Chapter Two
Plotting

Creating simple plots

For example in 2D, is to take a vector of x-coordinates, and a vector of y-coordinates

$x = (x_1,.......,x_N)$

$y = (y_1,....,y_N)$,

The MATLAB command to plot a graph is **plot(x , y)**.

The vectors x=(1,2,3,4,5,6) and y=(3,-1,2,4,5,1) produce the picture shown

```
>> x = [1 2 3 4 5 6] ;
>> y = [3 -1 2 4 5 1] ;
>> plot(x,y)
```

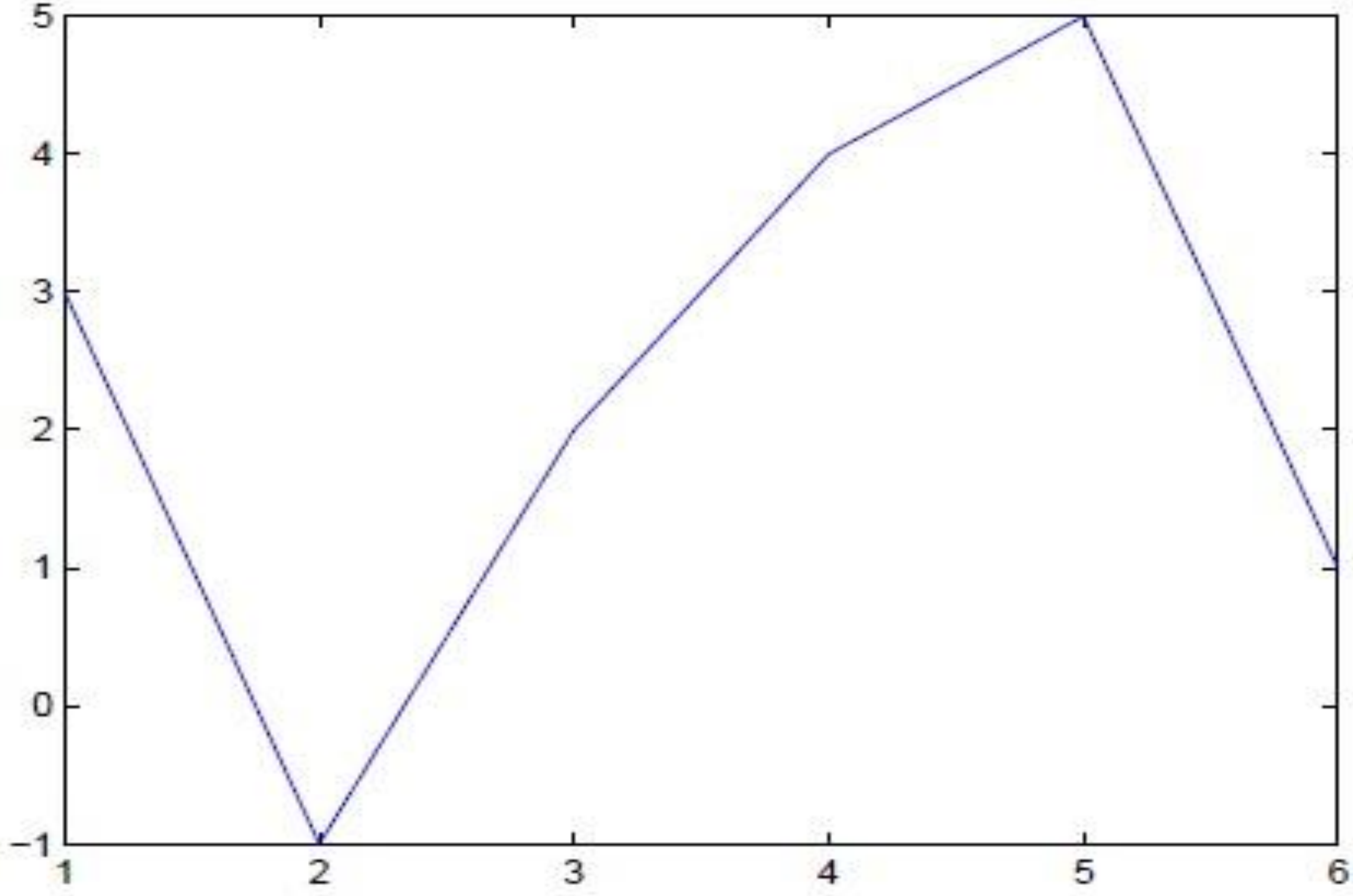


Figure 2.1: Plot for the vectors x and y

Note: The plot functions have different forms depending on the input arguments. If y is a vector plot(y) produces a piecewise linear graph of the elements of y versus the index of the elements of x.

For example, to plot the function sin(x) on the interval [0,2π], we first create a vector of x values ranging from 0 to 2π, then compute the sine of these values, and finally plot the result:

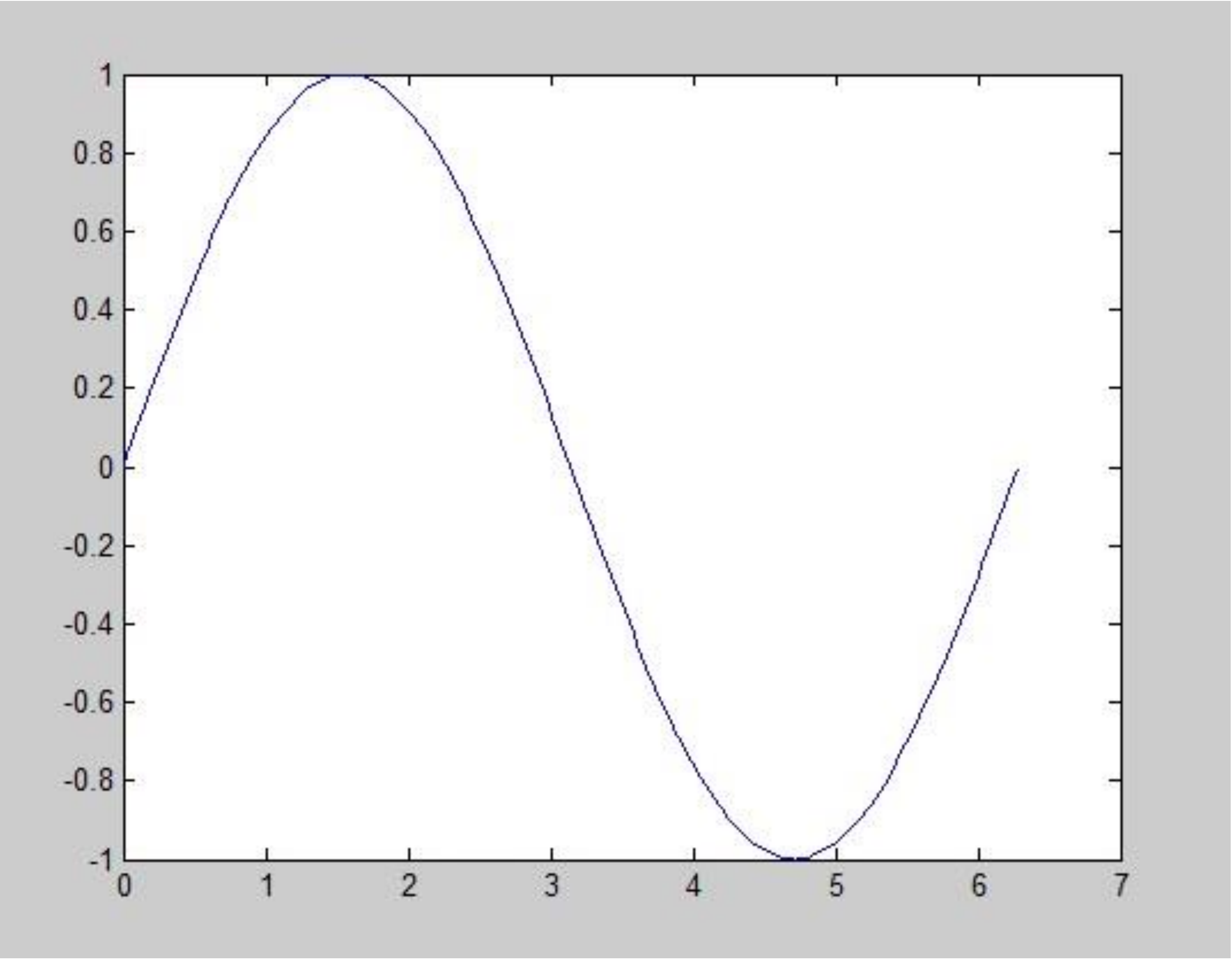
```
>> x = 0: pi/100 : 2*pi ;
>> y = sin(x) ;
>> plot ( x , y )
```

Notes:

1) 0 : pi/100 :2*pi yields a vector that

- ❑ Starts at 0,
- ❑ Takes steps (or increments) of π/100,
- ❑ Stops when 2π is reached.

2) If you omit the increment, MATLAB automatically increments by 1.


1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
1
2
3
4
5
6
7

Adding titles, axis labels, and annotations

MATLAB enables you to add axis labels and titles. For example add an x-axis and y-axis labels. Now label the axes and add a title.

The character 2pi creates the symbol π . An example of 2D plot is shown

```
>> plot(x,y)
>> xlabel (' x = 0 : 2 pi ')
>> ylabel (' Sine of x ')
>> title (' Plot of the Sine function ')
```

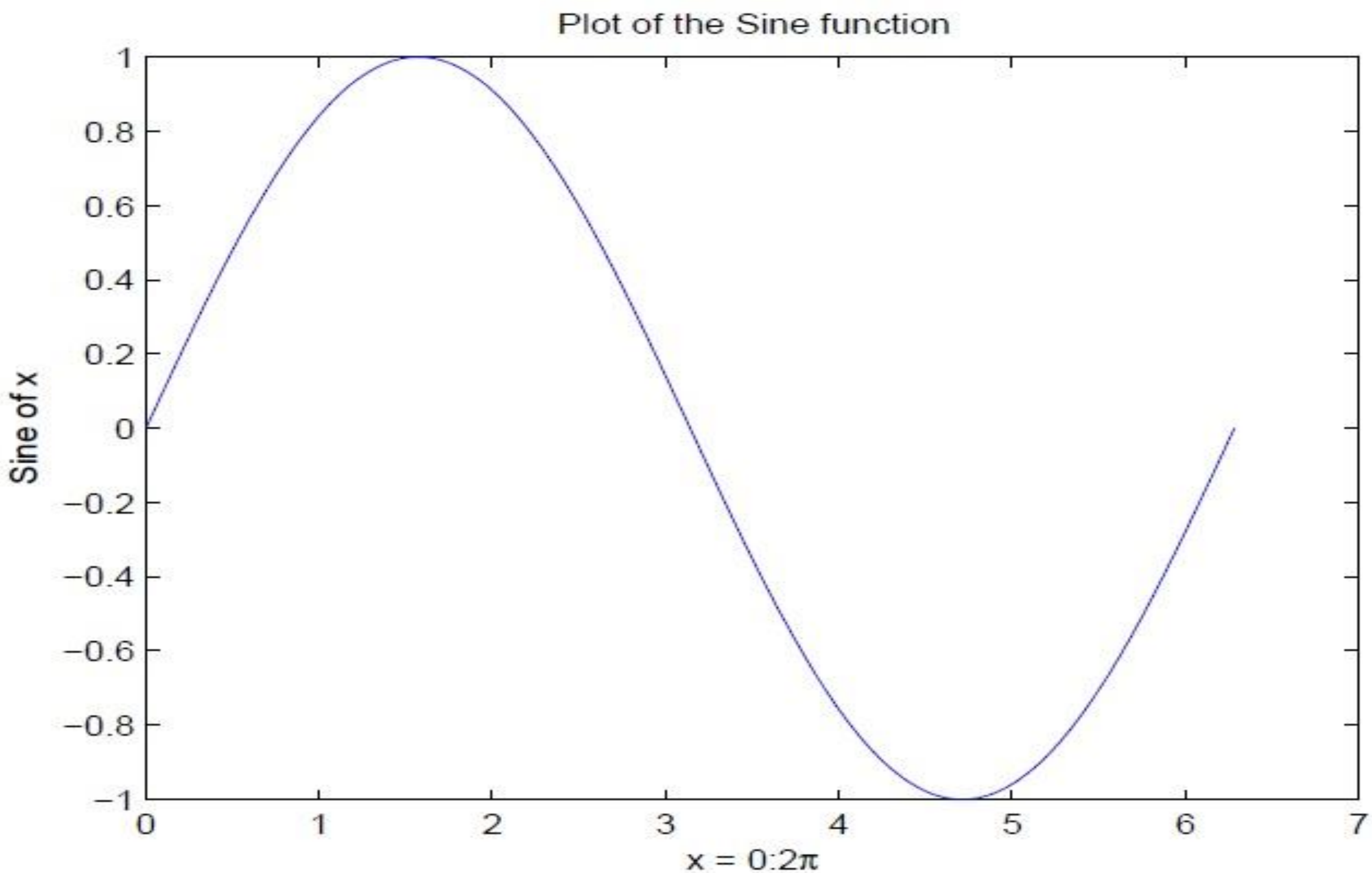


Figure 2.2: Plot of the Sine function

The color of a single curve is, by default, **blue**, but other colors are possible. The desired color is indicated by a third argument. For example, **red** is selected by plot(x,y,'r'). Note the single quotes ,'', around r.

Multiple data sets in one plot

Multiple(x , y) pairs arguments create multiple graphs with a single call to plot. For example, these statements plot three related functions of x : $y_1 = 2\cos(x)$, $y_2 = \cos(x)$, and $y_3 = 0.5\cos(x)$, in the interval $0 \leq x \leq 2\pi$.

```
>> x = 0 : pi/100 : 2*pi ;
>> y1 = 2*cos(x) ;
>> y2 = cos(x) ;
>> y3 = 0.5*cos(x) ;
>> plot ( x , y1 , '--' ,  x , y2 , '-' , x , y3 , ':' )
>> xlabel ( ' 0 \ leq x \ leq 2 \ pi ' )
>> ylabel ( ' Cosine functions ' )
>> legend ( ' 2*cos(x) ' , ' cos(x) ' , ' 0.5*cos(x) ' )
>> title ( ' Typical example of multiple plots ')
>> axis ( [ 0  2*pi   -3  3 ] )
```

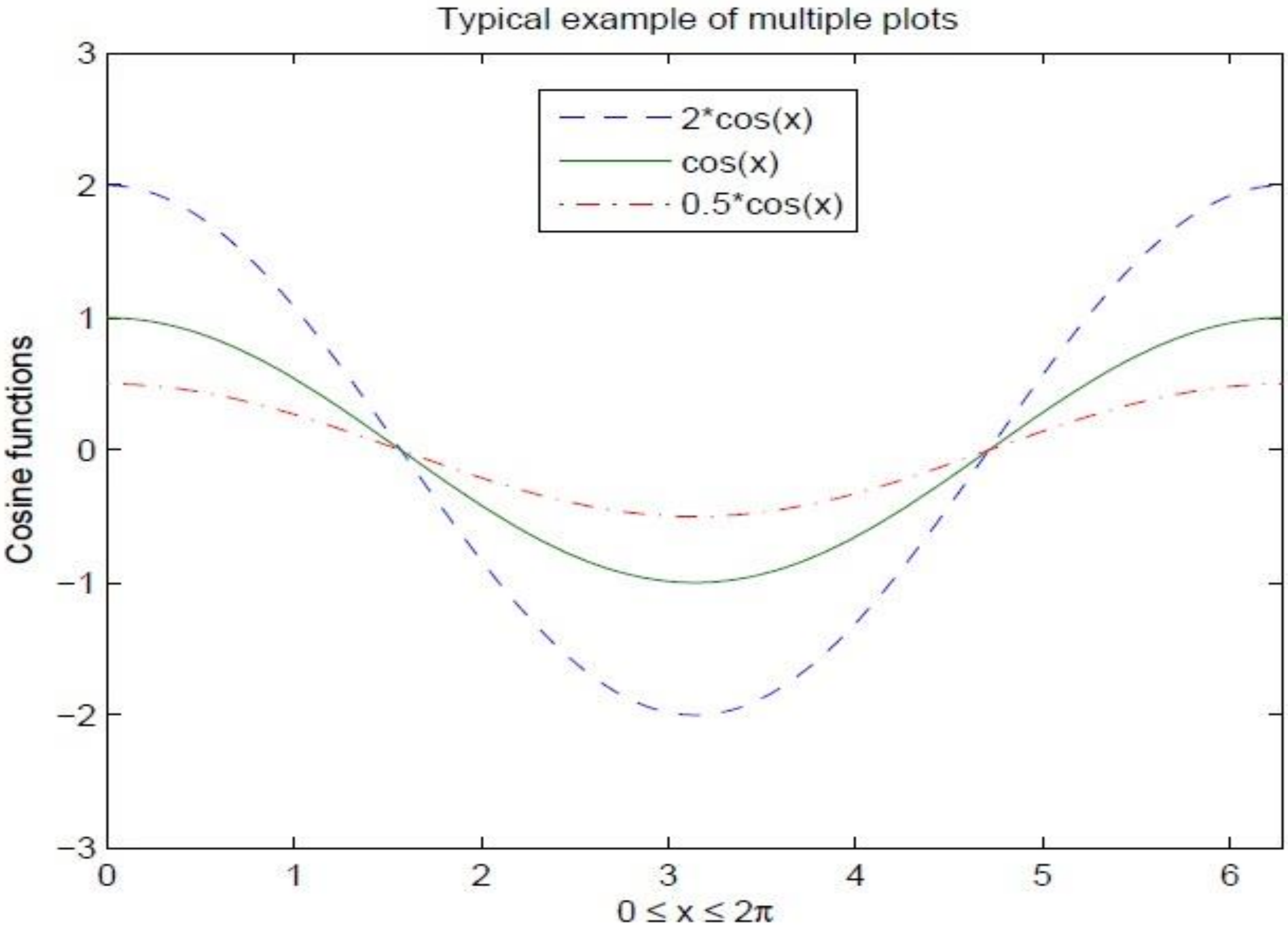


Figure 2.3: Typical example of multiple plots

Specifying line styles and colors

It is possible to specify line styles, colors, and markers (e.g., circles, plus signs,...) using the plot command:

plot (x , y , ' style_color_marker ')

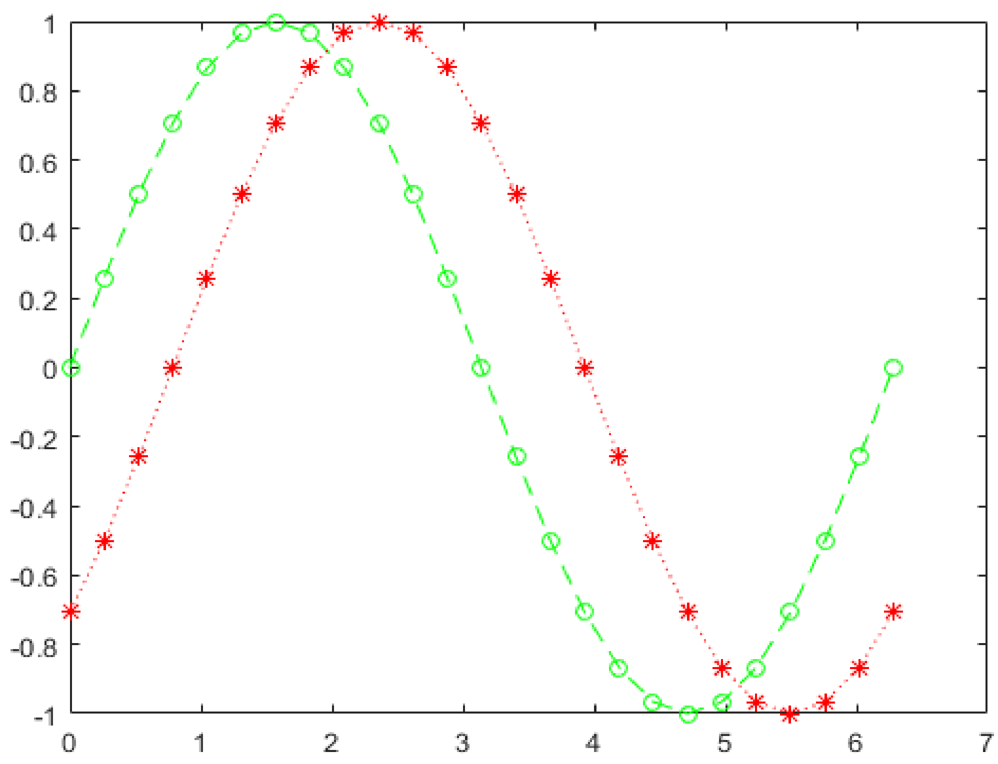
Table 5. Attributes for plot

SYMBOL	COLOR	SYMBOL	LINE STYLE	SYMBOL	MARKER
k	Black	–	Solid	+	Plus sign
r	Red	– –	Dashed	o	Circle
b	Blue	:	Dotted	*	Asterisk
g	Green	–.	Dash-dot	.	Point
c	Cyan	none	No line	×	Cross
m	Magenta			*s*	Square
y	Yellow			*d*	Diamond

Specifying line styles and colors

dashed line and circle markers using '--go'. Plot the second sine wave with a red dotted line and star markers using ':r*'.

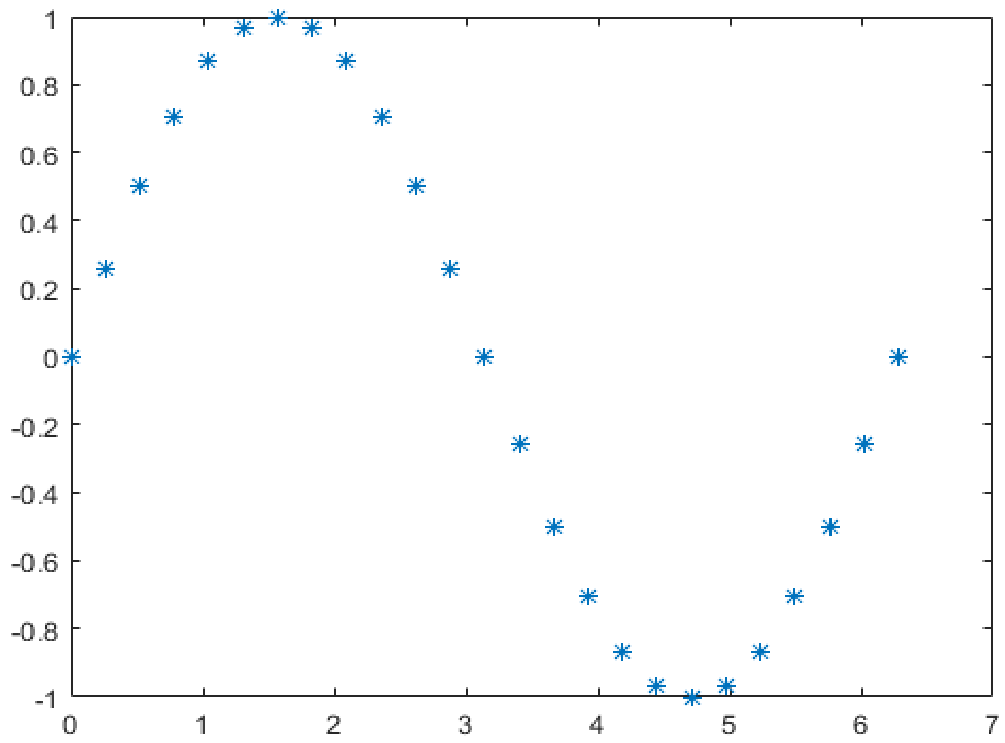
```
x = linspace(0,2*pi,25);
y1 = sin(x);
y2 = sin(x-pi/4);
figure % new figure window
plot(x,y1,'--go',x,y2,':r*')
```



Plot Only Data Points

Define the data x and y. Plot the data and display a star marker at each data point.

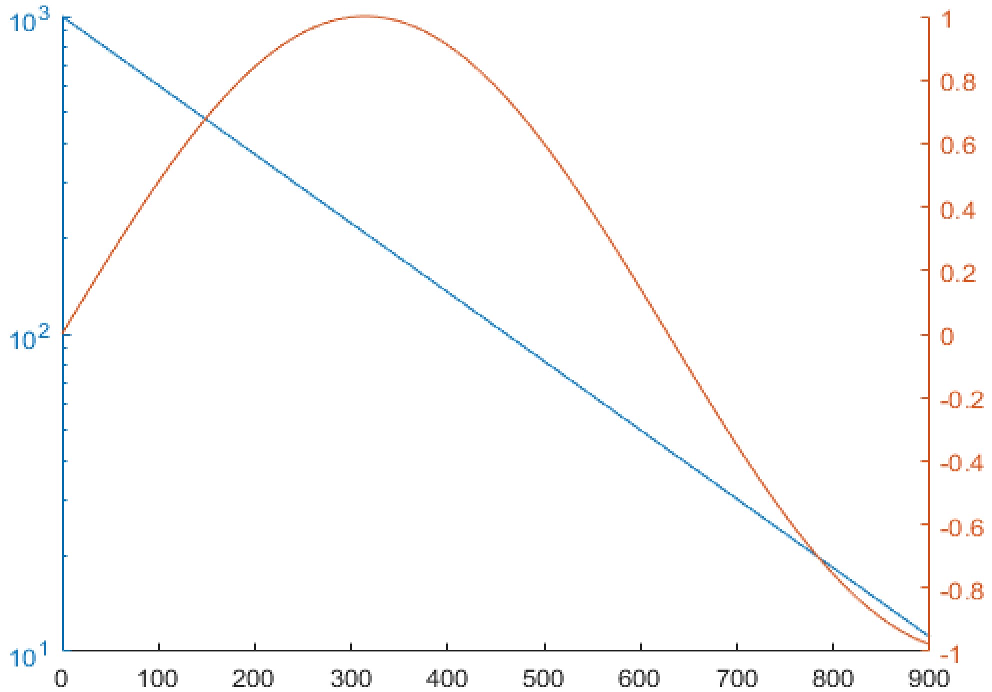
```
>>x = linspace(0,2*pi,25);
  y = sin(x);
  figure
 plot(x,y,'*')
```



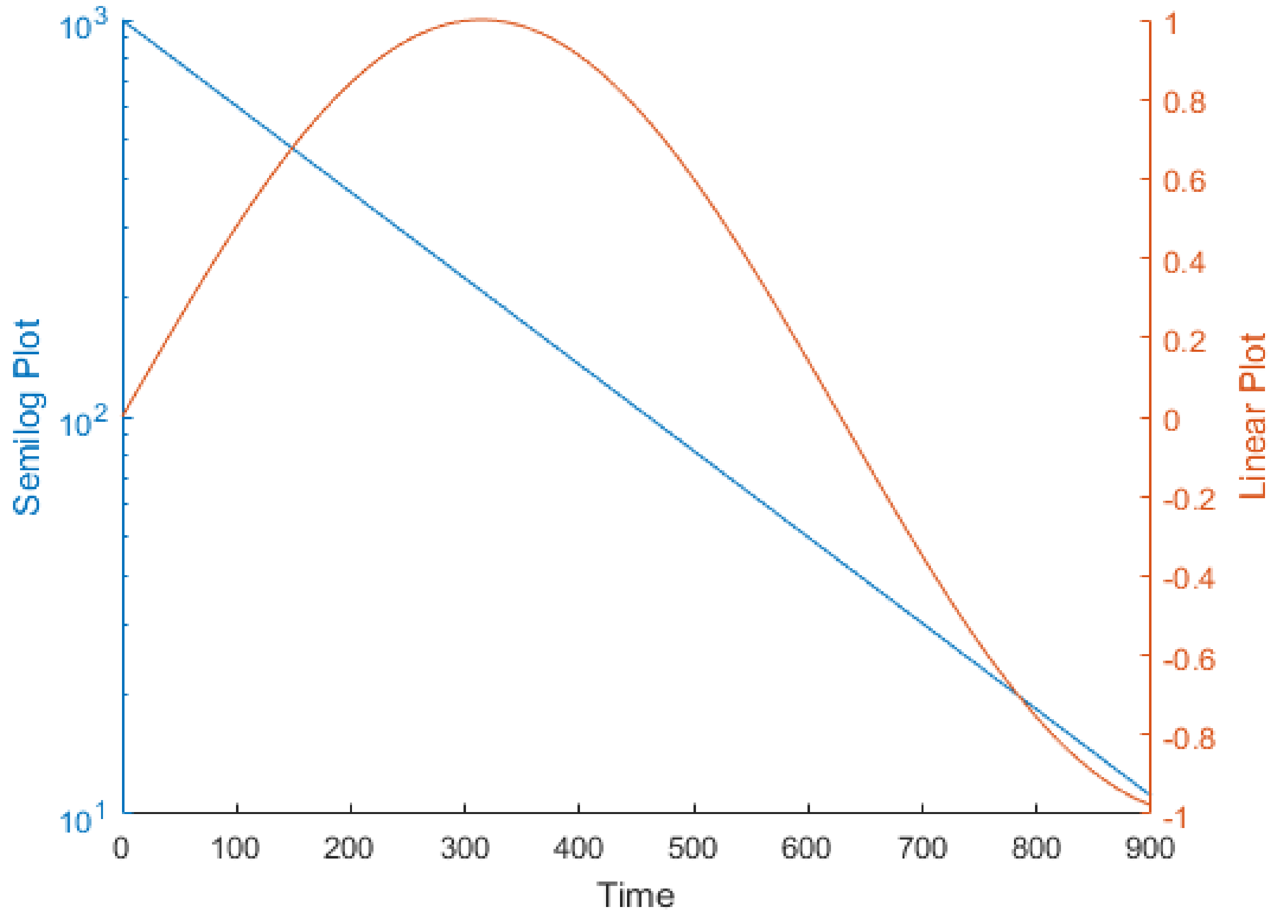
Create Graph with Two y-Axes

Plot z1 versus t using semi-logarithmic scaling. Plot z2 versus t using linear scaling. Return the two axes objects as array ax. Return the two lines as p1 and p2.

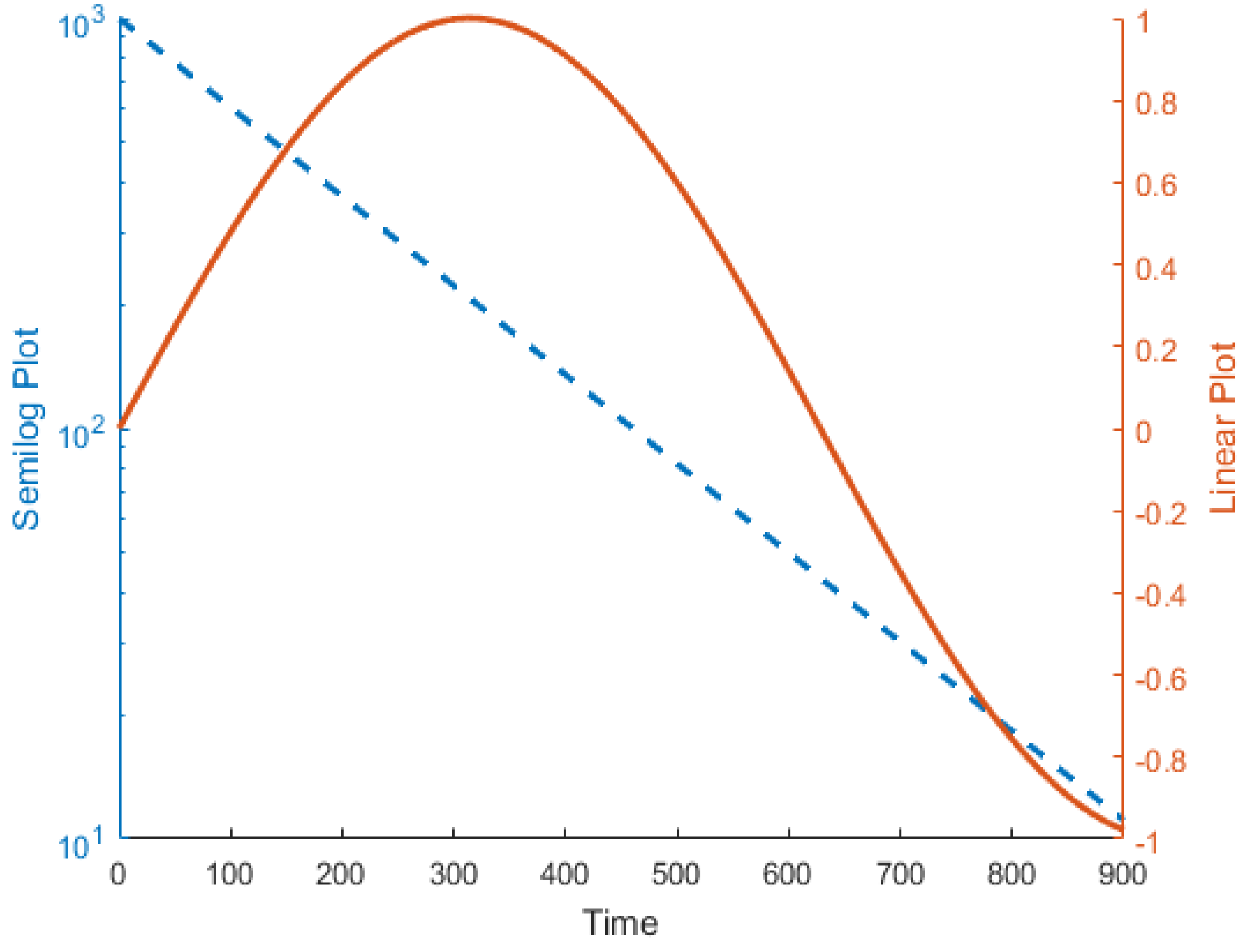
```
A = 1000 ;  a = 0.005 ;  b = 0.005 ;  t = 0:900 ;
z1 = A*exp(-a*t);
z2 = sin(b*t);
[ax,p1,p2] = plotyy(t,z1,t,z2,'semilogy','plot');
```



```
xlabel(ax(1),'Time') % label x-axis
ylabel(ax(1),'Semilog Plot') % label left y-axis
ylabel(ax(2),'Linear Plot') % label right y-axis
```

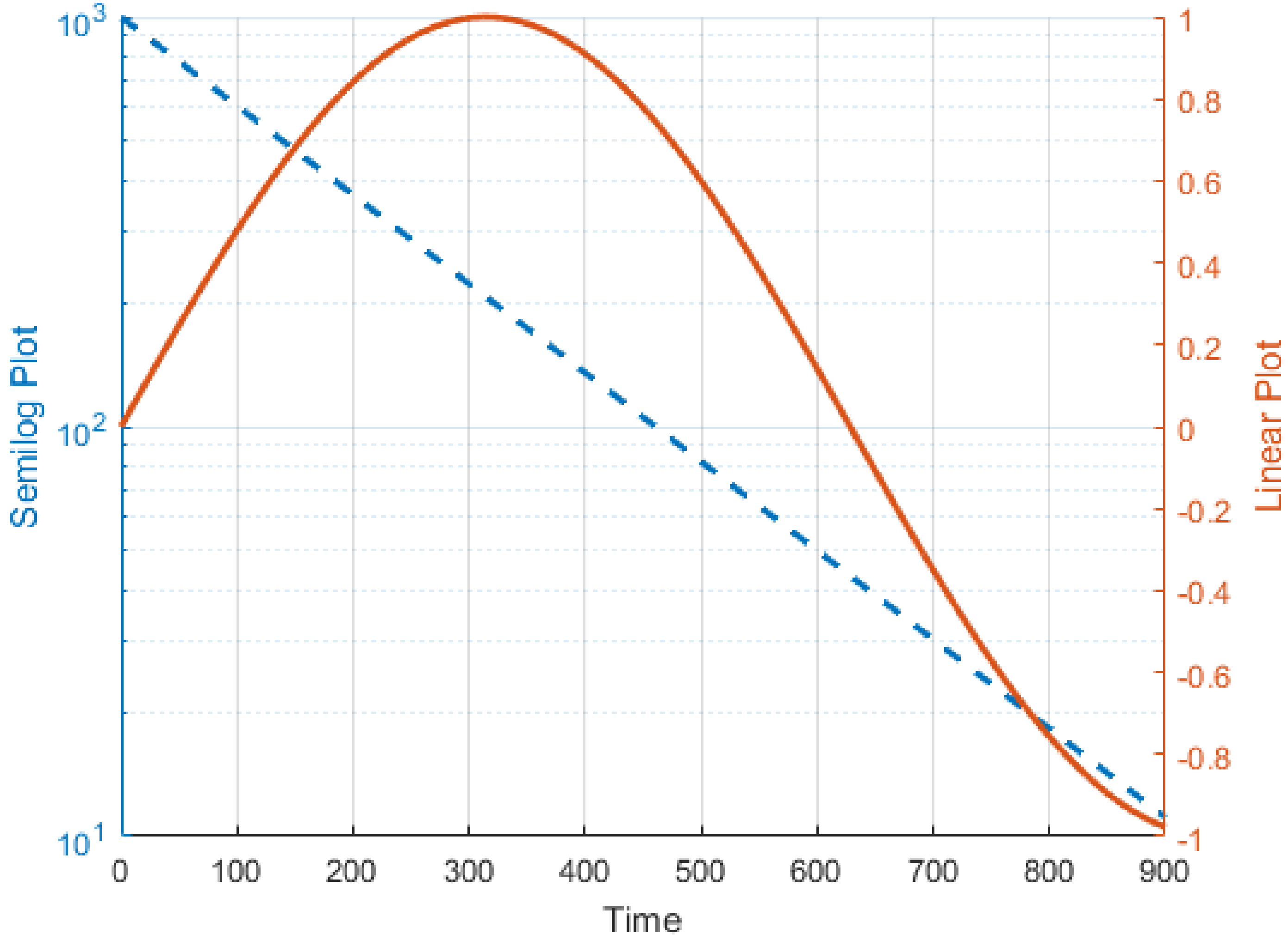


```
p1.LineStyle = '--';
p1.LineWidth = 2;
p2.LineWidth = 2;
```



- the grid function.

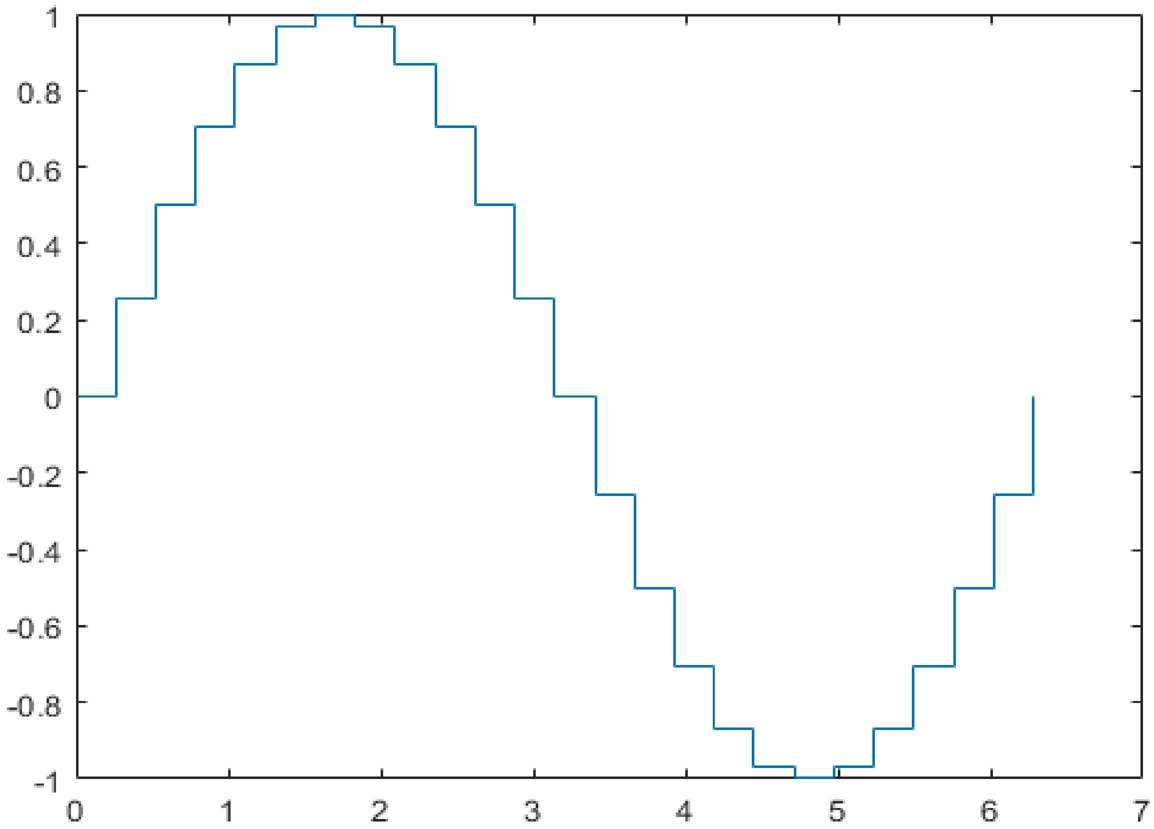
grid(ax(1),'on')



Stair graph

Ex: plot a stair graph for x range (0 to 2π) and y is sine x function.

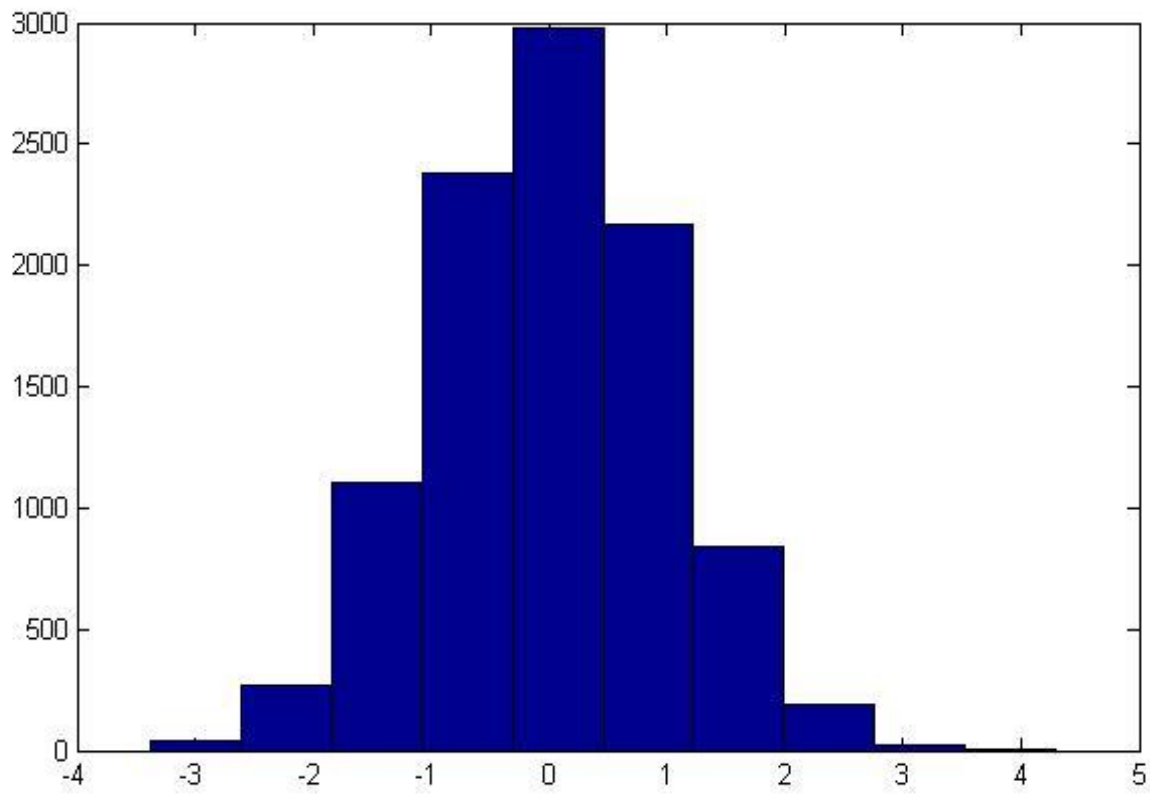
x = linspace (0, 2*pi, 25);
y = sin(x);
figure
stairs(x,y)



Histogram graph

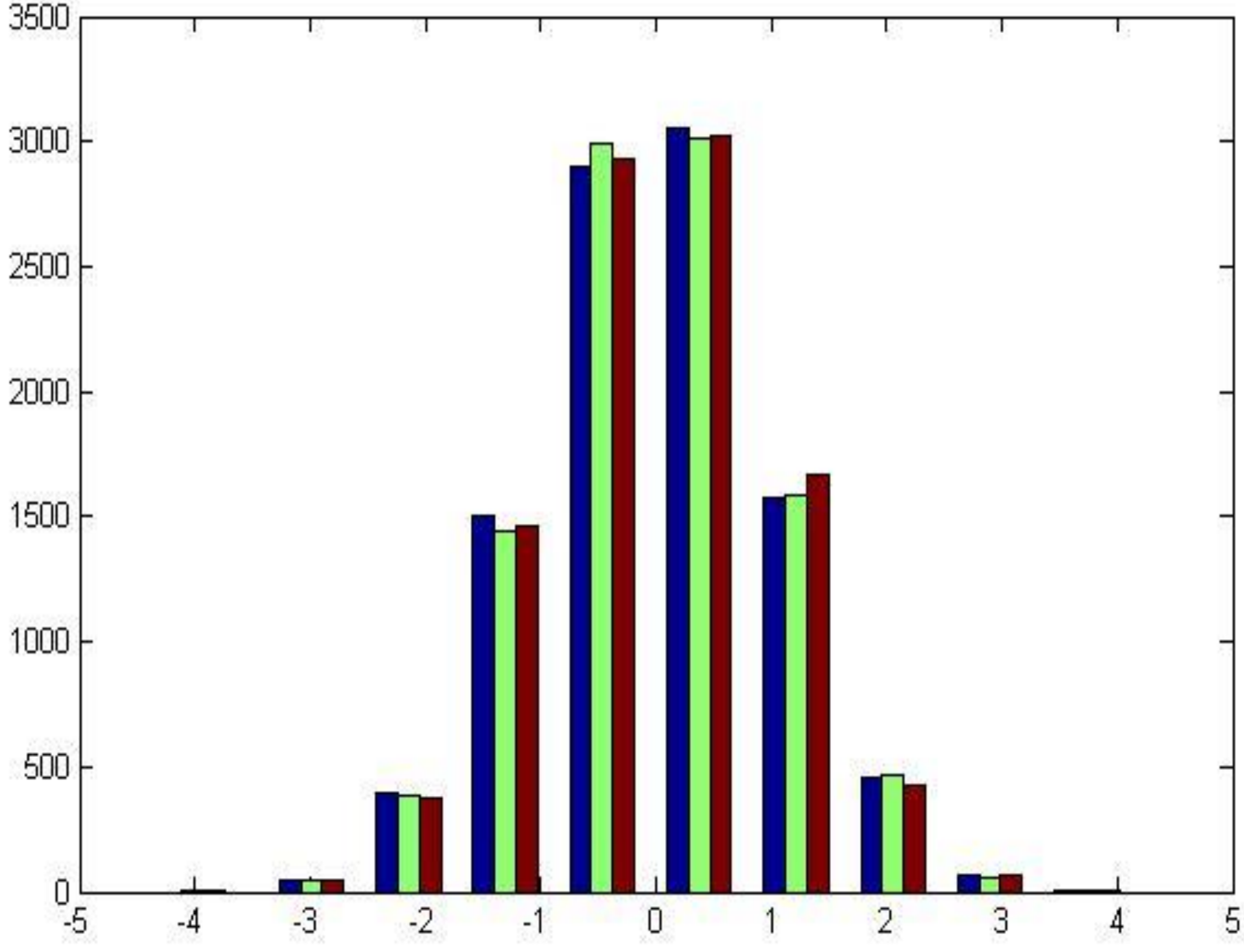
Generates 10,000 random numbers and creates a histogram with 10 bins distributed along the x-axis between the minimum and maximum values of yn.

>> yn = randn(10000,1);
hist(yn)



Matrix Input Argument, when Y is a matrix, hist creates a set of bins for each column, displaying each set in a separate color. The statements

>> Y = randn(10000,3);
hist(Y)

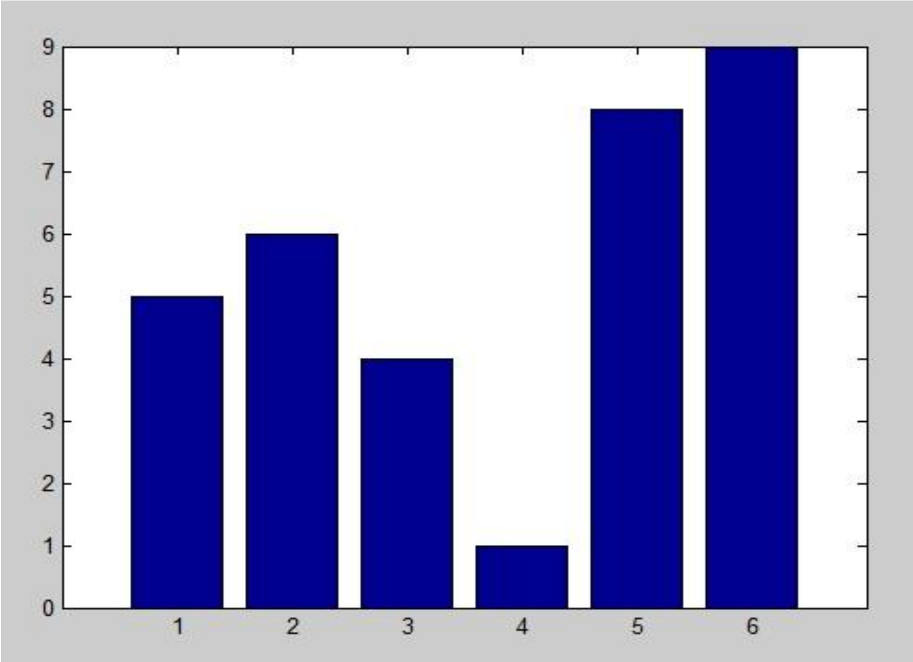


bar graph (vertical and horizontal)

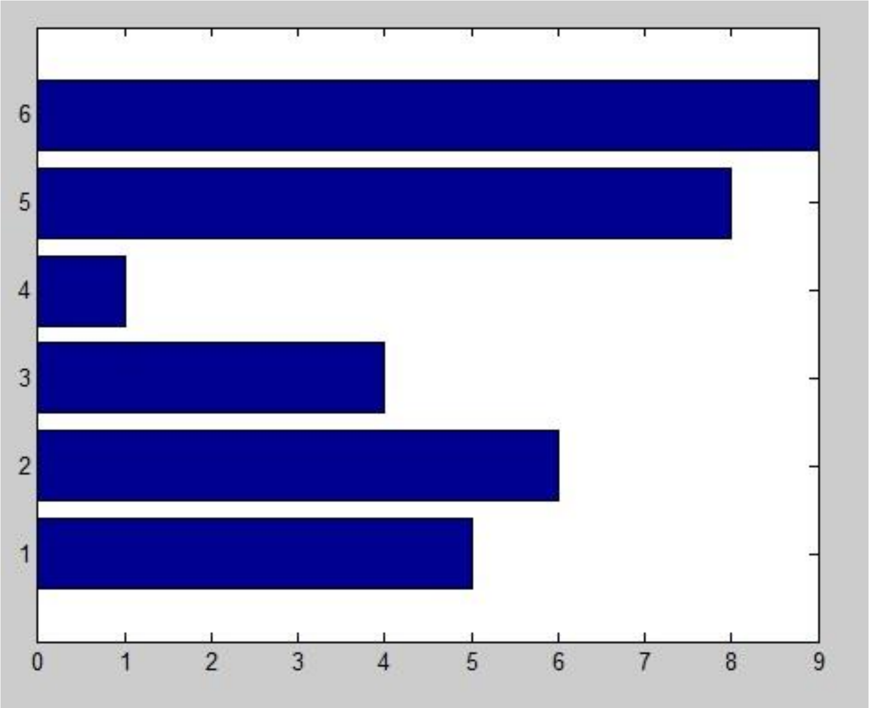
Syntax

bar(Y)
bar(x,Y)
bar(...,width)
bar(...,'style')
bar(...,'bar_color')

```
>> y=[5 6 4 1 8 9];
>> bar(y)
```




```
>> barh(y)
```

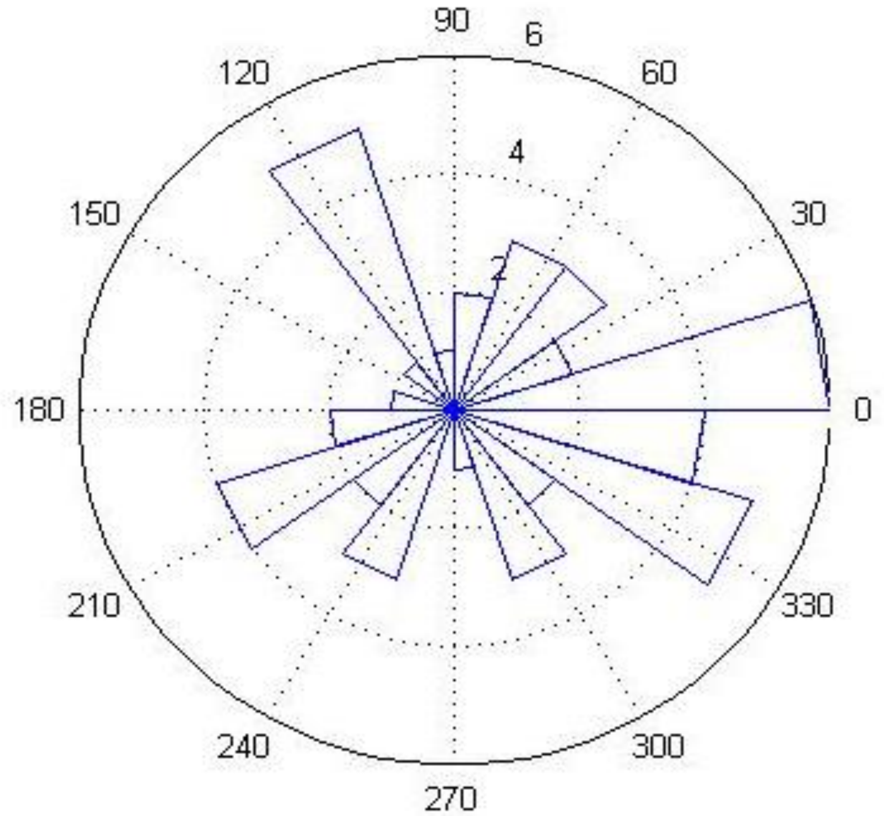


Rose graph

It's an angle histogram plot: rose(theta) creates an angle histogram, which is a polar plot showing the distribution of values grouped according to their numeric range, showing the distribution of theta in 20 angle bins or less.

Ex: Create a rose plot showing the distribution of 50 random numbers.

>> theta = 2*pi*rand(1,50);
rose(theta)

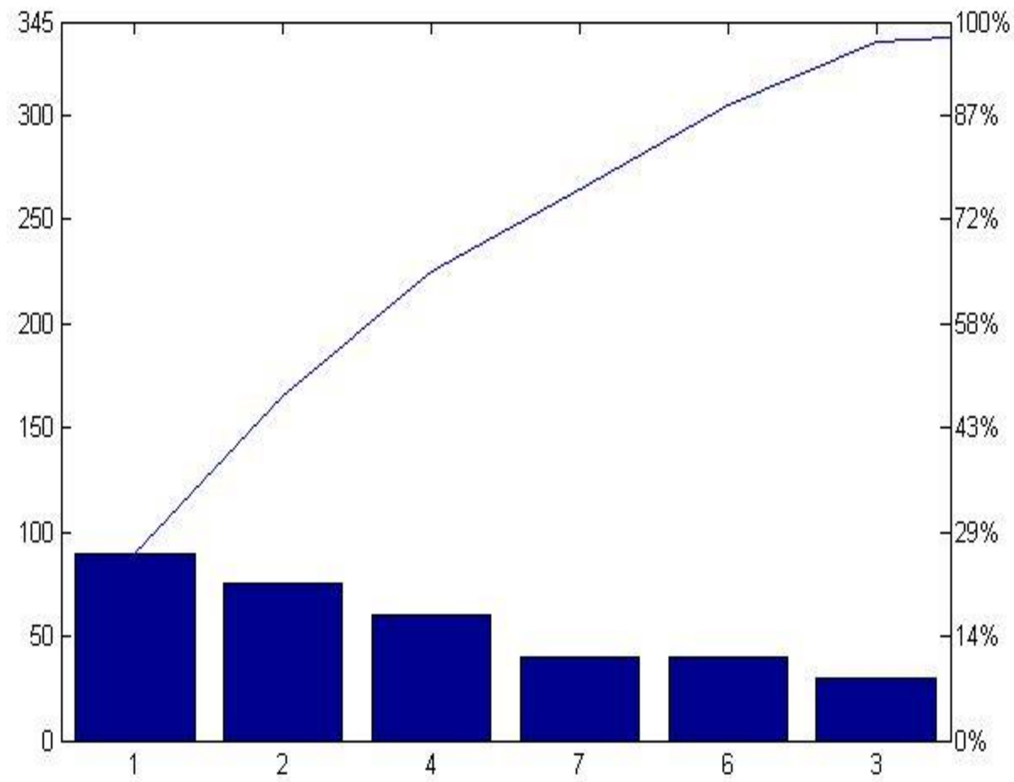


Pareto chart

The Pareto charts display the values in the vector Y as bars drawn in descending order. Values in Y must be nonnegative and not include NaNs. pareto (Y) labels each bar with its element index in Y and also plots a line displaying the cumulative sum of Y.

Ex: Create a Pareto chart of vector y.

```
y = [90,75,30,60,5,40,40,5];
figure
pareto(y)
```

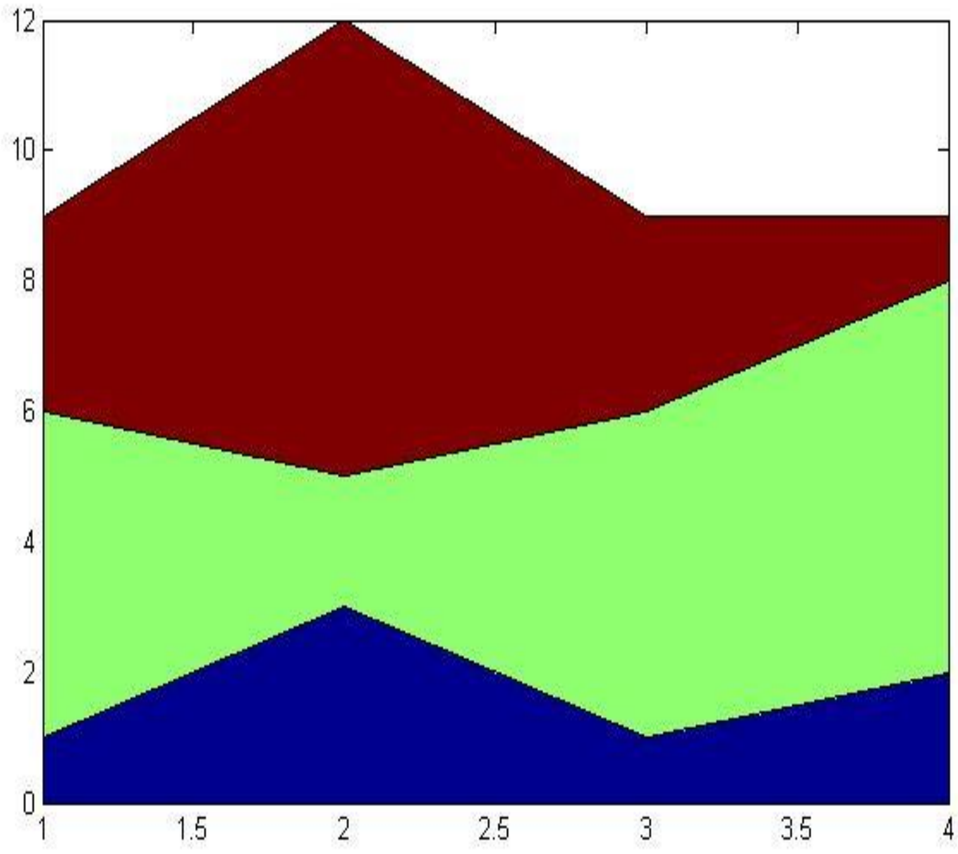


Area graph (2D)

An area graph displays elements in Y as one or more curves and fills the area beneath each curve

Ex: Plot the data in matrix Y as an area graph.

```
Y = [1, 5, 3;
    3, 2, 7;
    1, 5, 3;
    2, 6, 1];
figure
area(Y)
```



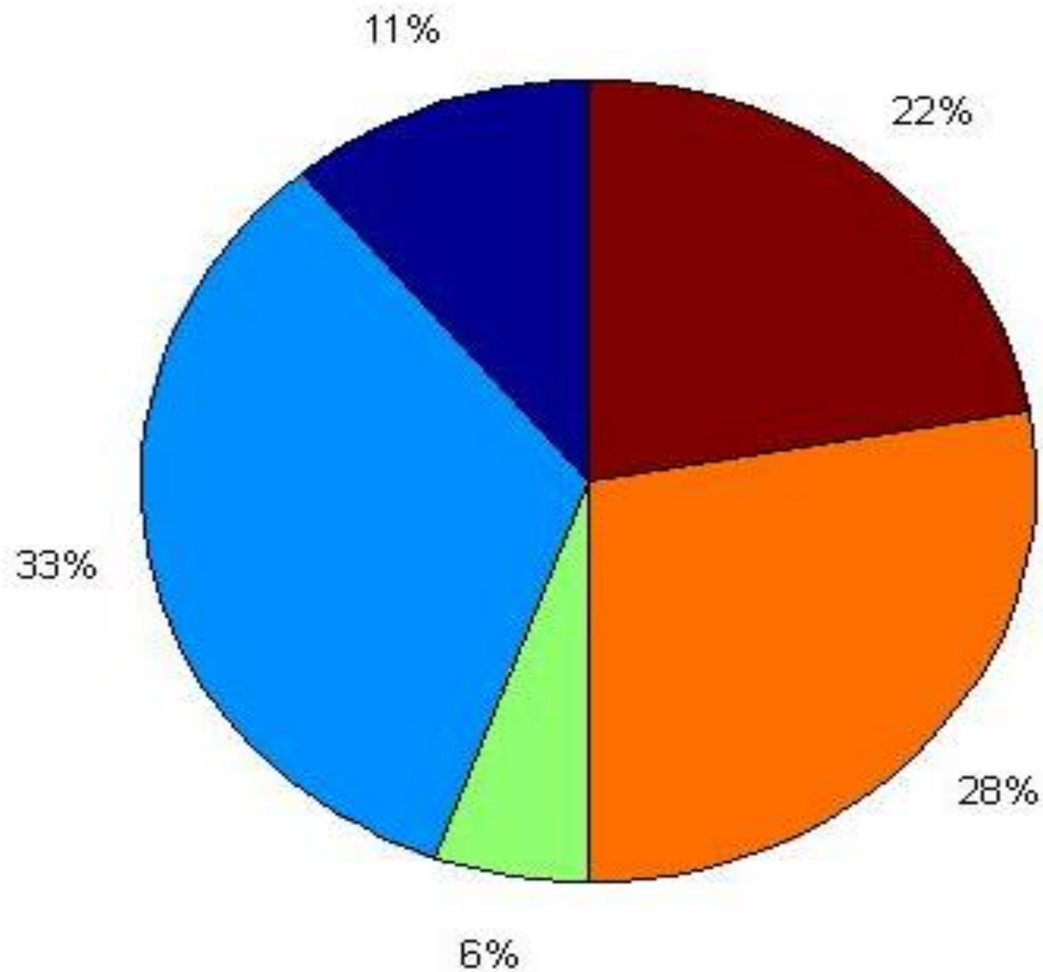
Pie chart

The pie(X) function can draw a pie chart using the data in X. Each slice of the pie chart represents an element in X.

- If sum (X) ≤ 1, then the values in X directly specify the areas of the pie slices. pie draws only a partial pie if sum(X)< 1.
- If sum(X) > 1, then pie normalizes the values by X/sum(X) to determine the area of each slice of the pie.
- If X is of data type categorical, the slices correspond to categories. The area of each slice is the number of elements in the category divided by the number of elements in X.

Ex: Create a pie chart of vector X.

X = [1 3 0.5 2.5 2];
pie(X)


11%
22%
33%
28%
6%

3D Graphs

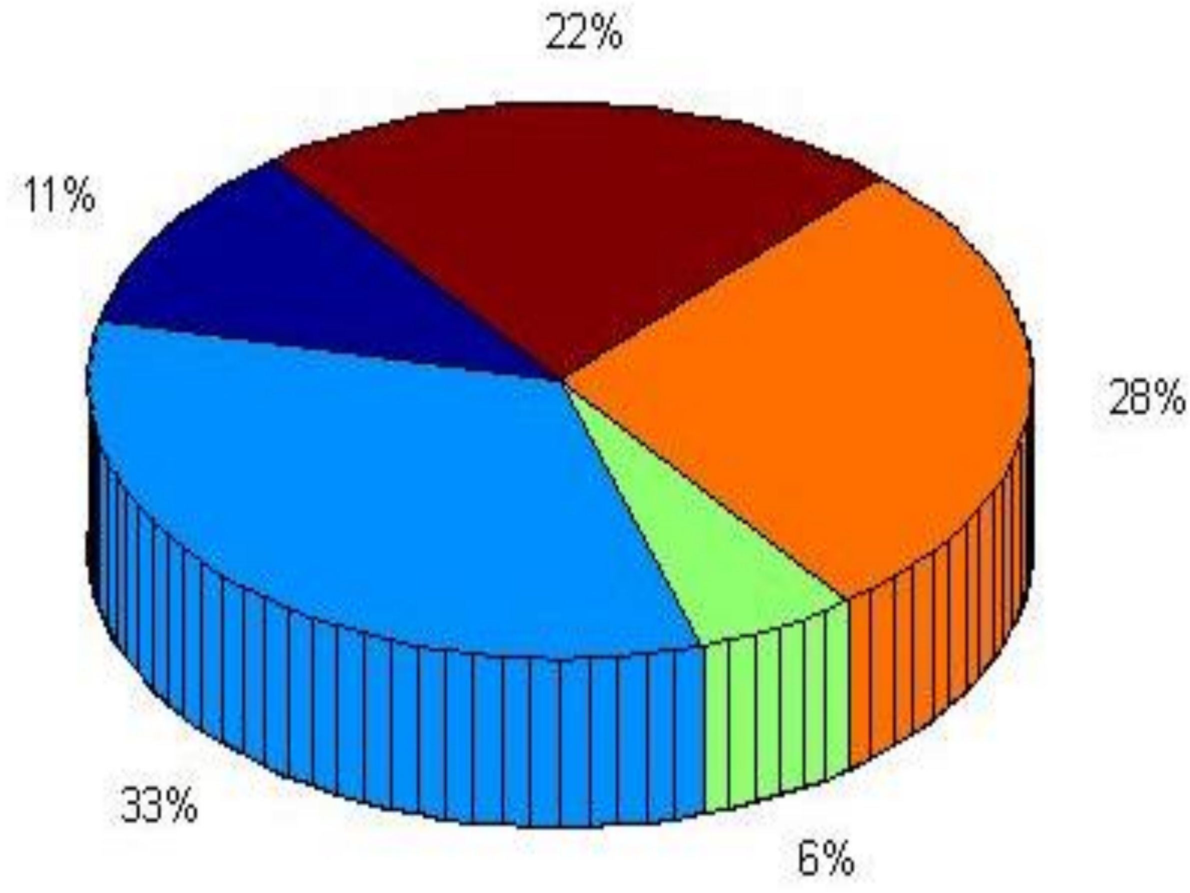
3D pie chart

The function pie3(X) can draw a three-dimensional pie chart using the data in X. Each element in X is represented as a slice in the pie chart.

- ❑ If sum(X) ≤ 1, then the values in X directly specify the area of the pie slices. pie3 draws only a partial pie if sum(X)< 1.
- ❑ If the sum of the elements in X is greater than one, then pie3 normalizes the values by X/sum(X) to determine the area of each slice of the pie.

Ex: Create a 3-D pie chart of vector x.

```
x = [1,3,0.5,2.5,2];
figure
pie3(x)
```

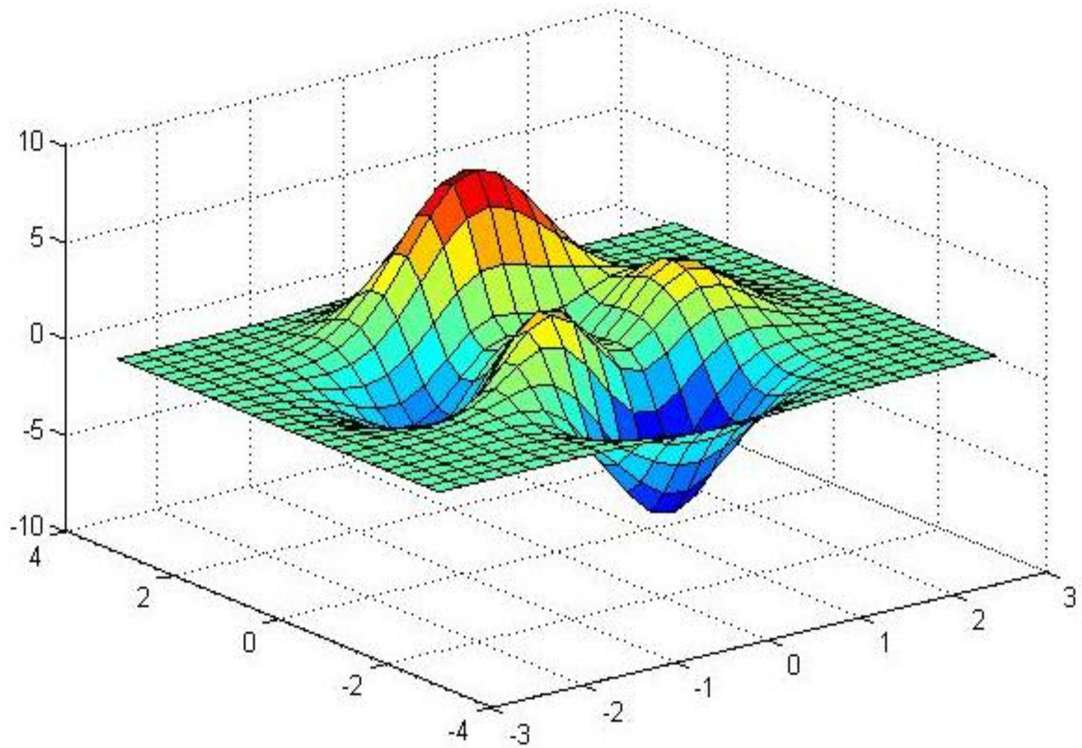

22%
11%
28%
33%
6%

3D shaded surface plot

The surf(Z) function creates a three-dimensional shaded surface from the *z* components in matrix Z, using x = 1:n and y = 1:m, where [m,n] = size(Z). The height, Z, is a single-valued function defined over a geometrically rectangular grid. Z specifies the color data, as well as surface height, so color is proportional to surface height.

Ex: Use the peaks function to define X, Y, and Z as 25-by-25 matrices. Then, create a surface plot.

```
[X,Y,Z] = peaks(25);
figure
surf(X,Y,Z);
```



Animating plot

In this section, we will use only the comet3 function. There are many function in MATLAB can be used in animating plot, but we will focus only on comet3. A comet plot is an animated graph in which a circle (the comet *head*) traces the data points on the screen. The comet *body* is a trailing segment that follows the head. The *tail* is a solid line that traces the entire function. comet3(z) displays a 3-D comet graph of the vector z.

Ex: Create 3-D Comet Graph

t = -10*pi:pi/250:10*pi;
x = (cos(2*t).^2).*sin(t);
y = (sin(2*t).^2).*cos(t);
comet3(x,y,t);