Exercise1 Matplotlib

April 15, 2018

1 Matplotlib

Documentation: http://matplotlib.org/

Matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

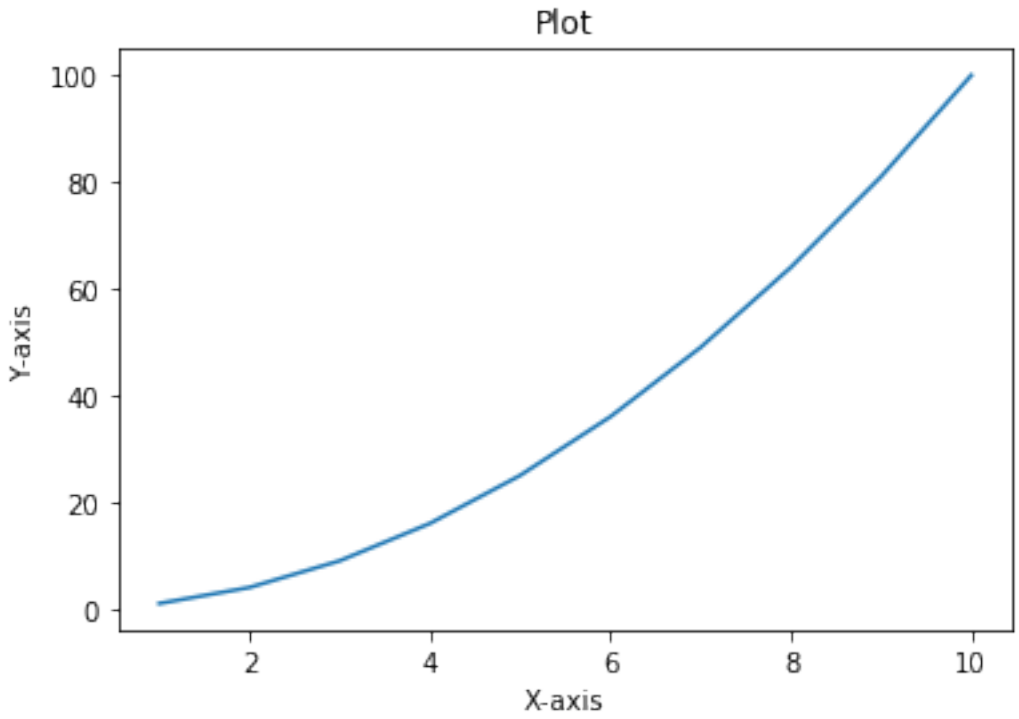
You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc.

```
In [1]: # needed to display the graphs
        %matplotlib inline
        import matplotlib.pyplot as plt
        import pylab
        from pylab import *

In [2]: import pandas as pd
```

1.1 Task 1

- Create a plot $y = x^2$ for $x \in [1 : 10]$
- Add Title and Axes (Replicate the plot below)

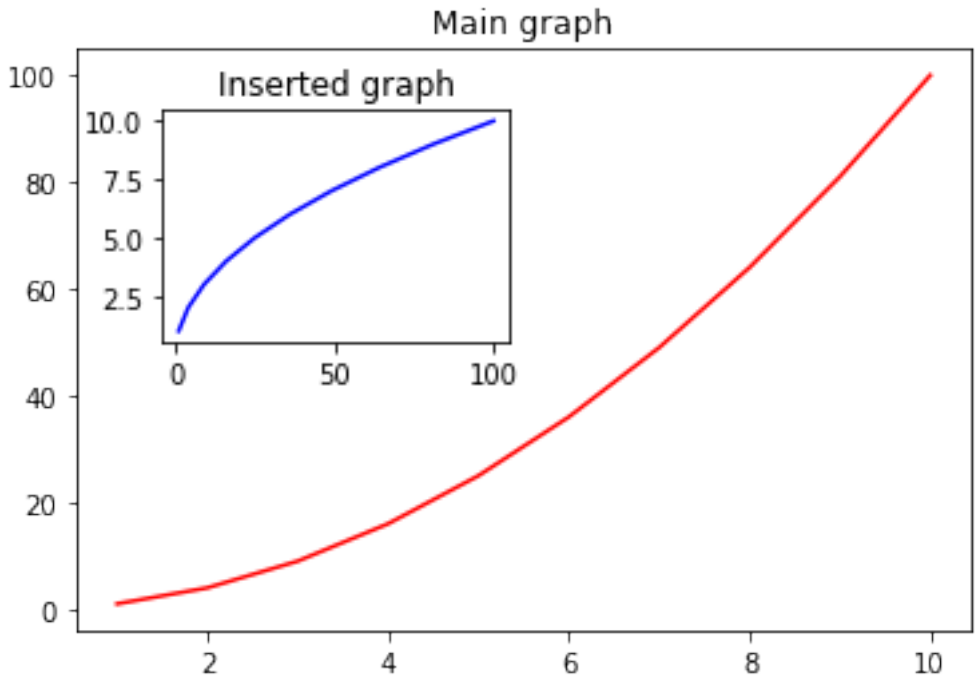
```
In [3]: x = range(1, 11)
        y = list()
        for i in x:
            y.append(i**2)
        plt.plot(x, y)
        plt.xlabel("X-axis")
        plt.ylabel("Y-axis")
        plt.title("Plot")
        plt.show()
```



1.2 Task 2

Create two plots: 'main' and 'insert' and place them such that - The 'insert' plot are included into the 'main' plot - The 'insert' is next to the 'main' plot (Replicate the plots below)

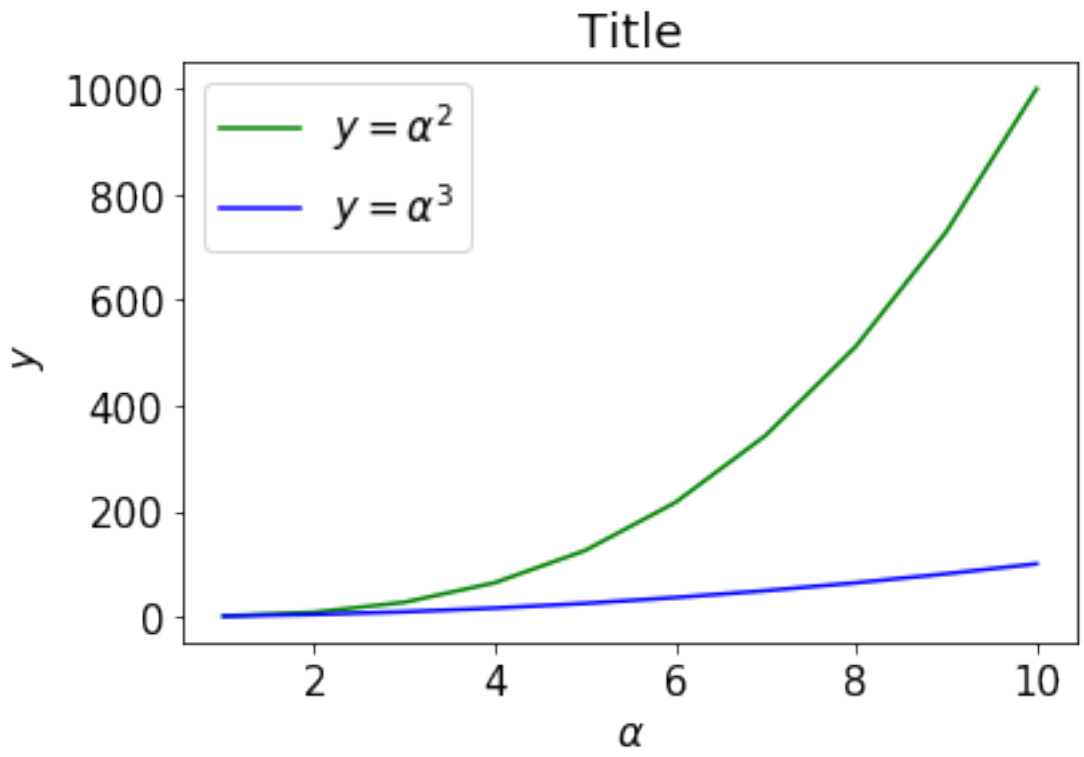
```
In [4]: ax1 = plt.plot(x,y,'r')  # standard axes
        plt.title("Main graph")
        ax2 = plt.axes([0.20, 0.5, 0.3, 0.3])
        ax2.plot(y,x,'b')
        plt.title("Inserted graph")

        plt.show()
```



1.3 Task 3

Create a plot with a legend and latex symbols

```
In [5]: matplotlib.rc('font', size=15)
        y1 = list()
        for i in x:
            y1.append(i**3)
        ax = plt.gca()
        ax.plot(x, y1, 'g', label=r'$y = \alpha^2$')
        ax.plot(x, y, 'b', label=r'$y = \alpha^3$')

        ax.legend(loc=2)
        plt.title("Title")
        plt.xlabel(r'$\alpha$')
        plt.ylabel(r'$y$')
        plt.show()
```



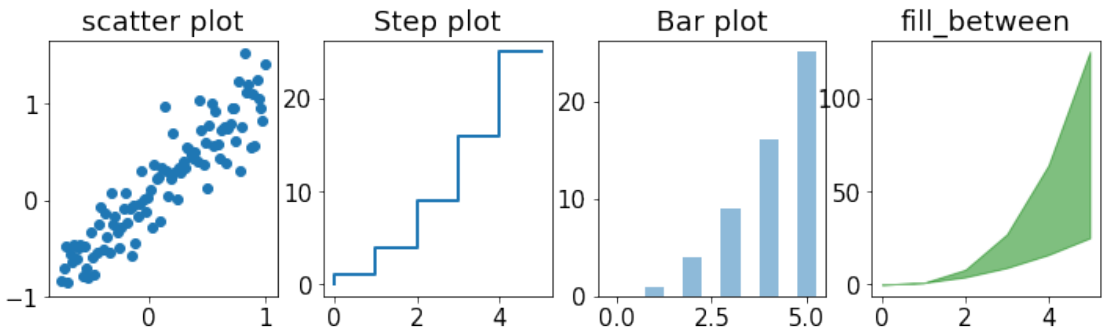
1.4 Task 4

Other plot styles. Given:

```
In [6]: xx = np.linspace(-0.75, 1., 100)
        n = array([0,1,2,3,4,5])
```

Generate: scatter, step, bar, fill_between

```
In [13]: fig, axes = plt.subplots(1, 4, figsize=(12,3))
         axes[0].scatter(xx, xx+0.25*np.random.randn(len(xx)))
         axes[0].set_title("scatter plot")

         axes[1].step(n, n**2, lw=2)
         axes[1].set_title("Step plot")

         axes[2].bar(n, n**2, align="center", width=0.5, alpha=0.5)
         axes[2].set_title("Bar plot")

         axes[3].fill_between(n, n**2, n**3, color="green", alpha=0.5);
         axes[3].set_title("fill_between");
```

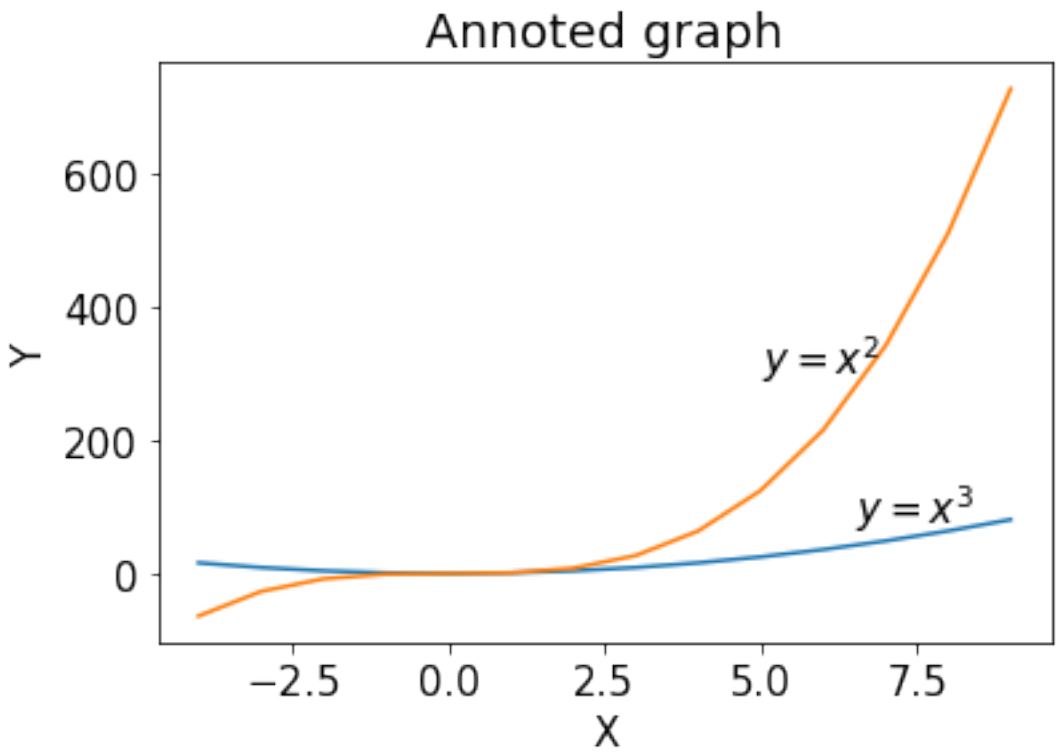


1.5 Task 5

Create a plot with annotations of the curves.

```
In [7]: a = range(-4, 10)
        b = list()
        c = list()
        print a
        for j in a:
            b.append(j**2)
            c.append(j**3)
        plt.plot(a, b)
        plt.plot(a, c)
        plt.annotate(r'$y=x^2$', (5,300))
        plt.annotate(r'$y=x^3$', (6.5,75))
        plt.title("Annoted graph")
        plt.xlabel("X")
        plt.ylabel("Y")
        plt.show()
```

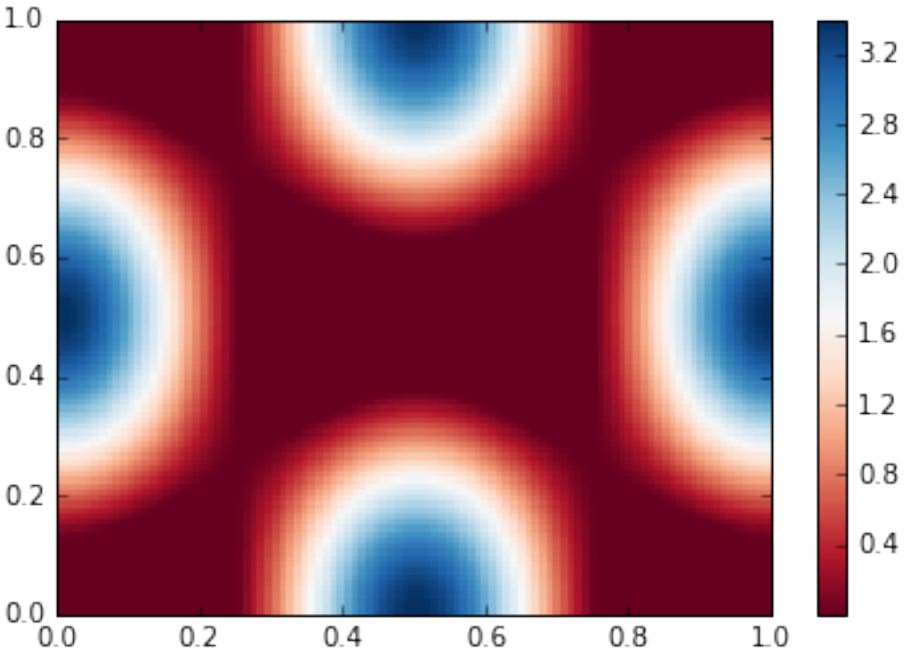
```
[-4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```



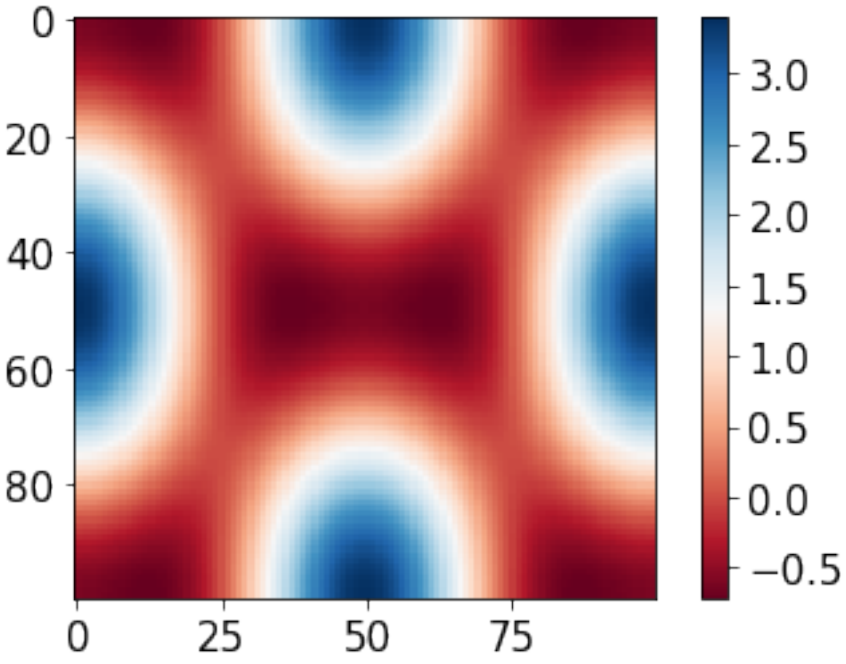
1.6 Task 6

Create a color map using pcolor and colorbar functions for the following X,Y and Z

```
In [8]: alpha = 0.7
        phi_ext = 2 * pi * 0.5

        def flux_qubit_potential(phi_m, phi_p):
            return ( + alpha - 2 * cos(phi_p)*cos(phi_m) -
                alpha * cos(phi_ext - 2*phi_p))

        phi_m = linspace(0, 2*pi, 100)
        phi_p = linspace(0, 2*pi, 100)
        X,Y = meshgrid(phi_p, phi_m)
        Z = flux_qubit_potential(X, Y).T

In [14]:
```



In [9]:
```
plt.imshow(Z, cmap='RdBu')
plt.colorbar()
plt.show()
```



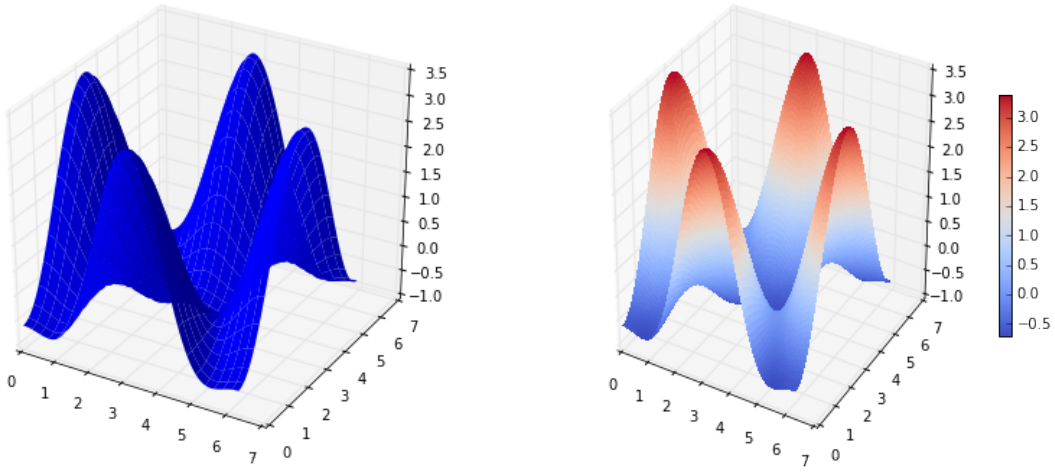
1.7 Task 7

For the same data (i.e. X,Y and Z) create plot_surface, plot_wireframe, contour plot with projections, using

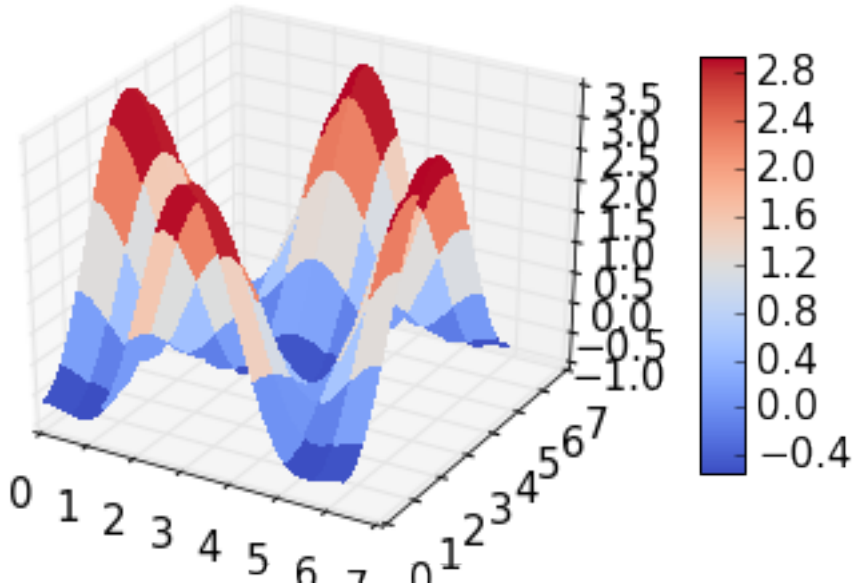
```
In [11]: from mpl_toolkits.mplot3d.axes3d import Axes3D
```

Replicate the plots introduced below (you can use your own data for this)

```
In [16]:
```




```
In [12]: fig = plt.figure()
         ax = fig.gca(projection='3d')
         surf = ax.plot_surface(X, Y, Z, cmap=cm.coolwarm, linewidth=0,  antialiased = False)

         fig.colorbar(surf, shrink=0.7, aspect=9)
         plt.show()
```



In [13]:
```
fig = plt.figure()
ax = fig.gca(projection='3d')
surf = ax.plot_surface(X, Y, Z, rstride=1, cstride=1)
plt.show()
```