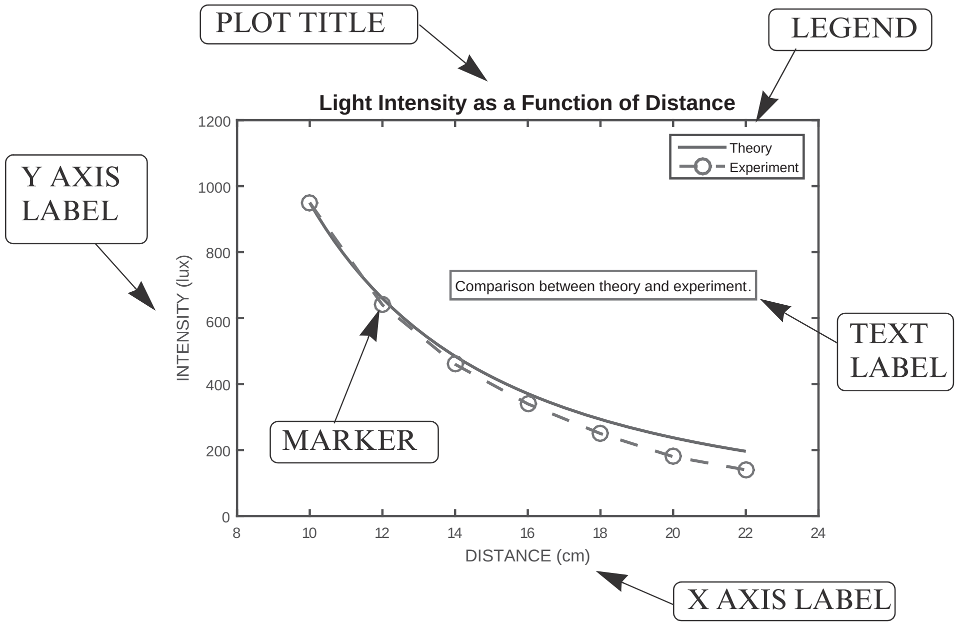
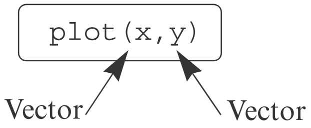


Figure 5-1: Example of a formatted two-dimensional plot.

5.1 *THE* `plot` *COMMAND*

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



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

The arguments `x` and `y` are each a vector (one-dimensional array). The two vectors *must* have the same number of elements. When the `plot` command is executed, a figure is created in the Figure Window. If not already open, the Figure Window opens automatically when the command is executed. The figure has a single curve with the `x` values on the abscissa (horizontal axis) and the `y` values on the ordinate (vertical axis). The curve is constructed of straight-line segments that connect the points whose coordinates are defined by the elements of the vectors `x` and `y`. Each of the vectors, of course, can have any name. The vector that is typed first in the `plot` command is used for the horizontal axis, and the vector that is typed second is used for the vertical axis. If only one vector is entered as an input argument in the `plot` command (for example `plot(y)`) than the figure will show a plot of the values of the elements of the vector ($y(1)$, $y(2)$, $y(3)$, …) versus the element number (1, 2, 3, …).

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 by typing the following in the Command Window:

```
>> x=[1.1  1.8  3.2  5.5  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 5-2.

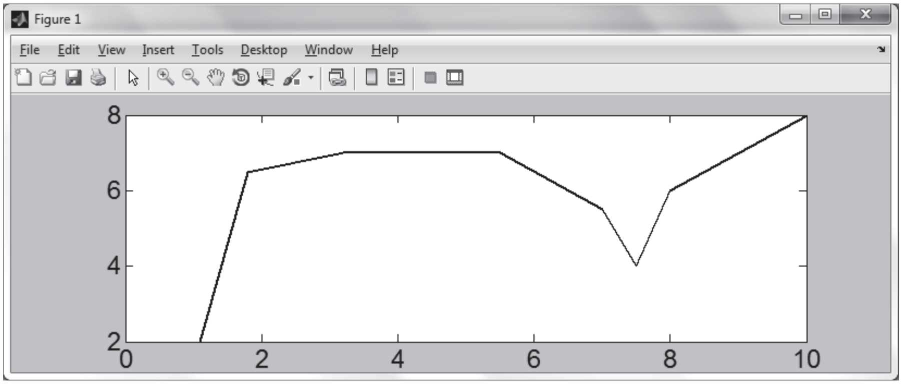


Figure 5-2: The Figure Window with a simple plot.

The plot appears on the screen in blue, which is the default line color.

The `plot` command has additional, optional arguments that can be used to specify the color and style of the line and the color and type of markers, if any are desired. With these options the command has the form:

```
plot(x,y,'line specifiers','PropertyName',PropertyValue)
```

Vector Vector

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

(Optional) Properties with values that can be used to specify the line width, and a marker's size and edge, and fill colors.

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`			

Notes about using the specifiers:

- The specifiers are typed inside the `plot` command as strings.
- Within the string the specifiers can be typed in any order.
- The specifiers are optional. This means that none, one, two, or all three types can be included in a command.

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.

`Property Name and Property Value:`

Properties are optional and can be used to specify the thickness of the line, the size of the marker, and the colors of the marker's edge line and fill. The Property Name is typed as a string, followed by a comma and a value for the property, all inside the `plot` command.

Four properties and their possible values are:

Property name	Description	Possible property values
`LineWidth` (or `linewidth`)	Specifies the width of the line.	A number in units of points (default 0.5).
`MarkerSize` (or `markersize`)	Specifies the size of the marker.	A number in units of points.
`MarkerEdgeColor` (or `markeredgecolor`)	Specifies the color of the marker, or the color of the edge line for filled markers.	Color specifiers from the table above, typed as a string.
`MarkerFaceColor` (or `markerfacecolor`)	Specifies the color of the filling for filled markers.	Color specifiers from the table above, typed as a string.

For example, the command

```
plot(x,y,‘-mo’,‘LineWidth’,2,‘markersize’,12,
      ‘MarkerEdgeColor’,‘g’,‘markerfacecolor’,‘y’)
```

creates a plot that connects the points with a magenta solid line and circles as markers at the points. The line width is 2 points and the size of the circle markers is 12 points. The markers have a green edge line and yellow filling.

A note about line specifiers and properties:

The three line specifiers, which indicate the style and color of the line, and the type of the marker can also be assigned with a `PropertyName` argument followed by a `PropertyValue` argument. The Property Names for the line specifiers are:

Specifier	Property Name	Possible property values
Line style	`linestyle` (or `LineStyle`)	Line style specifier from the table above, typed as a string.
Line color	`color` (or `Color`)	Color specifier from the table above, typed as a string.
Marker	`marker` (or `Marker`)	Marker specifier from the table above, typed as a string.

As with any command, the `plot` command can be typed in the Command Window, or it can be included in a script file. It also can be used in a function file (explained in Chapter 7). It should also be remembered that before the `plot` command can be executed, the vectors `x` and `y` must have assigned elements.

This can be done, as was explained in Chapter 2, by entering values directly, by using commands, or as the result of mathematical operations. The next two subsections show examples of creating simple plots.

5.1.1 Plot of Given Data

In this case given data is used to create vectors that are then used in the `plot` command. 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

To plot this data, the list of years is assigned to one vector (named `yr`), and the corresponding sales data is assigned to a second vector (named `sle`). The Command Window where the vectors are created and the `plot` command is used is shown below:

```
>> yr=[1988:1:1994];
>> sle=[8  12  20  22  18  24  27];
>> plot(yr,sle,'--r*','linewidth',2,'markersize',12)
>>
```

Line Specifiers: dashed red line and asterisk marker.

Property Name and Property Value: the line width is 2 points and the marker size is 12 points.

Once the `plot` command is executed, the Figure Window with the plot, as shown in Figure 5-3, opens. The plot appears on the screen in red.

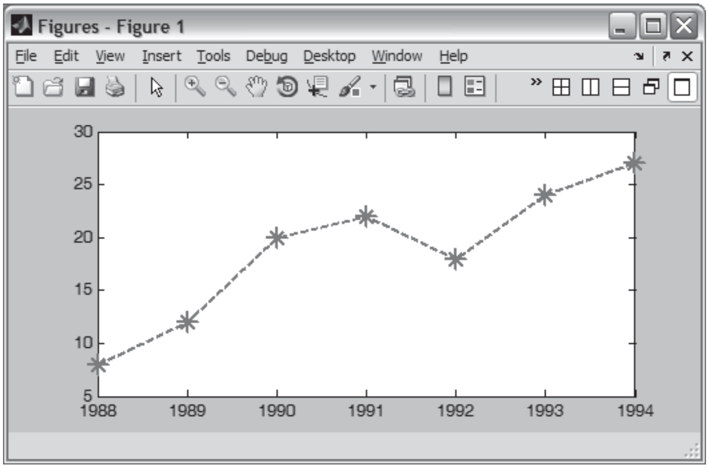


Figure 5-3: The Figure Window with a plot of the sales data.

5.1.2 Plot of a Function

In many situations there is a need to plot a given function. This can be done in MATLAB by using the `plot` or the `fplot` command. The use of the `plot` command is explained below. The `fplot` command is explained in detail in the next section.

In order to plot a function $y = f(x)$ 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 $f(x)$ by using element-by-element calculations (see Chapter 3). Once the two vectors are defined, they can be used in the `plot` command.

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];            Create vector x with the domain of the function.
y=3.5.^(-0.5*x).*cos(6*x);          Create vector y with the func-
                                    tion value at each x.
plot(x,y)                              Plot y as a function of x.
```

Once the script file is executed, the plot is created in the Figure Window, as shown in Figure 5-4. Since the plot is made up of segments of straight lines that connect the points, to obtain an accurate plot of a function, the spacing between the elements of the vector `x` must be appropriate. Smaller spacing is needed for a

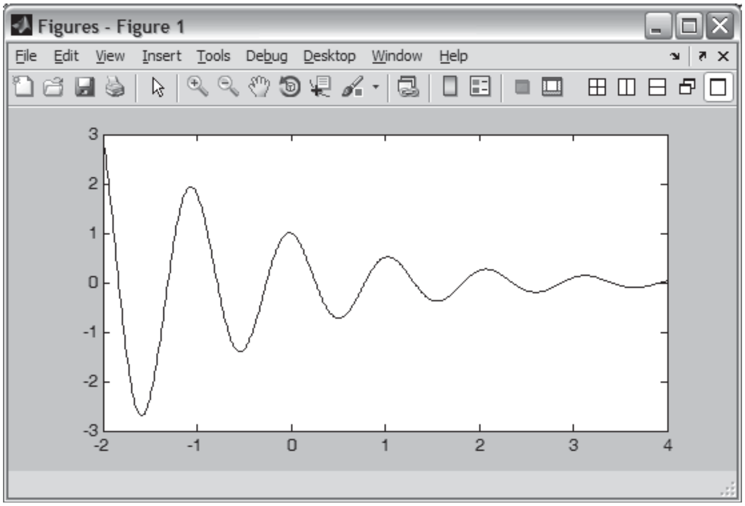


Figure 5-4: The Figure Window with a plot of the function $y = 3.5^{-0.5x}\cos(6x)$.