How to do basic plotting in Python

You will need:

- *numpy* and *matplotlib* (included in Anaconda).

Generating Values

Two simple ways to generate a list of x-values are:

1. numpy.arange(start,stop,step,dtype): returns evenly spaced values in a given interval.
    a. Example: x = numpy.arange(5) will give a list x = [0 1 2 3 4]
    b. Example: x = numpy.arange(2,5) will give a list x = [2 3 4]
    c. Example: x = numpy.arange(2,5,0.5) will give a list x = [2. 2.5 3. 3.5 4. 4.5]

2. numpy.linspace(start,stop,num,endpoint,retstep,dtype): returns evenly spaced values over the interval [start, stop] (note, inclusivity unless specified)
    a. Example: x = numpy.linspace(2,5,3) will give a list x = [2. 3.5 5.]
    b. Example: x = numpy.linspace(2,5,3,False) will give a list x = [2. 3. 4.]

To generate random values, you can use a variety of random sampling methods from the random sub-package.

1. rand(d0,d1,…,dn) perform random sampling of dimension i from Uniform(0,1).
    a. Example: numpy.random.rand(5) will generate a list of 5 values between 0 and 1
2. randn(d0,d1,…,dn) perform random sampling of dimension i from the Standard Normal

You can draw samples from a variety of distributions. See the random documentation (http://docs.scipy.org/doc/numpy/reference/routines.random.html).

Note: If you mix the functions in the *math* and *numpy* library, you might sometime get an error that reads:

```
TypeError: only length-1 arrays can be converted to Python scalars
```

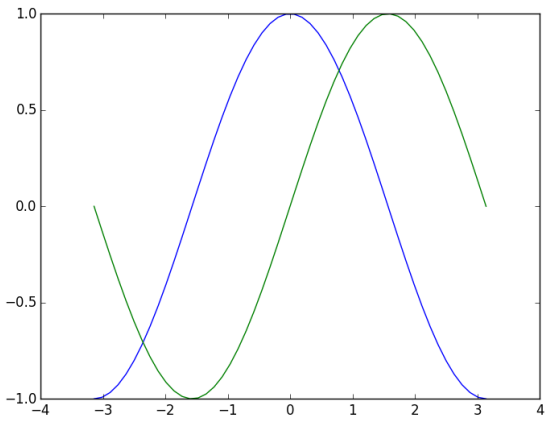
This is because *numpy* functions expect an array input whereas *math* expects scalar inputs. Both packages have common function names so be careful which you use.

Plotting

matplotlib is a plotting library with procedures to generate a variety of plots.

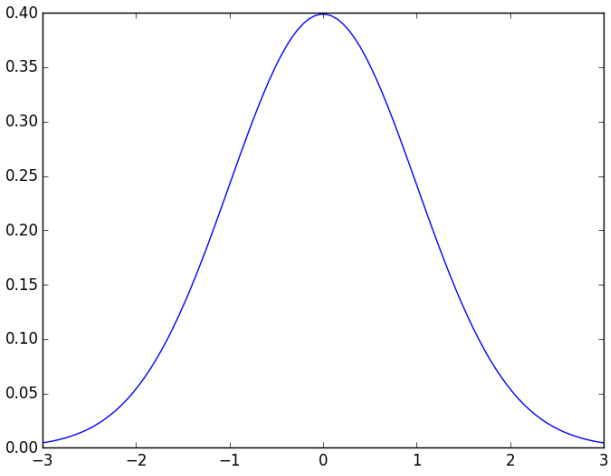
Example 1: Plotting sine and cosine function

```
# Importing the packages
import numpy as np
import matplotlib.pyplot as plt

# Generate values of X between
-pi and pi
X = np.linspace(-np.pi, np.pi)

# Compute the y-values
C = np.cos(X)
S = np.sin(X)

# Plot the points
plt.plot(X,C)
plt.plot(X,S)

# Display the plot
plt.show()
```



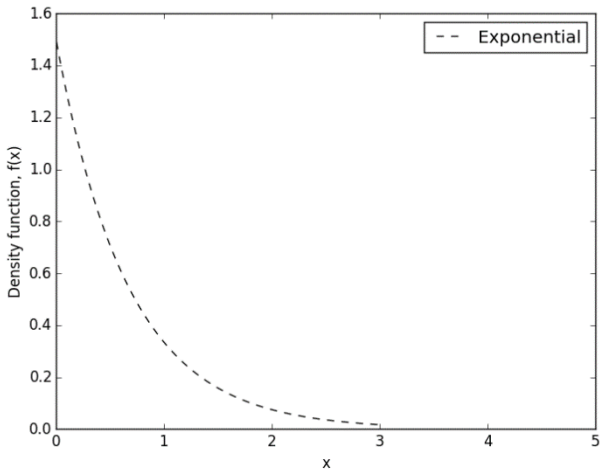
Example 2: Plotting the Normal distribution

```
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.mlab as mlab
import math

# Defining the parameters of
the Normal distribution
mean = 0
variance = 1
sigma = math.sqrt(variance)

# Generating the values of x
and y
x = np.linspace(-3,3,100)
y = mlab.normpdf(x,mean,sigma)

plt.plot(x,y)

plt.show()
```



Example 3: Plotting the Exponential distribution with labels and legend

```
import matplotlib.pyplot as plt
import numpy as np

# Initializing x, y and the
parameter for the exponential
lmbda = 1.5
x = np.arange(-3,3,0.001)
y = np.zeros(len(x))

# Compute the values of y
for j in range(0,len(x)):
    if (x[j] > 0):
        y[j] = 1.5 * np.exp(-
1.5*x[j])
    else:
        y[j] = 0

# Plot
plt.plot(x,y, 'k--',
label='Exponential')

# Label the axes and create a legend
plt.xlabel('x')
plt.ylabel('Density function, f(x)')
plt.xlim(0,5)
plt.legend()

# Display plot
plt.show()
```



Example 4: Plotting a bar graph

```
import matplotlib.pyplot as plt
import numpy as np

# Dictionary containing relevant
values
meanScore = {'1': 84.2, '2': 86.8,
'3': 75.1, '4': 92, '5': 88.7 }

# Plotting the dictionary
plt.bar(range(len(meanScore)),
meanScore.values(), align='center',
color='r')
plt.xticks(range(len(meanScore)),
list(meanScore.keys()))

# Creating the plot labels
plt.title("Average HW Score")
plt.ylabel('Score out of 100')
plt.xlabel('Homework')

# Display plot
plt.show()
```