Programlama -1

“ Curve Plotting”

Dr. Cahit Karakuş, 2020

Vector Lengths

- Başlangıçta vurgulanması gereken çok önemli bir gerçek, bir vektörü diğerine karşı çizmek için vektörlerin aynı sayıda öğeye sahip olması gerektiğidir. Aynı sayıda değere sahip olmaları koşuluyla, bir sütun vektörü veya bir satır vektörüne karşı bir sütun vektörü veya bir satır vektörü çizilebilir.
- İki boyutlu çizim komutlarında, yatay eksene x ekseni ve dikey eksene y ekseni olarak atıfta bulunulacaktır. Bununla birlikte, gerçek değişkenler herhangi bir nicelikle etiketlenebilir. Yalnızca çizim komutlarında x ve y kullanılır.

Command for Linear Array

- >> x = x1:xstep:x2
- where x1=beginning point, x2=final point, and xstep=step size. Assuming that the final point coincides with an integer multiple of xstep, the number of points *N* is

$$N = \frac{x_2 - x_1}{x_{step}} + 1$$

Alternate Command for Linear Array

- >> x = linspace(x1, x2, N)
- where x1=beginning point, x2=final point, and N=number of points. The name linspace represents "linear spacing". Again, the number of points *N* is

$$N = \frac{x_2 - x_1}{x_{step}} + 1$$

Plot komutu ile grafik çizme

- Plot komutunun genel kullanımı
- xlabel komutu ile x-ekseninin adlandırılması
- ylabel komutu ile ekseninin adlandırılması
- title komutu ile grafiğe isim verilmesi
- renk, şekil, kalınlık gibi grafiklerin özelliklerinin değiştirilmesi
- holdon komutu ile tek bir penceredе birden fazla grafik çizdirilmesi
- Grid komutu ile yatay ve dikey bölümlendirme
- Axis komutu ile eksen ölçeklendirme

plot Komutu ile Grafik Çizimi

genel kullanımı

- **İki boyutlu grafik çiziminde kullanılır.**
- **plot (x , y)**

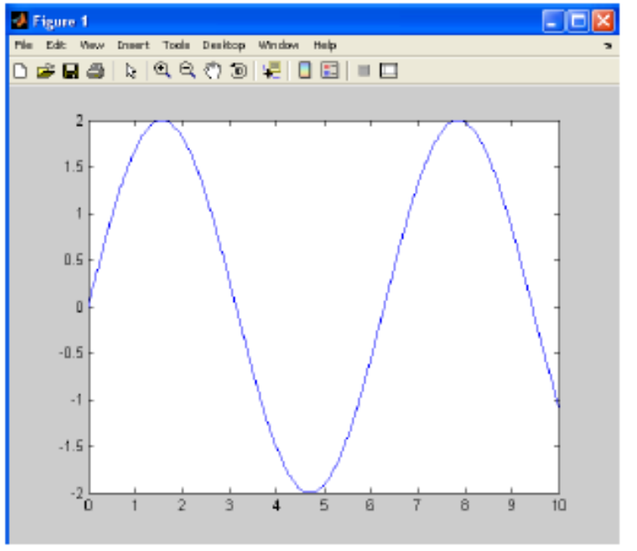
  y eksenine ait vektörel ifade

  x eksenine ait vektörel ifade

- Örnek: u(t) = 2Sin(ωt) sinyalini 0.01 adımlarla, 0 ile 10 sn zaman dilimi için çiziniz? Not: ω = 1

Komut penceresi

```
% 0.01 artışlar ile 0 – 10 sn zaman diliminin tanımlanması
>> t = 0: 0.01 : 10;

% Grafiğin y eksenini oluşturacak u(t) sinyalinin tanımlanması
>> u = 2*sin(t);

% Grafiğin çizdirilmesi
>> plot(t,u)
```



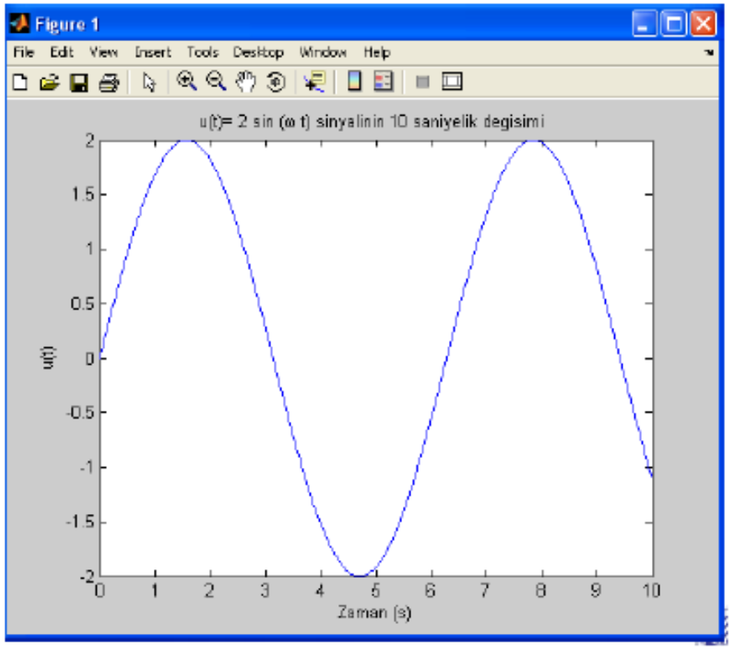
plot Komutu ile Grafik Çizimi

grafiklere ve eksenlere isim verilmesi

- Çizdirilen grafiklerin daha anlamlı olması için, grafiklere başlık ve x ile y eksenine de isim verilmesi gerekir.
  - title (' Grafiğin başlığı ')
  - xlabel (' x ekseninin etiketi ')
  - ylabel (' y ekseninin etiketi ')
- Önceki örnek çizdirilen grafik üzerinde isim verilmesi:

Komut penceresi

```
% Grafik üzerinde eksen açıklamalarının yapılması
>> xlabel ('Zaman (s) ')

>> ylabel (' u(t) ')

% Grafiğe başlık verilmesi
>> title ('u(t)= 2 sin (\omega t)
sinyalinin 10 saniyelik değişimi ')
```



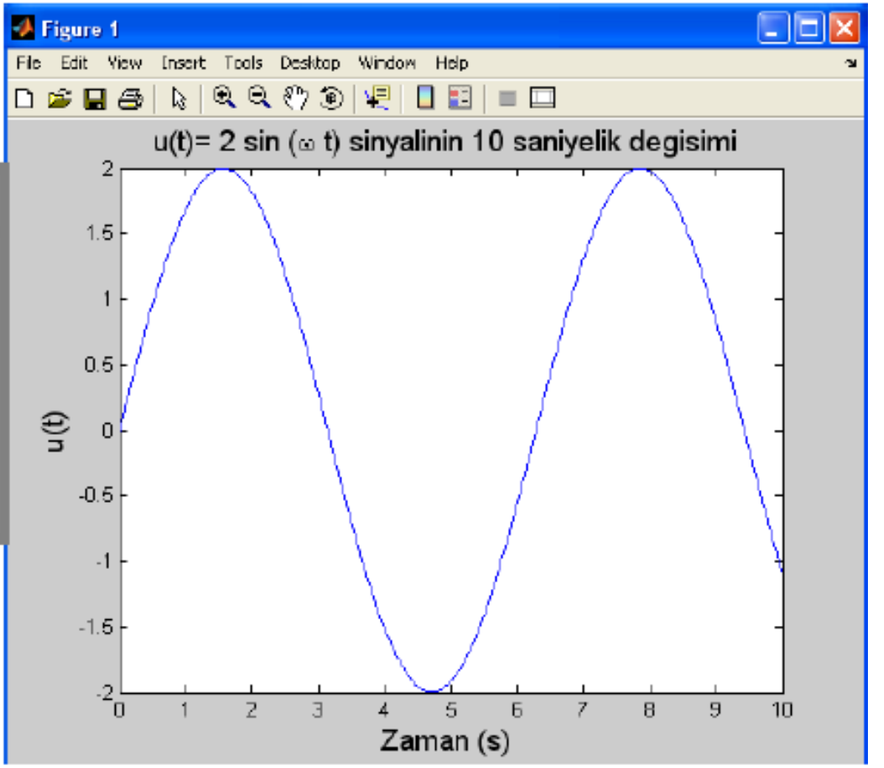
plot Komutu ile Grafik Çizimi

grafik başlık ve eksen isimlerinin boyutlandırılması

- **Bazı durumlarda eksen ve başlık isimlerinin daha koyu yazdırılması istenebilir. Bu durumda yazının büyüklük (font) ayarı değiştirilmelidir.**
  - **fontsize (' istenen punto ')**

Komut penceresi

```
% Grafik üzerinde eksen ve başlık  açıklamalarının 14 punto yazılması
>> xlabel ('Zaman (s) ', 'fontsize', [14])

>> ylabel (' u(t) ', 'fontsize', [14])

>> title ('u(t)= 2 sin (\omega t) sinyalinin 10
saniyelik değişimi ', 'fontsize', [14])
```



plot Komutu ile Grafik Çizimi

grafik çizgi-işaret stillerinin değiştirilmesi

- **plot komutu ile grafikler düz çizgi tarzındadır.**
- **Farklı türde çizgi ve işarete sahip grafik çizdirmek için plot komutu aşağıdaki gibi kullanılmalıdır.**

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

çizimde kullanılacak çizgi / renk tanımlaması
y eksenine ait vektörel ifade
x eksenine ait vektörel ifade

Çizgi çeşitleri

Düz çizgi	-	İki noktalı	:
Kesikli çizgi	--	Kesikli-noktalı	-.

İşaret çeşitleri

Nokta	.	Üçgen (aşağı)	v
Artı	+	Üçgen (yukarı)	^
Yıldız	*	Üçgen (sola)	<
Daire	o	Üçgen (sağa)	>
x-işareti	x	Beş köşeli	p
Kare	s	Altı köşeli	h
Baklava şekli	d		

Example 1. When air resistance can be ignored, the velocity (in m/s) of an object falling from rest is

$$v = 9.8t$$

- Use MATLAB to plot the velocity over a time interval from 0 to 10 s.

Example 1. Continuation.

- It should be emphasized that this is a simple linear equation with a vertical intercept of 0 so we actually **need only two points to plot the curve**. However, our purpose is to learn how to use MATLAB for plotting and we will utilize far more points than necessary as a learning process.

Example 1. Continuation.

- A time step of 0.1 s will be selected.

```
t = 0:0.1:10;
```

- Alternately,

```
t = linspace(0,10,101);
```

- t(1:5)

```
ans =   0    0.1000    0.2000    0.3000    0.4000
```

Example 1. Continuation.

- >> v = 9.8*t;
- This command generates 101 values of v corresponding to the 101 values of t. It can be plotted by the command
- >> plot(t, v)
- The result is a “raw” plot but various labels can be added as will be shown on the next slide.

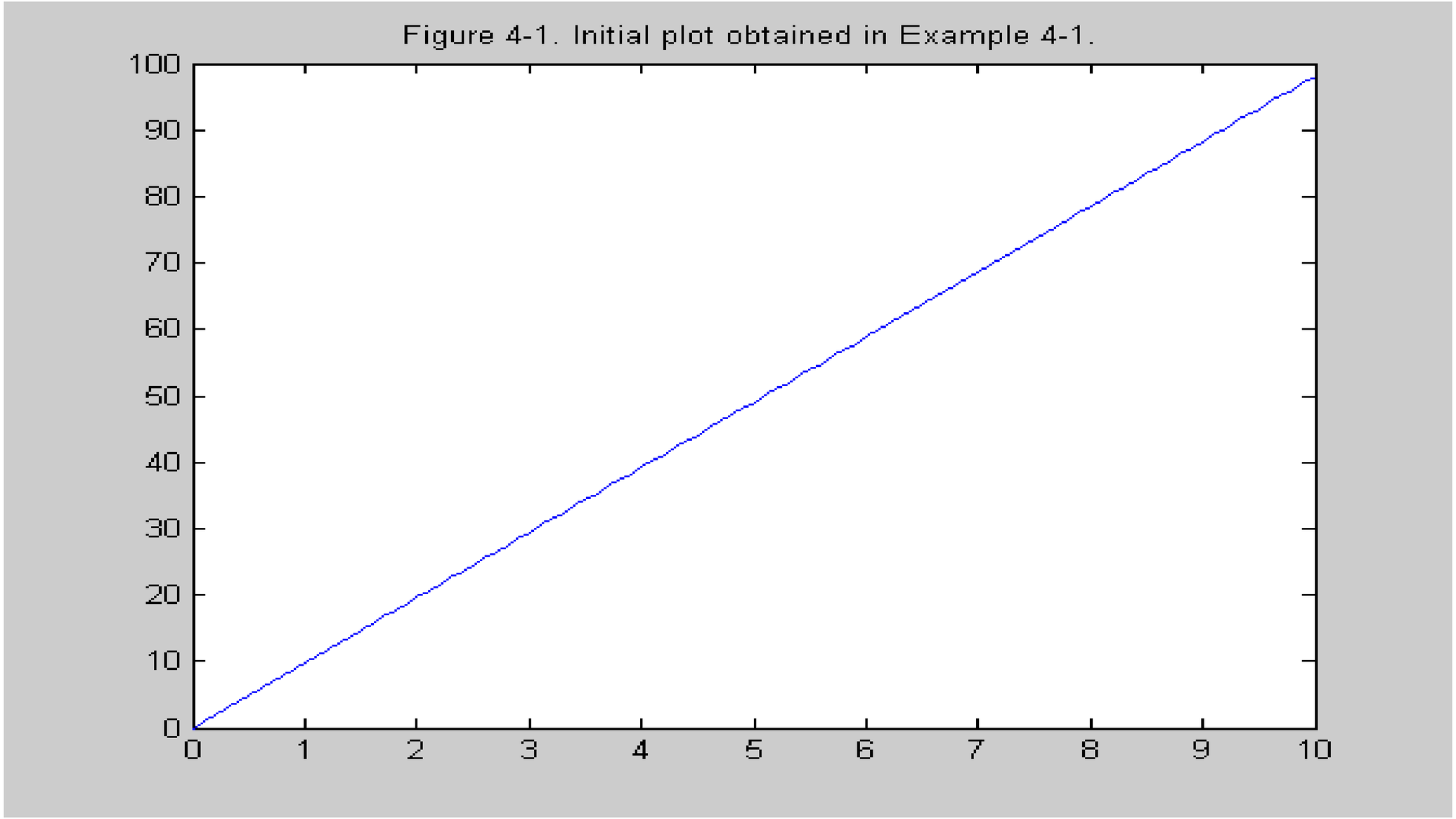


Figure 4-1. Initial plot obtained in Example 4-1.
100
90
80
70
60
50
40
30
20
10
0
0
1
2
3
4
5
6
7
8
9
10

Example 1. Continuation.

- A horizontal label is provided.
- >> xlabel(‘Time, seconds’)
- A vertical label is provided.
- >> ylabel(‘Velocity, meters/second’)
- A title is provided.
- >> title(‘Figure 4-3. Velocity of falling object of Example 4-1 with grid.’)
- A grid is added.
- >> grid

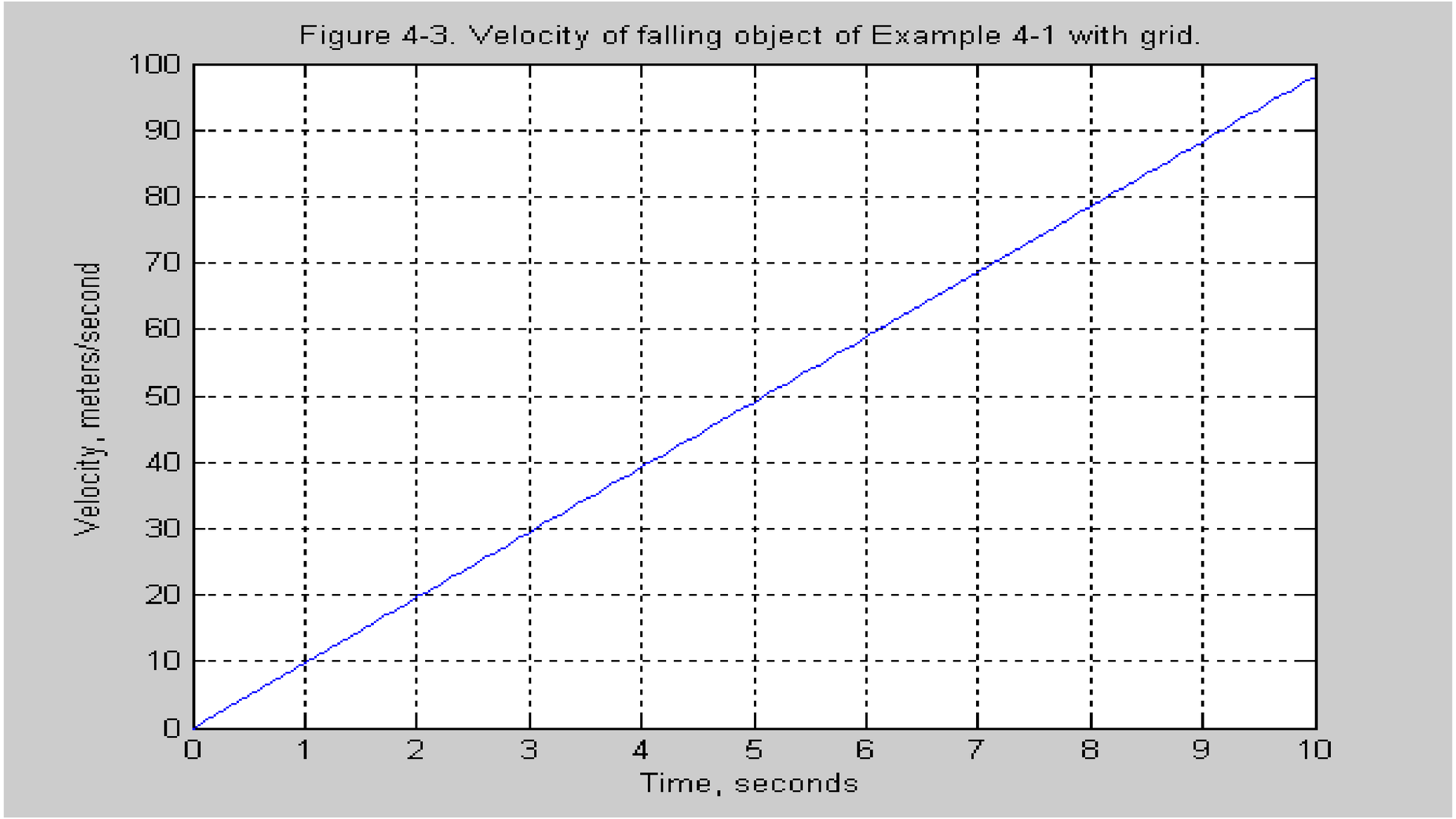


Figure 4-3. Velocity of falling object of Example 4-1 with grid.
Velocity, meters/second
100
90
80
70
60
50
40
30
20
10
0
0
1
2
3
4
5
6
7
8
9
10
Time, seconds

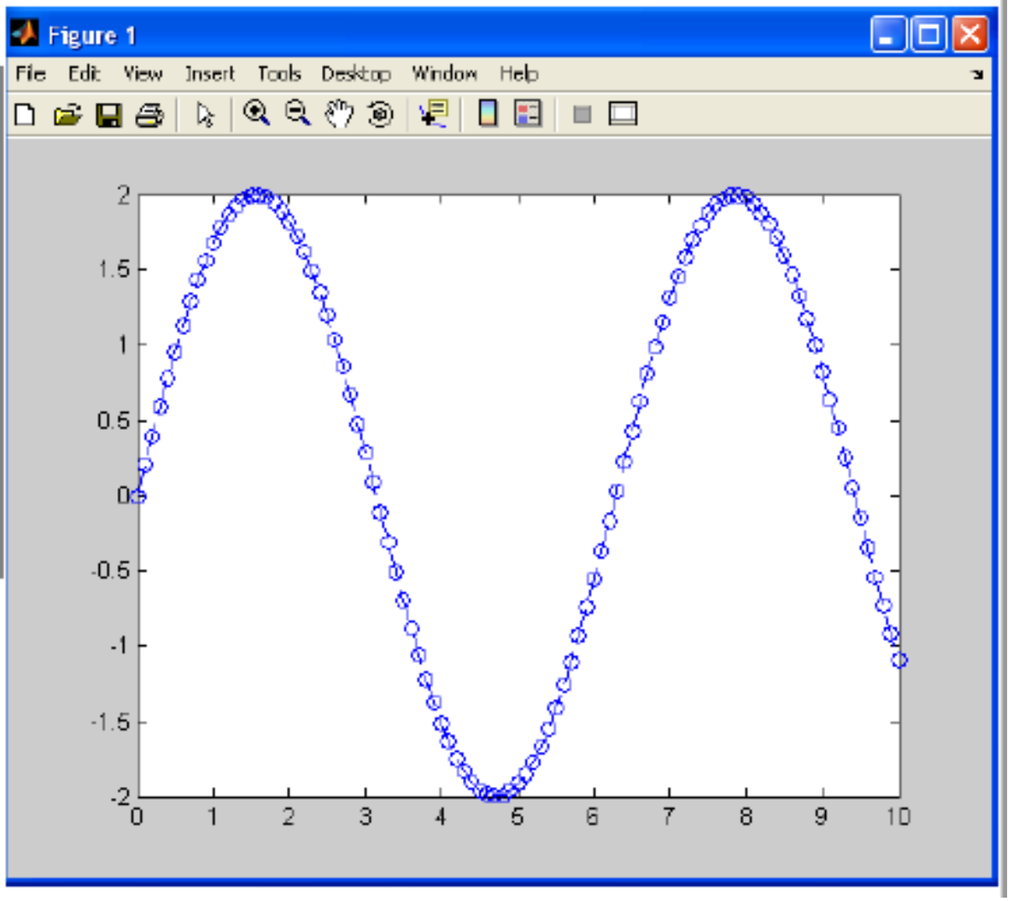
plot Komutu ile Grafik Çizimi

grafik çizgi-işaret stillerinin değiştirilmesi

- **Örnek: plot komutu ile kesik çizgili ve daire işaretlerine sahip grafik çizimi.**

Komut penceresi

```
% 0.1 artışlar ile 0 – 10 sn zaman diliminin tanımlanması
>> t = 0: 0.1 : 10;

% Grafiğin y eksenini oluşturacak u(t) sinyalinin tanımlanması
>> u = 2*sin(t);

% Grafiğin kesik çizgili ve daire işaretleri ile çizdirilmesi
>> plot(t,u, '-- o')
```



plot Komutu ile Grafik Çizimi

grafik çizgi renklerinin değiştirilmesi

- **Renk tanımlamalarında genel olarak, renklere ait ingilizce kelimelerin baş karakterleri kullanılmaktadır. Örneğin kırmızı için red 'r'**

- **Standart renk tanımlamalarının dışında [r g b] bir başka değişle [kırmızı yeşil mavi] şeklinde vektörel tanımlamada yapılabilir.**
- **Vektörel tanımlamadaki sayısal değerler 0 ile 1 arasında olmalıdır. 0 rengin olmayacağını, 1 oluşturulacak renk içerisindeki ana rengi gösterir. 0.4 gibi bir değer ise o renge ait katkı miktarını gösterir.**

Renk çeşitleri

Kırmızı	**r**	Beyaz	**w**
Yeşil	**g**	Siyah	**k**
Mavi	**b**	Çıyan	**c**
Sarı	**y**	Maganda	**m**
Görünmez	**i**		

Vektörel olarak tanımlanmış bazı renk çeşitleri

[1 1 1] Beyaz	[1 0 0] Kırmızı	[1 1 0] Sarı
[0 0 0] Siyah	[0 1 0] Yeşil	[1 0 1] Pembe
	[0 0 1] Mavi	[0 1 1] Açık mavi

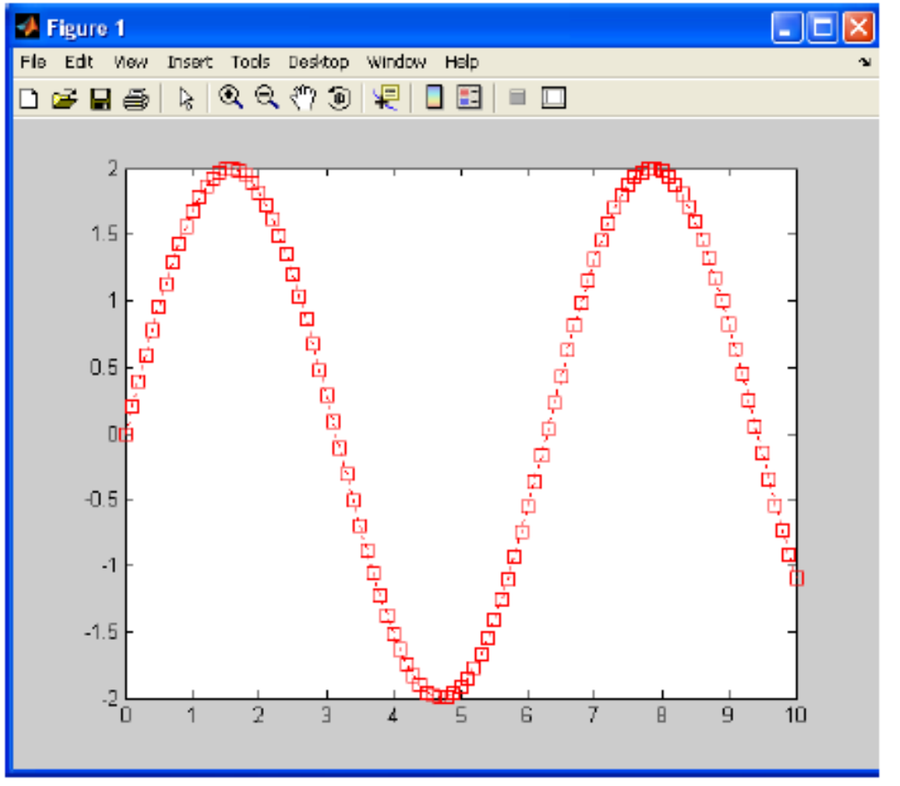
plot Komutu ile Grafik Çizimi

grafik çizgi-işaret ve renk stillerinin değiştirilmesi

- **Örnek: plot komutu ile iki noktalı çizgili ve kare işaretlerine sahip kırmızı renkli grafik çizimi.**

Komut penceresi

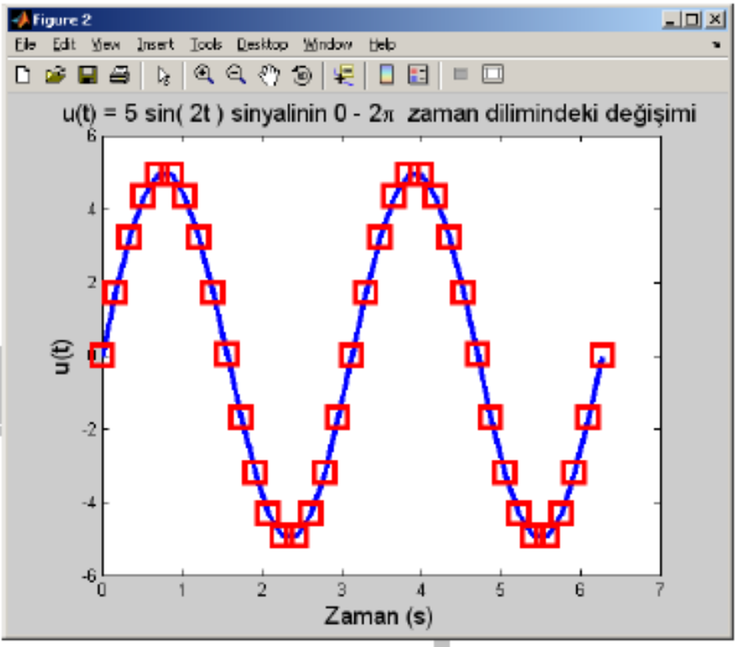
```
% 0.1 artışlar ile 0 – 10 sn zaman diliminin tanımlanması
>> t = 0: 0.1 : 10;

% Grafiğin y eksenini oluşturacak u(t) sinyalinin tanımlanması
>> u = 2*sin(t);

% iki noktalı, kare işaretli ve kırmızı renkte grafik
>> plot(t,u, ': s r ')
```



plot Komutu ile Grafik Çizimi

plot komutunun diğer özellikleri

```
plot(x,y,'LineStyle', '--',...
         'Color','r',...
         'LineWidth',2,...
         'Marker','square',...
         'MarkerEdgeColor','k',...
         'MarkerFaceColor','g',...
         'MarkerSize',10)
```



PROGRAM 3.3

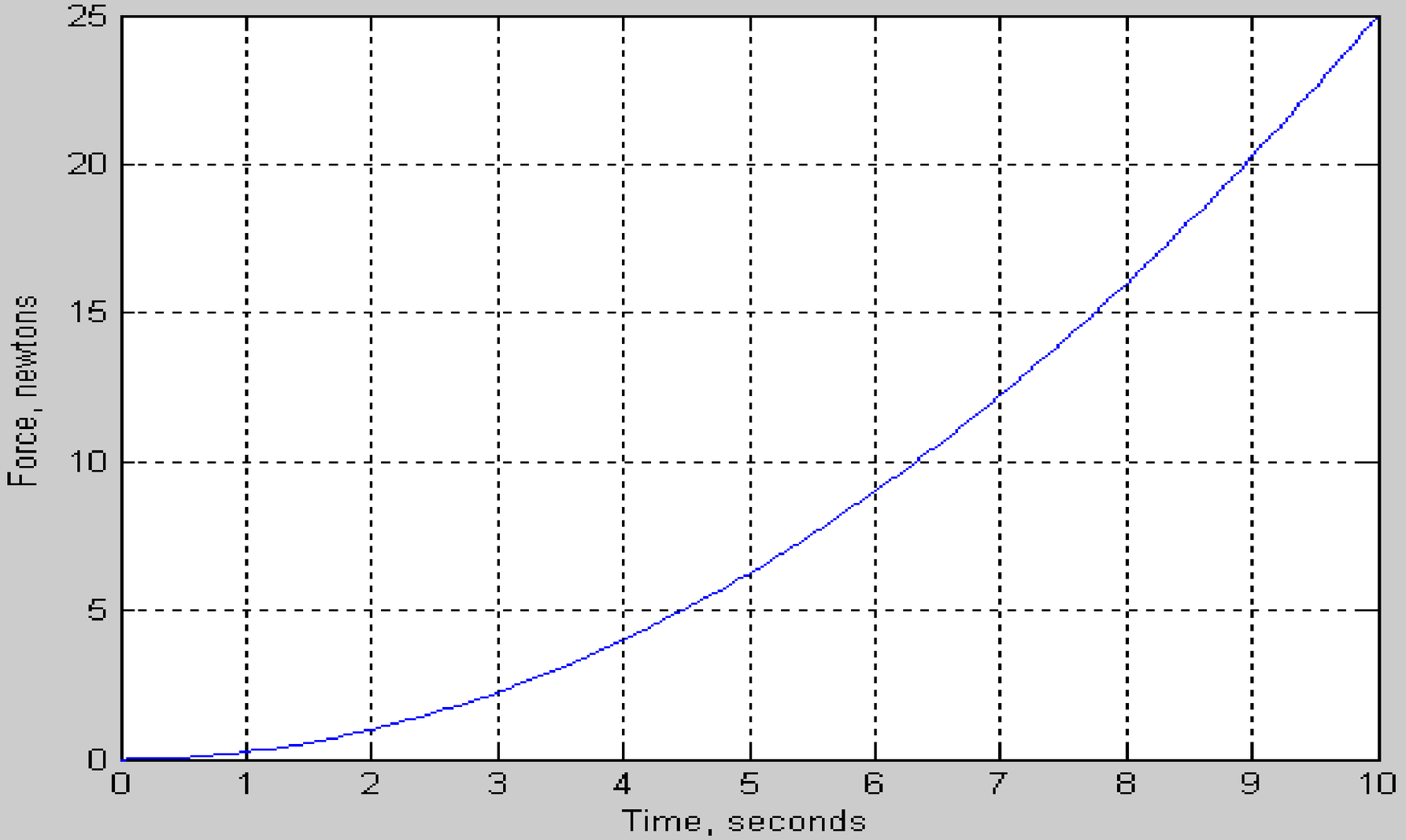
```
1    % Grafik çiziminde yer alan u(t) sinyalinin tanımlanması
2  - t = 0:pi/18:2*pi;
3  - u = 5*sin(2*t);
4
5    % Grafik çiziminin plot komutu ile gerçekleştirilmesi
6  - plot(t,u,'-bs','linewidth',3,'MarkerEdgeColor','[1 0 0]', ...
7         'MarkerSize',16)
8    % Grafik üzerinde eksen açıklamalarının yapılması
9  - xlabel ('Zaman (s)','fontsize',14)
10 - ylabel('u(t)','fontsize',14)
11 - title('u(t) = 5 sin( 2t ) sinyalinin 0 - 2\pi  zaman dilimindeki ...
12        değişimi','fontsize',14)
```

Example 4-2. A force in newtons (N) is given below. Plot the function.

$$f_1(t) = 0.25t^2$$

- Assume 101-point t vector is in memory.
- >> f1 = 0.25*t.*t; **or**
- >> f1 = 0.25*t.^2:
- >> plot(t, f1)
- >> xlabel('Time, seconds')
- >> ylabel('Force, newtons')
- >> title('Figure 4-4. Force as a function of time in Example 4-2.')
- >> grid

Figure 4-4. Force as a function of time in Example 4-2.



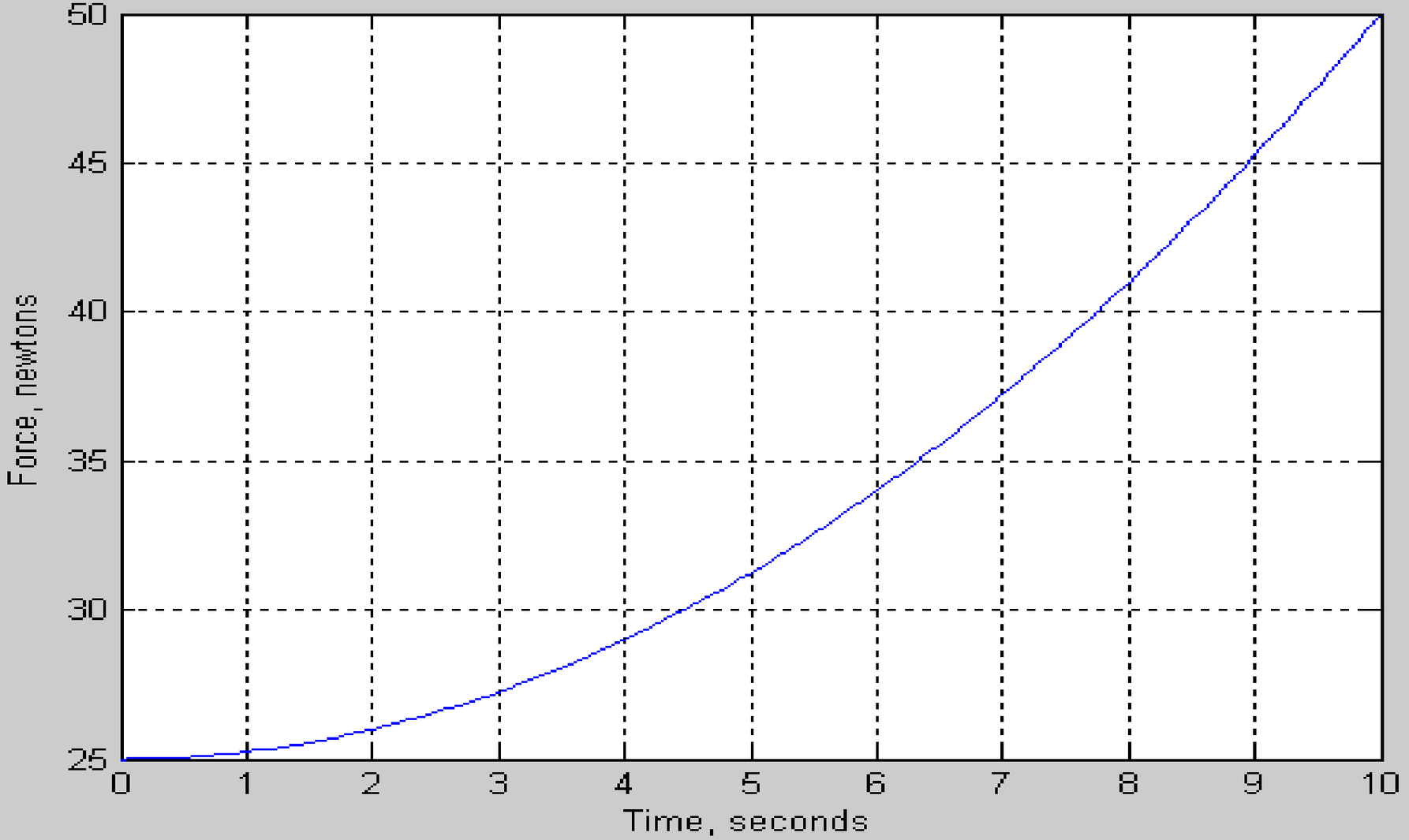
Example 4-3. A force in newtons (N) is given below. Plot the function.

$$f_2(t) = 25 + 0.25t^2$$

Assume 101-point t-vector is in memory.

```
>> f2 = 25+0.25*t.^2;
>> plot(t, f2)
>> xlabel('Time, seconds')
>> ylabel('Force, newtons')
>> title('Figure 4-6. Second force as initially obtained in Example 4-3.')
>> grid
```

Figure 4-5. Second force as initially obtained in Example 4-3.



Example 4-3. Continuation.

- Plot is modified by the command

- >> axis([0 10 0 50])


Figure 4-6. Second force in Example 4-3 after scale change.
Force, newtons
Time, seconds
0
5
10
15
20
25
30
35
40
45
50
0
1
2
3
4
5
6
7
8
9
10

Multiple Plots on Same Graph

- The two functions f_1 and f_2 of the previous two examples can be plotted on the same graph by the command
- >> plot(t, f1, t, f2)
- The command **gtext('label')** allows a label to placed on a graph using crosshairs. The resulting functions are shown on the next slide.

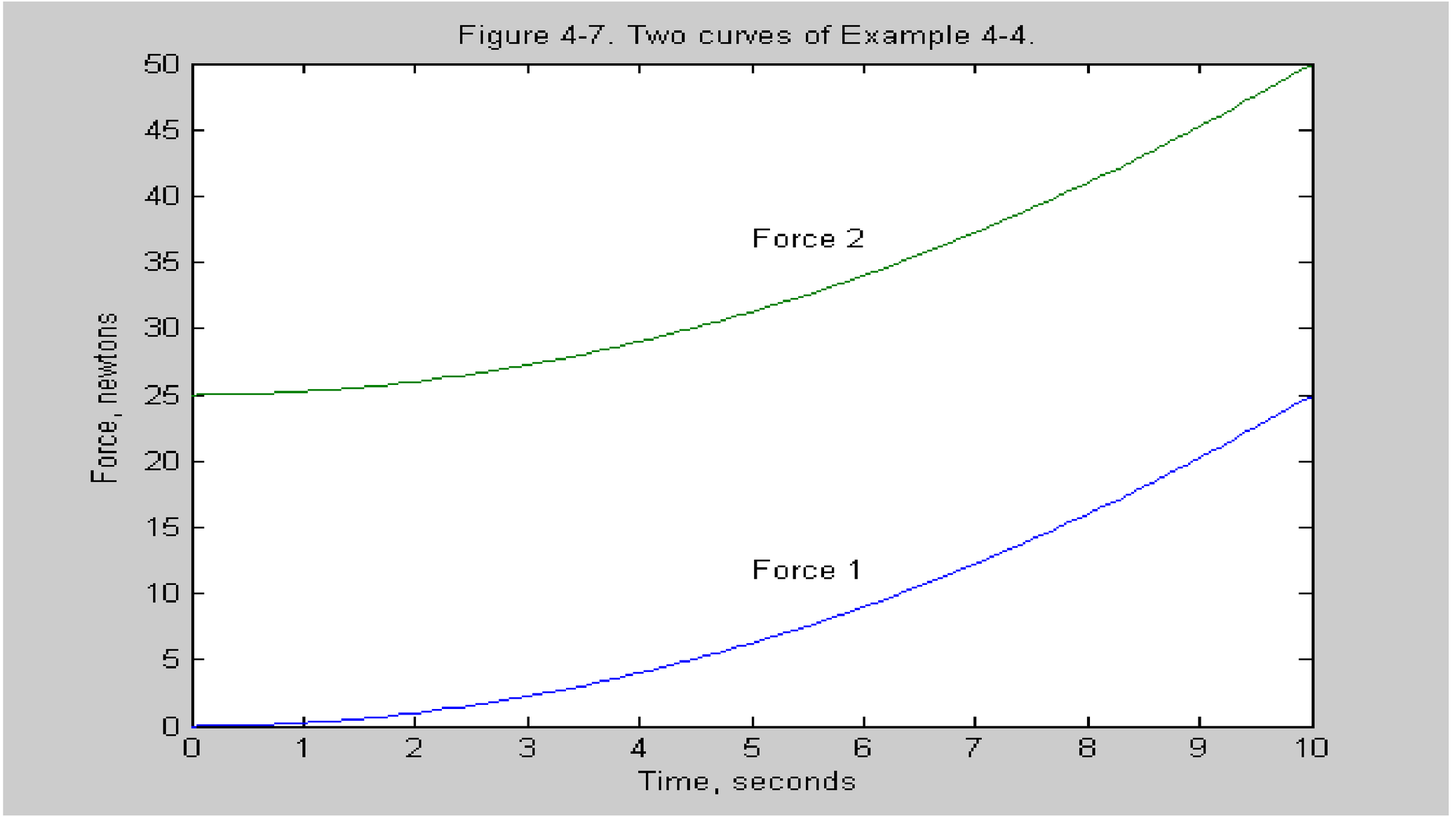
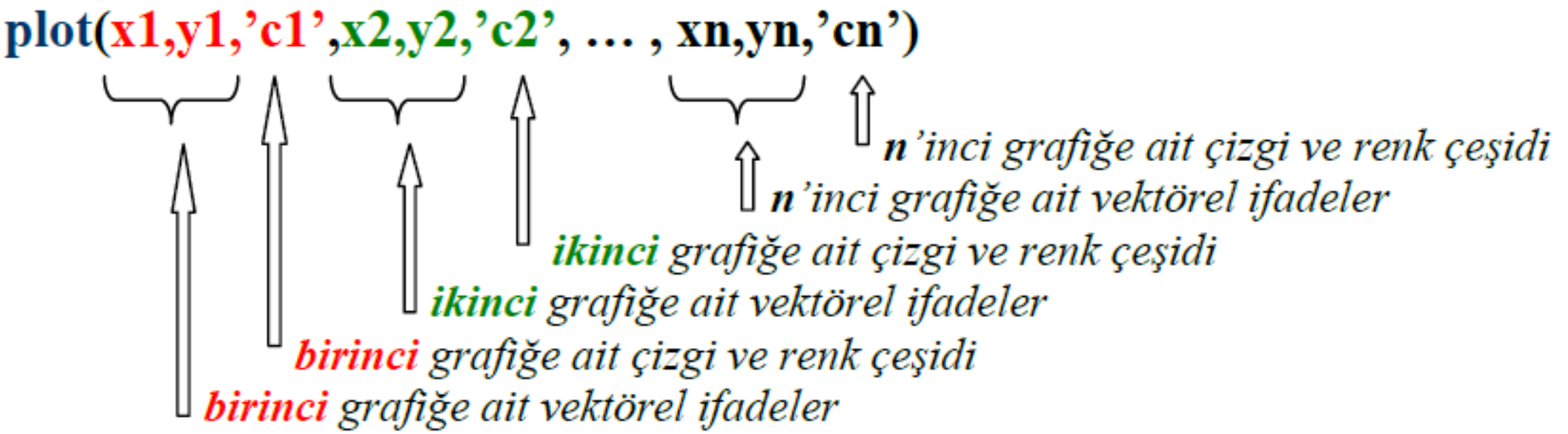


Figure 4-7. Two curves of Example 4-4.
Force, newtons
Time, seconds
Force 2
Force 1
0
5
10
15
20
25
30
35
40
45
50
0
1
2
3
4
5
6
7
8
9
10

plot Komutu ile Grafik Çizimi

tek bir figürde birden fazla grafik çizimi

- **Tek bir figure içerisinde farklı özelliklere sahip birden fazla grafik çizdirilmesi istenirse,**



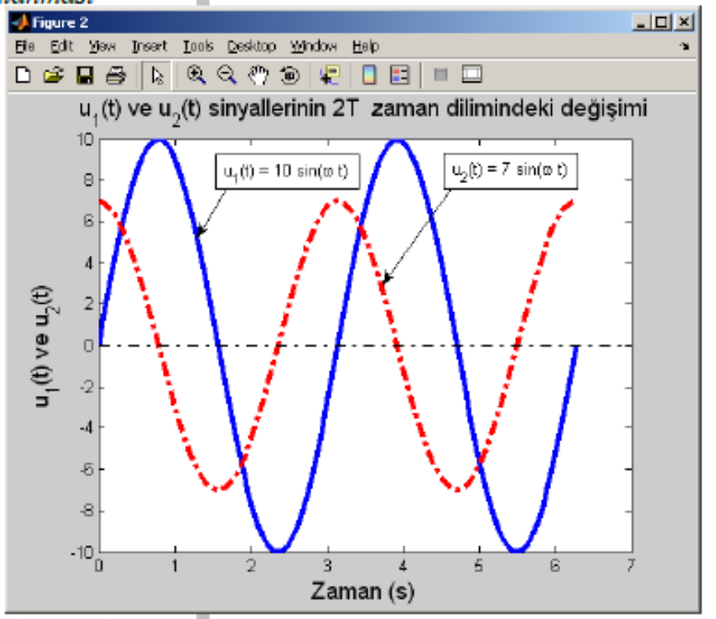
plot Komutu ile Grafik Çizimi

tek bir figürde birden fazla grafik çizimi

- **Örnek: Aşağıda belirtilen işlemleri bir m.file içerisinde yapınız.**
  - **u1(t)= 10sin(ωt) ve u2(t)= 7cos(ωt) iki ayrı sinyali tanımlayınız. ω= 2 rad/sn**
  - **Sinyallerin iki (2) periyotluk değişimlerini tek bir grafik üzerinde karşılaştırınız.**

PROGRAM

```
% Grafik çiziminde kullanılacak u1(t) ve u2(t) sinyallerinin 2*T'ye göre tanımlanması
w = 2;
T = 2*pi/w;
t = linspace(0,2*T);
u1 = 10*sin(w*t);
u2 = 7*cos(w*t);

% Grafik çiziminin tek plot komutu ile gerçekleştirilmesi
plot(t,u1,'-b',t,u2, '-.r', 'linewidth',3)

% Grafik üzerinde eksen açıklamalarının yapılması
xlabel ('Zaman (s)','fontsize',14)
ylabel('u_1(t) ve u_2(t)','fontsize',14)
title('u_1(t) ve u_2(t) sinyallerinin 2T  zaman dilimindeki ...
      değişimi','fontsize',14)
```



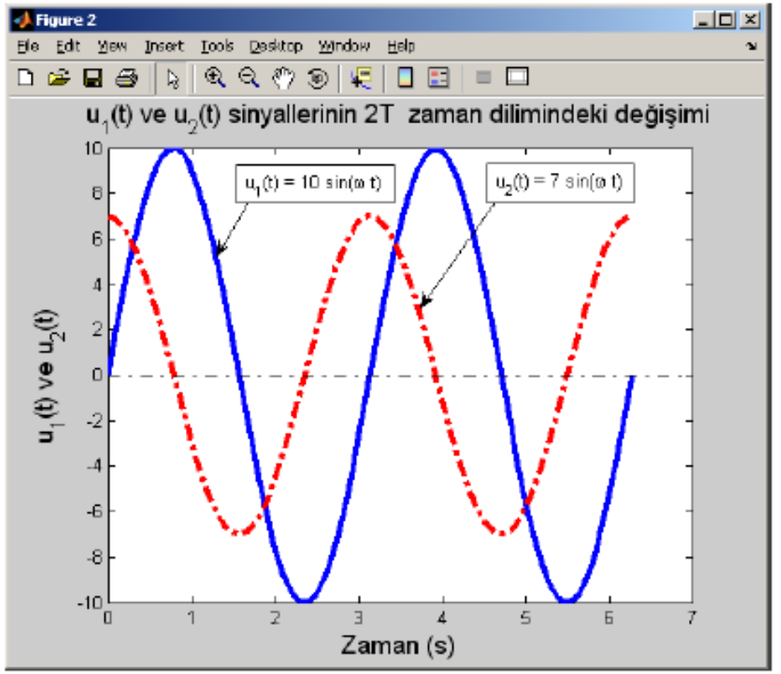
plot Komutu ile Grafik Çizimi

hold on komutu ile tek bir figürde birden fazla grafik çizimi

- Önceki örnekte elde edilen çizimi sıra ile elde ederek tek bir grafikte gösterelim.
- İlk önce u1(t) sinyali çizdirilir.
- hold on komutu çizdirilmiş grafiğin figür penceresinde tutulmasını sağlar.
- hold on komutu kullanıldıktan sonra çizdirilen grafik aynı figüre eklenir.
- hold on komutunu iptal etmek için hold off kullanılır.

PROGRAM

```
% u1(t)'e ait grafiğin çizimi
plot(t,u1,'-b' 'linewidth',3)

% birinci grafiğin figür penceresinde tutulması
hold on

% u2(t)'e ait grafiğin çizimi
plot(t,u2,'-.r' 'linewidth',3)

% elde edilen grafik üzerine eksen çizgisinin eklenmesi
plot([0 7],[0 0],'-.k')
```



Log-Log Plots

$$y = Cx^k$$

$$\log_{10} y = \log_{10}(Cx^k) = \log_{10} C + k \log_{10} x$$

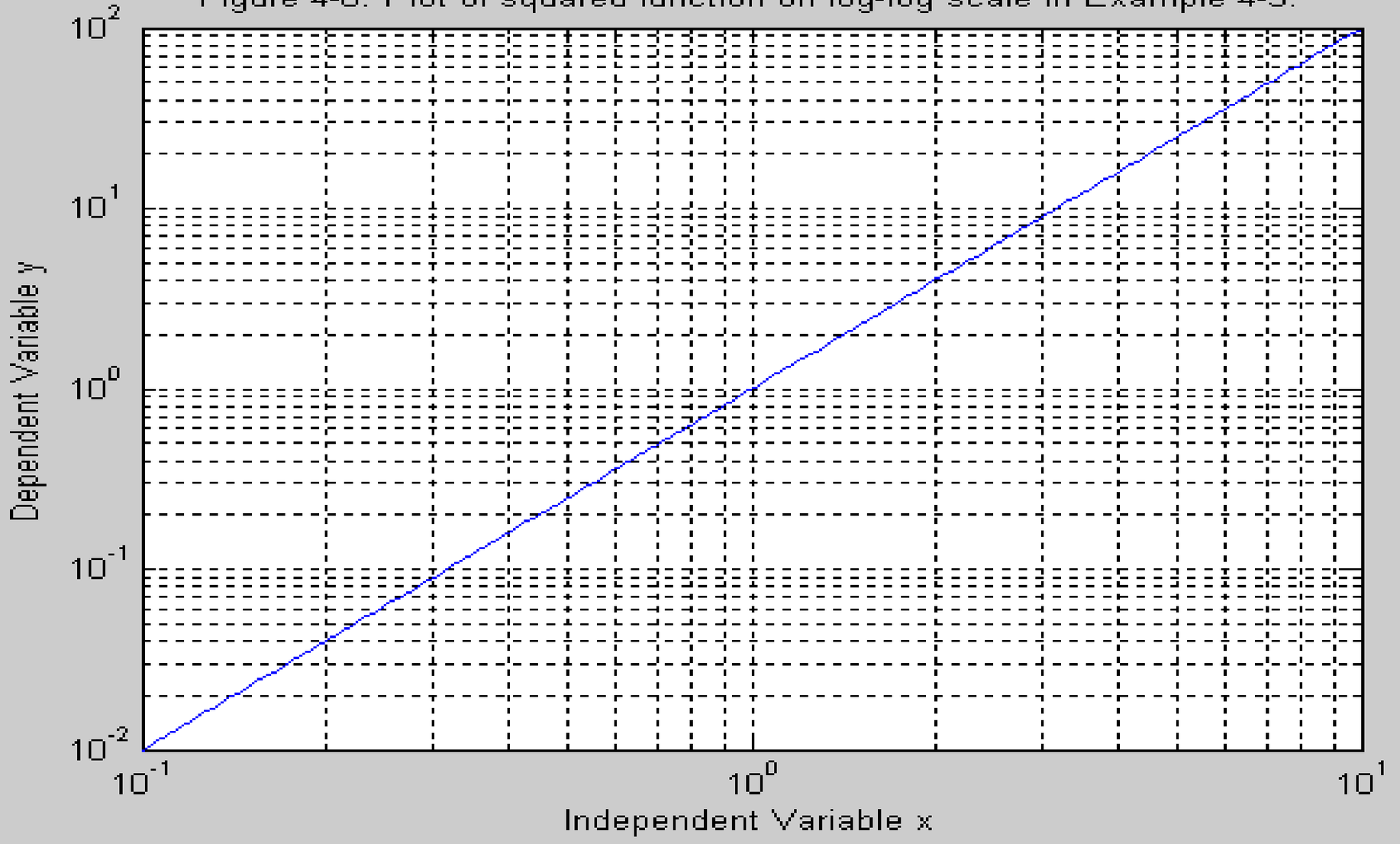
$$y' = mx' + b'$$

Example 4-5. Plot the 2nd degree function below on a log-log graph.

$$y = x^2$$

- >> x = logspace(-1, 1, 100);
- >> y = x.^2;
- >> loglog(x, y)
- A grid and additional labeling were provided and the curve is shown on the next slide.

Figure 4-8. Plot of squared function on log-log scale in Example 4-5.



Bar and Stem Plots

- Command for a bar plot:
- >> bar (x, y)

- Command for a stem plot:

- >> stem (x, y)

Example 4-6. The text contains the sales in *thousands of dollars* for a small business from 1993 through 2002. Construct a bar graph.

- >> year = 1993:2002;
- >> sales = [the 10 values in the text];
- >> bar(year, sales)
- The graph with additional labeling is shown on the next slide.

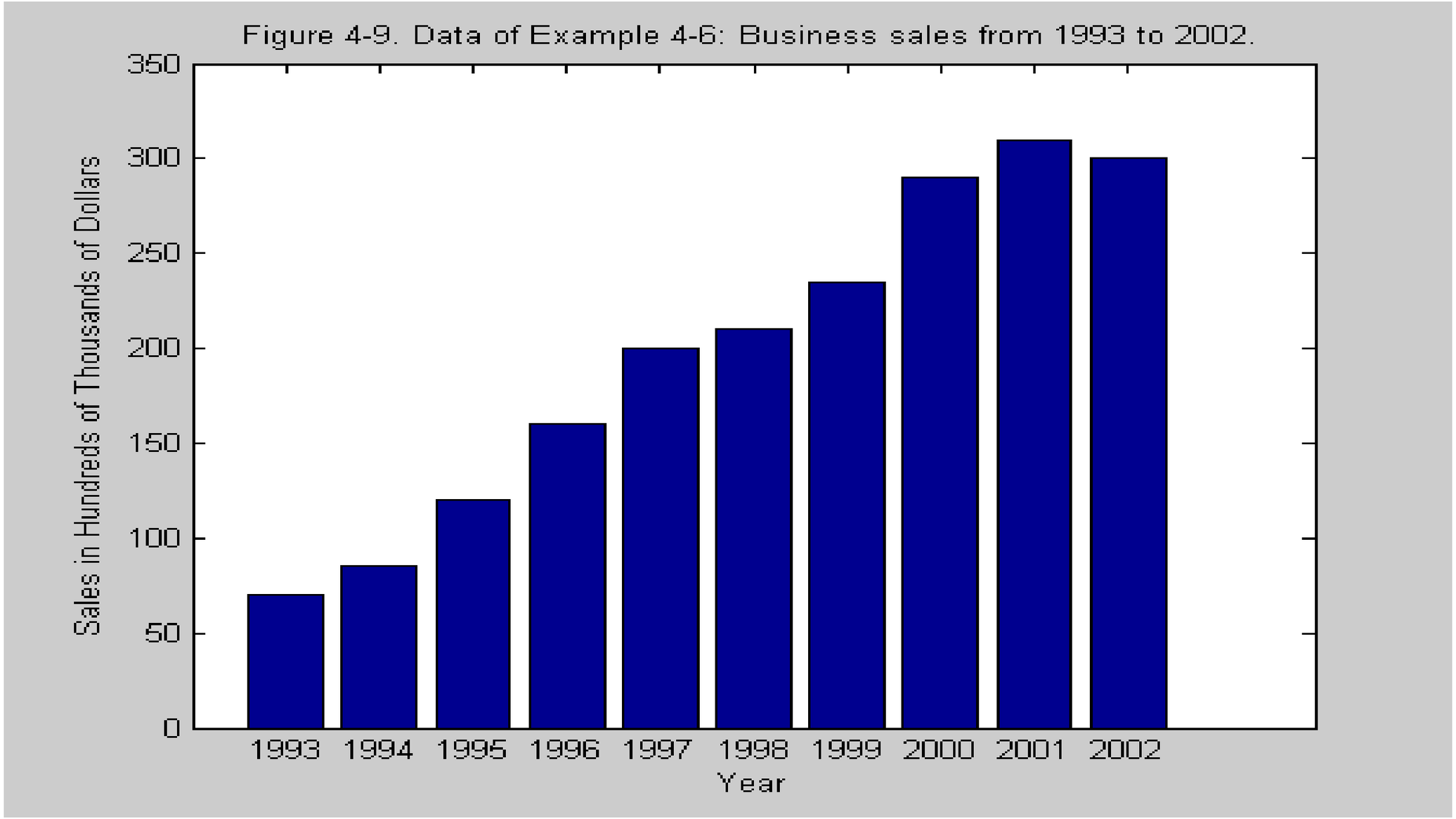


Figure 4-9. Data of Example 4-6: Business sales from 1993 to 2002.
Sales in Hundreds of Thousands of Dollars
350
300
250
200
150
100
50
0
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
Year

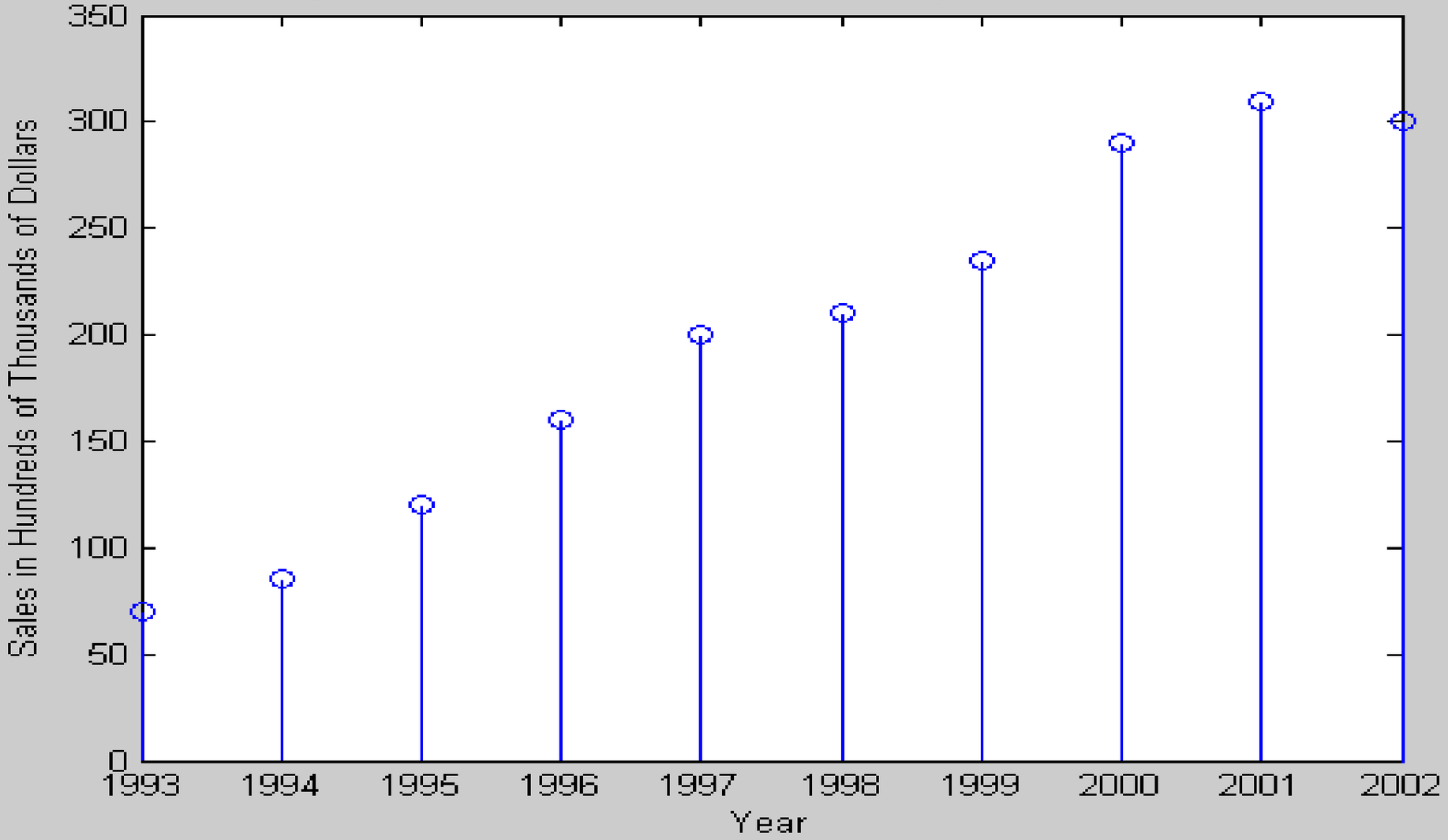
Example 4-7. Plot the data of the previous example using a stem plot.

- Assume that the variables **year** and **sales** are still in memory. The command is

- >> stem (year, sales)

- The plot with additional labeling is shown on the next slide.

Figure 4-10. Data of Example 4-6 displayed on a stem plot.



Chapter 5
Common Functions and their Properties

- The concept of *functions* is a very basic part of mathematics and one that appears in all forms of algebra, trigonometry, and calculus. While there are hundreds of different types of mathematical functions, certain common ones tend to occur quite often in applied engineering and scientific applications. In this chapter, we will explore some of these most common functions and study their behavior.

- Now that basic MATLAB matrix operations and curve plotting have been covered, much of the work that follows will provide coverage of the MATLAB commands immediately after introducing the mathematical forms. This will be the norm where the commands are fairly simple and are best introduced after discussing the mathematical form. In particular, the need to plot curves of functions immediately after introducing the functions will be best achieved with MATLAB commands.

Functions

- A **function** is a relationship between two or more variables. At this point in the text, we will consider only the variables *x* and *y* for a given function. In most cases, we will consider that *x is the independent variable* and *y is the dependent variable.* This does not necessarily mean that *x* causes *y* in all cases, but it suggests that we first assign a value of *x* and then determine the value or values of *y.*

Horizontal and Vertical Axes

- Normally, *x* is assigned to the *horizontal axis* and *y* is assigned to the *vertical axis*. The general notation indicating that *y* is a function of *x* will take the form $y=f(x)$ and letters other than *f* may be used when there are several functions. Alternately, subscripts may be added to different functions to give them separate identities. Sometimes, we will use the simpler notation $y=y(x)$ to mean the same thing.

Single-Valued versus Multi-Valued

- A *single-valued function* is one in which a single value of *x* results in a single value of *y*. *A multi-valued function* is one in which a single value of *x* results in more than one value of *y*. An example of a single-valued function is shown in Figure 5-1(a), and a multi-valued function is shown in Figure 5-1(b). Both appear on the next slide.

Figure 5-1(a). Example of a single-valued function.

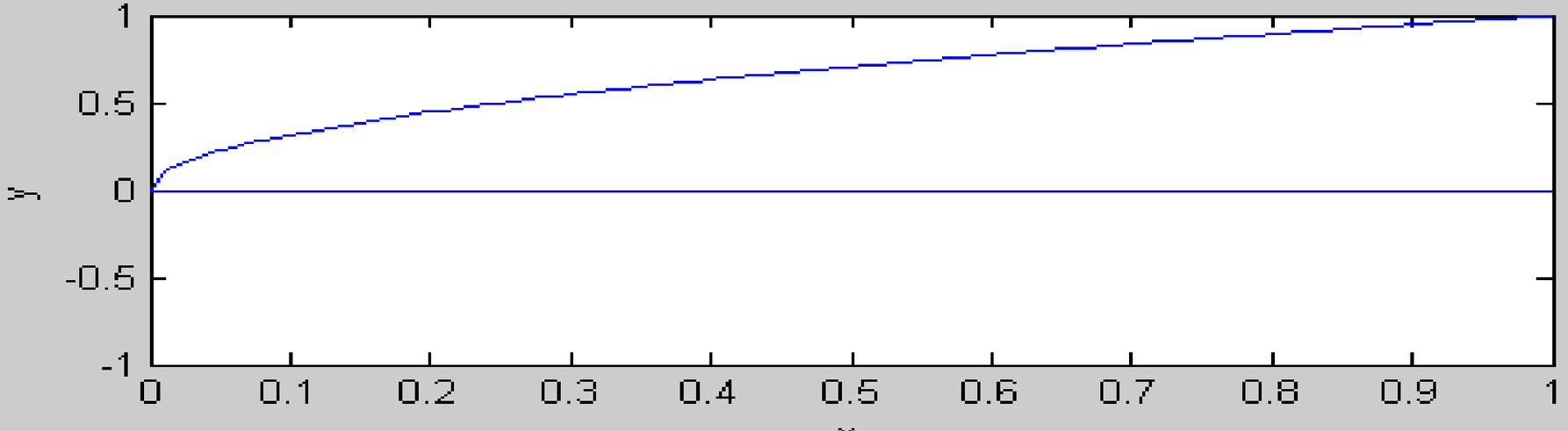
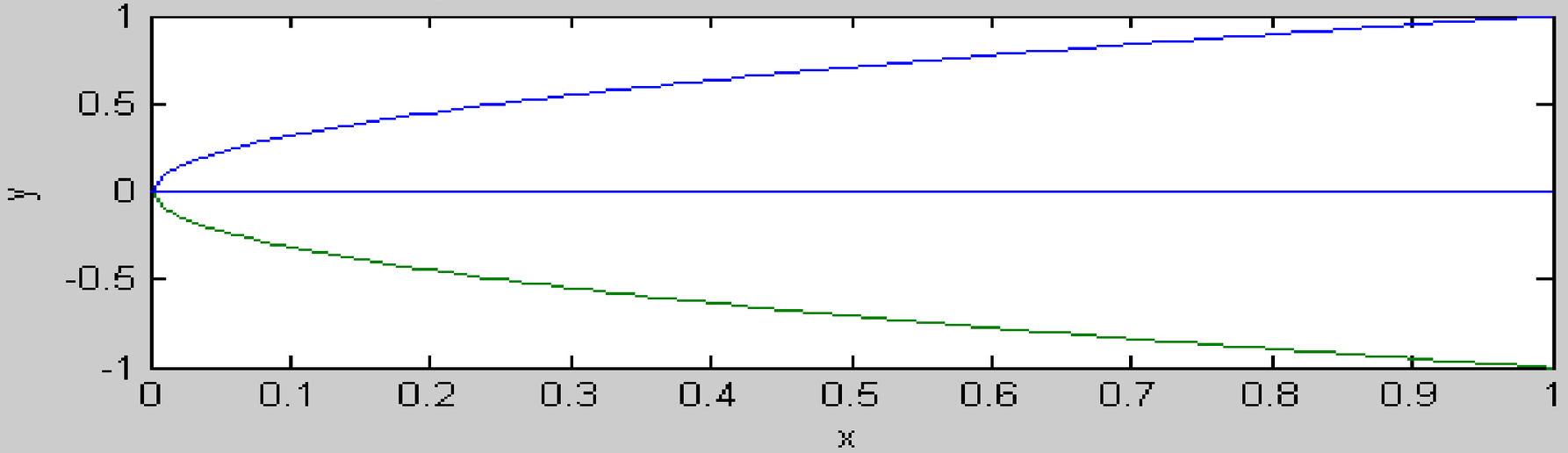


Figure 5-1(b). Example of a multi-valued function.



Continuous versus Discontinuous

The definition of a continuous function is one in which at any value of the independent variable, approaching the value from the left results in the same dependent value as approaching the value from the right. An example of a continuous function is shown in Figure 5-2(a), and a function that has one finite discontinuity is shown in Figure 5-2(b). Both appear on the next slide.

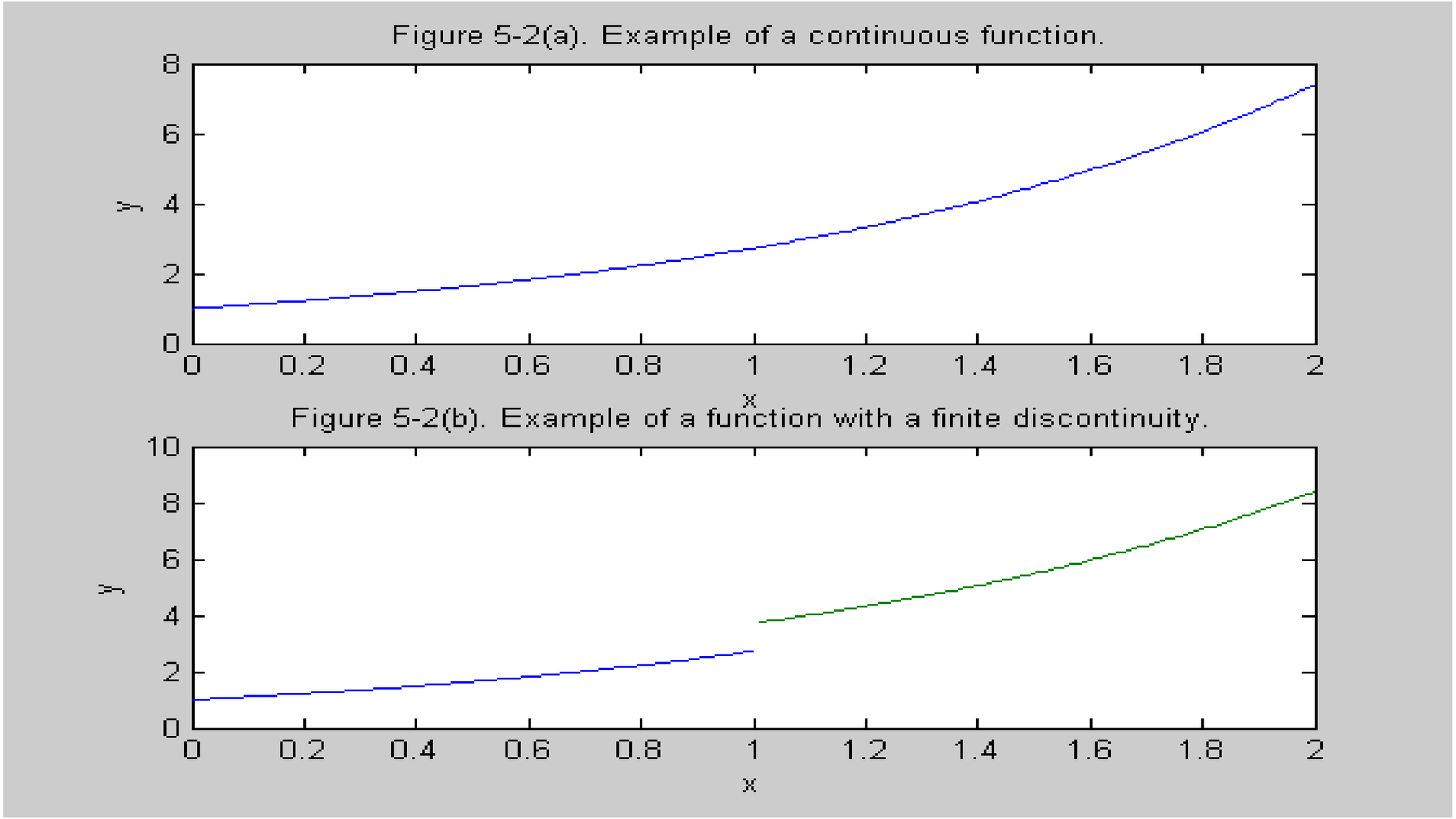


Figure 5-2(a). Example of a continuous function.
y
x
0
2
4
6
8
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Figure 5-2(b). Example of a function with a finite discontinuity.
y
x
0
2
4
6
8
10
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

Domain and Range

- Assume that a function is being evaluated over specific limits such as from x_1 to x_2. This portion of the *x*-axis is called the *domain*. All values of the dependent variable *y* that are produced in the process are called the *range*. In casual usage, engineers and technologists tend to refer to both as *ranges.*

Inverse Functions

- If we have a function $y=f(x)$, and we can reverse the process and solve for x in terms of y, we have the *inverse function*. For the moment, we will denote the inverse simply as $x=g(y)$. We will retain the original variable names and then consider y as the independent variable and x as the dependent variable.

Even and Odd Functions

- An *even function* is one that satisfies
- $f(-x)=f(x)$
- Figures 5-4 and 5-6 are even functions.

- An *odd function* is one that satisfies
- $f(-x)=-f(x)$
- Figures 5-5 and 5-7 are odd functions.

Example 5-1. Determine if the function below is single-valued or multi-valued.

$$y = f(x) = x^2 - 1$$

- With *x* as the independent variable and *y* as the dependent variable, there is only one value of *y* for a given value of *x*. Hence the function is single-valued.

Example 5-2. Is the function of Example 5-1 even, odd, or neither.

$$f(-x) = (-x)^2 - 1 = x^2 - 1 = f(x)$$

- Since $f(-x)=f(x)$, the function is even.

Example 5-3. Determine the inverse of the function of Example 5-1.

$$y = x^2 - 1$$

$$x^2 = y + 1$$

$$x = \pm\sqrt{y+1}$$

- We now consider *y* as the independent variable and *x* as the dependent variable.

Example 5-4. Is the inverse function of Example 5-3 single-valued or multi-valued?

- Since two values of *x* result from a given value of *y*, the inverse function is multi-valued. This tells us that a function may be single-valued but its inverse may be multi-valued or vice-versa. In many applications, only the positive square root would be of interest, so if the negative square root is rejected, we could interpret the result as being single-valued.

Example 5-5. Is the inverse function of Example 5-3 even, odd, or neither?

$$g(-y) = \pm\sqrt{-y+1}$$

- The inverse function is neither even nor odd.

MATLAB Subplot

- The **subplot** allows more than one plot to be prepared on the same printer page. In fact, Figures 5-1 and 5-2 were both prepared using that command.The syntax for the subplot command is as follows:
- >> subplot(m, n, k)
- Integers m and n define the number of rows and columns of subplots. The integer k defines the particular one based on left to right and top to bottom.

Example 5-6. Plot the function of Example 5-1 and the inverse of Example 5-3 using subplots.

- >> x = linspace(-2, 2, 201);
- >> y = x.^2-1;
- >> subplot(2, 1, 1)
- >> plot(x, y)
- Additional labeling commands were used.
- >> subplot(2, 1, 2)
- >>plot(y, x)
- Additional labeling commands were used.

Figure 5-3(a). Curve of Examples 5-1 through 5-5 with y versus x.

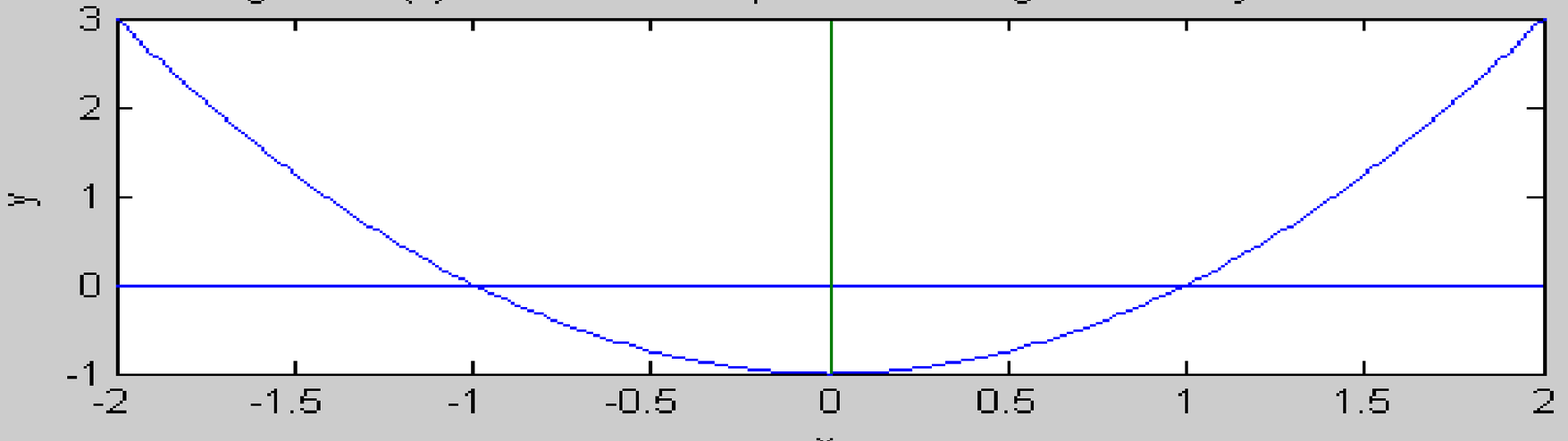
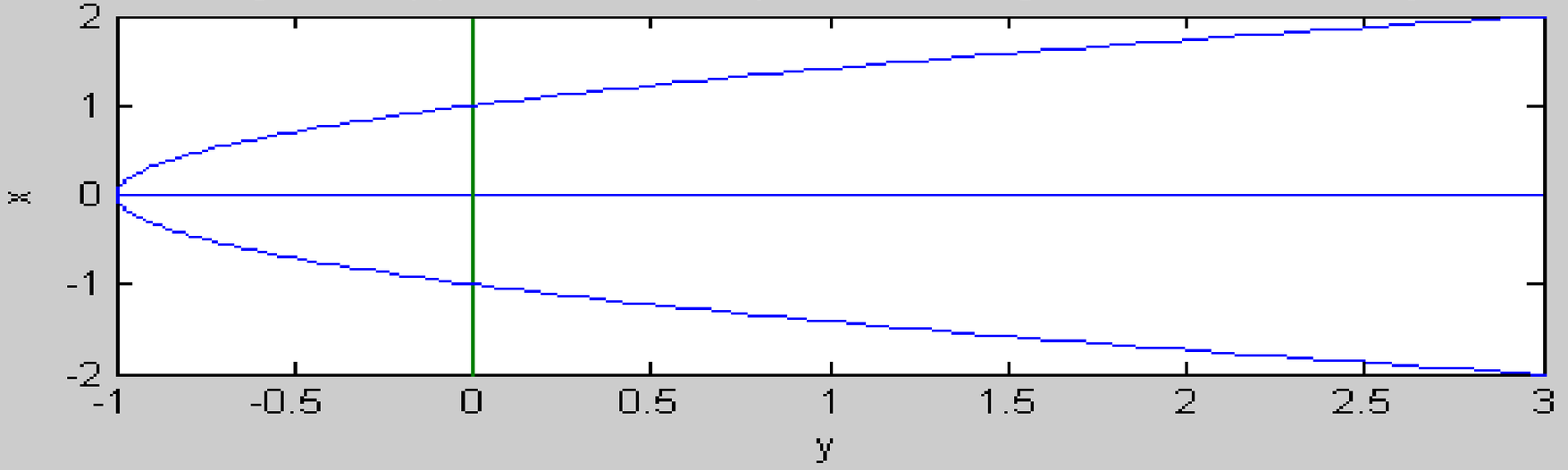


Figure 5-3(b). Curve of Examples 5-1 through 5-5 with x versus y.



Power and Polynomial Functions

$$y_n = x^n$$

$$y_0 = x^0 = 1$$

$$y_1 = x^1 = x$$

$$y_2 = x^2$$

$$y_3 = x^3$$

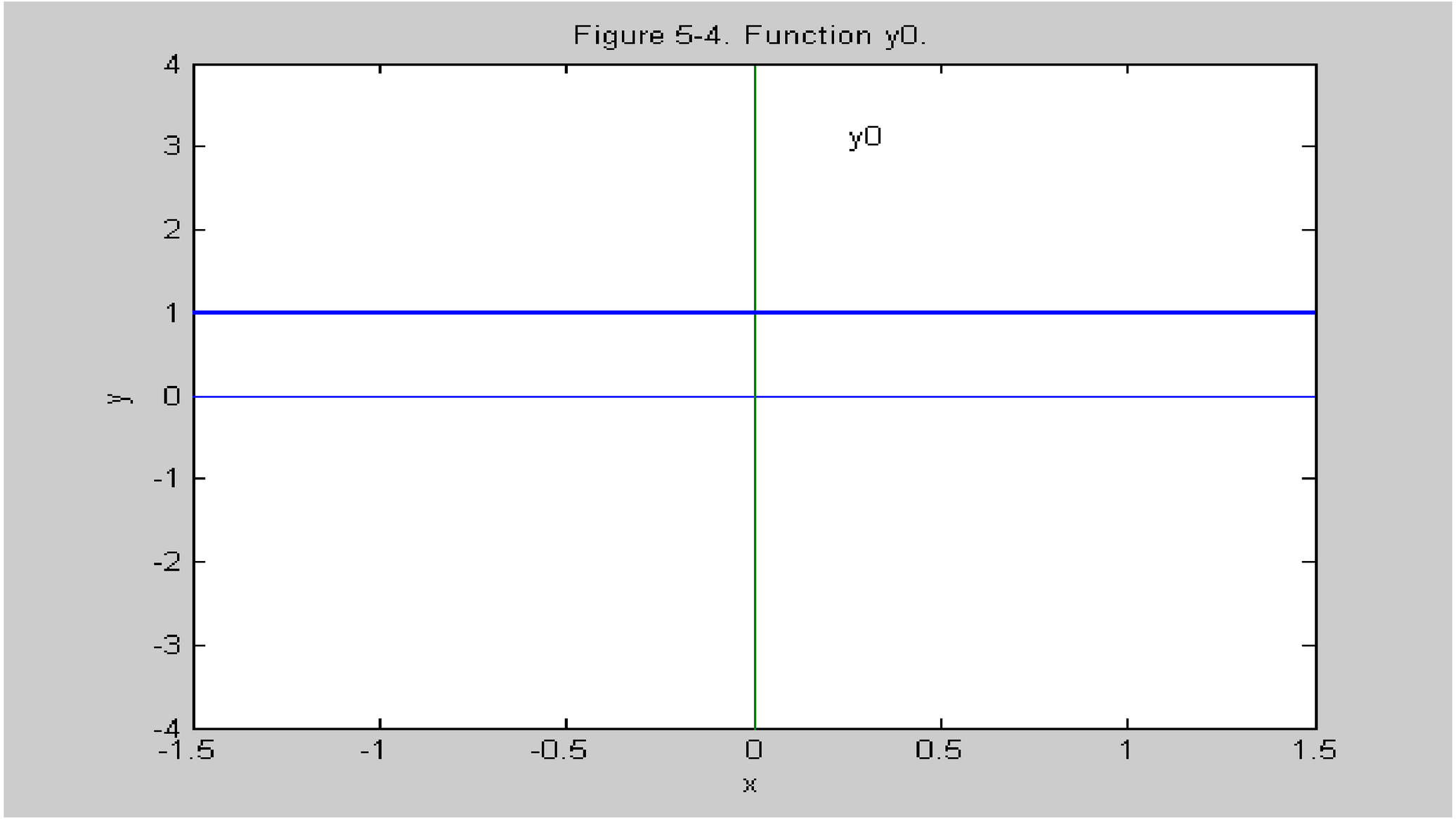


Figure 5-4. Function y0.
y0
4
3
2
1
0
-1
-2
-3
-4
y
-1.5
-1
-0.5
0
0.5
1
1.5
x

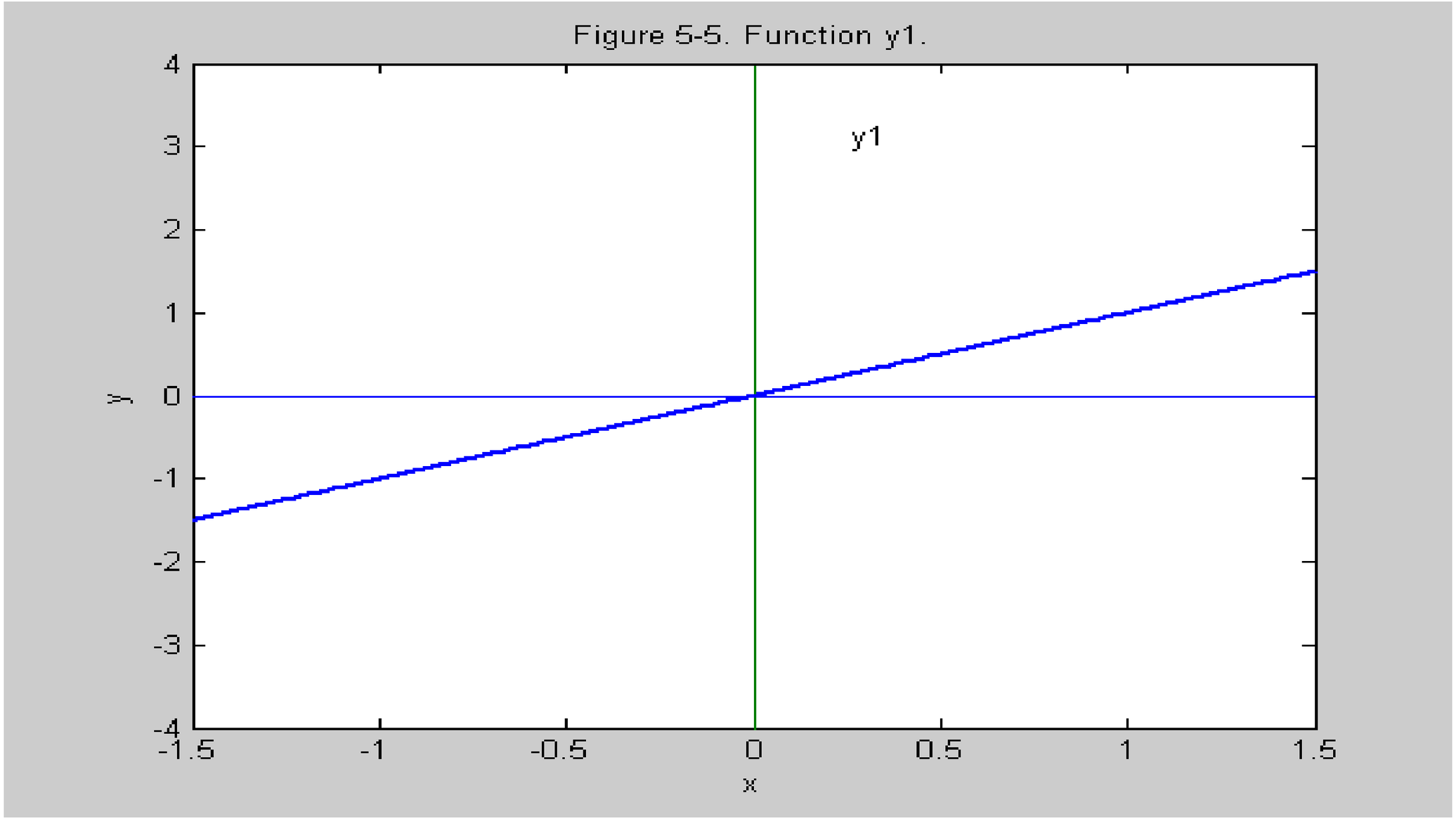


Figure 5-5. Function y1.
y1
4
3
2
1
0
-1
-2
-3
-4
y
-1.5
-1
-0.5
0
0.5
1
1.5
x

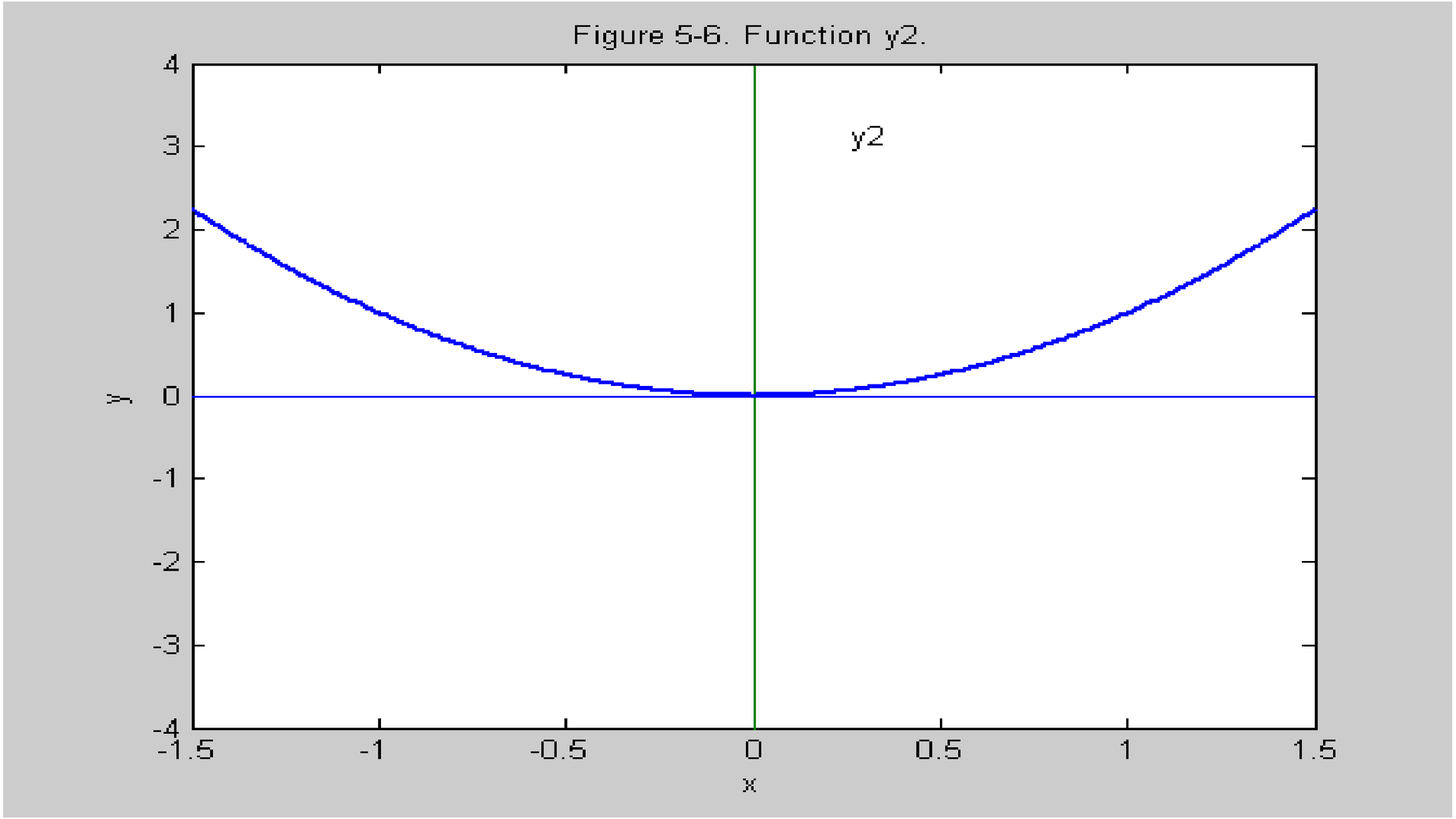


Figure 5-6. Function y2.
y2
4
3
2
1
0
-1
-2
-3
-4
y
-1.5
-1
-0.5
0
0.5
1
1.5
x

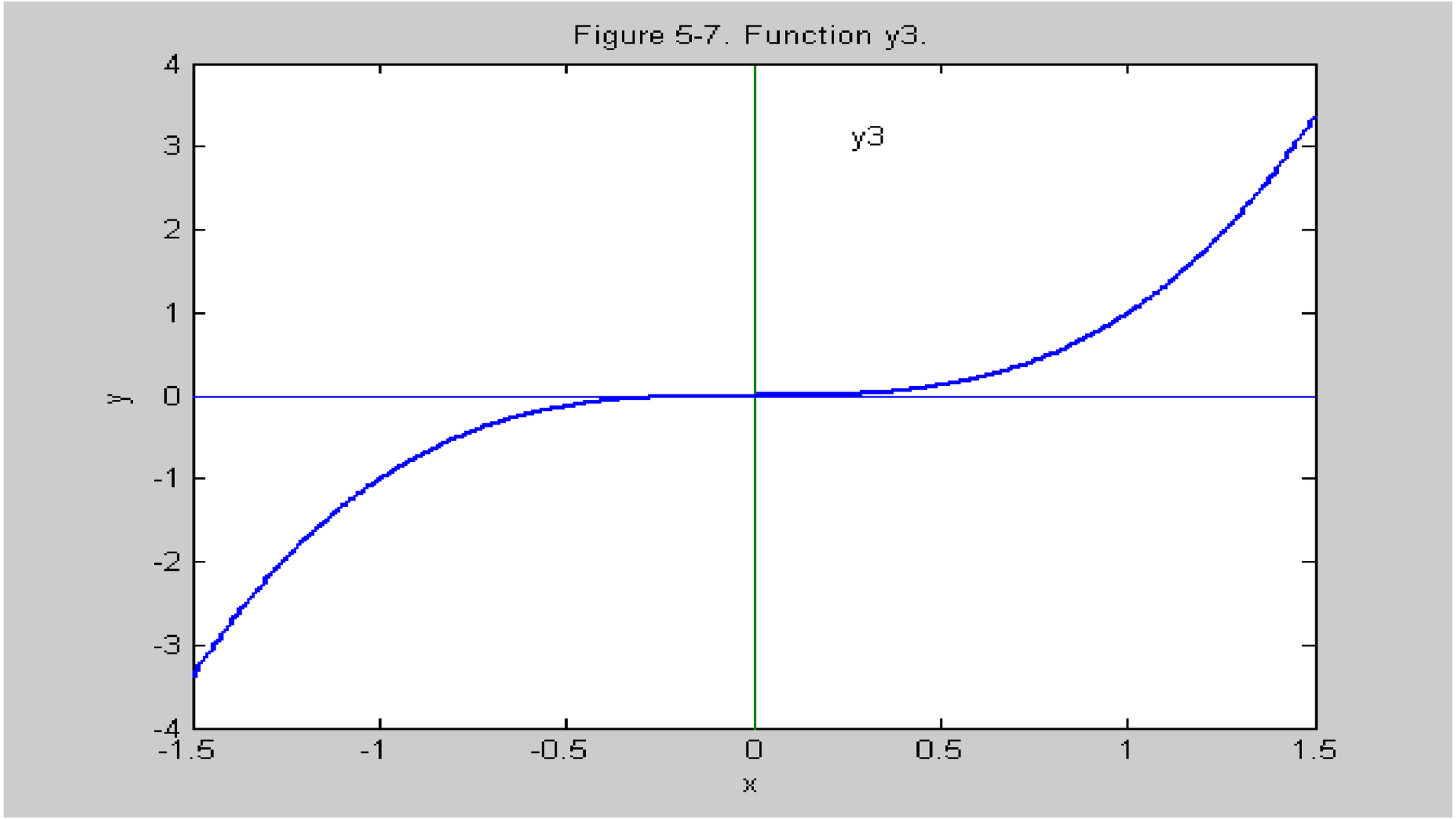


Figure 5-7. Function y3.
y3
4
3
2
1
0
-1
-2
-3
-4
y
-1.5
-1
-0.5
0
0.5
1
1.5
x

Straight-Line Equation

$$y = mx + b$$

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

- The quantity *m* is the *slope* of the line and *b* is the *vertical intercept*. For $m>0$, the slope is upward and for $m<0$, the slope is downward. The line crosses the vertical axis at a value *b*.

Example 5-7. Write the equation and plot the line having a slope of 2 and a vertical intercept of -4.

$$y = 2x - 4$$

- This case is about as simple as any can be since we are given the two parameters required in the slope/vertical intercept form. The straight-line is shown on the next slide.

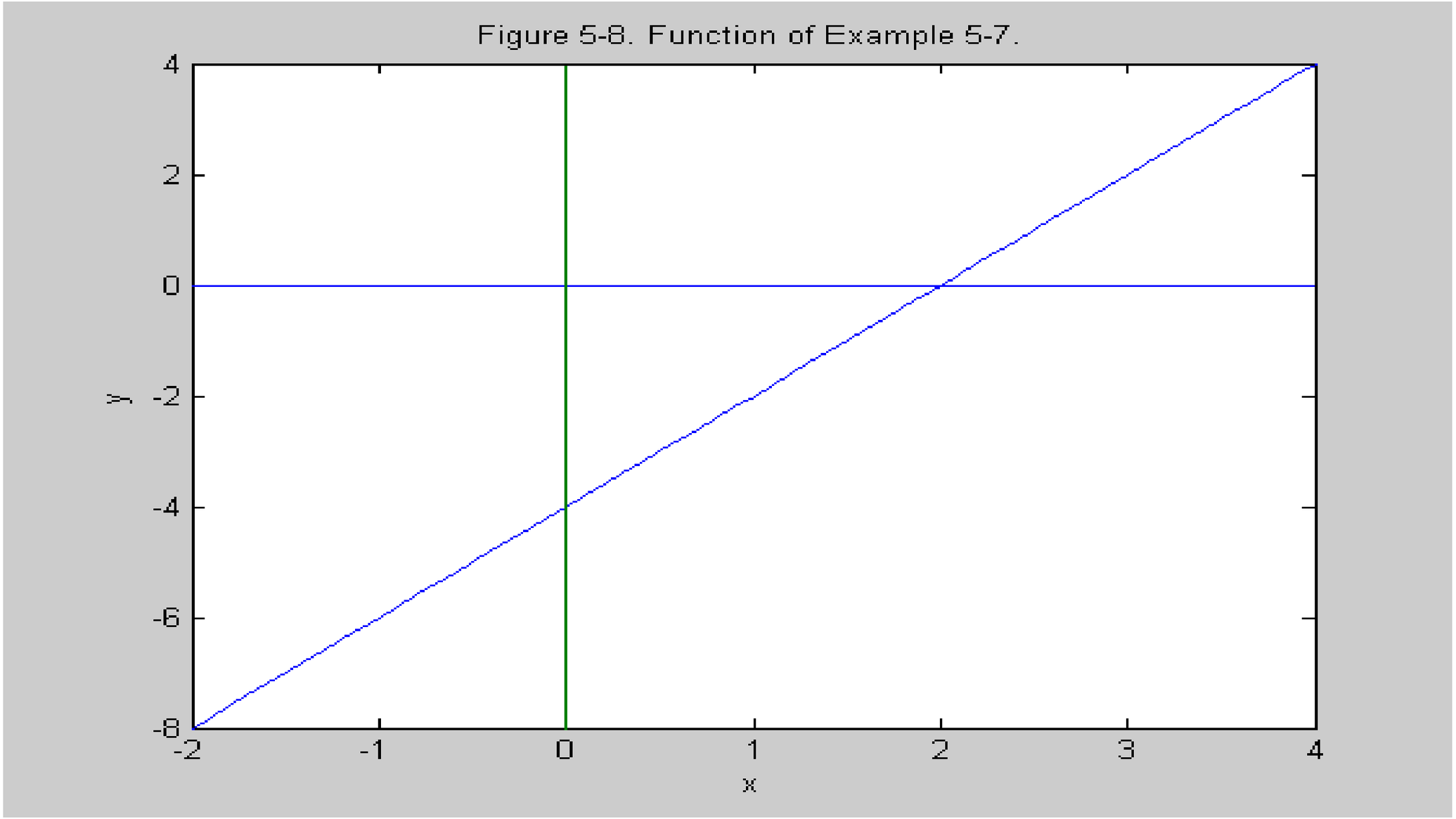


Figure 5-8. Function of Example 5-7.
4
2
0
-2
-4
-6
-8
y
-2
-1
0
1
2
3
4
x

Example 5-8. Write the equation and plot the line passing through the points (3, 5) and (6, -7).

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{-7-5}{6-3} = \frac{-12}{3} = -4$$

$$y = -4x + b \quad 5 = -4(3) + b \text{ or } b = 17$$

$$y = -4x + 17$$

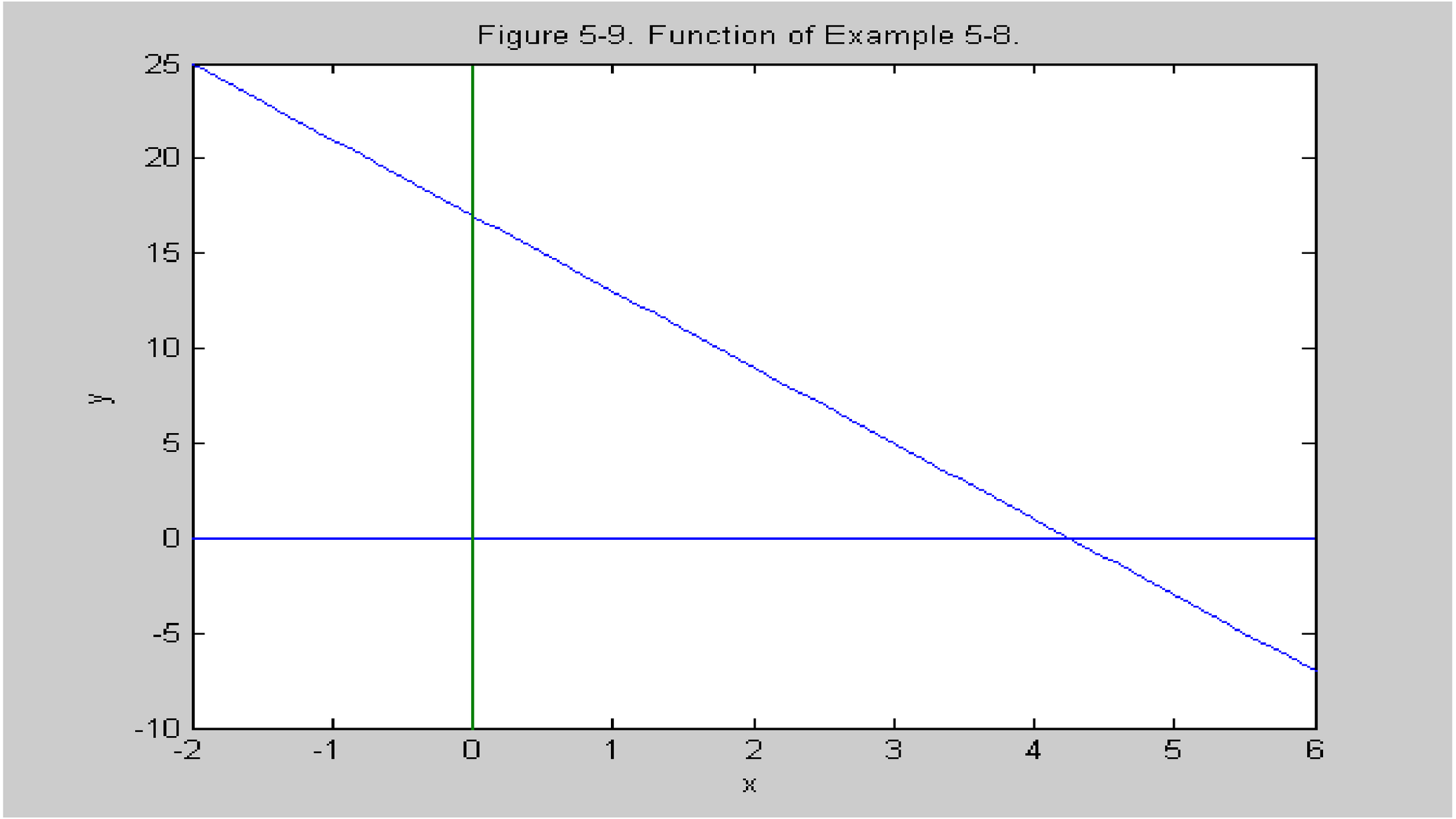


Figure 5-9. Function of Example 5-8.
25
20
15
10
5
0
-5
-10
y
-2
-1
0
1
2
3
4
5
6
x

Polynomial Functions

- A *polynomial function* is one composed of a sum of power terms of the form of x^n with integer values of n and constant factors. A typical polynomial function of degree N can be expressed in the following form:

$$y = p(x)$$
$$= A_N x^N + A_{N-1} x^{N-1} + \ldots + A_1 x + A_0$$

Roots of a Polynomial Function

- A *root* of a polynomial equation is a value of *x* such that the polynomial is zero when it is evaluated for that particular value of *x*. This means that for any root x_k of the polynomial $p(x)$ on the previous slide, the following equation is satisfied:

$$p(x_k) = 0$$

Theorem on Roots

- A polynomial of degree *N* has exactly *N* roots. These roots may be classified as
- 1. Real roots of first order
- 2. Complex roots of first order
- 3. Real roots of multiple order
- 4. Complex roots of multiple order
- In this classification scheme, purely imaginary roots may be considered as a special case of complex roots.

Complex Roots

- For real polynomial coefficients, any complex roots appear in conjugate pairs. Thus, if *a*+*ib* is a root, *a*-*ib* will also be a root. The value *a*-*ib* is the *complex conjugate* of *a*+*ib.* The quantity *i* is the basis for the complex number system and is given by

$$i = \sqrt{-1}$$

Factored Form of a Polynomial

$$y = p(x)$$
$$= A_N(x - x_1)(x - x_2)....(x - x_N)$$

MATLAB Evaluation of Polynomial

$$y = A_N x^N + A_{N-1} x^{N-1} + ... + A_1 x + A_0$$

- Assume that the vector x has been entered. To illustrate for the third degree case, one way to evaluate is shown below.

- >> y = A3*x.^3 + A2*x.^2 + A1*x + A0

- An easier way will be shown on the next slide.

Easier MATLAB Procedure for Polynomial Evaluation

- Define a row vector C as follows:
- >> C = [A3 A2 A1 A0];

- The polynomial will be evaluated at all values of x by the command

- >> y = polyval(C, x)

Factoring of Polynomials

- Define a row vector C as follows:
- >> C = [A3 A2 A1 A0];

- The roots will be determined by the command
- >> R =roots(C)
- The vector R as is a column vector whose values are the roots of the polynomial.

Forming the Polynomial from the Roots

- Assume that the roots of a polynomial are formed as either a row or a column vector and denoted as R. The coefficient matrix C of the polynomial is determined by
- >> C = poly(R)
- If the coefficient of the highest degree term is other than one, it is necessary to modify C as follows:
- >> C = AN*C

Multiplication of Polynomials

•Two polynomials can be multiplied together by the use of the **conv** command. The term **conv** is a contraction of the term *convolution* which has applications in signal processing and in both differential and difference equations. To illustrate, assume two 2nd degree polynomials.

$$p_1(x) = A_2x^2 + A_1x + A_0$$

$$p_2(x) = B_2x^2 + B_1x + B_0$$

$$p_3(x) = p_1(x)\,p_2(x)$$

Multiplication of Polynomials Continuation

- Form row vectors for the coefficients.
- >> C1 = [A2 A1 A0];
- >> C2 = [B2 B1 B0];

- The coefficient matrix of the product polynomial is obtained by the command that follows.
- C3 = conv(C1, C2)

Example 5-9. Use MATLAB to determine the roots of

$$y = 3x^2 + 12x + 39$$

- >> C = [3 12 39];
- >> R = roots(C)
- R =
- -2.0000 + 3.0000i
- -2.0000 - 3.0000i

Example 5-10. Reconstruct the coefficients of the polynomial from the roots of the preceding example.

- >> C1 = 3*poly(R)
- C1 =
- 3 12 39

- We could use the **polyval** command to evaluate the polynomial if desired.

Example 5-11. Determine the roots of the 5th degree polynomial below.

$$y = x^5 + 3.2361x^4 + 5.2361x^3 + 5.2361x^2 + 3.2361x + 1$$

- >> C=[1 3.2361 5.2361 5.2361 3.2361 1];

```
>> R = roots(C)
R =
  -0.3090 + 0.9511i
  -0.3090 - 0.9511i
  -1.0000
  -0.8090 + 0.5877i
  -0.8090 - 0.5877i
```

Example 5-12. Reconstruct the polynomial of Example 5-11 from the roots.

- Assume that the 5 roots are still in memory as a vector.

- >> C1 = poly(R)
- C1 =
- 1.0000 3.2361 5.2361 5.2361 3.2361 1.0000

Example 5-13. Evaluate the 5th degree polynomial for x = 0, 0.5, 1, and 2.

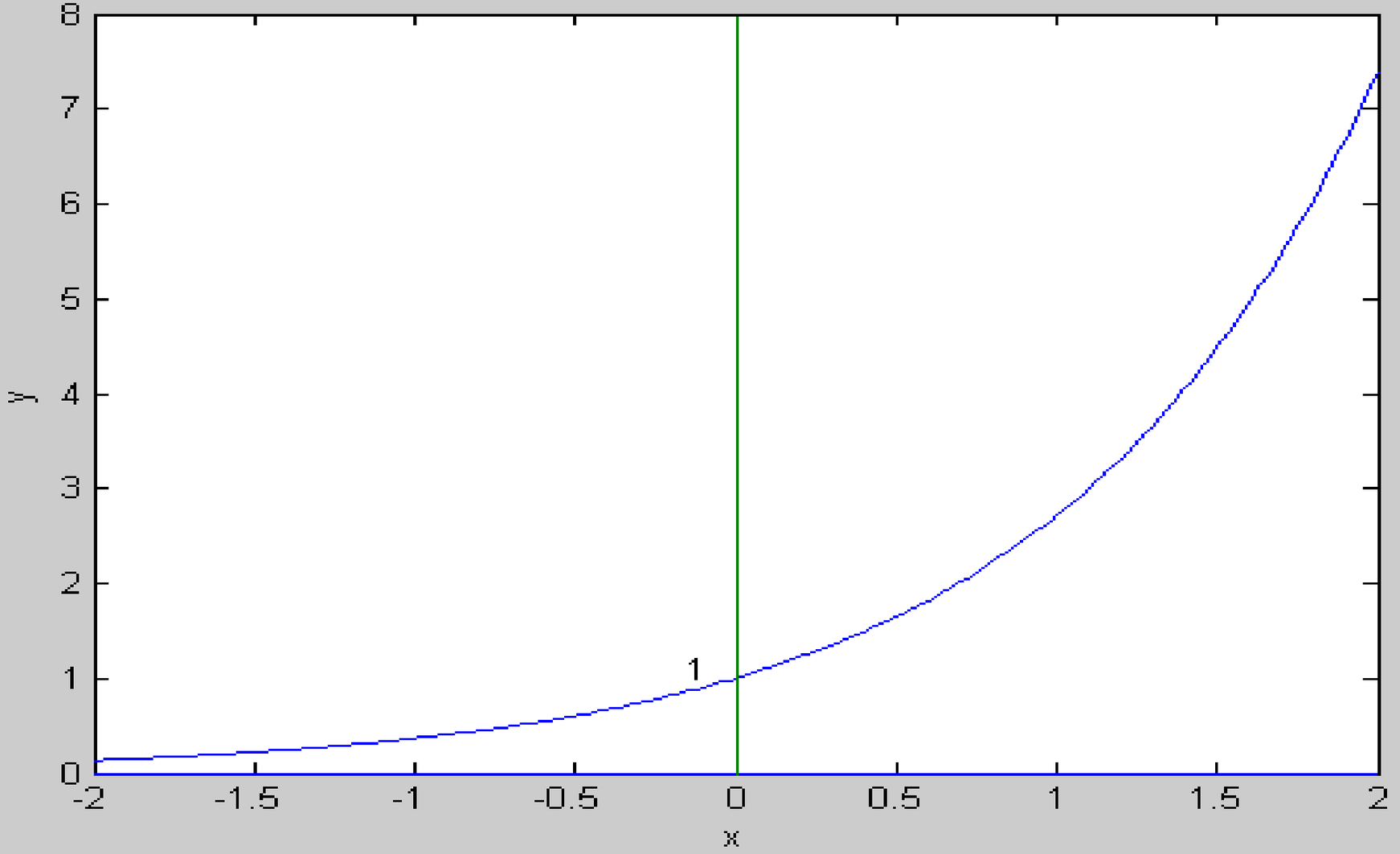
- Assume that C is still in memory.
- >> x = [0 0.5 1 2];
- >> y = polyval(C, x)
- y =
- 1.0000 4.8151 18.9444 154.0830

Exponential Function to the Base e

$$y = e^x$$

- The basic exponential function arises in a large number of scientific and engineering problems. The "purest" form of the exponential is as a power of the mathematical constant *e*=2.718 to four significant digits. The form of the function for both positive and negative *x* is shown on the next slide.

Figure 5-10. General form of the exponential function.



Decaying Exponential Function

•The most common form of the exponential function in practical engineering problems is the *decaying* or *damped exponential function*. Many applications involve time as the independent variable and the forms are shown below and on the next slide.

$$y = e^{-t/\tau}$$

$$y = e^{-\alpha t}$$

$$\alpha = 1/\tau$$

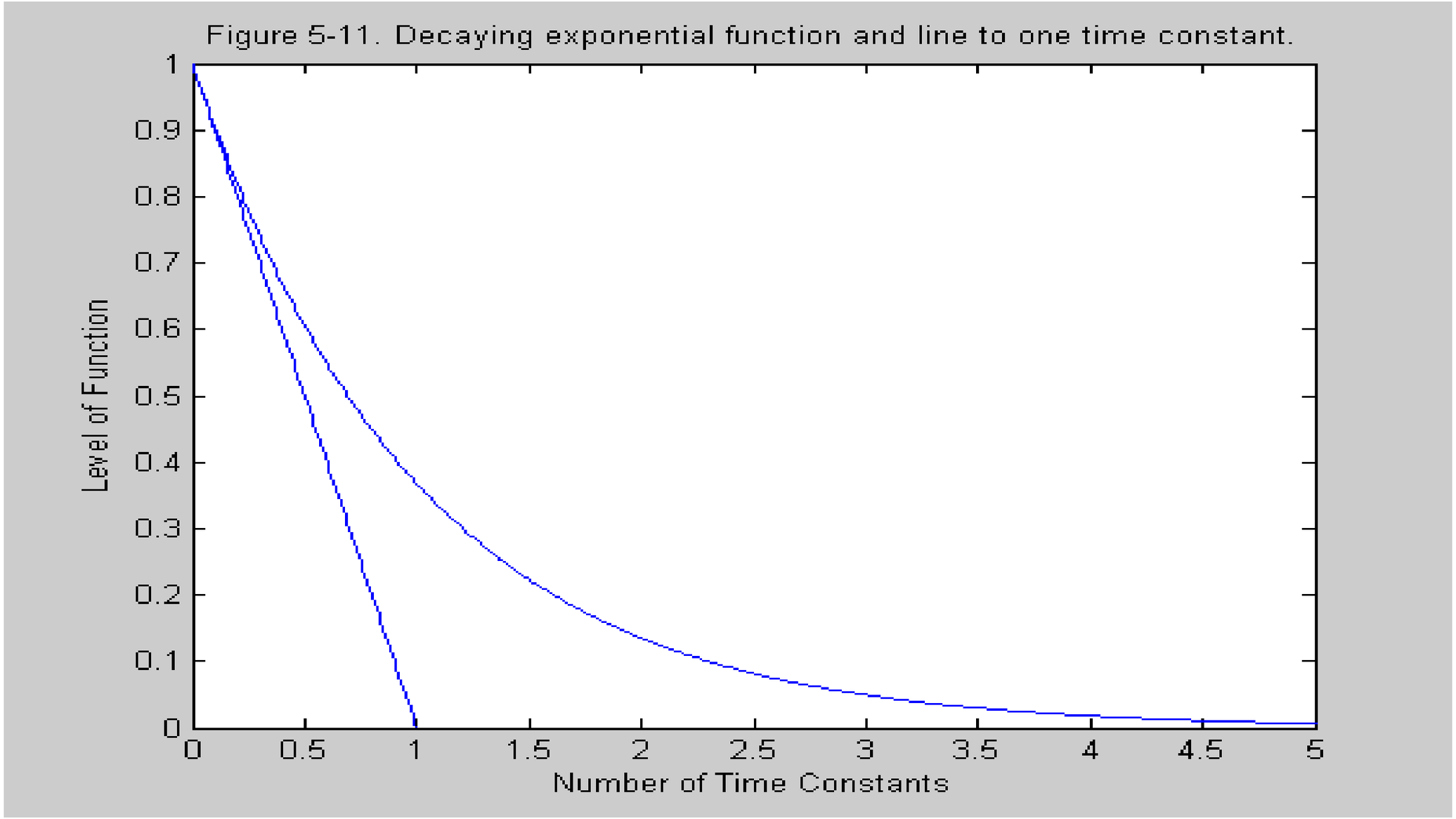


Figure 5-11. Decaying exponential function and line to one time constant.
Level of Function
Number of Time Constants
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

MATLAB Exponential Forms

- Assume that a vector x is in memory. MATLAB uses **exp** for *e* and the command to generate y is
- >> y = exp(x)
- If a base other than *e* is desired, the exponentiation operation is used. For example, if the base is 10, the command is
- >> y = 10.^x

Example 5-14. Consider the exponential function shown below.

$$y = e^{-t/0.01} = e^{-100t}$$

- Determine (a) the time constant and (b) the damping constant. (c). Based on the rule-of-thumb provided in the text, about how long would it take to reach a practical level of zero?

Example 5-14. Continuation.

$$\tau = 0.01 \text{ s} = 10 \text{ ms}$$

$$\alpha = \frac{1}{\tau} = \frac{1}{0.01} = 100 \text{ s}^{-1}$$

$$T = 5\tau = 5 \times 10 \text{ ms} = 50 \text{ ms}$$

Example 5-15. A force *f* begins with 20 N and decays exponentially with a time constant of 5 s. Write the equation.

$$f = 20e^{-t/5} = 20e^{-0.2t}$$

Example 5-16. Generate the two curves of Figure 5-11 and plot them.

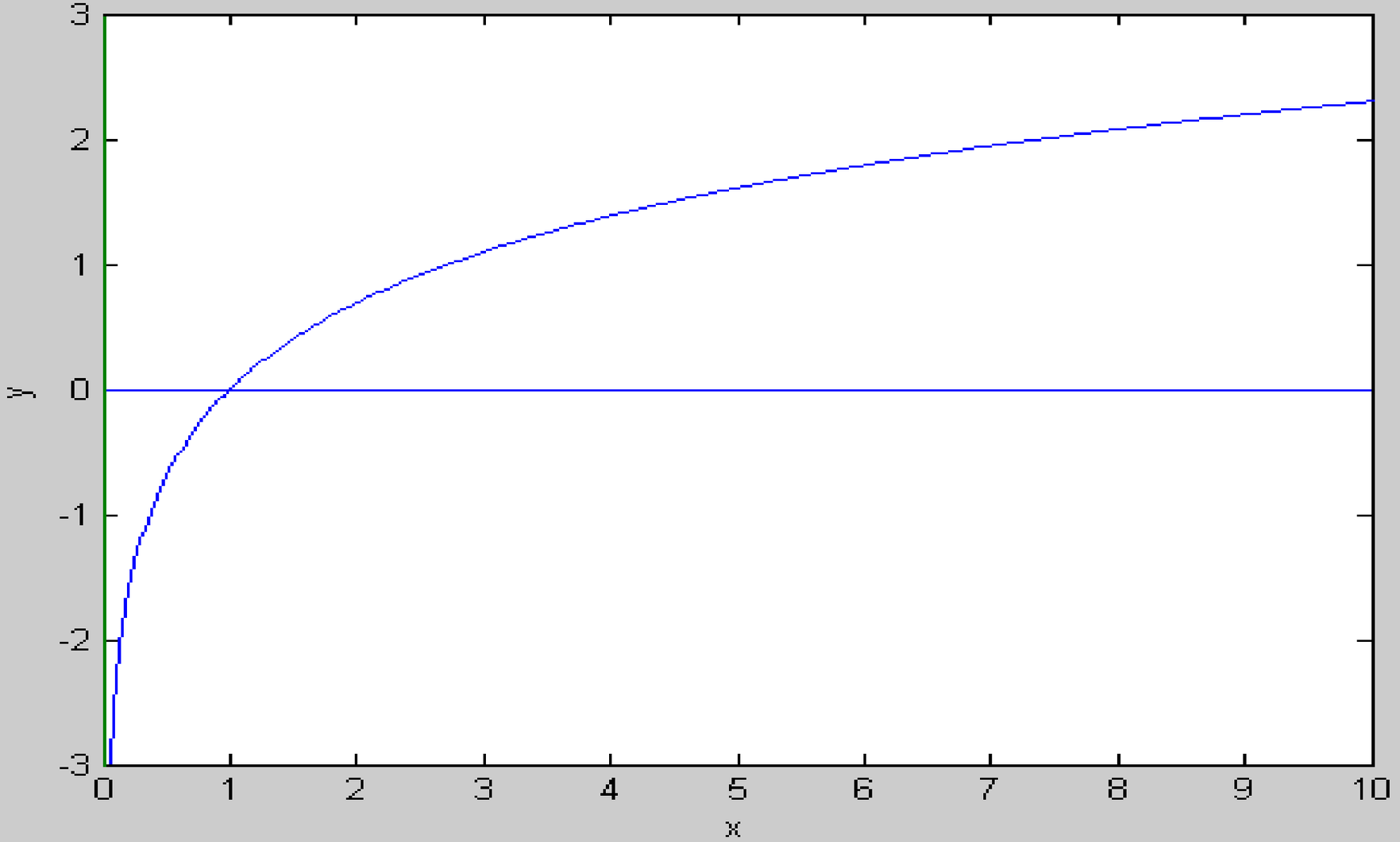
- One is the exponential function and the other is the straight-line y1 = 1 - x.
- >> x = linspace(0, 5, 501);
- >> y = exp(-x);
- >> x1 = linspace(0, 1, 11);
- >> y1 = 1-x1;
- >> plot(x, y, x1, y1)
- Other routine labeling was provided on Figure 5-11.

Logarithmic Function

- The logarithmic function is the inverse of the exponential function. However, because it arises in many applications, it will be represented in the usual form with *x* as the independent variable and *y* as the dependent variable. The mathematical form is provided below and a curve is shown on the next slide.

$$y = \ln x$$

Figure 5-12. Logarithmic function to the base e.



Logarithms to Other Bases

- In general, the logarithm to a base *a* other than *e* is determined by the first equation below. The base 2 and the base 10 are also considered.

$$\log_a x = \frac{\ln x}{\ln a}$$

$$\log_2 x = \frac{\ln x}{\ln 2} = 1.4427 \ln x$$

$$\log_{10} x = \frac{\ln x}{\ln 10} = \frac{\ln x}{2.3026} = 0.4343 \ln x$$

MATLAB Logarithmic Commands

- The logarithm to the base *e* in MATLAB is
- >> y = log(x)
- This could be confusing since some math books use log(x) to mean to the base 10.

- The logarithm to the base 10 in MATLAB is
- >> y = log10(x)

Example 5-17. Some definitions are provided below.

$$G = \frac{P}{P_{ref}} = \text{ absolute power ratio}$$

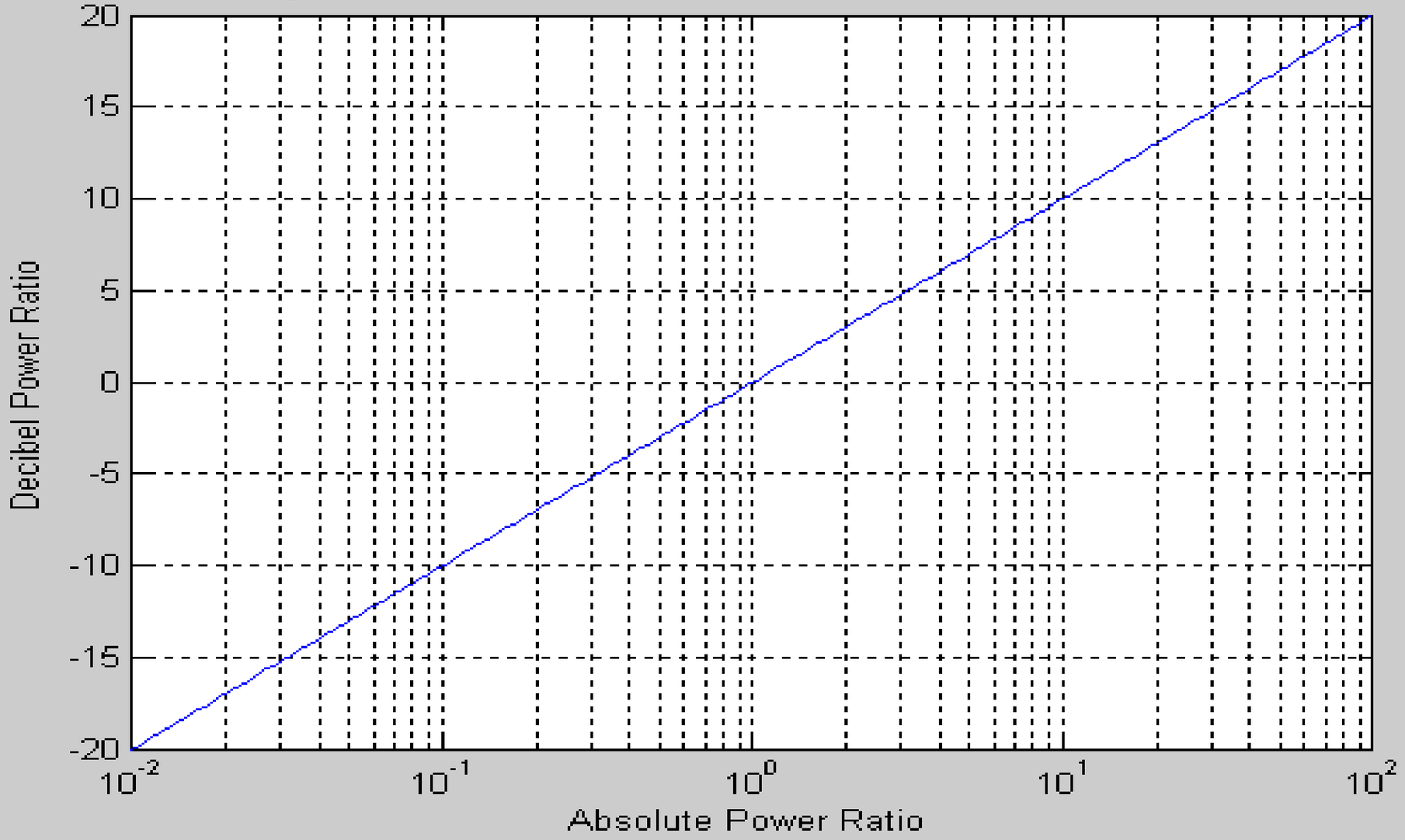
$$G_{\text{dB}} = 10 \log_{10} G$$

- Use MATLAB to develop a conversion curve in which G varies from 0.01 to 100. Use a semi-log plot with G on the horizontal logarithmic scale and G_{dB} on the vertical linear scale.

Example 5-17. Continuation.

- The command to generate G on a logarithmic scale from 0.01 to 100 is
- >> G = logspace(-2, 2, 200);
- The decibel gain is generated by
- >> GdB = 10*log10(G);
- A logarithmic x scale and a linear y scale are generated by the command
- >> semilogx(G, GdB)
- A grid and additional labeling are provided and the curve is shown on the next slide.

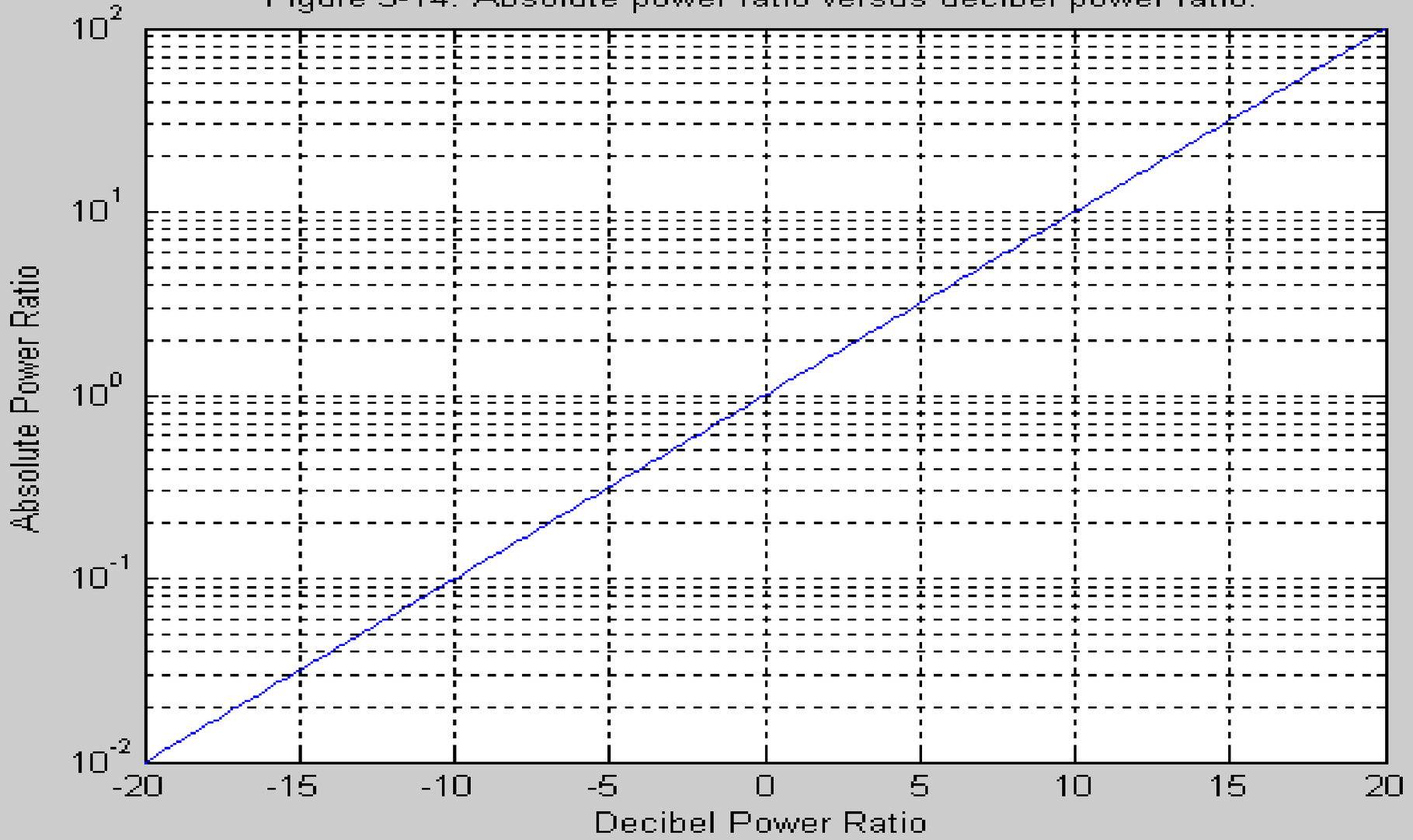
Figure 5-13. Decibel power ratio versus absolute power ratio.



Example 5-18. Plot the absolute gain versus the decibel gain from Example 5-17.

- We could solve for G in terms of GdB, but that is unnecessary since we have both G and GdB in memory. We simply reverse the order of the variables and change semilogx to semilogy. The command is
- >> semilogy(GdB, G)
- The plot with additional labeling is shown on the next slide.

Figure 5-14. Absolute power ratio versus decibel power ratio.



Example 5-19. Use MATLAB to plot the gaussian function shown below over the domain from -3 to 3.

$$y = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$$

- >> a = 1/(sqrt(2*pi));
- >> x = linspace(-3,3,301);
- >> y = a*exp(-0.5*x.^2);
- >> plot(x, y)
- With additional labeling, the curve is shown on the next slide.

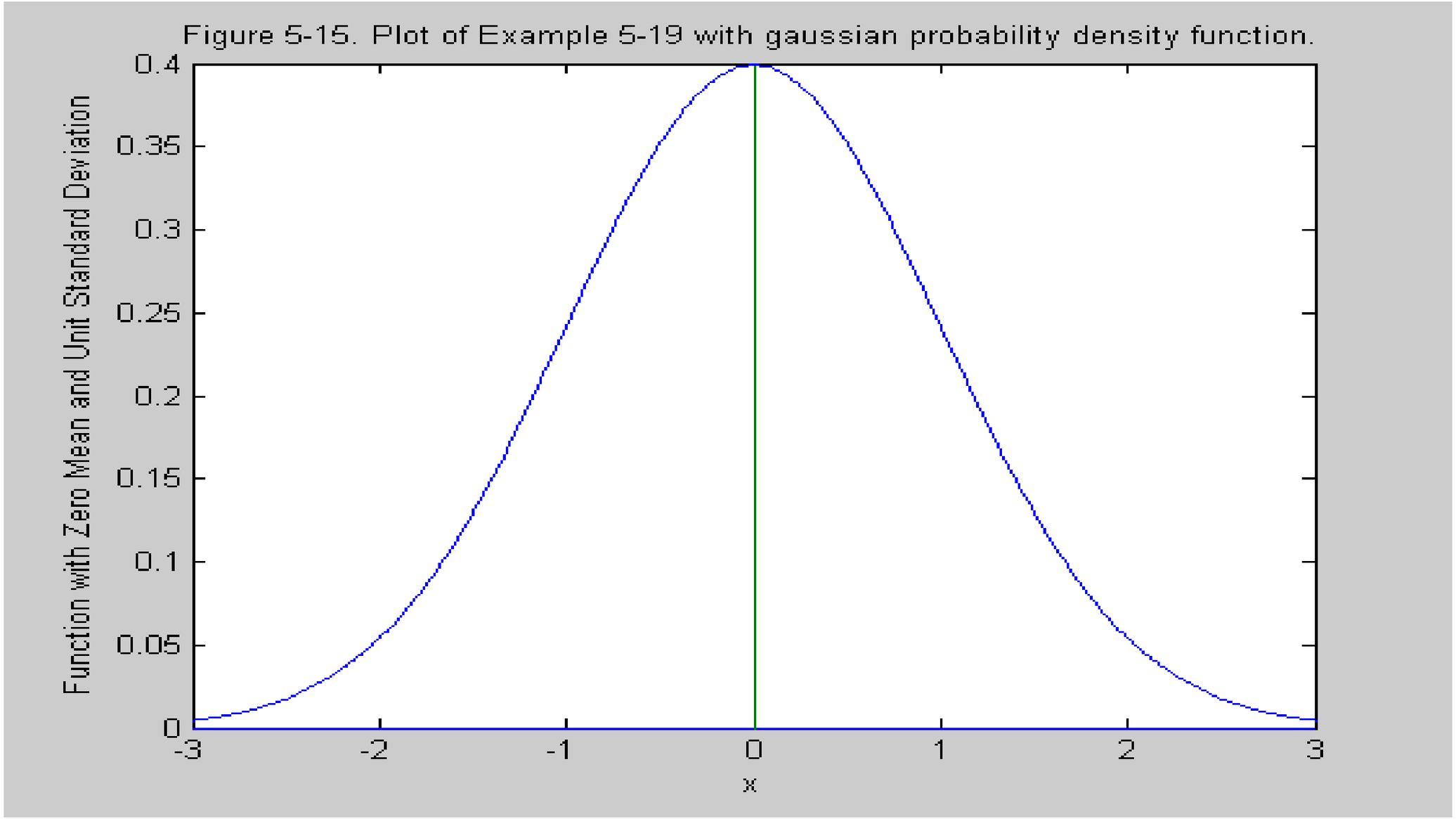


Figure 5-15. Plot of Example 5-19 with gaussian probability density function.
Function with Zero Mean and Unit Standard Deviation
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-3
-2
-1
0
1
2
3
x

Trigonometric Functions

- There are six basic trigonometric functions: (1) sine, (2) cosine, (3) tangent, (4) cotangent, (5) secant, and (6) cosecant. However, the first three tend to occur more often in practical applications than the latter three. Moreover, the latter three can be expressed as reciprocals of the first three (not in the order listed). Therefore, we will focus on the first three, but the definitions of the latter three will be provided for reference purposes.

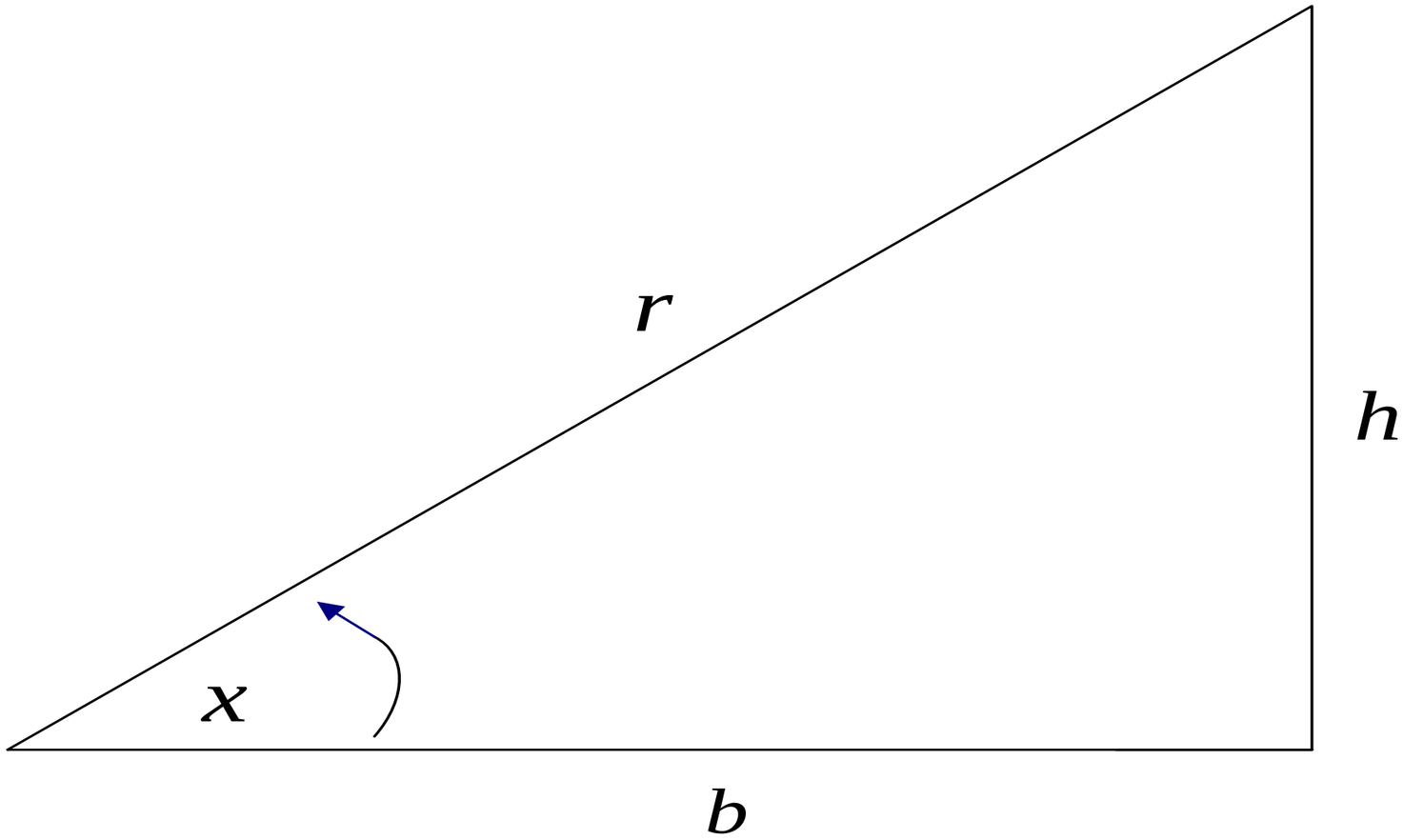
Angle Measurement

•The most basic mathematical unit for an angle is the *radian* (rad). It does have a mathematical basis for its form and does arise as a natural process. One complete revolution for a circle corresponds to 2π radians. To convert between radians and degrees, the following formulas can be used:

$$\text{Angle (degrees)} = \frac{180}{\pi} \times \text{Angle (radians)}$$

$$\text{Angle (radians)} = \frac{\pi}{180} \times \text{Angle (degrees)}$$

Figure 5-16. Right-triangle used to define trigonometric functions.



Trigonometric Definitions

$$\sin x = \frac{h}{r} \qquad \csc x = \frac{r}{h} = \frac{1}{\sin x}$$

$$\cos x = \frac{b}{r} \qquad \sec x = \frac{r}{b} = \frac{1}{\cos x}$$

$$\tan x = \frac{h}{b} \qquad \cot x = \frac{b}{h} = \frac{1}{\tan x}$$

Sine Function

- The form of the sine function over the domain from 0 to 2π is shown on the next slide.The function is *periodic*, meaning that it repeats the pattern shown for both positive and negative *x*. The domain shown constitutes *one cycle* of the periodic function and the *period* on an angular basis is 2π radians.

- The sine function is an odd function.

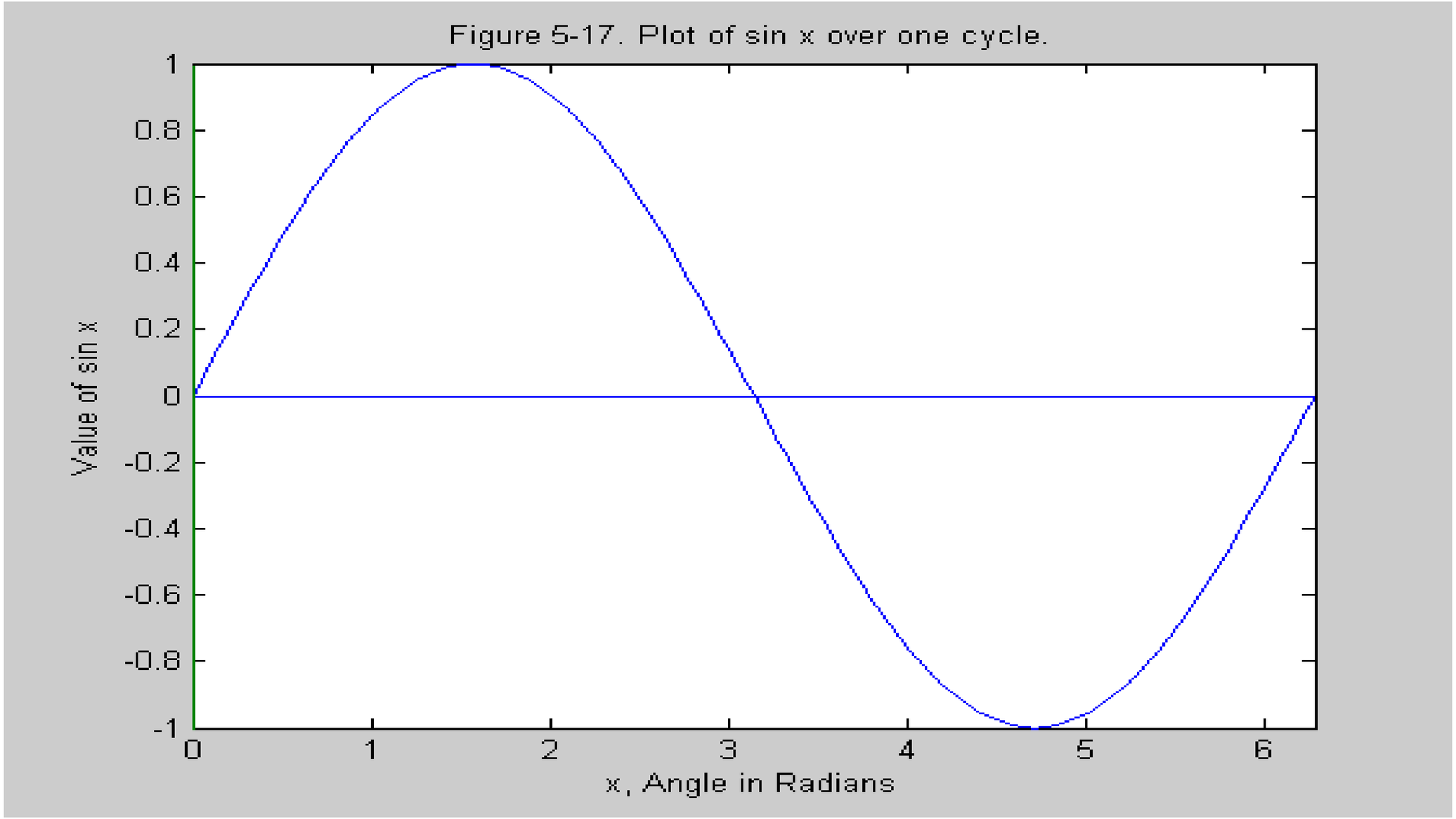


Figure 5-17. Plot of sin x over one cycle.
Value of sin x
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
1
2
3
4
5
6
x, Angle in Radians

Cosine Function

- The form of the cosine function over the domain from 0 to 2π is shown on the next slide. As in the case of the sine function, the cosine function is periodic with a period of 2π radians on an angular basis.

- The cosine function is an even function.

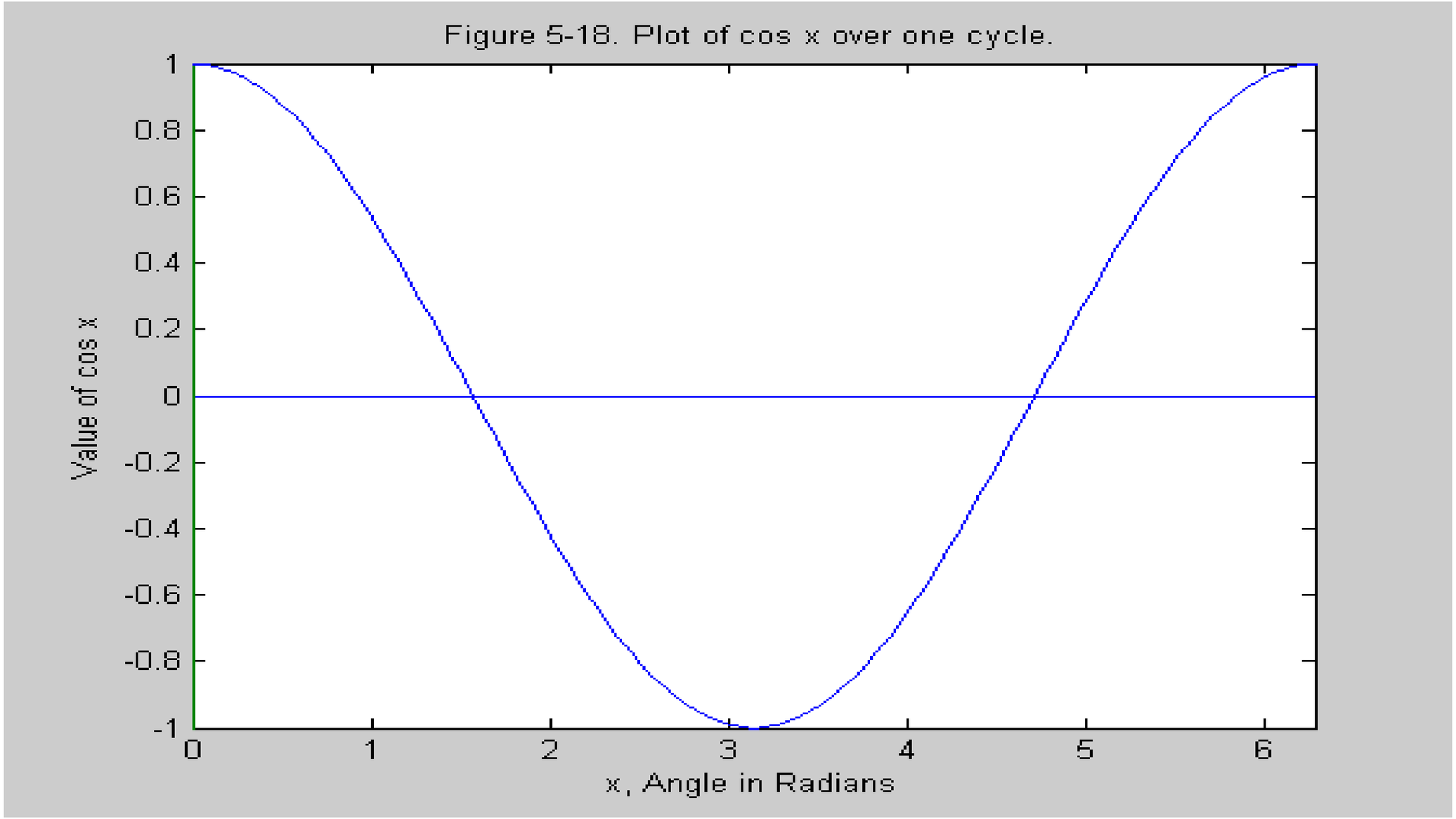


Figure 5-18. Plot of cos x over one cycle.
Value of cos x
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
1
2
3
4
5
6
x, Angle in Radians

Tangent Function

- The form of the tangent function over the domain from 0 to 2π is shown on the next slide.This function is periodic, but there are two cycles shown in the given domain. Hence, the tangent function is periodic with a period of π radians on an angular basis.
- The tangent function is an odd function. Moreover, it has infinite discontinuities at odd integer multiples of $\pi/2$.

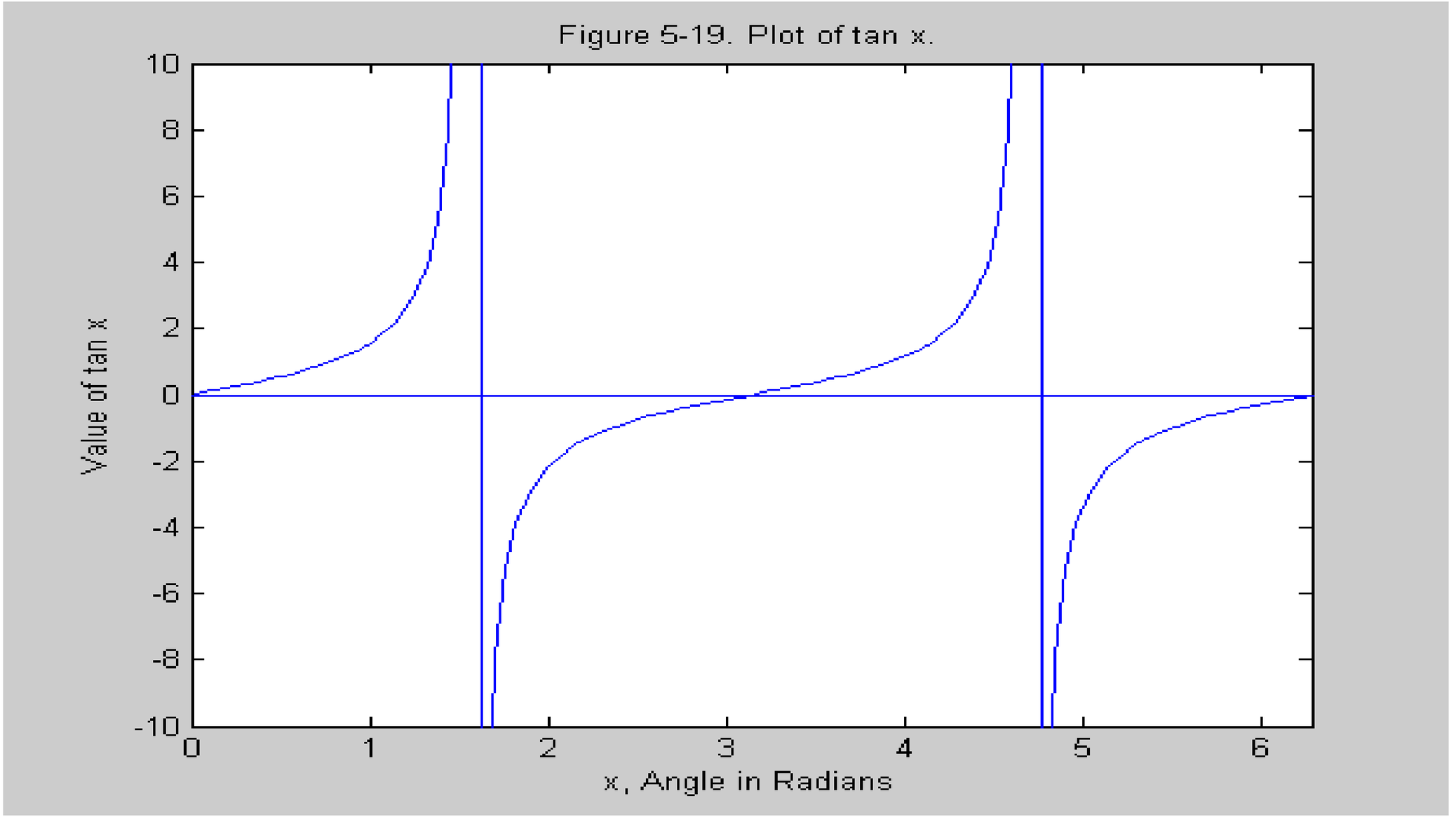


Figure 5-19. Plot of tan x.
Value of tan x
x, Angle in Radians
10
8
6
4
2
0
-2
-4
-6
-8
-10
0
1
2
3
4
5
6

MATLAB Trigonometric Functions

- The 6 MATLAB commands are
- >> y = sin(x)
- >> y = cos(x)
- >> y = tan(x)
- >> y = cot(x)
- >> y = sec(x)
- >> y = csc(x)

Sinusoidal Time Functions

$$y_s = B\sin\omega t$$

$$y_c = A\cos\omega t$$

$y_s =$ sine function and $y_c =$ cosine function

B or $A =$ amplitude or peak value of function

$\omega =$ angular frequency or angular velocity of function in radians/second (rad/s)

$t =$ time in seconds (s)

Period and Frequency

•For either the sine or cosine, the quantity ω is the number of radians per second that the function undergoes in the argument. This quantity is called the *angular velocity* in mechanics and is called the *angular frequency* in electricity. It is related to the cyclic frequency by the relationship

$$\omega = 2\pi f$$

$$T = 1/f$$

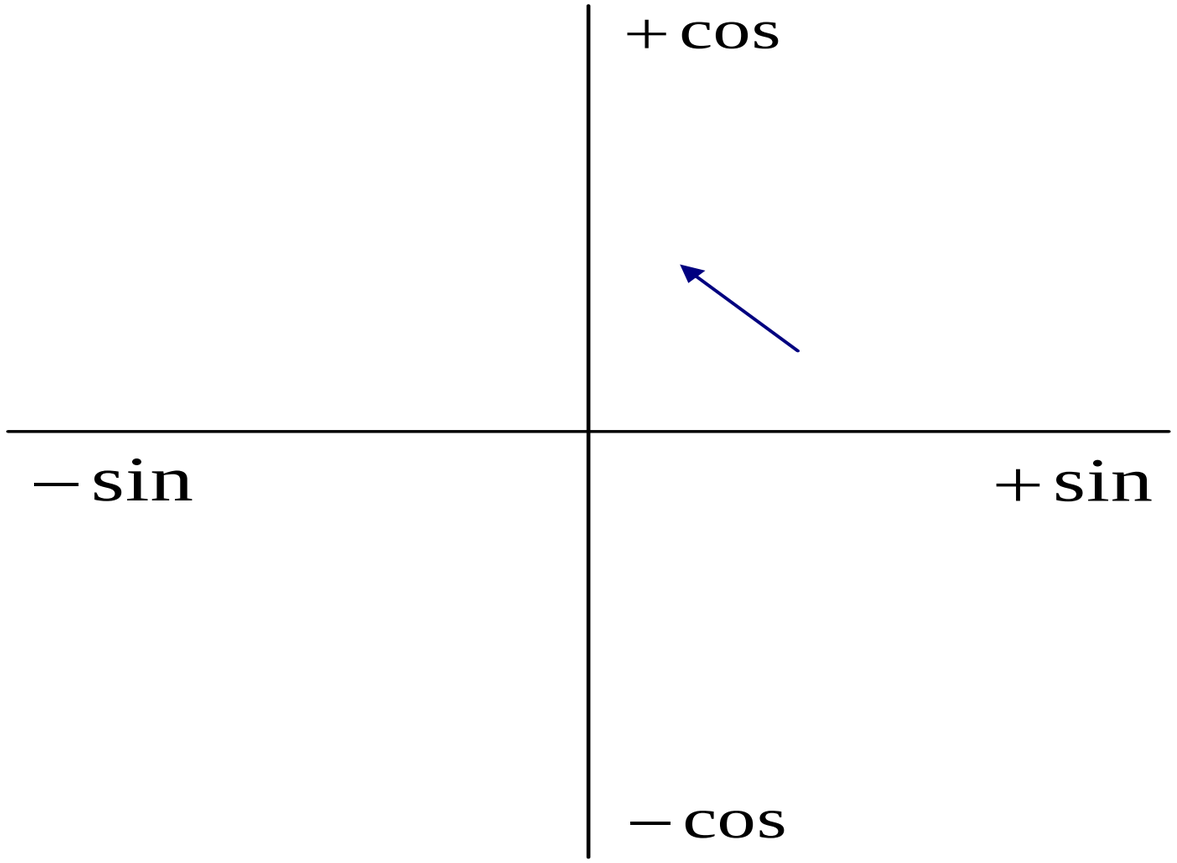
Combining Sine and Cosine Functions at the Same Frequency

$$y = B\sin\omega t + A\cos\omega t$$

$$= C\sin(\omega t + \theta) = C\cos(\omega t + \phi)$$

$$C = \sqrt{A^2 + B^2}$$

- The sum of a sine and a cosine function at the same frequency may be expressed as either a sine or a cosine function with an angle. The angle may be determined from the phase diagram on the next slide.


+ cos
− sin
+ sin
− cos

Example 5-20. Use MATLAB to plot the function below over two cycles.

$$y = 20\sin(\omega t + 30^{\circ})$$

$$= 20\sin(\omega t + \pi / 6)$$

$$\omega t = \frac{2\pi t}{T} = 2\pi x$$

```
x = linspace(0, 2, 201);
y = 20*sin(2*pi*x+pi/6);
plot(x, y)
```

The plot is shown on the next slide.

Figure 5-21. Sinusoidal function of Example 5-20.