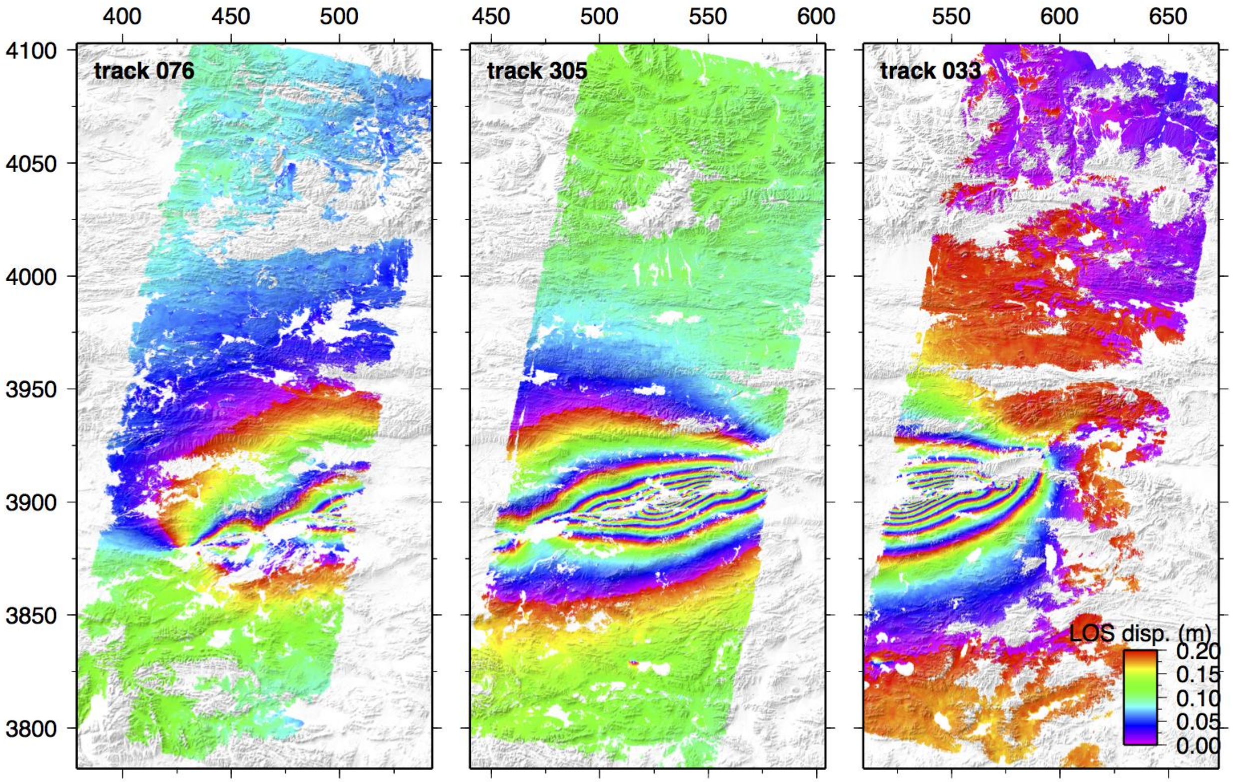
GEO 242: Numerical methods and modeling in the geosciences



Raster graphics and other ‘advanced’ GMT plots

The project

Per the syllabus, the largest contribution of the class grade (45%) is a project.

The final project should make use of all aspects used in the class:

- Analysis and modeling of a data set
- Figures plotted in GMT
- Report written in LaTeX
- Codes and scripts uploaded to GitHub

The subject can be anything geoscientific, and hopefully something that is useful for your research.

Please let me know your project ideas by the end of next week.

Outline

Binary (raster) image formats

Using grdfiles – grdimage, grdgradient

Supplemental packages

Binary file formats

The standard formats for binary data are:

8 bit (1 byte) integer – a.k.a. 'short integer'. Signed (between -128 and 127) or unsigned (between 0 and 255). Usually used for satellite image bands.

16 bit (2 byte) integer – signed (between -32768 and 32767) or unsigned (between 0 and 65535). Usually used for topography.

4 byte real (floating point) – a.k.a. 'single precision'. 1 bit for sign, 8 bits for exponent, 23 bits for fraction.

8 byte real (floating point) – a.k.a. 'double precision'. 1 bit for sign, 11 bits for exponent, 52 bits for fraction

grdfiles in GMT

grdfiles are the 'native' format for raster data in GMT

(Raster data = gridded numbers, e.g. digital topography)

A grdfile typically consists of an ASCII header and then a grid of single-precision floating point numbers (4 bytes per grid point/pixel)

They are a ‘netCDF’-compatible file format, a format that is commonly used for remote sensing data

Since GMT 6, GMT's 'grd' functions can be applied to any GDAL-compatible raster file (and write out multiple formats also)

Operations with grdfiles

grdinfo – read grdfile header information

grdimage – plot a grdfile using a color palette

grdgradient – calculate gradients in a grdfile

grdcontour – calculate contours from a grdfile

grdcut – generate a subset of a grdfile

grdpaste – merge grdfiles with a common edge

grdmath – perform mathematical operations on grdfiles

grd2cpt – generate a color palette file from a grdfile

grdinfo

read grdfile header information

```
gmt grdinfo <grdfile>
```

displays the information in the grdfile headers

grdimage

plot gridded image data

```
gmt grdimage <grdfile> -J<proj> -R -C<col_pal_file>
  -O -K >> <outfile>
```

plots the data in the grdfile using the 'col_pal_file' palette

grd2cpt

generate a color palette from grdfile extents

```
gmt grd2cpt <grdfile> -C<palette> > <col_pal_file>
```

creates a color palette file based on the maximum and minimun values in a grdfile

`-C` has the same function as in `makecpt` – selecting the color scheme that the new '.cpt' file will be based upn

grdgradient

calculate gradients for a grdfile

```
gmt grdgradient <grdfile> -G<outgrd> -A<azimuth>
  -N<normalization>
```

calculates gradients of the input grdfile in the specified azimuth direction (degrees clockwise from north)

The gradients can be normalized in various ways (see the man pages for details). A good option to try is `-Ne0.6` (exponential, amplitude 0.6).

grdimage with gradients

```
gmt grdimage <grdfile> -J<proj> -R
  -C<col_pal_file> -I<grad_grdfile> -O -K >>
  <outfile>
```

plots the data in the grdfile using the 'col_pal_file' palette, shaded using the gradient file ('grad_grdfile').

This can be used to plot shaded relief (hillshade) images. The overlaid image need not be topography.

grdcontour

plot contours for grdfile data

```
gmt grdcontour <grdfile> -J<proj> -R -C<contour_info>
  -O -K >> <outfile>
```

'contour_info' here can take three forms:

- a color palette file (color intervals are used as contour intervals)
- a text file with contour intervals in the 1st column and either 'C's (for 'contour') or 'A's (for 'annotate')
- a number which specifies a contour interval

grdcontour

Bonus options:

`-A<value>`

specify annotation interval

`-W+cl`

color the contours (if a color palette file is named under `-C`)

grdmath

conduct mathematical (Reverse Polish calculator) operations on grdfiles

```
gmt grdmath <grdfile1> <grdfile2> ADD = <out_grdfile>
```

adds ‘grdfile1’ and ‘grdfile2’ and saves the output as ‘out_grdfile’

```
gmt grdmath <grdfile1> <grdfile2> MUL = <out_grdfile>
```

multiplies ‘grdfile1’ and ‘grdfile2’ and saves the output as ‘out_grdfile’

grdmath

The arguments for grdmath need not be grdfiles – numbers work too, e.g.

```
gmt grdmath <grdfile> 0 NEQ = <out_grdfile>
```

which creates a non-zero mask (i.e. a file with zeros where ‘grdfile’ is zero and ones where ‘grdfile’ is non-zero)

grdcut

takes a subset of a grdfile

```
gmt grdcut <grdfile> -R<region> -G<out_grdfile>
```

this cuts a grdfile down to the size of the region specified

grdpaste

assembles grdfiles with common edges

```
gmt grdpaste <grdfile1> <grdfile2> -G<out_grdfile>
```

this merges two grdfiles which have a common edge and the same number of rows and columns

xyz2grd

converts ASCII or gridded binary data to a grdfile

```
gmt xyz2grd <ASCIIfile> -R<region> -I<spacing>
  -G<out_grdfile>
```

takes ASCII x,y,z data ('ASCIIfile') and makes a grdfile containing the 'z' values, with a grid spacing of 'spacing'.

xyz2grd

```
gmt xyz2grd <binaryfile> -R<region> -I<spacing>
  -Z<binary_options> -G<out_gridfile>
```

Converts gridded binary data to grdfile format. The `-Z` option is critical to getting this to work:

`-ZTLh` – 16 bit integer data with the first point in the **T**op **L**eft corner

`-ZTLf` – 4 byte real data with the first point in the **T**op **L**eft corner

If the number of datapoints implied by `-R` and `-I` does not match the size of the file, this will fail with an error

This command is mostly redundant these days, due to GDAL support

psmeca

plots earthquake focal mechanisms

```
gmt psmeca <infile> -J<proj> -R           –
  Sa<scale> -G<color> -O -K >> <outfile>
```

format of infile is:

x y depth strike dip rake magnitude newx newy name

Other input formats exist (e.g. –Sc for Global CMT format) – check out the manpages

psvelo

plots vectors (e.g. velocities)

```
gmt psvelo <infile> -J<proj> -R –Se<scale> -O -K
  >> <outfile>
```

format of infile is:

x y vel_x vel_y sig_x sig_y cor_xy name

This will plot error ellipses for each vector (to suppress, provide zeros for the sigmas)

It is also possible to plot strain crosses and other quantities (e.g. anisotropy) with other –S options

psvelo

plots vectors (e.g. velocities)

```
gmt psvelo <infile> -J<proj> -R –Sv<size> -O -K >>
  <outfile>
```

The `–Sv` option allows the use of the old (and by some, beloved) GMT4 vector plotting style, where 'size' is defined by three numbers: *tailwidth/headlength/halfheadwidth* and a `+n<norm>` option allows for scaling of the vectors by a given normalization

Today’s exercise

Make a map of a tectonically active area in the western US that illustrates the active tectonics. Your map should be at least 2° × 2°, outside of southern California, and include color-coded and shaded-relief topography

Other elements to include:

- network seismicity
- earthquake focal mechanisms
- NOTA GPS velocity vectors
- place names and other geographical markers
- traces of Holocene faults