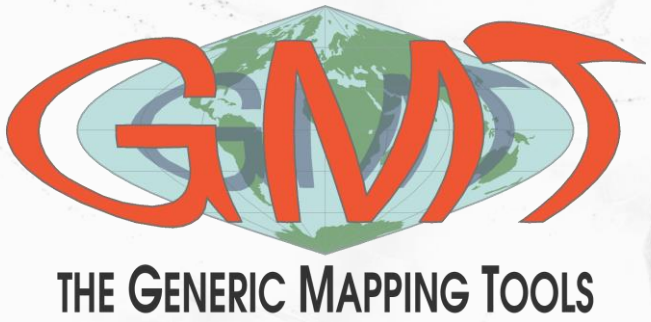
# PyGMT – Accessing and Integrating GMT with Python and the Scientific Python Ecosystem

AGU24 | Washington, D.C. | December 9, 2024 – **The impact of GMT in the Earth, Ocean and Space sciences: What's next?** – U12B-05

Yvonne **Fröhlich** | Dongdong **Tian** | Wei Ji **Leong** | Max **Jones** | Michael **Grund**

https://www.pygmt.org/ – https://github.com/GenericMappingTools/pygmt – https://www.generic-mapping-tools.org/ – https://forum.generic-mapping-tools.org/






OCE-1558403, EAR-1948602

# Motivation for Wrappers

```
gmt coast -JN10c -Rg -B -Ggray -png worldmap
```


- Open source / access
- Multi-platform support
- Processing of spatial data
- High-quality publication-ready figures

- Readability & intuitiveness of the code
- Getting started & learning curve
- Preparation & handling of input data
- Integration in the overall workflow

# Motivation for Wrappers & Goals of the project

```
gmt coast -JN10c -Rg -B -Ggray -png worldmap
```


- Open source / access
- Multi-platform support
- Processing of spatial data
- High-quality publication-ready figures

→ **Keep it!**

- Readability & intuitiveness of the code
- Getting started & learning curve
- Preparation & handling of input data
- Integration in the overall workflow

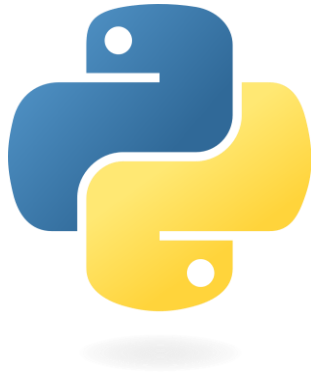
→ **Improve it!**

Integrate **GMT** with **Python** & **scientific Python ecosystem**

# Python

```
gmt coast -JN10c -Rg -B -Ggray -png worldmap
```

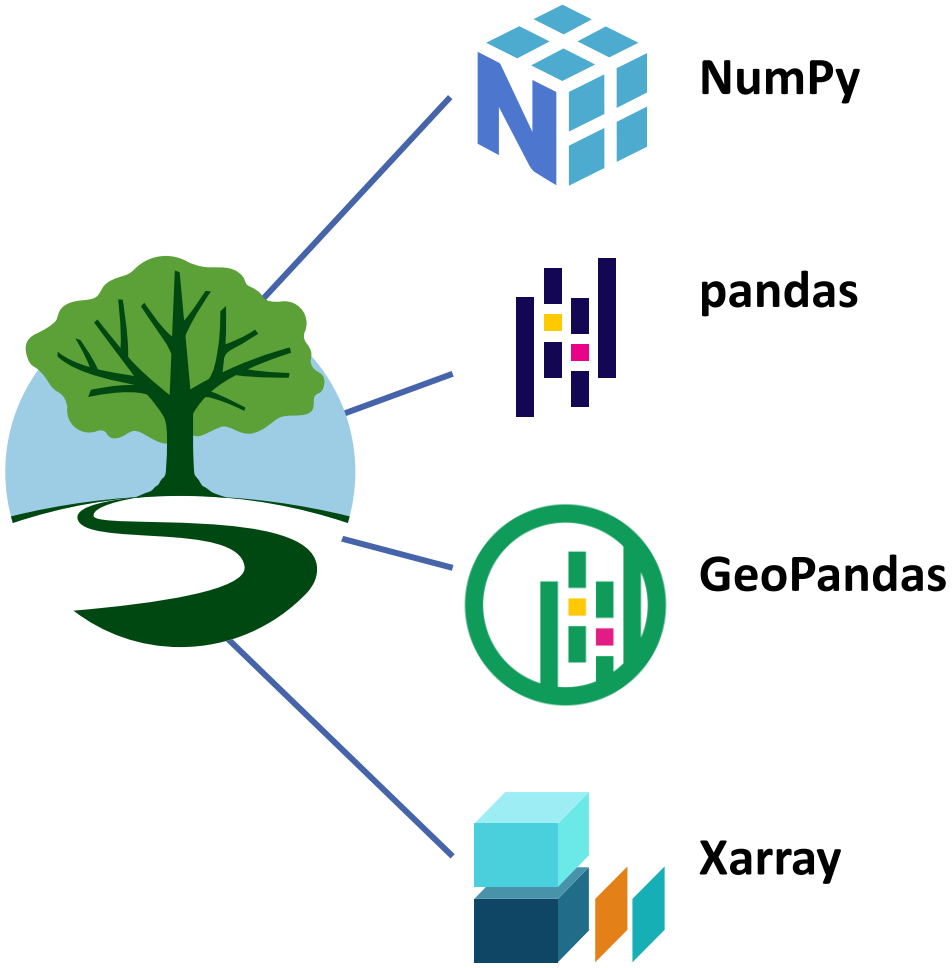
- Easy to get started
- Readability & intuitiveness
- Open source / access
- Multi-platform support



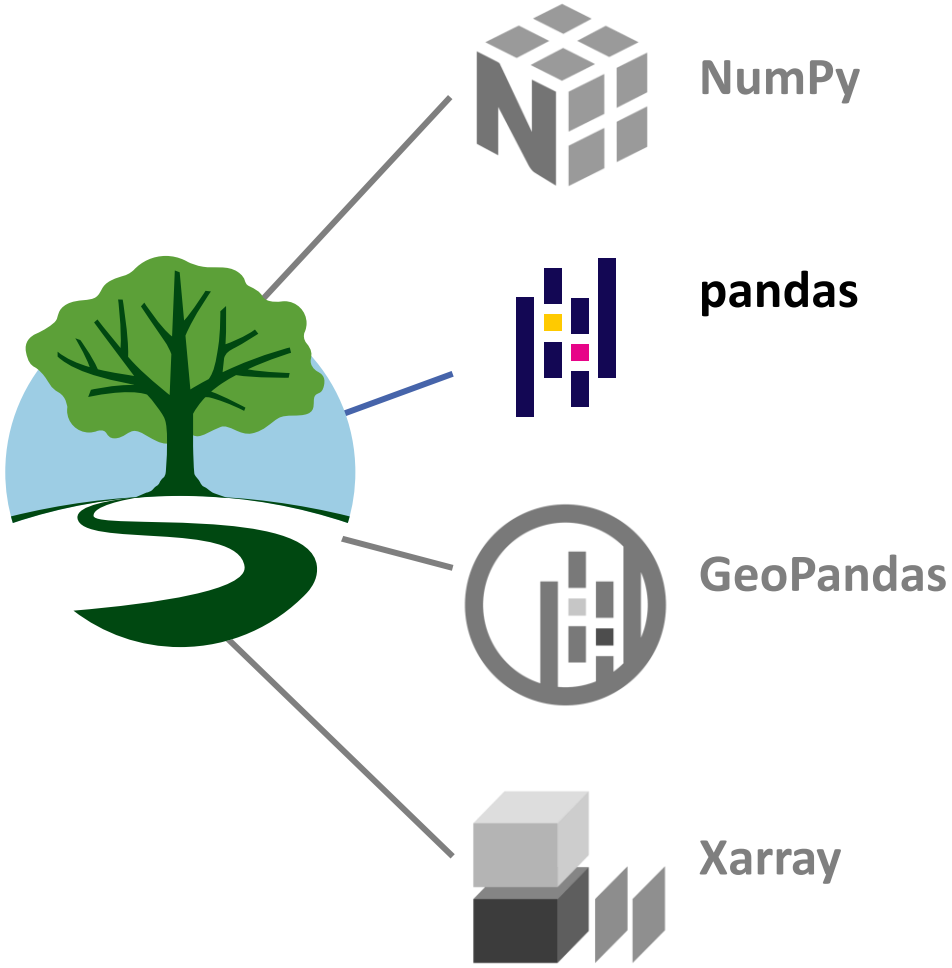
- Interact with GMT C API, requires **modern mode** introduced in GMT 6
- Use **functions**, **classes**, and **methods**
  - pygmt.Figure() class for plotting
- **Alias system**
  - Introduce descriptive names for flags, derivatives, modifiers

```
import pygmt

fig = pygmt.Figure()
fig.coast(projection="N10c", region="g", frame=True, land="gray")
fig.show()
fig.savefig(fname="worldmap.png")
```

# Scientific Python Ecosystem (SPE)



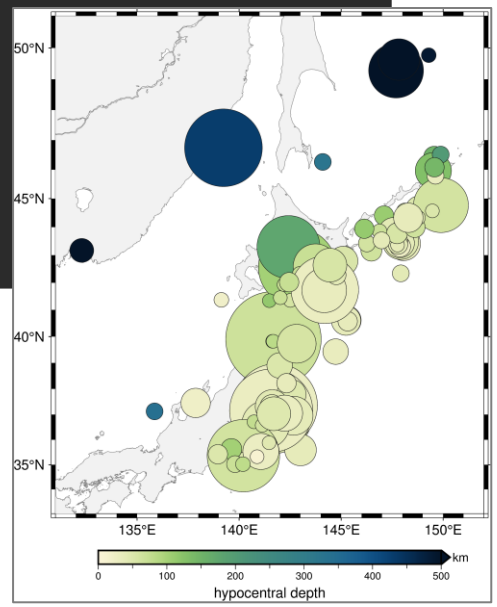
# Scientific Python Ecosystem (SPE)



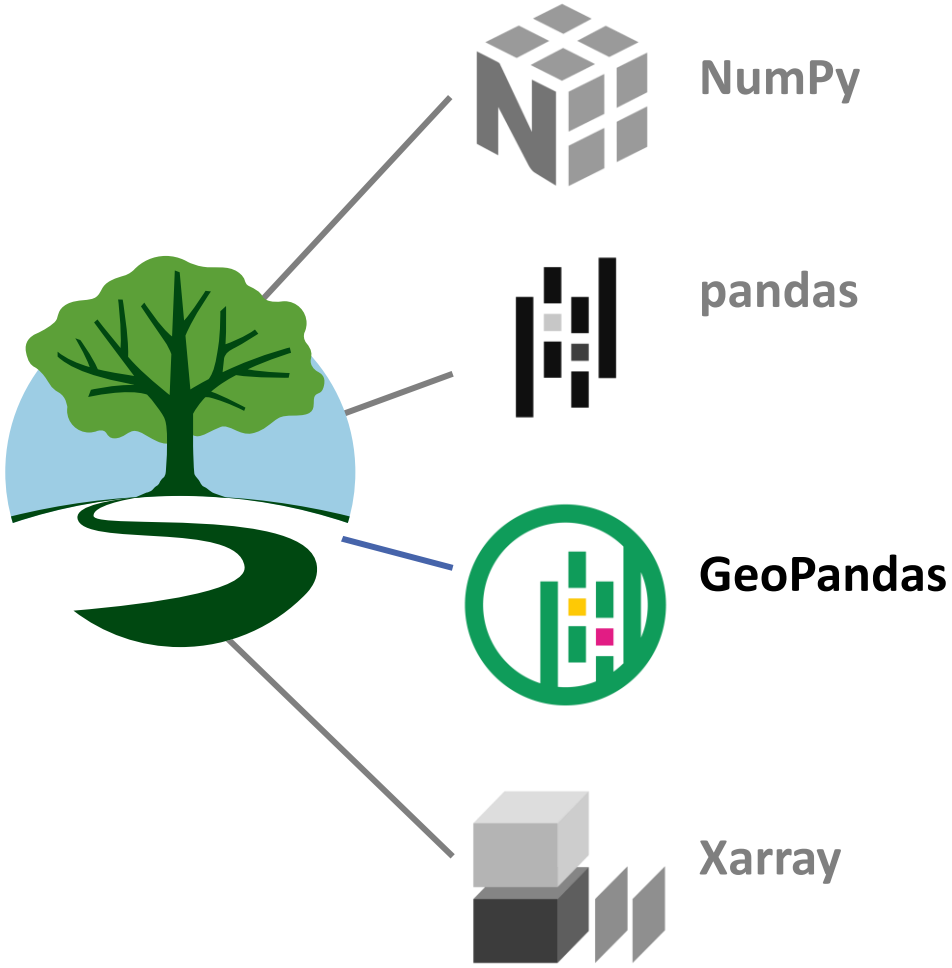

**Tabular data** → pandas.DataFrame

```python
import pygmt
import pandas as pd

# Load tabular data into a pandas.DataFrame
df_eqs = pd.read_csv("my_japan_quakes.csv")
# OR
df_eqs = pygmt.datasets.load_sample_data(name="japan_quakes")

fig = pygmt.Figure()
# Plot the epicenters as color- and size-coded circels based on depth or magnitude
fig.plot(
    x=df_eqs.longitude,
    y=df_eqs.latitude,
    size=0.02 * 2**df_eqs.magnitude,
    fill=df_eqs.depth_km,
    cmap=True,
    style="cc",
    pen="gray10",
)
fig.show()
```

Complete code example available at [pandas_example](pandas_example).



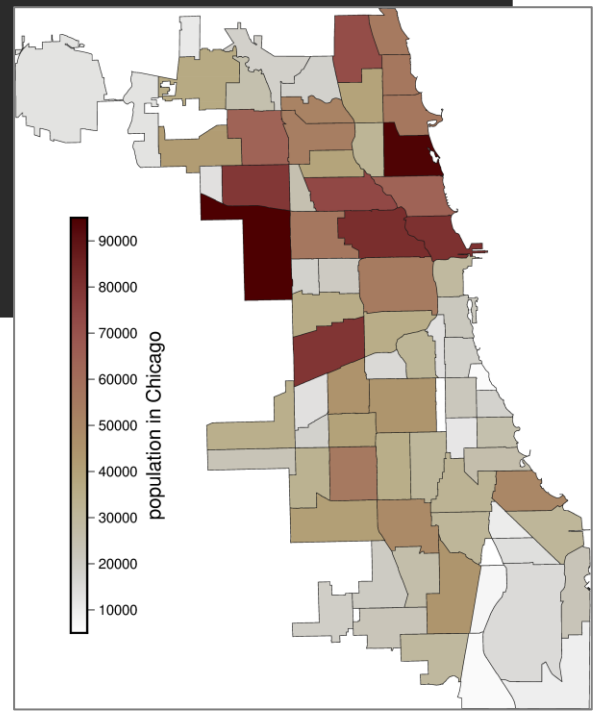
# Scientific Python Ecosystem (SPE)



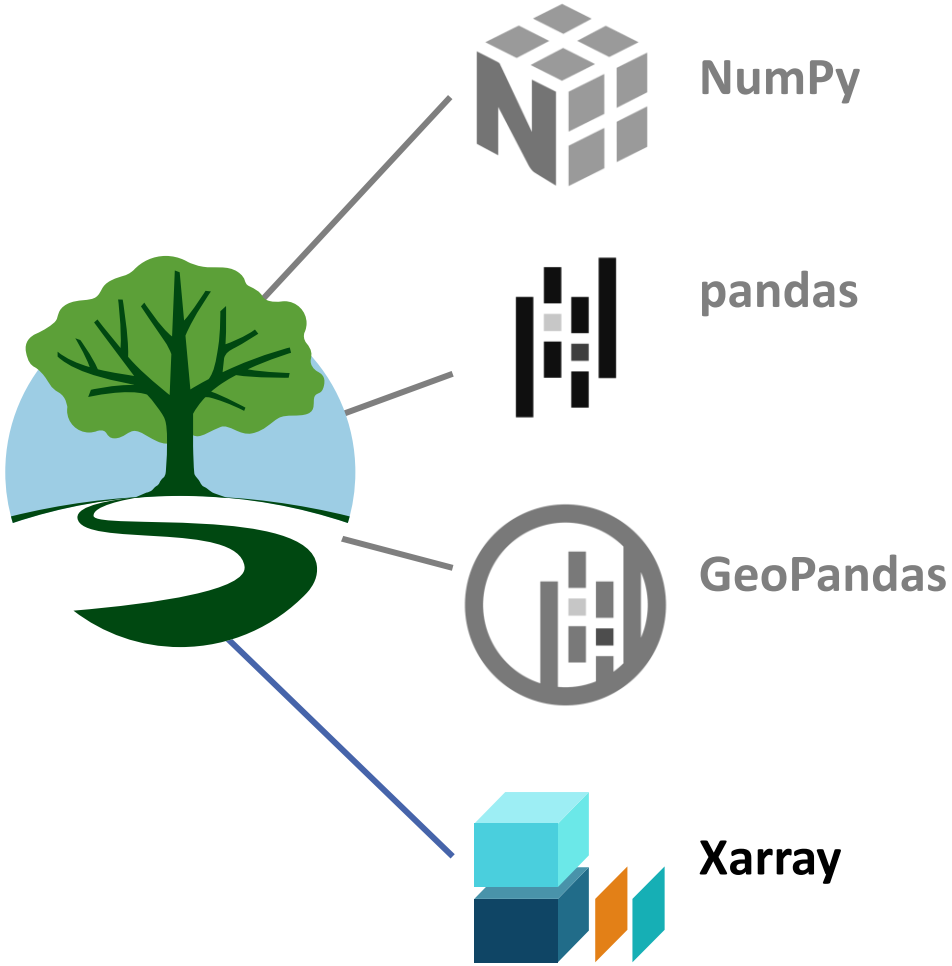

**Spatial data (points, lines, polygons)** → geopandas.GeoDataFrame

```python
import pygmt
import geopandas as gpd

# Download a dataset about Chicago into a geopandas.GeoDataFrame
gdf_airbnb = gpd.read_file("https://geodacenter.github.io/data-and-lab/data/airbnb.zip")

fig = pygmt.Figure()
# Plot the polygons with color-coding for the population
fig.plot(
    data=gdf_airbnb,
    fill="+z",
    aspatial="Z=population",
    cmap=True,
    pen="0.2p,gray10",
)
fig.show()
```

Complete code example available at geopandas_example.



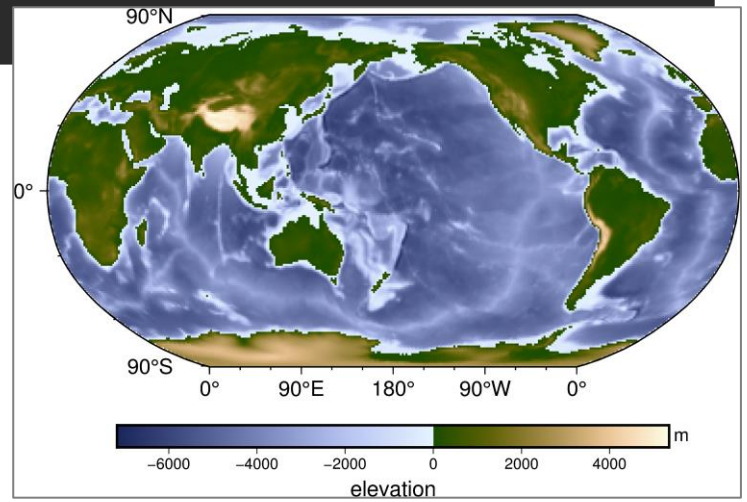
# Scientific Python Ecosystem (SPE)




**Gridded data** → xarray.DataArray

```python
import pygmt
import xarray as xr

# Download an elevation grid into a xarray.DataArray
da_ele = pygmt.datasets.load_earth_relief(resolution="01d")

fig = pygmt.Figure()
# Plot the grid with color-coding for the elevation
fig.grdimage(grid=da_ele, cmap="oleron")
fig.show()
```

Complete code example available at [xarray_example](xarray_example).



# More functionality & User friendliness

## **Background map** → contextily: tiled maps

```python
import pygmt
import contextily

fig = pygmt.Figure()
fig.tilemap(
    region=[-77.2, -76.7, 38.7, 39],
    projection="M10c",
    zoom=10,
    source="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png",
    frame=True,
)
fig.show()
```


Complete code example available at [contextily_example].

## **Auto-completion** → type hints

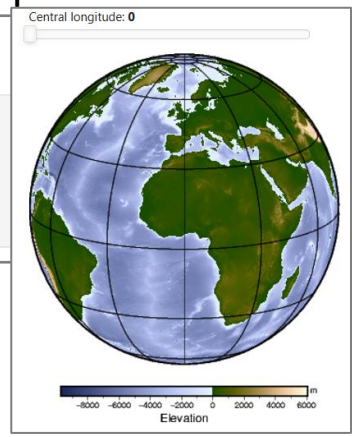
```python
import pygmt

# Load tabular data into a pandas.DataFrame
df_eqs = pygmt.datasets.load_sample_data(
  name=
```

```
load_sample_data(name: Literal[ "bathymetry",
                 "earth_relief_holes", "fractures",
                 "hotspots", "japan_quakes", "mars_shape",
                 "maunaloa_co2", "notre_dame_topography",
                 "ocean_ridge_points", "rock_compositions",
                 "usgs_quakes", ]) -> pd.DataFrame |
                 xr.DataArray

Load an example dataset from the GMT server.

The data are downloaded to a cache directory (usually
``~/.gmt/cache``) the first time you invoke this function.
Afterwards, it will load the data from the cache. So you'll
need an internet connection the first time around.

Parameters
----------
name ...
```

## **Notebook environment** → Jupyter: rich display, panel: interactivity

Import the required packages

```python
import numpy as np
import panel as pn
import pygmt

pn.extension()
```



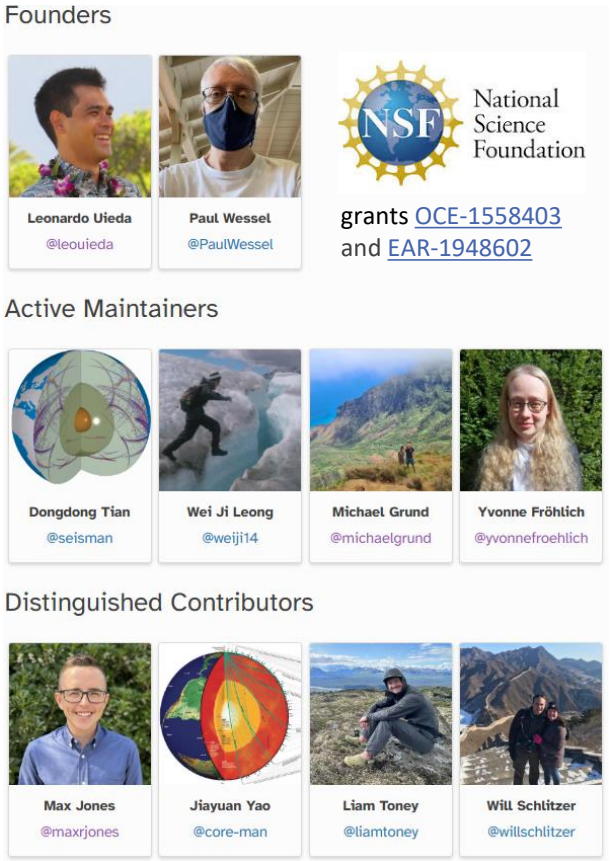

Complete tutorial available at [working with panel].

# Documentation & Development

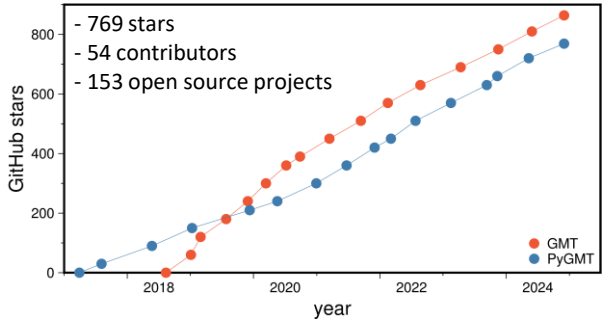
## www.pygmt.org



## https://github.com/GenericMappingTools/pygmt




- Project started in 2017 by Leonardo Uieda
- Now an international community-driven project
- Issue – pull request – review workflow
- Semantic versioning, backwards compatibility policy
- Contributions: besides code, also documentation, examples, review, and bug reports, feature requests
- New contributors: contributors guide, good first issue label, code of conduct




Data retrieved from https://star-history.com, last access 2024/12/01.

Complete code example available at **numpy_example**.

# Next steps

- **Alias system**
  - Main GMT modules are wrapped
    Some remaining modules are grdmask, movie, coupe

  → https://github.com/orgs/GenericMappingTools/projects/3/views/1

  - Handling of long-string arguments
    Introduce parameters, classes, or dictionaries?

  → https://github.com/GenericMappingTools/pygmt/issues/1082

```
position="jTR+jMC+o0.3c/0.6c+w3c"
```

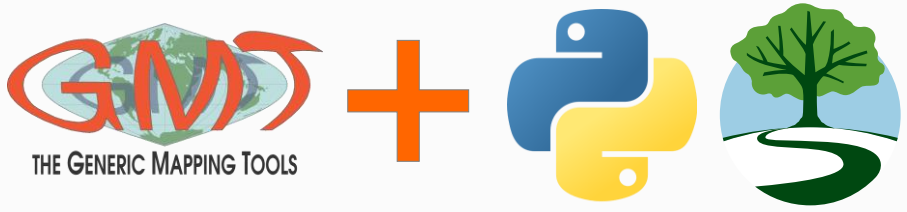
- **High-level plotting methods**
  - Scatter plot, error bars, stem plot, etc.

  → https://github.com/GenericMappingTools/pygmt/issues/2797

  - Scale bar, direction rose, magnetic rose

  → https://github.com/GenericMappingTools/pygmt/issues/2831

# A Python Interface for the Generic Mapping Tools



www.pygmt.org

- **Python, alias system**: Improvement of the readability / intuitiveness of the code
- **SPE**: Support of NumPy, pandas, GeoPandas, Xarray to handle data types for tabular, spatial, and gridded data
- **Visualizations**: Reproducibility of GMT figures plus more functionality like choropleth map, tiled maps, interactivity
- **User friendliness**: Support of type hints & Jupyter notebook

**GMT**

```
gmt coast -JN10c -Rg -B -Ggray -png worldmap
```

**PyGMT**

```
import pygmt

fig = pygmt.Figure()
fig.coast(projection="N10c", region="g", frame=True, land="gray")
fig.show()
fig.savefig(fname="worldmap.png")
```

- **mamba / conda, PyPi / pip**: Easy installation within a virtual environment
- **Discord forum**: Getting help → https://forum.generic-mapping-tools.org
- **GitHub**: Source code → https://github.com/GenericMappingTools/pygmt
- **Zenodo**: DOI for proper citation → https://doi.org/10.5281/zenodo.3781524
- **PyOpenSci**: Peer-reviewed → https://www.pyopensci.org/python-packages.html

Tian D, Uieda L, Leong W J, Fröhlich Y, Schlitzer W, Grund M, Jones M, Toney L, Yao J, Magen Y, Tong J-H, Materna K, Belem A, Newton T, Anant A, Ziebarth M, Quinn J, Wessel P. (2024). PyGMT: A Python interface for the Generic Mapping Tools (v0.13.0). Zenodo. https://doi.org/10.5281/zenodo.13679420.

The development of PyGMT has been supported by NSF grants OCE-1558403 and EAR-1948602.