PROFESSOR: Hello. I'm Ian Hutchinson, and the purpose of this short video is to give you illustration of how to use Octave. I'm certainly not an expert at [? Octave, ?] which is an open source equivalent of MATLAB, but that's part of the demonstration. Here's someone who's not an expert who can use this routine for carrying out the kinds of exercises that we want to do. So I'm going to delete my picture from the screen now, and just show you the rest of the screen.

Here I am in a directory. I'm going to start a new file. It's going to be called fitting. Let's say, fitting.m, is the correct extension for a MATLAB or Octave file. And I've opened Emax, and I'm ready to get started.

Actually, I'm going to run Octave simultaneously in the adjacent window. I run Octave here. So now I'm in a situation where I, essentially, have my own IDE, integrated development environment. It's not quite the same as you would have if you're running MATLAB itself, but it's pretty similar.

What I'm going to do now is develop this fitting program. I'm going to start with some parameters. Let's, first of all, define the number of points I'm going to fit. Let's make it a small number, so that it's kind of manageable. And now, let's generate some x and y values.

I can set x to be equal to an array. It might be an array with six points. And I'm going to make it be run from 1 to endpoints. And so I do that by saying x is equal to 1 colon endpoints like this.

I can save this file-- it's called fitting-- and then I can run it in Octave by just typing fitting. If I do that, what you see happens is, it tells me that endpoints is 6. Tells me that x is 1, 2, 3, 4, 5, and 6.

Actually, I don't want it to run from 1 to 6. I want it to run from, essentially, a small value up to 1. So let's divide it by end points. And I can save that file again, run it again, and now, of course, my screen shows that it runs from 1/6 up to 1.

This is, of course, a row vector. It's preferable that I have a column vector. I can transform a row vector into a column vector by taking its transpose. In MATLAB or Octave, transpose is prime. So I can say, transpose the numbers 1 through 6 into a column vector and then divide by endpoints. And if I do that and save the file, then I see I've turned it into a column vector.

Let me now make a y variable. y, I'm going to make equal to x, let's say. That would just give me a straight line, if I made it nothing but x. Then I'll add to it some small fraction-- let's say, 1/10th-- not 0.1 times the sine of, let us say, 10 x.

I can save that file. I can run it again. And it tells me, here's x and here's y. That's given me a start. Actually, it's a bit inconvenient to see it in quite those terms. So let's actually try to plot it.

So I can say plot x y. Save that and run it. And lo and behold, it brings up a plot. The labeling is rather too small, but that's kind of built into the rather poor graphics in Octave and MATLAB, for that matter. That can be fixed in ways that I'll go into later, but we won't worry about it now.

This is a line plot, which is a little bit annoying. So let's change that plot. I'll put the figure over here. I'll change that plot by telling it that it's going to have points, and those points are going to have the form of a plus sign.

So let's save that and run it again. And Lo and behold, it's changed it to a plot with points. Again, the markers are rather unpleasantly small. I can actually change that if I want by telling it a string marker size. Making that size larger than it would be-- let's say, 20-- and running it again. And lo and behold, the points are bigger.

So we've started, and we've made, some data. And we've used parameters. And in our next video, we'll move on to finding out how to proceed on that basis.