**Matplotlib**

**Syed Mansoor ul Hassan Bukhari**

**Introduction**

Matplotlib is a low-level graph plotting library in Python that serves as a visualization utility. It is open-source and can be used freely. Matplotlib is mostly written in Python, with some segments in C, Objective-C, and JavaScript for platform compatibility.

**Installation**

To install Matplotlib, you can use pip:

pip install matplotlib

**Importing**

To use Matplotlib, you need to import it. Most of the utilities lie under the pyplot submodule, usually imported under the alias plt:

import matplotlib.pyplot as plt

**Basic Plotting**

**Drawing a Line**

To draw a line from position (0,0) to position (6,250):

import matplotlib.pyplot as plt

import numpy as np

xpoints = np.array([0, 6])

ypoints = np.array([0, 250])

plt.plot(xpoints, ypoints)

plt.show()

**Plotting x and y Points**

The plot() function is used to draw points (markers) in a diagram. By default, it draws a line from point to point.

import matplotlib.pyplot as plt

import numpy as np

xpoints = np.array([1, 8])

ypoints = np.array([3, 10])

plt.plot(xpoints, ypoints)

plt.show()

To plot only the markers, use the shortcut string notation parameter 'o':

import matplotlib.pyplot as plt

import numpy as np

xpoints = np.array([1, 8])

ypoints = np.array([3, 10])

plt.plot(xpoints, ypoints, 'o')

plt.show()

**Multiple Points**

You can plot multiple points by ensuring you have the same number of points in both axes.

import matplotlib.pyplot as plt

import numpy as np

xpoints = np.array([1, 2, 6, 8])

ypoints = np.array([3, 8, 1, 10])

plt.plot(xpoints, ypoints)

plt.show()

**Default X-Points**

If you do not specify the points on the x-axis, they will get default values (0, 1, 2, 3, etc.).

import matplotlib.pyplot as plt

import numpy as np

ypoints = np.array([3, 8, 1, 10, 5, 7])

plt.plot(ypoints)

plt.show()

**Markers**

You can use the keyword argument marker to emphasize each point with a specified marker.

import matplotlib.pyplot as plt

import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, marker='o')

plt.show()

**Format Strings (fmt)**

You can use the shortcut string notation parameter to specify the marker, line style, and color.

import matplotlib.pyplot as plt

import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, 'o:r')

plt.show()

**Marker Size**

Set the size of the markers using the markersize or ms keyword argument.

import matplotlib.pyplot as plt

import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, marker='o', ms=20)

plt.show()

**Marker Color**

Set the color of the edge of the markers using markeredgecolor or mec.

import matplotlib.pyplot as plt

import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, marker='o', ms=20, mec='r')

plt.show()

**Linestyle**

Change the style of the plotted line using the linestyle or ls keyword argument.

import matplotlib.pyplot as plt

import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, linestyle='dotted')

plt.show()

**Line Color**

Set the color of the line using the color or c keyword argument.

import matplotlib.pyplot as plt

import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, color='r')

plt.show()

You can also use hexadecimal color values:

plt.plot(ypoints, c='#4CAF50')

**Line Width**

Set the width of the line using the linewidth or lw keyword argument.

import matplotlib.pyplot as plt

import numpy as np

ypoints = np.array([3, 8, 1, 10])

plt.plot(ypoints, linewidth=2.5)

plt.show()