These exercises are designed to help you get started in using computers to solve physics problems. They use a language called “Python”, which is an extremely powerful programming language useful for just about anything. This is not a programming course, though; to really learn Python will take a different course. This is just a set of exercises that will help you get started in using Python.

The exercises build on each other, and should mostly be done in order.

**Exercise 1: Installation and Test**

Python is an open-source language that is extremely powerful yet still easy to use. It’s also free, which is generally a good thing. Python by itself is a fairly basic computer language, but there are extra “libraries” that extend its capabilities to allow interesting things like plotting equations and data and 3D objects. The libraries you will need for these exercises are Numpy, Scipy, and MatPlotLib, which are often known collectively as “pylab”.

If you are already using Python, you may have these packages already; or you may need to do a more customized installation. If not, the simplest way to install what you need is to install “Anaconda”. Anaconda is a full Python environment with all the tools you will need (and more!) It is available for free download from [continuum](http://www.continuum.io/downloads). Download the appropriate version for your computer system, and follow the instructions to install it.

Once Anaconda is installed, launch it and choose the “spyder-app” Scientific Python Development Environment. (Feel free to experiment with the others also, “qtconsole” is also a good choice.) A new window should appear, with several sub-windows. On the left will be a text editor, where you can write Python programs. On the bottom right will be a console window, in which you can type Python commands directly. In the console window, type the following commands, one per line:

 from pylab import \*

 t = linspace(0, 10\*pi, 1000)

 z = sin(t)\*exp(-t/10)

 plot(t,z,'b-')

 title("Cool! It works!")

 xlabel("time")

 ylabel("displacement")

 show()

After entering the last of these lines, a graph like the one below will appear in the console window.



Here’s an explanation of what each of those lines do, for future reference.

 from pylab import \*

This command tells Python to load up a big batch of mathematical goodies, and should be the first command you enter just about every time you use Python for graphing or numeric work.

 t = linspace(0, 10\*pi, 1000)

tt is now an array of 1000 values, linearly spaced from 0 to 10ππ.

 z = sin(t)\*exp(-t/10)

zz is calculated with the formula given. Since tt is an array, zz is also an array; one value for each value of tt.

 plot(t,z,'b-')

Make a plot with the values contained in tt on the horizontal axis and the values contained in zz on the vertical axis. Show the data with a blue line (b−b−).

 title("Cool! It works!")

 xlabel("time")

 ylabel("displacement")

Make a title, a label for the horizontal axis, and a label for the vertical axis. (These are optional plot features, if you skip them the plot just won’t have these useful details.)

 show()

Show the resulting plot.

**ASSIGNMENT**

Make a change to the plot, print it out, and turn it in with a note calling attention to your change.