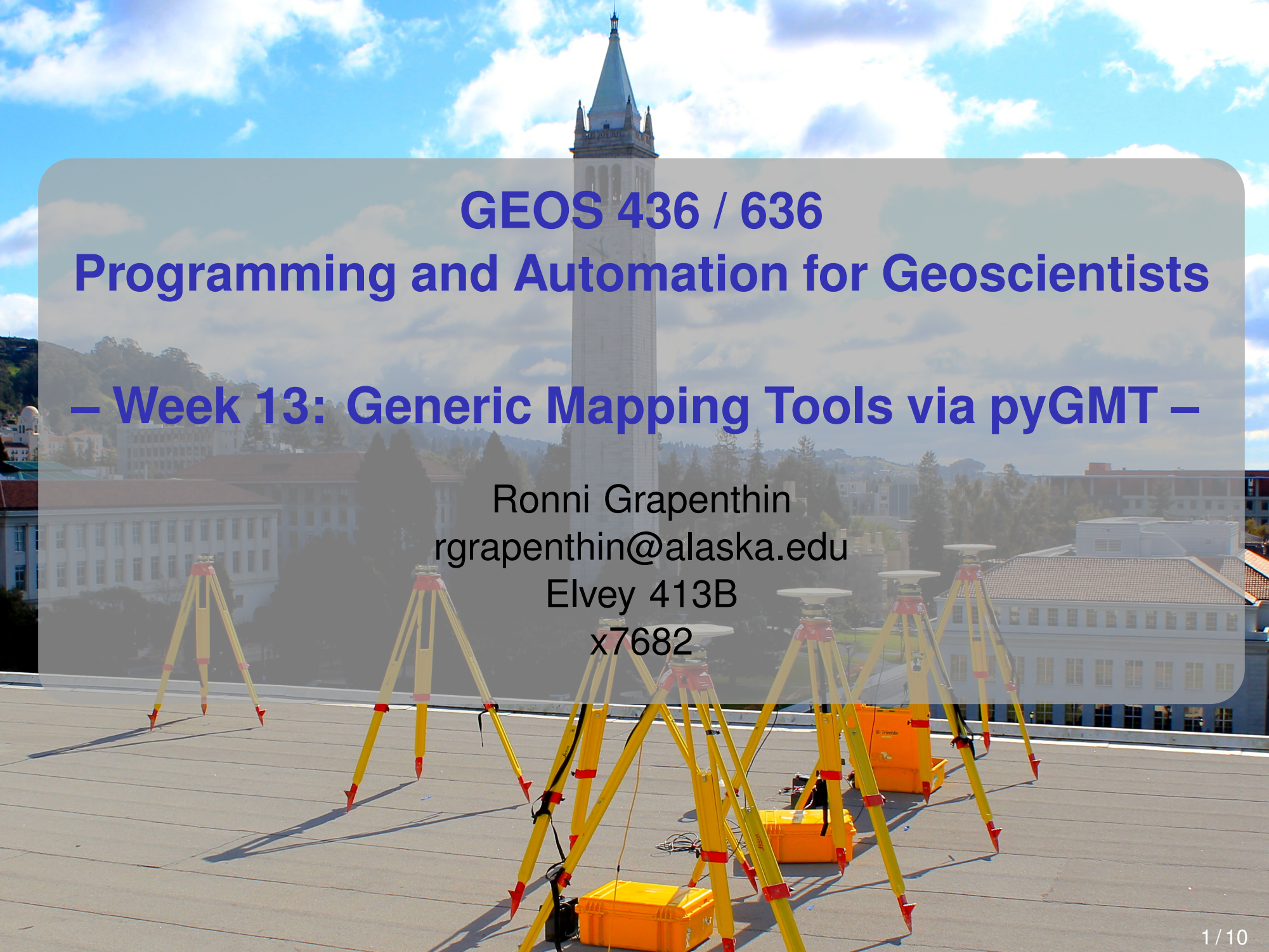


GEOS 436 / 636
Programming and Automation for Geoscientists

– Week 13: Generic Mapping Tools via pyGMT –

Ronni Grapenthin
rgrapenthin@alaska.edu
Elvey 413B
x7682

How to automate making publication-quality:

- maps,
- x-y plots,
- animations

using world class base data sets while having maximum flexibility regarding layout of your product?

https://www.pygmt.org/latest/index.html

pyGMT Goals

Project goals

- Make GMT more accessible to new users.
- Build a Pythonic API for GMT.
- Interface with the GMT C API directly using ctypes (no system calls).
- Support for rich display in the Jupyter notebook.
- Integration with the PyData ecosystem: `numpy.ndarray` or `pandas.DataFrame` for data tables and `xarray.DataArray` for grids.

https://www.pygmt.org/latest/index.html

pyGMT.Figure - base class

pygmt.Figure

class **pygmt.Figure** [source]

A GMT figure to handle all plotting.

Use the plotting methods of this class to add elements to the figure. You can preview the figure using `pygmt.Figure.show` and save the figure to a file using `pygmt.Figure.savefig`.

Unlike traditional GMT figures, no figure file is generated until you call `pygmt.Figure.savefig` or `pygmt.Figure.psconvert`.

Examples

```
>>> fig = Figure()
>>> fig.basemap(region=[0, 360, -90, 90], projection="W7i", frame=True)
>>> fig.savefig("my-figure.png")
>>> # Make sure the figure file is generated and clean it up
>>> import os
>>> os.path.exists("my-figure.png")
True
>>> os.remove("my-figure.png")
```

https://www.pygmt.org/latest/api/generated/pygmt.Figure.html#pygmt.Figure

pyGMT.Figure - methods

`Figure.basemap` (*[, region, projection, ...])	Plot base maps and frames for the figure.
`Figure.coast` (*[, region, projection, ...])	Plot continents, shorelines, rivers, and borders on maps
`Figure.colorbar` (*[, region, projection, ...])	Plot a gray or color scale-bar on maps.
`Figure.contour` ([data, x, y, z, annotation, ...])	Contour table data by direct triangulation.
`Figure.grdcontour` (grid, *[, annotation, ...])	Convert grids or images to contours and plot them on maps.
`Figure.grdimage` (grid, *[, img_out, frame, ...])	Project and plot grids or images.
`Figure.grdview` (grid, *[, region, ...])	Create 3-D perspective image or surface mesh from a grid.
`Figure.histogram` (data, *[, horizontal, ...])	Plots a histogram, and can read data from a file or list, array, or dataframe.
`Figure.image` (imagefile, *[, region, ...])	Place images or EPS files on maps.
`Figure.inset` (*[, position, box, margin, ...])	Create an inset figure to be placed within a larger figure.
`Figure.legend` ([spec, position, box, region, ...])	Plot legends on maps.
`Figure.logo` (*[, region, projection, ...])	Plot the GMT logo.
`Figure.meca` (spec, scale[, longitude, ...])	Plot focal mechanisms.
`Figure.plot` ([data, x, y, size, direction, ...])	Plot lines, polygons, and symbols in 2-D.
`Figure.plot3d` ([data, x, y, z, size, ...])	Plot lines, polygons, and symbols in 3-D.
`Figure.psconvert` (*[, crop, gs_option, dpi, ...])	Convert [E]PS file(s) to other formats.
`Figure.rose` ([data, length, azimuth, sector, ...])	Plot windrose diagrams or polar histograms.
`Figure.savefig` (fname[, transparent, crop, ...])	Save the figure to a file.

pyGMT comes loaded - Datasets

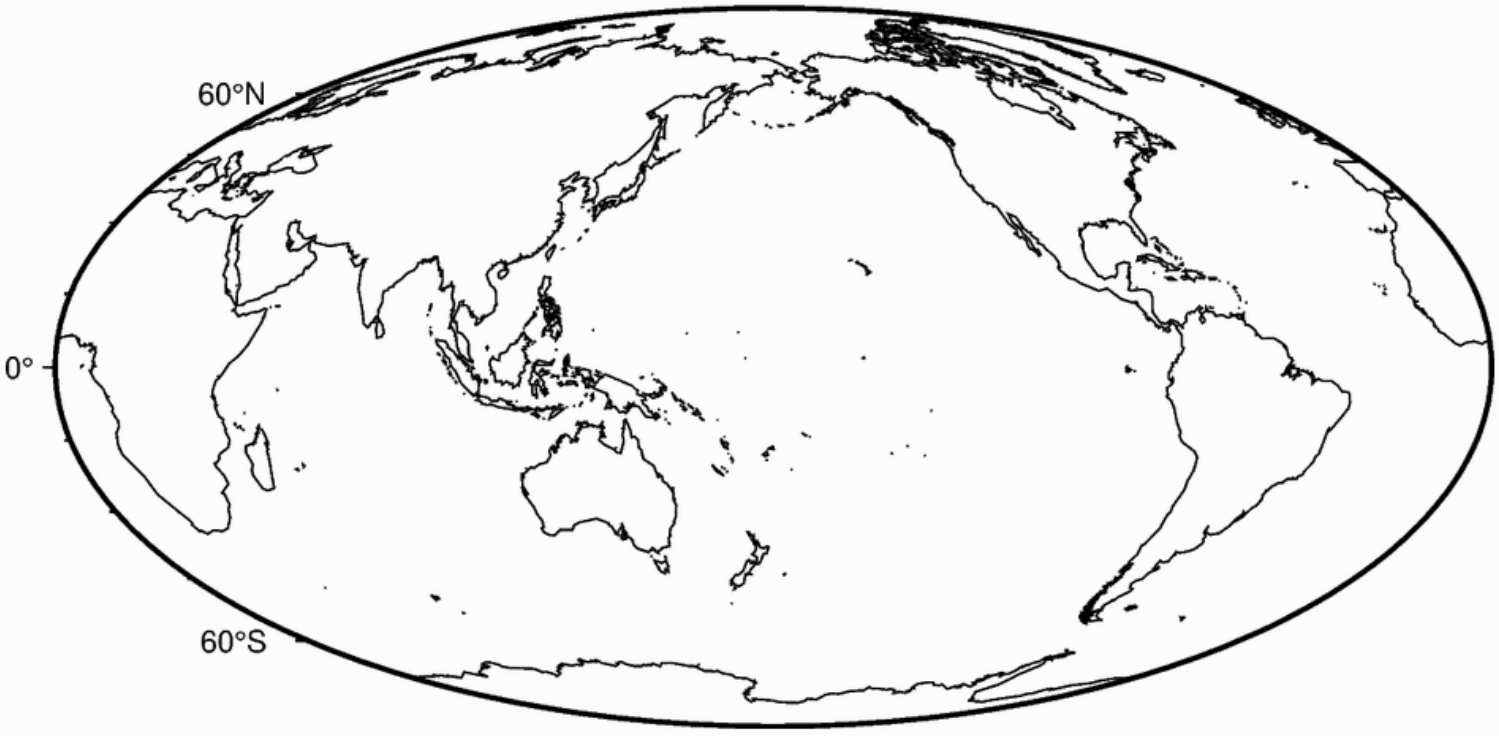
PyGMT provides access to GMT's datasets through the `pygmt.datasets` package. These functions will download the datasets automatically the first time they are used and store them in the GMT cache folder.

`datasets.load_earth_age` ([resolution, ...])	Load Earth seafloor crustal ages in various resolutions.
`datasets.load_earth_relief` ([resolution, ...])	Load Earth relief grids (topography and bathymetry) in various resolutions.
`datasets.load_fractures_compilation` ()	Load a table of fracture lengths and azimuths as hypothetically digitized from geological r
`datasets.load_hotspots` ()	Load a table with the locations, names, and suggested symbol sizes of hotspots.
`datasets.load_japan_quakes` ()	Load a table of earthquakes around Japan as a pandas.DataFrame.
`datasets.load_mars_shape` ()	Load a table of data for the shape of Mars.
`datasets.load_ocean_ridge_points` ()	Load a table of ocean ridge points for the entire world as a pandas.DataFrame.
`datasets.load_sample_bathymetry` ()	Load a table of ship observations of bathymetry off Baja California as a pandas.DataFrame
`datasets.load_usgs_quakes` ()	Load a table of global earthquakes form the USGS as a pandas.DataFrame.

https://www.pygmt.org/latest/api/index.html#datasets

pyGMT comes loaded - Coastlines

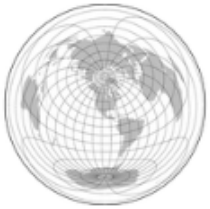
```
fig = pygmt.Figure()
fig.basemap(region="g", projection="W15c", frame=True)
fig.coast(shorelines="1/0.5p,black")
fig.show()
```



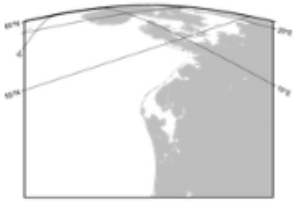
https://www.pygmt.org/latest/tutorials/coastlines.html

GMT comes loaded - The 31 Projections

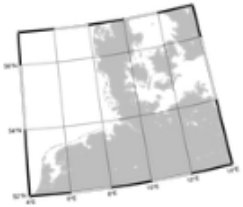
Azimuthal Projections



Azimuthal Equidistant



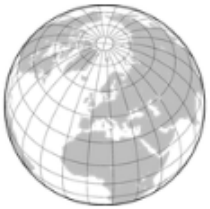
General Perspective



General Stereographic

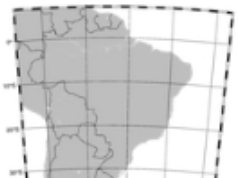
Gnomonic

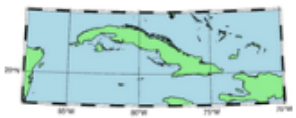
Lambert Azimuthal Equal Area



Orthographic

Conic Projections





GMT comes loaded - The 31 Projections

PyGMT Projection Argument	Projection Name
Alon$_0$/lat$_0$[/*horizon*]/*width*	Lambert azimuthal equal area
Blon$_0$/lat$_0$/lat$_1$/lat$_2$/*width*	Albers conic equal area
Clon$_0$/lat$_0$/*width*	Cassini cylindrical
Cyl_stere/[lon$_0$[/lat$_0$/]]*width*	Cylindrical stereographic
Dlon$_0$/lat$_0$/lat$_1$/lat$_2$/*width*	Equidistant conic
Elon$_0$/lat$_0$[/*horizon*]/*width*	Azimuthal equidistant
Flon$_0$/lat$_0$[/*horizon*]/*width*	Azimuthal gnomonic
Glon$_0$/lat$_0$[/*horizon*]/*width*	Azimuthal orthographic
Glon$_0$/lat$_0$/*alt*/*azim*/*tilt*/*twist*/*W*/*H*/*width*	General perspective
H[lon$_0$/]*width*	Hammer equal area
I[lon$_0$/]*width*	Sinusoidal equal area
J[lon$_0$/]*width*	Miller cylindrical
Kf[lon$_0$/]*width*	Eckert IV equal area
Ks[lon$_0$/]*width*	Eckert VI equal area
Llon$_0$/lat$_0$/lat$_1$/lat$_2$/*width*	Lambert conic conformal
M[lon$_0$[/lat$_0$]/]*width*	Mercator cylindrical
N[lon$_0$/]*width*	Robinson
Oalon$_0$/lat$_0$/*azim*/*width*[**+v**]	Oblique Mercator, 1: origin and azim

PyGMT Projection Argument	Projection Name
Oblon$_0$/lat$_0$/lon$_1$/lat$_1$/*width*[**+v**]	Oblique Mercator, 2: two points
Oclon$_0$/lat$_0$/lon$_p$/lat$_p$/*width*[**+v**]	Oblique Mercator, 3: origin and pole
P*width*[**+a**][**+f**[**e**|**p**|*radius*]][**+r***offset*][**+t***origin*][**+z**[**p**|*radius*]]	Polar [azimuthal] (θ, r) (or cylindrical)
Poly[lon$_0$[/lat$_0$]/]*width*	Polyconic
Q[lon$_0$[/lat$_0$/]]*width*	Equidistant cylindrical
R[lon$_0$/]*width*	Winkel Tripel
Slon$_0$/lat$_0$[/*horizon*]/*width*	General stereographic
T[lon$_0$[/lat$_0$]/]*width*	Transverse Mercator
U*zone*/*width*	Universal Transverse Mercator (UTM)
V[lon$_0$/]*width*	Van der Grinten
W[lon$_0$/]*width*	Mollweide
X*width*[**l**|**p***exp*|**T**|**t**][/*height*[**l**|**p***exp*|**T**|**t**]][**d**]	Linear, logarithmic, power, and time
Ylon$_0$/lat$_0$/*width*	Cylindrical equal area
X*width*[**l**|**p***exp*|**T**|**t**][/*height*[**l**|**p***exp*|**T**|**t**]][**d**]	Linear, logarithmic, power, and time
Ylon$_0$/lat$_0$/*width*	Cylindrical equal area
X*width*[**l**|**p***exp*|**T**|**t**][/*height*[**l**|**p***exp*|**T**|**t**]][**d**]	Linear, logarithmic, power, and time
Ylon$_0$/lat$_0$/*width*	Cylindrical equal area

https://www.pygmt.org/latest/projections/index.html