

# A modern **Python** interface for the Generic Mapping Tools

Leonardo Uieda\* and Paul Wessel

Department of Geology and Geophysics, SOEST, University of Hawai'i at Mānoa

## Introduction

The Generic Mapping Tools (GMT) are open-source programs for processing geospatial data and **making beautiful maps**. Python is one of the fastest growing languages for **scientific computing**. We are building a **bridge** to bring the power of GMT to the Python ecosystem.

## Project goals

Be modern: Python 3.5+ and GMT6 only.  
Provide a simple and Pythonic interface.  
Use the GMT C API instead of system calls.

Readable aliases for GMT arguments.  
Integrate with numpy, pandas, and xarray.  
Support for the Jupyter notebook.

## Development stage

Finished ~70% of the C API wrapper (LibGMT).  
Jupyter integration through the Figure class.  
Automated tests with > 90% code coverage.  
Heavy use of decorators and context managers.

Only a few modules wrapped.  
Working on: retrieving data from GMT modules.  
pandas and xarray integration.  
Windows support.

## Contact and contribute

🏠 [www.gmtpython.xyz](http://www.gmtpython.xyz)  
🔄 [github.com/GenericMappingTools](https://github.com/GenericMappingTools)  
✉ [leouieda@gmail.com](mailto:leouieda@gmail.com)  
🐦 [@leouieda](https://twitter.com/leouieda)

We welcome contributions: code, ideas, bugs.  
**Anyone can contribute**, regardless of skill level.  
We have a Code of Conduct to keep you safe.  
The project is open-source (BSD license).

## The GMT/Python library

```
import gmt, numpy
lon, lat, magnitude = numpy.loadtxt("usgs_quakes.txt", unpack=True)
fig = gmt.Figure()
fig.coast(region=[-270, 90, -70, 70], projection="M10i", land="#aaaaaa",
          water="white", resolution="1")
fig.plot(lon, lat, sizes=0.02*1.5**magnitude, style="cc", cmap="ocean",
         color=magnitude/magnitude.max())
fig.savefig("poster_background_inception.png", dpi=1000, show=True)
fig.show()
```

## Interacting with the GMT C API

```
@fmt_docstring
@use_alias(R="region", J="projection", B="frame", P="portrait", ...)
@kwargs_to_strings(R="sequence", i="sequence_comma")
def plot(self, x=None, y=None, sizes=None, **kwargs):
    "Plot lines, polygons, and symbols on maps."
    with LibGMT() as lib:
        with lib.vectors_to_vfile(x, y) as vfile:
            arg_str = " ".join([vfile, build_arg_string(kwargs)])
            lib.call_module("plot", arg_str)
```



Try an online demo!

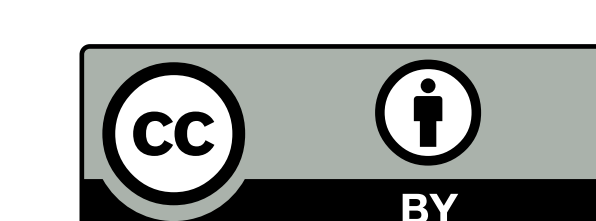
[github.com/leouieda/agu2017](https://github.com/leouieda/agu2017)



Download the poster

[doi:10.6084/m9.figshare.5662411](https://doi.org/10.6084/m9.figshare.5662411)

Feel free to photograph or share this poster.



This poster is licensed Creative Commons Attribution 4.0.

This project is supported by grant  
OCE-1558403 from the US  
National Science Foundation.

