Density and Contour Plots

Sometimes it is useful to display three-dimensional data in two dimensions using contours or color-coded regions. There are three Matplotlib functions that can be helpful for this task: `plt.contour` for contour plots, `plt.contourf` for filled contour plots, and `plt.imshow` for showing images. This section looks at several examples of using these. We'll start by setting up the notebook for plotting and importing the functions we will use:

```
In[1]: %matplotlib inline
       import matplotlib.pyplot as plt
       plt.style.use('seaborn-white')
       import numpy as np
```

Visualizing a Three-Dimensional Function

We'll start by demonstrating a contour plot using a function $z = f(x, y)$, using the following particular choice for f (we've seen this before in "Computation on Arrays: Broadcasting" on page 63, when we used it as a motivating example for array broadcasting):

```
In[2]: def f(x, y):
           return np.sin(x) ** 10 + np.cos(10 + y * x) * np.cos(x)
```

A contour plot can be created with the `plt.contour` function. It takes three arguments: a grid of x values, a grid of y values, and a grid of z values. The x and y values represent positions on the plot, and the z values will be represented by the contour levels. Perhaps the most straightforward way to prepare such data is to use the `np.meshgrid` function, which builds two-dimensional grids from one-dimensional arrays:

```
In[3]: x = np.linspace(0, 5, 50)
       y = np.linspace(0, 5, 40)

       X, Y = np.meshgrid(x, y)
       Z = f(X, Y)
```

Now let's look at this with a standard line-only contour plot (Figure 4-30):

```
In[4]: plt.contour(X, Y, Z, colors='black');
```