

# Data Visualisation v1

---

Dr Amir-Homayoun Javadi

[a.h.javadi@gmail.com](mailto:a.h.javadi@gmail.com)

[www.javadilab.com](http://www.javadilab.com)

## Prerequisite

MATLAB does not require any preparation to create and display figures. But in Python, you need to load the relevant libraries. The below example shows how you can create a plot showing four cycles of a sinusoidal curve.

`% MATLAB`

```
x = linspace(0, 8 * pi, 100);
```

```
y = sin(x);
```

```
plot(x, y);
```

`# Python`

```
import matplotlib.pyplot as plt      # loads toolbox matplotlib.pyplot
```

```
import numpy as np                  # loads toolbox numpy
```

```
x = np.linspace(0, 8 * np.pi, 100)
```

```
y = np.sin(x)
```

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

Note: `linspace(a, b, n)` is a function that creates a list of  $n$  numbers between  $a$  and  $b$  (inclusive). This function is very helpful in creating plots.

Note: `pi` refers to pi number (3.14).

Note: in Python, for more complex figures (such as 3-dimentional plots/surfaces) you might need to set more properties such as below.

`# Python`

```
import matplotlib.pyplot as plt
```

```
from mpl_toolkits.mplot3d import Axes3D
```

```
fig = plt.figure()
```

```
ax = fig.add_subplot(111, projection='3d')
```

then you use `ax` for plotting. For further information please refer to [this page](#).

## plot

The most basic plotting command is `plot`. `plot(y)` plots the values of  $y$ . In this plot, the x-axis represents the indices of values in  $y$ . `plot(x, y)` plots the values of  $y$  with the x-axis representing the values in  $x$ . There are a few properties that you can control.

### Line types

Symbol	Description
-	solid line (default)
:	dotted line
--	dashed line
-.	dash-dot line

### Symbols

Symbol	Description	Symbol	Description
+	plus sign	o	circle
*	asterisk	.	point
x	cross	s	square
v	downward pointing triangle	^	upward pointing triangle
<	left pointing triangle	>	right pointing triangle
H	six-pointed star (hexagram)	p	five-pointed star (pentagram)
d	diamond		

### Colours

Symbol	Description	Symbol	Description
y	yellow	m	magenta
c	cyan	r	red
g	green (more like lime)	b	blue
w	white	k	black

```
% MATLAB
plot(x, y, 'd:r')
```

```
# Python
plt.plot(x, y, 'd:r')
```

Alternatively you can use 'Color' property to indicate RGB values. These values are between 0 and 1 (inclusive). Below creates a dark green plot.

```
% MATLAB
plot(x, y, 'Color', [0, 0.50, 0])
```

```
# Python
plt.plot(x, y, Color=[0, 0.50, 0])
```

## More properties

Switch	Description
LineWidth	specifies the width (in points) of the line
MarkerEdgeColor	specifies the colour of the marker or the edge colour for filled markers
MarkerFaceColor	specifies the colour of the face of filled markers
MarkerSize	specifies the size of the marker in points

```
% MATLAB
plot(x, y, ...
     'o-.r', ...
     'LineWidth', 10, ...
     'MarkerEdgeColor', 'r', ...
     'MarkerFaceColor', 'y', ...
     'MarkerSize', 16)
```

```
# Python
plt.plot(x, y, 'o-.b',
         LineWidth=10,
         MarkerEdgeColor='r',
         MarkerFaceColor='y',
         MarkerSize=16)
```

## Other simple plots

Command	Description
scatter()	creates X-Y-diagram; every value of A is shown on x-axis, every corresponding value of B is shown on y-axis
bar()	values of A are shown in bar graph
histogram()	analyses abundance of all values of A and presents them in histogram with 10 classes.

```
% MATLAB - scatter
scatter(x, y + rand(1, length(y)))
```

```
% MATLAB - bar
y = [1 3 5; 3 2 7; 3 4 2];
bar(y)
```

```
% MATLAB - histogram
x = randn(1000,1);
nbins = 25; % number of bins
h = histogram(x, nbins)
```

```
# Python - scatter
plt.scatter(x, y + np.random.rand(len(y)))
```

```

# Python - bar
# in Python you need to display each bar-group separately,
# with a bit of shift on the x-axis.
# It gives you more flexibility (pro), but with more work (con)
    y = np.array([[1, 3, 5], [3, 2, 7], [3, 4, 2]]);
    x = np.arange(3) # similar to range, but the output is type of np.array

plt.bar(x-0.2, y[0, :], width=0.2, color='b', align='center')
plt.bar(x      , y[1, :], width=0.2, color='g', align='center')
plt.bar(x+0.2, y[2, :], width=0.2, color='r', align='center')

# Python - hist
    y = np.random.randn(1000)
    nbins = 25 # number of bins
    plt.hist(y, bins=nbins)

```

## Other commands

Command	Example	Explanation
hold	hold on hold off	turning on/ off the superposition of diagrams/ graphics
clf	clf	deletes content of graphics window
axis	axis tight	sets the axis limits to the range of the data
	axis square	makes the current axes region square (or cubed when three-dimensional)
	axis equal	sets the aspect ratio so that the data units are the same in every direction
	axis auto	sets to its default behaviour
	axis xy	draws the graph in the default Cartesian axes format (origin in the bottom-left corner)
	axis ij	places the coordinate system origin in the upper-left corner
legend	axis on	turns on/off all axis lines, tick marks and labels.
	axis off	
		displays legend and corresponding curve Location: combination of North, South, East and West and if wanted Outside to display the legend out of plot area.

```

% MATLAB
    clf;
    hold on;
    x = linspace(0, 8 * pi, 100);
    plot(x, sin(y), 'r-');
    plot(x, cos(y), 'b:');
    legend('sin wave', 'cos wave', 'Location', 'SouthEastOutside');

```

```

# Python
plt.hold(True)    # in contrast to MATLAB, default for Python is holding
plots
plt.hold(False)
plt.clf()
plt.axis('equal')

plt.clf()
plt.hold(True)
x = np.linspace(0, 8 * np.pi, 100)
plt.plot(x, np.sin(x), 'r-');
plt.plot(x, np.cos(x), 'b:');
plt.legend(['sin wave', 'cos wave'], loc=6)

```

In Python, positioning the legend out of the plot area is quite complicated. I refer you to [this page](#).

Finally to specify the range of the axes manually, you can use the following:

```

% MATLAB
axis([xmin, xmax, ymin, ymax])

# Python
plt.axis([xmin, xmax, ymin, ymax])

```

## Meshes and surfaces

Command	Explanation
mesh	plots a surface with or without fill
surf	
meshc	similar to mesh and surf, but with contours projected on the xy-plane showing elevation
surfc	in z-axis.

```

% MATLAB
[x, y] = meshgrid(-2:0.1:2, -2:0.1:2);
z = x.^ 2 + y.^ 2;
surf(x, y, z)

colormap jet
shading interp

# Python
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import numpy as np

```

```
fig = plt.figure()
ax = fig.gca(projection='3d')

X = np.arange(-2, 2, 0.1)
Y = np.arange(-2, 2, 0.1)
X, Y = np.meshgrid(X, Y)
Z = X**2 + Y**2

ax.plot_surface(X, Y, Z, cmap=cm.jet, linewidth=0, antialiased=False)
```

Note: colormap in MATLAB and cmap in Python represent colour maps. For the list of colour maps refer to [this page for MATLAB](#) and [this page for Python](#).