GARY SHERMAN

THE GEOSPATIAL DESKTOP

OPEN SOURCE GIS AND MAPPING

locate PRESS

The Geospatial Desktop: Open Source GIS and Mapping
by Gary Sherman

Published by Locate Press

978-0-9868052-1-9

Editor Tyler Mitchell

Cover Design Julie Springer

Interior Design Based on Tufte-LaTeX

Publisher Website `http://www.locatepress.com`

Book Website `http://geospatialdesktop.com`

Contents

To get the book, see **`http://locatepress.com`**

1

Forward

Locate Press is proud to bring this second edition of Gary's book into print. The original title, *Desktop GIS*, published by Pragmatic Press successfully sold out, yet there is increasing pressure to have it back in print.

It is special in another way too. It is the first title being published by Locate Press. As our flagship book for this new venture we are thrilled to have such a well-known and competent author as Gary Sherman. His enthusiasm and inspiration have been a pivotal influence in helping us start our publishing endeavour. Thank you Gary!

We've found that it is particularly sought-after by academics looking for a text book for introducing open source GIS into their courses. Seasoned open source GIS users also have been looking for copies to give to their friends.

The demand for having long term stable access to titles like this won't be going away anytime soon. Open source geospatial technology continues to take root around the world and across sectors. Financial crises tend to encourage open source adoption due to the

low price tag. Likewise more organisations need to adopt open standards for interoperability and open source geo applications have proven themselves regularly on that front. So now, more than ever, users are seeking out high quality training material from subject matter experts such as Gary.

The movement toward both cloud computing and mobile application platforms also puts increased stress on those products that cannot operate in a free and open source operating system. Fortunately, most open source geo applications are available across all major desktop and server operating systems. So when you need to move from one operating system to the other, your experience will still be similar to what you will learn in this book.

You are likely to find that this book will fill gaps in ways you didn't expect, from practical everyday tips for selecting software to in-depth examples for completing obscure tasks. I believe this is a special book, without comparison, as few other authors have yet taken on the challenge of covering this broad landscape with significant depth.

Enjoy the book,

Tyler Mitchell
Publisher

13

Using Command-Line Tools

(Partial Excerpt from The Geospatial Desktop)

Command-line tools provide a powerful way to manipulate data, especially when you want to process in batches using a script. This chapter describes some of the more common and useful command-line tools and illustrates how to use them to perform common data manipulation, conversion, and map generation tasks. We will take a look at the following:

- Generic mapping tools (GMT)
- Converting and appending data using GDAL/OGR
- PostGIS

13.1 GMT

For a very brief introduction to GMT, see Appendix 16, Appendix A: Survey of Desktop GIS Software, on page 299. In this section, we will take a look at using GMT to create nicely formatted maps for displaying and printing. But before we can do that, we need to make sure you have GMT installed. If not, take a look at Section 17.5, GMT, on page 320 for some hints to get you started.

The GMT commands create Encapsulated PostScript (EPS) output. If you are using Linux or OS X, you should already have the tools you need to view `eps` files. On Windows you will need a viewer that supports EPS. One such viewer is GSview, which allows EPS files to be viewed and printed. For other options, use your favorite Internet search engine to find a suitable application that works for you.

A Simple GMT Example

Figure 13.1: Hemisphere view of Earth created with GMT

To get started, let's take a look at how to generate a simple globe like that shown in Figure 13.1. The code is pretty simple, although it's a bit arcane at first glance:


```
pscoast -JA0/20/4.5i -Bg30/g15 -Dl -A2000 -G187/142/46 -S109/202/255 \
  -R0/360/-90/90 -P -N1 > simple_hemi.eps
```

Let's examine the switches used to generate the image. First, GMT's `pscoast` command requires information about the coordinate system you want to use. This is specified by using the `-J` switch. GMT supports a nice selection of coordinate systems including the following:

- Albers Conic Equal Area
- Lambert Conic Conformal
- Equidistant Conic
- Lambert Azimuthal Equal Area
- Stereographic Equal Angle
- Orthographic
- Azimuthal Equidistant
- Gnomonic
- Mercator
- Transverse Mercator
- Universal Transverse Mercator
- Oblique Mercator
- Cassini Cylindrical
- Cylindrical Equidistant
- General Cylindrical
- Miller Cylindrical
- Miscellaneous

Each projection has a specific argument that must be supplied to the `-J` switch. Looking back at the globe example, you'll see that `-JA` was used to specify the Lambert Azimuthal Equal Area projection. I know because the programs that make up GMT provide a complete description of what's expected as input when you run them with no options. For example, if we enter the `pscoast` command, we get several screens of options and switches. The first part contains the available projection switches and their syntax:


```
$ pscoast
pscoast 4.5.7 [64-bit] - Plot continents, shorelines, rivers, and borders on maps

usage: pscoast -B<params> -J<params>
[-A<min_area>[/<min_level>/<max_level>][+r|l][+p<percent>]]
[-R<west>/<east>/<south>/<north>[/<zmin/zmax>][r]]
    [-C[<feature>/]<fill>]
[-D<resolution>][+]
[-E<azim>/<elev>[+w<lon>/<lat>[<z>][+v<x0>/<y0>]]
[-G[<fill>]]
[-I<feature>[/<pen>]]
[-Jz|Z<params>]
[-K]
[-L[f][x]<lon0>/<lat0>[/<slon>]/<slat>/<length>[m|n|k][+l<label>][+j<just>]
    [+p<pen>][+f<fill>][+u]]
[-N<feature>[/<pen>]]
[-O] [-P] [-Q]
[-S<fill>]
[-T[f|m][x]<lon0>/<lat0>/<diameter>[/<info>][:w,e,s,n:][+<gint>[/<mint>]]]
[-U[<just>/<dx>/<dy>/][c|<label>]]
[-V]
[-W[<feature>/][<pen>]]
[-X[a|c|r]<x_shift>[u]]
[-Y[a|c|r]<x_shift>[u]]
[-Z<zlevel>]
[-bo[s|S|d|D[<ncol>]|c[<var1>/...]]]
[-c<ncopies>]
[-m[<flag>]]
-J Selects map proJection. (<scale> in cm/degree, <width> in cm)
     Append h for map height, + for max map dimension, and - for min map
     dimension.  Azimuthal projections set -Rg unless polar aspect or
     -R<...>r is given.

     -Ja|A<lon0>/<lat0>[/<horizon>]/<scale (or radius/lat)|width>
        (Lambert Azimuthal Equal Area)
   -Jb|B<lon0>/<lat0>/<lat1>/<lat2>/<scale|width> (Albers Equal-Area Conic)
     -Jcyl_stere|Cyl_stere/[<lon0>/[<lat0>/]]<lat1>/<lat2>/<scale|width>
        (Cylindrical Stereographic)
   -Jc|C<lon0>/<lat0><scale|width> (Cassini)
   -Jd|D<lon0>/<lat0>/<lat1>/<lat2>/<scale|width> (Equidistant Conic)
     -Je|E<lon0>/<lat0>[/<horizon>]/<scale (or radius/lat)|width>
        (Azimuthal Equidistant)
   -Jf|F<lon0>/<lat0>[/<horizon>]/<scale (or radius/lat)|width>  (Gnomonic)
   -Jg|G<lon0>/<lat0>/<scale (or radius/lat)|width>  (Orthographic)
     -Jg|G[<lon0>/]<lat0>[/<horizon>|/<altitude>/<azimuth>/<tilt>/<twist>/
        <Width>/<Height>]/<scale|width> (General Perspective)
   -Jh|H[<lon0>/]<scale|width> (Hammer-Aitoff)
   -Ji|I[<lon0>/]<scale|width> (Sinusoidal)
   -Jj|J[<lon0>/]<scale|width> (Miller)
   -Jkf|Kf[<lon0>/]<scale|width> (Eckert IV)
```



```
-Jks|Ks[<lon0>/]<scale|width> (Eckert VI)
-Jl|L<lon0>/<lat0>/<lat1>/<lat2>/<scale|width> (Lambert Conformal Conic)
-Jm|M[<lon0>/[<lat0>/]]<scale|width> (Mercator).
-Jn|N[<lon0>/]<scale|width> (Robinson projection)
-Jo|O (Oblique Mercator).  Specify one of three definitions:
   -Jo|O[a]<lon0>/<lat0>/<azimuth>/<scale|width>
   -Jo|O[b]<lon0>/<lat0>/<lon1>/<lat1>/<scale|width>
   -Jo|Oc<lon0>/<lat0>/<lonp>/<latp>/<scale|width>
-Jpoly|Poly/[<lon0>/[<lat0>/]]<scale|width> ((American) Polyconic)
-Jq|Q[<lon0>/[<lat0>/]]<scale|width> (Equidistant Cylindrical)
-Jr|R[<lon0>/]<scale|width> (Winkel Tripel)
  -Js|S<lon0>/<lat0>/[<horizon>/]<scale (or slat/scale or
     radius/lat)|width> (Stereographic)
-Jt|T<lon0>/[<lat0>/]<scale|width> (Transverse Mercator).
-Ju|U<zone>/<scale|width> (UTM)
-Jv|V<lon0>/<scale|width> (van der Grinten)
-Jw|W<lon0>/<scale|width> (Mollweide)
-Jy|Y[<lon0>/[<lat0>/]]<scale|width> (Cylindrical Equal-area)
-Jp|P[a]<scale|width>[/<origin>][r|z] (Polar [azimuth] (theta,radius))
  -Jx|X<x-scale|width>[d|l|p<power>|t|T][/<y-scale|height>[d|l|p<power>|t|T]]
     (Linear, log, and power projections)
(See psbasemap for more details on projection syntax)
```

Each projection requires different parameters. In our example we used `-JA0/20/4.5i`. This selects the projection (Lambert Azimuthal Equal Area) and sets the longitude to zero degrees, the latitude to 20 degrees, and the width of the map to 4.5 inches. Widths can be specified using `c`, `i`, `p`, or `m`, which correspond to centimeters, inches, points (1/72 of an inch), and meters.

You can quickly see two things:

- GMT has a lot of options.
- You might want to read the manual.

This is an important point—sometimes it takes a bit of digging and looking at examples to find the switches, arguments, or parameters needed to accomplish your goal. Reading the manual is a good place to start.

Let's see what happens if we modify the `-J` switch a bit. Let's flip the view around 180 degrees and move it closer to the North Pole.

To do this, use `-JA180/65/4.5i`. Leave all the other parameters the same, and run the `pscoast` command. Our command is now as follows:

```
pscoast -JA180/65/4.5i -Bg30/g15 -Dl -A2000 -G187/142/46 -S109/202/255 \
  -R0/360/-90/90 -P -N1 > simple_180_world.eps
```

Looking at Figure 13.2, you see that indeed we are now looking at the International Date Line, and our view is centered at 65 degrees north latitude.

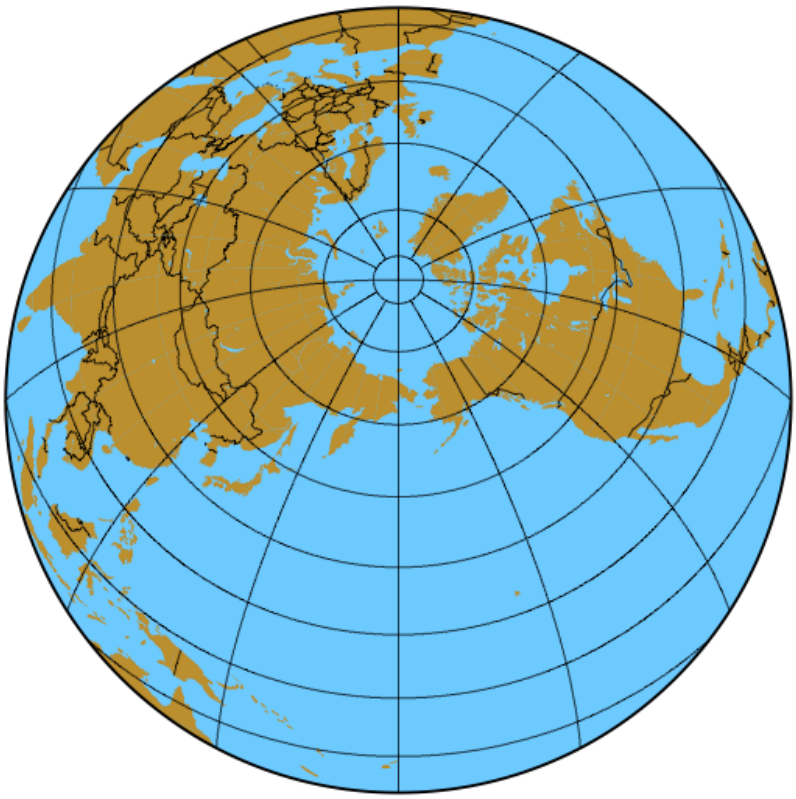

Figure 13.2: Globe centered on 180/65

Let's take a look at the other switches used to create the globe. The `-B` switch defines the intervals for the boundary tick marks. In the globe case, these are the lines of longitude and latitude. The arguments to the `-B` switch indicate a grid line spacing of 30 degrees

in the x (longitude) direction and 15 degrees in the y (latitude) direction. Note how the x and y settings are separated by a forward slash.

The `-D` switch selects the resolution of the dataset used in creating the globe. The available choices are `f`, `h`, `i`, `l`, and `c`, which correspond to full, high, intermediate, low, and crude. Some of these options may not be available to you if you didn't install all the data sets with GMT. For the globe, we used the low resolution data set.

Where Does the GMT Data Come From?

GMT actually provides a number of datasets in five resolutions ranging from fine to crude. When installing GMT, you can choose which resolutions you want to include.
There are also tools available to convert other formats for use with GMT. Use your favorite search engine to find tools applicable to your situation.

To control the display of features, the `-A` switch allows you to specify that features below a certain size not be drawn. In our example, we specified that features with an area greater smaller 2,000 square kilometers should not be displayed.

The fill color used for the countries is specified using the `-G` switch. The color can be specified using RGB notation, a shade of gray, or a pattern. In the globe, we used `187/142/46` to create a light brown color. We could have specified a fill pattern using `-Gp100/30`. This fills the land masses with pattern number 30 at a resolution of 100 dpi. If we want to get the highest possible resolution for the pattern, we can use a resolution of zero. Specifying `-GP` inverts the pattern. GMT has 90 predefined patterns available for your use, and you can find examples of each in the GMT Technical Reference. The same options apply for filling the water areas in GMT, except we use the `-S` switch. There are a number of variations for specifying fill colors, and these are well documented in the GMT manuals and tutorial.

The other major switch used in generating the globe is `-R`. This specifies the extent of the map we want to generate. In the case of the globe, we obviously wanted the entire planet, so we specified an x range of 0 to 360 degrees and a y range of -90 to 90. The range is specified as west/east/south/north. In our next example, we will use `-R` to constrain our map to a smaller area.

The other switch of interest is `-N1`. This tells GMT to draw national boundaries in addition to the coastline. Other arguments to `-N` allow you to draw state boundaries within the Americas and marine boundaries.

The `-P` switch simply sets the page orientation to portrait. Landscape is the default.

A Flat Example

Let's shift gears a bit and look at another example of using GMT, this time for a smaller area. For this example, we'll create a map of Alaska and annotate it. As I said before, the `-R` switch controls the extent of our map. Alaska ranges from about 172 degrees east longitude to 130 degrees west. Using 360 degrees for the entire globe, this translates to a region extending from 172 degrees to 230 degrees.

For the Alaska map, we will use the Albers Equal Area Conic projection. Looking at the syntax for `pscoast` reveals that this requires the use of the `-Jb` switch. In this case, we use the lowercase *b* to indicate that we will specify the size of the map using a scale. First let's look at the code in `gmt_alaska.sh`:

```
pscoast -Jb-154/50/55/65/1:12000000 -R172/230/51/72 -B10g5/5g5 -W1p/0/0/0 \
  -I1/2p/0/192/255 -I2/2p/0/192/255 -I3/1p/0/192/255 -I4/1p/0/192/255 \
  -G220/220/220 -S0/192/255 -L210/54/54/1000 -P -N1/1p/0/0/0 -Dl \
  >gmt_alaska_coast.eps
```

This looks like quite a complex command, but it's really not too bad

once you get past all the numbers and slashes.

Projection

First note we specified the projection using `-Jb-154/50/55/65/1:12000000`. Let's pick that apart a bit to see what's happening. The Albers projection requires the longitude of the central meridian, the latitude of the origin, and the latitude of the two standard parallels. That's what you see specified as `-154/50/55/65`. These are the standard values used for the Albers projection in Alaska. You can actually specify any values you want, but if there is a standard for the area you are mapping, you should use it.

The remaining part of the `-Jb` switch is the size of the output. In this case, we specified it as a scale of 1:12,000,000. This means that one unit on the map represents 12,000,000 units on the ground (in this case meters). If we just wanted output to fit on a page, we could specify `-JB-154/50/55/65/6.0i` to get a 6-inch-wide image.

If you find your map doesn't fit on the "paper" you can change the media size using `gmtset`. See the `gmtdefaults` documentation for available sizes.

Map Extent

To set the map extent, we use the `-R` switch. In this case we already determined that Alaska ranges from 172 to 230 degrees longitude and roughly 51 to 72 degrees north latitude. To create the map covering this area, we use `-R172/230/51/72`.

Grid Lines

In this example, we not only want to draw grid lines but also want to annotate them. This is done using `-B10g5/5g5`. This tells `pscoast` to draw grid lines 5 degrees apart for both latitude and longitude. The annotation is drawn at 10 degree intervals for longitude and 5 degree intervals for latitude. If you look at the documentation for `pscoast`, you will see that the first number after the `-B` is the annotation interval followed by the grid line interval. This notation gives you a lot of flexibility in drawing and labeling grid lines.

Rivers

To make our map more interesting, we'll add rivers to it. GMT comes with several levels of river detail that are specified with the `-I` switch. The levels we are using are as follows:

- Permanent major rivers
- Additional major rivers
- Additional rivers
- Minor rivers

Notice the `-I` options we specified in the `pscoast` command. One is required for each river level we want to include on the map. The first two (major rivers) are drawn using a pen width of `2` (2p), while the third and fourth level are drawn with a width of `1` (1p). We use the same color (0/192/255) for each river. If we wanted to include the intermittent major rivers (fifth level), we would add `-I5/1p/0/192/255` to the `pscoast` command.

Fill Colors

Next we specify the fill colors for the land and water areas using the `-G` and `-S` switches and add the RGB values to specify the color. For land we use a light gray with RGB values of `220/220/220`. For the water `0/192/255` gives us a nice cyan color. Keep in mind that we could also use a pattern or shade for filling land and water areas.

Scale Bar

A scale bar can easily be added to the map using the `-L` switch. Scale bars can be simple or fancy. In this case, we'll just create a simple one and place it in an open area on the map. How do we know it's open? Well, part of the process is running `pscoast` and tweaking the options and then running it again until we get the look we want. To create the scale bar, we need the latitude and longitude of the point where we want to place it. Since scale varies as we move further from the equator, we also specify the latitude at which we want the scale calculated. Lastly, we indicate the length the scale

bar should span. The default is kilometers, but you can append `m` for miles or `n` for nautical miles. Putting it all together, we have `-L210/54/54/1000`, which gives a 1,000 km scale bar calculated at 54 degrees north latitude and originating at 210 degrees longitude and 54 degrees latitude.

The Last Bits

The remainder of the command tells `pscoast` to use portrait mode (`-P`), draw country boundaries in black using a pen width of 1 (`-N1/1p/0/0/0`), and use the low-resolution data (`-Dl`). The low-resolution dataset is the default, but we specified it here so you could see the syntax.

The Result

You can see the result of all these command switches and options in Figure 13.3, on the following page. We have a nice map of Alaska, with grid lines, borders, and degree annotations. The land and water is filled as we specified, and the scale bar is sitting nicely in the Gulf of Alaska.

Overlaying Data

Now that we have used most of the common options, let's look at one more example with `pscoast`. This time we'll generate a world map and overlay point data from a delimited text file. You can take that concept and expand it to create overlays of multiple datasets. In this case, we will overlay the location of the world's volcanoes. First let's look at the command to generate the base map:

```
pscoast -JN0/38 -R-180/180/-90/90 -W -G220/220/220 -S0/192/255 -N1 \
    -P -B30g5:."World Volcanoes": > world_volcanoes.eps
```

About the only thing new in this command is we added a title to the map by appending a colon and a period to the `-B` arguments and then the title string. If you are getting the idea the `-B` switch has lots of permutations, you are correct. Some have called it the most complicated (or confusing) switch in the GMT suite of tools.

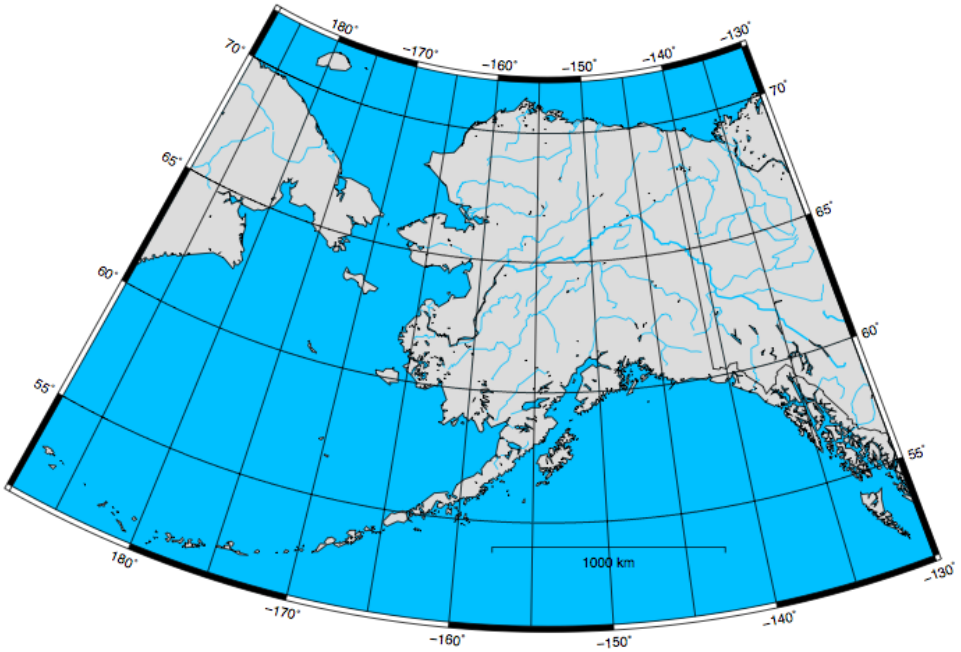

Figure 13.3: Alaska coastline generated with GMT

Fortunately, it's well documented.

Note that we used `-JN` to specify the Robinson projection, centering the map at 0 degrees longitude with a width of 38 centimeters.

GMT and Multiple Commands

When you want to create a more complex map, you will wind up running multiple GMT commands. The trick is to make sure you specify in the first command that more PostScript code will be appended to the output. Without this, you will end up with every command generating a new page in the output. This can be useful, but when you are trying to create an overlay of multiple commands, it's annoying. Create the base map using the `-K` switch, and then in subsequent commands include the `-O` to invoke overlay mode.

This gives us a Robinson base map of the world with grid lines and annotation of the tick marks. To add an overlay of data, we need to add a couple of things. First we need to specify that we want to be able to write to the output file in "append" mode. This is done using the `-K` switch. This allows us to overlay the data created with the next command. Without it, our next command would generate a new page in the output, and we would have to hold it up to the window to see the overlay.

Secondly we need to use the `psxy` command with a text file containing the coordinates of each volcano. Here is a snippet from the text file we will use to create the map:

```
131.6,34.5
-67.62,-23.3
-90.876,14.501
34.52,38.57
-69.05,-18.37
-176.6,51.98
```

These are just x and y values (longitude and latitude) for each point we want to create on the map. To add these, we use `psxy`, making sure to include the `-O` switch so the points overlay the base map. The complete code to generate the base map and the overlay is as follows:

```
pscoast -JN0/38 -R-180/180/-90/90 -K -W -G220/220/220 -S0/192/255 -N1 \
    -P -B30g5:."World Volcanoes": > world_volcanoes.eps
psxy volcanoes_lon_lat.txt -JN -O -R -Sc0.15c -G255/0/0 >> world_volcanoes.eps
```

Note that in the `psxy` command we didn't need to supply any arguments to the projection or extent switches since they were fully specified when the base map was created. We included the overlay switch and a color for the points using `-G`. The `-Sc` switch indicates we want to plot the points using circles. There are several other symbol types you can use including star, bar, diamond, and ellipse. We specified the symbol size as 0.15 cm. You should recognize the rest of the parameters from our previous discussion.

The final result is shown in Figure 13.4, on the next page. This combines the base map with the volcano data. We could have added

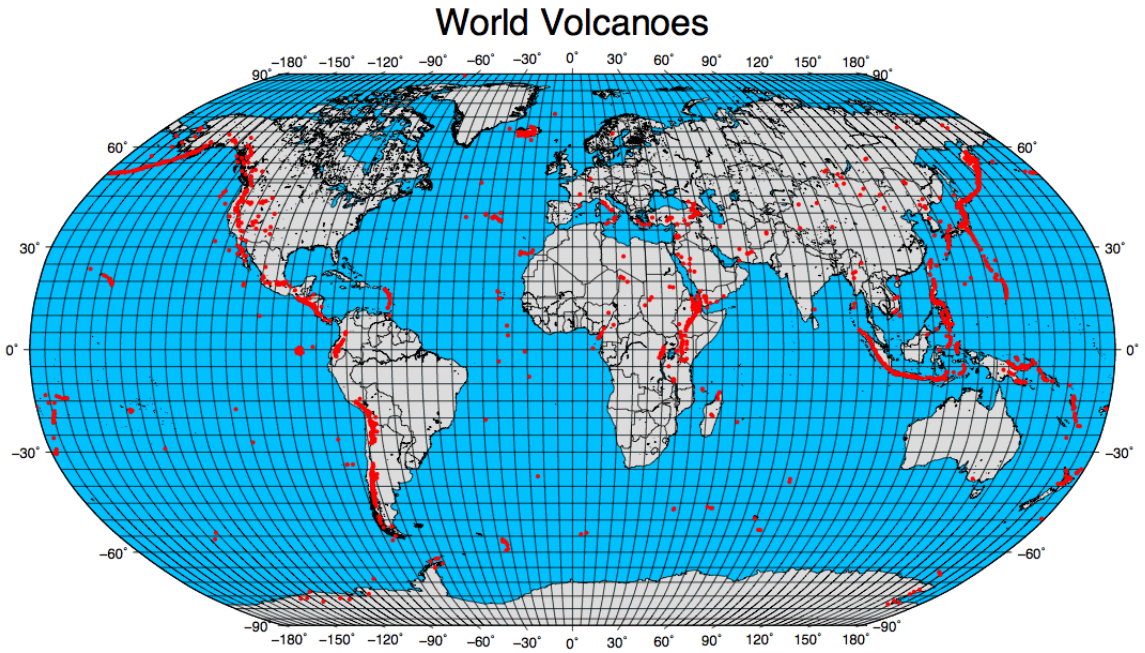

Figure 13.4: Volcanoes plotted on a Robinson projection using GMT

other point data by running another `psxy` command.

As you can see, GMT is a handy tool for creating maps from the command line. We've really only scratched the surface of its capability.

To get the rest of this chapter and the book, see `http://locatepress.com`

