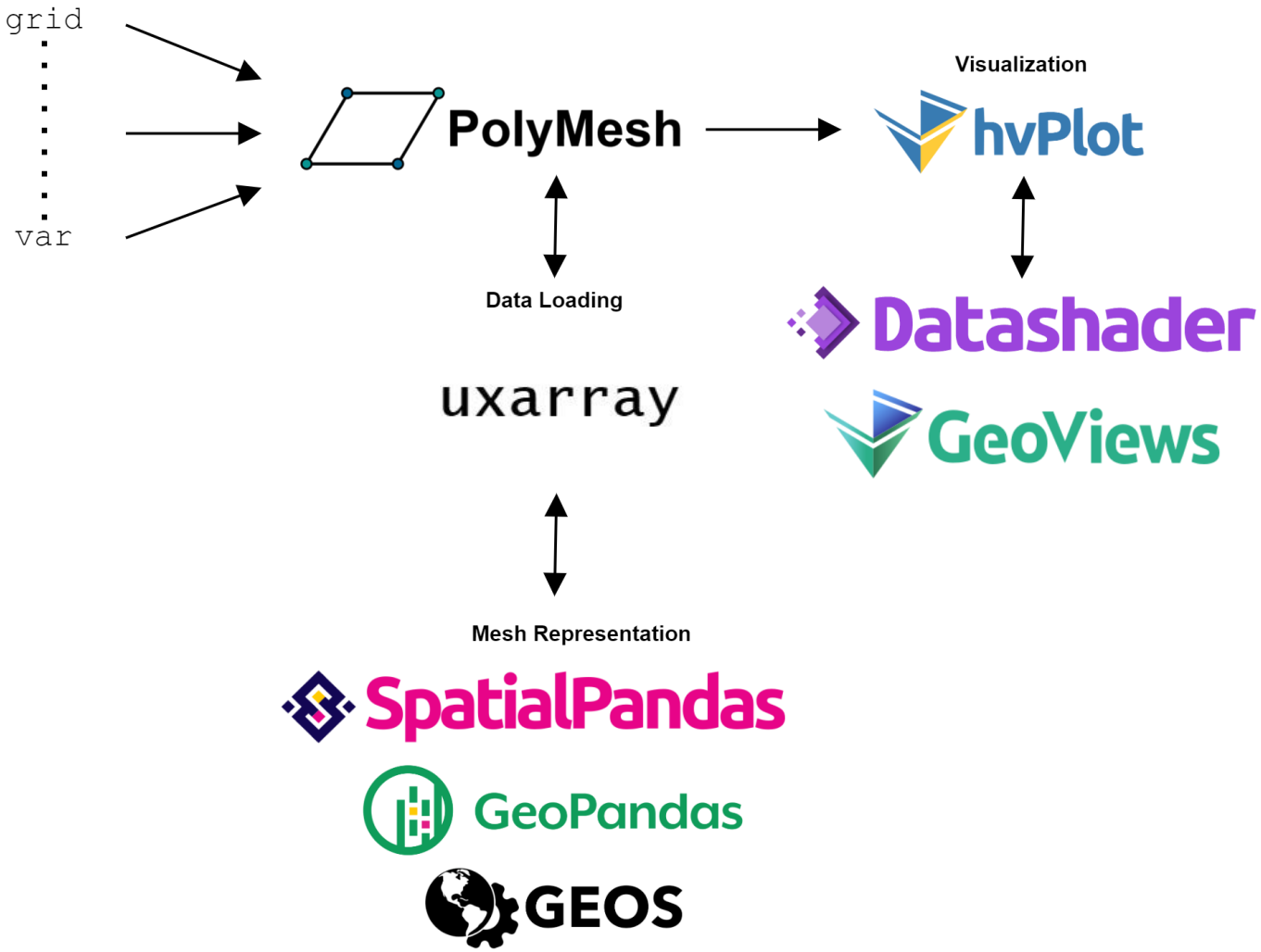
- Triangulation approximates our true unstructured mesh
- Computationally expensive on large datasets
- Connectivity information (face nodes) are ignored

Issues

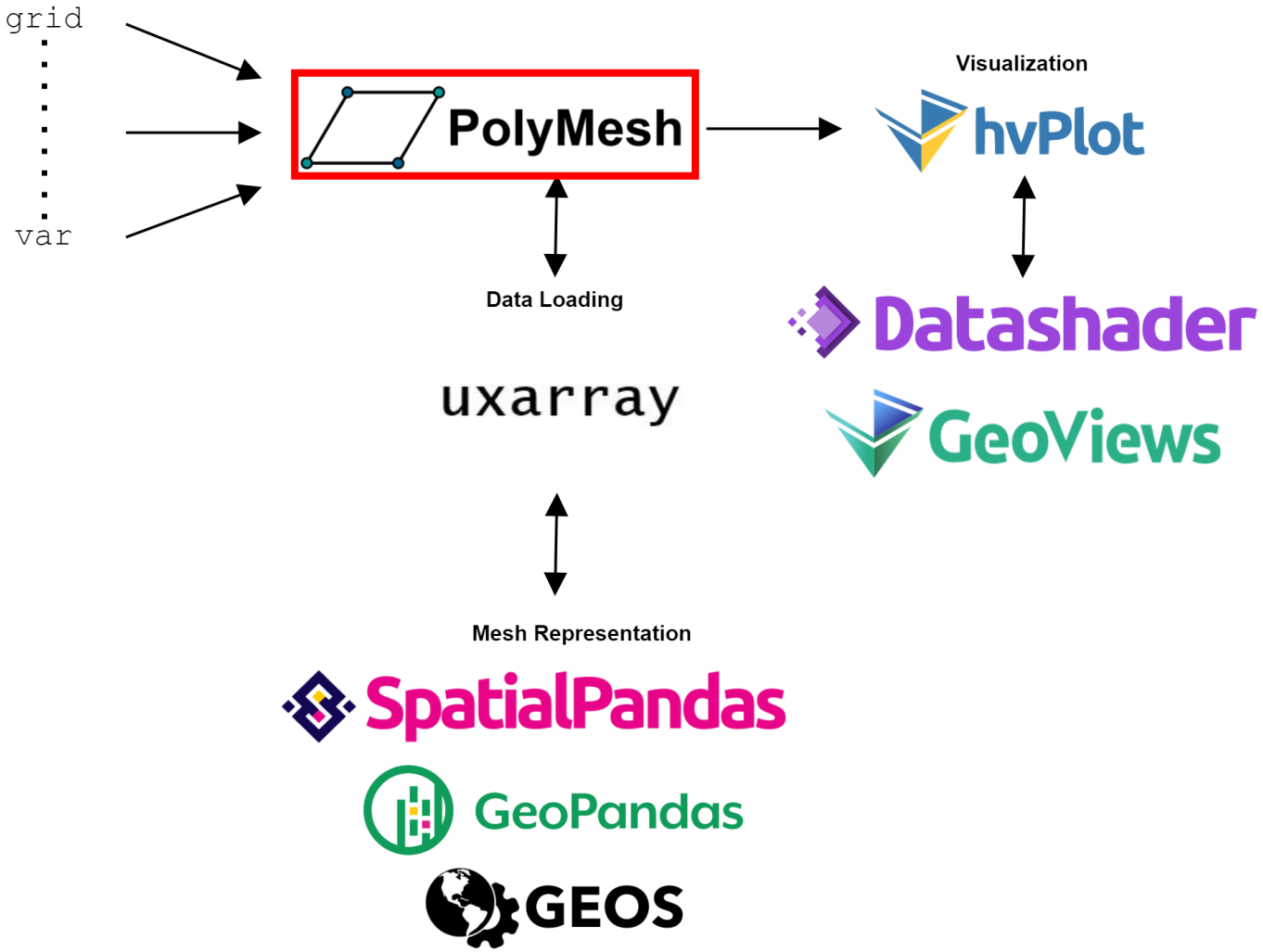


- Basic visualization workflows are more complex than MPL
- Requires our data to be in a specific format for rendering

Solution


grid
var
PolyMesh
Visualization
hvPlot
Data Loading
Datashader
uxarray
GeoViews
Mesh Representation
SpatialPandas
GeoPandas
GEOS

Solution



PolyMesh

- Tool for constructing unstructured meshes
- Compatible with most grid formats
  - UGRID, SCRIP, EXODUS, Shapefiles
- Native mesh construction & visualization
  - No need for Delaunay Triangulation
- Optimized geometric representation
  - Each cell is a polygon

geocat-scratch/polymesh

Data Loading

Uxarray

- Xarray-like package for unstructured grids
- Pure Python implementation (no compiled code)

Compatibility

- Support for standard unstructured conventions
  - UGRID, SCRIP, EXODUS, shapefiles
- Standardized variable access across conventions
- Groups grid and data files together for I/O

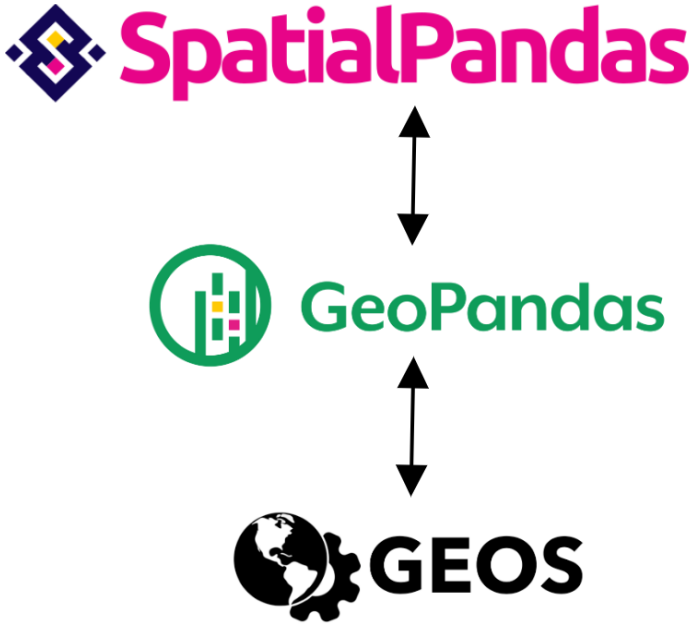
Mesh Representation

Geometry

- Each face is represented as a Polygon
  - Coordinates (x, y) define edge nodes
  - Face Nodes defines how they are connected

Optimized Libraries

- Polygons initially represented as Numpy arrays
- Efficiently converted to Shapely objects with PyGeos
  - Multithreading support
- Stored as a GeoDataFrame through SpatialPandas
  - Polygon Coordinates & Face Values

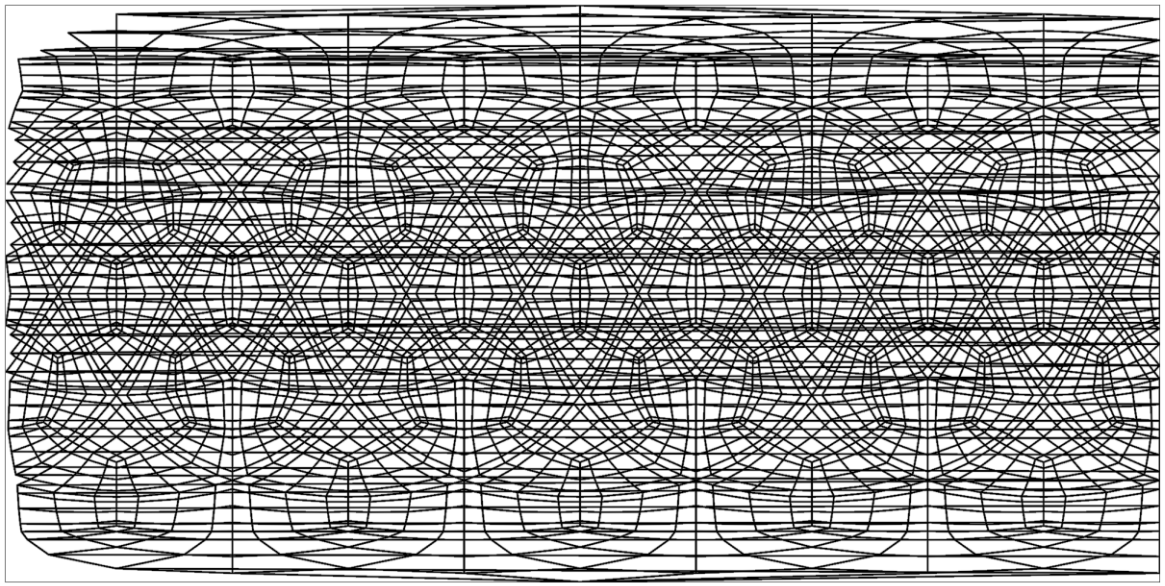


Mesh Representation

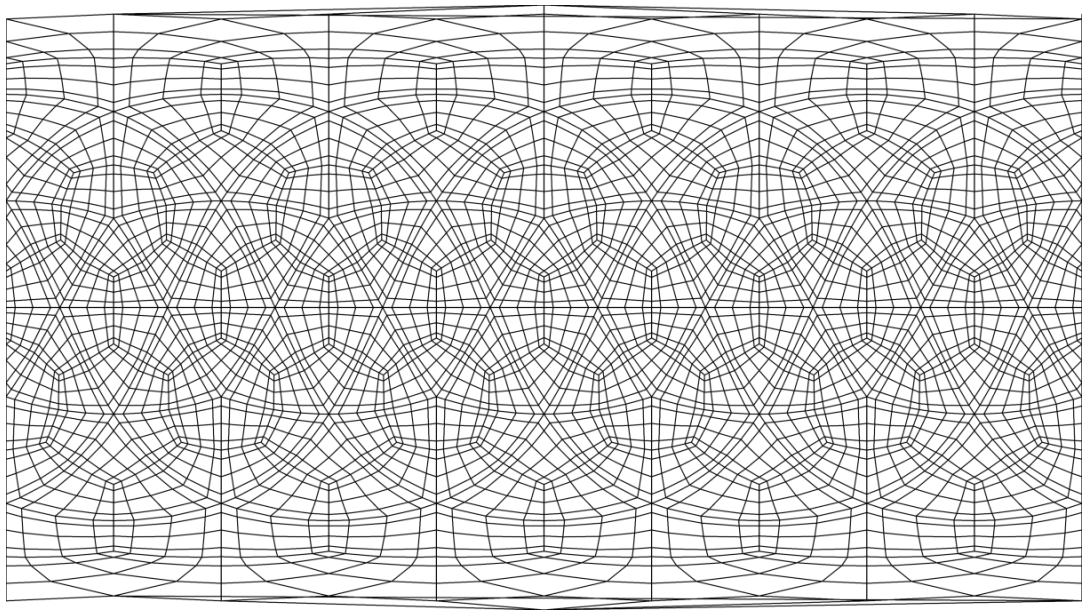
Cyclic Grid Cells

- Our grid may contain cells that wrap around from ±180 lon.
- Leads to noticeable artifacts when rendering
- Solution
  - Identify these polygons (longitude switches from (+) to (-)
  - Mirror them to the left and right side
  - Clip any data past |180| longitude
  - Ignore original when rendering

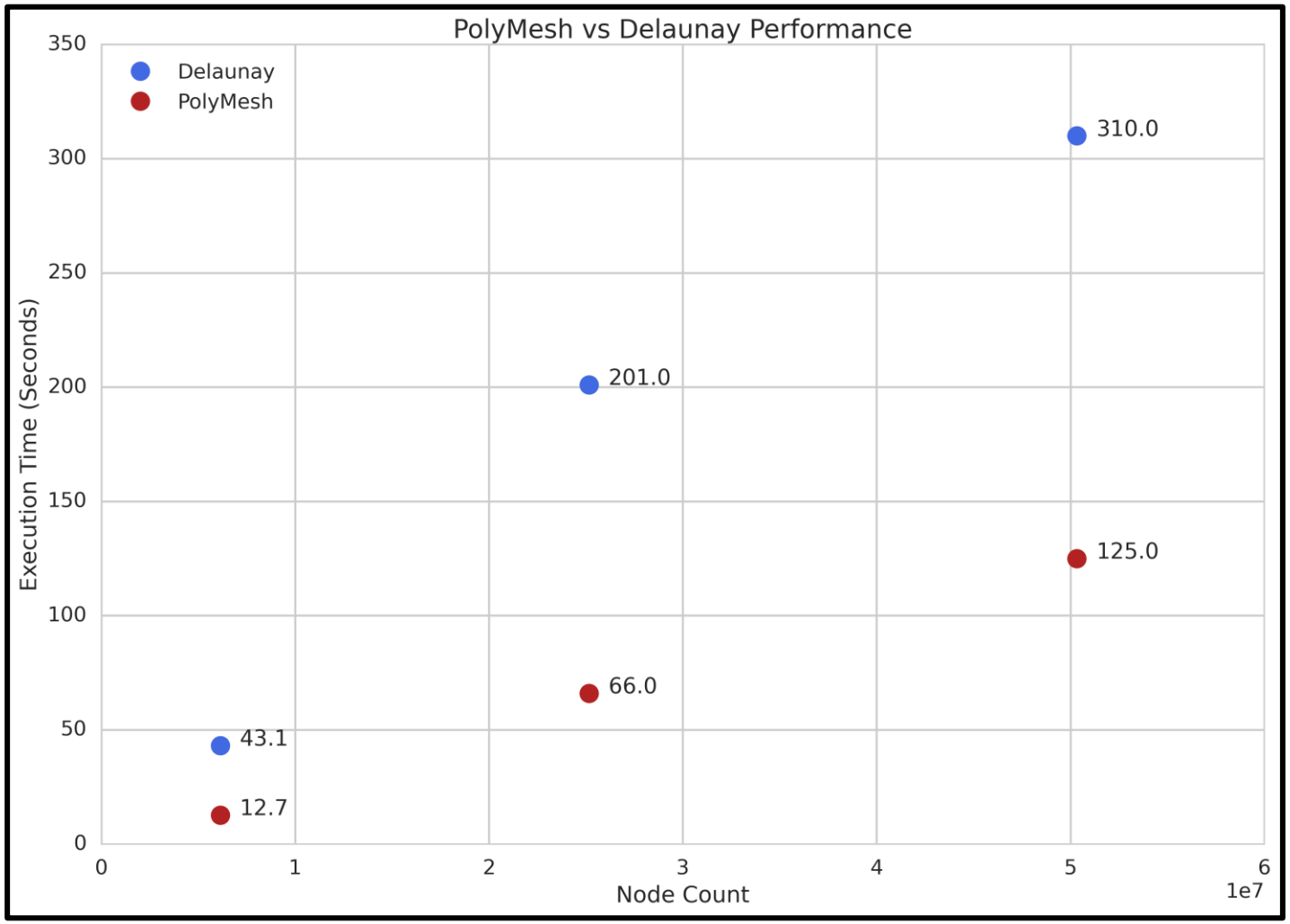
Original



Corrected



Performance



Test Configuration:
Single 128gb Node on Casper

Usage

Data Loading

```
import uxarray as ux
import hvplot.pandas
import cartopy.crs as ccrs
```

```
base_path = "/glade/p/cisl/vast/vapor/data/Source/UGRID/NOAA-geoflow/large/"

ugrid_large = ux.open_dataset(base_path + "grid.nc",
                              base_path + "v1.000000.nc",
                              base_path + "v2.000000.nc",
                              base_path + "v3.000000.nc")
```

Mesh Representation

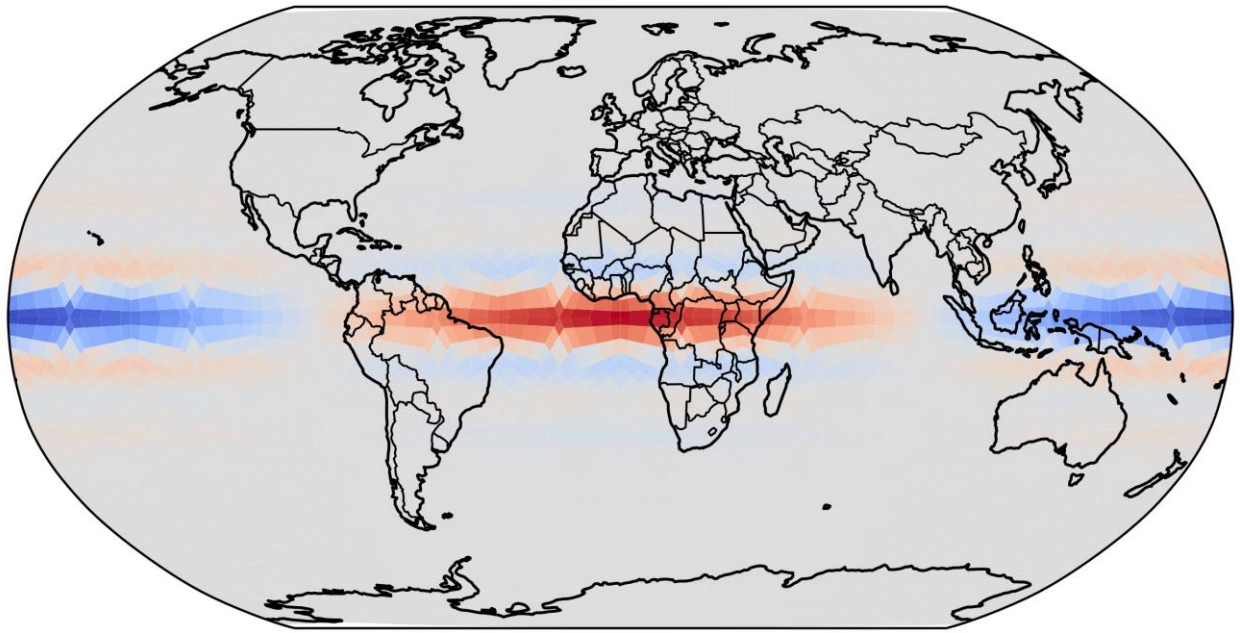
```
projection = ccrs.Robinson()
geoflow_small = Polymesh(ugrid=ugrid_large, projection=projection)
geoflow_small.construct_mesh()
```

Visualization

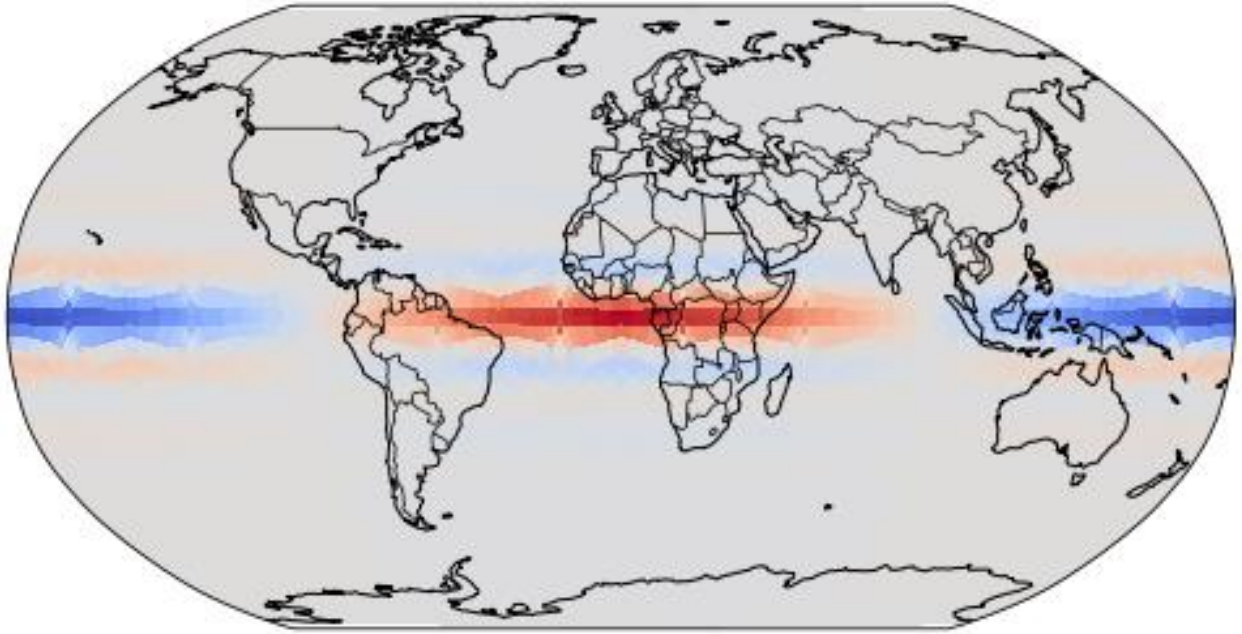
```
df = geoflow_large.data_mesh(name="v3", dims={"time" : 0, "meshLayers" : 0}, fill="nodes")
df.hvplot.polygons(rasterize=True,aggregator='mean', c='faces', cmap=cmap) * gf.coastline(projection=projection)
```

Results

Non-Rasterized

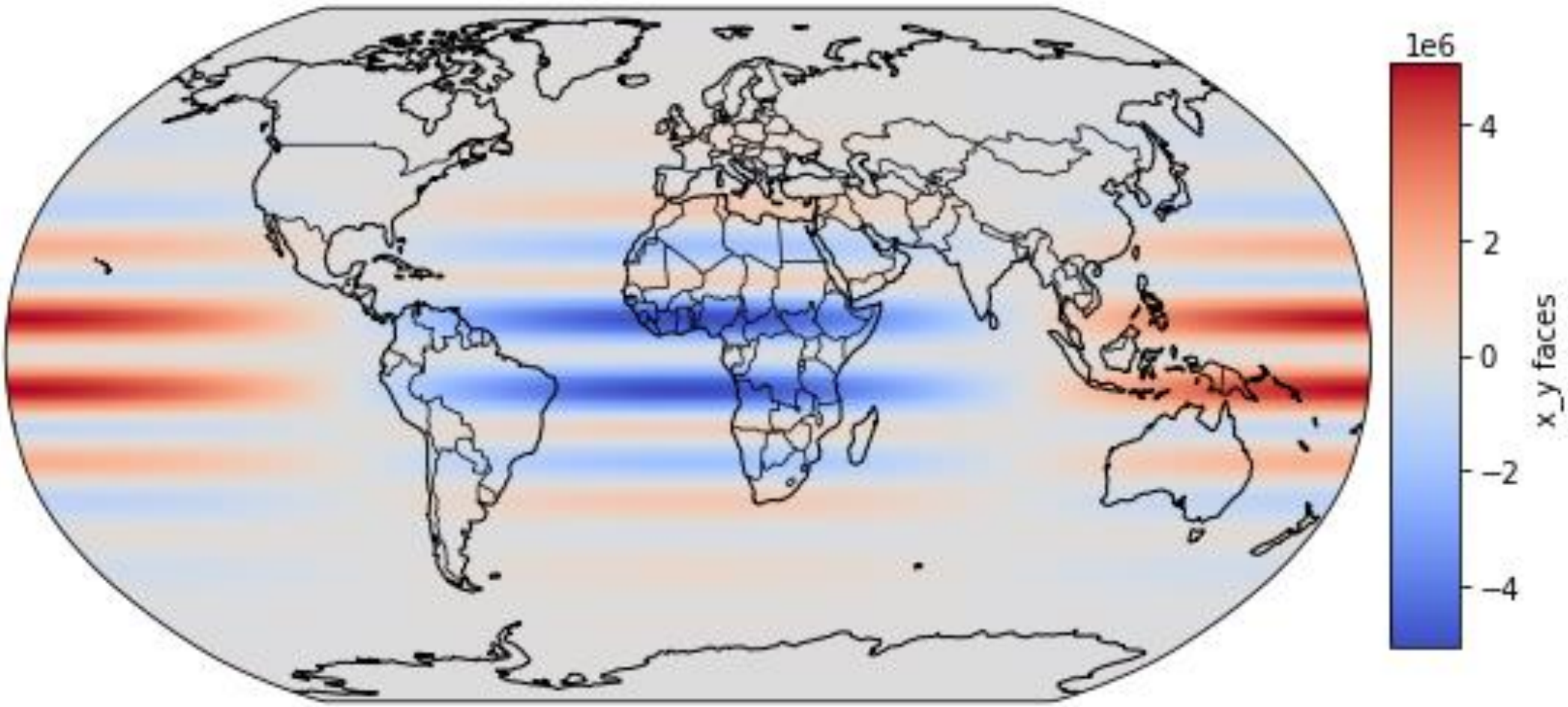


Rasterized



Results

Rasterized



UXarray

Data Access

```
import uxarray as ux
```

Original Approach

```
grid = uxr.open_dataset(grid_path, data_path)

x = grid.ds[grid.ds_var_names["Mesh2_node_x"]]
y = grid.ds[grid.ds_var_names["Mesh2_node_y"]]
face_nodes = grid.ds[grid.ds_var_names["Mesh2_face_nodes"]]
```

Data Access

```
import uxarray as ux
```

Original Approach

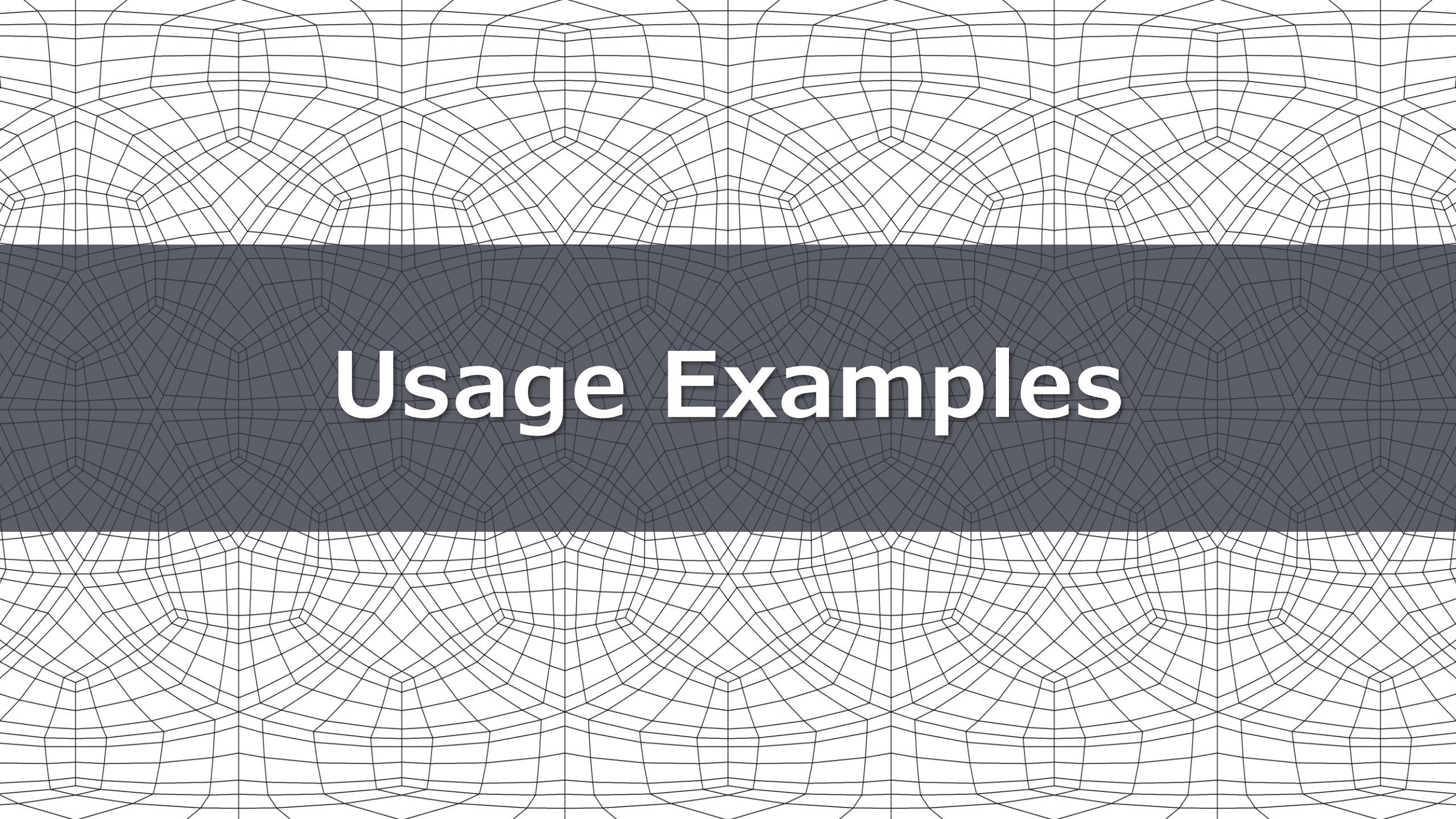
```
grid = uxr.open_dataset(grid_path, data_path)

x = grid.ds[grid.ds_var_names["Mesh2_node_x"]]
y = grid.ds[grid.ds_var_names["Mesh2_node_y"]]
face_nodes = grid.ds[grid.ds_var_names["Mesh2_face_nodes"]]
```

My Implementation

```
grid = uxr.open_dataset(grid_path, data_path)

x = grid.Mesh2_node_x
y = grid.Mesh2_node_y
face_nodes = grid.Mesh2_face_nodes
```


Usage Examples

Development

PolyMesh

↓

Uxarray

↓

Development

PolyMesh

↓

Example Notebook
Comparison Notebook
Performance Notebook
GeoFlow Example Notebook
SCREAMv0 Example Notebook

geocat-scratch/polymesh

Uxarray

↓

Development

PolyMesh

↓

Example Notebook
Comparison Notebook
Performance Notebook
GeoFlow Example Notebook
SCREAMv0 Example Notebook

geocat-scratch/polymesh

Uxarray

↓

Data Attribute Usage Example
Grid Format Conversion Example

Uxarray

Special Thanks

GeoCAT Team

Orhan Eroglu, Anissa Zacharias, Michaela Sizemore, Heather Craker, Mario Rodriquez, Alea Kootz, Daphne Quint, Alina Guha

Visualization Data

John Clyne (GeoFlow) & Paul Ullrich (SCREAMv0)

SIParCS Team

Virginia Do, Francesgladys Pudilo