Scilab Textbook Companion for Digital Signal Processing: A Modern Introduction
by A. Ashok[1]

Created by
Edulakante Nagarjun Reddy
B.Tech (pursing)
Electronics Engineering
NIT Surat(SVNIT)
College Teacher
Jigisha Patel
Cross-Checked by
K. Suryanarayan, IITB

August 11, 2013

[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: A Modern Introduction

Author: A. Ashok

Publisher: Cenage Learning India Private Limited

Edition: 5

Year: 2010

ISBN: 81-315-0179-5

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 2

Discrete Signals

Scilab code Exa 2.1a Signal energy and power

```
//example 2.1.a,pg no.11
for i=1:1:50
    x(1,i)=3*(0.5)^(i-1);
end
//summation of x
E=0
for i=1:1:50
    E=E+x(1,i)^2;
end
disp("the energy of given signal is")
E
```

Scilab code Exa 2.1b Average power of periodic signals

```
//example 2.1b,pg.no.11
n=1:1:10;
xn=6*cos((2*%pi*n')/4);
a=4;
```

```
p=0;
for i=1:1:a
    p=p+abs(xn(i)^2);
end
P=p/a;
disp("The average power of given signal is")
P
```

Scilab code Exa 2.1c Average power of periodic signals

```
//example 2.1c,pg.no.11
n=1:4;
xn=6*%e^((%i*%pi*n')/2);
a=4;
p=0;
for i=1:1:a
    p=p+abs(xn(i)^2);
end
P=p/a;
disp("The average power of given signal is")
P
```

Scilab code Exa 2.2 Operations on Discrete Signals

```
//example 2.2,pg no.12
x=[0 2 3 4 5 6 7];
n1=-3:1:3;
y=x;
n2=0:1:6;
f=x;
n3=-5:1:1;
```

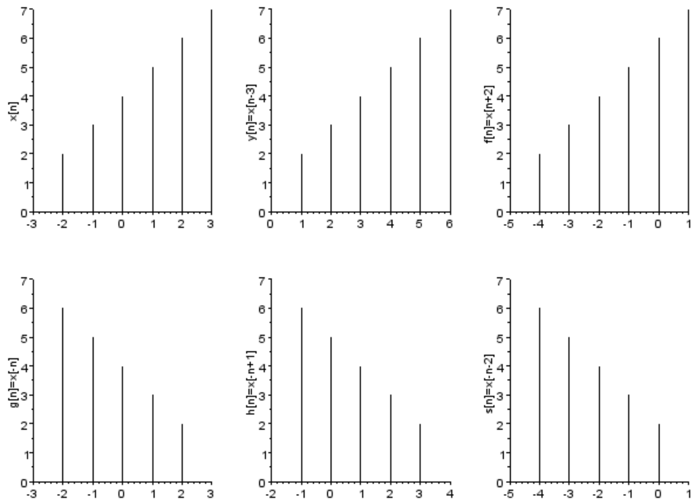

Figure 2.1: Operations on Discrete Signals

```
g=x(length(x):-1:1);
n4=-3:1:3;
h=x(length(x):-1:1);
n5=-2:1:4;
s=x(length(x):-1:1);
n6=-5:1:1;
a=gca();
subplot(231);
plot2d3('gnn',n1,x);
ylabel('x[n]');
subplot(232);
plot2d3('gnn',n2,y);
ylabel('y[n]=x[n-3]');
subplot(233);
plot2d3('gnn',n3,f);
ylabel('f[n]=x[n+2]');
subplot(234);
plot2d3('gnn',n4,g);
ylabel('g[n]=x[-n]');
subplot(235);
```

```
plot2d3('gnn',n5,h);
ylabel('h[n]=x[-n+1]');
subplot(236);
plot2d3('gnn',n6,s);
ylabel('s[n]=x[-n-2]');
```

Scilab code Exa 2.3a Even and Odd parts of Discrete signals

```
//example 2.3a pg.no.14
clear;clc;close;
n=-2:2;
x1=[4 -2 4 -6 0];
x2=0.5*x1//x[n]
x3=0.5*[x1(length(x1):-1:1)];//x[-n]
xe=(x2+x3);//even part
xo=(x2-x3);//odd part
a=gca();
a.thickness=2;
a.x_location="middle";
a.y_location="middle";
plot2d3('gnn',n,xe,rect=[-4 -6 4 6])
xtitle('graphical representation of even part of x[n
    ]','n','x[n]')
xset('window',1)
b=gca();
b.thickness=2;
b.y_location="middle";
b.x_location="middle";
plot2d3('gnn',n,xo,rect=[-2 -4 2 4])
xtitle('graphical representation of odd part of x[n]
    ','n','x[n]')
```

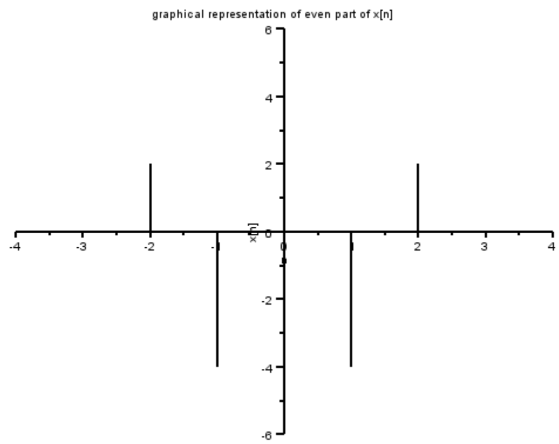

Figure 2.2: Even and Odd parts of Discrete signals

Scilab code Exa 2.3b Even and Odd parts of Discrete signals

```
//example 2.3a pg.no.14
clear;clc;close;
x1=[0 0 0 0 0 1 1 1 1 ];
n=-4:4;
x2=0.5*x1//x[n]
x3=0.5*[x1(length(x1):-1:1)]//x[-n]
xe=(x2+x3);//even part
xo=(x2-x3);//odd part
a=gca();
a.thickness=2;
a.y_location="middle";
a.x_location="middle";
plot2d3('gnn',n,xe,rect=[-4 -1 4 1]);
```

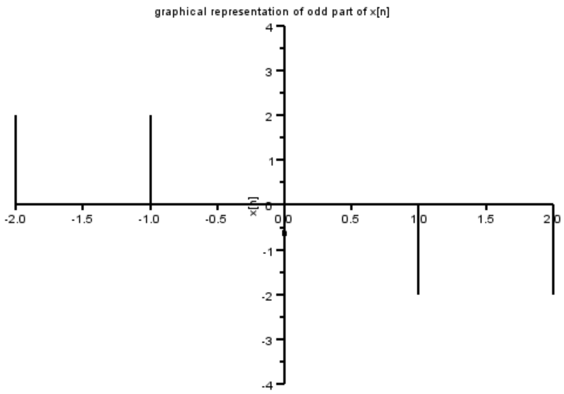

Figure 2.3: Even and Odd parts of Discrete signals

```
xtitle('graphical representation of even part of x[n
    ]','n','x[n]')
xset('window',1)
b=gca();
b.thickness=2;
b.y_location="middle";
b.x_location="middle";
plot2d3('gnn',n,xo,rect=[-4 -1 4 1]);
xtitle('graphical representation of odd part of x[n]
    ','n','x[n]')
```

Scilab code Exa 2.4a Decimation and Interpolation of Discrete signals

```
//example 2.4a pg.no.17
```

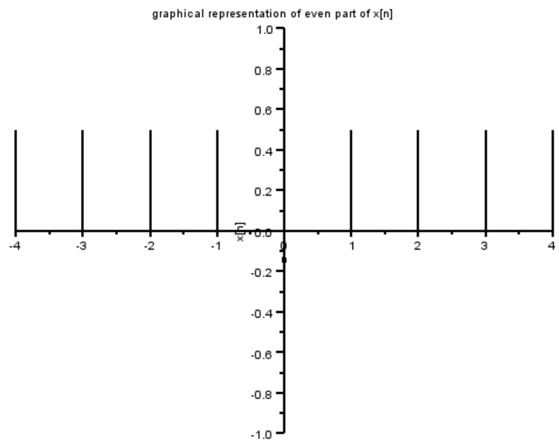

Figure 2.4: Even and Odd parts of Discrete signals

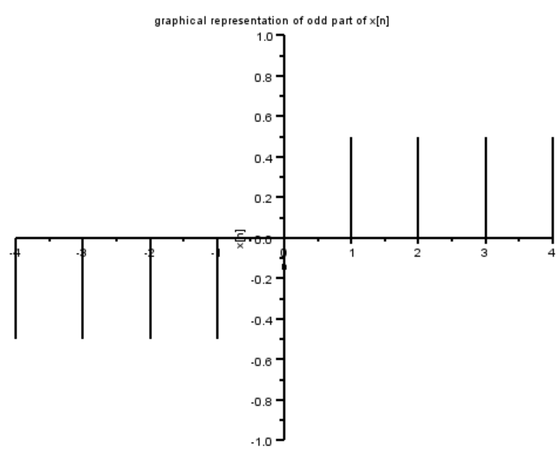

Figure 2.5: Even and Odd parts of Discrete signals

```
x=[1 2 5 -1];
xm=2;//denotes 2nd sample has pad.
y=[x(1:2:xm-2),x(xm:2:length(x))]//decimation
h=[x(1:1/3:length(x))]//step interpolated
g=h;
for i=2:3
    g(i:3:length(g))=0;
end
//zero interpolated
x1=1:3:3*length(x);
s=interpln([x1;x],1:10)//linear interpolated
```

Scilab code Exa 2.4b Decimation and Interpolation of Discrete signals

```
//example 2.4 b,c.pg.no.17
x=[3 4 5 6];
xm=3;//denotes 3rd sample has pad
xm=xm-1;//shifting
g=[x(xm-2:-2:1),x(xm:2:length(x))]//decimation
xm=3;
h=[x(1:1/2:length(x))]//step interpolated
```

Scilab code Exa 2.4c Decimation and Interpolation of Discrete signals

```
//example 2.4c,pg.no.17
x=[3 4 5 6];
xm=3;
xm=xm+1*(xm-1);//shift in pad due to interpolation
xm=xm-2//normal shifting
x1=[x(1:1/3:length(x))]//step interpolated
xm=3;
xm=xm+2*(xm-1)//shift in pad due to interpolation
y=[x1(1:2:xm-2),x1(xm:2:length(x1))]//decimation
```

Scilab code Exa 2.4d Decimation and Interpolation of Discrete signals

```
//example 2.4d,pg.no.17
x=[2 4 6 8]
xm=3;//denote 3rd sample has pad
x1=[1 3 5 7]
x2=interpln([x1;x],1:6)
xm=xm+1*(xm-1);//shift in pad due to interpolation
xm=xm-1//shift in pad due to delay
y=[x2(2:2:xm-2),x2(xm:2:length(x2))]//decimation
```

Scilab code Exa 2.5 Describing Sequences and signals

```
//example 2.5,pg.no.20
x=[1 2 4 8 16 32 64];
y=[0 0 0 1 0 0 0];
z=x.*y;
a=0;
for i=1:length(z)
    a=a+z(i);
end
z,a//a=summation of z
```

Scilab code Exa 2.6 Discrete time Harmonics and Periodicity

```
//example 2.6 pg.no.23
function[p]=period(x)
for i=2:length(x)
    v=i
```

```
    if (abs(x(i)-x(1))<0.00001)
        k=2
        for j=i+1:i+i
            if (abs(x(j)-x(k))<0.00001)
                v=v+1
            end
        k=k+1;
        end
     end
    if (v==(2*i)) then
        break
    end
end
p=i-1
endfunction
for i=1:60
    x1(i)=cos((2*%pi*8*i)/25);
end
for i=1:60
    x2(i)=exp(%i*0.2*i*%pi)+exp(-%i*0.3*i*%pi);
end
for i=1:45
    x3(i)=2*cos((40*%pi*i)/75)+sin((60*%pi*i)/75);
end
period(x1)
period(x2)
period(x3)
```

check Appendix AP 1 for dependency:

`Aliasfrequency.sci`

Scilab code Exa 2.7 Aliasing and its effects

```
//example 2.7.pg.no.27
f=100;
```

```
s=240;
s1=s;
aliasfrequency(f,s)
s=140;
s1=s;
aliasfrequency(f,s,s1)
s=90;
s1=s;
aliasfrequency(f,s,s1)
s=35;
s1=s;
aliasfrequency(f,s,s1)
```

check Appendix AP 1 for dependency:

`Aliasfrequency.sci`

Scilab code Exa 2.8 Signal Reconstruction

```
f=100;
s=210;
s1=420;
aliasfrequency(f,s,s1)
s=140;
aliasfrequency(f,s,s1)
```

Chapter 3

Response of Digital Filters

Scilab code Exa 3.5 FIR filter response

```
//Response of non-recursive Filters
for i=1:4
    x(i)=0.5^i;
end
x1=[0;1;x(1:2)]
for i=1:4
     y(i)=2*x(i)-3*x1(i);
end
y(1),y(2)
```

Scilab code Exa 3.19a Analytical Evaluation of Discrete Convolution

```
//Analytical evaluation of Discrete Convolution
clear;close;clc;
max_limit=10;
h=ones(1,max_limit);
n2=0:length(h)-1;
x=h;
```

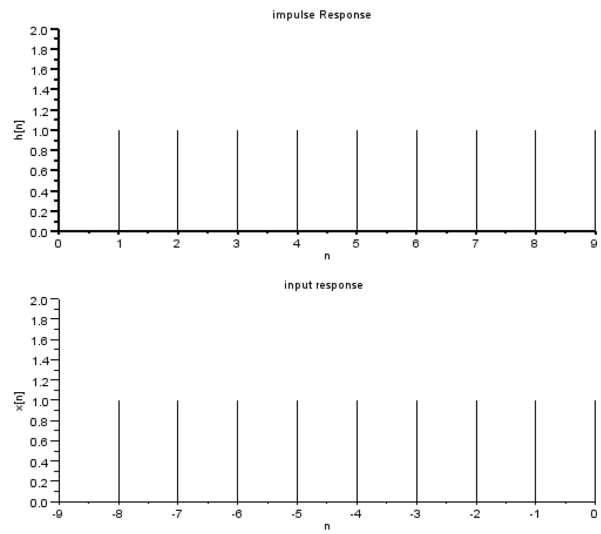

Figure 3.1: Analytical Evaluation of Discrete Convolution

```
n1=-length(x)+1:0;
y=convol(x,h);
n=-length(x)+1:length(h)-1;
a=gca();
subplot(211);
plot2d3('gnn',n2,h)
xtitle('impulse Response','n','h[n]');
a.thickness=2;
a.y_location="origin";
subplot(212);
plot2d3('gnn',n1,x)
a.y_location="origin";
xtitle('input response','n','x[n]');
xset("window",1);
a=gca();
plot2d3('gnn',n,y)
xtitle('output response','n','y[n]');
```

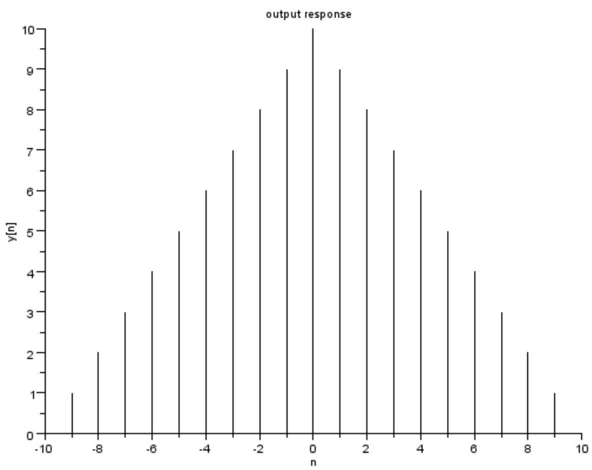

Figure 3.2: Analytical Evaluation of Discrete Convolution

Scilab code Exa 3.19b Analytical Evaluation of Discrete Convolution

```
clear;close;clc;
max_limit=10;
for n=1:max_limit
    h(n)=(0.4)^n;
end
n2=0:length(h)-1;
for n=1:max_limit
    x(n)=(0.8)^n;
end
n1=-length(x)+1:0;
y=convol(x,h)
n=-length(x)+1:length(h)-1;
a=gca();
```

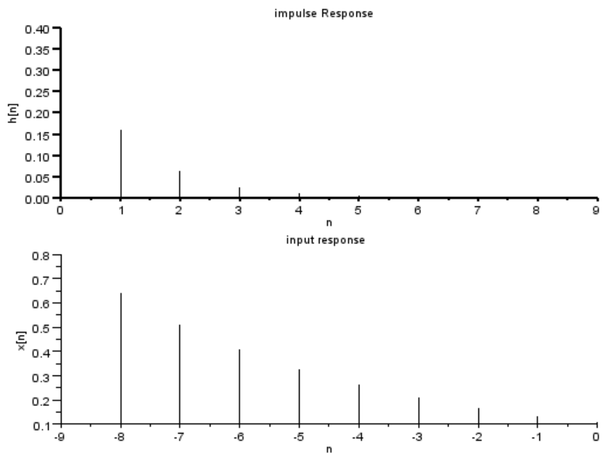

Figure 3.3: Analytical Evaluation of Discrete Convolution

```
subplot(211);
plot2d3('gnn',n2,h)
xtitle('impulse Response','n','h[n]');
a.thickness=2;
a.y_location="origin";
subplot(212);
plot2d3('gnn',n1,x)
a.y_location="origin";
xtitle('input response','n','x[n]');
xset("window",1);
a=gca();
plot2d3('gnn',n,y)
xtitle('output response','n','y[n]');
```

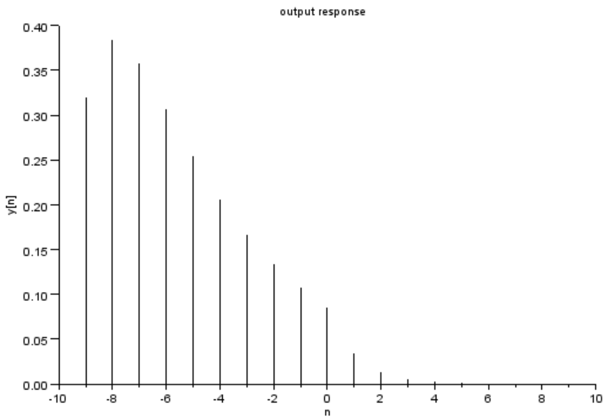

Figure 3.4: Analytical Evaluation of Discrete Convolution

Scilab code Exa 3.19c Analytical Evaluation of Discrete Convolution

```
//Analytical Evaluation of Discrete convolution
clear;close;clc;
max_limit=5;
h(1)=0;
for n=2:max_limit
    h(n)=0.8^n;
end
n2=0:length(h)-1;
x=[0 ones(1:max_limit)]
n1=-length(x)+1:0;
y=convol(x,h);
n=-length(x)+1:length(h)-1;
a=gca();
subplot(211);
plot2d3('gnn',n2,h)
xtitle('impulse Response','n','h[n]');
a.thickness=2;
a.y_location="origin";
```

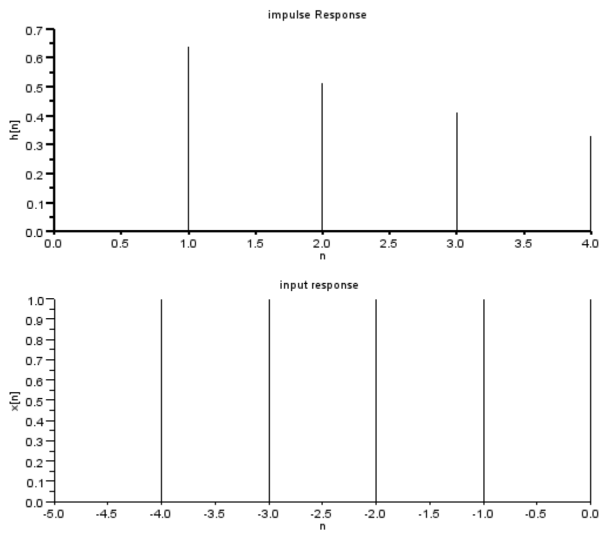

Figure 3.5: Analytical Evaluation of Discrete Convolution

```
a=gca();
subplot(212);
plot2d3('gnn',n1,x)
a.y_location="origin";
xtitle('input response','n','x[n]');
xset("window",1);
a=gca();
plot2d3('gnn',n,y)
a.y_location="origin";
a.x_location="origin";
xtitle('output response','n','y[n]');
```

Scilab code Exa 3.20a Properties of Convolution

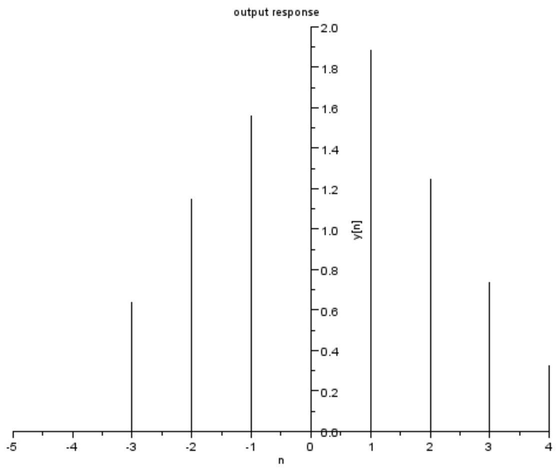

Figure 3.6: Analytical Evaluation of Discrete Convolution

```
//properties of convolution
x=[1 2 3 4 5];
h=[1 zeros(1:5)];
a=convol(x,h);
b=convol(h,x);
a==b
```

Scilab code Exa 3.20b Properties of Convolution

```
//Convolution with Step Function
x=[1 2 3 4 5];
h=[ones(1:5)];
a=convol(h,x);
b(1)=a(1);
for i=2:length(x)
    b(i)=b(i-1)+x(i);
end
```

```
disp(a(1:length(x)),b,'Step Response is running sum
   of impulses can be seen below');
```

Scilab code Exa 3.21a Convolution of finite length Signals

```
//convolution of finite length signals
clear;close;clc;
max_limit=10;
h=[1 2 2 3];
n2=0:length(h)-1;
x=[2 -1 3];
n1=0:length(x)-1;
y=convol(x,h);
n=0:length(h)+length(x)-2;
a=gca();
subplot(211);
plot2d3('gnn',n2,h)
xtitle('impulse Response','n','h[n]');
a.thickness=2;
a.y_location="origin";
a=gca();
subplot(212);
plot2d3('gnn',n1,x)
a.y_location="origin";
a.x_location="origin";
xtitle('input response','n','x[n]');
xset("window",2);
a=gca();
plot2d3('gnn',n,y)
a.y_location="origin";
xtitle('output response','n','y[n]');
```

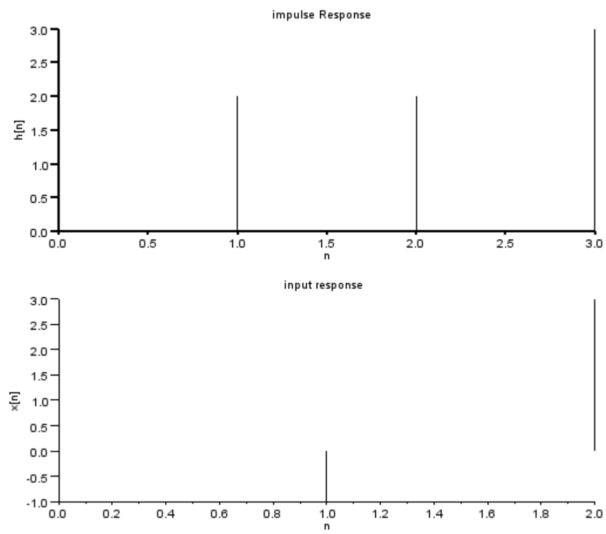

Figure 3.7: Convolution of finite length Signals

Scilab code Exa 3.21b Convolution of finite length Signals

```
clear;close;clc;
max_limit=10;
h=[2 5 0 4];
n2=-2:length(h)-3;
x=[4 1 3];
n1=-1:length(x)-2;
y=convol(x,h);
n=-3:length(x)+length(h)-5;
a=gca();
subplot(211);
plot2d3('gnn',n2,h)
xtitle('impulse Response','n','h[n]');
a.thickness=2;
```

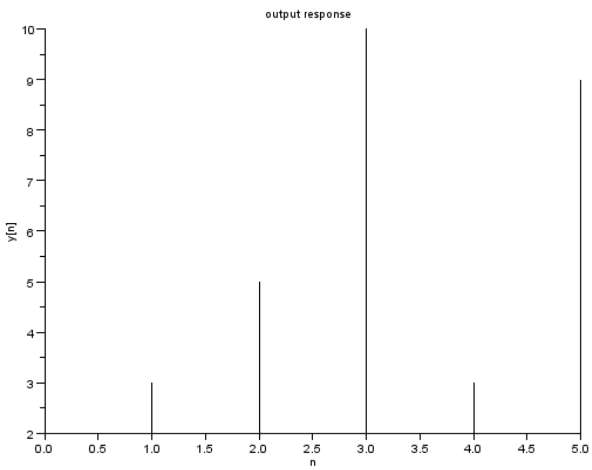

Figure 3.8: Convolution of finite length Signals

```
a.y_location="origin";
a=gca();
subplot(212);
plot2d3('gnn',n1,x)
a.y_location="origin";
xtitle('input response','n','x[n]');
xset("window",1);
a=gca();
plot2d3('gnn',n,y)
a.y_location="origin";
a.x_location="origin";
xtitle('output response','n','y[n]');
```

Scilab code Exa 3.21c Convolution of finite length Signals

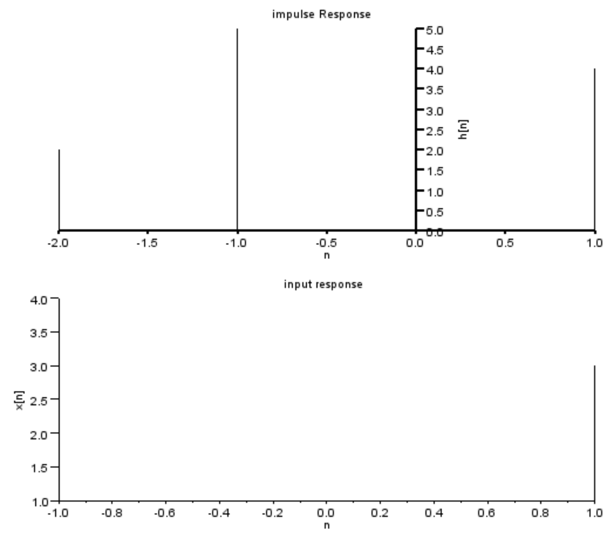

Figure 3.9: Convolution of finite length Signals

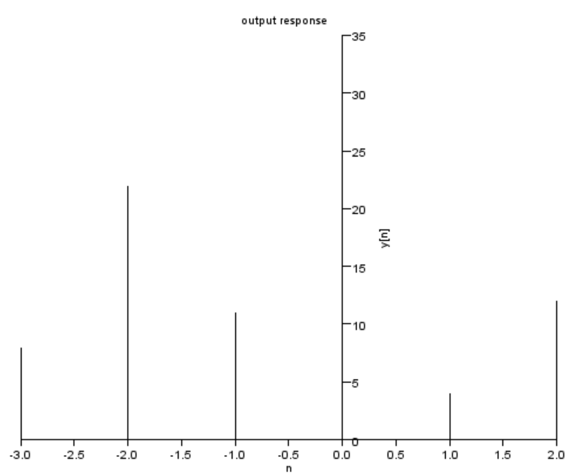

Figure 3.10: Convolution of finite length Signals

```
clear;close;clc;
max_limit=10;
h=[1/2 1/2 1/2];
n2=0:length(h)-1;
x=[2 4 6 8 10];
n1=0:length(x)-1;
y=convol(x,h);
n=0:length(x)+length(h)-2;
a=gca();
subplot(211);
plot2d3('gnn',n2,h);
xtitle('impulse Response','n','h[n]');
a.thickness=2;
a.y_location="origin";
a=gca();
subplot(212);
plot2d3('gnn',n1,x);
a.y_location="origin";
xtitle('input response','n','x[n]');
xset("window",1);
a=gca();
plot2d3('gnn',n,y)
a.y_location="origin";
a.x_location="origin";
xtitle('output response','n','y[n]');
```

Scilab code Exa 3.22 Convolution of finite length Signals

```
max_limit=10;
h=[2 5 0 4];
n2=0:length(h)-1;
```

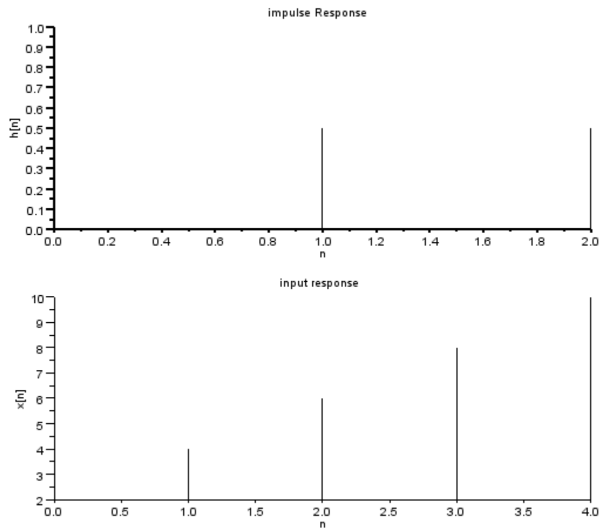

Figure 3.11: Convolution of finite length Signals

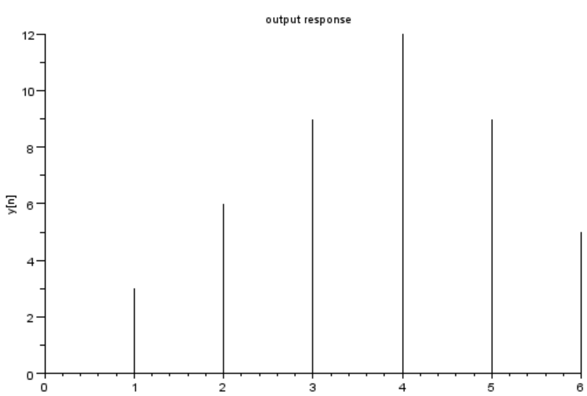

Figure 3.12: Convolution of finite length Signals

```
x=[4 1 3];
n1=0:length(x)-1;
y=convol(x,h);
n=0:length(x)+length(h)-2;
a=gca();
subplot(211);
plot2d3('gnn',n2,h)
xtitle('impulse Response','n','h[n]');
a.thickness=2;
a.y_location="origin";
a=gca();
subplot(212);
plot2d3('gnn',n1,x)
a.y_location="origin";
a.x_location="origin";
xtitle('input response','n','x[n]');
xset("window",1);
a=gca();
plot2d3('gnn',n,y)
a.y_location="origin";
a.x_location="origin";
xtitle('output response','n','y[n]');
```

Scilab code Exa 3.23 effect of Zero Insertion,Zero Padding on convol.

```
//convolution by polynomial method
x=[4 1 3];
h=[2 5 0 4];
z=%z;
n=length(x)-1:-1:0;
X=x*z^n';
```

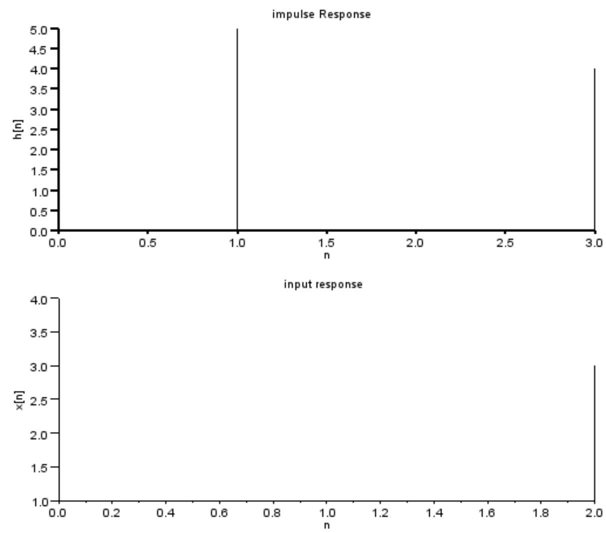

Figure 3.13: Convolution of finite length Signals

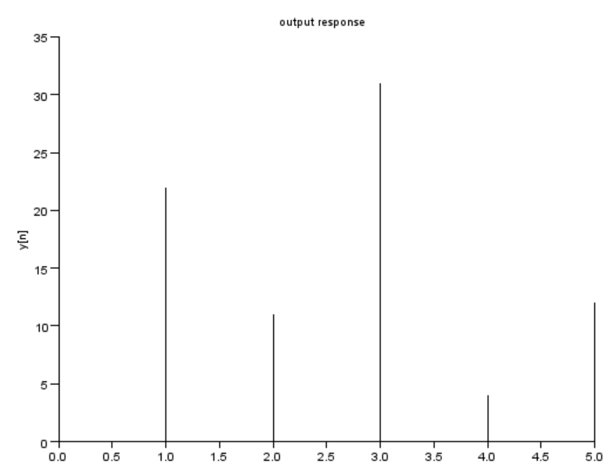

Figure 3.14: Convolution of finite length Signals

```
n1=length(h)-1:-1:0;
H=h*z^n1';
y=X*H
//effect of zero insertion on convolution
h=[2 0 5 0 0 0 4];
x=[4 0 1 0 3];
y=convol(x,h)
//effect of zero padding on convolution
h=[2 5 0 4 0 0];
x=[4 1 3 0];
y=convol(x,h)
```

Scilab code Exa 3.25 Stability and Causality

```
//concepts based on stability and Causality
function []=stability(X)
if (abs(roots(X))<1)
    disp("given system is stable")
else
    disp("given system is not stable")
end
endfunction
x=[1 -1/6 -1/6];
z=%z;
n=length(x)-1:-1:0;
//characteristic eqn is
X=x*(z)^n'
stability(X)
x=[1 -1];
n=length(x)-1:-1:0;
//characteristic eqn is
X=x*(z)^n'
stability(X)
x=[1 -2 1];
n=length(x)-1:-1:0;
```

```
//characteristic eqn is
X=x*(z)^n'
stability(X)
```

Scilab code Exa 3.26 Response to Periodic Inputs

```
//Response of periodic inputs
function[p]=period(x)
for i=2:length(x)
    v=i
    if (abs(x(i)-x(1))<0.00001)
        k=2
        for j=i+1:i+i
            if (abs(x(j)-x(k))<0.00001)
                v=v+1
            end
        k=k+1;
        end
     end
    if (v==(2*i)) then
        break
    end
end
p=i-1
endfunction
x=[1 2 -3 1 2 -3 1 2 -3];
h=[1 1];
y=convol(x,h)
y(1)=y(4);
period(x)
period(y)
h=[1 1 1];
y=convol(x,h)
```

Scilab code Exa 3.27 Periodic Extension

```
//to find periodic extension
x=[1 5 2;0 4 3;6 7 0];
y=[0 0 0];
for i=1:3
    for j=1:3
        y(i)=y(i)+x(j,i);
    end
end
y
```

Scilab code Exa 3.28 System Response to Periodic Inputs

```
//method of wrapping to fing convolution of periodic
    signal with one period
x=[1 2 -3];
h=[1 1];
y1=convol(h,x)
y1=[y1,zeros(5:9)]
y2=[y1(1:3);y1(4:6);y1(7:9)];
y=[0 0 0];
for i=1:3
    for j=1:3
        y(i)=y(i)+y2(j,i);
    end
end
y
x=[2 1 3];
h=[2 1 1 3 1];
y1=convol(h,x)
y1=[y1,zeros(8:9)]
```

```
y2=[y1(1:3);y1(4:6);y1(7:9)];
y=[0 0 0];
for i=1:3
    for j=1:3
        y(i)=y(i)+y2(j,i);
    end
end
y
```

Scilab code Exa 3.29 Periodic Convolution

```
//periodic or circular convolution
x=[1 0 1 1];
h=[1 2 3 1];
y1=convol(h,x)
y1=[y1,zeros(8:12)];
y2=[y1(1:4);y1(5:8);y1(9:12)];
y=[0 0 0 0];
for i=1:4
    for j=1:3
        y(i)=y(i)+y2(j,i);
    end
end
y
```

Scilab code Exa 3.30 Periodic Convolution by Circulant Matrix

```
//periodic convolution by circulant matrix
x=[1 0 2];
h=[1;2;3];
//generation of circulant matrix
c(1,:)=x;
for i=2:length(x)
```

```
    c(i,:)=[x(length(x):length(x)-i),x(1:length(x)-i)]
end
c'
```

Scilab code Exa 3.32 Deconvolution By polynomial Division

```
//deconvolution by polynomial division
x=[2 5 0 4];
y=[8 22 11 31 4 12];
z=%z
n=length(x)-1:-1:0;
X=x*(z)^n'
n1=length(y)-1:-1:0;
Y=y*(z)^n1'
h=Y/X
```

Scilab code Exa 3.33 Autocorrelation and Cross Correlation

```
//discrete auto correlation and cross correlation
x=[2 5 0 4];
h=[3 1 4];
x1=x(length(x):-1:1)
h1=h(length(h):-1:1)
rxhn=convol(x,h1)
rhxn=convol(x1,h)
rhxn1=rhxn(length(rhxn):-1:1)
//we observe that rhxn1=rxhn
x=[3 1 -4];
x1=x(length(x):-1:1)
rxxn=convol(x,x1)
//we observe that rxxn is even symmetric about origin
```

Scilab code Exa 3.35 Periodic Autocorrelation and Cross Correlation

```
//discrete periodic auto correlation and cross
   correlation
x=[2 5 0 4];
h=[3 1 -1 2];
x1=x(length(x):-1:1);
h1=h(length(h):-1:1);
rxhn=convol(x,h1)
rhxn=convol(x1,h)
rxxn=convol(x,x1)
rhhn=convol(h,h1)
y1=[rxhn,zeros(8:12)];
y2=[y1(1:4);y1(5:8);y1(9:12)];
y3=[rhxn,zeros(8:12)];
y4=[y3(1:4);y3(5:8);y3(9:12)];
y5=[rxxn,zeros(8:12)];
y6=[y5(1:4);y5(5:8);y5(9:12)];
y7=[rhhn,zeros(8:12)];
y8=[y7(1:4);y7(5:8);y7(9:12)];
rxhp=[0 0 0 0];
rhxp=[0 0 0 0];
rxxn=[0 0 0 0];
rhhp=[0 0 0 0];
for i=1:4
     for j=1:3
        rhxp(i)=rhxp(i)+y4(j,i);
        rxhp(i)=rxhp(i)+y2(j,i);
        rxxn(i)=rxxn(i)+y6(j,i);
        rhhp(i)=rhhp(i)+y8(j,i);
    end
end
rxhp
rhxp
```

```
rxxn
rhhp
```

Chapter 4

z Transform Analysis

Scilab code Exa 4.1b z transform of finite length sequences

```
function[za]=ztransfer(sequence,n)
    z=poly(0,'z','r')
    za=sequence*(1/z)^n'
endfunction
x1=[2 1 -5 4];
n=-1:length(x1)-2;
ztransfer(x1,n)
```

Scilab code Exa 4.4a Pole Zero Plots

```
//Pole-Zero plots
z=%z;
az=2*z*(z+1);
bz=(z-1/3)*((z^2)+1/4)*((z^2)+4*z+5);
poles=roots(bz)
zeroes=roots(az)
h=az/bz
```

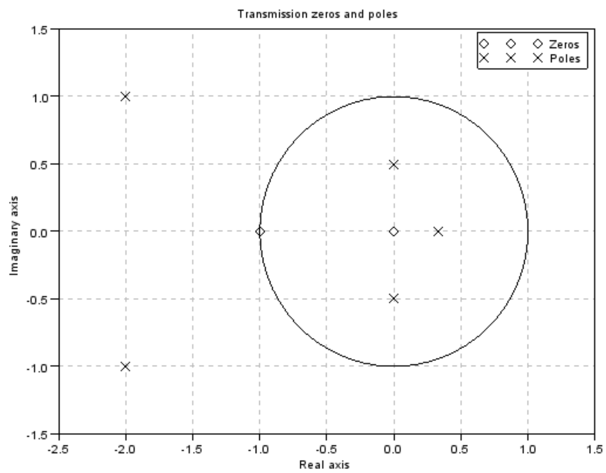

Figure 4.1: Pole Zero Plots

```
plzr(h)
```

Scilab code Exa 4.4b Pole Zero plots

```
//Pole-Zero plots
z=%z;
az=z^4+4.25*z^2+1;
bz=z^4;
poles=roots(bz)
zeroes=roots(az)
h=az/bz
plzr(h)
```

Scilab code Exa 4.8 Stability of Recursive Filters

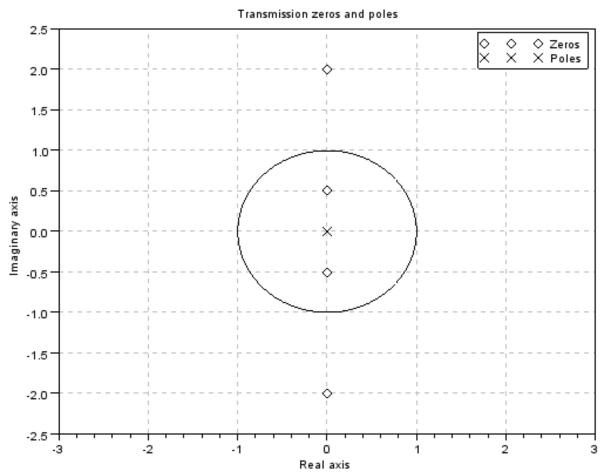

Figure 4.2: Pole Zero plots

```
//stability of recursive filter
//for roc:/z/>/a/
a=input('enter the value of alpha')
z=%z;
H=z/(z-a);
if (abs(a)<1)
    disp("system is stable")
else
    disp("system is not stable")
 end
 //for roc:/z/</a/
 if (abs(a)>1)
    disp("system is stable")
else
    disp("system is not stable")
 end
```

Scilab code Exa 4.9 Inverse Systems

```
//inverse systems
z=%z;
H=(1+2*(z^(-1)))/(1+3*(z^(-1)));
//inverse of H is
H1=1/H
H=1+2*(z^(-1))+3*(z^(-2));
H1=1/H
```

Scilab code Exa 4.10 Inverse Transform of sequences

```
//inverse transform of sequences
//(a)X(z)=3z^-1+5z^-3+2z^-4
z=%z;
X1=[3*z^-1;0;5*z^-3;2*z^-4];
n1=1:4;
ZI=z^n1';
x1=numer(X1.*ZI);
disp(x1,"x[n]=");
//(b)X(z)=2z^2-5z+5z^-1-2z^-2
X2=[2*z^2;-5*z;0;5*z^-1;-2*z^-2];
n2=-2:2;
ZI=z^n2';
x2=numer(X2.*ZI);
disp(x2,"x[n]=");
```

Scilab code Exa 4.11 Inverse Transform by Long Division

```
//inverse transform by long division
z=%z;
x=ldiv(z-4,1-z+z^2,5)
```

Scilab code Exa 4.12 Inverse transform of Right sided sequences

```
//inverse z transforms from standard transforms
z=%z;
xz=z/((z-0.5)*(z-0.25));
yz=xz/z;
pfss(yz)
//hence x[n]=-4(0.25)^n*un+4(0.5)^n*un;
xz=1/((z-0.5)*(z-0.25));
yz=xz/z;
pfss(yz)
//hence x[n]=-4(0.25)^n-1*u[n-1]+4(0.5)^n-1*u[n-1];
```

Scilab code Exa 4.20 z Transform of Switched periodic Signals

```
//z transform of switched periodic signals
z=%z;
//sol. for 4.20a
x1=[0 1 2];
n=0:2;
N=3;
x1z=x1*(1/z)^n'
xz=x1z/(1-z^-N)
//sol.for 4.20b
x1=[0 1 0 -1];
n=0:3;
N=4;
x1z=x1*(1/z)^n'
xz=x1z/(1-z^-N)
//sol.for 4.20c
xz=(2+z^-1)/(1-z^-3);
x1z=numer(xz)
//thus first period of xn is [2 1 0]
//sol.for 4.20d
xz=(z^-1-z^-4)/(1-z^-6);
```

```
x1z=numer(xz)
//thus first period of xn is [0 1 0 0 -1 0]
```

Chapter 5

Frequency Domain Analysis

Scilab code Exa 5.1c DTFT from Defining Relation

```
//DTFT of x[n]=(a)^n*u[n]
clear;
clc;close;
//DTS signal
a1=0.5;
a2=-0.5;
max_limit=10;
for n=0:max_limit-1
  x1(n+1)=(a1^n);
  x2(n+1)=(a2^n);
end
n=0:max_limit-1;
//discrete time fourier transform
wmax=2*%pi;
K=4;
k=0:(K/1000):K;
W=k*wmax/K;
x1=x1'
x2=x2'
XW1=x1*exp(%i*n'*W);
XW2=x2*exp(%i*n'*W);
```

```
XW1_Mag=abs(XW1);
XW2_Mag=abs(XW2);
W=[-mtlb_fliplr(W),W(2:1001)];//omega form
XW1_Mag=[mtlb_fliplr(XW1_Mag),XW1_Mag(2:1001)];
XW2_Mag=[mtlb_fliplr(XW2_Mag),XW2_Mag(2:1001)];
[XW1_phase,db]=phasemag(XW1);
[XW2_phase,db]=phasemag(XW2);
XW1_phase=[-mtlb_fliplr(XW1_phase),XW1_phase(2:1001)
    ];
XW2_phase=[-mtlb_fliplr(XW2_phase),XW2_phase(2:1001)
    ];

//plot for a>0
figure
subplot(3,1,1);
plot2d3('gnn',n,x1)
xtitle('Discrete time sequencex[n] a>0')
subplot(3,1,2);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d3(W,XW1_Mag);
title('magnitude Response abs(exp(jw))')
subplot(3,1,3);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW1_phase);
title('magnitude Response abs(exp(jw))')
//plot for a<0
figure
subplot(3,1,1);
plot2d3('gnn',n,x2);
xtitle('Discrete Time sequence x[n] for a>0')
subplot(3,1,2);
a=gca();
a.y_location="origin";
a.x_location="origin";
```

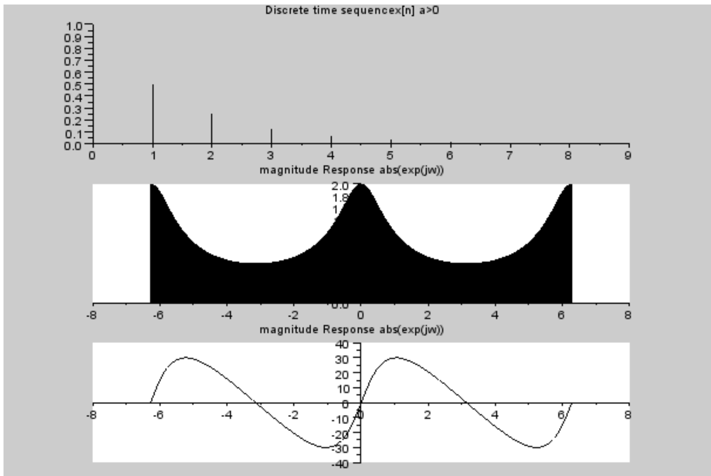

Figure 5.1: DTFT from Defining Relation

```
plot2d(W,XW2_Mag);
title('Magnitude Response abs(X(jw))')
subplot(3,1,3);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW2_phase);
title('phase Response<(X(jw))')
```

Scilab code Exa 5.3a Some DTFT pairs using properties

```
//DTFT of x[n]=n*(a)^n*u[n]
clear;
clc;close;
```

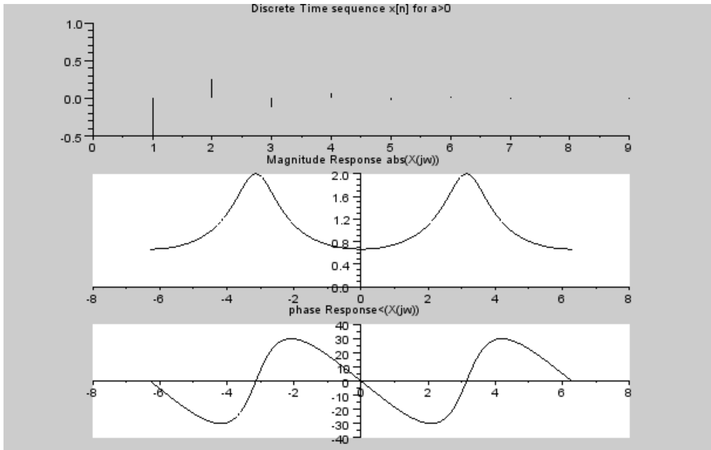

Figure 5.2: DTFT from Defining Relation

```
//DTS signal
a1=0.5;
a2=-0.5;
max_limit=10;
for n=0:max_limit-1
  x1(n+1)=n*(a1^n);
  x2(n+1)=n*(a2^n);
end
n=0:max_limit-1;
//discrete time fourier transform
wmax=2*%pi;
K=4;
k=0:(K/1000):K;
W=k*wmax/K;
x1=x1';
x2=x2';
XW1=x1*exp(%i*n'*W);
XW2=x2*exp(%i*n'*W);
XW1_Mag=abs(XW1);
XW2_Mag=abs(XW2);
```

```
W=[-mtlb_fliplr(W),W(2:1001)];//omega form
XW1_Mag=[mtlb_fliplr(XW1_Mag),XW1_Mag(2:1001)];
XW2_Mag=[mtlb_fliplr(XW2_Mag),XW2_Mag(2:1001)];
[XW1_phase,db]=phasemag(XW1);
[XW2_phase,db]=phasemag(XW2);
XW1_phase=[-mtlb_fliplr(XW1_phase),XW1_phase(2:1001)
   ];
XW2_phase=[-mtlb_fliplr(XW2_phase),XW2_phase(2:1001)
   ];

//plot for a>0
figure
subplot(3,1,1);
plot2d3('gnn',n,x1)
xtitle('Discrete time sequencex[n] a>0')
subplot(3,1,2);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d3(W,XW1_Mag);
title('magnitude Response abs(exp(jw))')
subplot(3,1,3);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW1_phase);
title('magnitude Response abs(exp(jw))')
//plot for a<0
figure
subplot(3,1,1);
plot2d3('gnn',n,x2);
xtitle('Discrete Time sequence x[n] for a>0')
subplot(3,1,2);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW2_Mag);
title('Magnitude Response abs(X(jw))')
```

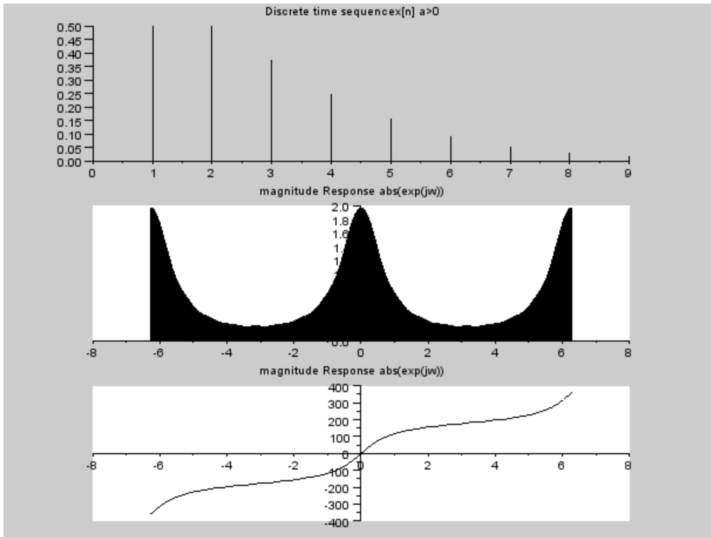

Figure 5.3: Some DTFT pairs using properties

```
subplot(3,1,3);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW2_phase);
title('phase Response<(X(jw))')
```

Scilab code Exa 5.3b Some DTFT pairs using properties

```
//DTFT of x[n]=n*(a)^n*u[n]
clear;
clc;close;
//DTS signal
a1=0.5;
```

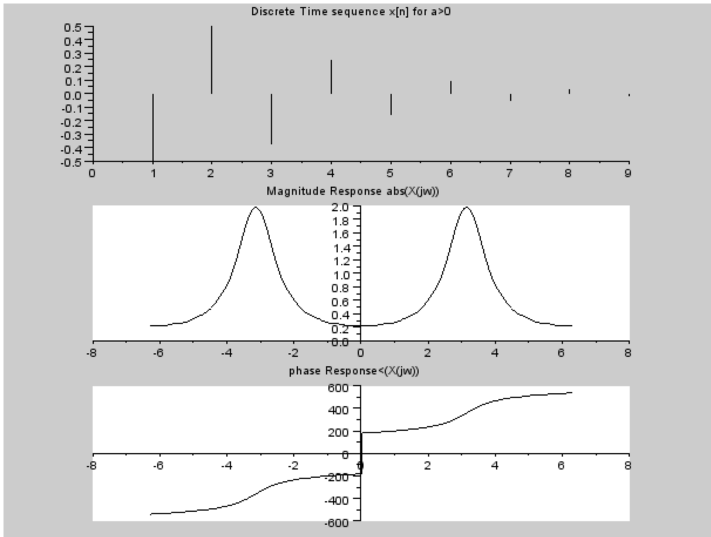

Figure 5.4: Some DTFT pairs using properties

```
a2=-0.5;
max_limit=10;
for n=0:max_limit-1
  x1(n+1)=(n+1)*(a1^n);
  x2(n+1)=(n+1)*(a2^n);
end
n=0:max_limit-1;
//discrete time fourier transform
wmax=2*%pi;
K=4;
k=0:(K/1000):K;
W=k*wmax/K;
x1=x1';
x2=x2';
XW1=x1*exp(%i*n'*W);
XW2=x2*exp(%i*n'*W);
XW1_Mag=abs(XW1);
XW2_Mag=abs(XW2);
W=[-mtlb_fliplr(W),W(2:1001)]; //omega form
XW1_Mag=[mtlb_fliplr(XW1_Mag),XW1_Mag(2:1001)];
```

```
XW2_Mag=[mtlb_fliplr(XW2_Mag),XW2_Mag(2:1001)];
[XW1_phase,db]=phasemag(XW1);
[XW2_phase,db]=phasemag(XW2);
XW1_phase=[-mtlb_fliplr(XW1_phase),XW1_phase(2:1001)
   ];
XW2_phase=[-mtlb_fliplr(XW2_phase),XW2_phase(2:1001)
   ];

//plot for a>0
figure
subplot(3,1,1);
plot2d3('gnn',n,x1)
xtitle('Discrete time sequencex[n] a>0')
subplot(3,1,2);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d3(W,XW1_Mag);
title('magnitude Response abs(exp(jw))')
subplot(3,1,3);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW1_phase);
title('magnitude Response abs(exp(jw))')
//plot for a<0
figure
subplot(3,1,1);
plot2d3('gnn',n,x2);
xtitle('Discrete Time sequence x[n] for a>0')
subplot(3,1,2);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW2_Mag);
title('Magnitude Response abs(X(jw))')
subplot(3,1,3);
a=gca();
```

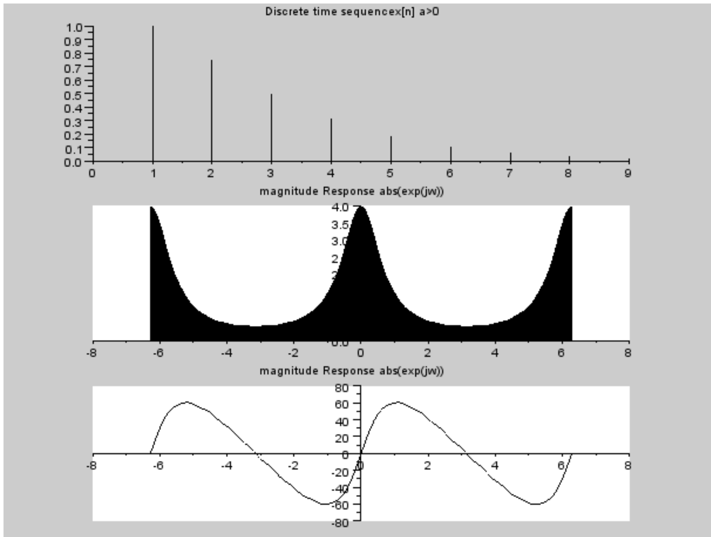

Figure 5.5: Some DTFT pairs using properties

```
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW2_phase);
title('phase Response<(X(jw))')
```

Scilab code Exa 5.3d Some DTFT pairs using properties

```
//DTFT of x[n]=a^abs(n)
a=0.5;
n=-9:9;
x=a^abs(n);
//Discrete time Fourier Transform
```

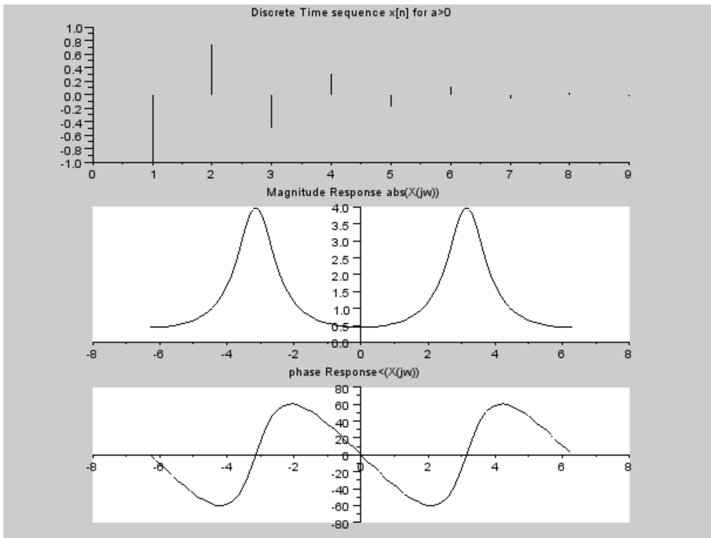

Figure 5.6: Some DTFT pairs using properties

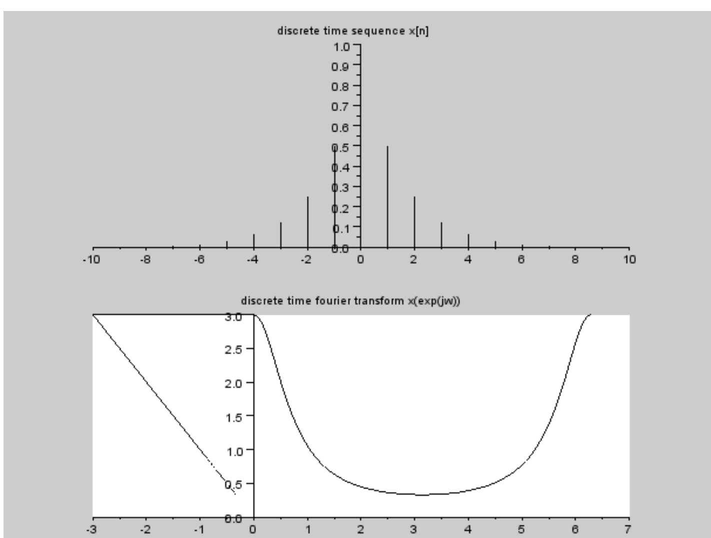

Figure 5.7: Some DTFT pairs using properties

```
k=0:(4/1000):4;
w=(2*%pi*k)/4;
xw=x*exp(%i*n'*w);
xw_mag=real(xw);
w=[-mtlb_fliplr(xw_mag),w(2:1001)];
xw_mag=[mtlb_fliplr(xw_mag),xw_mag(2:1001)];
figure
subplot(2,1,1);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d3('gnn',n,x);
xtitle('discrete time sequence x[n]');
subplot(2,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d(w,xw_mag);
title('discrete time fourier transform x(exp(jw))');
```

Scilab code Exa 5.3e Some DTFT pairs using properties

```
//DTFT of x[n]=n*(a)^n*u[n]
clear;
clc;close;
//DTS signal
a1=0.5;
a2=-0.5;
max_limit=10;
for n=0:max_limit-1
  x1(n+1)=4*(a1^(n+3));
  x2(n+1)=4*(a2^(n+3));
end
n=0:max_limit-1;
//discrete time fourier transform
```

```
wmax=2*%pi;
K=4;
k=0:(K/1000):K;
W=k*wmax/K;
x1=x1';
x2=x2';
XW1=x1*exp(%i*n'*W);
XW2=x2*exp(%i*n'*W);
XW1_Mag=abs(XW1);
XW2_Mag=abs(XW2);
W=[-mtlb_fliplr(W),W(2:1001)];//omega form
XW1_Mag=[mtlb_fliplr(XW1_Mag),XW1_Mag(2:1001)];
XW2_Mag=[mtlb_fliplr(XW2_Mag),XW2_Mag(2:1001)];
[XW1_phase,db]=phasemag(XW1);
[XW2_phase,db]=phasemag(XW2);
XW1_phase=[-mtlb_fliplr(XW1_phase),XW1_phase(2:1001)
   ];
XW2_phase=[-mtlb_fliplr(XW2_phase),XW2_phase(2:1001)
   ];

//plot for a>0
figure
subplot(3,1,1);
plot2d3('gnn',n,x1)
xtitle('Discrete time sequencex[n] a>0')
subplot(3,1,2);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d3(W,XW1_Mag);
title('magnitude Response abs(exp(jw))')
subplot(3,1,3);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW1_phase);
title('magnitude Response abs(exp(jw))')
//plot for a<0
```

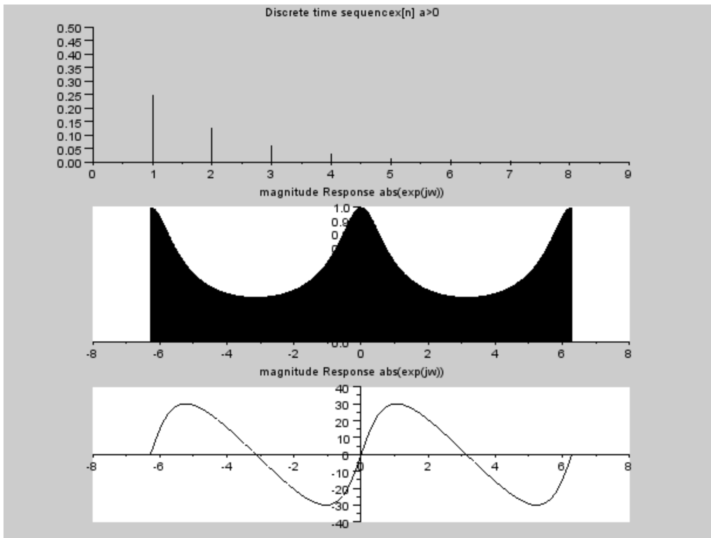

Figure 5.8: Some DTFT pairs using properties

```
figure
subplot(3,1,1);
plot2d3('gnn',n,x2);
xtitle('Discrete Time sequence x[n] for a>0')
subplot(3,1,2);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW2_Mag);
title('Magnitude Response abs(X(jw))')
subplot(3,1,3);
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d(W,XW2_phase);
title('phase Response <(X(jw))')
```

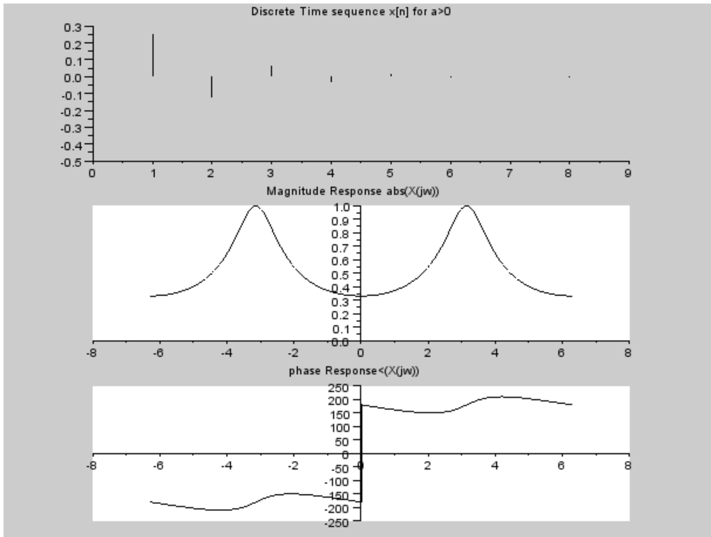

Figure 5.9: Some DTFT pairs using properties

Scilab code Exa 5.4 DTFT of periodic Signals

```
//DTfT of periodic signals
x=[3 2 1 2];//one period of signal
n=0:3;
k=0:3;
x1=x*exp(%i*n'*2*k*%pi/4)
dtftx=abs(x1)
x=[3 2 1 2 3 2 1 2 3];
n=-4:4;
a=gca();
a.y_location="origin";
a.x_location="origin";
plot2d3('gnn',n,x);
xtitle('discrete periodic time signal');
```

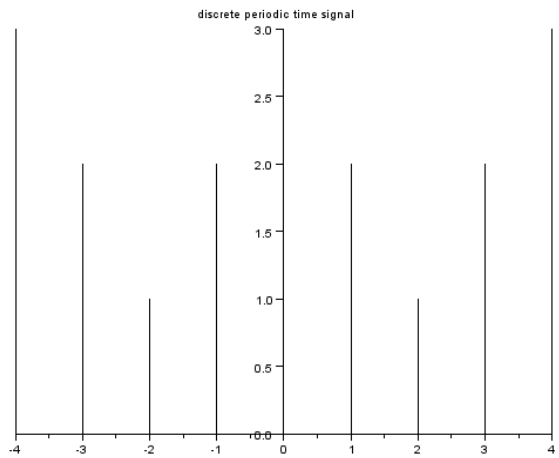

Figure 5.10: DTFT of periodic Signals

```
x2=[dtftx dtftx 8];
a=gca();
xset('window',1);
a.x_location="origin";
a.y_location="origin";
plot2d3('gnn',n,x2);
xtitle('DTFT of discrete periodic signal');
```

Scilab code Exa 5.5 The DFT,DFS and DTFT

```
x=[1 0 2 0 3];//one period of signal
n=0:4;
k=0:4;
x1=x*exp(%i*n'*2*k*%pi/4)
```

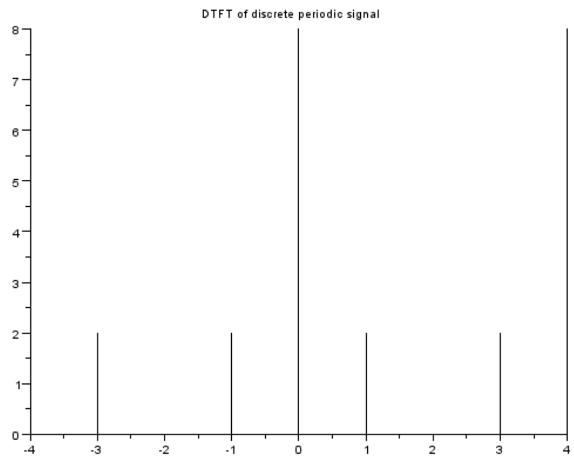

Figure 5.11: DTFT of periodic Signals

```
DTFTx=abs(x1)
DFT=dft(x,-1)
DFS=DFT/5
```

Scilab code Exa 5.7 Frequency Response of Recursive Filter

```
a=0.5;b=1;
n=0:50;
h=b*(a^n);
//Discrete-Time Fourier transform
K=500;
k=-250:1:250;
w=%pi*k/K;
H=h*exp(-%i*n'*w);
//caluculation of phase and magnitude of h(z)
[phase_H,m]=phasemag(H);
```

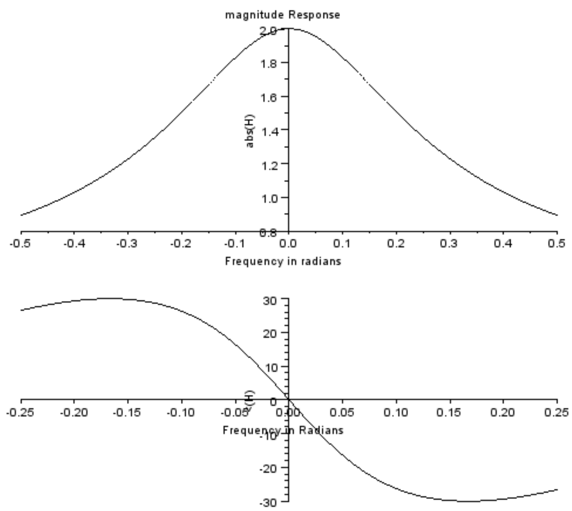

Figure 5.12: Frequency Response of Recursive Filter

```
H=abs(H);
a=gca();
subplot(2,1,1);
a.y_location="origin";
plot2d(w/%pi,H);
xlabel('Frequency in radians')
ylabel('abs(H)')
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'))
```

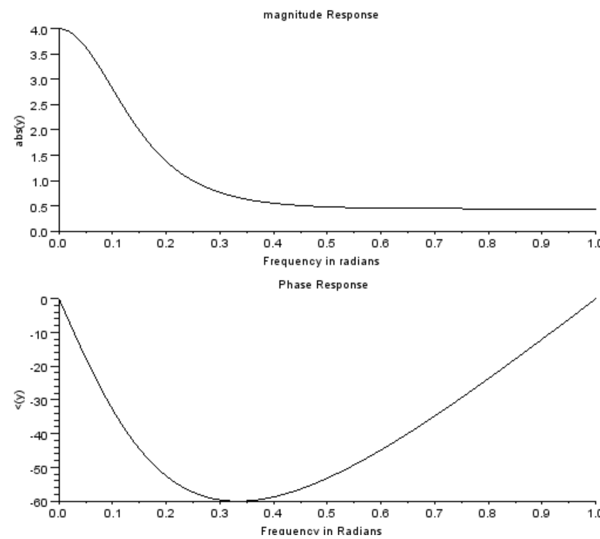

Figure 5.13: The DTFT in System Analysis

Scilab code Exa 5.8a The DTFT in System Analysis

```
//DTFT in system analysis
a=0.5;b=1;
n=0:50;
h=b*(a^n);
//Discrete-Time Fourier transform
K=500;
k=0:1:K;
w=%pi*k/K;
H=h*exp(-%i*n'*w);
//x[n] is given as (a)^n*u[n]
xw=h*exp(-%i*n'*w);
for i=1:501
    y(i)=H(i)*xw(i);
end
[phase_y,m]=phasemag(y);
```

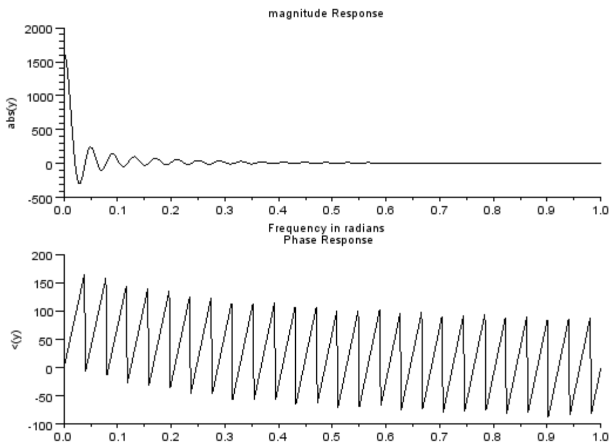

Figure 5.14: The DTFT in System Analysis

```
y=real(y);
subplot(2,1,1)
plot2d(w/%pi,y);
xlabel('Frequency in radians')
ylabel('abs(y)')
title('magnitude Response')
subplot(2,1,2)
plot2d(w/%pi,phase_y)
xlabel('Frequency in Radians');
ylabel('<(y)')
title('Phase Response')
```

Scilab code Exa 5.8b The DTFT in System Analysis

```
a=0.5;b=1;
n=0:50;
```

```
h=4*(a^n);
//Discrete-Time Fourier transform
K=500;
k=0:1:K;
w=%pi*k/K;
H=h*exp(-%i*n'*w);
//x[n] is given as (a)^n*u[n]
x=4*[ones(1:51)];
xw=x*exp(%i*n'*w);
for i=1:501
    y(i)=H(i)*xw(i);
end
[phase_y,m]=phasemag(y);
y=real(y);
subplot(2,1,1);
plot2d(w/%pi,y);
xlabel('Frequency in radians')
ylabel('abs(y)')
title('magnitude Response')
subplot(2,1,2)
plot2d(w/%pi,phase_y)
xlabel('Frequency in Radians');
ylabel('<(y)')
title('Phase Response')
```

Scilab code Exa 5.9a DTFT and steady state response

```
//DTFT and steady state response
a=0.5,b=1;F=0.25;
n=0:(5/1000):5;
h=(a^n);
x=10*cos(0.5*%pi*n'+%pi/3);
H=h*exp(-%i*n'*F);
```

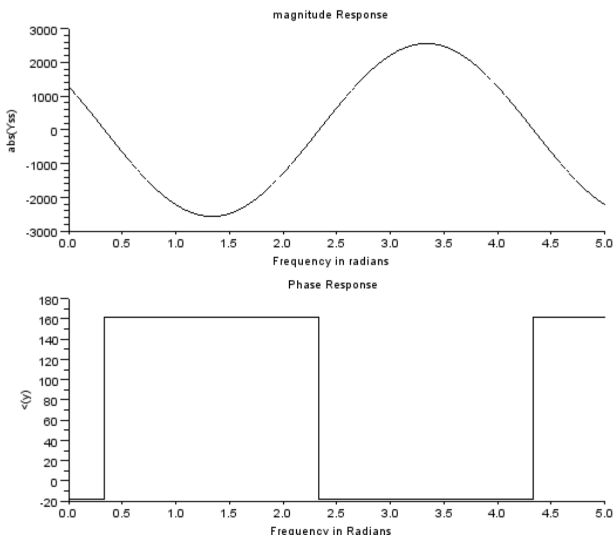

Figure 5.15: DTFT and steady state response

```
Yss=H*x;
[phase_Yss,m]=phasemag(Yss);
Yss=real(Yss);
subplot(2,1,1)
plot2d(n,Yss);
xlabel('Frequency in radians')
ylabel('abs(Yss)')
title('magnitude Response')
subplot(2,1,2)
plot2d(n,phase_Yss)
xlabel('Frequency in Radians');
ylabel('<(y)')
title('Phase Response')
```

Scilab code Exa 5.9b DTFT and steady state response

```
//DTFT and steady state response
a=0.8,b=1;F=0;
```

```
n=0:50;
h=(a^n);
x=4*[ones(1:10)];
H=h*exp(-%i*n'*F)
Yss=H*x
```

Scilab code Exa 5.10a System Representation in various forms

```
//System Representation in various forms
a=0.8;b=2;
n=0:50;
h=b*(a^n);
//Discrete-Time Fourier transform
K=500;
k=0:1:K;
w=%pi*k/K;
H=h*exp(-%i*n'*w);
//caluculation of phase and magnitude of h(z)
[phase_H,m]=phasemag(H);
H=abs(H);
subplot(2,1,1);
plot2d(w/%pi,H);
xlabel('Frequency in radians')
ylabel('abs(H)')
title('magnitude Response')
subplot(2,1,2)
plot2d(w/%pi,phase_H)
xlabel('Frequency in Radians');
ylabel('<(H)')
title('Phase Response')
```

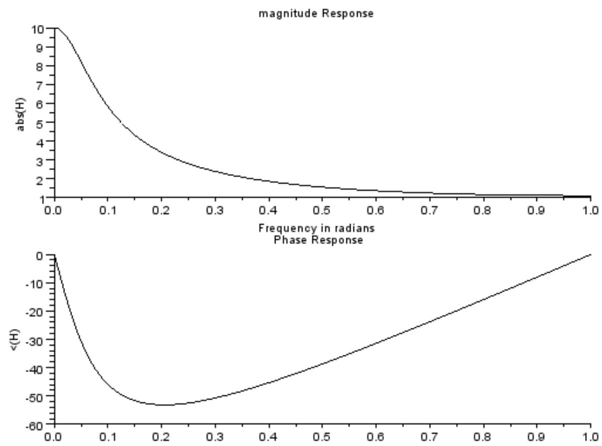

Figure 5.16: System Representation in various forms

Scilab code Exa 5.10b System Representation in various forms

```
//System Representation in various forms
a=0.6;b=1;
n=0:50;
h=b*(a^n);
//Discrete-Time Fourier transform
K=500;
k=0:1:K;
w=%pi*k/K;
H=h*exp(-%i*n'*w);
//caluculation of phase and magnitude of h(z)
[phase_H,m]=phasemag(H);
H=abs(H);
subplot(2,1,1);
plot2d(w/%pi,H);
xlabel('Frequency in radians')
ylabel('abs(H)')
title('magnitude Response')
subplot(2,1,2)
```

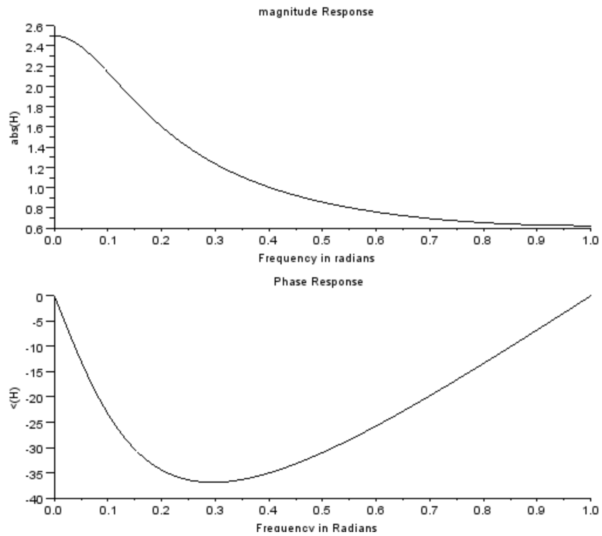

Figure 5.17: System Representation in various forms

```
plot2d(w/%pi,phase_H)
xlabel('Frequency in Radians');
ylabel('<(H)')
title('Phase Response')
```

Chapter 6

Filter Concepts

Scilab code Exa 6.1 The Minimum Phase Concept

```
//The minimum phase concept
z=%z;
F=0:(0.5/400):0.5;
z=exp(%i*2*%pi*F);
for i=1:401
H1Z(i)=((z(i)-1/2)*(z(i)-1/4))/((z(i)-1/3)*(z(i)
    -1/5));
end
for i=1:401
H2Z(i)=(((-1/2)*z(i)+1)*(z(i)-1/4))/((z(i)-1/3)*(z(i
    )-1/5));
end
for i=1:401
H3Z(i)=(((-1/2)*z(i)+1)*((-1/4)*z(i)+1))/((z(i)-1/3)
    *(z(i)-1/5));
end
[phase_H1Z,m]=phasemag(H1Z);
[phase_H2Z,m]=phasemag(H2Z);
[phase_H3Z,m]=phasemag(H3Z);
```

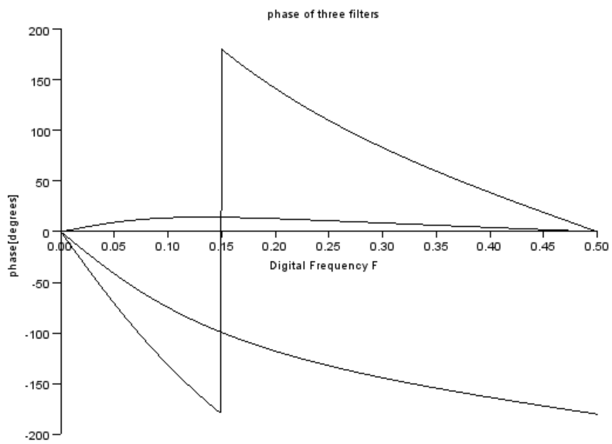

Figure 6.1: The Minimum Phase Concept

```
a=gca();
a.x_location="origin";
xlabel('Digital Frequency F');
ylabel('phase[degrees]');
xtitle('phase of three filters');
plot2d(F,phase_H1Z,rect=[0,-200,0.5,200]);
plot2d(F,phase_H2Z,rect=[0,-200,0.5,200]);
plot2d(F,phase_H3Z,rect=[0,-200,0.5,200]);
```

Scilab code Exa 6.4 Linear Phase Filters

```
//linear phase filters
z=%z
H1Z=((z^3)+2*(z^2)+2*z+1)/(z^3);
//from pole zero diagram its not a linear phase
   filter
H2Z=(z^4+4.25*z^2+1)/(z^4);
xset('window',1);
```

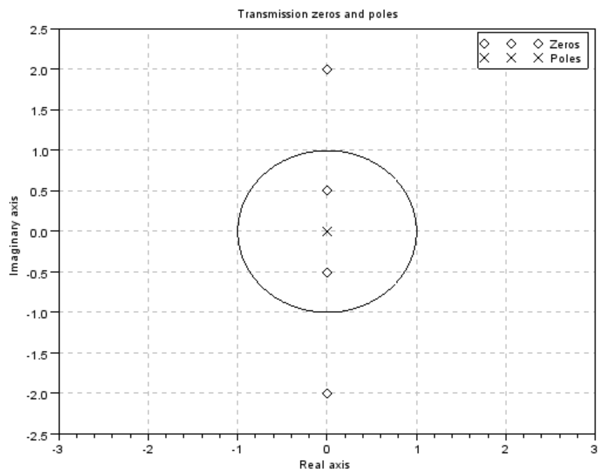

Figure 6.2: Linear Phase Filters

```
plzr(H2Z);
//from pole zero diagram and LPF
// characteristics its a linear phase filter
H3Z=((z^4+2.5*z^3-2.5*z-1)/(z^4));
xset('window',2);
plzr(H3Z);
//from pole zero diagram and LPF
// characteristics its a linear phase filter
```

Scilab code Exa 6.6 Frequency Response and Filter characteristics

```
//Frequency Response and filter characteristics
```

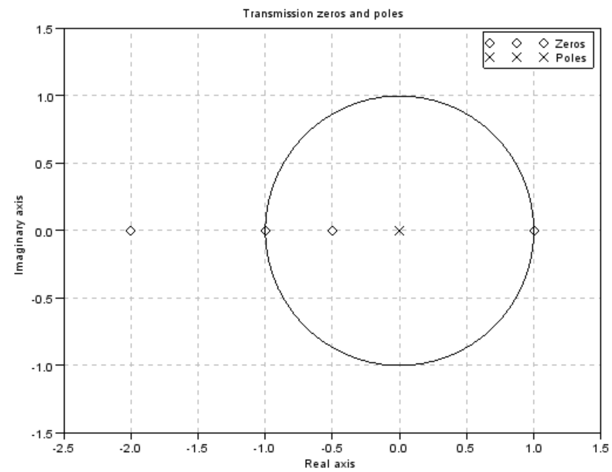

Figure 6.3: Linear Phase Filters

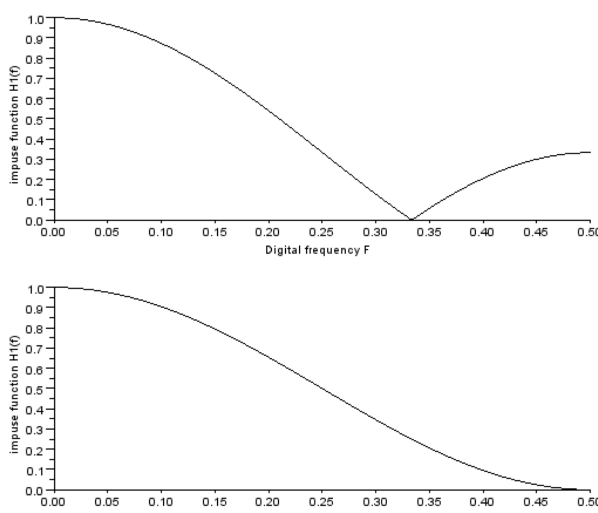

Figure 6.4: Frequency Response and Filter characteristics

```
z=%z;
F=0:(0.5/200):0.5;
z=exp(%i*2*%pi*F);
H1=(1/3)*(z+1+z^-1);
H2=(z/4)+(1/2)+(1/4)*(z^-1);
H1=abs(H1);
H2=abs(H2);
a=gca();
a.x_location="origin";
subplot(211);
plot2d(F,H1);
xlabel('Digital frequency F');
ylabel('impuse function H1(f)');
subplot(212);
plot2d(F,H2);
xlabel('Digital frequency F');
ylabel('impuse function H1(f)');
```

Scilab code Exa 6.7a Filters and Pole Zero Plots

```
z=%z;
s=%s;
F=0:(0.5/400):0.5;
s=exp(%i*2*%pi*F);
H1Z=(z^4+1)/(z^4+1.6982*z^2+0.7210);
for i=1:401
    H1(i)=(s(i)^4+1)/(s(i)^4+1.6982*s(i)^2+0.7210);
end
H1=abs(H1);
plzr(H1Z);
a=gca();
xset('window',1);
a.x_location="origin";
a.y_location="origin";
plot2d(F,H1)
```

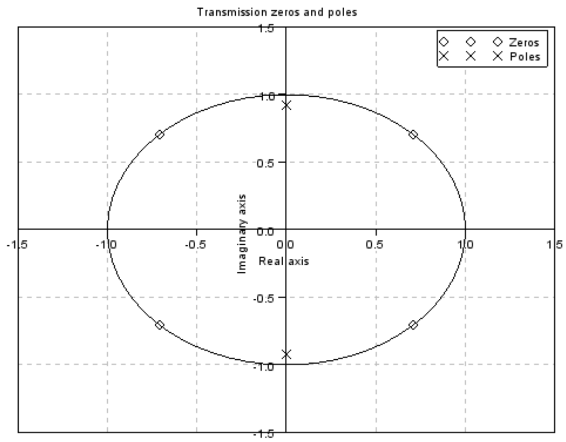

Figure 6.5: Filters and Pole Zero Plots

```
xlabel('Digital frequency F');
ylabel('magnitude');
xtitle('Magnitude spectrum of bandpass filter');
```

Scilab code Exa 6.7b Filters and Pole Zero Plots

```
z=%z;
s=%s;
F=0:(0.5/400):0.5;
s=exp(%i*2*%pi*F);
H1Z=(z^2+1-0.618*z)/(z^2-0.5857*z+0.898);
for i=1:401
    H1(i)=(s(i)^2+1-0.618*s(i))/(s(i)^2-0.5857*s(i)
       +0.898);
```

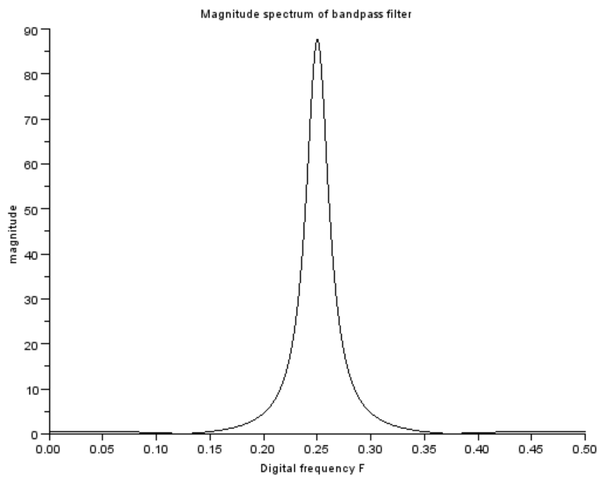

Figure 6.6: Filters and Pole Zero Plots

```
end
H1=abs(H1);
plzr(H1Z);
a=gca();
xset('window',1);
a.x_location="origin";
a.y_location="origin";
plot2d(F,H1)
xlabel('Digital frequency F');
ylabel('magnitude');
xtitle('Magnitude spectrum of bandpass filter');
```

Scilab code Exa 6.8 Digital resonator Design

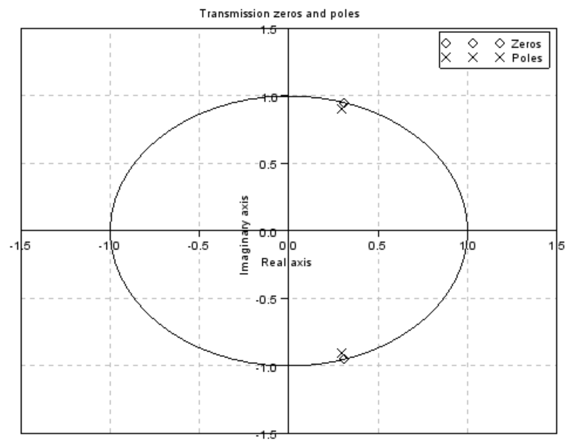

Figure 6.7: Filters and Pole Zero Plots

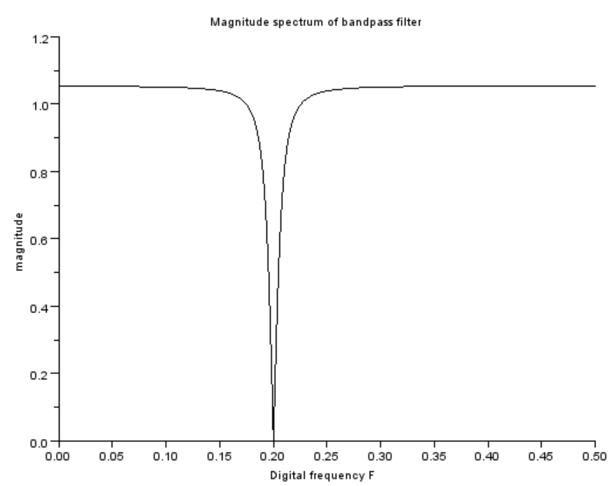

Figure 6.8: Filters and Pole Zero Plots

```
//Digital Resonator design with peak gain 50 HZ
//and 3 db bandwidth of 6HZ at sampling of 300 HZ
clf();
s=%s;
F=0:150;
f=F/300;
s=exp(%i*2*%pi*f);
for i=1:151
    H1(i)=(0.1054*(s(i)^2))/(s(i)^2-0.9372*s(i)
        +0.8783);
end
H1=abs(H1);
H2=H1(40:60);
F1=40:60;
f1=F1/300;
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d(F,H1)
xlabel('Analog frequency F');
ylabel('magnitude');
xtitle('Magnitude spectrum of digital resonator with
    peak 50HZ');
xset('window',1);
a.x_location="origin";
a.y_location="origin";
plot2d(F1,H2)
xlabel('Analog frequency F');
ylabel('magnitude');
xtitle('passband detail');
```

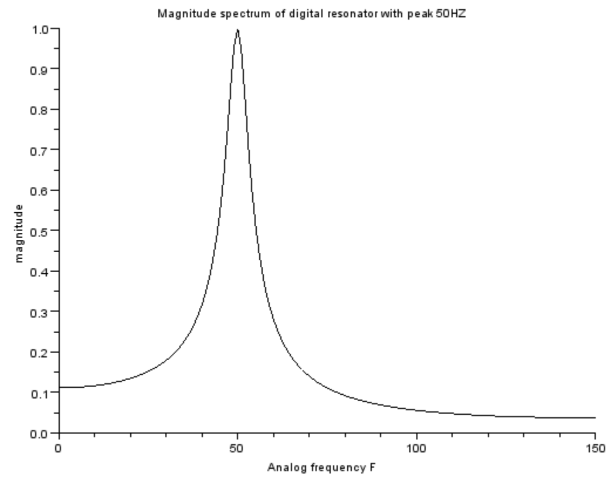

Figure 6.9: Digital resonator Design

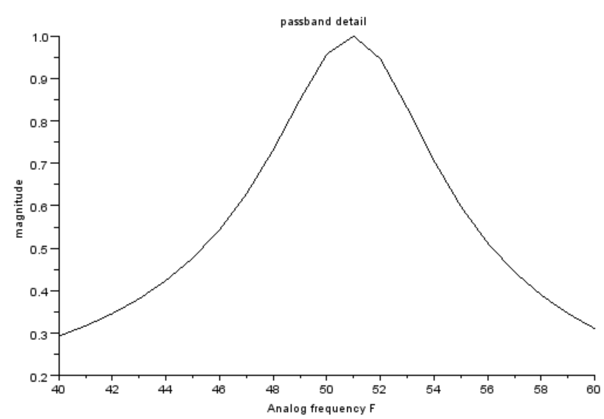

Figure 6.10: Digital resonator Design

Scilab code Exa 6.9 Periodic Notch Filter Design

```
//Periodic notch filter design at 60 HZ and sampling
    frequency 300HZ
z=%z;
f=0:(0.5/400):0.5;
z1=exp(%i*2*%pi*f);
for i=1:401
    H1Z(i)=(z1(i)^5-1)/((z1(i)^5)-(0.9^5));
    H2Z(i)=(z1(i)^5-1)/((z1(i)^5)-(0.99^5));
end
H1Z=abs(H1Z);
H2Z=abs(H2Z);
N1z=(1-z^-5)/(1-z^-1);
H3z=(N1z)/(horner(N1z,z/0.9));
H4z=(N1z)/(horner(N1z,z/0.99));
H3z=horner(H3z,z1);
H4z=horner(H4z,z1);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d(f,H1Z);
plot2d(f,H2Z);
xlabel('Digital frequency f');
ylabel('magnitude');
xtitle('Periodic Notch Filter N=5,R=0.9,0.99');
xset('window',1);
plot2d(f,H3z);
plot2d(f,H4z);
xlabel('Digital frequency f');
ylabel('magnitude');
xtitle('Notch Filter that also passes DC N=5,R
   =0.9,0.99');
```

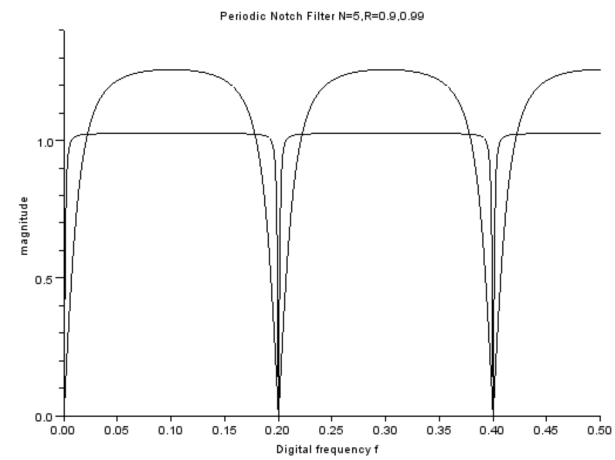

Figure 6.11: Periodic Notch Filter Design

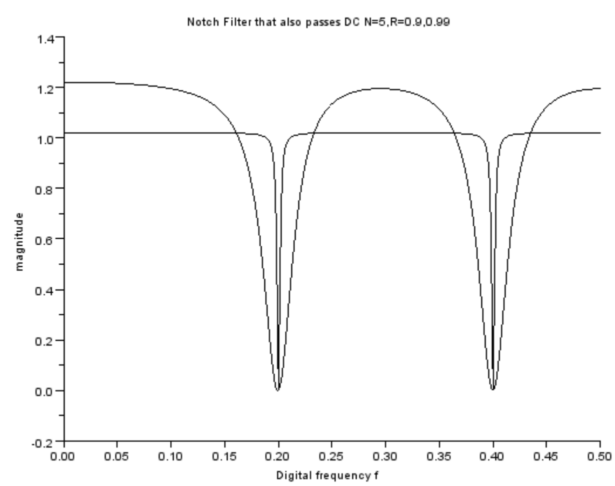

Figure 6.12: Periodic Notch Filter Design

Chapter 7

Digital Processing of Analog Signals

Scilab code Exa 7.3 Sampling oscilloscope

```
//Sampling Oscilloscope Concepts
fo=100;a=50;
s=(a-1)*fo/a;
B=100-s;
i=s/(2*B);
i=ceil(i);
disp(i,'The sampling frequency can at max divided by
    i');
disp(s,2*B,'range of sampling rate is between s and
   2*B');
fo1=100;
a=50;
s1=(a-1)*fo1/a;
B1=400-4*s1;
j=s1/(2*B1);
j=ceil(j);
disp(j,'The sampling frequency can at max divided by
    j');
disp(s1,2*B1,'range of sampling rate is between s1
```

```
and 2*B1');
```

Scilab code Exa 7.4 Sampling of Band pass signals

```
//sampling of bandpass signals
fc=4;fl=6;
B=fl-fc;
xt=[0 1 2 1];
xtt=[0 1 2];
a=0;b=1;c=2;
xta=[xt];
xtb=[0 0 2 1 0];
xtc=[0 0 0 2 1 0];
xt1=[xta xta xta];
xt2=[xtb xtb(length(xtb):-1:2) xtb(2:length(xtb))
   xtb(length(xtb):-1:2)];
xt3=[xtc(length(xtc):-1:2) xtc(3:length(xtc)) zeros
   (1:7) xtc(length(xtc):-1:2) xtc(3:length(xtc))];
f1=0:length(xt1)-1;
f2=[0 1 1.001 2:6 6.001 7 7.001 8:12 12.001];
f3=[-10:-8 -7.99 -7:-6 -5.99 -5:6 6.01 7:8 8.01
   9:10];
subplot(211);
plot2d(f1,xt1);
xtitle("spectrum of signal sampled at 4KHZ");
subplot(212);
plot2d(f2,xt2);
xtitle("spectrum of signal sampled at 7KHZ");
xset('window',1);
b=gca();
b.y_location="origin"
plot2d(f3,xt3);
xtitle("spectrum of signal sampled at 14KHZ");
```

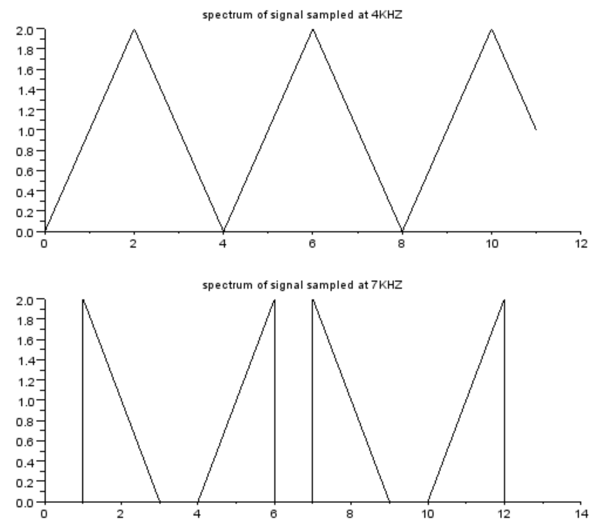

Figure 7.1: Sampling of Band pass signals

Scilab code Exa 7.6 Signal Reconstruction from Samples

```
//signal reconstruction from samples
//(a)By step interpolation method
x=[-1 2 3 2];
t=2.5;
ts=1;
t1=ceil(t);
t2=floor(t);
x1t=x(t2)
//(b)By linear interpolation method
x2t=(x(t1)+x(t2))/2
//(c)By sinc interpolation method
```

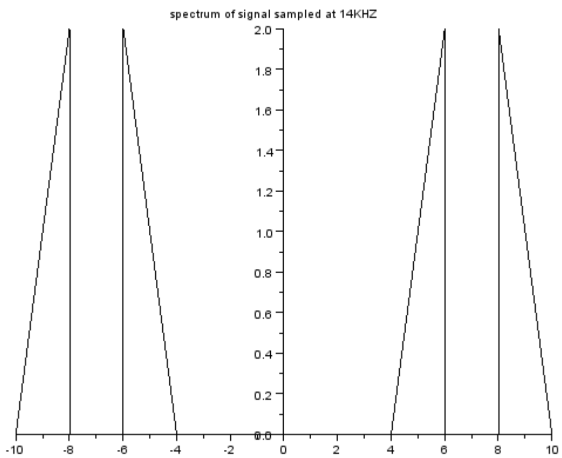

Figure 7.2: Sampling of Band pass signals

```
x3t=0;x1=[1 2 3 4];
for k=1:4
    x3t=x3t+(x1(k)*sinc(%pi*(t-(k-1))));
end
x3t//sinc interpolated value of x(2.5)
//(d)raised cosine interpolation method
x4t=0;
for k=1:4
    p=(cos(0.5*%pi*(t-k+1))/(1-(t-k+1)^2));
    xt=x1(k)*sinc(%pi*(t-k+1))*p;
    x4t=x4t+xt;
end
x4t//raised cosine interpolated value of x(2.5)
```

Scilab code Exa 7.7 Zero Interpolation and Spectrum Replication

```
//Zero interpolation and spectrum replication
XF=[0 1 2 1];
```

```
X1F=[XF XF XF 0];
YF=[X1F X1F];
DF=0.5*[XF XF 0];
GF=0.5*[XF 0 XF 0 XF 0];
f=-0.2:0.1:1;
f1=-0.1:0.05:1.15;
f2=-0.4:0.2:1.2;
f3=-0.2:0.1:1.2;
length(f3),length(GF)
a=gca();
a.y_location="origin";
subplot(211);
plot2d(f,X1F);
ylabel('X1F');
subplot(212);
a.y_location="origin";
plot2d(f1,YF);
ylabel('YF');
xset('window',1);
b=gca();
b.y_location="origin";
subplot(211);
plot2d(f2,DF);
ylabel('DF');
subplot(212);
b.y_location="origin";
plot2d(f3,GF);
ylabel('GF');
```

Scilab code Exa 7.8 Up Sampling and Filtering

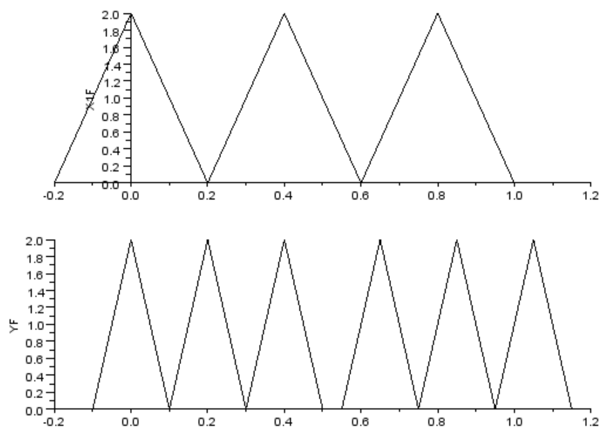

Figure 7.3: Zero Interpolation and Spectrum Replication

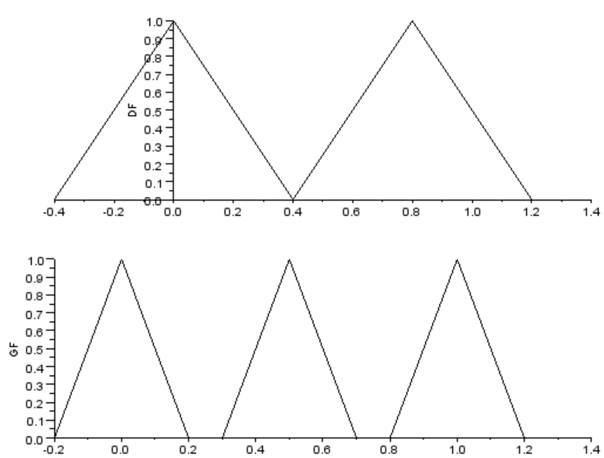

Figure 7.4: Zero Interpolation and Spectrum Replication

```
clf();
X=[0 0.5 1 0.5];
XF=[X 0];
WF=[X X X 0];
f=-0.5:0.25:0.5;
f1=-0.75:0.125:0.75;
HF=[0 1 1 1 0];
f2=[-0.126,-0.125:0.125:0.125,0.126];
for i=1:5
    YF(i)=WF(i)*HF(i);
end
f3=[-0.126 -0.125 0 0.125 0.126];
a=gca();
a.y_location="origin";
subplot(211);
plot2d(f,XF);
xtitle('spectrum of XF');
a.y_location="origin";
subplot(212);
plot2d(f1,WF);
xtitle('spectrum of WF');
xset('window',1);
b=gca();
b.y_location="origin";
subplot(211);
plot2d(f2,HF);
xtitle('spectrum of HF');
b.y_location="origin";
subplot(212);
plot2d(f3,YF);
xtitle('spectrum of YF');
```

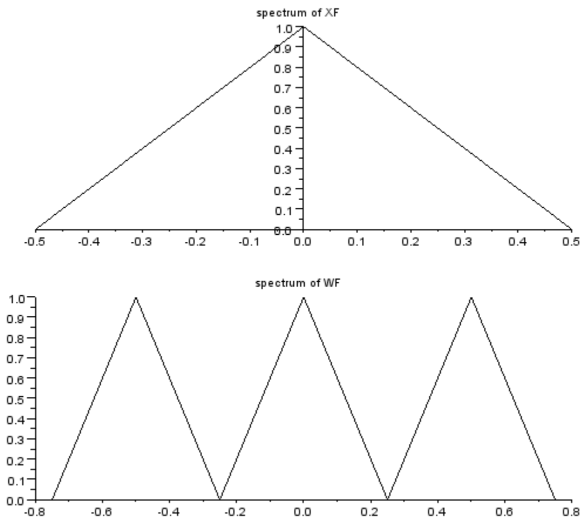

Figure 7.5: Up Sampling and Filtering

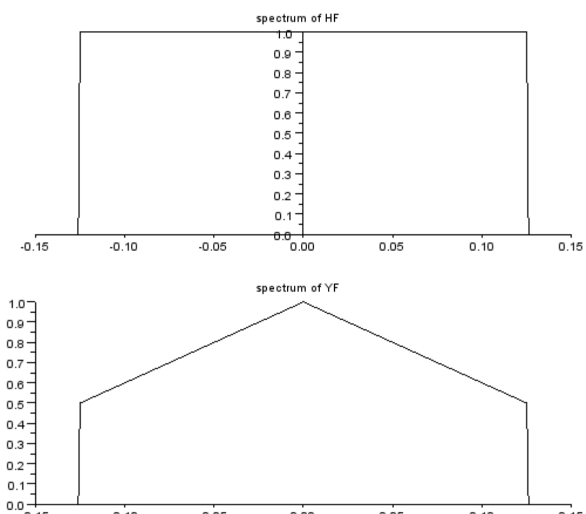

Figure 7.6: Up Sampling and Filtering

Scilab code Exa 7.9 Quantisation Effects

```
//(a)Quantisation effects
sig=0.005;
D=4;
B=log2(D/(sig*sqrt(12)));//no.of samples
//value of B to ensure quantisation error to 5mv
//(b)Quantisation error and noise
xn=0:0.2:2.0;
xqn=[0 0 0.5 0.5 1 1 1 1.5 1.5 2 2];
en=xn-xqn;//quantization error
//Quantisation signal top noise ratio
x=0;e=0;
for i=1:length(xn)
    x=x+xn(i)^2;
    e=e+en(i)^2;
end
//method 1
SNRQ=10*log10(x/e)
//method 2
SNRQ=10*log10(x/length(xn))+10.8+20*log10(4)-20*
    log10(2)
SNRS=10*log10(1.33)+10*log10(12)+20*log10(4)-20*
    log10(2)
//from results we see that SNRS is statistical
    estimate
```

Scilab code Exa 7.10 ADC considerations

```
//ADC considerations
//(a)Aperture time TA
B=12;
fo=15000;//band limited frquency
TAm=(1/((2^B)))/(%pi*fo);
TAm=TAm*10^9
```

```
//Hence TA must satisfy TA<=TAm nano sec
//(b)conversion time of quantizer
TA=4*10^-9;
TH=10*10^-6;//hold time
S=30*10^3;
TCm=1/S-TA-TH;
TCm=TCm*10^6
//Hence TC must satisfy TC<=TCm micro sec
//(c)Holding capacitance C
Vo=10;
TH=10*10^-6;
B=12;
R=10^6;//input resistance
delv=Vo/(2^(B+1));
Cm=(Vo*TH)/(R*delv);
Cm=Cm*10^9
//Hence C must satisfy C>=Cm nano farad
```

Scilab code Exa 7.11 Anti Aliasing Filter Considerations

```
//Anti Aliasing filter considerations
//minimum stop band attenuation As
B=input('enter no. of bits');//no. of samples
n=input('enter band width in KHZ');
As=20*log10(2^B*sqrt(6))
//nomalised frequency
Vs=(10^(0.1*As)-1)^(1/(2*n))
fp=4;//pass edge frequency
fs=Vs*fp//stop band frquency
S=2*fs//sampling frequency
fa=S-fp//aliaed frequency
Va=fa/fp;
//Attenuation at aliased frequency
Aa=10*log10(1+Va^(2*n))
```

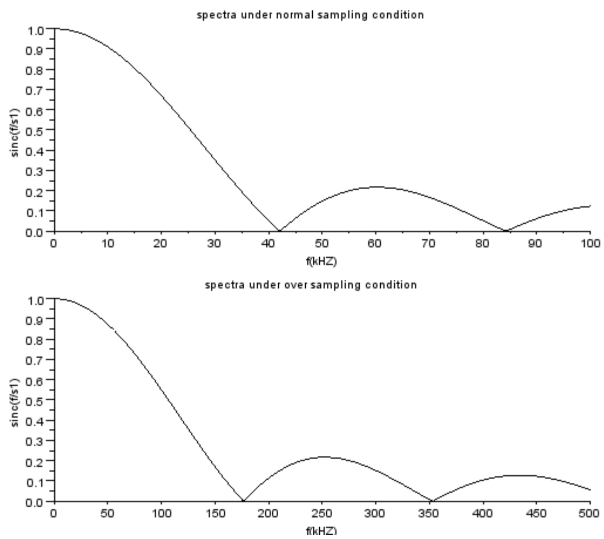

Figure 7.7: Anti Imaging Filter Considerations

Scilab code Exa 7.12 Anti Imaging Filter Considerations

```
//Anti Imaging Filter considerations
Ap=0.5;//passband attenuation
fp=20;//passband edge frequency
As=60;//stopband attenuation
S=42.1;
fs=S-fp;//stopband edge frequency
e=sqrt(10^(0.1*Ap)-1);
e1=sqrt(10^(0.1*As)-1);
n=(log10(e1/e))/(log10(fs/fp));
n=ceil(n)//design of nth order butworth filter
//(b)Assuming Zero-order hold sampling
S1=176.4;
fs1=S1-fp;
```

```
Ap=0.316;
e2=sqrt(10^(0.1*Ap)-1);
n1=(log10(e1/e2))/(log(fs1/fp));//new order of
    butworth filter
n1=ceil(n1)
f=0:100;
x=abs(sinc(f*%pi/S));
f1=0:500;
x1=abs(sinc(f1*%pi/S1));
a=gca();
subplot(211);
plot2d(f,x);
xtitle("spectra under normal sampling condition","f(
    kHZ)","sinc(f/s1)");
subplot(212);
plot2d(f1,x1);
xtitle("spectra under over sampling condition","f(
    kHZ)","sinc(f/s1)");
```

Chapter 8

The Discrete Fourier Transform and its Applications

Scilab code Exa 8.1 DFT from Defining Relation

```
//DFT from defining relation
//N-point DFT
x=[1 2 1 0];
XDFT=dft(x,-1);
disp(XDFT,'The DFT of x[n] is');
//DFT of periodic signal x with period N=4
```

Scilab code Exa 8.2 The DFT and conjugate Symmetry

```
//The DTFT and conjugate symmetry
//8-point DFT
x=[1 1 0 0 0 0 0 0];
XDFT=dft(x,-1);
disp(XDFT,'The DFT of x is');
disp('from conjugate symmetry we see XDFT[k]=XDFT[8-
   k]');
```

Scilab code Exa 8.3 Circular Shift and Flipping

```
//Circular shift and flipping
//(a)right circular shift
y=[1 2 3 4 5 0 0 6];
f=y;g=y;h=y;
for i=1:2
    b=f(length(f));
    for j=length(f):-1:2
        f(j)=f(j-1);
     end
     f(1)=b;
end
disp(f,'By right circular shift y[n-2] is');
//(b)left circular shift
for i=1:2
    a=g(1);
    for j=1:length(g)-1
        g(j)=g(j+1);
     end
     g(length(g))=a;
end
disp(g,'By left circular shift y[n+2] is');
//(c)flipping property
h=[h(1) h(length(h):-1:2)];
disp(h,'By flipping property y[-n] is');
```

Scilab code Exa 8.4 Properties of DFT

```
x=[1 2 1 0];
XDFT=dft(x,-1)
//(a)time shift property
```

```
y=x;
for i=1:2
    a=y(1);
    for j=1:length(y)-1
        y(j)=y(j+1);
     end
     y(length(y))=a;
end
YDFT=dft(y,-1)
disp(YDFT,'By Time-Shift property DFT of x[n-2] is');
//(b)flipping property
g=[x(1) x(length(x):-1:2)]
GDFT=dft(g,-1)
disp(GDFT,'By Time reversal property DFT of x[-n] is');
//(c)conjugation property
p=XDFT;
PDFT=[p(1);p(4:-1:2)];
disp(YDFT,'BY conjugation property DFT of x*[n] is');
```

Scilab code Exa 8.5a Properties of DFT

```
//properties of DFT
//a1)product
xn=[1 2 1 0];
XDFT=dft(xn,-1)
hn=xn.*xn
HDFT=dft(hn,-1)
HDFT1=1/4*(convol(XDFT,XDFT))
HDFT1=[HDFT1,zeros(8:12)];
HDFT2=[HDFT1(1:4);HDFT1(5:8);HDFT1(9:12)];
HDFT3=[0 0 0 0];
for i=1:4
```

```
    for j=1:3
        HDFT3(i)=HDFT3(i)+HDFT2(j,i);
    end
end
disp(HDFT3,'DFT of x[n]^2 is');
//a2)periodic convolution
vn=convol(xn,xn);
vn=[vn,zeros(8:12)];
vn=[vn(1:4);vn(5:8);vn(9:12)];
vn1=[0 0 0 0];
for i=1:4
    for j=1:3
        vn1(i)=vn1(i)+vn(j,i);
    end
end
VDFT=dft(vn1,-1);
VDFT1=XDFT.*XDFT;
disp(VDFT1,'DFT of x[n]*x[n] is');
//a3)signal energy(parcewell's theorem)
xn2=xn^2;
E=0;
for i=1:length(xn2)
    E=E+abs(xn2(i));
end
XDFT2=XDFT^2
E1=0;
for i=1:length(XDFT2)
    E1=E1+abs(XDFT2(i));
end
E,(1/4)*E1;
disp(1/4*E1,'The energy of the signal is');
```

Scilab code Exa 8.5b Properties of DFT

```
//b1)modulation
```

```
XDFT=[4 -2*%i 0 2*%i];
xn=dft(XDFT,1)
for i=1:length(xn)
    zn(i)=xn(i)*%e^((%i*%pi*(i-1))/2);
end
disp(zn,'The IDFT of XDFT[k-1] is');
ZDFT=[2*%i 4 -2*%i 0];
zn1=dft(ZDFT,1)
//b2)periodic convolution
HDFT=(convol(XDFT,XDFT))
HDFT=[HDFT,zeros(8:12)];
HDFT=[HDFT(1:4);HDFT(5:8);HDFT(9:12)];
HDFT1=[0 0 0 0];
for i=1:4
    for j=1:3
        HDFT1(i)=HDFT1(i)+HDFT(j,i);
    end
end
HDFT1;
hn=dft(HDFT1,1)
hn1=4*(xn.*xn);
disp(hn1,'The IDFT of XDFT*XDFT is');
//b3)product
WDFT=XDFT.*XDFT;
wn=dft(WDFT,1)
wn1=convol(xn,xn);
wn1=[wn1,zeros(8:12)];
wn1=[wn1(1:4);wn1(5:8);wn1(9:12)];
WN=[0 0 0 0];
for i=1:4
    for j=1:3
        WN(i)=WN(i)+wn1(j,i);
    end
end
disp(WN,'The IDFT of XDFT.XDFT is');
//b4)Central ordinates and signal Energy
E=0;
for i=1:length(xn)
```

```
    E=E+abs(xn(i)^2);
end
disp(E,'the signal energy is');
```

Scilab code Exa 8.5c Properties of DFT

```
//Regular convolution
xn=[1 2 1 0];
yn=[1 2 1 0 0 0 0];
YDFT=dft(yn,-1)
SDFT=YDFT.*YDFT
sn=dft(SDFT,1)
sn1=convol(xn,xn)
```

Scilab code Exa 8.6 Signal and Spectrum Replication

```
//Signal and spectrum replication
xn=[2 3 2 1];
XDFT=dft(xn,-1)
yn=[xn xn xn];
YDFT=dft(yn,-1)
YDFT1=3*[XDFT(1:1/3:length(XDFT))];
for i=2:3
    YDFT1(i:3:length(YDFT1))=0;
end
YDFT1(12:-1:11)=0;
disp(YDFT1,'the DFT of x[n/3] is');
hn=[xn(1:1/3:length(xn))]
for i=2:3
    hn(i:3:length(hn))=0;
end
hn(12:-1:11)=0;
hn
```

```
HDFT=dft(hn,-1)
HDFT1=[XDFT;XDFT;XDFT];
disp(HDFT1,'the DFT of y[n]=[x[n],x[n],x[n]] is');
```

Scilab code Exa 8.7 Relating DFT and DTFT

```
//relating DFT and IDFT
XDFT1=[4 -2*%i 0 2*%i];
xn1=dft(XDFT1,1);
disp(xn1,'The IDFT of XDFT1');
XDFT2=[12 -24*%i 0 4*%e^(%i*%pi/4) 0 4*%e^(-%i*%pi/4) 0 24*%i];
xn2=dft(XDFT2,1);
disp(xn2,'The IDFT of XDFT1');
```

Scilab code Exa 8.8 Relating DFT and DTFT

```
//Relating DFT and DTFT
xn=[1 2 1 0];
XDFT=dft(xn,-1);
//for F=k/4,k=0,1,2,3
for k=1:4
    XF(k)=1+2*%e^(-%i*%pi*(k-1)/2)+%e^(-%i*%pi*(k-1));
end
XF,XDFT
disp(XF,'The DFT of x[n] is');
```

Scilab code Exa 8.9a The DFT and DFS of sinusoids

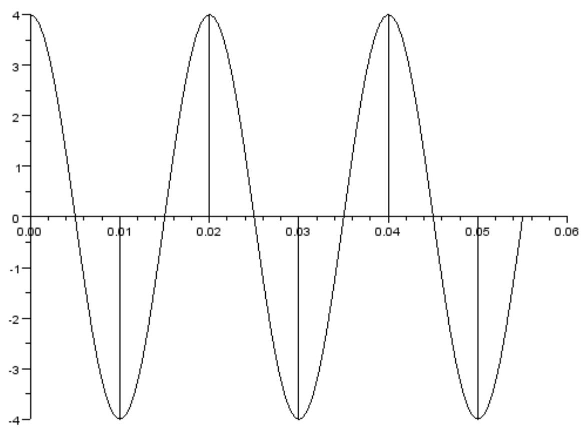

Figure 8.1: The DFT and DFS of sinusoids

```
//DFT and DFS of sinusoids
n2=0:0.5/1000:5.5/100;
xt=4*cos(100*%pi*n2');
n=0:(0.5)/100:(5.5)/100;//F=3/12 hence N=12
xn=4*cos(100*%pi*n');
XDFT=dft(xn,-1);
n1=0:11;
a=gca();
a.x_location="origin";
plot2d(n2,xt);
plot2d3('gnn',n,xn);
xset('window',1);
b=gca();
b.x_location="origin";
plot2d3('gnn',n1,XDFT);
```

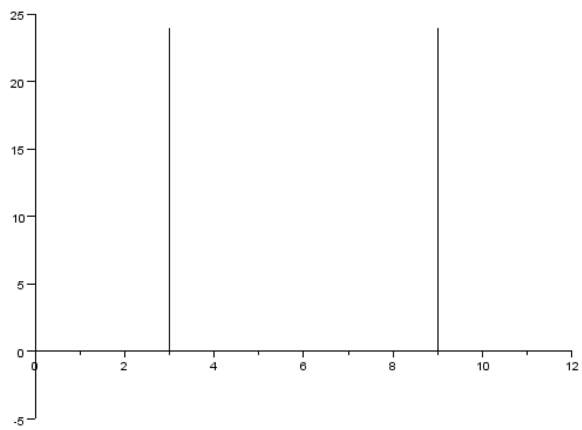

Figure 8.2: The DFT and DFS of sinusoids

Scilab code Exa 8.9b The DFT and DFS of sinusoids

```
//DFT and DFS of sinusoids
n2=0:1/1280:31/128;
xt=4*sin(72*%pi*n2');
n=0:1/128:31/128;//F=9/32 hence N=32
xn=4*sin(72*%pi*n');
XDFT=abs(dft(xn,-1));
n1=0:31;
a=gca();
a.x_location="origin";
plot2d(n2,xt);
plot2d3('gnn',n,xn);
xset('window',1);
b=gca();
b.x_location="origin";
plot2d3('gnn',n1,XDFT);
```

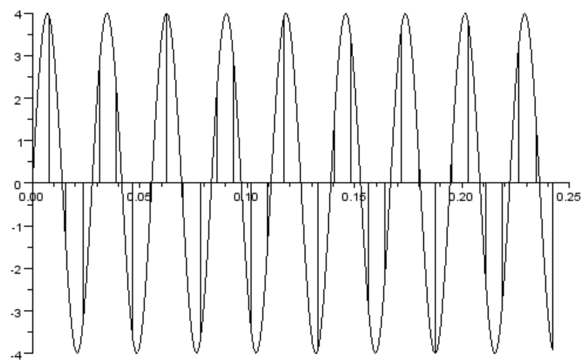

Figure 8.3: The DFT and DFS of sinusoids

Scilab code Exa 8.9c The DFT and DFS of sinusoids

```
//DFT and DFS of sinusoids
n2=0:1/840:6/21;
xt=4*sin(72*%pi*n2')-6*cos(12*%pi*n2');
n=0:1/21:6/21;//F=3/12 hence N=12
xn=4*sin(72*%pi*n')-6*cos(12*%pi*n');
XDFT=abs(dft(xn,-1));
n1=0:6;
a=gca();
a.x_location="origin";
plot2d(n2,xt);
plot2d3('gnn',n,xn);
```

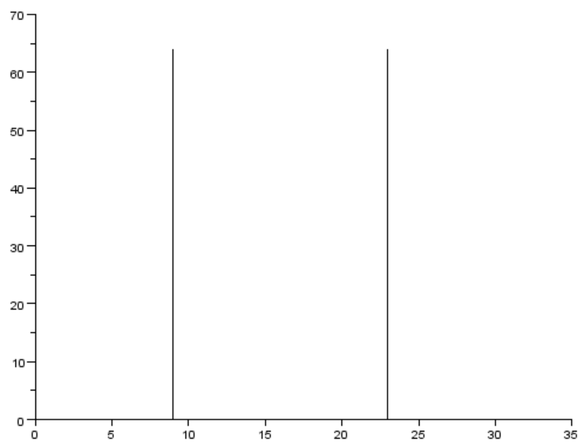

Figure 8.4: The DFT and DFS of sinusoids

```
xset('window',1);
b=gca();
b.x_location="origin";
plot2d3('gnn',n1,XDFT);
```

Scilab code Exa 8.9d The DFT and DFS of sinusoids

```
//DFT and DFS of sinusoids
n2=0:1/2400:23/240;
xt=1+4*sin(120*%pi*n2')+4*sin(40*%pi*n2');
n=0:1/240:23/240;//F=9/32 hence N=32
xn=1+4*sin(120*%pi*n')+4*sin(40*%pi*n');
XDFT=abs(dft(xn,-1));
n1=0:23;
```

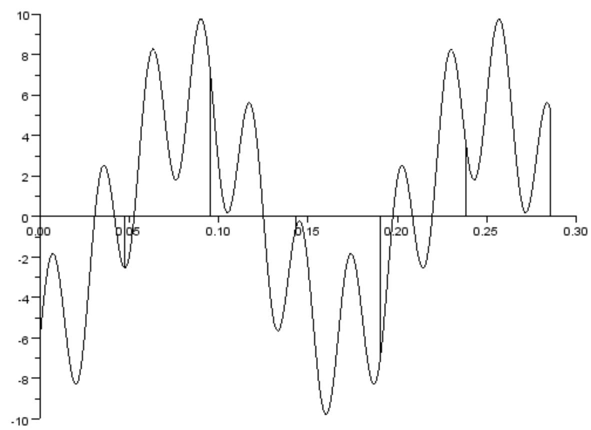

Figure 8.5: The DFT and DFS of sinusoids

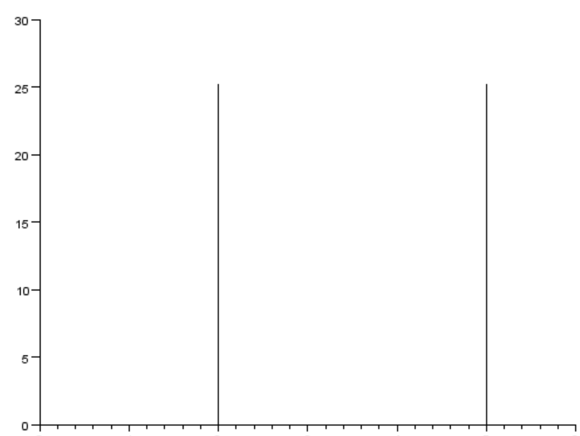

Figure 8.6: The DFT and DFS of sinusoids

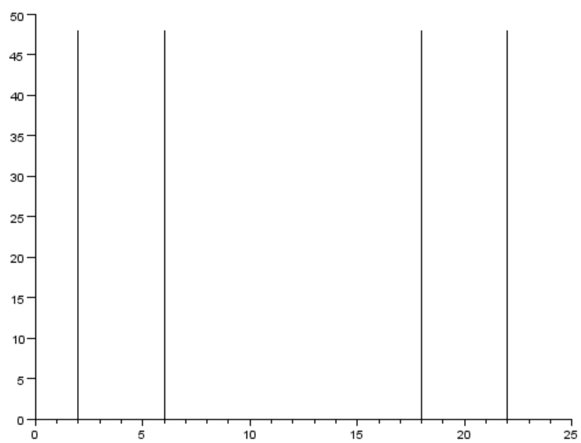

Figure 8.7: The DFT and DFS of sinusoids

```
a=gca();
a.x_location="origin";
plot2d(n2,xt);
plot2d3('gnn',n,xn);
xset('window',1);
b=gca();
b.x_location="origin";
plot2d3('gnn',n1,XDFT);
```

Scilab code Exa 8.10 DFS of sampled Periodic Signals

```
//DFS of sampled periodic signals
xn=[0 ones(2:16) 0 -ones(18:32)];
XDFS=0.032*dft(xn,-1);
```

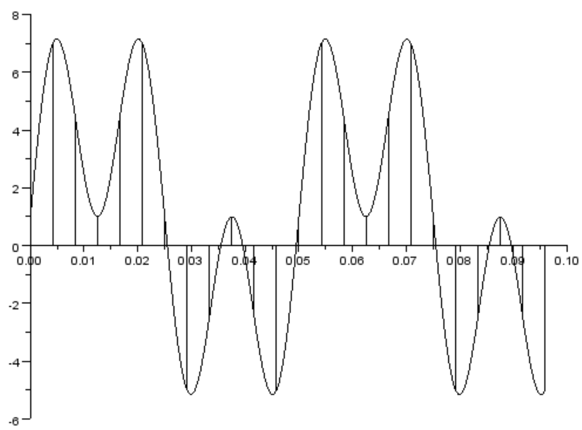

Figure 8.8: The DFT and DFS of sinusoids

```
for i=1:length(XDFS)
    if (abs(XDFS(i))<0.000001) then
        XDFS(i)=0;
    end
end
disp(XDFS,'The DFS of x[n] is');
```

Scilab code Exa 8.11 The effects of leakage

```
//Effects of leakage
n1=0:0.005:0.1;
n2=0:0.005:0.125;
n3=0:0.005:1.125;
xt1=(2*cos(20*%pi*n1')+5*cos(100*%pi*n1'));
xt2=(2*cos(20*%pi*n2')+5*cos(100*%pi*n2'));
xt3=(2*cos(20*%pi*n3')+5*cos(100*%pi*n3'));
XDFS1=abs(dft(xt1,-1))/20;
XDFS2=abs(dft(xt2,-1))/25;
```

```
XDFS3=abs(dft(xt3,-1))/225;
f1=0:5:100;
f2=0:4:100;
f3=0:100/225:100;
a=gca();
a.x_location="origin";
plot2d3('gnn',f1,XDFS1);
xlabel('analog frequency');
ylabel('Magnitude');
xset('window',1);
subplot(211);
plot2d3('gnn',f2,XDFS2);
xlabel('analog frequency');
ylabel('Magnitude');
subplot(212);
plot2d3('gnn',f3,XDFS3);
xlabel('analog frequency');
ylabel('Magnitude');
```

Scilab code Exa 8.15a Methods to find convolution

```
//overlapp-add and overlap-save methods of
   convolution
//overlap-add method
xn=[1 2 3 3 4 5];
xon=[1 2 3];
x1n=[3 4 5];
hn=[1 1 1];
yon=convol(xon,hn);
y1n=convol(x1n,hn);
yon=[yon,0,0,0];
```

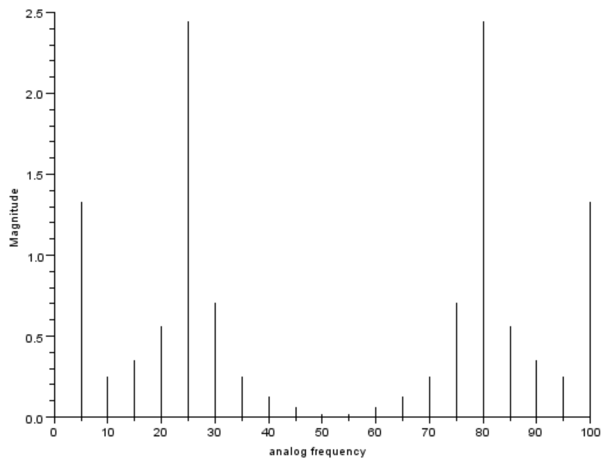

Figure 8.9: The effects of leakage

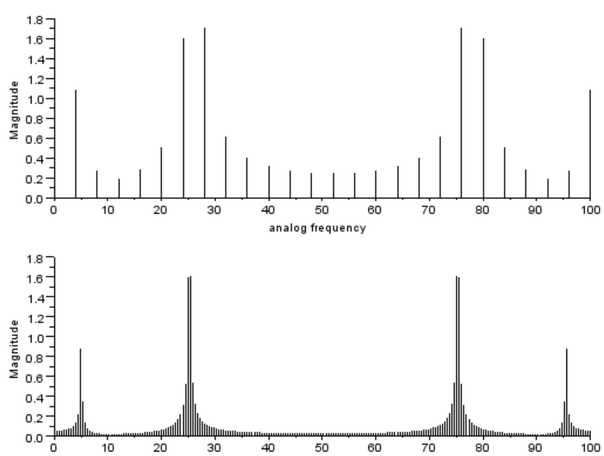

Figure 8.10: The effects of leakage

```
y1n=[0,0,0,y1n];
yn=yon+y1n
yn1=convol(xn,hn)
```

Scilab code Exa 8.15b Methods to find convolution

```
//(b)overlap-save method
xn=[1 2 3 3 4 5];
hn=[1 1 1];
xon=[0 0 1 2 3];
x1n=[2 3 3 4 5];
x2n=[4 5 0 0 0];
yon=convol(xon,hn);
y1n=convol(x1n,hn);
y2n=convol(x2n,hn);
yno=yon(3:5);
yn1=y1n(3:5);
yn2=y2n(3:5);
yn=[yno yn1 yn2]
YN=convol(xn,hn)
```

Scilab code Exa 8.16 Signal Interpolation using FFT

```
//signal interpolation using FFT
xn=[0 1 0 -1];
XDFT=dft(xn,-1)
ZT=[0 -2*%i 0 zeros(1:27) 0 2*%i];
xn1=dft(ZT,1);
t=0:1/length(xn1):1-(1/length(xn1));
a=gca();
a.x_location="origin";
```

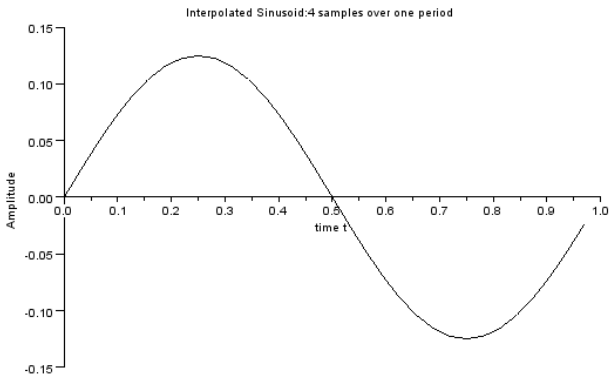

Figure 8.11: Signal Interpolation using FFT

```
plot2d(t,xn1);
xlabel('time t');
ylabel('Amplitude');
xtitle('Interpolated Sinusoid:4 samples over one
   period');
```

Scilab code Exa 8.17 The Concept of Periodogram

```
//concept of periodogram
xn=[0 1 0 -1];
N=4;
XDFT=dft(xn,-1);
for i=1:length(XDFT)
    p(i)=(1/N)*abs(XDFT(i)^2);
end
p//periodogram
```

Scilab code Exa 8.18 DFT from matrix formulation

```
//The DFT from the matrix formulation
xn=[1;2;1;0];
w=exp(-%i*%pi/2);
for i=1:4
    for j=1:4
        WN(i,j)=w^((i-1)*(j-1));
    end
end
XDFT=WN*xn
```

Scilab code Exa 8.19 Using DFT to find IDFT

```
//using DFT to find IDFT
XDFT=[4;-2*%i;0;2*%i];
XDFTc=[4;2*%i;0;-2*%i];
w=exp(-%i*%pi/2);
for i=1:4
    for j=1:4
        WN(i,j)=w^((i-1)*(j-1));
    end
end
xn=1/4*(WN*XDFTc)
```

Scilab code Exa 8.20 Decimation in Frequency FFT algorithm

```
//A four point decimation-in-frequency FFT algorithm
x=[1 2 1 0];
```

```
w=-%i;
xdft(1)=x(1)+x(3)+x(2)+x(4);
xdft(2)=x(1)-x(3)+w*(x(2)-x(4));
xdft(3)=x(1)+x(3)-x(2)-x(4);
xdft(4)=x(1)-x(3)-w*(x(2)-x(4));
XDFT=dft(x,-1);
xdft,XDFT
```

Scilab code Exa 8.21 Decimation in time FFT algorithm

```
//A four point decimation-in-time FFT algorithm
x=[1 2 1 0];
w=-%i;
xdft=[0 0 0 0];
for i=1:4
    for j=1:4
        xdft(i)=xdft(i)+x(j)*w^((i-1)*(j-1));
    end
end
XDFT=dft(x,-1);
xdft,XDFT
```

Scilab code Exa 8.22 4 point DFT from 3 point sequence

```
//A 4-point DFT from a 3-point sequence
xn=[1;2;1];
w=exp(-%i*%pi/2);
for i=1:4
    for j=1:3
        WN(i,j)=w^((i-1)*(j-1));
    end
end
XDFT=WN*xn
```

Scilab code Exa 8.23 3 point IDFT from 4 point DFT

```
//A 3-point IDFT from 4-point DFT
XDFT=[4;-2*%i;0;2*%i];
w=exp(-%i*%pi/2);
for i=1:4
    for j=1:3
        WN(i,j)=w^((i-1)*(j-1));
    end
end
WI=WN';
xn=1/4*(WI*XDFT)
```

Scilab code Exa 8.24 The importance of Periodic Extension

```
//The importance of Periodic extension
//(a)For M=3
x=[1 2 1];
XDFT=dft(x,-1)
w=exp(-%i*2*%pi/3);
for i=1:3
    for j=1:3
        WN(i,j)=w^((i-1)*(j-1));
    end
end
WI=WN';
xn=1/3*WI*XDFT
//The result is periodic with M=3 & 1 period equals
    x[n]
//(b)For M=4
y=[1 2 1 0];
```

```
YDFT=dft(y,-1)
w=exp(-%i*%pi/2);
for i=1:4
    for j=1:4
        WN(i,j)=w^((i-1)*(j-1));
    end
end
WI=WN';
yn=1/4*WI*YDFT
```

Chapter 9

Design of IIR Filters

Scilab code Exa 9.1 Response Invariant Mappings

```
//Response in variant mappings
s=%s;z=%z;
HS=1/(s+1);
f=0:0.05:0.5;
HS1=horner(HS,(%i*%pi*2*f'));
ts=1;
HZ=z/(z-0.3679);
HZ1=horner(HZ,exp(%i*%pi*2*f'));
a=gca();
a.x_location="origin";
subplot(211)
plot2d(f,HS1);
plot2d(f,HZ1);
xlabel('Analog frequency f(Hz)');
ylabel('Magnitude');
xtitle('magnitude of H(s) and H(z)');
HZ1=HZ1-0.582;//magnitude after gain matching at dc
b=gca();
b.x_location="origin";
subplot(212);
plot2d(f,HS1);
```

```
plot2d(f,HZ1);
xlabel('Analog frequency f(Hz)');
ylabel('Magnitude');
xtitle('magnitude after gain matching at DC');
//Impulse response of analog and digital filter
t=0:0.01:6;
ht=exp(-t');
n=0:6;
hn=exp(-n');
xset('window',1)
c=gca();
subplot(211);
plot2d(t,ht);
plot2d3('gnn',n,hn);
xlabel('DT index n and time t=nts');
ylabel('Amplitude');
xtitle('Impulse response of analog and digital
    filter');
//Step response of analog and digital filter
t=0:0.01:6;
st=1-exp(-t');
n=0:6;
sn=(%e-%e^(-n'))/(%e-1);
c=gca();
subplot(212);
plot2d(t,st);
plot2d3('gnn',n,sn);
xlabel('DT index n and time t=nts');
ylabel('Amplitude');
xtitle('Step response of analog and digital filter')
    ;
```

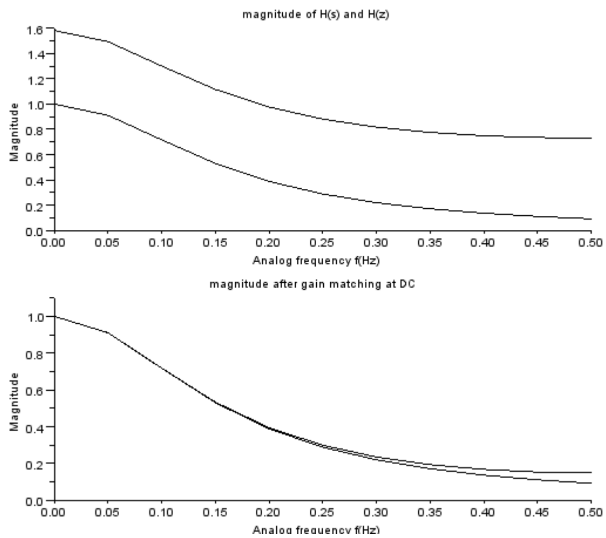

Figure 9.1: Response Invariant Mappings

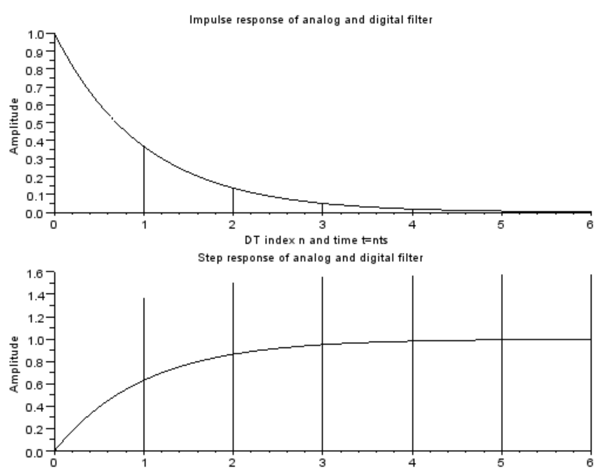

Figure 9.2: Response Invariant Mappings

Scilab code Exa 9.2 Impulse Invariant Mappings

```
//Impulse invariant mappings
//(a)converting H(s)=4s+7/s^2+5s+4 to H(z) using impulse invariance
s=%s;
z=%z;
HS=(4*s+7)/(s^2+5*s+4);
pfss(HS)
ts=0.5;
HZ=3*z/(z-%e^(-4*ts))+z/(z-%e^(-ts))
//(b)converting H(s)=4s+7/s^2+5s+4 to H(z) using impulse invariance
HS1=4/((s+1)*(s^2+4*s+5))
pfss(HS1);
HZ1=2*z/(z-%e^-ts)+(2*1.414*z^2*cos(-0.75*%pi)-2*1.414*(z/%e)*cos(0.5-0.75*%pi))/(z^2-2*(z/%e)*cos(0.5)+%e^-2)
```

Scilab code Exa 9.3ab Modified Impulse Invariant Design

```
//(a)Impulse invariant design
s=%s;z=%z;
HS=1/(s+1);
H1s=horner(HS,3*s/%pi)
H1z=%pi/3*z/(z-%e^(-%pi/3))
//Modified inmpulse invariant design
HZ=z/(z-1/%e);
HMZ=0.5*(z+1/%e)/(z-1/%e);//modified transfer function
H1Z=HZ/horner(HZ,1)
HM1Z=HMZ/horner(HMZ,1)
f=0:0.05:0.5;
```

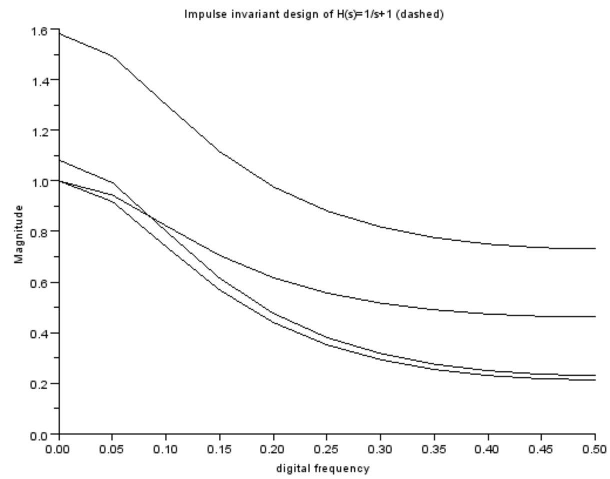

Figure 9.3: Modified Impulse Invariant Design

```
HZ1=horner(HZ,exp(%i*2*%pi*f'));
HMZ1=horner(HMZ,exp(%i*2*%pi*f'));
H1Z1=horner(H1Z,exp(%i*2*%pi*f'));
HM1Z1=horner(HM1Z,exp(%i*2*%pi*f'));
a=gca();
a.x_location="origin";
plot2d(f,HZ1);
plot2d(f,HMZ1);
plot2d(f,H1Z1);
plot2d(f,HM1Z1);
xlabel('digital frequency');
ylabel('Magnitude');
xtitle('Impulse invariant design of H(s)=1/s+1 (
    dashed)');
```

Scilab code Exa 9.3cd Modified Impulse Invariant Design

```
//modified Impulse invariant Design
```

```
//(c)H(s)=4s+7/s^2+5s+4
s=%s;z=%z;
HS=(4*s+7)/(s^2+5*s+4);
[d1]=degree(numer(HS));
[d2]=degree(denom(HS));
HZ=((3*z)/(z-%e^-2))+(z/(z-%e^-0.5))
if (d2-d1==1) then
    h=(4+7/%inf)/(1+5/%inf+4/%inf)
    HMZ=HZ-0.5*h
else
    HMZ=HZ
end
HS1=4/((s+1)*(s^2+4*s+5))
HZ1=(0.2146*z^2+0.093*z)/(z^3-1.2522*z^2+0.527*z
   -0.0821);
[d1]=degree(numer(HS1));
[d2]=degree(denom(HS1));
if (d2-d1==1) then
    HMZ1=HZ1-0.5*h
else
    HMZ1=HZ1
end
```

Scilab code Exa 9.5 Mappings from Difference Algorithms

```
//Mappings from difference algorithms
//Backward difference mappings
s=%s;z=%z;
ts=1;a=1;
HS=1/(s+a);
HZa=horner(HS,(z-1)/(z*ts))
z1=roots(denom(HZa))//for ts >0 HZa always stable
HZb=horner(HS,(z-1)/ts)
z2=roots(denom(HZb))//stable only for 0<ats<2
HZc=horner(HS,(z^2-1)/(2*z*ts))
```

```
z3=roots(denom(HZc))//magnitude of 1 pole is always
    >1 hence unstable
```

Scilab code Exa 9.6 Mappings From Integration Algorithms

```
//Mappings from integration algorithm
//(a)Trapezoidal integration algorithm
s=%s;z=%z;
a=input('enter the value of a')
ts=input('enter the value of sampling time')
HS=1/(s+1);
HZa=horner(HS,2*(z-1)/(ts*(z+1)))
za=roots(denom(HZa))//pole always less than 1 ,hence
    HZ is stable
//(b)simphson's algorithm
HZb=horner(HS,3*(z^2-1)/(ts*(z^2+4*z+1)))
zb=roots(denom(HZb))
//magnitude of 1 pole always greater than 1 ,hence
    HZ is unstable
```

Scilab code Exa 9.7 DTFT of Numerical Algorithms

```
//DTFT of numerical algorithms
//(a)For trapezoidal numerical integrator
ieee(2)
F=0:0.01:0.2;
HTF=1/(%i*2*tan(%pi*F'));
HIF=1/(%i*2*%pi*F');
Ha=1-((%pi*F')^2)/3-((%pi*F')^4/45);
//(b)For simphson's numerical integrator
Hb=((2*%pi*F')).*((2+cos(2*%pi*F'))./(3*sin(2*%pi*F
    ')));
//For forward difference operator
```

```
HFF=(%e^(%i*2*%pi*F'))-1;
HDF=1/(%i*2*%pi*F');
Hc=1+(%i*2*%pi*F')/2-(2*%pi*F')^2/6;
Hc=abs(Hc);
//for central difference operator
HCF=sin(2*%pi*F')./(%i*4*%pi*%pi*F'^2);
Hd=abs(sin(2*%pi*F')./(2*%pi*F'));
length(F),length(Ha)
a=gca();
a.x_location="origin";
plot2d(F,Ha,rect=[0,0.8,0.2,1.1]);
plot2d(F,Hb);
xtitle("Magnitude spectrum of Integration algorithms
    ","Digital Frequency F","Magnitude");
xset('window',1);
plot2d(F,Hc,rect=[0,0.8,0.2,1.1]);
plot2d(F,Hd);
xtitle("Magnitude spectrum of difference algorithms"
    ,"Digital Frequency F","Magnitude");
```

Scilab code Exa 9.8a Bilinear Transformation

```
//Bilinear transformation
//To convert bessel analog filter to digital filter
s=%s;
z=%z;
HS=3/(s^2+3*s+3);
Wa=4;//analog omega
Wd=%pi/2;//digital omega
```

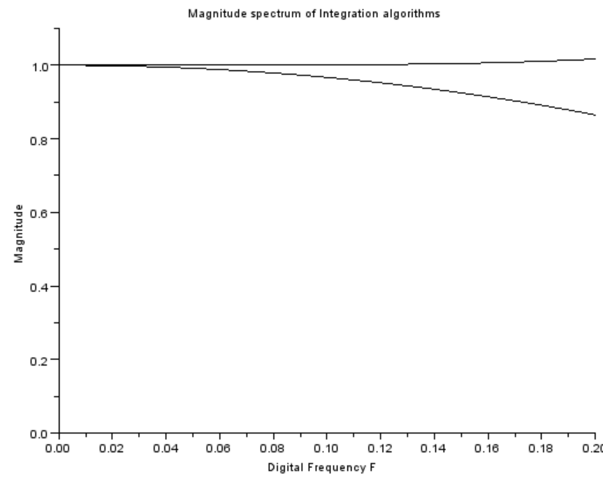

Figure 9.4: DTFT of Numerical Algorithms

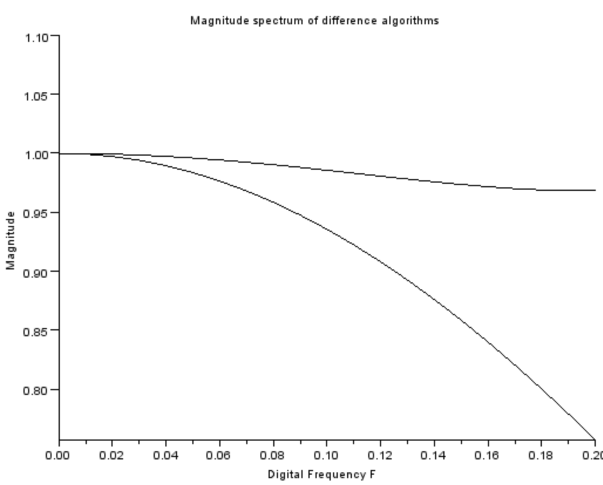

Figure 9.5: DTFT of Numerical Algorithms

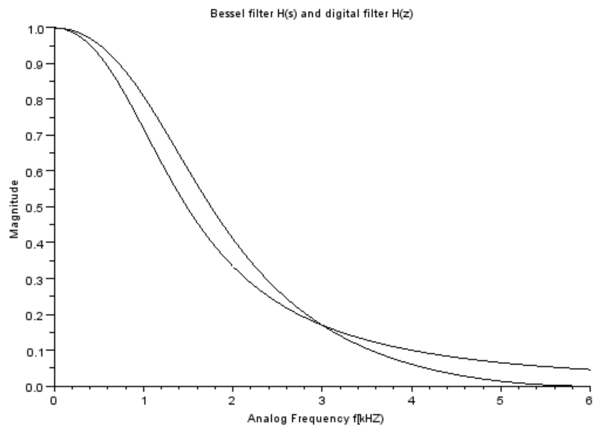

Figure 9.6: Bilinear Transformation

```
T=(2/Wa)*(tan(Wd/2));
HZ=horner(HS,(2/T)*(z-1)/(z+1))
f=0:0.1:6;
HS1=horner(HS,(%i*4*f'/3));
HS1=abs(HS1);
HZ1=horner(HZ,exp(-%i*%pi*f'/6));
HZ1=abs(HZ1);
a=gca();
a.x_location="origin";
plot2d(f,HS1);
plot2d(f,HZ1);
xlabel('Analog Frequency f[kHZ)');
ylabel('Magnitude');
xtitle('Bessel filter H(s) and digital filter H(z)')
   ;
```

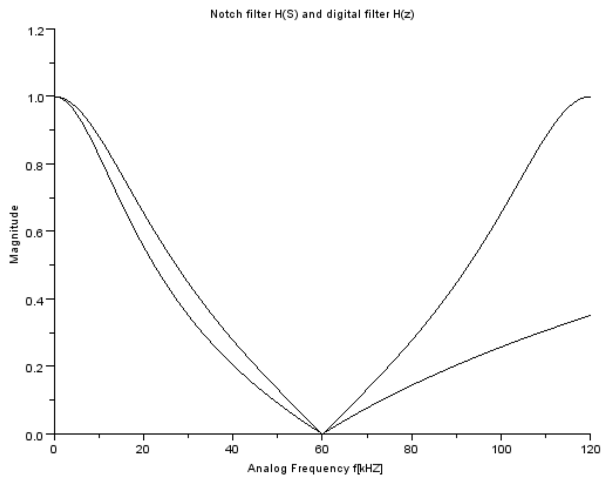

Figure 9.7: Bilinear Transformation

Scilab code Exa 9.8b Bilinear Transformation

```
//Bilinear transformation
//To convert twin−T notch analog filter to digital
   filter
s=%s;
z=%z;
HS=(s^2+1)/(s^2+4*s+1);
Wo=1;
S=240;f=60;//sampling and analog frequencies
W=0.5*%pi;//digital frequency
C=Wo/tan(0.5*W)
HZ=horner(HS,C*(z-1)/(z+1))
f=0:120;
HZ1=abs(horner(HZ,exp(-%i*%pi*f'/120)));
HS1=abs(horner(HS,(%i*f'/60)));
a=gca();
a.x_location="origin";
plot2d(f,HZ1);
plot2d(f,HS1);
```

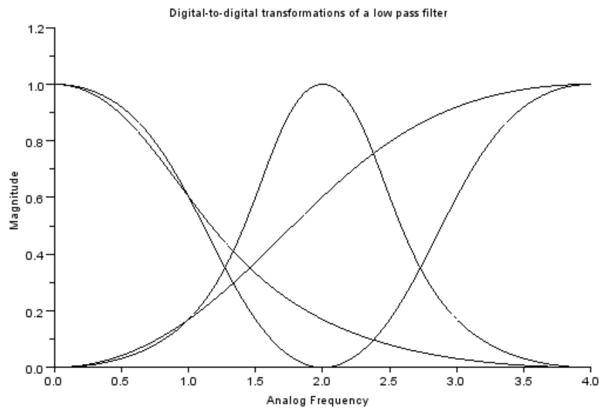

Figure 9.8: D2D transformations

```
xlabel('Analog Frequency f[kHZ]');
ylabel('Magnitude');
xtitle('Notch filter H(S) and digital filter H(z)');
```

Scilab code Exa 9.9 D2D transformations

```
//D2D transformations
//(a)LP 2 HP transformation
z=%z;
HZ=(3*(z+1)^2)/(31*z^2-26*z+7);
WD=0.5*%pi;WC=0.25*%pi;//Band edges
a=-cos(0.5*(WD+WC))/cos(0.5*(WD-WC))//Mapping parameter
HHPZ=horner(HZ,-(z+a)/(1+a*z))
//(b)LP 2 BP transformation
W1=0.25*%pi;
```

```
W2=0.75*%pi;
K=tan(0.5*WD)/tan(0.5*(W2-W1))
a=-cos(0.5*(W1+W2))/cos(0.5*(W1-W2))//Mapping
    parameters k,a,A1,A2
A1=2*K*a/(K+1),A2=(K-1)/(K+1)
HBPZ=horner(HZ,-(z^2+A1*z+A2)/(A2*z^2+A1*z+1))
//(c)Lp 2 BS transformation
w1=3/8*%pi;w2=5/8*%pi;//band edges
K1=tan(0.5*WD)*tan(0.5*(w2-w1))
a1=-cos(0.5*(w1+w2))/cos(0.5*(w2-w1))//Mapping
    parameters k1,a1,A1c,A2c
A1c=2*a1/(K1+1),A2c=-(K1-1)/(K1+1)
HBSZ=horner(HZ,(z^2+A1c*z+A2c)/(A2c*z^2+A1c*z+1))
f=0:1/256:4;
for i=1:length(f)
HZ1(i)=horner(HZ,exp(-%i*%pi*f(i)/4));
end
HZ1=abs(HZ1);
HHPZ1=abs(horner(HHPZ,exp(-%i*%pi*f'/4)));
HBPZ1=abs(horner(HBPZ,exp(-%i*%pi*f'/4)));
HBSZ1=abs(horner(HBSZ,exp(-%i*%pi*f'/4)));
a=gca();
a.x_location="origin";
plot2d(f,HZ1);
plot2d(f,HHPZ1);
plot2d(f,HBPZ1);
plot2d(f,HBSZ1);
xlabel('Analog Frequency');
ylabel('Magnitude');
xtitle('Digital-to-digital transformations of a low
    pass filter');
```

Scilab code Exa 9.10a Bilinear Design of Second Order Filters

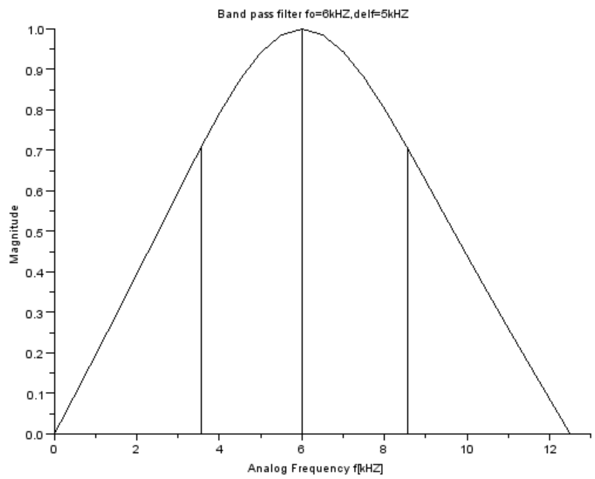

Figure 9.9: Bilinear Design of Second Order Filters

```
//Bilinear design of second order filters
s=%s;z=%z;
fo=6;Wo=2*%pi*fo/25;
delf=5;S=25;
B=cos(2*%pi*fo/25)
C=tan(%pi*delf/25)
HS=1/(s+1);
HZ=horner(HS,(z^2-(2*B*z)+1)/(C*(z^2)-C))
f=0:0.5:12.5;
HZ1=horner(HZ,exp(%i*2*%pi*f'/25));
HZ1=abs(HZ1);
W2=(%pi*delf/25)+acos(cos(Wo)*cos(%pi*delf/25))
W1=W2-(2*%pi*delf/25)
f1=S*W1/(2*%pi),f2=S*W2/(2*%pi)
f3=[f1;fo;f2];
HZf=abs(horner(HZ,exp(%i*2*%pi*f3'/25)));
a=gca();
a.x_location="origin";
plot2d(f,HZ1,rect=[0 0 13 1]);
plot2d3('gnn',f3,HZf);
```

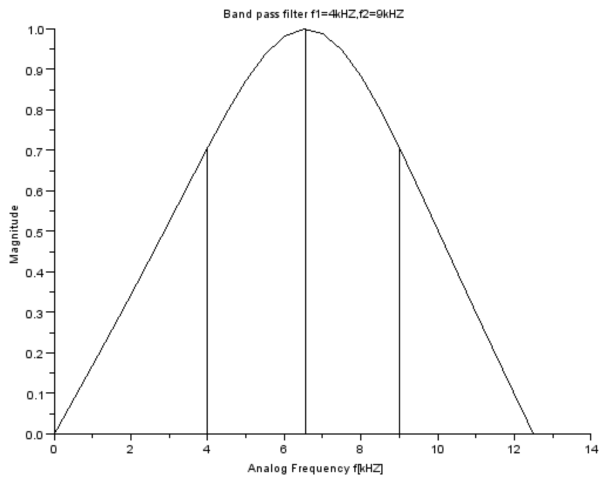

Figure 9.10: Bilinear Design of Second Order Filters

```
xlabel('Analog Frequency f[kHZ]');
ylabel('Magnitude');
xtitle('Band pass filter fo=6kHZ,delf=5kHZ');
```

Scilab code Exa 9.10b Bilinear Design of Second Order Filters

```
//Bilinear design of second order filters
s=%s;z=%z;
f1=4;f2=9;
delf=f2-f1;S=25;
B=cos(%pi*(f1+f2)/25)/cos(%pi*(f2-f1)/25)
C=tan(%pi*delf/25)
HS=1/(s+1);
HZ=horner(HS,(z^2-(2*B*z)+1)/(C*(z^2)-C))
f=0:0.5:12.5;
HZ1=horner(HZ,exp(%i*2*%pi*f'/25));
```

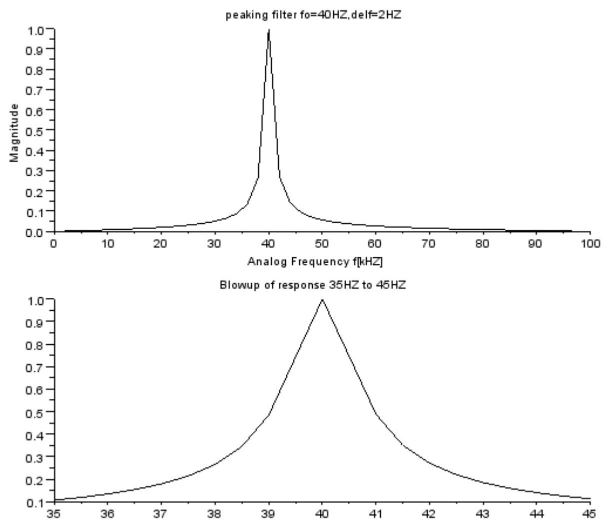

Figure 9.11: Bilinear Design of Second Order Filters

```
HZ1=abs(HZ1);
fo=S*acos(B)/(2*%pi)
f3=[f1 fo f2];
HZf=abs(horner(HZ,exp(-%i*2*%pi*f3'/25)));
a=gca();
a.x_location="origin";
plot2d(f,HZ1);
plot2d3('gnn',f3,HZf);
xlabel('Analog Frequency f[kHZ]');
ylabel('Magnitude');
xtitle('Band pass filter f1=4kHZ,f2=9kHZ');
```

Scilab code Exa 9.10c Bilinear Design of Second Order Filters

```
//Bilinear design of second order filters
s=%s;z=%z;
```

```
fo=40;Wo=2*%pi*fo/200;
delf=2;S=25;
delW=2*%pi*delf/200;
B=cos(2*%pi*fo/200)
K=0.557;
C=K*tan(0.5*delW)
HS=1/(s+1);
HZ=horner(HS,(z^2-(2*B*z)+1)/(C*(z^2)-C))
f=0:2:100;
f1=35:0.5:45;
HZ1=horner(HZ,exp(%i*2*%pi*f'/200));
HZ2=horner(HZ,exp(%i*2*%pi*f1'/200));
HZ1=abs(HZ1);
HZ2=abs(HZ2);
a=gca();
a.x_location="origin";
subplot(211);
plot2d(f,HZ1);
xlabel('Analog Frequency f[kHZ]');
ylabel('Magnitude');
xtitle('peaking filter fo=40HZ,delf=2HZ');
subplot(212);
plot2d(f1,HZ2);
xtitle('Blowup of response 35HZ to 45HZ');
```

Scilab code Exa 9.11 Interference Rejection

```
//interference Rejection
//design oh high-Q and low-Q notch filters
s=%s;z=%z;
Q=50;
fo=60;S=300;
delf=fo/Q;
```

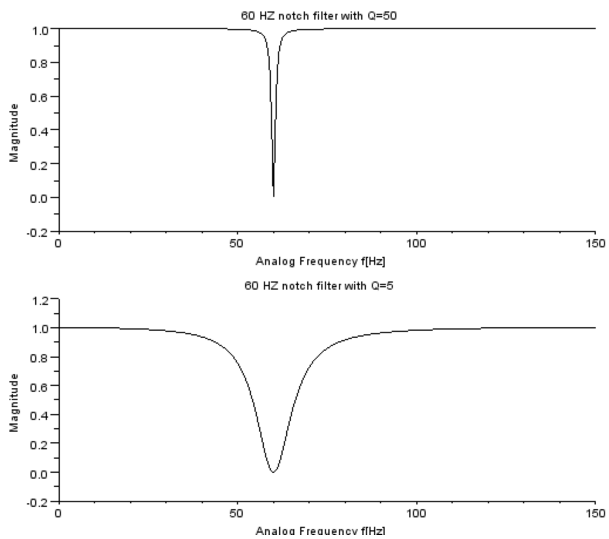

Figure 9.12: Interference Rejection

```
Wo=2*%pi*fo/S;
delW=2*%pi*delf/S;
C=tan(0.5*delW),B=cos(Wo)
HS=(s)/(s+1);
H1Z=horner(HS,(z^2-(2*B*z)+1)/(C*(z^2)-C))
Q1=5;delf1=fo/Q1;
delW1=2*%pi*delf1/S;
C1=tan(0.5*delW1),B1=cos(Wo)
H2Z=horner(HS,(z^2-(2*B1*z)+1)/(C1*(z^2)-C1))
f=0:0.5:150;
H1Z1=horner(H1Z,exp(%i*2*%pi*f'/S));
H2Z1=horner(H2Z,exp(%i*2*%pi*f'/S));
a=gca();
subplot(211);
plot2d(f,H1Z1);
xlabel('Analog Frequency f[Hz]');
ylabel('Magnitude');
xtitle('60 HZ notch filter with Q=50');
subplot(212);
plot2d(f,H2Z1);
```

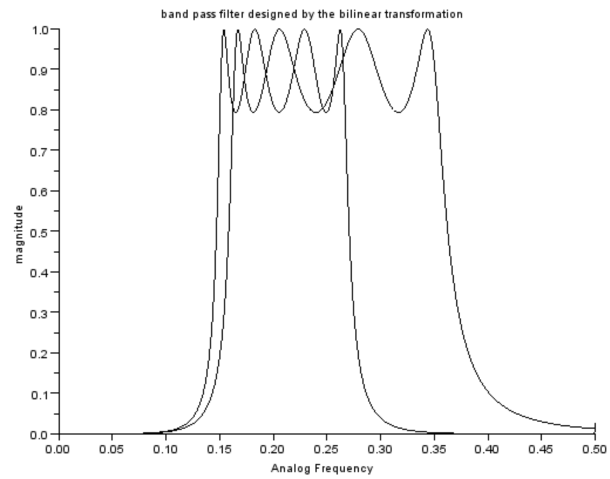

Figure 9.13: IIR Filter Design

```
xlabel('Analog Frequency f[Hz]');
ylabel('Magnitude');
xtitle('60 HZ notch filter with Q=5');
```

Scilab code Exa 9.12 IIR Filter Design

```
//IIR filter design
//Design of chebyshev IIR filter with following
    specifications
fp1=1.6;fp2=1.8;fs1=3.2;fs2=4.8;//pass band edges
Ap=2;As=20;S=12;
s=%s;z=%z;
//(a)Indirect Bilinear design
W=2*%pi*[fp1 fp2 fs1 fs2]/S
C=2;
```

```
omega=2*tan(0.5*W');//prewarping each band edge
   frequency
epsilon=sqrt(10^(0.1*Ap)-1);
n=acosh(((10^(0.1*As)-1)/epsilon^2)^1/2)/(acosh(fs1/
   fp1));
n=ceil(n)
alpha=(1/n)*asinh(1/epsilon);
for i=1:n
    B(i)=(2*i-1)*%pi/(2*n);
end
for i=1:n
    p(i)=-sinh(alpha)*sin(B(i))+%i*cosh(alpha)*cos(B
       (i));
end
Qs=1;
for i=1:n
    Qs=Qs*(s-p(i))
end
Qo=0.1634;
HPS=Qo/Qs
HBPS=horner(HPS,(s^2+1.5045^2)/(s*1.202))
HZ=horner(HBPS,2*(z-1)/(z+1))
f=0:0.001:0.5;
HZF=abs(horner(HZ,exp(%i*2*%pi*f')));
HBPF=abs(horner(HBPS,%i*2*%pi*f'));
a=gca();
plot2d(f,HZF);
plot2d(f,HBPF);
xlabel('Analog Frequency');
ylabel('magnitude');
xtitle('band pass filter designed by the bilinear
   transformation');
```

Chapter 10

Design of FIR filters

Scilab code Exa 10.2 Truncation and Windowing

```
//Truncation and Windowing
//(a)N=9,Barlett Window.
z=%z;
Fc=0.25;
n=-4:4;
hn=2*Fc*(sinc(0.5*n'*%pi))
Wn=1-(2*abs(n'))/8//Barlett window
hwn=hn.*Wn
Hcz=0;
for i=1:length(hwn)
    Hcz=Hcz+hwn(i)*(z^((2-i)));
end
Hcz//indicates delay of 0.15ms
//(b)N=6,vonhann Window
n1=-2.5:2.5;
hn1=2*Fc*(sinc(0.5*n1'*%pi))
Wn1=0.5+0.5*(cos(0.4*%pi*n1'))//Vonhann window
hwn1=hn1.*Wn1
Hcz1=0;
for i=1:length(hwn1)
    Hcz1=Hcz1+hwn1(i)*(z^((2-i)));
```

```
end
Hcz1//1st sample of hwn is 0 hence delay is 1.5ms
```

Scilab code Exa 10.3ab FIR lowpass Filter design

```
//FIR filter design using windows
//(a)Design of FIR filter to meet following specifications
fp=2;fs=4;Ap=2;As=40;S=20;
Fp=fp/S;Fs=fs/S;
Fc=0.15;
z=%z;
N1=3.21/(Fs-Fp);
N1=ceil(N1)
N2=5.71/(Fs-Fp);
N2=ceil(N2)
n1=-16:16;
n2=-28.5:1:28.5;
hn1=2*Fc*(sinc(2*Fc*n1'));
hn2=2*Fc*(sinc(2*Fc*n2'));
Wn1=0.5+0.5*(cos(2*%pi*n1'/(N1-1)));//Vonhann window
Wn2=0.42+0.5*(cos(2*%pi*n2'/(N2-1)))+0.08*(cos(4*%pi*n2'/(N2-1)));//Blackman window
hwn1=abs(hn1.*Wn1);
hwn2=abs(hn2.*Wn2);
[hwn1F,fr1]=frmag(hwn1,256);
[hwn2F,fr2]=frmag(hwn2,256);
hwn1F1=20*log10(hwn1F);
hwn2F1=20*log10(hwn2F);
plot2d(fr1,hwn1F1);
plot2d(fr2(1:length(fr2)-2),hwn2F1(1:length(fr2)-2));
xlabel('Digital frequency');
ylabel('Magnitude [dB]');
title('Low pass filter using vonhann and Blackmann
```

```
windows Fc=0.15,vonhann N=33,Blackman N=58');
//(b)Minimum length design
Fcv=0.1313;
Fcb=0.1278;
Nv=23;Nb=29;
nv=-11:11;
nb=-14:14;
hnv=2*Fcv*(sinc(2*Fcv*nv'));
hnb=2*Fcb*(sinc(2*Fcb*nb'));
Wnv=0.5+0.5*(cos(2*%pi*nv'/(Nv-1)));//Vonhann window
Wnb=0.42+0.5*(cos(2*%pi*nb'/(Nb-1)))+0.08*(cos(4*%pi*nb'/(Nb-1)));//Blackman window
hwnv=abs(hnv.*Wnv);
hwnb=abs(hnb.*Wnb);
[hwnvF,frv]=frmag(hwnv,256);
[hwnbF,frb]=frmag(hwnb,256);
hwnvF=20*log10(hwnvF);
hwnbF=20*log10(hwnbF);
b=gca();
xset('window',2);
plot(frv,hwnvF);
plot(frb,hwnbF);
xlabel('Digital frequency');
ylabel('Magnitude [dB]');
title('Vonhann Fc=0.1313,Minimum N=23,Blackmann Fc=0.1278,Minimum N=29');
```

Scilab code Exa 10.3cd FIR filter Design

```
//Design of high pass FIR filter with specifications
//fp=4kHZ;fs=2kHZ;Ap=2dB;As=40dB
```

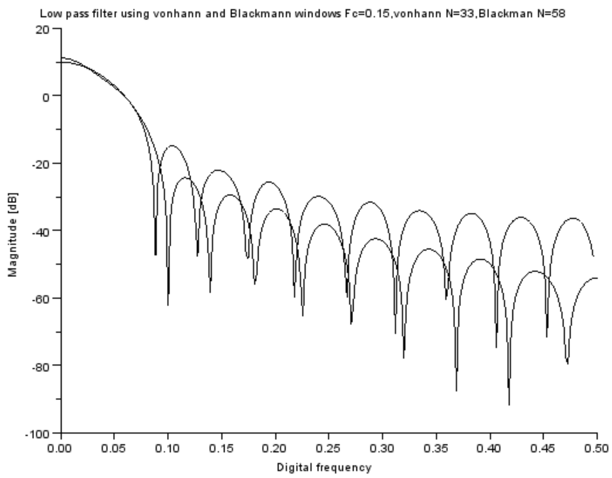

Figure 10.1: FIR lowpass Filter design

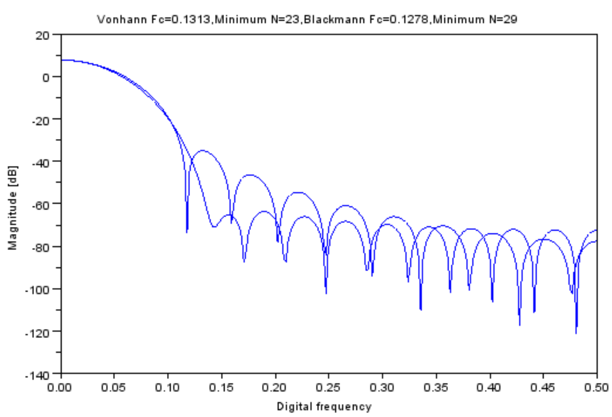

Figure 10.2: FIR lowpass Filter design

```
fp=2;fs=4;Ap=2;As=40;S=20;
Fp=fp/S;Fs=fs/S;
Ft=0.1;
Fc=0.15
N1=3.47/(Fs-Fp);//hamming
N1=int(N1)+1
N2=5.71/(Fs-Fp);//blackman
N2=int(N2)+1
[hn1]=eqfir(N1,[0 0.1;0.2 0.5],[0 1],[1 1]);
[HF1,fr1]=frmag(hn1,512);
Hf1=20*log10(HF1);
[hn2]=eqfir(58,[0 0.1;0.2 0.43],[0 1],[1 1]);
[HF2,fr2]=frmag(hn2,512);
Hf2=20*log10(HF2);
a=gca();
plot2d(fr1,Hf1,rect=[0 -120 0.5 4]);
plot2d(fr2(1:length(fr2)-5),Hf2(1:length(fr2)-5),
   rect=[0 -120 0.5 4]);
xlabel('Digital Frequency F');
ylabel('Magnitude [dB]');
xtitle('High pass filter using Hamming and Blackmann
   windows LPP Fc=0.35');
//Minimum Length Design
[hn3]=eqfir(22,[0 0.1;0.2 0.43],[0 1],[1 1]);
[HF3,fr3]=frmag(hn3,512);
Hf3=20*log10(HF3);
[hn4]=eqfir(29,[0 0.1;0.2 0.5],[0 1],[1 1]);
[HF4,fr4]=frmag(hn4,512);
Hf4=20*log10(HF4);
xset('window',1);
a=gca();
plot2d(fr3(1:length(fr3)-5),Hf3(1:length(fr3)-5),
   rect=[0 -120 0.5 4]);
plot2d(fr4,Hf4,rect=[0 -120 0.5 4]);
xlabel('Digital Frequency F');
ylabel('Magnitude [dB]');
xtitle('Hamming LPP Fc=0.3293 N=22;Blackmann LPP Fc
   =0.3277 N=29');
```

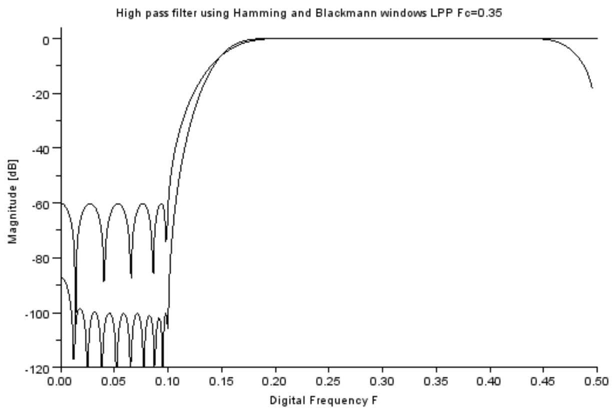

Figure 10.3: FIR filter Design

Scilab code Exa 10.4a Half Band lowpass FIR filter Design

```
//Half band FIR Filter Design
//(a)lowpass Half band Filter
s=%s;z=%z;
fp=8;fs=16;Ap=1;As=50;
S=2*(fs+fp);
Fp=fp/S;Fs=fs/S;Fc=0.25;
delp=(10^(Ap/20)-1)/(10^(Ap/20)+1);
dels=10^(-As/20);
del=min(delp,dels);
As0=-20*log10(del)
N1=(As0-7.95)/(14.36*(Fs-Fp))+1;
```

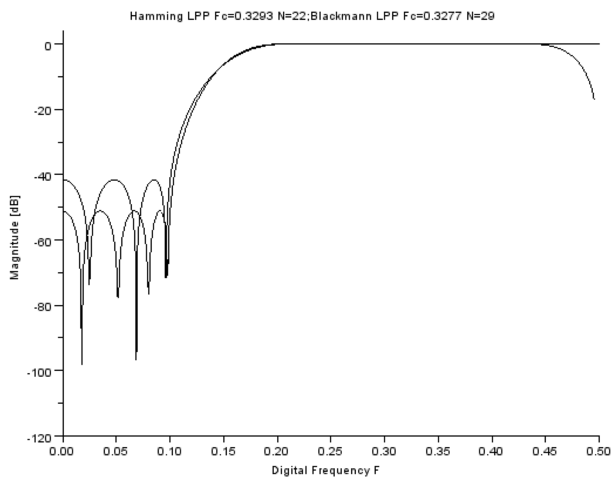

Figure 10.4: FIR filter Design

```
N1=int(N1)+1;
B=0.0351*(As0-8.7)
[hn1]=eqfir(19,[0 1/6;1/3 0.5],[1 0],[1 1]);
[HLPF1,fr1]=frmag(hn1,512);
HLPf1=20*log10(HLPF1);
a=gca();
plot2d(fr1,HLPf1);
xlabel('Digital Frequency');
ylabel('Magnitude in dB');
xtitle('Kaiser half band LPF:B=1.44;Fc=0.25');
[hn2]=eqfir(21,[0 1/6;1/3 0.5],[1 0],[1 1]);
[HLPF2,fr2]=frmag(hn2,512);
HLPf2=20*log10(HLPF2);
xset('window',1);
plot2d(fr2,HLPf2);
xlabel('Digital Frequency');
ylabel('Magnitude in dB');
xtitle('Hamming half-band LPF:N=21;Fc=0.25');
```

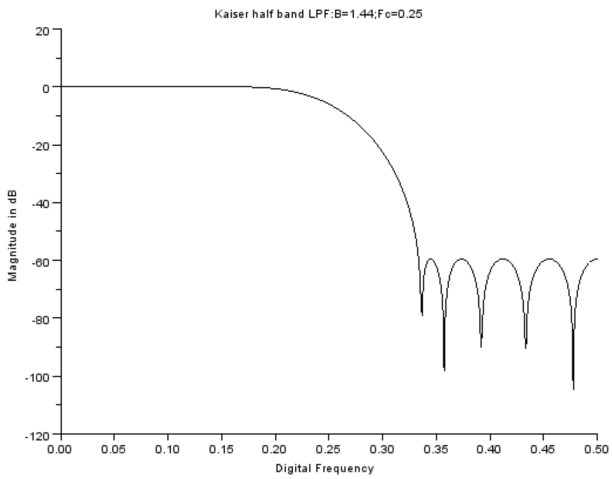

Figure 10.5: Half Band lowpass FIR filter Design

Scilab code Exa 10.4b Half Band bandstop FIR filter Design

```
//Half band FIR Filter Design
//(a)band-stop Half band Filter
s=%s;z=%z;
fp1=1;fs1=2;fp2=4;fs2=3;Ap=1;As=50;
S=2*(fs1+fs2);
Fp=0.5*(fs2/S-fs1/S);Fs=0.5*(fp2/S-fp1/S);
Fc=0.5*(Fp+Fs);Fo=0.25;
delp=(10^(Ap/20)-1)/(10^(Ap/20)+1);
dels=10^(-As/20);
del=min(delp,dels);
As0=-20*log10(del)
```

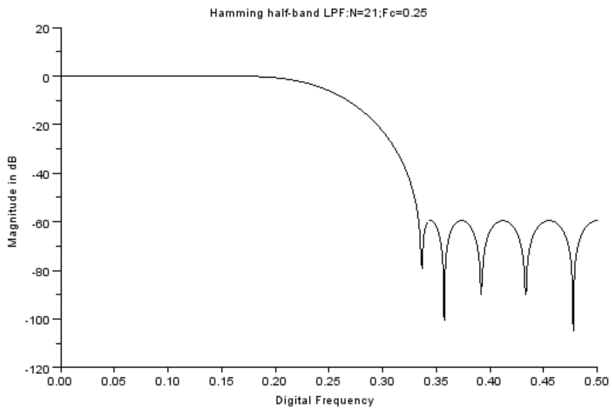

Figure 10.6: Half Band lowpass FIR filter Design

```
N1=(As0-7.95)/(14.36*(Fs-Fp))+1;
N1=ceil(N1);
B=0.0351*(As0-8.7)
[hn1]=eqfir(31,[0 0.1;0.2 0.3;0.4 0.5],[1 0 1],[1 1
   1]);
[HBSF1,fr1]=frmag(hn1,400);
HBSf1=20*log10(HBSF1);
a=gca();
plot2d(fr1,HBSf1);
xlabel('Digital Frequency');
ylabel('Magnitude in dB');
xtitle('Kaiser half band LPF:B=1.44;Fc=0.25');
[hn2]=eqfir(35,[0 0.1;0.2 0.3;0.4 0.5],[1 0 1],[1 1
   1]);
[HF2,fr2]=frmag(hn2,200);
HBSf2=20*log10(HF2);
xset('window',1);
plot2d(fr2,HBSf2);
xlabel('Digital Frequency');
ylabel('Magnitude in dB');
```

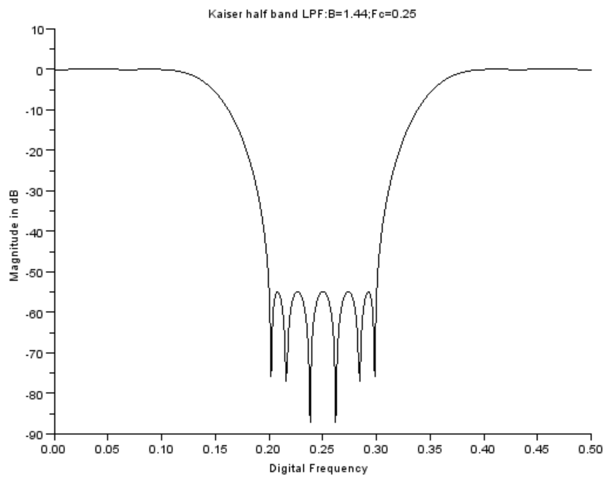

Figure 10.7: Half Band bandstop FIR filter Design

```
xtitle('Hamming half-band LPF:N=21;Fc=0.25');
```

Scilab code Exa 10.5a Design by Frequency Sampling

```
//Fir low pass filter design by frequency sampling
z=%z;
N=10;
magHk=[1 1 1 0 0 0 0 0 1 1];
k=[0:7 -1 -2];
fik=-%pi*k'*(N-1)/N;
for i=1:length(fik)
    H1k(i)=magHk(i)*exp(%i*fik(i));
```

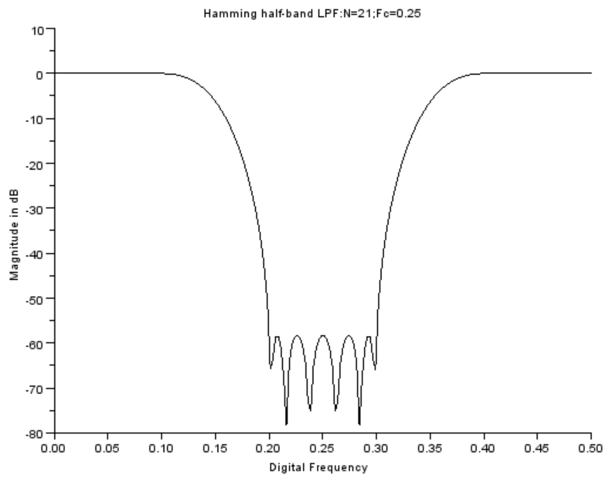

Figure 10.8: Half Band bandstop FIR filter Design

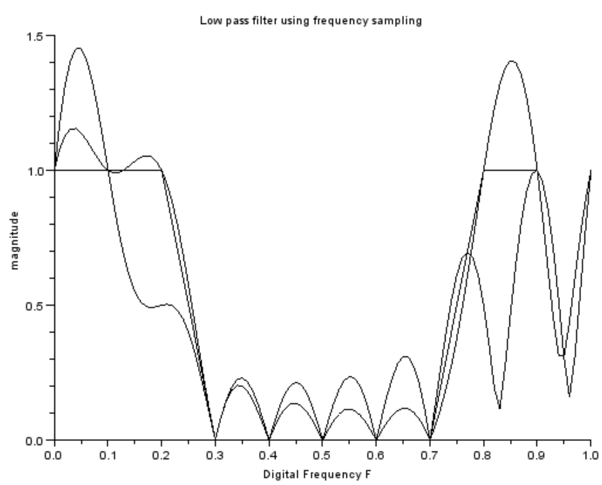

Figure 10.9: Design by Frequency Sampling

```
end
H1n=(dft(H1k,1));
H2k=H1k;
H2k(3)=0.5*%e^(-%i*1.8*%pi);
H2k(9)=0.5*%e^(%i*1.8*%pi);
H2n=(dft(H2k,1));
H1Z=0;H2Z=0;
for i=1:length(H1n)
    H1Z=H1Z+H1n(i)*z^(-i);
end
for i=1:length(H2n)
    H2Z=H2Z+H2n(i)*z^(-i);
end
F=0:0.01:1;
F1=0:0.1:0.9;
H1F=abs(horner(H1Z,exp(%i*2*%pi*F')));
H2F=abs(horner(H2Z,exp(%i*2*%pi*F')));
a=gca();
plot2d(F1,magHk);
plot2d(F,H2F);
plot2d(F,H1F);
xlabel('Digital Frequency F');
ylabel('magnitude');
xtitle('Low pass filter using frequency sampling');
```

Scilab code Exa 10.5b Design by Frequency Sampling

```
//Fir high pass filter design by frequency sampling
z=%z;
N=10;
magHk=[0 0 0 1 1 1 1 1 0 0];
k=[0:5 -4:-1:-1];
fik=(-%pi*k'*(N-1)/N)+(0.5*%pi);
```

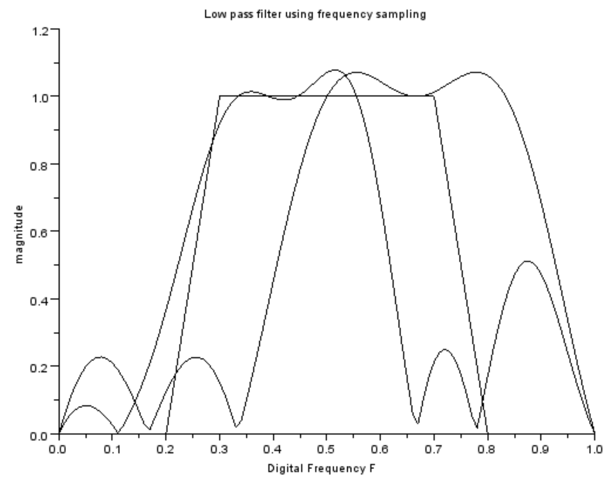

Figure 10.10: Design by Frequency Sampling

```
for i=1:length(fik)
    H1k(i)=magHk(i)*exp(%i*fik(i));
end
H1n=(dft(H1k,1));
H2k=H1k;
H2k(3)=0.5*%e^(-%i*1.3*%pi);
H2k(9)=0.5*%e^(%i*1.3*%pi);
H2n=(dft(H2k,1));
H1Z=0;H2Z=0;
for i=1:length(H1n)
    H1Z=H1Z+H1n(i)*z^(-i);
end
for i=1:length(H2n)
    H2Z=H2Z+H2n(i)*z^(-i);
end
F=0:0.01:1;
F1=0:0.1:0.9;
H1F=abs(horner(H1Z,exp(%i*2*%pi*F')));
H2F=abs(horner(H2Z,exp(%i*2*%pi*F')));
a=gca();
```

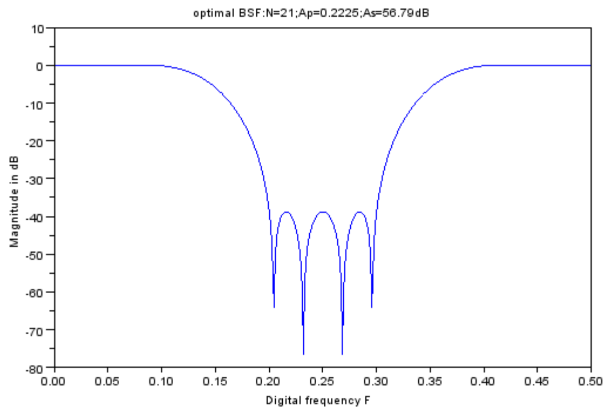

Figure 10.11: Optimal FIR Bandstop Filter Design

```
plot2d(F1,magHk);
plot2d(F,H2F);
plot2d(F,H1F);
xlabel('Digital Frequency F');
ylabel('magnitude');
xtitle('Low pass filter using frequency sampling');
```

Scilab code Exa 10.6a Optimal FIR Bandstop Filter Design

```
//optimal Fir band stop filter design
fp1=1;fp2=4;fs1=2;fs2=3;
Ap=1;As=50;S=10;
Fp1=fp1/S;Fp2=fp2/S;Fs1=fs1/S;Fs2=fs2/S;
FT=0.1;FC=0.25
//calculation of filter length
delp=(10^(Ap/20)-1)/(10^(Ap/20)+1);
```

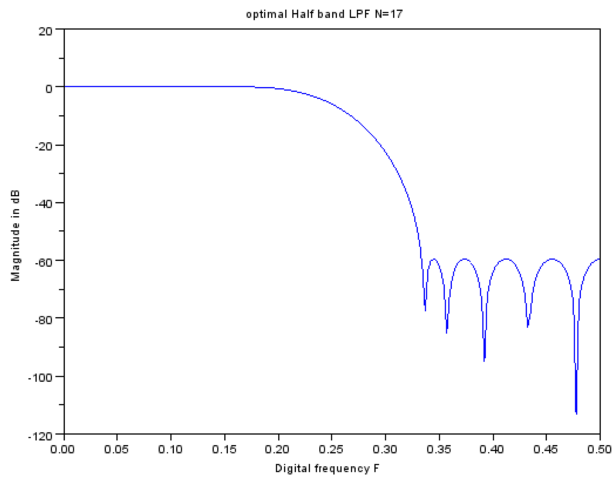

Figure 10.12: Optimal Half Band Filter Design

```
dels=10^(-As/20);
N=1+((-10*log10(delp*dels)-13)/(14.6*FT))
N1=21;
[hn]=eqfir(N1,[0 0.1;0.2 0.3;0.4 0.5],[1 0 1],[1 1 1]);
[HF,fr]=frmag(hn,512);
Hf=20*log10(HF);
a=gca();
plot(fr,Hf);
xlabel('Digital frequency F');
ylabel('Magnitude in dB');
xtitle('optimal BSF:N=21;Ap=0.2225;As=56.79dB');
```

Scilab code Exa 10.6b Optimal Half Band Filter Design

```
//optimal Fir band pass filter design
```

```
fp=8;fs=16;
Ap=1;As=50;S=48;
Fp=fp/S;Fs=fs/S;
FT=0.1;FC=0.25
//calculation of filter length
delp=(10^(Ap/20)-1)/(10^(Ap/20)+1);
dels=10^(-As/20);
del=min(delp,dels);
N=1+((-10*log10(del*del)-13)/(14.6*FT));
N1=19;
[hn]=eqfir(N1,[0 1/6;1/3 0.5],[1 0],[1 1]);
[HF,fr]=frmag(hn,200);
Hf=20*log10(HF);
a=gca();
plot(fr,Hf);
xlabel('Digital frequency F');
ylabel('Magnitude in dB');
xtitle('optimal Half band LPF N=17');
```

Scilab code Exa 10.7 Multistage Interpolation

```
//The concept of multistage Interpolation
//(a)Single stage interpolator
Sin=4;Sout=48;
fp=1.8;
fs=Sin-fp;
FT=(fs-fp)/Sout;
disp('By using single stage the total filter length
    is:')
L=4/FT
//(b)Two-stage interpolator
Sin=[4 12];
I=[3 4];//interpolating factors
Sout=[12 48];
fp=[1.8 1.8];
```

```
fs=Sin-fp;
L1=4*Sout./(fs-fp);
L=0;
for i=1:length(L1)
    L=L+L1(i);
end
disp('By using 2 stage interpolator filter length is
    :")
ceil(L)
//(c)3 stage interpolator with I1=2;I2=3;I3=2
Sin=[4 8 24];
I=[2 3 2];
Sout=[8 24 48];
fp=[1.8 1.8 1.8];
fs=Sin-fp;
L2=4*Sout./(fs-fp);L=0;
for i=1:length(L2)
    L=L+L2(i);
end
disp('By using 3 stage interpolator filter length is
    :")
ceil(L)
//(d)3 stage interpolator with I1=2;I2=3;I3=2
Sin=[4 12 24];
I=[3 2 2];
Sout=[12 24 48];
fp=[1.8 1.8 1.8];
fs=Sin-fp;
L3=4*Sout./(fs-fp);L=0;
for i=1:length(L3)
    L=L+L3(i);
end
disp('By using 2 stage interpolator filter length is
    :")
ceil(L)
```

Scilab code Exa 10.8 Design of Interpolating Filters

```
//Design of interpolating filters
//(a)Design using a single stage interpolator
fp=1.8;Sout=48;Sin=4;
Ap=0.6;As=50;
fs=Sin-fp;
//finding ripple parameters
delp=(10^(Ap/20)-1)/(10^(Ap/20)+1);
dels=10^(-As/20);
N=Sout*(-10*log10(delp*dels)-13)/(14.6*(fs-fp))+1;
disp('By using single stage interpolator the filter design is:');
ceil(N)
//Design using 3-stage interpolator with I1=2;I2=3;I3=2
Ap=0.2;
Sin=[4 8 24];
Sout=[8 24 48];
fp=[1.8 1.8 1.8];
fs=Sin-fp;
delp=(10^(Ap/20)-1)/(10^(Ap/20)+1);
dels=10^(-As/20);
p=14.6*(fs-fp);
N1=((-10*log10(delp*dels)-13)./p);
N1=(Sout.*N1)+1;N=0;
for i=1:length(N1)
    N=N+N1(i);
end
disp('By using single stage interpolator the filter design is:');
ceil(N)
```

Scilab code Exa 10.9 Multistage Decimation

```
//The concept of multistage Decimation
//(a)Single stage decimator
Sin=48;Sout=4;
fp=1.8;
fs=Sout-fp;
FT=(fs-fp)/Sin;
disp('By using single stage the total filter length
   is:')
L=4/FT
//(b)Two-stage decimator
Sin=[48 12];
D=[4 3];//decimating factors
Sout=[12 4];
fp=[1.8 1.8];
fs=Sout-fp;
L1=4*Sin./(fs-fp);
L=0;
for i=1:length(L1)
    L=L+L1(i);
end
disp('By using 2 stage decimator filter length is:")
ceil(L)
//3 stage decimator with D1=2;D2=3;D3=2
Sin=[48 24 8];
D=[2 3 2];
Sout=[24 8 4];
fp=[1.8 1.8 1.8];
fs=Sout-fp;
L2=4*Sin./(fs-fp);L=0;
for i=1:length(L2)
    L=L+L2(i);
end
```

```
disp('By using 3 stage decimator filter length is:")
ceil(L)
//3 stage decimator with I1=2;I2=3;I3=2
Sin=[48 24 12];
D=[2 2 3];
Sout=[24 12 4];
fp=[1.8 1.8 1.8];
fs=Sout-fp;
L3=4*Sin./(fs-fp);L=0;
for i=1:length(L3)
    L=L+L3(i);
end
disp('By using 3 stage decimator filter length is:")
ceil(L)
```

Scilab code Exa 10.10 Maximally Flat FIR filter Design

```
//Maximally flat FIR filter design
Fp=0.2;
Fs=0.4;
Fc=0.3;
Ft=0.2;
N0=1+0.5/Ft^2;
N0=ceil(N0);
alpha=(cos(%pi*Fc))^2;
k=5;Mmin=14;
L=Mmin-k;
N=2*Mmin-1;
disp(N,'Hence with this length we can get maximally
   flat FIR filter with no ripples in passband');
```

Appendix

Scilab code AP 1 Alias Frequency

```
function[F]=aliasfrequency(f,s,s1)
if (s>2*f) then
    disp("alias has not occured")
else
    disp("alias has occured")
end
F=f/s;
for i=1:100
    if (abs(F)>0.5)
        F=F-i;
    end
end
fa=F*s1;
disp(fa,"frequency of reconstructed signal is")
endfunction
```