**Module 3 Day2 Course Material**

**Basic Plotting Functions**

This section describes important graphics functions and provides examples of some typical applications. The plotting tools, described in previous sections, make use of MATLAB plotting functions and use these functions to generate code for graphs:

**•**“Creating a Plot”

**•**“Multiple Data Sets in One Graph”

**•**“Specifying Line Styles and Colors”

**•**“Plotting Lines and Markers”

**•**“Imaginary and Complex Data”

**•**“Adding Plots to an Existing Graph”

**•**“Figure Windows”

**•**“Multiple Plots in One Figure”

**•**“Controlling the Axes”

**•**“Axis Labels and Titles”

**•**“Saving Figures”

**Creating a 2D-Plot**

The plot function has 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 y. If you specify two vectors as arguments, plot(x,y) produces a graph of y versus x.

For example, these statements use the colon operator to create a vector of x values ranging from 0 to 2π, compute the sine of these values, and plot the result:

x = 0:pi/100:2\*pi;

y = sin(x);

plot(x,y)

Now label the axes and add a title. The characters \pi create the symbol π. See

text strings for more symbols:

xlabel('x = 0:2\pi')

ylabel('Sine of x')

title('Plot of the Sine Function','FontSize',12)

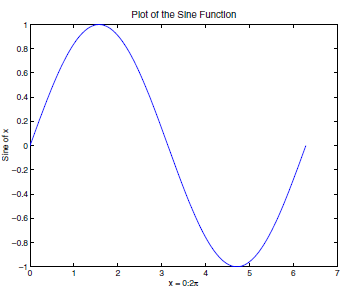


Fig1. Plot of Sine wave

**Multiple Data Sets in One Graph**

Multiple x-y pair arguments create multiple graphs with a single call to plot. MATLAB automatically cycles through a predefined (but user settable) list of colors to allow discrimination among sets of data. See the axes ColorOrder and LineStyleOrder properties.

For example, these statements plot three related functions of x, with each

curve in a separate distinguishing color:

x = 0:pi/100:2\*pi;

y = sin(x);

y2 = sin(x-.25);

y3 = sin(x-.5);

plot(x,y,x,y2,x,y3)

The legend command provides an easy way to identify the individual plots:

legend('sin(x)','sin(x-.25)','sin(x-.5)')

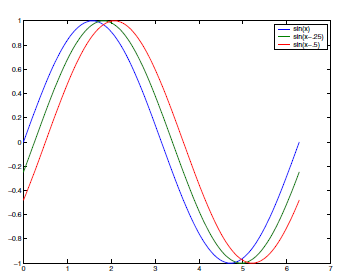


Fig2. Plot of Multiple data

**Specifying Line Styles and Colors**

It is possible to specify color, line styles, and markers (such as plus signs or circles) when you plot your data using the plot command: plot(x,y,'*color\_style\_marker*') *color\_style\_marker* is a string containing from one to four characters (enclosed in single quotation marks) constructed from a color, a line style, and a marker type:

* Color strings are 'c', 'm', 'y', 'r', 'g', 'b', 'w', and 'k'. These correspond

to cyan, magenta, yellow, red, green, blue, white, and black.

* Line style strings are '-' for solid, '--' for dashed, ':' for dotted, '-.' for

dash-dot. Omit the line style for no line.

* The marker types are '+', 'o', '\*', and 'x', and the filled marker types are

's' for square, 'd' for diamond, '^' for up triangle, 'v' for down triangle,

'>' for right triangle, '<' for left triangle, 'p' for pentagram, 'h' for

hexagram, and none for no marker.

You can also edit color, line style, and markers interactively. See “Editing

Plots” on page 3-16 for more information.

**Plotting Lines and Markers**

If you specify a marker type but not a line style, MATLAB draws only the

marker. For example,

plot(x,y,'ks')

plots black squares at each data point, but does not connect the markers with a line.

The statement

plot(x,y,'r:+')

plots a red dotted line and places plus sign markers at each data point.

**Placing Markers at Every Tenth Data Point**

You might want to use fewer data points to plot the markers than you use to plot the lines. This example plots the data twice using a different number of points for the dotted line and marker plots:

x1 = 0:pi/100:2\*pi;

x2 = 0:pi/10:2\*pi;

plot(x1,sin(x1),'r:',x2,sin(x2),'r+');

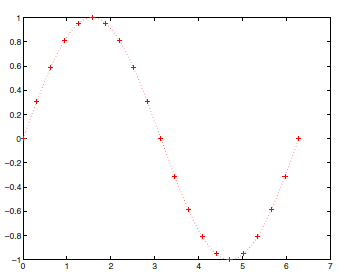


Fig3.Placing Markers

**Axis Labels and Titles**

The xlabel, ylabel, and zlabel commands add *x*-, *y*-, and *z*-axis labels. The title command adds a title at the top of the figure and the text function inserts text anywhere in the figure.

You can produce mathematical symbols using LaTeX notation in the text string, as the following example illustrates:

t = -pi:pi/100:pi;

y = sin(t);

plot(t,y)

axis([-pi pi -1 1])

xlabel('-\pi \leq {\itt} \leq \pi')

ylabel('sin(t)')

title('Graph of the sine function')

text(1,-1/3,'{\itNote the odd symmetry.}')

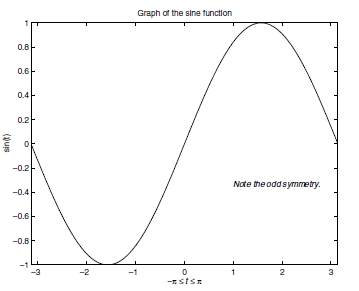


Fig4.Plot with Labels

**Saving Figures**

Save a figure by selecting **Save** from the **File** menu to display a file save dialog. MATLAB saves the data it needs to recreate the figure and its contents (i.e., the entire graph) in a file with a .fig extension.

To save a figure using a standard graphics format, such as TIFF, for use with other applications, select **Export Setup** from the **File** menu. You can also save from the command line — use the save as command, including any options to save the figure in a different format.

**Mesh and Surface Plots**

MATLAB defines a surface by the *z*-coordinates of points above a grid in the *x*-*y* plane, using straight lines to connect adjacent points. The mesh and surfplotting functions display surfaces in three dimensions. mesh produceswireframe surfaces that color only the lines connecting the defining points. surf displays both the connecting lines and the faces of the surface in color.

The figure colormap and figure properties determine how MATLAB colors the surface.

**Visualizing Functions of Two Variables**

To display a function of two variables, *z* = *f* (*x*,*y*),

* Generate X and Y matrices consisting of repeated rows and columns, respectively, over the domain of the function.
* Use X and Y to evaluate and graph the function. The meshgrid function transforms the domain specified by a single vector or two vectors x and y into matrices X and Y for use in evaluating functions of two variables. The rows of X are copies of the vector x and the columns of Y are copies of the vector y.

**Module 4**

**3D-Plotting**

3D line plot, We can plot a line through (x,y,z) points defined parametrically using the following instruction.

Example of 3D plot:

>> t = 0.01:.01:20\*pi;

>>x=cos(t);

>>y=sin(t);

>>z=t.^3;

>>plot3(x,y,z);

>>xlabel('x');

>>ylabel('y');

>>zlabel('z');

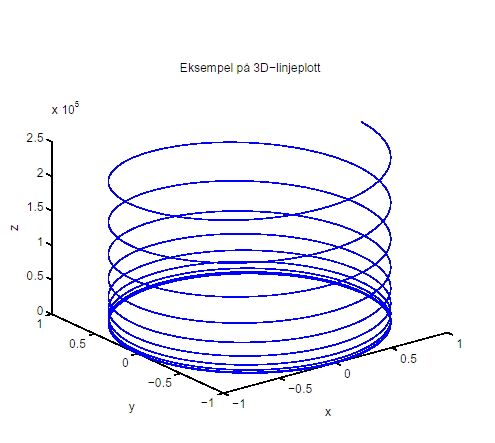
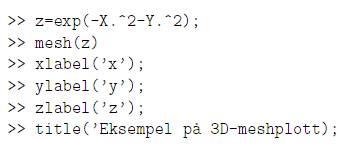


Fig. 3D line plot

3D surface plot We can plot surface functions as for example the surface given as

z = e-x2-y2. We want to plot this function over the square [-2,2] x [-2,2].To do this we have to generate a grid of point of computation.



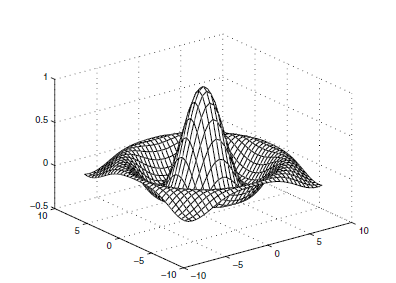


Fig. 3D Mesh plot

**Example — Graphing the sinc Function**

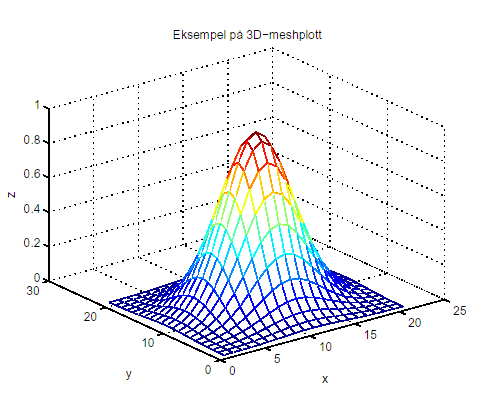
This example evaluates and graphs the two-dimensional *sinc* function, sin(*r*)/*r*, between the *x* and *y* directions. R is the distance from the origin, which is at the center of the matrix. Adding eps (a MATLAB command that returns a small floating-point number) avoids the indeterminate 0/0 at the origin:

[X,Y] = meshgrid(-8:.5:8);

R = sqrt(X.^2 + Y.^2) + eps;

Z = sin(R)./R;

mesh(X,Y,Z,'EdgeColor','black')



**Example — Colored Surface Plots**

A surface plot is similar to a mesh plot except that MATLAB colors the rectangular faces of the surface. The color of each faces is determined by the values of Z and the colormap (a colormap is an ordered list of colors). These statements graph the *sinc* function as a surface plot, specify a colormap, and add a color bar to show the mapping of data to color:

surf(X,Y,Z)

colormap hsv

colorbar

