
3D Graphics in MATLAB

We'll introduce different types of plotting in 3D.

MATLAB has different plotting approaches for showing data in 3D:

- **3D line plots** [MATLAB: *plot3*. Plot lines in 3-space]

- **3D mesh plots** [MATLAB: *mesh, meshc, meshz, waterfall*. Make wire-framed surfaces 3D]

- **3D surface plots** [MATLAB: *surf, shading, surfc, surfl, surfnorm,*. Like mesh, with patches filled in with color]

- **3D contour plots** [MATLAB: *contour, contour3, contourf, shading, clabel*. Contour plots in 2 & 3D]

- **3D volume plots** [MATLAB: *slice, isosurface, smooth3, isocaps, isonormals*. Visualizations of fully 3D data sets]

- **3D specialized plots** [MATLAB: *ribbon, quiver, quiver3, fill3, stem3, sphere, cylinder*. Special purpose 3D plotting]

There are many other MATLAB functions that relate to these renderings, including camera and lighting attributes.

3D LINE PLOTS

plot3 Plot lines and points in 3-D space

PLOT3() is a three-dimensional analogue of PLOT().

PLOT3(x,y,z), where x, y and z are three vectors of the same length, plots a line in 3-space through the points whose coordinates are the elements of x, y and z.

PLOT3(X,Y,Z), where X, Y and Z are three matrices of the same size, plots several lines obtained from the columns of X, Y and Z.

Various

line types,

plot symbols and

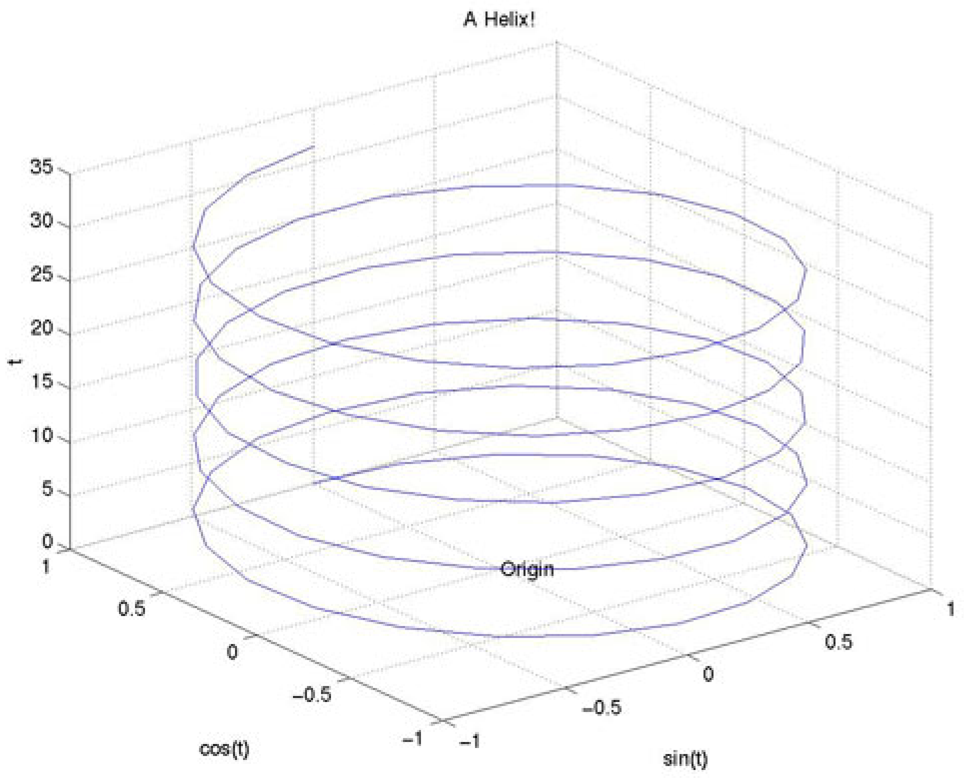
colors

may be obtained with PLOT3(X,Y,Z,s) where s is a 1, 2 or 3 character string made from the characters listed under the PLOT command.

PLOT3(x1,y1,z1,s1,x2, y2, z2, s2, x3, y3, z3, s3,...) combines the plots defined by the (x,y,z,s) fourtuples, where the x's, y's and z's are vectors or matrices and the s's are strings.

helix1.m

```
t = linspace(0,10*pi); %100 points between 0-10Pi
plot3(sin(t),cos(t),t)
xlabel('sin(t)'), ylabel('cos(t)'), zlabel('t')
text(0,0,0,'Origin')
grid on
title('A Helix! ')
```



Ex:

Less points

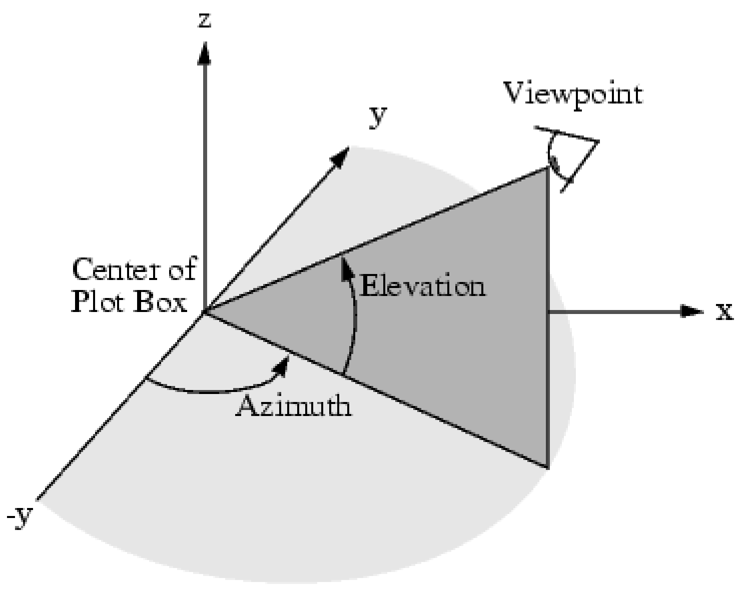
- more points
- diff functions

Azimuth and Elevation

The view command specifies the viewpoint by defining azimuth and elevation with respect to the axis origin.

Azimuth is a polar angle in the x-y plane, with positive angles indicating counter-clockwise rotation of the viewpoint.

Elevation is the angle above (positive angle) or below (negative angle) the x-y plane.



VIEW

3-D graph viewpoint specification.

VIEW(AZ,EL) and VIEW([AZ,EL])

set the angle of the view from which an observer sees the current 3-D plot.

- AZ is the azimuth or horizontal rotation and
- EL is the vertical elevation

(both in degrees).

Azimuth

revolves about the z-axis, with
- positive values indicating counter-clockwise rotation of the viewpoint.

- Positive values of elevation correspond to moving above the object;
- negative values move below.

Some examples:

AZ = -37.5, EL = 30 is the default 3-D view.

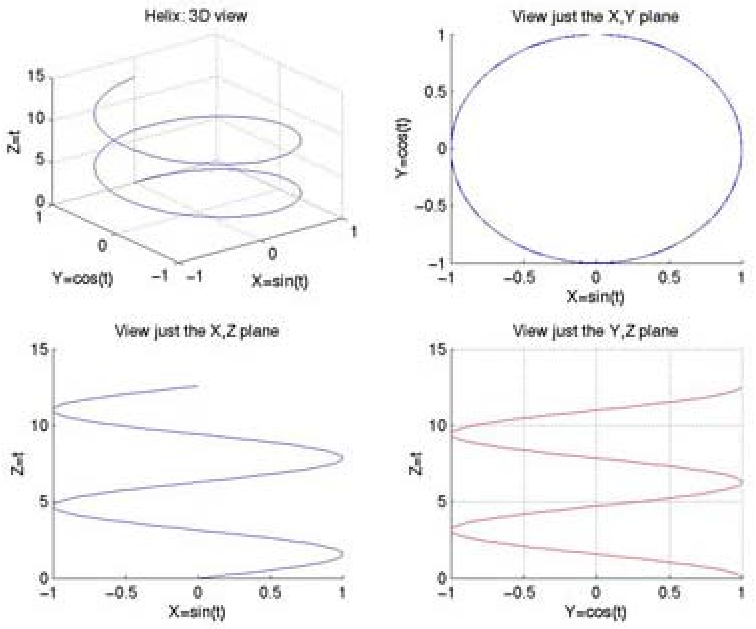
AZ = 0, EL = 90 is directly overhead and the default 2-D view.
AZ = EL = 0 looks directly up the first column of the matrix.
AZ = 180 is behind the matrix.

VIEW(2) sets the default 2-D view, AZ = 0, EL = 90.
VIEW(3) sets the default 3-D view, AZ = -37.5, EL = 30.

[AZ,EL] = VIEW returns the current azimuth and elevation.

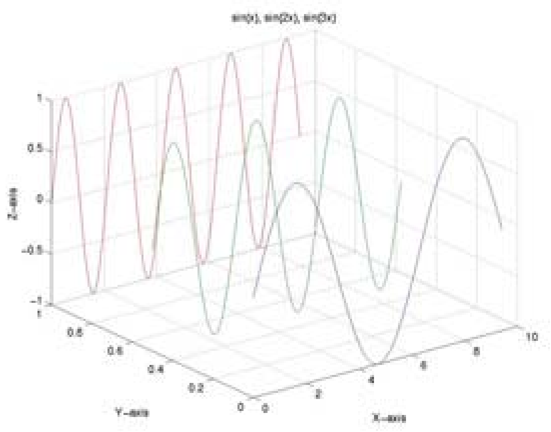
helix2.m

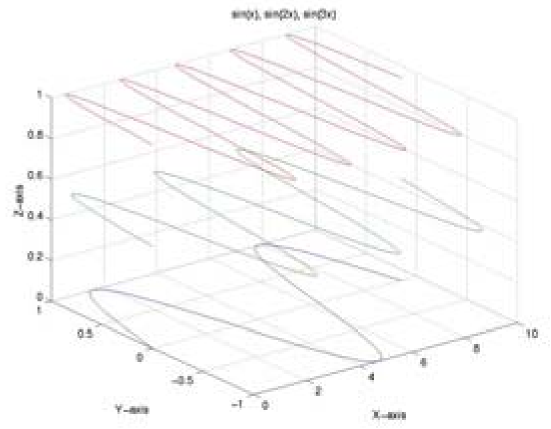
```
clf %clear figure window
t = linspace(0,6*pi,100); %100 points from 0 to
6pi

subplot(2,2,1)
plot3(sin(t),cos(t),t)
xlabel('X=sin(t)'), ylabel('Y=cos(t)'),
zlabel('Z=t')
grid on
title('Helix: 3D view')

subplot(2,2,2)
plot3(sin(t),cos(t),t),view(0,90)
xlabel('X=sin(t)'), ylabel('Y=cos(t)'),
zlabel('Z=t')
title('View just the X,Y plane')

subplot(2,2,3)
plot3(sin(t),cos(t),t),view(0,0)
xlabel('X=sin(t)'), ylabel('Y=cos(t)'),
zlabel('Z=t')
title('View just the X,Z plane')

subplot(2,2,4)
plot3(sin(t),cos(t),t,'r'),view(90,0),grid
xlabel('X=sin(t)'), ylabel('Y=cos(t)'),
zlabel('Z=t')
title('View just the Y,Z plane')
```



sine.m

```
clf %clear figure window
x = linspace(0,3*pi); % x-axis data
z1 = sin(x); % plot in x-z plane
z2 = sin(2*x);
z3 = sin(3*x);
y1 = zeros(size(x)); % spread out along y-axes
y3 = ones(size(x)); % by giving each curve
different y-axis values
y2 = y3/2;
plot3(x,y1,z1,x,y2,z2,x,y3,z3)
grid on
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('sin(x), sin(2x), sin(3x)')
pause
plot3(x,z1,y1,x,z2,y2,x,z3,y3)
grid on
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('sin(x), sin(2x), sin(3x)')
```





Representing a Matrix as a Surface

MATLAB defines a surface by the z-coordinates of points above a rectangular grid in the x-y plane.

The plot is formed by joining adjacent points with straight lines.

Surface plots are useful for
- visualizing matrices that are too large to display in numerical form and
- for graphing functions of two variables.

MATLAB can create different forms of surface plots.

Mesh plots

- are wire-frame surfaces that color only the lines connecting the defining points.

Surface plots

- display both the connecting lines and the faces of the surface in color.

Mesh and Surface Plots

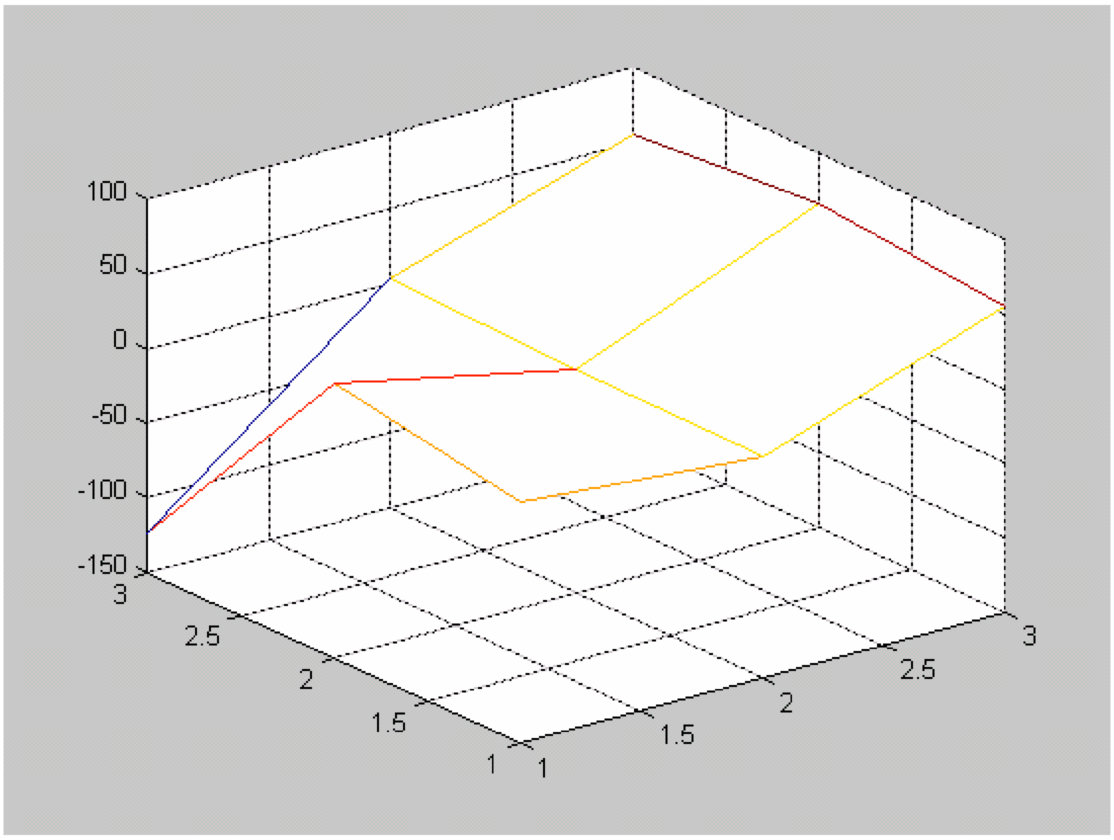
The mesh and surf commands create 3-D surface plots of <u>matrix</u> data.

If Z is a matrix for which the elements Z(i,j) define the height of a surface over an underlying (i,j) grid, then mesh(Z) generates a colored, wire-frame view of the surface and displays it in a 3-D view.

For example :

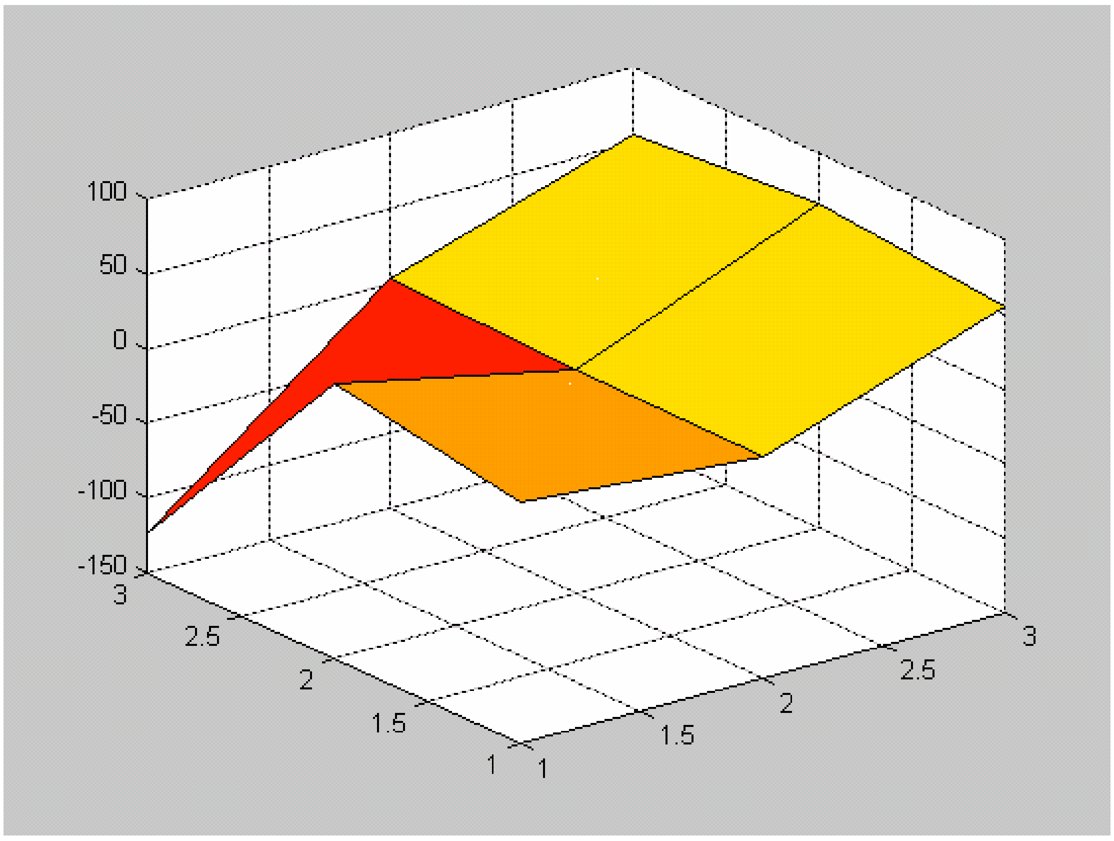
```
x =

   12    -1    55
   34    -1    66
 -123     3    56

>> mesh(x)
```

we'll get :



Similarly, surf(Z) generates a colored, faceted view of the surface and displays it in a 3-D view.

```
>>surf(x)
```

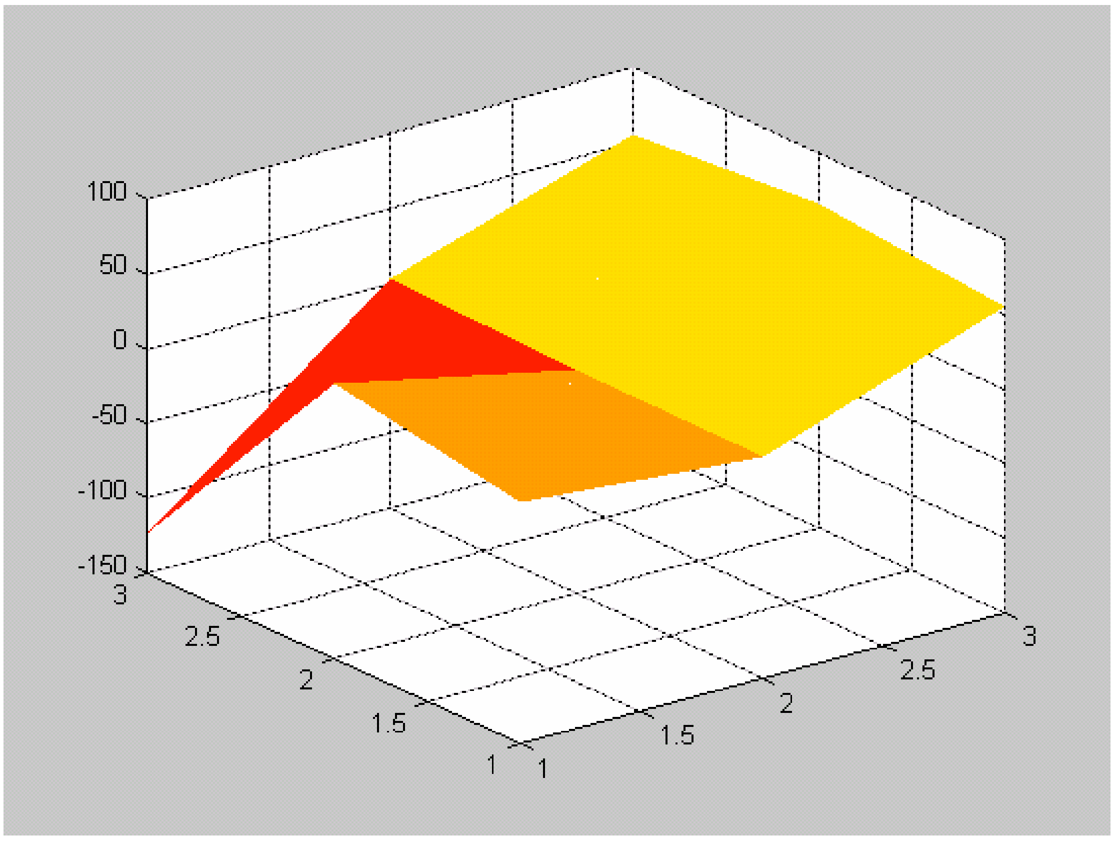


Ordinarily,

- the facets are quadrilaterals,
- each of which is a constant color,
- outlined with black mesh lines,

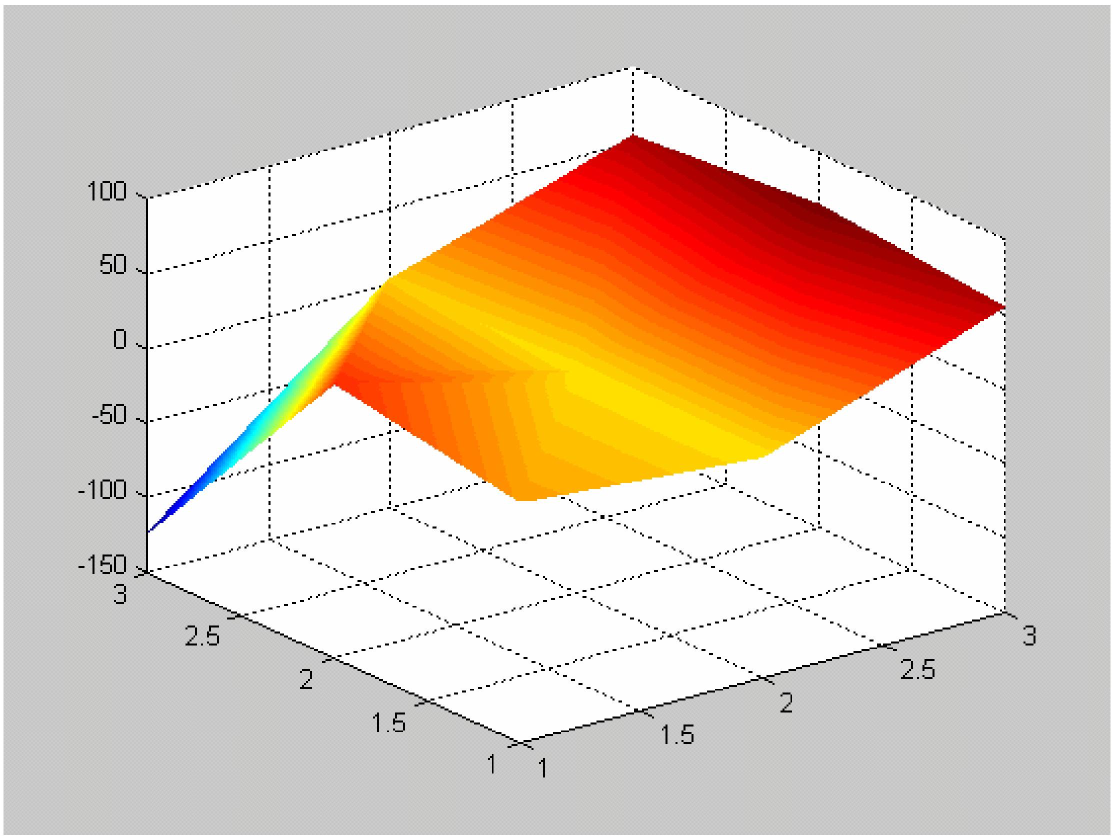
but

- the shading command allows you to eliminate the mesh lines (shading flat)



or

- to select interpolated shading across the facet (shading interp).



Surface object properties provide additional control over the visual appearance of the surface.

You can specify

- edge line styles,
- vertex markers,
- face coloring,
- lighting characteristics, and so on.

EX:
- create a 100x100 positive random matrix and display it in diff formats with mesh, surf
- similar but with pos/negative values .

Parametric Surfaces

The functions that draw surfaces can take two additional vector or matrix arguments to describe surfaces with specific x and y data.

If

- Z is an m-by-n matrix,
- x is an n-vector, and
- y is an m-vector,

then mesh(x,y,Z,C) describes a mesh surface with

- vertices having color C(i,j) and
- located at the points (x(j), y(i), Z(i,j))

where

- x corresponds to the columns of Z and
- y to its rows.

More generally,
if X, Y, Z, and C are matrices of the same dimensions,
then mesh(X,Y,Z,C) describes a mesh surface with vertices

- having color C(i,j) and
- located at the points (X(i,j), Y(i,j), Z(i,j))

Exercise:

- Create 4 100x100 pos/neg random matrices, use them with the prev mesh.
- Create 4 100x100 pos/neg matrices, whose values are drawn by 3 diff functions who deposit their output into the matrices cells. Use them with the prev mesh.

colormap
Set and get the current colormap

Syntax
colormap(map)
colormap('default')
cmap = colormap

Description

A colormap is an m-by-3 matrix of real numbers between 0.0 and 1.0.

Each row is an RGB vector that defines one color.

The kth row of the colormap defines the k-th color, where

```
map(k,:) = [r(k) g(k) b(k)])
```

specifies the intensity of red, green, and blue.

```
colormap(map)
```

sets the colormap to be the matrix map.

If any values in map are outside the interval [0 1], MATLAB returns the error:

```
Colormap must have values in [0,1].
```

```
colormap('default')
```

sets the current colormap to the default colormap.

Hadamard matrix

Syntax

```
H = hadamard(n)
```

Description

H = hadamard(n) returns the Hadamard matrix of order n.

Definition

Hadamard matrices are matrices of 1's and -1's whose columns are orthogonal,

```
H'*H = n*I
```

where [n n] = size(H) and I = eye(n,n). (eye : identity matrix)

They have applications in several different areas, including combinatorics, signal processing, and numerical analysis.

What is n ?
An n-by-n Hadamard matrix with n > 2 exists only if rem(n,4) = 0.

This function handles only the cases where n, n/12, or n/20 is a power of 2.

Examples
The command hadamard(4) produces the 4-by-4 matrix:

```
1   1   1   1
1  -1   1  -1
1   1  -1  -1
1  -1  -1   1
```

The command axis square
makes the x- and y-axes equal in length.

Example - Sphere

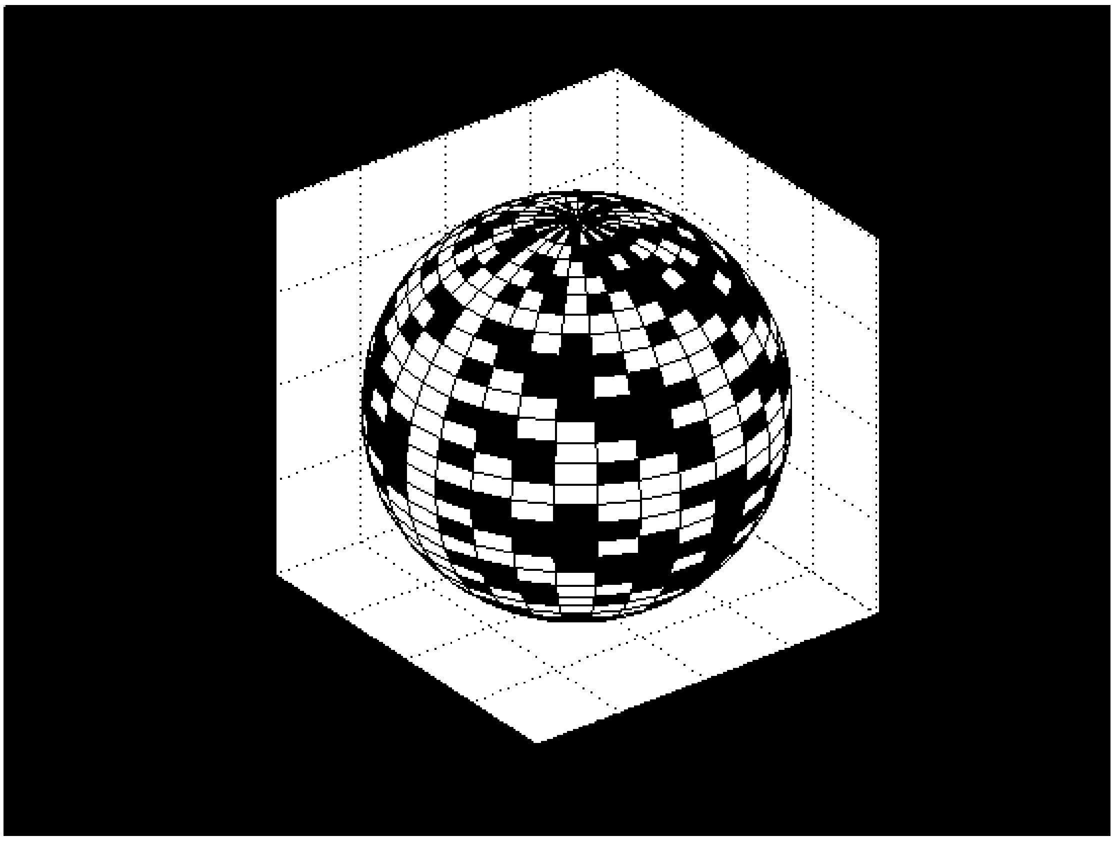
- uses spherical coordinates to draw a sphere and
- color it with the pattern of pluses and minuses in a Hadamard matrix, an orthogonal matrix used in signal processing coding theory.

The vectors theta and phi are in the range
-Pi <= theta <=Pi , and
-Pi/2 <= phi <= Pi/2.

Because theta is a row vector and phi is a column vector, the multiplications that produce the matrices X, Y, and Z are vector outer products. k = 5;

```
k=5;
n = 2^k-1;
theta = pi*(-n:2:n)/n;
```

```
phi = (pi/2)*(-n:2:n)'/n;
X = cos(phi)*cos(theta);
Y = cos(phi)*sin(theta);
Z = sin(phi)*ones(size(theta));
colormap([0 0 0;1 1 1])
C = hadamard(2^k);
surf(X,Y,Z,C)
axis square
```



3D MESH PLOTS

mesh 3D mesh surface

<u>MESH(X,Y,Z,C)</u> plots the colored parametric mesh defined by four matrix arguments. The view point is specified by VIEW.

The axis labels are determined by the range of X, Y and Z, or by the current setting of AXIS.

The color scaling is determined by the range of C, or by the current setting of CAXIS.

The scaled color values are used as indices into the current COLORMAP. MESH(X,Y,Z) uses C = Z, so color is proportional to mesh height.

meshc 3D mesh surface with contour plot underneath

MESHC(...) is the same as MESH(...) except that a contour plot is drawn beneath the mesh. Because CONTOUR does not handle irregularly spaced data, this routine only works for surfaces defined on a rectangular grid.

meshgrid
Generate X and Y matrices for three-dimensional plots

Syntax
[X,Y] = meshgrid(x,y)
[X,Y] = meshgrid(x)
[X,Y,Z] = meshgrid(x,y,z)

Description
[X,Y] = meshgrid(x,y) transforms the domain specified by vectors x and y into arrays X and Y, which can be used to evaluate :
- functions of two variables and
- three-dimensional mesh/surface plots.

- The rows of the output array X are copies of the vector x;
- The columns of the output array Y are copies of the vector y.

    [X,Y] = meshgrid(x)
is the same as
    [X,Y] = meshgrid(x,x).

[X,Y,Z] = meshgrid(x,y,z) produces three-dimensional arrays used to evaluate :
- functions of three variables and
- three-dimensional volumetric plots.

Ex: Check-
- what are the dimensions of X,Y,Z
- what are their contents ?

Examples

```
[X,Y] = meshgrid(1:3,10:14)

X =

    1    2    3
    1    2    3
    1    2    3
    1    2    3
    1    2    3

Y =

   10   10   10
   11   11   11
   12   12   12
   13   13   13
   14   14   14
```

PEAKS
A sample function of two variables.

PEAKS is a function of two variables, obtained by translating and
scaling Gaussian distributions, which is useful for demonstrating

MESH, SURF, etc.

There are several variants of the calling sequence:

```
        Z = PEAKS;
        Z = PEAKS(N);
        Z = PEAKS(V);
        Z = PEAKS(X,Y);

        PEAKS; % produces a 49-by-49 matrix.
        PEAKS(N);% produces an N-by-N matrix.
        PEAKS(V);% produces an N-by-N matrix where N =
length(V).
        PEAKS(X,Y);% evaluates the function at the given X and
Y,
                          which must be the same size.
                          The resulting Z is also that size.
```

The next variants, with no output arguments, do a SURF plot of the result.

```
        [X,Y,Z] = PEAKS;
        [X,Y,Z] = PEAKS(N);
        [X,Y,Z] = PEAKS(V);
```

Those variants also produce two matrices, X and Y, for use in commands such as SURF(X,Y,Z,DEL2(Z)).

If not given as input, the underlying matrices X and Y are

```
        [X,Y] = MESHGRID(V,V)
```

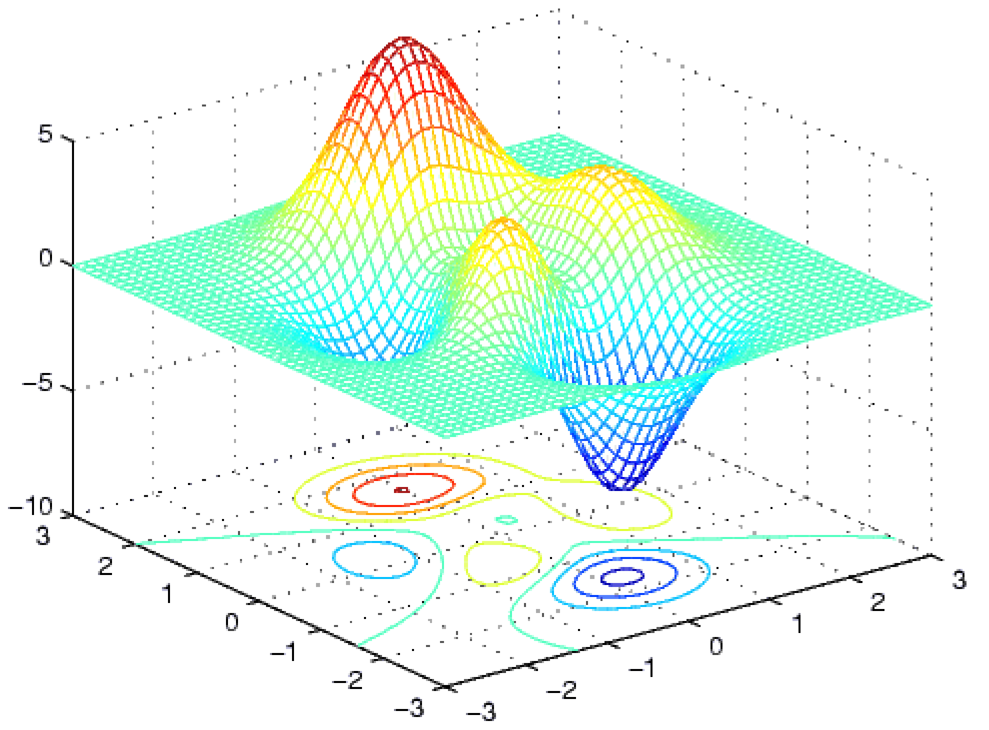
where V is a given vector, or V is a vector of length N with elements equally spaced from -3 to 3.

If no input argument is given, the default N is 49.

Examples

Produce a combination mesh and contour plot of the peaks surface:

- `[X,Y] = meshgrid(-3:.125:3);`
- `Z = peaks(X,Y);`
- `meshc(X,Y,Z);`
- `axis([-3 3 -3 3 -10 5])` `%specifies the scales of all 3 axes`



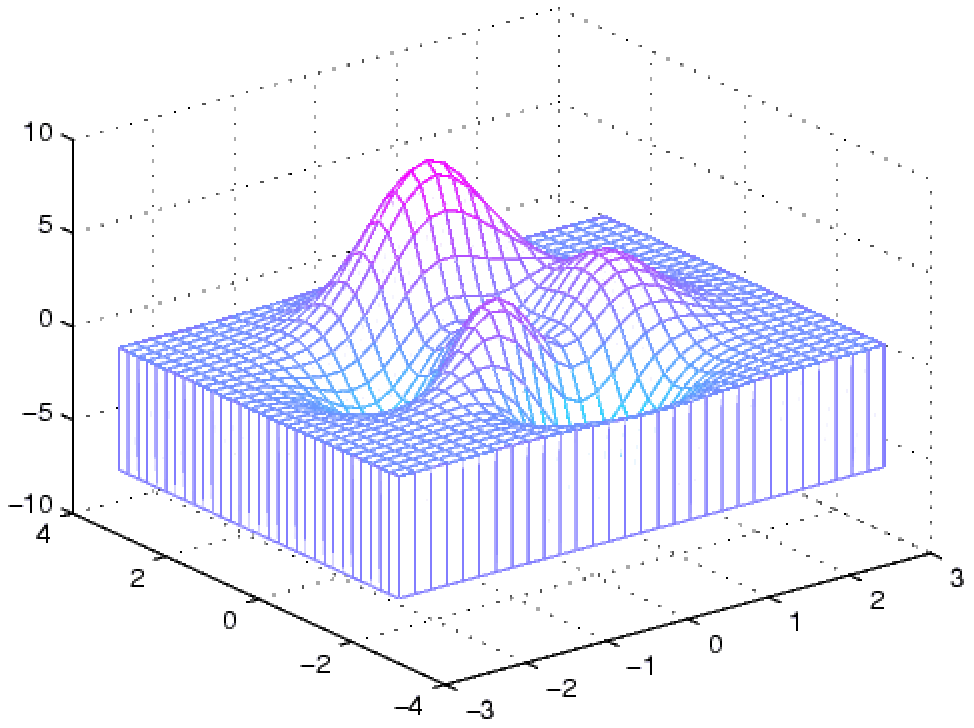
meshz 3D mesh surface with a curtain

MESHZ(...) is the same as MESH(...) except that a "curtain" or reference plane is drawn beneath. This routine only works for surfaces defined on a rectangular grid.

Generate the curtain plot for the peaks function:

- `[X,Y] = meshgrid(-3:.125:3);`
- `Z = peaks(X,Y);`

- `meshz(X,Y,Z)`



waterfall like meshz with lines in x-direction only

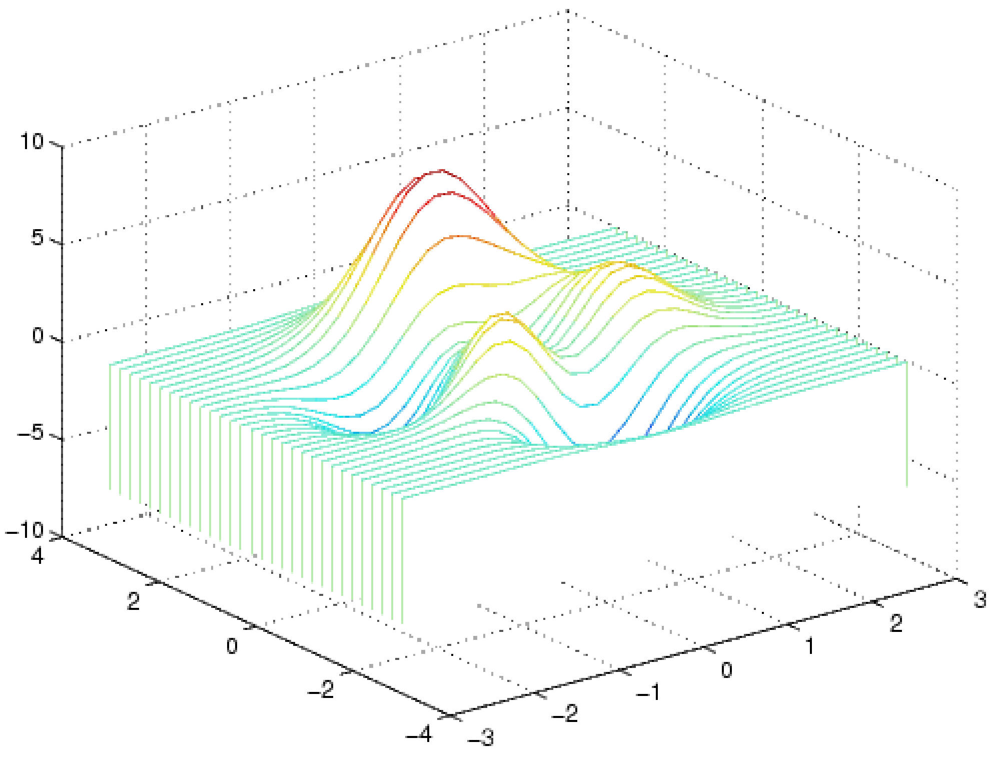
<u>WATERFALL(...)</u> is the same as MESH(...) except that the column lines of the mesh are not drawn - thus producing a "waterfall" plot. For column-oriented data analysis, use WATERFALL(Z') or WATERFALL(X',Y',Z').

Examples

Produce a waterfall plot of the peaks function.

- `[X,Y,Z] = peaks(30);`

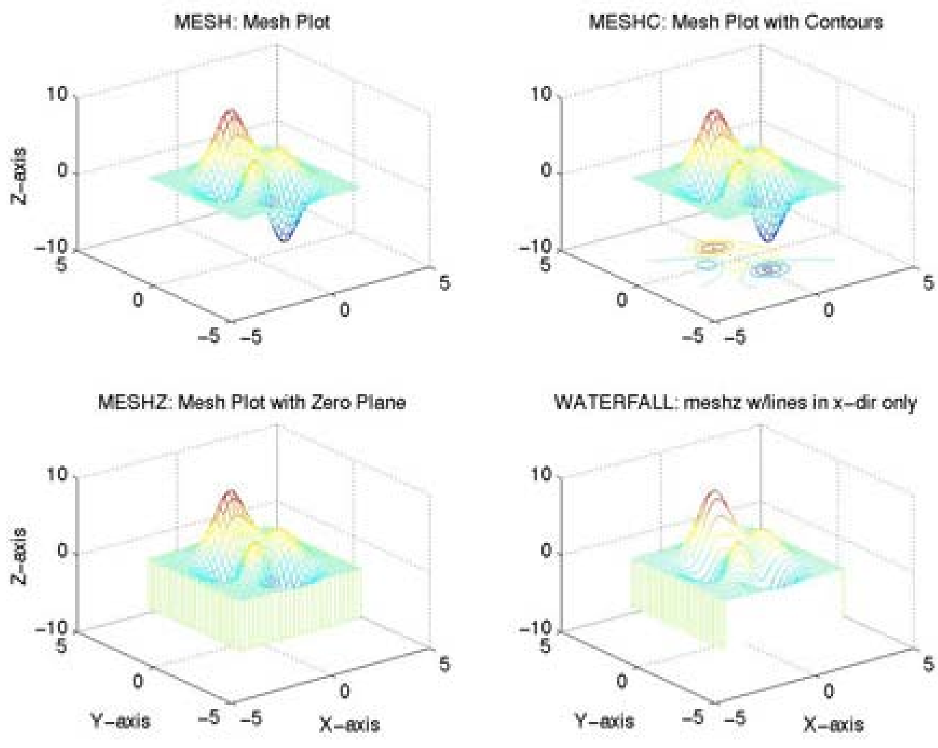
- **`waterfall(X,Y,Z)`**



meshes.m

```
clf
[X,Y,Z] = peaks(30);

subplot(2,2,1)
mesh(X,Y,Z)
xlabel(' '), ylabel(' '), zlabel('Z-axis')
title('MESH: Mesh Plot')

subplot(2,2,2)
meshc(X,Y,Z) % mesh plot with underlying contour
plot
title('MESHC: Mesh Plot with Contours')

subplot(2,2,3)
meshz(X,Y,Z) % mesh plot with zero plane
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('MESHZ: Mesh Plot with Zero Plane')

subplot(2,2,4)
waterfall(X,Y,Z)
```

```
xlabel('X-axis'), ylabel('Y-axis'), zlabel(' ')
title('WATERFALL: meshz w/lines in x-dir only')
```



3D SURFACE PLOTS

surf 3D colored surface

<u>SURF(X,Y,Z,C)</u> plots the colored parametric surface defined by four matrix arguments. The view point is specified by VIEW. The axis labels are determined by the range of X, Y and Z, or by the current setting of AXIS. The color scaling is determined by the range of C, or by the current setting of CAXIS. The scaled color values are used as indices into the current COLORMAP. The shading model is set by SHADING. SURF(X,Y,Z) uses C = Z, so color is proportional to surface height.

sphere

Generate sphere

Syntax

sphere
sphere(n)
[X,Y,Z] = sphere(...)

Description

The sphere function generates the x-, y-, and z-coordinates of a unit sphere for use with surf and mesh.

sphere
generates a sphere consisting of 20-by-20 faces.

sphere(n)
draws a surf plot of an n-by-n sphere in the current figure.

[X,Y,Z] = sphere(n)

returns the coordinates of a sphere in three matrices that are (n+1)-by-(n+1) in size.

You draw the sphere with surf(X,Y,Z) or mesh(X,Y,Z).

Examples

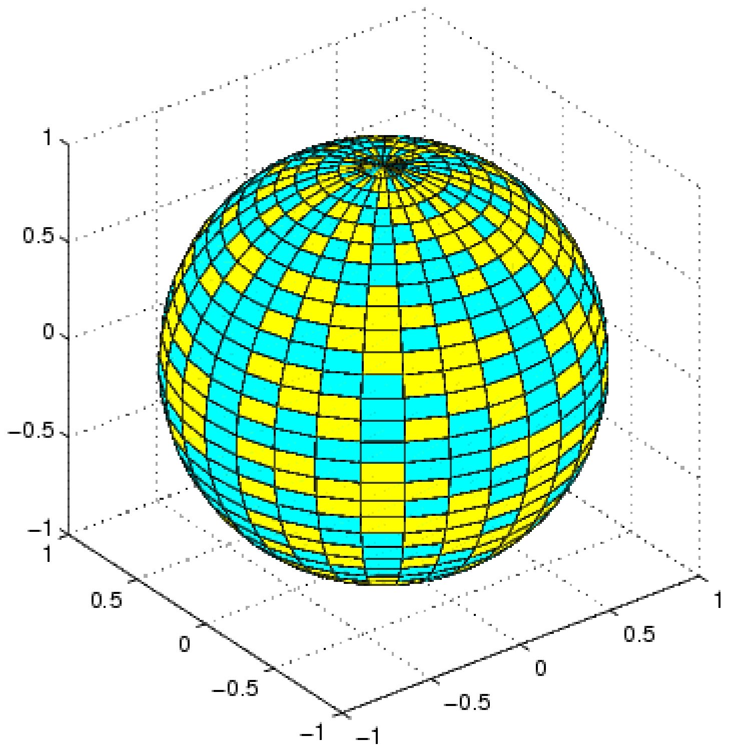
Generate and plot a sphere.

```
sphere
axis equal
```

Color a sphere with the pattern of +1s and -1s in a Hadamard matrix.

- `k = 5;`
- `n = 2^k-1;`

- `[x,y,z] = sphere(n);`
- `c = hadamard(2^k);`
- `surf(x,y,z,c);`
- `colormap([1  1  0; 0  1  1])`
- `axis equal`



surfc 3D colored surface w/ contours underneath

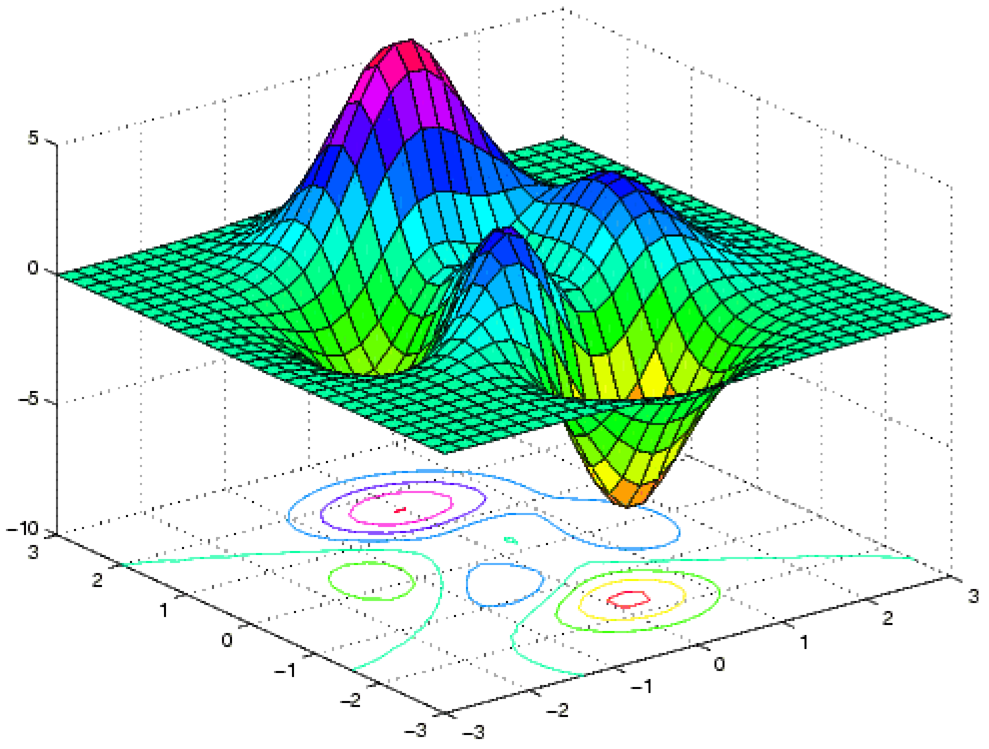
SURFC(...) is the same as SURF(...) except that a contour plot is drawn beneath the surface.

Examples

Display a surface and contour plot of the peaks surface.

- `[X,Y,Z] = peaks(30);`
- `surfc(X,Y,Z)`
- `colormap hsv`
- `axis([-3 3 -3 3 -10 5])`

•



surfl 3D shaded surface w/ lighting

SURFL(...) is the same as SURF(...) except that it draws the surface with highlights from a light source. SURFL(Z), SURFL(X,Y,Z), SURFL(Z,S), and SURFL(X,Y,Z,S) are all legal. S, if specified, is the three vector S = [Sx,Sy,Sz] that specifies the direction of the light source. S can also be specified in view coordinates, S = [AZ,EL].

shading
Set color shading properties

Syntax
shading flat
shading faceted
shading interp

Description

The shading function controls the color shading of surface and patch graphics objects.

shading flat

each mesh line segment and face has a constant color determined by the color value at the end point of the segment or the corner of the face that has the smallest index or indices.

shading faceted

flat shading with superimposed black mesh lines.
This is the default shading mode.

shading interp

varies the color in each line segment and face by interpolating the colormap index or true color value across the line or face.

Examples

Compare a flat, faceted, and interpolated-shaded sphere.

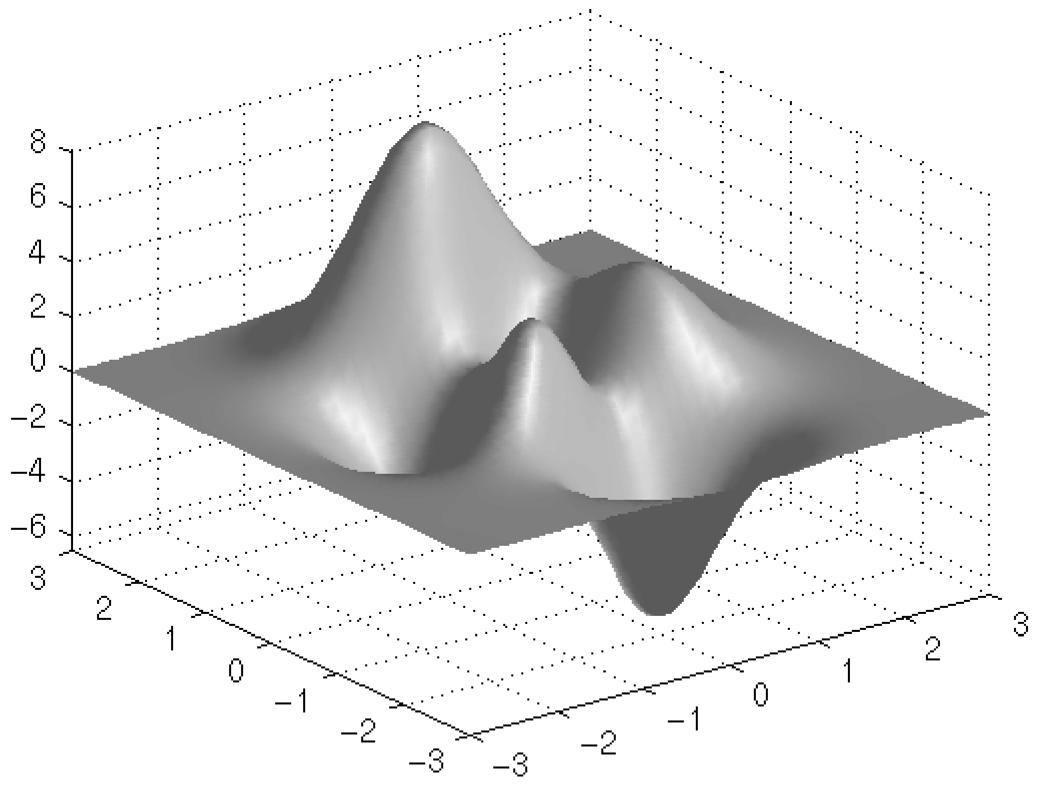
```
subplot(3,1,1)
sphere(16)
axis square
shading flat
title('Flat Shading')

subplot(3,1,2)
sphere(16)
axis square
shading faceted
title('Faceted Shading')

subplot(3,1,3)
sphere(16)
axis square
```

shading interp
title('Interpolated Shading

Examples

View peaks using colormap-based lighting.

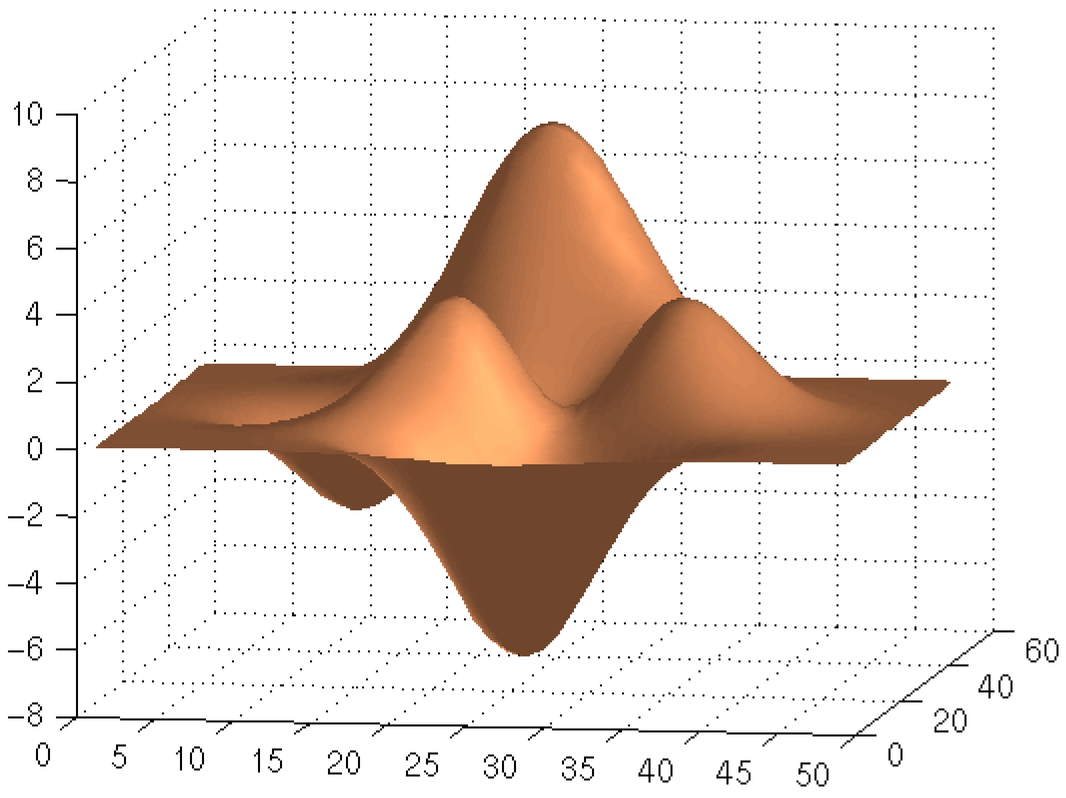
- `[x,y] = meshgrid(-3:1/8:3);`
- `z = peaks(x,y);`
- `surfl(x,y,z);`
- `shading interp`
- `colormap(gray);`
- `axis([-3  3  -3  3  -8  8])`



To plot a lighted surface from a view direction other than the default.

- `view([10 10])`
- `grid on`
- `hold on`
- `surfl(peaks)`
- `shading interp`

- colormap copper
- hold off

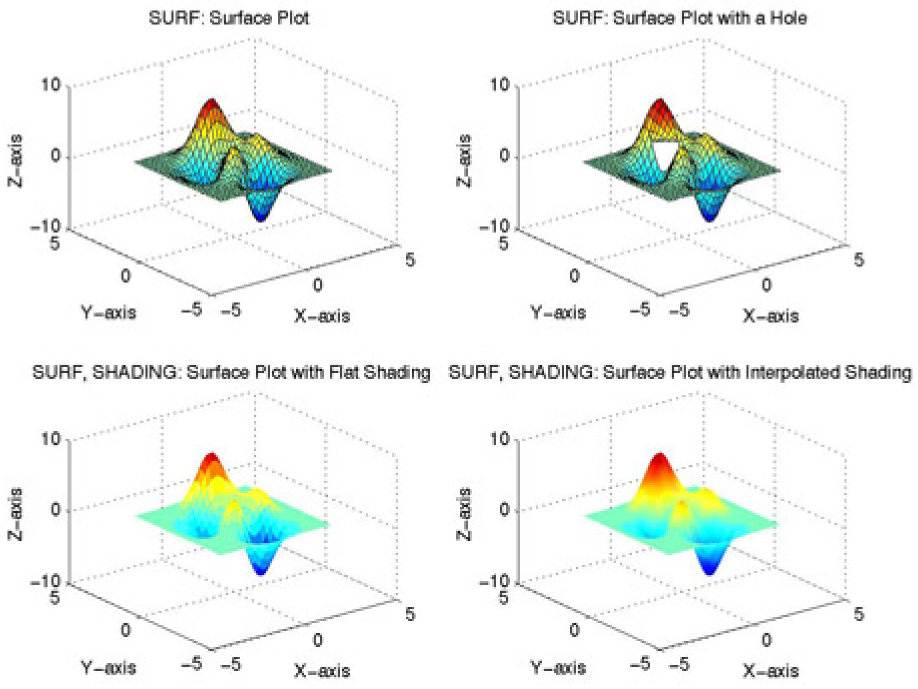


More Examples

surface1.m

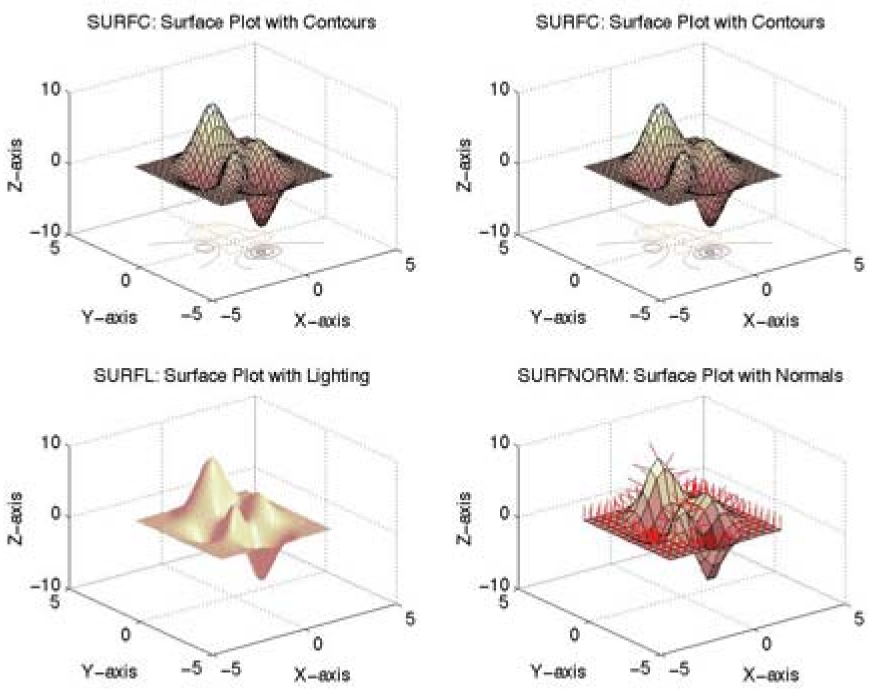
```
clf
[X,Y,Z] = peaks(30);
subplot(2,2,1)
surf(X,Y,Z)
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('SURF: Surface Plot ')

subplot(2,2,2)
x = X(1,:); % vector of x axis
y = Y(:,1); % vector of y axis
i = find(y>-0.5 & y<1.2); % find y axis indices
of hole
j = find(x>-.6 & x<.5); % find x axis indices of
hole
```

```
Z(i,j) = nan; % set values at hole indices to
NaNs
surf(X,Y,Z)
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('SURF: Surface Plot with a Hole')

subplot(2,2,3)
[X,Y,Z] = peaks(30);
surf(X,Y,Z) % same plot as above
shading flat
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('SURF, SHADING: Surface Plot with Flat
Shading')

subplot(2,2,4)
surf(X,Y,Z) % same surface plot
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
shading interp
title('SURF, SHADING: Surface Plot with
Interpolated Shading')
```



surface2.m

```
clf
colormap(winter)
subplot(2,2,1)
[X,Y,Z] = peaks(30);
surfc(X,Y,Z) % surf plot with contour plot
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('SURFC: Surface Plot with Contours')

subplot(2,2,2)
surfc(X,Y,Z) % surf plot with contour plot
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('SURFC: Surface Plot with Contours')

subplot(2,2,3)
surfl(X,Y,Z) % surf plot with lighting
shading interp % surfl plots look best with
interp shading
colormap pink % they also look better with shades
of a single color
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('SURFL: Surface Plot with Lighting')

subplot(2,2,4)
[X,Y,Z] = peaks(15);
surfnorm(X,Y,Z)
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('SURFNORM: Surface Plot with Normals')
```



3D CONTOUR PLOTS

contour a contour plot

CONTOUR(Z) is a contour plot of matrix Z treating the values in Z as heights above a plane. A contour plot are the level curves of Z for some values V. The values V are chosen automatically. CONTOUR(X,Y,Z) X and Y specify the (x,y) coordinates of the surface as for SURF. CONTOUR(Z,N) and CONTOUR(X,Y,Z,N) draw N contour lines, overriding the automatic value.

contour3 a 3D contour plot

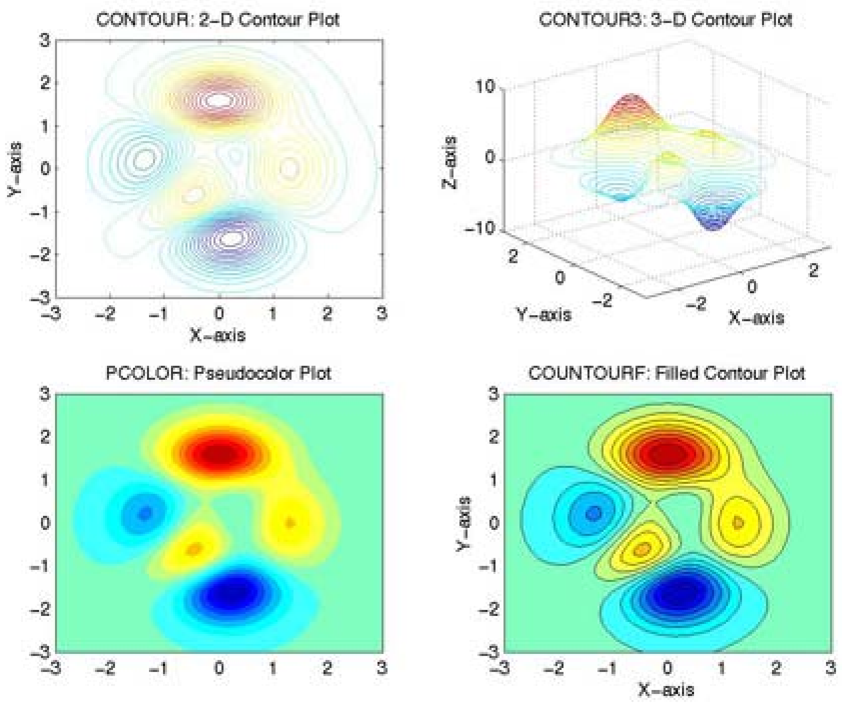
CONTOUR3(...) is the same as CONTOUR(...) except that the contours are drawn at their corresponding Z level.

contourf a filled contour plot

CONTOURF(...) is the same as CONTOUR(...) except that the contours are filled.

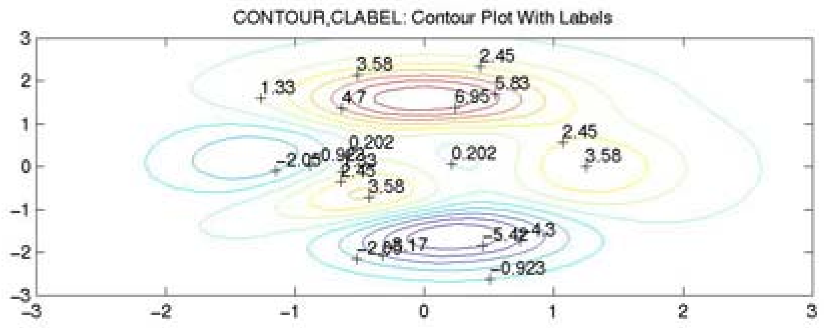
contour1.m

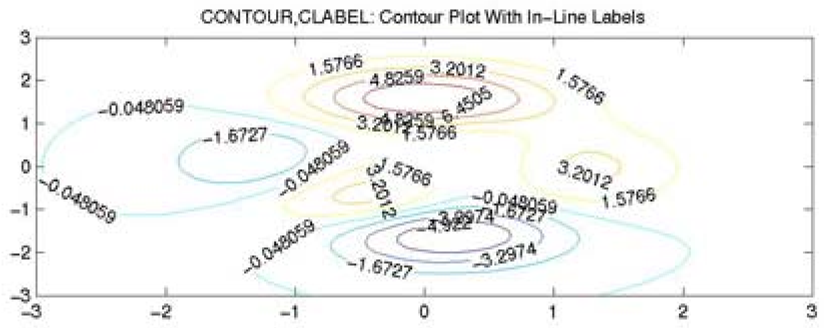
```
clf
colormap(jet)
subplot(2,2,1)
[X,Y,Z] = peaks;
contour(X,Y,Z,30) % generate 30 2-D contour lines
xlabel('X-axis'), ylabel('Y-axis')
title('CONTOUR: 2-D Contour Plot')

subplot(2,2,2)
contour3(X,Y,Z,30) % the same contour plot in 3-D
xlabel('X-axis'), ylabel('Y-axis'), zlabel('Z-
axis')
title('CONTOUR3: 3-D Contour Plot')

subplot(2,2,3)
pcolor(X,Y,Z)
shading interp % remove the grid lines
title('PCOLOR: Pseudocolor Plot')

subplot(2,2,4)
contourf(X,Y,Z,15) % filled contour plot with 12
contours
xlabel('X-axis'), ylabel('Y-axis')
title('COUNTOURF: Filled Contour Plot')
```



contour2.m

```
clf
subplot(2,1,1)
[X,Y,Z] = peaks;
C = contour(X,Y,Z,12);
clabel(C)
title('CONTOUR,CLABEL: Contour Plot With Labels')

subplot(2,1,2)
[C,h] = contour(X,Y,Z,8); % ask for fewer
contours
clabel(C,h)
title('CONTOUR,CLABEL: Contour Plot With In-Line
Labels')
```





3D SPECIAL PLOTS

quiver a velocity arrow plot

QUIVER(X,Y,U,V) plots velocity vectors as arrows with components (u,v) at the points (x,y). The matrices X,Y,U,V must all be the same size and contain corresponding position and velocity components (X and Y can also be vectors to specify a uniform grid). QUIVER automatically scales the arrows to fit within the grid. QUIVER(U,V) plots velocity vectors at equally spaced points in the x-y plane.

quiver3 a 3D velocity arrow plot

QUIVER3(X,Y,Z,U,V,W) plots velocity vectors as arrows with components (u,v,w) at the points (x,y,z). The matrices X,Y,Z,U,V,W must all be the same size and contain the corresponding position and velocity components. QUIVER3 automatically scales the arrows to fit. QUIVER3(Z,U,V,W) plots velocity vectors at the equally spaced surface points specified by the matrix Z. QUIVER3(Z,U,V,W,S) or QUIVER3(X,Y,Z,U,V,W,S) automatically scales the arrows to fit and then stretches them by S.

ribbon a ribbon form of plot(x,y)

RIBBON(X,Y) is the same as PLOT(X,Y) except that the columns of Y are plotted as separated ribbons in 3-D. RIBBON(Y) uses the default value of X=1:SIZE(Y,1). RIBBON(X,Y,WIDTH) specifies the width of the ribbons to be WIDTH. The default value is WIDTH = 0.75;

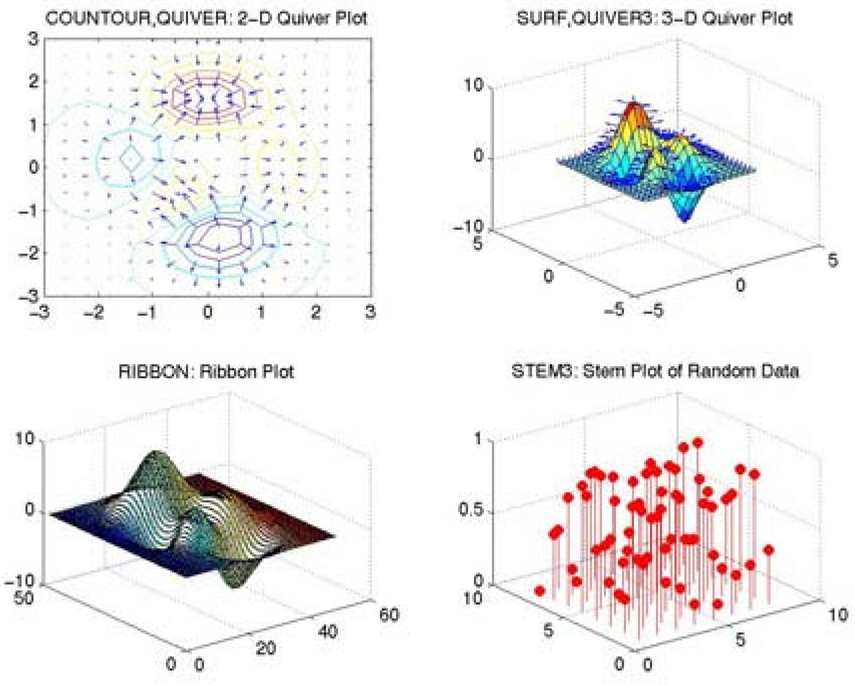
stem3 a 3D stem plot

STEM3(Z) plots the discrete surface Z as stems from the xy-plane terminated with circles for the data value. STEM3(X,Y,Z) plots the surface Z at the values specified in X and Y. STEM3(...,'filled') produces a stem plot with filled markers.

special.m

```
clf
subplot(2,2,1)
[X,Y,Z] = peaks(16);
[DX,DY] = gradient(Z,.5,.5);
contour(X,Y,Z,10)
hold on
quiver(X,Y,DX,DY)
hold off
title('COUNTOUR,QUIVER: 2-D Quiver Plot')

subplot(2,2,2)
[X,Y,Z] = peaks(20);
[Nx,Ny,Nz] = surfnorm(X,Y,Z);
surf(X,Y,Z)
hold on
quiver3(X,Y,Z,Nx,Ny,Nz)
hold off
title('SURF,QUIVER3: 3-D Quiver Plot')
```

```
subplot(2,2,3)
Z = peaks;
ribbon(Z)
title('RIBBON: Ribbon Plot')

subplot(2,2,4)
Z = rand(8);
stem3(Z,'ro','filled');
grid on
title('STEM3: Stem Plot of Random Data')
```


COUNTOUR,QUIVER: 2-D Quiver Plot
SURF,QUIVER3: 3-D Quiver Plot
RIBBON: Ribbon Plot
STEM3: Stem Plot of Random Data