Octave Version Tutorial

Kenny Vo

1. Install Octave: You can download and install Octave, a free and open-source program for numerical computation, on your computer.
2. Data must first be loaded in order to build plots, smoothers, and salters. The load command, which takes data from a file and creates a matrix variable in memory, can be used to load data into Octave. Or you can create your own data.

* Using the randn function, which creates random numbers from a typical normal distribution, we will create some random data for this example.

x = linspace(0, 10, 100);

y = randn(size(x));

1. Plotter creation: In Octave, you can use the plot function to build a plotter. The x- and y-coordinates of the points you want to plot are the second and third inputs that the plot function accepts.

* Now that we have our data, we can make a plotter using Octave's built-in plotting methods. We can use the plot function to build the following plot.

plot(x, y);

* With the x-axis ranging from 0 to 10 and the y-axis ranging from the least value of y to the maximum value of y, this will produce a plot of our data.

1. Build a salter: The Octave salt function can be used to build a salter. The data you want to salt and the salt factor or frequency are the two or more arguments that the salt function accepts.

* When we introduce some noise to our data, we can produce a salter. Using the rand function, we can accomplish this by multiplying y by a modest random number between 0 and 1.

salty\_y = y .\* rand(size(y));

* The two data sets can then be plotted side by side to compare the results:

plot(x, y, 'b', x, salty\_y, 'r');

* The original data will appear in blue on the plot, and the salted data will appear in red.

1. Create a smoother: In Octave, use the smooth function to build a smoother. The smooth function accepts two or more inputs, including the window size or smoothing factor and the data you wish to smooth.

smoother\_y = smooth(x, y);

* The original data and the smoothed data can then be plotted side by side to compare.

plot(x, y, 'b', x, smoother\_y, 'r');

* The smoothed data will appear in red on the figure, with the original data appearing in blue.

I learned that there are numerous uses for these three strategies, all of which can be helpful for displaying and interpreting data.