Matplotlib 1-3

Page No.
Date :

```
Import matplotlib.pyplot as plt
%matplotlib inline  # using this line to see plots in the notebooks.

Import numpy as np.
x = np.linspace(0,5,11)
y = x**2
```

Basic matplotlib commands

```
plt.plot(x,y,'r')  # 'r' is the colour red
plt.xlabel('x axis Title here')
plt.ylabel('y axis Title Here')
plt.title('String Title here')
plt.show()
```

Creating multiplots on same canvas

```
plt.subplot(1,2,1)
plt.plot(x,y,'r--')  # more on color options later.
plt.subplot(1,2,2)
plt.plot(y,x,'g*-');
```

Matplotlib Object Oriented Method

```
# Create figure (Empty canvas)
fig = plt.figure()
  # Add set of axes to figure.
axes = fig.add_axes([0.1,0.1,0.8,0.8])
                # left, bottom, width, height [Range 0 to 1]
axes.plot(x,y,'b')
axes.set_xlabel('Set x label') # Notice the use of set method
axes.set_ylabel('Set y label') # Set method for y axis.
axes.set_title('Set title')
```

```
# Create blank canvas

fig = plt.figure()
axes1 = fig.add_axes([0.1,0.1,0.8,0.8]) # main axes
axes2 = fig.add_axes([0.2,0.5,0.4,0.3]) # Insert axes

## Larger fig axes 1

axes1.plot(x,y,'b')
axes1.set_xlabel('x_label_axes2')
axes1.set_ylabel('y_label_axes2')
axes1.set_title('Axes 2 Title')

## Insert Figure Axes 2

Axes2.plot(y,x,'r')
axes2.set_xlabel('x-Label_axes2')
axes2.set_ylabel('y_label_axes2').
axes2.set_title('Axes 2 Title');
```

Subplots

The plt.subplots() object will act as a more automatic axis manager.

Use similar to plt.figure() except use tuple unpacking to Grade fig and axes.

```
fig, axes = plt.subplots()
# Now use the axes object to add stuff to plat.
axes.plot(x,y,'r')
axes.set_xlabel('x')
axes.set_ylabel('y')
axes.set_title('title').
```

```
# Empty canvas of 1 by 2 subplots.
fig, axes = plt.subplots(nrows=1, ncols=2)

Axes  # Axes is an array of axes to plot on
axes.

for ax in axes:
    ax.plot(x, y, 'b')
    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.set_title('title').
fig # display the figure object.
```

→ A common issue with matplotlib is overlapping subplots or figures. We can use plt.tight_layout() method, which automatically adjusts the position of the axes on the figure. canvas so that there is no overlapping content.

```
fig, axes = plt.subplots(nrows=1, ncols=2)
for ax in axes:
    ax.plot(x, y, 'g')
    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.set_title('title')
plt.tight_layout().
```

Figsize, aspect Ratio and DPI

Matplotlib allows the aspect ratio, DPI and figure size to be specified when the figure object is created. You can use the figsize and dpi keywords arguments.

Fig Size → Tuple of width and height of the fig in inches.

dpi → dots-per-inch (Pixel per inch)

For example :-

```
fig = plt.figure(figsize=(8,4), dpi=100)
```

* The same arguements can also be passed to layout managers, such as the subplots functions:

```
fig, axes = plt.subplots(figsize=(12,3))
axes.plot(x,y,'r')
axes.set_xlabel('x')
axes.set_ylabel('y')
axes.set_title('title')
```

Saving Figures

matplotlib can generate high-quality output in a number formats.

→ fig.savefig("filename.png")

→ fig.savefig("filename.png", dpi=200).

Legends, Labels and Titles

Page No.
Date :

Figure titles

→ A title can be added to each axis instance in figure. To set the title, use the set_title method in the axes instance.

→ ax.set_title("title")

Axes label

→ ax.set_xlabel("x")

→ ax.set_ylabel("y").

Legends

```
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.plot = (x, x**2, label="x**2")
ax.plot = (x, x**3, label="x**3").
ax.legend()
```

→ The Legend function takes an optional keyword argument loc that can be used to specify where in the figure the legend is to be drawn

```
ax.legend(loc=1)   ## upper right corner.
ax.legend(loc=2)   ## upper left corner
ax.legend(loc=3)   ## lower left corner
ax.legend(loc=4).  ## lower right corner.
ax.legend(loc=0)   ## let mat decide the optimal location.
```

Page No.
Date :

Setting Colors, linewidths, line styles

#matlab style line color and style

```
fig, ax = plt.subplots()
ax.plot(x, x**2, 'b.-') # blue line with dots
ax.plot(x, x**3, 'g--') # green dashed line
```

Colors with the color= parameter

```
fig, ax = plt.subplots()
ax.plot(x, x+1, color="blue", alpha=0.5) #half transparent
ax.plot(x, x+2, color="#8B008B") #RGB hex code.
ax.plot(x, x+3, color="#FF8C00") #RGB hex code.
```

Marker size and color

```
ax.plot(x, x+13, color="purple", lw=1, ls='-',
        marker='o', markersize=2)
fig, ax = plt.subplots(figsize=(12,6)).
```

plot Range

→ We can configure the ranges of the axes using the set_ylim and set_xlim methods in the axis object, or axis('tight') for automatically getting tightly fitted.

```
fig, axes = plt.subplots(1,3, figsize=(12,4))

axes[0].plot(x, x**2, x, x**3)
axes[0].set_title("default axes ranges");

axes[1].plot(x, x**2, x, x**3)
axes[1].axis('tight')
axes[1].set_title("tight axes")

axes[2].plot(x, x**2, x, x**3)
axes[2].set_ylim([0, 60])
axes[2].set_xlim([2, 5]);
axes[2].set_title("custom axes range");
```

☆ Special Plot Types

```
plt.scatter(x, y)
```