EXPERIMENT-FIVE

GRAPHICS

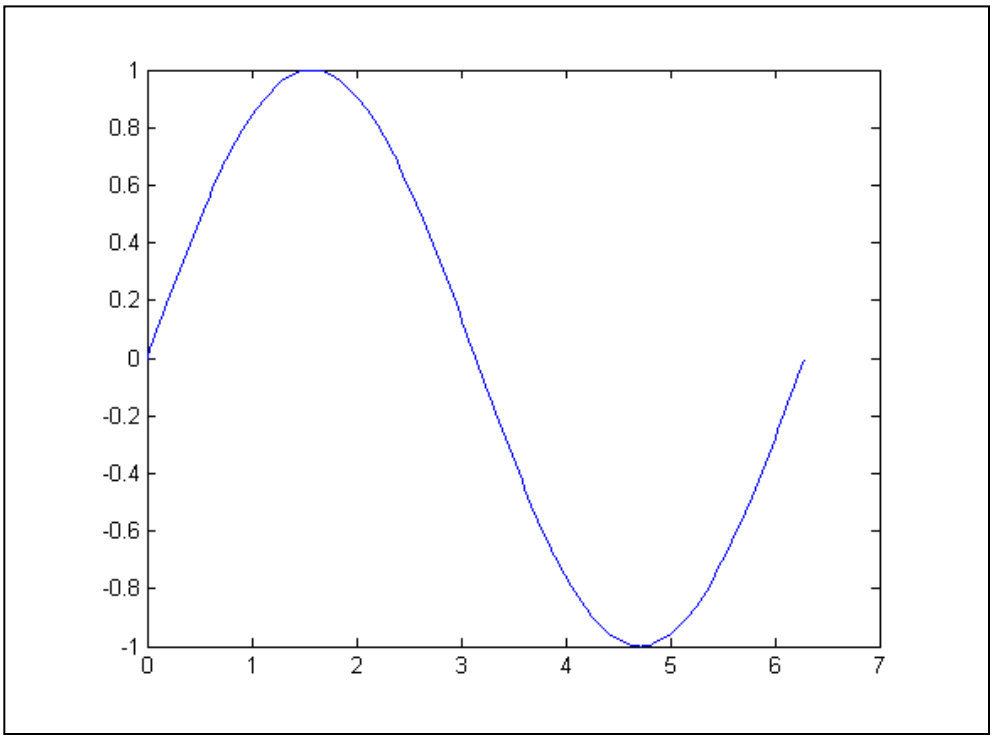
MATLAB has extensive facilities for displaying vectors and matrices as graphs, as well as annotating and printing these graphs. This chapter describes a few of the most important graphics functions and provides examples of some typical applications.

1- Creating a 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.

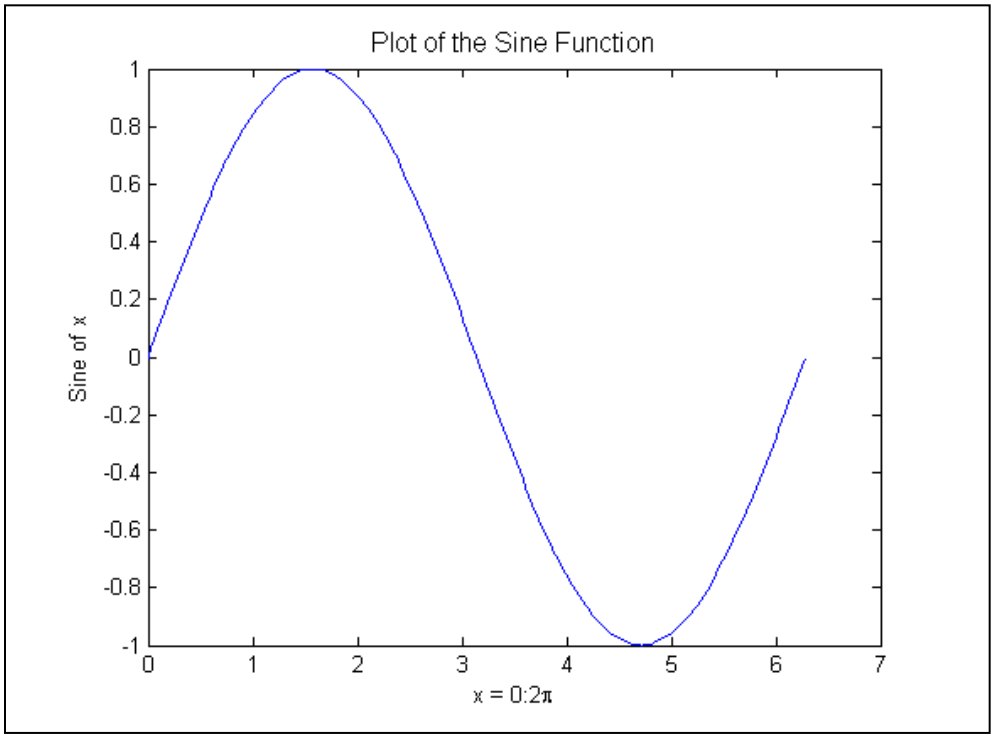
For example, these statements use the colon operator to create a vector of x values ranging from zero to 2π, compute the sine of these values, and plot the result.

```
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 in the plot.

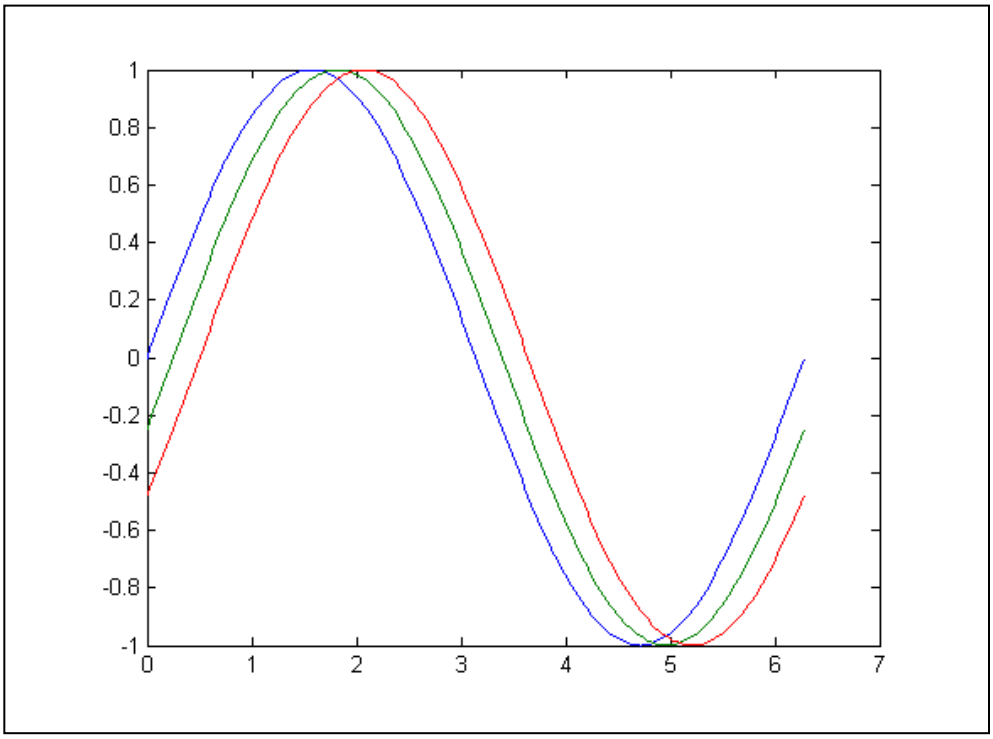
```
xlabel('x = 0:2\pi')
ylabel('Sine of x')
title('Plot of the Sine Function')
```



2- Multiple Data Sets in One Graph

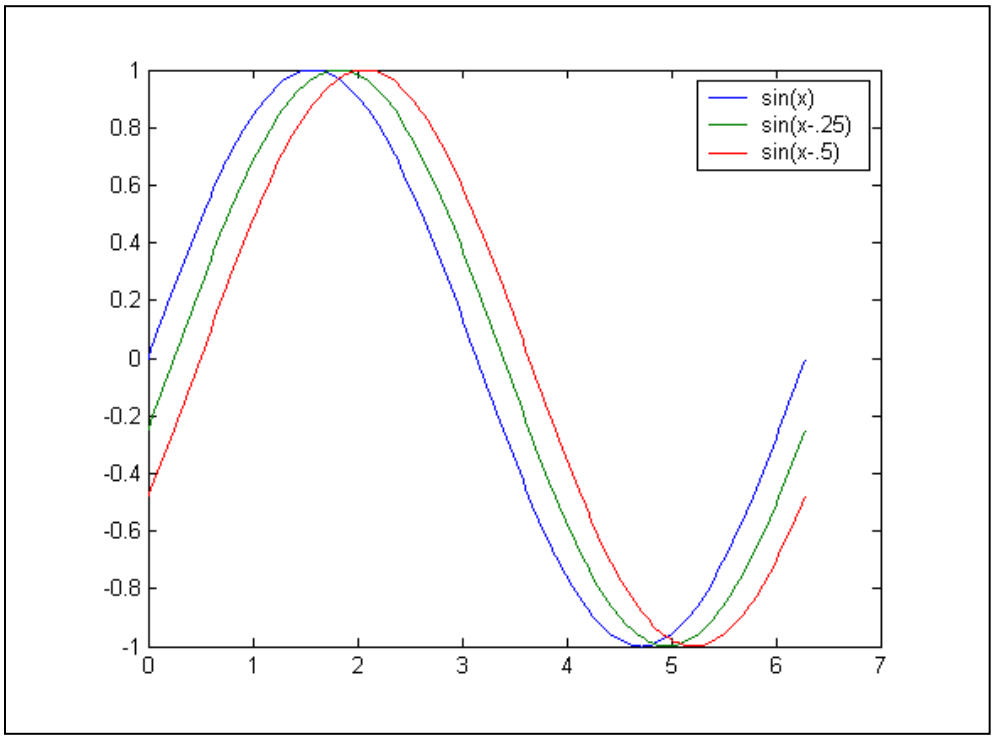
Multiple x-y pair arguments create multiple graphs with a single call to plot. MATLAB automatically cycles through a predefined (but user settable) list of colors to allow discrimination between each set of data. For example, these statements plot three related functions of x, each curve in a separate distinguishing color.

```
x = 0:pi/100:2*pi;
y = sin(x);
y2 = sin(x-0.25);
y3 = sin(x-0.5);
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)')
```



3- 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.
- Linestyle strings are '-' for solid, '--' for dashed, ':' for dotted, '-.' for dash-dot, and 'none' for no line.
- The marker types are '+', 'o', '*', and 'x' and the filled marker types '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.

For example,

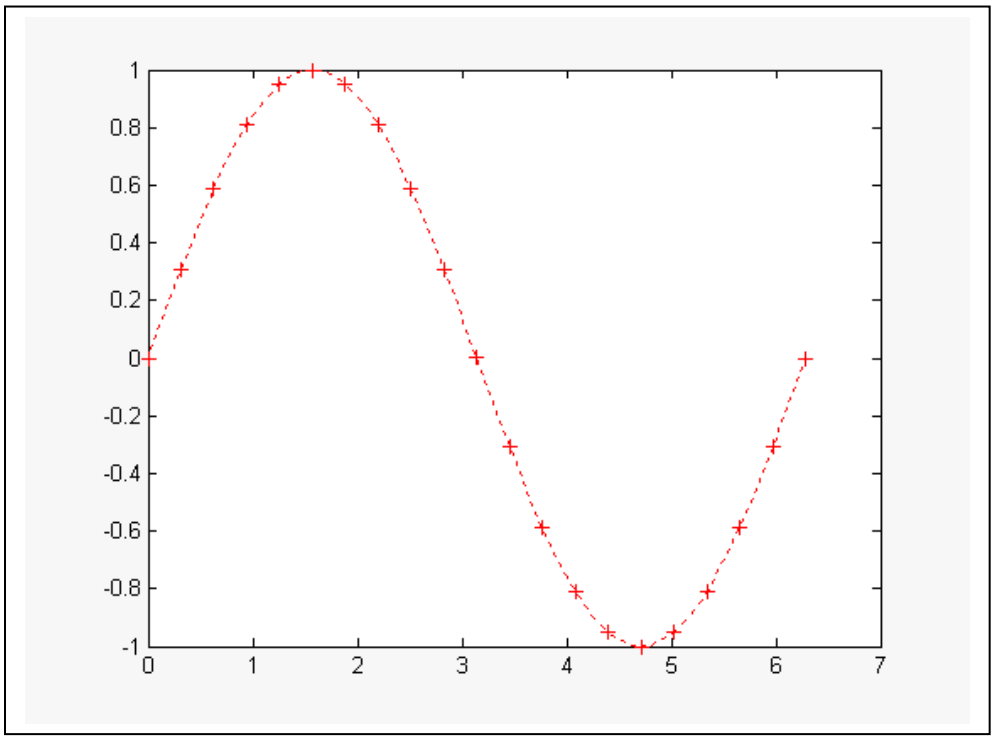
```
plot(x,y,'ko')
```

plots black circles 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. You may 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+')
```



4- Imaginary and Complex Data

When the arguments to plot are complex, the imaginary part is ignored *except* when plot is given a single complex argument. For this special case, the command is a shortcut for a plot of the real part versus the imaginary part. Therefore,

```
plot(Z)
```

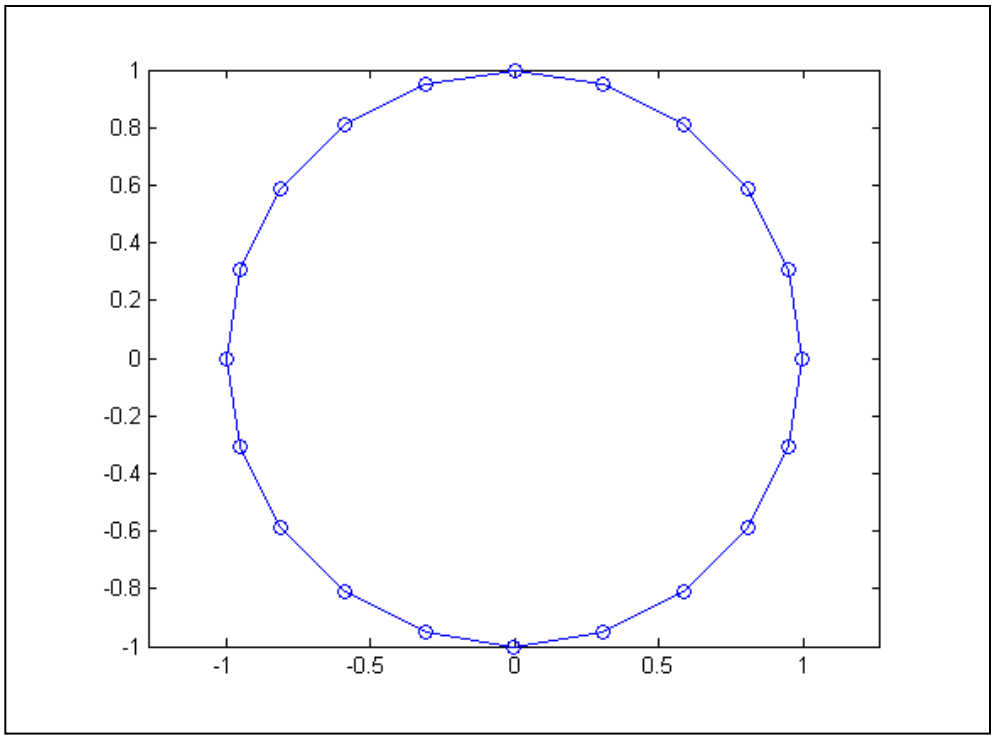
where Z is a complex vector or matrix, is equivalent to

```
plot(real(Z),imag(Z))
```

For example,

```
t = 0:pi/10:2*pi;
plot(exp(i*t),'-o')
axis equal
```

draws a 20-sided polygon with little circles at the vertices. The command, axis equal, makes the individual tick mark increments on the *x*- and *y*-axes the same length, which makes this plot more circular in appearance.



5- Adding Plots to an Existing Graph

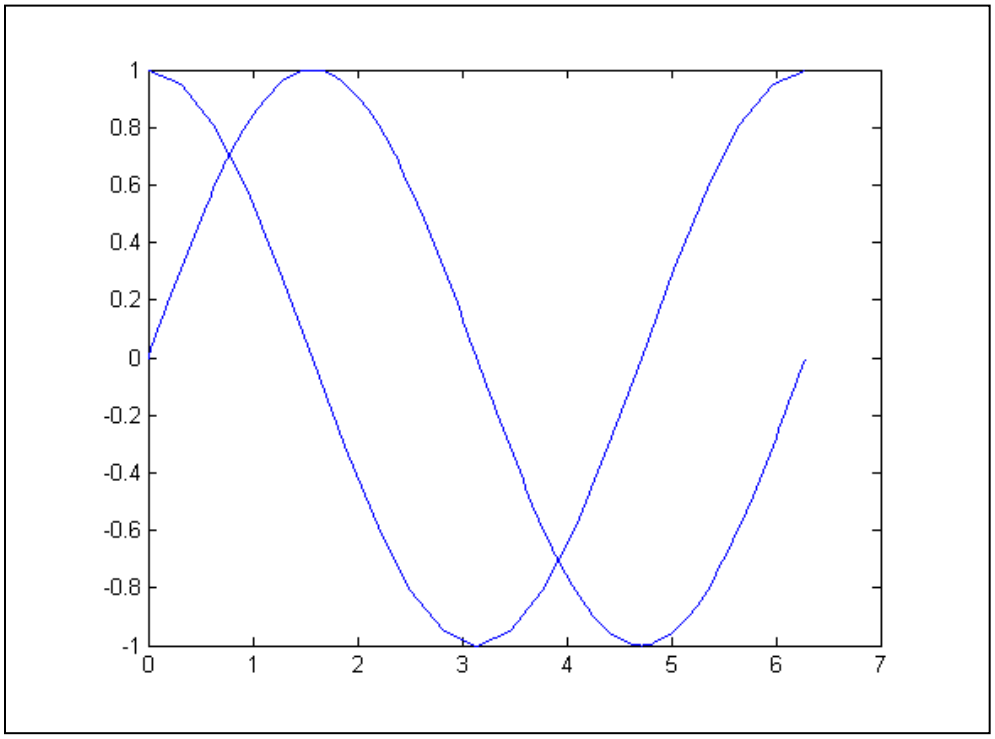
The hold command enables you to add plots to an existing graph. When you type

```
hold on
```

MATLAB does not replace the existing graph when you issue another plotting command; it adds the new data to the current graph, rescaling the axes if necessary. For example, these statements first create a contour plot of the peaks function, then superimpose a pseudocolor plot of the same function.

```
x = 0:pi/100:2*pi;
y = 0:pi/10:2*pi;
plot(x,sin(x))
hold on
plot(y,cos(y))
hold off
```

The hold on command causes the sin(x) plot to be combined with the cos(y) plot in one figure.



6- Figure Windows

Graphing functions automatically open a new figure window if there are no figure windows already on the screen. If a figure window exists, MATLAB uses that window for graphics output. If there are multiple figure windows open, MATLAB targets the one that is designated the "current figure" (the last figure used or clicked in).

To make an existing figure window the current figure, you can click the mouse while the pointer is in that window or you can type

```
figure(n)
```

where n is the number in the figure title bar. The results of subsequent graphics commands are displayed in this window.

To open a new figure window and make it the current figure, type

```
figure
```

7- Controlling the Axes

The axis command supports a number of options for setting the scaling, orientation, and aspect ratio of plots. You can also set these options interactively.

7.1- Setting Axis Limits:

By default, MATLAB finds the maxima and minima of the data to choose the axis limits to span this range. The axis command enables you to specify your own limits

```
axis([xmin xmax ymin ymax])
```

or for three-dimensional graphs,

```
axis([xmin xmax ymin ymax zmin zmax])
```

Use the command `axis auto` to re-enable MATLAB's automatic limit selection.

7.2- Setting Axis Aspect Ratio:

`axis` also enables you to specify a number of predefined modes. For example,

```
axis square
```

makes the *x*-axes and *y*-axes the same length and

```
axis equal
```

makes the individual tick mark increments on the *x*- and *y*-axes the same length. This means

```
plot(exp(i*[0:pi/10:2*pi]))
```

followed by either `axis square` or `axis equal` turns the oval into a proper circle.

```
axis auto normal
```

returns the axis scaling to its default, automatic mode.