

Scilab Textbook Companion for  
Principles Of Linear Systems And Signals  
by B. P. Lathi<sup>1</sup>

Created by  
A. Lasya Priya  
B.Tech (pursuing)  
Electrical Engineering  
NIT, Surathkal  
College Teacher  
H. Girisha Navada, NIT Surathkal  
Cross-Checked by  
S.M. Giridharan, IIT Bombay

July 30, 2019

<sup>1</sup>Funded by a grant from the National Mission on Education through ICT, <http://spoken-tutorial.org/NMEICT-Intro>. This Textbook Companion and Scilab codes written in it can be downloaded from the "Textbook Companion Project" section at the website <http://scilab.in>

# Book Description

**Title:** Principles Of Linear Systems And Signals

**Author:** B. P. Lathi

**Publisher:** Oxford University Press

**Edition:** 2

**Year:** 2009

**ISBN:** 0-19-806227-3

Scilab numbering policy used in this document and the relation to the above book.

**Exa** Example (Solved example)

**Eqn** Equation (Particular equation of the above book)

**AP** Appendix to Example(Scilab Code that is an Appednix to a particular Example of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means a scilab code whose theory is explained in Section 2.3 of the book.

# Contents

List of Scilab Codes	4
1 signals and systems	6
2 time domain analysis of continuous time systems	22
3 time domain analysis of discrete time systems	38
4 continuous time system analysis	56
5 discrete time system analysis using the z transform	74
6 continuous time signal analysis the fourier series	85
7 continuous time signal analysis the fourier transform	93
8 Sampling The bridge from continuous to discrete	107
9 fourier analysis of discrete time signals	113
10 state space analysis	130

# List of Scilab Codes

Exa 1.2	power and rms value . . . . .	6
Exa 1.3	time shifting . . . . .	7
Exa 1.4	time scaling . . . . .	10
Exa 1.5	time reversal . . . . .	11
Exa 1.6	basic signal models . . . . .	13
Exa 1.7	describing a signal in a single expression . . . . .	17
Exa 1.8	even and odd components of a signal . . . . .	18
Exa 1.10	input output equation . . . . .	19
Exa 2.5	unit impulse response for an LTIC system . . . . .	22
Exa 2.6	zero state response . . . . .	25
Exa 2.7	graphical convolution . . . . .	28
Exa 2.8	graphical convolution . . . . .	29
Exa 2.9	graphical convolution . . . . .	34
Exa 3.1	energy and power of a signal . . . . .	38
Exa 3.8	iterative solution . . . . .	40
Exa 3.9	iterative solution . . . . .	41
Exa 3.10	total response with given initial conditions . . . . .	42
Exa 3.11	iterative determination of unit impulse response . . . . .	46
Exa 3.13	convolution of discrete signals . . . . .	46
Exa 3.14	convolution of discrete signals . . . . .	48
Exa 3.16	sliding tape method of convolution . . . . .	49
Exa 3.17	total response with given initial conditions . . . . .	50
Exa 3.18	total response with given initial conditions . . . . .	51
Exa 3.19	forced response . . . . .	52
Exa 3.20	forced response . . . . .	55
Exa 4.1	laplace transform of exponential signal . . . . .	56
Exa 4.2	laplace transform of given fsignal . . . . .	56
Exa 4.3.a	laplace transform in case of different roots . . . . .	58

Exa 4.3.b	laplace transform in case of similar roots . . .	58
Exa 4.3.c	laplace transform in case of imaginary roots	59
Exa 4.4	laplace transform of a given signal . . . . .	59
Exa 4.5	inverse laplace transform . . . . .	60
Exa 4.8	time convolution property . . . . .	60
Exa 4.9	initial and final value . . . . .	61
Exa 4.10	second order linear differential equation . . .	61
Exa 4.11	solution to ode using laplace transform . . .	62
Exa 4.12	response to LTIC system . . . . .	62
Exa 4.15	loop current in a given network . . . . .	63
Exa 4.16	loop current in a given network . . . . .	63
Exa 4.17	voltage and current of a given network . . .	63
Exa 4.23	frequency response of a given system . . . .	64
Exa 4.24	frequency response of a given system . . . .	65
Exa 4.25	bode plots for given transfer function . . . .	65
Exa 4.26	bode plots for given transfer function . . . .	65
Exa 4.27	second order notch filter to suppress 60Hz hum	70
Exa 4.28	bilateral inverse transform . . . . .	71
Exa 4.29	current for a given RC network . . . . .	72
Exa 4.30	response of a noncausal sytem . . . . .	72
Exa 4.31	response of a fn with given tf . . . . .	73
Exa 5.1	z transform of a given signal . . . . .	74
Exa 5.2	z transform of a given signal . . . . .	74
Exa 5.3.a	z transform of a given signal with different roots . . . . .	76
Exa 5.3.c	z transform of a given signal with imaginary roots . . . . .	77
Exa 5.5	solution to differential equation . . . . .	78
Exa 5.6	response of an LTID system using difference eq . . . . .	78
Exa 5.10	response of an LTID system using difference eq . . . . .	79
Exa 5.12	maximum sampling timeinterval . . . . .	80
Exa 5.13	discrete time amplifier highest frequency . .	80
Exa 5.17	bilateral z transfrom . . . . .	80
Exa 5.18	bilateral inverse z transform . . . . .	81
Exa 5.19	transfer function for a causal system . . . .	82
Exa 5.20	zero state response for a given input . . . .	83

Exa 6.1	fourier coefficients of a periodic sequence . . .	85
Exa 6.2	fourier coefficients of a periodic sequence . . .	86
Exa 6.3	fourier spectra of a signal . . . . .	87
Exa 6.5	exponential fourier series . . . . .	88
Exa 6.7	exponential fourier series for the impulse train	90
Exa 6.9	exponential fourier series to find the output	91
Exa 7.1	fourier transform of exponential function . . .	93
Exa 7.4	inverse fourier transform . . . . .	95
Exa 7.5	inverse fourier transform . . . . .	97
Exa 7.6	fourier transform for everlasting sinusoid . . .	98
Exa 7.7	fourier transform of a periodic signal . . . . .	100
Exa 7.8	fourier transform of a unit impulse train . . .	101
Exa 7.9	fourier transform of unit step function . . . .	103
Exa 7.12	fourier transform of exponential function . . .	105
Exa 8.8	discrete fourier transform . . . . .	107
Exa 8.9	discrete fourier transform . . . . .	109
Exa 8.10	frequency response of a low pass filter . . . . .	110
Exa 9.1	discrete time fourier series . . . . .	113
Exa 9.2	DTFT for periodic sampled gate function . . .	114
Exa 9.3	discrete time fourier series . . . . .	116
Exa 9.4	discrete time fourier series . . . . .	119
Exa 9.5	DTFT for rectangular pulse . . . . .	121
Exa 9.6	DTFT for rectangular pulse spectrum . . . . .	123
Exa 9.9	DTFT of sinc function . . . . .	125
Exa 9.10.a	sketching the spectrum for a modulated signal	127
Exa 9.13	frequency response of LTID . . . . .	128
Exa 10.4	state space description by transfer function . .	130
Exa 10.5	finding the state vector . . . . .	130
Exa 10.6	state space description by transfer function . .	131
Exa 10.7	time domain method . . . . .	131
Exa 10.8	state space description by transfer function . .	132
Exa 10.9	state equations of a given systems . . . . .	132
Exa 10.10	diagonalized form of state equation . . . . .	133
Exa 10.11	controllability and observability . . . . .	133
Exa 10.12	state space description of a given description	134
Exa 10.13	total response using z transform . . . . .	134

# List of Figures

1.1	time shifting . . . . .	8
1.2	time shifting . . . . .	9
1.3	time scaling . . . . .	11
1.4	time scaling . . . . .	12
1.5	time reversal . . . . .	14
1.6	time reversal . . . . .	15
1.7	basic signal models . . . . .	16
1.8	describing a signal in a single expression . . . . .	17
1.9	even and odd components of a signal . . . . .	19
1.10	even and odd components of a signal . . . . .	20
2.1	unit impulse response for an LTIC system . . . . .	23
2.2	unit impulse response for an LTIC system . . . . .	24
2.3	zero state response . . . . .	26
2.4	zero state response . . . . .	27
2.5	graphical convolution . . . . .	30
2.6	graphical convolution . . . . .	31
2.7	graphical convolution . . . . .	32
2.8	graphical convolution . . . . .	33
2.9	graphical convolution . . . . .	35
2.10	graphical convolution . . . . .	36
3.1	energy and power of a signal . . . . .	39
3.2	iterative solution . . . . .	40



3.3	iterative solution . . . . .	41
3.4	total response with given initial conditions . . . . .	43
3.5	total response with given initial conditions . . . . .	44
3.6	iterative determination of unit impulse response . . . . .	45
3.7	convolution of discrete signals . . . . .	47
3.8	convolution of discrete signals . . . . .	48
3.9	sliding tape method of convolution . . . . .	49
3.10	total response with given initial conditions . . . . .	50
3.11	total response with given initial conditions . . . . .	51
3.12	forced response . . . . .	53
3.13	forced response . . . . .	54
4.1	laplace transform of exponential signal . . . . .	57
4.2	frequency response of a given system . . . . .	64
4.3	frequency response of a given system . . . . .	66
4.4	frequency response of a given system . . . . .	67
4.5	bode plots for given transfer function . . . . .	68
4.6	bode plots for given transfer function . . . . .	69
4.7	second order notch filter to suppress 60Hz hum . . . . .	70
5.1	z transform of a given signal . . . . .	75
5.2	response of an LTID system using difference eq . . . . .	79
6.1	fourier coefficients of a periodic sequence . . . . .	86
6.2	fourier coefficients of a periodic sequence . . . . .	87
6.3	exponential fourier series . . . . .	89
6.4	exponential fourier series for the impulse train . . . . .	90
6.5	exponential fourier series to find the output . . . . .	92
7.1	fourier transform of exponential function . . . . .	94
7.2	inverse fourier transform . . . . .	96
7.3	fourier transform for everlasting sinusoid . . . . .	99
7.4	fourier transform of a periodic signal . . . . .	100
7.5	fourier transform of a unit impulse train . . . . .	102
7.6	fourier transform of unit step function . . . . .	104
7.7	fourier transform of exponential function . . . . .	105
8.1	discrete fourier transform . . . . .	108
8.2	discrete fourier transform . . . . .	109

8.3	frequency response of a low pass filter . . . . .	111
9.1	discrete time fourier series . . . . .	114
9.2	DTFT for periodic sampled gate function . . . . .	115
9.3	discrete time fourier series . . . . .	116
9.4	discrete time fourier series . . . . .	119
9.5	discrete time fourier series . . . . .	120
9.6	DTFT for rectangular pulse . . . . .	122
9.7	DTFT for rectangular pulse spectrum . . . . .	124
9.8	DTFT of sinc function . . . . .	125
9.9	sketching the spectrum for a modulated signal . . . . .	127

# Chapter 1

## signals and systems

Scilab code Exa 1.2 power and rms value

```
1 //signals and systems
2 //power and rms value of a signal
3 clear
4 close
5 clc
6 //part a is a periodic function with period 2*pi/w0
7
8 disp("consider the power for almost infinite range")
9 ;
10 disp('part (a)')
11 disp("integrating ((c*cos(w0*t +theta))^2) for this
12     big range gives c^2/2 as the power which is
13     irrespective of w0");
14 disp("rms value is the square root of power and
15     therefpre equal to sqrt(c^2/2)\n\n");
16 //part b is the sum of 2 sinusoids
17 disp('part (b)')
18 disp("again integrating in the same way and ignoring
19     the zero terms we get (c1^2+c2^2)/2");
20 //part c deals with a complex signal
21 disp('part (c)')
```

```
17 disp("integrating the expression we get |D|^2 as the  
    power and |D| as the rms value");
```

---

### Scilab code Exa 1.3 time shifting

```
1 //signals and systems  
2 //time shifting  
3 clear  
4 close  
5 clc  
6 t=[-4:0.001:4];  
7 a=gca();  
8 plot(t,(exp(-2*t)).*(t>0))  
9 a.thickness=2;  
10 a.y_location="middle";  
11 xtitle('the signal x(t)')  
12 //delaying the function by 1 second we obtain  
13 figure  
14 a=gca();  
15 plot(t,(exp(-2*(t-1))).*((t>1)))  
16 a.thickness=2;  
17 a.y_location="middle";  
18 title('the signal x(t-1)')  
19 //advancing the function by 1 second we obtain  
20 figure  
21 a=gca();  
22 plot(t,(exp(-2*(t+1))).*(t>-1))  
23 a.thickness=2;  
24 a.y_location="middle";  
25 xtitle('the signal x(t+1)')
```

---

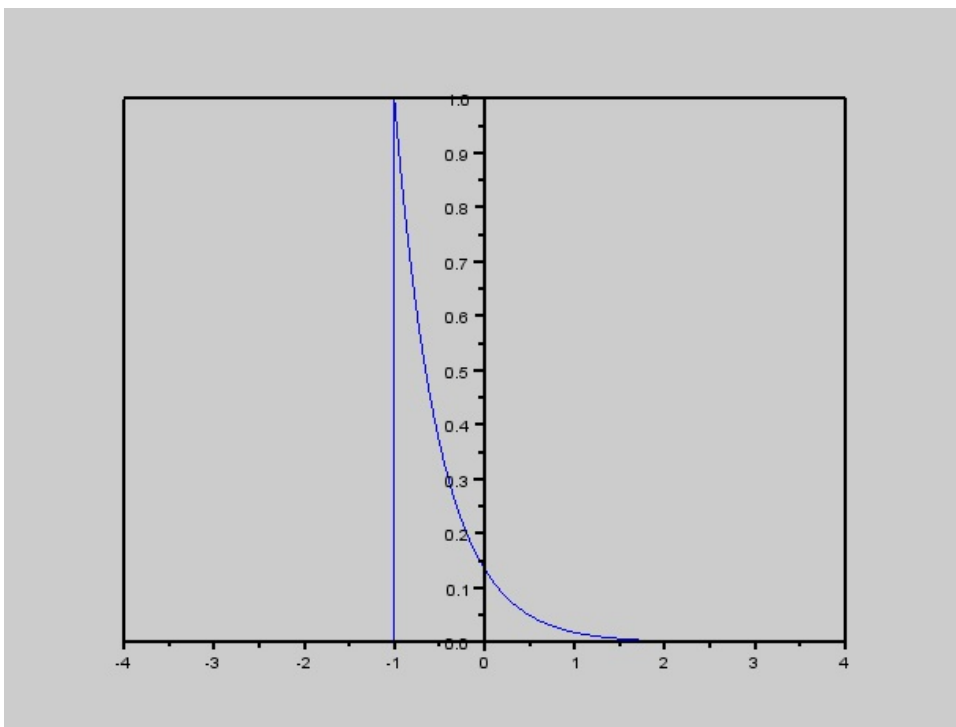


Figure 1.1: time shifting

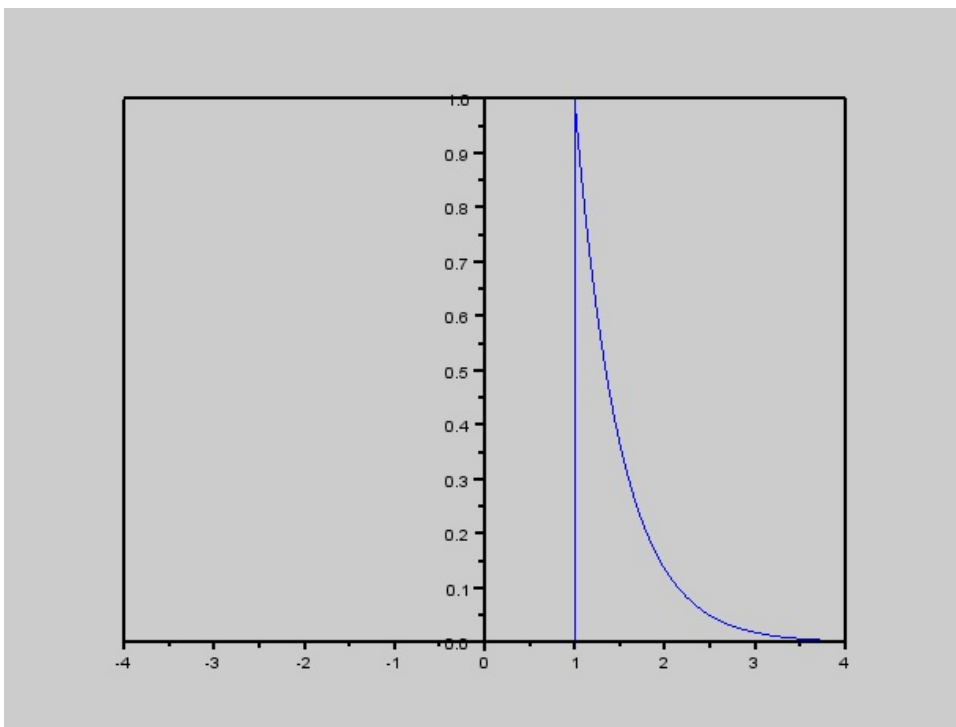


Figure 1.2: time shifting

### Scilab code Exa 1.4 time scaling

```
1 //signals and systems
2 //time scaling
3 clear
4 close
5 clc
6 t=[-4:0.1:6];
7 a=gca();
8 plot(t,2.*((t>-1.5)&(t<=0))+2*exp(-t/2).*((t>0)&(t
   <=3)));
9 figure
10 a.thickness=2;
11 a.y_location="middle";
12 xtitle('the signal x(t)');
13 //compressing this graph by a factor 3
14 a=gca();
15 plot(t,2.*((t>-0.5)&(t<=0))+2*exp(-3*t/2).*((t>0)&(t
   <=1)));
16 figure
17 a.thickness=2;
18 a.y_location="middle";
19 xtitle('the signal x(3t)');
20 //expanding this signal by a factor 2
21 a=gca();
22 plot(t,2.*((t>-3)&(t<=0))+2*exp(-t/4).*((t>0)&(t<=6)
   ));
23 a.thickness=2;
24 a.y_location="middle";
25 xtitle('the signal x(t/2)');
26 //the coordinates can b easily obtained from the
   graphs
```

---

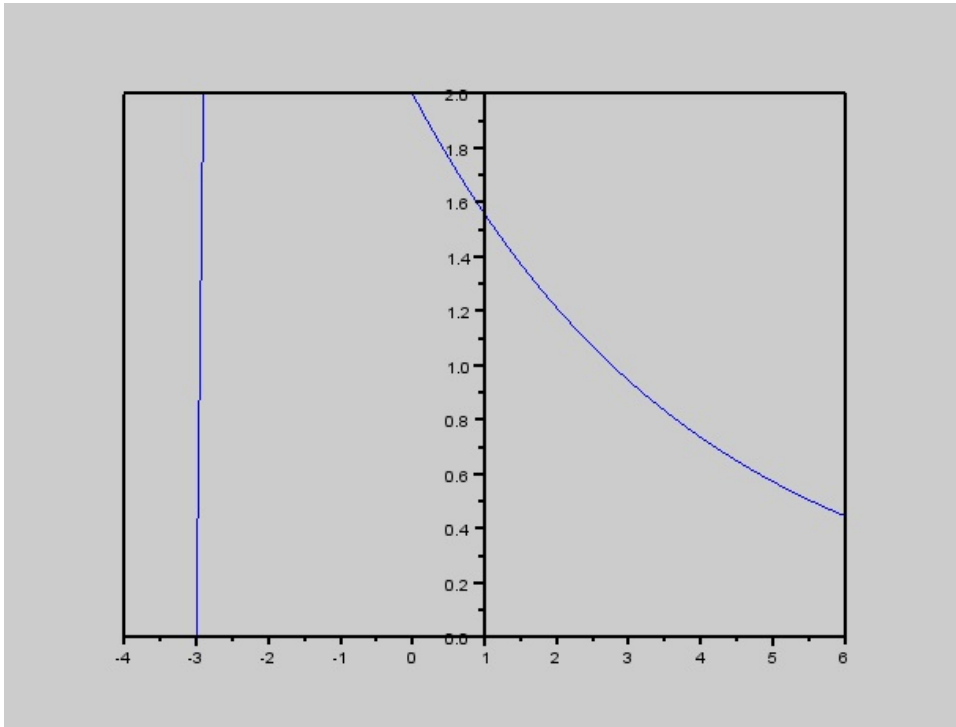


Figure 1.3: time scaling

**Scilab code Exa 1.5 time reversal**

```
1 //signals and systems
2 //time reversal
3 clear
4 close
5 clc
6 t=[-6:0.1:6];
```



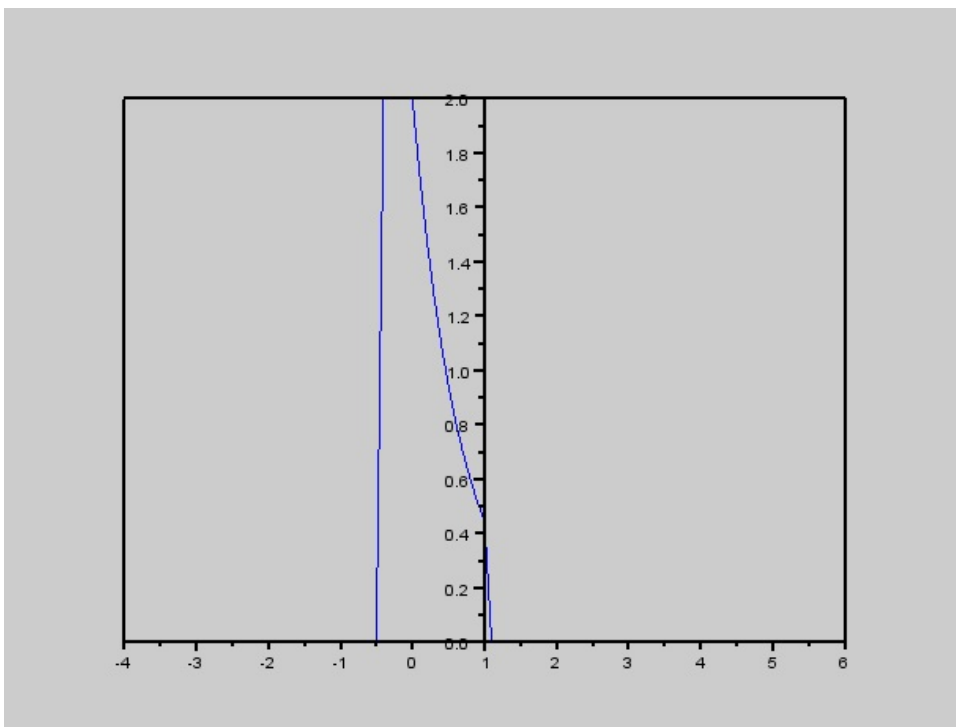


Figure 1.4: time scaling

```

7 a=gca();
8 plot(t,exp(t/2).*((t>=-5)&(t<=-1)));
9 figure
10 a.thickness=2;
11 a.y_location="middle";
12 xtitle('the signal x(t)')
13 //by replacing t by -t we get
14 a=gca();
15 plot(t,exp(-t/2).*((t>=1)&(t<5)));
16 a.thickness=2;
17 a.y_location="middle";
18 xtitle('the signal x(-t)')
19 //the coordinates can be easily observed from the
    graphs

```

---

### Scilab code Exa 1.6 basic signal models

```

1 //signals and systems
2 //representation of a signal
3 clear
4 close
5 clc
6 t=[0:0.1:5];
7 a=gca();
8 plot(t,t.*((t>=0)&(t<=2)) - 2*(t-3).*((t>2)&(t<=3)))
    ;
9 a.thickness=2;
10 a.y_location="middle";
11 xtitle('the signal x(t)')

```

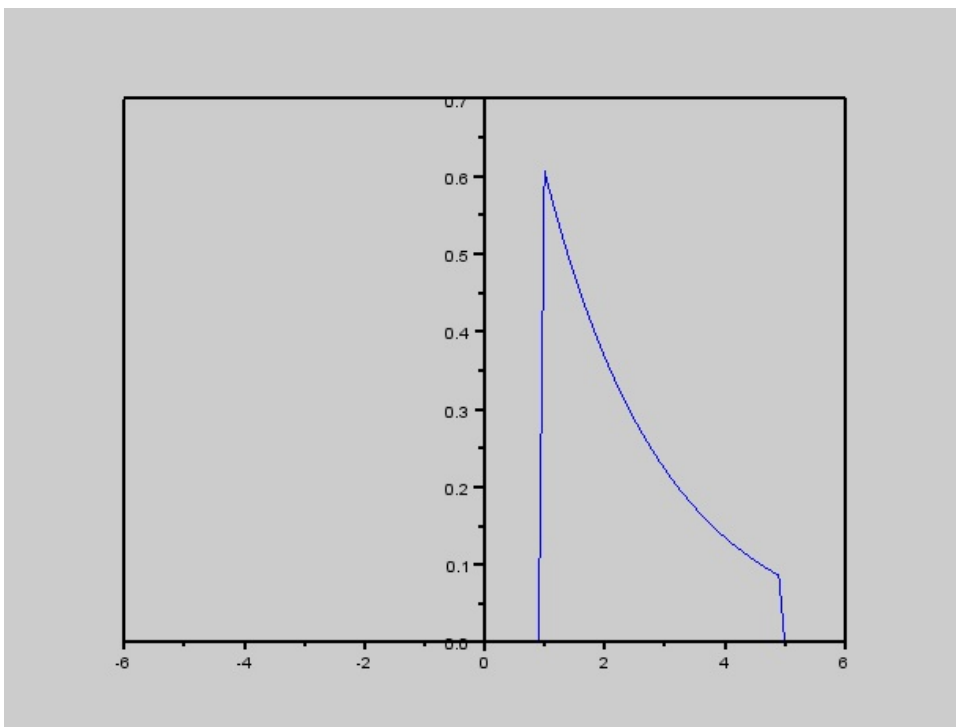


Figure 1.5: time reversal

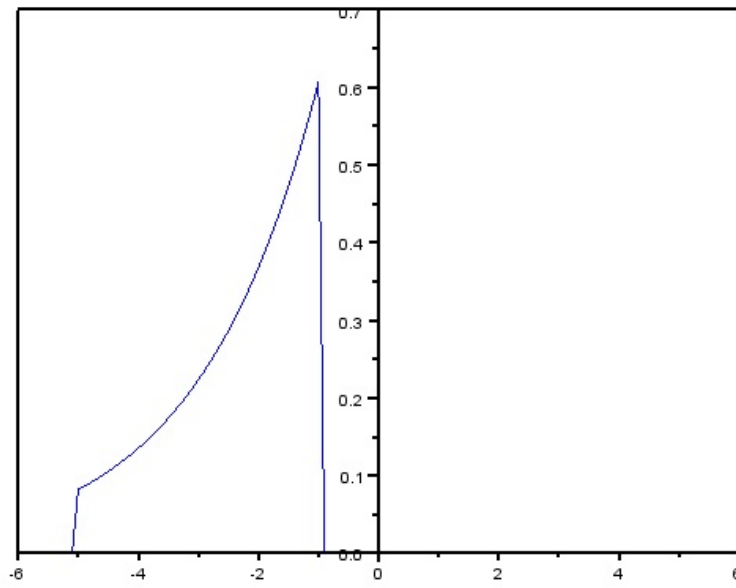


Figure 1.6: time reversal

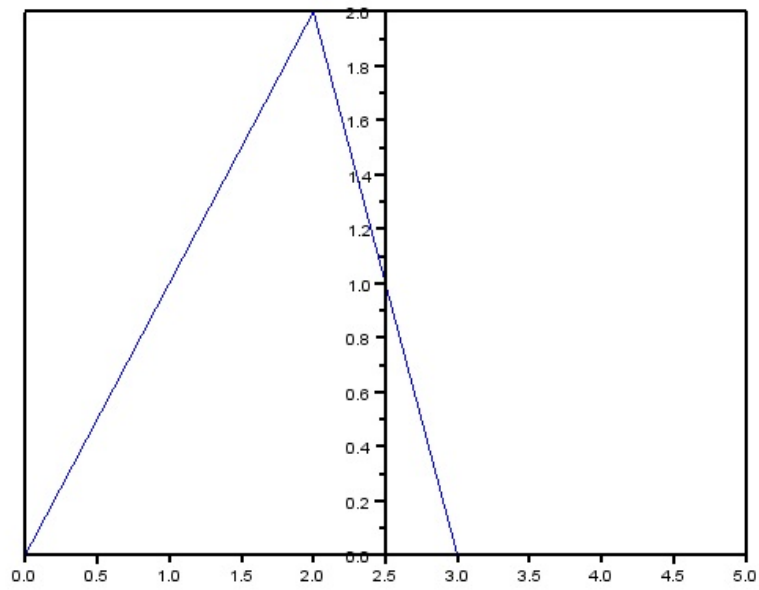


Figure 1.7: basic signal models

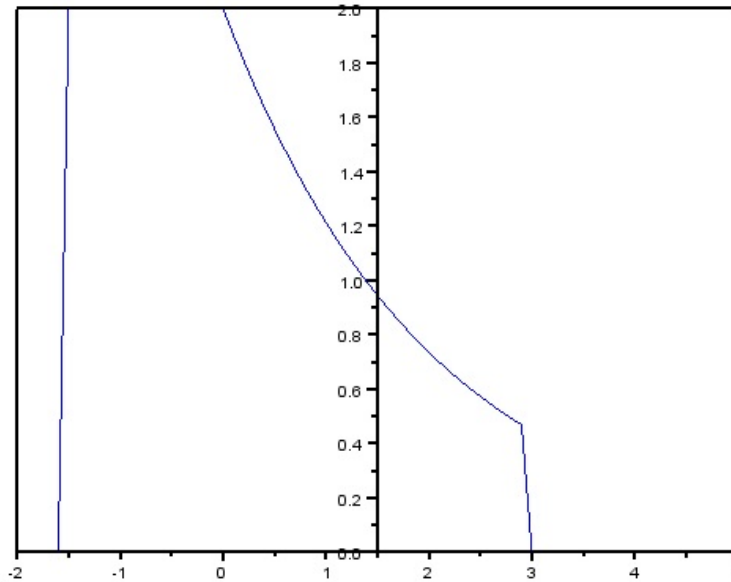


Figure 1.8: describing a signal in a single expression

```

12 //this can be written as a combination of 2 lines
13 disp("x(t)=x1(t)+x2(t)= tu(t) -3(t-2)u(t-2)+2(t-3)u(t-3)");

```

---

Scilab code Exa 1.7 describing a signal in a single expression

```

1 //signals and systems
2 //representation of a signal
3 clear
4 close
5 clc

```

```

6 t=[-2:0.1:5];
7 a=gca();
8 plot(t,2.*((t>=-1.5)&(t<0))+2*exp(-t/2).*((t>=0)&(t
    <3)));
9 a.thickness=2;
10 a.y_location="middle";
11 xtitle('the signal x(t-1)')
12 //this is a cobination of a constant function and an
    exponential function
13 disp("x(t)=x1(t)+x2(t)= 2u(t+1.5)-2(1-exp(-t/2))u(t)
    -2exp(-t/2)u(t-3)");

```

---

Scilab code Exa 1.8 even and odd components of a signal

```

1 //signals and systems
2 //odd and even components
3 clear
4 close
5 clc
6 t = 0:1/100:5;
7 x=exp(%i.*t);
8 y=exp(-%i.*t);
9 even=x./2+y./2;
10 odd=x./2-y./2;
11 figure
12 a=gca();
13 plot2d(t,even)
14 a.x_location='origin'
15 xtitle('even')
16 figure
17 a=gca();
18 plot2d(t,odd./%i)
19 a.x_location='origin'
20 xtitle('odd')

```

---

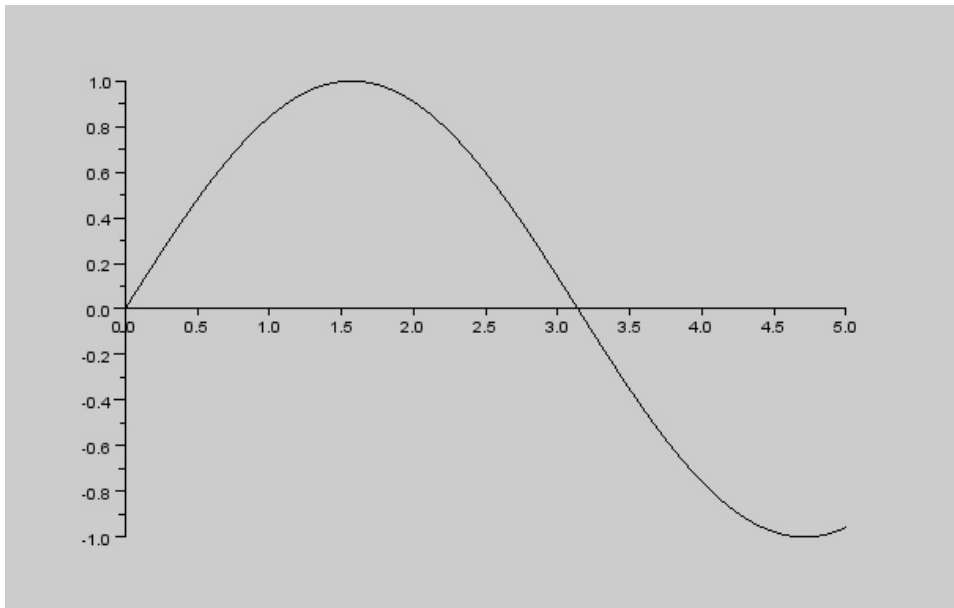


Figure 1.9: even and odd components of a signal

**Scilab code Exa 1.10** input output equation

```
1 //signals and systems
2 //formation of differential equation for a series RC
  circuit
3 clear
4 close
5 clc
6 r=15;
7 c=0.2;
8 //let the input voltage be x(t)
```



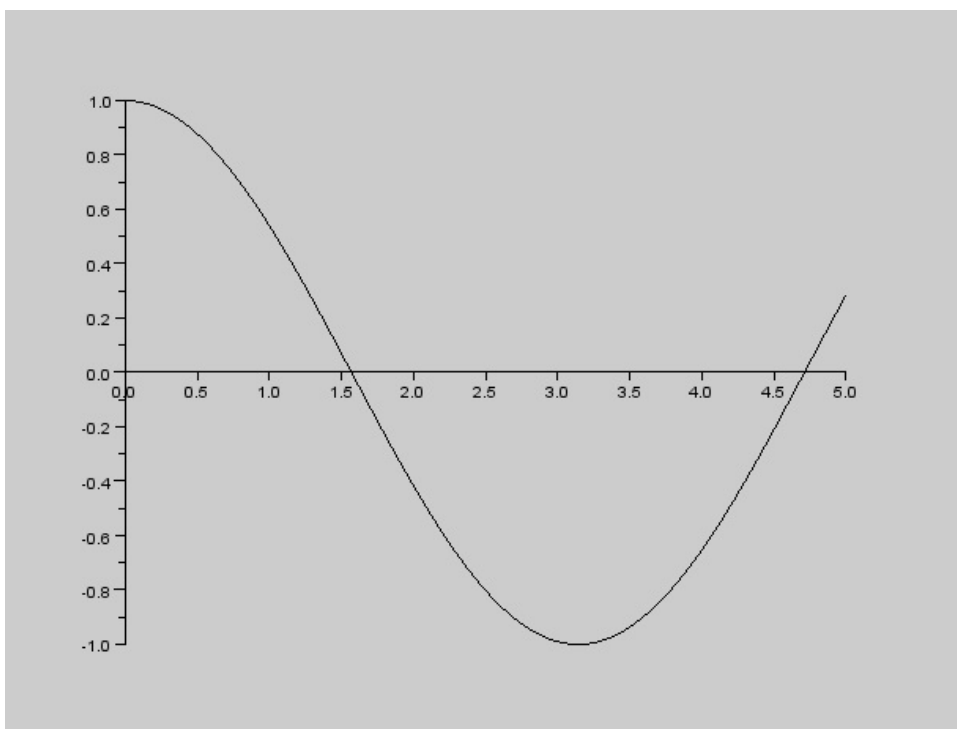


Figure 1.10: even and odd components of a signal

```
9 //let the loop current be i(t)
10 //let capacitor voltage be y(t)
11 disp("the loop equation 4 the circuit is given by r*
      i(t)+(5/D)*i(t)=x(t)")
12 disp("final form - (3D+1)y(t)=x(t)")
13 //the next few problems are of the same type where
      we have to frame the eqation based on the
      scenario
```

---

## Chapter 2

# time domain analysis of continuous time systems

Scilab code Exa 2.5 unit impulse response for an LTIC system

```
1 //time domain analysis of continuous time systems
2 //Convolution Integral of input  $x(t) = (e^{-t}).u(t)$ 
   and  $g(t) = (e^{-2*t}).u(t)$ 
3 clear;
4 close;
5 clc;
6 Max_Limit = 10;
7 t = 0:0.001:10;
8 for i=1:length(t)
9     g(i) = (exp(-2*t(i)));
10 end
11 x= exp(-(t));
12
13 y = convol(x,g)
14 figure
```

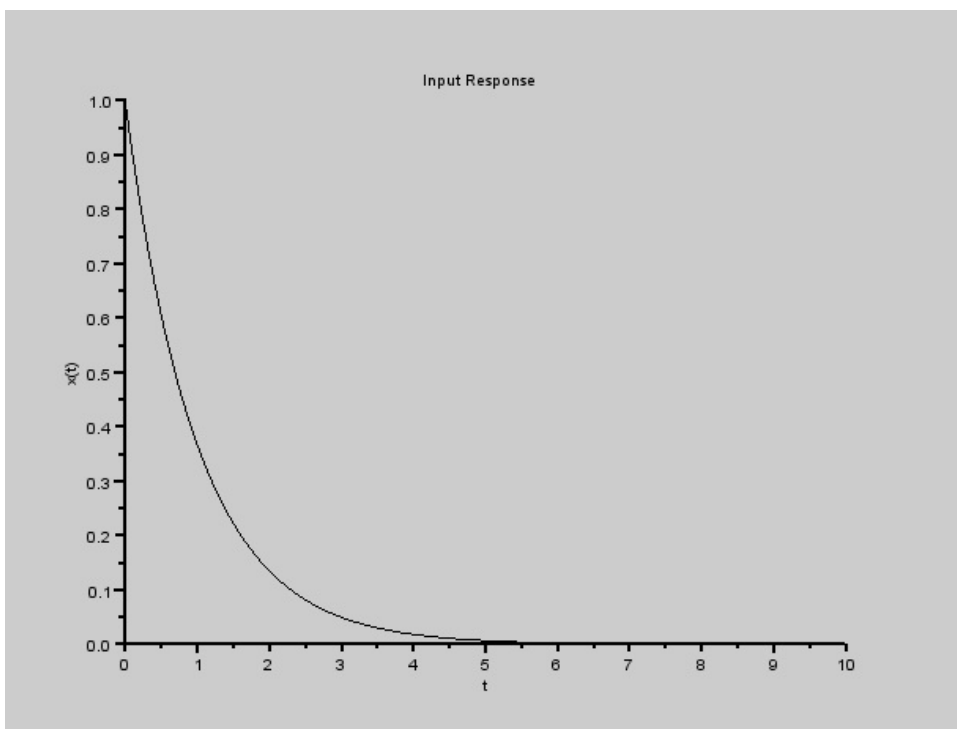


Figure 2.1: unit impulse response for an LTIC system

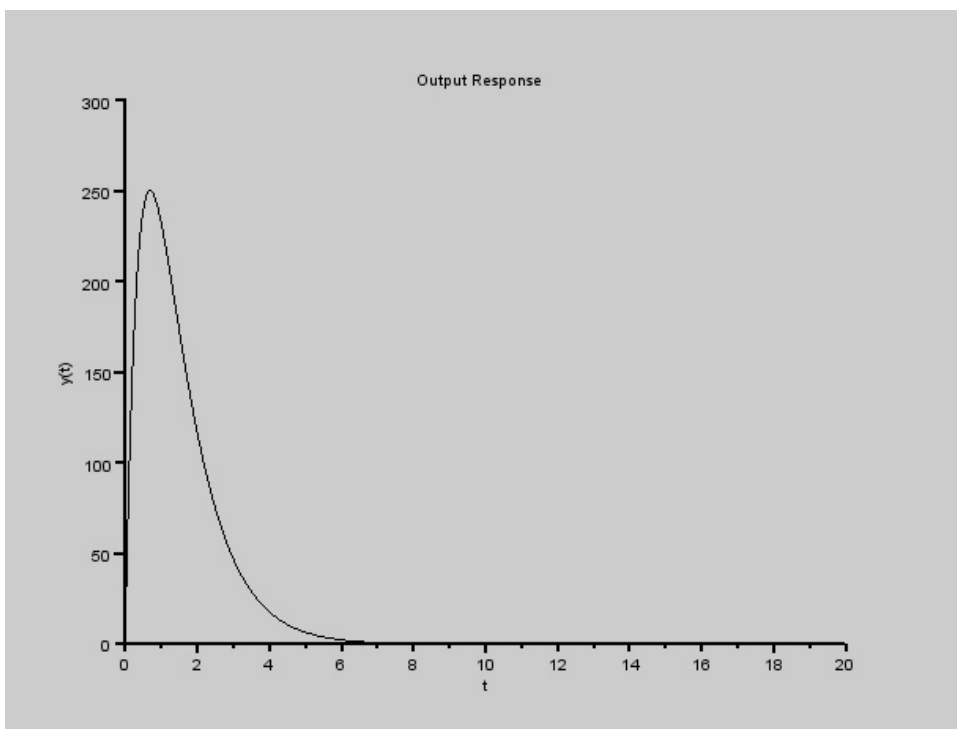


Figure 2.2: unit impulse response for an LTIC system

```

15 a=gca();
16 plot2d(t,g)
17 xtitle('Impulse Response','t','h(t)');
18 a.thickness = 2;
19 figure
20 a=gca();
21 plot2d(t,x)
22 xtitle('Input Response','t','x(t)');
23 a.thickness = 2;
24 figure
25 a=gca();
26 T=0:0.001:20;
27 plot2d(T,y)
28 xtitle('Output Response','t','y(t)');
29 a.thickness = 2;

```

---

### Scilab code Exa 2.6 zero state response

```

1 //time domain analysis of continuous time systems
2 //Convolution Integral of input  $x(t) = (e^{-3t}).u(t)$ 
   and  $h(t) = (2*e^{-2*t}-e^{-t}).u(t)$ 
3 clear;
4 close;
5 clc;
6 Max_Limit = 10;
7 t = 0:0.001:10;
8 for i=1:length(t)
9     g(i) = (2*exp(-2*t(i))-exp(-t(i)));
10 end
11 x= exp(-3*(t));
12

```

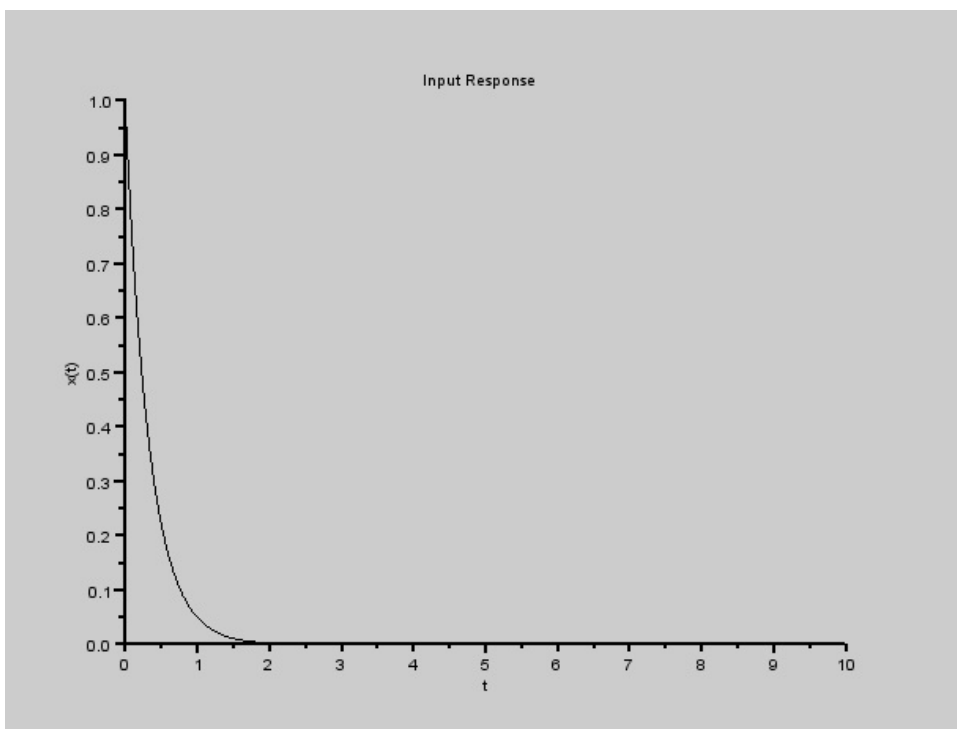


Figure 2.3: zero state response

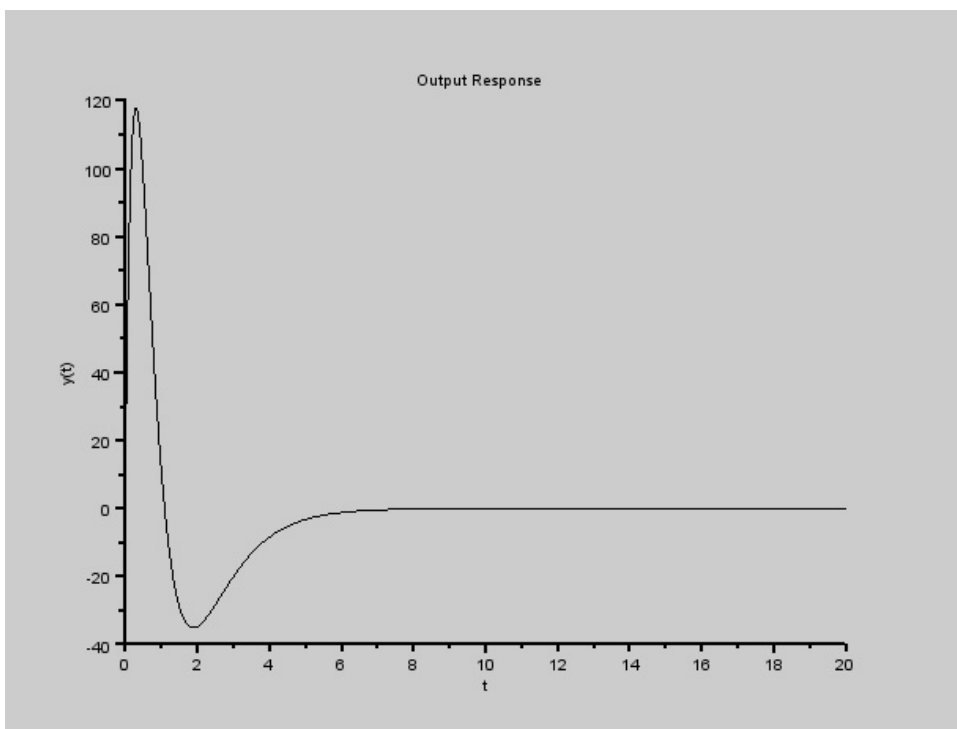


Figure 2.4: zero state response



```

13 y = convol(x,g)
14 figure
15 a=gca();
16 plot2d(t,g)
17 xtitle('Impulse Response','t','h(t)');
18 a.thickness = 2;
19 figure
20 a=gca();
21 plot2d(t,x)
22 xtitle('Input Response','t','x(t)');
23 a.thickness = 2;
24 figure
25 a=gca();
26 T=0:0.001:20;
27 plot2d(T,y)
28 xtitle('Output Response','t','y(t)');
29 a.thickness = 2;

```

---

### Scilab code Exa 2.7 graphical convolution

```

1 //time domain analysis of continuous time systems
2 //Convolution Integral of input  $x(t) = (e^{-t}).u(t)$ 
   and  $g(t) = u(t)$ 
3 clear;
4 close;
5 clc;
6 Max_Limit = 10;
7 t = -10:0.001:10;
8 for i=1:length(t)
9
10     g(i)=exp(-t(i));
11     x(i)=exp(-2*t(i));
12 end
13
14 y = convol(x,g)

```

```

15 figure
16 a=gca();
17 plot2d(t,g)
18 xtitle('Impulse Response','t','h(t)');
19 a.thickness = 2;
20 figure
21 a=gca();
22 plot2d(t,x)
23 xtitle('Input Response','t','x(t)');
24 a.thickness = 2;
25 figure
26 a=gca();
27 T=-20:0.001:20;
28 plot2d(T,y)
29 xtitle('Output Response','t','y(t)');
30 a.thickness = 2;

```

---

### Scilab code Exa 2.8 graphical convolution

```

1 //time domain analysis of continuous time systems
2 //Convolution Integral of input  $x(t) = (e^{-t}).u(t)$ 
   and  $g(t) =u(t)$ 
3 clear;
4 close;
5 clc;
6 Max_Limit = 10;
7 t = -10:0.001:10;

```

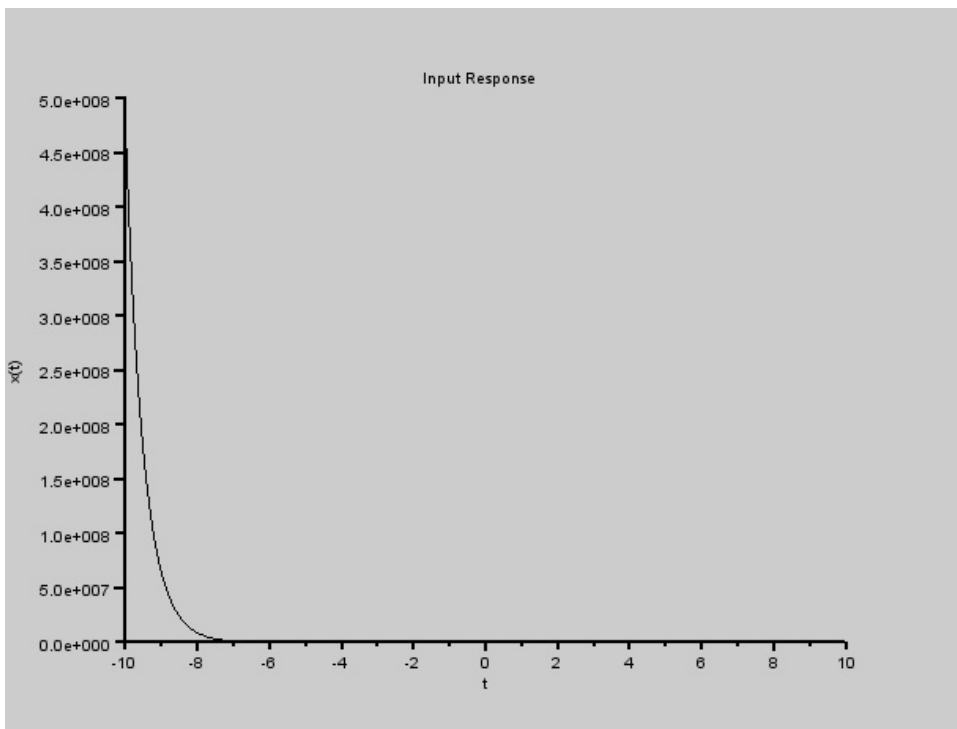


Figure 2.5: graphical convolution

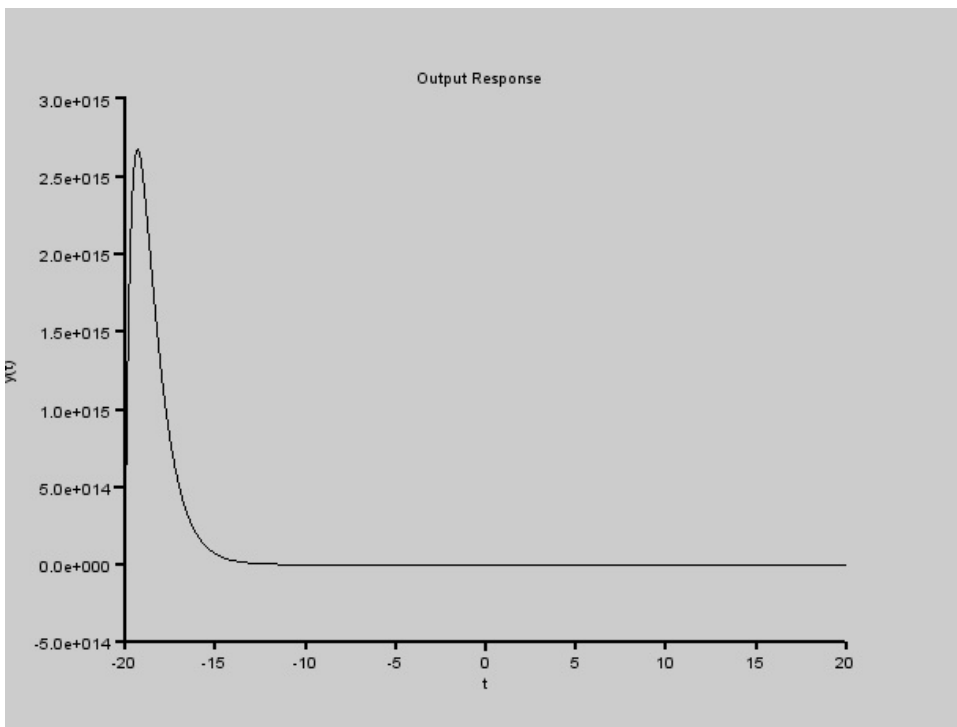


Figure 2.6: graphical convolution

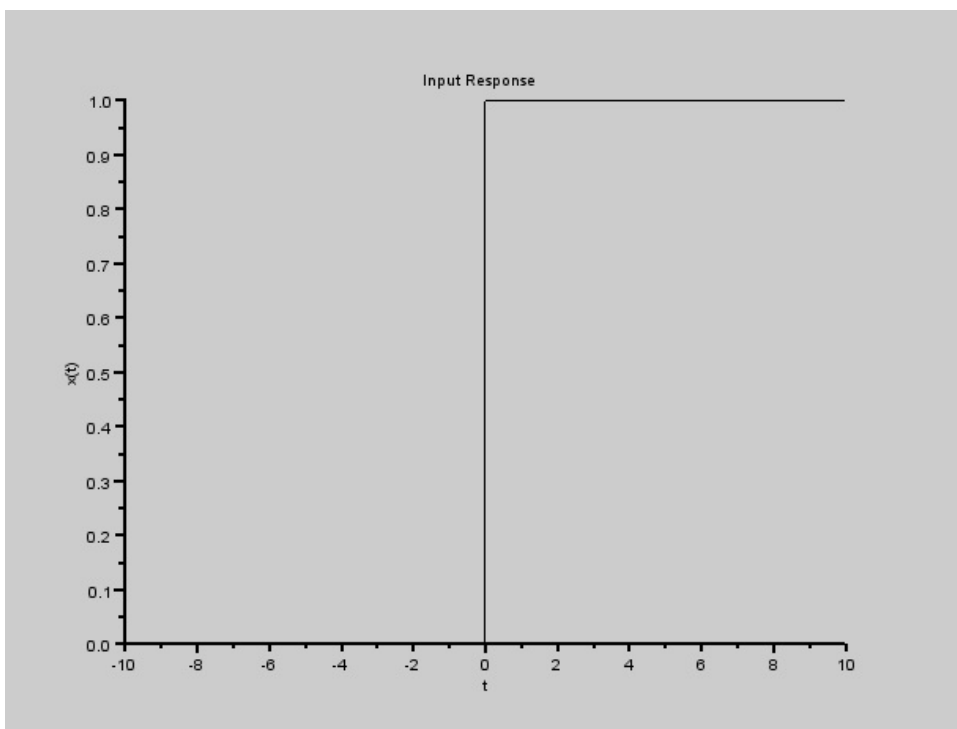


Figure 2.7: graphical convolution

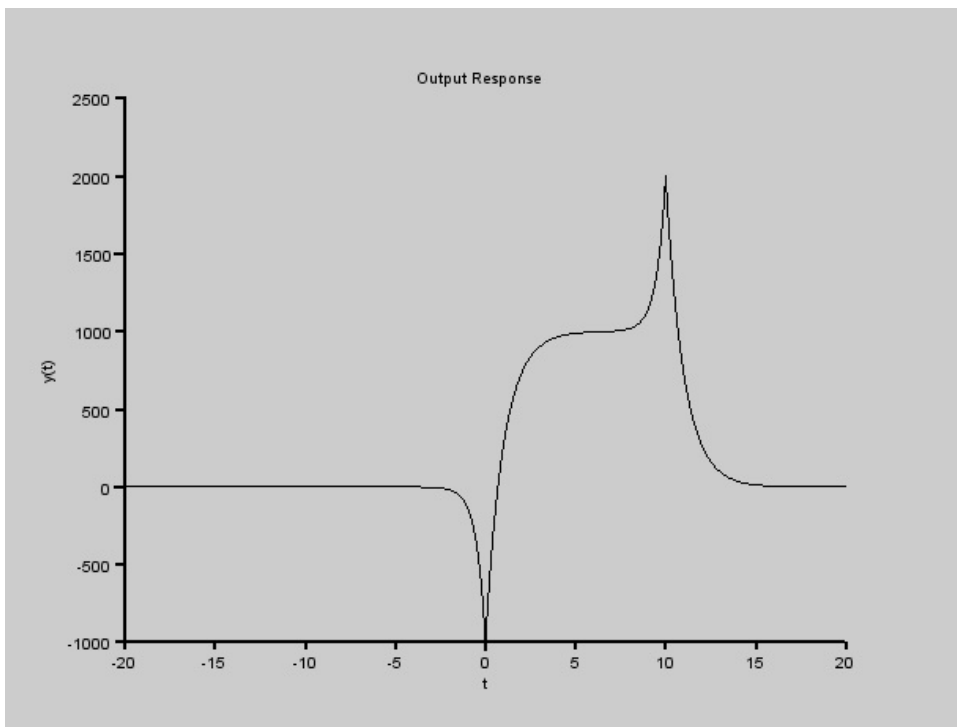


Figure 2.8: graphical convolution

```

8 for i=1:length(t)
9     if t(i)<0 then
10         g(i)=-2*exp(2*t(i));
11         x(i)=0;
12     else
13         g(i)=2*exp(-t(i));
14         x(i)=1;
15     end
16 end
17
18 y = convol(x,g)
19 figure
20 a=gca();
21 plot2d(t,g)
22 xtitle('Impulse Response','t','h(t)');
23 a.thickness = 2;
24 figure
25 a=gca();
26 plot2d(t,x)
27 xtitle('Input Response','t','x(t)');
28 a.thickness = 2;
29 figure
30 a=gca();
31 T=-20:0.001:20;
32 plot2d(T,y)
33 xtitle('Output Response','t','y(t)');
34 a.thickness = 2;

```

---

### Scilab code Exa 2.9 graphical convolution

```
1 //time domain analysis of continuous time systems
```

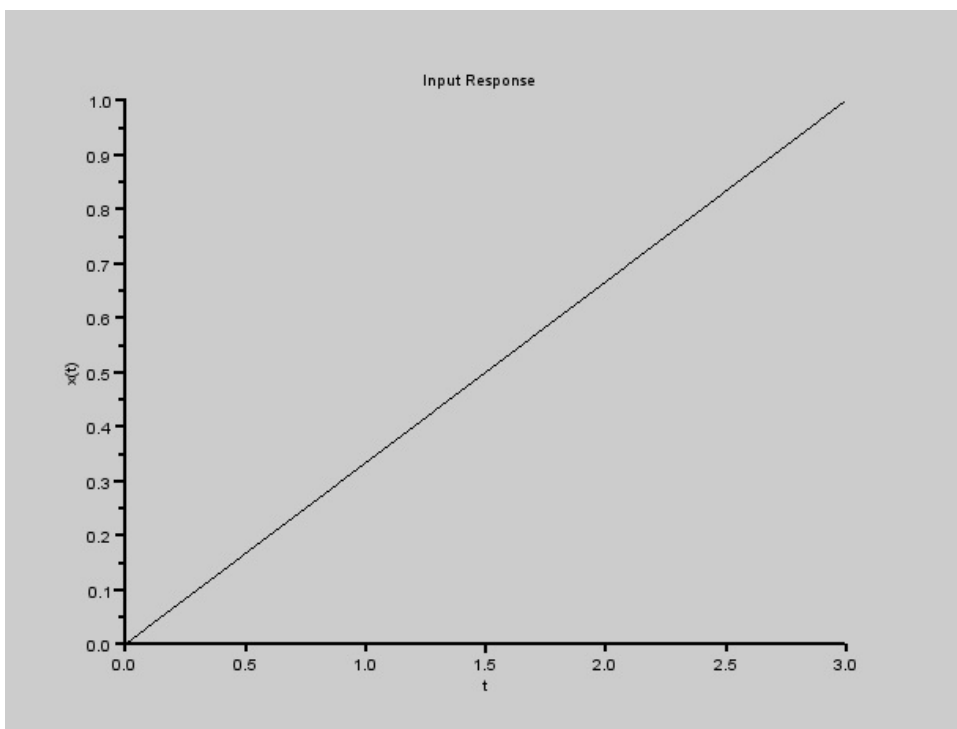


Figure 2.9: graphical convolution



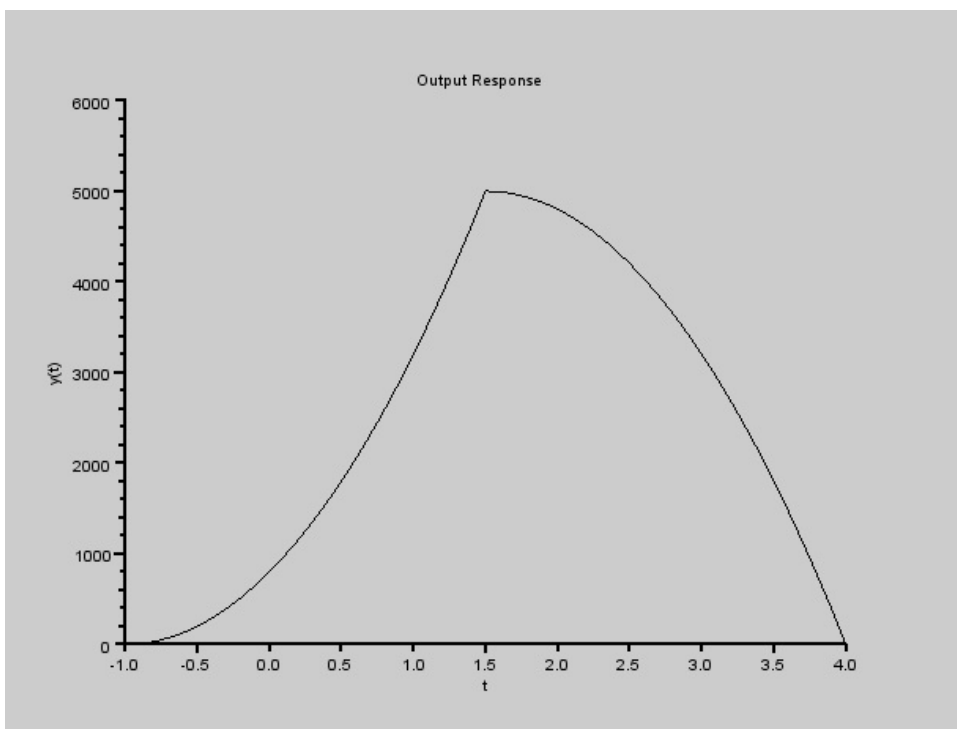


Figure 2.10: graphical convolution

```

2 //Convolution Integral of input  $x(t) = (e^{-t}).u(t)$ 
   and  $g(t) =u(t)$ 
3 clear;
4 close;
5 clc;
6 Max_Limit = 10;
7 t =linspace(-1,1,10001);
8 for i=1:length(t)
9     g(i)=1;
10 end
11 t1=linspace(0,3,10001);
12 for i=1:length(t1)
13 x(i)= t1(i)/3;
14 end
15 y = convol(x,g);
16 figure
17 a=gca();
18 size(t)
19 size(g)
20 plot2d(t,g)
21 xtitle('Impulse Response','t','h(t)');
22 a.thickness = 2;
23 figure
24 a=gca();
25 size(x)
26 plot2d(t1,x)
27 xtitle('Input Response','t','x(t)');
28 a.thickness = 2;
29 figure
30 a=gca();
31 T=linspace(-1,4,20001);
32 size(y)
33 plot2d(T,y)
34 xtitle('Output Response','t','y(t)');
35 a.thickness = 2;

```

---

# Chapter 3

## time domain analysis of discrete time systems

Scilab code Exa 3.1 energy and power of a signal

```
1 //signals and systems
2 //time domain analysis of discrete time systems
3 //energy of a signal
4 clear;
5 close;
6 clc;
7 n=0:1:5
8 figure
9 a=gca();
10 plot2d(n,n);
11 energy=sum(n^2)
12 power=(1/6)*sum(n^2)
13 disp(energy)
14 disp(power)
```

---

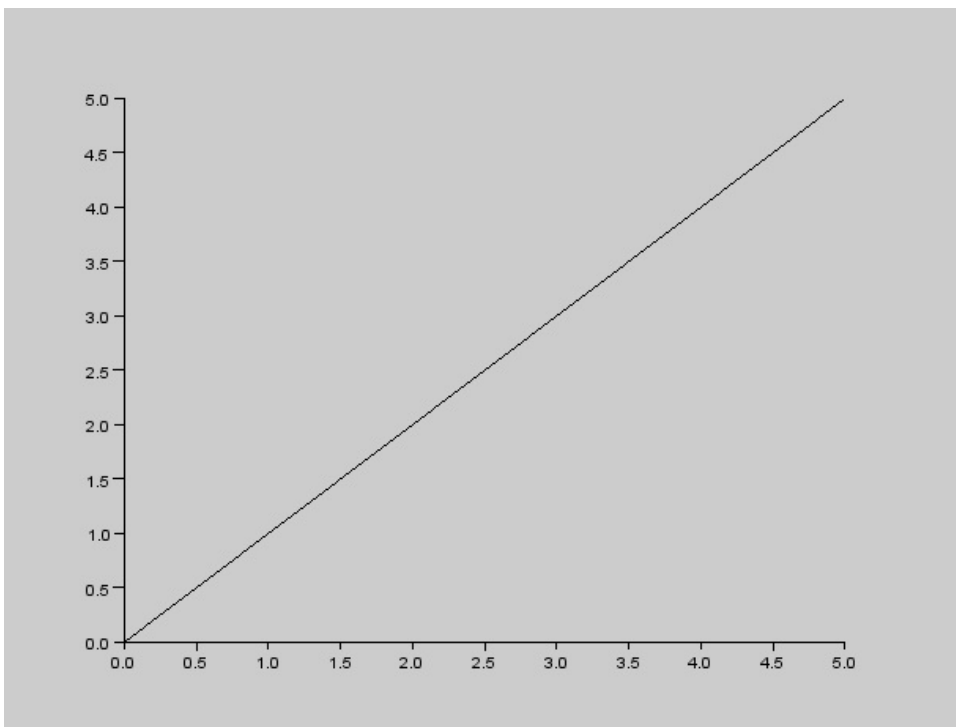


Figure 3.1: energy and power of a signal

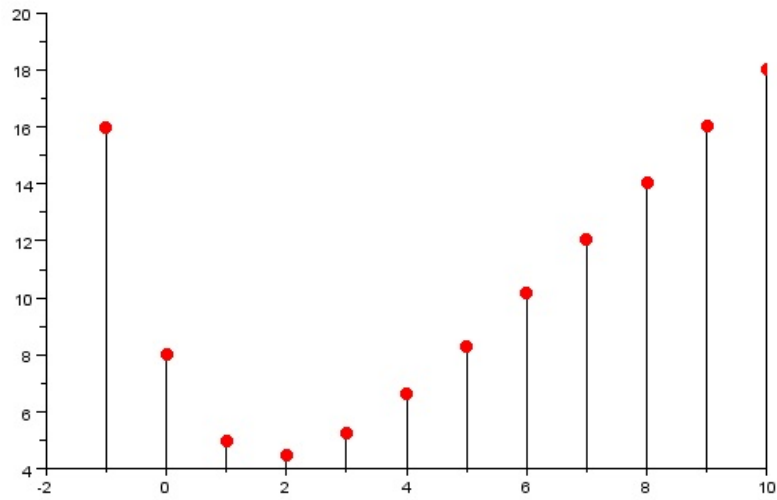


Figure 3.2: iterative solution

### Scilab code Exa 3.8 iterative solution

```

1 //signals and systems
2 //time domain analysis of discrete time systems
3 //iterative solution
4 clear;
5 close;
6 clf;
7 n=(-1:10)';
8 y=[16;0;zeros(length(n)-2,1)];
9 x=[0;0;n(3:length(n))];
10 for k=1:length(n)-1
11     y(k+1)=0.5*y(k)+x(k+1);
12 end;
13 clf;

```

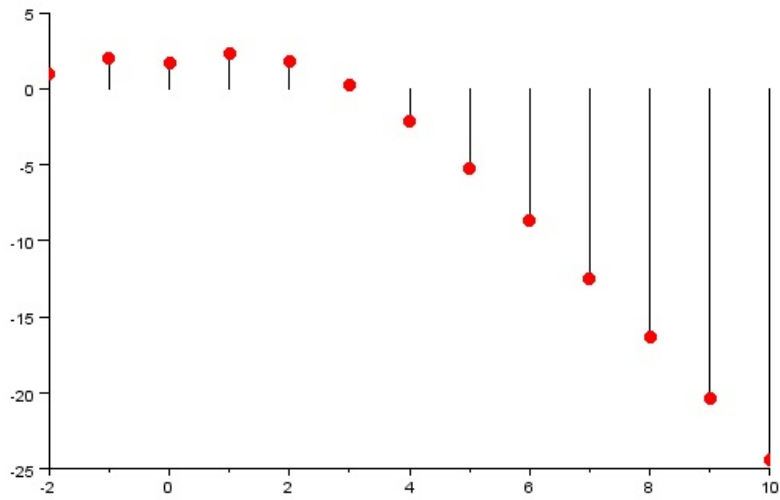


Figure 3.3: iterative solution

```

14 size(y)
15 size(n)
16 plot2d3(n,y);
17 plot(n,y,'r. ')
18 disp([sprintf('%d %d\n',[n,y])]);

```

---

### Scilab code Exa 3.9 iterative solution

```

1 //signals and systems
2 //time domain analysis of discrete time systems
3 //iterative solution
4 clear;
5 close;
6 clc;

```

```

7 n=(-2:10)';
8 y=[1;2;zeros(length(n)-2,1)];
9 x=[0;0;n(3:length(n))];
10 for k=1:length(n)-2
11     y(k+2)=y(k+1)-0.24*y(k)+x(k+2)-2*x(k+1);
12 end;
13 clf;
14 plot2d3(n,y);
15 disp([sprintf('%d %d\n',[n,y])]);

```

---

Scilab code Exa 3.10 total response with given initial conditions

```

1 //signals and systems
2 //time domain analysis of discrete time systems
3 //total response with initial conditions
4 clear;
5 close;
6 clc;
7 n=(-2:10)';
8 y=[25/4;0;zeros(length(n)-2,1)];
9 x=[0;0;4^-n(3:length(n))];
10 for k=1:length(n)-2
11     y(k+2)=0.6*y(k+1)+0.16*y(k)+5*x(k+2);
12 end;
13 clf;
14 a=gca();
15 plot2d3(n,y);
16
17 y1=[25/4;0;zeros(length(n)-2,1)];
18 x=[0;0;4^-n(3:length(n))];
19 for k=1:length(n)-2
20     y1(k+2)=-6*y1(k+1)-9*y1(k)+2*x(k+2)+6*x(k+1);
21 end
22 figure
23 a=gca();

```

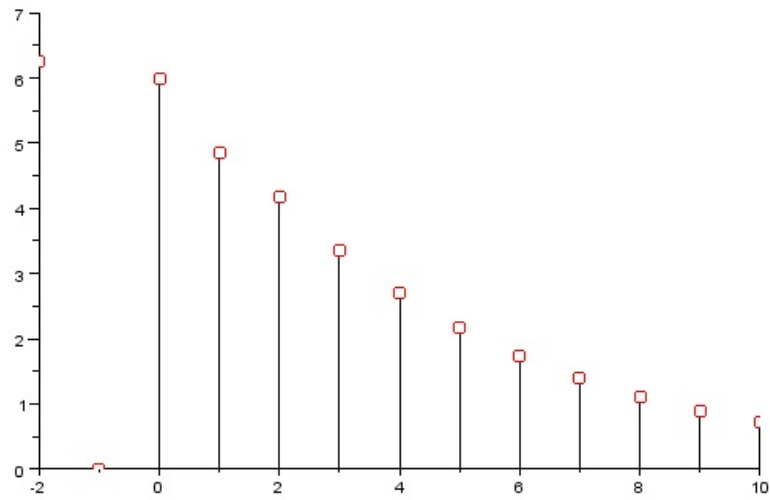


Figure 3.4: total response with given initial conditions

```

24 plot2d3(n, y1);
25
26
27 y2=[25/4;0; zeros(length(n)-2,1)];
28 x=[0;0; 4^-n(3:length(n))];
29 for k=1:length(n)-2
30     y2(k+2)=1.56*y2(k+1)-0.81*y2(k)+ x(k+1)+3*x(k);
31 end
32 figure
33 a=gca();
34 plot2d3(n, y2);

```

---



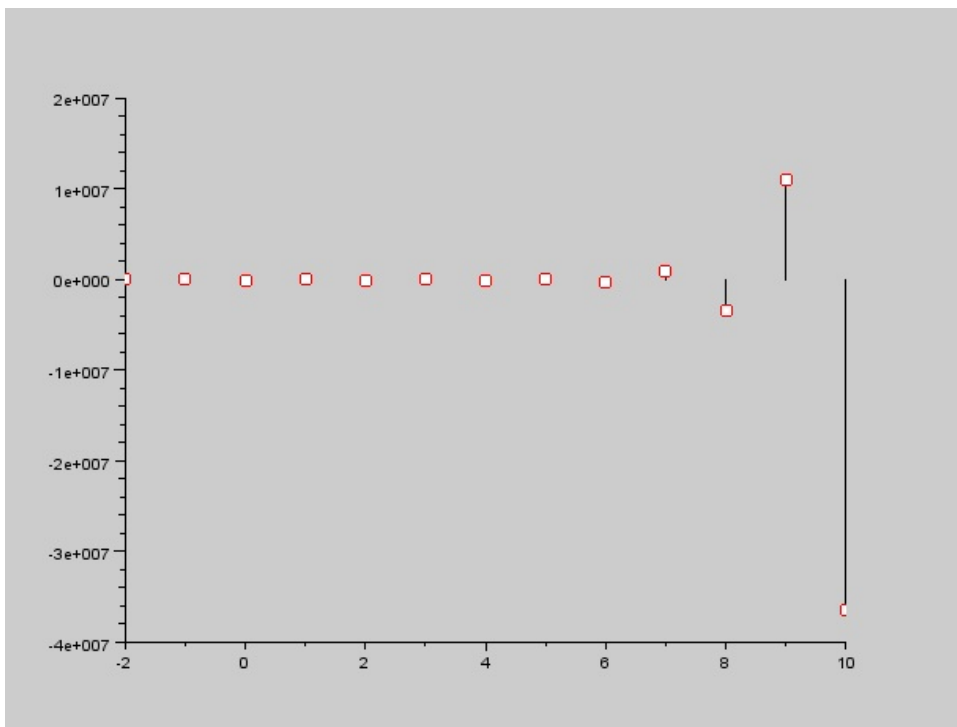


Figure 3.5: total response with given initial conditions

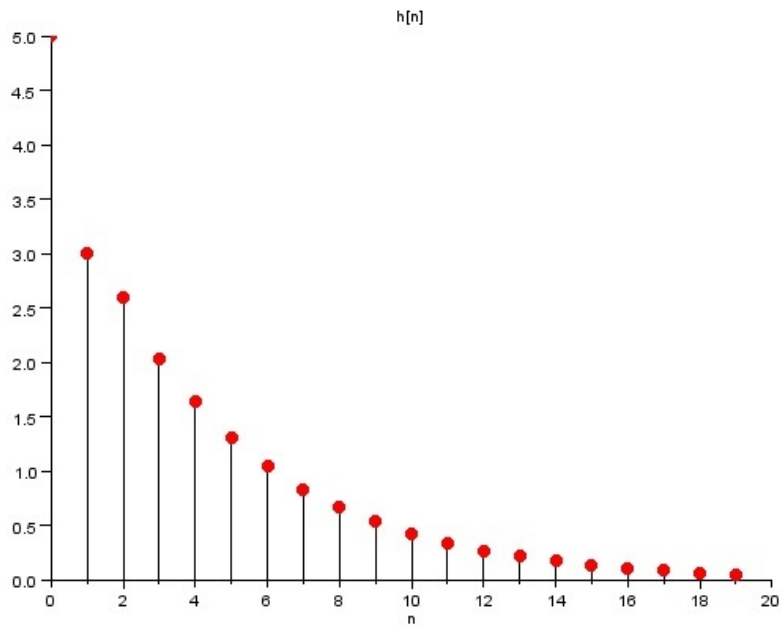


Figure 3.6: iterative determination of unit impulse response

**Scilab code Exa 3.11** iterative determination of unit impulse response

```
1 //signals and systems
2 //time domain analysis of discrete time systems
3 //impulse response with initial conditions
4 clear;
5 close;
6 clc;
7 n=(0:19);
8 x=[1 zeros(1,length(n)-1)];
9 a=[1 -0.6 -0.16];
10 b=[5 0 0];
11 h=filter(b,a,x);
12 clf;
13 plot2d3(n,h); xlabel('n'); ylabel('h[n]');
```

---

**Scilab code Exa 3.13** convolution of discrete signals

```
1 //signals and systems
2 //time domain analysis of discrete time systems
3 //convolution
4 clear;
5 close;
6 clc;
7 n=(0:19);
8 x=0.8^n;
9 g=0.3^n;
10 n1=(0:1:length(x)+length(g)-2);
11 c=convol(x,g);
12 plot2d3(n1,c);
```

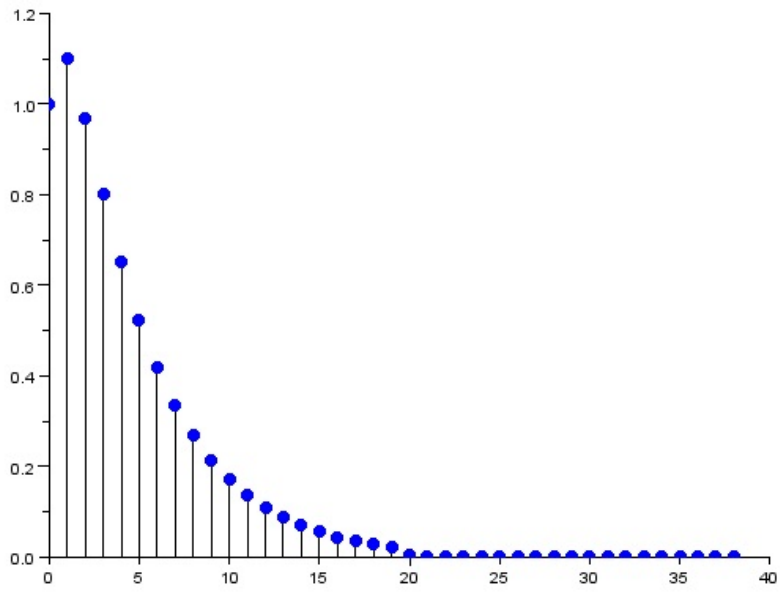


Figure 3.7: convolution of discrete signals

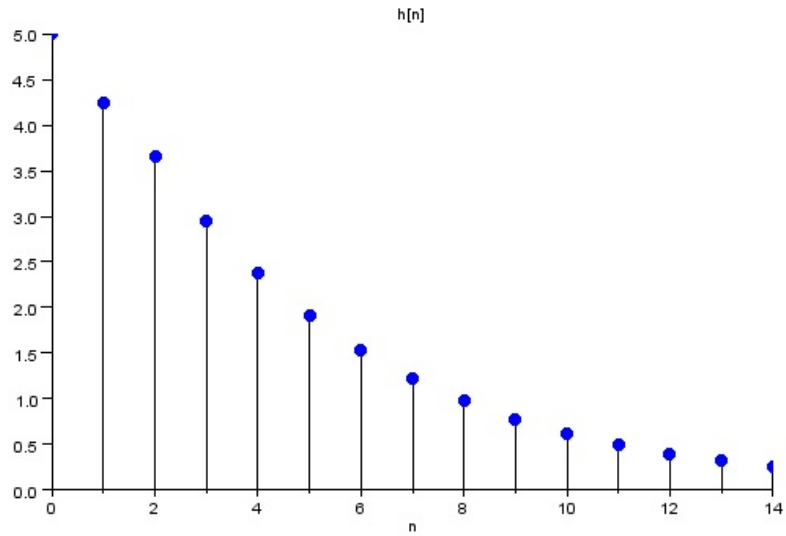


Figure 3.8: convolution of discrete signals

---

**Scilab code Exa 3.14** convolution of discrete signals

```

1 //signals and systems
2 //time domain analysis of discrete time systems
3 //convolution
4 clear;
5 close;
6 clc;
7 n=(0:14);
8 x=4^-n;
9 a=[1 -0.6 -0.16];
10 b=[5 0 0];
11 y=filter(b,a,x);

```

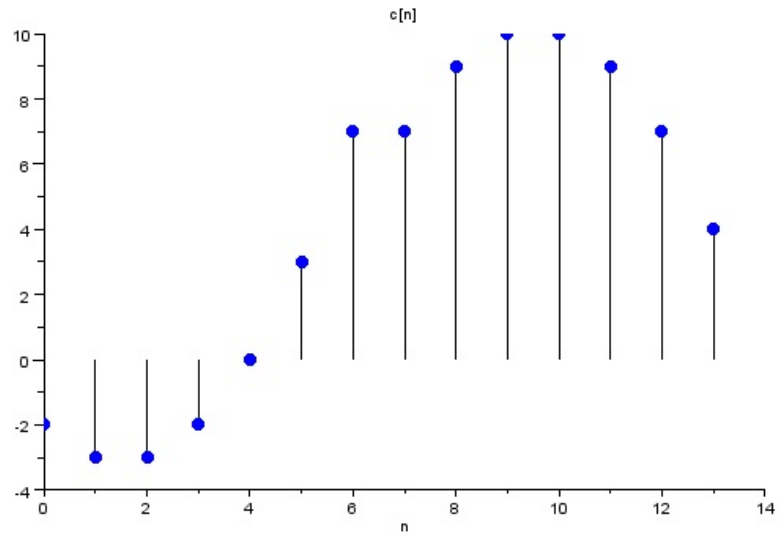


Figure 3.9: sliding tape method of convolution

```

12 clf;
13 plot2d3(n,y); xlabel('n'); ylabel('y[n]');

```

---

**Scilab code Exa 3.16** sliding tape method of convolution

```

1 //signals and systems
2 //time domain analysis of discrete time systems
3 //convolution by sliding tape method
4 clear;
5 close;
6 clc;
7 x=[-2 -1 0 1 2 3 4];
8 g=[1 1 1 1 1 1 1];
9 n=(0:1:length(x)+length(g)-2);

```

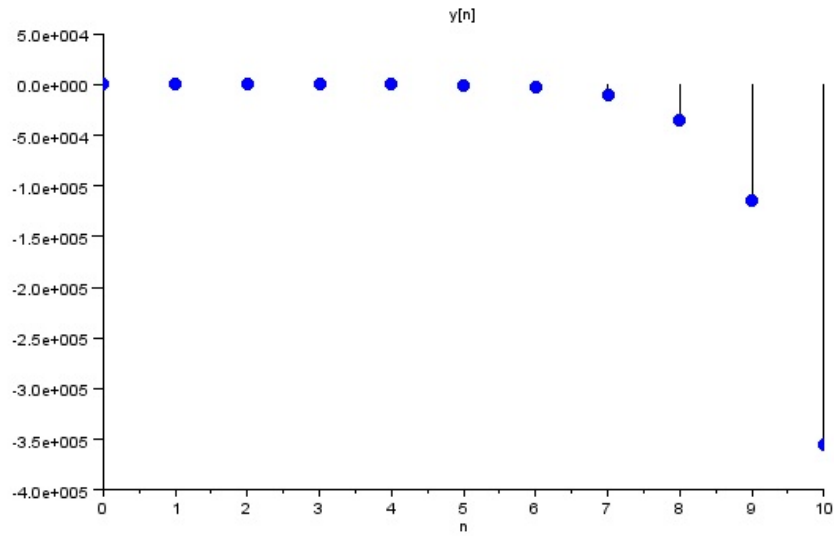


Figure 3.10: total response with given initial conditions

```

10 c=convol(x,g);
11 clf;
12 plot2d3(n,c); xlabel('n'); ylabel('c[n]');

```

---

**Scilab code Exa 3.17** total response with given initial conditions

```

1 //signals and systems
2 //time domain analysis of discrete time systems
3 //convolution by sliding tape method
4 clear;
5 close;
6 clc;
7 n=(0:10)';
8 y=[4;13;zeros(length(n)-2,1)];

```

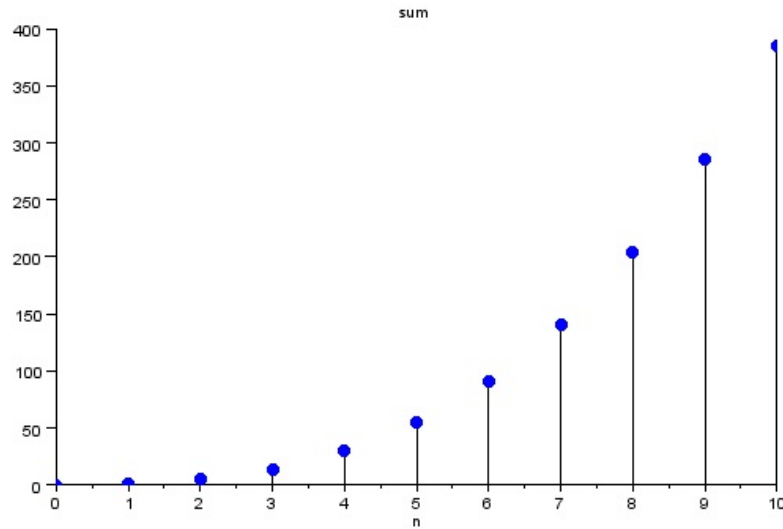


Figure 3.11: total response with given initial conditions

```

9 x=(3*n+5).*(n>=0);
10 for k=1:length(n)-2
11     y(k+2)=5*y(k+1)-6*y(k)+x(k+1)-5*x(k);
12 end
13 clf;
14 plot2d3(n,y); xlabel('n'); ylabel('y[n]');
15 disp('n          y');
16 disp(sprintf('%f\t\t%f\n',[n,y]));

```

---

**Scilab code Exa 3.18** total response with given initial conditions

```

1 //signals and systems
2 //time domain analysis of discreet time systems
3 //convolution by sliding tape method

```



```

4 clear;
5 close;
6 clc;
7 n=(0:10)';
8 y=[0;zeros(length(n)-1,1)];
9 x=(n+1)^2;
10 for k=1:length(n)-1
11     y(k+1)=y(k)+x(k);
12 end;
13 clf;
14 a=gca();
15 plot2d3(n,y);xtitle('sum','n')
16 plot(n,y,'b.')
```

---

### Scilab code Exa 3.19 forced response

```

1 //signals and systems
2 //time domain analysis of discrete time systems
3 //convolution by sliding tape method
4 clear;
5 close;
6 clc;
7 n=(0:14);
8 x=3^n;
9 a=[1 -3 2];
10 b=[0 1 2];
11 y=filter(b,a,x);
12 clf;
13 plot2d3(n,y); xlabel('n'); ylabel('y[n]');
```

---

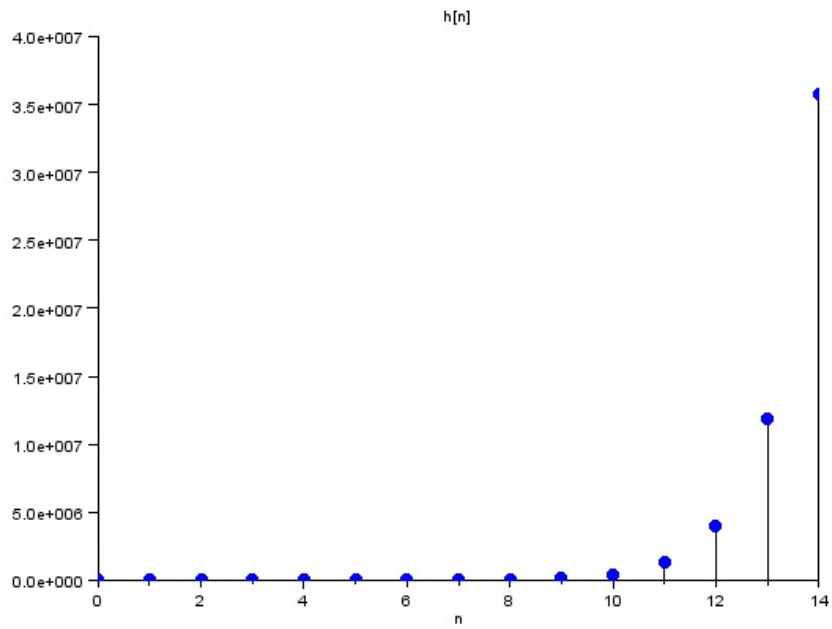


Figure 3.12: forced response

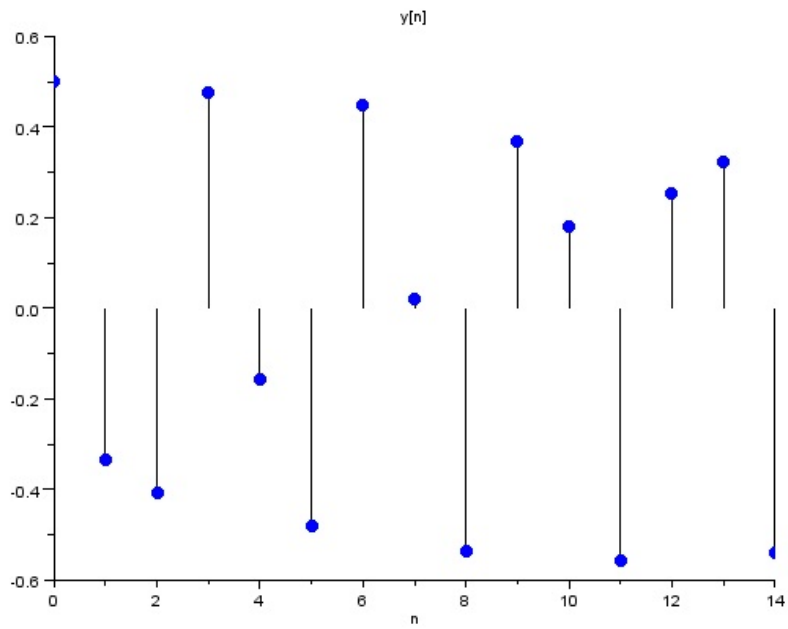


Figure 3.13: forced response

### Scilab code Exa 3.20 forced response

```
1 //signals and systems
2 //time domain analysis of discrete time systems
3 //convolution by sliding tape method
4 clear;
5 close;
6 clc;
7 pi=3.14;
8 n=(0:14);
9 x=cos(2*n*pi/3);
10 a=[1 -1 0.16];
11 b=[0 1 0.32];
12 y=filter(b,a,x);
13 clf;
14 plot2d3(n,y); xlabel('n'); ylabel('y[n]');
```

---

# Chapter 4

## continuous time system analysis

Scilab code Exa 4.1 laplace transform of exponential signal

```
1 //signals and systems
2 //Laplace Transform  $x(t) = \exp(-at) \cdot u(t)$  for t
   negative and positive
3 syms t s;
4 a = 3;
5 y =laplace('%e^(-a*t)',t,s);
6 t1=0:0.001:10;
7 plot2d(t1,exp(-a*t1));
8 disp(y)
9 y1 = laplace('%e^(a*-t)',t,s);
10 disp(y1)
```

---

Scilab code Exa 4.2 laplace transform of given fsignal

```
1 //signals and systems
2 //(a) laplace transform  $x(t) = \delta(t)$ 
3 syms t s;
```

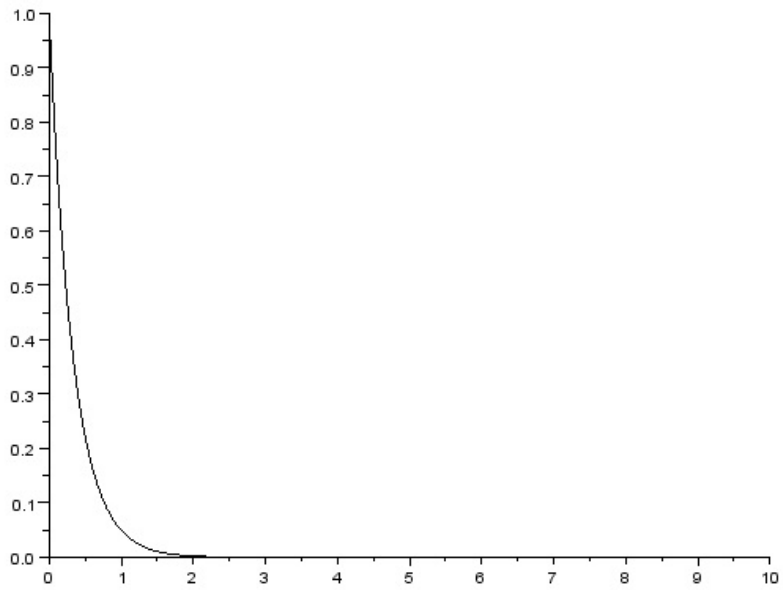


Figure 4.1: laplace transform of exponential signal

```

4
5 y =laplace('0',t,s)
6 disp(y)
7 //(b) Laplace Transform  $x(t) = u(t)$ 
8
9 y1 =laplace('1',t,s);
10 disp(y1)
11 //(c) laplace transform  $x(t) = \cos(w_0*t)u(t)$ 
12
13 y2 =laplace('cos(w0*t)',t,s);
14 disp(y2)

```

---

Scilab code Exa 4.3.a laplace transform in case of different roots

```

1 //signals and systems
2 //Inverse Lapalce Transform
3 //(a)  $X(S) = (7s-6)/s^2-s-6 \quad \text{Re}(s) > -1$ 
4 s =%s ;
5 syms t ;
6 [A]=pfs((7*s-6)/((s^2-s-6))); //partial fraction of
   F(s)
7 F1 = ilaplace(A(1),s,t)
8 F2 = ilaplace(A(2),s,t)
9 //F3 = ilaplace(A(3),s,t)
10 F = F1+F2;
11 disp(F,"f(t)=")

```

---

Scilab code Exa 4.3.b laplace transform in case of similar roots

```

1 //example 4.3
2 //(b)  $X(S) = (2*s^2+5)/s^2-3*s+2 \quad \text{Re}(s) > -1$ 
3 s =%s ;
4 syms t ;

```

```

5 [A]=pfs((2*s^2+5)/((s^2-3*s+2))); //partial
   fraction of F(s)
6 F1 = ilaplace(A(1),s,t)
7 F2 = ilaplace(A(2),s,t)
8 //F3 = ilaplace(A(3),s,t)
9 F = F1+F2;
10 disp(F,"f(t)=")

```

---

Scilab code Exa 4.3.c laplace transform in case of imaginary roots

```

1 //example4.3
2 //(c)  $X(S) = 6(s+34)/s(s^2+10*s+34)$   $\text{Re}(s)>-1$ 
3 s =%s ;
4 syms t ;
5 [A]=pfs((6*(s+34))/(s*(s^2+10*s+34))); //partial
   fraction of F(s)
6 F1 = ilaplace(A(1),s,t)
7 F2 = ilaplace(A(2),s,t)
8 //F3 = ilaplace(A(3),s,t)
9 F = F1+F2;
10 disp(F,"f(t)=")

```

---

Scilab code Exa 4.4 laplace transform of a given signal

```

1 //signals and systems
2 //Lapalce Transform  $x(t) = (t-1)u(t-1)-(t-2)u(t-2)-u$ 
    $(t-4), 0<t<T$ 
3 syms t s;
4 a = 3;
5 T = 1;
6 //t = T;
7 y1 = laplace('t',t,s);
8 y2 = laplace('t',t,s);

```



```

9 y3 = laplace('1',t,s);
10 y=y1*(%e^(-s))+y2*(%e^(-2*s))+y3*(%e^(-4*s))
11 disp(y)

```

---

#### Scilab code Exa 4.5 inverse laplace transform

```

1 //signals and systems
2 //example4.5
3 // X(S) = s+3+5*exp(-2*s)/(s+1)*(s+2)) Re(s)>-1
4 s1 =%s ;
5 syms t s;
6 [A]=pfss((s1+3)/((s1+1)*(s1+2))); //partial fraction
    of F(s)
7 F1 = ilaplace(A(1),s,t)
8 F2 = ilaplace(A(2),s,t)
9 //F3 = ilaplace(A(3),s,t)
10 Fa = F1+F2;
11 disp(Fa,"f1(t)=")
12 [B]=pfss((5)/((s1+1)*(s1+2))); //partial fraction of
    F(s)
13 F1 = ilaplace(B(1),s,t)
14 F2 = ilaplace(B(2),s,t)
15 Fb = (F1+F2)*(%e^(-2*s));
16 disp(Fb,"f2(t)=")
17 disp(Fa+Fb,"f(t)=")

```

---

#### Scilab code Exa 4.8 time convolution property

```

1 //signals and systems
2 //Example 4.8
3 //Lapalce Transform for convolution
4 s=%s
5 syms t ;

```

```

6 a=3;b=2;
7 [A]=pfss(1/(s^2-5*s+6)); //partial fraction of F(s)
8 F1 = ilaplace(A(1),s,t)
9 F2 = ilaplace(A(2),s,t)
10 //F3 = ilaplace(A(3),s,t)
11 F = F1+F2;
12 disp(F,"f(t)=")

```

---

#### Scilab code Exa 4.9 initial and final value

```

1 //Initial and final Value Theorem of Lapalace
  Transform
2 syms s;
3 num =poly([30 20], 's', 'coeff')
4 den =poly([0 5 2 1], 's', 'coeff')
5 X = num/den
6 disp (X,"X(s)=")
7 SX = s*X;
8 Initial_Value =limit(SX,s,%inf);
9 final_value =limit(SX,s,0);
10 disp(Initial_Value,"x(0)=")
11 disp(final_value,"x(inf)=")

```

---

#### Scilab code Exa 4.10 second order linear differential equation

```

1 //signals and systems
2 //Unilateral Laplace Transform:Solving Differential
  Equation
3 //example 4.10
4 s = %s;
5 syms t;
6 [A] = pfss((2*s^2+20*s+45)/((s+2)*(s+3)*(s+4)));
7 F1 = ilaplace(A(1),s,t)

```

```
8 F2 = ilaplace(A(2),s,t)
9 F3 = ilaplace(A(3),s,t)
10 F = F1+F2+F3
11 disp(F)
```

---

Scilab code Exa 4.11 solution to ode using laplace transform

```
1 //signals and systems
2 //Unilateral Laplace Transform:Solving Differential
   Equation
3 //example 4.11
4 s = %s;
5 syms t;
6 [A] = pfs((2*s)/(s^2+2*s+5));
7 F1 = ilaplace(A(1),s,t)
8 //F2 = ilaplace(A(2),s,t)
9 //F3 = ilaplace(A(3),s,t)
10 F = F1+F2+F3
11 disp(F)
```

---

Scilab code Exa 4.12 response to LTIC system

```
1 //signals and systems
2 //Unilateral Laplace Transform:Solving Differential
   Equation
3 //example 4.12
4 s = %s;
5 syms t;
6 [A] = pfs((3*s+3)/((s+5)*(s^2+5*s+6)));
7 F1 = ilaplace(A(1),s,t)
8 F2 = ilaplace(A(2),s,t)
9 F3 = ilaplace(A(3),s,t)
10 F = F1+F2+F3
```

```
11 disp(F)
```

---

Scilab code Exa 4.15 loop current in a given network

```
1 //signals and systems
2 //Unilateral Laplace Transform:Solving Differential
  Equation
3 //example 4.15
4 s = %s;
5 syms t;
6 [A] = pfss((10)/(s^2+3*s+2));
7 F1 = ilaplace(A(1),s,t)
8 F2 = ilaplace(A(2),s,t)
9 //F3 = ilaplace(A(3),s,t)
10 F = F1+F2+F3
11 disp(F)
```

---

Scilab code Exa 4.16 loop current in a given network

```
1 //signals and systems
2 //Unilateral Laplace Transform: transfer function
3 //example 4.16
4 s = %s;
5 syms t s;
6 y1 =laplace('24*%e^(-3*t)+48*%e^(-4*t)',t,s);
7 disp(y1)
8 y2 =laplace('16*%e^(-3*t)-12*%e^(-4*t)',t,s);
9 disp(y2)
```

---

Scilab code Exa 4.17 voltage and current of a given network

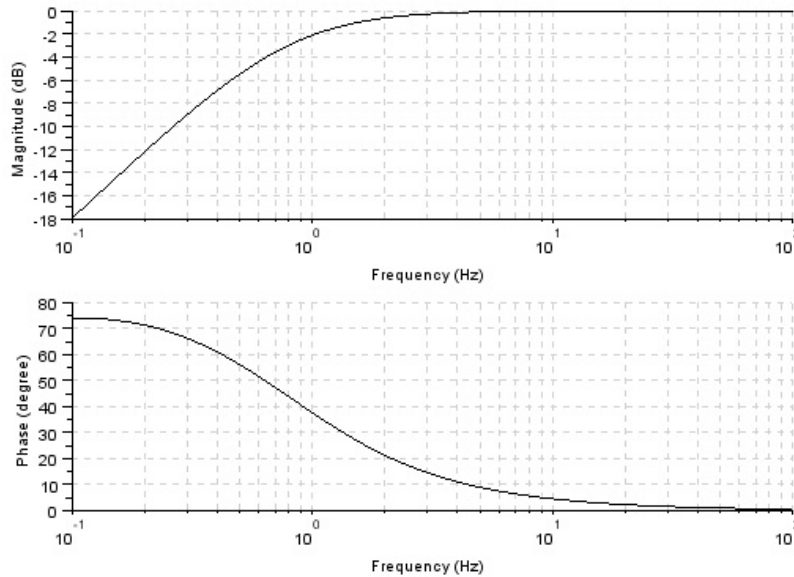


Figure 4.2: frequency response of a given system

```

1 //signals and systems
2 //Unilateral Laplace Transform:Solving Differential
  Equation
3 //example 4.17
4 s= %s;
5 syms t;
6 [A] = pfss(((2*s^2+9*s+4)/((s)*(s^2+3*s+1))));
7 F1 = ilaplace(A(1),s,t)
8 F2 = ilaplace(A(2),s,t)
9 F3 = ilaplace(A(3),s,t)
10 F = F1+F2+F3
11 disp(F)

```

---

Scilab code Exa 4.23 frequency response of a given system

```
1 s=poly(0, 's')
2 h=syslin('c', (s+0.1)/(s+5))
3 clf(); bode(h, 0.1, 100);
```

---

Scilab code Exa 4.24 frequency response of a given system

```
1 s=poly(0, 's')
2 h=syslin('c', (s^2/s))
3 clf(); bode(h, 0.1, 100);
4 h1=syslin('c', (1/s))
5 clf(); bode(h1, 0.1, 100);
```

---

Scilab code Exa 4.25 bode plots for given transfer function

```
1 s=poly(0, 's')
2 h=syslin('c', ((20*s^2+2000*s)/(s^2+12*s+20)))
3 clf(); bode(h, 0.1, 100);
```

---

Scilab code Exa 4.26 bode plots for given transfer function

```
1 s=poly(0, 's')
2 h=syslin('c', ((10*s+1000)/(s^2+2*s+100)))
3 clf(); bode(h, 0.1, 100);
```

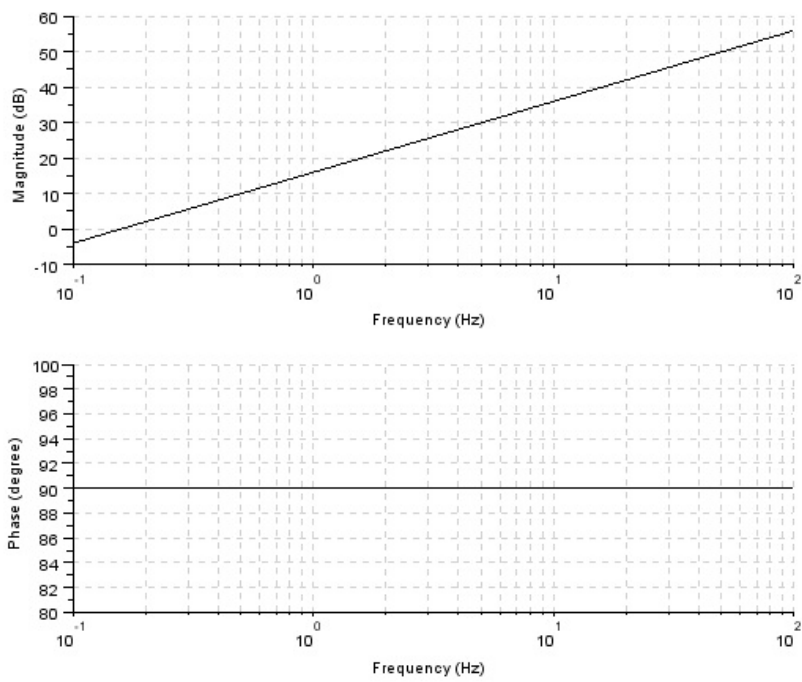


Figure 4.3: frequency response of a given system

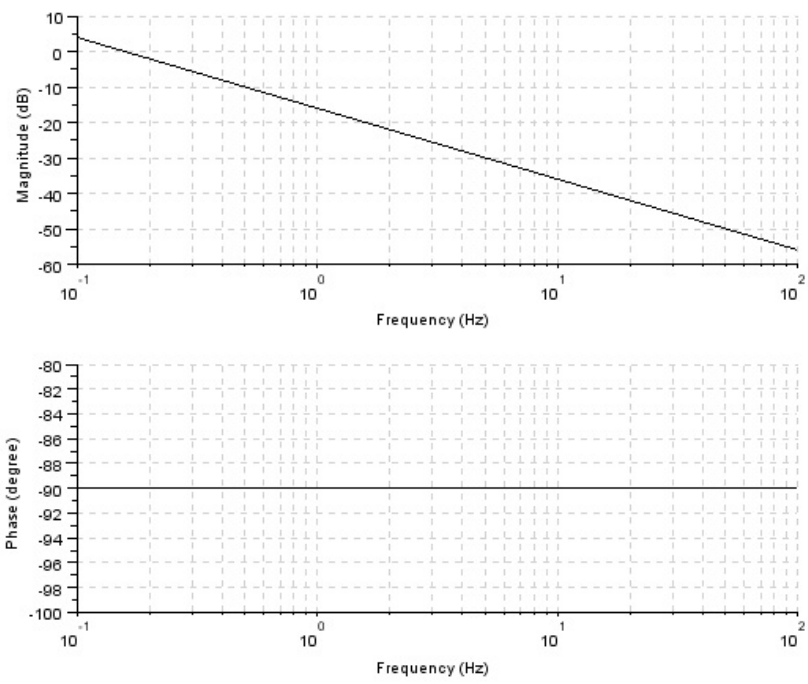


Figure 4.4: frequency response of a given system



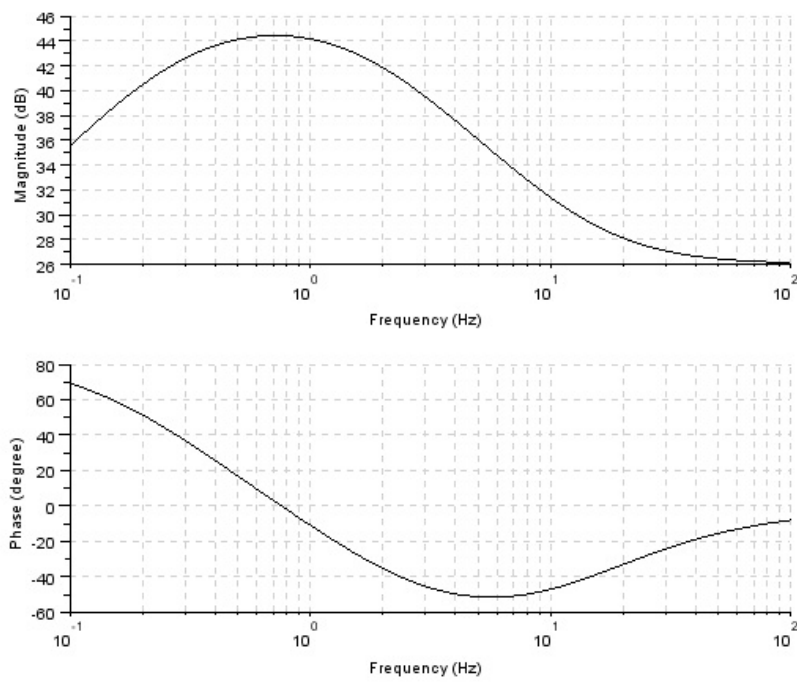


Figure 4.5: bode plots for given transfer function

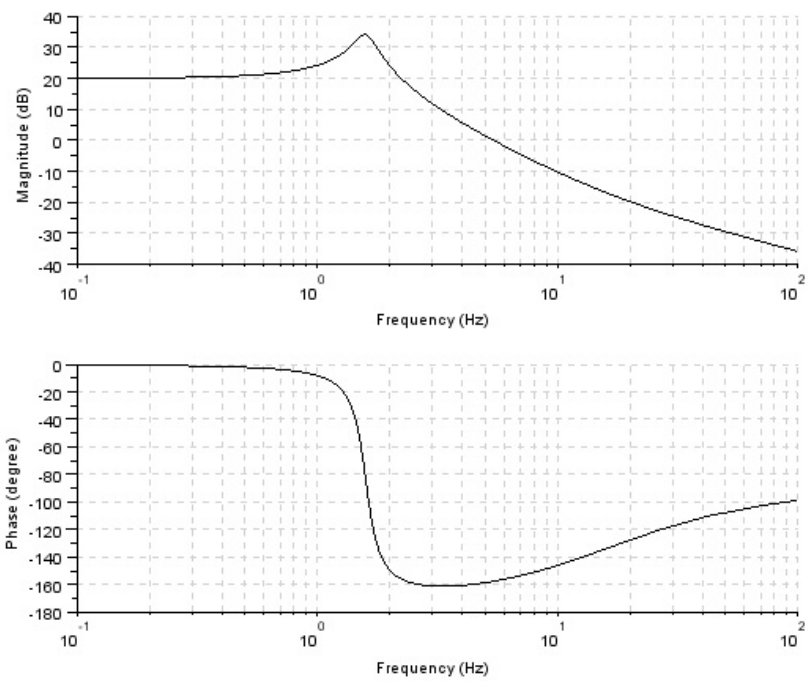


Figure 4.6: bode plots for given transfer function

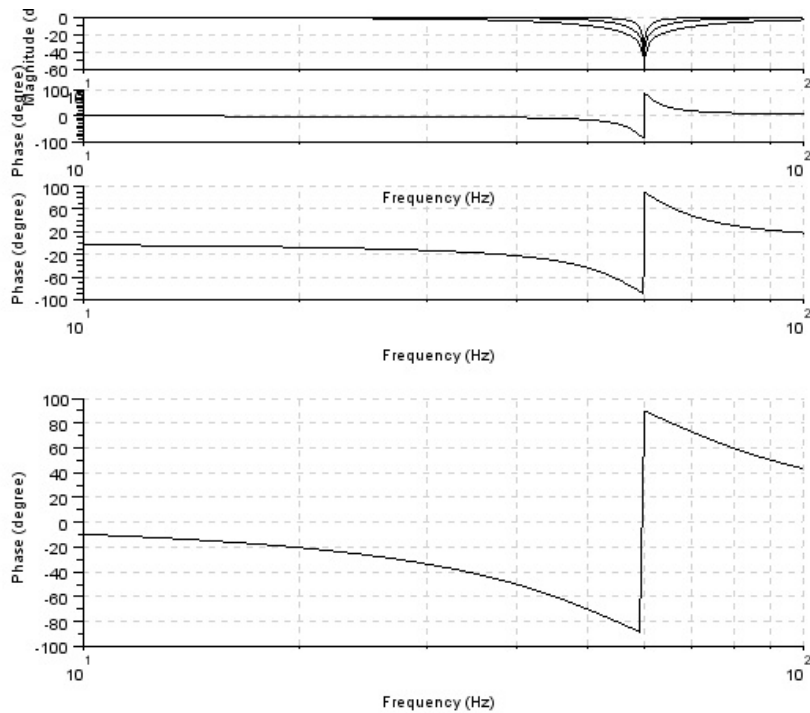


Figure 4.7: second order notch filter to suppress 60Hz hum

---

Scilab code Exa 4.27 second order notch filter to suppress 60Hz hum

```

1 omega_0=2*%pi*60; theta = [60 80 87]*(%pi/180);
2 omega = (0:0.5:1000)'; mag = zeros(3,length(omega));
3 s=poly(0,'s')
4 for m =1:length(theta)
5     H=syslin('c',((s^2+omega_0^2)/(s^2+2*omega_0*cos
6         (theta(m))*s +omega_0^2)));
7     bode(H,10,100);
7 end

```

```

8 f=omega/((2*pi));
9 plot(f,mag(1,:), 'k-', f,mag(2,:), 'k—', f,mag(3,:), '
    'k-.'');
10 xlabel('f [hz]'); ylabel('|H(j2/pi f)|');
11 legend('\theta=60^\circ', '\theta = 80^\circ', '\theta
    = 87^\circ', 1)

```

---

#### Scilab code Exa 4.28 bilateral inverse transform

```

1 //signals and systems
2 //bilateral Inverse Lapalce Transform
3 //X(S) = 1/((s-1)(s+2))
4 s =%s ;
5 syms t ;
6 [A]=pfss(1/((s-1)*(s+2))) //partial fraction of F(s)
7 F1 = ilaplace(A(1),s,t)
8 F2 = ilaplace(A(2),s,t)
9 F=F1+F2;
10 disp(F,"f(t)=")
11
12
13 //X(S) = 1/((s-1)(s+2)) Re(s)> -1,Re(s)< -2
14 s =%s ;
15 syms t ;
16 [A]=pfss(1/((s-1)*(s+2))) //partial fraction of F(s)
17 F1 = ilaplace(A(1),s,t)
18 F2 = ilaplace(A(2),s,t)
19 F = -F1-F2;
20 disp(F,"f(t)=")
21
22
23 //X(S) = 1/((s-1)(s+2)) -2< Re(s)< 1
24 s =%s ;
25 syms t ;
26 [A]=pfss(1/((s-1)*(s+2))) //partial fraction of F(s)

```

```

27 F1 = ilaplace(A(1),s,t)
28 F2 = ilaplace(A(2),s,t)
29 F = -F1+F2;
30 disp(F,"f(t)=")

```

---

Scilab code Exa 4.29 current for a given RC network

```

1 //signals and systems
2 //Unilateral Laplace Transform:Solving Differential
   Equation
3 //example 4.30
4 s= %s;
5 syms t;
6 [A] = pfss((-s)/((s-1)*(s-2)*(s+1)));
7 F1 = ilaplace(A(1),s,t)
8 F2 = ilaplace(A(2),s,t)
9 F3 = ilaplace(A(3),s,t)
10 F = F1+F2+F3
11 disp(F)

```

---

Scilab code Exa 4.30 response of a noncausal sytem

```

1 //signals and systems
2 //Unilateral Laplace Transform:Solving Differential
   Equation
3 //example 4.30
4 s= %s;
5 syms t;
6 [A] = pfss((-1)/((s-1)*(s+2)));
7 F1 = ilaplace(A(1),s,t)
8 F2 = ilaplace(A(2),s,t)
9 //F3 = ilaplace(A(3),s,t)
10 F = F1+F2

```

11 `disp(F)`

---

Scilab code Exa 4.31 response of a fn with given tf

```
1 //signals and systems
2 //Unilateral Laplace Transform:Solving Differential
   Equation
3 //example 4.17
4 s= %s;
5 syms t;
6 // Re s>-1
7 [A] = pfs(1/((s+1)*(s+5)));
8 F1 = ilaplace(A(1),s,t)
9 F2 = ilaplace(A(2),s,t)
10 //F3 = ilaplace(A(3),s,t)
11 F = F1+F2
12 disp(F)
13 //-5< Re s <-2
14 [B] = pfs(-1/((s+2)*(s+5)));
15 G1 = ilaplace(B(1),s,t)
16 G2 = ilaplace(B(2),s,t)
17 //F3 = ilaplace(A(3),s,t)
18 G = G1+G2
19 disp(G)
```

---

# Chapter 5

## discrete time system analysis using the z transform

Scilab code Exa 5.1 z transform of a given signal

```
1 //signals and systems
2 // Ztransform of  $x[n] = (a)^n \cdot u[n]$ 
3 syms n z;
4 a = 0.5;
5 x =(a)^n;
6 n1=0:10;
7 plot2d3(n1,a^n1); xtitle('a^n', 'n');
8 plot(n1,a^n1, 'r. ')
9 X = symsum(x*(z^(-n)),n,0,%inf)
10 disp(X,"ans=")
```

---

Scilab code Exa 5.2 z transform of a given signal

```
1 //example 5.2 (c)
```

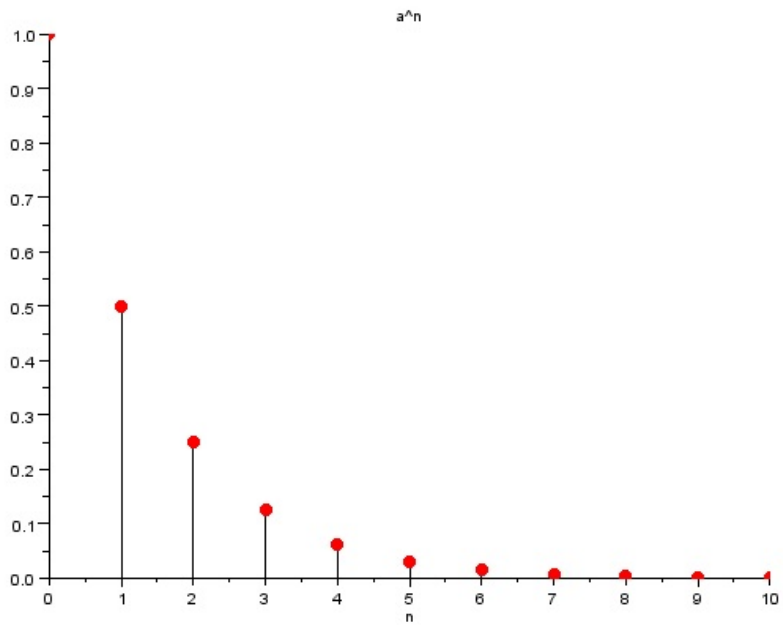


Figure 5.1: z transform of a given signal



```

2 //Z-transform of sine signal
3 syms n z;
4 Wo =%pi/4;
5 a = (0.33)^n;
6 x1=%e^(sqrt(-1)*Wo*n);
7 X1=symsum(a*x1*(z^(-n)),n,0,%inf)
8 x2=%e^(-sqrt(-1)*Wo*n)
9 X2=symsum(a*x2*(z^(-n)),n,0,%inf)
10 X =(1/(2*sqrt(-1)))*(X1+X2)
11 disp(X,"ans=")
12
13 //example 5.2 (a)
14 //Z-transform of Impulse Sequence
15 syms n z;
16 X=symsum(1*(z^(-n)),n,0,0);
17 disp(X,"ans=")
18
19 //example 5.2 (d)
20 //Z-transform of given Sequence
21 syms n z;
22 X=symsum(1*(z^(-n)),n,0,4);
23 disp(X,"ans=")
24
25 //example 5.2 (b)
26 //Z-transform of unit function Sequence
27 syms n z;
28 X=symsum(1*(z^(-n)),n,0,%inf);
29 disp(X,"ans=")

```

---

**Scilab code Exa 5.3.a** z transform of a given signal with different roots

```

1 //signals and systems
2 //Inverse Z Transform:ROC |z|>1/3
3 z = %z;
4 syms n z1;//To find out Inverse z transform z must

```

```

    be linear z = z1
5 X  =(8*z-19)/((z-2)*(z-3))
6 X1 = denom(X);
7 zp = roots(X1);
8 X1 = (8*z1-19)/((z1-2)*(z1-3))
9 F1 = X1*(z1^(n-1))*(z1-zp(1));
10 F2 = X1*(z1^(n-1))*(z1-zp(2));
11 h1 = limit(F1,z1,zp(1));
12 disp(h1,'h1[n]=')
13 h2 = limit(F2,z1,zp(2));
14 disp(h2,'h2[n]=')
15 h = h1+h2;
16 disp(h,'h[n]=')

```

---

Scilab code Exa 5.3.c z transform of a given signal with imaginary roots

```

1 //signals and systems
2 //Inverse Z Transform:ROC |z|>1/3
3 z = %z;
4 syms n z1;//To find out Inverse z transform z must
    be linear z = z1
5 X  =(2*z*(3*z+17))/((z-1)*(z^2-6*z+25))
6 X1 = denom(X);
7 zp = roots(X1);
8 X1 = 2*z1*(3*z1+17)/((z1-1)*(z1^2-6*z1+25))
9 F1 = X1*(z1^(n-1))*(z1-zp(1));
10 F2 = X1*(z1^(n-1))*(z1-zp(2));
11 h1 = limit(F1,z1,zp(1));
12 disp(h1,'h1[n]=')
13 h2 = limit(F2,z1,zp(2));
14 disp(h2,'h2[n]=')
15 h = h1+h2;
16 disp(h,'h[n]=')

```

---

**Scilab code Exa 5.5** solution to differential equation

```
1 //LTi Systems characterized by Linear Constant
2 //Coefficient Difference equations
3 //Inverse Z Transform
4 //z = %z;
5 syms n z;
6 H1 = (26/15)/(z-(1/2));
7 H2 = (7/3)/(z-2);
8 H3 = (18/5)/(z-3);
9 F1 = H1*z^(n)*(z-(1/2));
10 F2 = H2*z^(n)*(z-2);
11 F3 = H3*z^(n)*(z-3);
12 h1 = limit(F1,z,1/2);
13 disp(h1,'h1[n]=')
14 h2 = limit(F2,z,2);
15 disp(h2,'h2[n]=')
16 h3 = limit(F3,z,3);
17 disp(h3,'h3[n]=')
18 h = h1-h2+h3;
19 disp(h,'h[n]=')
```

---

**Scilab code Exa 5.6** response of an LTID system using difference eq

```
1 //LTi Systems characterized by Linear Constant
2 //Coefficient Difference equations
3 //Inverse Z Transform
4 //z = %z;
5 syms n z;
6 H1 = (2/3)/(z+0.2);
7 H2 = (8/3)/(z+0.8);
8 H3 = (2)/(z+0.5);
```

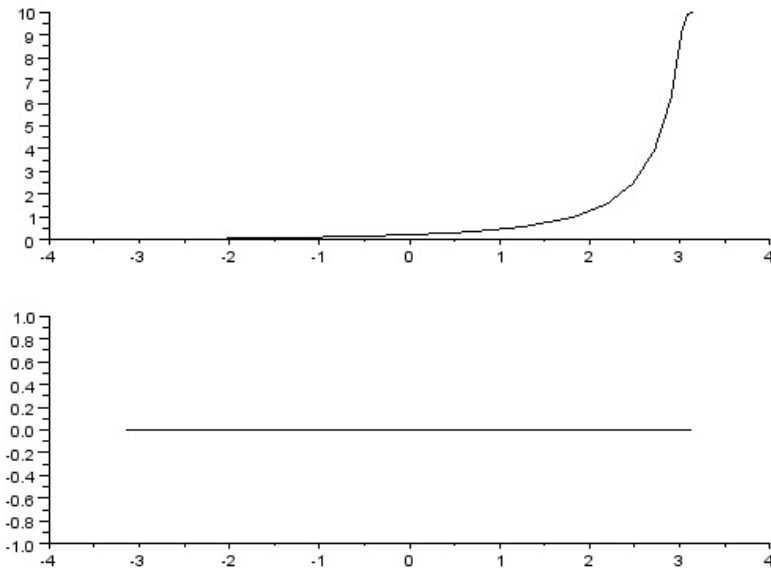


Figure 5.2: response of an LTID system using difference eq

```

9 F1 = H1*z^(n)*(z+0.2);
10 F2 = H2*z^(n)*(z+0.8);
11 F3 = H3*z^(n)*(z+0.5);
12 h1 = limit(F1,z,-0.2);
13 disp(h1,'h1[n]=')
14 h2 = limit(F2,z,-0.8);
15 disp(h2,'h2[n]=')
16 h3 = limit(F3,z,-0.5);
17 disp(h3,'h3[n]=')
18 h = h1-h2+h3;
19 disp(h,'h[n]=')

```

---

Scilab code Exa 5.10 response of an LTID system using difference eq

```

1 omega= linspace(-%pi,%pi,106);
2 s = %s;
3 H= syslin('c',(s/(s-0.8)));
4 H_omega= squeeze(calfrq(H,0.01,10));
5 size(H_omega)
6 subplot(2,1,1); plot2d(omega, abs(H_omega));
7 //xlabel('\omega');
8 //ylabel('|H[e^{j\omega}]|');
9 subplot(2,1,2); plot2d(omega, atan(imag(H_omega),real
    (H_omega))*180/%pi);
10 //xlabel('\omega');
11 //ylabel('\angle H[e^{j \omega}] [deg]');

```

---

Scilab code Exa 5.12 maximum sampling timeinterval

```

1 //signals and systems
2 //maximum sampling interval
3 f=50*10^3;
4 T=0.5/f;
5 disp(T)//in seconds

```

---

Scilab code Exa 5.13 discrete time amplifier highest frequency

```

1 //signals and systems
2 //highest frequency of a signal
3 T=25*10^-6
4 f=0.5/T
5 disp(f)//in hertz

```

---

Scilab code Exa 5.17 bilateral z transform

```

1 //Z transform of  $x[n] = a^n \cdot u[n] + b^{-n} \cdot u[-n-1]$ 
2 syms n z;
3 a=0.9
4 b = 1.2;
5
6 x1=(a)^(n)
7 x2=(b)^(-n)
8 //plot2d3(n1,x1+x2)
9 X1=symsum(x1*(z^(-n)),n,0,%inf)
10 X2=symsum(x2*(z^(n)),n,1,%inf)
11 X = X1+X2;
12 disp(X,"ans=")

```

---

#### Scilab code Exa 5.18 bilateral inverse z transform

```

1 //signals and systems
2 //Inverse Z Transform:ROC  $|z|>2$ 
3 z = %z;
4 syms n z1;//To find out Inverse z transform z must
   be linear z = z1
5 X = -z*(z+0.4)/((z-0.8)*(z-2))
6 X1 = denom(X);
7 zp = roots(X1);
8 X1 = -z1*(z1+0.4)/((z1-0.8)*(z1-2))
9 F1 = X1*(z1^(n-1))*(z1-zp(1));
10 F2 = X1*(z1^(n-1))*(z1-zp(2));
11 h1 = limit(F1,z1,zp(1));
12 disp(h1,'h1[n]=')
13 h2 = limit(F2,z1,zp(2));
14 disp(h2,'h2[n]=')
15 h = h1+h2;
16 disp(h,'h[n]=')
17
18 //Inverse Z Transform:ROC  $0.8<|z|<2$ 
19 z = %z;

```

```

20 syms n z1;
21 X = -z*(z+0.4)/((z-0.8)*(z-2))
22 X1 = denom(X);
23 zp = roots(X1);
24 X1 = -z1*(z1+0.4)/((z1-0.8)*(z1-2))
25 F1 = X1*(z1^(n-1))*(z1-zp(1));
26 F2 = X1*(z1^(n-1))*(z1-zp(2));
27 h1 = limit(F1,z1,zp(1));
28 disp(h1*'u(n)', 'h1[n]=')
29 h2 = limit(F2,z1,zp(2));
30 disp((h2)*'u(-n-1)', 'h2[n]=')
31 disp((h1)*'u(n)'-(h2)*'u(n-1)', 'h[n]=')
32
33 //Inverse Z Transform:ROC |z|<0.8
34 z = %z;
35 syms n z1;
36 X = -z*(z+0.4)/((z-0.8)*(z-2))
37 X1 = denom(X);
38 zp = roots(X1);
39 X1 = -z1*(z1+0.4)/((z1-0.8)*(z1-2))
40 F1 = X1*(z1^(n-1))*(z1-zp(1));
41 F2 = X1*(z1^(n-1))*(z1-zp(2));
42 h1 = limit(F1,z1,zp(1));
43 disp(h1*'u(-n-1)', 'h1[n]=')
44 h2 = limit(F2,z1,zp(2));
45 disp((h2)*'u(-n-1)', 'h2[n]=')
46 disp(-(h1)*'u(-n-1)'-(h2)*'u(-n-1)', 'h[n]=')

```

---

**Scilab code Exa 5.19** transfer function for a causal system

```

1 //LTi Systems characterized by Linear Constant
2 //Coefficient Difference equations
3 //Inverse Z Transform
4 //z = %z;
5 syms n z;

```

```

6 H1 = -z/(z-0.5);
7 H2 = (8/3)*z/(z-0.8);
8 H3=(-8/3)*z/(z-2);
9 F1 = H1*z^(n-1)*(z-0.5);
10 F2 = H2*z^(n-1)*(z-0.8);
11 F3 = H3*z^(n-1)*(z-2);
12 h1 = limit(F1,z,0.5);
13 disp(h1,'h1[n]=')
14 h2 = limit(F2,z,0.8);
15 disp(h2,'h2[n]=')
16 h3 = limit(F3,z,2);
17 disp(h3,'h3[n]=')
18 h = h1+h2+h3;
19 disp(h,'h[n]=')

```

---

**Scilab code Exa 5.20** zero state response for a given input

```

1 //LTi Systems characterized by Linear Constant
2 //Coefficient Difference equations
3 //Inverse Z Transform
4 //z = %z;
5 syms n z;
6 H1 = (-5/3)*z/(z-0.5);
7 H2 = (8/3)*z/(z-0.8);
8 H3=5*z/(z-0.5);
9 H4=-6*z/(z-0.6);
10 F1 = H1*z^(n-1)*(z-0.5);
11 F2 = H2*z^(n-1)*(z-0.8);
12 F3 = H3*z^(n-1)*(z-0.5);
13 F4 = H4*z^(n-1)*(z-0.6);
14 h1 = limit(F1,z,0.5);
15 disp(h1,'h1[n]=')
16 h2 = limit(F2,z,0.8);
17 disp(h2,'h2[n]=')
18 h3 = limit(F3,z,0.5);

```



```
19 disp(h3, 'h3[n]= ')
20 h4 = limit(F4,z,0.6);
21 disp(h4, 'h4[n]= ')
22 h = h1+h2+h3+h4;
23 disp(h, 'h[n]= ')
```

---

# Chapter 6

## continuous time signal analysis the fourier series

Scilab code Exa 6.1 fourier coefficients of a periodic sequence

```
1 n=0:10;
2 a_n=0.504*2*ones(1,length(n))./(1+16*n.^2);
3 a_n(1)=0.504
4 b_n=0.504*8*n./(1+16*n.*n);
5 size(n)
6 size(a_n)
7 size(b_n)
8 disp(b_n(1))
9 C_n=sqrt(a_n.^2+(b_n).^2);
10 theta_n(1)=0; theta_n=atan(-b_n,a_n);
11 //n=[0,n];
12 clf;
13 size(n)
14 subplot(2,2,1); plot2d3(n,a_n);xtitle('a_n','n');
    plot(n,a_n,'ro');
15 subplot(2,2,2); plot2d3(n,b_n);xtitle('b_n','n');
    plot(n,b_n,'r.');
```

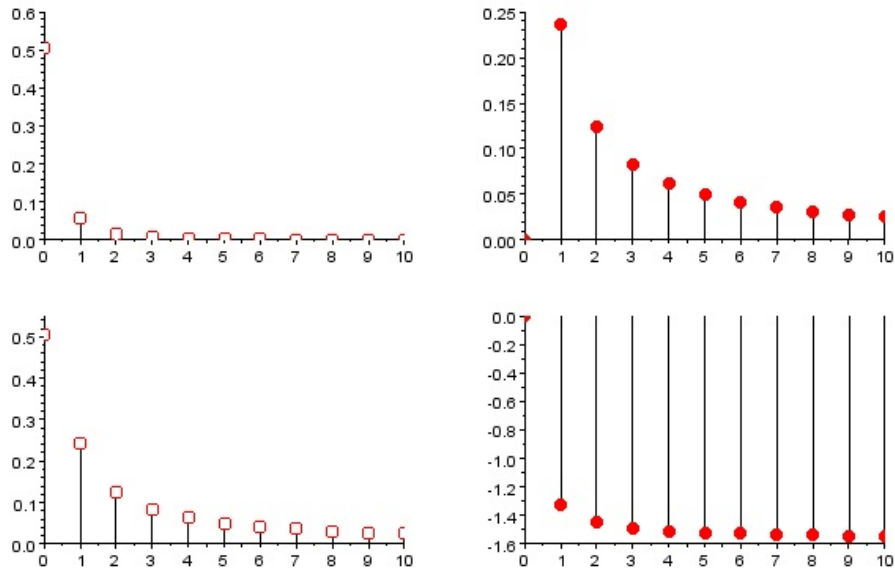


Figure 6.1: fourier coefficients of a periodic sequence

```

16 subplot(2,2,3); plot2d3(n,C_n);xtitle('C_n','n');
    plot(n,C_n,'ro');
17 subplot(2,2,4); plot2d3(n,theta_n,);xtitle('theta_n'
    ,'n');plot(n,theta_n,'r.')
```

---

Scilab code Exa 6.2 fourier coefficients of a periodic sequence

```

1 n=0:10;
2 a_n=zeros(1,length(n));
3 size(a_n)
4 b_n=(8/%pi^2*n.^2).*sin(n.*%pi/2);
5 size(n)
6 size(a_n)
7 size(b_n)
```

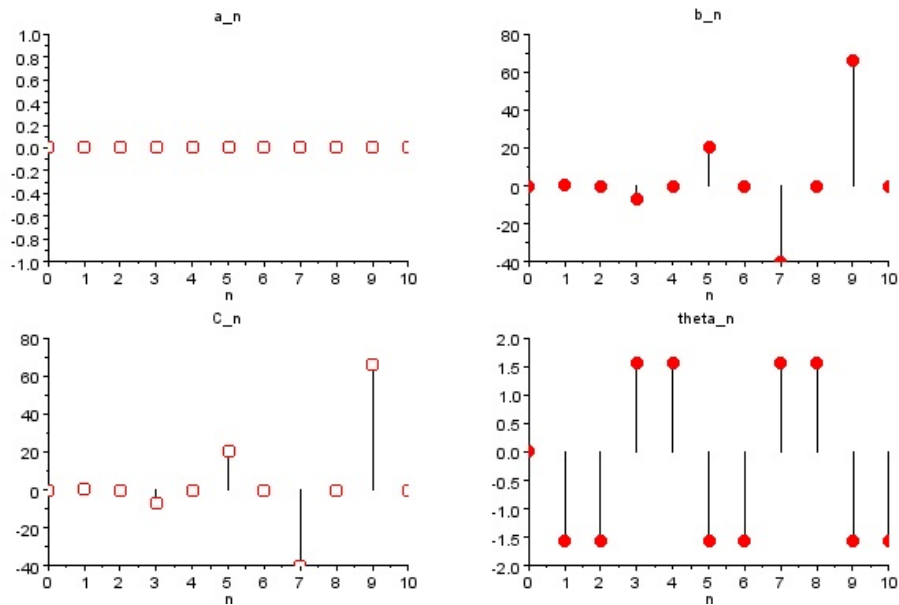


Figure 6.2: fourier coefficients of a periodic sequence

```

8 disp(b_n(1))
9 C_n=b_n
10 //theta_n(1)=0;
11 theta_n=atan(-b_n,a_n);
12 //n=[0,n];
13 clf;
14 size(n)
15 subplot(2,2,1); plot2d3(n,a_n); xtitle('a_n','n');
    plot(n,a_n,'ro')
16 subplot(2,2,2); plot2d3(n,b_n); xtitle('b_n','n');
    plot(n,b_n,'r.')
17 subplot(2,2,3); plot2d3(n,C_n); xtitle('C_n','n');
    plot(n,C_n,'ro')
18 subplot(2,2,4); plot2d3(n,theta_n,); xtitle('theta_n',
    'n');plot(n,theta_n,'r.')

```

---

### Scilab code Exa 6.3 fourier spectra of a signal

```
1 n=0:10;
2
3 for n=0:10
4     // if (n%2==0)
5     //   a_n=0;
6     // else
7     //   if (n==4*n-3)
8     //     a_n=2/(%pi.*n);
9     //   else if (n==4*n-1)
10    //     a_n=-2/(%pi.*n);
11    //   end end end
12
13 b_n=zeros(1,length(n));
14 size(n)
15 size(a_n)
16 size(b_n)
17 disp(b_n(1))
18 C_n=sqrt(a_n.^2+(b_n).^2);
19 theta_n(1)=0; theta_n=atan(-b_n,a_n);
20 //n=[0,n];
21 clf;
22 size(n)
23 subplot(2,2,1); plot2d3(n,a_n);xtitle('a_n','n');
    plot(n,a_n,'ro');
24 subplot(2,2,2); plot2d3(n,b_n);xtitle('b_n','n');
    plot(n,b_n,'r. ');
25 subplot(2,2,3); plot2d3(n,C_n);xtitle('C_n','n');
    plot(n,C_n,'ro');
26 subplot(2,2,4); plot2d3(n,theta_n,);xtitle('theta_n'
    , 'n');plot(n,theta_n,'r. ');
```

---

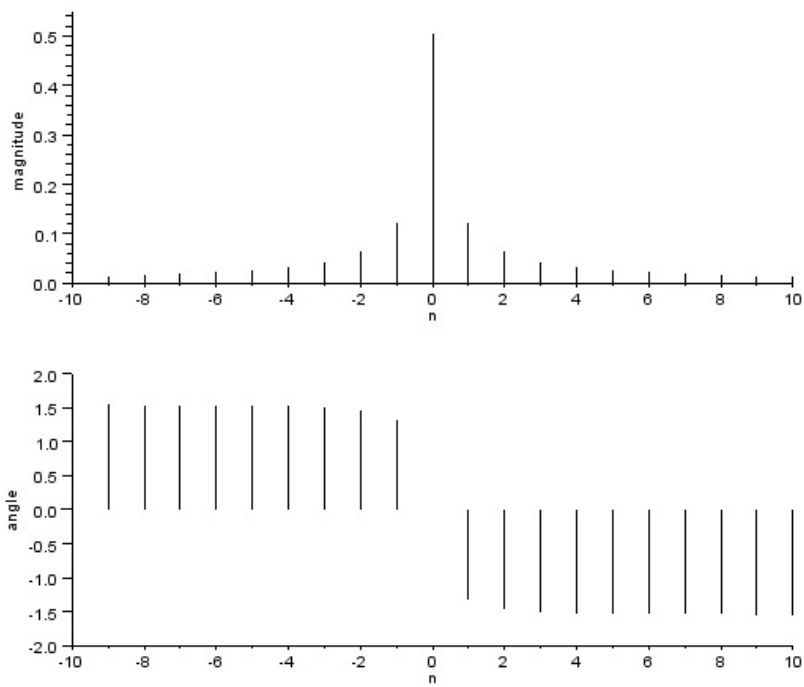


Figure 6.3: exponential fourier series

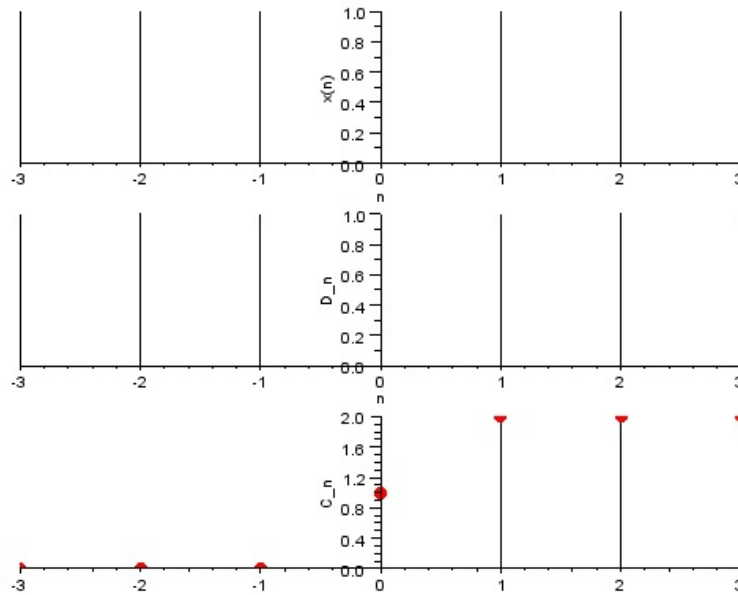


Figure 6.4: exponential fourier series for the impulse train

#### Scilab code Exa 6.5 exponential fourier series

```

1 n=(-10:10); D_n=0.504./(1+ %i*4*n);
2 clf;
3 subplot(2,1,1); plot2d3(n,abs(D_n));
4 subplot(2,1,2); plot2d3(n,atan(imag(D_n),real(D_n)))
  ;

```

---

#### Scilab code Exa 6.7 exponential fourier series for the impulse train

```

1 //signals and systems
2 //fourier series for train of impulses
3 clear;
4 close;

```

```

5  clc;
6  n=-3:1:3
7  x = ones(1,length(n))
8  D_n=ones(1,length(n));
9  C_n=[0 0 0 1 2 2 2]
10 subplot(3,1,1)
11 a = gca();
12 a.y_location = "origin";
13 a.x_location = "origin";
14 plot2d3(n,x)
15 subplot(3,1,2)
16 a = gca();
17 a.y_location = "origin";
18 a.x_location = "origin";
19 plot2d3(n,D_n)
20 subplot(3,1,3)
21 a = gca();
22 a.y_location = "origin";
23 a.x_location = "origin";
24 plot2d3(n,C_n); plot(n,C_n,'r.')
```

---

**Scilab code Exa 6.9** exponential fourier series to find the output

```

1  n=(-10:10); D_n=2/(3.14*(1-4.*n.^2).*(%i*6.*n+1));
2  clf;
3  subplot(2,1,1); plot2d3(n,abs(D_n));
4  subplot(2,1,2); plot2d3(n,atan(imag(D_n),real(D_n)))
   ;
```

---



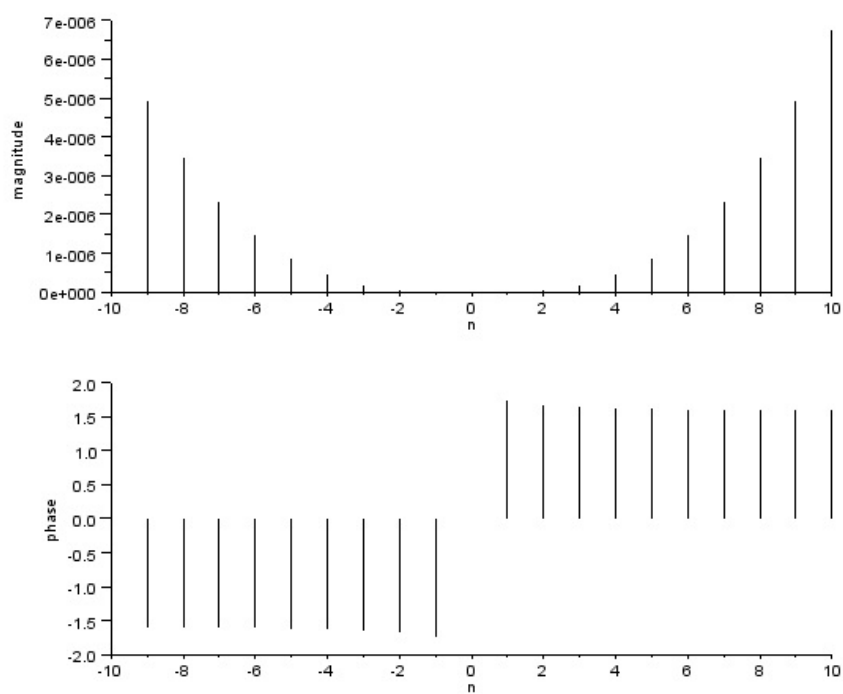


Figure 6.5: exponential fourier series to find the output

# Chapter 7

## continuous time signal analysis the fourier transform

Scilab code Exa 7.1 fourier transform of exponential function

```
1 //signals and systems
2 //continuous time signal analysis the fourier
  transform
3 //fourier transform of exp(-A*t)
4 clear;
5 clc;
6 A =1; //Amplitude
7 Dt = 0.005;
8 t = -4.5:Dt:4.5;
9 xt = exp(-A*abs(t));
10 Wmax = 2*pi*1; //Analog Frequency = 1Hz
11 K = 4;
12 k = 0:(K/1000):K;
13 W = k*Wmax/K;
14 XW = xt* exp(-sqrt(-1)*t'*W) * Dt;
15 XW = real(XW);
16 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -
```

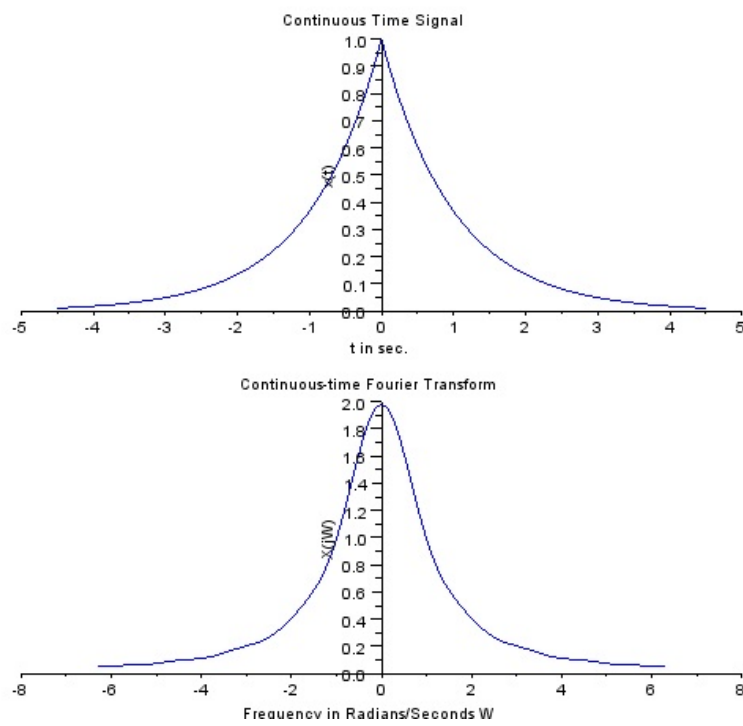


Figure 7.1: fourier transform of exponential function

```

    Wmax to Wmax
17 XW = [mtlbfliplr(XW), XW(2:1001)];
18 subplot(2,1,1);
19 a = gca();
20 a.y_location = "origin";
21 plot(t,xt);
22 xlabel('t in sec. ');
23 ylabel('x(t) ');
24 title('Continuous Time Signal')
25 subplot(2,1,2);
26 a = gca();
27 a.y_location = "origin";
28 plot(W,XW);
29 xlabel('Frequency in Radians/Seconds W');
30 ylabel('X(jW) ');
31 title('Continuous-time Fourier Transform')

```

---

#### Scilab code Exa 7.4 inverse fourier transform

```

1 //Example 4.5
2 // Inverse Continuous Time Fourier Transform
3 // impulse funtion
4 clear;
5 clc;
6 close;
7 // CTFT
8 A =1; //Amplitude
9 Dw = 0.005;
10 W1 = 4; //Time in seconds
11 w = -W1/2:Dw:W1/2;
12 for i=1:length(w)
13     XW(1)=1;
14     end

```

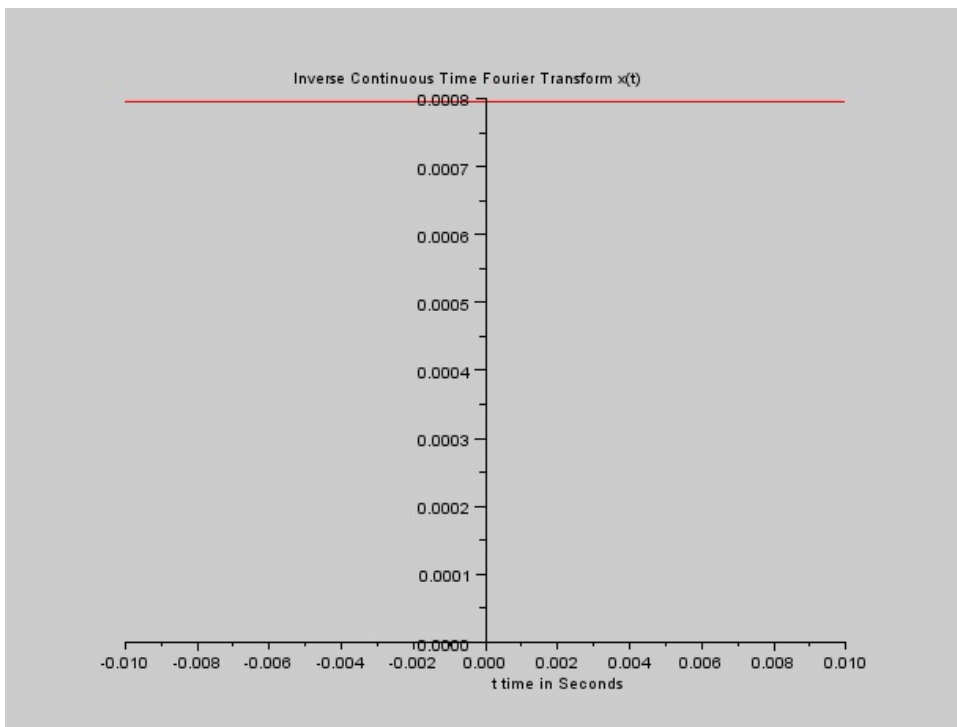


Figure 7.2: inverse fourier transform

```

15 XW = XW';
16
17 //Inverse Continuous-time Fourier Transform
18 t = -0.01:1/length(w):0.01;
19 xt = (1/(2*pi))*XW * exp(sqrt(-1)*w'*t)*Dw;
20 xt = real(xt);
21 figure
22 a = gca();
23 a.y_location = "origin";
24 a.x_location = "origin";
25 plot(t,xt);
26 xlabel('                                t time
           in Seconds');
27 title('Inverse Continuous Time Fourier Transform x(t
        )')

```

---

#### Scilab code Exa 7.5 inverse fourier transform

```

1 //signals and systems
2 // Inverse Continuous Time Fourier Transform
3 // shifted impulse function
4 clear;
5 clc;
6 close;
7 w0=1
8 A =1; //Amplitude
9 Dw = 0.005;
10 W1 = 4; //Time in seconds
11 w = -W1/2:Dw:W1/2;
12 XW=[zeros(1,length(w)/2) 1 zeros(1,length(w/2))];
13 XW = XW';
14
15 //Inverse Continuous-time Fourier Transform
16 t = -0.01:1/length(w):0.01;
17 size(XW)

```

```

18 size(t)
19 xt =(1/(2*%pi))*XW *exp(sqrt(-1)*w'.*t).*exp(sqrt
    (-1).*t)*Dw;
20 xt = real(xt);
21 figure
22 a = gca();
23 a.y_location =" origin";
24 a.x_location =" origin";
25 plot(t,xt);
26 xlabel('                                t time
    in Seconds');
27 title('Inverse Continuous Time Fourier Transform x(t
    )')

```

---

Scilab code Exa 7.6 fourier transform for everlasting sinusoid

```

1 //signals and systems
2 // Continuous Time Fourier Transforms
3 // Sinusoidal waveforms cos(Wot)
4 clear;
5 clc;
6 close;
7
8 T1 = 2;
9 T = 4*T1;
10 Wo = 2*%pi/T;
11 W = [-Wo,0,Wo];
12 ak = (2*%pi*Wo*T1/%pi)/sqrt(-1);
13 XW = [-ak,0,ak];
14 ak1 = (2*%pi*Wo*T1/%pi);
15 XW1 =[ak1,0,ak1];
16
17 figure

```

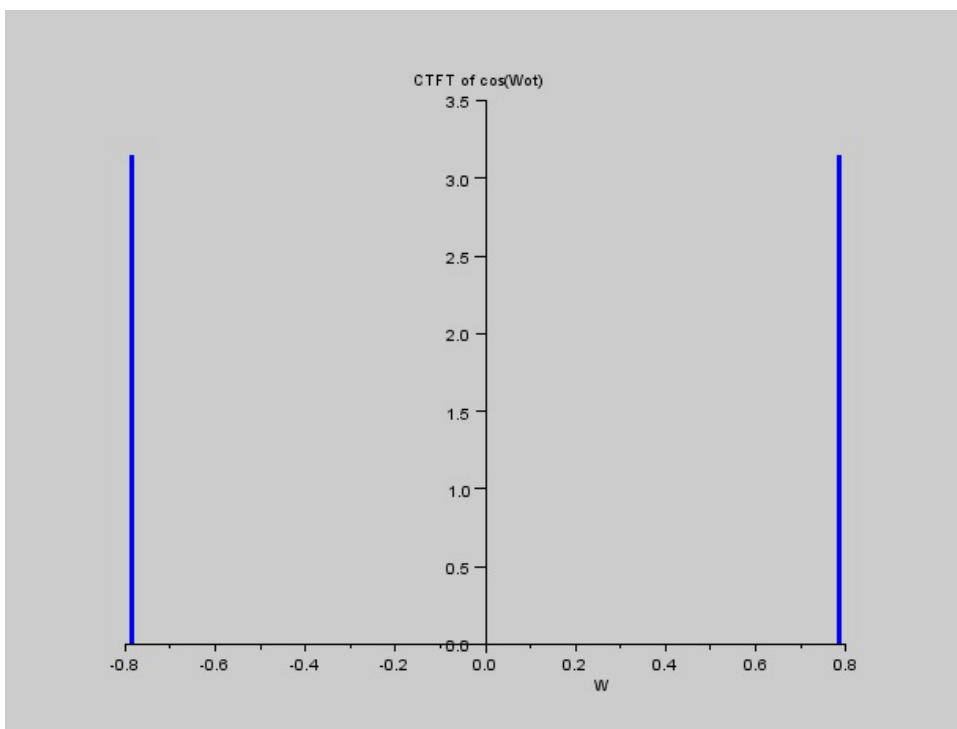


Figure 7.3: fourier transform for everlasting sinusoid



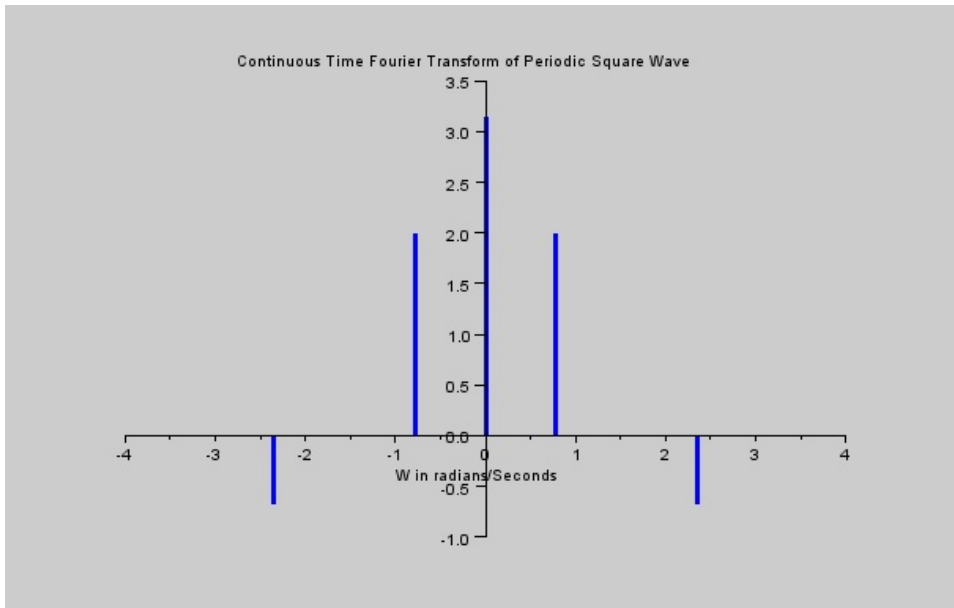


Figure 7.4: fourier transform of a periodic signal

```

18 a = gca();
19 a.y_location = "origin";
20 a.x_location = "origin";
21 plot2d3('gnn',W,XW1,2);
22 poly1 = a.children(1).children(1);
23 poly1.thickness = 3;
24 xlabel('
    W');
25 title('CTFT of cos(Wot)')

```

---

Scilab code Exa 7.7 fourier transform of a periodic signal

```

1 //signals and systems

```

```

2 // Continuous Time Fourier Transform of Symmetric
3 // periodic Square waveform
4 clear;
5 clc;
6 close;
7
8 T1 = 2;
9 T = 4*T1;
10 Wo = 2*%pi/T;
11 W = -%pi:Wo:%pi;
12 delta = ones(1,length(W));
13 XW(1) = (2*%pi*Wo*T1/%pi);
14 mid_value = ceil(length(W)/2);
15 for k = 2:mid_value
16     XW(k) = (2*%pi*sin((k-1)*Wo*T1)/(%pi*(k-1)));
17 end
18 figure
19 a = gca();
20 a.y_location = "origin";
21 a.x_location = "origin";
22 plot2d3('gnn',W(mid_value:$),XW,2);
23 poly1 = a.children(1).children(1);
24 poly1.thickness = 3;
25 plot2d3('gnn',W(1:mid_value-1),XW($:-1:2),2);
26 poly1 = a.children(1).children(1);
27 poly1.thickness = 3;
28 xlabel('W in radians/Seconds');
29 title('Continuous Time Fourier Transform of Periodic
        Square Wave')

```

---

Scilab code Exa 7.8 fourier transform of a unit impulse train

```

1 //signals and systems

```

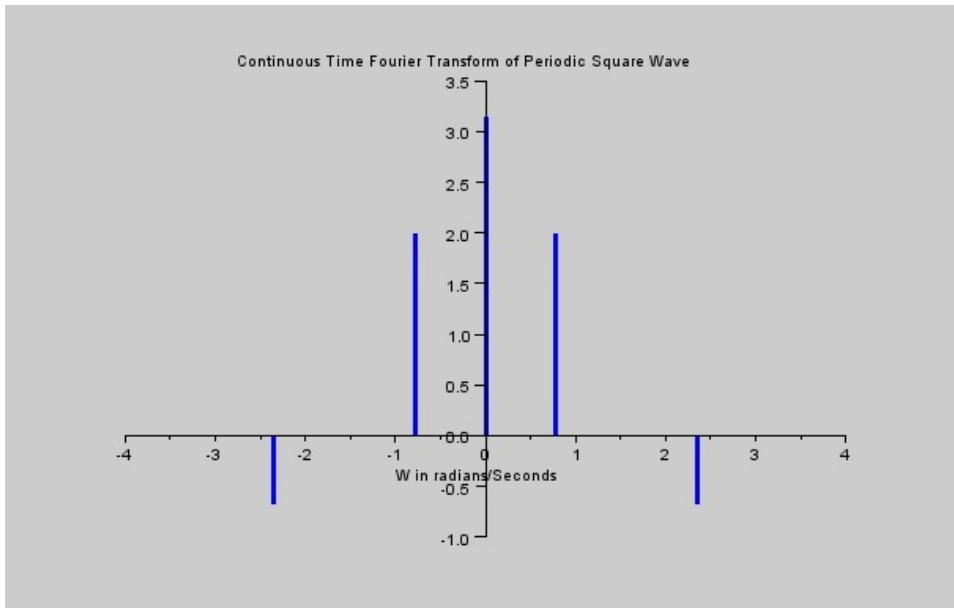


Figure 7.5: fourier transform of a unit impulse train

```

2 //continuous time signal analysis the fourier
  transform
3 // Periodic Impulse Train
4 clear;
5 clc;
6 close;
7 T = -4:4;;
8 T1 = 1; //Sampling Interval
9 xt = ones(1,length(T));
10 ak = 1/T1;
11 XW = 2*%pi*ak*ones(1,length(T));
12 Wo = 2*%pi/T1;
13 W = Wo*T;
14 figure
15 subplot(2,1,1)
16 a = gca();
17 a.y_location = "origin";
18 a.x_location = "origin";

```

```

19 plot2d3('ggn',T,xt,2);
20 poly1 = a.children(1).children(1);
21 poly1.thickness = 3;
22 xlabel('

        t');
23 title('Periodic Impulse Train')
24 subplot(2,1,2)
25 a = gca();
26 a.y_location = "origin";
27 a.x_location = "origin";
28 plot2d3('ggn',W,XW,2);
29 poly1 = a.children(1).children(1);
30 poly1.thickness = 3;
31 xlabel('

        t');
32 title('CTFT of Periodic Impulse Train')

```

---

**Scilab code Exa 7.9** fourier transform of unit step function

```

1 //signals and systems
2 //continuous time signal analysis the fourier
  transform
3 //fourier transform of unit step function u(t)
4 clear;
5 clc;
6 A =0.000000001;      //Amplitude
7 Dt = 0.005;
8 t = 0:Dt:4.5;
9 xt = exp(-A*abs(t));
10 Wmax = 2*pi*1;      //Analog Frequency = 1Hz
11 K = 4;

```

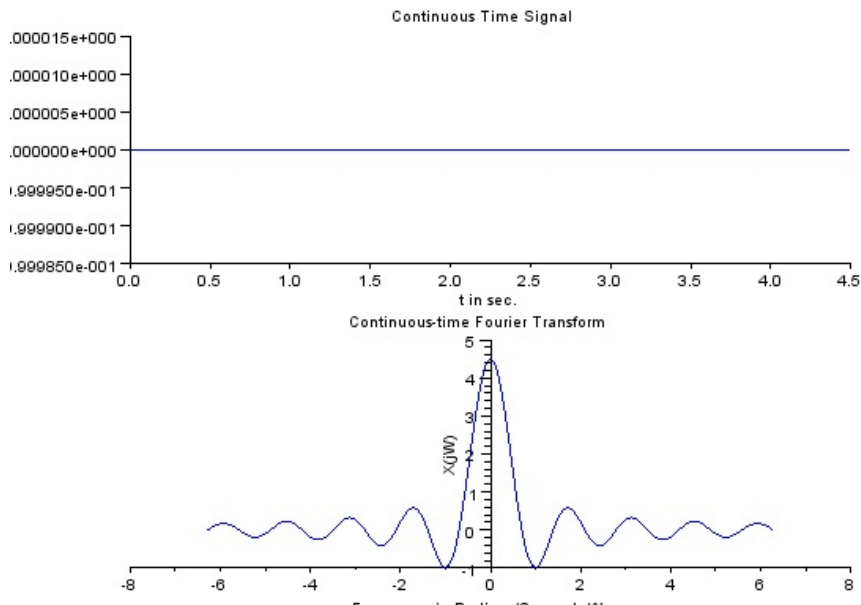


Figure 7.6: fourier transform of unit step function

```

12 k = 0:(K/500):K;
13 W = k*Wmax/K;
14 XW = xt* exp(-sqrt(-1)*t'*W) * Dt;
15 XW = real(XW);
16 W = [-mtlb_fliplr(W), W(2:501)]; // Omega from -Wmax
    to Wmax
17 XW = [mtlb_fliplr(XW), XW(2:501)];
18 subplot(2,1,1);
19 a = gca();
20 a.y_location = "origin";
21 plot(t,xt);
22 xlabel('t in sec. ');
23 ylabel('x(t) ');
24 title('Continuous Time Signal')
25 subplot(2,1,2);
26 a = gca();
27 a.y_location = "origin";
28 plot(W,XW);

```

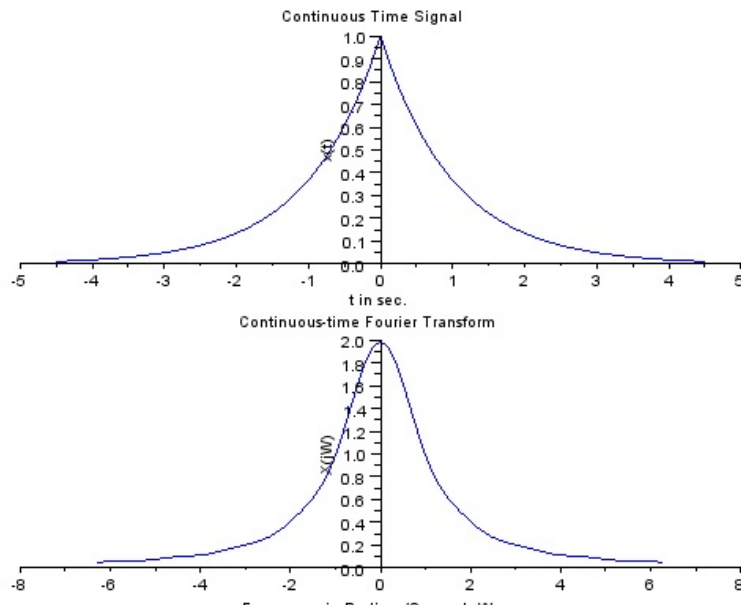


Figure 7.7: fourier transform of exponential function

```

29 xlabel('Frequency in Radians/Seconds W');
30 ylabel('X(jW)')
31 title('Continuous-time Fourier Transform')

```

---

**Scilab code Exa 7.12** fourier transform of exponential function

```

1 //signals and systems
2 //Continuous Time Fourier Transform
3 //Continuous Time Signal x(t)= exp(-A*abs(t))
4 clear;
5 clc;
6 close;
7
8 A =1; //Amplitude

```

```

 9 Dt = 0.005;
10 t = -4.5:Dt:4.5;
11 xt = exp(-A*abs(t));
12
13 Wmax = 2*pi*1;           //Analog Frequency = 1Hz
14 K = 4;
15 k = 0:(K/1000):K;
16 W = k*Wmax/K;
17 XW = xt* exp(-sqrt(-1)*t'*W) * Dt;
18 XW = real(XW);
19 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -
    Wmax to Wmax
20 XW = [mtlbfliplr(XW), XW(2:1001)];
21 subplot(1,1,1)
22 subplot(2,1,1);
23 a = gca();
24 a.y_location = "origin";
25 plot(t,xt);
26 xlabel('t in sec. ');
27 ylabel('x(t)')
28 title('Continuous Time Signal')
29 subplot(2,1,2);
30 a = gca();
31 a.y_location = "origin";
32 plot(W,XW);
33 xlabel('Frequency in Radians/Seconds W');
34 ylabel('X(jW)')
35 title('Continuous-time Fourier Transform')

```

---

# Chapter 8

## Sampling The bridge from continuous to discrete

Scilab code Exa 8.8 discrete fourier transform

```
1 //signals and systems
2 //sampling:the bridge from continuous to discrete
3 //DFT to compute the fourier transform of  $e^{-2t}.u(t)$ 
4 T_0 = 4;
5 N_0 = 256;
6 T = T_0/N_0;
7 t = (0:T:T*(N_0-1))';
8 x = T*exp(-2*t);
9 x = mtlb_i(x,1,(T*(exp(-2*T_0)+1))/2);
10 X_r = fft(x);
11 r = (-N_0/2:N_0/2-1)';
12 omega_r = ((r*2)*%pi)/T_0;
13 omega = linspace(-%pi/T,%pi/T,4097);
14 X = 1 ./(%i*omega+2);
15 subplot(2,1,1);
16 a = gca();
17 a.y_location = "origin";
```



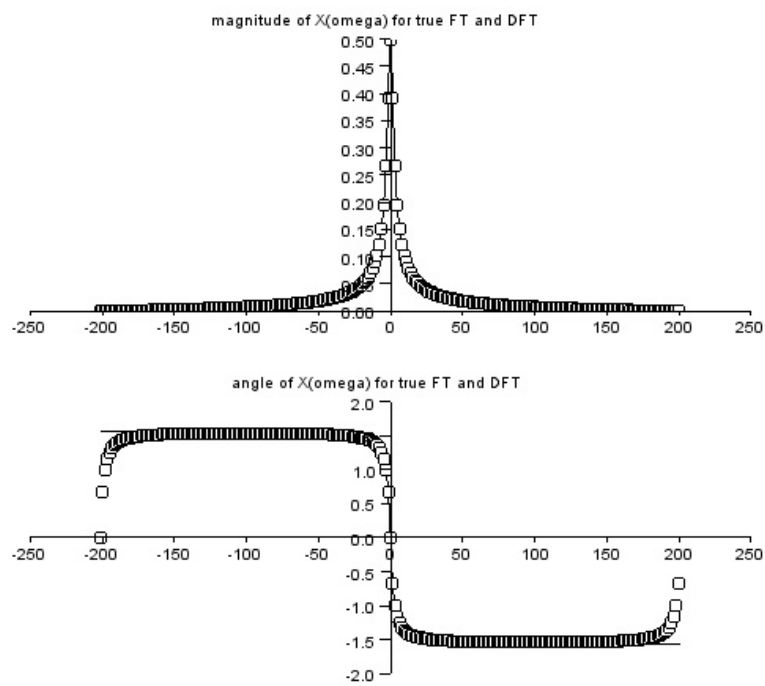


Figure 8.1: discrete fourier transform

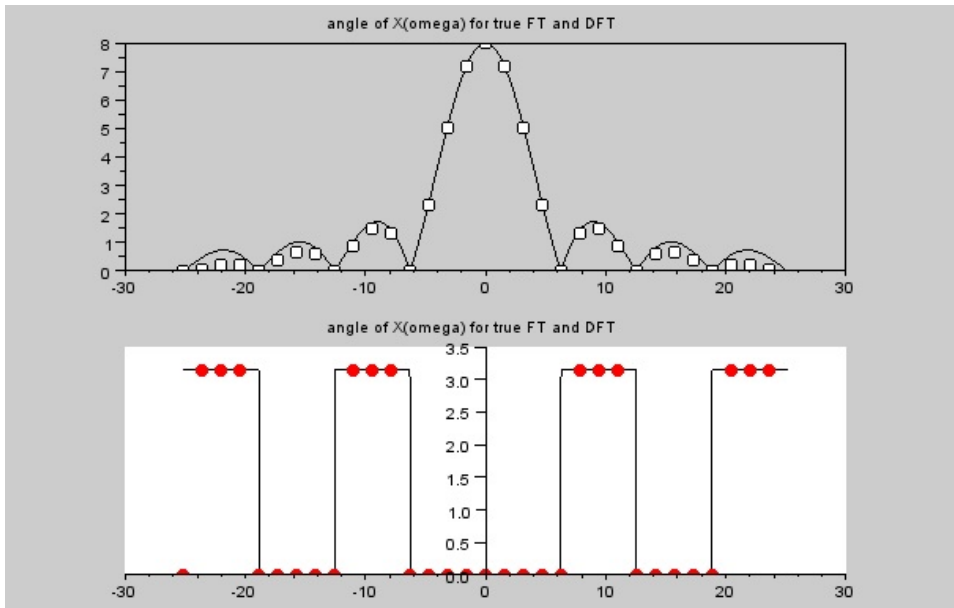


Figure 8.2: discrete fourier transform

```

18 a.x_location = "origin";
19 plot(omega,abs(X),"k",omega_r,fftshift(abs(X_r)),"ko
   ");
20 xtitle("magnitude of X(omega) for true FT and DFT");
21 subplot(2,1,2);
22 a = gca();
23 a.y_location = "origin";
24 a.x_location = "origin";
25 plot(omega,atan(imag(X),real(X)),"k",omega_r,
   fftshift(atan(imag(X_r),real(X_r))),"ko");
26 xtitle("angle of X(omega) for true FT and DFT");

```

---

Scilab code Exa 8.9 discrete fourier transform

```

1 //signals and systems
2 //sampling:the bridge from continuous to discrete
3 //DFT to compute the fourier transform of 8rect(t)
4 T_0 = 4;
5 N_0 = 32;
6 T = T_0/N_0;
7 x_n = [ones(1,4) 0.5 zeros(1,23) 0.5 ones(1,3)]';
8 size(x_n)
9 x_r = fft(x_n);r = (-N_0/2:(N_0/2)-1)';
10 omega_r = ((r*2)*%pi)/T_0;
11 omega = linspace(-%pi/T,%pi/T,4097);
12 size(omega_r)
13 size(omega)
14 X = 8*(sinc(omega/2));
15 size(X)
16 figure(1);
17 subplot(2,1,1);
18 plot(omega,abs(X),"k");
19 plot(omega_r,fftshift(abs(x_r)),"ko")
20 xtitle("angle of X(omega) for true FT and DFT");
21 a=gca();
22 subplot(2,1,2);
23 a = gca();
24 a.y_location = "origin";
25 a.x_location = "origin";
26 plot(omega,atan(imag(X),real(X)),"k",omega_r,
      fftshift(atan(imag(x_r),real(x_r))), 'r. ');
27 xtitle("angle of X(omega) for true FT and DFT");

```

---

Scilab code Exa 8.10 frequency response of a low pass filter

```

1 //signals and systems
2 // sampling: the bridge between continuous to

```

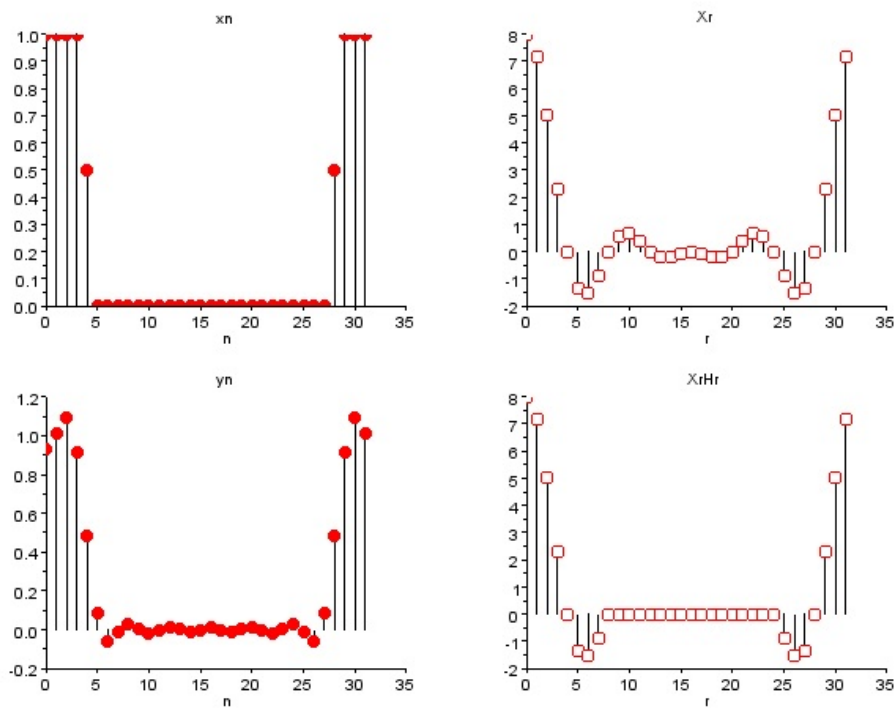


Figure 8.3: frequency response of a low pass filter

```

        discrete
3  T_0 = 4;
4  N_0 = 32;
5  T = T_0/N_0;n = 0:N_0-1;r = n;
6  x_n = [ones(1,4),0.5,zeros(1,23),0.5,ones(1,3)]';
7  H_r = [ones(1,8),0.5,zeros(1,15),0.5,ones(1,7)]';
8  X_r = fft(x_n,-1);
9  Y_r = H_r.*(X_r);y_n = mtlb_ifft(Y_r);
10 subplot(2,2,1);
11 plot2d3(n,x_n);
12 plot(n,x_n,'r. ');
13 xtitle('xn','n')
14 subplot(2,2,2);
15 plot2d3(r,real(X_r));
16 plot(r,real(X_r),'ro ')
17 xtitle('Xr','r')
18 subplot(2,2,3);
19 plot2d3(n,real(y_n));
20 plot(n,real(y_n),'r. ');
21 xtitle('yn','n')
22 subplot(2,2,4);
23 plot2d3(r,(X_r).*H_r);
24 plot(r,(X_r).*H_r,'ro ')
25 xtitle('XrHr','r')

```

---

# Chapter 9

## fourier analysis of discrete time signals

Scilab code Exa 9.1 discrete time fourier series

```
1 //signals and systems
2 //fourier analysis of discrete time signals
3 //Example5.5:Discrete Time Fourier Transform:x[n]=
   sin(nWo)
4 clear;
5 clc;
6 close;
7 N = 0.1;
8 Wo = %pi;
9 W = [-Wo/10,0,Wo/10];
10 XW = [0.5,0,0.5];
11 //
12 figure
13 a = gca();
14 a.y_location = "origin";
15 a.x_location = "origin";
16 plot2d3('gnn',W,XW,2);
```

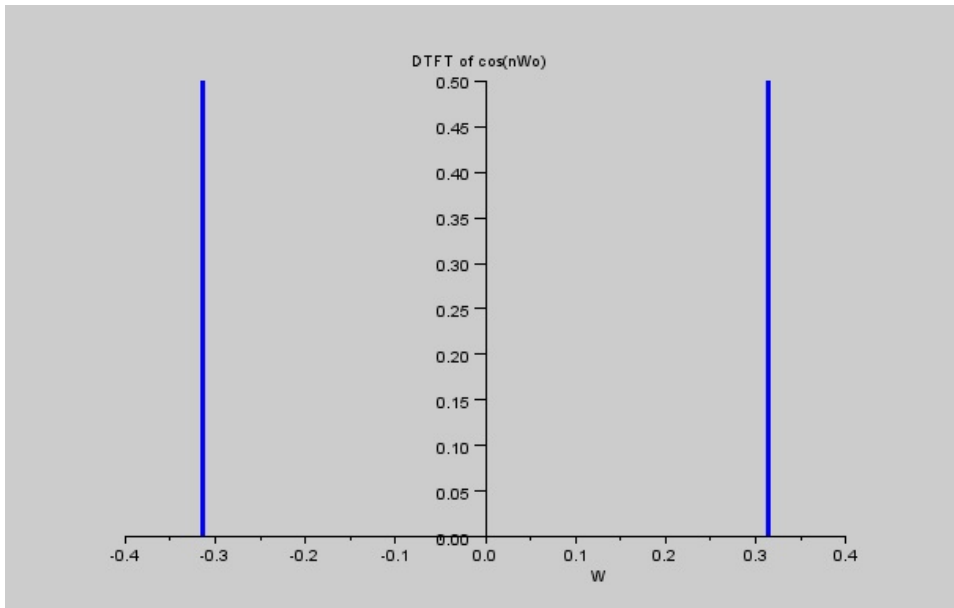


Figure 9.1: discrete time fourier series

```

17 poly1 = a.children(1).children(1);
18 poly1.thickness = 3;
19 xlabel('
    W');
20 title('DTFT of cos(nWo)')
21 disp(Wo/10)

```

---

**Scilab code Exa 9.2 DTFT for periodic sampled gate function**

```

1 N_0=32; n=(0:N_0-1);
2 x_n= [ones(1,5) zeros(1,23) ones(1,4)];
3 for r=0:31
4     X_r(r+1)=sum(x_n.*exp(-sqrt(-1)*r*2*3.14/N_0*n))

```

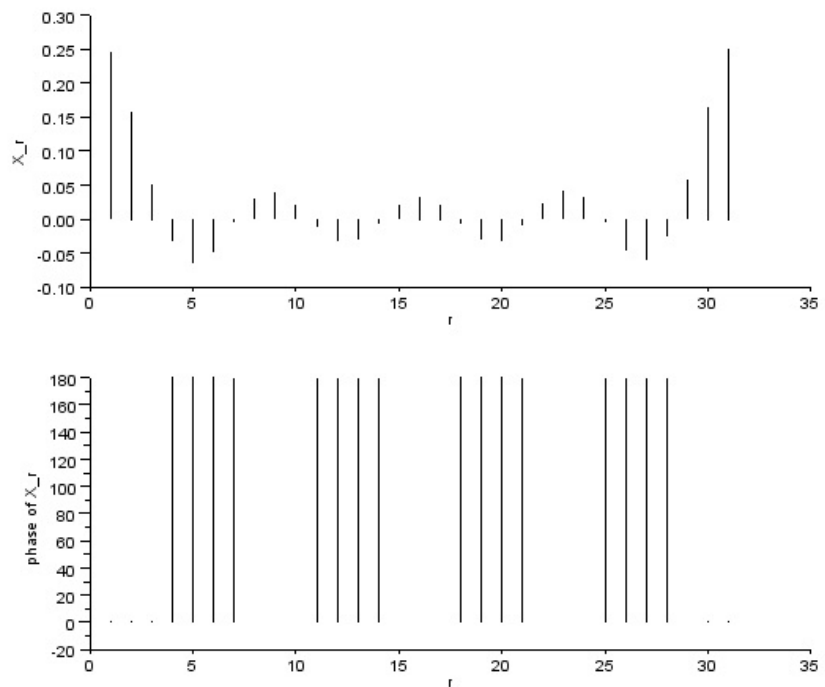


Figure 9.2: DTFT for periodic sampled gate function



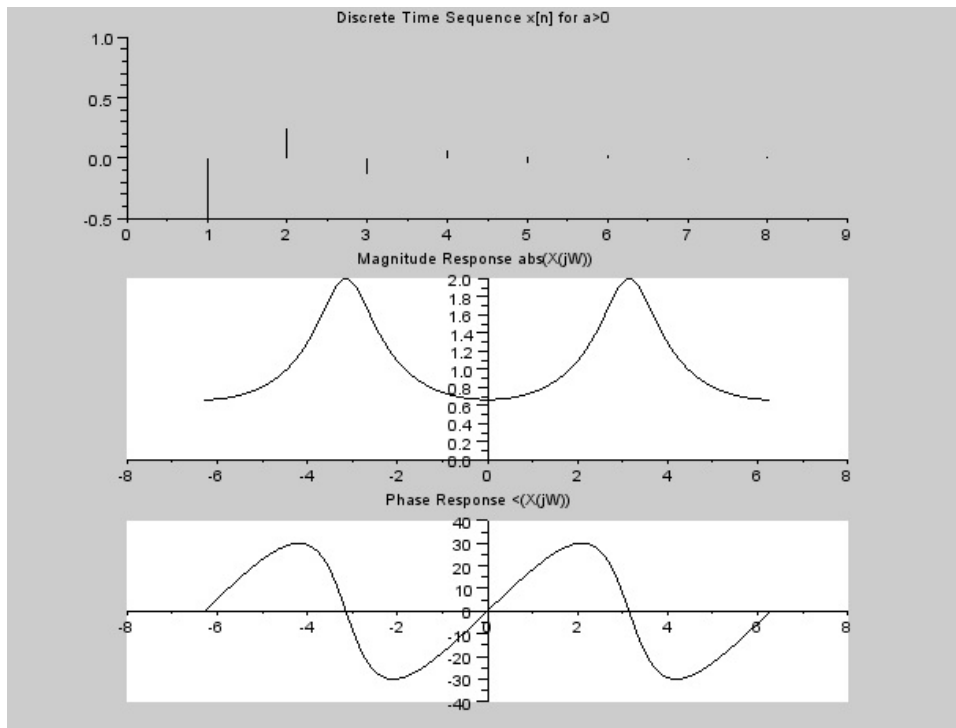


Figure 9.3: discrete time fourier series

```

/32;
5 end
6 subplot(2,1,1); r=n; plot2d3(r,real(X_r));
7 xlabel('r'); ylabel('X_r');
8 X_r=fft(x_n)/N_0;
9 subplot(2,1,2);
10 plot2d3(r,phasemag(X_r));
11 xlabel('r'); ylabel('phase of X_r');
12 disp(N_0,'period=')
13 disp(2*%pi/N_0,'omega=')

```

---

### Scilab code Exa 9.3 discrete time fourier series

```
1 //signals and systems
2 //Discrete Time Fourier Transform of discrete
   sequence
3 //x[n]= (a^n).u[n], a>0 and a<0
4 clear;
5 clc;
6 close;
7 // DTS Signal
8 a1 = 0.5;
9 a2 = -0.5;
10 max_limit = 10;
11 for n = 0:max_limit-1
12     x1(n+1) = (a1^n);
13     x2(n+1) = (a2^n);
14 end
15 n = 0:max_limit-1;
16 // Discrete-time Fourier Transform
17 Wmax = 2*%pi;
18 K = 4;
19 k = 0:(K/1000):K;
20 W = k*Wmax/K;
21 x1 = x1';
22 x2 = x2';
23 XW1 = x1* exp(-sqrt(-1)*n'*W);
24 XW2 = x2* exp(-sqrt(-1)*n'*W);
25 XW1_Mag = abs(XW1);
26 XW2_Mag = abs(XW2);
27 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -
   Wmax to Wmax
28 XW1_Mag = [mtlbfliplr(XW1_Mag), XW1_Mag(2:1001)];
29 XW2_Mag = [mtlbfliplr(XW2_Mag), XW2_Mag(2:1001)];
30 [XW1_Phase,db] = phasemag(XW1);
31 [XW2_Phase,db] = phasemag(XW2);
32 XW1_Phase = [-mtlbfliplr(XW1_Phase),XW1_Phase
   (2:1001)];
33 XW2_Phase = [-mtlbfliplr(XW2_Phase),XW2_Phase
```

```

        (2:1001)];
34 //plot for a>0
35 figure
36 subplot(3,1,1);
37 plot2d3('gmn',n,x1);
38 xtitle('Discrete Time Sequence x[n] for a>0')
39 subplot(3,1,2);
40 a = gca();
41 a.y_location = "origin";
42 a.x_location = "origin";
43 plot2d(W,XW1_Mag);
44 title('Magnitude Response abs(X(jW))')
45 subplot(3,1,3);
46 a = gca();
47 a.y_location = "origin";
48 a.x_location = "origin";
49 plot2d(W,XW1_Phase);
50 title('Phase Response <(X(jW))')
51 //plot for a<0
52 figure
53 subplot(3,1,1);
54 plot2d3('gmn',n,x2);
55 xtitle('Discrete Time Sequence x[n] for a>0')
56 subplot(3,1,2);
57 a = gca();
58 a.y_location = "origin";
59 a.x_location = "origin";
60 plot2d(W,XW2_Mag);
61 title('Magnitude Response abs(X(jW))')
62 subplot(3,1,3);
63 a = gca();
64 a.y_location = "origin";
65 a.x_location = "origin";
66 plot2d(W,XW2_Phase);
67 title('Phase Response <(X(jW))')

```

---

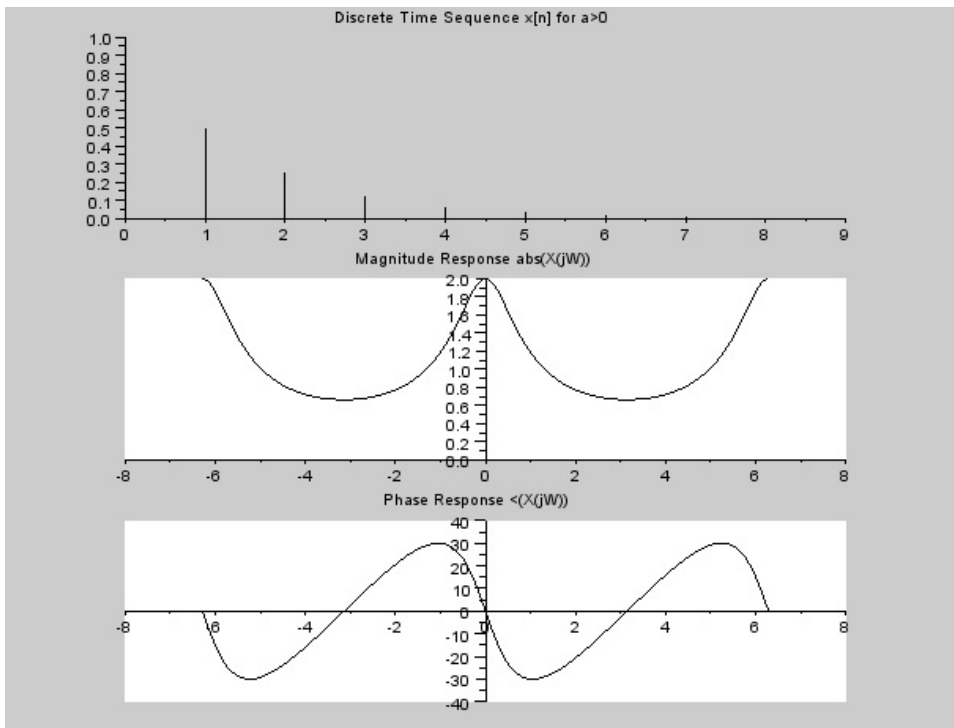


Figure 9.4: discrete time fourier series

Scilab code Exa 9.4 discrete time fourier series

```

1 //signals and systems
2 //Discrete Time Fourier Transform of discrete
  sequence
3 //x[n]= (a^n).u[-n], a>0 and a<0
4 clear;
5 clc;

```

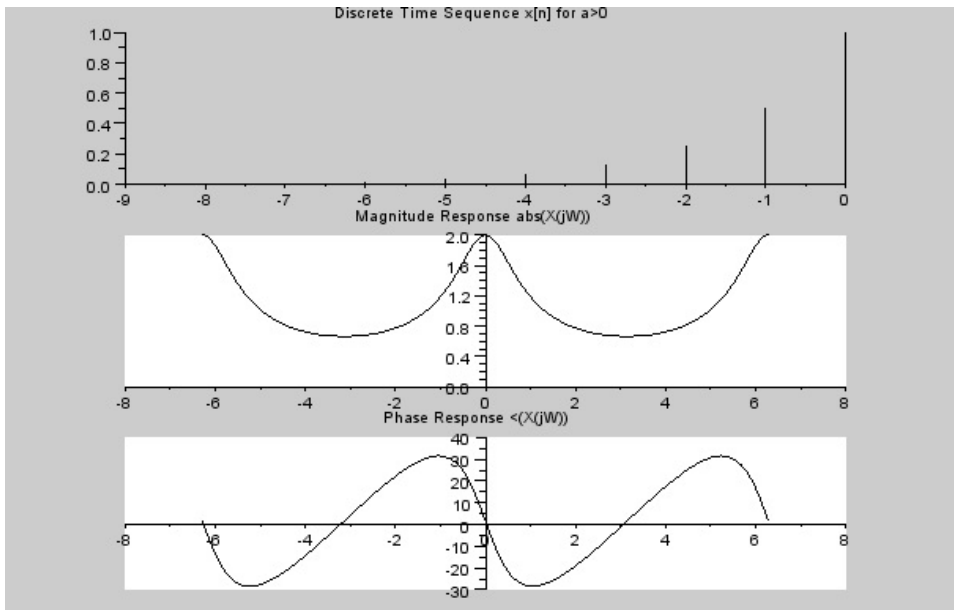


Figure 9.5: discrete time fourier series

```

6 close;
7 // DTS Signal
8 a = 0.5;
9 max_limit = 10;
10 for n = 0:max_limit-1
11     x1(n+1) = (a^n);
12 end
13 n = 0:max_limit-1;
14 // Discrete-time Fourier Transform
15 Wmax = 2*pi;
16 K = 4;
17 k = 0:(K/1000):K;
18 W = k*Wmax/K;
19 x1 = x1';
20 XW1 = x1* exp(-sqrt(-1)*n'*W);
21
22 XW1_Mag = abs(XW1);
23 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -

```

```

    Wmax to Wmax
24 XW1_Mag = [mtlbfliplr(XW1_Mag), XW1_Mag(2:1001)];
25 [XW1_Phase,db] = phasemag(XW1);
26 XW1_Phase = [-mtlbfliplr(XW1_Phase),XW1_Phase
    (2:1001)];
27 //plot for a>0
28 figure
29 subplot(3,1,1);
30 plot2d3('gmn',-n,x1);
31 xtitle('Discrete Time Sequence x[n] for a>0')
32 subplot(3,1,2);
33 a = gca();
34 a.y_location = "origin";
35 a.x_location = "origin";
36 plot2d(W,XW1_Mag);
37 title('Magnitude Response abs(X(jW))')
38 subplot(3,1,3);
39 a = gca();
40 a.y_location = "origin";
41 a.x_location = "origin";
42 plot2d(W,XW1_Phase+%pi/2);
43 title('Phase Response <(X(jW))')

```

---

### Scilab code Exa 9.5 DTFT for rectangular pulse

```

1 //signals and systems
2 //Discrete Time Fourier Transform
3 //x[n]= 1 , abs(n)<=N1
4 clear;
5 clc;
6 close;
7 // DTS Signal
8 N1 = 2;

```

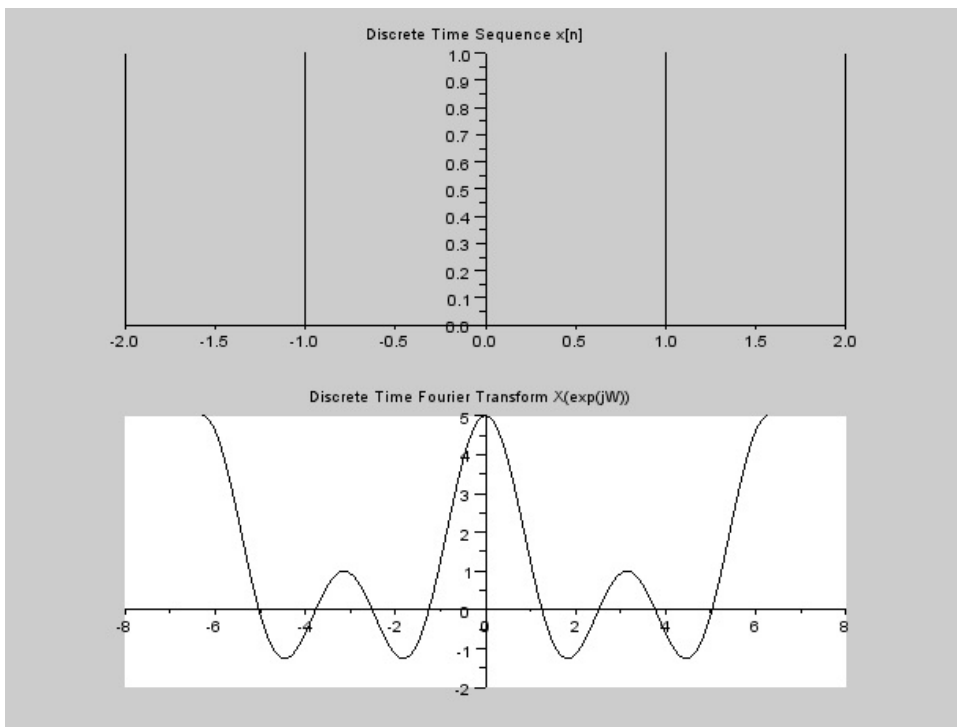


Figure 9.6: DTFT for rectangular pulse

```

9  n = -N1:N1;
10 x = ones(1,length(n));
11 // Discrete-time Fourier Transform
12 Wmax = 2*%pi;
13 K = 4;
14 k = 0:(K/1000):K;
15 W = k*Wmax/K;
16 XW = x* exp(-sqrt(-1)*n'*W);
17 XW_Mag = real(XW);
18 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -
    Wmax to Wmax
19 XW_Mag = [mtlbfliplr(XW_Mag), XW_Mag(2:1001)];
20 //plot for abs(a)<1
21 figure
22 subplot(2,1,1);
23 a = gca();
24 a.y_location = "origin";
25 a.x_location = "origin";
26 plot2d3('gnn',n,x);
27 xtitle('Discrete Time Sequence x[n]')
28 subplot(2,1,2);
29 a = gca();
30 a.y_location = "origin";
31 a.x_location = "origin";
32 plot2d(W,XW_Mag);
33 title('Discrete Time Fourier Transform X(exp(jW))')

```

---

Scilab code Exa 9.6 DTFT for rectangular pulse spectrum

```

1 //signals and systems
2 //discreet time fourier series
3 //IDTFT:Impulse Response of Ideal Low pass Filter
4 clear;

```



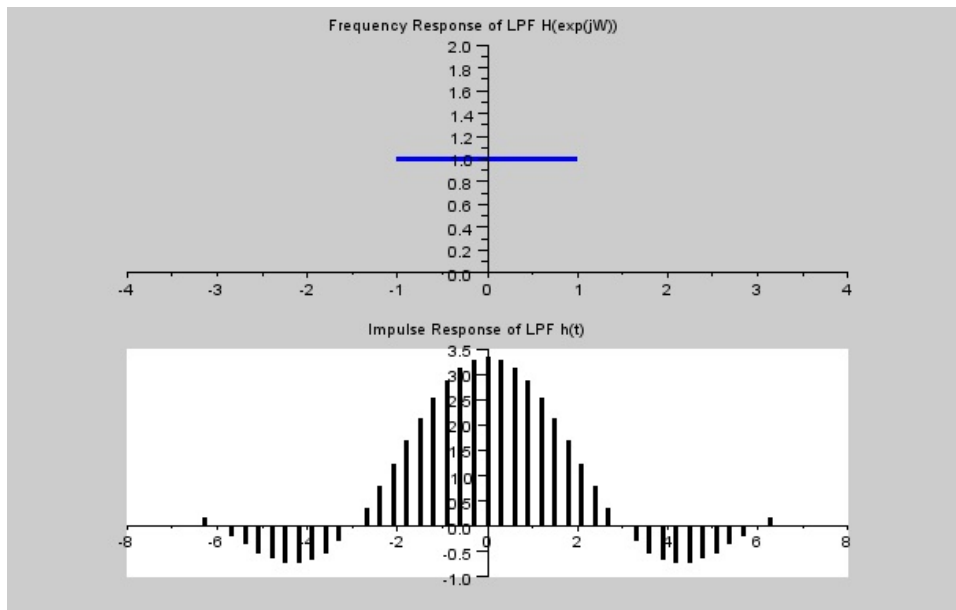


Figure 9.7: DTFT for rectangular pulse spectrum

```

5  clc;
6  close;
7  Wc = 1; //1 rad/sec
8  W = -Wc:0.1:Wc; //Passband of filter
9  H0 = 1; //Magnitude of Filter
10 HlpW = H0*ones(1,length(W));
11 //Inverse Discrete-time Fourier Transform
12 t = -2*%pi:2*%pi/length(W):2*%pi;
13 ht = (1/(2*%pi))*HlpW *exp(sqrt(-1)*W'*t);
14 ht = real(ht);
15 figure
16 subplot(2,1,1)
17 a = gca();
18 a.y_location = "origin";
19 a.x_location = "origin";
20 a.data_bounds = [-%pi,0;%pi,2];
21 plot2d(W,HlpW,2);
22 poly1 = a.children(1).children(1);

```

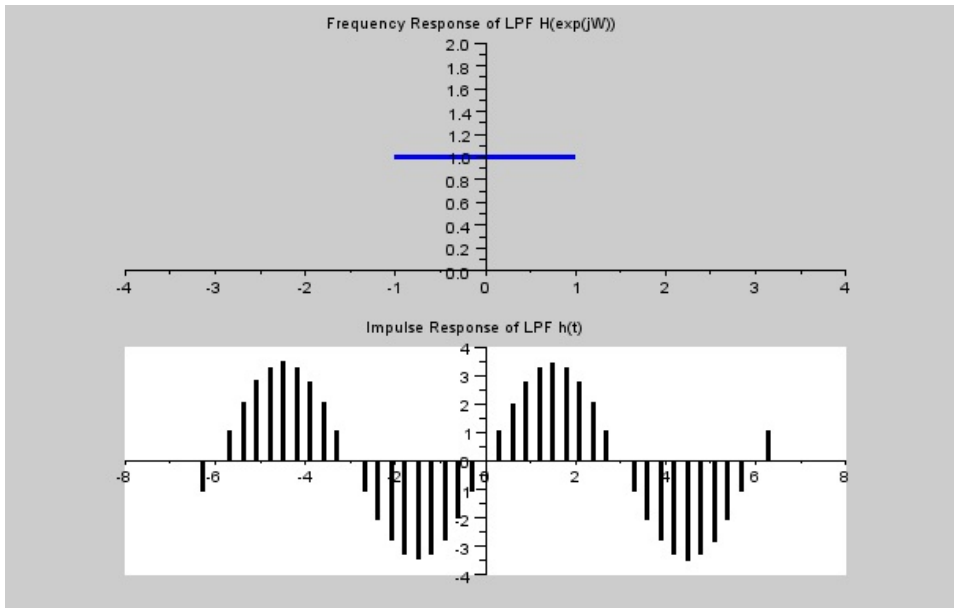


Figure 9.8: DTFT of sinc function

```

23 poly1.thickness = 3;
24 xtitle('Frequency Response of LPF H(exp(jW))')
25 subplot(2,1,2)
26 a = gca();
27 a.y_location = "origin";
28 a.x_location = "origin";
29 a.data_bounds = [-2*pi, -1; 2*pi, 2];
30 plot2d3('gnn', t, ht);
31 poly1 = a.children(1).children(1);
32 poly1.thickness = 3;
33 xtitle('Impulse Response of LPF h(t)')

```

---

Scilab code Exa 9.9 DTFT of sinc function

```

1 //signals and systems
2 //discreet time fourier series
3 //IDTFT:Impulse Response of Ideal Low pass Filter
4 clear;
5 clc;
6 close;
7 Wc = 1; //1 rad/sec
8 W = -Wc:0.1:Wc; //Passband of filter
9 H0 = 1; //Magnitude of Filter
10 HlpW = H0*ones(1,length(W));
11 //Inverse Discrete-time Fourier Transform
12 t = -2*%pi:2*%pi/length(W):2*%pi;
13 ht1 = (1/(2*%pi))*HlpW *exp(sqrt(-1)*W'*t);
14 size(ht1)
15 n=-21:21;
16 size(n)
17 ht=ht1.*(%e^i*2*t);
18 ht = real(ht);
19 figure
20 subplot(2,1,1)
21 a = gca();
22 a.y_location = "origin";
23 a.x_location = "origin";
24 a.data_bounds=[-%pi,0;%pi,2];
25 plot2d(W,HlpW,2);
26 poly1 = a.children(1).children(1);
27 poly1.thickness = 3;
28 xtitle('Frequency Response of LPF H(exp(jW))')
29 subplot(2,1,2)
30 a = gca();
31 a.y_location = "origin";
32 a.x_location = "origin";
33 a.data_bounds=[-2*%pi,-1;2*%pi,2];
34 size(t)
35 size(ht)
36 plot2d3('gmn',t,ht);
37 poly1 = a.children(1).children(1);
38 poly1.thickness = 3;

```

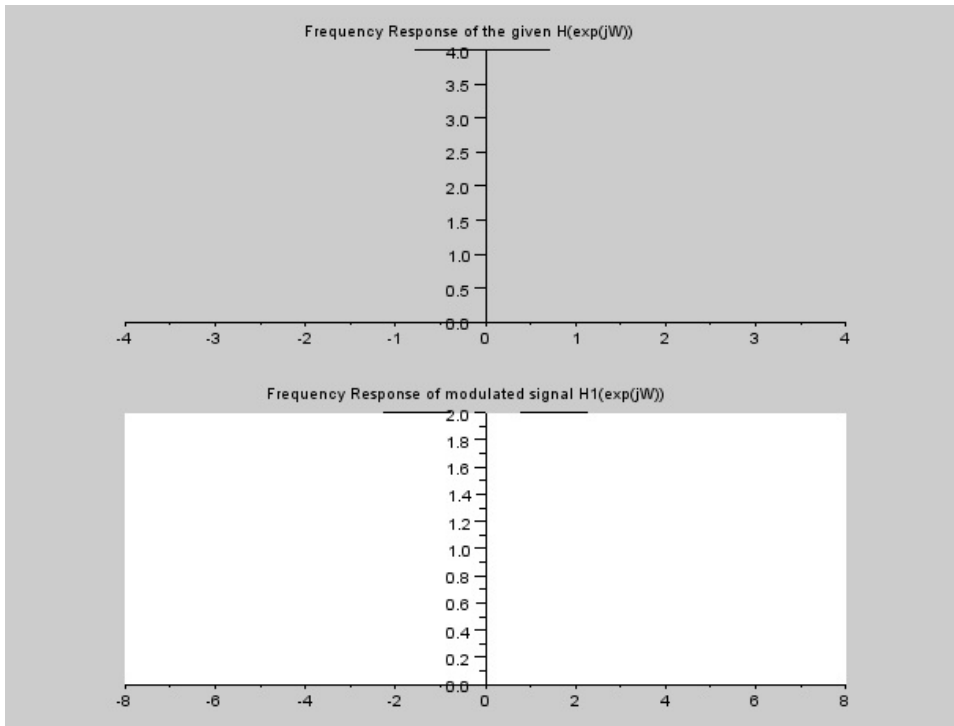


Figure 9.9: sketching the spectrum for a modulated signal

39 `xtitle('Impulse Response of LPF h(t)')`

---

Scilab code Exa 9.10.a sketching the spectrum for a modulated signal

```

1 //signals and systems
2 //discrete fourier transform
3 //Frequency Shifting Property of DTFT
4 clear;
5 clc;
6 close;
7 mag = 4;

```

```

8 W = -%pi/4:0.1:%pi/4;
9 H1 = mag*ones(1,length(W));
10 W1 =W+%pi/2;
11 W2 = -W-%pi/2;
12 figure
13 subplot(2,1,1)
14 a = gca();
15 a.y_location = "origin";
16 a.x_location = "origin";
17 a.data_bounds=[-%pi,0;%pi,2];
18 plot2d(W,H1);
19 xtitle('Frequency Response of the given H(exp(jW))')
20 subplot(2,1,2)
21 a = gca();
22 a.y_location = "origin";
23 a.x_location = "origin";
24 a.data_bounds=[-2*%pi,0;2*%pi,2];
25 plot2d(W1,0.5*H1);
26 plot2d(W2,0.5*H1);
27 xtitle('Frequency Response of modulated signal H1(
    exp(jW))')

```

---

### Scilab code Exa 9.13 frequency response of LTID

```

1 //LTi Systems characterized by Linear Constant
2 //fourier analysis of discrete systems
3 //Inverse Z Transform
4 //z = %z;
5 syms n z;
6 H1 = (-5/3)/(z-0.5);
7 H2 = (8/3)/(z-0.8);
8 F1 = H1*z^(n)*(z-0.5);
9 F2 = H2*z^(n)*(z-0.8);
10 h1 = limit(F1,z,0.5);
11 disp(h1, 'h1 [n]= ')

```

```
12 h2 = limit(F2,z,0.8);
13 disp(h2, 'h2[n]= ')
14 h = h1-h2;
15 disp(h, 'h[n]= ')
```

---

# Chapter 10

## state space analysis

Scilab code Exa 10.4 state space description by transfer function

```
1 //signals and systems
2 //state space analysis
3 //state space description
4 clear;
5 close;
6 clc;
7 s=poly(0, 's');
8 H=[(4/3)/(1+s), -2/(3+s), (2/3)/(4+s)];
9 Sys=tf2ss(H)
10 clean(ss2tf(Sys))
11 disp(Sys)
```

---

Scilab code Exa 10.5 finding the state vector

```
1 syms t s
2 A=[-12 2/3; -36 -1]; B=[1/3; 1]; q0=[2; 1]; X=1/s;
3 size(A)
4 size(s*eye(2,2))
```

```

5 Q=inv(s*eye(2,2)-A)*(q0+B*X);
6 q=[];
7 q(1)=ilaplace(Q(1));
8 q(2)=ilaplace(Q(2));
9 disp(q*'u(t)', "[q1(t) ; q2(t)]")

```

---

**Scilab code Exa 10.6** state space description by transfer function

```

1 A=[0 1;-2 -3];
2 B=[1 0;1 1];
3 C=[1 0;1 1;0 2];
4 D=[0 0;1 0; 0 1];
5 syms s;
6 H=C*inv(s*eye(2,2)-A)*B+D;
7 disp(H,"the transfer function matrix H(s)=")
8 disp(H(3,2),"the transfer function relating y3 and
   x2 is H32(s)=")

```

---

**Scilab code Exa 10.7** time domain method

```

1 //signals and systems
2 //state space
3 //time domain method to find the state vector
4 clc;
5 clf;
6 s=poly(0,'s');
7 A=[s+12 -2/3; 36 s+1];
8 y=roots(det(A))
9 t=poly(0,'t');
10 beta=inv([1 y(1); 1 y(2)])*[%e^-y(1)*t; %e^-y(2)*t];
11 disp(beta)
12 size(beta)
13 W=beta(1)*[1 0;0 1]+ beta(2)*[-12 2/3;-36 -1];

```



```

14 zir=W*[2;1];
15 disp(zir);
16 zsr=W*[1/3;1];
17 disp(zsr);
18 total=zir+zsr;
19 disp(total);

```

---

Scilab code Exa 10.8 state space description by transfer function

```

1 syms t s;
2 F1=ilaplace((s+3)/((s+1)*(s+2)))
3 F2=ilaplace(1/((s+1)*(s+2)))
4 F3=ilaplace(-2/((s+1)*(s+2)))
5 F4=ilaplace(s/((s+1)*(s+2)))
6 F=[F1 F2;F3 F4];
7 disp(F,"f(t)=")
8 A=[1 0;1 1;0 2];
9 B=[0 0;1 0;0 1];
10 h=A*F*[1 0;1 1]+B*eye(2,2); //here 1 represents del(t
)
11 disp(h,"h(t)=")

```

---

Scilab code Exa 10.9 state equations of a given systems

```

1 A=[0 1;-2 -3];
2 B=[1;2];
3 P=[1 1;1 -1];
4 Ahat= P*A*inv(P)
5 Bhat=P*B
6 disp(Ahat,"A^=")
7 disp(Bhat,"B^=")

```

---

Scilab code Exa 10.10 diagonalized form of state equation

```
1 A=[0 1;-2 -3];
2 P = [2 1; 1 1]
3 [V,lambda]=spec(A);
4 B=[1;2];
5 Bhat=P*B
6 disp(P,"P=")
7 disp(Bhat,"B^=")
8 disp(lambda,"lambda=")
```

---

Scilab code Exa 10.11 controllability and observability

```
1 A=[1 0;1 -1];
2 [V,lambda]=spec(A);
3 B=[1;0];
4 C=[1 -2];
5 P=inv(V);
6 Bhat=P*B
7 Chat=C*inv(P)
8 disp(' (a): ')
9 disp(Bhat,"B^=")
10 disp(Chat,"C^=")
11 A=[-1 0;-2 1];
12 [V,lambda]=spec(A);
13 B=[1;1];
14 C=[0 1];
15 P=inv(V);
16 Bhat=P*B
17 Chat=C*inv(P)
18 disp(' Part (b): ')
19 disp(Bhat,"B^=")
```

```
20 disp(Chat, 'c^=')
```

---

Scilab code Exa 10.12 state space description of a given description

```
1 A=[0 1;-1/6 5/6];
2 B=[0;1];
3 C=[-1 5];
4 D=0;
5 sys=syslin('d',A,B,C,D);
6 N=25;
7 x=ones(1,N+1);n=(0:N);
8 q0=[2;3];
9 [y q]=csim('step',n,sys);
10 y=dsimul(sys,x);
11 plot2d3(y)
```

---

Scilab code Exa 10.13 total response using z transform

```
1 //LTi Systems characterized by Linear Constant
2 //Inverse Z Transform
3 //z = %z;
4 syms n z;
5 H1 = (-2*z)/(z-(1/3));
6 H2 = (3*z)/(z-0.5);
7 H3 = (24*z)/(z-1);
8 F1 = H1*z^(n-1)*(z-(1/3));
9 F2 = H2*z^(n-1)*(z-0.5);
10 F3 = H3*z^(n-1)*(z-1);
11 h1 = limit(F1,z,(1/3));
12 disp(h1, 'h1[n]=')
13 h2 = limit(F2,z,0.5);
14 disp(h2, 'h2[n]=')
15 h3 = limit(F3,z,1);
```

```
16 disp(h3, 'h3[n]= ')  
17 h = h1+h2+h3;  
18 disp(h, 'h[n]= ')
```

---