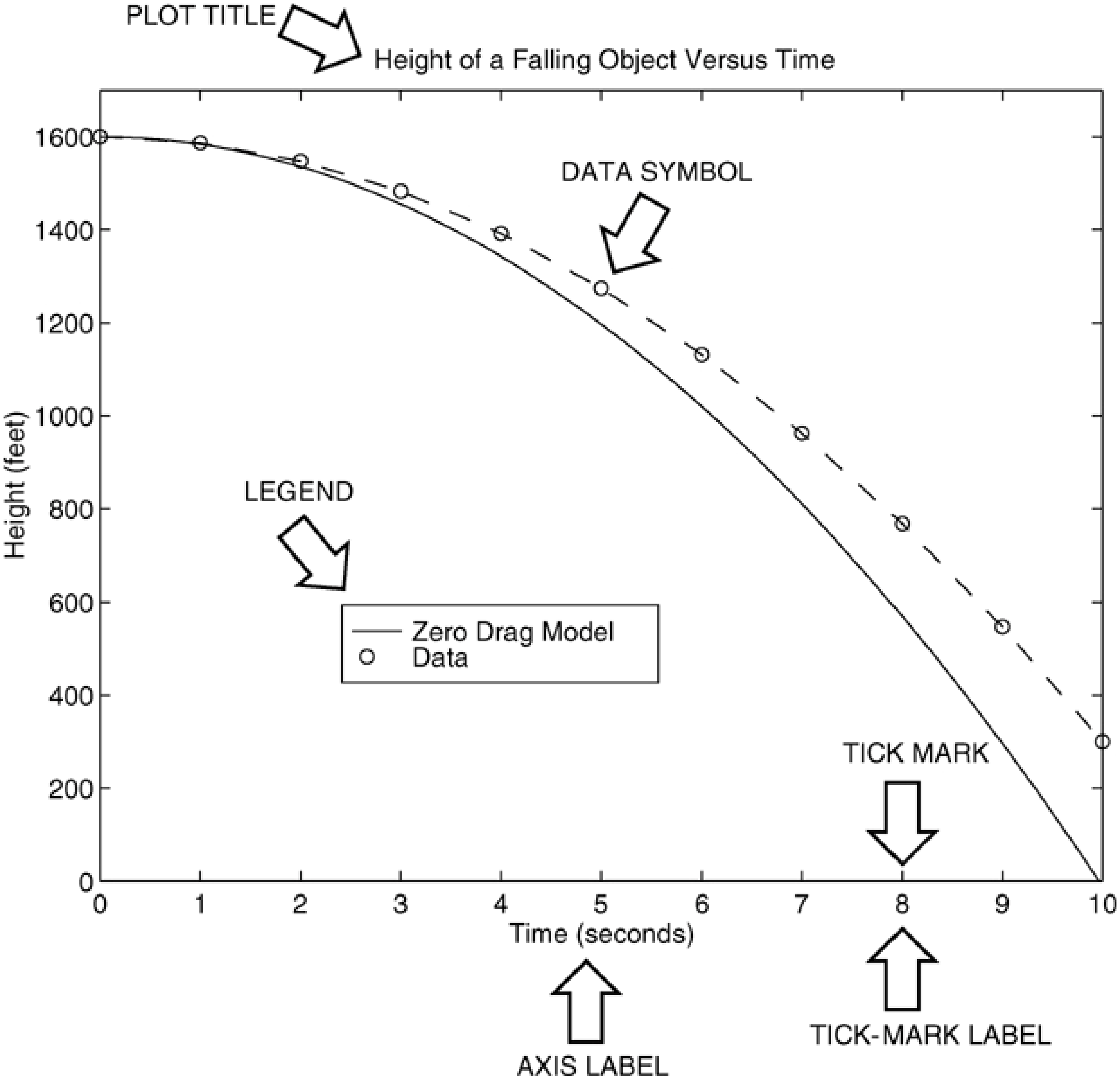
®

Lecture 5

Two Dimensional Plotting

Nomenclature for a typical *xy* plot.

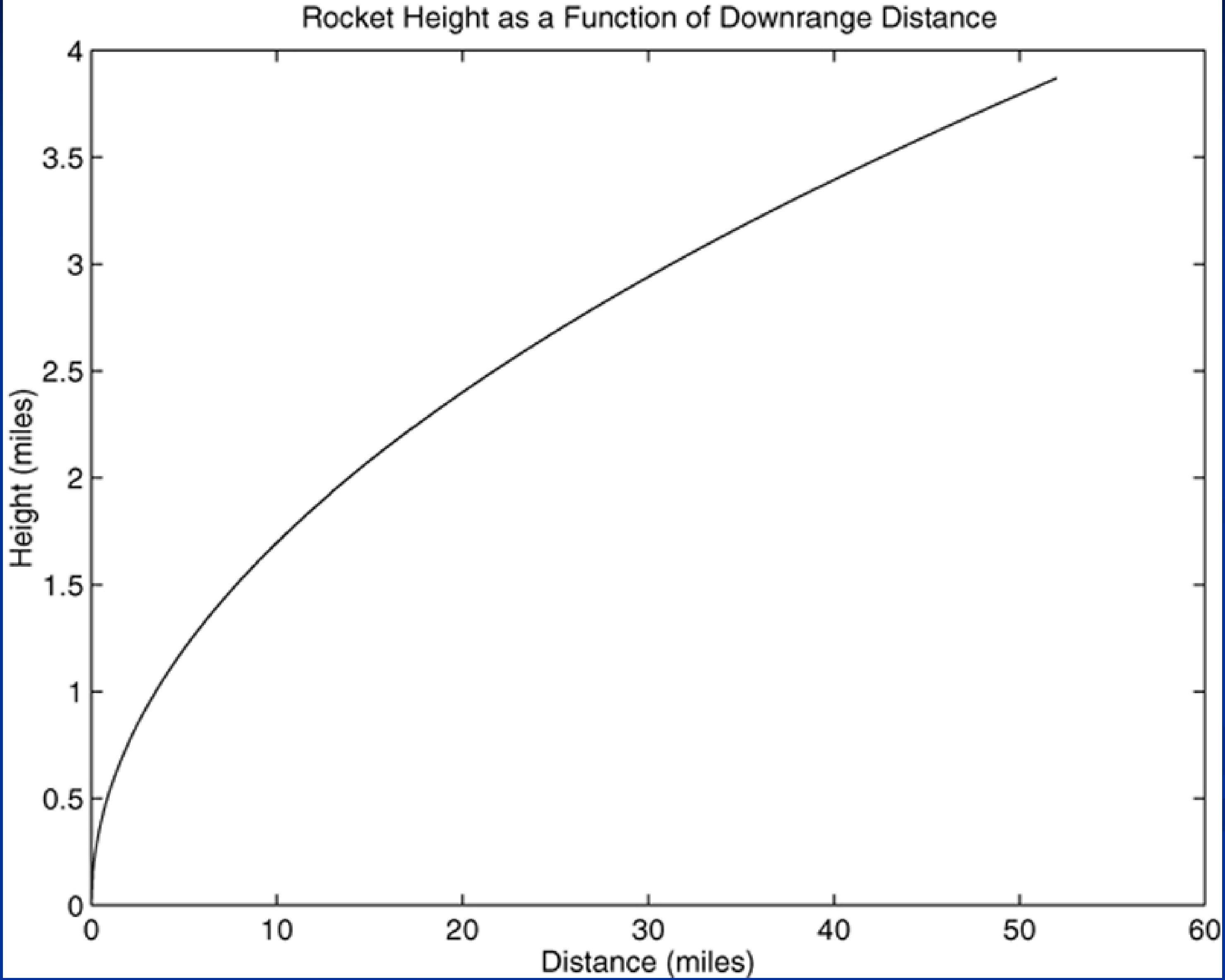


xy Plots: The following MATLAB session plots $y = 0.4\sqrt{1.8x}$ for $0 \le x \le 52$,

where *y* represents the height of a rocket after launch, in miles, and *x* is the horizontal (downrange) distance in miles.

```
>>x = [0:0.1:52];
>>y = 0.4*sqrt(1.8*x);
>>plot(x,y)
>>xlabel(’Distance (miles)’)
>>ylabel(’Height (miles)’)
>>title(’Rocket Height as a Function of
  Downrange Distance’)
```

The resulting plot is shown on the next slide.


Rocket Height as a Function of Downrange Distance
Height (miles)
Distance (miles)
0
0.5
1
1.5
2
2.5
3
3.5
4
0
10
20
30
40
50
60

The `grid` and `axis` Commands

The `grid` command displays gridlines at the tick marks corresponding to the tick labels. Type `grid on` to add gridlines; type `grid off` to stop plotting gridlines. When used by itself, `grid` toggles this feature on or off, but you might want to use `grid on` and `grid off` to be sure.

You can use the `axis` command to override the MATLAB selections for the axis limits. The basic syntax is `axis([xmin xmax ymin ymax])`. This command sets the scaling for the *x*- and *y*-axes to the minimum and maximum values indicated. Note that, unlike an array, this command does not use commas to separate the values.

.

Additional Commands and Plot Types

Subplots

You can use the `subplot` command to obtain several smaller "subplots" in the same figure. The syntax is `subplot(m,n,p).` This command divides the Figure window into an array of rectangular panes with *m* rows and *n* columns. The variable `p` tells MATLAB to place the output of the `plot` command following the `subplot` command into the *p*th pane.

For example, `subplot(3,2,5)` creates an array of six panes, three panes deep and two panes across, and directs the next plot to appear in the fifth pane (in the bottom-left corner).

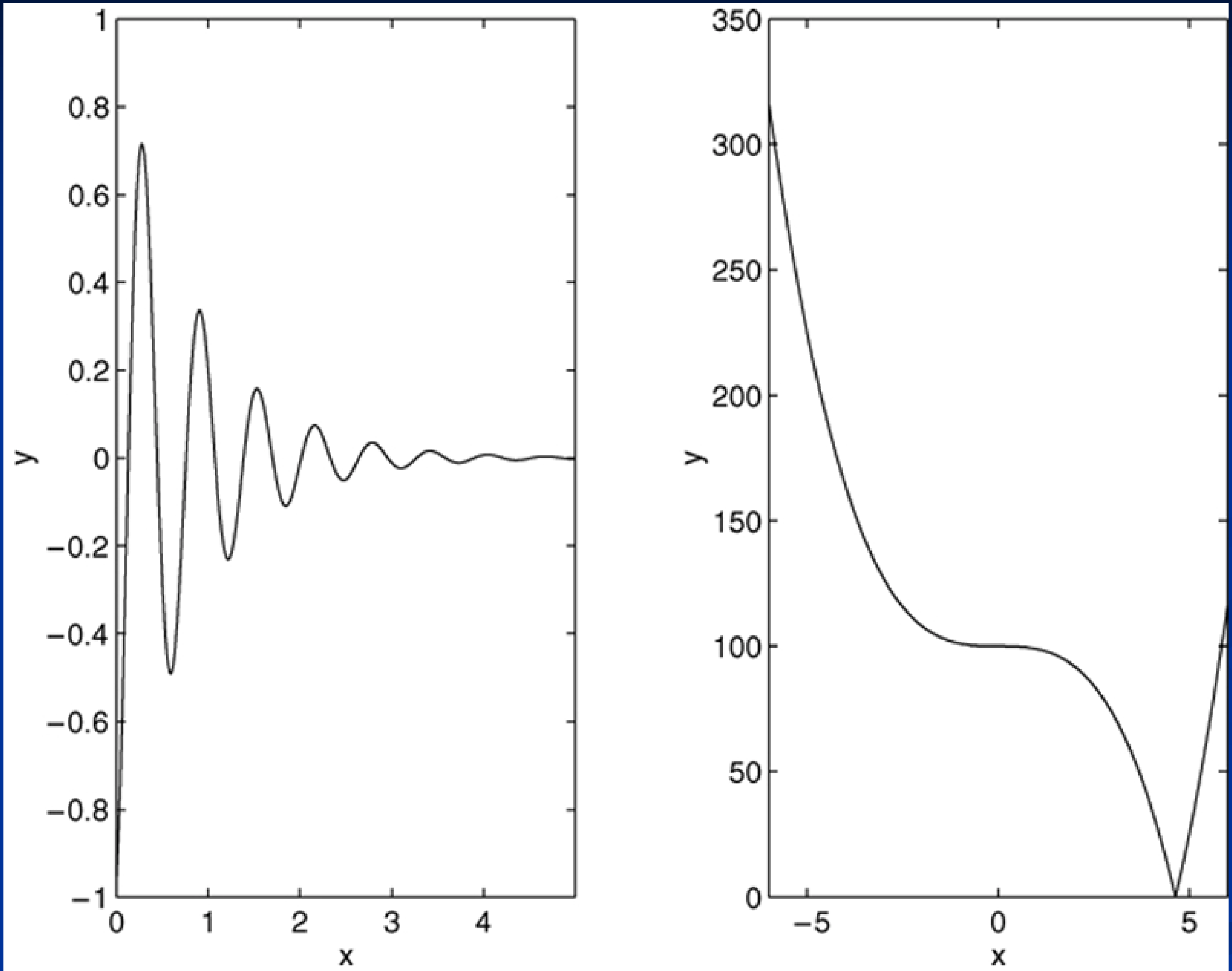
Example

The following script file created , shows the plots of the functions $y = e^{-1.2x} \sin(10x + 5)$ for $0 \leq x \leq 5$ and $y = |x^3 - 100|$ for $-6 \leq x \leq 6$.

```
x = [0:0.01:5];
y = exp(-1.2*x).*sin(10*x+5);
subplot(1,2,1)
plot(x,y),axis([0 5 -1 1])
x = [-6:0.01:6];
y = abs(x.^3-100);
subplot(1,2,2)
plot(x,y),axis([-6 6 0 350])
```

The figure is shown on the next slide.

Application of the `subplot` command.



Data Markers and Line Types

the symbols for line types are given in table.

Example

To plot `y` versus `x` with asterisks (*) connected with a dotted line, you must plot the data twice by typing

```
>> x = [0:0.01:5];
>> y = exp(-1.2*x).*sin(10*x+5);
>> plot(x,y,'*',x,y,':')
```

```
See next figure
```

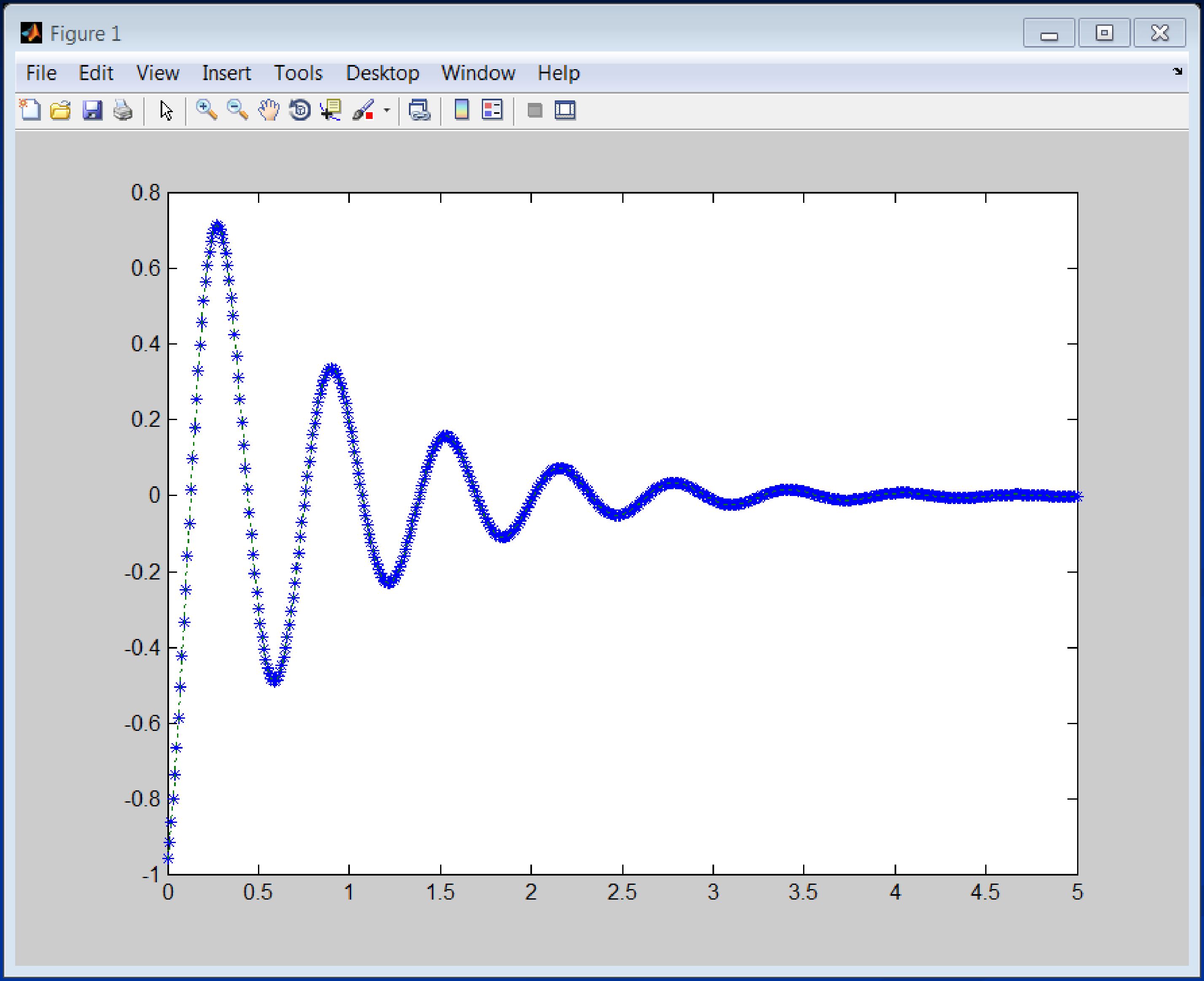


Figure 1
File
Edit
View
Insert
Tools
Desktop
Window
Help
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

Example

To plot `y` versus `x` with green asterisks (∗) connected with a red dashed line, you must plot the data twice by typing

```
>> x = [0:0.01:5];

>> y = exp(-1.2*x).*sin(10*x+5);

plot(x,y,’g*’,x,y,’r--’)
```

See figure on next page

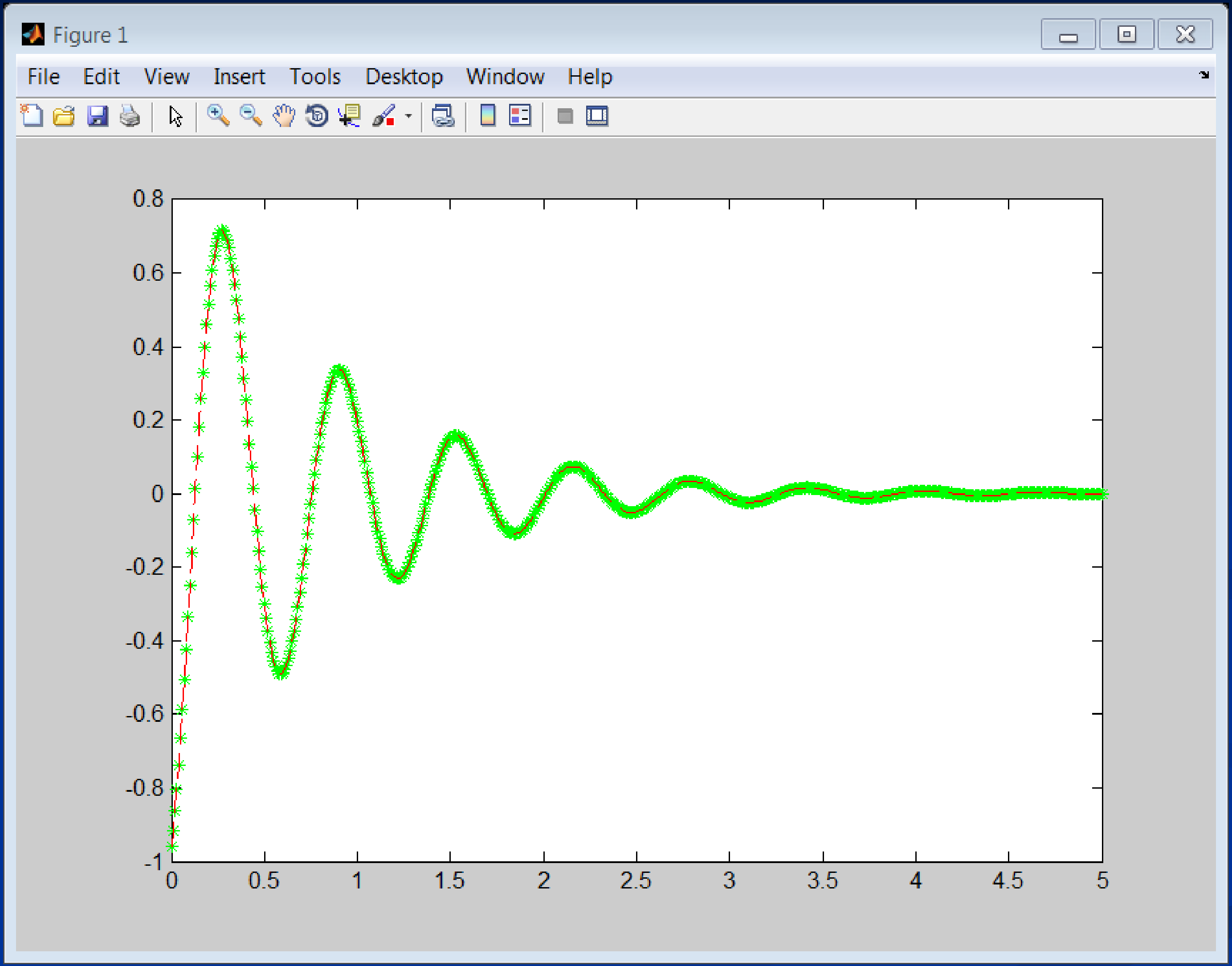


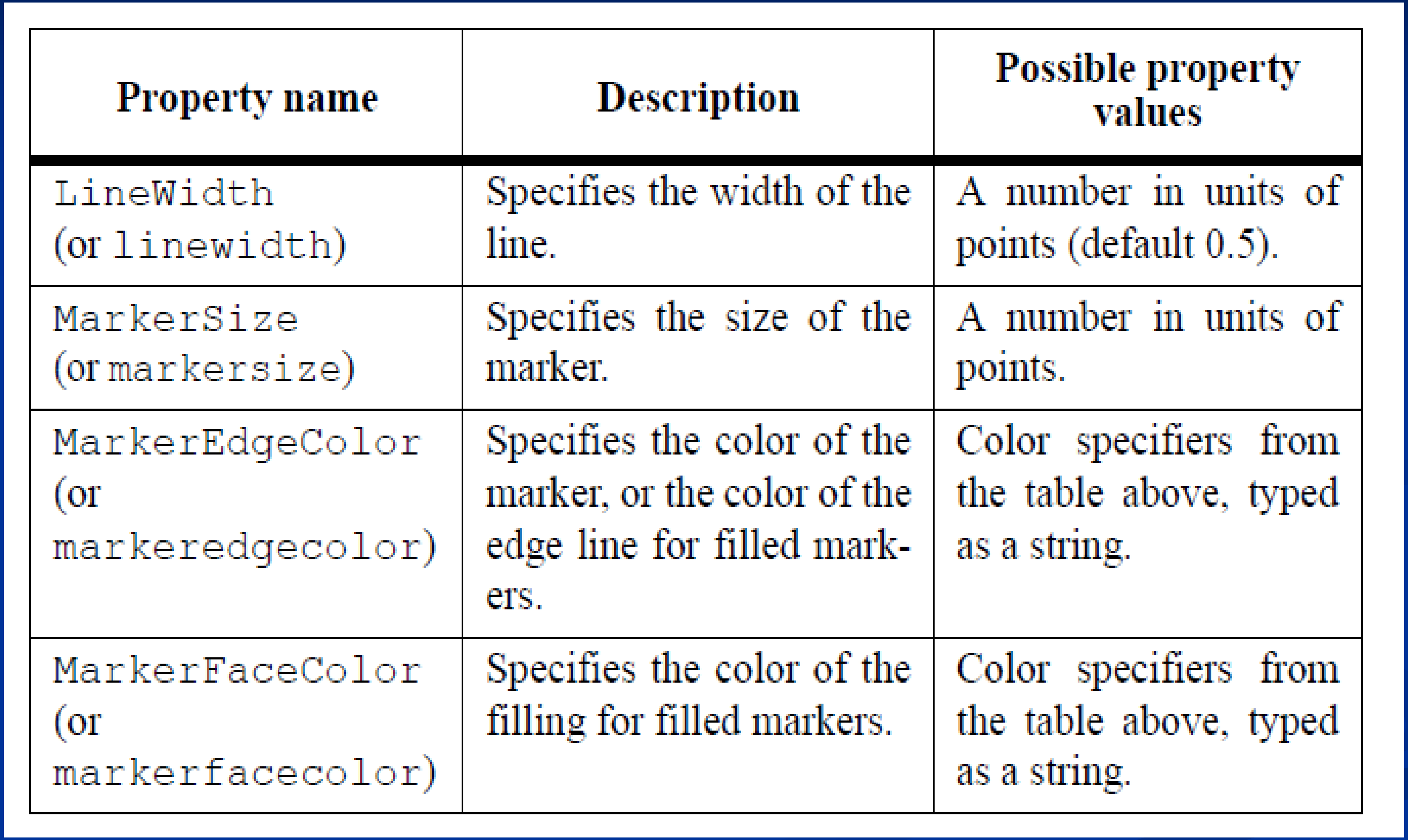
Figure 1
File
Edit
View
Insert
Tools
Desktop
Window
Help
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

Specifiers for data markers, line types, and colors.

Data markers†		Line types		Colors	
Dot (.)	.	Solid line	—	Black	k
Asterisk (*)	*	Dashed line	– –	Blue	b
Cross (×)	×	Dash-dotted line	– .	Cyan	c
Circle (◦)	◦	Dotted line	….	Green	g
Plus sign (+)	+			Magenta	m
Square (□)	s			Red	r
Diamond (◇)	d			White	w
Five-pointed star	p			Yellow	y

†Other data markers are available. Search for “markers” in MATLAB help.

Property Name and Property Value



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.

Examples

```
>> x=[0:0.1:2*pi];
>> y=cos(x);
 plot(x,y,'-mo','LineWidth',2,'MarkerSize',6,...
'MarkerEdgeColor','g','MarkerFaceColor','y')
 xlabel('x')
>> ylabel('cos(x)')
 >>title('plot of cos(x) vs x')
>> legend('cos(x)')
```

See figure on next slide

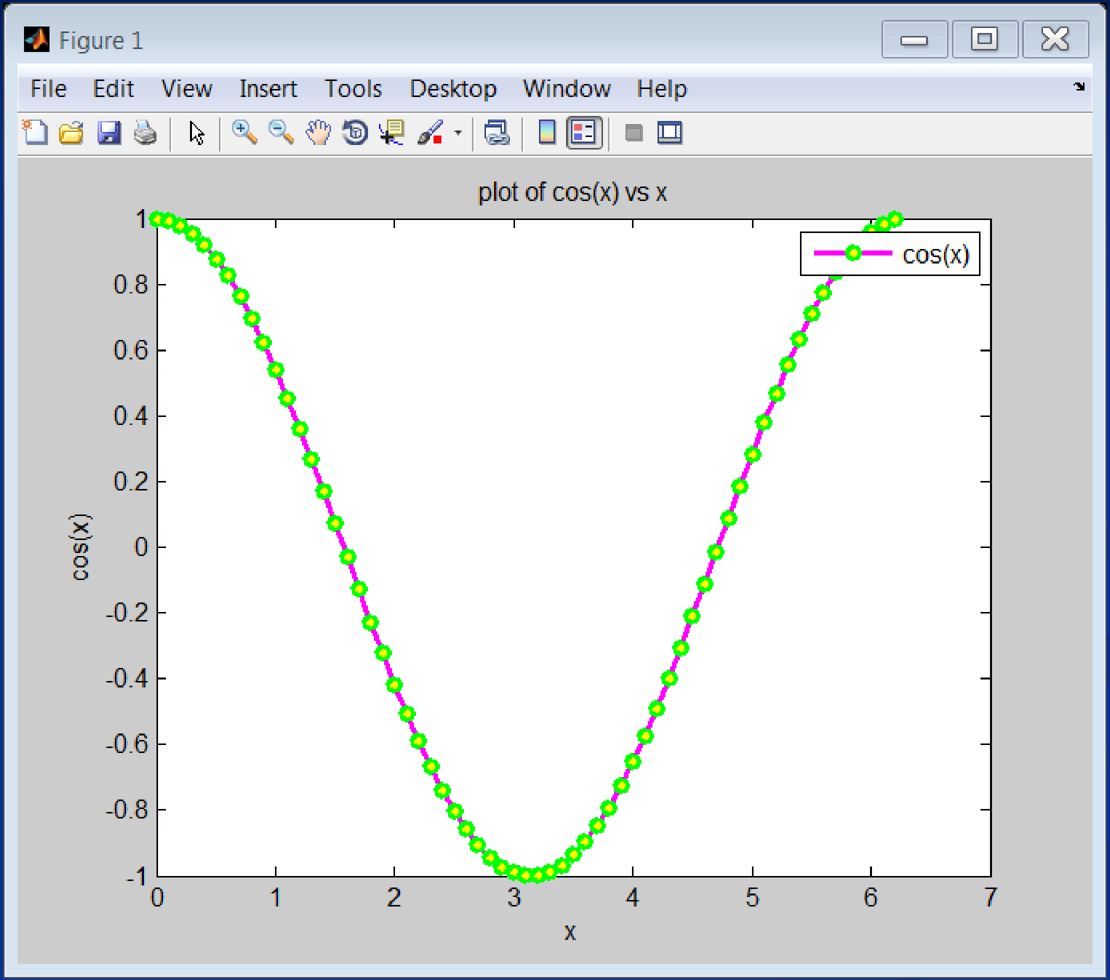
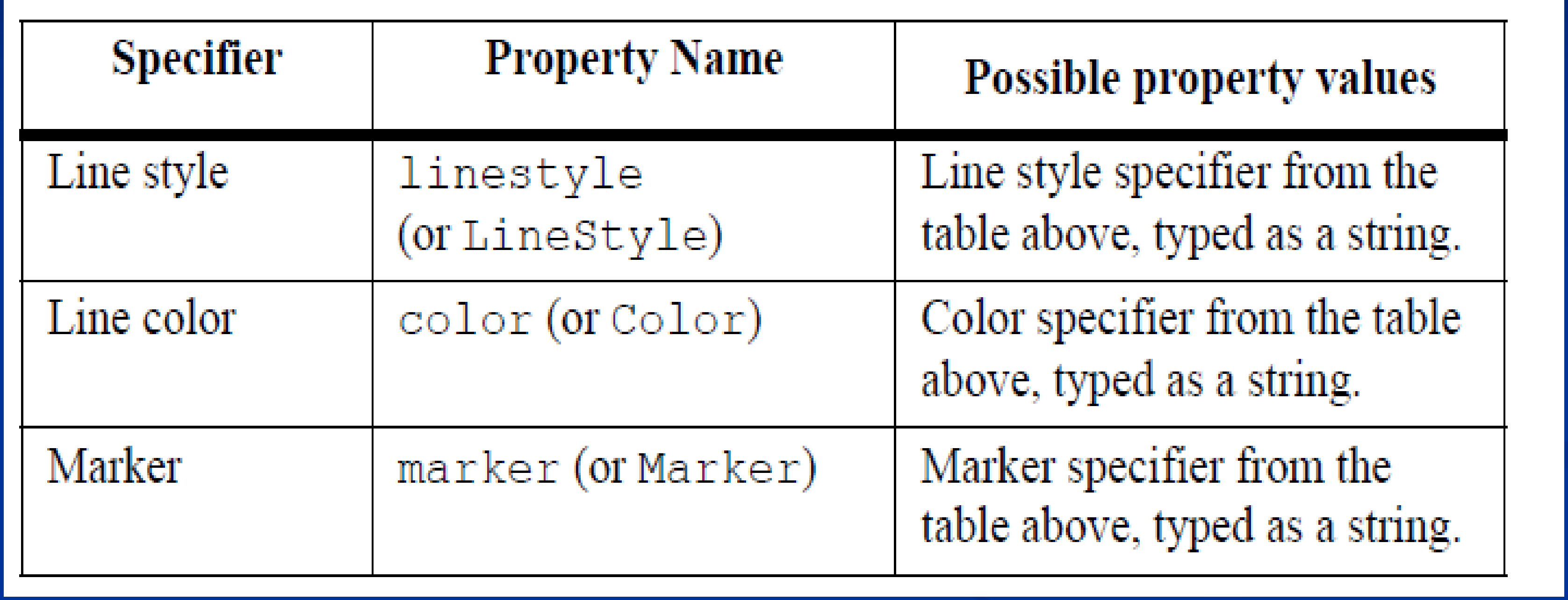


Figure 1
File Edit View Insert Tools Desktop Window Help
plot of cos(x) vs x
cos(x)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
1
2
3
4
5
6
7
x

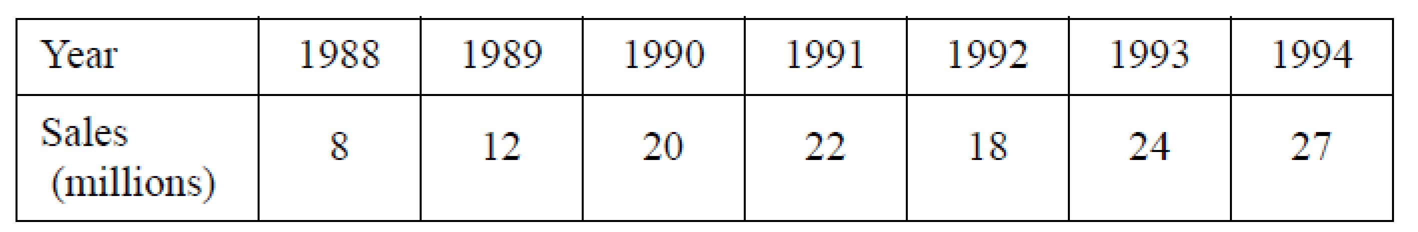
Line Specifiers and properties



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.

Example

Plot the following data .This data is related to 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

Solution

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

See next slide

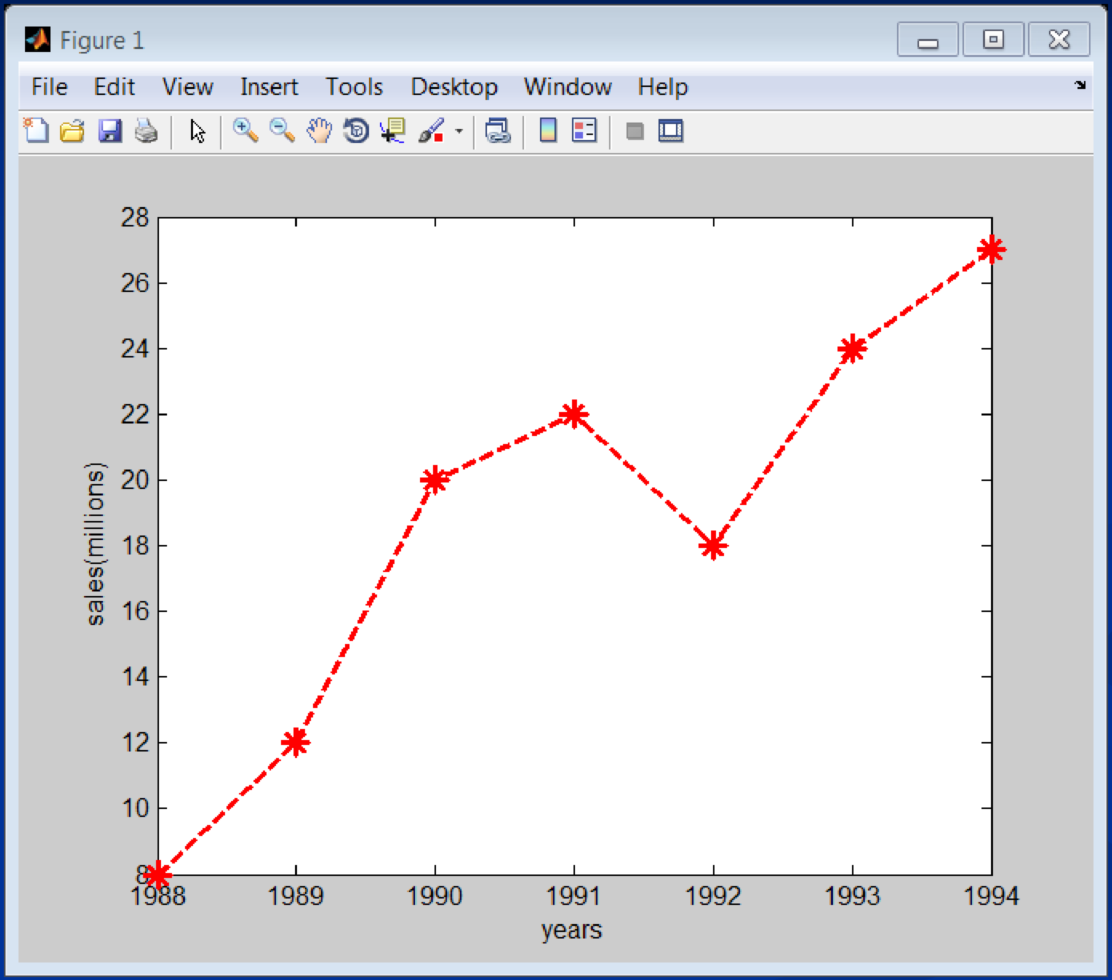


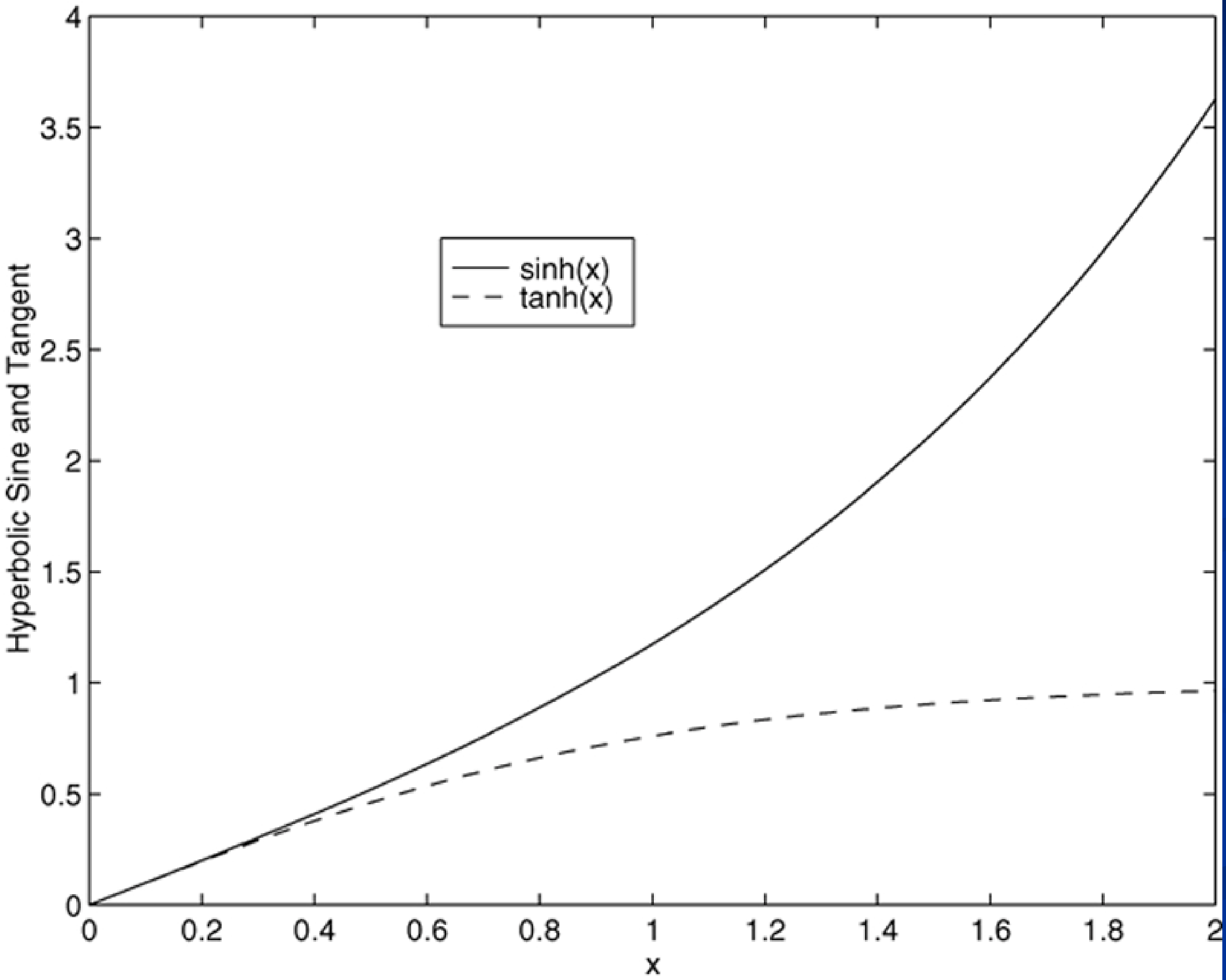
Figure 1
File
Edit
View
Insert
Tools
Desktop
Window
Help
28
26
24
22
20
18
16
14
12
10
8
sales(millions)
1988
1989
1990
1991
1992
1993
1994
years

Labeling Curves and Data

The `legend` command automatically obtains from the plot the line type used for each data set and displays a sample of this line type in the legend box next to the string you selected. The following script file produced the plot in Figure given on next slide.

Example

```
x = [0:0.01:2];
y = sinh(x);
z = tanh(x);
plot(x,y,x,z,’--’),xlabel(’x’), ...
ylabel(’Hyperbolic Sine and
Tangent’), ...
legend(’sinh(x)’,’tanh(x)’)
```


Hyperbolic Sine and Tangent
4
3.5
3
2.5
2
1.5
1
0.5
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x
sinh(x)
tanh(x)

hold command.

Example

```
>> x=[-1:0.01:1];
>> y1=3+exp(-x).*sin(6*x);
>> y2=4+exp(-x).*cos(6*x);
>> plot(x,y1,'.k')
>> hold on
>> plot(x,y2,'--r')
>> hold off
>> gtext('plot of y1(x)' )
>> gtext('plot of y2(x)')
```

See figure on next slide

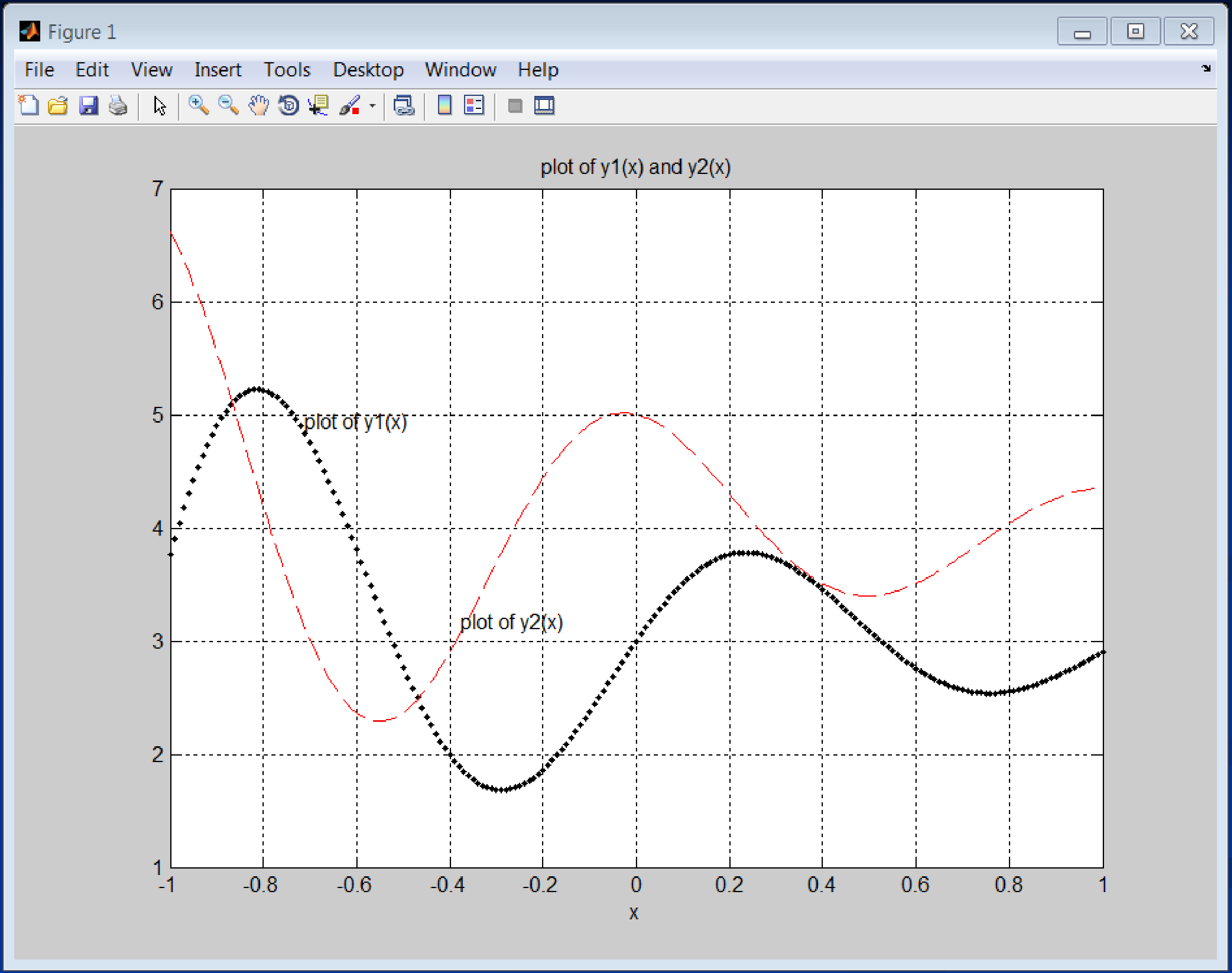


Figure 1
File Edit View Insert Tools Desktop Window Help
plot of y1(x) and y2(x)
7
6
5
4
3
2
1
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
x
plot of y1(x)
plot of y2(x)

Logarithmic Plots:

Why use log scales?

Rectilinear scales cannot properly display

variations over wide ranges.

Logarithmic Plots

It is important to remember the following points when using log scales:

1. You cannot plot negative numbers on a log scale, because the logarithm of a negative number is not defined as a real number.

2. You cannot plot the number 0 on a log scale, because $\log_{10} 0 = \ln 0 = -\infty$. You must choose an appropriately small number as the lower limit on the plot.

(continued…)

Logarithmic Plots (continued)

3. The tick-mark labels on a log scale are the actual values being plotted; they are not the logarithms of the numbers. For example the range of x values in the log-log plot (see slide) is from $10^{-1} = 0.1 \ \ to\, 10^{2}$

4. Gridlines and tick marks within a decade are unevenly spaced. If 8 gridlines or tick marks occur within the decade, they correspond to values equal to 2, 3, 4, $. \ . \ . \ .$, 8, 9 times the value represented by the first gridline or tick mark of the decade.

(continued…)

Logarithmic Plots (continued)

5. Equal distances on a log scale correspond to multiplication by the same constant (as opposed to addition of the same constant on a rectilinear scale).

For example, all numbers that differ by a factor of 10 are separated by the same distance on a log scale. That is, the distance between 0.3 and 3 is the same as the distance between 30 and 300. This separation is referred to as a *decade* or *cycle.*

The plot shown in slide covers three decades in *x* (from 0.1 to 100) and four decades in *y* and is thus called a *four-by-three-cycle plot.*

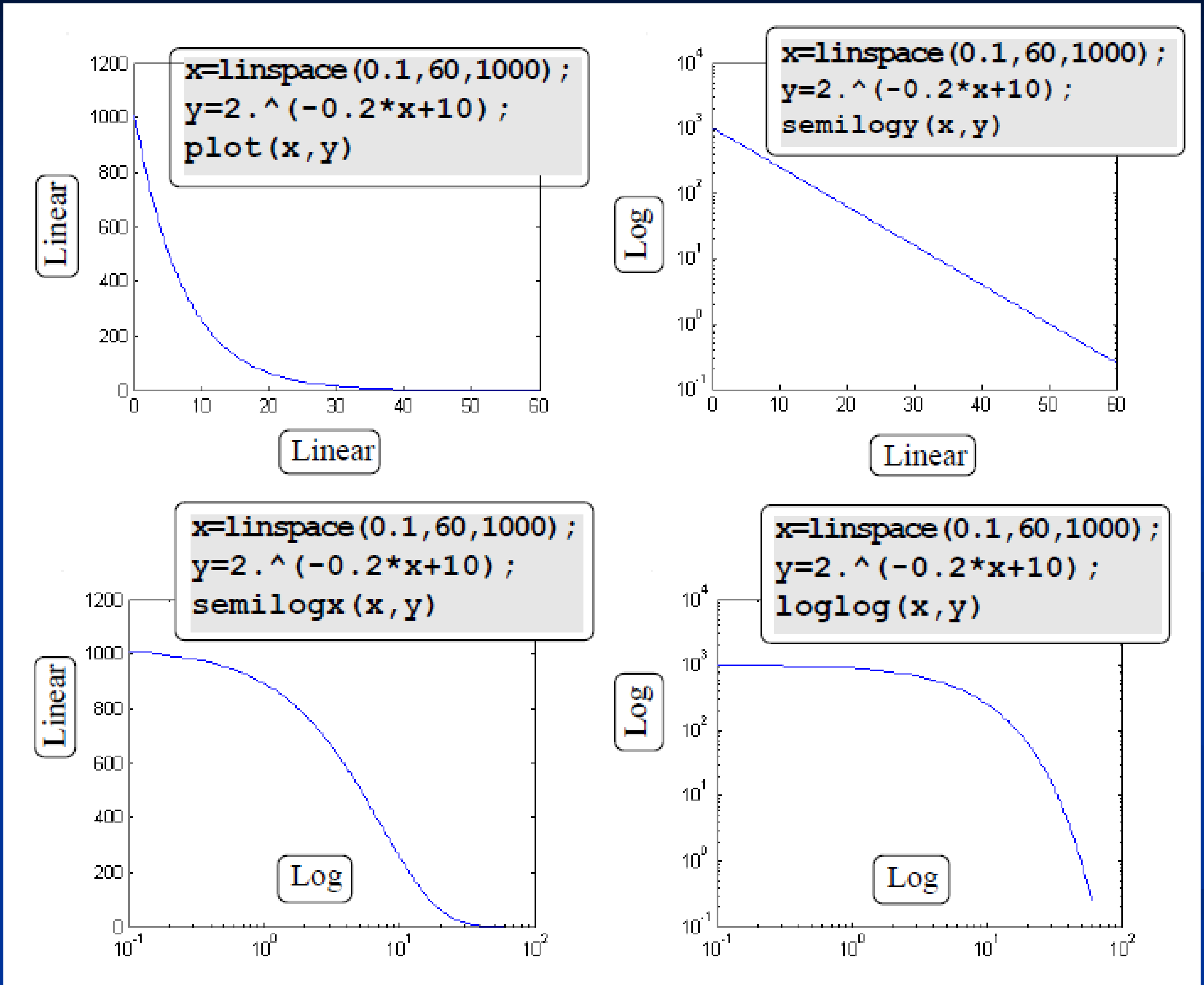
MATLAB has three commands for generating plots having log scales. The appropriate command depends on which axis must have a log scale.

1. Use the `loglog(x,y)` command to have both scales logarithmic.

2. Use the `semilogx(x,y)` command to have the *x* scale logarithmic and the *y* scale rectilinear.

3. Use the `semilogy(x,y)` command to have the *y* scale logarithmic and the *x* scale rectilinear.

Plots of $y = 2^{(-0.2x+10)}$ with linear, semilog, and log-log scales.



```
x=linspace(0.1,60,1000);
y=2.^(-0.2*x+10);
plot(x,y)
```

```
x=linspace(0.1,60,1000);
y=2.^(-0.2*x+10);
semilogy(x,y)
```

```
x=linspace(0.1,60,1000);
y=2.^(-0.2*x+10);
semilogx(x,y)
```

```
x=linspace(0.1,60,1000);
y=2.^(-0.2*x+10);
loglog(x,y)
```

Specialized plot commands.

Command	Description
`bar(x,y)`	Creates a bar chart of `y` versus `x`.
`plotyy(x1,y1,x2,y2)`	Produces a plot with two *y*-axes, `y1` on the left and `y2` on the right.
`polar(theta,r,’type’)`	Produces a polar plot from the polar coordinates `theta` and `r`, using the line type, data marker, and colors specified in the string `type`.
`stairs(x,y)`	Produces a stairs plot of `y` versus `x`.
`stem(x,y)`	Produces a stem plot of `y` versus `x`.

Polar Plot

Polar plots are two dimensional plots made using polar plots. If the polar coordinates are (r, θ) where θ is the angular coordinate and r is the radial coordinate then the command polar(theta,r) will pruduce the polar plot.

Example

Plot the equation

$$r = \frac{p}{1 - \epsilon \cos(\theta)}$$

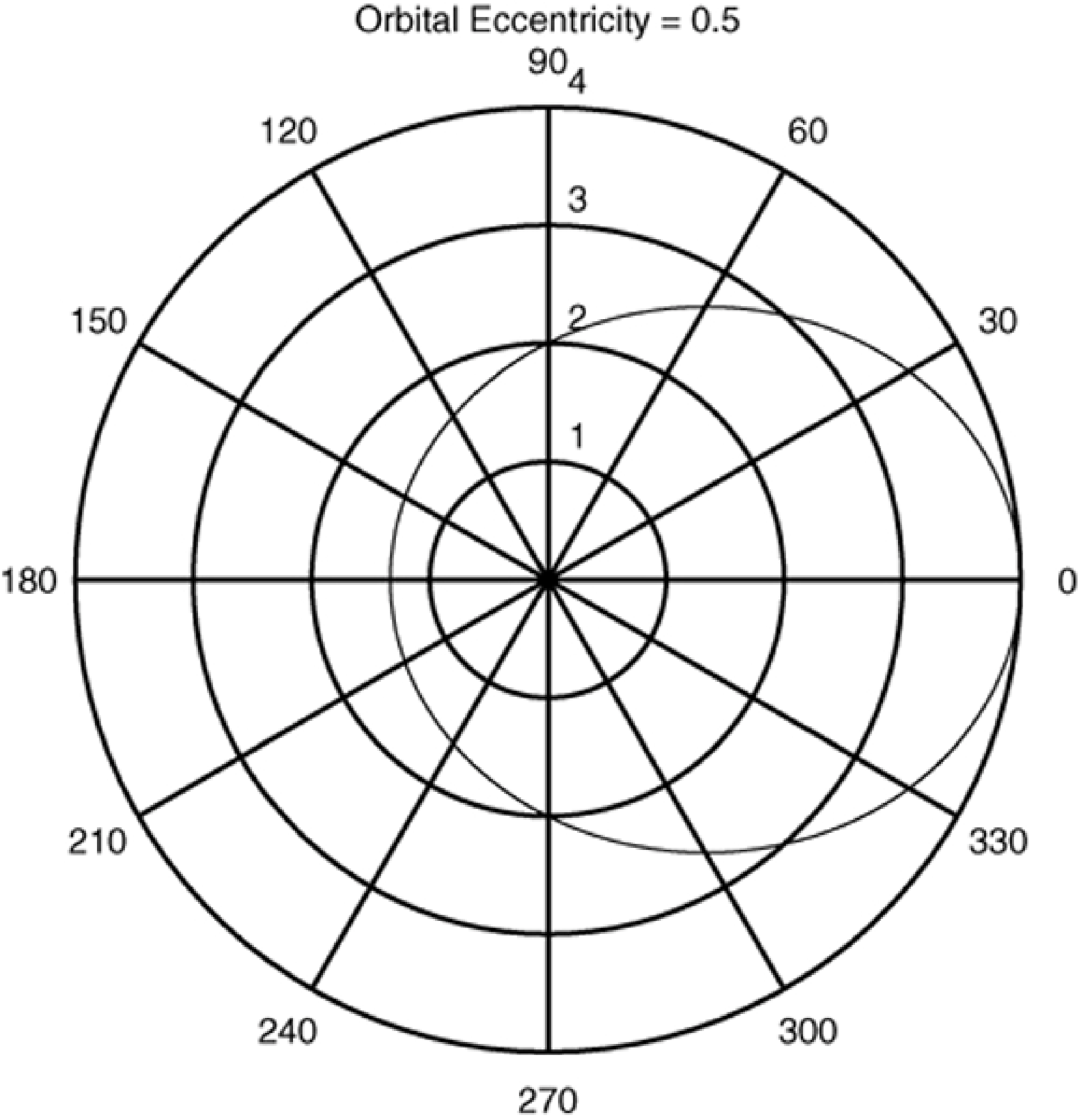
Where $\in = 0.5$ and p=2.

Solution

```
>> clear
>> theta=[0:pi/90:2*pi];
>> r=2./(1-0.5*cos(theta));
>> polar(theta,r),title('Orbital eccentricity=0.5')
```

See next slide

A polar plot


Orbital Eccentricity = 0.5
90
4
120
60
3
150
30
2
1
180
0
210
330
240
300
270