Scilab Manual for Digital Signal Processing by Dr R.senthilkumar Electronics and Telecommunication Engineering Government College Of Engineering Erode[1]

Solutions provided by
Dr R.senthilkumar
Electronics and Telecommunication Engineering
Government College Of Engineering Erode

May 10, 2025

[1]Funded by a grant from the National Mission on Education through ICT, http://spoken-tutorial.org/NMEICT-Intro. This Scilab Manual and Scilab codes written in it can be downloaded from the "Migrated Labs" section at the website http://scilab.in

Contents

List of Scilab Solutions 3

1 Generation of elementary Discrete-Time sequences 5

2 Linear and Circular convolutions 13

3 Auto correlation and Cross Correlation 18

4 Frequency Analysis using DFT 22

5 Design of FIR filters (LPF/HPF/BPF/BSF) and demonstrates the filtering operation 25

6 Design of Butterworth and Chebyshev IIR filters (LPF/HPF/BPF/BSF) and demonstrate the filtering operations 34

List of Experiments

Solution 1.1	Unit Sample Sequence	5
Solution 1.2	Unit Step Sequence	7
Solution 1.3	Discrete Ramp Sequence	8
Solution 1.4	Exponent Increasing Signal	9
Solution 1.5	Exponential Decreasing Signal	12
Solution 2.1	Linear Convolution	13
Solution 2.2	Circular Convolution	15
Solution 3.1	Auto Correlation	18
Solution 3.2	Cross Correlation	19
Solution 4.0	Spectrum Using DFT	22
Solution 5.1	FIR LPF	25
Solution 5.2	FIR HPF	28
Solution 5.3	FIR BPF	29
Solution 5.4	FIR BSF	31
Solution 6.1	Butterworth IIR LPF	34
Solution 6.2	Butterworth IIR HPF	35
Solution 6.3	Butterworth IIR BPF	36
Solution 6.4	Butterworth IIR BSF	37
Solution 6.5	Chebyshev IIR LPF	38
Solution 6.6	Chebyshev IIR HPF	39
Solution 6.7	Chebyshev IIR BPF	40
Solution 6.8	Chebychev IIR BSF	41

List of Figures

1.1 Unit Sample Sequence 6
1.2 Unit Step Sequence 7
1.3 Discrete Ramp Sequence 8
1.4 Exponent Increasing Signal 10
1.5 Exponential Decreasing Signal 11

2.1 Linear Convolution 14

4.1 Spectrum Using DFT 23

5.1 FIR LPF 26
5.2 FIR HPF 27
5.3 FIR BPF 29
5.4 FIR BSF 32

6.1 Butterworth IIR LPF 35
6.2 Butterworth IIR HPF 36
6.3 Butterworth IIR BPF 37
6.4 Butterworth IIR BSF 38
6.5 Chebyshev IIR LPF 39
6.6 Chebyshev IIR HPF 40
6.7 Chebyshev IIR BPF 41
6.8 Chebychev IIR BSF 42

Experiment: 1

Generation of elementary Discrete-Time sequences

Scilab code Solution 1.1 Unit Sample Sequence

```
1  //1.GENERATION OF ELEMENTARY DISCRETE TIME SEQUENCE
2  //1.1Caption:Unit Sample Sequence
3  clear;
4  clc;
5  close;
6  L = 4;  //Upperlimit
7  n = -L:L;
8  x = [zeros(1,L),1,zeros(1,L)];
9  b = gca();
10 b.y_location = "middle";
11 plot2d3('gnn',n,x)
12 a=gce();
13 a.children(1).thickness =4;
14 xtitle('Graphical Representation of Unit Sample
     Sequence','n','x[n]');
```

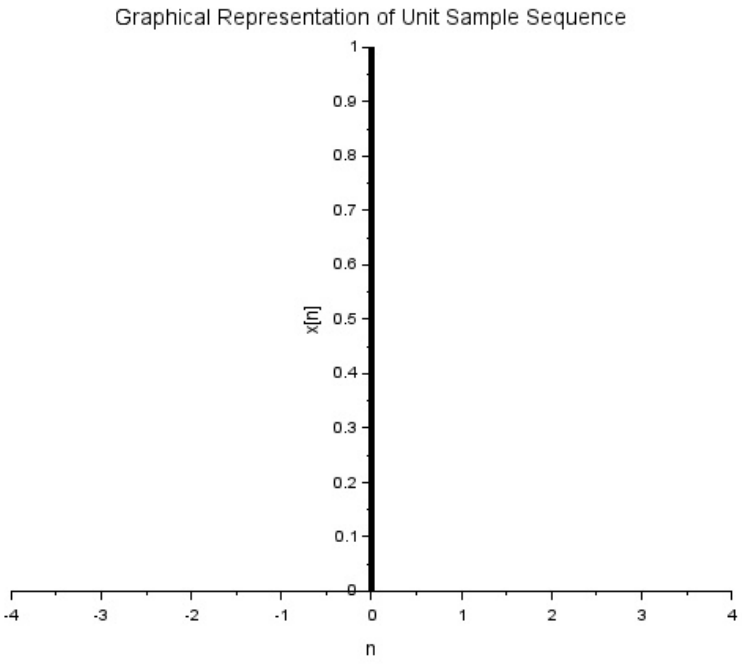

Figure 1.1: Unit Sample Sequence

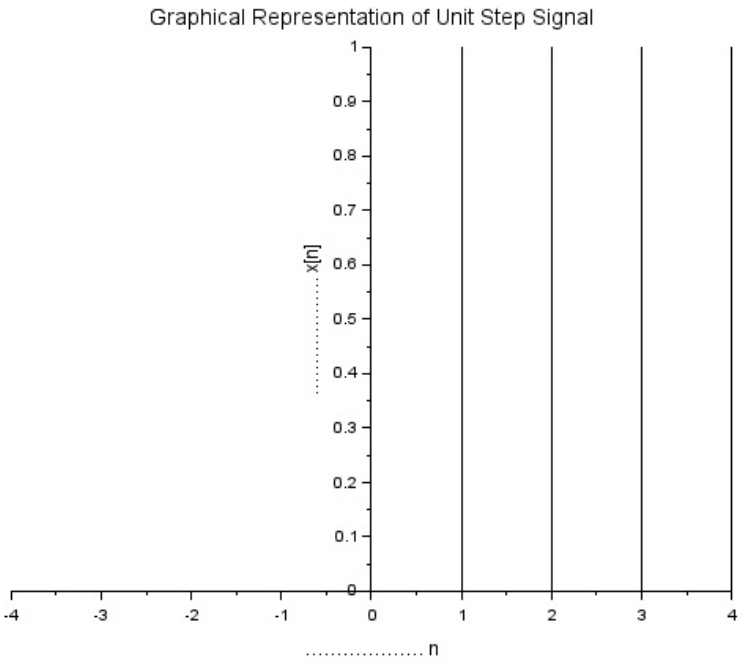

Figure 1.2: Unit Step Sequence

Scilab code Solution 1.2 Unit Step Sequence

```
1 //1.GENERATION OF ELEMENTARY DISCRETE TIME SEQUENCE
2 //1.2 Caption: Unit Step Sequence
3 clear;
4 clc;
5 close;
6 L = 4;  //Upperlimit
7 n = -L:L;
8 x = [zeros(1,L),ones(1,L+1)];
```

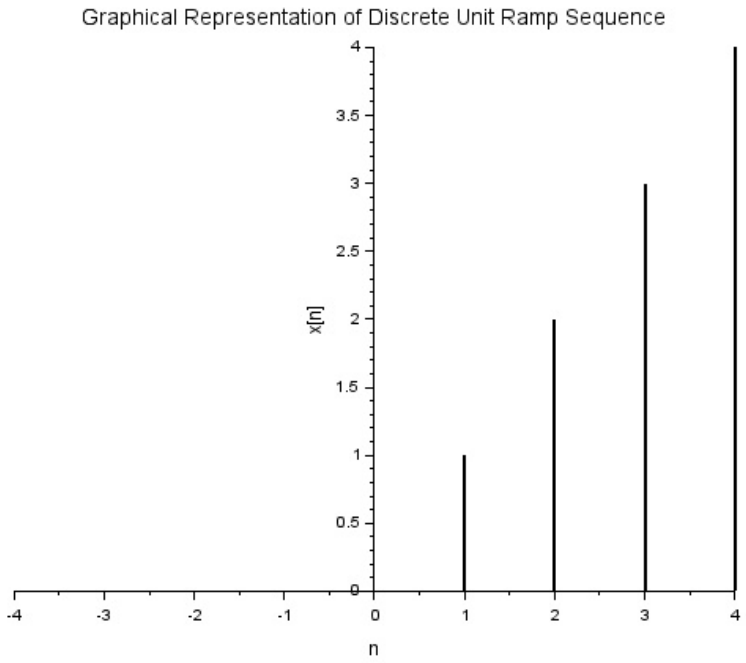

Figure 1.3: Discrete Ramp Sequence

```
9  a=gca();
10 a.y_location = "middle";
11 plot2d3('gnn',n,x)
12 title('Graphical Representation of Unit Step Signal'
     )
13 xlabel('                    . n');
14 ylabel('                    . x[n]');
```

Scilab code Solution 1.3 Discrete Ramp Sequence

```
1 //1.GENERATION OF ELEMENTARY DISCRETE TIME SEQUENCE
```

```
2 //1.3 Caption: Discrete Ramp Sequence
3 clear;
4 clc;
5 close;
6 L = 4;  //Upperlimit
7 n = -L:L;
8 x = [zeros(1,L),0:L];
9 b = gca();
10 b.y_location = 'middle';
11 plot2d3('gnn',n,x)
12 a=gce();
13 a.children(1).thickness =2;
14 xtitle('Graphical Representation of Discrete Unit
   Ramp Sequence','n','x[n]');
```

Scilab code Solution 1.4 Exponent Increasing Signal

```
1 //1.GENERATION OF ELEMENTARY DISCRETE TIME SEQUENCE
2 //1.4 Caption:Exponentially Increasing Signal
3 clear;
4 clc;
5 close;
6 a =1.5;
7 n =1:10;
8 x = (a)^n;
9 a=gca();
10 a.thickness = 2;
11 plot2d3('gnn',n,x)
12 xtitle('Graphical Representation of Exponentially
   Increasing Signal','n','x[n]');
```

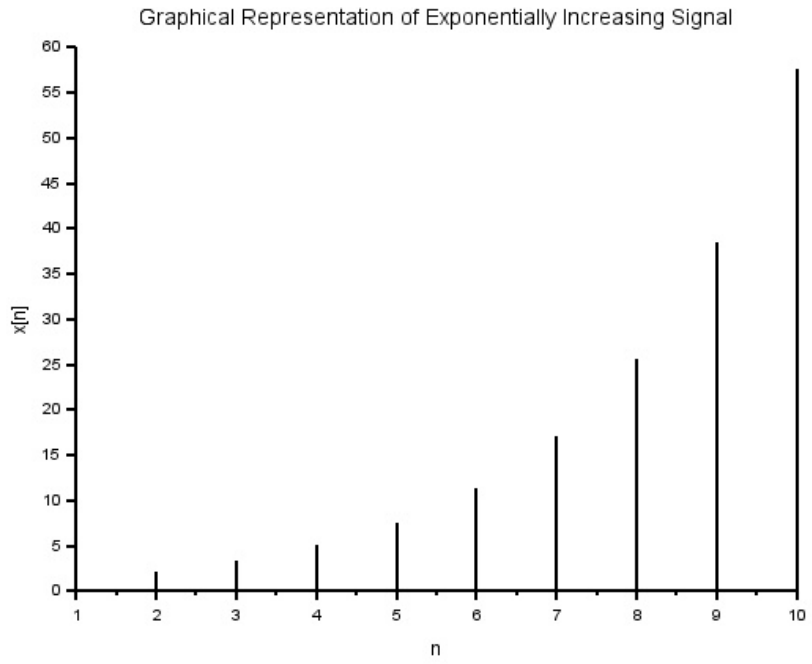

Figure 1.4: Exponent Increasing Signal

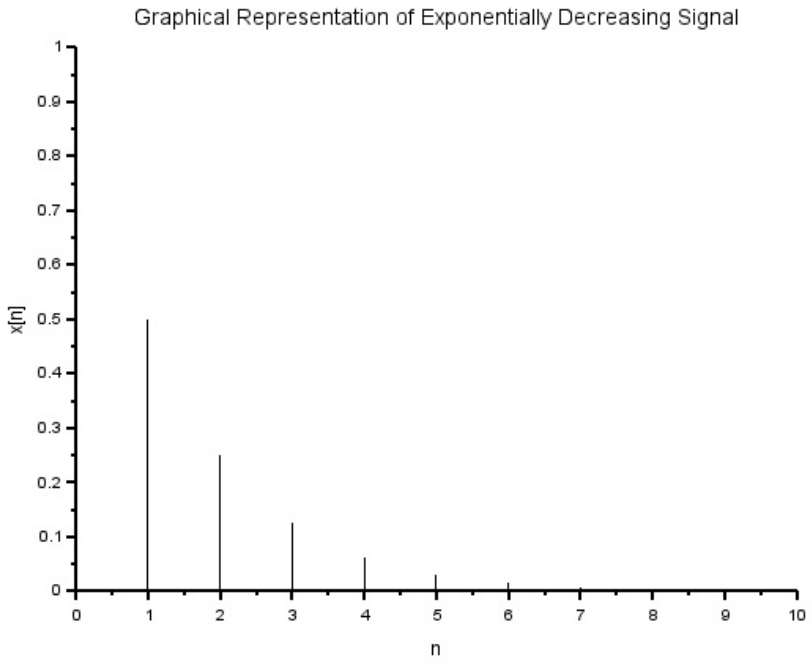

Figure 1.5: Exponential Decreasing Signal

Scilab code Solution 1.5 Exponential Decreasing Signal

```
1  //1.GENERATION OF ELEMENTARY DISCRETE TIME SEQUENCE
2  //Caption: Exponentially Decreasing Signal
3  clear;
4  clc;
5  close;
6  a =0.5;
7  n = 0:10;
8  x = (a)^n;
9  a=gca();
10 a.x_location = "origin";
11 a.y_location = "origin";
12 plot2d3('gnn',n,x)
13 a.thickness = 2;
14 xtitle('Graphical Representation of Exponentially
      Decreasing Signal','n','x[n]');
```

Experiment: 2

Linear and Circular convolutions

Scilab code Solution 2.1 Linear Convolution

```
1  //Caption:Program for Linear Convolution
2  clc;
3  clear all;
4  close ;
5  x = input('enter x seq');
6  h = input('enter h seq');
7  m = length(x);
8  n = length(h);
9  //Method 1 Using Direct Convolution Sum Formula
10 for i = 1:n+m-1
11     conv_sum = 0;
12     for j = 1:i
13         if (((i-j+1) <= n)&(j <= m))
14             conv_sum = conv_sum + x(j)*h(i-j+1);
15         end;
16         y(i) = conv_sum;
17     end;
```

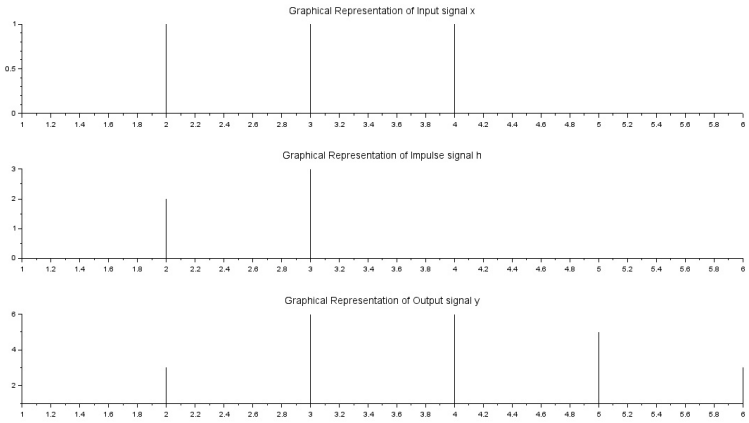

Figure 2.1: Linear Convolution

```
18 end;
19 disp(y','Convolution Sum using Direct Formula Method
       =')
20 //Method 2 Using Inbuilt Function
21 f = convol(x,h)
22 disp(f,'Convolution Sum Result using Inbuilt Funtion
       =')
23 //Method 3 Using frequency Domain multiplication
24 N = n+m-1;
25 x = [x zeros(1,N-m)];
26 h = [h zeros(1,N-n)];
27 f1 = fft(x)
28 f2 = fft(h)
29 f3 = f1.*f2;    // freq domain multiplication
30 f4 = ifft(f3)
31 disp(f4,'Convolution Sum Result DFT - IDFT method ='
      )
32 //f4 =  real(f4)
33 subplot(3,1,1);
34 plot2d3('gnn',x)
35 xtitle('Graphical Representation of Input signal x')
      ;
36 subplot(3,1,2);
```

```
37 plot2d3('gnn',h)
38 xtitle('Graphical Representation of Impulse signal h
   ');
39 subplot(3,1,3);
40 plot2d3('gnn',y)
41 xtitle('Graphical Representation of Output signal y'
   );
42
43
44 //OUTPUT Test case
45
46 //enter x seq [1,1,1,1]
47 //enter h seq [1,2,3]
48
49 //Result
50 //  1.   3.   6.   6.   5.   3.
51 //  "Convolution Sum using Direct Formula Method ="
52 //   1.   3.   6.   6.   5.   3.
53 //  "Convolution Sum Result using Inbuilt Funtion ="
54 //   1.   3.   6.   6.   5.   3.
55 //  "Convolution Sum Result DFT - IDFT method ="
```

Scilab code Solution 2.2 Circular Convolution

```
1 //Caption: Program to find the Cicrcular Convolution
     of given discrete sequences
2 clear all;
3 clc;
4 x1 = input('Enter the first sequence x1[n]=')
5 x2 = input('Enter the second sequence x2[n]=')
6 //x1 = [1,3,5,7];
7 //x2 = [2,4,6,8];
8
9 //Method 1: Circular Convolution using built-in fft
   function
```

```
X1 = fft(x1,-1); //fft of x1
X2 = fft(x2,-1); //fft of x2
X3 = X1.*X2; //X1(k)*X2(K)
x3 = fft(X3,1); //ifft of X3(K)
disp('Circular Convolution using built-in fft
   function:')
disp(x3)



//Method 2: Circular Convolution of two sequences
   using direct formula
m = length(x1)
n = length(x2)
a = zeros(1,max(m,n))
if (m >n)
   for i = n+1:m
     x2(i) = 0;
   end
elseif (n>m)
  for i = m+1:n
    x1(i) = 0;
  end
end
N = length(x1)
x3 = zeros(1,N);
a(1) = x2(1);
for j = 2:N
  a(j) = x2(N-j+2);
end
for i =1:N
  x3(1) = x3(1)+x1(i)*a(i);
end
for k = 2:N
 for j =2:N
   x2(j) = a(j-1);
 end
   x2(1) = a(N);
```

```
46      x2
47    for i = 1:N
48        a(i) = x2(i);
49        x3(k) = x3(k)+x1(i)*a(i);
50    end
51  end
52  disp('Circular Convolution using Direct formula
      method:')
53  disp('Circular Convolution Result x3 = ')
54  disp(x3)
55
56  //OUTPUT Test case
57  //Enter the first sequence x1[n]=[1,2,3,4]
58  //
59  //Enter the second sequence x2[n]=[1,1,1,1]
60  //
61  //
62  //   "Circular Convolution using built-in fft
      function:"
63  //
64  //    10.    10.    10.    10.
65  //
66  //   "Circular Convolution using Direct formula
      method:"
67  //
68  //   "Circular Convolution Result x3 = "
69  //
70  //    10.    10.    10.    10.
```

Experiment: 3

Auto correlation and Cross Correlation

Scilab code Solution 3.1 Auto Correlation

```
1 //Caption: Progrem to find the Autocorrelation of a
     given Input Sequence
2 clear all;
3 clc;
4 x = input('Enter the given discrete time sequence');
5 L = length(x);
6 h = zeros(1,L);
7 for i = 1:L
8   h(L-i+1) = x(i);
9 end
10 N = 2*L-1;
11 Rxx = zeros(1,N);
12 for i = L+1:N
13    h(i) = 0;
14 end
15 for i = L+1:N
16     x(i) = 0;
17 end
18
```

```
for n = 1:N
  for k = 1:N
    if(n >= k)
      Rxx(n) = Rxx(n)+x(n-k+1)*h(k);
    end
  end
end
disp('Auto Correlation Result is')
disp(Rxx)


//OUTPUT Test case
//Enter the given discrete time sequence [1,2,3,4]
//Result
//"Auto Correlation Result is"
//4.   11.   20.   30.   20.   11.   4.
```

Scilab code Solution 3.2 Cross Correlation

```
//Caption: Program to find the Cross-correlation of
   two Sequences
clear all;
clc;
x = [1,2,1,1];
L = length(x);
h1 = [1,1,2,1];
disp('The input sequence 1 =')
disp(x)
disp('The input sequence 2=')
disp(h1)

//Cross Correlation using Built-in function xcorr()

Y  = xcorr(x,h1)
disp('Cross Correlation using built-in function
```

```
    xcorr():')
disp(Y)

//Cross Correlation using Direct Formula Method
for i = 1:L
  h(L-i+1) = h1(i);
end
N = 2*L-1;
Rxy = zeros(1,N);
for i = L+1:N
   h(i) = 0;
end
for i = L+1:N
    x(i) = 0;
end

for n = 1:N
  for k = 1:N
    if(n >= k)
      Rxy(n) = Rxy(n)+x(n-k+1)*h(k);
    end
  end
end

disp('Cross Correlation Result using direct formula
    method is')
disp(Rxy)


//OUTPUT Test Case
//Result
//"The input sequence 1 ="
//
//    1.    2.    1.    1.
//
//   "The input sequence 2="
//
//    1.    1.    2.    1.
```

```
52 //
53 //  "Cross Correlation using built-in function xcorr
       ():"
54 //
55 //    1.    4.    6.    6.    5.    2.    1.
56 //
57 //  "Cross Correlation Result using direct formula
       method is"
58 //
59 //    1.    4.    6.    6.    5.    2.    1.
```

Experiment: 4

Frequency Analysis using DFT

Scilab code Solution 4.0 Spectrum Using DFT

```
1  //Caption:Discrete Periodic Spectrum Plot of N-point
       Sequece
2  clear all;
3  clc;
4  x=[0,1,2,3];
5  //Computing DFT and IDFT
6  X=fft(x,-1);
7  disp('The DFT of given sequence x[n] is X(k)=')
8  disp(X)
9  Phase=atan(imag(X),real(X));
10 figure
11 subplot(2,1,1)
12 a=gca();
13 a.data_bounds=[0,0;5,6];
14 a.x_location='origin'
15 a.y_location='origin'
16 plot2d3('gnn',[0:length(x)-1],abs(X))
17 poly1=a.children(1).children(1);
18 poly1.thickness=2;
```

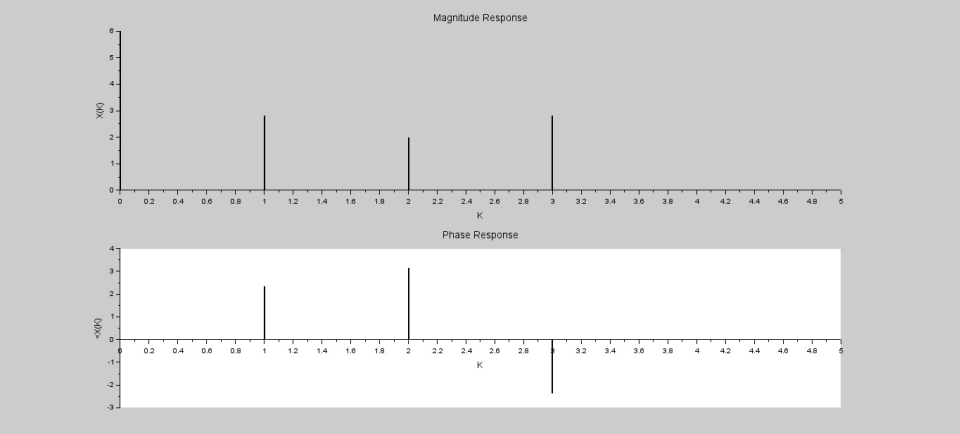

Figure 4.1: Spectrum Using DFT

```
19 xtitle('Magnitude Response','K','X(K)');
20 subplot(2,1,2)
21 a=gca();
22 a.data_bounds=[0,0;5,2];
23 a.x_location='origin'
24 a.y_location='origin'
25 plot2d3('gnn',[0:length(x)-1],Phase)
26 poly1=a.children(1).children(1);
27 poly1.thickness=2;
28 xtitle('Phase Response','K','<X(K)');
29 disp('The Magnitude Response is |X(k)|=')
30 disp(abs(X))
31 disp('The Phase Response is <X(K)=')
32 disp(Phase)
33
34
35 //OUTPUT Test case
36 //Result
37 //"The DFT of given sequence x[n] is X(k)="
38 //
39 //    6. + 0.i   -2. + 2.i   -2. + 0.i   -2. - 2.i
40 //
41 //   "The Magnitude Response is |X(k)|="
```

```
42 //
43 //    6.    2.8284271    2.    2.8284271
44 //
45 //   "The Phase Response is <X(K)="
46 //
47 //    0.    2.3561945    3.1415927   -2.3561945
```

Experiment: 5

Design of FIR filters (LPF/HPF/BPF/BSF) and demonstrates the filtering operation

Scilab code Solution 5.1 FIR LPF

```
1  //Caption: Program to Design FIR Low Pass Filter
2  clc;
3  close;
4  M = input('Enter the Odd Filter Length =');
                  //Filter length
5  Wc = input('Enter the Digital Cutoff frequency =');
     //Digital Cutoff frequency
6  Tuo = (M-1)/2      //Center Value
7  for n = 1:M
8      if (n == Tuo+1)
9        hd(n) = Wc/%pi;
10     else
11       hd(n) =  sin(Wc*((n-1)-Tuo))/(((n-1)-Tuo)*%pi)
```

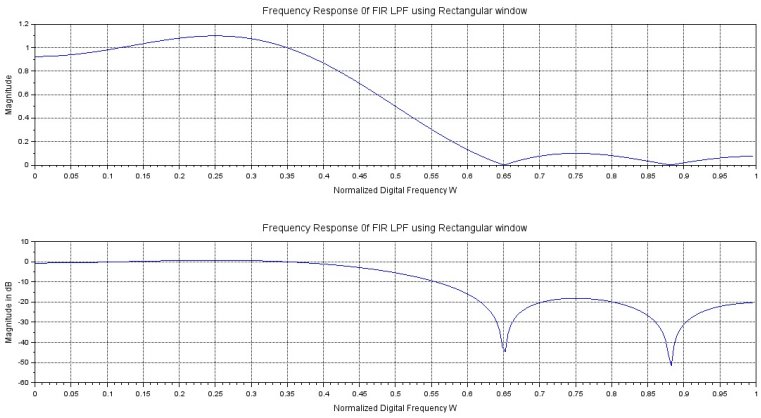

Figure 5.1: FIR LPF

```
            ;
12      end
13  end
14  //Rectangular Window
15  for n = 1:M
16    W(n) = 1;
17  end
18  //Windowing Fitler Coefficients
19  h = hd.*W;
20  disp('Filter Coefficients are')
21  disp(h)
22
23  [hzm,fr]=frmag(h,256);
24  hzm_dB = 20*log10(hzm)./max(hzm);
25  subplot(2,1,1)
26  plot(2*fr,hzm)
27  xlabel('Normalized Digital Frequency W');
28  ylabel('Magnitude');
29  title('Frequency Response 0f FIR LPF using
       Rectangular window')
30  xgrid(1)
31  subplot(2,1,2)
32  plot(2*fr,hzm_dB)
```

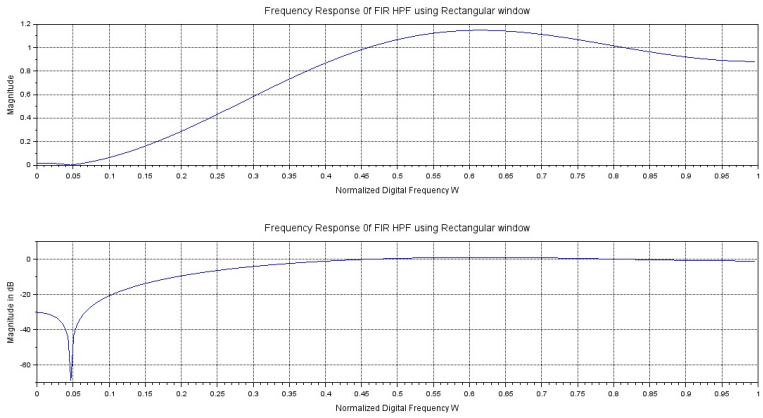

Figure 5.2: FIR HPF

```
33 xlabel('Normalized Digital Frequency W');
34 ylabel('Magnitude in dB');
35 title('Frequency Response 0f FIR LPF using
      Rectangular window')
36 xgrid(1)
37
38 //OUTPUT Test case
39 //Enter the Odd Filter Length = 7
40 //Enter the Digital Cutoff frequency = %pi/2
41 //Result
42 // "Filter Coefficients are"
43 //   -0.1061033
44 //    1.949D-17
45 //    0.3183099
46 //    0.5
47 //    0.3183099
48 //    1.949D-17
49 //   -0.1061033
```

Scilab code Solution 5.2 FIR HPF

```
1  //Caption: Program to Design FIR High Pass Filter
2  clear;
3  clc;
4  close;
5  M = input('Enter the Odd Filter Length =');
                  //Filter length
6  Wc = input('Enter the Digital Cutoff frequency =');
     //Digital Cutoff frequency
7  Tuo = (M-1)/2      //Center Value
8  for n = 1:M
9      if (n == Tuo+1)
10       hd(n) = 1-Wc/%pi;
11     else
12       hd(n) = (sin(%pi*((n-1)-Tuo)) -sin(Wc*((n-1)-
            Tuo)))/(((n-1)-Tuo)*%pi);
13     end
14 end
15 //Rectangular Window
16 for n = 1:M
17   W(n) = 1;
18 end
19 //Windowing Fitler Coefficients
20 h = hd.*W;
21 disp('Filter Coefficients are')
22 disp(h)
23 [hzm,fr]=frmag(h,256);
24 hzm_dB = 20*log10(hzm)./max(hzm);
25 subplot(2,1,1)
26 plot(2*fr,hzm)
27 xlabel('Normalized Digital Frequency W');
28 ylabel('Magnitude');
29 title('Frequency Response 0f FIR HPF using
     Rectangular window')
30 xgrid(1)
31 subplot(2,1,2)
32 plot(2*fr,hzm_dB)
```

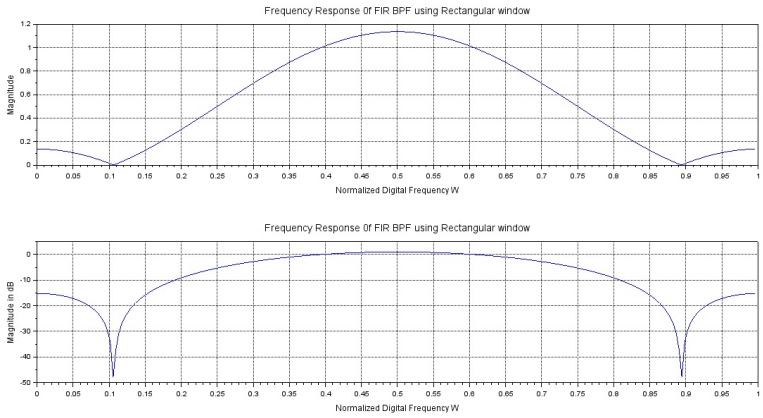

Figure 5.3: FIR BPF

```
33 xlabel('Normalized Digital Frequency W');
34 ylabel('Magnitude in dB');
35 title('Frequency Response 0f FIR HPF using
      Rectangular window')
36 xgrid(1)
37
38 //OUTPUT Test case
39 //Enter the Odd Filter Length = 5
40 //Enter the Digital Cutoff frequency = %pi/4
41 //"Filter Coefficients are"
42 //   -0.1591549
43 //   -0.2250791
44 //    0.75
45 //   -0.2250791
46 //   -0.1591549
```

Scilab code Solution 5.3 FIR BPF

```
1 //Caption: Program to Design FIR Band Pass Filter
```

```
clear;
clc;
close;
M = input('Enter the Odd Filter Length =');
                    //Filter length
//Digital Cutoff frequency [Lower Cutoff, Upper
   Cutoff]
Wc = input('Enter the Digital Cutoff frequency =');
Wc2 = Wc(2)
Wc1 = Wc(1)
Tuo = (M-1)/2      //Center Value
hd = zeros(1,M);
W = zeros(1,M);
for n = 1:11
  if (n == Tuo+1)
     hd(n) = (Wc2-Wc1)/%pi;
  else
        n
     hd(n) = (sin(Wc2*((n-1)-Tuo)) -sin(Wc1*((n-1)-
        Tuo)))/(((n-1)-Tuo)*%pi);
  end
  if(abs(hd(n))<(0.00001))
    hd(n)=0;
  end
end
hd;
//Rectangular Window
for n = 1:M
  W(n) = 1;
end
//Windowing Fitler Coefficients
h = hd.*W;
disp('Filter Coefficients are')
disp(h)
[hzm,fr]=frmag(h,256);
hzm_dB = 20*log10(hzm)./max(hzm);
subplot(2,1,1)
plot(2*fr,hzm)
```

```
37 xlabel('Normalized Digital Frequency W');
38 ylabel('Magnitude');
39 title('Frequency Response 0f FIR BPF using
      Rectangular window')
40 xgrid(1)
41 subplot(2,1,2)
42 plot(2*fr,hzm_dB)
43 xlabel('Normalized Digital Frequency W');
44 ylabel('Magnitude in dB');
45 title('Frequency Response 0f FIR BPF using
      Rectangular window')
46 xgrid(1)
47
48
49 //OUTPUT Test case
50 //Enter the Odd Filter Length = 11
51 //Enter the Digital Cutoff frequency = [%pi/4,3*%pi
      /4]
52 //Result
53 //  "Filter Coefficients are"
54 // 0.    0.    0.   -0.3183099    0.    0.5    0.
      -0.3183099    0.    0.    0.
```

Scilab code Solution 5.4 FIR BSF

```
1 //Caption: Program to Design FIR Band Reject Filter
2 clear ;
3 clc;
4 close;
5 M = input('Enter the Odd Filter Length =');
                  //Filter length
6 //Digital Cutoff frequency [Lower Cutoff, Upper
      Cutoff]
```

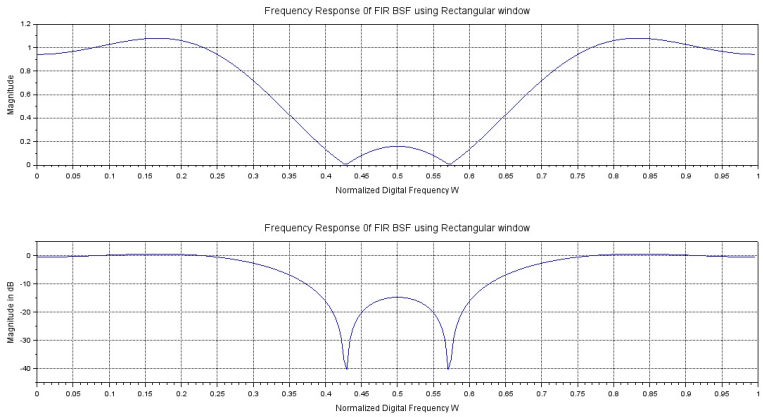

Figure 5.4: FIR BSF

```
Wc = input('Enter the Digital Cutoff frequency =');
Wc2 = Wc(2)
Wc1 = Wc(1)
Tuo = (M-1)/2      //Center Value
hd = zeros(1,M);
W = zeros(1,M);
for n = 1:M
 if (n == Tuo+1)
 hd(n) = 1-((Wc2-Wc1)/%pi);
 else
 hd(n)=(sin(%pi*((n-1)-Tuo))-sin(Wc2*((n-1)-Tuo))+
    sin(Wc1*((n-1)-Tuo)))/(((n-1)-Tuo)*%pi);
 end
  if(abs(hd(n))<(0.00001))
    hd(n)=0;
  end
end

//Rectangular Window
for n = 1:M
  W(n) = 1;
end
//Windowing Fitler Coefficients
```

```
29 h = hd.*W;
30 disp('Filter Coefficients are')
31 disp(h)
32 [hzm,fr]=frmag(h,256);
33 hzm_dB = 20*log10(hzm)./max(hzm);
34 subplot(2,1,1)
35 plot(2*fr,hzm)
36 xlabel('Normalized Digital Frequency W');
37 ylabel('Magnitude');
38 title('Frequency Response 0f FIR BSF using
      Rectangular window')
39 xgrid(1)
40 subplot(2,1,2)
41 plot(2*fr,hzm_dB)
42 xlabel('Normalized Digital Frequency W');
43 ylabel('Magnitude in dB');
44 title('Frequency Response 0f FIR BSF using
      Rectangular window')
45 xgrid(1)
46
47 //OUTPUT Test case
48 //Enter the Odd Filter Length = 11
49 //Enter the Digital Cutoff frequency = [%pi/3,2*%pi
      /3]
50 //Result
51 //  "Filter Coefficients are"
52 // 0.  -0.1378322   0.   0.2756644   0.   0.6666667
       0.   0.2756644   0.  -0.1378322   0.
```

Experiment: 6

Design of Butterworth and Chebyshev IIR filters (LPF/HPF/BPF/BSF) and demonstrate the filtering operations

Scilab code Solution 6.1 Butterworth IIR LPF

```
1  //IIR Butterworth LPF Filter
2  clear;
3  clc;
4  n=3;
5  ftype='lp';
6  fdesign = 'butt';
7  frq=[0.15,0.25];
8  delta=[0.08,0.02]
9  hz=iir(n,ftype,fdesign,frq);
10 [hzm,fr]=frmag(hz,256);
11 plot2d(fr',hzm')
```

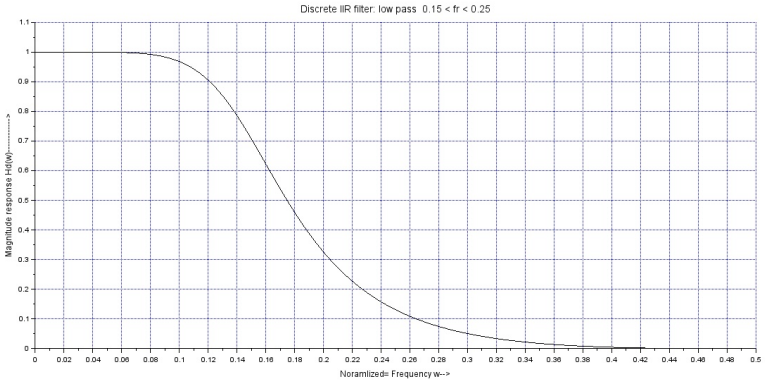

Figure 6.1: Butterworth IIR LPF

```
12 xtitle('Discrete IIR filter: low pass  0.15 < fr <
     0.25 ',' ',' ');
13 xlabel('Noramlized= Frequency w-->')
14 ylabel('Magnitude response Hd(w)------------------>')
15 xgrid(2)
```

Scilab code Solution 6.2 Butterworth IIR HPF

```
1 //IIR Butterworth HPF Filter
2
3 clear;
4 clc;
5 n=3;
6 ftype='hp';
7 fdesign = 'butt';
8 frq=[0.15,0.25];
9 delta=[0.08,0.02]
10
11 hz=iir(n,ftype,fdesign,frq);
```

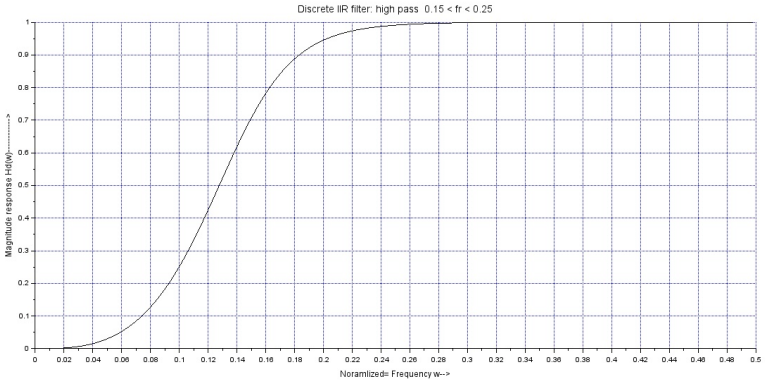

Figure 6.2: Butterworth IIR HPF

```
12 [hzm,fr]=frmag(hz,256);
13 plot2d(fr',hzm')
14 xtitle('Discrete IIR filter: high pass  0.15 < fr <
     0.25 ',' ',' ');
15 xlabel('Noramlized= Frequency w-->')
16 ylabel('Magnitude response Hd(w)------------------>')
17 xgrid(2)
```

Scilab code Solution 6.3 Butterworth IIR BPF

```
1 //IIR Butterworth BPF Filter
2 clear all;
3 clc;
4 n=3;
5 ftype='bp';
6 fdesign = 'butt';
7 frq=[0.15,0.25];
8 delta=[0.08,0.02]
9 hz=iir(n,ftype,fdesign,frq);
```

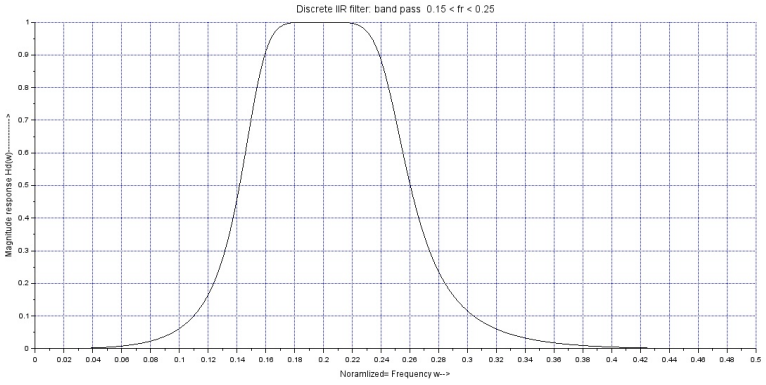

Figure 6.3: Butterworth IIR BPF

```
10 [hzm,fr]=frmag(hz,256);
11 plot2d(fr',hzm')
12 xtitle('Discrete IIR filter: band pass  0.15 < fr <
     0.25 ',' ',' ');
13 xlabel('Noramlized= Frequency w-->')
14 ylabel('Magnitude response Hd(w)---------------->')
15 xgrid(2)
```

Scilab code Solution 6.4 Butterworth IIR BSF

```
1 //IIR Butterworth BSF Filter
2 clear;
3 clc;
4 n=3;
5 ftype='sb';
6 fdesign = 'butt';
7 frq=[0.15,0.25];
8 delta=[0.08,0.02]
9 hz=iir(n,ftype,fdesign,frq);
```

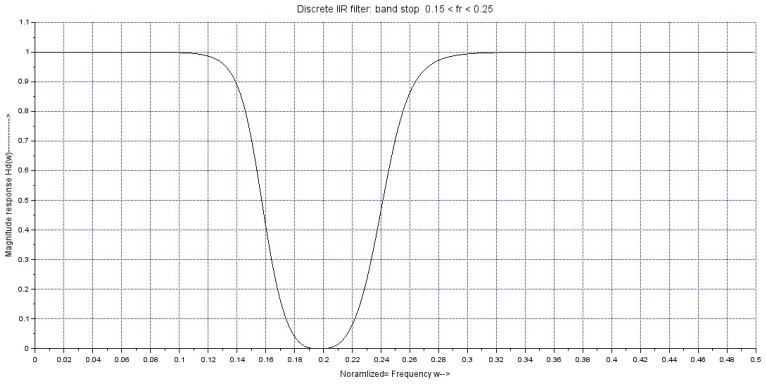

Figure 6.4: Butterworth IIR BSF

```
10 [hzm,fr]=frmag(hz,256);
11 plot2d(fr',hzm')
12 xtitle('Discrete IIR filter: band stop  0.15 < fr <
     0.25 ',' ',' ');
13 xlabel('Noramlized= Frequency w-->')
14 ylabel('Magnitude response Hd(w)------------------>')
15 xgrid(2)
```

Scilab code Solution 6.5 Chebyshev IIR LPF

```
1 //IIR Chebyshev LPF Filter
2 clear;
3 clc;
4 n=3;
5 ftype='lp';
6 fdesign = 'cheb1';
7 frq=[0.15,0.25];
8 delta=[0.08,0.02]
9 hz=iir(n,ftype,fdesign,frq);
```

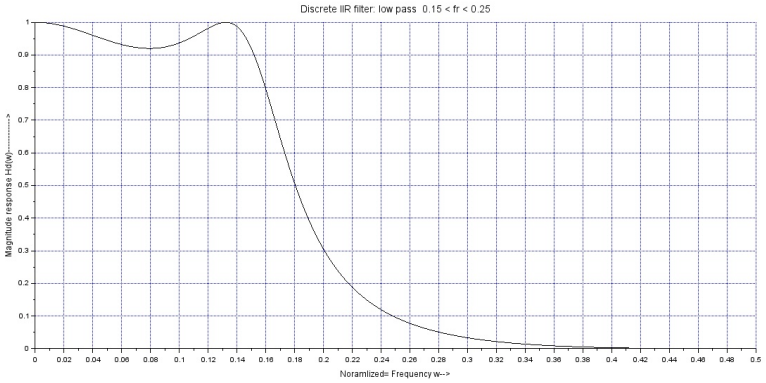

Figure 6.5: Chebyshev IIR LPF

```
10 [hzm,fr]=frmag(hz,256);
11 plot2d(fr',hzm')
12 xtitle('Discrete IIR filter: low pass  0.15 < fr <
     0.25 ',' ',' ');
13 xlabel('Noramlized= Frequency w-->')
14 ylabel('Magnitude response Hd(w)------------->')
15 xgrid(2)
```

Scilab code Solution 6.6 Chebyshev IIR HPF

```
1 //IIR Chebyshev HPF Filter
2
3 clear;
4 clc;
5 n=3;
6 ftype='hp';
7 fdesign = 'cheb1';
8 frq=[0.15,0.25];
9 delta=[0.08,0.02]
```

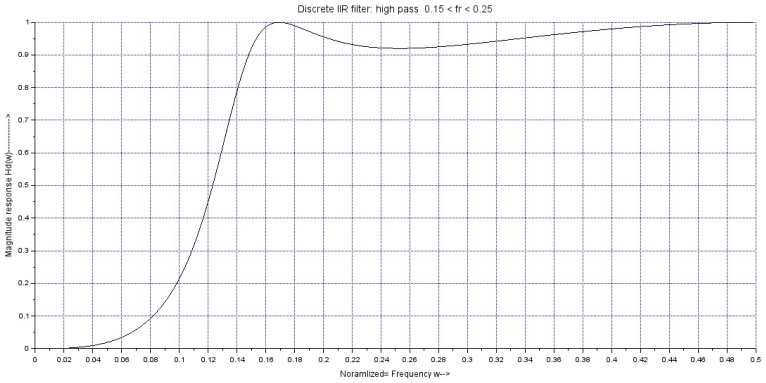

Figure 6.6: Chebyshev IIR HPF

```
10
11 hz=iir(n,ftype,fdesign,frq);
12 [hzm,fr]=frmag(hz,256);
13 plot2d(fr',hzm')
14 xtitle('Discrete IIR filter: high pass  0.15 < fr <
     0.25 ',' ',' ');
15 xlabel('Noramlized= Frequency w-->')
16 ylabel('Magnitude response Hd(w)------------------>')
17 xgrid(2)
```

Scilab code Solution 6.7 Chebyshev IIR BPF

```
1 //IIR Chebyshev BPF Filter
2 clear all;
3 clc;
4 n=3;
5 ftype='bp';
6 fdesign = 'cheb1';
7 frq=[0.15,0.25];
```

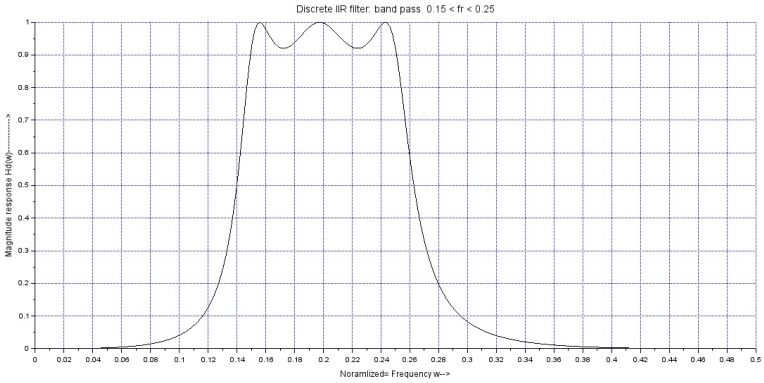

Figure 6.7: Chebyshev IIR BPF

```
8  delta=[0.08,0.02]
9  hz=iir(n,ftype,fdesign,frq);
10 [hzm,fr]=frmag(hz,256);
11 plot2d(fr',hzm')
12 xtitle('Discrete IIR filter: band pass  0.15 < fr <
     0.25 ',' ',' ');
13 xlabel('Noramlized= Frequency w-->')
14 ylabel('Magnitude response Hd(w)------------------>')
15 xgrid(2)
```

Scilab code Solution 6.8 Chebychev IIR BSF

```
1  //IIR Chebyshev BSF Filter
2  clear;
3  clc;
4  n=3;
5  ftype='sb';
6  fdesign = 'cheb1';
7  frq=[0.15,0.25];
```

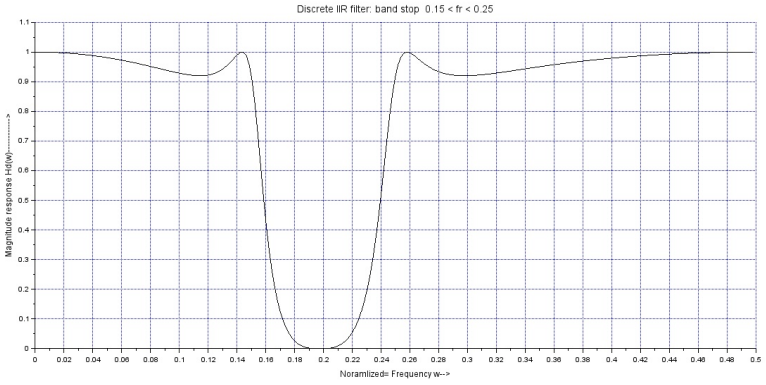

Figure 6.8: Chebychev IIR BSF

```
delta=[0.08,0.02]
hz=iir(n,ftype,fdesign,frq);
[hzm,fr]=frmag(hz,256);
plot2d(fr',hzm')
xtitle('Discrete IIR filter: band stop  0.15 < fr < 0.25 ',' ',' ');
xlabel('Noramlized= Frequency w-->')
ylabel('Magnitude response Hd(w)------------------>')
xgrid(2)
```