

Scilab Textbook Companion for
Digital Signal Processing
by S. Salivahanan, A. Vallavaraj And C.
Gnanapriya¹

Created by
Priya Sahani
B TECH EXTC
Electrical Engineering
V.J.T.I
College Teacher
Rizwn Ahmed
Cross-Checked by
Lavitha Pareira

July 30, 2019

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: Digital Signal Processing

Author: S. Salivahanan, A. Vallavaraj And C. Gnanapriya

Publisher: Tata McGraw - Hill, New Delhi

Edition: 1

Year: 2008

ISBN: 978-0-07-463996-2

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 Classifications of signals and systems	5
2 Fourier Analysis of Preiodic and Aperiodic Continuous Time Signals and Systems	7
3 Applications of Laplace Transform to System Analysis	16
4 Z Transforms	18
5 Linear Time Invariant Systems	22
6 Discrete and Fast Fourier Transforms	24
7 Finite Impulse Response Filters	41
8 Infinite Impulse Response Filters	43
9 Realisation of Digital Linear Systems	51
10 Effects of Finite Word Length in Digital Filters	53
11 Multirate Digital Signal Processing	57

List of Scilab Codes

Exa 1.2.a	Rectangular wave	5
Exa 1.2.b	Rectangular wave	5
Exa 1.2.c	Cosine wave	6
Exa 1.2.d	Ramp wave	6
Exa 2.1	Fourier Series of Periodic Square Wave . . .	7
Exa 2.2	Fourier Series of Periodic Rectangular Wave	8
Exa 2.3	Fourier Series of Periodic Half Wave Rectified Sine Wave	9
Exa 2.4	Fourier Series of Periodic Triangular Wave .	10
Exa 2.5	Fourier Series of Periodic Rectangular Pulse	11
Exa 2.6	Fourier Series of Square Wave	12
Exa 2.8	Complex fourier series representation	14
Exa 3.10	Poles and Zeros	16
Exa 3.11	Poles and zeros	16
Exa 3.12	Poles and zeros	17
Exa 4.2	Z transform	18
Exa 4.4	Z transform	19
Exa 4.13	Convolution	19
Exa 4.14	Convolution	20
Exa 4.16	Cross correlation	20
Exa 4.19	System response	21
Exa 5.20	System response	22
Exa 5.21	Poles and zeros	22
Exa 6.1	Linear and Circular convolution	24
Exa 6.2	FIR filter response	25
Exa 6.3	Convolution	26
Exa 6.4	Convolution	26
Exa 6.5	Convolution	27

Exa 6.6	Convolution	28
Exa 6.8	DFT	29
Exa 6.9	DFT	29
Exa 6.10	DFT	30
Exa 6.11	DFT	30
Exa 6.12	DFT	31
Exa 6.13	Inverse DFT	31
Exa 6.14	Inverse DFT	32
Exa 6.15	DIT FFT	32
Exa 6.16	DIT FFT	32
Exa 6.17	DIT FFT	33
Exa 6.18	DIT FFT	33
Exa 6.19	DIF FFT	33
Exa 6.20	DIF FFT	34
Exa 6.21	DIF FFT	34
Exa 6.22	DIF FFT	34
Exa 6.23	IFFT	35
Exa 6.24	IFFT	35
Exa 6.25	IFFT	36
Exa 6.26	IFFT	36
Exa 6.27	IFFT	36
Exa 6.34	Overlap Add Convolution	37
Exa 6.35	Overlap Save Convolution	38
Exa 6.36	Cross Correlation	38
Exa 6.37	Circular Correlation	39
Exa 7.3	Low pass filter using fourier series method .	41
Exa 7.4	Low pass filter using Type 1 frequency sam- pling technique	42
Exa 8.1	IIR filter Design byBackward Difference For Derivative method	43
Exa 8.2	IIR filter Design byBackward Difference For Derivative method	43
Exa 8.3	IIR filter Design byBackward Difference For Derivative method	44
Exa 8.4	IIR filter Design by Impulse Invariant method	44
Exa 8.5	IIR filter Design by Impulse Invariant method	45
Exa 8.6	IIR filter Design by Impulse Invariant method	45

Exa 8.7	IIR filter Design by Bilinear Transformation method	46
Exa 8.8	IIR filter Design by Bilinear Transformation method	46
Exa 8.9	IIR filter Design by Bilinear Transformation method	46
Exa 8.10	IIR filter Design by Bilinear Transformation method	47
Exa 8.11	Butterworth Filter using Impulse Invariant transformation	47
Exa 8.12	Butterworth Filter using Bilinear transformation	48
Exa 8.14	Filter transformation	50
Exa 8.15	Filter transformation	50
Exa 9.4	Cascade Realisation	51
Exa 9.5.a	Parallel Realisation	51
Exa 9.5.b	Parallel Realisation	52
Exa 10.2	Output Quantisation Noise	53
Exa 10.3	Deadband Interval	54
Exa 10.4	Deadband Interval	54
Exa 10.5	Output Quantisation Noise for Cascade realisation	55
Exa 11.1	Time Decimation	57
Exa 11.2	Interpolation	57
Exa 11.4	Polyphase Decomposition	58
Exa 11.5	Decimator implementation	59
Exa 11.6	Decimator implementation	60
Exa 12.2	Power Spectrum	63
Exa 12.4	Frequency resolution	64

Chapter 1

Classifications of signals and systems

Scilab code Exa 1.2.a Rectangular wave

```
1 //Example 1.2 (a)
2 clc;clear;
3 t=-5:0.01:5;
4 x=1*(abs(2*t+3)<0.5);
5 plot(t,x);
6 title('x(t)=rect(2t+3)');
```

Scilab code Exa 1.2.b Rectangular wave

```
1 //Example 1.2 (b)
2 clc;clear;
3 t=-5:0.01:5;
4 x=2*(abs(t-1/4)<0.5);
5 plot(t,x);
6 title('x(t)=2*rect(t-1/4)');
```

Scilab code Exa 1.2.c Cosine wave

```
1 //Example 1.2 (c)
2 clc;clear;
3 pi=22/7;
4 t=-5:0.01:5;
5 x=cos(2*pi*t-50*pi);
6 plot(t,x);
7 title('x(t)=cos(2*pi*t-50*pi)');
```

Scilab code Exa 1.2.d Ramp wave

```
1 //Example 1.2 (d)
2
3 clc;clear;
4 t=-5:0.01:5;
5 x=-0.5*(t-4);
6 plot(t,x);
7 title('x(t)=r(-0.5t+2)');
8 zoom_rect([-5 0 5 5]);
```

Chapter 2

Fourier Analysis of Periodic and Aperiodic Continuous Time Signals and Systems

Scilab code Exa 2.1 Fourier Series of Periodic Square Wave

```
1 //Example 2.1
2
3 clc;clear;close;
4 A=1;T=2;
5 w0=2*pi/T;
6
7 //Calculation of trigonometric fourier series co-
  coefficients
8 a0=A/T*(integrate('-1','t',-T/2,-T/4)+integrate('+1',
  't',-T/4,T/4)+integrate('-1','t',T/4,T/2));
9 for n=1:10;
10 a(1,n)=2*A/T*(integrate('-cos(n*w0*t)','t',-T/2,-T
  /4)+integrate('+cos(n*w0*t)','t',-T/4,T/4)+
  integrate('-cos(n*w0*t)','t',T/4,T/2));
11 b(1,n)=2*A/T*(integrate('-sin(n*w0*t)','t',-T/2,-T
  /4)+integrate('+sin(n*w0*t)','t',-T/4,T/4)+
  integrate('-sin(n*w0*t)','t',T/4,T/2));
```

```

12 end
13
14 //Displaying fourier coefficients
15 disp(T, 'fundamental period T= ', A, 'Assumption:
    Amplitude A= ');
16 disp('Trigonometric fourier series coefficients:');
17 disp(a0, 'a0= '); disp(a, 'an= '); disp(b, 'bn= ');
18
19 x=[-A*ones(1,25) A*ones(1,50) -A*ones(1,25)] //
    Function for plotting purpose
20 t=-T/2:0.01*T:T/2-0.01;
21 subplot(311); plot(t,x);
22 title('x(t)'); xlabel('time t');
23 subplot(312); plot2d3(a);
24 title('Coefficients an'); xlabel('n');
25 subplot(313); plot2d3(b);
26 title('Coefficients bn'); xlabel('n');

```

Scilab code Exa 2.2 Fourier Series of Periodic Rectangular Wave

```

1 //Example 2.2
2
3 clc; clear; close;
4 A=1; T=2;
5 w0=2*pi/T;
6
7 //Calculation of trigonometric fourier series co-
    efficients
8 a0=A/T*integrate('1', 't', -T/4, T/4);
9 for n=1:10;
10 a(1,n)=2*A/T*integrate('cos(n*w0*t)', 't', -T/4, T/4);
11 b(1,n)=2*A/T*integrate('sin(n*w0*t)', 't', -T/4, T/4);
12 end
13
14 //Displaying fourier coefficients

```

```

15 disp(T, 'fundamental period T= ', A, 'Assumption:
    Amplitude A= ');
16 disp('Trigonometric fourier series coefficients:');
17 disp(a0, 'a0= '); disp(a, 'an= '); disp(b, 'bn= ');
18
19 x=[zeros(1,25) A*ones(1,50) zeros(1,25)];
20 t=-T/2:0.01*T:T/2-0.01;
21 subplot(311); plot(t,x);
22 title('x(t)'); xlabel('time t');
23 subplot(312); plot2d3(a);
24 title('Coefficients an'); xlabel('n');
25 subplot(313); plot2d3(b);
26 title('Coefficients bn'); xlabel('n');

```

Scilab code Exa 2.3 Fourier Series of Periodic Half Wave Rectified Sine Wave

```

1 //Example 2.3
2
3 clc; clear; close;
4 A=1; T=2;
5 w0=2*pi/T;
6
7 //Calculation of trigonometric fourier series coefficients
8 a0=A/T*integrate('sin(w0*t)', 't', 0, T/2);
9 for n=1:10;
10 a(1,n)=2*A/T*integrate('sin(w0*t)*cos(n*w0*t)', 't',
    ,0, T/2);
11 b(1,n)=2*A/T*integrate('sin(w0*t)*sin(n*w0*t)', 't',
    ,0, T/2);
12 end
13
14 //Displaying fourier coefficients
15 disp(T, 'fundamental period T= ', A, 'Assumption:
    Amplitude A= ');

```

```

16 disp('Trigonometric fourier series coefficients:');
17 disp(a0, 'a0= '); disp(a, 'an= '); disp(b, 'bn= ');
18
19 t=0:0.01*T:T/2;
20 x=[A*sin(w0*t) zeros(1,50)];
21 t=0:0.01*T:T;
22 subplot(311); plot(t,x);
23 title('x(t)'); xlabel('time t');
24 subplot(312); plot2d3(a);
25 title('Coefficients an'); xlabel('n');
26 subplot(313); plot2d3(b);
27 title('Coefficients bn'); xlabel('n');

```

Scilab code Exa 2.4 Fourier Series of Periodic Triangular Wave

```

1 //Example 2.4
2
3 clc; clear; close;
4 A=1; T=2;
5 w0=2*pi/T;
6
7 //Calculation of trigonometric fourier series co-
  coefficients
8 a0=4*A/T*(integrate('t-0.5*T', 't', -T/2, -T/4)+
  integrate('t', 't', -T/4, T/4)+integrate('-t+0.5*T',
  't', T/4, T/2));
9 for n=1:10;
10 a(1,n)=2*4*A/T*(integrate('(t-0.5*T)*cos(n*w0*t)', 't',
  -T/2, -T/4)+integrate('t*cos(n*w0*t)', 't', -T/4, T/4)+
  integrate('(-t+0.5*T)*cos(n*w0*t)', 't', T/4, T/2));
11 b(1,n)=2*4*A/T*(integrate('(t-0.5*T)*sin(n*w0*t)', 't',
  -T/2, -T/4)+integrate('t*sin(n*w0*t)', 't', -T/4, T/4)+
  integrate('(-t+0.5*T)*sin(n*w0*t)', 't', T/4, T/2));

```

```

12 end
13
14 //Displaying fourier coefficients
15 disp(T, 'fundamental period T= ', A, 'Assumption:
    Amplitude A= ');
16 disp('Trigonometric fourier series coefficients:');
17 disp(a0, 'a0= '); disp(a, 'an= '); disp(b, 'bn= ');
18
19 t=-T/2:0.01*T:T/2;
20 x=[-4*A/T*t(1:25)-2*A 4*A/T*t(26:75) -4*A/T*t
    (76:101)+2*A];
21 subplot(311); plot(t,x);
22 title('x(t)'); xlabel('time t');
23 subplot(312); plot2d3(a);
24 title('Coefficients an'); xlabel('n');
25 subplot(313); plot2d3(b);
26 title('Coefficients bn'); xlabel('n');

```

Scilab code Exa 2.5 Fourier Series of Periodic Rectangular Pulse

```

1 //Example 2.5
2
3 clc; clear; close;
4 A=1; T=2; d=0.1;
5 w0=2*pi/T;
6
7 //Calculation of trigonometric fourier series co-
    efficients
8 a0=A/T*integrate('1', 't', -T/4, T/4);
9 for n=1:10;
10 a(1,n)=2*A/T*integrate('cos(n*w0*t)', 't', -d/2, d/2);
11 b(1,n)=2*A/T*integrate('sin(n*w0*t)', 't', -d/2, d/2);
12 end
13
14 //Displaying fourier coefficients

```

```

15 disp(d, 'pulse width d= ', T, 'fundamental period T= ',
    A, 'Assumption: Amplitude A= ');
16 disp('Trigonometric fourier series coefficients:');
17 disp(a0, 'a0= '); disp(a, 'an= '); disp(b, 'bn= ');
18
19 n=round(50*d/T); //
    Variable used for plotting pulses accurately
20 x=[zeros(1,50-n) A*ones(1,2*n+1) zeros(1,50-n)]
21 t=-T/2:0.01*T:T/2;
22 subplot(311); plot(t,x);
23 title('x(t)'); xlabel('time t');
24 subplot(312); plot2d3(a);
25 title('Coefficients an'); xlabel('n');
26 subplot(313); plot2d3(b);
27 title('Coefficients bn'); xlabel('n');

```

Scilab code Exa 2.6 Fourier Series of Square Wave

```

1 //Example 2.6
2
3 clc;clear;close;
4 A=1;T=2;
5 w0=2*pi/T;
6
7 //Calculation of trigonometric fourier series coefficients
8 a0=A/T*(integrate('-1','t',-T/2,0)+integrate('+1','t',0,T/2));
9 for n=1:10
10 a(1,n)=2*A/T*(integrate('-cos(n*w0*t)','t',-T/2,0)+
    integrate('+cos(n*w0*t)','t',0,T/2));
11 b(1,n)=2*A/T*(integrate('-sin(n*w0*t)','t',-T/2,0)+
    integrate('+sin(n*w0*t)','t',0,T/2));
12 end
13 a=clean(a);b=clean(b); //Function used to

```



```

        round small entities to zero
14
15 //Calculation of exponential fourier series co-
    efficients
16 function y=f(t),y=complex(cos(n*w0*t),-sin(n*w0*t)),
    endfunction;
17 for n=-10:10
18 c(1,n+11)=A/T*(-1*intc(-T/2,0,f)+intc(0,T/2,f));
19 end
20 c=clean(c); //Function used to
    round small entities to zero
21
22 //Calculation of trigonometric fourier series co-
    efficients from exponential fourie series
    coefficients
23 a01=c(1);
24 a1=2*real(c(12:21));
25 b1=-2*imag(c(12:21));
26
27 //Displaying fourier coefficients
28 disp(T,'fundamental period T= ',A,'Assumption:
    Amplitude A= ');
29 disp('Tigonometric fourier series co-efficients:');
30 disp(a0,'a0= ');disp(a,'an= ');disp(b,'bn= ');
31 disp('Exponential fourier series co-efficients');
32 disp(c(11),'c0= ');disp(c(12:21),'cn= ');disp(c
    (10:-1:1),'c-n= ');
33 disp('Trigonometric fourier series co-efficients from
    exponential coefficients:');
34 disp(a01,'a0= ');disp(a1,'an= ');disp(b1,'bn= ');
35 disp('The co-effifcients obtained are same by both
    methods')
36
37 x=[-A*ones(1,50) A*ones(1,51)];
38 t=-T/2:0.01*T:T/2;
39 n=-10:10;
40 subplot(311);plot(t,x);
41 title('x(t)');xlabel('time t');

```

```

42 subplot(312);plot2d3(a);
43 title('Coefficients an');xlabel('n');
44 subplot(313);plot2d3(b);
45 title('Coefficients bn');xlabel('n');
46 figure;
47 subplot(311);plot(t,x);
48 title('x(t)');xlabel('time t');
49 subplot(312);plot2d3(n,abs(c));
50 title('Magnitude of Coefficients |c|');xlabel('n');
51 subplot(313);plot2d3(n,atan(c));
52 title('Phase of Coefficients /_c');xlabel('n');

```

Scilab code Exa 2.8 Complex fourier series representation

```

1 //Example 2.8
2
3 clc;clear;close;
4 t=poly(0,'t');
5 //cn=3/(4+(n*pi)^2)
6 Pt=0.669; //Total energy
7 Preq=0.999*Pt; //Required energy
8 c0=3/(4+(0*pi)^2);
9 disp(c0,'c0=');
10 P=(abs(c0))^2;
11 c=[];n=0;
12 while P<Preq
13     n=n+1;
14     c(n)=3/(4+(n*pi)^2);
15     disp(c(n),'cn=',n,'n=');
16     P=P+2*(abs(c(n)))^2;
17 end
18 disp(Pt,'Total power Pt=');
19 disp(Preq,'99.9% of total power Preq=');
20 disp(n,'To include 99.9% of energy, we need to retain
    n terms where n=');

```


Chapter 3

Applications of Laplace Transform to System Analysis

Scilab code Exa 3.10 Poles and Zeros

```
1 //Example 3.10
2
3 clc;clear;close;
4 s=poly(0,'s');
5 I=3*s/(s+2)/(s+4);
6 disp(I,'Given Transfer Function:');
7 zero=roots(numer(I));
8 pole=roots(denom(I));
9 disp(zero,'Zeros of transfer function: ');
10 disp(pole,'Poles of transfer function: ');
11 plzr(I);
```

Scilab code Exa 3.11 Poles and zeros

```
1 //Example 3.11
2
```

```
3 clc; clear; close;
4 s=poly(0, 's');
5 F=4*(s+1)*(s+3)/(s+2)/(s+4);
6 disp(F, 'Given Transfer Function: ');
7 zero=roots(numer(F));
8 pole=roots(denom(F));
9 disp(zero, 'Zeros of transfer function: ');
10 disp(pole, 'Poles of transfer function: ');
11 plzr(F);
```

Scilab code Exa 3.12 Poles and zeros

```
1 //Example 3.12
2
3 clc; clear; close;
4 s=poly(0, 's');
5 F=10*s/(s^2+2*s+2);
6 disp(F, 'Given Transfer Function: ');
7 zero=roots(numer(F));
8 pole=roots(denom(F));
9 disp(zero, 'Zeros of transfer function: ');
10 disp(pole, 'Poles of transfer function: ');
11 plzr(F);
```

Chapter 4

Z Transforms

Scilab code Exa 4.2 Z transform

```
1 //Example 4.2
2
3 clc;clear;close;
4 z=poly(0,'z');
5 x1=[3 1 2 5 7 0 1];n1=-3:3;
6 X1=x1*(z^-n1)';
7 x2=[2 4 5 7 0 1 2];n2=-2:4;
8 X2=x2*(z^-n2)';
9 x3=[1 2 5 4 0 1]; n3=0:5;
10 X3=x3*(z^-n3)';
11 x4=[0 0 1 2 5 4 0 1];n4=0:7;
12 X4=x4*(z^-n4)';
13 X5=z^0;
14 X6=z^-5;
15 X7=z^5;
16 disp(X1,'x1(n)={3,1,2,5,7,0,1} X1(z)=');
17 disp(X2,'x2(n)={2,4,5,7,0,1,2} X2(z)=');
18 disp(X3,'x3(n)={1,2,5,4,0,1} X3(z)=');
19 disp(X4,'x4(n)={0,0,1,2,5,4,0,1} X4(z)=');
20 disp(X5,'x5(n)=delta(n) X5(z)=');
21 disp(X6,'x6(n)=delta(n-5) X6(z)=');
```

```
22 disp(X7, 'x7(n)=delta(n+5) X7(z)=');
```

Scilab code Exa 4.4 Z transform

```
1 //Example 4.4
2
3 clc;clear;close;
4 z=poly(0, 'z');
5 x=[1 3 0 0 6 -1];n=-1:4;
6 X=x*(z^-n)';
7 disp(X, 'x(n)={1,3,0,0,6,-1} X(z)=');
```

Scilab code Exa 4.13 Convolution

```
1 //Example 4.13
2
3 clc;clear;close;
4 z=poly(0, 'z');
5 x1=[4 -2 1];n1=0:length(x1)-1;
6 X1=x1*(z^-n1)';
7 x2=[1 1 1 1 1];n2=0:length(x2)-1;
8 X2=x2*(z^-n2)';
9 X3=X1*X2;
10 l=coeff( numer(X3) );
11 x3=l(:, $:-1:1);
12 disp(X1, 'x1(n)={4,-2,1} X1(z)=');
13 disp(X2, 'x2(n)={1,1,1,1,1} X2(z)=');
14 disp(X3, 'Z transform of convolution of the two
    signals X3(z)=');
15 disp(x3, 'Convolution result of the two signals=');
```

Scilab code Exa 4.14 Convolution

```
1 //Example 4.14
2
3 clc; clear; close;
4 z=poly(0, 'z');
5 x1=[2 1 0 0.5]; n1=0:length(x1)-1;
6 X1=x1*(z-n1)';
7 x2=[2 2 1 1]; n2=0:length(x2)-1;
8 X2=x2*(z-n2)';
9 X3=X1*X2;
10 l=coeff(numer(X3));
11 x3=l(:,$:-1:1);
12 disp(X1, 'x1(n)={2,1,0,0.5} X1(z)=');
13 disp(X2, 'x2(n)={2,2,1,1} X2(z)=');
14 disp(X3, 'Z transform of convolution of the two
    signals X3(z)=');
15 disp(x3, 'Convolution result of the two signals=')
```

Scilab code Exa 4.16 Cross correlation

```
1 //Example 4.16
2
3 clc; clear; close;
4 z=poly(0, 'z');
5 x1=[1 2 3 4]; n1=0:length(x1)-1;
6 X1=x1*(z-n1)';
7 x2=[4 3 2 1]; n2=0:length(x2)-1;
8 X2=x2*(z-n2)';
9 X2_=x2*(zn2)';
10 X3=X1*X2_;
11 l=coeff(numer(X3));
12 x3=l(:,$:-1:1);
13 disp(X1, 'x1(n)={4,-2,1} X1(z)=');
14 disp(X2, 'x2(n)={4,-2,1} X2(z)=');
```



```
15 disp(X3, 'Z transform of cross crrelation of the two
    signals X3(z)=');
16 disp(x3, 'Cross correlation result of the two signals
    = ')
```

Scilab code Exa 4.19 System response

```
1 //Example 4.19
2
3 clc;clear;close;
4 z=poly(0, 'z ');
5 h=[1 2 3];n1=0:length(h)-1;
6 H=h*(z^-n1)';
7 y=[1 1 2 -1 3];n2=0:length(y)-1;
8 Y=y*(z^-n2)';
9 X=Y/H;
10 l=coeff( numer(X));
11 x=l(:, $:-1:1);
12 disp(H, 'h(n)={1,2,3} H(z)=');
13 disp(Y, 'y(n)={1,1,2,-1,3} Y(z)=');
14 disp(X, 'Z transform of input sequence X(z)=');
15 disp(x, 'Inpput Sequence = ')
```

Chapter 5

Linear Time Invariant Systems

Scilab code Exa 5.20 System response

```
1 //Example 5.20
2
3 clc;clear;close;
4 z=poly(0,'z');
5 x=[-1 1 0 -1];n=0:length(x)-1;
6 X=x*(z^-n)';
7 H=0.2-0.5*z^-2+0.4*z^-3
8 Y=H*X;
9 l=coeff( numer(Y));
10 y=l(:, $:-1:1);
11 disp(X, 'Input sequence x(n)={-1,1,0,-1} X(z)=');
12 disp(H, 'System Transfer Function H(z)=');
13 disp(Y, 'Z transform of output response Y(z)=');
14 disp(y, 'Digital output sequence y=')
```

Scilab code Exa 5.21 Poles and zeros

```
1 //Example 5.21
```

```
2  clc;clear;close;
3  z=poly(0,'z');
4  H=(1+z^-1)/(1+3/4*z^-1+1/8*z^-2);
5  pole=roots( numer(H));
6  zero=roots( denom(H));
7  disp(H,'System Transfer Function H(z)=');
8  disp(zero,'System zeros are at ');
9  disp(pole,'System poles are at ');
10 plzr(H);
```

Chapter 6

Discrete and Fast Fourier Transforms

Scilab code Exa 6.1 Linear and Circular convolution

```
1 //Example 6.1
2 clc;clear;close;
3 x1=[1 1 2 2];
4 x2=[1 2 3 4];
5 ylength=length(x1);
6 //Calculation of linear convolution
7 z=convol(x1,x2);
8 //Calculation of circular convolution
9 for n=1:ylength
10     y(n)=0;
11     for k=1:ylength,
12         l=n-k+1;
13         if l <= 0 then
14             l=1+ylength;
15         end
16         y(n)=y(n)+(x1(k)*x2(l));
17     end
18 end
19 //Calculation of circular convolution using DFT and
```

```

IDFT
20 X1=fft(x1,-1);
21 X2=fft(x2,-1);
22 Y1=X1.*X2;
23 y1=fft(Y1,1);
24 y1=clean(y1);
25 disp(z,'Linear Convolution sequence is z(n): ');
26 disp(y,'Circular Convolution sequence is y(n): ');
27 disp(y1,'Circular Convolution sequence calculated
    using DFT-IDFT method is y(n): ');

```

Scilab code Exa 6.2 FIR filter response

```

1 //Example 6.2
2
3 clc;clear;close;
4 x=[1 2];
5 h=[1 2 4];
6
7 //Calculation of linear convolution
8 y=convol(x,h);
9 disp(x,'Input Sequence is x(n): ');
10 disp(h,'Impulse response of FIR filter h(n): ');
11 disp(y,'Output sequence is y(n): ');
12 subplot(3,1,1);
13 plot2d3(x);
14 title('Input Sequence x[n]: ');ylabel('Amplitude-->');
    xlabel('n-->')
15 subplot(3,1,2);
16 plot2d3(h);
17 title('Impulse Response h[n]: ');ylabel('Amplitude-->');
    xlabel('n-->')
18 subplot(3,1,3);
19 plot2d3(y);
20 title('Output Sequence y[n]=x[n]*h[n] : ');ylabel('

```

Amplitude—>');xlabel('n—>')

Scilab code Exa 6.3 Convolution

```
1 //Example 6.3
2
3 clc;clear;close;
4 x=[1 1 1];
5 h=[1 1 1];
6
7 //Calculation of linear convolution
8 y=convol(x,h);
9 disp(x,'First Sequence is x(n): ');
10 disp(h,'Second Sequence is h(n): ');
11 disp(y,'Output sequence is y(n): ');
12 subplot(3,1,1);
13 plot2d3(x);
14 title('First Sequence x[n]: ');ylabel('Amplitude—>');
    xlabel('n—>')
15 subplot(3,1,2);
16 plot2d3(h);
17 title('Second Sequence h[n]: ');ylabel('Amplitude—>')
    ;xlabel('n—>')
18 subplot(3,1,3);
19 plot2d3(y);
20 title('Convolution Sequence y[n]=x[n]*h[n] : ');ylabel
    ('Amplitude—>');xlabel('n—>')
```

Scilab code Exa 6.4 Convolution

```
1 //Example 6.4
2
3 clc;clear;close;
```

```

4 a=0.5;
5 n=1:50;
6 x=ones(1,50);
7 h=a^n;
8
9 //Calculation of linear convolution
10 for i=1:50
11     y(1,i)=sum(h(1:i));
12 end
13 disp('First Sequence is x(n)=u(n) ');
14 disp(a,'Second Sequence is h(n)=a^n*u(n) where a= ');
15     ;
16 disp(y,'Output sequence is y(n): ');
17 subplot(3,1,1);
18 plot2d3(x);
19 title('First Sequence x[n]: ');ylabel('Amplitude—>');
20     xlabel('n—>')
21 subplot(3,1,2);
22 plot2d3(h);
23 title('Second Sequence h[n]: ');ylabel('Amplitude—>');
24     xlabel('n—>')
25 subplot(3,1,3);
26 plot2d3(y);
27 title('Convolution Sequence y[n]=x[n]*h[n] : ');ylabel
28     ('Amplitude—>');xlabel('n—>')

```

Scilab code Exa 6.5 Convolution

```

1 //Example 6.5
2 clc;clear;close;
3 x=[1 2 3];xmin=0;nx=xmin:length(x)+xmin-1;
4 h=[1 2 -2 -1];hmin=-1;nh=length(h)+hmin-1;
5
6 //Calculation of linear convolution
7 y=convol(x,h);

```

```

8  ymin=xmin+hmin;ny=ymin:length(y)+ymin-1;
9
10 disp(x,'First Sequence is x(n): ');
11 disp(h,'Second Sequence is h(n): ');
12 disp(y,'Output sequence is y(n): ');
13 subplot(3,1,1);
14 plot2d3(nx,x);
15 title('First Sequence x[n]: ');ylabel('Amplitude—>');
    xlabel('n—>')
16 subplot(3,1,2);
17 plot2d3(nh,h);
18 title('Second Sequence h[n]: ');ylabel('Amplitude—>')
    ;xlabel('n—>')
19 subplot(3,1,3);
20 plot2d3(ny,y);
21 title('Convolution Sequence y[n]=x[n]*h[n] : ');ylabel
    ('Amplitude—>');xlabel('n—>')

```

Scilab code Exa 6.6 Convolution

```

1 //Example 6.6
2
3 clc;clear;close;
4 x=[1 1 0 1 1];xmin=-2;nx=xmin:length(x)+xmin-1;
5 h=[1 -2 -3 4];hmin=-3;nh=length(h)+hmin-1;
6
7 //Calculation of linear convolution
8 y=convol(x,h);
9 ymin=xmin+hmin;ny=ymin:length(y)+ymin-1;
10
11 disp(x,'First Sequence is x(n): ');
12 disp(h,'Second Sequence is h(n): ');
13 disp(y,'Output sequence is y(n): ');
14 subplot(3,1,1);
15 plot2d3(nx,x);

```



```

16 title('First Sequence x[n]: ');ylabel('Amplitude—>');
    xlabel('n—>')
17 subplot(3,1,2);
18 plot2d3(nh,h);
19 title('Second Sequence h[n]: ');ylabel('Amplitude—>')
    ;xlabel('n—>')
20 subplot(3,1,3);
21 plot2d3(ny,y);
22 title('Convolution Sequence y[n]=x[n]*h[n] : ');ylabel
    ('Amplitude—>');xlabel('n—>')

```

Scilab code Exa 6.8 DFT

```

1 //Example 6.8
2
3 clc;clear;close;
4 L=3;A=1/4;
5 x=A*ones(1,L);
6 //Calculation of DFT
7 X=fft(x,-1);
8 X=clean(X);
9 disp(x,'Given Sequence is x(n): ');
10 disp(X,'DFT of the Sequence is X(k): ');

```

Scilab code Exa 6.9 DFT

```

1 //Example 6.9
2
3 clc;clear;close;
4 L=3;A=1/5;
5 n=-1:1;
6 x=A*ones(1,L);
7 //Calculation of DFT

```

```
8 X=fft(x,-1);
9 X=clean(X);
10 disp(x,'Given Sequence is x(n): ');
11 disp(X,'DFT of the Sequence is X(k): ');
```

Scilab code Exa 6.10 DFT

```
1 //Example 6.10
2
3 clc;clear;close;
4 x=[1 1 2 2 3 3];
5 //Calculation of DFT
6 X=fft(x,-1);
7 X=clean(X);
8 disp(x,'Given Sequence is x(n): ');
9 disp(X,'DFT of the Sequence is X(k): ');
10 subplot(3,1,1);
11 plot2d3(x);
12 title('Given Sequence x[n]: ');ylabel('Amplitude—&gt;');
13 xlabel('n—&gt;');
14 subplot(3,1,2);
15 plot2d3(abs(X));
16 title('Magnitude Spectrum |X(k)| ');xlabel('k—&gt;');
17 ;
18 subplot(3,1,3);
19 plot2d3(atan(X));
20 title('Phase Spectrum /_X(k) ');xlabel('k—&gt;');
```

Scilab code Exa 6.11 DFT

```
1 //Example 6.11
```

```

2
3 clc;clear;close;
4 N=8;A=1/4;
5 n=0:N-1;
6 x=A^n;
7 //Calculation of DFT
8 X=fft(x,-1);
9 X=clean(X);
10 disp(x,'Given Sequence is x(n): ');
11 disp(N,'N=')
12 disp(X,'N-point DFT of the Sequence is X(k): ');

```

Scilab code Exa 6.12 DFT

```

1 //Example 6.12
2
3 clc;clear;close;
4 N=4;
5 n=0:N-1;
6 x=cos(%pi/4*n);
7 //Calculation of DFT
8 X=fft(x,-1);
9 X=clean(X);
10 disp(x,'Given Sequence is x(n): ');
11 disp(X,'DFT of the Sequence is X(k): ');

```

Scilab code Exa 6.13 Inverse DFT

```

1 //Example 6.13
2 clc;clear;close;
3 X=[1 2 3 4];
4 //Calculation of IDFT
5 x=fft(X,1);

```

```
6 x=clean(x);
7 disp(X, 'DFT of the Sequence is X(k): ');
8 disp(x, 'Sequence is x(n): ');
```

Scilab code Exa 6.14 Inverse DFT

```
1 //Example 6.14
2 clc;clear;close;
3 X=[3 2+%i 1 2-%i];
4 //Calculation of IDFT
5 x=fft(X,1);
6 x=clean(x);
7 disp(X, 'DFT of the Sequence is X(k): ');
8 disp(x, 'Sequence is x(n): ');
```

Scilab code Exa 6.15 DIT FFT

```
1 //Example 6.15
2
3 clc;clear;
4 x=[1 2 3 4 4 3 2 1];
5 X=clean(fft(x));
6 disp(x, 'x(n)=');
7 disp(X, 'X(k)=');
```

Scilab code Exa 6.16 DIT FFT

```
1 //Example 6.16
2
3 clc;clear;
```

```
4 x=[0 1 2 3 4 5 6 7];
5 X=clean(fft(x));
6 disp(x, 'x(n)=');
7 disp(X, 'X(k)=');
```

Scilab code Exa 6.17 DIT FFT

```
1 //Example 6.17
2
3 clc;clear;
4 n=0:7;
5 x=2^n;
6 X=clean(fft(x));
7 disp(x, 'x(n)=');
8 disp(X, 'X(k)=');
```

Scilab code Exa 6.18 DIT FFT

```
1 //Example 6.18
2
3 clc;clear;
4 x=[0 1 2 3];
5 X=clean(fft(x));
6 disp(x, 'x(n)=');
7 disp(X, 'X(k)=');
```

Scilab code Exa 6.19 DIF FFT

```
1 //Example 6.19
2
```

```
3 clc; clear;
4 x=[1 2 3 4 4 3 2 1];
5 X=clean(fft(x));
6 disp(x, 'x(n)=');
7 disp(X, 'X(k)=');
```

Scilab code Exa 6.20 DIF FFT

```
1 //Example 6.20
2
3 clc; clear;
4 n=0:7;
5 x=2n;
6 X=clean(fft(x));
7 disp(x, 'x(n)=');
8 disp(X, 'X(k)=');
```

Scilab code Exa 6.21 DIF FFT

```
1 //Example 6.21
2
3 clc; clear;
4 n=0:7;
5 x=n+1;
6 X=clean(fft(x));
7 disp(x, 'x(n)=');
8 disp(X, 'X(k)=');
```

Scilab code Exa 6.22 DIF FFT

```

1 //Example 6.21
2
3 clc;clear;
4 n=0:3;
5 x=cos(n*pi/2);
6 X=clean(fft(x));
7 disp(x, 'x(n)=');
8 disp(X, 'X(k)=');

```

Scilab code Exa 6.23 IFFT

```

1 //Example 6.23
2
3 clc;clear;
4 X=[6 -2+2*i -2 -2-2*i];
5 x=clean(ifft(X));
6 disp(X, 'X(k)=');
7 disp(x, 'x(n)=');

```

Scilab code Exa 6.24 IFFT

```

1 //Example 6.24
2
3 clc;clear;
4 X=[20 -5.828-2.414*i 0 -0.172-0.414*i 0
      -0.172+0.414*i 0 -5.828+2.414*i];
5 x=round(clean(ifft(X)));
6 disp(X, 'X(k)=');
7 disp(x, 'x(n)=');

```

Scilab code Exa 6.25 IFFT

```
1 //Example 6.25
2
3 clc;clear;
4 X=[255 48.63+166.05*%i -51+102*%i -78.63+46.05*%i
    -85 -78.63-46.05*%i -51-102*%i 48.63-166.05*%i];
5 x=round(clean(iff(X)));
6 disp(X, 'X(k)=');
7 disp(x, 'x(n)=');
```

Scilab code Exa 6.26 IFFT

```
1 //Example 6.26
2
3 clc;clear;
4 X=[36 -4+9.656*%i -4+4*%i -4+1.656*%i -4 -4-1.656*%i
    -4-4*%i -4-9.656*%i ];
5 x=round(clean(iff(X)));
6 disp(X, 'X(k)=');
7 disp(x, 'x(n)=');
```

Scilab code Exa 6.27 IFFT

```
1 //Example 6.27
2
3 clc;clear;
4 t=0:0.0025:0.0175;
5 f=50;
6 x=sin(2*%pi*f*t);
7 X=clean(fft(x));
8 disp(x, 'x(n)=');
9 disp(X, 'X(k)=');
```

Scilab code Exa 6.34 Overlap Add Convolution

```
1 //Example 6.34
2
3 clc;clear;close;
4 h=[2 2 1];
5 x=[3 0 -2 0 2 1 0 -2 -1 0];
6 M=length(h); //length of impulse
   response
7 L=2^M; //length of FFT/IFFT
   operation
8 N=L-M+1;
9 x1=length(x);
10 K=ceil(x1/N); //number of iterations
11 h=[h zeros(1,L-M)];
12 x=[x x(1:K*N-x1)];
13 H=fft(h);
14 y=zeros(1,M-1);
15 for k=0:K-1
16     xk=[x(k*N+1:(k+1)*N) zeros(1,M-1)];
17     Xk=fft(xk);
18     Yk=H.*Xk;
19     yk=ifft(Yk);
20     yk=clean(yk);
21     y=[y(1:k*N) y(k*N+1:k*N+M-1)+yk(1:M-1) yk(M:L)];
22     disp(k+1,'Segment =');
23     disp(xk,'xk(n)=');
24     disp(yk,'yk(n)=');
25 end
26 y=y(1:x1+M-1);
27 disp(y,'Output Sequence is y(n): ');
```

Scilab code Exa 6.35 Overlap Save Convolution

```
1 //Example 6.35
2
3 clc;clear;close;
4 h=[2 2 1];
5 x=[3 0 -2 0 2 1 0 -2 -1 0];
6 M=length(h); //length of impulse
   response
7 L=2^M; //length of FFT/IFFT operation
8 N=L-M+1;
9 x1=length(x);
10 K=ceil(x1/N); //number of iterations
11 h=[h zeros(1,L-M)];
12 x=[zeros(1,M-1) x x(1:K*N-x1)];
13 H=fft(h);
14 for k=0:K-1
15     xk=x(k*N+1:(k+1)*N+M-1);
16     Xk=fft(xk);
17     Yk=H.*Xk;
18     yk=ifft(Yk);
19     yk=clean(yk);
20     y=[yk(1:k*N) yk(M:L)];
21     disp(k+1,'Segment =');
22     disp(xk,'xk(n)=');
23     disp(yk,'yk(n)=');
24 end
25 disp(y,'Output Sequence is y(n): ');
```

Scilab code Exa 6.36 Cross Correlation

```
1 //Example 6.36
2
3 clc;clear;close;
4 x=[1 0 0 1];
```

```

5 h=[4 3 2 1];
6 ylength=length(x)+length(h)-1;
7 xlength=length(x);
8 x=[zeros(1,length(h)-1) x zeros(1,length(h)-1)];
9 y=0;
10 //Calculation of cross correlation
11 for n=1:ylength;
12     y(n)=x*[zeros(1,n-1) h zeros(1,ylength-n)];
           //this instruction performs cross
           correlation of x & h
13 end
14
15 disp(x,'First Sequence is x(n): ');
16 disp(h,'Second Sequence is h(n): ');
17 disp(y,'Correlation Sequence y[n] is ');
18 figure;
19 subplot(3,1,1);
20 plot2d3(x);
21 title('First Sequence x[n]: ');ylabel('Amplitude—>');
   xlabel('n—>')
22 subplot(3,1,2);
23 plot2d3(h);
24 title('Second Sequence h[n]: ');ylabel('Amplitude—>')
   ;xlabel('n—>')
25 subplot(3,1,3);
26 plot2d3(y);
27 title('Correlation Sequence y[n]: ');ylabel('Amplitude
   —>');xlabel('n—>')

```

Scilab code Exa 6.37 Circular Correlation

```

1 //Example 6.37
2
3 clc;clear;close;
4 x=[1 0 0 1];

```

```

5 h=[4 3 2 1];
6 ylength=length(x);
7 y=0;
8 //Calculation of circular correlation
9 for n=1:ylength,
10     y(n)=0;
11     for k=1:ylength,
12         l=k-n+1;
13         if l <= 0 then
14             l=l+ylength;
15         end
16         y(n)=y(n)+(x(k)*h(l));
17     end
18     y(n)=y(n)/4;
19 end
20
21 disp(x,'First Sequence is x(n): ');
22 disp(h,'Second Sequence is h(n): ');
23 disp(y,'Correlation Sequence y[n] is ');
24 figure;
25 subplot(3,1,1);
26 plot2d3(x);
27 title('First Sequence x[n]: ');ylabel('Amplitude—>');
    xlabel('n—>')
28 subplot(3,1,2);
29 plot2d3(h);
30 title('Second Sequence h[n]: ');ylabel('Amplitude—>')
    ;xlabel('n—>')
31 subplot(3,1,3);
32 plot2d3(y);
33 title('Correlation Sequence y[n]: ');ylabel('Amplitude
    —>');xlabel('n—>')

```

Chapter 7

Finite Impulse Response Filters

Scilab code Exa 7.3 Low pass filter using fourier series method

```
1 //Example 7.3
2
3 clc;clear;close;
4 fp=2000; //passband frequency
5 F=9600; //sampling frequency
6
7 //Calculation of filter co-efficients
8 a0=1/F*integrate('1','t',-fp,fp);
9 for n=1:10;
10 a(1,n)=2/F*integrate('cos(2*%pi*n*f/F)','f',-fp,fp);
11 end
12 h=[a(:, $:-1:1)/2 a0 a/2];
13
14 //Displaying filter co-efficients
15 disp(F, 'Sampling frequency F= ',fp, 'Assumption:
    Passband frequency fp= ');
16 disp('Filter co-efficients: ');
17 disp(a0, 'h(0)= ');disp(a/2, 'h(n)=h(-n)=');
18
19 n=-10:10;
20 plot2d3(n,h);
```

```
21 title('Filter transfer function h(n)');xlabel('n-->')
    );
```

Scilab code Exa 7.4 Low pass filter using Type 1 frequency sampling technique

```
1 //Example 7.4
2
3 clc;clear;close;
4 M=7;w=2*%pi/M;
5
6 //Calculation of filter co-efficients
7 k=[0 1 6];
8 for n=0:M-1
9     h(n+1)=sum(exp(-%i*3*w*k).*exp(%i*w*k*n))/M;
10 end
11 h=clean(h);
12
13 //Displaying filter co-efficients
14 disp(M,'Filter Order M= ');
15 disp('Filter co-efficients:');
16 disp(h,'h(n)= ');
17
18 plot2d3(h);
19 title('Filter transfer function h(n)');xlabel('n-->')
    );
```

Chapter 8

Infinite Impulse Response Filters

Scilab code Exa 8.1 IIR filter Design by Backward Difference For Derivative method

```
1 //Example 8.1
2 clc;clear;close;
3 s=poly(0,'s');
4 z=poly(0,'z');
5 T=1;
6 Hs=1/(s+2);
7 Hz=horner(Hs,(1-1/z)/T);
8 disp('Using Backward difference formula for
      derivative:')
9 disp(Hs,'H(s)=');
10 disp(Hz,'H(z)=');
```

Scilab code Exa 8.2 IIR filter Design by Backward Difference For Derivative method

```
1 //Example 8.2
2 clc;clear;close;
```

```

3 s=poly(0, 's');
4 z=poly(0, 'z');
5 T=1;
6 Hs=1/(s^2+16);
7 Hz=horner(Hs, (1-1/z)/T);
8 disp('Using Backward difference formula for
      derivative:')
9 disp(Hs, 'H(s)=');
10 disp(Hz, 'H(z)=');

```

Scilab code Exa 8.3 IIR filter Design by Backward Difference For Derivative method

```

1 //Example 8.3
2 clc;clear;close;
3 s=poly(0, 's');
4 z=poly(0, 'z');
5 T=1;
6 Hs=1/((s+0.1)^2+9);
7 Hz=horner(Hs, (1-1/z)/T);
8 disp('Using Backward difference formula for
      derivative:')
9 disp(Hs, 'H(s)=');
10 disp(Hz, 'H(z)=');

```

Scilab code Exa 8.4 IIR filter Design by Impulse Invariant method

```

1 //Example 8.4
2 clc;clear;close;
3 s=poly(0, 's');
4 z=poly(0, 'z');
5 T=1;
6 Hs=(s+0.2)/((s+0.2)^2+9);
7 Hz=horner(Hs, (1-1/z)/T);

```



```
8 disp('Using Impulse Invariant Technique:')
9 disp(Hs, 'H(s)=');
10 disp(Hz, 'H(z)=');
```

Scilab code Exa 8.5 IIR filter Design by Impulse Invariant method

```
1 //Example 8.5
2 clc;clear;close;
3 s=poly(0, 's');
4 z=poly(0, 'z');
5 T=1;
6 Hs=1/(s+1)/(s+2);
7 Hz=horner(Hs, (1-1/z)/T);
8 disp('Using Impulse Invariant Technique:')
9 disp(Hs, 'H(s)=');
10 disp(Hz, 'H(z)=');
```

Scilab code Exa 8.6 IIR filter Design by Impulse Invariant method

```
1 //Example 8.6
2 clc;clear;close;
3 s=poly(0, 's');
4 z=poly(0, 'z');
5 T=1;
6 Hs=1/(s+0.5)/(s^2+0.5*s+2);
7 Hz=horner(Hs, (1-1/z)/T);
8 disp('Using Impulse Invariant Technique:')
9 disp(Hs, 'H(s)=');
10 disp(Hz, 'H(z)=');
```

Scilab code Exa 8.7 IIR filter Design by Bilinear Transformation method

```
1 //Example 8.7
2 clc;clear;close;
3 s=poly(0,'s');
4 z=poly(0,'z');
5 T=0.276;
6 Hs=(s+0.1)/((s+0.1)^2+9);
7 Hz=ss2tf(cls2dls(tf2ss(Hs),T));
8 disp('Using Bilinear Transformation:');
9 disp(Hs,'H(s)=');
10 disp(Hz,'H(z)=');
```

Scilab code Exa 8.8 IIR filter Design by Bilinear Transformation method

```
1 //Example 8.8
2 clc;clear;close;
3 s=poly(0,'s');
4 z=poly(0,'z');
5 T=0.1;
6 Hs=2/(s+1)/(s+2);
7 Hz=ss2tf(cls2dls(tf2ss(Hs),T));
8 disp(Hs,'H(s)=');
9 disp(Hz,'H(z)=');
```

Scilab code Exa 8.9 IIR filter Design by Bilinear Transformation method

```
1 //Example 8.9
2 clc;clear;close;
3 s=poly(0,'s');
4 z=poly(0,'z');
5 T=0.1;
6 wr=0.25*%pi; //Given cut off frequency
```

```

7 fc=2/T*tan(wr/2);
8 Hs=fc/(s+fc);
9 Hz=ss2tf(cls2dls(tf2ss(Hs),T));
10 disp('Using Bilinear Transformation:');
11 disp(Hs,'H(s)=');
12 disp(Hz,'H(z)=');

```

Scilab code Exa 8.10 IIR filter Design by Bilinear Transformation method

```

1 //Example 8.10
2 clc;clear;close;
3 s=poly(0,'s');
4 z=poly(0,'z');
5 T=0.1;
6 Hs=1/(s+1)^2;
7 Hz=ss2tf(cls2dls(tf2ss(Hs),T));
8 disp('Using Bilinear Transformation:');
9 disp(Hs,'H(s)=');
10 disp(Hz,'H(z)=');

```

Scilab code Exa 8.11 Butterworth Filter using Impulse Invariant transformation

```

1 //Example 8.11
2 clc;clear;close;
3 rp=0.707 //passband ripple
4 rs=0.2 //stopband ripple
5 wp=%pi/2; //passband frequency
6 ws=3*%pi/4; //stopband frequency
7 T=1;
8 fp=wp/T;
9 fs=ws/T;
10 s=poly(0,'s');
11 z=poly(0,'z');

```

```

12 hs=1;
13 //Calculating the order of filter
14 num=log((rs^-2 -1)/(rp^-2 -1));
15 den=2*log(fs/fp);
16 N=ceil(num/den);
17
18 //Calculation of cut-off frequency
19 fc=fp/(rp^-2 -1)^(0.5/N);
20
21 //Calculating filter response
22 if modulo(N,2)==1 then
23     b=-2*sin(%pi/(2*N));
24     hs=hs*fc/(s+fc);
25 end
26 for k=1:N/2
27     b=2*sin((2*k-1)*%pi/(2*N));
28     hs=hs*fc^2/(s^2+b*fc*s+fc^2);
29 end
30 hs=clean(hs);
31 sys=syslin('c',hs);
32 hz=horner(ss2tf(dscr(sys,T)),1/z); //
    converting H(s) to H(z)
33
34 //Displaying filter response
35 [hzm,fr]=frmag(hz,256);
36 disp(hz,'Filter Transfer function: ');
37 plot(fr,hzm);
38 title('Lowpass Butterworth Filter Response');ylabel(
    'Amplitude—>');xlabel('Normalised frequency f/fs
    —>');

```

Scilab code Exa 8.12 Butterworth Filter using Bilinear transformation

```

1 //Example 8.12
2 clc;clear;close;

```

```

3  rp=0.9                                //passband ripple
4  rs=0.2                                //stopband ripple
5  wp=%pi/2;                             //passband frequency
6  ws=3*%pi/4;                           //stopband frequency
7  T=1;
8  fp=2/T*tan(wp/2);
9  fs=2/T*tan(ws/2);
10 s=poly(0, 's');
11 z=poly(0, 'z');
12 hs=1;
13 //Calculating the order of filter
14 num=log((rs^-2 -1)/(rp^-2 -1));
15 den=2*log(fs/fp);
16 N=ceil(num/den);
17
18 //Calculation of cut-off frequency
19 fc=fp/(rp^-2 -1)^(0.5/N);
20
21 //Calculating filter response
22 if modulo(N,2)==1 then
23     hs=hs*fc/(s+fc);
24 end
25 for k=1:N/2
26     b=2*sin((2*k-1)*%pi/(2*N));
27     hs=hs*fc^2/(s^2+b*fc*s+fc^2);
28 end
29 hs=clean(hs);
30 sys=syslin('c',hs);
31 hz=ss2tf(cls2dls(tf2ss(sys),T));      //converting
    H(s) to H(z)
32
33 //Displaying filter response
34 [hzm,fr]=frmag(hz,256);
35 disp(hz,'Filter Transfer function: ');
36 plot(fr,hzm);
37 title('Lowpass Butterworth Filter Response');ylabel(
    'Amplitude—>');xlabel('Normalised frequency f/fs
    —>');

```

Scilab code Exa 8.14 Filter transformation

```
1 //Example 8.14
2 clc;clear;close;
3 s=poly(0,'s');
4 fc=1; //Assumed cut off frequency
5 Q=10;f0=2; //Given data
6 Hs=1/(s^2+2*s+1);
7 l=fc*(s^2+f0^2)/(s*f0/Q);
8 Hs1=horner(Hs,l);
9 disp(Hs,'Low pass filter H(s)=');
10 disp(Hs1,'Band pass filterH(s)=');
```

Scilab code Exa 8.15 Filter transformation

```
1 //Example 8.15
2 clc;clear;close;
3 s=poly(0,'s');
4 fc=1; //Assumed cut off frequency of
   low pass filter
5 f0=5; //Assumed cut off frequency of
   high pass filter
6 Hs=fc/(s+fc);
7 Hs1=horner(Hs,fc*f0/s);
8 disp(Hs,'H(s)=',fc,'Low pass filter with fc=');
9 disp(Hs1,'H(s)=',f0,'High pass filter with fc=');
```

Chapter 9

Realisation of Digital Linear Systems

Scilab code Exa 9.4 Cascade Realisation

```
1 //Example 9.4
2 clc;clear;close;
3 z=poly(0,'z');
4 Hz=2*(z+2)/(z*(z-0.1)*(z+0.5)*(z+0.4));
5 H=dscr(Hz,0.1);
6 disp(Hz,'System Function H(z)=');
7 disp(H,'System Function for cascade realisation Hk(z
   )=');
```

Scilab code Exa 9.5.a Parallel Realisation

```
1 //Example 9.5.a
2 clc;clear;close;
3 z=poly(0,'z');
4 s=poly(0,'s');
5 Hz=3*(2*z^2+5*z+4)/(2*z+1)/(z+2);
```

```

6 H=pfss(Hz/z);
7 for k=1:length(H)
8     H(k)=clean(H(k));
9     H1(k)=z*horner(H(k),z);
10 disp(H1(k), 'System Function for parallel realisation
    Hk(z)=');
11 end
12 disp(Hz, 'System Function H(z)=');

```

Scilab code Exa 9.5.b Parallel Realisation

```

1 //Example 9.5.b
2 clc;clear;close;
3 z=poly(0, 'z ');
4 s=poly(0, 's ');
5 Hz=3*z*(5*z-2)/(z+1/2)/(3*z-1);
6 H=pfss(Hz/z);
7 for k=1:length(H)
8     H(k)=clean(H(k));
9     H1(k)=z*horner(H(k),z);
10 disp(H1(k), 'System Function for parallel realisation
    Hk(z)=');
11 end
12 disp(Hz, 'System Function H(z)=');

```

Chapter 10

Effects of Finite Word Length in Digital Filters

Scilab code Exa 10.2 Output Quantisation Noise

```
1 //Example 10.2
2
3 clc;clear;close;
4 z=poly(0,'z');
5 H=0.5*z/(z-0.5);
6 B=8;
7 pn=2^(-2*B)/12; //Noise power
8 X=H*horner(H,1/z)/z;
9 r=roots(denom(X));
10 r1=length(r);
11 rc=coeff(denom(X))
12 q1=[];q2=[];
13 for n=1:r1 //Loop to separate poles
    inside the unit circle
14     if (abs(r(n))<1) then
15         q1=[q1 r(n)];
16     else
17         q2=[q2 r(n)];
18     end
```

```

19 end
20 P=numer(X)/rc(length(rc));
21 Q1=poly(q1, 'z');
22 Q2=poly(q2, 'z');
23 I=residu(P,Q1,Q2);           //Residue Calculation
24 po=pn*I;                   //Output Noise power
25 disp(pn, 'Input Noise power');
26 disp(po, 'Output Noise power');

```

Scilab code Exa 10.3 Deadband Interval

```

1 //Example 10.3
2
3 clc; clear;
4 //y(n)=0.9y(n-1)+x(n)
5 //Input x(n)=0
6 n=-1;y=12;           //Initial Condition y(-1)=12
7 flag=1;
8 while n<8
9     n=n+1;
10    y=[y 0.9*y(n+1)];
11    yr=round(y);
12 end
13 disp(n, 'n=');
14 disp(y, 'y(n)-exact');
15 disp(yr, 'y(n)-rounded');
16 disp([-yr(n+2) yr(n+2)], 'Deadband interval ')

```

Scilab code Exa 10.4 Deadband Interval

```

1 //Example 10.4
2
3 clc; clear;

```

```

4 //y(n)=0.9y(n-1)+x(n)
5 a=0.9;
6 l=ceil(0.5/(1-abs(a)));
7 disp([-1 l], 'Deadband interval ')

```

Scilab code Exa 10.5 Output Quantisation Noise for Cascade realisation

```

1 //Example 10.5
2
3 clc;clear;close;
4 x=poly(0,'x'); //x=2^-2B
5 z=poly(0,'z');
6 H1=1/(1-0.9/z);
7 H2=1/(1-0.8/z);
8 H=H1*H2;
9 pn=x/12; //Input Noise power
10
11 //Calculation of output noise for H1(z)
12 X1=H*horner(H,1/z)/z;
13 r1=roots(denom(X1));
14 rc1=coeff(denom(X1));
15 q1=[];s1=[];
16 for n=1:length(r1) //Loop to separate
    poles inside the unit circle
17     if (abs(r1(n))<1) then
18         q1=[q1 r1(n)];
19     else
20         s1=[s1 r1(n)];
21     end
22 end
23 P1=numer(X1)/rc1(length(rc1));
24 Q1=poly(q1,'z');
25 S1=poly(s1,'z');
26 I1=abs(residu(P1,Q1,S1)); //Residue
    Calculation

```

```

27 po1=pn*I1;                               //Output Noise power
28
29 //Calculation of output noise for H2(z)
30 X2=H2*horner(H2,1/z)/z;
31 r2=roots(denom(X2));
32 rc2=coeff(denom(X2));
33 q2=[];s2=[];
34 for n=1:length(r2)                         //Loop to separate
      poles inside the unit circle
35     if (abs(r2(n))<1) then
36         q2=[q2 r2(n)];
37     else
38         s2=[s2 r2(n)];
39     end
40 end
41 P2=numer(X2)/rc2(length(rc2));
42 Q2=poly(q2,'z');
43 S2=poly(s2,'z');
44 I2=abs(residu(P2,Q2,S2));                 //Residue
      Calculation
45 po2=pn*I2;                               //Output Noise
      power
46
47 po=po1+po2;
48 disp(pn,'Input Noise power');
49 disp(I1,'I1=');disp(I2,'I2=');
50 disp(po1,'Output Noise power for H1(z)');
51 disp(po2,'Output Noise power for H2(z)');
52 disp(po,'Total Output Noise power');

```

Chapter 11

Multirate Digital Signal Processing

Scilab code Exa 11.1 Time Decimation

```
1 //Example 11.1
2
3 clc;clear;close;
4 x=[0:6 0:6];
5 y=x(1:3:length(x));
6 disp(x,'Input signal x(n)=');
7 disp(y,'Output signal of decimation process by
   factor three y(n)');
8 subplot(2,1,1);
9 plot2d3(x);title('Input signal x(n)');
10 subplot(2,1,2);
11 plot2d3(y);title('Output signal y(n)');
```

Scilab code Exa 11.2 Interpolation

```
1 //Example 11.2
```

```

2
3 clc; clear; close;
4 x=0:5;
5 y=[];
6 for i=1:length(x)
7     y(1,2*i)=x(i);
8 end
9 disp(x, 'Input signal x(n)=');
10 disp(y, 'Output signal of interpolation process with
    factor two y(n)');
11 subplot(2,1,1);
12 plot2d3(x); title('Input signal x(n)');
13 subplot(2,1,2);
14 plot2d3(y); title('Output signal y(n)');

```

Scilab code Exa 11.4 Polyphase Decomposition

```

1 //Example 11.4
2
3 clc; clear;
4 z=poly(0, 'z');
5 num=1-4*z^-1;
6 den=1+5*z^-1;
7 H=num/den;
8 num1=num*(1-5*z^-1);
9 den1=den*(1-5*z^-1);
10 H1=num1/den1;
11 c=coeff(numer(num1));
12 clength=length(c);
13 c=[c zeros(1, pmodulo(clength,2))]; //make
    length of 'c' multiple of 2
14 c0=[]; c1=[];
15 for n=1:ceil(clength/2) //loop to
    separate even and odd powers of z
16     c0=[c0 c(2*n-1) 0];

```

```

17     c1=[c1 c(2*n) 0];
18 end
19 E0=poly(c0,'z','coeff')/z^n/den1;
20 E1=poly(c1,'z','coeff')/z^(n-2)/den1;
21 disp('Polyphase Components')
22 disp(E0,'E0(z)');
23 disp(E1,'E1(z)');

```

Scilab code Exa 11.5 Decimator implementation

```

1 //Example 11.5
2
3 clc;clear;
4
5 function [N,R]=func(Fs,Fp,Ft,Fti,dp,ds,M)
6     dF=(Fs-Fp)/Ft;
7
8     //
9     Normalised transition bandwidth
10    N=round((-20*log10(sqrt(dp*ds))-13)/(14.6*dF))
11    ; //FIR Filter length
12    R=N*Fti/M;
13
14    //
15    Number of Multiplications per second
16 endfunction
17
18
19 Ft=20000; //Sampling rate of input signal
20 Fp=40; //Passband frequency
21 Fs=50; //Stopband frequency
22 dp=0.01; //Passband ripple
23 ds=0.002; //Stopband ripple
24 M=100; //Decimation Factor
25 Fti=Ft; //Input sampling rate
26 //Single stage implementation
27 [N1,R1]=func(Fs,Fp,Ft,Fti,dp,ds,M);
28

```

```

21 //Two stage implementation
22 //Stage 1 F(z) with decimation factor 50
23 Fpf=Fp;           //Passband frequency
24 Fsf=190;         //Stopband frequency
25 dpf=0.005;      //Passband ripple
26 dsf=0.002;      //Stopband ripple
27 Mf=50;          //Decimation Factor
28 Fti=Ft;         //Input sampling rate
29 [N2f, R2f]=func(Fsf, Fpf, Ft, Fti, dpf, dsf, Mf);
30
31 //Stage 2 G(z) with decimation factor 2
32 Fpg=50*Fp;      //Passband frequency
33 Fsg=50*Fs;      //Stopband frequency
34 dpg=0.005;     //Passband ripple
35 dsg=0.002;     //Stopband ripple
36 Mg=2;          //Decimation Factor
37 Fti=Ft/50;     //Input sampling rate
38 [N2g, R2g]=func(Fsg, Fpg, Ft, Fti, dpg, dsg, Mg);
39 N2=N2f+50*N2g+2; //Total filter length
40 R2=R2f+R2g;     //Total Number of
    Multiplications per second
41 disp(R1, 'Number of Multiplications per second =', N1,
    'FIR filter length =', 'For Single stage
    implementation:');
42 disp('For Two stage implementation:');
43 disp(R2f, 'Number of Multiplications per second =',
    N2f, 'FIR filter length =', 'For F(z):');
44 disp(R2g, 'Number of Multiplications per second =',
    N2g, 'FIR filter length =', 'For G(z):');
45 disp(R2, 'Total Number of Multiplications per second
    =', N2, 'Overall FIR filter length =');

```

Scilab code Exa 11.6 Decimator implementation

```
1 //Example 11.6
```



```

2
3 clc; clear;
4
5
6 function [N,R]=func(Fs ,Fp ,Ft ,Fti ,dp ,ds ,M)
7     dF=(Fs-Fp)/Ft;
8
9     Normalised transition bandwidth //
10    N=round((-20*log10(sqrt(dp*ds))-13)/(14.6*dF))
11    ; //FIR Filter length
12    R=N*Fti/M;
13
14    Number of Multiplications per second //
15 endfunction
16
17 Ft=10000; //Sampling rate of input signal
18 Fp=150; //Passband frequency
19 Fs=180; //Stopband frequency
20 dp=0.002; //Passband ripple
21 ds=0.001; //Stopband ripple
22 M=20; //Decimation Factor
23 Fti=Ft; //Input sampling rate
24 //Single stage implementation
25 [N1 ,R1]=func(Fs ,Fp ,Ft ,Fti ,dp ,ds ,M);
26
27 //Two stage implementation
28 //Stage 1 F(z) with decimation factor 50
29 Fpf=Fp; //Passband frequency
30 Fsf=720; //Stopband frequency
31 dpf=0.001; //Passband ripple
32 dsf=0.001; //Stopband ripple
33 Mf=10; //Decimation Factor
34 Fti=Ft; //Input sampling rate
35 [N2f ,R2f]=func(Fsf ,Fpf ,Ft ,Fti ,dpf ,dsf ,Mf);
36
37 //Stage 2 G(z) with decimation factor 2
38 Fpg=10*Fp; //Passband frequency
39 Fsg=10*Fs; //Stopband frequency

```

```

35 dpg=0.001;           //Passband ripple
36 dsg=0.001;           //Stopband ripple
37 Mg=2;                //Decimation Factor
38 Fti=Ft/10;           //Input sampling rate
39 [N2g,R2g]=func(Fsg,Fpg,Ft,Fti,dpg,dsg,Mg);
40 N2=N2f+10*N2g+2;     //Total filter length
41 R2=R2f+R2g;          //Total Number of
    Multiplications per second
42
43 disp(R1,'Number of Multiplications per second =',N1,
    'FIR filter length =','For Single stage
    implementation:');
44 disp('For Two stage implementation:');
45 disp(R2f,'Number of Multiplications per second =',
    N2f,'FIR filter length =','For F(z):');
46 disp(R2g,'Number of Multiplications per second =',
    N2g,'FIR filter length =','For G(z):');
47 disp(R2,'Total Number of Multiplications per second
    =',N2,'Overall FIR filter length =');

```

Chapter 12

Spectral Estimation

Scilab code Exa 12.2 Power Spectrum

```
1 //Example 12.2
2 clc;clear;close;
3 N=8;n=0:N-1;
4 f1=0.6;f2=0.62;
5 x=cos(2*%pi*f1*n)+cos(2*%pi*f2*n);
6 L1=8;
7 for k=0:L1-1
8     P1(k+1)=1/N*abs(x*(cos(%pi*n*k/L1)-%i*sin(%pi*n*
9         k/L1)))^2
10 end
11 L2=16;
12 for k=0:L2-1
13     P2(k+1)=1/N*abs(x*(cos(%pi*n*k/L2)-%i*sin(%pi*n*
14         k/L2)))^2;
15 end
16 L3=32;
17 for k=0:L3-1
18     P3(k+1)=1/N*abs(x*(cos(%pi*n*k/L3)-%i*sin(%pi*n*
19         k/L3)))^2;
20 end
21 subplot(311);
```

```
19 plot2d3(0:L1-1,P1);title('L=8');
20 subplot(312);
21 plot2d3(0:L2-1,P2);title('L=16');
22 subplot(313);
23 plot2d3(0:L3-1,P3);title('L=32');
```

Scilab code Exa 12.4 Frequency resolution

```
1 //Example 12.4
2 clc;clear;close;
3 N=1000;
4 Q=10;
5 disp(N,'Length of sample sequence N=',Q,'Quality
   factor Q=');
6 f_bart=Q/(1.11*N);
7 f_w=Q/(1.39*N);
8 f_bt=Q/(2.34*N);
9 disp(f_bart,'Bartlett Frequency resolution =');
10 disp(f_w,'Welch Frequency resolution =');
11 disp(f_bt,'Blackman Turkey Frequency resolution =');
```
