**Transcript**

**1. Plotting time-series with different variables**

00:00 - 00:14

To relate two time-series that coincide in terms of their times, but record the values of different variables, we might want to plot them on the same Axes.

**2. Plotting two time-series together**

00:14 - 00:35

For example, consider the climate-underscore-change DataFrame that we've seen previously. This DataFrame contains two variables measured every month from 1958 until 2016: levels of carbon dioxide and relative temperatures.

**3. Plotting two time-series together**

00:35 - 01:10

As before, we can create a Figure and Axes and add the data from one variable to the plot. And we can add the data from the other variable to the plot. We also add axis labels and show the plot. But this doesn't look right. The line for carbon dioxide has shifted upwards, and the line for relative temperatures looks completely flat. The problem is that the scales for these two measurements are different.

**4. Using twin axes**

01:10 - 02:14

You've already seen how you could plot these time-series in separate sub-plots. Here, we're going to plot them in the same sub-plot, using two different y-axis scales. Again, we start by adding the first variable to our Axes. Then, we use the twinx method to create a twin of this Axes. This means that the two Axes share the same x-axis, but the y-axes are separate. We add the other variable to this second Axes object and show the figure. There is one y-axis scale on the left, for the carbon dioxide variable, and another y-axis scale to the right for the temperature variable. Now you can see the fluctuations in temperature more clearly. But this is still not quite right. The two lines have the same color. Let's take care of that.

**5. Separating variables by color**

02:14 - 02:53

To separate the variables, we'll encode each one with a different color. We add color to the first variable, using the color key-word argument in the call to the plot function. We also set the color in our call to the set-underscore-ylabel function. We repeat this in our calls to plot and set-underscore-ylabel from the twin Axes object. In the resulting figure, each variable has its own color and the y-axis labels clearly tell us which scale belongs to which variable.

**6. Coloring the ticks**

02:53 - 03:46

We can make encoding by color even more distinct by setting not only the color of the y-axis labels but also the y-axis ticks and the y-axis tick labels. This is done by adding a call to the tick-underscore-params method. This method takes either y or x as its first argument, pointing to the fact that we are modifying the parameters of the y-axis ticks and tick labels. To change their color, we use the colors key-word argument, setting it to blue. Similarly, we call the tick-underscore-params method from the twin Axes object, setting the colors for these ticks to red.

**7. Coloring the ticks**

03:46 - 04:04

Coloring both the axis label and ticks makes it clear which scale to use with which variable. This seems like a useful pattern. Before we move on, let's implement this as a function that we can reuse.

**8. A function that plots time-series**

04:04 - 04:42

We use the def key-word to indicate that we are defining a function called plot-underscore-timeseries. This function takes as arguments an Axes object, x and y variables to plot, a color to associate with this variable, as well as x-axis and y-axis labels. The function calls the methods of the Axes object that we have seen before: plot, set-underscore-xlabel, set-underscore-ylabel, and tick-underscore-params.

**9. Using our function**

04:42 - 04:49

Using our function, we don't have to repeat these calls, and the code is simpler.

**10. Create your own function!**

04:49 - 04:58

In the exercises, you will gradually implement your own function from scratch.