Scilab Textbook Companion for Digital Signal Processing by P. Ramesh Babu[1]

Created by
Mohammad Faisal Siddiqui
B.Tech (pursuing)
Electronics Engineering
Jamia Milia Islamia
College Teacher
Dr. Sajad A. Loan, JMI, New D
Cross-Checked by
Santosh Kumar, IITB

July 12, 2011

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

Book Description

Title: Digital Signal Processing

Author: P. Ramesh Babu

Publisher: Scitech Publications (INDIA) Pvt. Ltd.

Edition: 4

Year: 2010

ISBN: 81-8371-081-7

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

List of Figures

Chapter 1

DISCRETE TIME SIGNALS AND LINEAR SYSTEMS

Scilab code Exa 1.1 Continuous Time Plot and Discrete Time Plot

```
//Example 1.1
//Sketch the continuous time signal x(t)=2*exp(-2t)
   and also its discrete time equivalent signal with
    a sampling period T = 0.2sec
clear all;
clc ;
close ;
t=0:0.01:2;
x1=2*exp(-2*t);
subplot(1,2,1);
plot(t,x1);
xlabel('t');
ylabel('x(t)');
title('CONTINUOUS TIME PLOT');
n=0:0.2:2;
x2=2*exp(-2*n);
subplot(1,2,2);
```

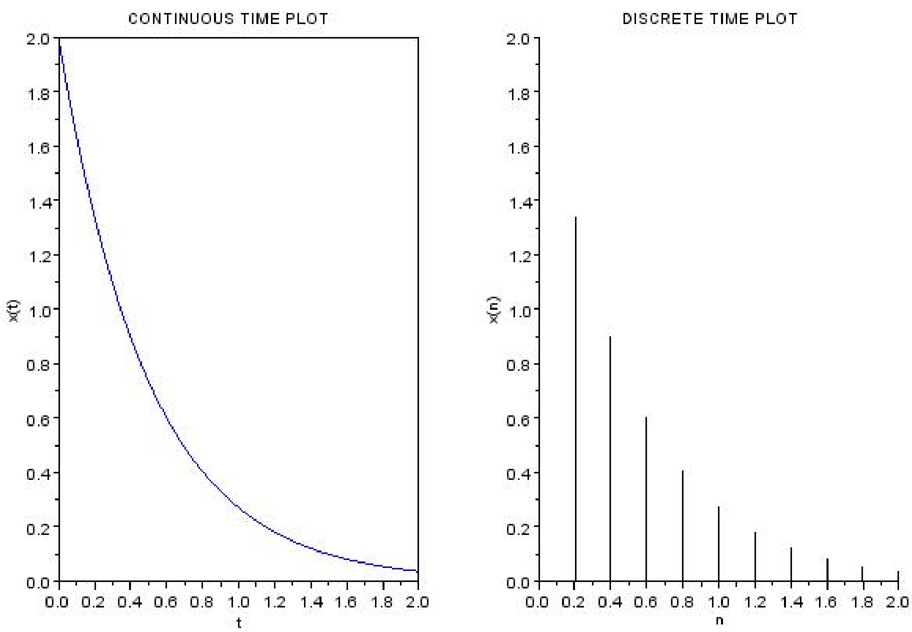

Figure 1.1: Continuous Time Plot and Discrete Time Plot

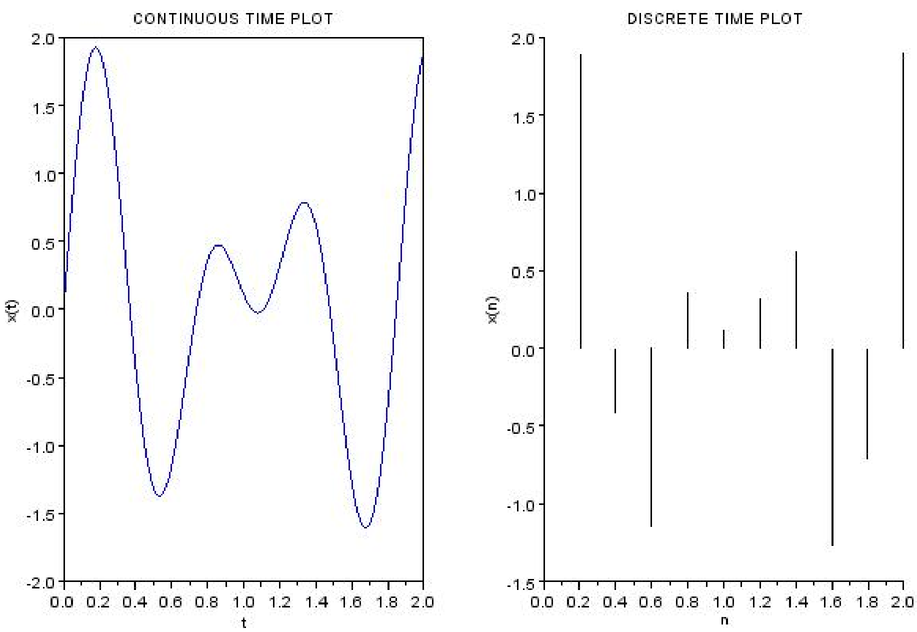

Figure 1.2: Continuous Time Plot and Discrete Time Plot

```
plot2d3(n,x2);
xlabel('n');
ylabel('x(n)');
title('DISCRETE TIME PLOT');
```

Scilab code Exa 1.2 Continuous Time Plot and Discrete Time Plot

```
//Example 1.2
//Sketch the continuous time signal x=sin(7*t)+sin
    (10*t) and also its discrete time equivalent
    signal with a sampling period T = 0.2sec
```

```
clear all;
clc ;
close ;
t=0:0.01:2;
x1=sin(7*t)+sin(10*t);
subplot(1,2,1);
plot(t,x1);
xlabel('t');
ylabel('x(t)');
title('CONTINUOUS TIME PLOT');
n=0:0.2:2;
x2=sin(7*n)+sin(10*n);
subplot(1,2,2);
plot2d3(n,x2);
xlabel('n');
ylabel('x(n)');
title('DISCRETE TIME PLOT');
```

Scilab code Exa 1.3.a Evaluate the Summations

```
//Example 1.3 (a)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Calculate Following Summations
clear all;
clc ;
close ;
syms n;
X= symsum (sin(2*n),n ,2, 2);
//Display the result in command window
disp (X,"The Value of summation comes out to be:");
```

Scilab code Exa 1.3.b Evaluate the Summations

```
//Example 1.3 (b)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Calculate Following Summations
clear all;
clc ;
close ;
syms n;
X= symsum (%e^(2*n),n ,0, 0);
//Display the result in command window
disp (X,"The Value of summation comes out to be:");
```

Scilab code Exa 1.4.a Check for Energy or Power Signals

```
//Example 1.4 (a)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Find Energy and Power of Given Signals
clear all;
clc ;
close ;
syms n N;
x=(1/3)^n;
E= symsum (x^2,n ,0, %inf);
//Display the result in command window
disp (E,"Energy:");
p=(1/(2*N+1))*symsum (x^2,n ,0, N);
P=limit(p,N,%inf);
disp (P,"Power:");
//The Energy is Finite and Power is 0. Therefore the
    given signal is an Energy Signal
```

Scilab code Exa 1.4.d Check for Energy or Power Signals

```
//Example 1.4 (d)
```

```
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Find Energy and Power of Given Signals
clear all;
clc ;
close ;
syms n N;
x=%e^(2*n);
E= symsum (x^2,n ,0, %inf);
//Display the result in command window
disp (E,"Energy:");
p=(1/(2*N+1))*symsum (x^2,n ,0, N);
P=limit(p,N,%inf);
disp (P,"Power:");
//The Energy andPower is infinite. Therefore the
   given signal is an neither Energy Signal nor
   Power Signal
```

Scilab code Exa 1.5.a Determining Periodicity of Signal

```
//Example 1.5 (a)
//To Determine Whether Given Signal is Periodic or
   not
clear all;
clc ;
close ;
t=0:0.01:2;
x1=exp(%i*6*%pi*t);
subplot(1,2,1);
plot(t,x1);
xlabel('t');
ylabel('x(t)');
title('CONTINUOUS TIME PLOT');
n=0:0.2:2;
```

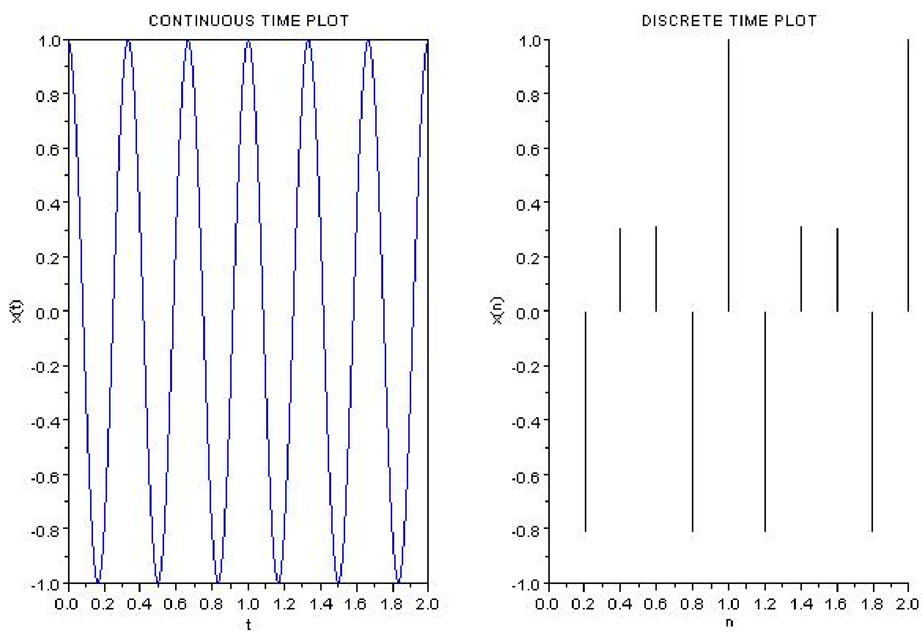

Figure 1.3: Determining Periodicity of Signal

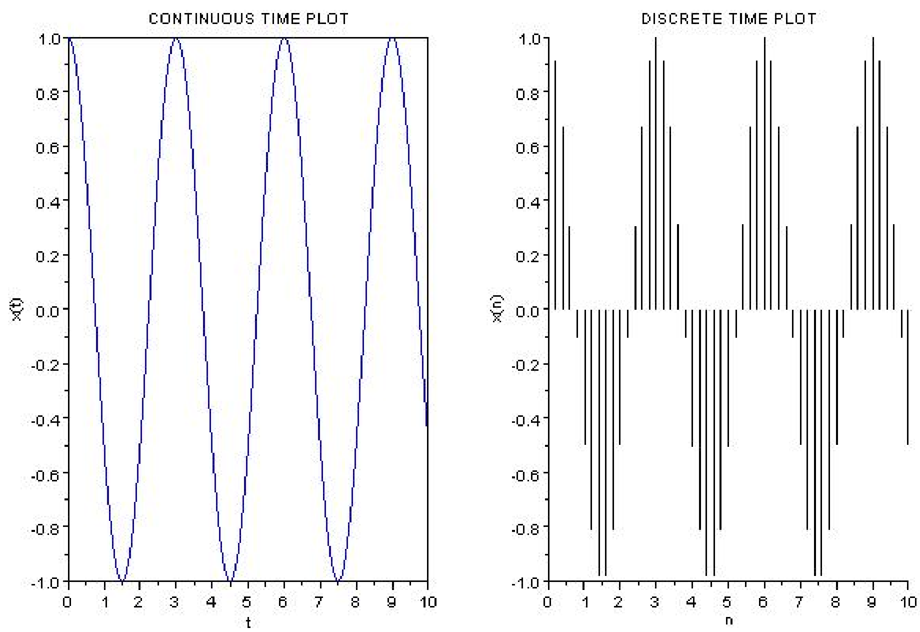

Figure 1.4: Determining Periodicity of Signal

```
x2=exp(%i*6*%pi*n);
subplot(1,2,2);
plot2d3(n,x2);
xlabel('n');
ylabel('x(n)');
title('DISCRETE TIME PLOT');
//Hence Given Signal is Periodic with N=1
```

Scilab code Exa 1.5.c Determining Periodicity of Signal

```
//Example 1.5 (c)
```

```
//To Determine Whether Given Signal is Periodic or
   not
clear all;
clc ;
close ;
t=0:0.01:10;
x1=cos(2*%pi*t/3);
subplot(1,2,1);
plot(t,x1);
xlabel('t');
ylabel('x(t)');
title('CONTINUOUS TIME PLOT');
n=0:0.2:10;
x2=cos(2*%pi*n/3);
subplot(1,2,2);
plot2d3(n,x2);
xlabel('n');
ylabel('x(n)');
title('DISCRETE TIME PLOT');
//Hence Given Signal is Periodic with N=3
```

Scilab code Exa 1.5.d Determining Periodicity of Signal

```
//Example 1.5 (d)
//To Determine Whether Given Signal is Periodic or
   not
clear all;
clc ;
close ;
t=0:0.01:50;
x1=cos(%pi*t/3)+cos(3*%pi*t/4);
subplot(1,2,1);
plot(t,x1);
```

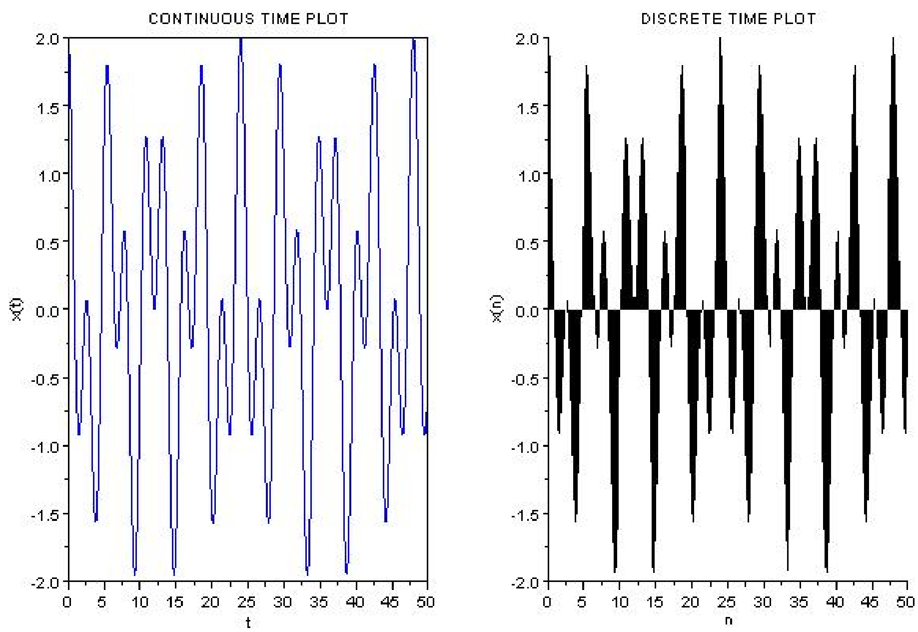

Figure 1.5: Determining Periodicity of Signal

```
xlabel('t');
ylabel('x(t)');
title('CONTINUOUS TIME PLOT');
n=0:0.2:50;
x2=cos(%pi*n/3)+cos(3*%pi*n/4);
subplot(1,2,2);
plot2d3(n,x2);
xlabel('n');
ylabel('x(n)');
title('DISCRETE TIME PLOT');
//Hence Given Signal is Periodic with N=24
```

Scilab code Exa 1.11 Stability of the System

```
//Example 1.11
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Testing Stability of Given System
clear all;
clc ;
close ;
syms n;
x =(1/2)^n
X= symsum (x,n ,0, %inf );
//Display the result in command window
disp (X,"Summation is :");
disp('Hence Summation < infinity. Given System is
   Stable');
```

Scilab code Exa 1.12 Convolution Sum of Two Sequences

```
//Example 1.12
//Program to Compute convolution of given sequences
//x(n)=[3 2 1 2], h(n)=[1 2 1 2];
```

```
clear all;
clc ;
close ;
x=[3 2 1 2];
h=[1 2 1 2];
y=convol(x,h);
disp(y);
```

Scilab code Exa 1.13 Convolution of Two Signals

```
//Example 1.13
//Program to Compute convolution of given sequences
//x(n)=[1 2 1 1], h(n)=[1 −1 1 −1];
clear all;
clc ;
close ;
x=[1 2 1 1];
h=[1 -1 1 -1];
y=convol(x,h);
disp(round(y));
```

Scilab code Exa 1.18 Cross Correlation of Two Sequences

```
//Example 1.18
//Program to Compute Cross−correlation  of given
   sequences
//x(n)=[1 2 1 1], h(n)=[1 1 2 1];
clear all;
clc ;
close ;
x=[1 2 1 1];
h=[1 1 2 1];
h1=[1 2 1 1];
```

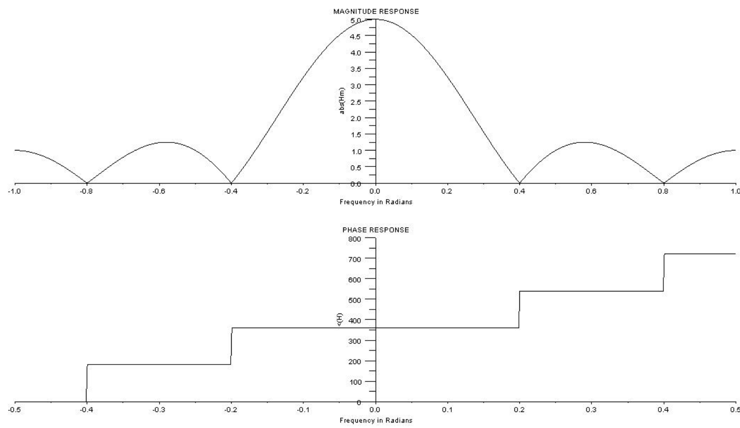

Figure 1.6: Plot Magnitude and Phase Response

```
y=convol(x,h1);
disp(round(y));
```

Scilab code Exa 1.19 Determination of Input Sequence

```
//Example 1.19
//To find input x(n)
//h(n)=[1 2 1], y(n)=[1 5 10 11 8 4 1]
clear all;
clc ;
close ;
z=%z;
a=z^6+5*(z^(5))+10*(z^(4))+11*(z^(3))+8*(z^(2))+4*(z
   ^(1))+1;
b=z^6+2*z^(5)+1*z^(4);
x =ldiv(a,b,5);
disp (x,"x(n)=");
```

Scilab code Exa 1.32.a Plot Magnitude and Phase Response

```
//Example 1.32
//Program to Plot Magnitude and Phase Responce
clear all;
clc ;
close ;
w=-%pi:0.01:%pi;
H=1+2*cos(w)+2*cos(2*w);
//caluculation of Phase and Magnitude of H
[phase_H,m]=phasemag(H);
Hm=abs(H);
a=gca();
subplot(2,1,1);
a.y_location="origin";
plot2d(w/%pi,Hm);
xlabel('Frequency in Radians')
ylabel('abs(Hm)');
title('MAGNITUDE RESPONSE');
subplot(2,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d(w/(2*%pi),phase_H);
xlabel('Frequency in Radians');
ylabel('<(H)');
title('PHASE RESPONSE');
```

Scilab code Exa 1.37 Sketch Magnitude and Phase Response

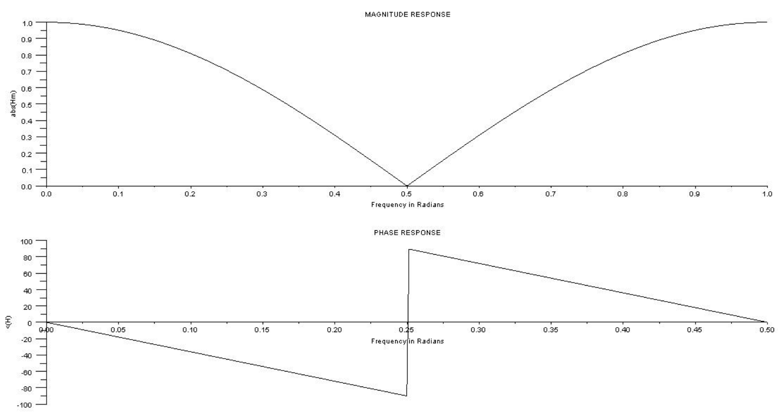

Figure 1.7: Sketch Magnitude and Phase Response

```
//Example 1.37
//Program to Plot Magnitude and Phase Responce
//y(n)=1/2[x(n)+x(n-2)]
clear all;
clc ;
close ;
w=0:0.01:%pi;
H=(1+cos(2*w)-%i*sin(2*w))/2;
//caluculation of Phase and Magnitude of H
[phase_H,m]=phasemag(H);
Hm=abs(H);
a=gca();
subplot(2,1,1);
a.y_location="origin";
plot2d(w/%pi,Hm);
xlabel('Frequency in Radians')
ylabel('abs(Hm)');
title('MAGNITUDE RESPONSE');
subplot(2,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d(w/(2*%pi),phase_H);
```

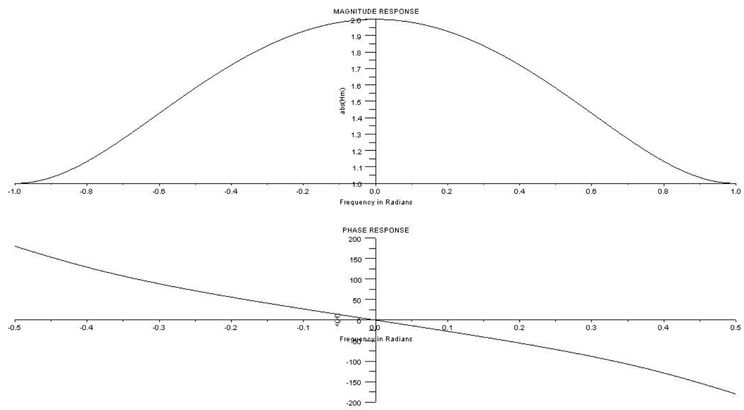

Figure 1.8: Plot Magnitude and Phase Response

```
xlabel('Frequency in Radians');
ylabel('<(H)');
title('PHASE RESPONSE');
```

Scilab code Exa 1.38 Plot Magnitude and Phase Response

```
//Example 1.38
//Program to Plot Magnitude and Phase Responce
//0.5delta(n)+delta(n-1)+0.5delta(n-2)
clear all;
clc ;
close ;
w=-%pi:0.01:%pi;
H=0.5+exp(-%i*w)+0.5*exp(-%i*w);
//caluculation of Phase and Magnitude of H
[phase_H,m]=phasemag(H);
Hm=abs(H);
a=gca();
subplot(2,1,1);
```

```
a.y_location="origin";
plot2d(w/%pi,Hm);
xlabel('Frequency in Radians')
ylabel('abs(Hm)');
title('MAGNITUDE RESPONSE');
subplot(2,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d(w/(2*%pi),phase_H);
xlabel('Frequency in Radians');
ylabel('<(H)');
title('PHASE RESPONSE');
```

Scilab code Exa 1.45 Filter to Eliminate High Frequency Component

```
//Example 1.45
//
clear all;
clc ;
close ;
t=0:0.01:10;
x=2*cos(5*t)+cos(300*t);
x1=2*cos(5*t);
b=[0.05 0.05];
a=[1 -0.9];
y=filter(b,a,x);
subplot(2,1,1);
plot(t,x);
xlabel('Time in Sec');
ylabel('Amplitude');
subplot(2,1,2);
plot(t,y);
```

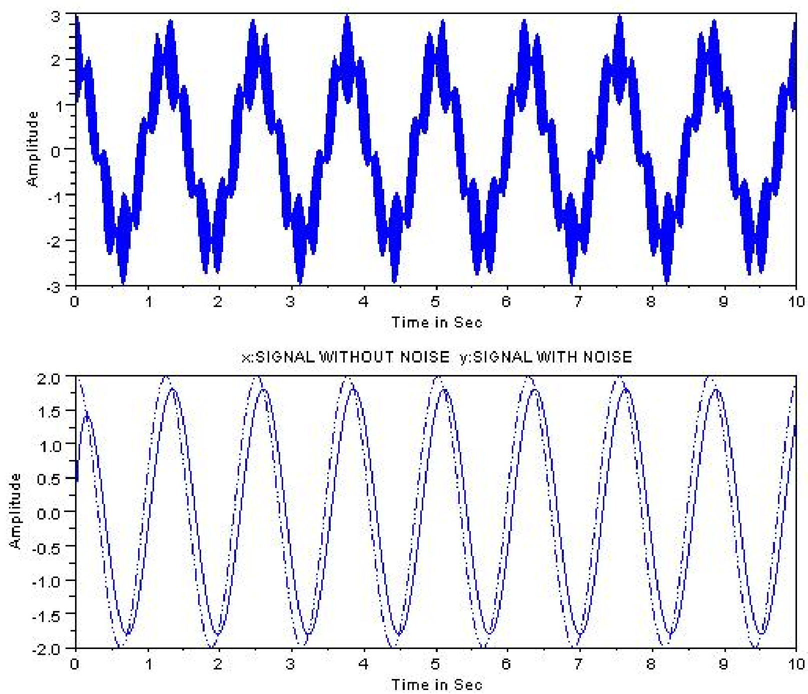

Figure 1.9: Filter to Eliminate High Frequency Component

```
subplot(2,1,2);
plot(t,x1,':');
title('x:SIGNAL WITHOUT NOISE  y:SIGNAL WITH NOISE')
    ;
xlabel('Time in Sec');
ylabel('Amplitude');
```

Scilab code Exa 1.57.a Discrete Convolution of Sequences

```
//Example 1.57 (a)
//Program to Compute discrete convolution of given
    sequences
//x(n)=[1 2 -1 1], h(n)=[1 0 1 1];
clear all;
clc ;
close ;
x=[1 2 -1 1];
h=[1 0 1 1];
y=convol(x,h);
disp(round(y));
```

Scilab code Exa 1.61 Fourier Transform

```
//Example 1.61
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Fourier transform of (3)^n u(n)
clear all;
clc ;
close ;
syms n;
x =(3) ^n;
X= symsum (x,n ,0, %inf )
//Display the result in command window
```

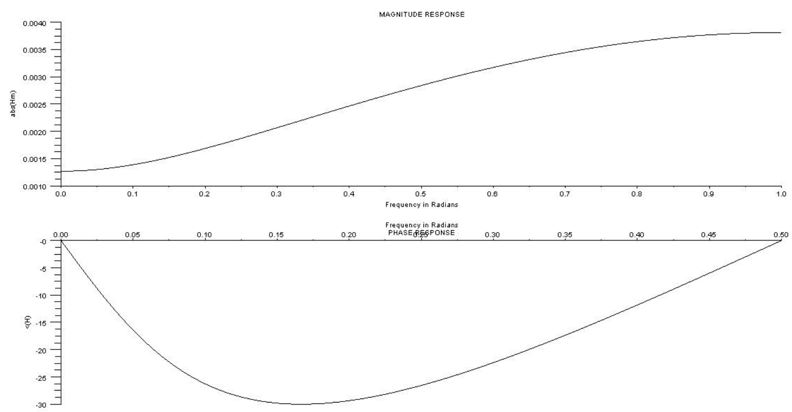

Figure 1.10: Frequency Response of LTI System

```
disp (X,"The Fourier Transform does not exit as x(n)
    is not absolutely summable and approaches
   infinity i.e.");
```

Scilab code Exa 1.62 Fourier Transform

```
//Example 1.62
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Fourier transform of (0.8)^|n| u(n)
clear all;
clc ;
close ;
syms w n;
X= symsum ((0.8)^n*%e^(%i*w*n),n ,1, %inf )+symsum
   ((0.8)^n*%e^(-%i*w*n),n ,0, %inf )
//Display the result in command window
disp (X,"The Fourier Transform comes out to be:");
```

Scilab code Exa 1.64.a Frequency Response of LTI System

```
//Example 1.64 (a)
//Program to Calculate Plot Magnitude and Phase
   Responce
clear all;
clc ;
close ;
w=0:0.01:%pi;
H=1/(1-0.5*%e^(-%i*w));
//caluculation of Phase and Magnitude of H
[phase_H,m]=phasemag(H);
Hm=abs(H);
a=gca();
subplot(2,1,1);
a.y_location="origin";
plot2d(w/%pi,Hm);
xlabel('Frequency in Radians')
ylabel('abs(Hm)');
title('MAGNITUDE RESPONSE');
subplot(2,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d(w/(2*%pi),phase_H);
xlabel('Frequency in Radians');
ylabel('<(H)');
title('PHASE RESPONSE');
```

Scilab code Exa 1.64.c Frequency Response of LTI System

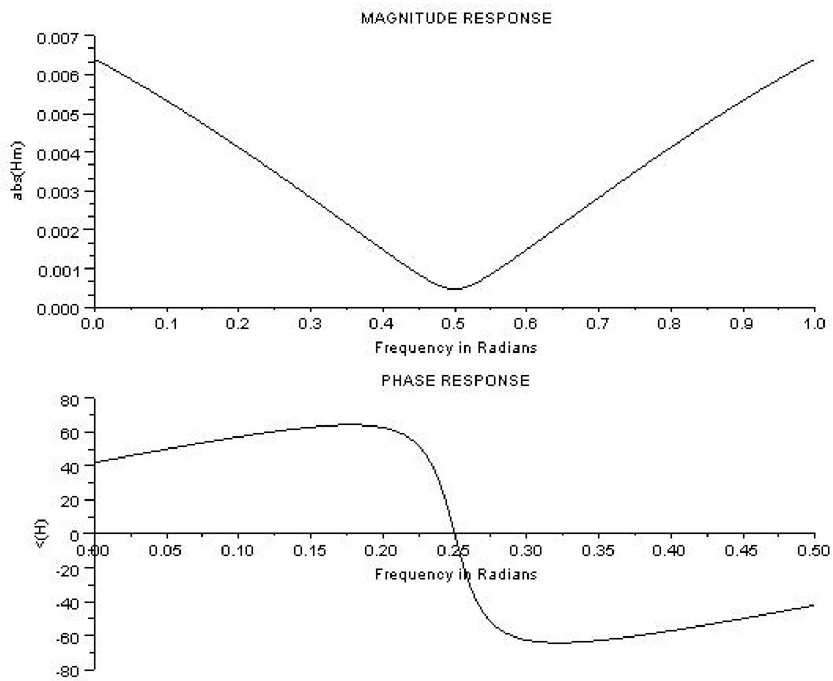

Figure 1.11: Frequency Response of LTI System

```
//Example 1.64 (c)
//Program to Calculate Plot Magnitude and Phase Responce
clear all;
clc ;
close ;
w=0:0.01:%pi;
H=1/(1-0.9*%i*%e^(-%i*w));
//caluculation of Phase and Magnitude of H
[phase_H,m]=phasemag(H);
Hm=abs(H);
a=gca();
subplot(2,1,1);
a.y_location="origin";
plot2d(w/%pi,Hm);
xlabel('Frequency in Radians')
ylabel('abs(Hm)');
title('MAGNITUDE RESPONSE');
subplot(2,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d(w/(2*%pi),phase_H);
xlabel('Frequency in Radians');
ylabel('<(H)');
title('PHASE RESPONSE');
```

Chapter 2

THE Z TRANSFORM

Scilab code Exa 2.1 z Transform and ROC of Causal Sequence

```
//Example 2.1
//Z- transform of [1 0 3 -1 2]
clear all;
clc ;
close ;
function[za]=ztransfer(sequence,n)
z=poly(0,'z','r')
za=sequence*(1/z)^n'
endfunction
x1=[1 0 3 -1 2];
n=0:length(x1)-1;
zz=ztransfer(x1,n);
//Display the result in command window
disp (zz,"Z-transform of sequence is:");
disp('ROC is the entire plane except z = 0');
```

Scilab code Exa 2.2 z Transform and ROC of Anticausal Sequence

```
//Example 2.2
//Z- transform of [-3 -2 -1 0 1]
clear all;
clc ;
close ;
function[za]=ztransfer(sequence,n)
z=poly(0,'z','r')
za=sequence*(1/z)^n'
endfunction
x1=[-3 -2 -1 0 1];
n=-(length(x1)-1):0;
zz=ztransfer(x1,n);
//Display the result in command window
disp (zz,"Z-transform of sequence is:");
disp('ROC is the entire plane except z = %inf');
```

Scilab code Exa 2.3 z Transform of the Sequence

```
//Example 2.3
//Z- transform of [2 -1 3 2 1 0 2 3 -1]
clear all;
clc ;
close ;
function[za]=ztransfer(sequence,n)
z=poly(0,'z','r')
za=sequence*(1/z)^n'
endfunction
x1=[2 -1 3 2 1 0 2 3 -1];
n=-4:4;
zz=ztransfer(x1,n);
//Display the result in command window
disp (zz,"Z-transform of sequence is:");
disp('ROC is the entire plane except z = 0 and z =
    %inf');
```

Scilab code Exa 2.4 z Transform and ROC of the Signal

```
//Example 2.4
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of a^n u(n)
clear all;
clc ;
close ;
syms a n z;
x =a^n
X= symsum (x*(z^(-n)),n ,0, %inf );
//Display the result in command window
disp (X,"Z-transform of a^n u(n) with is:");
disp('ROC is the Region mod(z) > a')
```

Scilab code Exa 2.5 z Transform and ROC of the Signal

```
//Example 2.5
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of -b^n u(-n-1)
clear all;
clc ;
close ;
syms b n z;
x =b^n
X= symsum (x*(z^(-n)),n ,0, %inf );
//Display the result in command window
disp (X,"Z-transform of b^n u(n) with is:");
disp('ROC is the Region mod(z) < b')
```

Scilab code Exa 2.6 Stability of the System

```
//Example 2.6
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of 2^n u(n)
clear all;
clc ;
close ;
syms n z;
x =(2) ^n
X= symsum (x*(z^(-n)),n ,0, %inf );
//Display the result in command window
disp (X,"Z-transform of 2^n u(n) is:");
disp('ROC is the Region mod(z) > 2');
```

Scilab code Exa 2.7 z Transform of the Signal

```
//Example 2.7
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of [3(3^n)-4(2)^n] u(n)
clear all;
clc ;
close ;
syms n z;
x1 =(3) ^(n);
X1= symsum (3* x1 *(z^(-n)),n ,0, %inf );
x2 =(4) ^(n);
X2= symsum (4* x2 *(z^(-n)),n ,0, %inf );
X = (X1 -X2);
//Display the result in command window
disp (X,"Z-transform of [3(3^n)-4(2)^n] u(n)  is:");
```

Scilab code Exa 2.8.a z Transform of the Signal

```
//Example 2.8 (a)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform of cos(Wo*n)
clc;
syms Wo n z;
x1=exp(sqrt(-1)*Wo*n);
X1=symsum(x1*(z^-n),n,0,%inf);
x2=exp(-sqrt(-1)*Wo*n);
X2=symsum(x2*(z^-n),n,0,%inf);
X=(X1+X2)/2;
disp(X,'X(z)=');
```

Scilab code Exa 2.9 z Transform of the Sequence

```
//Example 2.9
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of (1/3)^n u(n-1)
clear all;
clc ;
close ;
syms n z;
x =(1/3)^n;
X= (1/z)*symsum (x*(z^(-n)),n ,0, %inf );
//Display the result in command window
disp (X,"Z-transform of (1/3)^n u(n-1) is:");
```

Scilab code Exa 2.10 z Transform Computation

```
//Example 2.10
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform of r^n.cos(Wo*n)
clc;
syms r Wo n z;
```

```
x1=(r^n)*exp(sqrt(-1)*Wo*n);
X1=symsum(x1*(z^-n),n,0,%inf);
x2=(r^n)*exp(-sqrt(-1)*Wo*n);
X2=symsum(x2*(z^-n),n,0,%inf);
X=(X1+X2)/2;
disp(X,'X(z)=');
```

Scilab code Exa 2.11 z Transform of the Sequence

```
//Example 2.11
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of n.a^n u(n)
clear all;
clc ;
close ;
syms a n z;
x =(a) ^n;
X= symsum (x*(z^(-n)),n ,0, %inf )
Y = diff (X,z);
//Display the result in command window
disp (Y,"Z-transform of n.a^n u(n)  is:");
```

Scilab code Exa 2.13.a z Transform of Discrete Time Signals

```
//Example 2.13 (a)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of (-1/5)^n u(n)+5(1/2)^(-n)u(-n-1)
clear all;
clc ;
close ;
syms n z;
x1 =(-1/5)^n ;
X1= symsum (x1 *(z^(-n)),n ,0, %inf );
```

```
x2 =(1/2)^(-n);
X2= symsum (5* x2 *(z^(-n)),n ,0, %inf );
X = (X1 -X2);
//Display the result in command window
disp (X,"Z-transform of [3(3^n)-4(2)^n] u(n)  is:");
disp('ROC is the Region 1/5 < mod(z) < 2');
```

Scilab code Exa 2.13.b z Transform of Discrete Time Signals

```
//Example 2.13 (b)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform
clc;
syms n z k;
x1=1;
X1=symsum(x1*z^(-n),n,0,0);
x2=1;
X2=symsum(x2*z^(-n),n,1,1);
x3=1;
X3=symsum(x3*z^(-n),n,2,2);
X=0.5*X1+X2-1/3*X3;
disp(X,'X(z)=');
```

Scilab code Exa 2.13.c z Transform of Discrete Time Signals

```
//Example 2.13 (c)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of u(n-2)
clear all;
clc ;
close ;
syms n z;
x =1;
```

```
X= (1/(z^2))*symsum (x*(z^(-n)),n ,0, %inf );
//Display the result in command window
disp (X,"Z-transform of u(n-2) is:");
```

Scilab code Exa 2.13.d z Transform of Discrete Time Signals

```
//Example 2.13 (d)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of (n+0.5)((1/3)^n)u(n)
clear all;
clc ;
close ;
syms n z;
x1 =(1/3)^n;
X11= symsum (x1*(z^(-n)),n ,0, %inf )
X1 = diff (X11,z);
x2 =(1/3) ^(n);
X2= symsum (0.5* x2 *(z^(-n)),n ,0, %inf );
X = (X1+X2);
//Display the result in command window
disp (X,"Z-transform of (n+0.5)((1/3)^n)u(n)  is:");
```

Scilab code Exa 2.16 Impulse Response of the System

```
//Example 2.16
//To find input h(n)
//a=[1 2 -4 1], b=[1]
clear all;
clc ;
close ;
z=%z;
a=z^3+2*(z^(2))-4*(z)+1;
b=z^3;
```

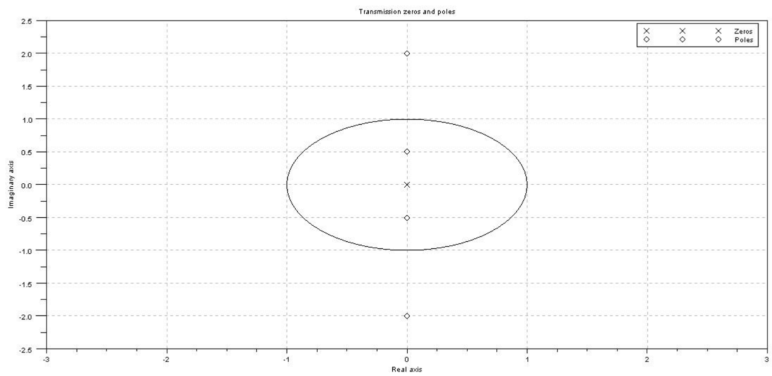

Figure 2.1: Pole Zero Plot of the Difference Equation

```
h =ldiv(a,b,4);
disp (h,"h(n)=");
```

Scilab code Exa 2.17 Pole Zero Plot of the Difference Equation

```
//Example 2.17
//To draw the pole-zero plot
clear all;
clc ;
close ;
z=%z
H1Z=((z)*(z-1))/((z-0.25)*(z-0.5));
xset('window',1);
plzr(H1Z);
```

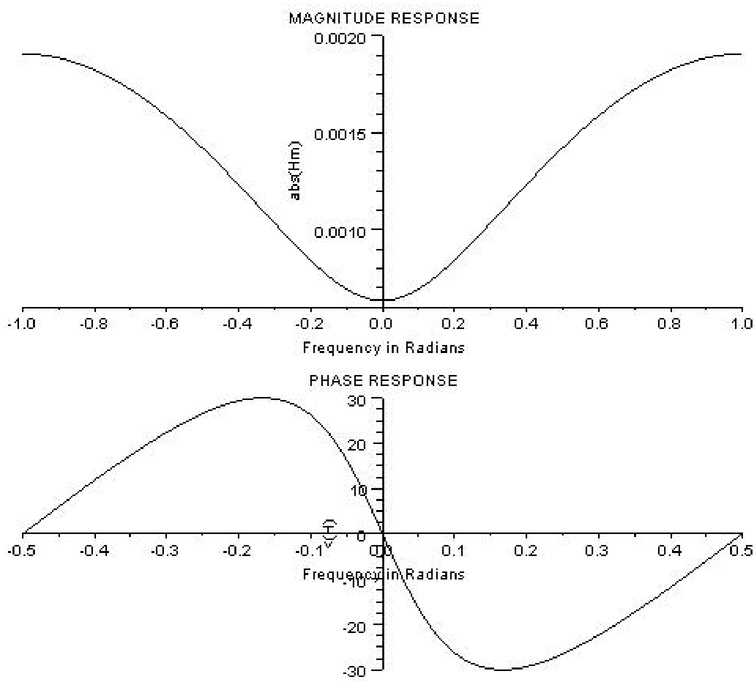

Figure 2.2: Frequency Response of the System

Scilab code Exa 2.19 Frequency Response of the System

```
//Example 2.19
//Program to Plot Magnitude and Phase Responce
clear all;
clc ;
close ;
w=-%pi:0.01:%pi;
H=1/(1-0.5*(cos(w)-%i*sin(w)));
//caluculation of Phase and Magnitude of H
[phase_H,m]=phasemag(H);
Hm=abs(H);
a=gca();
subplot(2,1,1);
a.y_location="origin";
plot2d(w/%pi,Hm);
xlabel('Frequency in Radians');
ylabel('abs(Hm)');
title('MAGNITUDE RESPONSE');
subplot(2,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d(w/(2*%pi),phase_H);
xlabel('Frequency in Radians');
ylabel('<(H)');
title('PHASE RESPONSE');
```

Scilab code Exa 2.20.a Inverse z Transform Computation

```
//Example 2.10 (a)
//To find input h(n)
//X(z)=(z+0.2)/((z+0.5)(z-1);
clear all;
clc ;
```

```
close ;
z=%z;
a=(z+0.5)*(z-1);
b=z+0.2;
h =ldiv(b,a,4);
disp (h,"h(n)=");
```

Scilab code Exa 2.22 Inverse z Transform Computation

```
//Example 2.22
//To find input x(n)
//X(z)=1/(2*z^(-2)+2*z^(-1)+1);
clear all;
clc ;
close ;
z=%z;
a=(2+2*z+z^2);
b=z^2;
h =ldiv(b,a,6);
disp (h,"First six values of h(n)=");
```

Scilab code Exa 2.23 Causal Sequence Determination

```
//Example 2.23
//To find input x(n)
//X(z)=1/(1-2z^(-1))(1-z^(-1))^2;
clear all;
clc ;
close ;
z=%z;
a=(z-2)*(z-1)^2;
b=z^3;
h =ldiv(b,a,6);
```

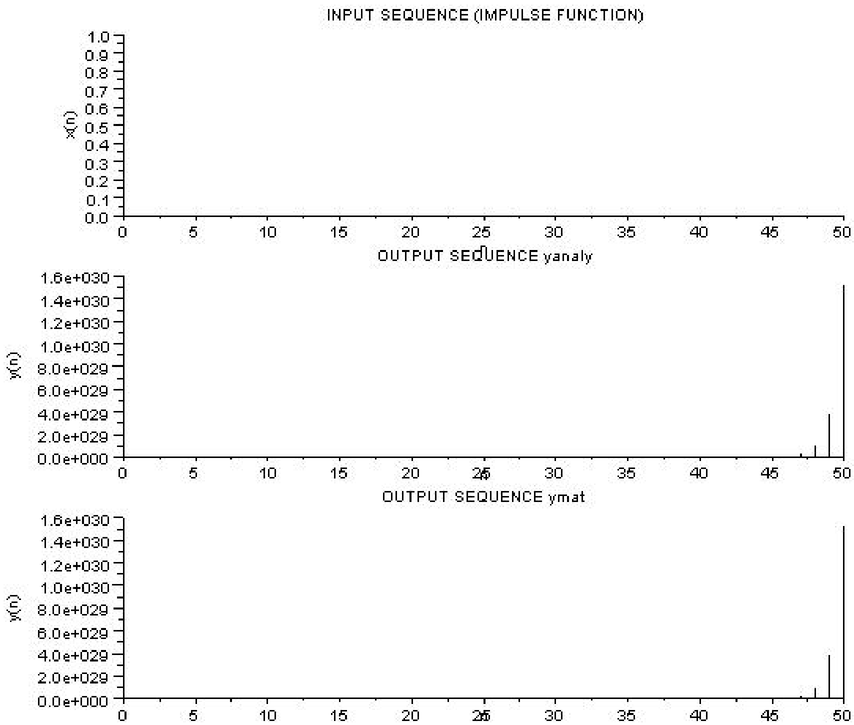

Figure 2.3: Impulse Response of the System

```
disp (h,"First six values of h(n)=");
```

Scilab code Exa 2.34 Impulse Response of the System

```
//Example 2.34
//To plot the impulse responce of the system
   analyically and using scilab
clear all;
clc ;
close ;
n=0:1:50;
```

```
x=[1,zeros(1,50)];
b=[1 2];
a=[1 -3 -4];
yanaly=6/5*4.^n-1/5*(-1).^n;//Analytical Solution
ymat=filter(b,a,x);
subplot(3,1,1);
plot2d3(n,x);
xlabel('n');
ylabel('x(n)');
title('INPUT SEQUENCE (IMPULSE FUNCTION)');
subplot(3,1,2);
plot2d3(n,yanaly);
xlabel('n');
ylabel('y(n)');
title('OUTPUT SEQUENCE yanaly');
subplot(3,1,3);
plot2d3(n,ymat);
xlabel('n');
ylabel('y(n)');
title('OUTPUT SEQUENCE ymat');
//As the Analtical Plot matches the Scilab Plot
   hence it is the Responce of the system
```

Scilab code Exa 2.35.a Pole Zero Plot of the System

```
//Example 2.35 (a)
//To draw the pole-zero plot
clear all;
clc ;
close ;
z=%z
H1Z=(z)/(z^2-z-1);
xset('window',1);
```

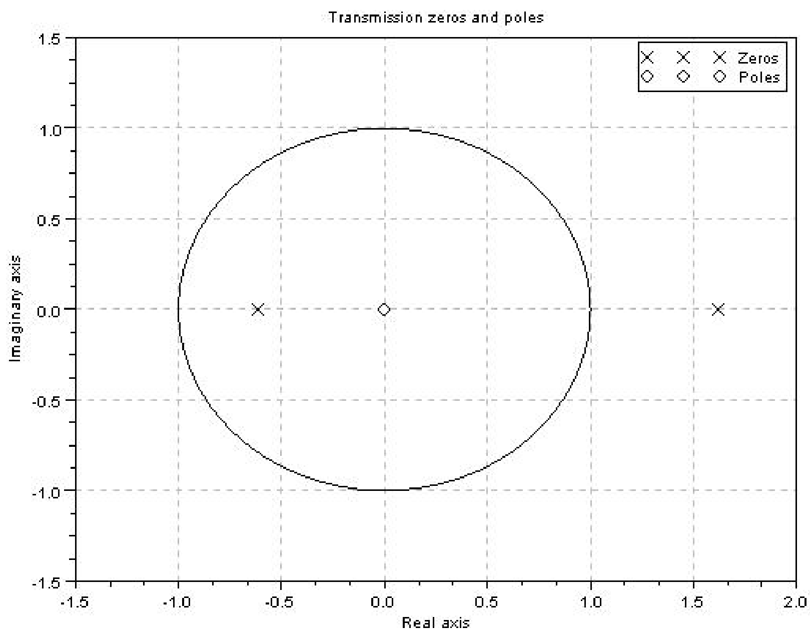

Figure 2.4: Pole Zero Plot of the System

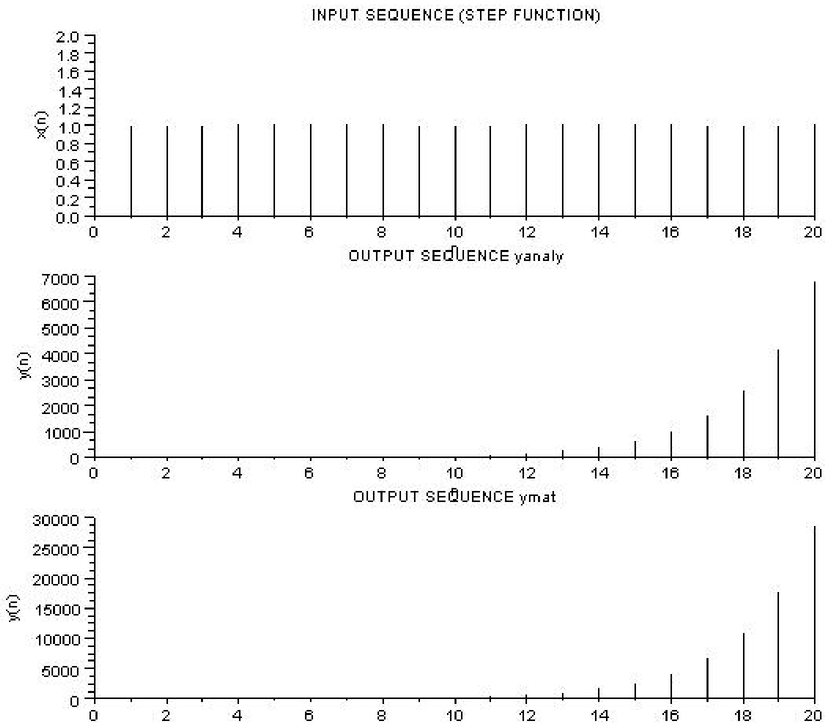

Figure 2.5: Unit Sample Response of the System

```
plzr(H1Z);
```

Scilab code Exa 2.35.b Unit Sample Response of the System

```
//Example 2.35 (b)
//To plot the responce of the system analyically and
    using scilab
clear all;
clc ;
close ;
n=0:1:20;
```

```
x=ones(1,length(n));
b=[0 1];
a=[1 -1 -1];
yanaly=0.447*(1.618).^n-0.447*(-0.618).^n;//Analytical Solution
[ymat,zf]=filter(b, a, x);
subplot(3,1,1);
plot2d3(n,x);
xlabel('n');
ylabel('x(n)');
title('INPUT SEQUENCE (STEP FUNCTION)');
subplot(3,1,2);
plot2d3(n,yanaly);
xlabel('n');
ylabel('y(n)');
title('OUTPUT SEQUENCE yanaly');
subplot(3,1,3);
plot2d3(n,ymat,zf);
xlabel('n');
ylabel('y(n)');
title('OUTPUT SEQUENCE ymat');
//As the Analtical Plot matches the Scilab Plot hence it is the Responce of the system
```

Scilab code Exa 2.38 Determine Output Response

```
//Example 2.38
//To plot the responce of the system analyically and using scilab
clear all;
clc ;
close ;
n=0:1:20;
```

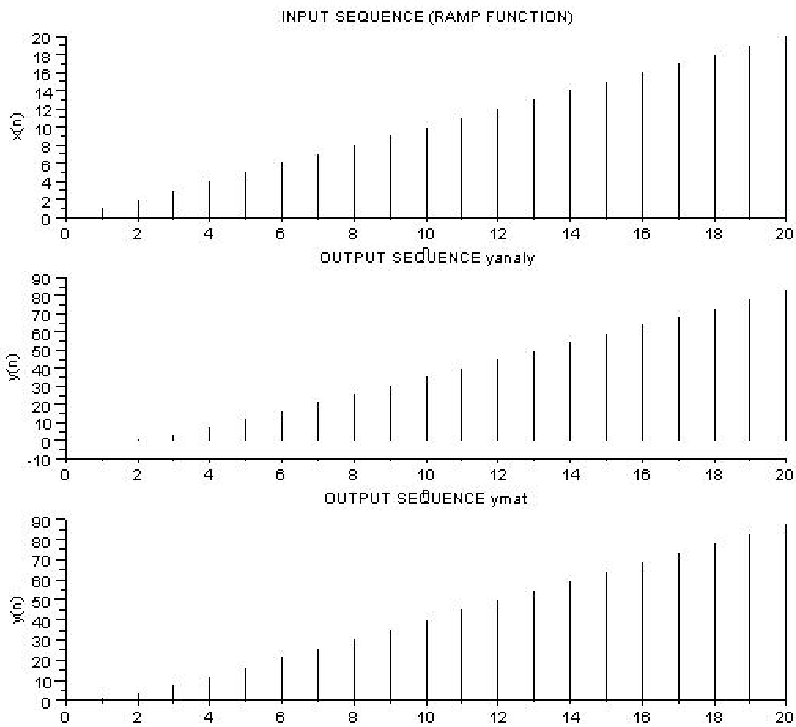

Figure 2.6: Determine Output Response

```
x=n;
b=[0 1 1];
a=[1 -0.7 0.12];
yanaly=38.89*(0.4).^n-26.53*(0.3).^n-12.36+4.76*n;//
    Analytical Solution
ymat=filter(b,a,x);
subplot(3,1,1);
plot2d3(n,x);
xlabel('n');
ylabel('x(n)');
title('INPUT SEQUENCE (RAMP FUNCTION)');
subplot(3,1,2);
plot2d3(n,yanaly);
xlabel('n');
ylabel('y(n)');
title('OUTPUT SEQUENCE yanaly');
subplot(3,1,3);
plot2d3(n,ymat);
xlabel('n');
ylabel('y(n)');
title('OUTPUT SEQUENCE ymat');
//As the Analtical Plot matches the Scilab Plot
    hence it is the Responce of the system
```

Scilab code Exa 2.40 Input Sequence Computation

```
//Example 2.40
//To find input x(n)
//h(n)=1 2 3 2, y(n)=[1 3 7 10 10 7 2]
clear all;
clc ;
close ;
z=%z;
a=z^6+3*(z^(5))+7*(z^(4))+10*(z^(3))+10*(z^(2))+7*(z
    ^(1))+2;
```

```
b=z^6+2*z^(5)+3*z^(4)+2*z^(3);
x =ldiv(a,b,4);
disp (x,"x(n)=");
```

Scilab code Exa 2.41.a z Transform of the Signal

```
//Example 2.41 (a)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of n.(-1)^n u(n)
clear all;
clc ;
close ;
syms a n z;
x =(-1) ^n;
X= symsum (x*(z^(-n)),n ,0, %inf )
Y = diff (X,z);
//Display the result in command window
disp (Y,"Z-transform of n.(-1)^n u(n)  is:");
```

Scilab code Exa 2.41.b z Transform of the Signal

```
//Example 2.41 (b)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z- transform of n^2 u(n)
clear all;
clc ;
close ;
syms n z;
x =1;
X= symsum (x*(z^(-n)),n ,0, %inf )
Y = diff(diff (X,z),z);
//Display the result in command window
disp (Y,"Z-transform of n^2 u(n)  is:");
```

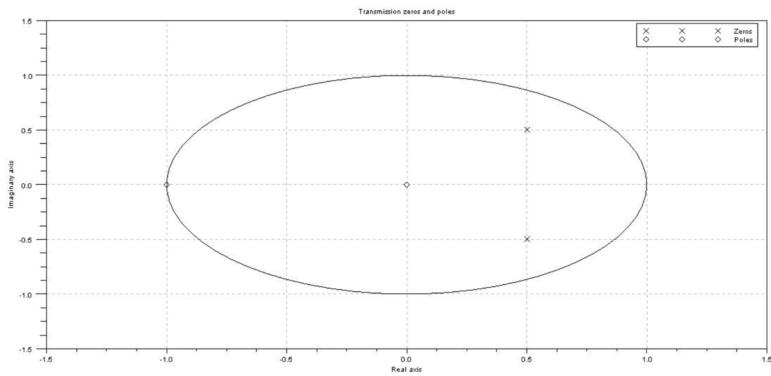

Figure 2.7: Pole Zero Pattern of the System

Scilab code Exa 2.41.c z Transform of the Signal

```
//Example 2.41 (c)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform of (-1)^n.cos(%pi/3*n)
clc;
syms n z;
Wo=%pi/3;
x1=exp(sqrt(-1)*Wo*n);
X1=(-1)^n*symsum(x1*(z^-n),n,0,%inf);
x2=exp(-sqrt(-1)*Wo*n);
X2=(-1)^n*symsum(x2*(z^-n),n,0,%inf);
X=(X1+X2)/2;
disp(X,'X(z)=');
```

Scilab code Exa 2.45 Pole Zero Pattern of the System

```
//Example 2.45
//To draw the pole-zero plot
clear all;
clc ;
close ;
z=%z
H1Z=((z)*(z+1))/(z^2-z+0.5);
xset('window',1);
plzr(H1Z);
```

Scilab code Exa 2.53.a z Transform of the Sequence

```
//Example 2.53 (a)
//Z- transform of [3 1 2 5 7 0 1]
clear all;
clc ;
close ;
function[za]=ztransfer(sequence,n)
z=poly(0,'z','r')
za=sequence*(1/z)^n'
endfunction
x1=[3 1 2 5 7 0 1];
n=-3:3;
zz=ztransfer(x1,n);
//Display the result in command window
disp (zz,"Z-transform of sequence is:");
```

Scilab code Exa 2.53.b z Transform of the Signal

```
//Example 2.53 (b)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
```

```
//Z transform of delta(n)
clc;
syms n z;
x=1;
X=symsum(x*z^(-n),n,0,0);
disp(X,'X(z)=');
```

Scilab code Exa 2.53.c z Transform of the Signal

```
//Example 2.53 (c)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform of delta(n)
clc;
syms n z k;
x=1;
X=symsum(x*z^(-n),n,k,k);
disp(X,'X(z)=');
```

Scilab code Exa 2.53.d z Transform of the Signal

```
//Example 2.53 (d)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform of delta(n)
clc;
syms n z kc;
x=1;
X=symsum(x*z^(-n),n,-k,-k);
disp(X,'X(z)=');
```

Scilab code Exa 2.54 z Transform of Cosine Signal

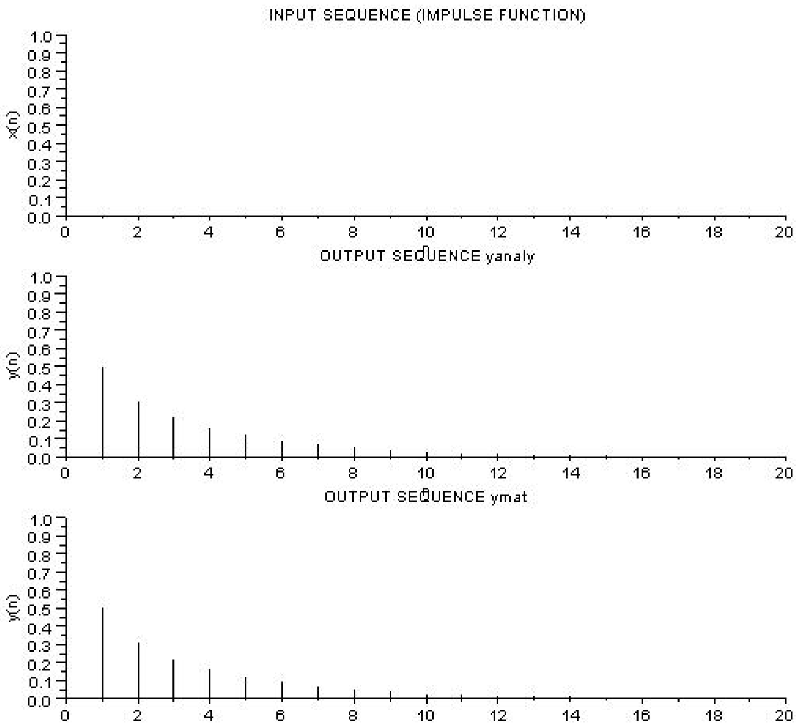

Figure 2.8: Impulse Response of the System

```
//Example 2.54
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform of cos(Wo*n)
clc;
syms Wo n z;
x1=exp(sqrt(-1)*Wo*n);
X1=symsum(x1*(z^-n),n,0,%inf);
x2=exp(-sqrt(-1)*Wo*n);
X2=symsum(x2*(z^-n),n,0,%inf);
X=(X1+X2)/2;
disp(X,'X(z)=');
```

Scilab code Exa 2.58 Impulse Response of the System

```
//Example 2.58
//To plot the responce of the system analyicically and
    using scilab
clear all;
clc ;
close ;
n=0:1:20;
x=[1 zeros(1,20)];
b=[1 -0.5];
a=[1 -1 3/16];
yanaly=0.5*(0.75).^n+0.5*(0.25).^n;//Analytical
   Solution
ymat=filter(b,a,x);
subplot(3,1,1);
plot2d3(n,x);
xlabel('n');
ylabel('x(n)');
title('INPUT SEQUENCE (IMPULSE FUNCTION)');
subplot(3,1,2);
plot2d3(n,yanaly);
xlabel('n');
ylabel('y(n)');
title('OUTPUT SEQUENCE yanaly');
subplot(3,1,3);
plot2d3(n,ymat);
xlabel('n');
ylabel('y(n)');
title('OUTPUT SEQUENCE ymat');
//As the Analtical Plot matches the Scilab Plot
   hence it is the Responce of the system
```

Chapter 3

THE DISCRETE FOURIER TRANSFORM

Scilab code Exa 3.1 DFT and IDFT

```
//Example 3.1
//Program to Compute the DFT of a Sequence x[n
    ]=[1,1,0,0]
//and IDFT of a Sequence Y[k]=[1,0,1,0]
clear all;
clc ;
close ;
x = [1,1,0,0];
//DFT Computation
X = fft (x , -1);
Y = [1,0,1,0];
//IDFT Computation
y = fft (Y , 1);
//Display sequence X[k] and y[n] in command window
disp(X,"X[k]=");
disp(y,"y[n]=");
```

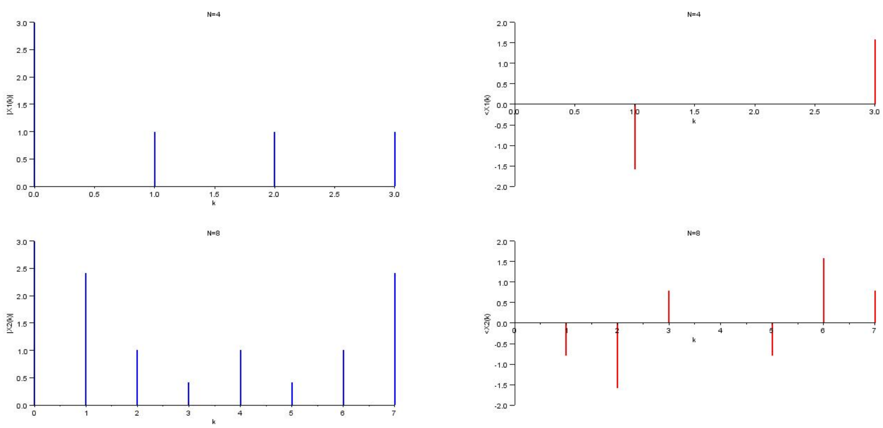

Figure 3.1: DFT of the Sequence

Scilab code Exa 3.2 DFT of the Sequence

```
//Example 3.2
//Program to Compute the DFT of a Sequence x[n]=1,
   0<=n<=2; and 0 otherwise
//for N=4 and N=8. Plot Magnitude and phase plots of
    each.
clear all;
clc ;
close ;
//N=4
x1 = [1,1,1,0];
//DFT Computation
X1 = fft (x1 , -1);
//N=8
x2 = [1,1,1,0,0,0,0,0];
//DFT Computation
X2 = fft (x2 , -1);
//Display sequence X1[k] and X2[k] in command window
disp(X1,"X1[k]=");
```

```
disp(X2,"X2[k]=");
//Plots for N=4
n1=0:1:3;
subplot(2,2,1);
a = gca ();
a.y_location ="origin";
a.x_location ="origin";
plot2d3(n1,abs(X1),2);
poly1=a.children(1).children (1);
poly1.thickness=2;
xtitle('N=4','k','|X1(k)|');
subplot(2,2,2);
a = gca ();
a.y_location ="origin";
a.x_location ="origin";
plot2d3(n1,atan(imag(X1),real(X1)),5);
poly1=a.children(1).children (1);
poly1.thickness=2;
xtitle('N=4','k','<X1(k)');
//Plots for N=8
n2=0:1:7;
subplot(2,2,3);
a = gca ();
a.y_location ="origin";
a.x_location ="origin";
plot2d3(n2,abs(X2),2);
poly1=a.children(1).children (1);
poly1.thickness=2;
xtitle('N=8','k','|X2(k)|');
subplot(2,2,4);
a = gca ();
a.y_location ="origin";
a.x_location ="origin";
plot2d3(n2,atan(imag(X2),real(X2)),5);
poly1=a.children(1).children (1);
poly1.thickness=2;
xtitle('N=8','k','<X2(k)');
```

Scilab code Exa 3.3 8 Point DFT

```
//Example 3.3
//Program to Compute the 8-point DFT of the Sequence
    x[n]=[1,1,1,1,1,1,0,0]
clear all;
clc ;
close ;
x = [1,1,1,1,1,1,0,0];
//DFT Computation
X = fft (x , -1);
//Display sequence X[k] in command window
disp(X,"X[k]=");
```

Scilab code Exa 3.4 IDFT of the given Sequence

```
//Example 3.4
//Program to Compute the IDFT of the Sequence X[k
   ]=[5,0,1-j,0,1,0,1+j,0]
clear all;
clc ;
close ;
j=sqrt(-1);
X = [5,0,1-j,0,1,0,1+j,0]
//IDFT Computation
x = fft (X , 1);
//Display sequences x[n]in command window
disp(x,"x[n]=");
```

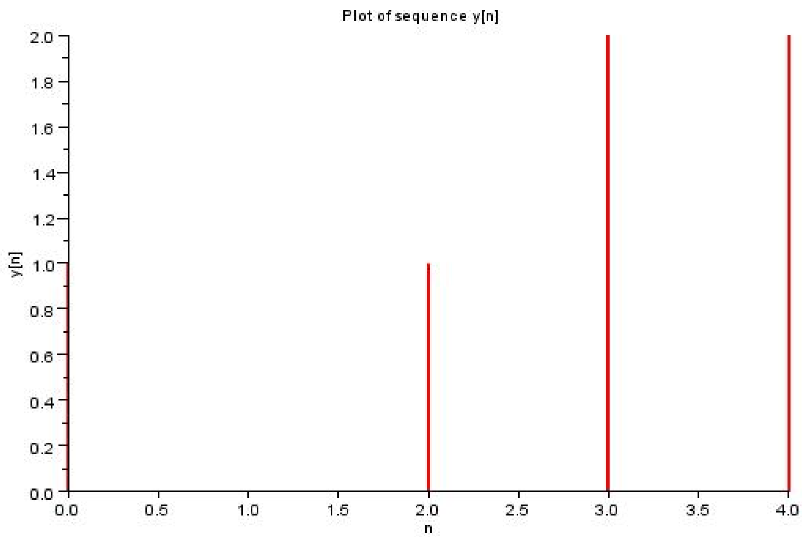

Figure 3.2: Plot the Sequence

Scilab code Exa 3.7 Plot the Sequence

```
//Example 3.7
//Program to Compute circular convolution of
   following sequences
//x[n]=[1,2,2,1,0]
//Y[k]=exp(-j*4*pi*k/5).X[k]
clear all;
clc ;
close ;
x=[1,2,2,1,0];
X=fft(x,-1);
k=0:1:4;
j=sqrt(-1);
pi=22/7;
```

```
H=exp(-j*4*pi*k/5);
Y=H.*X;
//IDFT Computation
y=fft(Y,1);
//Display sequence y[n] in command window
disp(round(y),"y[n]=");
//Plots
n=0:1:4;
a = gca ();
a.y_location ="origin";
a.x_location ="origin";
plot2d3(n,round(y),5);
poly1=a.children(1).children (1);
poly1.thickness=2;
xtitle('Plot of sequence y[n]','n','y[n]');
```

Scilab code Exa 3.9 Remaining Samples

```
//Example 3.9
//Program to remaining samples of the sequence
//X(0)=12,X(1)=-1+j3,X(2)=3+j4,X(3)=1-j5,X(4)=-2+j2,X(5)=6+j3,X(6)=-2-j3,X(7)=10
clear all;
clc ;
close ;
j=sqrt(-1);
z=1;
X(0+z)=12,X(1+z)=-1+j*3,X(2+z)=3+j*4,X(3+z)=1-j*5,X(4+z)=-2+j*2,X(5+z)=6+j*3,X(6+z)=-2-j*3,X(7+z)=10;
for a=9:1:14 do X(a)=conj(X(16-a)), end;
//Display the complete sequence X[k] in command window
disp(X,"X[k]=");
```

Scilab code Exa 3.11 DFT Computation

```
//Example 3.11
//Program to Compute the 8-point DFT of the
    following sequences
//x1[n]=[1,0,0,0,0,1,1,1]
//x2[n]=[0,0,1,1,1,1,0,0]
clear all;
clc ;
close ;
x1=[1,0,0,0,0,1,1,1];
x2=[0,0,1,1,1,1,0,0];
//DFT Computation
X1 = fft (x1 , -1);
X2 = fft (x2 , -1);
//Display sequences X1[k] and X2[k] in command
    window
disp(X1,"X1[k]=");
disp(X2,"X2[k]=");
```

Scilab code Exa 3.13 Circular Convolution

```
//Example 3.13
//Program to Compute circular convolution of
    following sequences
//x1[n]=[1,-1,-2,3,-1]
//x2[n]=[1,2,3]
clear all;
clc ;
close ;
x1=[1,-1,-2,3,-1];
x2=[1,2,3];
```

```
//Loop for zero padding the smaller sequence out of the two
n1=length(x1);
n2=length(x2);
n3=n2-n1;
if (n3>=0) then
 x1=[x1,zeros(1,n3)];
else
 x2=[x2,zeros(1,-n3)];
end
//DFT Computation
X1=fft(x1,-1);
X2=fft(x2,-1);
Y=X1.*X2;
//IDFT Computation
y=fft(Y,1);
//Display sequence y[n] in command window
disp(y,"y[n]=");
```

Scilab code Exa 3.14 Circular Convolution

```
//Example 3.14
//Program to Compute circular convolution of following sequences
//x1[n]=[1,2,2,1]
//x2[n]=[1,2,3,1]
clear all;
clc ;
close ;
x1=[1,2,2,1];
x2=[1,2,3,1];
//DFT Computation
X1=fft(x1,-1);
X2=fft(x2,-1);
Y=X1.*X2;
```

```
//IDFT Computation
y=fft(Y,1);
//Display sequence y[n] in command window
disp(y,"y[n]=");
```

Scilab code Exa 3.15 Determine Sequence x3

```
//Example 3.15
//Program to Compute x3[n] where X3[k]=X1[k].X2[k]
//x1[n]=[1,2,3,4]
//x2[n]=[1,1,2,2]
clear all;
clc ;
close ;
x1=[1,2,3,4];
x2=[1,1,2,2];
//DFT Computation
X1=fft(x1,-1);
X2=fft(x2,-1);
X3=X1.*X2;
//IDFT Computation
x3=fft(X3,1);
//Display sequence x3[n] in command window
disp(x3,"x3[n]=");
```

Scilab code Exa 3.16 Circular Convolution

```
//Example 3.16
//Program to Compute circular convolution of
   following sequences
//x1[n]=[1,1,2,1]
//x2[n]=[1,2,3,4]
clear all;
```

```
clc ;
close ;
x1=[1,1,2,1];
x2=[1,2,3,4];
//DFT Computation
X1=fft(x1,-1);
X2=fft(x2,-1);
X3=X1.*X2;
//IDFT Computation
x3=fft(X3,1);
//Display sequence x3[n] in command window
disp(x3,"x3[n]=");
```

Scilab code Exa 3.17 Circular Convolution

```
//Example 3.17
//Program to Compute y[n] where Y[k]=X1[k].X2[k]
//x1[n]=[0,1,2,3,4]
//x2[n]=[0,1,0,0,0]
clear all;
clc ;
close ;
x1=[0,1,2,3,4];
x2=[0,1,0,0,0];
//DFT Computation
X1=fft(x1,-1);
X2=fft(x2,-1);
Y=X1.*X2;
//IDFT Computation
y=round(fft(Y,1));
//Display sequence y[n] in command window
disp(y,"y[n]=");
```

Scilab code Exa 3.18 Output Response

```
//Example 3.18
//Program to Compute output responce of following
   sequences
//x[n]=[1,2,3,1]
//h[n]=[1,1,1]
//(1)Linear Convolution
//(2)Circular Convolution
//(3)Circular Convolution with zero padding
clear all;
clc ;
close ;
x=[1,2,3,1];
h=[1,1,1];
//(1)Linear Convolution Computation
ylinear=convol (x,h);
//Display Linear Convoluted Sequence y[n] in command
    window
disp(ylinear,"ylinear[n]=");
//(2)Circular Convolution Computation
//Now zero padding in h[n] sequence to make length
   of x[n] and h[n] equal
h1=[h,zeros(1,1)];
//Now Performing Circular Convolution by DFT method
X=fft(x,-1);
H=fft(h1,-1);
Y=X.*H;
ycircular=fft(Y,1);
//Display Circular Convoluted Sequence y[n] in
   command window
disp(ycircular,"ycircular[n]=");
//(3)Circular Convolution Computation with zero
   Padding
x2=[x,zeros(1,2)];
h2=[h,zeros(1,3)];
//Now Performing Circular Convolution by DFT method
X2=fft(x2,-1);
```

```
H2=fft(h2,-1);
Y2=X2.*H2;
ycircularp=fft(Y2,1);
//Display Circular Convoluted Sequence with zero
    Padding y[n] in command window
disp(ycircularp,"ycircularp[n]=");
```

Scilab code Exa 3.20 Output Response

```
//Example 3.20
//Program to Compute Linear Convolution of following
    sequences
//x[n]=[3,-1,0,1,3,2,0,1,2,1]
//h[n]=[1,1,1]
clear all;
clc ;
close ;
x=[3,-1,0,1,3,2,0,1,2,1];
h=[1,1,1];
// Linear Convolution Computation
y=convol (x,h);
//Display Sequence y[n] in command window
disp(y,"y[n]=");
```

Scilab code Exa 3.21 Linear Convolution

```
//Example 3.21
//Program to Compute Linear Convolution of following
    sequences
//x[n]=[1,2,-1,2,3,-2,-3,-1,1,1,2,-1]
//h[n]=[1,2]
clear all;
clc ;
```

```
close ;
x=[1,2,-1,2,3,-2,-3,-1,1,1,2,-1];
h=[1,2];
// Linear Convolution Computation
y=convol (x,h);
//Display Sequence y[n] in command window
disp(y,"y[n]=");
```

Scilab code Exa 3.23.a N Point DFT Computation

```
//Example 3.23 (a)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//N point DFT of delta(n)
clc;
syms n k N;
x=1;
X=symsum(x*exp(-%i*2*%pi*n*k/N),n,0,0);
disp(X,'X(k)=');
```

Scilab code Exa 3.23.b N Point DFT Computation

```
//Example 3.23 (b)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//N point DFT of delta(n-no)
clc;
syms n k N no;
x=1;
X=symsum(x*exp(-%i*2*%pi*n*k/N),n,-no,-no);
disp(X,'X(k)=');
```

Scilab code Exa 3.23.c N Point DFT Computation

```
//Example 3.23 (c)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//N point DFT of a^n
clc;
syms a n k N;
x=a^n;
X=symsum(x*exp(-%i*2*%pi*n*k/N),n,0,N-1);
disp(X,'X(k)=');
```

Scilab code Exa 3.23.d N Point DFT Computation

```
//Example 3.23 (d)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//N point DFT of x(n)=1, 0<=n<=N/2-1
clc;
syms n k N;
x=1;
X=symsum(x*exp(-%i*2*%pi*n*k/N),n,0,(N/2)-1);
disp(X,'X(k)=');
```

Scilab code Exa 3.23.e N Point DFT Computation

```
//Example 3.23 (e)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//N point DFT of x(n)=exp(%i*2*%pi*ko*n/N);
clc;
syms n k N ko;
x=exp(%i*2*%pi*ko*n/N);
X=symsum(x*exp(-%i*2*%pi*n*k/N),n,0,(N/2)-1);
disp(X,'X(k)=');
```

Scilab code Exa 3.23.f N Point DFT Computation

```
//Example 3.23 (f)
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//N point DFT of x(n)=1, for n=even and 0, for n=odd
clc;
syms n k N ;
x=1;//x(2n)=1,for all n
X=symsum(x*exp(-%i*4*%pi*n*k/N),n,0,N/2-1);
disp(X,'X(k)=');
```

Scilab code Exa 3.24 DFT of the Sequence

```
//Example 3.24
//Program to Compute the DFT of the Sequence x[n
  ]=(-1)^n, for N=4
clear all;
clc ;
close ;
N=4;
n=0:1:N-1;
x=(-1)^n;
//DFT Computation
X = fft (x,-1);
//Display Sequence X[k] in command window
disp(X,"X[k]=");
```

Scilab code Exa 3.25 8 Point Circular Convolution

```
//Example 3.25
//Program to Compute the 8-point Circular
   Convolution of the Sequences
//x1[n]=[1,1,1,1,0,0,0,0]
//x2[n]=sin(3*pi*n/8)
clear all;
clc ;
close ;
x1=[1,1,1,1,0,0,0,0];
n=0:1:7;
pi=22/7;
x2=sin(3*pi*n/8);
//DFT Computation
X1=fft (x1,-1);
X2=fft (x2,-1);
//Circular Convolution using DFT
Y=X1.*X2;
//IDFT Computation
y= fft (Y,1);
//Display sequence y[n] in command window
disp(y,"y[n]=");
```

Scilab code Exa 3.26 Linear Convolution using DFT

```
//Example 3.26
//Program to Compute the Linear Convolution of the
   following Sequences
//x[n]=[1,-1,1]
//h[n]=[2,2,1]
clear all;
clc ;
close ;
x=[1,-1,1];
h=[2,2,1];
//Convolution Computation
```

```
y= convol(x,h);
//Display sequence y[n] in command window
disp(y,"y[n]=");
```

Scilab code Exa 3.27.a Circular Convolution Computation

```
//Example 3.27 (a)
//Program to Compute the Convolution of the
   following Sequences
//x1[n]=[1,1,1]
//x2[n]=[2,-1,2]
clear all;
clc ;
close ;
x1=[1,1,1];
x2=[2,-1,2];
//Convolution Computation
X1=fft (x1,-1);
X2=fft (x2,-1);
Y=X1.*X2;
y=fft (Y,1);
//Display Sequence y[n] in command window
disp(y,"y[n]=");
```

Scilab code Exa 3.27.b Circular Convolution Computation

```
//Example 3.27 (b)
//Program to Compute the Convolution of the
   following Sequences
//x1[n]=[1,1,-1,-1,0]
//x2[n]=[1,0,-1,0,1]
clear all;
clc ;
```

```
close ;
x1=[1,1,-1,-1,0];
x2=[1,0,-1,0,1];
//Convolution Computation
X1=fft (x1,-1);
X2=fft (x2,-1);
Y=X1.*X2;
y= fft (Y,1);
//Display Sequence y[n] in command window
disp(y,"y[n]=");
```

Scilab code Exa 3.30 Calculate value of N

```
//Example 3.30
//Program to Calculate N from given data
//fm=5000Hz
//df=50Hz
//t=0.5sec
clear all;
clc ;
close ;
fm=5000 //Hz
df=50   //Hz
t=0.5   //sec
N1=2*fm/df;
N=2;
while N<=N1, N=N*2,end
//Displaying the value of N in command window
disp(N,"N=");
```

Scilab code Exa 3.32 Sketch Sequence

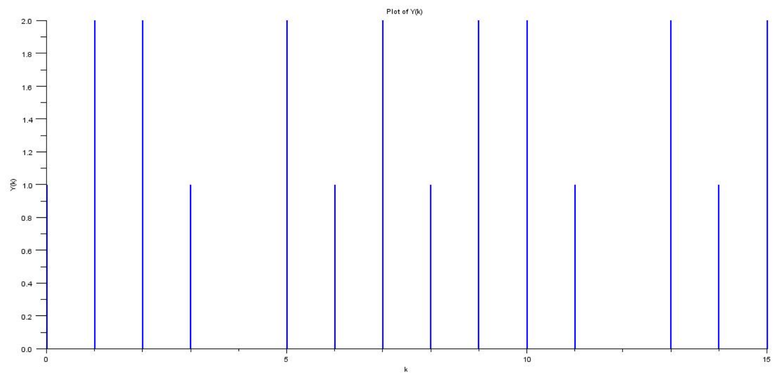

Figure 3.3: Sketch Sequence

```
//Example 3.32
//Program to plot the result of the given sequence
//X[k]=[1,2,2,1,0,2,1,2]
//y[n]=x[n/2] for n=even,0 for n=odd
clear all;
clc ;
close ;
X=[1,2,2,1,0,2,1,2];
x = fft (X , 1);
y=[x(1),0,x(2),0,x(3),0,x(4),0,x(5),0,x(6),0,x(7),0,x(8),0];
Y = fft (y , -1);
//Display sequence Y[k] and in command window
disp(Y,"Y[k]=");
//Plotting the sequence Y[k]
k=0:1:15;
a = gca ();
a.y_location ="origin";
a.x_location ="origin";
plot2d3(k,Y,2);
poly1=a.children(1).children (1);
poly1.thickness=2;
xtitle('Plot of Y(k)','k','Y(k)');
```

Scilab code Exa 3.36 Determine IDFT

```
//Example 3.36
//Program to Compute the IDFT of the following
    Sequence
//X[k]=[12,−1.5+j2.598,−1.5+j0.866,0,−1.5−j0
    .866,−1.5−j2.598]
clear all;
clc ;
close ;
j=sqrt(-1);
X=[12,-1.5+j*2.598,-1.5+j*0.866,0,-1.5-j*0.866,-1.5-
    j*2.598];
//IDFT Computation
x = fft (X , 1);
//Display Sequence x[n] in command window
disp(round(x),"x[n]=");
```

Chapter 4

THE FAST FOURIER TRANSFORM

Scilab code Exa 4.3 Shortest Sequence N Computation

```
//Example 4.3
//Program to calculate shortest sequence N such that
    algorithm B runs //faster than A
clear all;
clc ;
close ;
i=0;
N=32; //Given
//Calculation of Twiddle factor exponents for each
   stage
while 1==1
  i=i+1;
  N=2^i;
  A=N^2;
  B=5*N*log2(N);
  if A>B then break;
    end;
end
disp(N,'SHORTEST SEQUENCE N =');
```

Scilab code Exa 4.4 Twiddle Factor Exponents Calculation

```
//Example 4.4
//Program to calculate Twiddle factor exponents for
   each stage
clear all;
clc ;
close ;
N=32; //Given
//Calculation of Twiddle factor exponents for each
   stage
for m=1:5
  disp(m,'Stage: m =');
  disp(' k =');
  for t=0:(2^(m-1)-1)
    k=N*t/2^m;
    disp(k);
  end
end
```

Scilab code Exa 4.6 DFT using DIT Algorithm

```
//Example 4.6
//Program to find the DFT of a Sequence x[n
   ]=[1,2,3,4,4,3,2,1]
//using DIT Algorithm.
clear all;
clc ;
close ;
x = [1,2,3,4,4,3,2,1];
//FFT Computation
```

```
X = fft (x , -1);
disp(X, 'X(z) = ');
```

Scilab code Exa 4.8 DFT using DIF Algorithm

```
//Example 4.8
//Program to find the DFT of a Sequence x[n
    ]=[1,2,3,4,4,3,2,1]
//using DIF Algorithm.
clear all;
clc ;
close ;
x = [1,2,3,4,4,3,2,1];
//FFT Computation
X = fft (x , -1);
disp(X, 'X(z) = ');
```

Scilab code Exa 4.9 8 Point DFT of the Sequence

```
//Example 4.9
//Program to find the 8-point DFT of a Sequence x[n
    ]=1, 0<=n<=7
//using DIT,DIF Algorithm.
clear all;
clc ;
close ;
x = [1,1,1,1,1,1,1,1];
//FFT Computation
X = fft (x , -1);
disp(X, 'X(z) = ');
```

Scilab code Exa 4.10 4 Point DFT of the Sequence

```
//Example 4.10
//Program to Compute the 4-point DFT of a Sequence x
    [n]=[0,1,2,3]
//using DIT-DIF Algorithm.
clear all;
clc ;
close ;
x = [0,1,2,3];
//FFT Computation
X = fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.11 IDFT of the Sequence using DIT Algorithm

```
//Example 4.11
//Program to Compute the IDFT of a Sequence using
    DIT Algorithm.
//X[k] = [7,-0.707-j0.707,-j,0.707-j0.707,1,0.707+j0
    .707,j,-0.707+j0.707]
clear all;
clc ;
close ;
j=sqrt(-1);
X = [7,-0.707-j*0.707,-j,0.707-j*0.707,1,0.707+j
    *0.707,j,-0.707+j*0.707];
//Inverse FFT Computation
x = fft (X , 1);
disp(x,'x(n) = ');
```

Scilab code Exa 4.12 8 Point DFT of the Sequence

```
//Example 4.12
//Program to Compute the 8-point DFT of a Sequence
//x[n]=[0.5,0.5,0.5,0.5,0,0,0,0] using radix-2 DIT Algorithm.
clear all;
clc ;
close ;
x=[0.5,0.5,0.5,0.5,0,0,0,0];
//FFT Computation
X = fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.13 8 Point DFT of the Sequence

```
//Example 4.13
//Program to Compute the 8-point DFT of a Sequence
//x[n]=[0.5,0.5,0.5,0.5,0,0,0,0] using radix-2 DIF Algorithm.
clear all;
clc ;
close ;
x=[0.5,0.5,0.5,0.5,0,0,0,0];
//FFT Computation
X = fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.14 DFT using DIT Algorithm

```
//Example 4.14
//Program to Compute the 4-point DFT of a Sequence x[n]=[1,-1,1,-1]
//using DIT Algorithm.
clear all;
```

```
clc ;
close ;
x=[1,-1,1,-1];
//FFT Computation
X =fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.15 DFT using DIF Algorithm

```
//Example 4.15
//Program to Compute the 4-point DFT of a Sequence x
    [n]=[1,0,0,1]
//using DIF Algorithm.
clear all;
clc ;
close ;
x=[1,0,0,1];
//FFT Computation
X =fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.16.a 8 Point DFT using DIT FFT

```
//Example 4.16 (a)
//Program to Evaluate and Compare the 8-point DFT of
    the given Sequence
//x1[n]=1, -3<=n<=3 using DIT-FFT Algorithm.
clear all;
clc ;
close ;
x1=[1,1,1,1,0,1,1,1];
//FFT Computation
X1 = fft (x1 , -1);
```

```
disp(X1,'X1(k) = ');
```

Scilab code Exa 4.16.b 8 Point DFT using DIT FFT

```
//Example 4.16 (b)
//Program to Evaluate and Compare the 8-point DFT of
    the given Sequence
//x2[n]=1, 0<=n<=6 using DIT-FFT Algorithm.
clear all;
clc ;
close ;
x2=[1,1,1,1,1,1,1,0];
//FFT Computation
X2 = fft (x2 , -1);
disp(X2,'X2(k) = ');
```

Scilab code Exa 4.17 IDFT using DIF Algorithm

```
//Example 4.17
//Program to find the IDFT of the Sequence using DIF
    Algorithm.
//X[k]= [4,1-j2.414,0,1-j0.414,0,1+j0.414,0,1+j2
   .414]
clear all;
clc ;
close ;
j=sqrt(-1);
X= [4,1-j*2.414,0,1-j*0.414,0,1+j*0.414,0,1+j
   *2.414];
//Inverse FFT Computation
x = fft (X , 1);
disp(x,'x(n) = ');
```

Scilab code Exa 4.18 IDFT using DIT Algorithm

```
//Example 4.18
//Program to find the IDFT of the Sequence X[k]=
    [10,-2+j2,-2,-2-j2]
//using DIT Algorithm.
clear all;
clc ;
close ;
j=sqrt(-1);
X = [10,-2+j*2,-2,-2-j*2];
//Inverse FFT Computation
x = fft (X , 1);
disp(x,'x(n) = ');
```

Scilab code Exa 4.19 FFT Computation of the Sequence

```
//Example 4.19
//Program to Compute the FFT of given Sequence x[n
    ]=[1,0,0,0,0,0,0,0].
clear all;
clc ;
close ;
x = [1,0,0,0,0,0,0,0];
//FFT Computation
X = fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.20 8 Point DFT by Radix 2 DIT FFT

```
//Example 4.20
//Program to Compute the 8-point DFT of given
    Sequence
//x[n]=[2,2,2,2,1,1,1,1] using DIT, radix-2,FFT
    Algorithm.
clear all;
clc ;
close ;
x = [2,2,2,2,1,1,1,1];
//FFT Computation
X = fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.21 DFT using DIT FFT Algorithm

```
//Example 4.21
//Program to Compute the DFT of given Sequence
//x[n]=[1,-1,-1,-1,1,1,1,-1] using DIT-FFT Algorithm
    .
clear all;
clc ;
close ;
x = [1,-1,-1,-1,1,1,1,-1];
//FFT Computation
X = fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.22 Compute X using DIT FFT

```
//Example 4.22
//Program to Compute the DFT of given Sequence
//x[n]=2^n and N=8 using DIT-FFT Algorithm.
clear all;
```

```
clc ;
close ;
N=8;
n=0:1:N-1;
x =2^n;
//FFT Computation
X = fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.23 DFT using DIF FFT Algorithm

```
//Example 4.23
//Program to Compute the DFT of given Sequence
//x[n]=cos(n*pi/2), and N=4 using DIF-FFT Algorithm.
clear all;
clc ;
close ;
N=4;
pi=22/7;
n=0:1:N-1;
x =cos(n*pi/2);
//FFT Computation
X = fft (x , -1);
disp(X,'X(z) = ');
```

Scilab code Exa 4.24 8 Point DFT of the Sequence

```
//Example 4.24
//Program to Compute the 8-point DFT of given
    Sequence
//x[n]=[0,1,2,3,4,5,6,7] using DIF, radix-2,FFT
    Algorithm.
clear all;
```

```
clc ;
close ;
x = [0,1,2,3,4,5,6,7];
//FFT Computation
X = fft (x , -1);
disp(X,'X(z) = ');
```

Chapter 5

INFINITE IMPULSE RESPONSE FILTERS

Scilab code Exa 5.1 Order of the Filter Determination

```
//Example 5.1
//To Find out the order of the filter
clear all;
clc ;
close ;
ap=1;//db
as=30;//db
op=200;//rad/sec
os=600;//rad/sec
N=log(sqrt((10^(0.1*as)-1)/(10^(0.1*ap)-1)))/log(os/
    op);
disp(ceil(N),'Order of the filter , N =');
```

Scilab code Exa 5.2 Order of Low Pass Butterworth Filter

```
//Example 5.2
```

```
//To Find out the order of a Low Pass Butterworth
    Filter
clear all;
clc ;
close ;
ap=3;//db
as=40;//db
fp=500;//Hz
fs=1000;//Hz
op=2*%pi*fp;
os=2*%pi*fs;
N=log(sqrt((10^(0.1*as)-1)/(10^(0.1*ap)-1)))/log(os/
    op);
disp(ceil(N),'Order of the filter , N =');
```

Scilab code Exa 5.4 Analog Butterworth Filter Design

```
//Example 5.4
//To Design an Analog Butterworth Filter
clear all;
clc ;
close ;
ap=2;//db
as=10;//db
op=20;//rad/sec
os=30;//rad/sec
N=log(sqrt((10^(0.1*as)-1)/(10^(0.1*ap)-1)))/log(os/
    op);
disp(ceil(N),'Order of the filter , N =');
s=%s;
HS=1/((s^2+0.76537*s+1)*(s^2+1.8477*s+1));//Transfer
     Function for N=4
oc=op/(10^(0.1*ap)-1)^(1/(2*ceil(N)));
HS1=horner(HS,s/oc);
disp(HS1,'Normalized Transfer Function , H(s) =');
```

Scilab code Exa 5.5 Analog Butterworth Filter Design

```
//Example 5.5
//To Design an Analog Butterworth Filter
clear all;
clc ;
close ;
op=0.2*%pi;
os=0.4*%pi;
e1=0.9;
l1=0.2;
epsilon=sqrt(1/(e1^2)-1);
lambda=sqrt(1/(l1^2)-1);
N=log(lambda/epsilon)/log(os/op);
disp(ceil(N),'Order of the filter , N =');
s=%s;
HS=1/((s^2+0.76537*s+1)*(s^2+1.8477*s+1));//Transfer Function for N=4
oc=op/epsilon^(1/ceil(N));
HS1=horner(HS,s/oc);
disp(HS1,'Normalized Transfer Function , H(s) =');
```

Scilab code Exa 5.6 Order of Chebyshev Filter

```
//Example 5.6
//To Find out the order of the Filter using Chebyshev Approximation
clear all;
clc ;
close ;
ap=3;//db
```

```
as=16;//db
fp=1000;//Hz
fs=2000;//Hz
op=2*%pi*fp;
os=2*%pi*fs;
N=acosh(sqrt((10^(0.1*as)-1)/(10^(0.1*ap)-1)))/acosh
    (os/op);
disp(ceil(N),'Order of the filter , N =');
```

Scilab code Exa 5.7 Chebyshev Filter Design

```
//Example 5.7
//To Design an analog Chebyshev Filter with Given
    Specifications
clear all;
clc ;
close ;
os=2;
op=1;
ap=3;//db
as=16;//db
e1=1/sqrt(2);
l1=0.1;
epsilon=sqrt(1/(e1^2)-1);
lambda=sqrt(1/(l1^2)-1);
N=acosh(lambda/epsilon)/acosh(os/op);
disp(ceil(N),'Order of the filter , N =');
```

Scilab code Exa 5.8 Order of Type 1 Low Pass Chebyshev Filter

```
//Example 5.8
//To Find out the order of the poles of the Type 1
    Lowpass Chebyshev Filter
```

```
clear all;
clc ;
close ;
ap=1;//dB
as=40;//dB
op=1000*%pi;
os=2000*%pi;
N=acosh(sqrt((10^(0.1*as)-1)/(10^(0.1*ap)-1)))/acosh
    (os/op);
disp(ceil(N),'Order of the filter , N =');
```

Scilab code Exa 5.9 Chebyshev Filter Design

```
//Example 5.9
//To Design a Chebyshev Filter with Given
    Specifications
clear all;
clc ;
close ;
ap=2.5;//db
as=30;//db
op=20;//rad/sec
os=50;//rad/sec
N=acosh(sqrt((10^(0.1*as)-1)/(10^(0.1*ap)-1)))/acosh
    (os/op);
disp(ceil(N),'Order of the filter , N =');
```

Scilab code Exa 5.10 HPF Filter Design with given Specifications

```
//Example 5.10
//To Design a H.P.F. with given specifications
clear all;
clc ;
```

```
close ;
ap=3;//db
as=15;//db
op=500;//rad/sec
os=1000;//rad/sec
N=log(sqrt((10^(0.1*as)-1)/(10^(0.1*ap)-1)))/log(os/op);
disp(ceil(N),'Order of the filter , N =');
s=%s;
HS=1/((s+1)*(s^2+s+1));//Transfer Function for N=3
oc=1000//rad/sec
HS1=horner(HS,oc/s);
disp(HS1,'Normalized Transfer Function, H(s) =');
```

Scilab code Exa 5.11 Impulse Invariant Method Filter Design

```
//Example 5.11
//To Design the Filter using Impulse Invarient Method
clear all;
clc ;
close ;
s=%s;
T=1;
HS=(2)/(s^2+3*s+2);
elts=pfss(HS);
disp(elts,'Factorized HS = ');
//The poles comes out to be at -2 and -1
p1=-2;
p2=-1;
z=%z;
HZ=(2/(1-%e^(p2*T)*z^(-1)))-(2/(1-%e^(p1*T)*z^(-1)))
disp(HZ,'HZ = ');
```

Scilab code Exa 5.12 Impulse Invariant Method Filter Design

```
//Example 5.12
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//To Design the Filter using Impulse Invarient
   Method
clear all;
clc ;
close ;
s=%s;
HS=1/(s^2+sqrt(2)*s+1);
pp=ilaplace(HS);
syms n z;
t=1;
X= symsum (pp*(z^(-n)),n ,0, %inf );
disp(X,'Factorized HS = ');
```

Scilab code Exa 5.13 Impulse Invariant Method Filter Design

```
//Example 5.13
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//To Design the 3rd Order Butterworth Filter using
   Impulse Invarient Method
clear all;
clc ;
close ;
s=%s;
HS=1/((s+1)*(s^2+s+1));
pp=ilaplace(HS);//Inverse Laplace
syms n z;
t=1;
X= symsum (pp*(z^(-n)),n ,0, %inf );//Z Transform
```

```
disp(X,'H(z)= ');
```

Scilab code Exa 5.15 Impulse Invariant Method Filter Design

```
//Example 5.15
//To Design the Filter using Impulse Invarient
   Method
clear all;
clc ;
close ;
s=%s;
T=0.2;
HS=10/(s^2+7*s+10);
elts=pfss(HS);
disp(elts,'Factorized HS = ');
//The poles comes out to be at -5 and -2
p1=-5;
p2=-2;
z=%z;
HZ=T*((-3.33/(1-%e^(p1*T)*z^(-1)))+(3.33/(1-%e^(p2*T
   )*z^(-1))))
disp(HZ,'HZ = ');
```

Scilab code Exa 5.16 Bilinear Transformation Method Filter Design

```
//Example 5.16
//To Find out Bilinear Transformation of HS=2/((s+1)
   *(s+2))
clear all;
clc ;
close ;
s=%s;
z=%z;
```

```
HS=2/((s+1)*(s+2));
T=1;
HZ=horner(HS,(2/T)*(z-1)/(z+1));
disp(HZ,'H(z) =');
```

Scilab code Exa 5.17 HPF Design using Bilinear Transform

```
//Example 5.17
//To Design an H.P.F. monotonic in passband using
   Bilinear Transform
clear all;
clc ;
close ;
ap=3;//db
as=10;//db
fp=1000;//Hz
fs=350;//Hz
f=5000;
T=1/f;
wp=2*%pi*fp;
ws=2*%pi*fs;
op=2/T*tan(wp*T/2);
os=2/T*tan(ws*T/2);
N=log(sqrt((10^(0.1*as)-1)/(10^(0.1*ap)-1)))/log(op/
   os);
disp(ceil(N),'Order of the filter , N =');
s=%s;
HS=1/(s+1)//Transfer Function for N=1
oc=op//rad/sec
HS1=horner(HS,oc/s);
disp(HS1,'Normalized Transfer Function , H(s) =');
z=%z;
HZ=horner(HS,(2/T)*(z-1)/(z+1));
disp(HZ,'H(z) =');
```

Scilab code Exa 5.18 Bilinear Transformation Method Filter Design

```
//Example 5.18
//To Find out Bilinear Transformation of H(s)=(s
    ^2+4.525)/(s^2+0.692*s+0.504)
clear all;
clc ;
close ;
s=%s;
z=%z;
HS=(s^2+4.525)/(s^2+0.692*s+0.504);
T=1;
HZ=horner(HS,(2/T)*(z-1)/(z+1));
disp(HZ,'H(z) =');
```

Scilab code Exa 5.19 Single Pole LPF into BPF Conversion

```
//Example 5.19
//To Convert a single Pole LPF into BPF
clear all;
clc ;
close ;
s=%s;
z=%z;
HZ=(0.5*(1+z^(-1)))/(1-0.302*z^(-2));
T=1;
wu=3*%pi/4;
wl=%pi/4;
wp=%pi/6;
k=tan(wp/2)/tan((wu-wl)/2);
a=cos((wu+wl)/2)/cos((wu-wl)/2);
```

```
transf=-((((k-1)/(k+1))*(z^(-2)))-((2*a*k/(k+1))*(z
    ^(-1)))+1)/(z^(-2)-(2*a*k/(1+k)*z^(-1))+((k-1)/(k
    +1)));
HZ1=horner(HZ,transf);
disp(HZ1,'H(z) of B.P.F =');
```

Scilab code Exa 5.29 Pole Zero IIR Filter into Lattice Ladder Structure

```
//Example 5.29
//Program to convert given IIR pole-zero Filter into
    Lattice Ladder Structure.
clear all;
clc ;
close ;
U=1;            //Zero Adjust
a(3+U,0+U)=1;
a(3+U,1+U)=13/24;
a(3+U,2+U)=5/8;
a(3+U,3+U)=1/3;
a(2+U,0+U)=1; //a(m,0)=1
a(2+U,3+U)=1/3;
m=3,k=1;
a(m-1+U,k+U)=(a(m+U,k+U)-a(m+U,m+U)*a(m+U,m-k+U))
    /(1-a(m+U,m+U)*a(m+U,m+U));
m=3,k=2;
a(m-1+U,k+U)=(a(m+U,k+U)-a(m+U,m+U)*a(m+U,m-k+U))
    /(1-a(m+U,m+U)*a(m+U,m+U));
m=2,k=1;
a(m-1+U,k+U)=(a(m+U,k+U)-a(m+U,m+U)*a(m+U,m-k+U))
    /(1-a(m+U,m+U)*a(m+U,m+U));
disp('LATTICE COEFFICIENTS');
disp(a(1+U,1+U),'k1');
disp(a(2+U,2+U),'k2');
disp(a(3+U,3+U),'k3');
b0=1;
```

```
b1=2;
b2=2;
b3=1;
c3=b3;
c2=b2-c3*a(3+U,1+U);
c1=b1-(c2*a(2+U,1+U)+c3*a(3+U,2+U));
c0=b0-(c1*a(1+U,1+U)+c2*a(2+U,2+U)+c3*a(3+U,3+U));
disp('LADDER COEFFICIENTS');
disp(c0,'c0 =');
disp(c1,'c1 =');
disp(c2,'c2 =');
disp(c3,'c3 =');
```

Chapter 6

FINITE IMPULSE RESPONSE FILTERS

Scilab code Exa 6.1 Group Delay and Phase Delay

```
//Example 6.1
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Program to Calculate Group Delay and Phase Delay
//y(n)=0.25x(n)+x(n-1)+0.25x(n-2)
clear all;
clc ;
close ;
//w=poly(0,"w");
syms w;
theeta=-w;
gd= -diff (theeta,w); //Group Delay
pd=-theeta/w; //Phase Delay
disp(gd,'GROUP DELAY =');
disp(pd,'PHASE DELAY =');
```

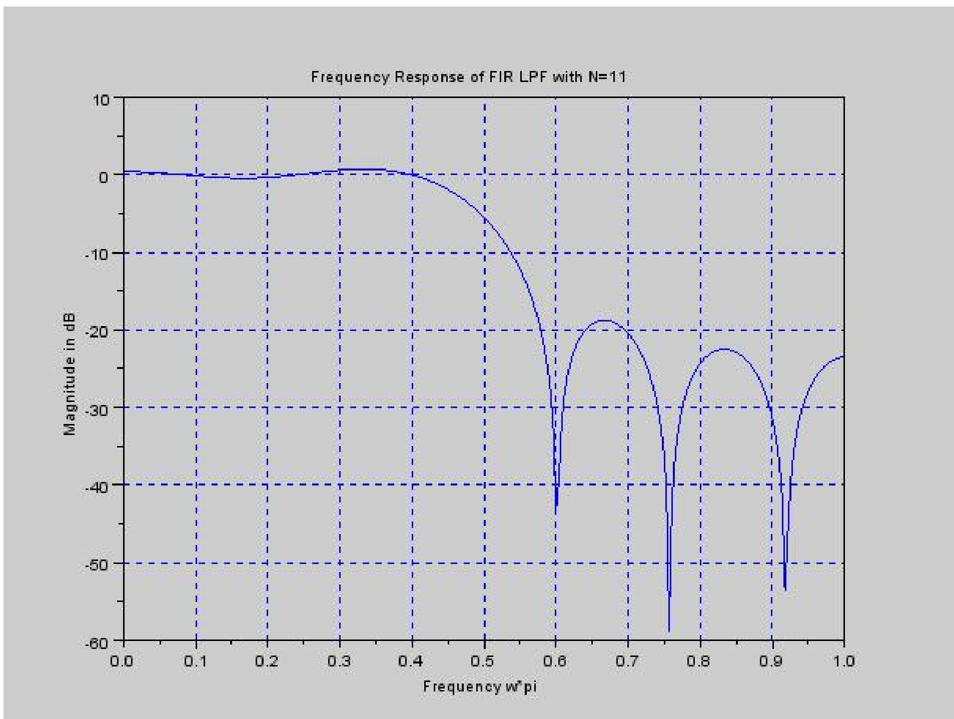

Figure 6.1: LPF Magnitude Response

Scilab code Exa 6.5 LPF Magnitude Response

```
//Example 6.5
//Program to Plot Magnitude Responce of ideal L.P.F.
    with wc=0.5*pi
//N=11
clear all;
clc ;
close ;
N=11;
U=6;
for n=-5+U:1:5+U
if n==6
hd(n)=0.5;
else
hd(n)=(sin(%pi*(n-U)/2))/(%pi*(n-U));
end
end
[hzm ,fr ]= frmag (hd ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure;
plot (2*fr , hzm_dB );
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of FIR LPF with N=11');
xgrid (2);
```

Scilab code Exa 6.6 HPF Magnitude Response

```
//Example 6.6
//Program to Plot Magnitude Responce of ideal H.P.F.
    with wc=0.25*pi
```

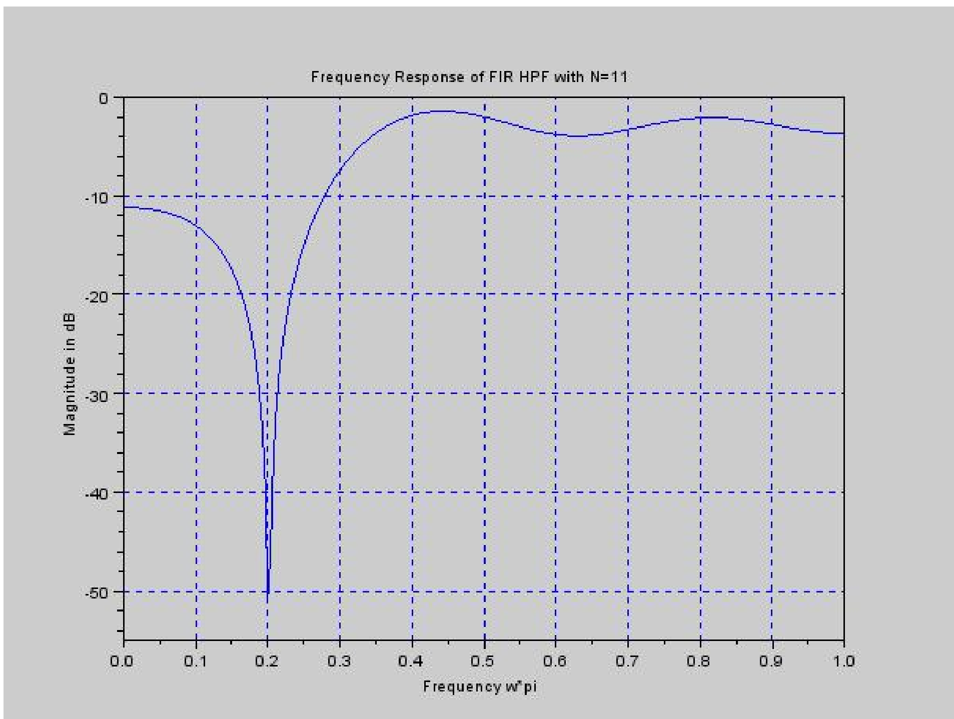

Figure 6.2: HPF Magnitude Response

```
//N=11
clear all;
clc ;
close ;
N=11;
U=6;
for n=-5+U:1:5+U
if n==6
hd(n)=0.5;
else
hd(n)=(sin(%pi*(n-U))-sin(%pi*(n-U)/4))/(%pi*(n-U));
end
end
[hzm ,fr ]= frmag (hd ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of FIR HPF with N=11');
xgrid (2);
```

Scilab code Exa 6.7 BPF Magnitude Response

```
//Example 6.7
//Program to Plot Magnitude Responce of ideal B.P.F.
    with
//wc1=0.25*pi and wc2=0.75*pi
//N=11
clear all;
clc ;
close ;
```

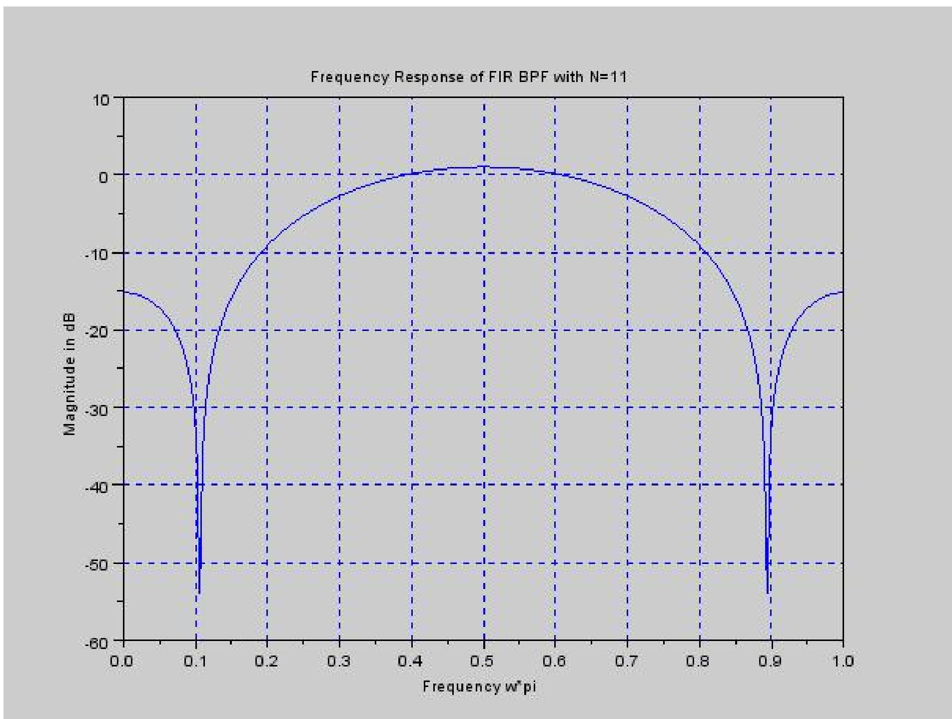

Figure 6.3: BPF Magnitude Response

```
N=11;
U=6;
for n=-5+U:1:5+U
if n==6
hd(n)=0.5;
else
hd(n)=(sin(%pi*3*(n-U)/4)-sin(%pi*(n-U)/4))/(%pi*(n-
   U));
end
end
[hzm ,fr ]= frmag (hd ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of FIR BPF with N=11');
xgrid (2);
```

Scilab code Exa 6.8 BRF Magnitude Response

```
//Example 6.8
//Program to Plot Magnitude Responce of ideal B.R.F.
    with
//wc1=0.33*pi and wc2=0.67*pi
//N=11
clear all;
clc ;
close ;
N=11;
U=6;
for n=-5+U:1:5+U
```

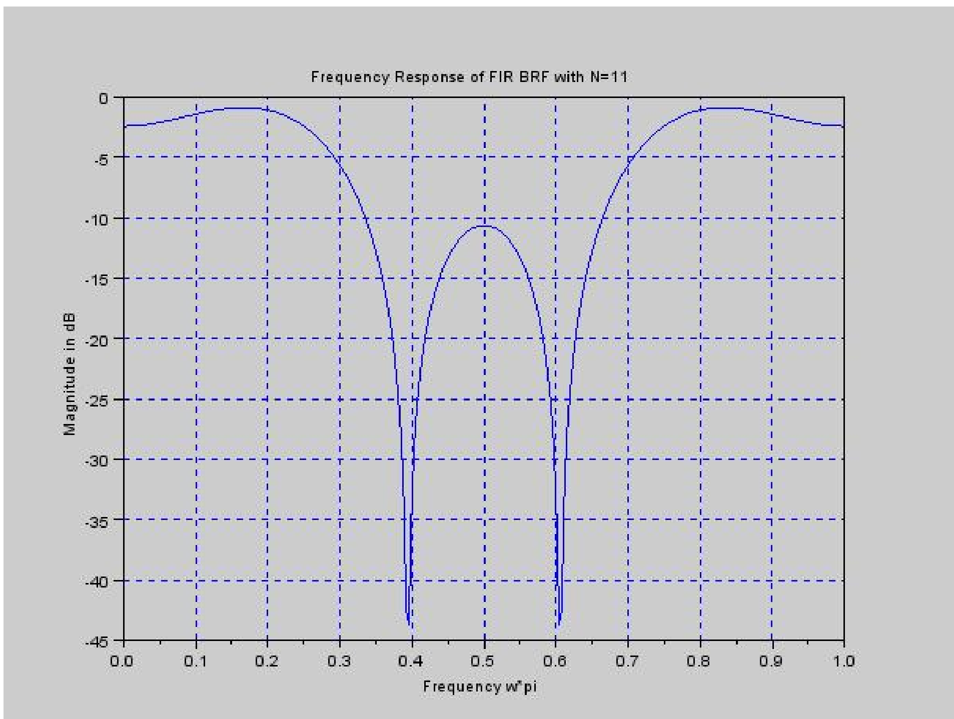

Figure 6.4: BRF Magnitude Response

```
if n==6
hd(n)=0.5;
else
hd(n)=(sin(%pi*(n-U))+sin(%pi*(n-U)/3)-sin(%pi*2*(n-
   U)/3))/(%pi*(n-U));
end
end
[hzm ,fr ]= frmag (hd ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of FIR BRF with N=11');
xgrid (2);
```

Scilab code Exa 6.9.a HPF Magnitude Response using Hanning Window

```
//Example 6.9a
//Program to Plot Magnitude Responce of ideal H.P.F.
//using Hanning Window
//wc1=0.25*pi
//N=11
clear all;
clc ;
close ;
N=11;
U=6;
h_hann=window('hn',N);
for n=-5+U:1:5+U
if n==6
hd(n)=0.75;
```

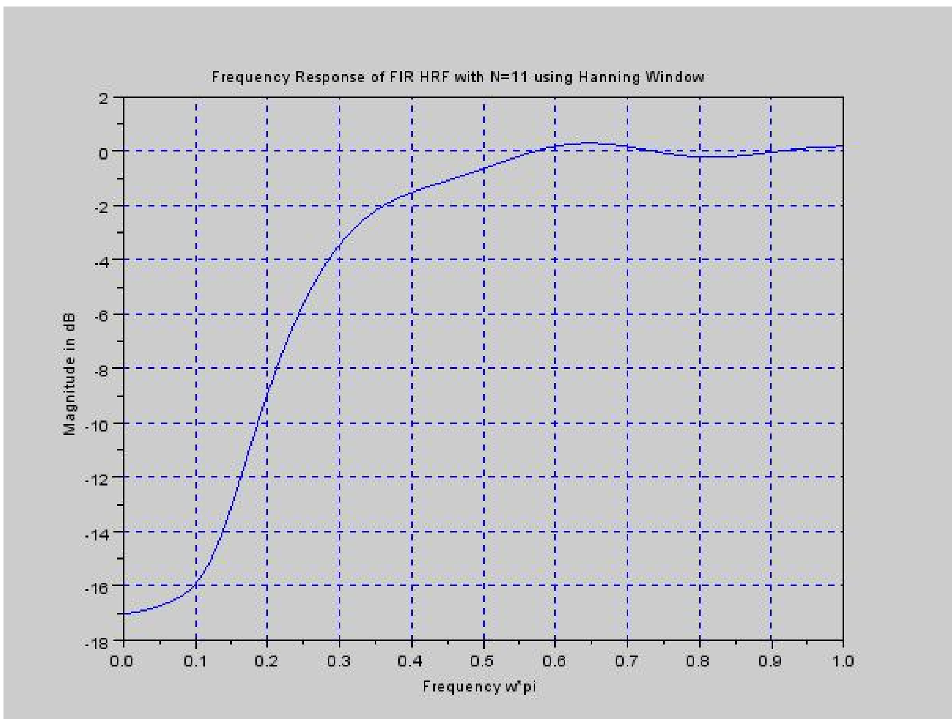

Figure 6.5: HPF Magnitude Response using Hanning Window

```
else
hd(n)=(sin(%pi*(n-U))-sin(%pi*(n-U)/4))/(%pi*(n-U));
end
h(n)=h_hann(n)*hd(n);
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of FIR HRF with N=11
   using Hanning Window');
xgrid (2);
```

Scilab code Exa 6.9.b HPF Magnitude Response using Hamming Window

```
//Example 6.9b
//Program to Plot Magnitude Responce of ideal H.P.F.
//using Hamming Window
//wc1=0.25*pi
//N=11
clear all;
clc ;
close ;
N=11;
U=6;
h_hamm=window('hm',N);
for n=-5+U:1:5+U
if n==6
hd(n)=0.75;
```

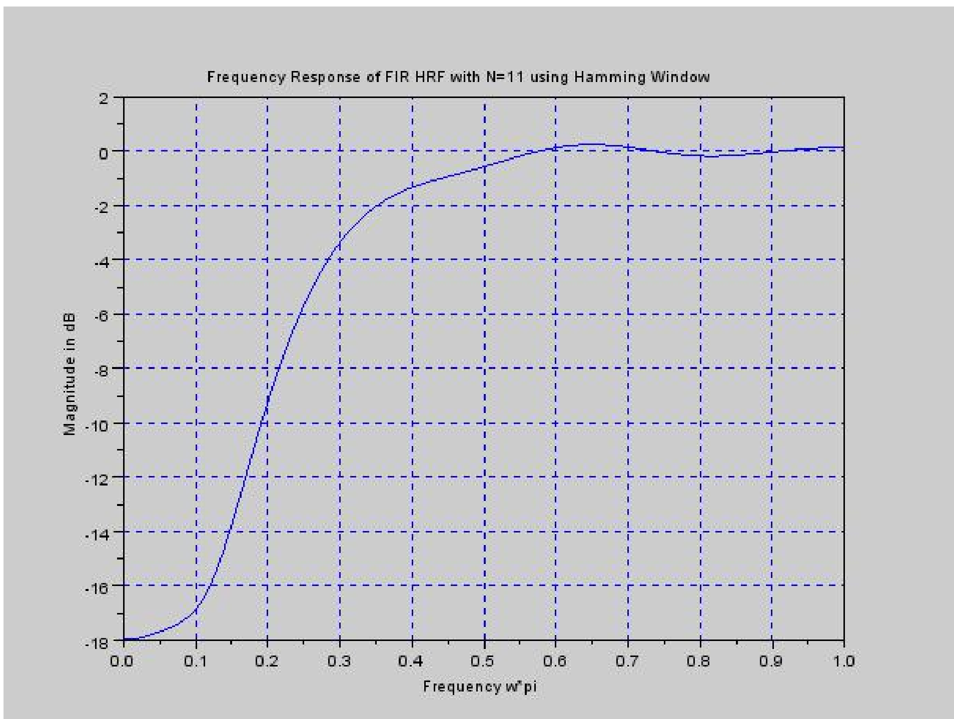

Figure 6.6: HPF Magnitude Response using Hamming Window

```
else
hd(n)=(sin(%pi*(n-U))-sin(%pi*(n-U)/4))/(%pi*(n-U));
end
h(n)=h_hamm(n)*hd(n);
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of FIR HRF with N=11
   using Hamming Window');
xgrid (2);
```

Scilab code Exa 6.10 Hanning Window Filter Design

```
//Example 6.10
//Program to Plot Magnitude Responce of given L.P.F.
    with specifications:
//N=7,w=pi/4
//Using Hanning Window
clear all;
clc ;
close ;
N=7;
alpha=3;
U=1;
h_hann=window('hn',N);
for n=0+U:1:6+U
if n==4
hd(n)=0.25;
```

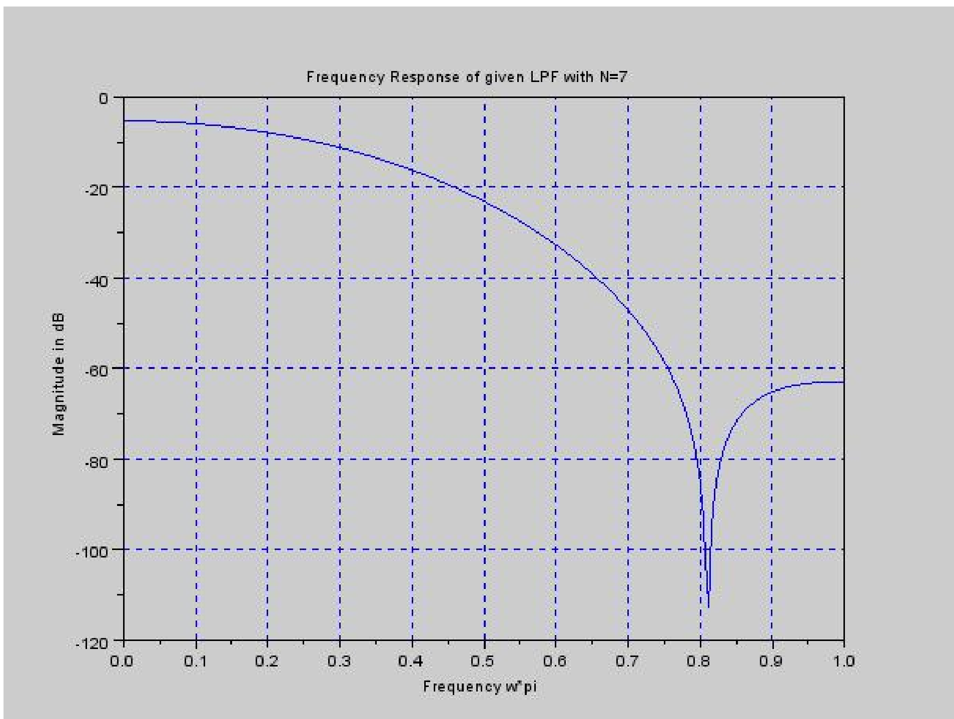

Figure 6.7: Hanning Window Filter Design

```
else
hd(n)=(sin(%pi*(n-U-alpha)/4))/(%pi*(n-U-alpha));
end
h(n)=hd(n)*h_hann(n);
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of given LPF with N=7');
xgrid (2);
```

Scilab code Exa 6.11 LPF Filter Design using Kaiser Window

```
//Example 6.11
//Program to Plot Magnitude Responce of given L.P.F.
    with specifications:
//wp=20rad/sec, ws=30rad/sec, wsf=100rad/sec
//as=44.0dB, ap=0.1dB
//Using Kaiser Window
clear all;
clc ;
close ;
wsf=100//rad/sec
ws=30;//rad/sec
wp=20;//rad/sec
as=44.0//dB
ap=0.1//dB
B=ws-wp;
wc=0.5*(ws+wp);
```

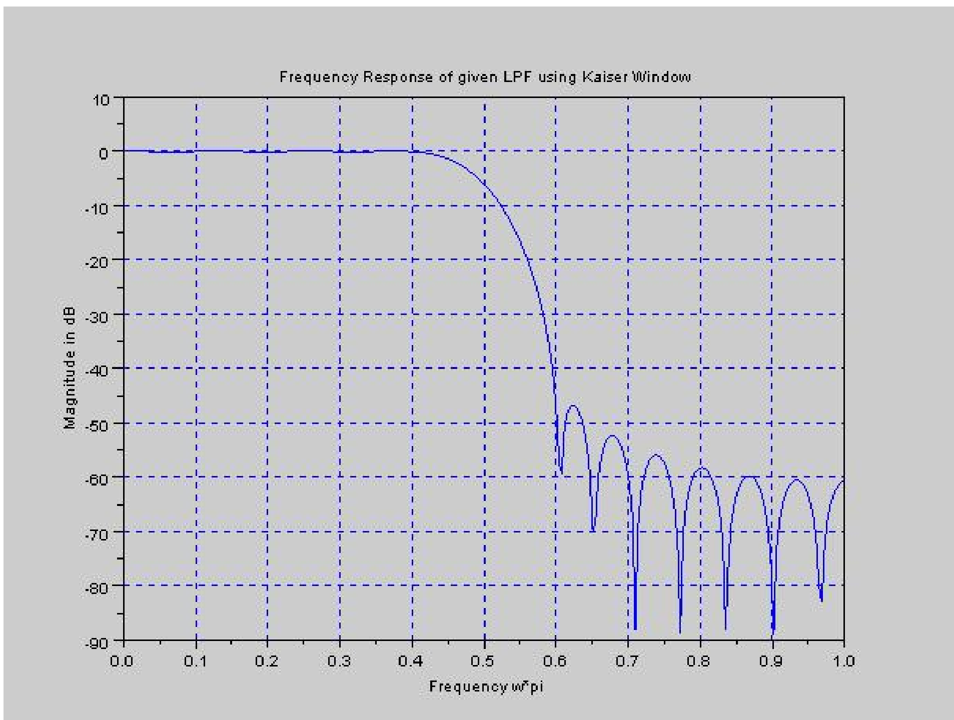

Figure 6.8: LPF Filter Design using Kaiser Window

```
wc1=wc*2*%pi/wsf;
delta1=10^(-0.05*as);
delta2=(10^(0.05*as)-1)/(10^(0.05*as)+1);
delta=min(delta1,delta2);
alphas=-20*log10(delta);
alpha=0.5842*(alphas-21)^0.4+0.07886*(alphas-21)
D=(alphas-7.95)/14.36;
N1=wsf*D/B+1;
N=ceil(N1);
U=ceil(N/2);
win_l=window('kr',N,alpha);
for n=-floor(N/2)+U:1:floor(N/2)+U
if n==ceil(N/2);
hd(n)=0.5;
else
hd(n)=(sin(%pi*(n-U)/2))/(%pi*(n-U));
end
h(n)=hd(n)*win_l(n);
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of given LPF using Kaiser
    Window');
xgrid (2);
disp(h,"Filter Coefficients,h(n)=");
```

Scilab code Exa 6.12 BPF Filter Design using Kaiser Window

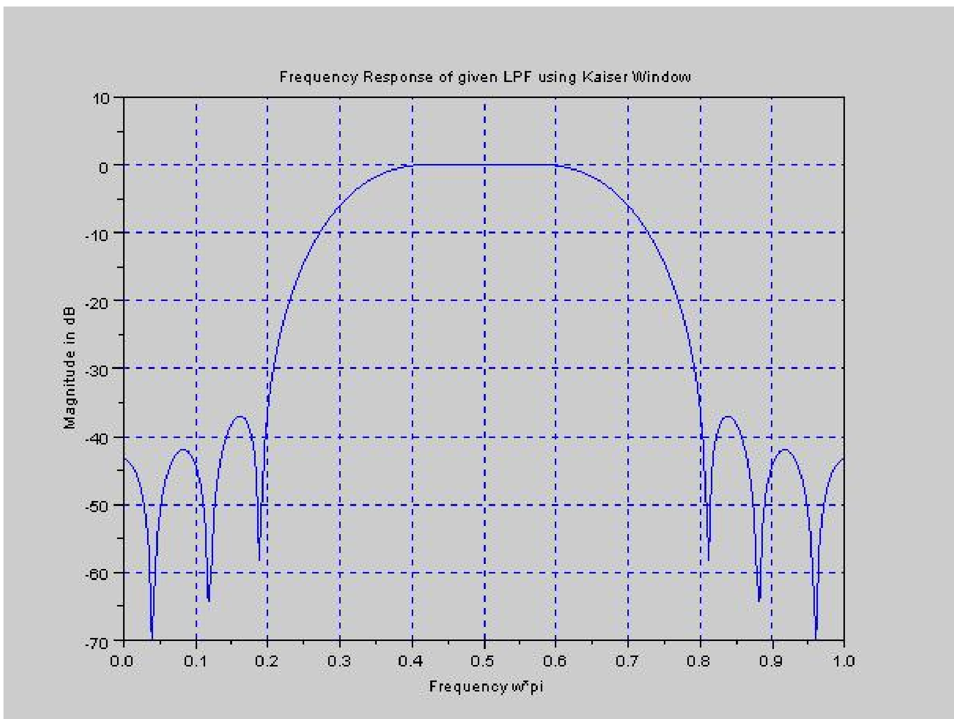

Figure 6.9: BPF Filter Design using Kaiser Window

```
//Example 6.12
//Program to Plot Magnitude Responce of given B.P.F.
    with specifications:
//wp1=40pi rad/sec, wp2=60pi rad/sec
//ws1=20pi rad/sec, ws2=80pi rad/sec
//as=30dB, ap=0.5dB
//F=100 Hz
//Using Kaiser Window
clear all;
clc ;
close ;
wsf=200*%pi;//rad/sec
ws1=20*%pi;//rad/sec
ws2=80*%pi;//rad/sec
wp1=40*%pi;//rad/sec
wp2=60*%pi;//rad/sec
as=30//dB
ap=0.5//dB
B=min(wp1-ws1,ws2-wp2);
wc1=wp1-B/2;
wc2=wp2+B/2;
wc1=wc1*2*%pi/wsf;
wc2=wc2*2*%pi/wsf;
delta1=10^(-0.05*as);
delta2=(10^(0.05*as)-1)/(10^(0.05*as)+1);
delta=min(delta1,delta2);
alphas=-20*log10(delta);
alpha=0.5842*(alphas-21)^0.4+0.07886*(alphas-21)
D=(alphas-7.95)/14.36;
N1=wsf*D/B+1;
N=ceil(N1);
U=ceil(N/2);
win_l=window('kr',N,alpha);
for n=-floor(N/2)+U:1:floor(N/2)+U
if n==ceil(N/2);
hd(n)=0.4;
else
hd(n)=(sin(0.7*%pi*(n-U))-sin(0.3*%pi*(n-U)))/(%pi*(
```

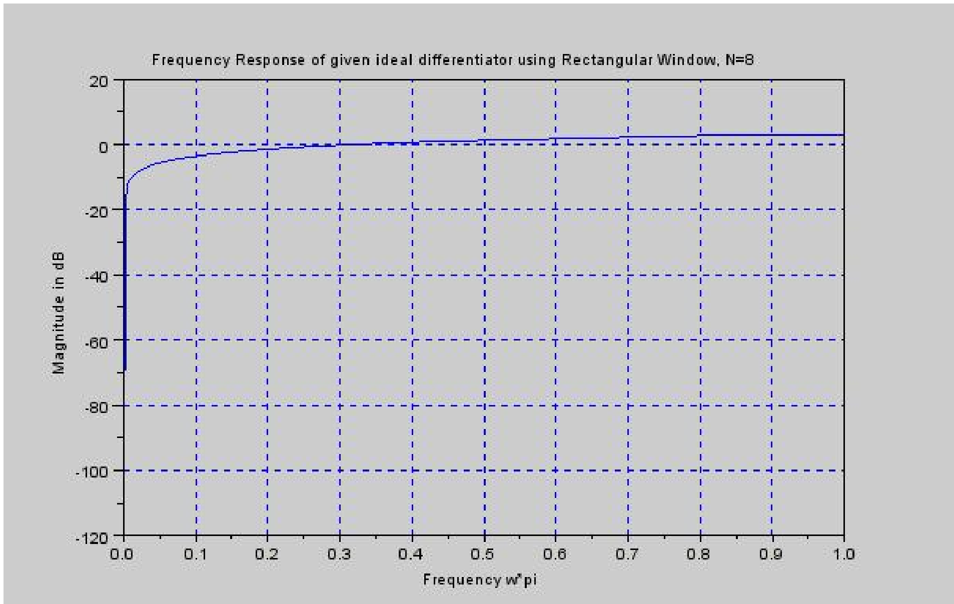

Figure 6.10: Digital Differentiator using Rectangular Window

```
    n-U));
end
h(n)=hd(n)*win_l(n);
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of given LPF using Kaiser
    Window');
xgrid (2);
disp(h,"Filter Coefficients,h(n)=");
```

Scilab code Exa 6.13.a Digital Differentiator using Rectangular Window

```
//Example 6.13a
//Program to Plot Magnitude Responce of ideal
    differentiator with specifications:
//N=8,w=pi
//using Rectangular window
clear all;
clc ;
close ;
N=8;
alpha=7/2;
U=1;
h_Rect=window('re',N);
for n=0+U:1:7+U
hd(n)=-(sin(%pi*(n-U-alpha)))/(%pi*(n-U-alpha)*(n-U-
    alpha));
h(n)=hd(n)*h_Rect(n);
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of given ideal
    differentiator using Rectangular Window, N=8');
xgrid (2)
```

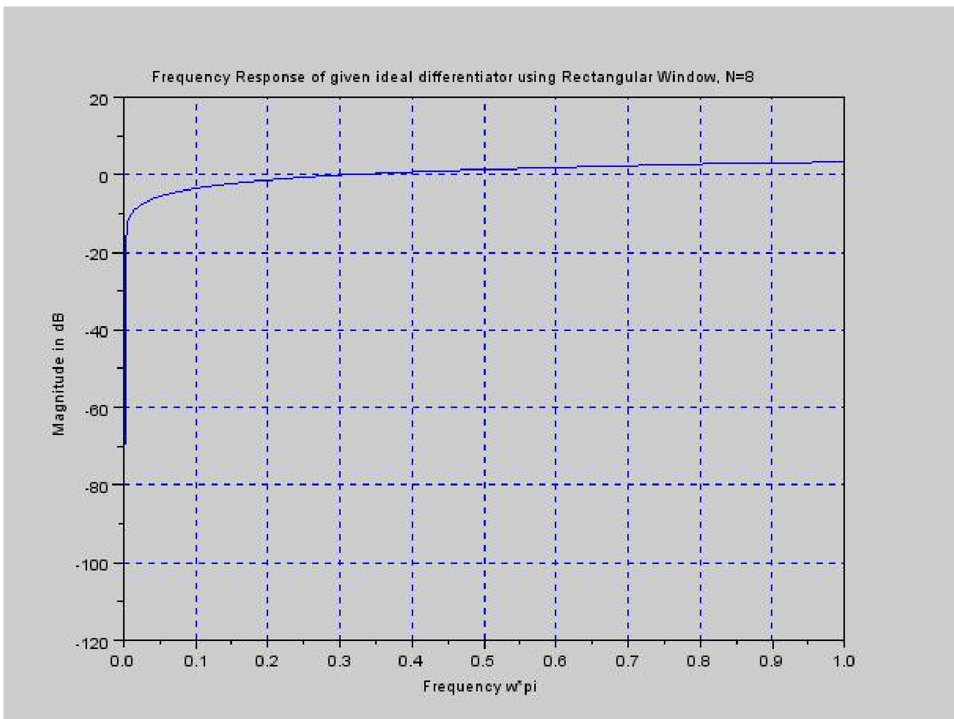

Figure 6.11: Digital Differentiator using Hamming Window

Scilab code Exa 6.13.b Digital Differentiator using Hamming Window

```
//Example 6.13b
//Program to Plot Magnitude Responce of ideal
   differentiator with specifications:
//N=8,w=pi
//using Hamming window
clear all;
clc ;
close ;
N=8;
alpha=7/2;
U=1;        //Zero Adjust
h_hamm=window('hm',N);
for n=0+U:1:7+U
hd(n)=-(sin(%pi*(n-U-alpha)))/(%pi*(n-U-alpha)*(n-U-
   alpha));
h(n)=hd(n)*h_hamm(n);
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of given ideal
   differentiator using Hamming Window, N=8');
xgrid (2)
```

Scilab code Exa 6.14.a Hilbert Transformer using Rectangular Window

```
//Example 6.14a
```

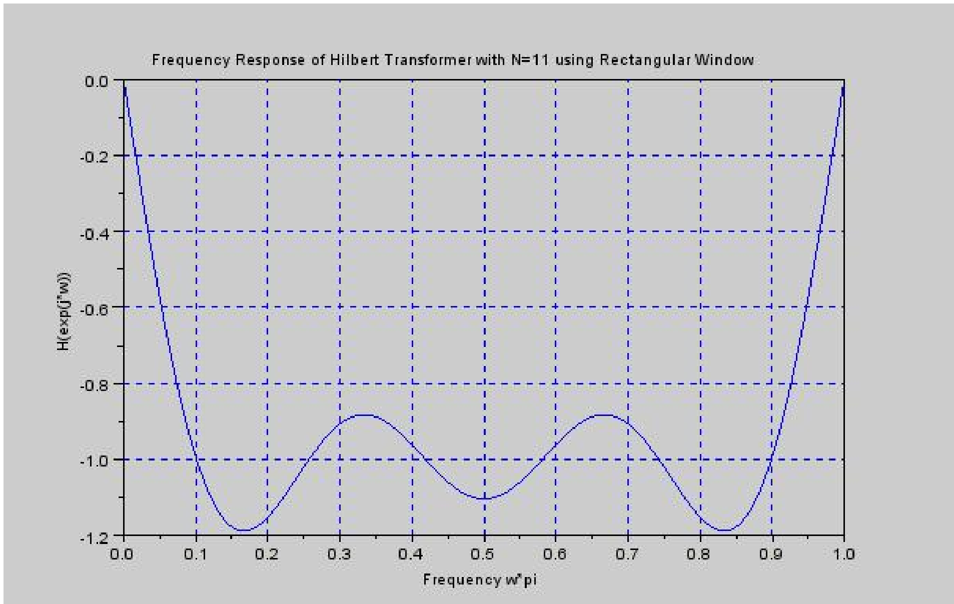

Figure 6.12: Hilbert Transformer using Rectangular Window

```
//Program to Plot Magnitude Responce of ideal
  Hilbert Transformer
//using Rectangular Window
//N=11
clear all;
clc ;
close ;
N=11;
U=6;
h_Rect=window('re',N);
for n=-5+U:1:5+U
if n==6
hd(n)=0;
else
hd(n)=(1-cos(%pi*(n-U)))/(%pi*(n-U));
end
h(n)=hd(n)*h_Rect(n);
end
[hzm ,fr]= frmag (h,256) ;
```

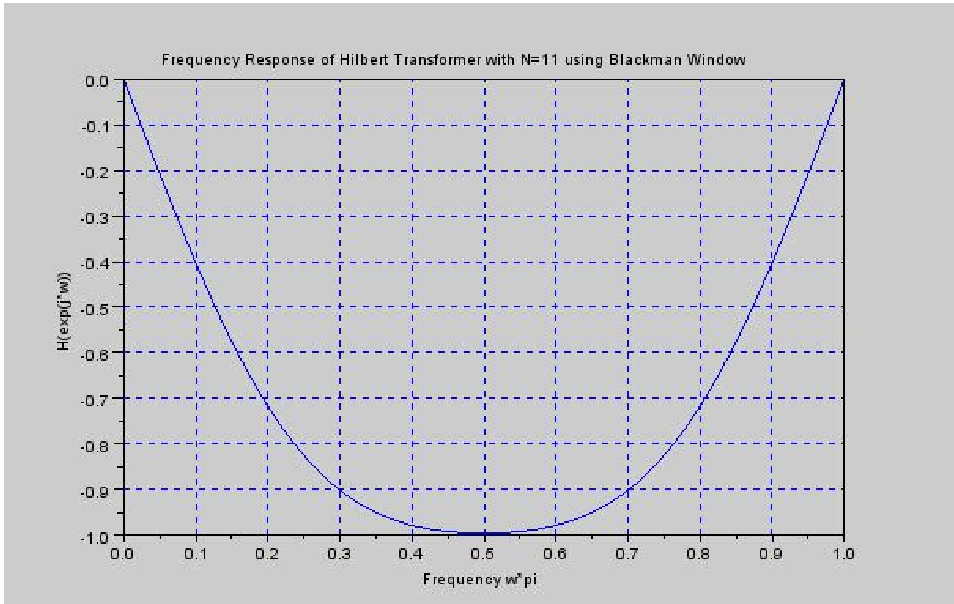

Figure 6.13: Hilbert Transformer using Blackman Window

```
figure
plot (2*fr ,-hzm);
a = gca ();
xlabel ('Frequency w*pi');
ylabel ('H(exp(j*w))');
title ('Frequency Response of Hilbert Transformer
    with N=11 using Rectangular Window');
xgrid (2);
```

Scilab code Exa 6.14.b Hilbert Transformer using Blackman Window

```
//Example 6.14b
//Program to Plot Magnitude Responce of ideal
    Hilbert Transformer
```

```
//using Blackman Window
//N=11
clear all;
clc ;
close ;
N=11;
U=6;
for n=-5+U:1:5+U
h_balckmann(n) = 0.42+0.5*cos(2*%pi*(n-U)/(N-1))
    +0.08*cos(4*%pi*(n-U)/(N-1));
if n==6
hd(n)=0;
else
hd(n)=(1-cos(%pi*(n-U)))/(%pi*(n-U));
end
h(n)=hd(n)*h_balckmann(n);
end
[hzm ,fr]= frmag (h,256) ;
figure
plot (2*fr ,-hzm);
a = gca ();
xlabel ('Frequency w*pi');
ylabel ('H(exp(j*w))');
title ('Frequency Response of Hilbert Transformer
    with N=11 using Blackman Window');
xgrid (2);
```

Scilab code Exa 6.15 Filter Coefficients obtained by Sampling

```
//Example 6.15
//Program to determine filter coefficients obtained
    by sampling:
//N=7,w=pi/2
clear all;
clc ;
```

```
close ;
N=7;
U=1;        //Zero Adjust
for n=0+U:1:N-1+U
h(n)=(1+2*cos(2*%pi*(n-U-3)/7))/N
end
disp(h,"Filter Coefficients,h(n)=")
```

Scilab code Exa 6.16 Coefficients of Linear phase FIR Filter

```
//Example 6.16
//Program to determine filter coefficients obtained by sampling:
//N=15
clear all;
clc ;
close ;
N=15;
U=1;        //Zero Adjust
for n=0:1:N-1
h(n+U)=(1+2*cos(2*%pi*(7-n)/N)+2*cos(4*%pi*(7-n)/N)+2*cos(6*%pi*(7-n)/N))/N;
end
disp(h,"Filter Coefficients,h(n)=");
```

Scilab code Exa 6.17 BPF Filter Design using Sampling Method

```
//Example 6.17
//Program to design bandpass filter with following specifications:
//N=7, fc1=1000Hz, fc2=3000Hz, F=8000Hz
clear all;
clc ;
```

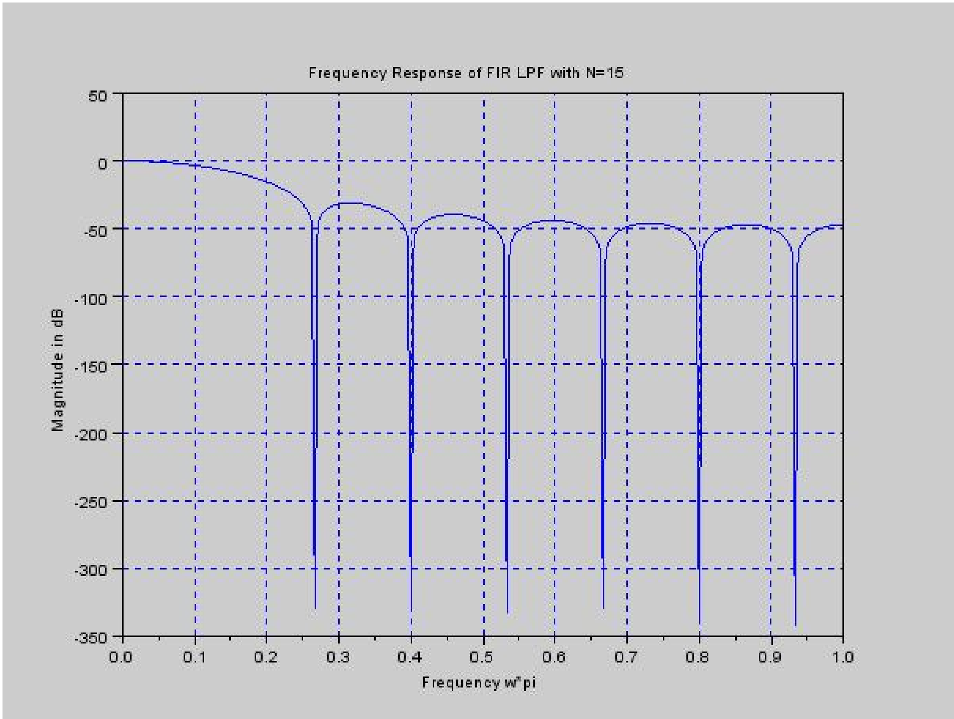

Figure 6.14: Frequency Sampling Method FIR LPF Filter

```
close ;
N=7;
U=1;          //Zero Adjust
for n=0:1:N-1
h(n+U)=2*(cos(2*%pi*(3-n)/N)+cos(4*%pi*(3-n)/N))/N;
end
disp(h,"Filter Coefficients ,h(n)=");
```

Scilab code Exa 6.18.a Frequency Sampling Method FIR LPF Filter

```
//Example 6.18a
```

```
//Program to design L.P.F. filter with following
    specifications:
//N=15, wc=pi/4
clear all;
clc ;
close ;
N=15;
U=1;
for n=0+U:1:N-1+U
h(n)=(1+cos(2*%pi*(7-n)/N))/N;
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure;
plot (2*fr , hzm_dB );
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of FIR LPF with N=15');
xgrid (2)
```

Scilab code Exa 6.18.b Frequency Sampling Method FIR LPF Filter

```
//Example 6.18b
//Program to design L.P.F. filter with following
    specifications:
//N=15, wc=pi/4
clear all;
clc ;
close ;
N=15;
U=1;
for n=0+U:1:N-1+U
```

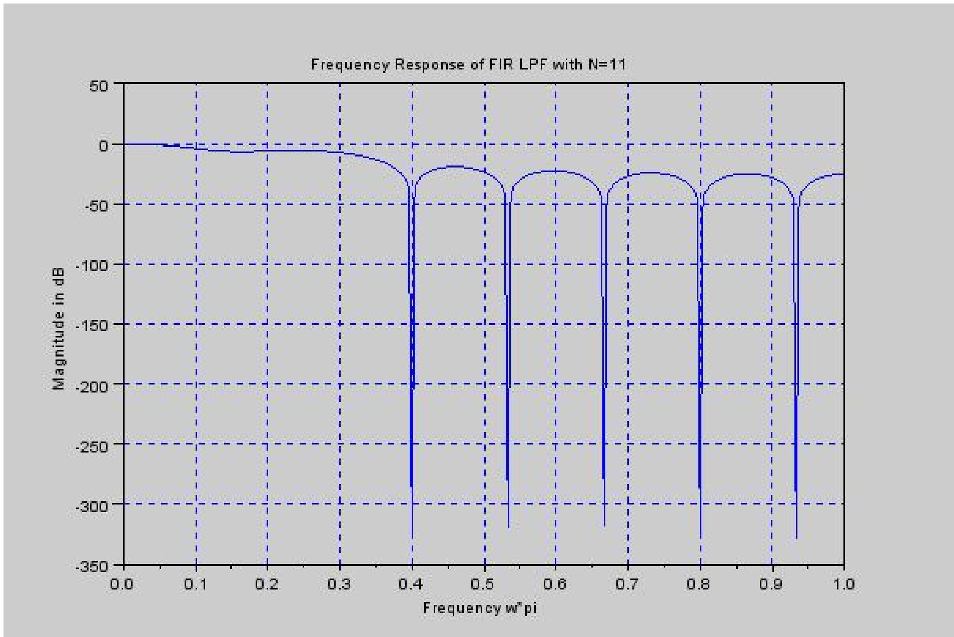

Figure 6.15: Frequency Sampling Method FIR LPF Filter

```
h(n)=(1+cos(2*%pi*(7-n)/N)+cos(4*%pi*(7-n)/N))/N;
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure;
plot (2*fr , hzm_dB );
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of FIR LPF with N=11');
xgrid (2)
```

Scilab code Exa 6.19 Filter Coefficients Determination

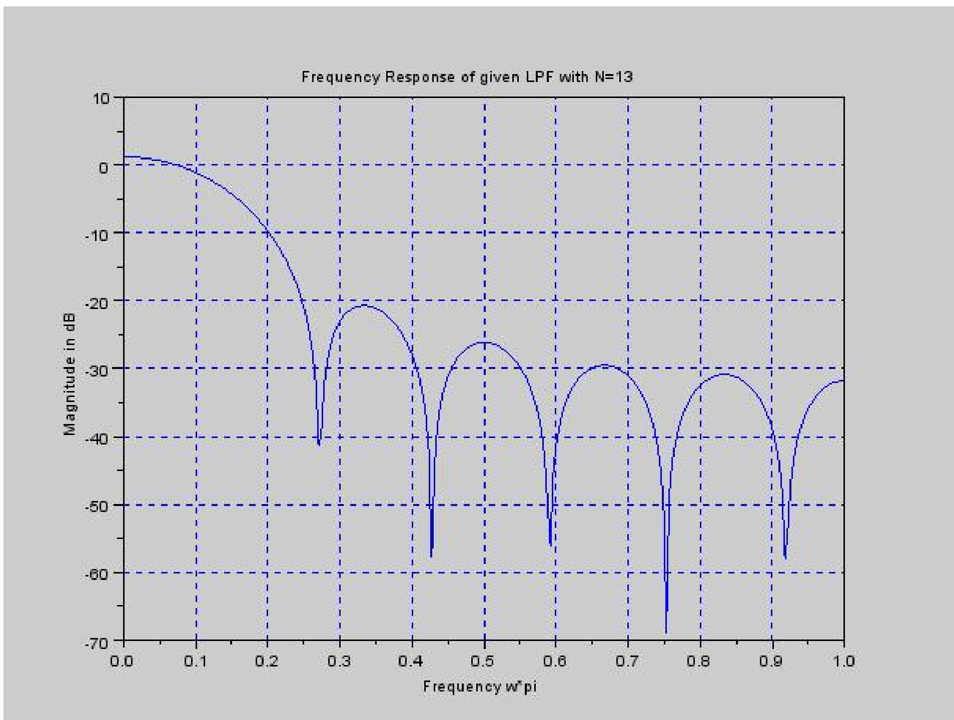

Figure 6.16: Filter Coefficients Determination

```
//Example 6.19
//Program to Plot Magnitude Responce of given L.P.F.
    with specifications:
//N=13,w=pi/6
clear all;
clc ;
close ;
alpha=6;
U=1;
for n=0+U:1:12+U
if n==7
hd(n)=0.167;
else
hd(n)=(sin(%pi*(n-U-alpha)/6))/(%pi*(n-U-alpha));
end
end
[hzm ,fr ]= frmag (hd ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of given LPF with N=13');
xgrid (2)
disp(hd,"Filter Coefficients,h(n)=");
```

Scilab code Exa 6.20 Filter Coefficients using Hamming Window

```
//Example 6.20
//Program to Plot Magnitude Responce of given L.P.F.
    with specifications:
//N=13,w=pi/6
```

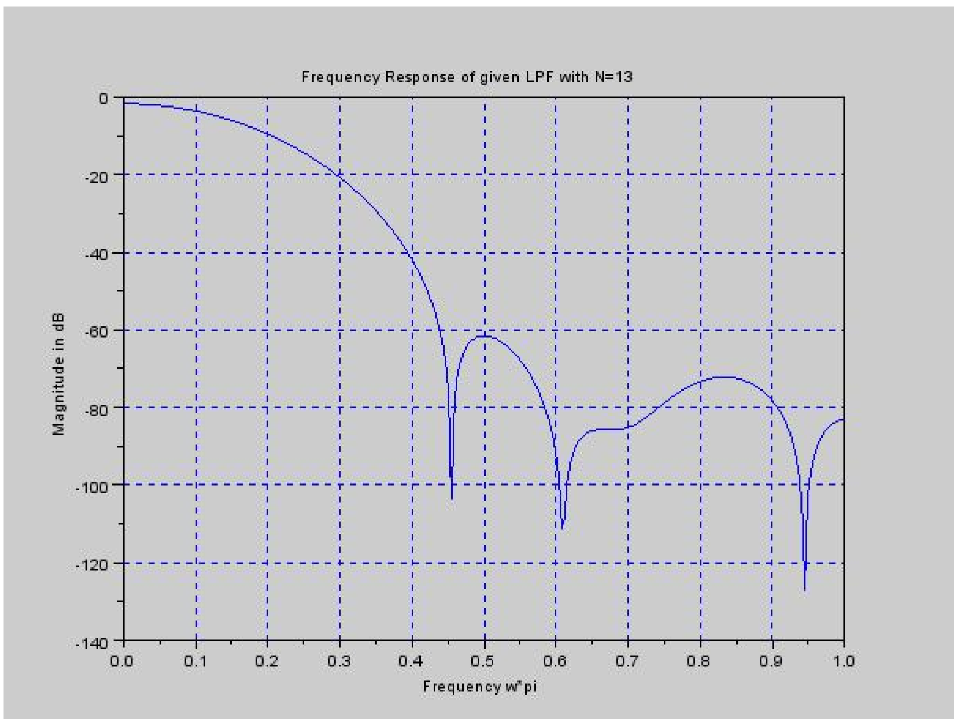

Figure 6.17: Filter Coefficients using Hamming Window

```
//Using Hamming Window
clear all;
clc ;
close ;
N=13;
alpha=6;
U=1;
h_hamm=window('hm',N);
for n=0+U:1:12+U
if n==7
hd(n)=0.167;
else
hd(n)=(sin(%pi*(n-U-alpha)/6))/(%pi*(n-U-alpha));
end
h(n)=hd(n)*h_hamm(n);
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of given LPF with N=13');
xgrid (2)
disp(h,"Filter Coefficients,h(n)=");
disp(h,"Filter Coefficients,h(n)=");
```

Scilab code Exa 6.21 LPF Filter using Rectangular Window

```
//Example 6.21
//Program to Plot Magnitude Responce of given L.P.F.
    with specifications:
```

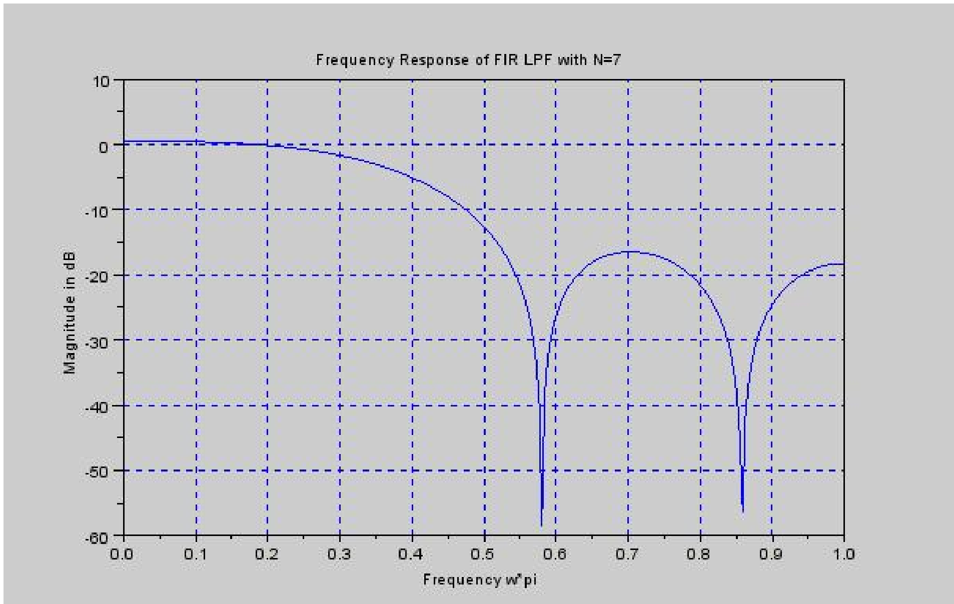

Figure 6.18: LPF Filter using Rectangular Window

```
//N=7,fc=1000Hz,F=5000Hz
clear all;
clc ;
close ;
N=7;
U=4;
h_Rect=window('re',N);
for n=-3+U:1:3+U
if n==4
hd(n)=0.4;
else
hd(n)=(sin(2*%pi*(n-U)/5))/(%pi*(n-U));
end
h(n)=hd(n)*h_Rect(n);
end
[hzm ,fr ]= frmag (h ,256) ;
hzm_dB = 20* log10 (hzm)./ max ( hzm );
figure
plot (2*fr , hzm_dB )
```

```
a= gca ();
xlabel ('Frequency w*pi');
ylabel ('Magnitude in dB');
title ('Frequency Response of FIR LPF with N=7');
xgrid (2)
disp(h,"Filter Coefficients ,h(n)=");
```

Scilab code Exa 6.28 Filter Coefficients for Direct Form Structure

```
//Example 6.28
//Program to calculate FIR Filter coefficients for
   the direct form structure
//k1=1/2 , k2=1/3 , k3=1/4
clear all;
clc ;
close ;
U=1;
k1=1/2;
k2=1/3;
k3=1/4;
a(3+U,0+U)=1;
a(1+U,1+U)=k1;
a(2+U,2+U)=k2;
a(3+U,3+U)=k3;
m=2,k=1;
a(m+U,k+U)=a(m-1+U,k+U)+a(m+U,m+U)*a(m-1+U,m-k+U);
m=3,k=1;
a(m+U,k+U)=a(m-1+U,k+U)+a(m+U,m+U)*a(m-1+U,m-k+U);
m=3,k=2;
a(m+U,k+U)=a(m-1+U,k+U)+a(m+U,m+U)*a(m-1+U,m-k+U);
disp(a(3+U,0+U),'a(3,0)');
disp(a(3+U,1+U),'a(3,1)');
disp(a(3+U,2+U),'a(3,2)');
disp(a(3+U,3+U),'a(3,3)');
```

Scilab code Exa 6.29 Lattice Filter Coefficients Determination

```
//Example 6.29
//Program to calculate given FIR Filter's Lattice
   form coefficients.
clear all;
clc ;
close ;
U=1;              //Zero Adjust
a(3+U,0+U)=1;
a(3+U,1+U)=2/5;
a(3+U,2+U)=3/4;
a(3+U,3+U)=1/3;
a(2+U,0+U)=1; //a(m,0)=1
a(2+U,3+U)=1/3;
m=3,k=1;
a(m-1+U,k+U)=(a(m+U,k+U)-a(m+U,m+U)*a(m+U,m-k+U))
   /(1-a(m+U,m+U)*a(m+U,m+U));
m=3,k=2;
a(m-1+U,k+U)=(a(m+U,k+U)-a(m+U,m+U)*a(m+U,m-k+U))
   /(1-a(m+U,m+U)*a(m+U,m+U));
m=2,k=1;
a(m-1+U,k+U)=(a(m+U,k+U)-a(m+U,m+U)*a(m+U,m-k+U))
   /(1-a(m+U,m+U)*a(m+U,m+U));
disp(a(1+U,1+U),'k1');
disp(a(2+U,2+U),'k2');
disp(a(3+U,3+U),'k3');
```

Chapter 7

FINITE WORD LENGTH EFFECTS IN DIGITAL FILTERS

Scilab code Exa 7.2 Subtraction Computation

```
//Example 7.2
//To Compute Subtraction
//(a) 0.25 from 0.5
clear all;
clc ;
close ;
a=0.5;
b=0.25;
c=a-b;
disp(c,'=',b,'-',a,'PART 1');
//(a) 0.5 from 0.25
d=b-a;
disp(d,'=',a,'-',b,'PART 2');
```

Scilab code Exa 7.14 Variance of Output due to AD Conversion Process

```
//Example 7.14
//To Compare the Varience of Output due to A/D
    Conversion process
//y(n)=0.8y(n-1)+x(n)
clear all;
clc ;
close ;
n=8;  //Bits
r=100;  //Range
Q=2*r/(2^n);  //Quantization Step Size
Ve=(Q^2)/12;
Vo=Ve*(1/(1-0.8^2));
disp(Q,'QUANTIZATION STEP SIZE =');
disp(Ve,'VARIANCE OF ERROR SIGNAL =');
disp(Vo,'VARIANCE OF OUTPUT =');
```

Chapter 8

MULTIRATE SIGNAL PROCESSING

Scilab code Exa 8.9 Two Component Decomposition

```
//Example 8.9
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Develop a two component decomposition for the
   transfer function
//and determine P0(z) and P1(z)
clear all;
clc ;
close ;
syms z a n;
HZ=(z)/(z-a);
hn=a^n;//Inverse Z Transform of HZ
h2n=a^(2*n);
P0=symsum(h2n*z^(-n),n,0,%inf);
h2n1=a^(2*n+1);
P1=symsum(h2n1*z^(-n),n,0,%inf);
disp(P0,'P0(Z) = ');
disp(P1,'P1(Z) = ');
```

Scilab code Exa 8.10 Two Band Polyphase Decomposition

```
//Example 8.10
//Develop a two band polyphase decomposition for the
    transfer function
//H(z)=z^2+z+2/z^2+0.8z+0.6
clear all;
clc ;
close ;
z=%z;
HZ=(z^2+z+2)/(z^2+0.8*z+0.6);
HZa=horner(HZ,-z);
P0=0.5*(HZ+HZa);
P1=0.5*(HZ-HZa);
disp(P1/z,'+',P0,'H(z) =')
```

Chapter 9

STATISTICAL DIGITAL SIGNAL PROCESSING

Scilab code Exa 9.7.a Frequency Resolution Determination

```
//Example 9.7 (a)
//Program To Determine Frequency Resolution of Bartlett,
//Welch(50% Overlap) and Blackmann-Tukey Methods
clear all;
clc;
close;
//Data
Q=10;//Quality Factor
N=1000;//Samples
//FREQUENCY RESOLUTION CALCULATION
K=Q;
rb=0.89*(2*%pi*K/N);
rw=1.28*(2*%pi*9*Q)/(16*N);
rbt=0.64*(2*%pi*2*Q)/(3*N);
//Display the result in command window
disp(rb,"Resolution of Bartlett Method");
disp(rw,"Resolution of Welch(50% overlap) Method");
disp(rbt,"Resolution of Blackmann-Tukey Method");
```

Scilab code Exa 9.7.b Record Length Determination

```
//Example 9.7 (b)
//Program To Determine Record Length of Bartlett ,
//Welch(50% Overlap) and Blackmann−Tukey Methods
clear all;
clc;
close;
//Data
Q=10;//Quality Factor
N=1000;//Samples
//RECORD LENGTH CALCULATION
lb=N/Q;
lw=16*N/(9*Q);
lbt=3*N/(2*Q);
//Display the result in command window
disp(lb,"Record Length of Bartlett Method");
disp(lw,"Record Length of Welch(50% overlap) Method"
    );
disp(lbt,"Record Length of Blackmann−Tukey Method");
```

Scilab code Exa 9.8.a Smallest Record Length Computation

```
//Example 9.8 (a)
//Program To Determine Smallest Record Length of
    Bartlett Method
clear all;
clc;
close;
//Data
fr=0.01;//Frequency Resolution
```

```
N=2400; //Samples
//RECORD LENGTH CALCULATION
lb=0.89/fr;
//Display the result in command window
disp(lb,"Record Length of Bartlett Method");
```

Scilab code Exa 9.8.b Quality Factor Computation

```
//Example 9.8 (b)
//Program To Determine Quality Factor of Bartlett
   Method
clear all;
clc;
close;
//Data
fr=0.01;//Frequency Resolution
N=2400; //Samples
lb=0.89/fr;
//QUALITY FACTOR CALCULATION
Q=N/lb;
//Display the result in command window
disp(Q,"Quality Factor of Bartlett Method");
```

Chapter 11

DIGITAL SIGNAL PROCESSORS

Scilab code Exa 11.3 Program for Integer Multiplication

```
//Program 11.3
//Program To Calculate the value of the function
//Y=A*B
clear all;
clc;
close;
//Input Data
A=input('Enter Integer Number A =');
B=input('Enter Integer Number B =');
//Multiplication Computation
Y=A*B;
//Display the result in command window
disp(Y,"Y = A*B = ");
```

Scilab code Exa 11.5 Function Value Calculation

```
//Program 11.5
//Program To Calculate the value of the function
//Y=A*X1+B*X2+C*X3
clear all;
clc;
close;
//Data
A=1;
B=2;
C=3;
X1=4;
X2=5;
X3=6;
//Compute the function
Y=A*X1+B*X2+C*X3;
//Display the result in command window
disp(Y,"Y = A*X1+B*X2+C*X3 = ");
```