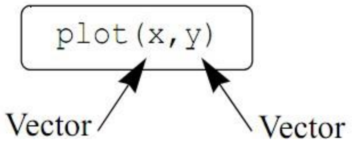
MATLAB plots

Introduction

In this chapter we introduced the use of the function plot in the MATLAB® software to get simple, two-dimensional plots of x and y points represented by two vectors, x and y. We also have seen some functions that allow customization of these plots. And we will explore other types of plots, ways of customizing plots, and some applications that combine plotting with functions and file input. Additionally, animation, three-dimensional plots, and graphics properties will be introduced.

4.1 plot command

The `plot` command is used to create two-dimensional plots. The simplest form of the command is:



The figure that is created has axes with a linear scale and default range. For example, if a vector x has the elements 1, 2, 3, 5, 7, 7.5, 8, 10, and a vector y has the elements 2, 6.5, 7, 7, 5.5, 4, 6, 8, a simple plot of y versus x can be created bytyping the following in the Command Window:

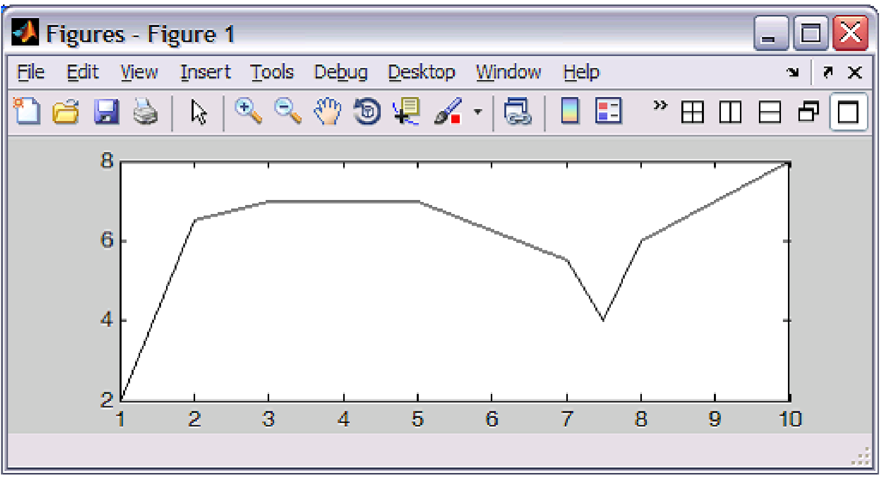
```
>> x=[1 2 3 5 7 7.5 8  10];
```

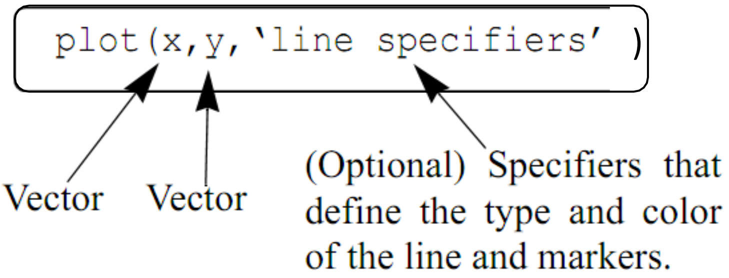
```
>> y=[2 6.5 7 7 5.5 4 6  8];
```

```
>> plot(x,y)
```

Once the plot command is executed, the Figure Window opens and the plot is

displayed, as shown in Figure .




plot(x,y,'line specifiers')
Vector
Vector
(Optional) Specifiers that define the type and color of the line and markers.

Line Specifiers:

Line specifiers are optional and can be used to define the style and color of the line and the type of markers (if markers are desired). The line style specifiers are:

Line Style	Specifier
solid (default)	-
dashed	--

Line Style	Specifier
dotted	:
dash-dot	-.

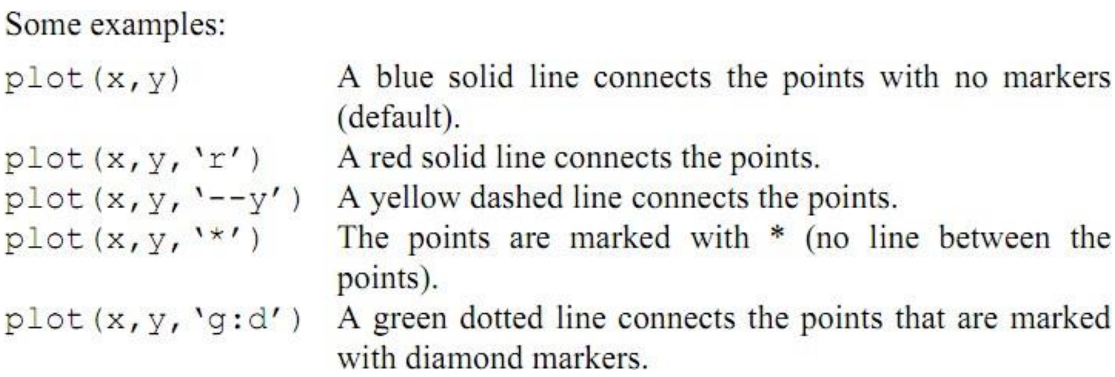
The line color specifiers are:

Line Color	Specifier
red	r
green	g
blue	b
cyan	c

Line Color	Specifier
magenta	m
yellow	y
black	k
white	w

The marker type specifiers are:

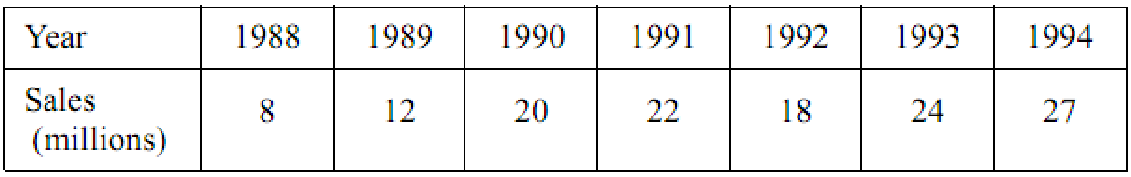
Marker Type	Specifier		Marker Type	Specifier
plus sign	+		square	s
circle	o		diamond	d
asterisk	*		five-pointed star	p
point	.		six-pointed star	h
cross	x		triangle (pointed left)	<
triangle (pointed up)	^		triangle (pointed right)	>
triangle (pointed down)	v			



Some examples:

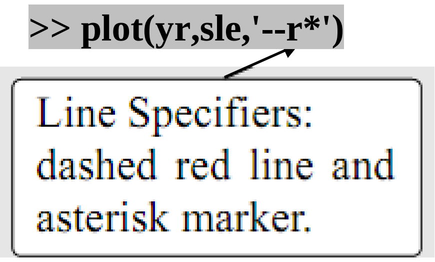
`plot(x,y)`	A blue solid line connects the points with no markers (default).
`plot(x,y,'r')`	A red solid line connects the points.
`plot(x,y,'--y')`	A yellow dashed line connects the points.
`plot(x,y,'*')`	The points are marked with * (no line between the points).
`plot(x,y,'g:d')`	A green dotted line connects the points that are marked with diamond markers.

Example , plot the following table contains sales data of a company from 1988 to 1994.

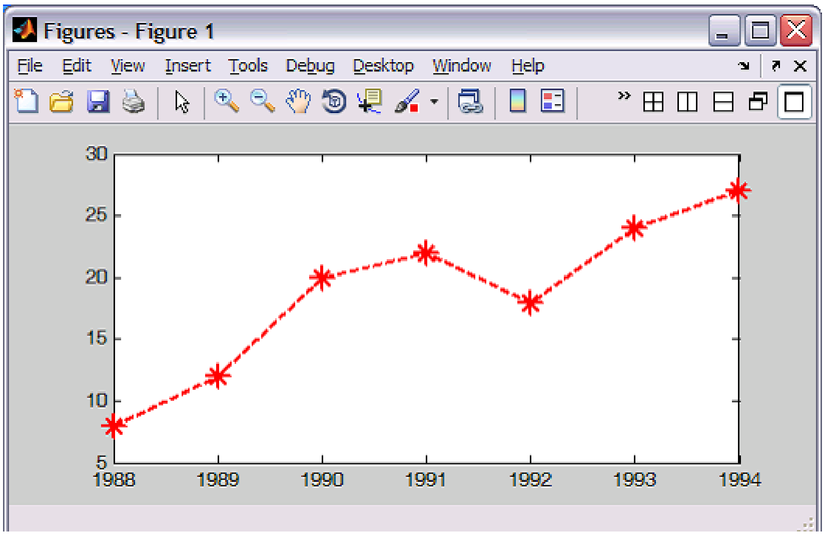


Year	1988	1989	1990	1991	1992	1993	1994
Sales (millions)	8	12	20	22	18	24	27

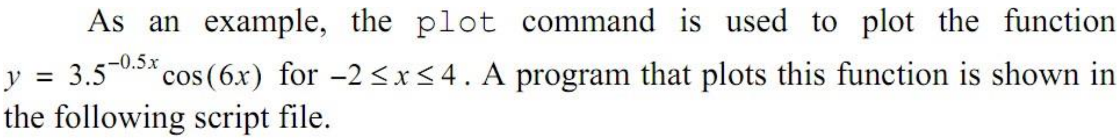
```
>> yr=[1988:1:1994];

>> sle=[8 12 20 22 18 24  27];

>> plot(yr,sle,'--r*')
```



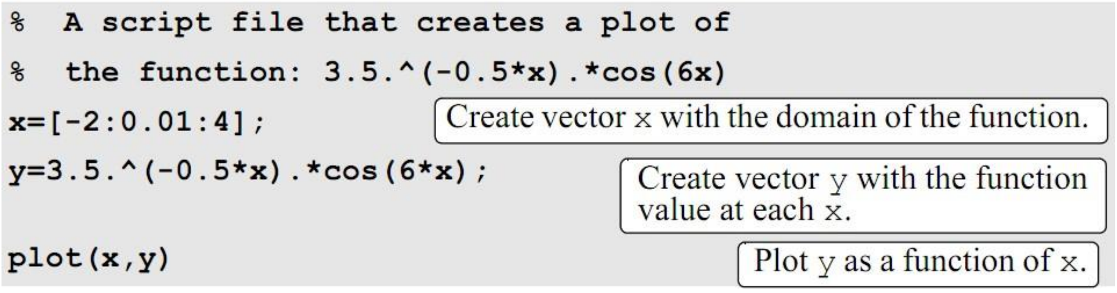
Line Specifiers: dashed red line and asterisk marker.



In order to plot a function with the plot command, the user needs to first create a vector of values of x for the domain over which the function will be plotted. Then a vector y is created with the corresponding values of by using element-by-element calculations .



As an example, the `plot` command is used to plot the function $y = 3.5^{-0.5x}\cos(6x)$ for $-2 \le x \le 4$. A program that plots this function is shown in the following script file.

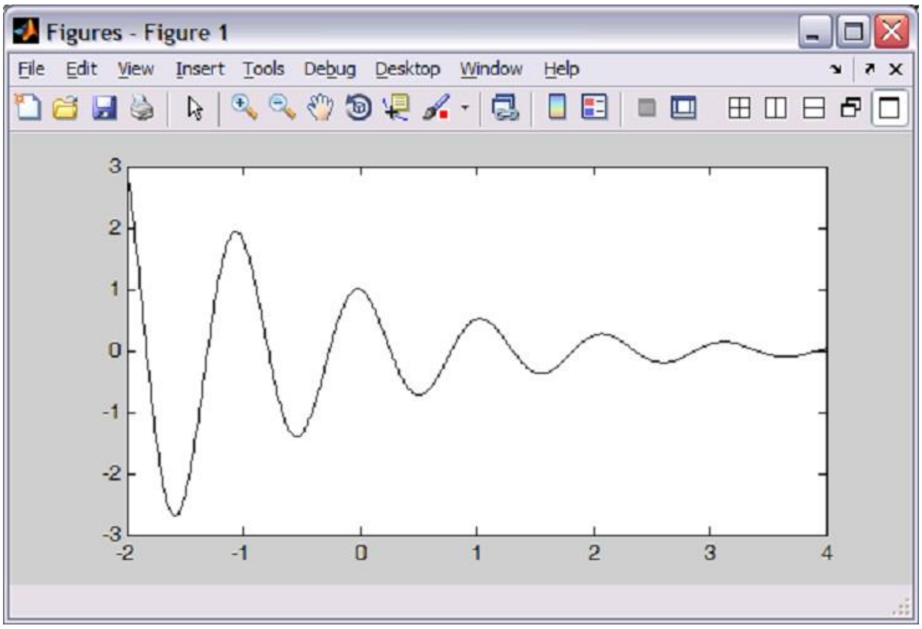


```
%  A script file that creates a plot of
%  the function: 3.5.^(-0.5*x).*cos(6x)
x=[-2:0.01:4];
y=3.5.^(-0.5*x).*cos(6*x);

plot(x,y)
```

Create vector x with the domain of the function.

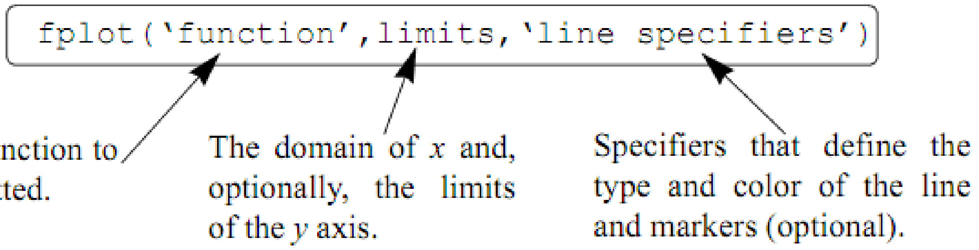
Create vector y with the function value at each x.

Plot y as a function of x.



4.2 THE *fplot* COMMAND

The `fplot` command plots a function with the form $y = f(x)$ between specified limits. The command has the form:



```
fplot('function',limits,'line specifiers')
```

The function to be plotted.

The domain of x and, optionally, the limits of the y axis.

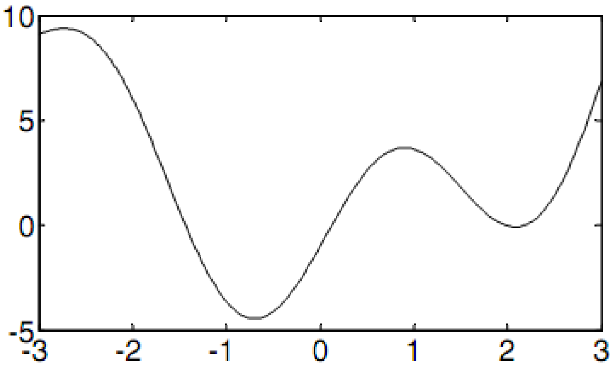
Specifiers that define the type and color of the line and markers (optional).

For example, a plot of the function $y = x^2 + 4\sin(2x) - 1$ for $-3 \le x \le 3$

be created with the `fplot` command by typing:

```
>> fplot('x^2+4*sin(2*x)-1',[-3 3])
```

The figure that is obtained in the Figure Window is shown in Figure



4.3 Plotting multiple graphs in the same plot

In many situations there is a need to make several graphs in the same plot. One is by using the plot command, the second is by using the hold on and hold off commands, and the third is by using the line command

1. Using the plot Command

Two or more graphs can be created in the same plot by typing pairs of vectors inside the `plot` command. The command

```
plot(x,y,u,v,t,h)
```

creates three graphs—`y` vs. `x`, `v` vs. `u`, and `h` vs. `t`—all in the same plot. The vectors of each pair must be of the same length. MATLAB automatically plots the graphs in different colors so that they can be identified. It is also possible to add line specifiers following each pair. For example the command

```
plot(x,y,'-b',u,v,'--r',t,h,'g:')
```

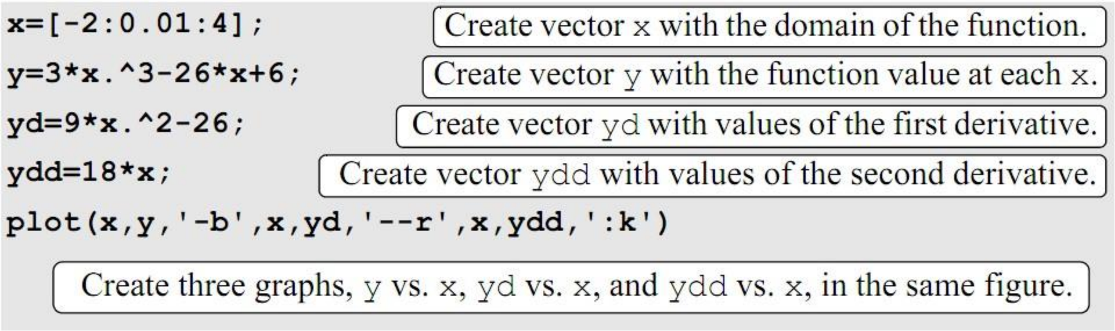
Example:

Plot the function $y = 3x^3 - 26x + 10$, and its first and second derivatives, for $-2 \le x \le 4$, all in the same plot.

The first derivative of the function is: $y' = 9x^2 - 26$.

The second derivative of the function is: $y'' = 18x$.

A script file that creates a vector x and calculates the values of y, y', and y'' is:



```
x=[-2:0.01:4];
y=3*x.^3-26*x+6;
yd=9*x.^2-26;
ydd=18*x;
plot(x,y,'-b',x,yd,'--r',x,ydd,':k')
```

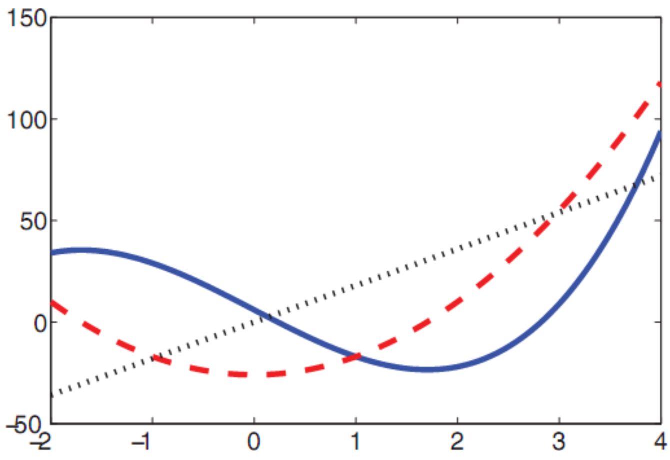
Create vector x with the domain of the function.

Create vector y with the function value at each x.

Create vector yd with values of the first derivative.

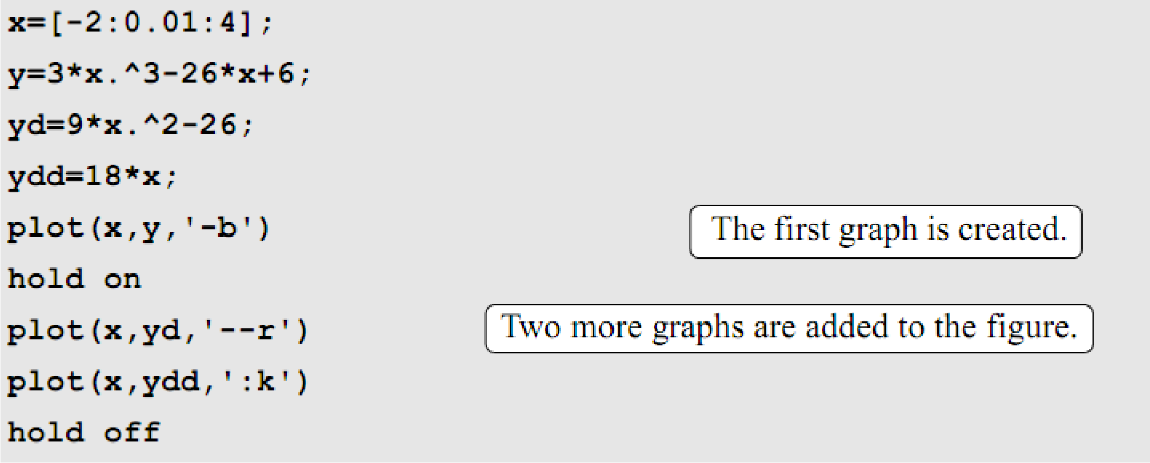
Create vector ydd with values of the second derivative.

Create three graphs, y vs. x, yd vs. x, and ydd vs. x, in the same figure.



2. Using the *hold on* and *hold off* Commands

To plot several graphs using the hold on and hold off commands, one graph is plotted first with the plot command. Then the hold on command is typed. This keeps the Figure Window with the first plot open, including the axis properties and formatting, As an example, a solution of last Problem using the hold on and hold off commands is shown in the following script file:



```
x=[-2:0.01:4];
y=3*x.^3-26*x+6;
yd=9*x.^2-26;
ydd=18*x;
plot(x,y,'-b')
hold on
plot(x,yd,'--r')
plot(x,ydd,':k')
hold off
```

The first graph is created.

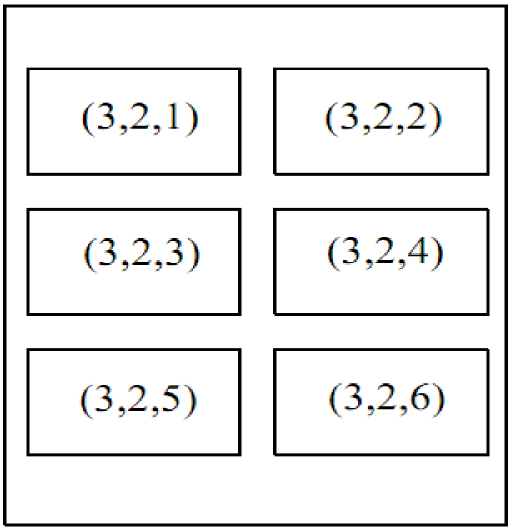
Two more graphs are added to the figure.

4.4 Putting multiple plots on the same page

Multiple plots can be created on the same page with the subplot command, which has the form:

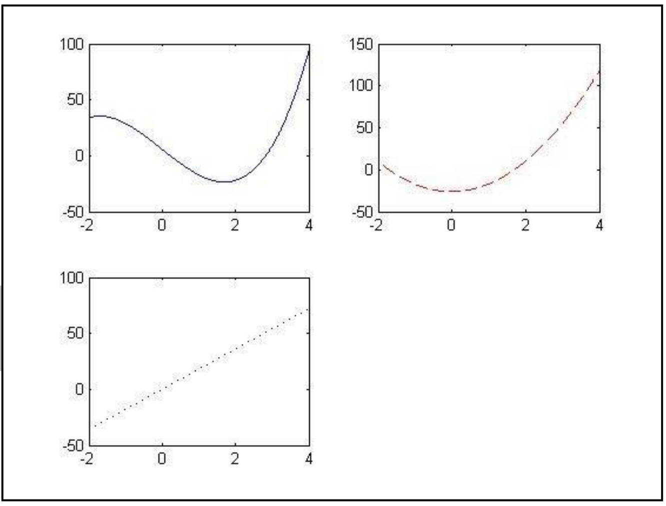
```
subplot(m,n,p)
```

The command divides the Figure Window (and the page when printed) into $m \times n$ rectangular subplots. The subplots are arranged like elements in an $m \times n$ matrix where each element is a subplot. The subplots are numbered from 1 through $m \cdot n$. The upper left subplot is numbered 1 and the lower right subplot is numbered $m \cdot n$. The numbers increase from left to right within a row, from the first row to the last. The command `subplot(m,n,p)` makes the subplot p current.



,if we solution last example using sub plot getting:

```
clear all
clc
x=[-2:0.01:4];
y=3*x.^3-26*x+6;
yd=9*x.^2-26;
ydd=18*x;
subplot(2,2,1),plot(x,y,'-b')
subplot(2,2,2),plot(x,yd,'--r')
subplot(2,2,3),plot(x,ydd,':k')
```


100
50
0
-50
-2
0
2
4
150
100
50
0
-50
-2
0
2
4
100
50
0
-50
-2
0
2
4