Professur Technische Informatik
Prof. Dr. Wolfram Hardt

Introduction to MatPlot Library

Exercise 1.7

Shadi Saleh

Agenda

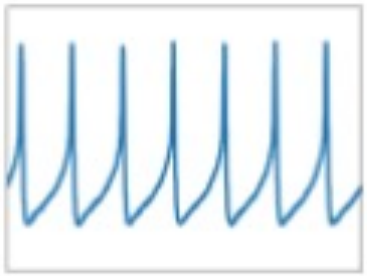
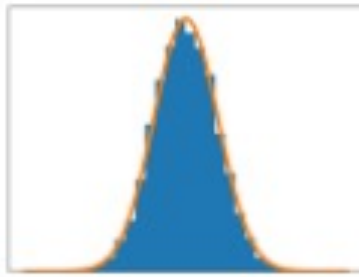
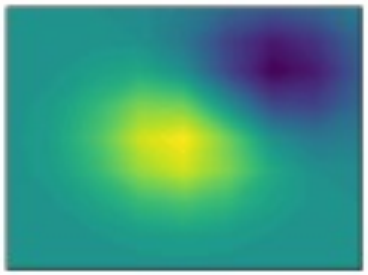
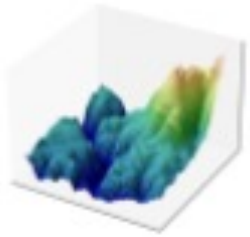
- Introduction to Matplot lib
- Histogram plot
- Multi plot
- Matrix of subplots
- Figure settings

Introduction to Matplotlib

- Matplotlib is an amazing visualization library in Python for 2D plots of arrays
- Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.
- One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals.
- Matplotlib consists of several plots like line, bar, scatter, histogram etc

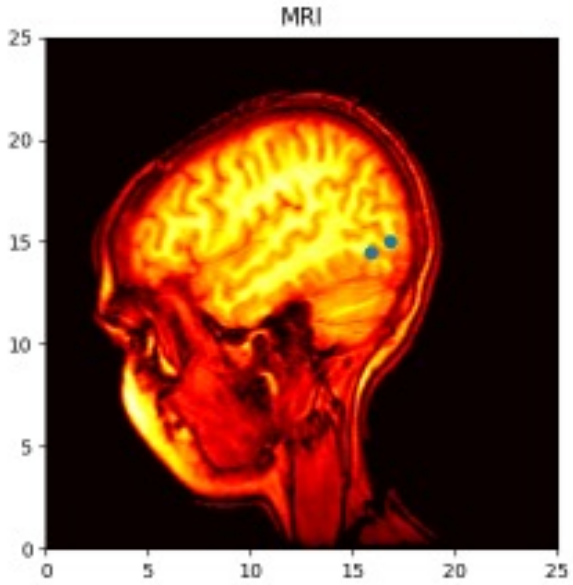
Visualization

- Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
- Matplotlib makes easy things easy and hard things possible.

Images

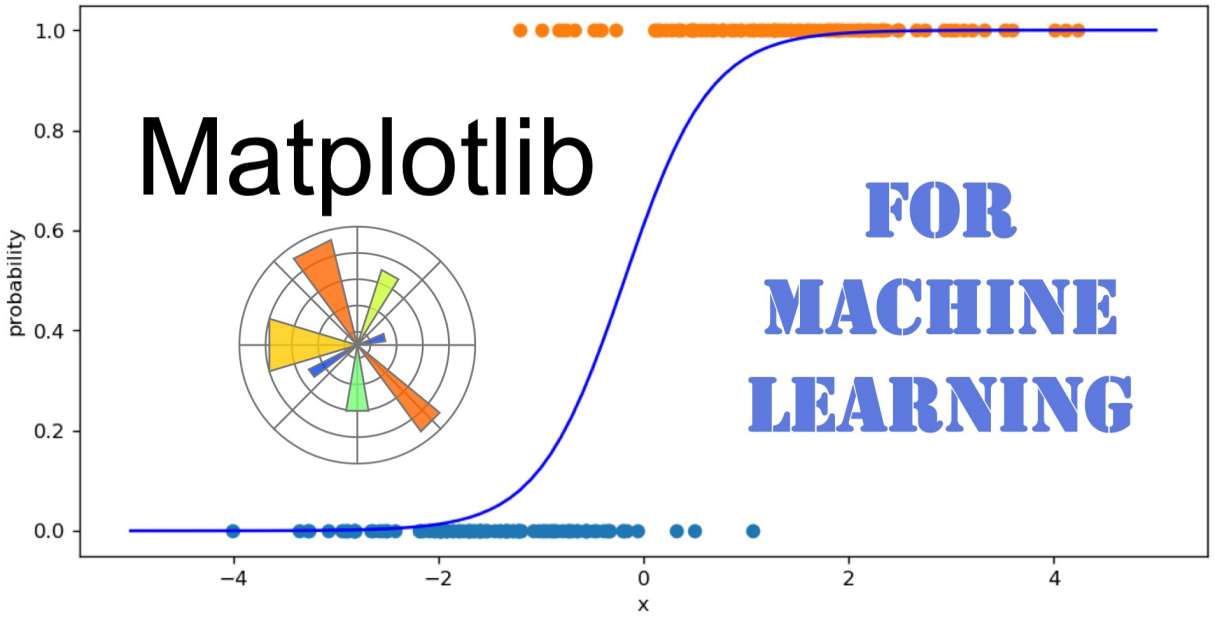
Matplotlib can display images (assuming equally spaced horizontal dimensions) using the imshow() function.



Example of using imshow() to display a CT Scan

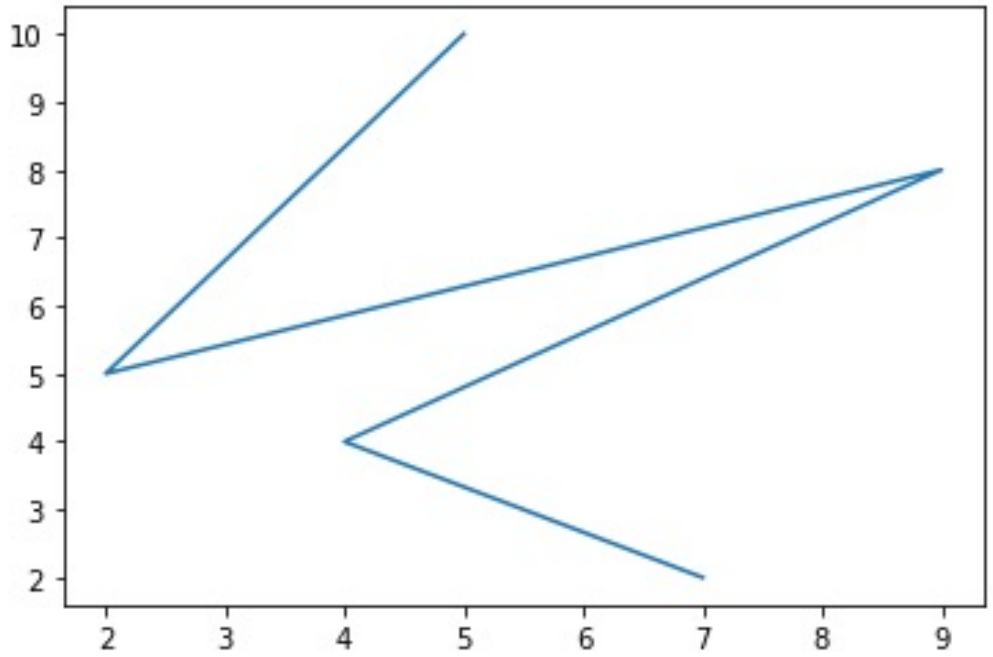
Python: Plotting

- Matplotlib designed to look like MATLAB plot
- 200 subroutines for various plots.
- Generally available with Python



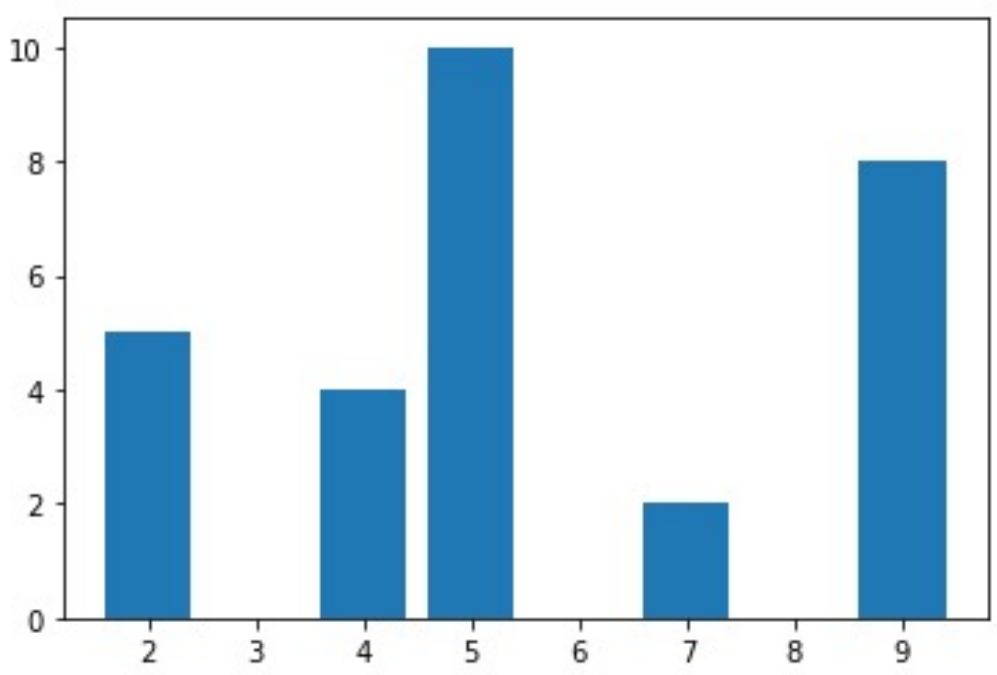
Line plot

```
from matplotlib import pyplot as plt
 # x-axis values
 x = [5, 2, 9, 4, 7]

 # Y-axis values
 y = [10, 5, 8, 4, 2]

 # Function to plot
 plt.plot(x,y)

 # function to show the plot
 plt.show()
```



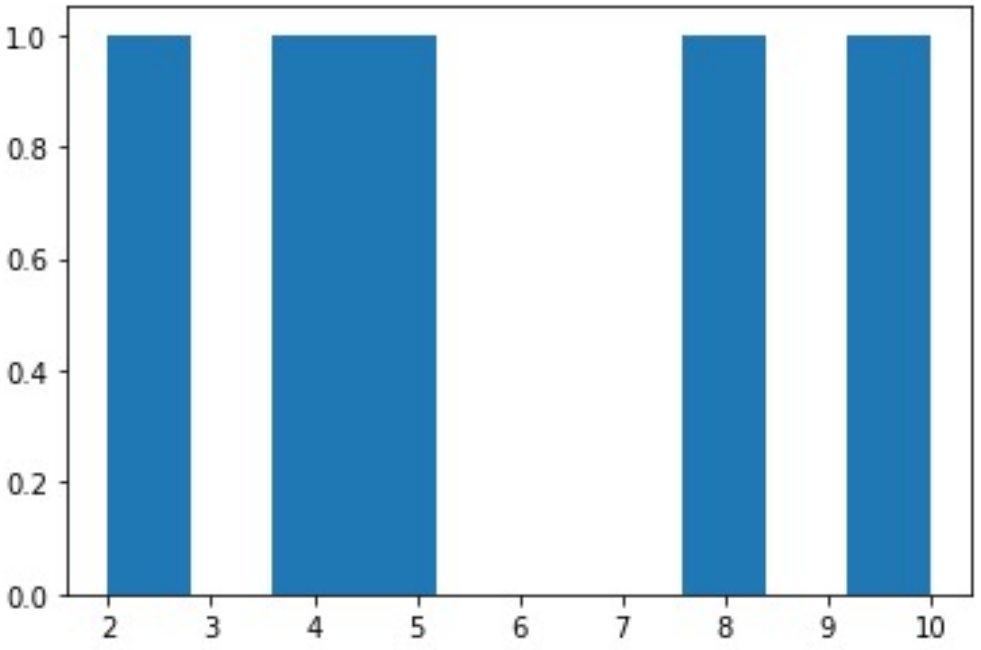
Bar plot

```
from matplotlib import pyplot as plt
 # x-axis values
 x = [5, 2, 9, 4, 7]

 # Y-axis values
 y = [10, 5, 8, 4, 2]

 # Function to plot the bar
 plt.bar(x,y)

 # function to show the plot
 plt.show()
```



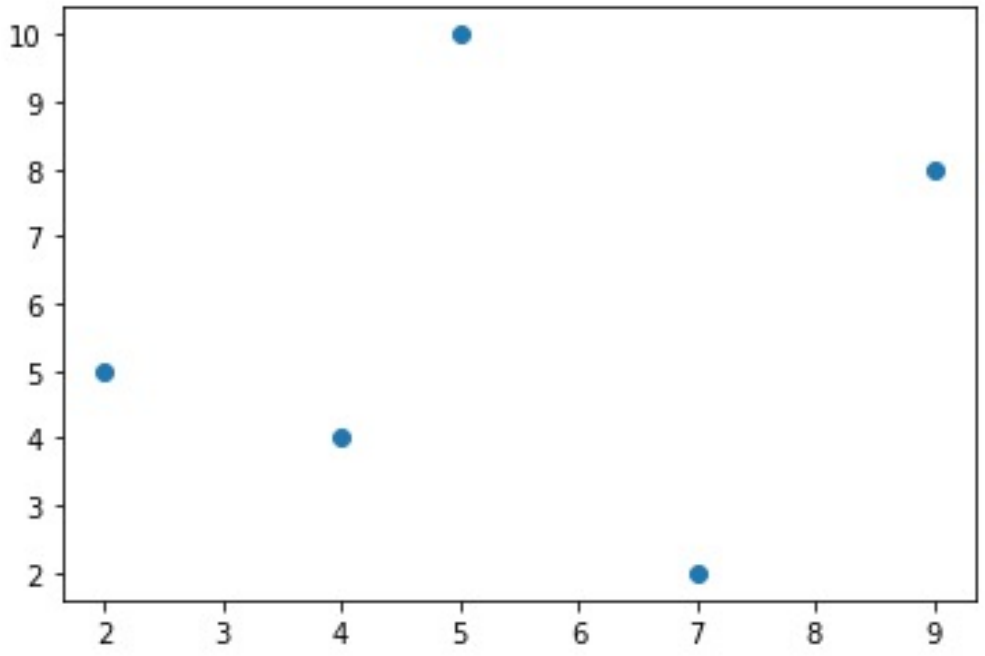
Histogram

```
from matplotlib import pyplot as plt
 # Y-axis values
 y = [10, 5, 8, 4, 2]

 # Function to plot histogram
 plt.hist(y)

 # function to show the plot
 plt.show()
```



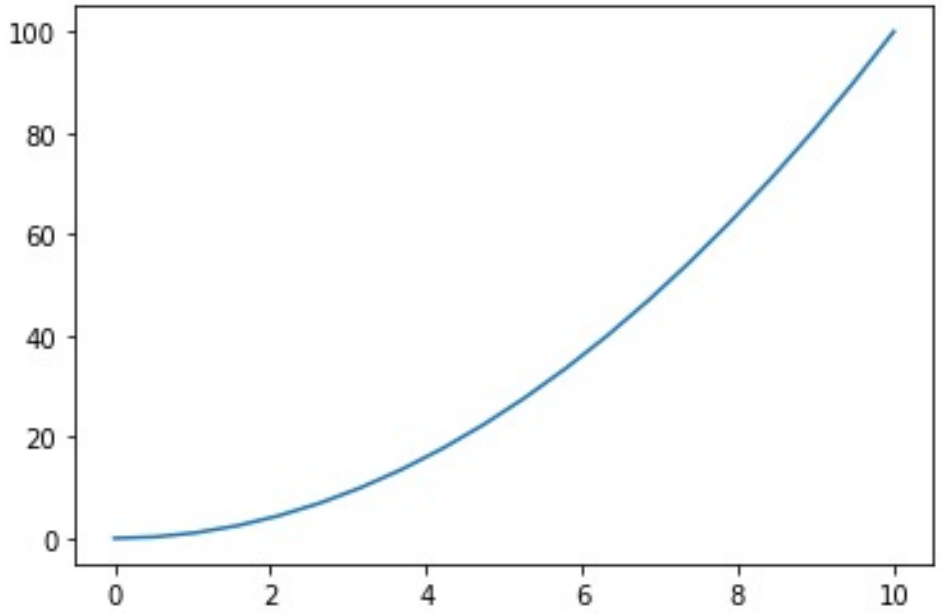
Scatter Plot

```
from matplotlib import pyplot as plt
 # x-axis values
 x = [5, 2, 9, 4, 7]

 # Y-axis values
 y = [10, 5, 8, 4, 2]

 # Function to plot scatter
 plt.scatter(x,y)

 # function to show the plot
 plt.show()
```



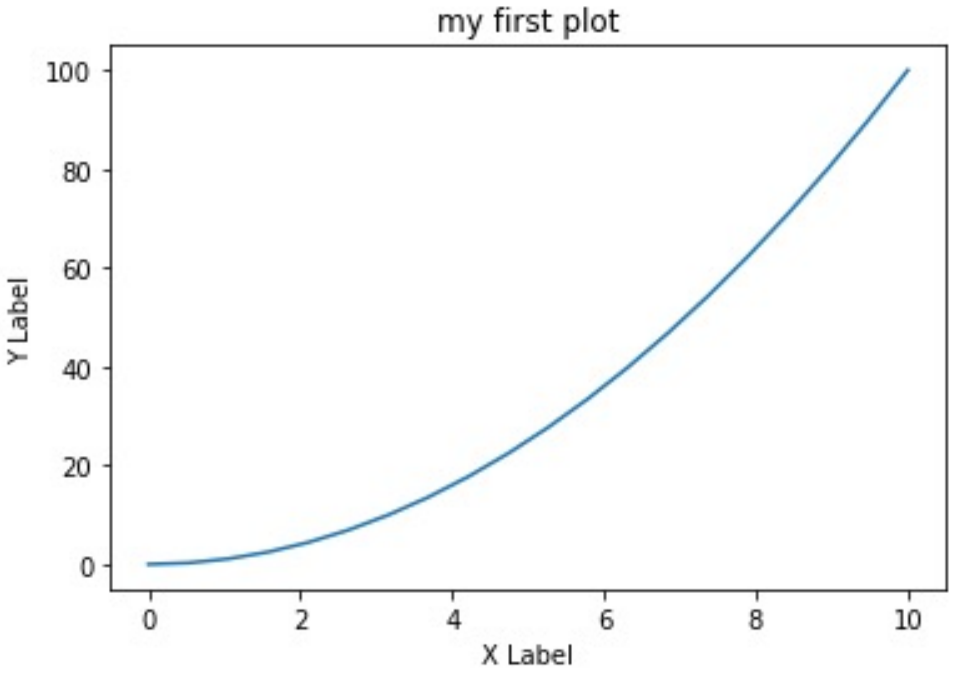
Functional approach

- import numpy as np
- x= np.linspace(0,10,20) # generate 20 datapoints between 0 and 20
- y= x**2 # generate array ‘y’ from square of ’x’
- plt.plot(x,y)



Example

```
from matplotlib import pyplot as plt
plt.plot(x,y)
plt.title('my first plot')
plt.xlabel('X Label')
plt.ylabel('Y Label')
```



Line 2D Properties

Property	Value type
Alpha	Float
animated	[True| False]
Clip_box	A matplotlib.transform.Bbox instance
Clip_on	[True| False]
Color or c	Any matplotlib color
data	(np.arrat xdata, np.array ydata)
label	Any string
picker	Used in interactive line selection
pickradius	The line pick selection radius

Multi plots

- Matplotlib allows us easily create multi-plots on the same figure using the .subplot() method
- This .subplot() method takes in three parameters, namely:
  - nrows:
    - the number of rows the Figure should have.
  - ncols:
    - the number of columns the Figure should have.
  - plot_number :
    - which refers to a specific plot in the Figure.

Multi plots

```
from matplotlib import pyplot as plt
plt.subplot(1,2,1)
plt.plot(x,y,'red')
plt.subplot(1,2,2)
plt.plot(y,x,'green')
```

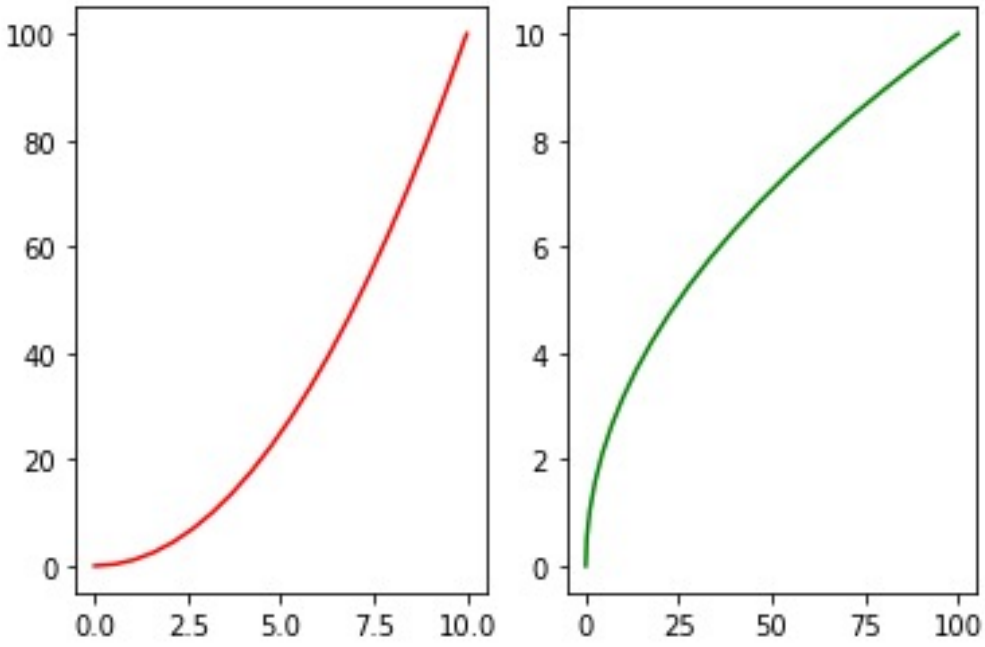
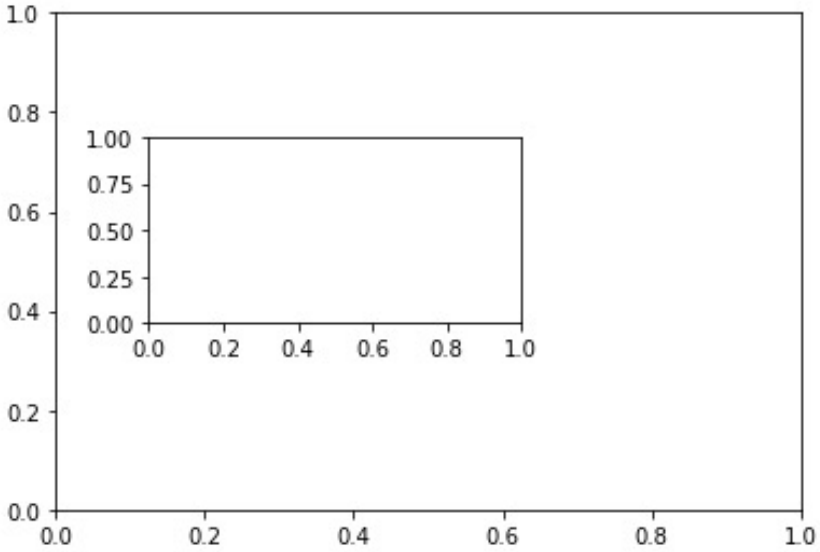


Figure in figure

```
#from matplotlib import pyplot as plt
from matplotlib import pyplot as plt
fig = plt.figure()
axis1= fig.add_axes([0.1,  0.1,  0.8, 0.8])

axis2= fig.add_axes([0.2,  0.5,  0.4, 0.3])
```



Matrix of subplot

```
#from matplotlib import pyplot as plt
from matplotlib import pyplot as plt
fig = plt.figure()
fig.axes= plt.subplots(nrows=3, ncols=3)
```

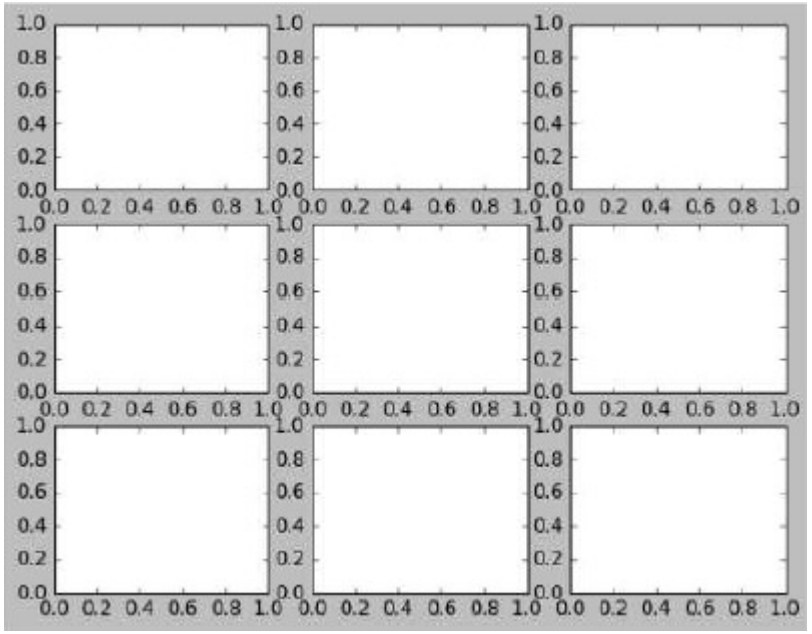
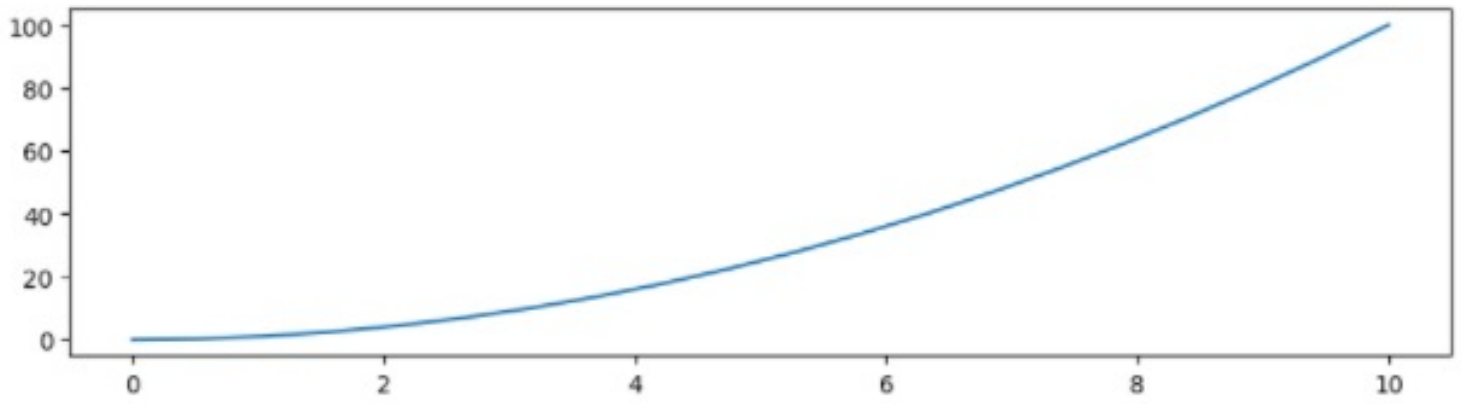


Figure size, aspect ratio, and DPI in figure

```
#from matplotlib import pyplot as plt
from matplotlib import pyplot as plt
fig = plt.figure(figsize=(8,2), dpi = 100)
ax = fig.add_axes([0,0,1,1])
ax.plot(x,y)
```



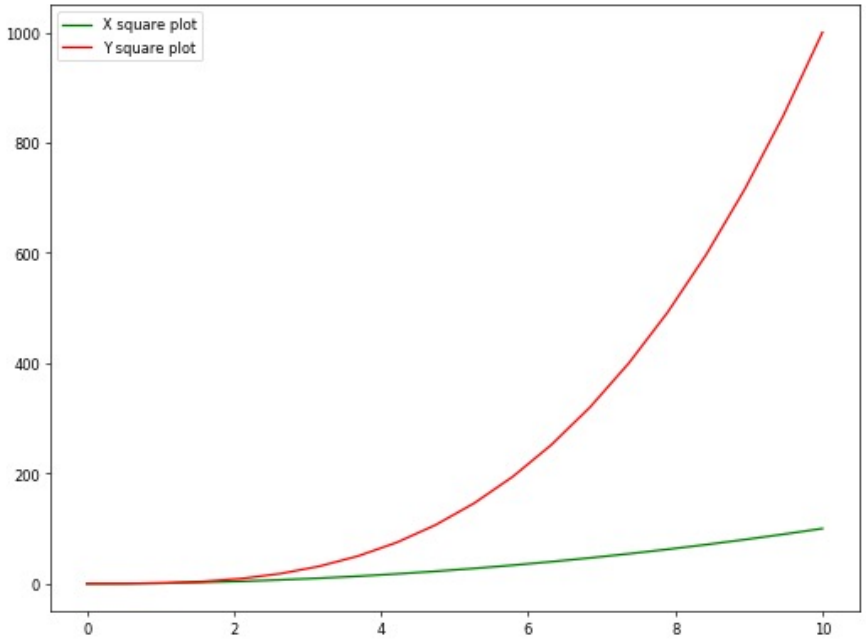
Legends

- Legends allows us to distinguish between plots. With Legends, you can use label texts to identify or differentiate one plot from another. For example, say we have a figure having two plots like below:

```
fig = plt.figure(figsize=(8,6), dpi = 60)
ax = fig.add_axes([0,0,1,1])
ax.plot(x, x**2)
ax.plot(x, x**3, 'red')
```

Legend

```
from matplotlib import pyplot as plt
import numpy as np
fig = plt.figure(figsize=(8,6), dpi = 60)
ax = fig.add_axes([0,0,1,1])
x= np.linspace(0,10,20)
ax.plot(x, x**2, 'green', label = "X square plot")
ax.plot(x, x**3, 'red', label = "Y square plot")
ax.legend()
```



WHAT YOU HAVE LEARN AND WHY

- For data visualization
- We can integrate the visualization technique in our model ML model
- We can sell visualization to company by their data
- Plotly, seaborn and matplotlib are replacement for each other

Exercise 1

- Write a text file
  - test.txt

    1 2
    2 4
    3 1
- Write a Python program to draw a line using given axis values taken from a text file, with suitable label in the x axis, y axis and a title.

Exercise 2

- Write a Python program to plot two or more lines with legends, different widths and colors.

Thank you for your attention

Shadi Saleh

Professur Technische Informatik
Fakultät für Informatik

Technische Universität Chemnitz
Straße der Nationen 62
09111 Chemnitz
Germany

shadi.saleh@informatik.tu-chemnitz.de