parameter for a given view; the best route is to start with a fast, low-resolution plot and increase the resolution as needed.

Plotting Data on Maps

Perhaps the most useful piece of the Basemap toolkit is the ability to over-plot a variety of data onto a map background. For simple plotting and text, any `plt` function works on the map; you can use the `Basemap` instance to project latitude and longitude coordinates to `(x, y)` coordinates for plotting with `plt`, as we saw earlier in the Seattle example.

In addition to this, there are many map-specific functions available as methods of the `Basemap` instance. These work very similarly to their standard Matplotlib counterparts, but have an additional Boolean argument `latlon`, which if set to `True` allows you to pass raw latitudes and longitudes to the method, rather than projected `(x, y)` coordinates.

Some of these map-specific methods are:

`contour()`/`contourf()`
: Draw contour lines or filled contours

`imshow()`
: Draw an image

`pcolor()`/`pcolormesh()`
: Draw a pseudocolor plot for irregular/regular meshes

`plot()`
: Draw lines and/or markers

`scatter()`
: Draw points with markers

`quiver()`
: Draw vectors

`barbs()`
: Draw wind barbs

`drawgreatcircle()`
: Draw a great circle

We'll see examples of a few of these as we continue. For more information on these functions, including several example plots, see the online Basemap documentation.

Example: California Cities

Recall that in "Customizing Plot Legends" on page 249, we demonstrated the use of size and color in a scatter plot to convey information about the location, size, and population of California cities. Here, we'll create this plot again, but using Basemap to put the data in context.

We start with loading the data, as we did before:

```
In[10]: import pandas as pd
        cities = pd.read_csv('data/california_cities.csv')

        # Extract the data we're interested in
        lat = cities['latd'].values
        lon = cities['longd'].values
        population = cities['population_total'].values
        area = cities['area_total_km2'].values
```

Next, we set up the map projection, scatter the data, and then create a colorbar and legend (Figure 4-109):

```
In[11]: # 1. Draw the map background
        fig = plt.figure(figsize=(8, 8))
        m = Basemap(projection='lcc', resolution='h',
                    lat_0=37.5, lon_0=-119,
                    width=1E6, height=1.2E6)
        m.shadedrelief()
        m.drawcoastlines(color='gray')
        m.drawcountries(color='gray')
        m.drawstates(color='gray')

        # 2. scatter city data, with color reflecting population
        # and size reflecting area
        m.scatter(lon, lat, latlon=True,
                  c=np.log10(population), s=area,
                  cmap='Reds', alpha=0.5)

        # 3. create colorbar and legend
        plt.colorbar(label=r'$\log_{10}({\rm population})$')
        plt.clim(3, 7)

        # make legend with dummy points
        for a in [100, 300, 500]:
            plt.scatter([], [], c='k', alpha=0.5, s=a,
                        label=str(a) + ' km$^2$')
        plt.legend(scatterpoints=1, frameon=False,
                   labelspacing=1, loc='lower left');
```