Lab 2: Plotting with matplotlib

1 Graphing Functions

Matplotlib Application Interfaces (APIs): Matplotlib has two major application interfaces:

An explicit “Axes” interface that uses methods on a Figure or Axes object to create other Artists, and build a visualization step by step, called an “object-oriented” interface.

An implicit “pyplot” interface that keeps track of the last Figure and Axes created, and adds Artists to the object it thinks the user wants. **We will mostly use this approach in this course.**

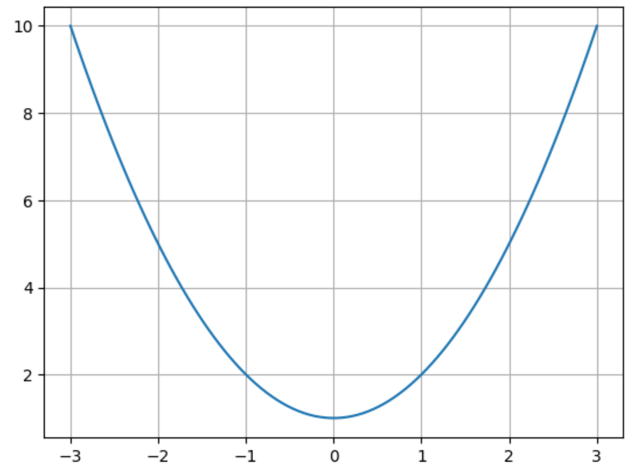
See more info about **Matplotlib** in the following links:
`https://matplotlib.org/stable/users/explain/api_interfaces.html`
`https://matplotlib.org/stable/gallery/pyplots/index.html`

Example 1.

Consider the function
$$f(x) = 1 + x^2.$$
The following commands can be used to generate a plot of the function f on the interval $[-3, 3]$.

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

x = np.linspace(-3,3,100) # 100 points in (-3 ,3)
y = 1+x**2
plt.plot(x,y) # graphing y as a function of x
plt.grid()
plt.show()
```

We should get a figure that looks like this.


10
8
6
4
2
−3
−2
−1
0
1
2
3

The following example provides all the information we need to graph basic functions:

Example 2.

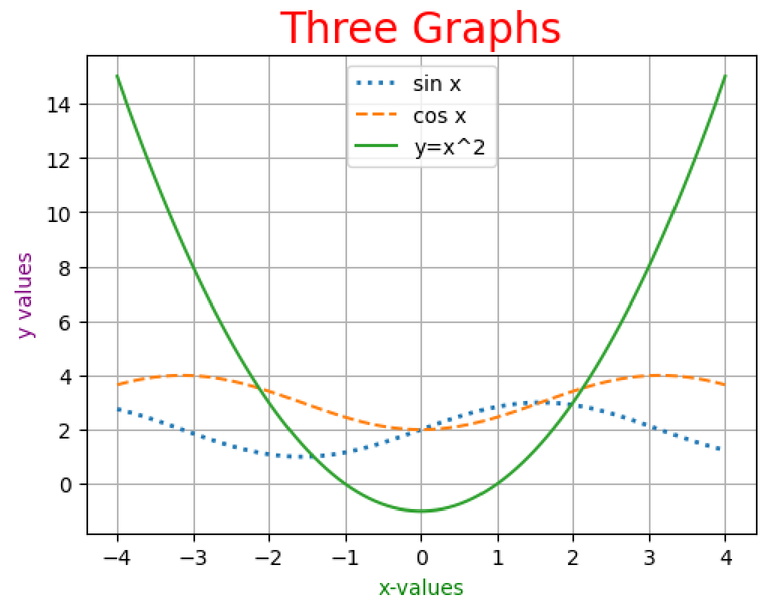
Graph the following three functions on the same domain $[-4, 4]$.

$$y = 2 + \sin(x)$$
$$y1 = 3 - \cos(x)$$
$$y2 = x^2 - 1$$

The following code will generate the desired graphs.

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

plt.figure(figsize=(5, 4), layout='constrained')
x = np.linspace(-4 , 4, num=100) # domain
# define the three functions
y = 2 + np.sin(x)
y1 = 3 - np.cos(x)
y2 = x**2-1

# plot
# default linewidth = 1
# play around different line styles and colors

plt.plot(x, y, linestyle=':', linewidth= 2, label='sin x') #dotted line
plt.plot(x, y1, linestyle = '--', label = 'cos x')
plt.plot(x, y2, label = 'y=x^2')
# plt.plot(x, y2, label = r '$y=x^2$') if you know LaTex (math majors)

# graphics standard

plt.title('Three Graphs', fontsize = 20, color = 'r')
plt.xlabel('x-values', color = 'g')
plt.ylabel('y values', color='purple')
plt.legend()
plt.grid(True)
plt.show()
```

We will get the following graphs given on the next page.



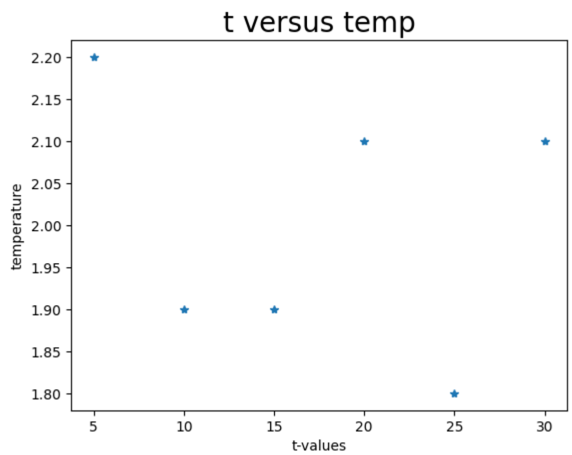
Example 3.

Let's graph the following discrete data.

t	5	10	15	20	25	30
temp	2.2	1.9	1.9	2.1	1.8	2.1

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

t = np.arange(5,31,5)
temp = [2.2, 1.9, 1.9, 2.1, 1.8, 2.1]
plt.plot(t , temp, '*') # we don't want lines here
plt.title('t versus temp', fontsize = 20)
plt.xlabel('t-values')
plt.ylabel('temperature')
plt.show()
```

We get the following graph:



---

2 Graphics Standards

The following standards should be followed for all graphics submitted in class. See the graph below for an example of a well-labeled graph.

1. **Labels.** The horizontal and vertical axes should be labeled. If the graph shows a function $y = f(x)$, then it is sufficient to label the axes with `x` and `y`. However, if the graph has meaning in some larger context, then more descriptive labels should be used. It is always a good idea to include the units of measurement in the axes labels if they are known.

2. **Title.** A title should be provided which clearly identifies the significance of the plot. The title should be descriptive, and it should include information that is not given in the axes labels. For example, if the axes are labeled `x` and `y`, then a title reading `Plot of y versus x` does not provide any additional information. The title should refer to the particular problem or context of the graph. Too much information is much better than not enough.

3. **Legend.** Multiple lines on the same plot should be distinguished by use of different line types or colors. The significance of each line should be explained in the caption or a legend.

4. **Font Size and Line Width.** The graph should be easy to read in the format that is being used (screen, print, etc.). Adjust the font size and line width accordingly. Also keep in mind that color graphs may be printed or copied in black-and-white. Use different line types so that graphs can still be read without color.

3 Exercises

For each of the following exercises you need to submit both the script and plot(s) as a pdf file. Make sure your plots are compliant with the *graphics standards* outlined in the previous section.

Please submit one page pdf for each of the problems so there will be three pages.

1. Consider a simple electric circuit consisting of a resistor, a capacitor, and an inductor. Suppose the charge on the capacitor $q(t)$ as a function of time can be computed as

   $$q(t) = q_0 e^{-Rt/(2L)} \cos\left[\sqrt{\frac{1}{LC} - \left(\frac{R}{2L}\right)^2}\, t\right]$$

   where t = time, q_0 = the initial charge, R = the resistance, L = inductance, and C = capacitance. Use MATLAB to generate a plot of this function from $t = 0$ to 0.5, given that $q_0 = 10$, $R = 50$, $L = 5$, and $C = 10^{-4}$.

2. Graph the following three functions on the same set of axes on $[0, 5]$.

   $$\begin{aligned} y1 &= e^{-x} \\ y2 &= -e^{-x} \\ y2 &= e^{-x}\cos(6x) \end{aligned}$$

   Don't forget to follow *graphing standards*.

3. It is general practice in engineering and science that equations be plotted as lines and discrete data as symbols. Here is some data for concentration (c) versus time (t) for the photodegradation of aqueous bromine:

t, min	10	20	30	40	50	60
c, ppm	3.4	2.6	1.6	1.3	1.0	0.5

   This data can be described by the following function:

   $$C(t) = 4.84e^{-0.034t}, \quad 0 \le t \le 70$$

   Create a plot displaying both the data (using square symbols) and the function (using a dashed line). Note that there should be six discrete data points. However, the graph of the function should be a smooth curve. Note that you are graphing a discrete data (t, c), and a continuous function $(t, C(t))$ on the same set of axes. (You may want to use different variable than t in the continuous function.)