Dr. Saji P. K.
Assistant Professor, CUSAT.

Sessions

Session - 1

Introduction

What is GMT?

- GMT stands for Generic Mapping Tools.
- GMT is a collection of commands that can be executed at the Unix/Linux terminal to produce graphic outputs.
- Often, GMT commands are written to a text file (with extension .gmt or .sh) and execute the file as a shell script.
- GMT need UNIX/Linux Environment.
- Graphic output from GMT is in the postscript format.

A sample GMT file


```
    makecpt -T0/150/10 > color.cpt
    awk '$$1 == 7 && $$4 > 0 {print $$3, $$2, $$4}' mld.txt > junk
    psbasemap -P -JM8 -R40/110/-40/30 -B20 -X5 -Y15 -K > fig.ps
#   pscontour junk  -R -J -Ccolor.cpt -I -O >> fig.ps
    psxy junk  -R -J -Ccolor.cpt -Sc0.1 -O -K >> fig.ps
#   pscoast -J -R -B -W -O >> fig.ps

    makecpt -T0/12/1 > color.cpt
    awk '$$1 == 7 && $$4 > -999 {print $$3, $$2, $$4}' ../../Wind/lev1/
wind.txt > junk
    psbasemap -P -JM -R40/110/-40/30 -B20 -Y-12 -O -K >> fig.ps
    psxy junk  -R -J -Ccolor.cpt -Sc0.03 -O >> fig.ps


3:
    makecpt -T20/30/1 > color.cpt
    awk '$$1 == 7 && $$2 == 0 && $$5 > -99 {print $$4, $$3, $$5}' ../../
WOA2005/lev1/temp.txt > junk
    psbasemap -P -JM8 -R40/110/-40/30 -B20 -X5 -Y18 -K > fig.ps
    psxy junk  -R -J -Ccolor.cpt -Sc0.3 -O -K >> fig.ps

    makecpt -T0/10/1 > color.cpt
    awk '$$1 == 7 && $$4 > -999 {print $$3, $$2, $$4}' wind.txt > junk
    psbasemap -P -JM -R40/110/-40/30 -B20 -Y-9 -O -K >> fig.ps
    psxy junk  -R -J -Ccolor.cpt -Sc0.03 -O >> fig.ps

4:
# time series at a point
    awk '$$2 == -0.5 && $$3 == 95.5 {print $$1, $$4}' mld.txt > junk
    psxy junk -R0/12/0/150 -P -JX12/8 -B1/50WS -W,red -X5 -Y10 -K > fig.ps
```

History

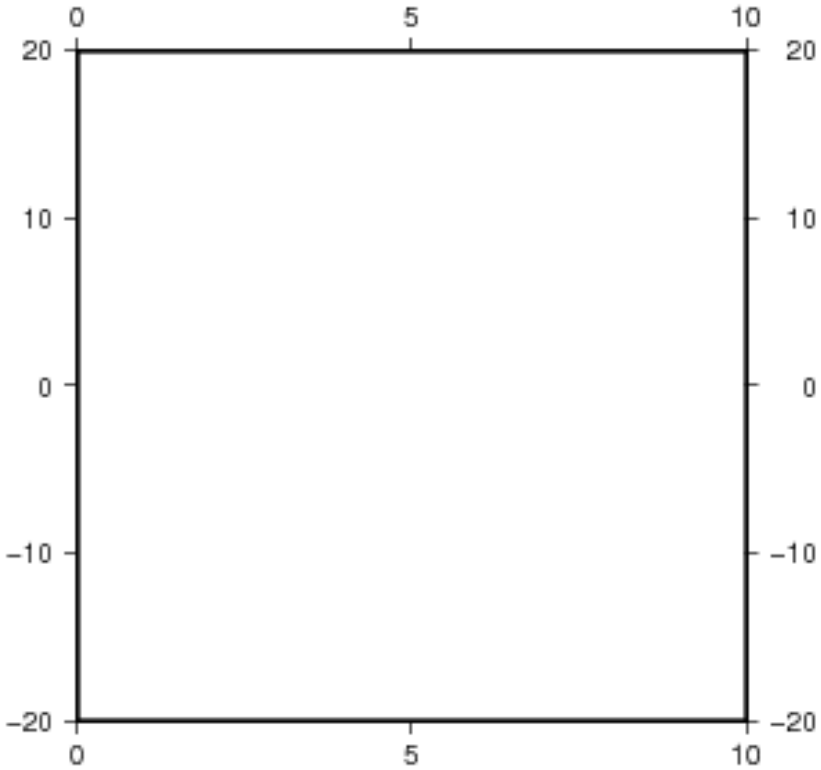
- Initiated in 1987 at Lamont-Doherty Earth Observatory, Columbia University by graduate students Paul Wessel and Walter H. F. Smith.
- Grew out of frustration with the existing geophysics software.
- Version 1.0 in 1988.
- Version 2.0 was released 1991, and quickly spread worldwide.
- A major upgrade (GMT 4.0) n 2004.
- GMT 5.0 in 2013.
- GMT 6.0 is under development.

Paul Wessel

Let us Begin with An Exercise

Method - 1 Run at Linux terminal.

```
gmt psbasemap -P -JX10 -R0/10/-20/20 -B5/10 > fig.ps
```



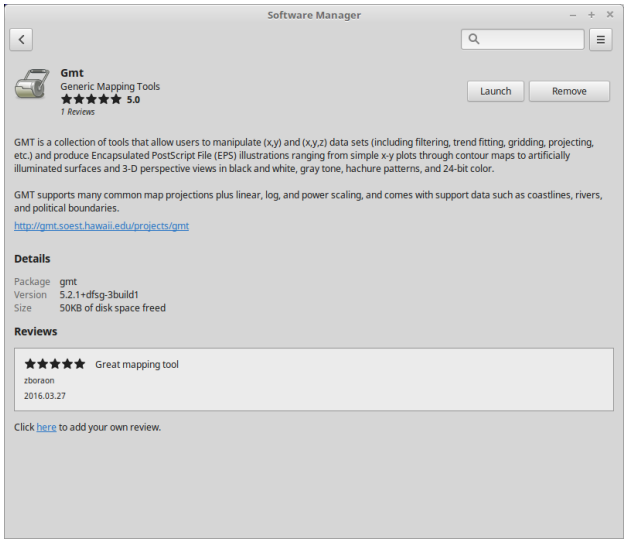
Method - 2 Run as a file

(1) Save the command to a file, say, 1.gmt

(2) Execute the command 'sh 1.gmt' at the terminal

How to Install GMT

- Using software manager. Search for 'gmt' and install.



- Installing from source. For this, download the files from GMT website and follow instructions.
- Using command (in Debian based systems)

```
sudo apt-get install gmt gmt-dcw gmt-gshhg-full
```

Design Philosophy

- GMT consists of modules (programs) focussing on specific tasks.
- Users have to combine several GMT modules to get the desired output.
- Integrates with Unix tools
- Can run modules in command-line or as shell script files.

Why use GMT?

- Free of cost.
- Open source code.
- Works in all OS platforms
- Support over 30 map projections with high resolution coastline.
- Works with ascii, netcdf file formats.
- Produces quality postscript outputs that can be converted to other formats
- Developers are also geophysical scientists who uses GMT.

What GMT can do?

- Data processing and manipulation
  Filter time series, filter 2D data, trend fitting, gridding xyz data, resampling, grid masking, optimal triangulations, spectral estimation, data reformatting, geospatial operations, subset extraction.

- Postscript plot generation
  Points, lines, symbols, polygons Text, labels, legends Histograms, geographical map, Contour maps, Color images, Vector fields

GMT suppliments

- Earthquake focal mechanisms
- Sandwell/Smith IMG files
- Plate tectonics
- MGD77 analysis and plotting
- Crossover analysis
- Spherical gridding

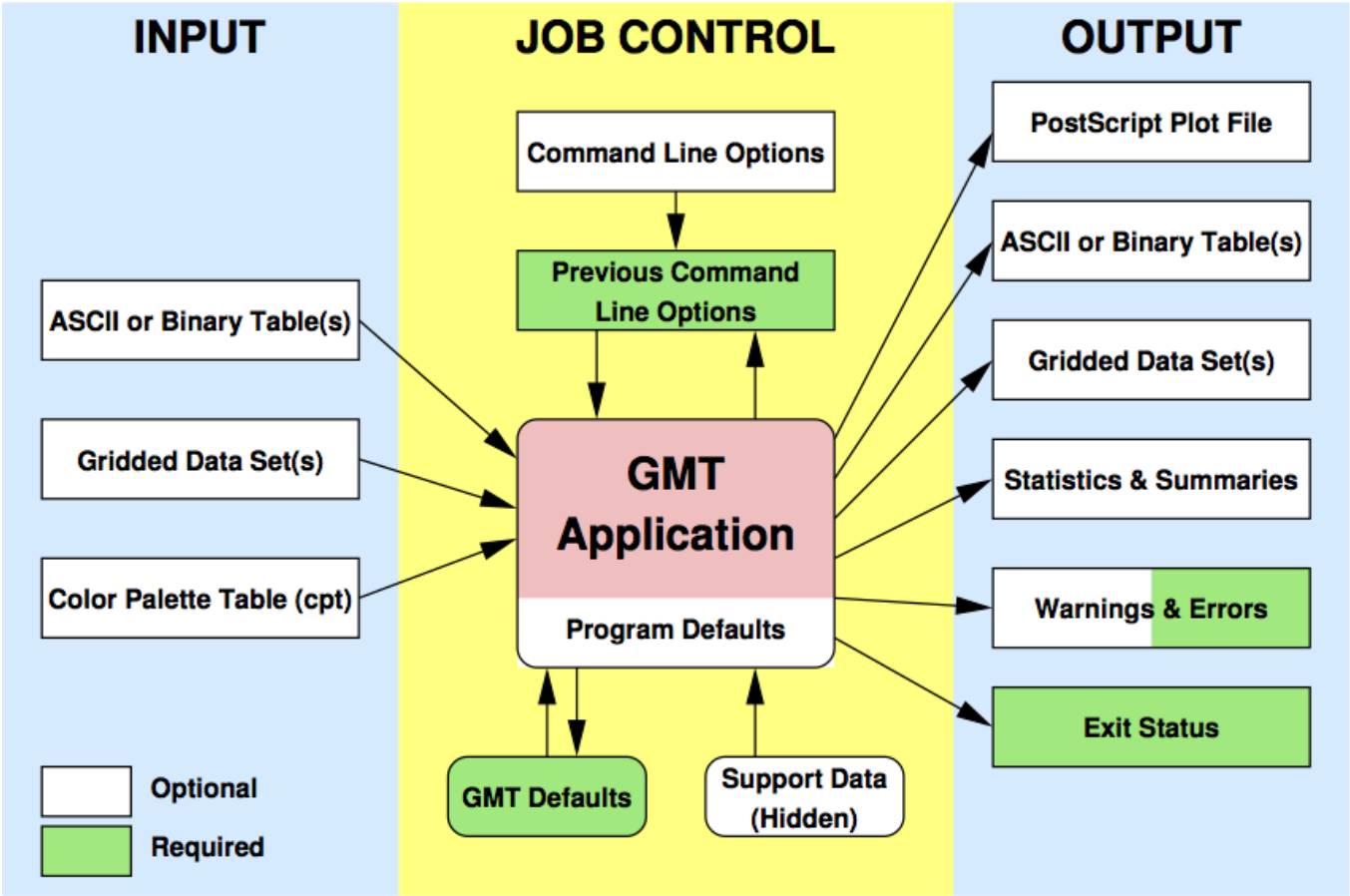
Getting Help

- GMT Website
  - https://www.soest.hawaii.edu/gmt/
  - http://gmt.soest.hawaii.edu/
- Man pages (from terminal or google)

Session - 2

How GMT Works?

Process



GMT Module

Syntax for a GMT module :

gmt *command options redirection outputfile*

where

command Graphic and data processing GMT commands

options Various options which start with a negative sign followed by a capital letter and values

redirection Either single arrow (>) or double arrow (>>). Single arrow will delete all the present contents of the output file and write new contents to it. Double arrow will keep its present content and append new content to it.

outputfile This can be either a postscript graphic file or a text file or a binary file depends upon the gmt command used.

GMT Commands

GMT has about 80 programs/commands to produce graphic and data outputs.

FILTERING OF 1-D AND 2-D DATA:

- blockmean L2 (x,y,z) data filter/decimator
- blockmedian L1 (x,y,z) data filter/decimator
- blockmode Mode-estimating (x,y,z) data filter/decimator
- filter1d Filter 1-D data (time series)
- grdfilter Filter 2-D data in space domain

GMT Commands ...

PLOTTING OF 1-D and 2-D DATA:

- grdcontour Contouring of 2-D gridded data
- grdimage Produce images from 2-D gridded datar
- grdvector Plot vector fields from 2-D gridded data
- grdview 3-D perspective imaging of 2-D gridded data
- psbasemap Create a basemap frame
- psclip Use polygon files as clipping paths
- pscoast Plot coastlines, filled continents, rivers, and political borders
- pscontour Direct contouring or imaging of xyz-data by triangulation
- pshistogram Plot a histogram
- psimage Plot Sun rasterfiles on a map
- pslegend Plot legend on a map
- psmask Create overlay to mask specified regions of a map
- psrose Plot sector or rose diagrams
- psscale Plot grayscale or colorscale
- pstext Plot textstrings
- pswiggle Draw anomalies along track
- psxy Plot symbols, polygons, and lines in 2-D
- psxyz Plot symbols, polygons, and lines in 3-D

GMT Commands ...

GRIDDING OF (X,Y,Z) DATA:

- greenspline Gridding using Green's function splines
- nearneighbor Nearest-neighbor gridding scheme
- surface Continuous curvature gridding algorithm
- triangulate Perform optimal Delauney triangulation on xyz data

SAMPLING OF 1-D AND 2-D DATA:

- grdsample Resample a 2-D gridded data onto new grid
- grdtrack Sampling of 2-D data along 1-D track
- sample1d Resampling of 1-D data

PROJECTION AND MAP-TRANSFORMATION:

- grdproject Project gridded data onto new coordinate system
- mapproject Transformation of coordinate systems
- project Project data onto lines/great circles

GMT Commands ...

INFORMATION:

- gmtcolors Information on how to specify colors in GMT
- gmtdefaults List the current default settings
- gmtset Edit parameters in the .gmtdefaults file
- grdinfo Get information about grd files
- minmax Report extreme values in table datafiles

CONVERT OR EXTRACT SUBSETS OF DATA:

- gmt2rgb Convert Sun raster or grdfile to red, green, blue component grids
- gmtconvert Convert table data from one format to another
- gmtmath Reverse Polish calculator for table data
- gmtselect Select table subsets based on multiple spatial criteria
- grd2xyz Convert 2-D gridded data to table
- grdcut Cut a sub-region from a grd file
- grdpaste Paste together grdfiles along common edge
- grdreformat Convert from one grdformat to another
- splitxyz Split xyz files into several segments
- xyz2grd Convert table to 2-D grd file

MISCELLANEOUS:

- makecpt Create GMT color palette tables
- spectrum1d Compute spectral estimates from time-series
- triangulate Perform optimal Delauney triangulation on xyz data

GMT Commands ...

DETERMINE TRENDS IN 1-D AND 2-D DATA:

- fitcircle Finds best-fitting great or small circles
- grdtrend Fits polynomial trends to grdfiles (z = f(x,y))
- trend1d Fits polynomial or Fourier trends to y = f(x) series
- trend2d Fits polynomial trends to z = f(x,y) series

OTHER OPERATIONS ON 2-D GRIDS:

- grd2cpt Make color palette table from grdfile
- grdblend Blend several gridded data sets into one
- grdclip Limit the z-range in gridded data sets
- grdedit Modify grd header information
- grdfft Operate on grdfiles in frequency domain
- grdgradient Compute directional gradient from grdfiles
- grdhisteq Histogram equalization for grdfiles
- grdlandmask Creates mask grdfile from coastline database
- grdmask Set nodes outside a clip path to a constant
- grdmath Reverse Polish calculator for grdfiles
- grdvolume Calculating volume under a surface within a contour

Standardized Options

- -J Map projection
- -R Extent of the map/plot region
- -B Defines tickmarks, annotations, and labels for basemaps and axes
- -P Selects Portrait plot orientation [Default is landscape]
- -X Sets the x-coordinate for the plot origin on the page
- -Y Sets the y-coordinate for the plot origin on the page
- -K Allows more plot code to be appended to this plot later
- -O Allows this plot code to be appended to an existing plot
- -V Selects verbose operation; reporting on progress

Projections

GMT offers 31 map projections that are classified into

- Azimuthal Projection
- Conical Projection
- Cylindrical Projection
- Miscellaneous Projections
- Non-geographic Projections

WITH GMT PROJECTION CODES	
-J (upper case for *width*, lower case for *scale*) Map projection	
-JA$\text{lon}_0/\text{lat}_0$[/*horizon*]/*width*	Lambert azimuthal equal area ...
-JB$\text{lon}_0/\text{lat}_0/\text{lat}_1/\text{lat}_2$*width*	Albers conic equal area ...
-JC$\text{lon}_0/\text{lat}_0$*width*	Cassini cylindrical ...
-JCyl_stere/[lon_0[/lat_0/]]*width*	Cylindrical stereographic ...
-JD$\text{lon}_0/\text{lat}_0/\text{lat}_1/\text{lat}_2$*width*	Equidistant conic ...
-JE$\text{lon}_0/\text{lat}_0$[/*horizon*]/*width*	Azimuthal equidistant ...
-JF$\text{lon}_0/\text{lat}_0$[/*horizon*]/*width*	Azimuthal gnomonic ...
-JG$\text{lon}_0/\text{lat}_0$[/*horizon*]/*width*	Azimuthal orthographic ...
-JG$\text{lon}_0/\text{lat}_0$*alt/azim/tilt/twist/W/H/width*	General perspective ...
-JH[lon_0/]*width*	Hammer equal area ...
-JI[lon_0/]*width*	Sinusoidal equal area ...
-JJ[lon_0/]*width*	Miller cylindrical ...
-JKf[lon_0/]*width*	Eckert IV equal area ...
-JKs[lon_0/]*width*	Eckert VI equal area ...
-JL$\text{lon}_0/\text{lat}_0/\text{lat}_1/\text{lat}_2$*width*	Lambert conic conformal ...
-JM[lon_0[/lat_0/]]*width*	Mercator cylindrical ...
-JN[lon_0/]*width*	Robinson ...
-JOa$\text{lon}_0/\text{lat}_0$*azim/width*	Oblique Mercator, 1: origin and azim ...
-JOb$\text{lon}_0/\text{lat}_0/\text{lon}_1/\text{lat}_1$*width*	Oblique Mercator, 2: two points ...
-JOc$\text{lon}_0/\text{lat}_0/\text{lon}_p/\text{lat}_p$*width*	Oblique Mercator, 3: origin and pole ...
-JP[**a**]*width*[/*origin*]	Polar [azimuthal] (θ, r) (or cylindrical)
-JPoly[lon_0[/lat_0/]]*width*	(American) polyconic ...
-JQ[lon_0[/lat_0/]]*width*	Equidistant cylindrical ...
-JR[lon_0/]*width*	Winkel Tripel ...
-JS$\text{lon}_0/\text{lat}_0$[/*horizon*]/*width*	General stereographic ...

Session - 3

Some Unix/Linux Basics

Redirection

To save the results of a command to an output file.

- Single arrow (>) to start new output.
  Syntax: gmt *module inputfile* > *outputfile*
  For example,
  gmt pscoast -P -JM10 -R30/100/-30/30 -B10 -W1 -G200 > fig.ps
- Double arrow (>>) to update/append the output
  Syntax: gmt module inputfile >> outputfile
  For example,
  gmt psxy sample.txt -P -JX10 -R0/100/0/30 -B10 -W1 -O >> fig.ps

Piping

Used when the output from one module in taken as the input to another module.

```
Someprogram | gmt module1 | gmt module1 > OutputFile
```

"wild cards"

Code	Meaning
*	Matches everything
?	Matches any single character

awk

- awk is one of the most prominent text/data processing utility on GNU/Linux.
- It is very powerful and uses simple programming language.
- It is used to extract data from files.
- Writing codes in other languages (C, fortran) is time-consuming and inconvenient.
- awk can perform this in one or two lines of codes.

Example:

Create a file that store the output of the 'ls -l' command

```
ls -l * > x.txt
```

Print first line

```
awk ' {print $1}' x.txt
```

Multiply 7th column with 2

```
awk ' {print $7*2}' x.txt
```

Printing 7th column for lines from 10th onwards

```
awk ' NR > 10 {print $7 }' x.txt
```

Another example

```
awk ' NR > 10 && $3 > 100 { print $7 }' x.txt
```

echo command

- The echo program displays text.
- It is a handy tool to create customized output to terminal or files.

Example

```
echo 'Hello'
echo '4'
echo '4+3'
echo 'Hello' > hello.txt
```

Session - 4

Basemap Plot

- Basemap is to set the various aspects of a plot such as projection, axis ranges, tick mark labels and intervals, position etc.
- The GMT command 'psbasemap' is used to generate basemaps.
- Syntax

  gmt psbasemap options > psfile

Options for psbasemap

Mandatory Options

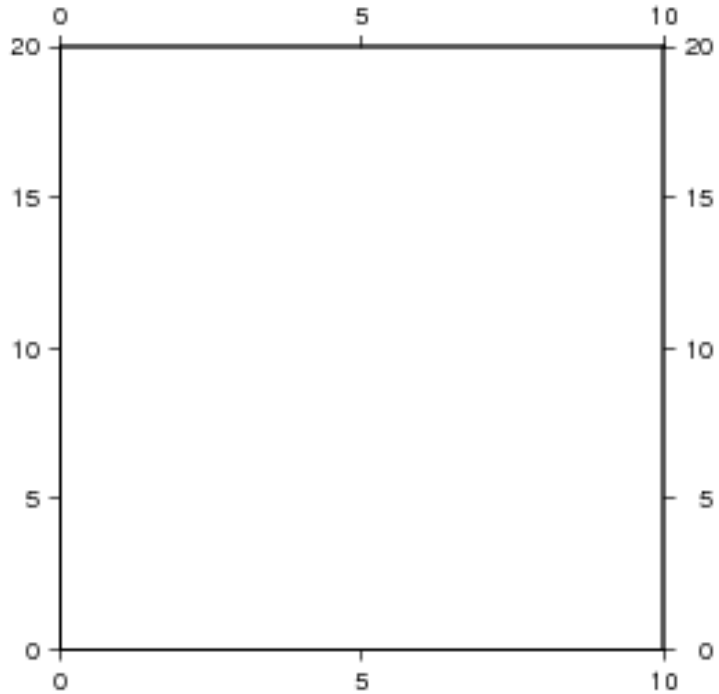
- -J Projection (geographic or non-geographic)
- -R Region or range of X and Y axes. Should be given as -Rxmin/xmax/ymin/ymax
- -B Tick mark details (labels, intervals, grids, axis title)

Additional options

- -G Sets background color
- -X X position
- -Y Y position

Exercise (Linear Projection)

gmt psbasemap -P -JX10 -R0/10/0/20 -Ba5 > fig.ps



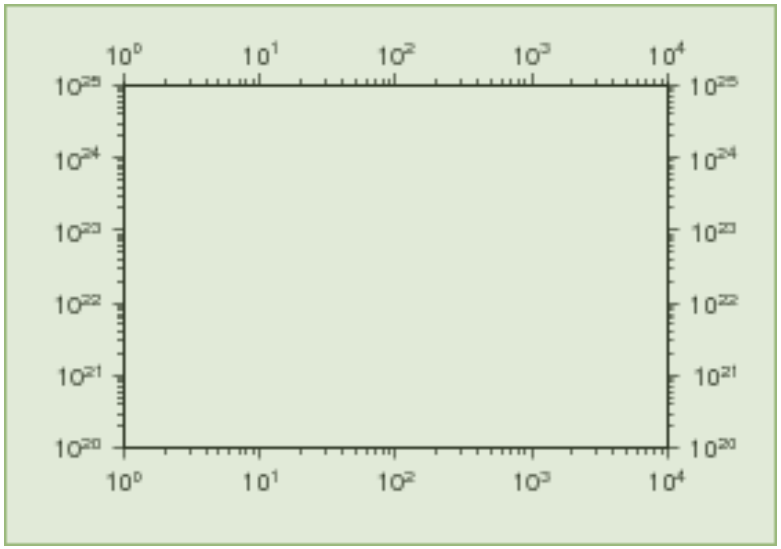
Modify -J (Axis length)

- -JX10/5 (Change different axis lengths)
- -JX10/-5 (Reverse Y axis)
- -Jx0.5 (Scale)

Modify -J (Logarithmic Projection)

Append 'l' to -JX

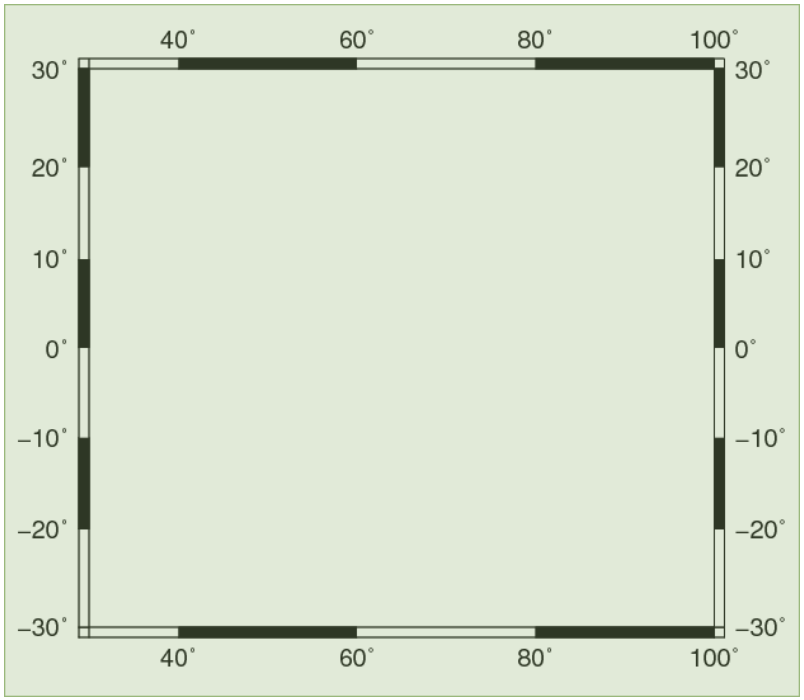
gmt psbasemap -R1/10000/1e20/1e25 -P -JX9l/6l -Ba1pf3/a1pf3 > fig.ps



Modify -J (Metcator Projection)

Mercator Projection (-Jm or -JM)

gmt psbasemap -P -Jm0.15 -R30/100/-30/30 -B20/10 > fig.ps



Here the -R option should have the geographic limits.

Modify -B (Axis title, annotation etc.)

- -Ba5/a10 [Different label annotations]
- -Ba5f1/a10f5 [Adding minor ticks]
- -Ba5f1g1/a10f5g5 [Adding grids]
- -Ba5f1g1:"X axis":/a10f5g5:"Y axis": [Axis labels]
- -Ba5f1g1:"X axis":/a10f5g5:"Y axis":WS [Select west and south axis]

Add Basemap background color with -G

- -Gblue Fill with blue color
- -G255/0/0 Red colour in RGB format

Change figure Position with -X, -Y

Positions are relative to the previous figure.

Default position is at the bottom left corner of the page.

- -X5 (shift 5cm to right)
- -Y10 (shift 10cm up)
- -X-8 (shift 8cm to the left)

Using variables

Variables can be defined in the beginning of a gmt file and can be referred within the script. While referring, the variable must be preceded with '$'. For example,

```
psfile=fig.ps
region=30/100/-30/30
gmt psbasemap -P -Jm0.15 -R$region -B20/10 > $psfile
```

More Plots with -K, -O Options

The -K and -O options control the generation of PostScript code for multiple overlay plots. All PostScript files must have a header (for initializations), a body (drawing the figure), and a trailer (printing it out).

Thus, when overlaying several GMT plots we must make sure that the first plot call ommits the trailer, that all intermediate calls omit both header and trailer, and that the final overlay omits the header.

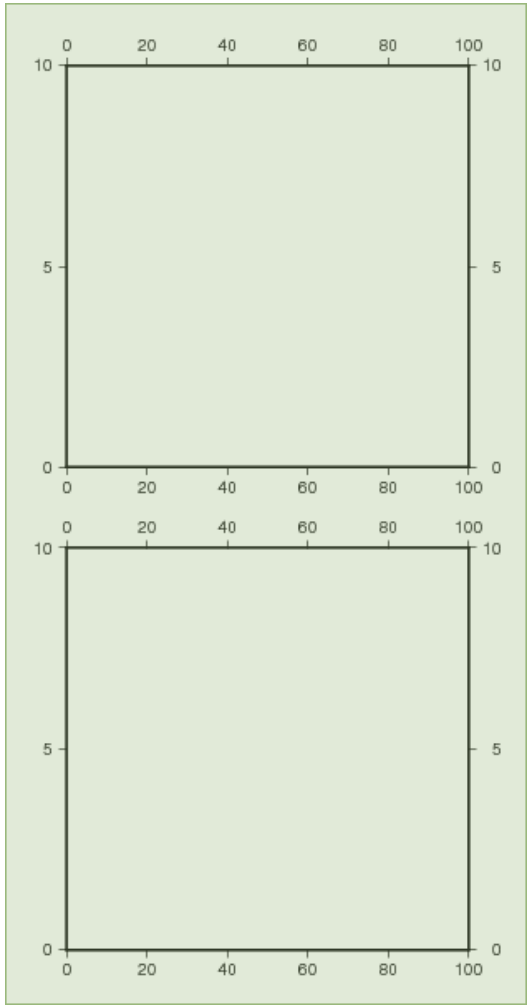
-K omits the trailer which implies that more PostScript code will be appended later.

-O selects Overlay plot mode and ommits the header information

Example -1 : Two Plots

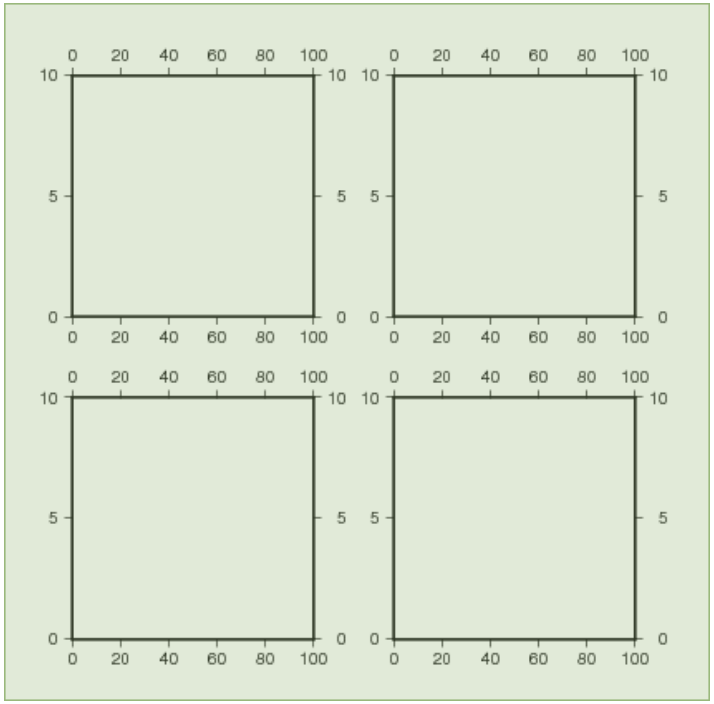
gmt psbasemap -P -JX10 -R0/100/0/10 -B20/5 -X5 -Y5 -K > fig.ps

gmt psbasemap -P -JX10 -R0/100/0/10 -B20/5 -X0 -Y12 -O >> fig.ps



Example - 2: Four Plots


```
gmt psbasemap -P -JX6 -R0/100/0/10 -B20/5 -X5 -Y5 -K > fig.ps
gmt psbasemap -P -JX -R -B -X8 -O -K >> fig.ps
gmt psbasemap -P -JX -R -B -X-8 -Y8 -O -K >> fig.ps

gmt psbasemap -P -JX -R -B -X8 -O >> fig.ps
```



Session - 5

Geographical Map

pscoast

The pscoast command can plot coastline, river and national border derived from the GSHHG database. It is available at different resolutions such as low, crude, high, and full.

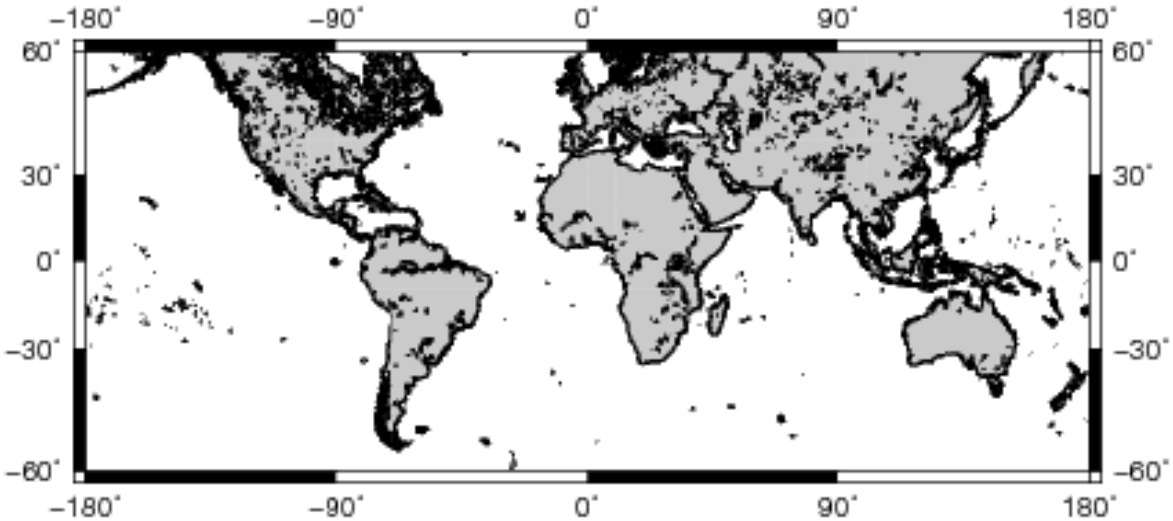
Standardized options: -P -J -R -B
Additional options :

- -D Data resolution (-Dl, -Dc, -Dh, -Df)
- -W Draw coastline and line thickness.
- -G Color land area
- -S Color water area
- -I Draw rivers
- -L Plot map scale
- -A Exclude small features
- -N Draw political borders

Note : One of -W, -G, and -S is mandatory.

Mercator Projection

gmt pscoast -P -JM15 -R180W/180E/-60/60 -B90/30 -Ggrey -W1 > fig.ps



Lambert azimuthal equal-area projection

pscoast -Rg -JA280/30/3.5i -B30g30/15g15 -Dc -A1000 -Gblack -P

Figure 6.5: Hemisphere map using the Lambert azimuthal equal-area projection.

Albers equal area Conic Projection

pscoast -R110/140/20/35 -JB125/20/25/45/5i -B10g5 -Dl -Glightgray -W0.25p -A250 -P

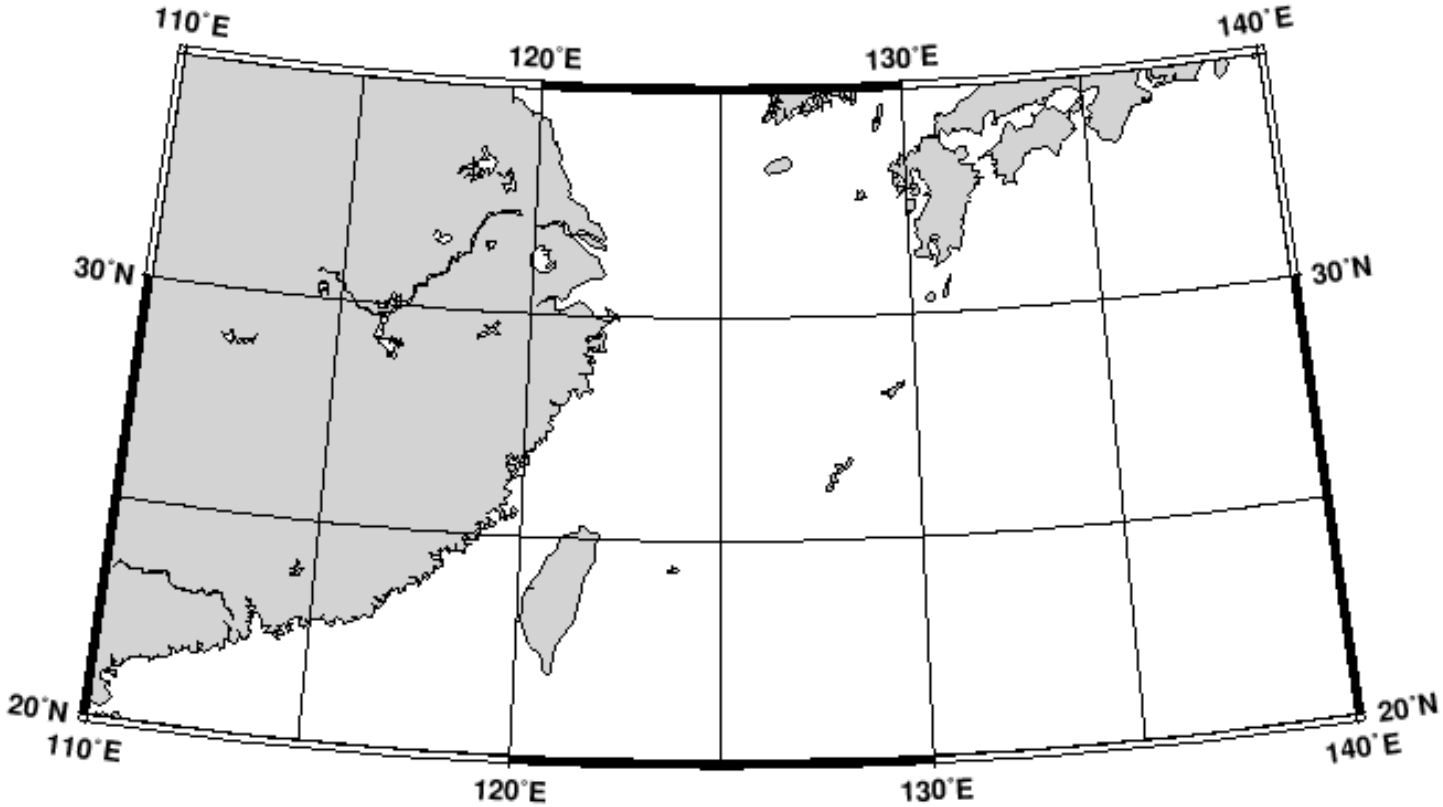


Figure 6.1: Albers equal-area conic map projection

Polar stereographic conformal projection with rectangular borders.

pscoast -R-25/59/70/72r -JS10/90/11c -B30g10/5g5 -Dl -A250 -Glightgray -W.25p -P

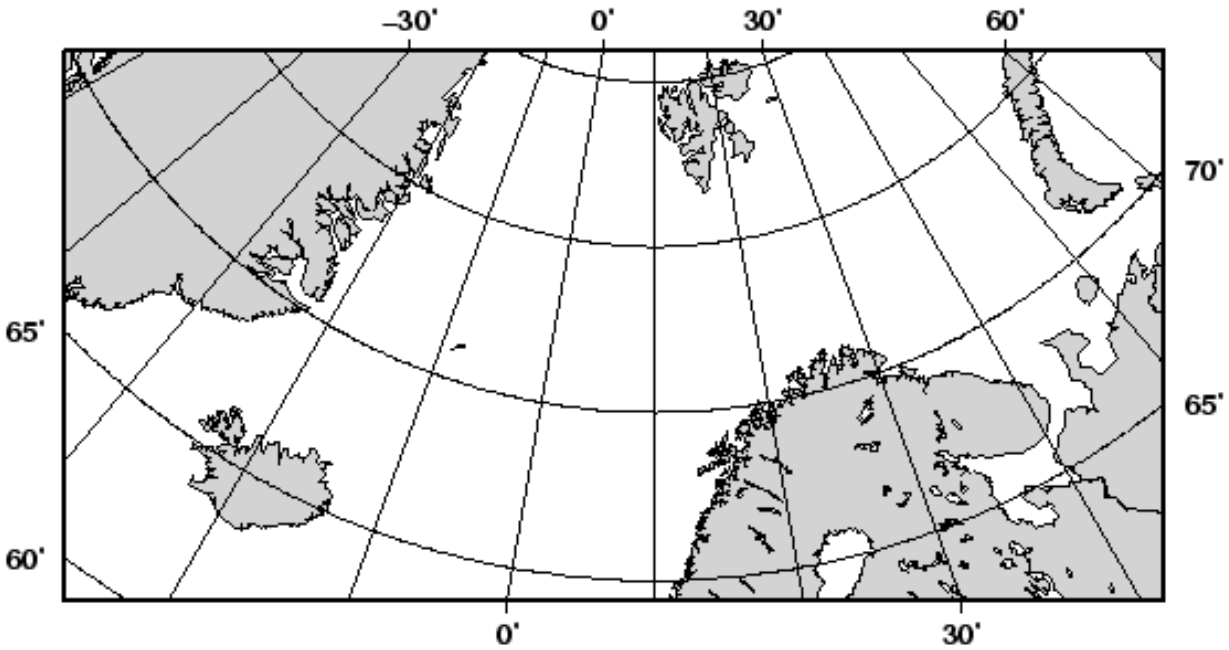
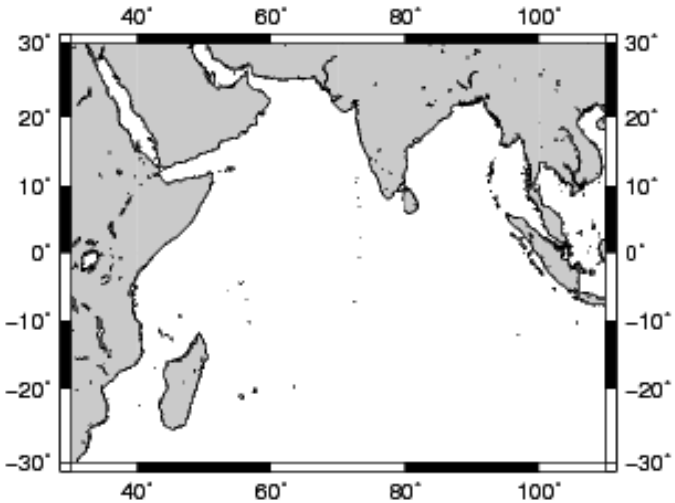


Figure 6.8: Polar stereographic conformal projection with rectangular borders.

psbasemap and pscoast


```
gmt psbasemap -P -JM10 -R30/110/-30/30 -Ba20/10 -X5 -Y10 -K > fig.ps
gmt pscoast -J -R -B -Dl -W -Ggrey -O >> fig.ps
```



Session - 6

Write text with pstext

pstext Options

pstext command is used to write text on maps with variable size, font, and orientation. The options are:

- fileFile name of the input file. The file should contain data in the format "x,y,Text"
- -J Projection
- -R Region
- -F Font parameters
- -N Plot text outside the domain (-R)

-F Option

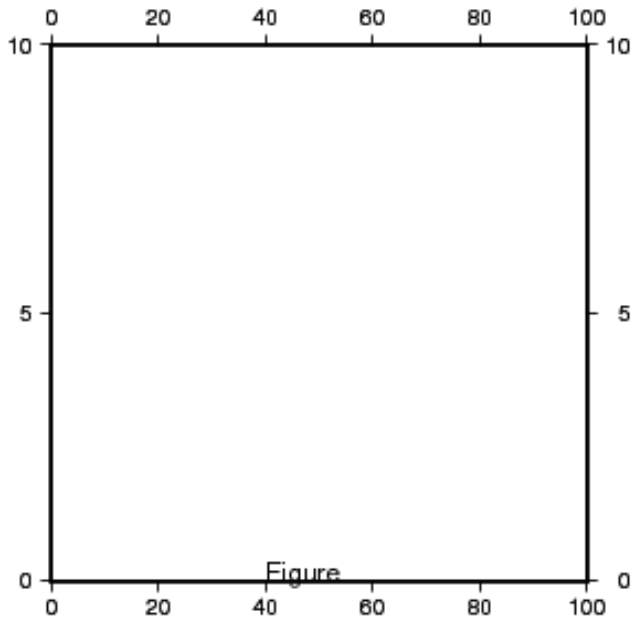
Syntax: -F+a+f+j+t

+a Font angle. Example, +a90
+f Font (size,type,color). Example. +f12p,Helvetica,red
+j Justify (L,C,R,T,M,B). Example, +jLM
+t Text

Example-1

The file sample1.txt contains only one line as "40 0 Figure".

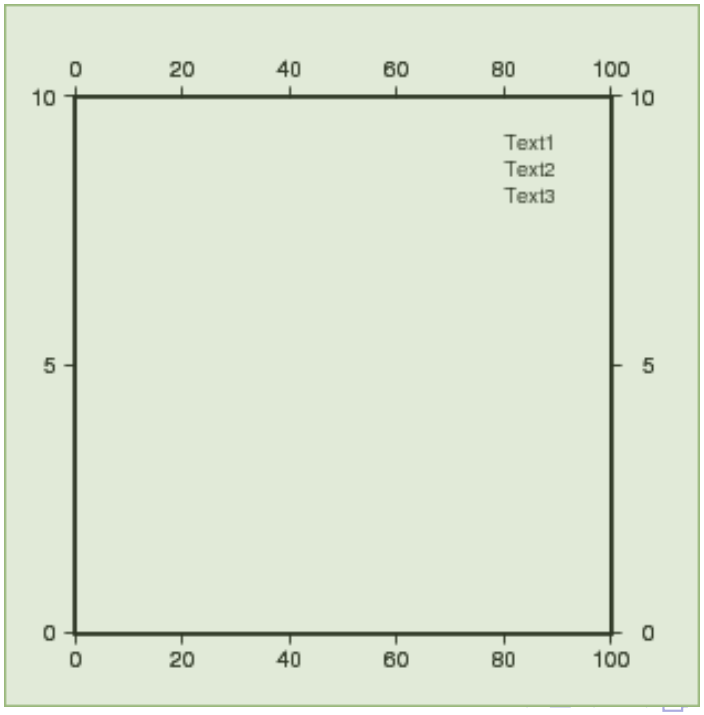
```
gmt psbasemap -P -JX10 -R0/100/0/10 -B20/5 -X5 -Y5 -K > fig.ps
gmt pstext sample1.txt -N -R -J -F+f14p,Helvetica,black+a0+jLB -B -O >>
fig.ps
```



Example-2

The sample2.txt file contains the following data
80 9 Text1
80 8.5 Text2
80 8 Text3

```
gmt psbasemap -P -JX10 -R0/100/0/10 -B20/5 -X5 -Y5 -K > fig.ps
gmt pstext sample2.txt -N -R -J -F+f11p,Helvetica,black+a0+jLB -B -O >>
fig.ps
```



Session - 7

Line Plot

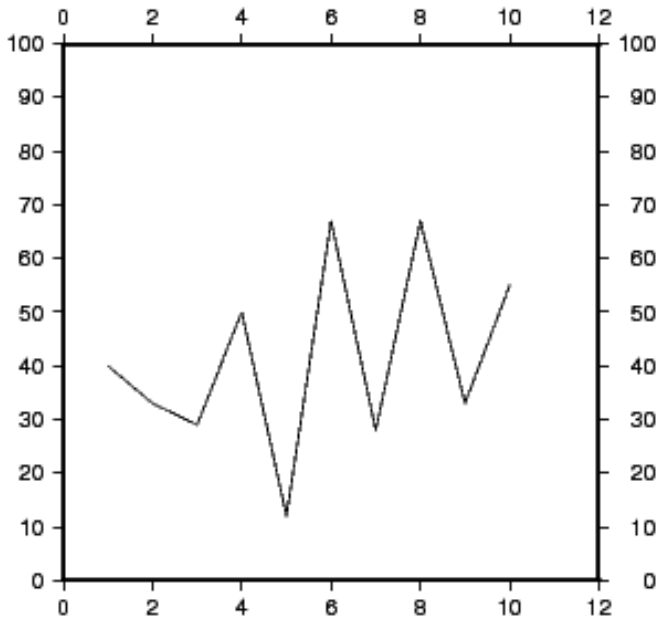
psxy command

- The psxy gmt command is used plot lines, polygons, and symbols on maps
- In addition to the basemap options (**-P -J -R -B**, psxy requires the option **-W** that represent line properties.
- psxy needs an input file as input. It takes the first column data in the file as X series and second as Y series.

Example - 1

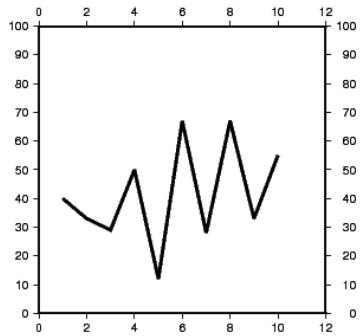
gmt psbasemap -P -JX10 -R0/12/0/100 -B2/10 -X4 -Y10 -K > fig.ps

psxy sample.txt -J -R -B -W -O >> fig.ps

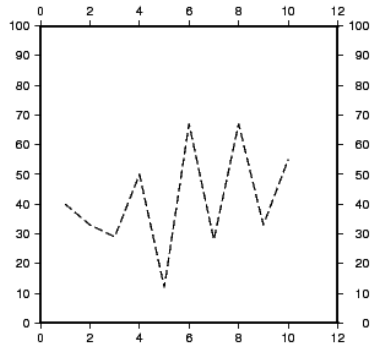


Additional options for -W

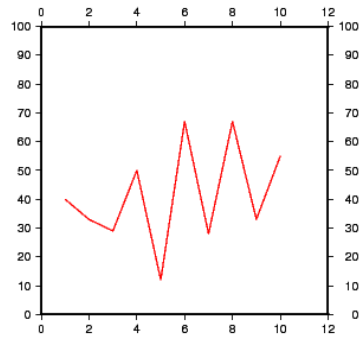
- -W Normal line (with thickness 1)
- -W3 Thicker line
- -W1,'–' Dashed line
- -W1,red Red color line using color name
- -W1/255/0/0 Red color line with R/G/B



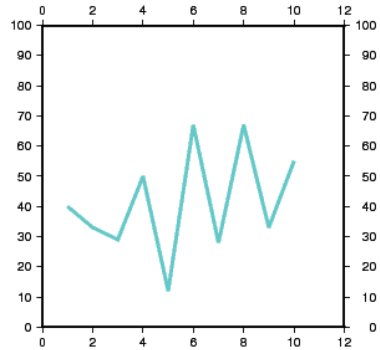
-W3



-W1,'-'



-W,red

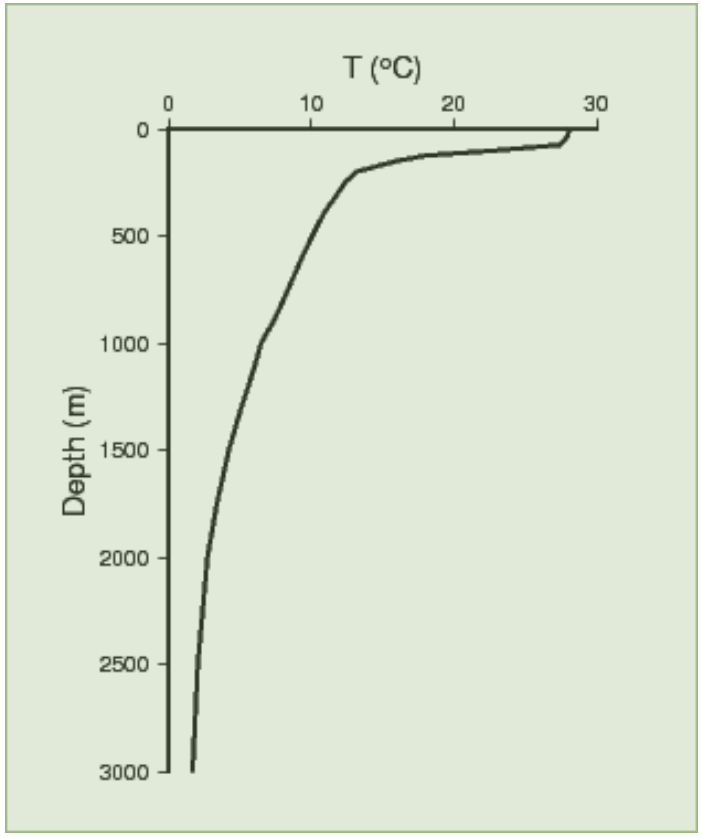


-W3,100/200/200

Example -2: Ocean Temperature profile


```
gmt psbasemap -P -JX8/-12 -R0/30/0/3000 -B10:"T (@+o@+C)":/500:"Depth (m)":WN -X4 -Y10 -K > fig.ps
gmt psxy profile-t.txt -J -R -B -W2 -O >> fig.ps
```

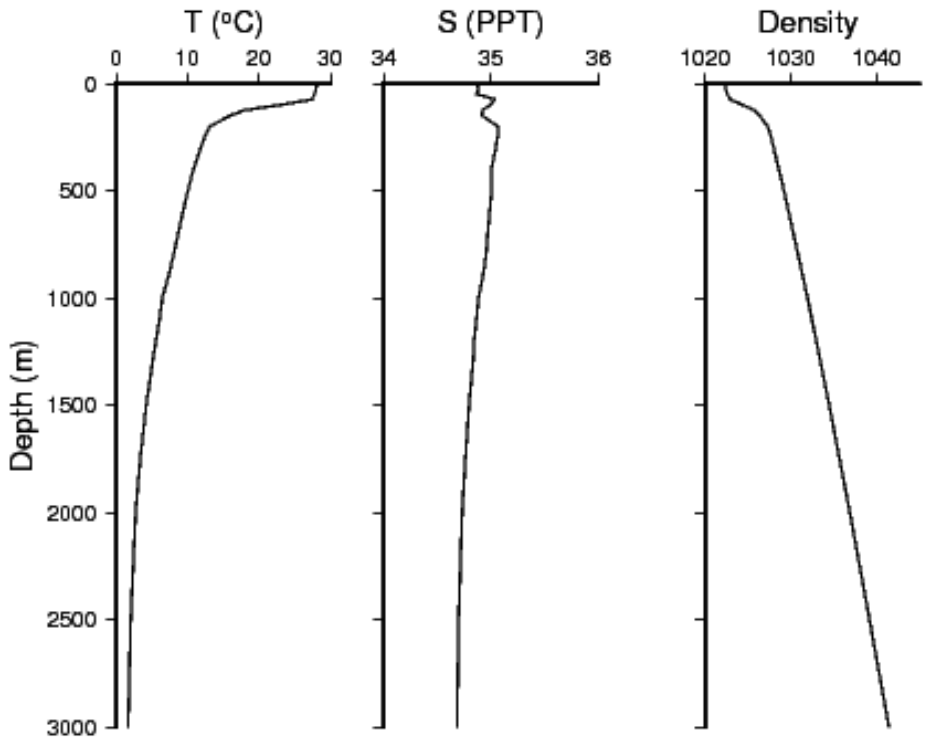
```
28.082001 0.0000000
28.049000 10.000000
28.031000 20.000000
28.002001 30.000000
27.784000 50.000000
27.457001 75.000000
23.105000 100.00000
18.014999 125.00000
16.023001 150.00000
13.165000 200.00000
12.395000 250.00000
11.878000 300.00000
10.846000 400.00000
10.083000 500.00000
9.3690004 600.00000
8.7110004 700.00000
8.0590000 800.00000
7.3579998 900.00000
6.4889998 1000.0000
6.0900002 1100.0000
5.6090002 1200.0000
5.1100001 1300.0000
4.6729999 1400.0000
4.2399998 1500.0000
3.3950000 1750.0000
2.7400000 2000.0000
2.0660000 2500.0000
1.7020000 3000.0000
```



Example -3: Temperature, Salinity, Density

Input file : profile.txt
Data format : Depth, Temperature, Salinity, Density
gmt code :

```
#
# Temperature
#
gmt psbasemap -P -JX4/-12 -R0/30/0/3000 -B10:"T (@+o@+C)":/500:"Depth (m)":WN -X4 -Y10 -K > fig.ps
awk '{print $2, $1}' profile.txt > junk
gmt psxy junk -J -R -B -W1 -O -K >> fig.ps

#
# Salinity
#
gmt psbasemap -J -R34/36/0/3000 -B1:"S (PPT)":/500:"Depth (m)":wN -X5 -O -K >> fig.ps
awk '{print $3, $1}' profile.txt > junk
gmt psxy junk -J -R -B -W1 -O -K >> fig.ps

#
# Density
#
gmt psbasemap -J -R1020/1045/0/3000 -B10:"Density ":/500:"Depth (m)":wN -X6 -O -K >> fig.ps
awk '{print $4, $1}' profile.txt > junk
gmt psxy junk -J -R -B -W1 -O >> fig.ps
```



Session - 8

Symbol Plot

To plot symbols, gmt uses the command psxy itself with the option -S.

Options for -S

−Sr	**r**ectangle. No size needs to be specified, but the x- and y-dimensions must be found in columns 3 and 4.
−Ss	**s**quare. *size* is diameter of circumscribing circle.
−St	**t**riangle. *size* is diameter of circumscribing circle.
−Sv	**v**ector. Direction (in degrees counter-clockwise from horizontal) and length must be found in columns 3 and 4. *size*, if present, will be interpreted as *arrowwidth/headlength/headwidth* [Default unit is 0.075**c**/0.3**c**/0.25**c** (or 0.03**i**/0.12**i**/0.1**i**)]. By default arrow attributes remains invariant to the length of the arrow. To have the size of the vector scale down with decreasing size, append **n***norm*, where vectors shorter than *norm* will have their attributes scaled by length/*norm*. To center vector on balance point, use **−Svb**; to align point with the vector head, use **−Svh**; to align point with the vector tail, use **−Svt** [Default]. To give the head point's coordinates instead of direction and length, use **−Svs**. Upper case **B, H, T, S** will draw a double-headed vector [Default is single head].
−SV	Same as **−Sv**, except azimuth (in degrees east of north) should be given instead of direction. The azimuth will be mapped into an angle based on the chosen map projection (**−Sv** leaves the directions unchanged.)
−Sw	pie **w**edge. Start and stop directions (in degrees counter-clockwise from horizontal) for pie slice must be found in columns 3 and 4.
−SW	Same as **−Sw**, except azimuths (in degrees east of north) should be given instead of the two directions. The azimuths will be mapped into angles based on the chosen map projection (**−Sw** leaves the directions unchanged.)
−Sx	cross (x). *size* is diameter of circumscribing circle.
−Sy	y-dash (|). *size* is the length of a short vertical line segment.

−S- x-dash (-). *size* is the length of a short horizontal line segment.

−S+ plus (+). *size* is diameter of circumscribing circle.

−Sa st**a**r. *size* is diameter of circumscribing circle.

−Sb Vertical **b**ar extending from *base* to y. *size* is bar width. Append **u** if *size* is in x-units [Default is plot-distance units]. By default, *base* = ymin. Append **b***base* to change this value.

−SB Horizontal **b**ar extending from *base* to x. *size* is bar width. Append **u** if *size* is in y-units [Default is plot-distance units]. By default, *base* = xmin. Append **b***base* to change this value.

−Sc **c**ircle. *size* is diameter of circle.

−Sd **d**iamond. *size* is diameter of circumscribing circle.

−Se **e**llipse. Direction (in degrees counter-clockwise from horizontal), major_axis, and minor_axis must be found in columns 3, 4, and 5.

−SE Same as **−Se**, except azimuth (in degrees east of north) should be given instead of direction. The azimuth will be mapped into an angle based on the chosen map projection (**−Se** leaves the directions unchanged.) Furthermore, the axes lengths must be given in km instead of plot-distance units. An exception occurs for a linear projection in which we assume the ellipse axes are given in the same units as **−R**.

−Sf **f**ront. **−Sf***gap/size*[*dir*][*type*][:*offset*]. Supply distance gap between symbols and symbol size. If *gap* is negative, it is interpreted to mean the number of symbols along the front instead. Append *dir* to plot symbols on the **l**eft or **r**ight side of the front [Default is centered]. Append *type* to specify which symbol to plot: **b**ox, **c**ircle, **f**ault, **s**lip, or **t**riangle. [Default is fault]. Slip means left-lateral or right-lateral strike-slip arrows (centered is not an option). Append :*offset* to offset the first symbol from the beginning of the front by that amount [Default is 0].

−Sg octa**g**on. *size* is diameter of circumscribing circle.

−Sh **h**exagon. *size* is diameter of circumscribing circle.

−Si **i**nverted triangle. *size* is diameter of circumscribing circle.

−Sj Rotated rectangle. Direction (in degrees counter-clockwise from horizontal), x-dimension, and y-dimension must be found in columns 3, 4, and 5.

−SJ Same as **−Sj**, except azimuth (in degrees east of north) should be given instead of direction. The azimuth will be mapped into an angle based on the chosen map projection (**−Sj** leaves the directions unchanged.) Furthermore, the dimensions must be given in km instead of plot-distance units. An exception occurs for a linear projection in which we assume the dimensions are given in the same units as **−R**.

−Sk **k**ustom symbol. Append <name>/*size*, and we will look for a definition file called <name>.def in (1) the current directory or (2) in ~/.gmt or (3) in **$GMT_SHAREDIR**/custom. The symbol as defined in that file is of size 1.0 by default; the appended *size* will scale symbol accordingly. Users may add their own custom *.def files; see CUSTOM SYMBOLS below.

−Sl **l**etter or text string (less than 64 characters). Give size, and append /*string* after the size. Note that the size is only approximate; no individual scaling is done for different characters. Remember to escape special characters like *. Optionally, you may append %*font* to select a particular font [Default is **ANNOT_FONT_PRIMARY**].

−Sm **m**ath angle arc, optionally with one or two arrow heads. The *size* is the radius of the arc. Start and stop directions (in degrees counter-clockwise from horizontal) for arc must be found in columns 3 and 4. Use **−Smf** to add arrow head at first angle, **−Sml** for arrow head at last angle, and **−Smb** for both [Default is no arrow heads].

−Sn pe**n**tagon. *size* is diameter of circumscribing circle.

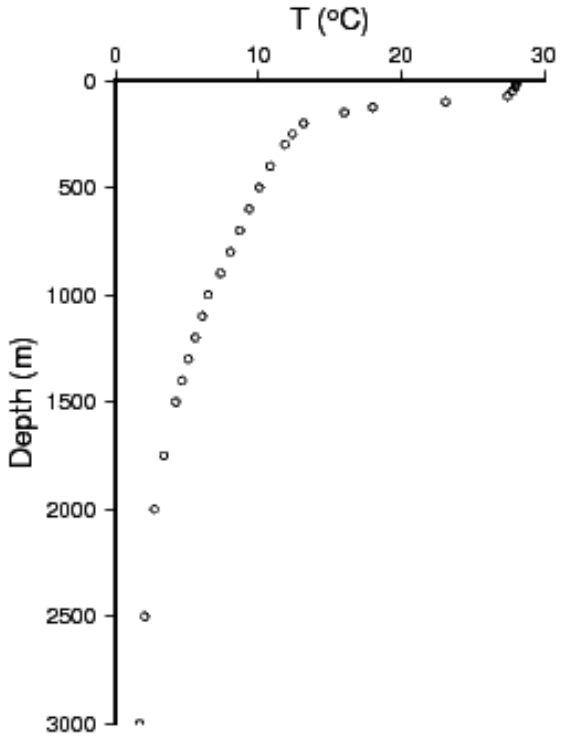
−Sp **p**oint. No size needs to be specified (1 pixel is used).

−Sq **q**uoted line, i.e., lines with annotations such as contours. Append **[d|D|f|l|L|n|x|X]***info*[:*labelinfo*]. The required argument controls the placement of labels along the quoted lines. Choose among five controlling algorithms:

Example-1: Temperature profile

Circle with 0.15cm diameter

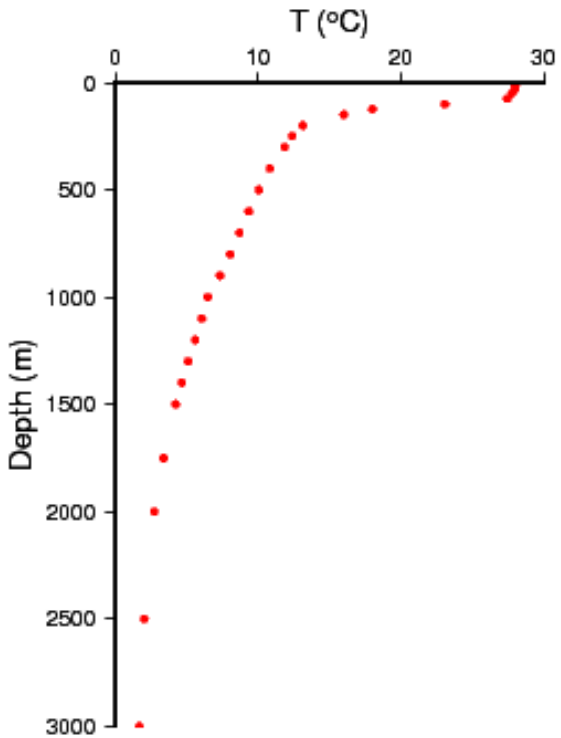
gmt psxy profile-t.txt -J -R -B -Sc0.15 -O >> fig.ps



Example-2: Temperature profile

Red color filled circle with 0.15cm diameter

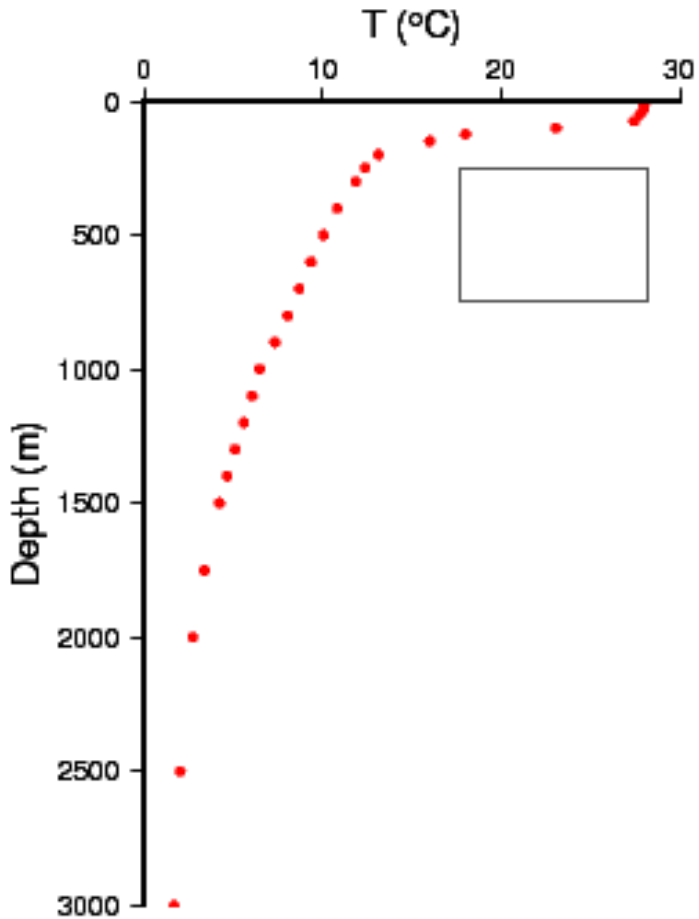
```
gmt psxy profile-t.txt -J -R -B -Sc0.15 -G255/0/0 -O >> fig.ps
```



Example-3: Temperature profile

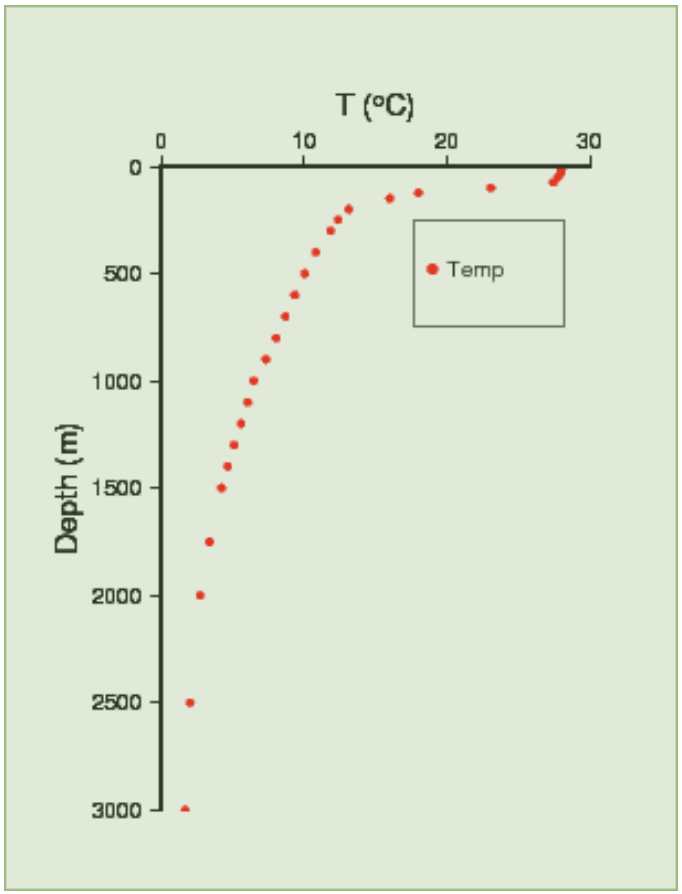
Drawing a rectangle in the plot

```
gmt psbasemap -P -JX8/-12 -R0/30/0/3000 -B10:"T (@+o@+C)":/500:"Depth (m)":WN -X4 -Y10 -K > fig.ps

gmt psxy profile-t.txt -J -R -B -Sc0.15 -G255/0/0 -O -K >> fig.ps

echo "23 500 2.8 2" > junk

gmt psxy junk -J -R -B -Sr -O >> fig.ps
```



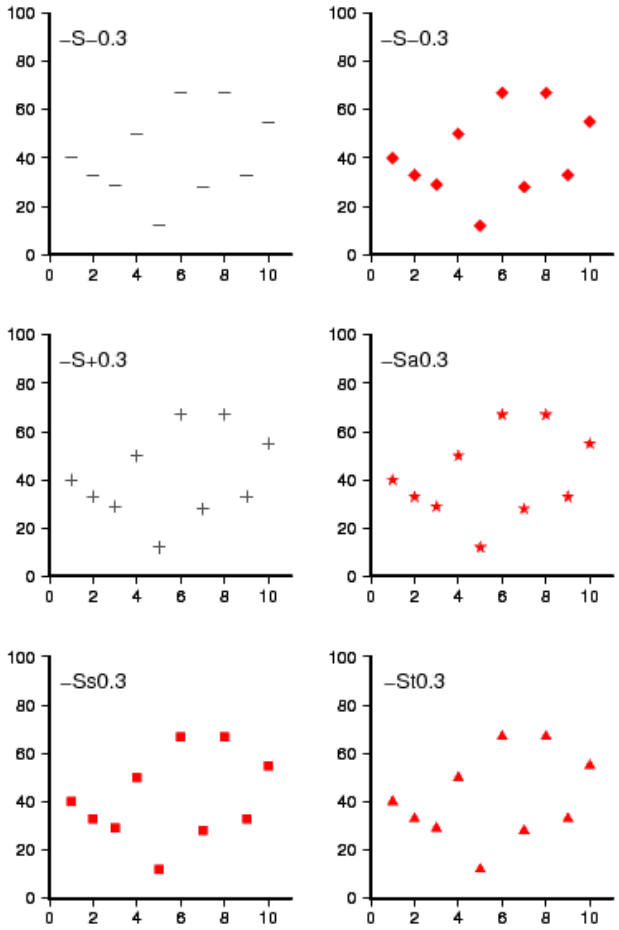
Adding text inside the box


```
# Basemap
gmt psbasemap -P -JX8/-12 -R0/30/0/3000 -B10:"T (@+o@+C)":/500:"Depth (m)":WN -X4 -Y10 -K > fig

# Symbol plot
gmt psxy profile-t.txt -J -R -B -Sc0.15 -G255/0/0 -O -K >> fig.ps

# Draw box
echo "23 500 2.8 2" > junk
gmt psxy junk -J -R -B -Sr -O -K >> fig.ps

# circle inside box with red color
echo "19 480" > junk
gmt psxy junk -J -R -B -Sc0.2 -Gred -O -K >> fig.ps

# text
echo "20 480 Temp" > junk
gmt pstext junk -N -R -J -F+f12p,Helvetica,black+a0+jLM -B -O >> fig.ps
```



Other symbols


```
# Square
gmt psbasemap -P -JX6 -R0/11/0/100 -B2:"":/20:"":WS -X4 -Y5 -K > fig.ps
gmt psxy sample.txt -J -R -B -Ss0.3 -G255/0/0 -O -K >> fig.ps
echo "2 90 -Ss0.3" | gmt pstext -J -R -B -N -F+f15p -O -K >> fig.ps

# triangle
gmt psbasemap -P -JX -R -B -X8 -O -K >> fig.ps
gmt psxy sample.txt -J -R -B -St0.3 -G255/0/0 -O -K >> fig.ps
echo "2 90 -St0.3" | gmt pstext -J -R -B -N -F+f15p -O -K >> fig.ps

# plus
gmt psbasemap -P -JX -R -B -X-8 -Y8 -O -K >> fig.ps
gmt psxy sample.txt -J -R -B -S+0.3 -G255/0/0 -O -K >> fig.ps
echo "2 90 -S+0.3" | gmt pstext -J -R -B -N -F+f15p -O -K >> fig.ps

# star
gmt psbasemap -P -JX -R -B -X8 -O -K >> fig.ps
gmt psxy sample.txt -J -R -B -Sa0.3 -G255/0/0 -O -K >> fig.ps
echo "2 90 -Sa0.3" | gmt pstext -J -R -B -N -F+f15p -O -K >> fig.ps

# dash
gmt psbasemap -P -JX -R -B -X-8 -Y8 -O -K >> fig.ps
gmt psxy sample.txt -J -R -B -S-0.3 -G255/0/0 -O -K >> fig.ps
echo "2 90 -S-0.3" | gmt pstext -J -R -B -N -F+f15p -O -K >> fig.ps

# diamond
gmt psbasemap -P -JX -R -B -X8 -O -K >> fig.ps
gmt psxy sample.txt -J -R -B -Sd0.3 -G255/0/0 -O -K >> fig.ps
echo "2 90 -S-0.3" | gmt pstext -J -R -B -N -F+f15p -O >> fig.ps
```



Session - 9

Contour Plots

'makecpt' command

Purpose : To create a color palette table used to plot contour maps.

Syntax : makecpt -T -C > outputfile

Options
-T Define the range of values of the parameter to be plotted given in the format "low/high/increment"
-C Type of color palette (rainbow, polar, grey, gebco, no_green)

Example : makecpt -T20/40/2 -Crainbow > out.cpt

Contour commands

1. pscontour
   Contour table data by direct triangulation
2. grdcontour
   Make contour map using grided data
3. grdimage
   Color map using grided data

(1) pscontour

pscontour requires an input file which contains data in the x,y,parameter format.

Basemap options : -J, -R, -B

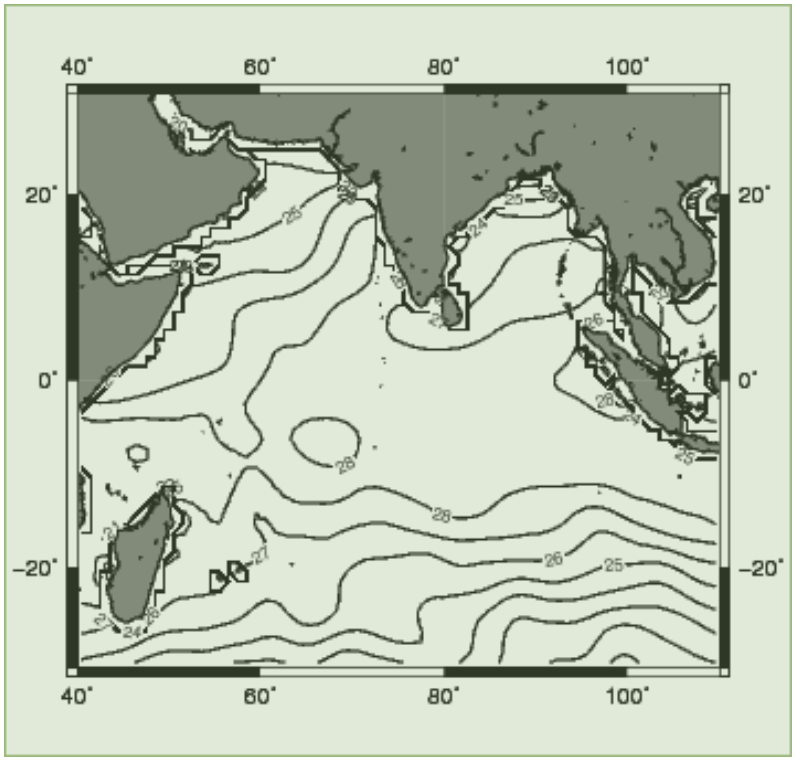
Additional options :
-C : filename of the color palette table
-A : Annotation interval
-W : Line thickness

Example: SST map

GMT script

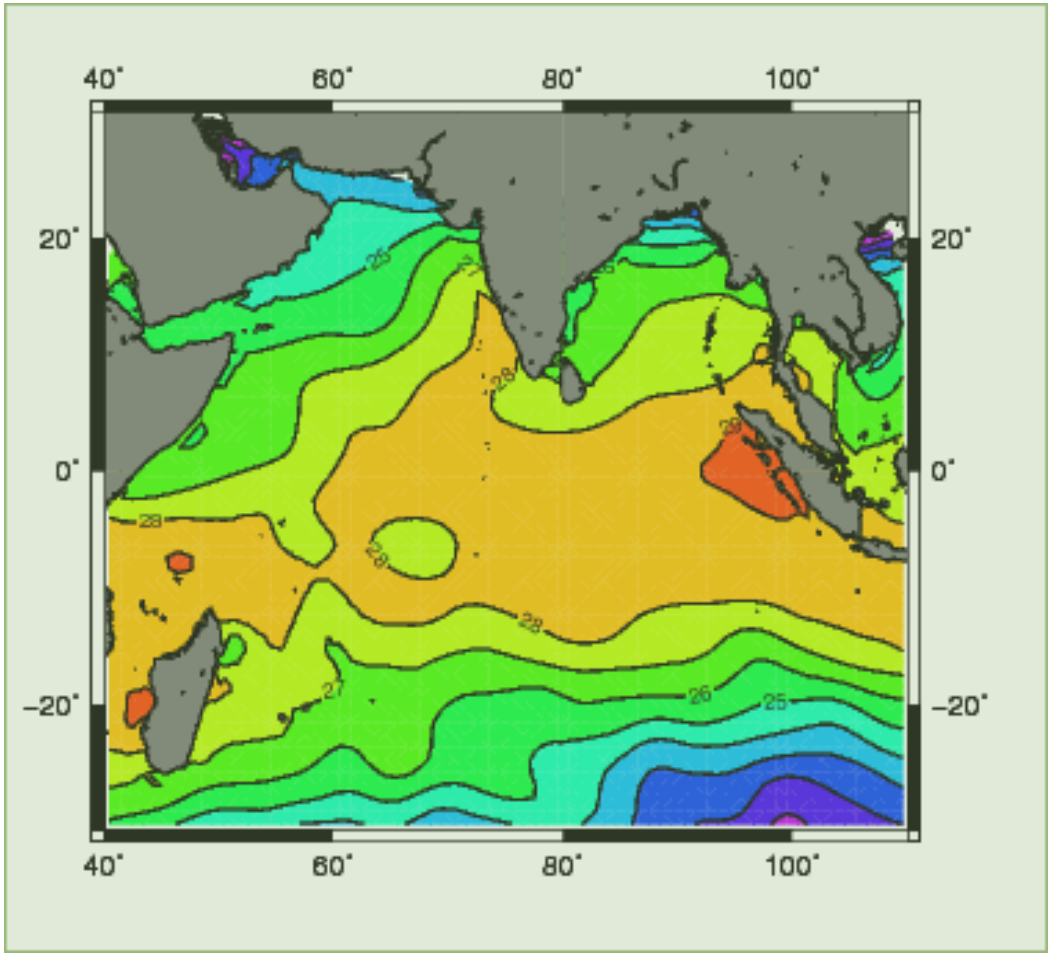
```
# create palette table
gmt makecpt -T20/30/1 -Crainbow > color.cpt

# basemap
gmt psbasemap -P -JM12 -R40/110/-30/30 -B20 -X5 -Y5 -K > fig.ps

# geography map
gmt pscoast -J -R -B -G100 -W1 -O -K >> fig.ps

# contour with pscontour
gmt pscontour temp-jan.out -J -R -B -Ccolor.cpt -A1 -W1 -O >> fig.ps
```

Plot



Color map with -I option

```
# create palette table
gmt makecpt -T20/30/1 -Crainbow > color.cpt

# basemap
gmt psbasemap -P -JM12 -R40/110/-30/30 -B20 -X5 -Y5 -K > fig.ps


# contour with pscontour
awk '$3 > 0 {print $0}' temp-jan.out > junk
gmt pscontour junk -J -R -B -Ccolor.cpt -I -A1 -W1 -O -K >> fig.ps


# geography map
gmt pscoast -J -R -B -G100 -W1 -O >> fig.ps
```



(2) grdcontour

grdcontour requires the input file in the gridded (binary) format. Hence the xyz file in the ascii format has to be converted to binary using one of the three gridding commands.

The GMT modules for gridding are:

1. nearneighbor (Grid table data using a "Nearest neighbor" algorithm")
2. triangulate (Do optimal (Delaunay) triangulation and gridding of Cartesian table data)
3. surface Grid table data using adjustable tension continuous curvature splines

Example : Gridding with nearneighbor

gmt nearneighbor junk -V -R -I1 -Gjunk.grd -S100

-I Grid size

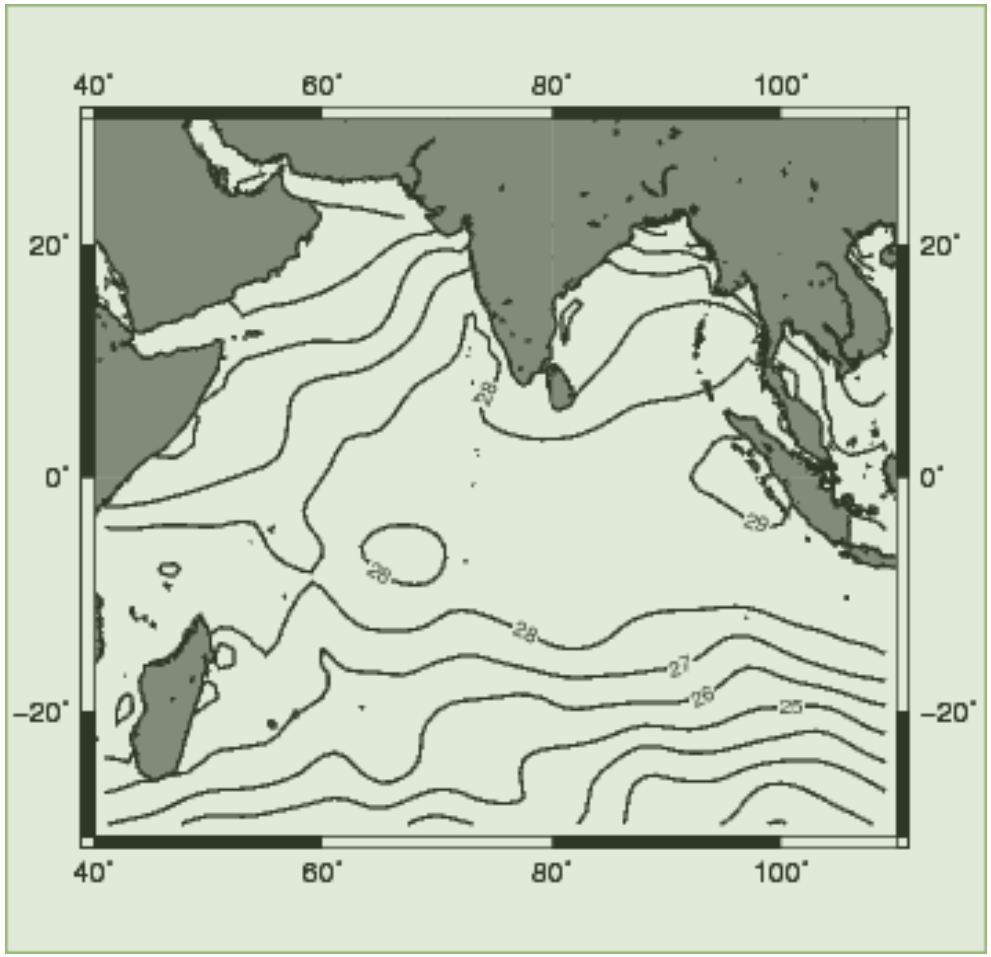
-G Output filename

-S Search radius hat determines which data points are considered close to a node.

GMT Script

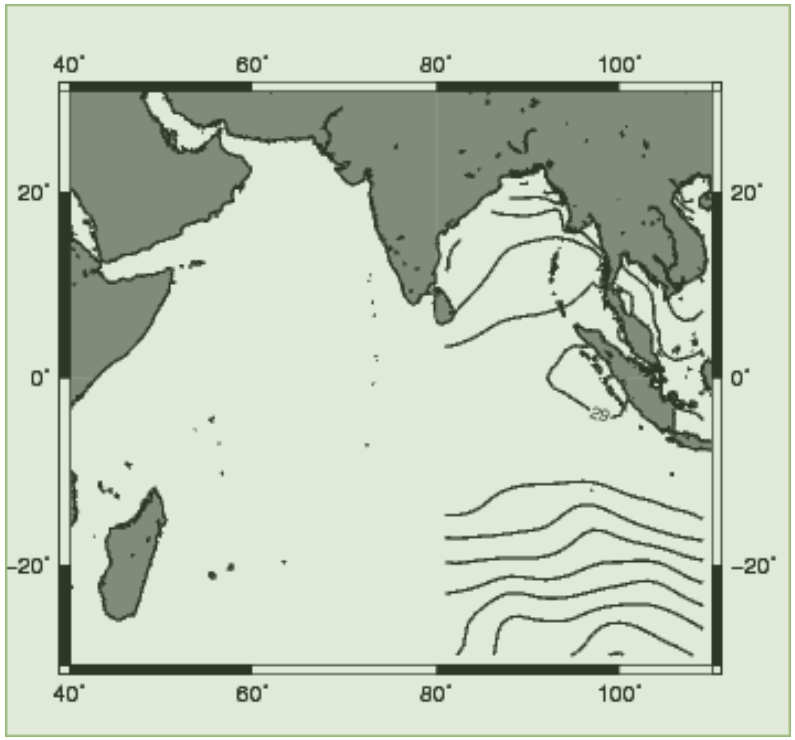
```
#
# Gridding with neighbour
#

# create palette table
gmt makecpt -V -T20/30/1 -Crainbow > color.cpt

# basemap
gmt psbasemap -V -P -JM12 -R40/110/-30/30 -B20 -X5 -Y5 -K > fig.ps


# Remove flag data with awk
awk '$3 > 0 {print $0}' temp-jan.out > junk

# gridding using surface
gmt nearneighbor junk -V -R -I1 -Gjunk.grd -S100

# contour with grdcontour
gmt grdcontour junk.grd -V -J -R -B -C1 -A1f8 -W1 -O -K >> fig.ps

# geography map
gmt pscoast -V -J -R -B -G100 -W1 -O >> fig.ps
```

Figure



Restricting data


```
#
# Gridding with neighbour restricting data
#

# create palette table
gmt makecpt -V -T20/30/1 -Crainbow > color.cpt

# basemap
gmt psbasemap -V -P -JM12 -R40/110/-30/30 -B20 -X5 -Y5 -K > fig.ps


# Remove flag data with awk
awk '$3 > 0 && $1 > 80 {print $0}' temp-jan.out > junk

# gridding using surface
gmt nearneighbor junk -V -R -I1 -Gjunk.grd -S100

# contour with grdcontour
gmt grdcontour junk.grd -V -J -R -B -C1 -A1f8 -W1 -O -K >> fig.ps

# geography map
gmt pscoast -V -J -R -B -G100 -W1 -O >> fig.ps
```



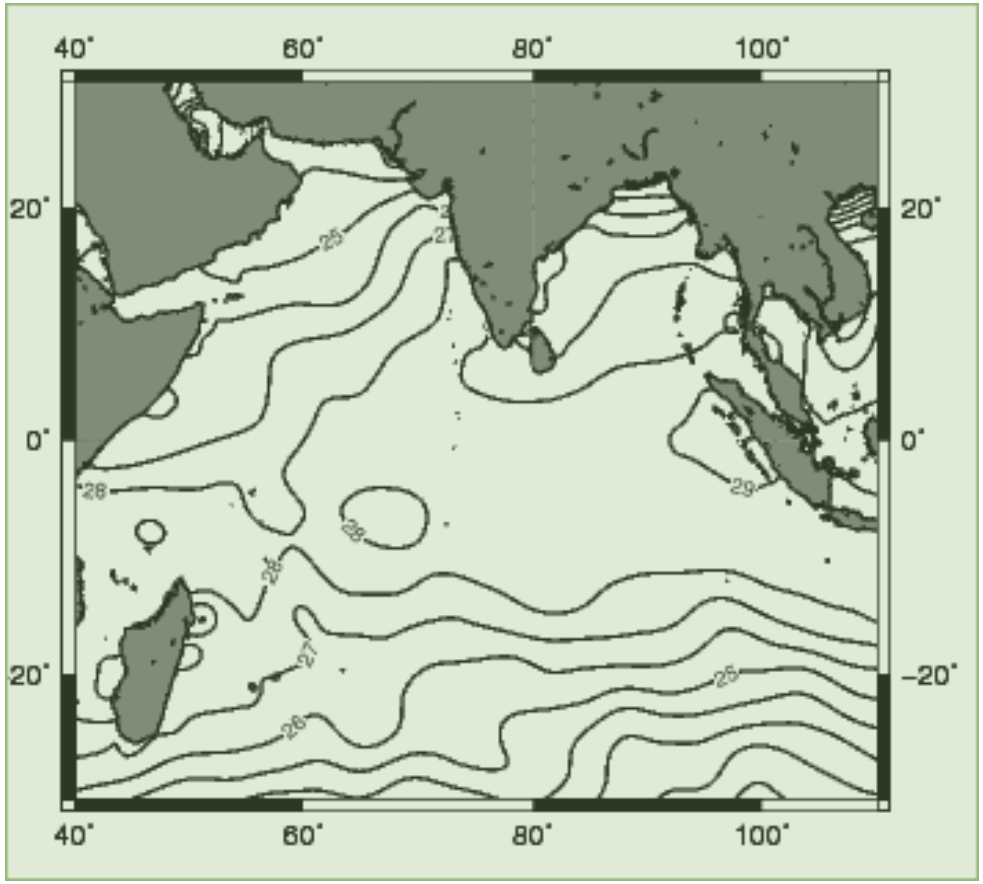
Gridding with triangulate

- triangulate reads one or more ASCII [or binary] files containing x,y[,z] and performs Delaunay triangulation.
- It finds how the points should be connected to give the most equilateral triangulation possible.

Gridding with 'surface'

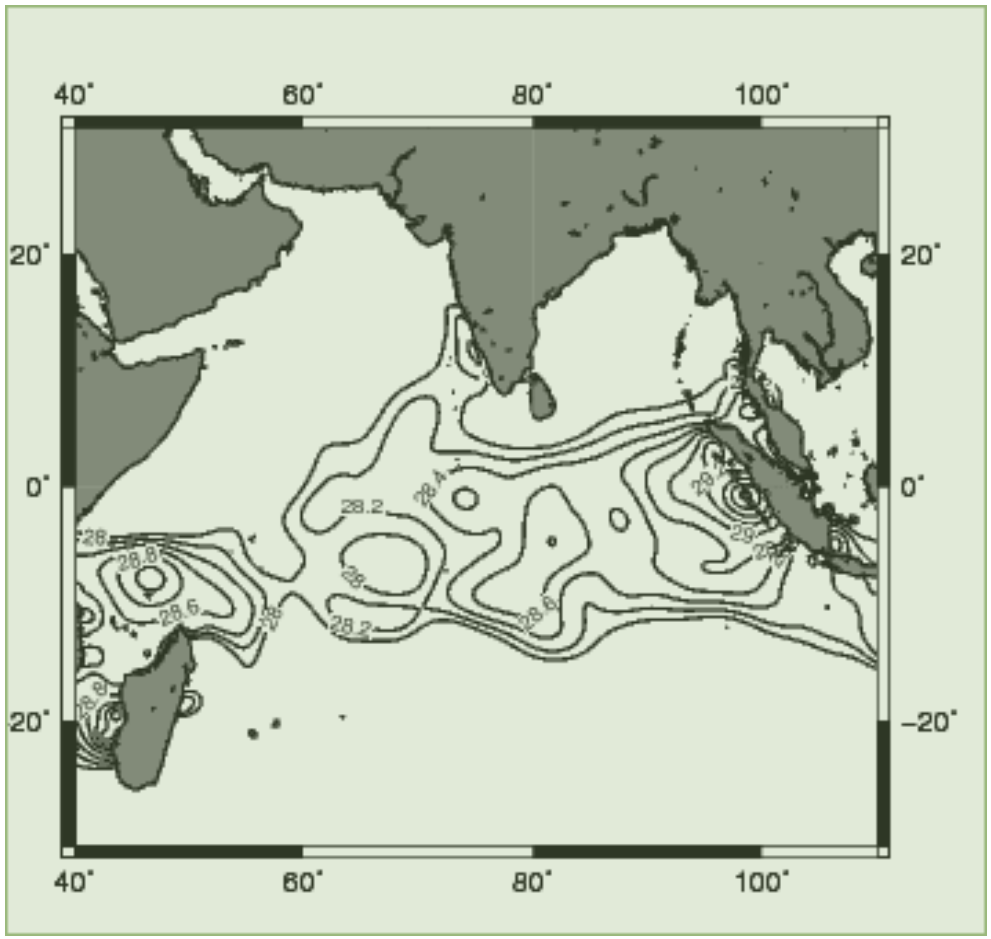
GMT script:

```
# create palette table
gmt makecpt -V -T20/30/1 -Crainbow > color.cpt

# basemap
gmt psbasemap -V -P -JM12 -R40/110/-30/30 -B20 -X5 -Y5 -K > fig.ps


# Remove flag data with awk
awk '$3 > 0 {print $0}' temp-jan.out > junk

# gridding using surface
gmt surface junk -V -R -I0.1 -Gjunk.grd

# contour with grdcontour
gmt grdcontour junk.grd -V -J -R -B -C1 -A1f8 -W1 -O -K >> fig.ps

# geography map
gmt pscoast -V -J -R -B -G100 -W1 -O >> fig.ps
```



Restricting data plotting with -L option.
GMT script:

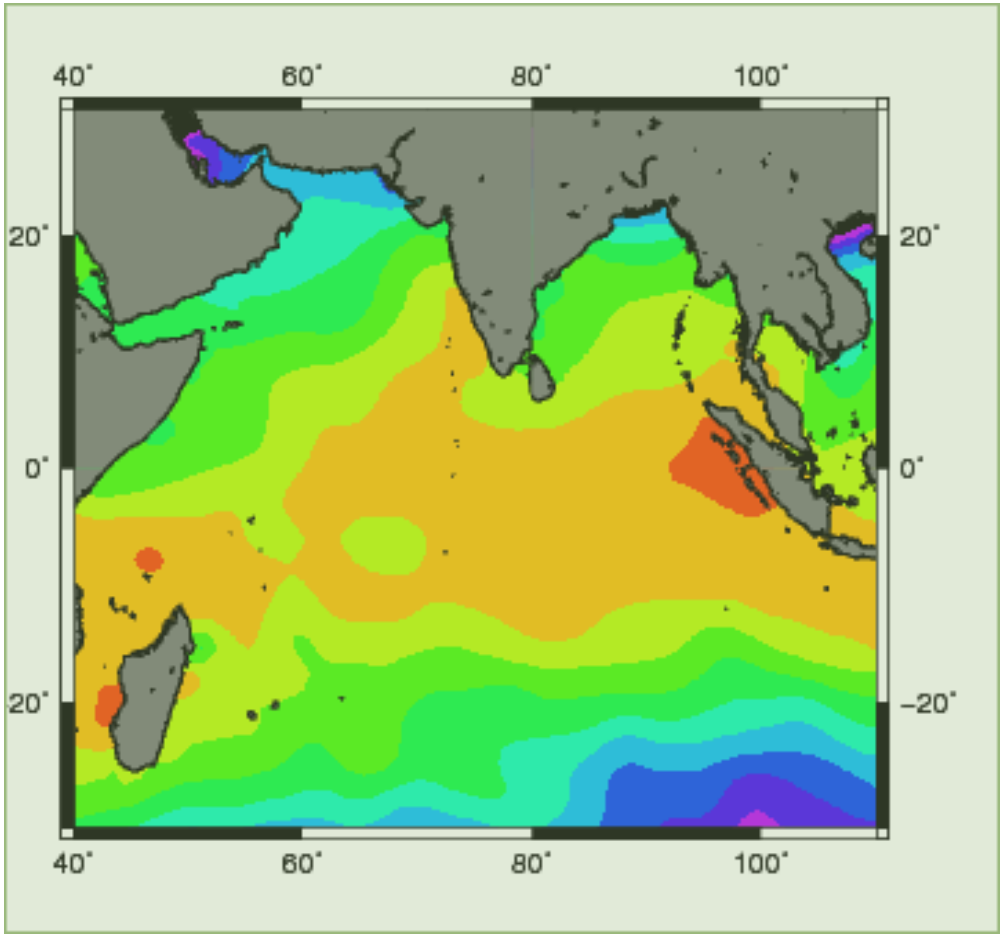
```
# create palette table
gmt makecpt -V -T28/30/0.2 -Crainbow > color.cpt

# basemap
gmt psbasemap -V -P -JM12 -R40/110/-30/30 -B20 -X5 -Y5 -K > fig.ps


# Remove flag data
awk '$3 > 0 {print $0}' temp-jan.out > junk

# gridding using surface
gmt surface junk -V -R -I0.1 -Gjunk.grd

# contour with grdcontour
gmt grdcontour junk.grd -V -J -R -B -C0.2 -A0.2f8 -L28/30 -W1 -O -K >> fig.p

# geography map
gmt pscoast -V -J -R -B -G100 -W1 -O >> fig.ps
```



(3) grdimage

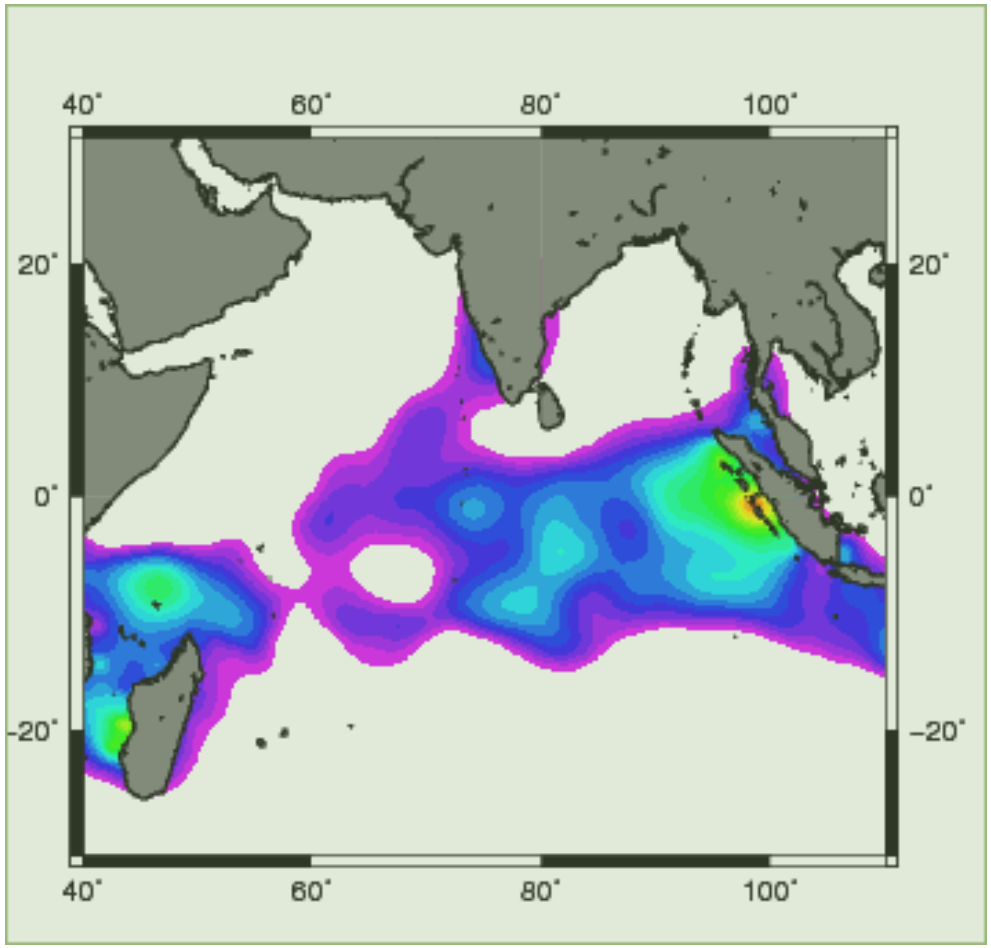
Makes color map using gridded data.


```
# create palette table
gmt makecpt -V -T20/30/1 -Crainbow > color.cpt

# basemap
gmt psbasemap -V -P -JM12 -R40/110/-30/30 -B20 -X5 -Y5 -K > fig.ps


# Remove flag data with awk
awk '$3 > 0 {print $0}' temp-jan.out > junk

# gridding using surface
gmt surface junk -V -R -I0.1 -Gjunk.grd

# contour with grdcontour
gmt grdimage junk.grd -V -J -Ccolor.cpt -O -K >> fig.ps

# geography map
gmt pscoast -V -J -R -B -G100 -W1 -O >> fig.ps
```



Restricting data

```
# Background color as white
gmt gmtset COLOR_BACKGROUND white

# create palette table
gmt makecpt -V -T28/30/0.1 -Crainbow > color.cpt

# basemap
gmt psbasemap -V -P -JM12 -R40/110/-30/30 -B20 -X5 -Y5 -K > fig.ps

# Select data in the range
awk '$3 >= 28 {print $0}' temp-jan.out > junk

# gridding using surface
gmt surface junk -V -R -I0.1 -Gjunk.grd

# contour with grdcontour
gmt grdimage junk.grd -V -J -Ccolor.cpt -O -K >> fig.ps

# geography map
gmt pscoast -V -J -R -B -G100 -W1 -O >> fig.ps
```



Color Scale with psscale

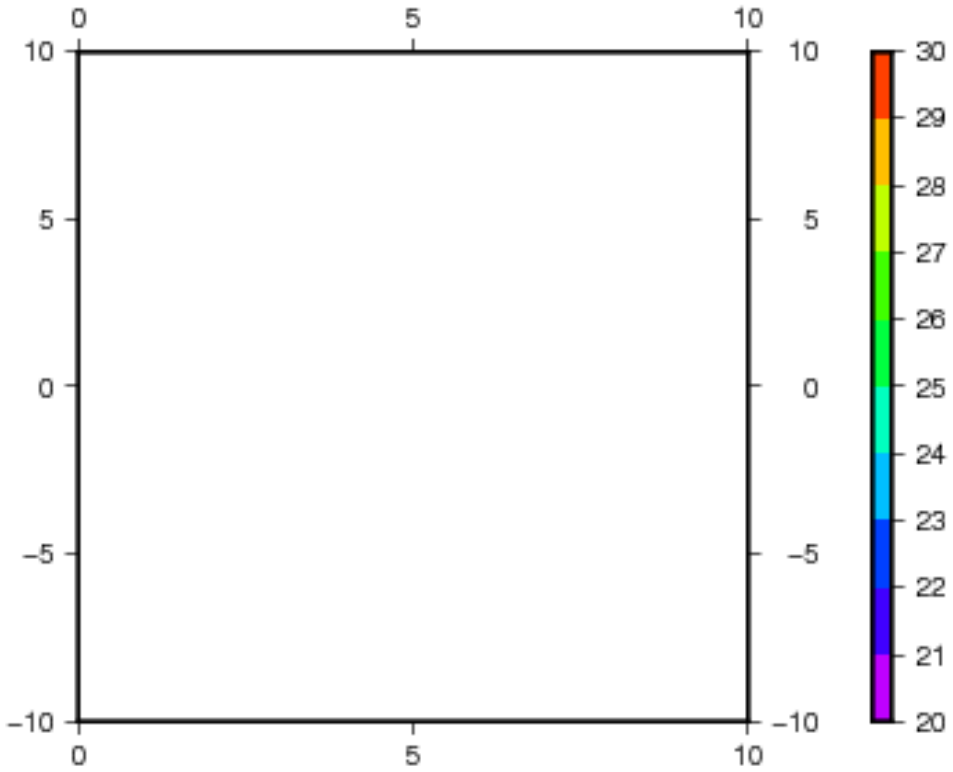
Purpose: Plot a gray or color scale-bar on maps

psscale Options

- -D Defines the reference point on the map for the color scale
- -C Name of the cpt file
- -B Set annotation, tick, and gridline interval for the colorbar

Example: Vertical scale

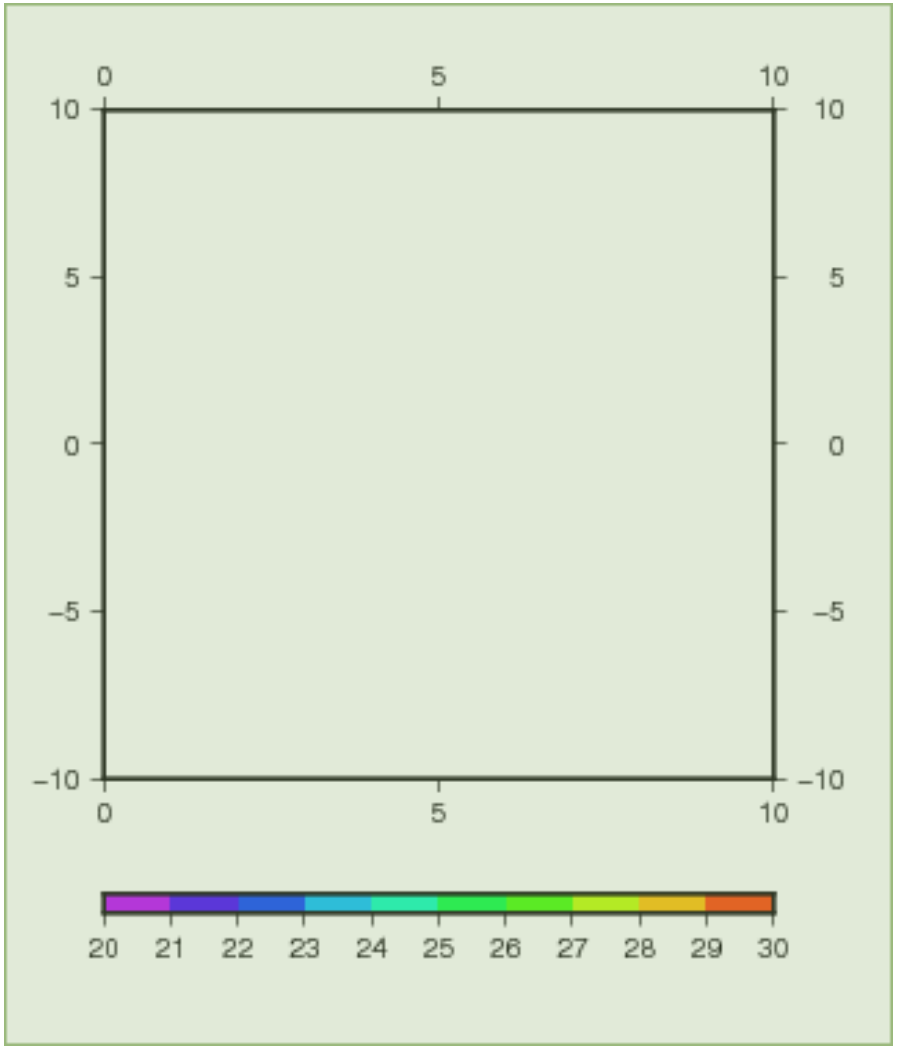

```
# Create cpt file
gmt makecpt -T20/30/1 -Crainbow > color.cpt

# basemap
gmt psbasemap -P -JX10 -R0/10/-10/10 -B5 -X5 -Y10 -K > fig.ps

# psscale
gmt psscale -R -J -Dx12c/0c+w10c/0.3c+jCB -Ccolor.cpt -B1 -O >> fig.ps
```



Example: Horizontal scale


```
# Create cpt file
gmt makecpt -T20/30/1 -Crainbow > color.cpt

# basemap
gmt psbasemap -P -JX10 -R0/10/-10/10 -B5 -X5 -Y10 -K > fig.ps

# psscale
gmt psscale -R -J -Dx5c/-2c+w10c/0.3c+jCB+h -Ccolor.cpt -B1 -O >> fig.ps
```

Example: Horizontal scale



Session - 10

Vector Plots

Methods

1. Using psxy
2. Using grdvector

(1) Vector Plot using psxy

The options -Sv and -SV in psxy is used to plot vectors.

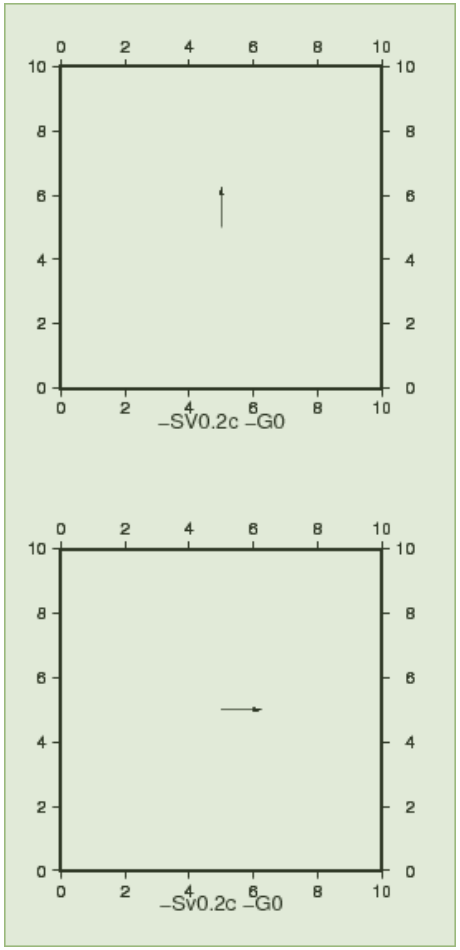
-Sv Syntax : -Svsize
Direction (in degrees counterclockwise from horizontal) and length must be found in column 3 and 4. size provides the size of the arrow head.

-SV Syntax : -Svsize
Direction (in degrees clockwise with respect to north) and length must be found in column 3 and 4.

The option -G can be used to color the arrow.

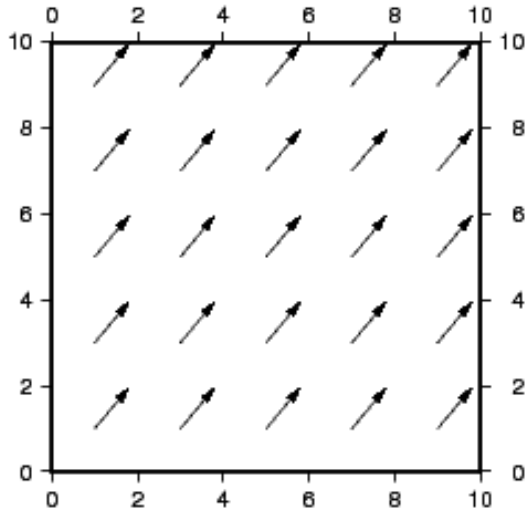
Example-1


```
#
# Plot vector using psxy
#

#basemap
gmt psbasemap -P -JX8 -R0/10/0/10 -B2 -X5 -Y4 -K > fig.ps
# Create a data
echo "5 5 0 1" > junk
# draw vector
gmt psxy junk -J -R -B -Sv0.2c+e -G0 -O -K >> fig.ps
# text
echo "5 -1 -Sv0.2c -G0" | gmt pstext -N -J -F+f15p -R -B -O -K >> fig.ps


#basemap
gmt psbasemap -J -R0/10/0/10 -B2 -Y12 -O -K >> fig.ps
# Create a data
echo "5 5 0 1" > junk
# draw vector
gmt psxy junk -J -R -B -SV0.2c+e -G0 -O -K >> fig.ps
# text
echo "5 -1 -SV0.2c -G0" | gmt pstext -N -J -F+f15p -R -B -O >> fig.ps
```



Example-2


```
#
# Plot vector using psxy
#

#basemap
gmt psbasemap -P -JX8 -R0/10/0/10 -B2 -X5 -Y4 -K > fig.ps
# draw vector
gmt psxy vector2.txt -J -R -B -SV0.3c+e -G0 -O >> fig.ps
```



Example -3: Vector on geographical map

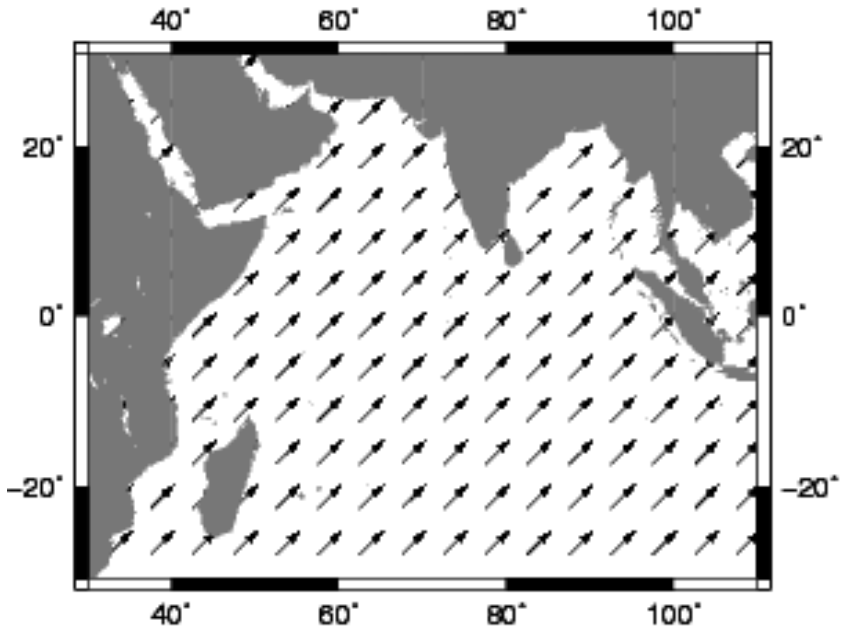
A sample current data for Indian Ocean was generated using a matlab/octave program.

```
% create a current vector data for Indian Ocean

fid=fopen("vector3.out", "w");

for y=-27.5:5.0:27.5
  for x=32.5:5.0:107.5
    fprintf(fid,"%f %f %f,%f\n", x,y,45.0,0.5)
  end
end

fclose(fid)
```

GMT script

```
#
# Plot vector using psxy for Indian Ocean
#

#basemap
gmt psbasemap -P -JM10 -R30/110/-30/30 -B20 -X5 -Y10 -K > fig.ps
# draw vector
gmt psxy vector3.out -J -R -B -SV0.2c+e -G0 -O -K >> fig.ps
gmt pscoast -JM -R -B -Dl -G100 -O >> fig.ps
```

Plot



(2) Vector plot using grdvector

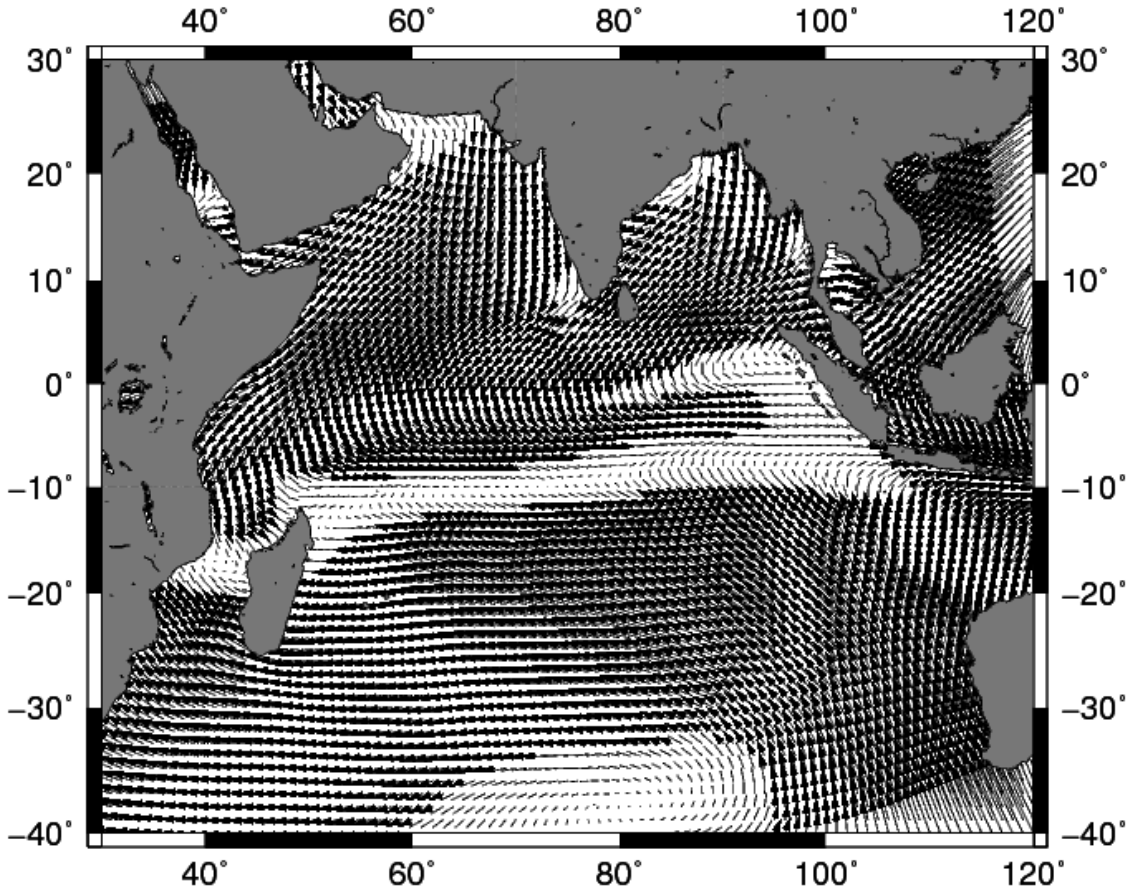
grdvector plots vector field from two velocity component grided data sets.

Options

- -J Projection
- -W Line thickness
- -Q Vector parameters
- -C Color to indicate vector magnitude.

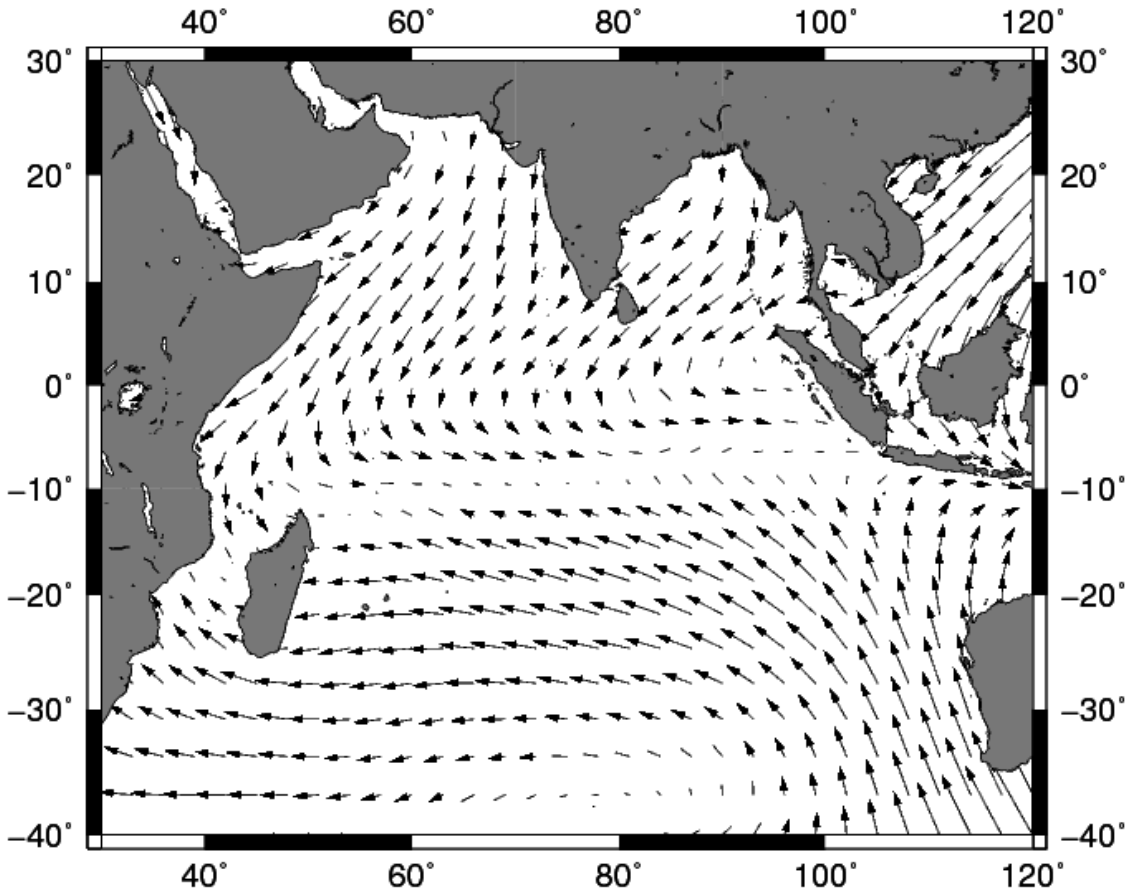
Example-1


```
#
# Vector using grdvector
#

# Basemap
gmt psbasemap -P -JM12 -R30/120/-40/30 -B20/10 -X5 -Y5 -K > fig.ps

# Extract U and V component data
awk '$3 > -999 {print $1, $2, $3*50}' wind.txt > u.txt
awk '$3 > -999 {print $1, $2, $4*50}' wind.txt > v.txt

# Gridding using surface
gmt surface u.txt -V -R -I1 -Gu.grd
gmt surface v.txt -V -R -I1 -Gv.grd

# Vector plot with grdvector
gmt grdvector u.grd v.grd -J -W -Q0.07i+e -G0 -O -K >> fig.ps

# geography
gmt pscoast -J -R -B -Dl -W -G100 -O >> fig.ps
```



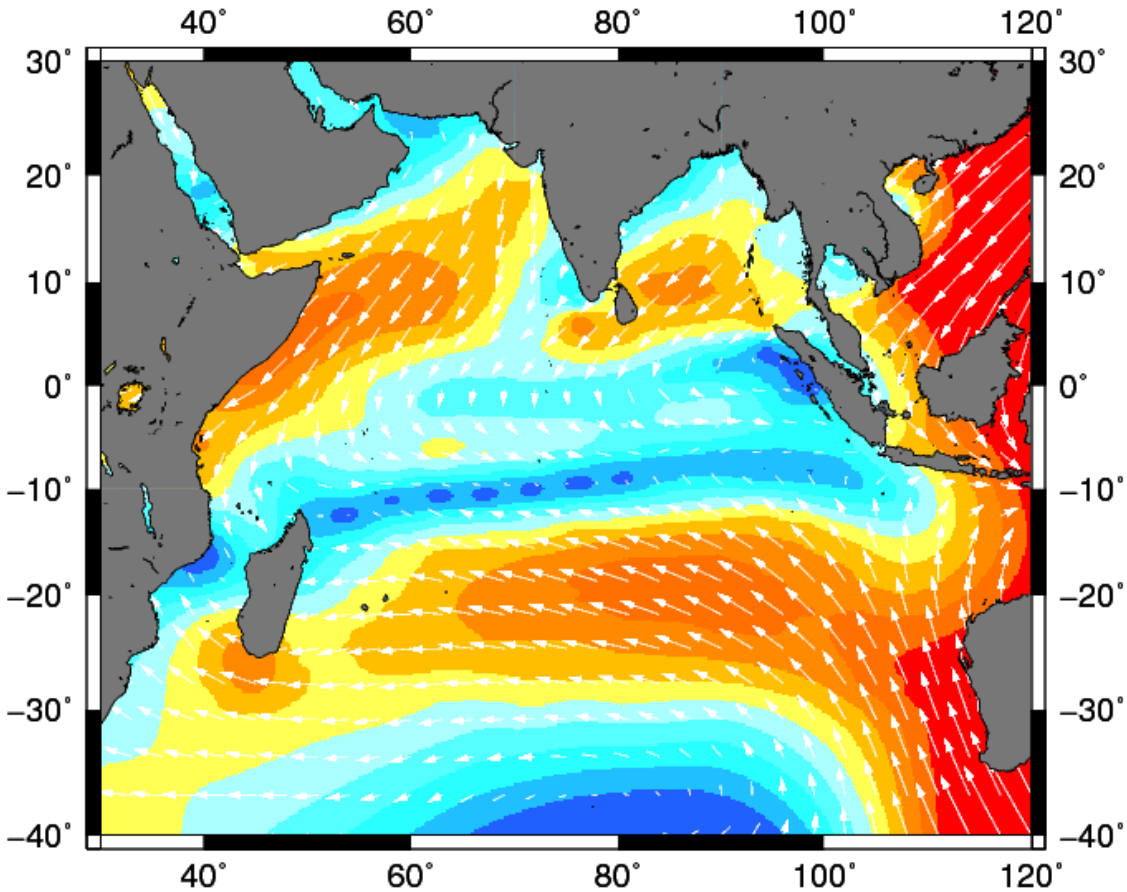
Example - 2:


```
#
# Vector using grdvector (-I3 in grdvector)
#

# Basemap
gmt psbasemap -P -JM12 -R30/120/-40/30 -B20/10 -X5 -Y5 -K > fig.ps

# Extract U and V component data
awk '$3 > -999 {print $1, $2, $3*50}' wind.txt > u.txt
awk '$3 > -999 {print $1, $2, $4*50}' wind.txt > v.txt

# Gridding using surface
gmt surface u.txt -V -R -I3 -Gu.grd
gmt surface v.txt -V -R -I3 -Gv.grd

# Vector plot with grdvector
gmt grdvector u.grd v.grd -J -W -Q0.07i+e -G0 -O -K >> fig.ps

# geography
gmt pscoast -J -R -B -Dl -W -G100 -O >> fig.ps
```



Example-3: Magnitude and vector


```
#
# Vector using grdvector (COlor vector)
#

# Create cpt file
gmt makecpt -T0/10/1 -Crainbow > color.cpt
gmt makecpt -T0/10/1 -Cno_green > spd.cpt

# Basemap
gmt psbasemap -P -JM12 -R30/120/-40/30 -B20/10 -X5 -Y5 -K > fig.ps

# Extract U and V component data
awk '$3 > -999 {print $1, $2, $3*50}' wind.txt > u.txt
awk '$3 > -999 {print $1, $2, $4*50}' wind.txt > v.txt
awk '$3 > -999 {print $1, $2, $5}' wind.txt > spd.txt

# Gridding using surface
gmt surface u.txt -V -R -I3 -Gu.grd
gmt surface v.txt -V -R -I3 -Gv.grd
gmt surface spd.txt -V -R -I0.2 -Gspd.grd

# contour
gmt grdimage spd.grd -J -Cspd.cpt -O -K >> fig.ps

# Vector plot with grdvector
gmt grdvector u.grd v.grd -J -Ccolor.cpt -W -Q0.07i+e -G0 -O -K >> fig.ps

# geography
gmt pscoast -J -R -B -Dl -W -G100 -O >> fig.ps
```



Thank You!