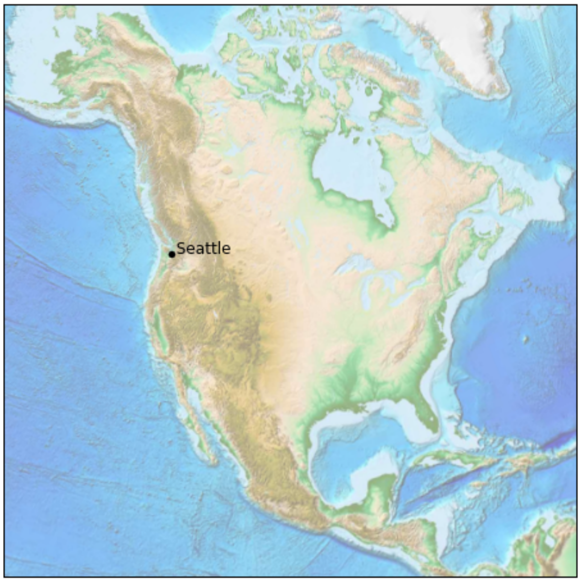

Figure 4-103. Plotting data and labels on the map

This gives you a brief glimpse into the sort of geographic visualizations that are possible with just a few lines of Python. We'll now discuss the features of Basemap in more depth, and provide several examples of visualizing map data. Using these brief examples as building blocks, you should be able to create nearly any map visualization that you desire.

Map Projections

The first thing to decide when you are using maps is which projection to use. You're probably familiar with the fact that it is impossible to project a spherical map, such as that of the Earth, onto a flat surface without somehow distorting it or breaking its continuity. These projections have been developed over the course of human history, and there are a lot of choices! Depending on the intended use of the map projection, there are certain map features (e.g., direction, area, distance, shape, or other considerations) that are useful to maintain.

The Basemap package implements several dozen such projections, all referenced by a short format code. Here we'll briefly demonstrate some of the more common ones.

We'll start by defining a convenience routine to draw our world map along with the longitude and latitude lines:

```
In[4]: from itertools import chain

       def draw_map(m, scale=0.2):
           # draw a shaded-relief image
           m.shadedrelief(scale=scale)

           # lats and longs are returned as a dictionary
           lats = m.drawparallels(np.linspace(-90, 90, 13))
           lons = m.drawmeridians(np.linspace(-180, 180, 13))

           # keys contain the plt.Line2D instances
           lat_lines = chain(*(tup[1][0] for tup in lats.items()))
           lon_lines = chain(*(tup[1][0] for tup in lons.items()))
           all_lines = chain(lat_lines, lon_lines)

           # cycle through these lines and set the desired style
           for line in all_lines:
               line.set(linestyle='-', alpha=0.3, color='w')
```

Cylindrical projections

The simplest of map projections are cylindrical projections, in which lines of constant latitude and longitude are mapped to horizontal and vertical lines, respectively. This type of mapping represents equatorial regions quite well, but results in extreme distortions near the poles. The spacing of latitude lines varies between different cylindrical projections, leading to different conservation properties, and different distortion near the poles. In Figure 4-104, we show an example of the *equidistant cylindrical projection*, which chooses a latitude scaling that preserves distances along meridians. Other cylindrical projections are the Mercator (`projection='merc'`) and the cylindrical equal-area (`projection='cea'`) projections.

```
In[5]: fig = plt.figure(figsize=(8, 6), edgecolor='w')
       m = Basemap(projection='cyl', resolution=None,
                   llcrnrlat=-90, urcrnrlat=90,
                   llcrnrlon=-180, urcrnrlon=180, )
       draw_map(m)
```

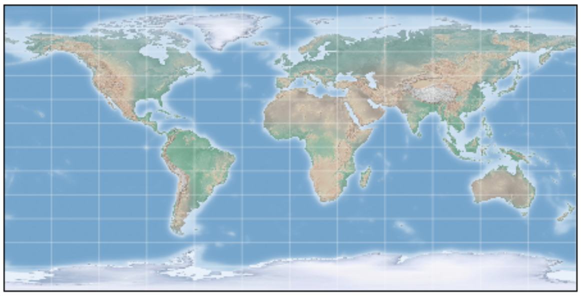

Figure 4-104. Cylindrical equal-area projection

The additional arguments to Basemap for this view specify the latitude (`lat`) and longitude (`lon`) of the lower-left corner (`llcrnr`) and upper-right corner (`urcrnr`) for the desired map, in units of degrees.

Pseudo-cylindrical projections

Pseudo-cylindrical projections relax the requirement that meridians (lines of constant longitude) remain vertical; this can give better properties near the poles of the projection. The Mollweide projection (`projection='moll'`) is one common example of this, in which all meridians are elliptical arcs (Figure 4-105). It is constructed so as to preserve area across the map: though there are distortions near the poles, the area of small patches reflects the true area. Other pseudo-cylindrical projections are the sinusoidal (`projection='sinu'`) and Robinson (`projection='robin'`) projections.

```
In[6]: fig = plt.figure(figsize=(8, 6), edgecolor='w')
       m = Basemap(projection='moll', resolution=None,
                   lat_0=0, lon_0=0)
       draw_map(m)
```

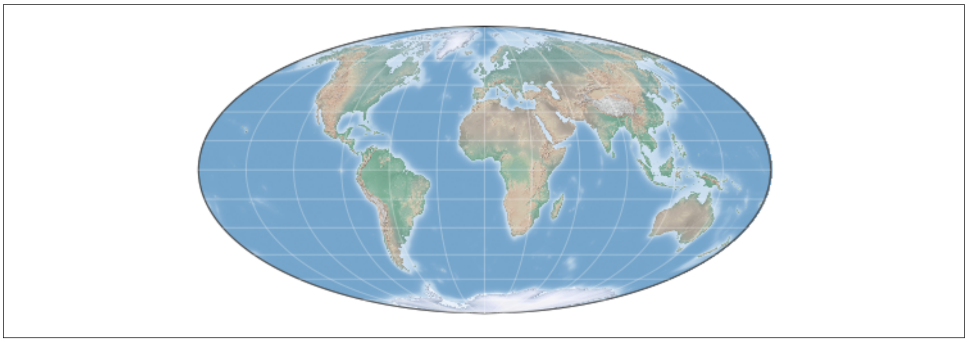

Figure 4-105. The Molleweide projection

The extra arguments to `Basemap` here refer to the central latitude (`lat_0`) and longitude (`lon_0`) for the desired map.

Perspective projections

Perspective projections are constructed using a particular choice of perspective point, similar to if you photographed the Earth from a particular point in space (a point which, for some projections, technically lies within the Earth!). One common example is the orthographic projection (`projection='ortho'`), which shows one side of the globe as seen from a viewer at a very long distance. Thus, it can show only half the globe at a time. Other perspective-based projections include the gnomonic projection (`projection='gnom'`) and stereographic projection (`projection='stere'`). These are often the most useful for showing small portions of the map.

Here is an example of the orthographic projection (Figure 4-106):

```
In[7]: fig = plt.figure(figsize=(8, 8))
       m = Basemap(projection='ortho', resolution=None,
                   lat_0=50, lon_0=0)
       draw_map(m);
```

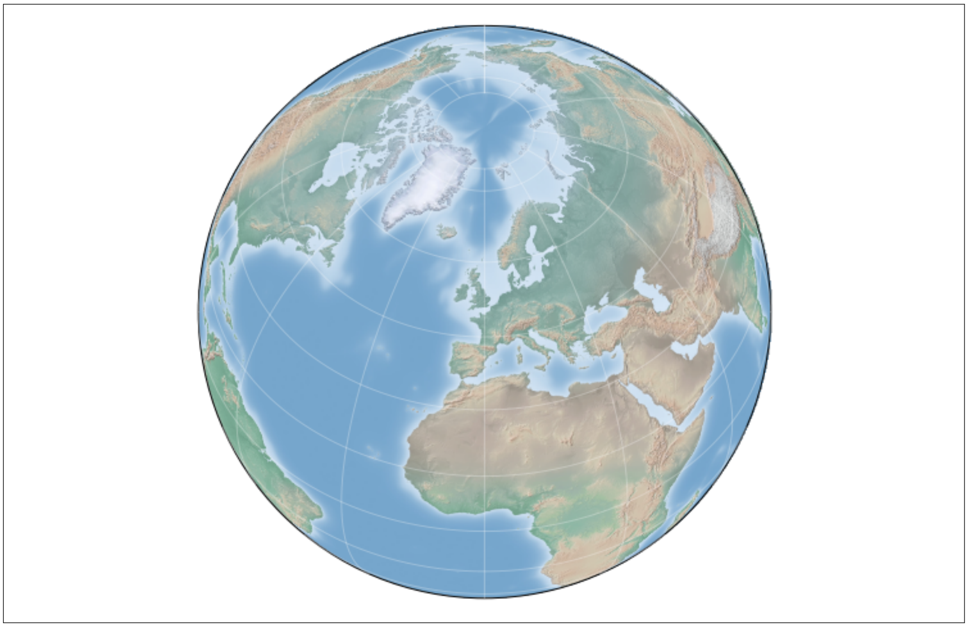

Figure 4-106. The orthographic projection

Conic projections

A conic projection projects the map onto a single cone, which is then unrolled. This can lead to very good local properties, but regions far from the focus point of the cone may become very distorted. One example of this is the Lambert conformal conic projection (`projection='lcc'`), which we saw earlier in the map of North America. It projects the map onto a cone arranged in such a way that two standard parallels (specified in `Basemap` by `lat_1` and `lat_2`) have well-represented distances, with scale decreasing between them and increasing outside of them. Other useful conic projections are the equidistant conic (`projection='eqdc'`) and the Albers equal-area (`projection='aea'`) projection (Figure 4-107). Conic projections, like perspective projections, tend to be good choices for representing small to medium patches of the globe.

```
In[8]: fig = plt.figure(figsize=(8, 8))
       m = Basemap(projection='lcc', resolution=None,
                   lon_0=0, lat_0=50, lat_1=45, lat_2=55,
                   width=1.6E7, height=1.2E7)
       draw_map(m)
```

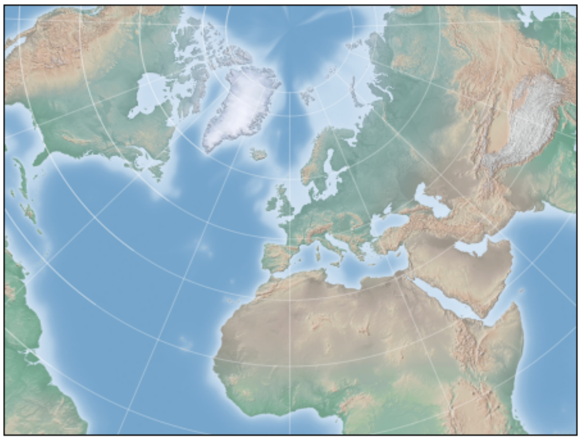

Figure 4-107. The Albers equal-area projection

Other projections

If you're going to do much with map-based visualizations, I encourage you to read up on other available projections, along with their properties, advantages, and disadvantages. Most likely, they are available in the Basemap package. If you dig deep enough into this topic, you'll find an incredible subculture of geo-viz geeks who will be ready to argue fervently in support of their favorite projection for any given application!

Drawing a Map Background

Earlier we saw the `bluemarble()` and `shadedrelief()` methods for projecting global images on the map, as well as the `drawparallels()` and `drawmeridians()` methods for drawing lines of constant latitude and longitude. The Basemap package contains a range of useful functions for drawing borders of physical features like continents, oceans, lakes, and rivers, as well as political boundaries such as countries and US states and counties. The following are some of the available drawing functions that you may wish to explore using IPython's help features:

- Physical boundaries and bodies of water

  `drawcoastlines()`
  : Draw continental coast lines

  `drawlsmask()`
  : Draw a mask between the land and sea, for use with projecting images on one or the other