

Scilab Textbook Companion for
Signals And Systems
by A. V. Oppenheim, A. S. Willsky And S. H.
Nawab¹

Created by
Prof. R. Senthilkumar
B.Tech
Electrical Engineering
Institute of Road and Transport Technology
College Teacher
Na
Cross-Checked by
Prof. Saravanan Vijayakumaran

July 19, 2017

¹Funded 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: Signals And Systems

Author: A. V. Oppenheim, A. S. Willsky And S. H. Nawab

Publisher: PHI Learning, New Delhi

Edition: 2

Year: 1992

ISBN: 978-81-203-1246-3

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	5
2 Linear Time Invariant Systems	20
3 Fourier Series Representation of Periodic Signals	28
4 The Cotntinuous Time Fourier Transform	48
5 The Discreet Time Fourier Transform	62
6 Time and Frequency Characterization of Signals and Systems	73
7 Sampling	78
9 The Laplace Transform	84
10 The Z Transform	98
11 Linear Feedback Systems	110

List of Scilab Codes

Exa 1.1	Time Shifting	5
Exa 1.2	Time Scaling	6
Exa 1.3	Time Scaling and Time Shifting	7
Exa 1.4	Combination two periodic signals	8
Exa 1.6	Fundamental period of composite discrete time signal	8
Exa 1.12	Classification of system	9
Exa 01.13	Determination of stability of a given system	10
Exa 1.13	Determination of stability of a given system	11
Exa 1.14	Time Invariance Property	12
Exa 0.15	Sum of two complex exponentials	13
Exa 1.15	Classification of a System	13
Exa 1.16	Time Invariance Property	14
Exa 1.17	Linearity Property	15
Exa 1.18	Linearity Property	16
Exa 1.20	Linearity Property	17
Exa 2.1	Linear Convolution Sum	20
Exa 2.3	Convolution of $x[n]$ and Unit Impulse response $h[n]$	21
Exa 2.4	Convolution Sum of finite duration sequences	22
Exa 2.5	Convolution Sum of input sequence	23
Exa 2.6	Convolution Integral of input	24
Exa 2.7	Convolution Integral of finite duration signals	25
Exa 2.8	Convolution Integral of input	26
Exa 3.2	CTFS of a periodic signal $x(t)$	28
Exa 3.3	Continuous Time Fourier Series Coefficients	30
Exa 3.4	CTFS coefficients of a periodic signal	31
Exa 3.5	CTFS coefficients of a periodic signal	32

Exa 3.6	Time Shift Property of CTFS	34
Exa 3.7	Derivative Property of CTFS	35
Exa 3.8	Fourier Series Representation of Periodic Im- pulse Train	37
Exa 3.10	DTFS of $x[n]$	39
Exa 3.11	DTFS of $x[n]$	40
Exa 3.12	DTFS coefficients of periodic square wave	42
Exa 3.13	Periodic sequence	43
Exa 3.14	Parseval's relation of DTFS	45
Exa 3.15	DTFS:Periodic Convolution Property	46
Exa 4.1	clear	48
Exa 4.2	clear	50
Exa 4.4	clear	51
Exa 4.5	clear	52
Exa 4.6	clear	53
Exa 4.7	clear	54
Exa 4.8	clear	55
Exa 4.9	clear	56
Exa 4.12	clear	57
Exa 4.18	clear	58
Exa 4.22	clear	59
Exa 4.23	clear	60
Exa 5.1	Discrete Time Fourier Transform of discrete sequence	62
Exa 5.2	Discrete Time Fourier Transform	64
Exa 5.3	Discrete Time Fourier Transform	65
Exa 5.5	Time Fourier Transform: $x[n]= \cos(n\omega_0)$	66
Exa 5.6	Discrete Time Fourier Transform	67
Exa 5.7	Frequency Shifting Property of DTFT	68
Exa 5.9	Expansion Property of DTFT	69
Exa 5.12	IDTFT:Impulse Response of Ideal Low pass Filter	70
Exa 5.15	Multiplication Property of DTFT	71
Exa 6.1	Phase Response and Group Delay	73
Exa 6.3	Analog Lowpass IIR filter design	76
Exa 6.4	Bode Plot	77
Exa 6.5	Bode Plot	77
Exa 7.1	Sinusoidal signal	78

Exa 7.2	Digital Differentiator	79
Exa 7.3	Half Sample Delay system	79
Exa 7.4	Period of the sampled signal and Sampling frequency	80
Exa 7.5	Multirate Signal Processing	81
Exa 9.1	Lapalce Transform $x(t)$	84
Exa 9.2	Lapalce Transform $x(t)$	84
Exa 9.3	Lapalce Transform $x(t)$	85
Exa 9.4	clear	85
Exa 9.5	clear	85
Exa 9.6	clear	86
Exa 9.7	clear	86
Exa 9.8	clear	87
Exa 9.9	clear	87
Exa 9.10	Inverse Lapalce Transform	88
Exa 9.11	Inverse Lapalce Transform	88
Exa 9.12	Inverse Lapalce Transform	89
Exa 9.13	Inverse Lapalce Transform	89
Exa 9.14	Lapalce Transform	90
Exa 9.15	Inverse Lapalce Transform	90
Exa 9.16	Initial Value Theorem of Lapalce Transform	91
Exa 9.17	Analysis and Characterization of LTI System	91
Exa 9.18	Analysis and Characterization of LTI System	92
Exa 9.19	Analysis and Characterization of LTI System	92
Exa 9.20	Inverse Lapalce Transform	92
Exa 9.21	Analysis and Characterization of LTI System	93
Exa 9.25	Finding Transfer function $H(S)$ of LTI system	93
Exa 9.31	Partial Fraction	94
Exa 9.33	Unilateral Laplace Transform	95
Exa 9.34	Unilateral Laplace Transform	95
Exa 9.35	clear	95
Exa 9.36	clear	96
Exa 9.37	clear	96
Exa 9.38	clear	97
Exa 10.1	Ztransform of $x[n]$	98
Exa 10.2	Z transform of $x[n] = -a^n \cdot u[-n - 1]$	98
Exa 10.3	Z transform of $x[n]$	99
Exa 10.4	Z-transform of sine signal	99

Exa 10.5	Z-transform of Impulse Sequence	100
Exa 10.6	Z transform of $x[n]$	100
Exa 10.7	Z transform of $x[n]$	101
Exa 10.9	clear	101
Exa 10.10	Inverse Z Transform	102
Exa 10.11	Inverse Z Transform	102
Exa 10.12	Inverse z tranform	103
Exa 10.13	Inverse z tranform of InFinite duration discrete sequence	104
Exa 10.18	Ztransform-Differentiation Property	104
Exa 10.19	Z Transform : Initial Value Theorem	104
Exa 10.23	Inverse Z Transform $H(z) = z/z-a$	105
Exa 10.25	Coefficient Difference equations	106
Exa 10.33	Differentiation Property of Unilateral Ztransform	106
Exa 10.34	Unilateral Ztransform- partial fraction	107
Exa 10.36	Output response of an LTI System	108
Exa 10.37	Output response of an LTI System	108
Exa 11.1	Root locus Analysis of Linear Feedback Systems	110
Exa 11.2	Continuous Time Systems	110
Exa 11.3	Discrete time system	111
Exa 11.05	Nyquist criterion for Continuous Time Systems	111
Exa 11.5	Bode Plot	112
Exa 11.6	Nyquist Plot	112
Exa 11.7	Nyquist Plot	113
Exa 11.8	Nyquist Plot	113
Exa 11.09	Root locus analysis of Linear feedback systems	113
Exa 11.9	Gain and Phase Margins	114

Chapter 1

Signals and Systems

Scilab code Exa 1.1 Time Shifting

```
1 //clear//
2 //Example 1.1: Time Shifting
3 //SIGNALS & SYSTEMS, Second Edition
4 //V.OPPENHEIM, S.WILLSKY, S.HAMID NAMWAB
5 //PHI, 2008 Edition
6 //Page 10
7 clear;
8 clc;
9 close;
10 t = 0:1/100:1;
11 for i = 1:length(t)
12     x(i) = 1 ;
13 end
14 for i = length(t)+1:2*length(t)
15     x(i) = 1-t(i-length(t));
16 end
17 t1 = 0:1/100:2;
18 t2 = -1:1/100:1;
19 //t3 = 0:1/100:4/3;
20 //t4 = 0:1/length(t3):1;
21 //Mid =ceil(length(t3)/2);
```

```

22 //for i = 1:Mid
23 //  x3(i) = 1 ;
24 //end
25 //for i = Mid+1:length(t3)
26 //  x3(i) = 1-t4(i-Mid);
27 //end
28 figure
29 a=gca();
30 plot2d(t1,x(1:$-1))
31 a.thickness=2;
32 xtitle('The signal x(t)')
33 figure
34 a=gca();
35 plot2d(t2,x(1:$-1))
36 a.thickness=2;
37 a.y_location = "middle";
38 xtitle('The signal x(t+1)')
39 figure
40 a=gca();
41 plot2d(t2,x($:-1:2))
42 a.thickness=2;
43 a.y_location = "middle";
44 xtitle('The signal x(-t+1)')

```

Scilab code Exa 1.2 Time Scaling

```

1 //clear//
2 //Example 1.2:Time Scaling
3 //SIGNALS & SYSTEMS, Second Edition
4 //V.OPPENHEIM, S.WILLSKY, S.HAMID NAMWAB
5 //PHI, 2008 Edition
6 //Page 11
7 clear;
8 clc;
9 close;

```

```

10 t3 = 0:1/100:4/3;
11 t4 = 0:1/length(t3):1;
12 Mid =ceil(length(t3)/2);
13 for i = 1:Mid
14     x3(i) = 1 ;
15 end
16 for i = Mid+1:length(t3)
17     x3(i) = 1-t4(i-Mid);
18 end
19 figure
20 a=gca();
21 plot2d(t3,x3)
22 a.thickness=2;
23 xtitle('Time Scaling x(3t/2)')

```

Scilab code Exa 1.3 Time Scaling and Time Shifting

```

1 //clear//
2 //Example 1.3:Time Scaling and Time Shifting
3 //SIGNALS & SYSTEMS, Second Edition
4 //V.OPPENHEIM, S.WILLSKY, S.HAMID NAMWAB
5 //PHI, 2008 Edition
6 //Page 11
7 clear;
8 clc;
9 close;
10 t3 = 0:1/100:4/3;
11 t4 = 0:1/length(t3):1;
12 Mid =ceil(length(t3)/2);
13 for i = 1:Mid
14     x3(i) = 1 ;
15 end
16 for i = Mid+1:length(t3)
17     x3(i) = 1-t4(i-Mid);
18 end

```

```

19 t5 = -2/3:1/100:2/3;
20 figure
21 a=gca();
22 plot2d(t5,x3)
23 a.thickness=2;
24 a.y_location = "middle";
25 xtitle('Time Scaling and Time Shifting  $x((3t/2)+1)$ ')

```

Scilab code Exa 1.4 Combination two periodic signals

```

1 //clear//
2 //Example 1.4: Combination two periodic signals
3 // Aperiodic signal
4 //Page 12
5 clear;
6 clc;
7 close;
8 F=1; //Frequency = 1 Hz
9 t1 = 0:-1/100:-2*pi;
10 x1 = cos(F*t1);
11 t2 = 0:1/100:2*pi;
12 x2 = sin(F*t2);
13 a=gca();
14 plot(t2,x2);
15 plot(t1,x1);
16 a.y_location = "middle";
17 a.x_location = "middle";
18 xtitle('The signal  $x(t) = \cos t$  for  $t < 0$  and  $\sin t$ 
for  $t > 0$ : Aperiodic Signal')

```

Scilab code Exa 1.6 Fundamental period of composite discrete time signal

```

1 //clear//

```

```

2 //Example 1.6:Determine the fundamental period of
  composite
3 // discrete time signal
4 //x[n] = exp(j(2*%pi/3)n)+exp(j(3*%pi/4)n)
5 clear;
6 clc;
7 close;
8 Omega1 = 2*%pi/3; //Angular frequency signal 1
9 Omega2 = 3*%pi/4; //Angular frequency signal 2
10 N1 = (2*%pi)/Omega1; //Peirod of signal 1
11 N2 = (2*%pi)/Omega2; //Period of signal 2
12 //To find rational period of signal 1
13 for m1 = 1:100
14     period = N1*m1;
15     if(modulo(period,1)==0)
16         period1 = period;
17         integer_value = m1
18         break;
19     end
20 end
21 //To find rational period of signal 2
22 for m2 = 1:100
23     period = N2*m2;
24     if(modulo(period,1)==0)
25         period2 = period;
26         integer_value = m2
27         break;
28     end
29 end
30 disp(period1)
31 disp(period2)
32 //To determine the fundamental period N
33 N = period1*period2

```

Scilab code Exa 1.12 Classification of system

```

1 //clear//
2 //Example 1.12: Classification of system: Causality
  property
3 //Page 47
4 //To check whether the given discrete system is a
  Causal System (or) Non-Causal System
5 //Given discrete system  $y[n]=x[-n]$ 
6 clear;
7 clc;
8 x = [2,4,6,8,10,0,0,0,1]; //Assign some value to
  input
9 n = -length(x)/2:length(x)/2;
10 count = 0;
11 mid = ceil(length(x)/2);
12 y = zeros(1,length(x));
13 y(mid+1:$) = x($:-1:mid+1);
14 for n = -1:-1:-mid
15   y(n+1+mid) = x(-n);
16 end
17 for i = 1:length(x)
18   if (y(i)==x(i))
19     count = count+1;
20   end
21 end
22 if (count==length(x))
23   disp('The given system is a causal system')
24 else
25   disp('Since it depends on future input value')
26   disp('The given system is a non-causal system')
27 end

```

Scilab code Exa 01.13 Determination of stability of a given system

```

1 //clear//
2 //Example 1.13(b): Determination of stability of a

```

```

        given system
3 //Page 50
4 //given system  $y(t) = \exp(x(t))$ 
5 clear;
6 clc;
7 Maximum_Limit = 10;
8 S = 0;
9 for t = 0:Maximum_Limit-1
10     x(t+1)= -2^t;           //Input some bounded value
11     S = S+exp(x(t+1));
12 end
13 if (S >Maximum_Limit)
14     disp('Eventhough input is bounded output is
           unbounded')
15     disp('The given system is unstable');
16     disp('S =');
17     S
18 else
19     disp('The given system is stable');
20     disp(S);
21 end

```

Scilab code Exa 1.13 Determination of stability of a given system

```

1 //clear//
2 //Example 1.13:Determination of stability of a
   given system
3 //Page 49
4 //given system  $y(t) = t.x(t)$ 
5 clear;
6 clc;
7 x = [1,2,3,4,0,2,1,3,5,8]; //Assign some input
8 Maximum_Limit = 10;
9 S = 0;
10 for t = 0:Maximum_Limit-1

```

```

11   S = S+t*x(t+1);
12   end
13   if (S >Maximum_Limit)
14     disp('Eventhough input is bounded output is
           unbounded')
15     disp('The given system is unstable');
16     disp('S =');
17     S
18   else
19     disp('The given system is stable');
20     disp('The value of S =');
21     S
22   end

```

Scilab code Exa 1.14 Time Invariance Property

```

1 //clear//
2 //Example 1.14:classification of a system:Time
  Invariance Property
3 //Page 51
4 //To check whether the given system is a Time
  variant (or) Time In-variant
5 // The given discrete signal is  $y(t) = \sin(x(t))$ 
6 clear;
7 clc;
8 to = 2; //Assume the amount of time shift =2
9 T = 10; //Length of given signal
10 for t = 1:T
11   x(t) = (2*%pi/T)*t;
12   y(t) = sin(x(t));
13 end
14 //First shift the input signal only
15 Input_shift = sin(x(T-to));
16 Output_shift = y(T-to);
17 if(Input_shift == Output_shift)

```



```

18     disp('The given discrete system is a Time In-
        variant system');
19 else
20     disp('The given discrete system is a Time Variant
        system');
21 end

```

Scilab code Exa 0.15 Sum of two complex exponentials

```

1 //clear//
2 //Example 1.5:To express sum of two complex
  exponentials
3 //as a single sinusoid
4 clear;
5 clc;
6 close;
7 t =0:1/100:2*%pi;
8 x1 = exp(sqrt(-1)*2*t);
9 x2 = exp(sqrt(-1)*3*t);
10 x = x1+x2;
11 for i = 1:length(x)
12     X(i) = sqrt((real(x(i)).^2)+(imag(x(i)).^2));
13 end
14 plot(t,X);
15 xtitle('Full wave rectified sinusoid','time t','
        Magnitude');

```

Scilab code Exa 1.15 Classification of a System

```

1 //clear//
2 //Example 1.15: Classification of a System:Time
  Invariance Property
3 //Page 51

```

```

4 //To check whether the given system is a Time
    variant (or) Time In-variant
5 // The given discrete signal is  $y[n] = n \cdot x[n]$ 
6 clear;
7 clc;
8 no = 2; //Assume the amount of time shift =2
9 L = 10; //Length of given signal
10 for n = 1:L
11     x(n) = n;
12     y(n) = n*x(n);
13 end
14 //First shift the input signal only
15 Input_shift = x(L-no);
16 Output_shift = y(L-no);
17 if(Input_shift == Output_shift)
18     disp('The given discrete system is a Time In-
        variant system');
19 else
20     disp('The given discrete system is a Time Variant
        system');
21 end

```

Scilab code Exa 1.16 Time Invariance Property

```

1 //clear//
2 //Example 1.16: Classification of system:Time
    Invariance Property
3 //Page 52
4 //To check whether the given system is a Time
    variant (or) Time In-variant
5 // The given discrete signal is  $y(t) = x(2t)$ 
6 clear;
7 clc;
8 to = 2; //Assume the amount of time shift =2
9 T = 10; //Length of given signal

```

```

10 x = [1,2,3,4,5,6,7,8,9,10];
11 y = zeros(1,length(x));
12 for t = 1:length(x)/2
13     y(t) = x(2*t);
14 end
15 //First shift the input signal only
16 Input_shift = x(T-to);
17 Output_shift = y(T-to);
18 if(Input_shift == Output_shift)
19     disp('The given discrete system is a Time In-
        variant system');
20 else
21     disp('The given discrete system is a Time Variant
        system');
22 end

```

Scilab code Exa 1.17 Linearity Property

```

1 //clear//
2 //Example 1.17: Classification of system:Linearity
  Property
3 //Page 54
4 //To check whether the given discrete system is a
  Linear System (or) Non-Linear System
5 //Given discrete system  $y(t) = t*x(t)$ 
6 clear;
7 clc;
8 x1 = [1,1,1,1];
9 x2 = [2,2,2,2];
10 a = 1;
11 b = 1;
12 for t = 1:length(x1)
13     x3(t) = a*x1(t)+b*x2(t);
14 end
15 for t = 1:length(x1)

```

```

16   y1(t) = t*x1(t);
17   y2(t) = t*x2(t);
18   y3(t) = t*x3(t);
19   end
20   for t = 1:length(y1)
21     z(t) = a*y1(t)+b*y2(t);
22   end
23   count = 0;
24   for n =1:length(y1)
25     if(y3(t)== z(t))
26       count = count+1;
27     end
28   end
29   if(count == length(y3))
30     disp('Since It satisfies the superposition
          principle')
31     disp('The given system is a Linear system')
32     y3
33     z
34   else
35     disp('Since It does not satisfy the
          superposition principle')
36     disp('The given system is a Non-Linear system')
37   end

```

Scilab code Exa 1.18 Linearity Property

```

1 //clear//
2 //Example 1.18: Classsification of a system:Linearity
  Property
3 //Page 54
4 //To check whether the given discrete system is a
  Linear System (or) Non-Linear System
5 //Given discrete system  $y(t) = (x(t))^2$ 
6 clear;

```

```

7  clc;
8  x1 = [1,1,1,1];
9  x2 = [2,2,2,2];
10 a = 1;
11 b = 1;
12 for t = 1:length(x1)
13     x3(t) = a*x1(t)+b*x2(t);
14 end
15 for t = 1:length(x1)
16     y1(t) = (x1(t)^2);
17     y2(t) = (x2(t)^2);
18     y3(t) = (x3(t)^2);
19 end
20 for t = 1:length(y1)
21     z(t) = a*y1(t)+b*y2(t);
22 end
23 count = 0;
24 for n =1:length(y1)
25     if(y3(t)== z(t))
26         count = count+1;
27     end
28 end
29 if(count == length(y3))
30     disp('Since It satisfies the superposition
31         principle ')
32     disp('The given system is a Linear system')
33     y3
34     z
35 else
36     disp('Since It does not satisfy the
37         superposition principle ')
38     disp('The given system is a Non-Linear system')
39 end

```

Scilab code Exa 1.20 Linearity Property

```

1 //clear//
2 //Example 1.20: Classification of a system:Linearity
   Property
3 //Page 55
4 //To check whether the given discrete system is a
   Linear System (or) Non-Linear System
5 //Given discrete system  $y[n]= 2*x[n]+3$ 
6 clear;
7 clc;
8 x1 = [1,1,1,1];
9 x2 = [2,2,2,2];
10 a = 1;
11 b = 1;
12 for n = 1:length(x1)
13     x3(n) = a*x1(n)+b*x2(n);
14 end
15 for n = 1:length(x1)
16     y1(n) = 2*x1(n)+3;
17     y2(n) = 2*x2(n)+3;
18     y3(n) = 2*x3(n)+3;
19 end
20 for n = 1:length(y1)
21     z(n) = a*y1(n)+b*y2(n);
22 end
23 count = 0;
24 for n =1:length(y1)
25     if(y3(n)== z(n))
26         count = count+1;
27     end
28 end
29 if(count == length(y3))
30     disp('Since It satisfies the superposition
        principle')
31     disp('The given system is a Linear system')
32     y3
33     z
34 else
35     disp('Since It does not satisfy the

```

```
        superposition principle')
36     disp('The given system is a Non-Linear system')
37 end
```

Chapter 2

Linear Time Invariant Systems

Scilab code Exa 2.1 Linear Convolution Sum

```
1 //clear//
2 //Example 2.1:Linear Convolution Sum
3 //page 80
4 clear;
5 close;
6 clc;
7 h = [0,0,1,1,1,0,0];
8 N1 = -2:4;
9 x = [0,0,0.5,2,0,0,0];
10 N2 = -2:4;
11 y = convol(x,h);
12 for i = 1:length(y)
13     if (y(i)<=0.0001)
14         y(i)=0;
15     end
16 end
17 N = -4:8;
18 figure
19 a=gca();
20 plot2d3('gnn',N1,h)
21 xtitle('Impulse Response','n','h[n]');
```



```

22 a.thickness = 2;
23 figure
24 a=gca();
25 plot2d3('gmn',N2,x)
26 xtitle('Input Response','n','x[n]');
27 a.thickness = 2;
28 figure
29 a=gca();
30 plot2d3('gmn',N,y)
31 xtitle('Output Response','n','y[n]');
32 a.thickness = 2;

```

Scilab code Exa 2.3 Convolution of $x[n]$ and Unit Impulse response $h[n]$

```

1 //clear//
2 //Example 2.3: Convolution Sum: Convolution of x[n]
   and
3 //Unit Impulse response h[n]
4 clear;
5 close;
6 clc;
7 Max_Limit = 10;
8 h = ones(1,Max_Limit);
9 N1 = 0:Max_Limit-1;
10 Alpha = 0.5; //alpha < 1
11 for n = 1:Max_Limit
12     x(n) = (Alpha^(n-1))*1;
13 end
14 N2 = 0:Max_Limit-1;
15 y = convol(x,h);
16 N = 0:2*Max_Limit-2;
17 figure
18 a=gca();
19 plot2d3('gmn',N1,h)
20 xtitle('Impulse Response Fig 2.5.(b)','n','h[n]');

```

```

21 a.thickness = 2;
22 figure
23 a=gca();
24 plot2d3('gmn',N2,x)
25 xtitle('Input Response Fig 2.5.(a)', 'n', 'x[n]');
26 a.thickness = 2;
27 figure
28 a=gca();
29 plot2d3('gmn',N(1:Max_Limit),y(1:Max_Limit),5)
30 xtitle('Output Response Fig 2.7', 'n', 'y[n]');
31 a.thickness = 2;

```

Scilab code Exa 2.4 Convolution Sum of finite duration sequences

```

1 //clear//
2 //Example 2.4: Convolution Sum of finite duration
   sequences
3 clear;
4 close;
5 clc;
6 x = ones(1,5);
7 N1 =0:length(x)-1;
8 Alpha = 1.4; //alpha > 1
9 for n = 1:7
10   h(n)= (Alpha^(n-1))*1;
11 end
12 N2 =0:length(h)-1;
13 y = convol(x,h);
14 N = 0:length(x)+length(h)-2;
15 figure
16 a=gca();
17 plot2d3('gmn',N2,h)
18 xtitle('Impulse Response', 'n', 'h[n]');
19 a.thickness = 2;
20 figure

```

```

21 a=gca();
22 plot2d3('gmn',N1,x)
23 xtitle('Input Response','n','x[n]');
24 a.thickness = 2;
25 figure
26 a=gca();
27 plot2d3('gmn',N,y)
28 xtitle('Output Response','n','y[n]');
29 a.thickness = 2;

```

Scilab code Exa 2.5 Convolution Sum of input sequence

```

1 //clear//
2 //Example 2.5: Convolution Sum of input sequence x[n
   ]=(2^n).u[-n]
3 //and h[n] = u[n]
4 clear;
5 close;
6 clc;
7 Max_Limit = 10;
8 h = ones(1,Max_Limit);
9 N2 =0:length(h)-1;
10 for n = 1:Max_Limit
11     x1(n)= (2^(-(n-1)))*1;
12 end
13 x = x1($:-1:1);
14 N1 = -length(x)+1:0;
15 y = convol(x,h);
16 N = -length(x)+1:length(h)-1;
17 figure
18 a=gca();
19 plot2d3('gmn',N2,h)
20 xtitle('Impulse Response','n','h[n]');
21 a.thickness = 2;
22 figure

```

```

23 a=gca();
24 a.y_location = "origin";
25 plot2d3('gmn',N1,x)
26 xtitle('Input Response Fig 2.11(a)', 'n', 'x[n]');
27 a.thickness = 2;
28 figure
29 a=gca();
30 a.y_location = "origin";
31 plot2d3('gmn',N,y)
32 xtitle('Output Response Fig 2.11(b)', 'n', 'y[n]');
33 a.thickness = 2;

```

Scilab code Exa 2.6 Convolution Integral of input

```

1 //clear//
2 //Example 2.6: Convolution Integral of input  $x(t) = ($ 
    $e^{-at}).u(t)$ 
3 //and  $h(t) = u(t)$ 
4 clear;
5 close;
6 clc;
7 Max_Limit = 10;
8 h = ones(1,Max_Limit);
9 N2 = 0:length(h)-1;
10 a = 0.5; //constant  $a > 0$ 
11 for t = 1:Max_Limit
12     x(t) = exp(-a*(t-1));
13 end
14 N1 = 0:length(x)-1;
15 y = convol(x,h)-1;
16 N = 0:length(x)+length(h)-2;
17 figure
18 a=gca();
19 plot2d(N2,h)
20 xtitle('Impulse Response', 't', 'h(t)');

```

```

21 a.thickness = 2;
22 figure
23 a=gca();
24 plot2d(N1,x)
25 xtitle('Input Response','t','x(t)');
26 a.thickness = 2;
27 figure
28 a=gca();
29 plot2d(N(1:Max_Limit),y(1:Max_Limit))
30 xtitle('Output Response','t','y(t)');
31 a.thickness = 2;

```

Scilab code Exa 2.7 Convolution Integral of finite duration signals

```

1 //clear//
2 //Example 2.7: Convolution Integral of finite
   duration signals
3 //page99
4 clear;
5 close;
6 clc;
7 T = 10;
8 x = ones(1,T); //Input Response
9 for t = 1:2*T
10   h(t) = t-1; //Impulse Response
11 end
12 N1 = 0:length(x)-1;
13 N2 = 0:length(h)-1;
14 y = convol(x,h);
15 N = 0:length(x)+length(h)-2;
16 figure
17 a=gca();
18 a.x_location="origin";
19 plot2d(N2,h)
20 xtitle('Impulse Response','t','h(t)');

```

```

21 a.thickness = 2;
22 figure
23 a=gca();
24 plot2d(N1,x)
25 xtitle('Input Response','t','x(t)');
26 a.thickness = 2;
27 figure
28 a=gca();
29 plot2d(N,y)
30 xtitle('Output Response','t','y(t)');
31 a.thickness = 2;

```

Scilab code Exa 2.8 Convolution Integral of input

```

1 //clear//
2 //Example 2.8: Convolution Integral of input  $x(t)=(e^{2t}).u(-t)$  and
    $h(t) = u(t-3)$ 
3 //h(t) = u(t-3)
4 clear;
5 close;
6 clc;
7 Max_Limit = 10;
8 h = [0,0,0,ones(1,Max_Limit-3)]; //h(n-3)
9 a = 2;
10 t = -9:0;
11 x= exp(a*t);
12 //x = x1($:-1:1)
13 N2 = 0:length(h)-1;
14 N1 = -length(x)+1:0;
15 t1 = -6:3;
16 y1 = (1/a)*exp(a*(t1-3));
17 y2 = (1/a)*ones(1,Max_Limit);
18 y = [y1 y2]
19 N = -length(h)+1:length(x)-1;
20 figure

```

```
21 a=gca();
22 a.x_location="origin";
23 a.y_location="origin";
24 plot2d(-Max_Limit+1:0,h($:-1:1))
25 xtitle('Impulse Response','t','h(t-T)');
26 a.thickness = 2;
27 figure
28 a=gca();
29 a.y_location = "origin";
30 plot2d(t,x)
31 xtitle('Input Response','t','x(t)');
32 a.thickness = 2;
33 figure
34 a=gca();
35 a.y_location = "origin";
36 a.x_location = "origin";
37 a.data_bounds=[-10,0;13,1];
38 plot2d(-Max_Limit+4:Max_Limit+3,y)
39 xtitle('Output Response','t','y(t)');
40 a.thickness = 2;
```

Chapter 3

Fourier Series Representation of Periodic Signals

Scilab code Exa 3.2 CTFS of a periodic signal $x(t)$

```
1 //clear//
2 //Example 3.2:CTFS of a periodic signal x(t)
3 //Expression of continuous time signal
4 //using continuous time fourier series
5 clear;
6 close;
7 clc;
8 t = -3:0.01:3;
9 //t1 = -%pi*4:(%pi*4)/100:%pi*4;
10 //t2 =-%pi*6:(%pi*6)/100:%pi*6;
11 xot = ones(1,length(t));
12 x1t = (1/2)*cos(%pi*2*t);
13 xot_x1t = xot+x1t;
14 x2t = cos(%pi*4*t);
15 xot_x1t_x2t = xot+x1t+x2t;
16 x3t = (2/3)*cos(%pi*6*t);
17 xt = xot+x1t+x2t+x3t;
18 //
19 figure
```



```

20 a = gca();
21 a.y_location = "origin";
22 a.x_location = "origin";
23 a.data_bounds=[-4,0;2 4];
24 plot(t,xot)
25 ylabel('t')
26 title('xot =1')
27 //
28 figure
29 subplot(2,1,1)
30 a = gca();
31 a.y_location = "origin";
32 a.x_location = "origin";
33 a.data_bounds=[-4,-3;2 4];
34 plot(t,x1t)
35 ylabel('t')
36 title('x1(t) =1/2*cos(2*pi*t)')
37 subplot(2,1,2)
38 a = gca();
39 a.y_location = "origin";
40 a.x_location = "origin";
41 a.data_bounds=[-4,0;2 4];
42 plot(t,xot_x1t)
43 ylabel('t')
44 title('xo(t)+x1(t)')
45 //
46 figure
47 subplot(2,1,1)
48 a = gca();
49 a.y_location = "origin";
50 a.x_location = "origin";
51 a.data_bounds=[-4,-2;4 2];
52 plot(t,x2t)
53 ylabel('t')
54 title('x2(t) =cos(4*pi*t)')
55 subplot(2,1,2)
56 a = gca();
57 a.y_location = "origin";

```

```

58 a.x_location = "origin";
59 a.data_bounds=[-4,0;4 4];
60 plot(t,xot_x1t_x2t)
61 ylabel('t')
62 title('xo(t)+x1(t)+x2(t)')
63 //
64 figure
65 subplot(2,1,1)
66 a = gca();
67 a.y_location = "origin";
68 a.x_location = "origin";
69 a.data_bounds=[-4,-3;4 3];
70 plot(t,x3t)
71 ylabel('t')
72 title('x1(t) =2/3*cos(6*pi*t)')
73 subplot(2,1,2)
74 a = gca();
75 a.y_location = "origin";
76 a.x_location = "origin";
77 a.data_bounds=[-4,-3;4 3];
78 plot(t,xt)
79 ylabel('t')
80 title('x(t)=xo(t)+x1(t)+x2(t)+x3(t)')

```

Scilab code Exa 3.3 Continuous Time Fourier Series Coefficients

```

1 //clear//
2 //Example3.3:Continuous Time Fourier Series
   Coefficients of
3 //a periodic signal  $x(t) = \sin(\omega_0 t)$ 
4 clear;
5 close;
6 clc;
7 t = 0:0.01:1;
8 T = 1;

```

```

9  Wo = 2*%pi/T;
10 xt = sin(Wo*t);
11 for k =0:5
12     C(k+1,:) = exp(-sqrt(-1)*Wo*t.*k);
13     a(k+1) = xt*C(k+1,:)'/length(t);
14     if(abs(a(k+1)))<=0.01)
15         a(k+1)=0;
16     end
17 end
18 a =a';
19 ak = [-a,a(2:$)];

```

Scilab code Exa 3.4 CTFS coefficients of a periodic signal

```

1 //clear//
2 //Example3.4:CTFS coefficients of a periodic signal
3 //x(t) = 1+sin(Wot)+2cos(Wot)+cos(2Wot+%pi/4)
4 clear;
5 close;
6 clc;
7 t = 0:0.01:1;
8 T = 1;
9 Wo = 2*%pi/T;
10 xt =ones(1,length(t))+sin(Wo*t)+2*cos(Wo*t)+cos(2*Wo
    *t+%pi/4);
11 for k =0:5
12     C(k+1,:) = exp(-sqrt(-1)*Wo*t.*k);
13     a(k+1) = xt*C(k+1,:)'/length(t);
14     if(abs(a(k+1)))<=0.1)
15         a(k+1)=0;
16     end
17 end
18 a =a';
19 a_conj =conj(a);
20 ak = [a_conj($:-1:1),a(2:$)];

```

```

21 Mag_ak = abs(ak);
22 for i = 1:length(a)
23     Phase_ak(i) = atan(imag(ak(i))/(real(ak(i))
        +0.0001));
24 end
25 Phase_ak = Phase_ak'
26 Phase_ak = [Phase_ak(1:$) -Phase_ak($-1:-1:1)];
27 figure
28 subplot(2,1,1)
29 a = gca();
30 a.y_location = "origin";
31 a.x_location = "origin";
32 plot2d3('ggn',[-k:k],Mag_ak,5)
33 poly1 = a.children(1).children(1);
34 poly1.thickness = 3;
35 title('abs(ak)')
36 xlabel('

        k')
37 subplot(2,1,2)
38 a = gca();
39 a.y_location = "origin";
40 a.x_location = "origin";
41 plot2d3('ggn',[-k:k],Phase_ak,5)
42 poly1 = a.children(1).children(1);
43 poly1.thickness = 3;
44 title('<(ak)')
45 xlabel('

        k')

```

Scilab code Exa 3.5 CTFS coefficients of a periodic signal

```

1 //clear//
2 //Example3.5:CTFS coefficients of a periodic signal

```

```

3 //x(t) = 1, |t|<T1, and 0, T1<|t|<T/2
4 clear;
5 close;
6 clc;
7 T =4;
8 T1 = T/4;
9 t = -T1:T1/100:T1;
10 Wo = 2*%pi/T;
11 xt =ones(1,length(t));
12 //
13 for k =0:5
14     C(k+1,:) = exp(-sqrt(-1)*Wo*t.*k);
15     a(k+1) = xt*C(k+1,:)/length(t);
16     if(abs(a(k+1))<=0.1)
17         a(k+1)=0;
18     end
19 end
20 a =a';
21 a_conj = real(a(:))-sqrt(-1)*imag(a(:));
22 ak = [a_conj($:-1:1)',a(2:$)];
23 k = 0:5;
24 k = [-k($:-1:1),k(2:$)];
25 Spectrum_ak = (1/2)*real(ak);
26 //
27 figure
28 a = gca();
29 a.y_location = "origin";
30 a.x_location = "origin";
31 a.data_bounds=[-2,0;2,2];
32 plot2d(t,xt,5)
33 poly1 = a.children(1).children(1);
34 poly1.thickness = 3;
35 title('x(t)')
36 xlabel('
    t')
37 //
38 figure

```

```

39 a = gca();
40 a.y_location = "origin";
41 a.x_location = "origin";
42 plot2d3('gnn',k,Spectrum_ak,5)
43 poly1 = a.children(1).children(1);
44 poly1.thickness = 3;
45 title('abs(ak)')
46 xlabel('

```

```

    k')

```

Scilab code Exa 3.6 Time Shift Property of CTFS

```

1 //clear//
2 //Example3.6: Time Shift Property of CTFS
3 clear;
4 close;
5 clc;
6 T =4;
7 T1 = T/2;
8 t = 0:T1/100:T1;
9 Wo = 2*%pi/T;
10 gt =(1/2)*ones(1,length(t));
11 a(1)=0; //k=0, ak =0
12 d(1)=0;
13 for k =1:5
14     a(k+1) = (sin(%pi*k/2)/(k*%pi));
15     if(abs(a(k+1))<=0.01)
16         a(k+1)=0;
17     end
18     d(k+1) = a(k+1)*exp(-sqrt(-1)*k*%pi/2);
19 end
20 k = 0:5
21 disp('Fourier Series Coefficients of Square Wave')
22 a

```

```

23 disp('Fourier Series Coefficients of  $g(t)=x(t-1)-0.5$ 
    ')
24 d
25 //
26 figure
27 a = gca();
28 a.y_location = "origin";
29 a.x_location = "origin";
30 a.data_bounds=[-1,-2;1,4];
31 plot2d([-t($:-1:1),t(1:$)],[-gt,gt],5)
32 poly1 = a.children(1).children(1);
33 poly1.thickness = 3;
34 title('g(t)')
35 xlabel('
    t')

```

Scilab code Exa 3.7 Derivative Property of CTFS

```

1 //clear//
2 //Example3.7: Derivative Property of CTFS
3 clear;
4 clc;
5 close;
6 T =4;
7 T1 = T/2;
8 t = 0:T1/100:T1;
9 xt = [t($:-1:1) t]/T1;
10 gt = (1/2)*ones(1,length(t));
11 e(1) = 1/2; //k =0, e0 = 1/2
12 for k =1:5
13     a(k+1) = (sin(%pi*k/2)/(k*%pi));
14     if(abs(a(k+1))<=0.01)
15         a(k+1)=0;
16     end

```

```

17     d(k+1) = a(k+1)*exp(-sqrt(-1)*k*pi/2);
18     e(k+1) = 2*d(k+1)/(sqrt(-1)*k*pi);
19 end
20 k = 0:5
21 disp('Fourier Series Coefficients of Square Wave')
22 a
23 disp('Fourier Series Coefficients of g(t)=x(t-1)-0.5
      ')
24 d
25 disp('Fourier Series Coefficients of Triangular Wave
      ')
26 e
27 //Plotting the time shifted square waveform
28 figure
29 a = gca();
30 a.y_location = "origin";
31 a.x_location = "origin";
32 a.data_bounds=[-1,-2;1,2];
33 plot2d([-t($:-1:1),t(1:$)],[-gt,gt],5)
34 poly1 = a.children(1).children(1);
35 poly1.thickness = 3;
36 title('g(t)')
37 xlabel('
      t')
38 //Plotting the Triangular waveform
39 figure
40 a = gca();
41 a.y_location = "origin";
42 a.x_location = "origin";
43 a.data_bounds=[-1,0;1,2];
44 plot2d([-t($:-1:1),t(1:$)],xt,5)
45 poly1 = a.children(1).children(1);
46 poly1.thickness = 3;
47 title('x(t)')
48 xlabel('t')

```

Scilab code Exa 3.8 Fourier Series Representation of Periodic Impulse Train

```
1 //clear//
2 //Example3.8:Fourier Series Representation of
   Periodic Impulse Train
3 clear;
4 clc;
5 close;
6 T =4;
7 T1 = T/4;
8 t = [-T,0,T];
9 xt = [1,1,1]; //Generation of Periodic train of
   Impulses
10 t1 = -T1:T1/100:T1;
11 gt = ones(1,length(t1));//Generation of periodic
   square wave
12 t2 = [-T1,0,T1];
13 qt = [1,0,-1];//Derivative of periodic square wave
14 Wo = 2*%pi/T;
15 ak = 1/T;
16 b(1) = 0;
17 c(1) = 2*T1/T;
18 for k =1:5
19     b(k+1) = ak*(exp(sqrt(-1)*k*Wo*T1)-exp(-sqrt(-1)*k
   *Wo*T1));
20     if(abs(b(k+1)) <=0.1)
21         b(k+1) =0;
22     end
23     c(k+1) = b(k+1)/(sqrt(-1)*k*Wo);
24     if(abs(c(k+1)) <=0.1)
25         c(k+1) =0;
26     end
27 end
28 k = 0:5
```

```

29 disp('Fourier Series Coefficients of periodic Square
      Wave')
30 disp(b)
31 disp('Fourier Series Coefficients of derivative of
      periodic square wave')
32 disp(c)
33 //Plotting the periodic train of impulses
34 figure
35 subplot(3,1,1)
36 a = gca();
37 a.y_location = "origin";
38 a.x_location = "origin";
39 a.data_bounds=[-6,0;6,2];
40 plot2d3('gmn',t,xt,5)
41 poly1 = a.children(1).children(1);
42 poly1.thickness = 3;
43 title('x(t)')
44 //Plotting the periodic square waveform
45 subplot(3,1,2)
46 a = gca();
47 a.y_location = "origin";
48 a.x_location = "origin";
49 a.data_bounds=[-6,0;6,2];
50 plot2d(t1,gt,5)
51 poly1 = a.children(1).children(1);
52 poly1.thickness = 3;
53 plot2d(T+t1,gt,5)
54 poly1 = a.children(1).children(1);
55 poly1.thickness = 3;
56 plot2d(-T+t1,gt,5)
57 poly1 = a.children(1).children(1);
58 poly1.thickness = 3;
59 title('g(t)')
60 //Plotting the periodic square waveform
61 subplot(3,1,3)
62 a = gca();
63 a.y_location = "origin";
64 a.x_location = "origin";

```

```

65 a.data_bounds=[-6,-2;6,2];
66 poly1.thickness = 3;
67 plot2d3('gmn',t2,qt,5)
68 poly1 = a.children(1).children(1);
69 poly1.thickness = 3;
70 plot2d3('gmn',T+t2,qt,5)
71 poly1 = a.children(1).children(1);
72 poly1.thickness = 3;
73 plot2d3('gmn',-T+t2,qt,5)
74 poly1 = a.children(1).children(1);
75 poly1.thickness = 3;
76 title('q(t)')

```

Scilab code Exa 3.10 DTFS of $x[n]$

```

1 //clear//
2 //Example3.10:DTFS of  $x[n] = \sin(\omega_0 n)$ 
3 clear;
4 close;
5 clc;
6 n = 0:0.01:5;
7 N = 5;
8  $\omega_0 = 2*\pi/N$ ;
9 xn = sin( $\omega_0*n$ );
10 for k = 0:N-2
11     C(k+1,:) = exp(-sqrt(-1)* $\omega_0*n.*k$ );
12     a(k+1) = xn*C(k+1,:)/length(n);
13     if(abs(a(k+1))<=0.01)
14         a(k+1)=0;
15     end
16 end
17 a =a'
18 a_conj = conj(a);
19 ak = [a_conj($:-1:1),a(2:$)]
20 k = -(N-2):(N-2);

```

```

21 //
22 figure
23 a = gca();
24 a.y_location = "origin";
25 a.x_location = "origin";
26 a.data_bounds=[-8,-1;8,1];
27 poly1 = a;
28 poly1.thickness = 3;
29 plot2d3('gmn',k,-imag(ak),5)
30 poly1 = a;
31 poly1.thickness = 3;
32 plot2d3('gmn',N+k,-imag(ak),5)
33 poly1 = a.children(1).children(1);
34 poly1.thickness = 3;
35 plot2d3('gmn',-(N+k),-imag(ak($:-1:1)),5)
36 poly1 = a;
37 poly1.thickness = 3;
38 title('ak')

```

Scilab code Exa 3.11 DTFS of $x[n]$

```

1 //clear//
2 //Example3.11:DTFS of
3 //x[n] = 1+sin(2*%pi/N)n+3cos(2*%pi/N)n+cos[(4*%pi/N
   )n+%pi/2]
4 clear;
5 close;
6 clc;
7 N = 10;
8 n = 0:0.01:N;
9 Wo = 2*%pi/N;
10 xn =ones(1,length(n))+sin(Wo*n)+3*cos(Wo*n)+cos(2*Wo
   *n+%pi/2);
11 for k = 0:N-2
12     C(k+1,:) = exp(-sqrt(-1)*Wo*n.*k);

```

```

13     a(k+1) = xn*C(k+1,:)'/length(n);
14     if(abs(a(k+1))<=0.1)
15         a(k+1)=0;
16     end
17 end
18 a =a';
19 a_conj =conj(a);
20 ak = [a_conj($:-1:1),a(2:$)];
21 Mag_ak = abs(ak);
22 for i = 1:length(a)
23     Phase_ak(i) = atan(imag(ak(i))/(real(ak(i))
        +0.0001));
24 end
25 Phase_ak = Phase_ak'
26 Phase_ak = [Phase_ak(1:$-1) -Phase_ak($:-1:1)];
27 k = -(N-2):(N-2);
28 //
29 figure
30 subplot(2,1,1)
31 a = gca();
32 a.y_location = "origin";
33 a.x_location = "origin";
34 plot2d3('gnn',k,real(ak),5)
35 poly1 = a.children(1).children(1);
36 poly1.thickness = 3;
37 title('Real part of(ak)')
38 xlabel('
        k')
39 subplot(2,1,2)
40 a = gca();
41 a.y_location = "origin";
42 a.x_location = "origin";
43 plot2d3('gnn',k,imag(ak),5)
44 poly1 = a.children(1).children(1);
45 poly1.thickness = 3;
46 title('imaginary part of(ak)')
47 xlabel('

```

```

    k')
48 //
49 figure
50 subplot(2,1,1)
51 a = gca();
52 a.y_location = "origin";
53 a.x_location = "origin";
54 plot2d3('gnn',k,Mag_ak,5)
55 poly1 = a.children(1).children(1);
56 poly1.thickness = 3;
57 title('abs(ak)')
58 xlabel('

    k')
59 subplot(2,1,2)
60 a = gca();
61 a.y_location = "origin";
62 a.x_location = "origin";
63 plot2d3('gnn',k,Phase_ak,5)
64 poly1 = a.children(1).children(1);
65 poly1.thickness = 3;
66 title('<(ak)')
67 xlabel('

    k')

```

Scilab code Exa 3.12 DTFS coefficients of periodic square wave

```

1 //clear//
2 //Example3.12:DTFS coefficients of periodic square
  wave
3 clear;
4 close;
5 clc;

```

```

6 N = 10;
7 N1 = 2;
8 Wo = 2*%pi/N;
9 xn = ones(1,length(N));
10 n = -(2*N1+1):(2*N1+1);
11 a(1) = (2*N1+1)/N;
12 for k = 1:2*N1
13     a(k+1) = sin((2*%pi*k*(N1+0.5))/N)/sin(%pi*k/N);
14     a(k+1) = a(k+1)/N;
15     if(abs(a(k+1))<=0.1)
16         a(k+1) = 0;
17     end
18 end
19 a = a';
20 a_conj = conj(a);
21 ak = [a_conj($:-1:1),a(2:$)];
22 k = -2*N1:2*N1;
23 //
24 figure
25 a = gca();
26 a.y_location = "origin";
27 a.x_location = "origin";
28 plot2d3('gnn',k,real(ak),5)
29 poly1 = a.children(1).children(1);
30 poly1.thickness = 3;
31 title('Real part of(ak)')
32 xlabel('

```

```

k')

```

Scilab code Exa 3.13 Periodic sequence

```

1 //clear//
2 //Example3.13:DTFS
3 //Expression of periodic sequence using

```

```

4 //the summation two different sequence
5 clear;
6 close;
7 clc;
8 N = 5;
9 n = 0:N-1;
10 x1 = [1,1,0,0,1];
11 x1 = [x1($:-1:1) x1(2:$)]; // Square Wave x1[n]
12 x2 = [1,1,1,1,1];
13 x2 = [x2($:-1:1) x2(2:$)]; //DC sequence of x2[n]
14 x = x1+x2; //sum of x1[n] & x2[n]
15 //Zeroth DTFS coefficient of dc sequence
16 c(1) = 1;
17 //Zeroth DTFS coefficient of square waveform
18 b(1) = 3/5;
19 //Zeroth DTFS coefficient of sum of x1[n] & x2[n]
20 a(1) = b(1)+c(1);
21 //
22 Wo = 2*%pi/N;
23 for k =1:N-1
24     a(k+1) = sin((3*%pi*k)/N)/sin(%pi*k/N);
25     a(k+1) = a(k+1)/N;
26     if(abs(a(k+1))<=0.1)
27         a(k+1) =0;
28     end
29 end
30 a =a';
31 a_conj =conj(a);
32 ak = [a_conj($:-1:1),a(2:$)];
33 k = -(N-1):(N-1);
34 n = -(N-1):(N-1);
35 //
36 figure
37 subplot(3,1,1)
38 a = gca();
39 a.y_location = "origin";
40 a.x_location = "origin";
41 plot2d3('gmn',n,x,5)

```



```

42 poly1 = a.children(1).children(1);
43 poly1.thickness = 3;
44 title('x[n]')
45 xlabel('

    n')
46 subplot(3,1,2)
47 a = gca();
48 a.y_location = "origin";
49 a.x_location = "origin";
50 plot2d3('gmn',n,x1,5)
51 poly1 = a.children(1).children(1);
52 poly1.thickness = 3;
53 title('x1[n]')
54 xlabel('

    n')
55 subplot(3,1,3)
56 a = gca();
57 a.y_location = "origin";
58 a.x_location = "origin";
59 plot2d3('gmn',n,x2,5)
60 poly1 = a.children(1).children(1);
61 poly1.thickness = 3;
62 title('x2[n]')
63 xlabel('

    n')

```

Scilab code Exa 3.14 Parseval's relation of DTFS

```

1 //clear//
2 //Example3.14:DTFS
3 //Finding x[n] using parseval's relation of DTFS
4 clear;

```

```

5  close;
6  clc;
7  N = 6;
8  n = 0:N-1;
9  a(1) = 1/3;
10 a(2)=0;
11 a(4)=0;
12 a(5)=0;
13 a1 = (1/6)*((-1)^n);
14 x =0;
15 for k = 0:N-2
16     if(k==2)
17         x = x+a1;
18     else
19         x =  x+a(k+1);
20     end
21 end
22 x = [x($:-1:1),x(2:$)];
23 n = -(N-1):(N-1);
24 //
25 figure
26 a = gca();
27 a.y_location = "origin";
28 a.x_location = "origin";
29 plot2d3('gnn',n,x,5)
30 poly1 = a.children(1).children(1);
31 poly1.thickness = 3;
32 title('x[n]')
33 xlabel('

```

```

    n')

```

Scilab code Exa 3.15 DTFS:Periodic Convolution Property

```
1 //clear//
```

```

2 //Example3.15:DTFS:Periodic Convolution Property
3 clear;
4 clc;
5 close;
6 x = [1,1,0,0,0,0,1];
7 X = fft(x);
8 W = X.*X;
9 w = ifft(W);
10 w = abs(w);
11 for i =1:length(x)
12     if (abs(w(i))<=0.1)
13         w(i) = 0;
14     end
15 end
16 w = [w($:-1:1) w(2:$)];
17 N = length(x);
18 figure
19 a = gca();
20 a.y_location = "origin";
21 a.x_location = "origin";
22 plot2d3('gnn',[-(N-1):0,1:N-1],w,5)
23 poly1 = a.children(1).children(1);
24 poly1.thickness = 3;
25 title('w[n]')
26 xlabel('

```

```

n')

```

Chapter 4

The Cotntinuous Time Fourier Transform

Scilab code Exa 4.1 clear

```
1 //clear//
2 //Example 4.1: Continuous Time Fourier Transform of a
3 //Continuous Time Signal  $x(t) = \exp(-A*t)u(t)$ ,  $t > 0$ 
4 clear;
5 clc;
6 close;
7 // Analog Signal
8 A = 1; //Amplitude
9 Dt = 0.005;
10 t = 0:Dt:10;
11 xt = exp(-A*t);
12 //
13 // Continuous-time Fourier Transform
14 Wmax = 2*pi*1; //Analog Frequency = 1Hz
15 K = 4;
16 k = 0:(K/1000):K;
17 W = k*Wmax/K;
18 XW = xt* exp(-sqrt(-1)*t'*W) * Dt;
19 XW_Mag = abs(XW);
```

```

20 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -
    Wmax to Wmax
21 XW_Mag = [mtlbfliplr(XW_Mag), XW_Mag(2:1001)];
22 [XW_Phase,db] = phasemag(XW);
23 XW_Phase = [-mtlbfliplr(XW_Phase),XW_Phase(2:1001)
    ];
24 //Plotting Continuous Time Signal
25 figure
26 a = gca();
27 a.y_location = "origin";
28 plot(t,xt);
29 xlabel('t in sec. ');
30 ylabel('x(t)')
31 title('Continuous Time Signal')
32 figure
33 //Plotting Magnitude Response of CTS
34 subplot(2,1,1);
35 a = gca();
36 a.y_location = "origin";
37 plot(W,XW_Mag);
38 xlabel('Frequency in Radians/Seconds—> W');
39 ylabel('abs(X(jW))')
40 title('Magnitude Response (CTFT)')
41 //Plotting Phase Reponse of CTS
42 subplot(2,1,2);
43 a = gca();
44 a.y_location = "origin";
45 a.x_location = "origin";
46 plot(W,XW_Phase*%pi/180);
47 xlabel('                                Frequency in
    Radians/Seconds—> W');
48 ylabel('                                <X
    (jW)')
49 title('Phase Response(CTFT) in Radians')

```

Scilab code Exa 4.2 clear

```
1 //clear//
2 //Example 4.2: Continuous Time Fourier Transform of a
3 //Continuous Time Signal  $x(t) = \exp(-A \cdot \text{abs}(t))$ 
4 clear;
5 clc;
6 close;
7 // Analog Signal
8 A =1; //Amplitude
9 Dt = 0.005;
10 t = -4.5:Dt:4.5;
11 xt = exp(-A*abs(t));
12 //
13 // Continuous-time Fourier Transform
14 Wmax = 2*%pi*1; //Analog Frequency = 1Hz
15 K = 4;
16 k = 0:(K/1000):K;
17 W = k*Wmax/K;
18 XW = xt* exp(-sqrt(-1)*t'*W) * Dt;
19 XW = real(XW);
20 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -
    Wmax to Wmax
21 XW = [mtlbfliplr(XW), XW(2:1001)];
22 subplot(1,1,1)
23 subplot(2,1,1);
24 a = gca();
25 a.y_location = "origin";
26 plot(t,xt);
27 xlabel('t in sec. ');
28 ylabel('x(t)')
29 title('Continuous Time Signal')
30 subplot(2,1,2);
31 a = gca();
```

```

32 a.y_location = "origin";
33 plot(W,XW);
34 xlabel('Frequency in Radians/Seconds W');
35 ylabel('X(jW)')
36 title('Continuous-time Fourier Transform')

```

Scilab code Exa 4.4 clear

```

1 //clear//
2 //Example 4.4
3 // Continuous Time Fourier Transform
4 //and Frequency Response of a Square Waveform
5 // x(t)= A, from -T1 to T1
6 clear;
7 clc;
8 close;
9 // CTS Signal
10 A =1; //Amplitude
11 Dt = 0.005;
12 T1 = 4; //Time in seconds
13 t = -T1/2:Dt:T1/2;
14 for i = 1:length(t)
15     xt(i) = A;
16 end
17 //
18 // Continuous-time Fourier Transform
19 Wmax = 2*pi*1; //Analog Frequency = 1Hz
20 K = 4;
21 k = 0:(K/1000):K;
22 W = k*Wmax/K;
23 xt = xt';
24 XW = xt* exp(-sqrt(-1)*t'*W) * Dt;
25 XW_Mag = real(XW);
26 W = [-mtlb_fliplr(W), W(2:1001)]; // Omega from -
    Wmax to Wmax

```

```

27 XW_Mag = [mtlbfliplr(XW_Mag), XW_Mag(2:1001)];
28 //
29 subplot(2,1,1);
30 a = gca();
31 a.data_bounds=[-4,0;4,2];
32 a.y_location = "origin";
33 plot(t,xt);
34 xlabel('t in msec. ');
35 title('Contiuous Time Signal x(t)')
36 subplot(2,1,2);
37 a = gca();
38 a.y_location = "origin";
39 plot(W,XW_Mag);
40 xlabel('Frequency in Radians/Seconds');
41 title('Continuous-time Fourier Transform X(jW)')

```

Scilab code Exa 4.5 clear

```

1 //clear//
2 //Example 4.5
3 // Inverse Continuous Time Fourier Transform
4 // X(jW)= 1, from -T1 to T1
5 clear;
6 clc;
7 close;
8 // CTFT
9 A =1; //Amplitude
10 Dw = 0.005;
11 W1 = 4; //Time in seconds
12 w = -W1/2:Dw:W1/2;
13 for i = 1:length(w)
14     XW(i) = A;
15 end
16 XW = XW';
17 //

```



```

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

```

Scilab code Exa 4.6 clear

```

1 //clear//
2 //Example 4.6
3 // Continuous Time Fourier Transform of Symmetric
4 // periodic Square waveform
5 clear;
6 clc;
7 close;
8 // CTFT
9 T1 = 2;
10 T = 4*T1;
11 Wo = 2*%pi/T;
12 W = -%pi:Wo:%pi;
13 delta = ones(1,length(W));
14 XW(1) = (2*%pi*Wo*T1/%pi);
15 mid_value = ceil(length(W)/2);
16 for k = 2:mid_value
17     XW(k) = (2*%pi*sin((k-1)*Wo*T1)/(%pi*(k-1)));
18 end
19 figure

```

```

20 a = gca();
21 a.y_location = "origin";
22 a.x_location = "origin";
23 plot2d3('gnn',W(mid_value:$),XW,2);
24 poly1 = a.children(1).children(1);
25 poly1.thickness = 3;
26 plot2d3('gnn',W(1:mid_value-1),XW($:-1:2),2);
27 poly1 = a.children(1).children(1);
28 poly1.thickness = 3;
29 xlabel('W in radians/Seconds');
30 title('Continuous Time Fourier Transform of Periodic
        Square Wave')

```

Scilab code Exa 4.7 clear

```

1 //clear//
2 //Example 4.7
3 // Continuous Time Fourier Transforms of
4 // Sinusoidal waveforms (a) sin(Wot) (b) cos(Wot)
5 clear;
6 clc;
7 close;
8 // CTFT
9 T1 = 2;
10 T = 4*T1;
11 Wo = 2*%pi/T;
12 W = [-Wo,0,Wo];
13 ak = (2*%pi*Wo*T1/%pi)/sqrt(-1);
14 XW = [-ak,0,ak];
15 ak1 = (2*%pi*Wo*T1/%pi);
16 XW1 =[ak1,0,ak1];
17 //
18 figure
19 a = gca();
20 a.y_location = "origin";

```

```

21 a.x_location = "origin";
22 plot2d3('ggn',W,imag(XW),2);
23 poly1 = a.children(1).children(1);
24 poly1.thickness = 3;
25 xlabel('

        W');
26 title('CTFT of sin(Wot)')
27 //
28 figure
29 a = gca();
30 a.y_location = "origin";
31 a.x_location = "origin";
32 plot2d3('ggn',W,XW1,2);
33 poly1 = a.children(1).children(1);
34 poly1.thickness = 3;
35 xlabel('

        W');
36 title('CTFT of cos(Wot)')

```

Scilab code Exa 4.8 clear

```

1 //clear//
2 //Example 4.8
3 // Continuous Time Fourier Transforms of
4 // Periodic Impulse Train
5 clear;
6 clc;
7 close;
8 // CTFT
9 T = -4:4;;
10 T1 = 1; //Sampling Interval
11 xt = ones(1,length(T));
12 ak = 1/T1;

```

```

13 XW = 2*%pi*ak*ones(1,length(T));
14 Wo = 2*%pi/T1;
15 W = Wo*T;
16 figure
17 subplot(2,1,1)
18 a = gca();
19 a.y_location = "origin";
20 a.x_location = "origin";
21 plot2d3('ggn',T,xt,2);
22 poly1 = a.children(1).children(1);
23 poly1.thickness = 3;
24 xlabel('

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

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

```

Scilab code Exa 4.9 clear

```

1 //clear//
2 //Example 4.9: Continuous Time Fourier Transform
  Properties:
3 //Linearity and Time Shift Property
4 clear;
5 clc;

```

```

6 close;
7 // CTFT
8 t1 = -1/2:0.1:1/2;
9 t2 = -3/2:0.1:3/2;
10 x1 = ones(1,length(t1));
11 x2 = ones(1,length(t2));
12 t3 = t1+2.5;
13 t4 = t2+2.5;
14 x1 = (1/2)*x1;
15 x = [x2(1:floor(length(x2)/3)),x1+x2(ceil(length(x2)
    /3):$-floor(length(x2)/3)),x2(($-ceil(length(x2)
    /3))+2:$)];
16 subplot(3,1,1)
17 a = gca();
18 a.x_location = "origin";
19 a.y_location = "origin";
20 plot(t1,x1)
21 xtitle('x1(t)')
22 subplot(3,1,2)
23 a = gca();
24 a.x_location = "origin";
25 a.y_location = "origin";
26 plot(t2,x2)
27 xtitle('x2(t)')
28 subplot(3,1,3)
29 a = gca();
30 a.x_location = "origin";
31 a.y_location = "origin";
32 plot(t4,x)
33 xtitle('x(t)')

```

Scilab code Exa 4.12 clear

```

1 //clear//
2 //Example 4.12: Continuous Time Fourier Transform:

```

```

3 //Derivative property
4 clear;
5 clc;
6 close;
7 // CFTF
8 t = -1:0.1:1;
9 x1 = ones(1,length(t));
10 x2 = [-1,zeros(1,length(t)-2),-1];
11 x = t;
12 //differentiation of x can be expressed as
13 //summation of x1 and x2
14 subplot(3,1,1)
15 a = gca();
16 a.x_location = "origin";
17 a.y_location = "origin";
18 plot(t,x1)
19 xtitle('x1(t)')
20 subplot(3,1,2)
21 a = gca();
22 a.x_location = "origin";
23 a.y_location = "origin";
24 plot2d3('gnn',t,x2)
25 xtitle('x2(t)')
26 subplot(3,1,3)
27 a = gca();
28 a.x_location = "origin";
29 a.y_location = "origin";
30 plot(t,x)
31 xtitle('x(t)')

```

Scilab code Exa 4.18 clear

```

1 //clear//
2 //Example 4.18:Frequency Response of Ideal Low pass
  Filter

```

```

3 // X(jW)= 1, from -T1 to T1
4 clear;
5 clc;
6 close;
7 Wc = 10; //1 rad/sec
8 W = -Wc:0.1:Wc; //Passband of filter
9 HW0 = 1; //Magnitude of Filter
10 HW = HW0*ones(1,length(W));
11 //Inverse Continuous-time Fourier Transform
12 t = -%pi:%pi/length(W):%pi;
13 Dw = 0.1;
14 ht = (1/(2*%pi))*HW *exp(sqrt(-1)*W'*t)*Dw;
15 ht = real(ht);
16 figure
17 subplot(2,1,1)
18 a = gca();
19 a.y_location = "origin";
20 a.x_location = "origin";
21 plot(W,HW);
22 xtitle('Frequency Response of Filter H(jW)')
23 subplot(2,1,2)
24 a = gca();
25 a.y_location = "origin";
26 a.x_location = "origin";
27 plot(t,ht);
28 xtitle('Impulse Response of Filter h(t)')

```

Scilab code Exa 4.22 clear

```

1 //clear//
2 //Figure 4.22
3 //Plotting Continuous Time Fourier Transform of
4 //Impulse Response h(t)= exp(-A*t)u(t), t>0
5 clear;
6 clc;

```

```

7 close;
8 // Analog Signal
9 A =1; //Amplitude
10 Dt = 0.005;
11 t = 0:Dt:10;
12 ht = exp(-A*t);
13 // Continuous-time Fourier Transform
14 Wmax = 2*pi*1; //Analog Frequency = 1Hz
15 K = 4;
16 k = 0:(K/1000):K;
17 W = k*Wmax/K;
18 HW = ht* exp(-sqrt(-1)*t'*W) * Dt;
19 HW_Mag = abs(HW);
20 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -
    Wmax to Wmax
21 HW_Mag = [mtlbfliplr(HW_Mag), HW_Mag(2:1001)];
22 //Plotting Continuous Time Signal
23 figure
24 a = gca();
25 a.y_location = "origin";
26 plot(t,ht);
27 xlabel('t in sec. ');
28 title('Impulse Response h(t)')
29 figure
30 //Plotting Magnitude Response of CTS
31 a = gca();
32 a.y_location = "origin";
33 plot(W,HW_Mag);
34 xlabel('Frequency in Radians/Seconds—> W');
35 title('Frequency Response H(jW)')

```

Scilab code Exa 4.23 clear

```

1 //Figure 4.23: Multiplication Property of CTFT
2 clear;

```



```

3  clc;
4  close;
5  W1 = -1:0.1:1;
6  W2 = -2:0.1:2;
7  W = -3:0.1:3;
8  //Fourier Transform of sinc function is square wave
9  XW1 = (1/%pi)*ones(1,length(W1)); //CTFT of x1(t)
10 XW2 = (1/(2*%pi))*ones(1,length(W2)); //CTFT of x2(t)
11 XW = (1/2)*convol(XW1,XW2); //CTFT of x(t)=x1(t)*x2(t
)
12 //X(jw) = linear convolution of X1(jw)and X2(jw)
13 figure
14 a = gca();
15 a.y_location = "origin";
16 a.x_location = "origin";
17 plot(W,XW);
18 xlabel('Frequency in Radians/Seconds—> W');
19 title('Multiplication Property X(jW)')

```

Chapter 5

The Discrete Time Fourier Transform

Scilab code Exa 5.1 Discrete Time Fourier Transform of discrete sequence

```
1 //clear//
2 //Example 5.1: 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('gnn',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('gnn',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))')

```

Scilab code Exa 5.2 Discrete Time Fourier Transform

```

1 //clear//
2 //Example 5.2: Discrete Time Fourier Transform of
3 //x[n]= (a^abs(n)) a>0 and a<0
4 clear;
5 clc;
6 close;
7 // DTS Signal
8 a = 0.5;
9 max_limit = 10;
10 n = -max_limit+1:max_limit-1;
11 x = a^abs(n);
12 // Discrete-time Fourier Transform
13 Wmax = 2*%pi;
14 K = 4;
15 k = 0:(K/1000):K;
16 W = k*Wmax/K;
17 XW = x* exp(-sqrt(-1)*n'*W);
18 XW_Mag = real(XW);

```

```

19 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -
      Wmax to Wmax
20 XW_Mag = [mtlbfliplr(XW_Mag), XW_Mag(2:1001)];
21 //plot for abs(a)<1
22 figure
23 subplot(2,1,1);
24 a = gca();
25 a.y_location = "origin";
26 a.x_location = "origin";
27 plot2d3('gmn',n,x);
28 xtitle('Discrete Time Sequence x[n] for a>0')
29 subplot(2,1,2);
30 a = gca();
31 a.y_location = "origin";
32 a.x_location = "origin";
33 plot2d(W,XW_Mag);
34 title('Discrete Time Fourier Transform X(exp(jW))')

```

Scilab code Exa 5.3 Discrete Time Fourier Transform

```

1 //clear//
2 //Example 5.3: Discrete Time Fourier Transform of
3 //x[n]= 1 , abs(n)<=N1
4 clear;
5 clc;
6 close;
7 // DTS Signal
8 N1 = 2;
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('gmn',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 5.5 Time Fourier Transform: $x[n] = \cos(nW_0)$

```

1 //clear//
2 //Example5.5: Discrete Time Fourier Transform:  $x[n] =$ 
     $\cos(nW_0)$ 
3 clear;
4 clc;
5 close;
6 N = 5;
7 Wo = 2*%pi/N;
8 W = [-Wo,0,Wo];
9 XW = [%pi,0,%pi];
10 //
11 figure
12 a = gca();

```

```

13 a.y_location = "origin";
14 a.x_location = "origin";
15 plot2d3('ggn',W,XW,2);
16 poly1 = a.children(1).children(1);
17 poly1.thickness = 3;
18 xlabel('

    W');
19 title('DTFT of cos(nWo)')
20 disp(Wo)

```

Scilab code Exa 5.6 Discrete Time Fourier Transform

```

1 //clear//
2 //Example5.6:Discrete Time Fourier Transform of
3 // Periodic Impulse Train
4 clear;
5 clc;
6 close;
7 N = 5;
8 N1 = -3*N:3*N;
9 xn = [zeros(1,N-1),1];
10 x = [1 xn xn xn xn xn];
11 ak = 1/N;
12 XW = 2*%pi*ak*ones(1,2*N);
13 Wo = 2*%pi/N;
14 n = -N:N-1;
15 W = Wo*n;
16 figure
17 subplot(2,1,1)
18 a = gca();
19 a.y_location = "origin";
20 a.x_location = "origin";
21 plot2d3('ggn',N1,x,2);
22 poly1 = a.children(1).children(1);

```

```

23 poly1.thickness = 3;
24 xlabel('
    n');
25 title('Periodic Impulse Train')
26 subplot(2,1,2)
27 a = gca();
28 a.y_location = "origin";
29 a.x_location = "origin";
30 plot2d3('ggn',W,XW,2);
31 poly1 = a.children(1).children(1);
32 poly1.thickness = 3;
33 xlabel('
    W');
34 title('DTFT of Periodic Impulse Train')
35 disp(Wo)

```

Scilab code Exa 5.7 Frequency Shifting Property of DTFT

```

1 //clear//
2 //Example 5.7:Frequency Shifting Property of DTFT:
  Frequency Response of Ideal Low pass Filter and
  HPF
3 clear;
4 clc;
5 close;
6 Wc = 1; //1 rad/sec
7 W = -Wc:0.1:Wc; //Passband of filter
8 H0 = 1; //Magnitude of Filter
9 HlpW = H0*ones(1,length(W));
10 Whp1 = W+%pi;
11 Whp2 = -W-%pi;
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, HlpW);
19 xtitle('Frequency Response of LPF 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(Whp1, HlpW);
26 plot2d(Whp2, HlpW);
27 xtitle('Frequency Response of HPF H(exp(jW))')

```

Scilab code Exa 5.9 Expansion Property of DTFT

```

1 //clear//
2 //Example 5.9:Time Expansion Property of DTFT
3 clear;
4 close;
5 clc;
6 n = -1:11;
7 x = [0, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 0, 0];
8 y = [1, 1, 1, 1, 1];
9 y_2_n = zeros(1, 2*length(y)+1);
10 y_2_n(1:2:2*length(y)) = y;
11 y_2_n = [0 y_2_n 0];
12 y_2_n_1 = [0, y_2_n(1:$-1)];
13 x_r = y_2_n+2*y_2_n_1;
14 y = [0, y, zeros(1, 7)];
15 figure
16 subplot(4, 1, 1)
17 plot2d3('gnn', n, y)
18 title('y[n]')

```

```

19 subplot(4,1,2)
20 plot2d3('gmn',n,y_2_n)
21 title('y(2)[n]')
22 subplot(4,1,3)
23 plot2d3('gmn',n,y_2_n_1)
24 title('y(2)[n-1]')
25 subplot(4,1,4)
26 plot2d3('gmn',n,x)
27 title('x[n]=y(2)[n]+2*y(2)[n-1]')

```

Scilab code Exa 5.12 IDTFT: Impulse Response of Ideal Low pass Filter

```

1 //clear//
2 //Example 5.12:IDTFT:Impulse Response of Ideal Low
  pass Filter
3 clear;
4 clc;
5 close;
6 Wc = 1; //1 rad/sec
7 W = -Wc:0.1:Wc; //Passband of filter
8 H0 = 1; //Magnitude of Filter
9 HlpW = H0*ones(1,length(W));
10 //Inverse Discrete-time Fourier Transform
11 t = -2*%pi:2*%pi/length(W):2*%pi;
12 ht = (1/(2*%pi))*HlpW *exp(sqrt(-1)*W'*t);
13 ht = real(ht);
14 figure
15 subplot(2,1,1)
16 a = gca();
17 a.y_location = "origin";
18 a.x_location = "origin";
19 a.data_bounds = [-%pi,0;%pi,2];
20 plot2d(W,HlpW,2);
21 poly1 = a.children(1).children(1);
22 poly1.thickness = 3;

```

```

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

```

Scilab code Exa 5.15 Multiplication Property of DTFT

```

1 //clear//
2 //Example5.15: Multiplication Property of DTFT
3 clear;
4 clc;
5 close;
6 n = 1:100;
7 x2 = [3/4, sin(0.75*pi*n)./(%pi*n)];
8 x1 = [1/2, sin(0.5*pi*n)./(%pi*n)];
9 x = x1.*x2;
10 Wmax = %pi;
11 K = 1;
12 k = 0:(K/1000):K;
13 W = k*Wmax/K;
14 n = 0:100;
15 XW1 = x1* exp(-sqrt(-1)*n'*W);
16 XW2 = x2* exp(-sqrt(-1)*n'*W);
17 XW = x* exp(-sqrt(-1)*n'*W);
18 XW1_Mag = real(XW1);
19 XW2_Mag = real(XW2);
20 XW_Mag = real(XW);
21 W = [-mtlb_fliplr(W), W(2:$)]; // Omega from -Wmax
    to Wmax

```

```

22 XW1_Mag = [mtlbfliplr(XW1_Mag), XW1_Mag(2:$)];
23 XW2_Mag = [mtlbfliplr(XW2_Mag), XW2_Mag(2:$)];
24 XW_Mag = [mtlbfliplr(XW_Mag), XW_Mag(2:$)];
25 figure
26 subplot(3,1,1)
27 a = gca();
28 a.y_location = "origin";
29 a.x_location = "origin";
30 plot(W,XW1_Mag);
31 title('DTFT X1(exp(jW))');
32 subplot(3,1,2)
33 a = gca();
34 a.y_location = "origin";
35 a.x_location = "origin";
36 plot(W,XW2_Mag);
37 title('DTFT X2(exp(jW))');
38 subplot(3,1,3)
39 a = gca();
40 a.y_location = "origin";
41 a.x_location = "origin";
42 plot(W,XW_Mag);
43 title('Multiplication Property of DTFT');

```

Chapter 6

Time and Frequency Characterization of Signals and Systems

Scilab code Exa 6.1 Phase Response and Group Delay

```
1 //clear//
2 //Example6.1:Phase Response and Group Delay
3 clear;
4 clc;
5 close;
6 f1 = 50;
7 f2 = 150;
8 f3 = 300;
9 w1 = 315;
10 tuo1 = 0.066;
11 w2 = 943;
12 tuo2 = 0.033;
13 w3 = 1888;
14 tuo3 = 0.058;
15 f = 0:0.1:400;
16 W = 2*%pi*f;
17 for i =1:length(f)
```

```

18 num1(i) = (1+(sqrt(-1)*f(i)/f1)^2-2*sqrt(-1)*tuo1*(
    f(i)/f1));
19 den1(i) = (1+(sqrt(-1)*f(i)/f1)^2+2*sqrt(-1)*tuo1*(
    f(i)/f1));
20 H1W(i) = num1(i)/den1(i);
21 num2(i) = (1+(sqrt(-1)*f(i)/f2)^2-2*sqrt(-1)*tuo2*(
    f(i)/f2));
22 den2(i) = (1+(sqrt(-1)*f(i)/f2)^2+2*sqrt(-1)*tuo2*(
    f(i)/f2));
23 H2W(i) = num2(i)/den2(i);
24 num3(i) = (1+(sqrt(-1)*f(i)/f3)^2-2*sqrt(-1)*tuo3*(
    f(i)/f3));
25 den3(i) = (1+(sqrt(-1)*f(i)/f3)^2+2*sqrt(-1)*tuo3*(
    f(i)/f3));
26 H3W(i) = num3(i)/den3(i);
27 H_W(i) = H1W(i)*H2W(i);
28 HW(i) = H_W(i)*H3W(i);
29 phase1(i) = -2*atan((2*tuo1*(f(i)/f1))/(1.001-(f(i)
    )/f1)^2));
30 phase2(i) = -2*atan((2*tuo2*(f(i)/f2))/(1.001-(f(i)
    )/f2)^2));
31 phase3(i) = -2*atan((2*tuo3*(f(i)/f3))/(1.001-(f(i)
    )/f3)^2));
32 phase_total(i) = phase1(i)+phase2(i)+phase3(i);
33 if(f(i)<=50)
34     W_phase1(i) = -2*atan((2*tuo1*(f(i)/f1))
        /(1.001-(f(i)/f1)^2));
35     W_phase2(i) = -2*atan((2*tuo2*(f(i)/f2))
        /(1.001-(f(i)/f2)^2));
36     W_phase3(i) = -2*atan((2*tuo3*(f(i)/f3))
        /(1.001-(f(i)/f3)^2));
37     group_delay(i) = -phase_total(i)*0.1/%pi;    //
        delta_f= 0.1
38 elseif(f(i)>=50 & f(i)<=150)
39     W_phase1(i)= -2*%pi-2*atan((2*tuo1*(f(i)/f1))
        /(1.001-(f(i)/f1)^2));
40     W_phase2(i)= -2*atan((2*tuo2*(f(i)/f2))/(1.001-(
        f(i)/f2)^2));

```

```

41     W_phase3(i)= -2*atan((2*tuo3*(f(i)/f3))/(1.001-(
        f(i)/f3)^2));
42     group_delay(i) = -phase_total(i)*0.1/(2*pi);
43 elseif(f(i)>=150 & f(i)<=300)
44     W_phase1(i)= -2*atan((2*tuo1*(f(i)/f1))/(1.001-(
        f(i)/f1)^2));
45     W_phase2(i)= -4*pi-2*atan((2*tuo2*(f(i)/f2))
        /(1.001-(f(i)/f2)^2));
46     W_phase3(i)= -2*atan((2*tuo3*(f(i)/f3))/(1.001-(
        f(i)/f3)^2));
47     group_delay(i) = -phase_total(i)*0.1/(4*pi);
48 elseif(f(i)>300 & f(i)<=400)
49     W_phase1(i)= -2*atan((2*tuo1*(f(i)/f1))/(1.001-(
        f(i)/f1)^2));
50     W_phase2(i)= -2*atan((2*tuo2*(f(i)/f2))/(1.001-(
        f(i)/f2)^2));
51     W_phase3(i)= -6*pi-2*atan((2*tuo3*(f(i)/f3))
        /(1.001-(f(i)/f3)^2));
52     group_delay(i) = -phase_total(i)*0.1/(4*pi);
53 end
54 if(f(i)==300.1)
55     W_phase_total(i) = 2*pi+W_phase1(i)+W_phase2(i)+
        W_phase3(i);
56 else
57     W_phase_total(i) = W_phase1(i)+W_phase2(i)+
        W_phase3(i);
58 end
59 end
60 figure
61 plot2d(f,phase_total,2)
62 xtitle('Principal phase','Frequency (Hz)','Phase (rad)
    ');
63 figure
64 plot2d(f,W_phase_total,2)
65 xtitle('unwrapped phase','Frequency (Hz)','Phase (rad)
    ');
66 figure
67 plot2d(f,abs(group_delay),2)

```

```
68 xtitle('group delay','Frequency(Hz)','Group Delay(  
    sec)');
```

Scilab code Exa 6.3 Analog Lowpass IIR filter design

```
1 //clear//  
2 //Example6.3:Analog Lowpass IIR filter design  
3 //Cutoff frequency Fc = 500Hz  
4 //Passband ripple 1-0.05 and stopband ripple = 0.05  
5 clear;  
6 close;  
7 clc;  
8 hs_butt = analpf(5,'butt',[0.05,0.05],500);  
9 hs_ellip = analpf(5,'ellip',[0.05,0.05],500);  
10 fr=0:.1:2000;  
11 hf_butt=freq(hs_butt(2),hs_butt(3),%i*fr);  
12 hm_butt = abs(hf_butt);  
13 hf_ellip=freq(hs_ellip(2),hs_ellip(3),%i*fr);  
14 hm_ellip = abs(hf_ellip);  
15 //Plotting Magnitude Response of Analog IIR Filters  
16 a = gca();  
17 plot2d(fr,hm_butt)  
18 poly1 = a.children(1).children(1);  
19 poly1.foreground = 2;  
20 poly1.thickness = 2;  
21 poly1.line_style = 3;  
22 plot2d(fr,hm_ellip)  
23 poly1 = a.children(1).children(1);  
24 poly1.foreground = 5;  
25 poly1.thickness = 2;  
26 xlabel('Frequency(Hz)')  
27 ylabel('Magnitude of frequency response')  
28 legend(['Butterworth Filter';'Elliptic Filter'])
```

Scilab code Exa 6.4 Bode Plot

```
1 //clear//
2 //Example 6.4:Bode Plot
3 s = %s;
4 //Open Loop Transfer Function
5 H = syslin('c',[20000/(s^2+100*s+10000)]); //jw
   replaced by s
6 clf;
7 bode(H,0.01,10000)
```

Scilab code Exa 6.5 Bode Plot

```
1 //clear//
2 //Example 6.5:Bode Plot
3 s = %s;
4 //Open Loop Transfer Function
5 H = syslin('c',[(100*(1+s))/((10+s)*(100+s))]); //jw
   replaced by s
6 clf;
7 bode(H,0.01,10000)
```

Chapter 7

Sampling

Scilab code Exa 7.1 Sinusoidal signal

```
1 //clear//
2 //Example7.1: Sinusoidal signal
3 clear;
4 close;
5 clc;
6 Wm = 2*%pi;
7 Ws = 2*Wm;
8 t = -2:0.01:2;
9 phi = -%pi/2;
10 x = cos((Ws/2)*t+phi);
11 y = sin((Ws/2)*t);
12 subplot(2,1,1)
13 a = gca();
14 a.x_location = "origin";
15 a.y_location = "origin";
16 plot(t,x)
17 title('cos(Ws/2*t+phi)')
18 subplot(2,1,2)
19 a = gca();
20 a.x_location = "origin";
21 a.y_location = "origin";
```

```

22 plot(t,y)
23 title('sin(Ws/2*t)')

```

Scilab code Exa 7.2 Digital Differentiator

```

1 //clear//
2 //Example7.2: Digital Differentiator
3 syms t n;
4 T = 0.1; //Sampling time in seconds
5 xct = sin(%pi*t/T)/(%pi*t);
6 yct = diff(xct,t);
7 disp(yct, 'yc(t)=');
8 t = n*T;
9 xdn = sin(%pi*t/T)/(%pi*t);
10 ydn = diff(xdn,n);
11 disp(ydn, 'yd[n]=');
12 hdn = T*ydn;
13 disp(hdn, 'hd[n]=');
14 //Result
15 //yc(t) = (10*cos(31.415927*t)/t) - (0.3183099*sin
      (31.415927*t)/(t^2))
16 //yd[n]=(10*cos(3.1415927*n)/n) - 3.183*sin(3.1415927*
      n)/(n^2)
17 //hd[n]=(cos(3.1415927*n)/n) - 0.3183*sin(3.1415927*n)
      /(n^2)

```

Scilab code Exa 7.3 Half Sample Delay system

```

1 //clear//
2 //Example7.3: Half Sample Delay system
3 syms t n T;
4 //T = 0.1; //Sampling time in seconds
5 xct = sin(%pi*t/T)/(%pi*t);

```

```

6 t = t-T/2;
7 yct_del = sin(%pi*t/T)/(%pi*t);
8 disp(yct_del, 'Output of Half Sample delay system
   continuous =');
9 t = n*T-T/2;
10 xdn = sin(%pi*t/T)/(%pi*t);
11 ydn_del = xdn;
12 disp(ydn_del, 'Output of Half Sample delay system
   discrete =');
13 hdn = T*ydn_del;
14 disp(hdn, 'Impulse Response of discrete time half
   sample delay system=');
15 //Result
16 //Output of Half Sample delay system continuous =
17 //sin(3.14*(t-T/2)/T)/(3.14*(t-T/2))
18 //Output of Half Sample delay system discrete =
19 // sin(3.14*(n*T-T/2)/T)/(3.14*(n*T-T/2))
20 // Impulse Response of discrete time half sample
   delay system=
21 // T*sin(3.14*(n*T-T/2)/T)/(3.14*(n*T-T/2))

```

Scilab code Exa 7.4 Period of the sampled signal and Sampling frequency

```

1 //clear//
2 //Example7.4: Finding the period of the sampled
   signal
3 //and Sampling frequency
4 clear;
5 close;
6 clc;
7 Wm = 2*%pi/9;
8 N = floor(2*%pi/(2*Wm))
9 disp(N, 'Period of the discrete signal')
10 Ws = 2*%pi/N;
11 disp(Ws, 'The Sampling frequency corresponding to the

```

period N')

Scilab code Exa 7.5 Multirate Signal Processing

```
1 //clear//
2 //Example7.5:Multirate Signal Processing:Sampling
  Rate Conversion
3 //(1)Downsampling by 4
4 //(2)Upsampling by 2
5 //(3)Upsampling by 2 and followed by downsampling by
  9
6 clear;
7 close;
8 clc;
9 Wm = 2*%pi/9;//Maximum frequency of signal
10 Ws = 2*Wm; //Sampling frequency
11 N = floor(2*%pi/Ws);//period of discrete signal
12 //Original discrete time signal generation and
  Magnitude response
13 n = 0:0.01:N;
14 x = sin(Wm*n);
15 Wmax = 2*%pi/9;
16 K = 4;
17 k = 0:(K/1000):K;
18 W = k*Wmax/K;
19 XW = x* exp(-sqrt(-1)*n'*W);
20 XW_Mag = real(XW);
21 XW_Mag = XW_Mag/max(XW_Mag);
22 W = [-mtlbfliplr(W), W(2:1001)]; // Omega from -
  Wmax to Wmax
23 XW_Mag = [mtlbfliplr(XW_Mag), XW_Mag(2:1001)];
24 //(1)downsampling by 4 and corresponding magnitude
  response
25 n1 = 0:0.01:N/4;
26 y = x(1:4:length(x));
```

```

27 k1 = 0:(K/2000):K;
28 W1 = k1*4*Wmax/K;
29 XW4 = y* exp(-sqrt(-1)*n1'*W1);
30 XW4_Mag = real(XW4);
31 XW4_Mag = XW4_Mag/max(XW4_Mag);
32 W1 = [-mtlbfliplr(W1), W1(2:$)]; // Omega from -
    Wmax to Wmax
33 XW4_Mag = [mtlbfliplr(XW4_Mag), XW4_Mag(2:$)];
34 //(2)Upsampling by 2 and corresponding magnitude
    response
35 n2 = 0:0.01:2*N;
36 z = zeros(1,length(n2));
37 z([1:2:length(z)]) = x;
38 k2 = 0:(K/500):K;
39 W2 = k2*Wmax/(2*K);
40 XW2 = z* exp(-sqrt(-1)*n2'*W2);
41 XW2_Mag = real(XW2);
42 XW2_Mag = XW2_Mag/max(XW2_Mag);
43 W2 = [-mtlbfliplr(W2), W2(2:$)]; // Omega from -
    Wmax to Wmax
44 XW2_Mag = [mtlbfliplr(XW2_Mag), XW2_Mag(2:$)];
45 //(3)Upsampling by 2 and Downsampling by 9
    corresponding magnitude response
46 n3 = 0:0.01:2*N/9;
47 g = z([1:9:length(z)]);
48 k3 = 0:K/(9*500):K;
49 W3 = k3*9*Wmax/(2*K);
50 XW3 = g* exp(-sqrt(-1)*n3'*W3);
51 XW3_Mag = real(XW3);
52 XW3_Mag = XW3_Mag/max(XW3_Mag);
53 W3 = [-mtlbfliplr(W3), W3(2:$)]; // Omega from -
    Wmax to Wmax
54 XW3_Mag = [mtlbfliplr(XW3_Mag), XW3_Mag(2:$)];
55 //
56 figure
57 subplot(2,2,1)
58 a = gca();
59 a.y_location = "origin";

```

```

60 a.x_location = "origin";
61 a.data_bounds = [-%pi,0;%pi,1.5];
62 plot2d(W,XW_Mag,5);
63 title('Spectrum of Discrete Signal X(exp(jW))')
64 subplot(2,2,2)
65 a = gca();
66 a.y_location = "origin";
67 a.x_location = "origin";
68 a.data_bounds = [-%pi,0;%pi,1.5];
69 plot2d(W1,XW4_Mag,5);
70 title('Spectrum of downsampled signal by 4 X(exp(jW
    /4))')
71 subplot(2,2,3)
72 a = gca();
73 a.y_location = "origin";
74 a.x_location = "origin";
75 a.data_bounds = [-%pi,0;%pi,1.5];
76 plot2d(W2,XW2_Mag,5);
77 title('Spectrum of Upsampled signal by 2 X(exp(2jW)
    )')
78 subplot(2,2,4)
79 a = gca();
80 a.y_location = "origin";
81 a.x_location = "origin";
82 a.data_bounds = [-%pi,0;%pi,1.5];
83 plot2d(W3,XW3_Mag,5);
84 title('Spectrum of Upsampled by 2 and Downsampled by
    9 X(exp(2jW/9))')

```

Chapter 9

The Laplace Transform

Scilab code Exa 9.1 Lapalce Transform $x(t)$

```
1 // clear //
2 //Example9.1:Lapalce Transform  $x(t) = \exp(-at) \cdot u(t)$ 
3 syms t s;
4 a = 3;
5 y =laplace('%e^(-a*t)',t,s);
6 disp(y)
7 //Result
8 //1/(s+a)
```

Scilab code Exa 9.2 Lapalce Transform $x(t)$

```
1 // clear //
2 //Example9.2:Lapalce Transform  $x(t) = -\exp(-at) \cdot u(-t)$ 
3 syms t s;
4 a =3;
5 y = laplace('%e^(a*-t)',t,s);
6 disp(y)
```



```
7 // Result
8 // 1/(s+a)
```

Scilab code Exa 9.3 Lapalce Transform $x(t)$

```
1 // clear //
2 // Example 9.3: Lapalce Transform  $x(t) = 3\exp(-2t)u(t) - 2\exp(-t)u(t)$ 
3 syms t s;
4 y = laplace('3*%e^(-2*t)-2*%e^(-t)', t, s);
5 disp(y)
6 // Result
7 // (3/(s+2)) - (2/(s+1))
```

Scilab code Exa 9.4 clear

```
1 // clear //
2 // Example 9.4: Lapalce Transform  $x(t) = \exp(-2t)u(t) + \exp(-t)(\cos 3t)u(t)$ 
3 syms t s;
4 y = laplace('%e^(-2*t)+%e^(-t)*cos(3*t)', t, s);
5 disp(y)
6 // Result
7 // [(s+1)/(s^2+2*s+10)] + [1/(s+2)] refer equation
   9.29
8 // Equivalent to (2*s^2+5*s+12)/((s^2+2*s+10)*(s+2))
   refer equation 9.30
```

Scilab code Exa 9.5 clear

```

1 // clear //
2 //Example9.5:Lapalce Transform of  $x(t)=s(t) - (4/3)\exp(-t)u(t) + (1/3)\exp(2t)u(t)$ 
3 syms t s;
4 y = laplace('-(4/3)*%e^(-t)+(1/3)*%e^(2*t)',t,s);
5 y = 1+y;
6 disp(y)
7 //Result
8 //[-4/(3*(s+1))]+[1/(3*(s-2))]+1

```

Scilab code Exa 9.6 clear

```

1 // clear //
2 //Example9.6
3 //Lapalce Transform  $x(t) = \exp(-at)u(t)$ ,  $0 < t < T$ 
4 syms t s;
5 a = 3;
6 T = 10;
7 //t = T;
8 y = laplace('%e^(-a*t)-%e^(-a*t)*%e^(-(s+a)*T)',t,s)
9 ;
10 disp(y)
11 //Result
12 // [1/(s+a)] - [(exp((-s-a)*T))/(s+a)]

```

Scilab code Exa 9.7 clear

```

1 // clear //
2 //Example9.7
3 //Lapalce Transform  $x(t) = \exp(-b.\text{abs}(t)).u(t)$ ,  $0 < t < T$ 
4 // $x(t) = \exp(-bt).u(t) + \exp(bt).u(-t)$ 
5 syms t s;

```

```

6 b = 3;
7 y = laplace(' %e^(-b*t)-%e^(b*t)',t,s);
8 disp(y)
9 //Result
10 // [1/(s+b)]-[1/(s-b)]

```

Scilab code Exa 9.8 clear

```

1 //clear//
2 //Example9.8:Inverse Lapalce Transform
3 //X(S) = 1/((s+1)(s+2))
4 s =%s ;
5 G =syslin('c',(1/((s+1)*(s+2)))) ;
6 disp(G,"G( s )=")
7 plzr(G)
8 x=denom(G) ;
9 disp(x,"Ch a r a c t e r i s t i c s Polynomial=" )
10 y = roots(x) ;
11 disp(y,"Poles of a system=" )
12 //Result
13 // -1 and -2

```

Scilab code Exa 9.9 clear

```

1 //clear//
2 //Example9.9:Inverse Lapalce Transform
3 //X(S) = 1/((s+1)(s+2))
4 s =%s ;
5 syms t ;
6 [A]=pfs(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 //Result
12 // (%e^-t)-(%e^-(2*t))

```

Scilab code Exa 9.10 Inverse Lapalce Transform

```

1 //clear//
2 //Example9.10:Inverse Lapalce Transform
3 //X(S) = 1/((s+1)(s+2)) Re(s)< -1,Re(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 //Result
12 // %e^-(2*t)-%e^-t

```

Scilab code Exa 9.11 Inverse Lapalce Transform

```

1 //clear//
2 //Example9.11:Inverse Lapalce Transform
3 //X(S) = 1/((s+1)(s+2)) -2< Re(s)< -1
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 //Result
12 // -(%e^-t)-(%e^-(2*t))

```

Scilab code Exa 9.12 Inverse Lapalce Transform

```
1 //clear//
2 //Example9.12: Inverse Lapalce Transform
3 //X(S) = 1/(s+(1/2))   Re(s)> -1/2
4 s =%s ;
5 G =syslin('c', (1/(s+0.5)))
6 disp(G, "G( s )=")
7 plzr(G)
```

Scilab code Exa 9.13 Inverse Lapalce Transform

```
1 //clear//
2 //Example9.13
3 //Inverse Lapalce Transform
4 //X1(S) = 1/(s+1)   Re(s)> -1
5 //X2(S) = 1/((s+1)(s+2))   Re(s)>-1
6 s =%s ;
7 syms t ;
8 G1 =syslin('c', (1/(s+1)));
9 disp(G1, "G( s )=")
10 figure
11 plzr(G1)
12 G2 =syslin('c', (1/((s+1)*(s+2))));
13 disp(G2, "G( s )=")
14 figure
15 plzr(G2)
16 G3 = syslin('c', (1/(s+1)) - (1/((s+1)*(s+2))));
17 disp(G3, "G( s )=")
18 figure
19 plzr(G3)
```

Scilab code Exa 9.14 Lapalce Transform

```
1 // clear //
2 //Example9.14:Lapalce Transform
3 //x(t) = t.exp(-at), t>0
4 //x(t) = (t^2)/2.exp(-at), t>0
5 s =%s ;
6 syms t ;
7 a =10;
8 x1 = laplace('t*%e^(-10*t)',t,s);
9 disp(x1)
10 x2 = laplace('((t^2)/2)*%e^(-10*t)',t,s);
11 disp(x2)
12 //Result
13 //1/((s+10)^2)
14 // 1/((s+10)^3)
```

Scilab code Exa 9.15 Inverse Lapalce Transform

```
1 // clear //
2 //Example9.15:Inverse Lapalce Transform
3 //X(S) = (2s^2+5s+5)/((s+1)^2)(s+2) Re(s)>-1
4 s =%s ;
5 syms t ;
6 [A]=pfs((2*(s^2)+5*s+5)/(((s+1)^2)*(s+2))); //
7     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)=")
13 //Result
```

```
13 // (2*t*(%e^-t)) - (%e^-t) + (3*%e^-(2*t))
```

Scilab code Exa 9.16 Initial Value Theorem of Lapalace Transform

```
1 // clear //
2 // Example9.16: Initial Value Theorem of Lapalace
  Transform
3 syms s;
4 num =poly([12 5 2], 's', 'coeff')
5 den =poly([20 14 4 1], 's', 'coeff')
6 X = num/den
7 disp(X, "X(s)=")
8 SX = s*X;
9 Initial_Value =limit(SX,s,%inf);
10 disp(Initial_Value, "x(0)=")
11 // Result
12 // (2*%inf^3+5*%inf^2+12*%inf)/( %inf^3+4*%inf^2+14*
  %inf+20) =2
```

Scilab code Exa 9.17 Analysis and Characterization of LTI System

```
1 // clear //
2 // Example9.17: Analysis and Characterization of LTI
  System
3 // Lapalce Transform h(t) = exp(-t).u(t)
4 syms t s;
5 h =laplace('%e^(-t)',t,s);
6 disp(h)
7 // Result
8 // 1/(s+1)
```

Scilab code Exa 9.18 Analysis and Characterization of LTI System

```
1 //clear//
2 //Example9.18: Analysis and Characterization of LTI
  System
3 //Lapalce Transform  $x(t) = \exp(-\text{abs}(t))$ 
4 // $x(t) = \exp(-t) \cdot u(t) + \exp(t) \cdot u(-t)$ 
5 syms t s;
6 y = laplace('%e^(-t)-%e^(t)',t,s);
7 disp(y)
8 //Result
9 //  $(1/(s+1)) - (1/(s-1))$ 
```

Scilab code Exa 9.19 Analysis and Characterization of LTI System

```
1 //clear//
2 //Example9.19: Analysis and Characterization of LTI
  System
3 //Inverse Lapalce Transform
4 // $X(S) = (e^s)/(s+1)$ 
5 syms s t ;
6 h1 = exp(-1); //Inverse Laplace Transform of exp(s)
7 H2 = 1/(s+1);
8 h2 = ilaplace(H2,s,t)
9 h = h1*h2;
10 disp(h,"h(t)=")
11 //Result
12 //  $(18089*(\%e^{-t}))/49171 = 0.3678794(\%e^{-t})$ 
```

Scilab code Exa 9.20 Inverse Lapalce Transform

```
1 //clear//
2 //Example9.20: Inverse Lapalce Transform
```



```

3 //X(S) = ((s-1)/((s+1)*(s-2)))
4 s =%s ;
5 syms t ;
6 [A] = pfs(s/((s+1)*(s-2)));
7 [B] = pfs(1/((s+1)*(s-2)));
8 F1 = ilaplace(A(1),s,t)
9 F2 = ilaplace(A(2),s,t)
10 F3 = ilaplace(B(1),s,t)
11 F4 = ilaplace(B(2),s,t)
12 F = F1+F2-F3-F4;
13 disp(F,"f(t)=")
14 //Result
15 //f(t)= 33333329999999*exp(2*t)
    /99999970000000+6666664*%e^-t/9999997
16 //i.e. f(t) =0.3333334*exp(2*t)+0.6666666*%e^(-t)
17 //Refer equation 9.120. (1/3)=0.3333 and (2/3) =
    0.66666

```

Scilab code Exa 9.21 Analysis and Characterization of LTI System

```

1 //clear//
2 //Example9.21: Analysis and Characterization of LTI
    System
3 //Laplace Transform h(t) = exp(2t)u(t), Re(s)>2
4 syms t s;
5 X = laplace('%e^(2*t)',t,s);
6 disp(X)
7 //Result
8 //1/(s-2)

```

Scilab code Exa 9.25 Finding Transfer function H(S) of LTI system

```

1 //clear//

```

```

2 //Example9.25:LTI Systems Characterized by Linear
  Constant
3 //Coefficient differential Equation
4 //Finding Transfer function H(S) of LTI system
5 //x(t) = exp(-3t).u(t)
6 //y(t) = [exp(-t)-exp(-2t)].u(t)
7 syms t s;
8 X = laplace('%e^(-3*t)',t,s);
9 Y = laplace('%e^(-t)-%e^(-2*t)',t,s);
10 H = Y/X;
11 disp(H)
12 //Result
13 //(s+3)*(1/(s+1)-1/(s+2))

```

Scilab code Exa 9.31 Partial Fraction

```

1 //clear//
2 //Example9.31:Causal LTI Systems described by
  differential equations
3 //and Rational System functions
4 //Partial Fraction
5 //H(S) = ((s-1)/((s+1)*(s-2)))
6 s =%s ;
7 syms t ;
8 [A] = pfs((2*s^2+4*s-6)/(s^2+3*s+2));
9 disp(A,"H(S)=")
10 //Result H(S)=
11 // - 8
12 // -----
13 // 1 + s
14 // 6
15 // -----
16 // 2 + s
17 //
18 // 2

```

Scilab code Exa 9.33 Unilateral Laplace Transform

```
1 // clear //
2 //Example9.33: Unilateral Laplace Transform: Time
   Shifting Property
3 //x(t) = exp(-a*(t+1)).u(t+1)
4 syms t s;
5 a = 2;
6 X = laplace('%e^(-a*(t+1))', t, s);
7 disp(X)
8 //Result
9 //%e^-a/(s+a)
```

Scilab code Exa 9.34 Unilateral Laplace Transform

```
1 // clear //
2 //Example9.34: Unilateral Laplace Transform
3 //x(t) = s(t)+2u(t)+e^t.u(t)
4 syms t s;
5 a = 2;
6 X = laplace('2+%e^(t)', t, s);
7 Y = 1+X;
8 disp(X)
9 disp(Y)
10 //Result
11 // (2/s)+(1/(s-1))+1
```

Scilab code Exa 9.35 clear

```

1 //clear//
2 //Example9.35: Unilateral Inverse Laplace Transform
3 //X(S) = 1/((s+1)(s+2))
4 s = %s;
5 syms t;
6 X = 1/((s+1)*(s+2));
7 x = ilaplace(X,s,t);
8 disp(X)
9 disp(x)
10 //Result
11 // (%e^-t)-(%e^-(2*t))

```

Scilab code Exa 9.36 clear

```

1 //clear//
2 //Example9.36: Unilateral Laplace Transform
3 //X(S) = ((s^2)-3)/(s+2)
4 s = %s;
5 syms t;
6 [X] = pfs((s^2)-3)/(s+2));
7 disp(X)

```

Scilab code Exa 9.37 clear

```

1 //clear//
2 //Example9.37: Unilateral Laplace Transform: Solving
  Differential Equation
3 //Y(S) = alpha/(s(s+1)(s+2))
4 s = %s;
5 syms t;
6 alpha = 1; //Alpha value assigned as some constant
  one
7 [A] = pfs(alpha/(s*(s+1)*(s+2)));

```

```

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+F3
12 disp(F)
13 //result
14 // (-%e^-t)+((%e^-(2*t))/2)+(1/2 )

```

Scilab code Exa 9.38 clear

```

1 //clear//
2 //Example9.38: Unilateral Laplace Transform: Solving
   Differential Equation
3 //Y(S)=[beta(s+3)/((s+1)(s+2))]+[gamma/((s+2)(s+2))
   ]+[alpha/(s(s+1)(s+2))]
4 s = %s;
5 syms t;
6 alpha = 2; //input constant
7 beta_B = 3; //intial condition
8 gamma_v = -5; //initial condition
9 Y1 = 1/s;
10 Y2 = 1/(s+1);
11 Y3 = 3/(s+2);
12 Y = Y1-Y2+Y3;
13 disp(Y)
14 y = ilaplace(Y,s,t)
15 disp(y)
16 //result
17 // (-%e^(-t))+3*(%e^(-(2*t)))+1

```

Chapter 10

The Z Transform

Scilab code Exa 10.1 Ztransform of $x[n]$

```
1 // clear //
2 // Example10.1: Ztransform of  $x[n] = (a)^n \cdot u[n]$ 
3 syms n z;
4 a = 0.5;
5 x =(a)^n
6 X = symsum(x*(z^(-n)),n,0,%inf)
7 disp(X,"ans=")
8 //Result
9 //  $1.0 \cdot (2^{(-\%inf-1)} \cdot z^{(-\%inf-1)} - 1) / (1 / (2 \cdot z) - 1)$ 
10 // Equivalent to  $-1 / (0.5 \cdot (z^{-1} - 1))$ 
```

Scilab code Exa 10.2 Z transform of $x[n] = -a^n \cdot u[-n-1]$

```
1 // clear //
2 //Example 10.2: Z transform of  $x[n] = -a^n \cdot u[-n-1]$ 
3 //a = 0.5
4 clear;
5 close;
```

```

6  clc;
7  syms n z;
8  a = 0.5;
9  x=-(0.5)^(-n)
10 X=symsum(x*(z^(n)),n,1,%inf)
11 disp(X,"ans=")
12 //Result
13 //  $-1.0*(2^{(\%inf+1)}*z^{(\%inf+1)}-2*z)/(2*z-1)$ 
14 //Equivalent to  $-1*-2*z/(2*z-1) = 1/(1-0.5*z^{-1})$ 

```

Scilab code Exa 10.3 Z transform of $x[n]$

```

1  //clear//
2  //Example 10.3:Z transform of  $x[n] = 7.(1/3)^n.u[n]$ 
    $-6.(1/2)^n.u[n]$ 
3  syms n z;
4  x1=(0.33)^(n)
5  X1=symsum(7*x1*(z^(-n)),n,0,%inf)
6  x2=(0.5)^(n)
7  X2=symsum(6*x2*(z^(-n)),n,0,%inf)
8  X = X1-X2
9  disp(X,"ans=")
10 //Result
11 //  $-6.0*(2^{(-\%inf-1)}*z^{(-\%inf-1)}-1)/(1/(2*z)-1)$ 
12 //Equivalent to  $-6*-1/(0.5*z^{-1}-1)$ 
13 //The Region of Convergence is  $|z|>1/2$ 

```

Scilab code Exa 10.4 Z-transform of sine signal

```

1  //clear//
2  //Example10.4: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=")

```

Scilab code Exa 10.5 Z-transform of Impulse Sequence

```

1 //clear//
2 //Example10.5:Z-transform of Impulse Sequence
3 syms n z;
4 X=symsum(1*(z^(-n)),n,0,0);
5 disp(X,"ans=")
6 //Result
7 // 1

```

Scilab code Exa 10.6 Z transform of $x[n]$

```

1 //clear//
2 //Example 10.6:Z transform of  $x[n] = a^n, 0 < n < N$ 
   -1
3 syms n z;
4 a = 0.5;
5 N =6;
6 x=(a)^(n)
7 X=symsum(x*(z^(-n)),n,0,N)
8 disp(X,"ans=")
9 //Result
10 //0.5/z+0.25/z^2+0.125/z^3+0.0625/z^4+0.03125/z
   ^5+0.015625/z^6+1.0

```

Scilab code Exa 10.7 Z transform of $x[n]$

```
1 //clear//
2 //Example 10.7:Z transform of  $x[n] = b^n.u[n]+b^{-n}.u$ 
    $[-n-1]$ 
3 syms n z;
4 b = 0.5;
5 x1=(b)^(n)
6 x2=(b)^(-n)
7 X1=symsum(x1*(z^(-n)),n,0,%inf)
8 X2=symsum(x2*(z^(n)),n,1,%inf)
9 X = X1+X2;
10 disp(X,"ans=")
11 //Result
12 //+1.0*(2^(-%inf-1)*z^(-%inf-1)-1)/(1/(2*z)-1)
13 //Equivalent to  $-1/(0.5*z^{-1} - 1)$ 
14 //Region of Convergence  $|z|>0.5$ 
```

Scilab code Exa 10.9 clear

```
1 //clear//
2 //Example10.9: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 =z*(3*z-(5/6))/((z-(1/4))*(z-(1/3)))
6 X1 = denom(X);
7 zp = roots(X1);
8 X1 = z1*(3*z1-(5/6))/((z1-(1/4))*(z1-(1/3)))
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]= ')
17 ////Result
18 //h[n]= (1/4)^n+(2/3)^n

```

Scilab code Exa 10.10 Inverse Z Transform

```

1 //clear//
2 //Example10.10:Inverse Z Transform:ROC 1/4<|z|<1/3
3 z = %z;
4 syms n z1;//To find out Inverse z transform z must
   be linear z = z1
5 X =z*(3*z-(5/6))/((z-(1/4))*(z-(1/3)))
6 X1 = denom(X);
7 zp = roots(X1);
8 X1 = z1*(3*z1-(5/6))/((z1-(1/4))*(z1-(1/3)))
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*'u(n)', 'h1[n]= ')
13 h2 = limit(F2, z1, zp(2));
14 disp((h2)*'u(-n-1)', 'h2[n]= ')
15 disp((h1)*'u(n)'-(h2)*'u(n-1)', 'h[n]= ')
16 ////Result
17 // h[n]= u(n)/4^n-2*u(n-1)/3^n
18 //Equivalent to h[n] =(1/4)^n.u[n]-2*(1/3)^n.u[-n-1]

```

Scilab code Exa 10.11 Inverse Z Transform

```

1 //clear//

```

```

2 //Example10.11:Inverse Z Transform:ROC |z|<1/4
3 z = %z;
4 syms n z1;//To find out Inverse z transform z must
   be linear z = z1
5 X =z*(3*z-(5/6))/((z-(1/4))*(z-(1/3)))
6 X1 = denom(X);
7 zp = roots(X1);
8 X1 = z1*(3*z1-(5/6))/((z1-(1/4))*(z1-(1/3)))
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*'u(-n-1)', 'h1[n]=')
13 h2 = limit(F2,z1,zp(2));
14 disp((h2)*'u(-n-1)', 'h2[n]=')
15 disp(-(h1)*'u(-n-1)'-(h2)*'u(-n-1)', 'h[n]=')
16 ////Result
17 // h[n]= -u(-n-1)/4^n-2*u(-n-1)/3^n
18 //Equivalent to h[n] =-(1/4)^n.u[-n-1]-2*(1/3)^n.u[-
   n-1]

```

Scilab code Exa 10.12 Inverse z tranform

```

1 //clear//
2 //Example10.12:Inverse z tranform:For Finite
   duration discrete sequence
3 syms z;
4 X = [4*z^2 0 2 3*z^-1];
5 n = -2:1;
6 for i = 1:length(X)
7   x(i) = X(i)*(z^n(i));
8 end
9 disp(x, 'x[n]=')

```

Scilab code Exa 10.13 Inverse z tranform of InFinite duration discrete sequence

```
1 //clear//
2 //Example10.13:Inverse z tranform ofInFinite
   duration discrete sequence
3 //Power Series Method (OR)//Long Division Method
4 z = %z;
5 a = 2;
6 X = ldiv(z,z-a,5)
```

Scilab code Exa 10.18 Ztransform-Differentiation Property

```
1 //clear//
2 // Example10.18:Ztransform-Differentiation Property
3 // x[n] = (a)^n.u[n]
4 syms n z;
5 a = 0.5;
6 x =(a)^n
7 X = symsum(x*(z^(-n)),n,0,%inf)
8 X1 = -1/((1/(2*z))-1) //z transform of 0.5^n.u[
   n]
9 Y = -z*diff(X,z) //Differentiation property of z-
   transform
10 disp(X,"ans=")
11 disp(Y,"ans=")
12 //Result
13 //X(z) = 1.0*(2^(-%inf-1)*z^(-%inf-1)-1)/(1/(2*z)-1)
14 //Y(z) = -1.0*(-%inf-1)*2^(-%inf-1)*z^(-%inf-1)
   /(1/(2*z)-1)
15 //Y1(z) = 1/(2*(1/(2*z)-1)^2*z)
16 //Equivalent to Y1(z) = 0.5*z^-1/((1-0.5*z^-1)^2)
```

Scilab code Exa 10.19 Z Transform : Initial Value Theorem

```

1 //clear//
2 //Example10.19:Z Transform : Initial Value Theorem
3 z = %z;
4 syms n z1;//To find out Inverse z transform z must
   be linear z = z1
5 X =z*(z-(3/2))/((z-(1/3))*(z-(1/2)))
6 X1 = denom(X);
7 zp = roots(X1);
8 X1 = z1*(z1-(3/2))/((z1-(1/3))*(z1-(1/2)))
9 F1 = X1*(z1^(n-1))*(z1-zp(1));
10 F2 = X1*(z1^(n-1))*(z1-zp(2));
11 x1 = limit(F1,z1,zp(1));
12 x2 = limit(F2,z1,zp(2));
13 x = x1+x2;
14 disp(x, 'x[n]= ')
15 x_initial = limit(x,n,0);
16 disp(x_initial, 'x[0]= ')
17 ////Result
18 //x[n]= 7/3^n-3*2^(1-n)
19 //x[0]= 1; Initial Value

```

Scilab code Exa 10.23 Inverse Z Transform $H(z) = z/z-a$

```

1 //clear//
2 //Example10.23:Inverse Z Transform H(z) =z/z-a
3 //z = %z;
4 syms n z;
5 a = 2;
6 H = z/(z-a);
7 F = H*z^(n-1)*(z-a);
8 h = limit(F,z,a);
9 disp(h, 'h[n]= ')

```

Scilab code Exa 10.25 Coefficient Difference equations

```
1 //clear//
2 //Example10.25:LTi Systems characterized by Linear
   Constant
3 //Coefficient Difference equations
4 //Inverse Z Transform
5 //z = %z;
6 syms n z;
7 H1 = z/(z-(1/2));
8 H2 = (1/3)/(z-(1/2));
9 F1 = H1*z^(n-1)*(z-(1/2));
10 F2 = H2*z^(n-1)*(z-(1/2));
11 h1 = limit(F1,z,1/2);
12 disp(h1,'h1[n]=')
13 h2 = limit(F2,z,1/2);
14 disp(h2,'h2[n]=')
15 h = h1+h2;
16 disp(h,'h[n]=')
17 //Result
18 //h[n]= [(1/2)^n]+[2^(1-n)]/3
19 //Which is Equivalent to h[n] =[(1/2)^n]+[(1/2)^(n
   -1)]/3
```

Scilab code Exa 10.33 Differentiation Property of Unilateral Ztransform

```
1 //clear//
2 // Example10.33: Differentiation Property of
   Unilateral Ztransform
3 // x[n] = (a)^(n+1).u[n+1]
4 syms n z;
5 a = 0.5;
6 x =(a)^(n+1)
7 X = symsum(x*(z^(-n)),n,-1,%inf)
8 disp(X,"ans=")
```

```

9 //Result
10 //X(z)= 0.5*(2^(-%inf-1)*z^(-%inf-1)-2*z)/(1/(2*z)
    -1)
11 //Equivalent to z/(1-0.5*z^-1)

```

Scilab code Exa 10.34 Unilateral Ztransform- partial fraction

```

1 //clear//
2 // Example10.34: Unilateral Ztransform- partial
  fraction
3 // X(z) =(3-(5/6)*(z^-1))/((1-(1/4)*(z^-1))*(1-(1/3)
  *(z^-1)))
4 z = %z;
5 s = %s;
6 syms n t;
7 a = 0.5;
8 [A]=pfss((3-(5/6)*(z^-1))/((1-(1/4)*(z^-1))*(1-(1/3)
  *(z^-1))))
9 x1 = horner(A(1),z)
10 x2 = horner(A(2),z)
11 x3 = A(3)
12 x = x1+x2+x3
13 disp(x1,"ans=")
14 disp(x2,"ans=")
15 disp(x3,"ans=")
16 disp(x,"ans=")
17 //Result
18
19 //      0.6666667
20 //      -----
21 // - 0.3333333 + z
22
23 //      0.25
24 //      -----
25 // - 0.25 + z

```

```

26
27 //3
28
29 //sum of these , gives the original value
30 //
31 //      2
32 //      - 0.8333333z + 3z
33 //      -----
34 //      2
35 //      0.0833333 - 0.5833333z + z

```

Scilab code Exa 10.36 Output response of an LTI System

```

1 //clear//
2 //Example 10.36:To find output response of an LTI
  System
3 syms n z;
4 H = z/(z+3)
5 X = z/(z-1)
6 Y = X*H
7 F1 = Y*(z^(n-1))*(z-1);
8 y1 = limit(F1,z,1);
9 F2 = Y*(z^(n-1))*(z+3);
10 y2 = limit(F2,z,-3);
11 disp(y1*"u(n)" + y2*"u(n)", 'y[n]= ')
12 //Result
13 //y[n] = u(n)/4 - (-3)^(n+1)*u(n)/4
14 //Equivalent to = (1/4).u[n] - (3/4)(-3)^n.u[n]

```

Scilab code Exa 10.37 Output response of an LTI System

```

1 //clear//
2 //Example 10.37:To find output response of an LTI
  System

```



```

3 syms n z;
4 alpha = 8; //input constant
5 beta_b = 1; //initial condition y[-1] = 1
6 Y1 = -((3*beta_b*z)/(z+3))
7 Y2 = (alpha*z^2/((z+3)*(z-1)))
8 F1 = Y1*(z^(n-1))*(z+3);
9 y1 = limit(F1,z,-3);
10 F2 = Y2*(z^(n-1))*(z+3);
11 y2 = limit(F2,z,-3);
12 F3 = Y2*(z^(n-1))*(z-1);
13 y3 = limit(F3,z,1);
14 disp((y1+y2+y3)*'u(n)', 'y[n]=')
15 //Result
16 //y[n] = (2-(-3)^(n+1))*u(n)

```

Chapter 11

Linear Feedback Systems

Scilab code Exa 11.1 Root locus Analysis of Linear Feedback Systems

```
1 //clear//
2 //Example11.1:Root locus Analysis of Linear Feedback
  Systems
3 //Continuous Time Systems
4 //Refer figure 11.12(a) in Openhiem &Willksy page
  840
5 s = %s;
6 H = syslin('c',[1/(s+1)]);
7 G = syslin('c',[1/(s+2)]);
8 F = G*H;
9 clf;
10 evans(F,3)
```

Scilab code Exa 11.2 Continuous Time Systems

```
1 //clear//
2 //Example11.2:Root locus Analysis of Linear Feedback
  Systems
```

```

3 //Continuous Time Systems
4 //Refer figure 11.14(a) in Openhiem &Willksy page
   844
5 s = %s;
6 G = syslin('c',[(s-1)/((s+1)*(s+2))]);
7 clf;
8 evans(G,2)

```

Scilab code Exa 11.3 Discrete time system

```

1 //clear//
2 //Example11.3:Root locus Analysis of Linear Feedback
   Systems
3 ////Discrete time system
4 //Refer figure 11.16(a) in Openhiem &Willksy page
   846
5 z = %z;
6 G = syslin('d',[z/((z-0.5)*(z-0.25))]);
7 clf;
8 evans(G,2)

```

Scilab code Exa 11.05 Nyquist criterion for Continuous Time Systems

```

1 //clear//
2 //Example 11.5:Nyquist criterion for Continuous Time
   Systems
3 //Nyquist Plot
4 s = %s;
5 //Open Loop Transfer Function
6 G = syslin('c',[1/(s+1)]);
7 H = syslin('c',[1/(0.5*s+1)]);
8 F = G*H;
9 clf;

```

```
10 nyquist(F)
11 show_margins(F, 'nyquist')
```

Scilab code Exa 11.5 Bode Plot

```
1 //clear//
2 //Example 11.5:Nyquist criterion for Continuous Time
  Systems
3 //Bode Plot
4 s = %s;
5 //Open Loop Transfer Function
6 G = syslin('c',[1/(s+1)]);
7 H = syslin('c',[1/(0.5*s+1)]);
8 F = G*H;
9 clf;
10 bode(F,0.01,100)
11 show_margins(F)
```

Scilab code Exa 11.6 Nyquist Plot

```
1 //clear//
2 //Example 11.6:Nyquist criterion for Continuous Time
  Systems
3 //Nyquist Plot
4 s = %s;
5 //Open Loop Transfer Function
6 F = syslin('c',[(s+1)/((s-1)*(0.5*s+1))])
7 clf;
8 nyquist(F)
9 show_margins(F, 'nyquist')
```

Scilab code Exa 11.7 Nyquist Plot

```
1 //clear//
2 //Example 11.7
3 //Nyquist Plot
4 s = %s;
5 T =1;
6 //Open Loop Transfer Function
7 G = syslin('c',[-%e^(-s*T)]);
8 clf;
9 nyquist(G)
10 show_margins(G,'nyquist')
```

Scilab code Exa 11.8 Nyquist Plot

```
1 //clear//
2 //Example 11.8:Nyquist criterion for Discrete Time
   Systems
3 //Nyquist Plot
4 //Discrete Time System
5 z = %z;
6 //Open Loop Transfer Function
7 F = syslin('d',[1/(z*(z+0.5))])
8 clf;
9 nyquist(F)
10 show_margins(F,'nyquist')
```

Scilab code Exa 11.09 Root locus analysis of Linear feedback systems

```
1 //clear//
2 //Figure11.9:Root locus analysis of Linear feedback
   systems
3 s = %s;
```

```

4 beta_b1 = 1;
5 beta_b2 = -1;
6 G1 = syslin('c',[2*beta_b1/s]);
7 G2 = syslin('c',[2*beta_b2/s]);
8 H = syslin('c',[s/(s-2)]);
9 F1 = G1*H;
10 F2 = G2*H;
11 clf;
12 evans(F1,2)
13 figure
14 evans(F2,2)

```

Scilab code Exa 11.9 Gain and Phase Margins

```

1 //clear//
2 //Example 11.9:Gain and Phase Margins and their
3 //associated cross over frequencies
4 s =poly(0,'s'); // Define ss as polynomial variable
5 //Create s transfer function in forward path
6 F = syslin('c',[(4*(1+0.5*s))/(s*(1+2*s)*(1+0.05*s
   +(0.125*s)^2))])
7 B = syslin('c',(1+0*s)/(1+0*s))
8 OL = F*B;
9 fmin = 0.01; // Min freq in Hz
10 fmax = 10; // Max freq in Hz
11 scf(1);
12 //clf;
13 // Plot frequency response of open loop transfer
   function
14 bode(OL,0.01,10);
15 // display gain and phase margin and cross over
   frequencies
16 show_margins(OL);
17 [gm,fr1] = g_margin(OL)
18 [phm,fr2] = p_margin(OL)

```

```
19 disp(gm, 'gain margin in dB')
20 disp(fr1, 'gain cross over frequency in Hz')
21 disp(phm, 'phase margin in dB')
22 disp(fr2, 'phase cross over frequency in Hz')
```
