8 Plotting

We'll now spend a few minutes discussing how to make plots in python. There are many ways to achieve similar results with different packages. In this course do everything with matplotlib as it works well from within python and it is easy to use.

Before we get started, we need to talk about moving files between different computers as we want to download the pdf file we generate from the linux machine to your own computer. There are a couple of ways to do that, depending on the operating system and terminal program you decided to use. The easiest way to copy files to windows is by using the MobaXterm application. On the left hand side, you can see tree of the remote directory structure. Simply go to the subdirectory where your file is in and drag it to any folder on your windows machine to copy the file.

If you are using MacOSX, Linux or another unix-like operating system, you can use the command line on your computer to copy files with the scp program. The basic syntax is as follows

```
scp username@rein001.utsc.utoronto.ca:~/PSCB57/
   assignment_06/plot.pdf .
```

where you would of course replace `username` with your own username and you would adjust the path to the file according to what you want to transfer. Don't forget the dot at the end. This tells the scp program to copy the file to the current directory you're in on your local machine. When you execute the scp program it will ask for your password.

Now, let us start to plot some data. Note that for the examples that follow, we will simply use some mock data. Matplotlib is a python module. It is basically a big chunk of code that we can use to do a certain task (plotting) without having to write everything ourselves. There are many modules for python. In fact, that is one reason why python is so popular. In this course we won't use many modules, because the main goal of the course is to teach you numerical algorithms. However, if you want to solve a problem as quickly as possible, there is a good chance that you'll be using a lot of modules. Rarely, you'll have to start completely from scratch.

Back to plotting. First, we import the matplotlib module. This is done with the import statement. Then, we specify the output format that we want. There are many options (png, jpg, LaTeX). We will use the pdf format. Then, we import yet another part of matplotlib, pyplot. This module includes the functions that allow us to actually plot something. In the first example we simply plot a curve whose joining three points. The points are given as two arrays, x and y. Both are of length 3. The complete script is only five lines long and looks like this:

```
import matplotlib
matplotlib.use('pdf')
import matplotlib.pyplot as plt
plt.plot([0.1,1,1.9],[2.9,4.4,4.9])
plt.savefig("plot.pdf")
```

Code 44: Matplotlib example 1.

The above script produces a pdf file that looks as follow. Note that there are many properties of the plot which are now set to default, but which you might want to change. For example, the colour of the line, the size of the plot, the labels on the axis, etc.

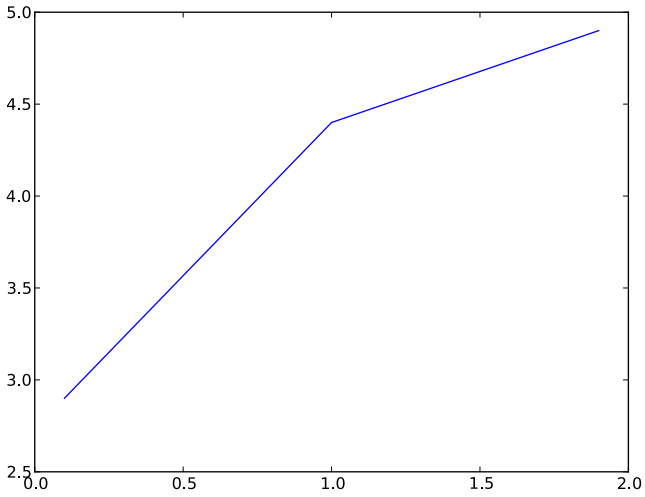

Figure 8: Plot from matplotlib example 1.

Our first example joined the data-points with a line. We can also plot only the actual data-points without joining them by adding an additional option to the plot command.

```
import matplotlib
matplotlib.use('pdf')
import matplotlib.pyplot as plt
plt.plot([0.1,1,1.9],[2.9,4.4,4.9],'ro')
plt.savefig("plot.pdf")
```

Code 45: Matplotlib example 2.

The output now looks like this:

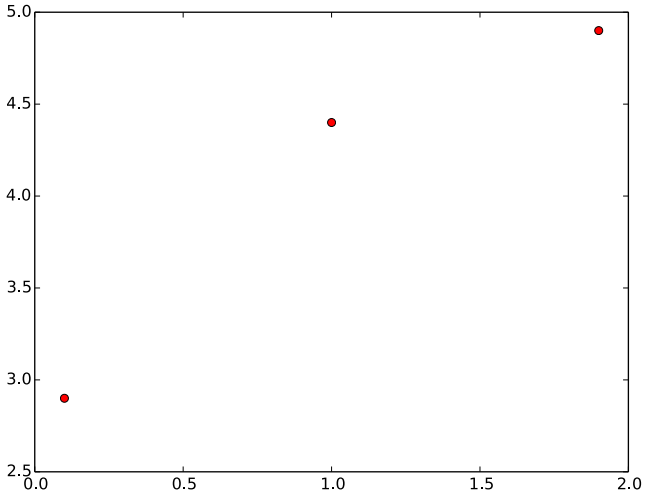

Figure 9: Plot from matplotlib example 2.

So far we only plotted data-points. If we want to plot a function, the easiest way is to create a set of data-points ourselves. How many data-points we use depends on how accurate we want to plot to be. Have a look at the following code, which plots the sine function.

```
import matplotlib
matplotlib.use('pdf')
import matplotlib.pyplot as plt
x = []
y = []
import math
N=1000
for i in xrange(N):
    xp = 2.*math.pi * i/N
    x.append( xp )
    y.append( math.sin(xp) )
plt.plot(x,y)
plt.savefig("plot.pdf")
```

Code 46: Matplotlib example 3.

The output of the above code produces a sine curve sampled at $N = 1000$ data-points. It looks like this:

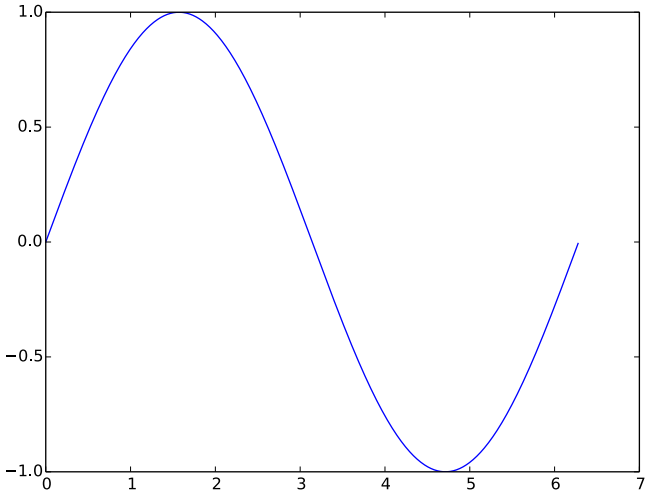

Figure 10: Plot from matplotlin example 3.

This is all that we'll need for plotting in this course. Google is your friend when you want to customize your plots. There are an almost infinite number of options. And if there isn't the right option available for your task, there is likely another package out there that does it for you.

End of lecture 6.