

Scilab Textbook Companion for
Digital Signal Processing: A Modern
Introduction
by A. Ashok¹

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

July 31, 2019

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

Book Description

Title: Digital Signal Processing: 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	4
2 Discrete Signals	5
3 Response of Digital Filters	16
4 z Transform Analysis	38
5 Frequency Domain Analysis	44
6 Filter Concepts	68
7 Digital Processing of Analog Signals	80
8 The Discrete Fourier Transform and its Applications	92
9 Design of IIR Filters	114
10 Design of FIR filters	134

List of Scilab Codes

Exa 2.1a	Signal energy and power	5
Exa 2.1b	Average power of periodic signals	5
Exa 2.1c	Average power of periodic signals	6
Exa 2.2	Operations on Discrete Signals	6
Exa 2.3a	Even and Odd parts of Discrete signals	8
Exa 2.3b	Even and Odd parts of Discrete signals	9
Exa 2.4a	Decimation and Interpolation of Discrete signals	10
Exa 2.4b	Decimation and Interpolation of Discrete signals	12
Exa 2.4c	Decimation and Interpolation of Discrete signals	12
Exa 2.4d	Decimation and Interpolation of Discrete signals	13
Exa 2.5	Describing Sequences and signals	13
Exa 2.6	Discrete time Harmonics and Periodicity	13
Exa 2.7	Aliasing and its effects	14
Exa 2.8	Signal Reconstruction	15
Exa 3.5	FIR filter response	16
Exa 3.19a	Analytical Evaluation of Discrete Convolution	16
Exa 3.19b	Analytical Evaluation of Discrete Convolution	18
Exa 3.19c	Analytical Evaluation of Discrete Convolution	19
Exa 3.20a	Properties of Convolution	21
Exa 3.20b	Properties of Convolution	22
Exa 3.21a	Convolution of finite length Signals	23
Exa 3.21b	Convolution of finite length Signals	24
Exa 3.21c	Convolution of finite length Signals	25
Exa 3.22	Convolution of finite length Signals	27

Exa 3.23	effect of Zero Insertion,Zero Padding on con- vol.	29
Exa 3.25	Stability and Causality	31
Exa 3.26	Response to Periodic Inputs	32
Exa 3.27	Periodic Extension	33
Exa 3.28	System Response to Periodic Inputs	33
Exa 3.29	Periodic Convolution	34
Exa 3.30	Periodic Convolution by Circulant Matrix	34
Exa 3.32	Deconvolution By polynomial Division	35
Exa 3.33	Autocorrelation and Cross Correlation	35
Exa 3.35	Periodic Autocorrelation and Cross Correla- tion	36
Exa 4.1b	z transform of finite length sequences	38
Exa 4.4a	Pole Zero Plots	38
Exa 4.4b	Pole Zero plots	39
Exa 4.8	Stability of Recursive Filters	39
Exa 4.9	Inverse Systems	40
Exa 4.10	Inverse Transform of sequences	41
Exa 4.11	Inverse Transform by Long Division	41
Exa 4.12	Inverse transform of Right sided sequences	42
Exa 4.20	z Transform of Switched periodic Signals	42
Exa 5.1c	DTFT from Defining Relation	44
Exa 5.3a	Some DTFT pairs using properties	46
Exa 5.3b	Some DTFT pairs using properties	49
Exa 5.3d	Some DTFT pairs using properties	52
Exa 5.3e	Some DTFT pairs using properties	54
Exa 5.4	DTFT of periodic Signals	57
Exa 5.5	The DFT,DFS and DTFT	58
Exa 5.7	Frequency Response of Recursive Filter	59
Exa 5.8a	The DTFT in System Analysis	61
Exa 5.8b	The DTFT in System Analysis	62
Exa 5.9a	DTFT and steady state response	63
Exa 5.9b	DTFT and steady state response	64
Exa 5.10a	System Representation in various forms	65
Exa 5.10b	System Representation in various forms	66
Exa 6.1	The Minimum Phase Concept	68
Exa 6.4	Linear Phase Filters	69
Exa 6.6	Frequency Response and Filter characteristics	70

Exa 6.7a	Filters and Pole Zero Plots	72
Exa 6.7b	Filters and Pole Zero Plots	73
Exa 6.8	Digital resonator Design	74
Exa 6.9	Periodic Notch Filter Design	76
Exa 7.3	Sampling oscilloscope	80
Exa 7.4	Sampling of Band pass signals	81
Exa 7.6	Signal Reconstruction from Samples	82
Exa 7.7	Zero Interpolation and Spectrum Replication	83
Exa 7.8	Up Sampling and Filtering	84
Exa 7.9	Quantisation Effects	88
Exa 7.10	ADC considerations	88
Exa 7.11	Anti Aliasing Filter Considerations	89
Exa 7.12	Anti Imaging Filter Considerations	90
Exa 8.1	DFT from Defining Relation	92
Exa 8.2	The DFT and conjugate Symmetry	92
Exa 8.3	Circular Shift and Flipping	93
Exa 8.4	Properties of DFT	93
Exa 8.5a	Properties of DFT	94
Exa 8.5b	Properties of DFT	95
Exa 8.5c	Properties of DFT	97
Exa 8.6	Signal and Spectrum Replication	97
Exa 8.7	Relating DFT and DTFT	98
Exa 8.8	Relating DFT and DTFT	98
Exa 8.9a	The DFT and DFS of sinusoids	98
Exa 8.9b	The DFT and DFS of sinusoids	100
Exa 8.9c	The DFT and DFS of sinusoids	101
Exa 8.9d	The DFT and DFS of sinusoids	102
Exa 8.10	DFS of sampled Periodic Signals	104
Exa 8.11	The effects of leakage	105
Exa 8.15a	Methods to find convolution	106
Exa 8.15b	Methods to find convolution	108
Exa 8.16	Signal Interpolation using FFT	108
Exa 8.17	The Concept of Periodogram	109
Exa 8.18	DFT from matrix formulation	110
Exa 8.19	Using DFT to find IDFT	110
Exa 8.20	Decimation in Frequency FFT algorithm	110
Exa 8.21	Decimation in time FFT algorithm	111
Exa 8.22	4 point DFT from 3 point sequence	111

Exa 8.23	3 point IDFT from 4 point DFT	112
Exa 8.24	The importance of Periodic Extension	112
Exa 9.1	Response Invariant Mappings	114
Exa 9.2	Impulse Invariant Mappings	117
Exa 9.3ab	Modified Impulse Invariant Design	117
Exa 9.3cd	Modified Impulse Invariant Design	118
Exa 9.5	Mappings from Difference Algorithms	119
Exa 9.6	Mappings From Integration Algorithms	120
Exa 9.7	DTFT of Numerical Algorithms	120
Exa 9.8a	Bilinear Transformation	121
Exa 9.8b	Bilinear Transformation	123
Exa 9.9	D2D transformations	125
Exa 9.10a	Bilinear Design of Second Order Filters	126
Exa 9.10b	Bilinear Design of Second Order Filters	128
Exa 9.10c	Bilinear Design of Second Order Filters	129
Exa 9.11	Interference Rejection	130
Exa 9.12	IIR Filter Design	132
Exa 10.2	Truncation and Windowing	134
Exa 10.3ab	FIR lowpass Filter design	135
Exa 10.3cd	FIR filter Design	136
Exa 10.4a	Half Band lowpass FIR filter Design	139
Exa 10.4b	Half Band bandstop FIR filter Design	141
Exa 10.5a	Design by Frequency Sampling	143
Exa 10.5b	Design by Frequency Sampling	145
Exa 10.6a	Optimal FIR Bandstop Filter Design	147
Exa 10.6b	Optimal Half Band Filter Design	148
Exa 10.7	Multistage Interpolation	149
Exa 10.8	Design of Interpolating Filters	151
Exa 10.9	Multistage Decimation	152
Exa 10.10	Maximally Flat FIR filter Design	153

Chapter 2

Discrete Signals

Scilab code Exa 2.1a Signal energy and power

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

Scilab code Exa 2.1b Average power of periodic signals

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

```

5 p=0;
6 for i=1:1:a
7     p=p+abs(xn(i)^2);
8 end
9 P=p/a;
10 disp("The average power of given signal is")
11 P

```

Scilab code Exa 2.1c Average power of periodic signals

```

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

```

Scilab code Exa 2.2 Operations on Discrete Signals

```

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

```

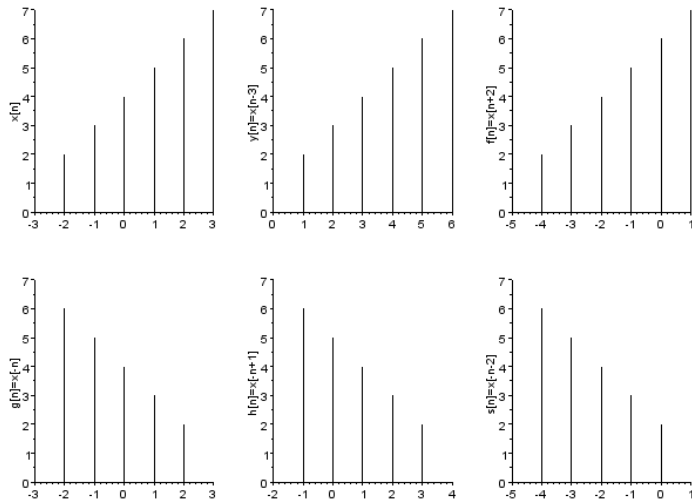


Figure 2.1: Operations on Discrete Signals

```

8 g=x(length(x):-1:1);
9 n4=-3:1:3;
10 h=x(length(x):-1:1);
11 n5=-2:1:4;
12 s=x(length(x):-1:1);
13 n6=-5:1:1;
14 a=gca();
15 subplot(231);
16 plot2d3('gnn',n1,x);
17 ylabel('x[n]');
18 subplot(232);
19 plot2d3('gnn',n2,y);
20 ylabel('y[n]=x[n-3]');
21 subplot(233);
22 plot2d3('gnn',n3,f);
23 ylabel('f[n]=x[n+2]');
24 subplot(234);
25 plot2d3('gnn',n4,g);
26 ylabel('g[n]=x[-n]');
27 subplot(235);

```

```

28 plot2d3('gnn',n5,h);
29 ylabel('h[n]=x[-n+1]');
30 subplot(236);
31 plot2d3('gnn',n6,s);
32 ylabel('s[n]=x[-n-2]');

```

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

```

1 //example 2.3a pg.no.14
2 clear;clc;close;
3 n=-2:2;
4 x1=[4 -2 4 -6 0];
5 x2=0.5*x1//x[n]
6 x3=0.5*[x1(length(x1):-1:1)];//x[-n]
7 xe=(x2+x3);//even part
8 xo=(x2-x3);//odd part
9 a=gca();
10 a.thickness=2;
11 a.x_location="middle";
12 a.y_location="middle";
13 plot2d3('gnn',n,xe,rect=[-4 -6 4 6])
14 xtitle('graphical representation of even part of x[n]
        ','n','x[n]')
15 xset('window',1)
16 b=gca();
17 b.thickness=2;
18 b.y_location="middle";
19 b.x_location="middle";
20 plot2d3('gnn',n,xo,rect=[-2 -4 2 4])
21 xtitle('graphical representation of odd part of x[n]
        ','n','x[n]')

```

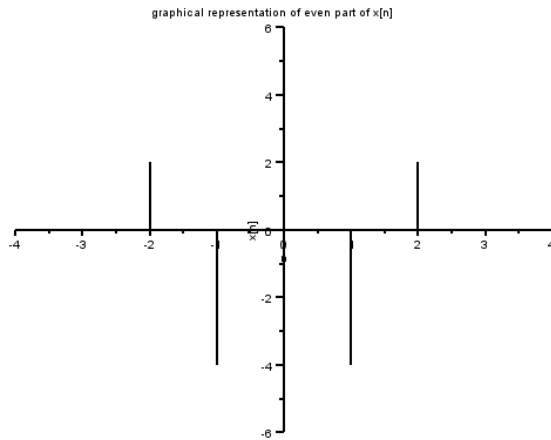


Figure 2.2: Even and Odd parts of Discrete signals

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

```

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

```

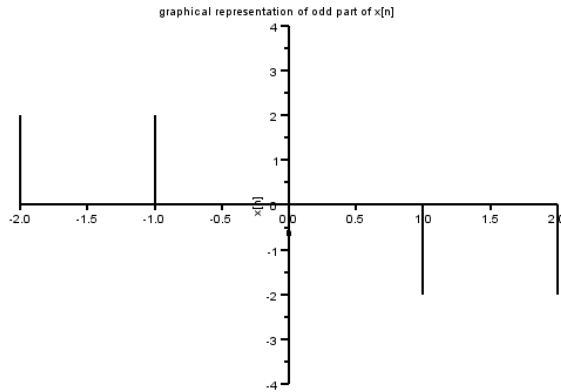


Figure 2.3: Even and Odd parts of Discrete signals

```

14 xtitle('graphical representation of even part of x[n
    ]', 'n', 'x[n]')
15 xset('window', 1)
16 b=gca();
17 b.thickness=2;
18 b.y_location="middle";
19 b.x_location="middle";
20 plot2d3('gnn', n, xo, rect=[-4 -1 4 1]);
21 xtitle('graphical representation of odd part of x[n]
    ', 'n', 'x[n]')

```

Scilab code Exa 2.4a Decimation and Interpolation of Discrete signals

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

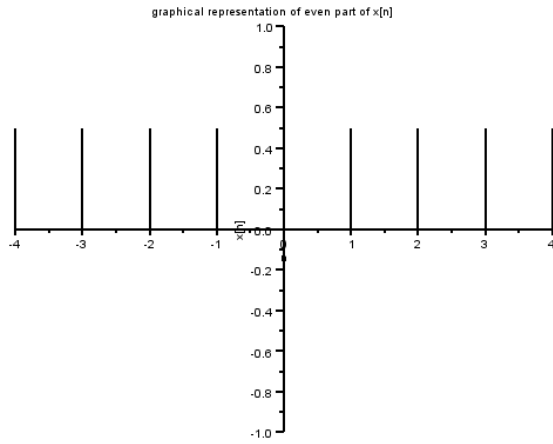


Figure 2.4: Even and Odd parts of Discrete signals

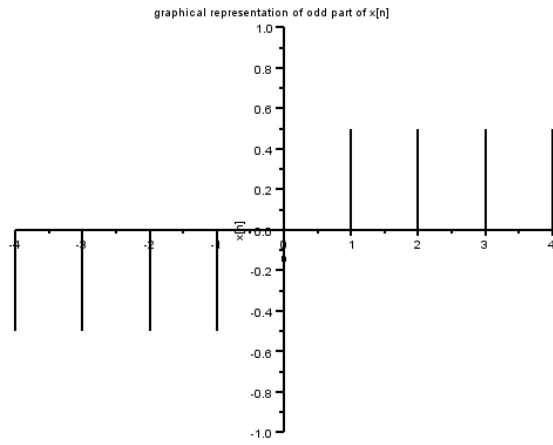


Figure 2.5: Even and Odd parts of Discrete signals

```

2 x=[1 2 5 -1];
3 xm=2; //denotes 2nd sample has pad.
4 y=[x(1:2:xm-2),x(xm:2:length(x))] //decimation
5 h=[x(1:1/3:length(x))] //step interpolated
6 g=h;
7 for i=2:3
8     g(i:3:length(g))=0;
9 end
10 //zero interpolated
11 x1=1:3:3*length(x);
12 s=interpvn([x1;x],1:10) //linear interpolated

```

Scilab code Exa 2.4b Decimation and Interpolation of Discrete signals

```

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

```

Scilab code Exa 2.4c Decimation and Interpolation of Discrete signals

```

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

```

Scilab code Exa 2.4d Decimation and Interpolation of Discrete signals

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

Scilab code Exa 2.5 Describing Sequences and signals

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

Scilab code Exa 2.6 Discrete time Harmonics and Periodicity

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

```

5     if (abs(x(i)-x(1)) < 0.00001)
6         k=2
7         for j=i+1:i+i
8             if (abs(x(j)-x(k)) < 0.00001)
9                 v=v+1
10            end
11            k=k+1;
12        end
13    end
14    if (v==(2*i)) then
15        break
16    end
17 end
18 p=i-1
19 endfunction
20 for i=1:60
21     x1(i)=cos((2*pi*8*i)/25);
22 end
23 for i=1:60
24     x2(i)=exp(i*0.2*i*pi)+exp(-i*0.3*i*pi);
25 end
26 for i=1:45
27     x3(i)=2*cos((40*pi*i)/75)+sin((60*pi*i)/75);
28 end
29 period(x1)
30 period(x2)
31 period(x3)

```

check Appendix [AP 1](#) for dependency:

Aliasfrequency.sci

Scilab code Exa 2.7 Aliasing and its effects

```

1 //example 2.7.pg.no.27
2 f=100;

```

```
3 s=240;
4 s1=s;
5 aliasfrequency(f,s)
6 s=140;
7 s1=s;
8 aliasfrequency(f,s,s1)
9 s=90;
10 s1=s;
11 aliasfrequency(f,s,s1)
12 s=35;
13 s1=s;
14 aliasfrequency(f,s,s1)
```

check Appendix [AP 1](#) for dependency:

Aliasfrequency.sci

Scilab code Exa 2.8 Signal Reconstruction

```
1 f=100;
2 s=210;
3 s1=420;
4 aliasfrequency(f,s,s1)
5 s=140;
6 aliasfrequency(f,s,s1)
```

Chapter 3

Response of Digital Filters

Scilab code Exa 3.5 FIR filter response

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

Scilab code Exa 3.19a Analytical Evaluation of Discrete Convolution

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

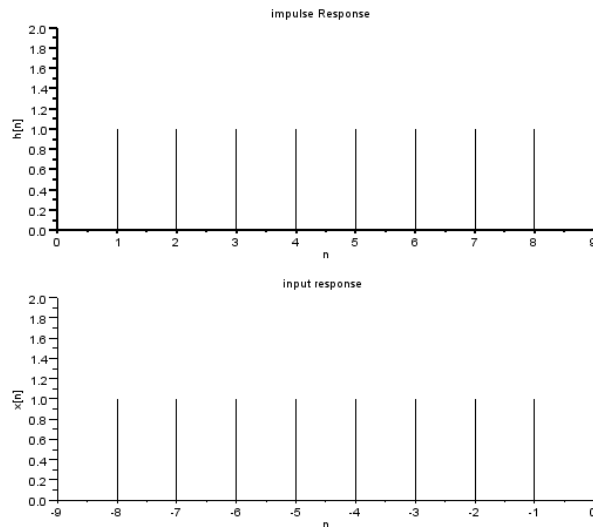


Figure 3.1: Analytical Evaluation of Discrete Convolution

```

7 n1=-length(x)+1:0;
8 y=convol(x,h);
9 n=-length(x)+1:length(h)-1;
10 a=gca();
11 subplot(211);
12 plot2d3('gnn',n2,h)
13 xtitle('impulse Response','n','h[n]');
14 a.thickness=2;
15 a.y_location="origin";
16 subplot(212);
17 plot2d3('gnn',n1,x)
18 a.y_location="origin";
19 xtitle('input response','n','x[n]');
20 xset("window",1);
21 a=gca();
22 plot2d3('gnn',n,y)
23 xtitle('output response','n','y[n]');

```

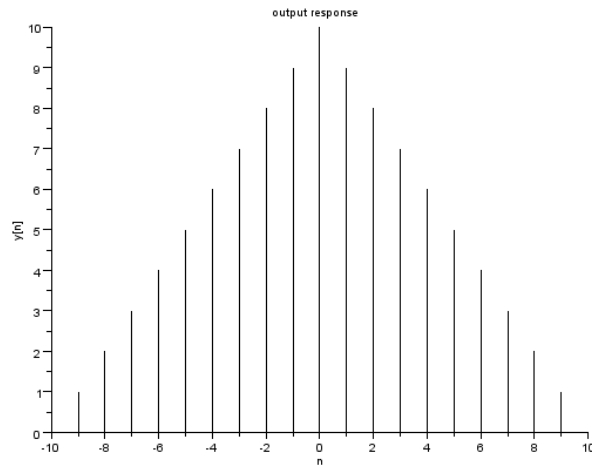


Figure 3.2: Analytical Evaluation of Discrete Convolution

Scilab code Exa 3.19b Analytical Evaluation of Discrete Convolution

```

1  clear; close; clc;
2  max_limit=10;
3  for n=1:max_limit
4      h(n)=(0.4)^n;
5  end
6  n2=0:length(h)-1;
7  for n=1:max_limit
8      x(n)=(0.8)^n;
9  end
10 n1=-length(x)+1:0;
11 y=convol(x,h)
12 n=-length(x)+1:length(h)-1;
13 a=gca();

```

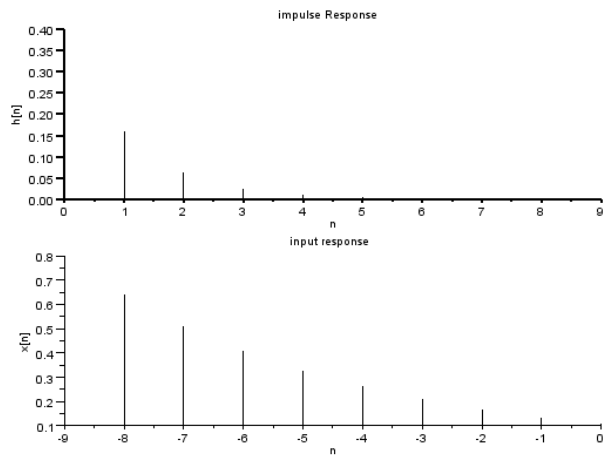


Figure 3.3: Analytical Evaluation of Discrete Convolution

```

14 subplot(211);
15 plot2d3('gnn',n2,h)
16 xtitle('impulse Response','n','h[n]');
17 a.thickness=2;
18 a.y_location="origin";
19 subplot(212);
20 plot2d3('gnn',n1,x)
21 a.y_location="origin";
22 xtitle('input response','n','x[n]');
23 xset("window",1);
24 a=gca();
25 plot2d3('gnn',n,y)
26 xtitle('output response','n','y[n]');

```

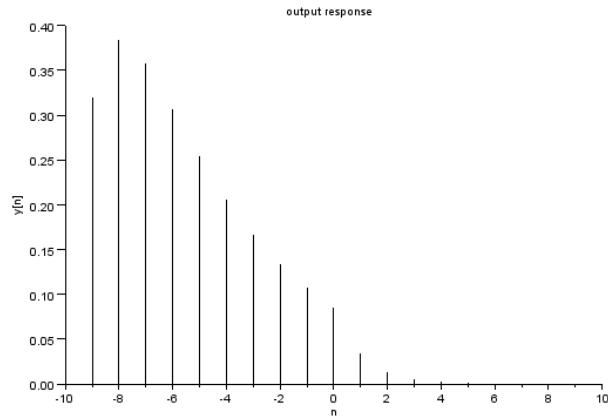


Figure 3.4: Analytical Evaluation of Discrete Convolution

Scilab code Exa 3.19c Analytical Evaluation of Discrete Convolution

```

1 //Analytical Evaluation of Discrete convolution
2 clear;close;clc;
3 max_limit=5;
4 h(1)=0;
5 for n=2:max_limit
6     h(n)=0.8^n;
7 end
8 n2=0:length(h)-1;
9 x=[0 ones(1:max_limit)]
10 n1=-length(x)+1:0;
11 y=convol(x,h);
12 n=-length(x)+1:length(h)-1;
13 a=gca();
14 subplot(211);
15 plot2d3('gnn',n2,h)
16 xtitle('impulse Response','n','h[n]');
17 a.thickness=2;
18 a.y_location="origin";

```

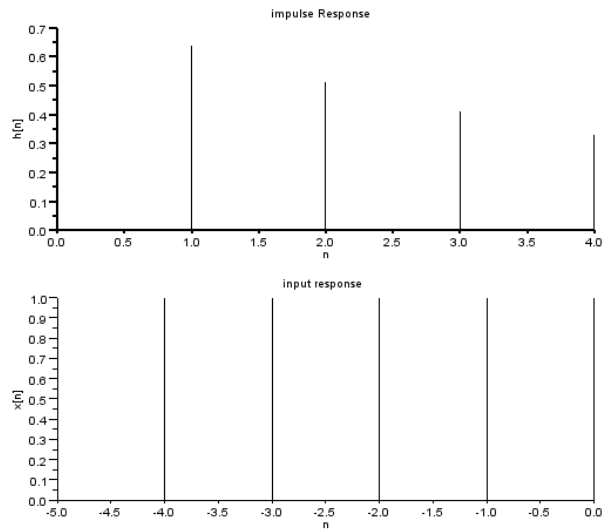



Figure 3.5: Analytical Evaluation of Discrete Convolution

```

19 a=gca();
20 subplot(212);
21 plot2d3('gnn',n1,x)
22 a.y_location="origin";
23 xtitle('input response','n','x[n]');
24 xset("window",1);
25 a=gca();
26 plot2d3('gnn',n,y)
27 a.y_location="origin";
28 a.x_location="origin";
29 xtitle('output response','n','y[n]');

```

Scilab code Exa 3.20a Properties of Convolution

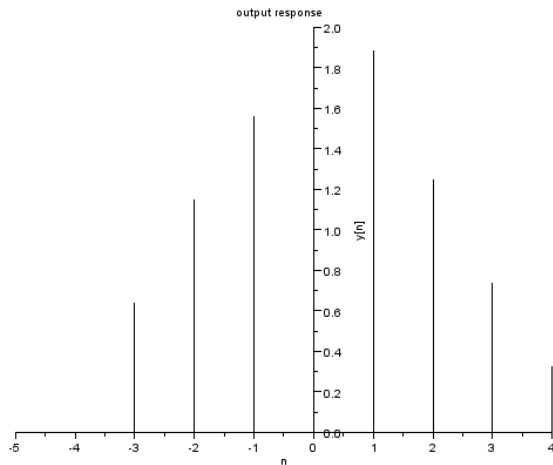


Figure 3.6: Analytical Evaluation of Discrete Convolution

```

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

```

Scilab code Exa 3.20b Properties of Convolution

```

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

```

```
9 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

```
1 //convolution of finite length signals  
2 clear;close;clc;  
3 max_limit=10;  
4 h=[1 2 2 3];  
5 n2=0:length(h)-1;  
6 x=[2 -1 3];  
7 n1=0:length(x)-1;  
8 y=convol(x,h);  
9 n=0:length(h)+length(x)-2;  
10 a=gca();  
11 subplot(211);  
12 plot2d3('ggn',n2,h)  
13 xtitle('impulse Response','n','h[n]');  
14 a.thickness=2;  
15 a.y_location="origin";  
16 a=gca();  
17 subplot(212);  
18 plot2d3('ggn',n1,x)  
19 a.y_location="origin";  
20 a.x_location="origin";  
21 xtitle('input response','n','x[n]');  
22 xset("window",2);  
23 a=gca();  
24 plot2d3('ggn',n,y)  
25 a.y_location="origin";  
26 xtitle('output response','n','y[n]');
```

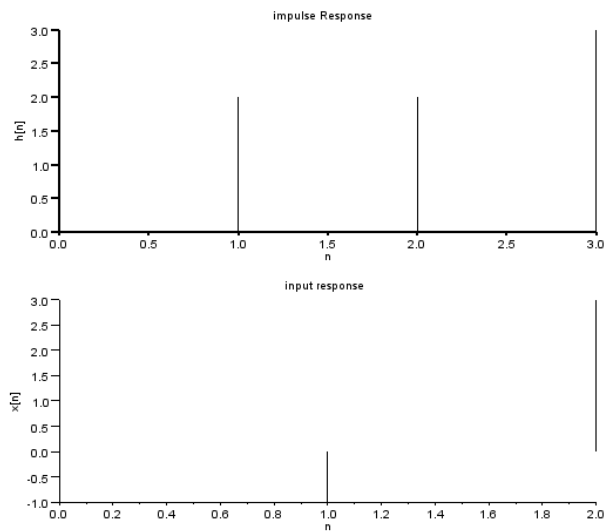


Figure 3.7: Convolution of finite length Signals

Scilab code Exa 3.21b Convolution of finite length Signals

```

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

```

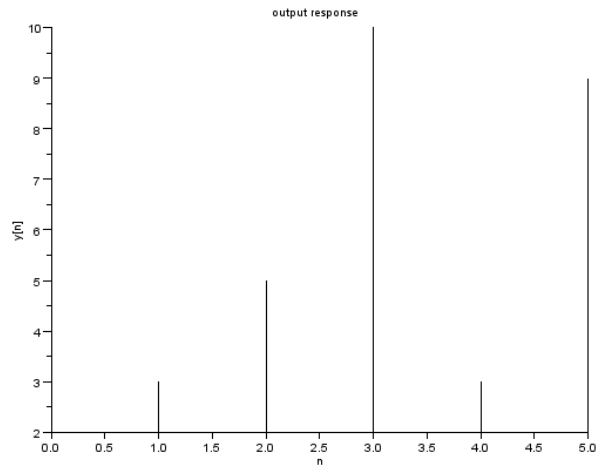


Figure 3.8: Convolution of finite length Signals

```

14 a.y_location="origin";
15 a=gca();
16 subplot(212);
17 plot2d3('gnn',n1,x)
18 a.y_location="origin";
19 xtitle('input response','n','x[n]');
20 xset("window",1);
21 a=gca();
22 plot2d3('gnn',n,y)
23 a.y_location="origin";
24 a.x_location="origin";
25 xtitle('output response','n','y[n]');

```

Scilab code Exa 3.21c Convolution of finite length Signals

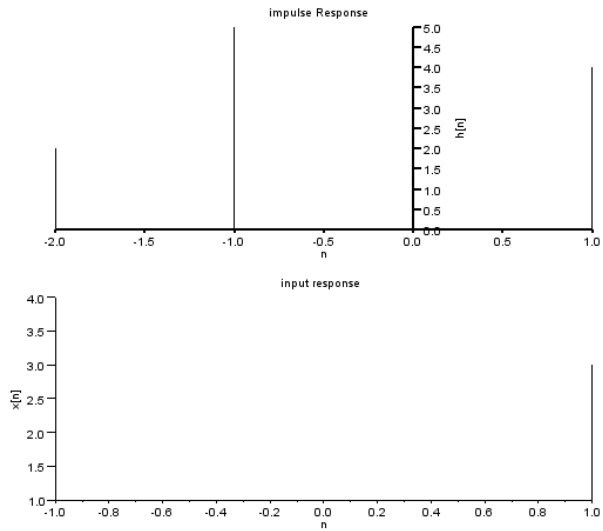


Figure 3.9: Convolution of finite length Signals

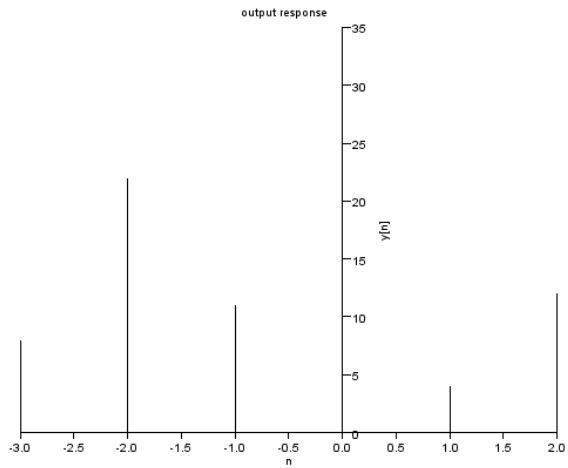


Figure 3.10: Convolution of finite length Signals

```

1 clear;close;clc;
2 max_limit=10;
3 h=[1/2 1/2 1/2];
4 n2=0:length(h)-1;
5 x=[2 4 6 8 10];
6 n1=0:length(x)-1;
7 y=convol(x,h);
8 n=0:length(x)+length(h)-2;
9 a=gca();
10 subplot(211);
11 plot2d3('gnn',n2,h);
12 xtitle('impulse Response','n','h[n]');
13 a.thickness=2;
14 a.y_location="origin";
15 a=gca();
16 subplot(212);
17 plot2d3('gnn',n1,x);
18 a.y_location="origin";
19 xtitle('input response','n','x[n]');
20 xset("window",1);
21 a=gca();
22 plot2d3('gnn',n,y)
23 a.y_location="origin";
24 a.x_location="origin";
25 xtitle('output response','n','y[n]');

```

Scilab code Exa 3.22 Convolution of finite length Signals

```

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

```

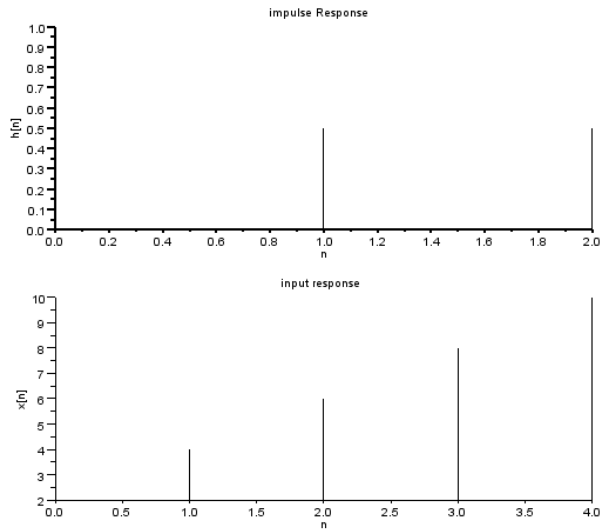


Figure 3.11: Convolution of finite length Signals

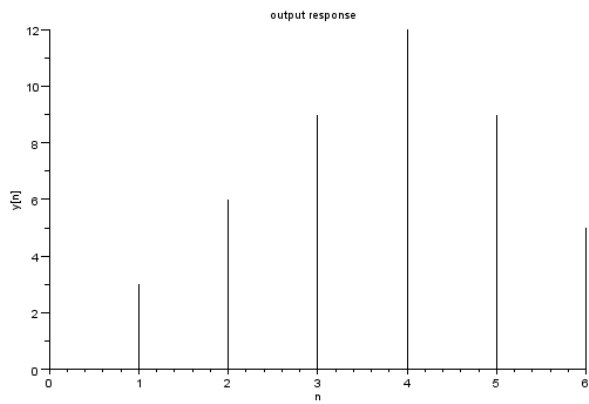


Figure 3.12: Convolution of finite length Signals


```

4 x=[4 1 3];
5 n1=0:length(x)-1;
6 y=convol(x,h);
7 n=0:length(x)+length(h)-2;
8 a=gca();
9 subplot(211);
10 plot2d3('gmn',n2,h)
11 xtitle('impulse Response','n','h[n]');
12 a.thickness=2;
13 a.y_location="origin";
14 a=gca();
15 subplot(212);
16 plot2d3('gmn',n1,x)
17 a.y_location="origin";
18 a.x_location="origin";
19 xtitle('input response','n','x[n]');
20 xset("window",1);
21 a=gca();
22 plot2d3('gmn',n,y)
23 a.y_location="origin";
24 a.x_location="origin";
25 xtitle('output response','n','y[n]');

```

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

```

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

```

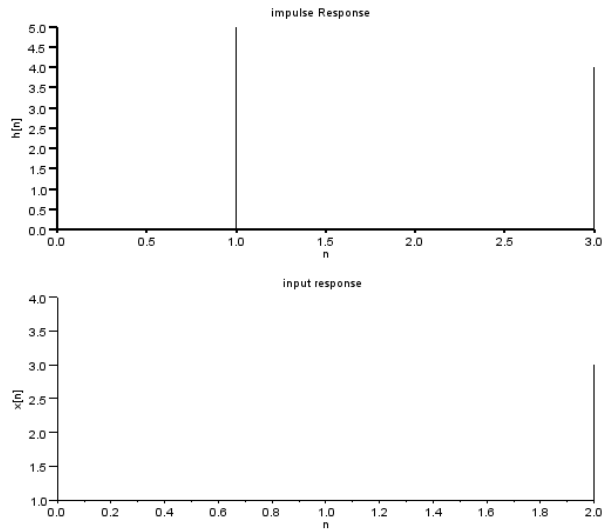


Figure 3.13: Convolution of finite length Signals

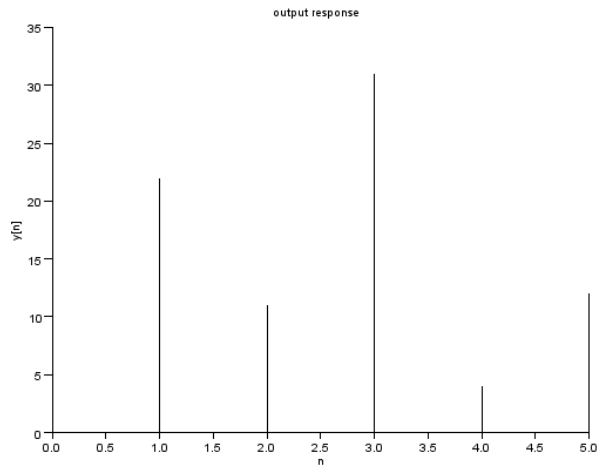


Figure 3.14: Convolution of finite length Signals

```

7 n1=length(h)-1:-1:0;
8 H=h*z^n1';
9 y=X*H
10 //effect of zero insertion on convolution
11 h=[2 0 5 0 0 0 4];
12 x=[4 0 1 0 3];
13 y=convol(x,h)
14 //effect of zero padding on convolution
15 h=[2 5 0 4 0 0];
16 x=[4 1 3 0];
17 y=convol(x,h)

```

Scilab code Exa 3.25 Stability and Causality

```

1 //concepts based on stability and Causality
2 function []=stability(X)
3 if (abs(roots(X))<1)
4     disp("given system is stable")
5 else
6     disp("given system is not stable")
7 end
8 endfunction
9 x=[1 -1/6 -1/6];
10 z=%z;
11 n=length(x)-1:-1:0;
12 //characteristic eqn is
13 X=x*(z)^n'
14 stability(X)
15 x=[1 -1];
16 n=length(x)-1:-1:0;
17 //characteristic eqn is
18 X=x*(z)^n'
19 stability(X)
20 x=[1 -2 1];
21 n=length(x)-1:-1:0;

```

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

Scilab code Exa 3.26 Response to Periodic Inputs

```
1 //Response of periodic inputs
2 function [p]=period(x)
3 for i=2:length(x)
4     v=i
5     if (abs(x(i)-x(1))<0.00001)
6         k=2
7         for j=i+1:i+i
8             if (abs(x(j)-x(k))<0.00001)
9                 v=v+1
10            end
11            k=k+1;
12        end
13    end
14    if (v==(2*i)) then
15        break
16    end
17 end
18 p=i-1
19 endfunction
20 x=[1 2 -3 1 2 -3 1 2 -3];
21 h=[1 1];
22 y=convol(x,h)
23 y(1)=y(4);
24 period(x)
25 period(y)
26 h=[1 1 1];
27 y=convol(x,h)
```

Scilab code Exa 3.27 Periodic Extension

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

Scilab code Exa 3.28 System Response to Periodic Inputs

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

```

18 y2=[y1(1:3);y1(4:6);y1(7:9)];
19 y=[0 0 0];
20 for i=1:3
21     for j=1:3
22         y(i)=y(i)+y2(j,i);
23     end
24 end
25 y

```

Scilab code Exa 3.29 Periodic Convolution

```

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

```

Scilab code Exa 3.30 Periodic Convolution by Circulant Matrix

```

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

```

```

7     c(i,:)=[x(length(x):length(x)-i),x(1:length(x)-i
           )]
8 end
9 c'

```

Scilab code Exa 3.32 Deconvolution By polynomial Division

```

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

```

Scilab code Exa 3.33 Autocorrelation and Cross Correlation

```

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

```

Scilab code Exa 3.35 Periodic Autocorrelation and Cross Correlation

```
1 //discrete periodic auto correlation and cross
  correlation
2 x=[2 5 0 4];
3 h=[3 1 -1 2];
4 x1=x(length(x):-1:1);
5 h1=h(length(h):-1:1);
6 rxhn=convol(x,h1)
7 rhxn=convol(x1,h)
8 rxxn=convol(x,x1)
9 rhhn=convol(h,h1)
10 y1=[rxhn,zeros(8:12)];
11 y2=[y1(1:4);y1(5:8);y1(9:12)];
12 y3=[rhxn,zeros(8:12)];
13 y4=[y3(1:4);y3(5:8);y3(9:12)];
14 y5=[rxxn,zeros(8:12)];
15 y6=[y5(1:4);y5(5:8);y5(9:12)];
16 y7=[rhhn,zeros(8:12)];
17 y8=[y7(1:4);y7(5:8);y7(9:12)];
18 rxhp=[0 0 0 0];
19 rhxp=[0 0 0 0];
20 rxxn=[0 0 0 0];
21 rhhp=[0 0 0 0];
22 for i=1:4
23     for j=1:3
24         rhxp(i)=rhxp(i)+y4(j,i);
25         rxhp(i)=rxhp(i)+y2(j,i);
26         rxxn(i)=rxxn(i)+y6(j,i);
27         rhhp(i)=rhhp(i)+y8(j,i);
28     end
29 end
30 rxhp
31 rhxp
```


32 rxxn

33 rhhp

Chapter 4

z Transform Analysis

Scilab code Exa 4.1b z transform of finite length sequences

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

Scilab code Exa 4.4a Pole Zero Plots

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

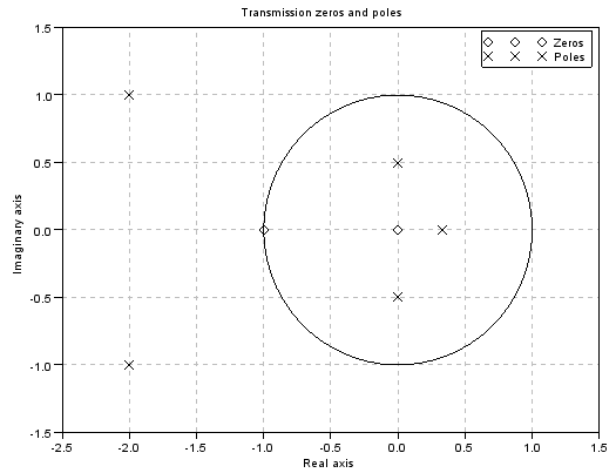


Figure 4.1: Pole Zero Plots

8 `plzr(h)`

Scilab code Exa 4.4b Pole Zero plots

```

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

```

Scilab code Exa 4.8 Stability of Recursive Filters

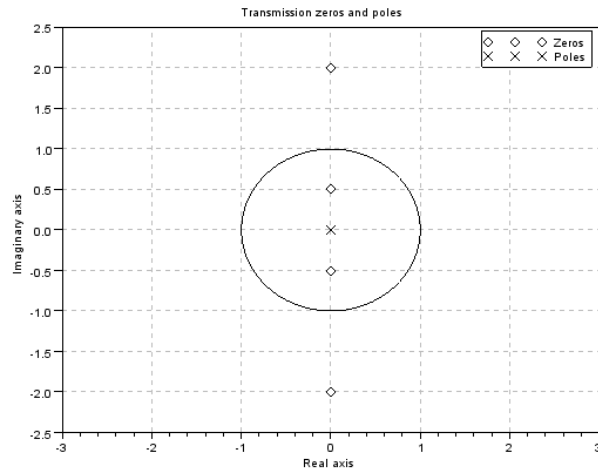


Figure 4.2: Pole Zero plots

```

1 //stability of recursive filter
2 //for roc: /z/ > /a/
3 a=input('enter the value of alpha')
4 z=%z;
5 H=z/(z-a);
6 if (abs(a)<1)
7     disp("system is stable")
8 else
9     disp("system is not stable")
10 end
11 //for roc: /z/ < /a/
12 if (abs(a)>1)
13     disp("system is stable")
14 else
15     disp("system is not stable")
16 end

```

Scilab code Exa 4.9 Inverse Systems

```

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

```

Scilab code Exa 4.10 Inverse Transform of sequences

```

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

```

Scilab code Exa 4.11 Inverse Transform by Long Division

```

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

```

Scilab code Exa 4.12 Inverse transform of Right sided sequences

```
1 //inverse z transforms from standard transforms
2 z=%z;
3 xz=z/((z-0.5)*(z-0.25));
4 yz=xz/z;
5 pfs(yz)
6 //hence  $x[n] = -4(0.25)^n u[n] + 4(0.5)^n u[n]$ ;
7 xz=1/((z-0.5)*(z-0.25));
8 yz=xz/z;
9 pfs(yz)
10 //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

```
1 //z transform of switched periodic signals
2 z=%z;
3 //sol. for 4.20a
4 x1=[0 1 2];
5 n=0:2;
6 N=3;
7 x1z=x1*(1/z)^n;
8 xz=x1z/(1-z^-N)
9 //sol. for 4.20b
10 x1=[0 1 0 -1];
11 n=0:3;
12 N=4;
13 x1z=x1*(1/z)^n;
14 xz=x1z/(1-z^-N)
15 //sol. for 4.20c
16 xz=(2+z^-1)/(1-z^-3);
17 x1z=numer(xz)
18 //thus first period of xn is [2 1 0]
19 //sol. for 4.20d
20 xz=(z^-1-z^-4)/(1-z^-6);
```

```
21 x1z=numer(xz)
22 //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

```
1 //DTFT of  $x[n]=(a)^n*u[n]$ 
2 clear;
3 clc;close;
4 //DTS signal
5 a1=0.5;
6 a2=-0.5;
7 max_limit=10;
8 for n=0:max_limit-1
9     x1(n+1)=(a1^n);
10    x2(n+1)=(a2^n);
11 end
12 n=0:max_limit-1;
13 //discrete time fourier transform
14 wmax=2*%pi;
15 K=4;
16 k=0:(K/1000):K;
17 W=k*wmax/K;
18 x1=x1';
19 x2=x2';
20 XW1=x1*exp(%i*n'*W);
21 XW2=x2*exp(%i*n'*W);
```



```

22 XW1_Mag=abs(XW1);
23 XW2_Mag=abs(XW2);
24 W=[-mtlbfliplr(W),W(2:1001)]; //omega form
25 XW1_Mag=[mtlbfliplr(XW1_Mag),XW1_Mag(2:1001)];
26 XW2_Mag=[mtlbfliplr(XW2_Mag),XW2_Mag(2:1001)];
27 [XW1_phase,db]=phasemag(XW1);
28 [XW2_phase,db]=phasemag(XW2);
29 XW1_phase=[-mtlbfliplr(XW1_phase),XW1_phase(2:1001)
    ];
30 XW2_phase=[-mtlbfliplr(XW2_phase),XW2_phase(2:1001)
    ];
31
32 //plot for a>0
33 figure
34 subplot(3,1,1);
35 plot2d3('gnn',n,x1)
36 xtitle('Discrete time sequencex[n] a>0')
37 subplot(3,1,2);
38 a=gca();
39 a.y_location="origin";
40 a.x_location="origin";
41 plot2d3(W,XW1_Mag);
42 title('magnitude Response abs(exp(jw))')
43 subplot(3,1,3);
44 a=gca();
45 a.y_location="origin";
46 a.x_location="origin";
47 plot2d(W,XW1_phase);
48 title('magnitude Response abs(exp(jw))')
49 //plot for a<0
50 figure
51 subplot(3,1,1);
52 plot2d3('gnn',n,x2);
53 xtitle('Discrete Time sequence x[n] for a>0')
54 subplot(3,1,2);
55 a=gca();
56 a.y_location="origin";
57 a.x_location="origin";

```

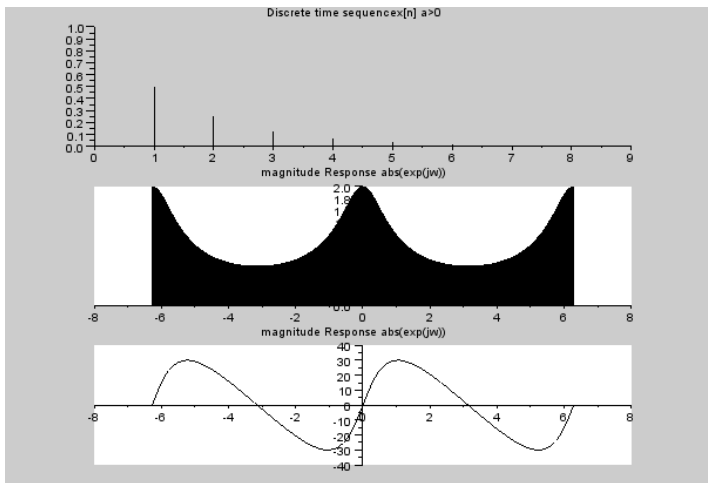


Figure 5.1: DTFT from Defining Relation

```

58 plot2d(W,XW2_Mag);
59 title('Magnitude Response abs(X(jw))')
60 subplot(3,1,3);
61 a=gca();
62 a.y_location="origin";
63 a.x_location="origin";
64 plot2d(W,XW2_phase);
65 title('phase Response <(X(jw))')

```

Scilab code Exa 5.3a Some DTFT pairs using properties

```

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

```

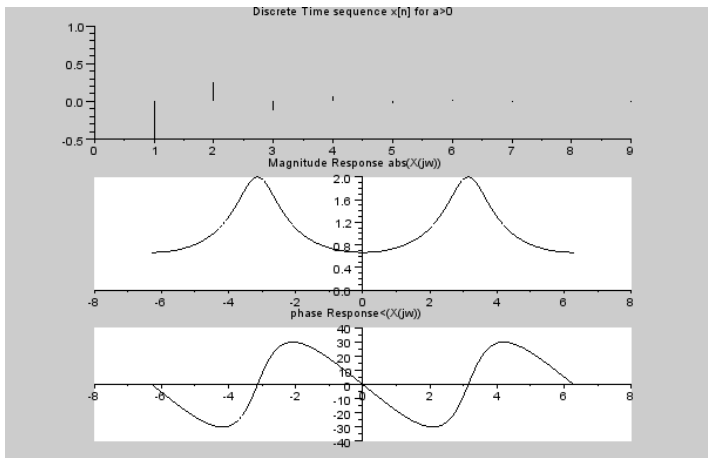


Figure 5.2: DTFT from Defining Relation

```

4 //DTS signal
5 a1=0.5;
6 a2=-0.5;
7 max_limit=10;
8 for n=0:max_limit-1
9     x1(n+1)=n*(a1^n);
10    x2(n+1)=n*(a2^n);
11 end
12 n=0:max_limit-1;
13 //discrete time fourier transform
14 wmax=2*%pi;
15 K=4;
16 k=0:(K/1000):K;
17 W=k*wmax/K;
18 x1=x1';
19 x2=x2';
20 XW1=x1*exp(%i*n'*W);
21 XW2=x2*exp(%i*n'*W);
22 XW1_Mag=abs(XW1);
23 XW2_Mag=abs(XW2);

```

```

24 W=[-mtlbfliplr(W),W(2:1001)]; //omega form
25 XW1_Mag=[mtlbfliplr(XW1_Mag),XW1_Mag(2:1001)];
26 XW2_Mag=[mtlbfliplr(XW2_Mag),XW2_Mag(2:1001)];
27 [XW1_phase,db]=phasemag(XW1);
28 [XW2_phase,db]=phasemag(XW2);
29 XW1_phase=[-mtlbfliplr(XW1_phase),XW1_phase(2:1001)
    ];
30 XW2_phase=[-mtlbfliplr(XW2_phase),XW2_phase(2:1001)
    ];
31
32 //plot for a>0
33 figure
34 subplot(3,1,1);
35 plot2d3('ggn',n,x1)
36 xtitle('Discrete time sequence x[n] a>0')
37 subplot(3,1,2);
38 a=gca();
39 a.y_location="origin";
40 a.x_location="origin";
41 plot2d3(W,XW1_Mag);
42 title('magnitude Response abs(exp(jw))')
43 subplot(3,1,3);
44 a=gca();
45 a.y_location="origin";
46 a.x_location="origin";
47 plot2d(W,XW1_phase);
48 title('magnitude Response abs(exp(jw))')
49 //plot for a<0
50 figure
51 subplot(3,1,1);
52 plot2d3('ggn',n,x2);
53 xtitle('Discrete Time sequence x[n] for a>0')
54 subplot(3,1,2);
55 a=gca();
56 a.y_location="origin";
57 a.x_location="origin";
58 plot2d(W,XW2_Mag);
59 title('Magnitude Response abs(X(jw))')

```

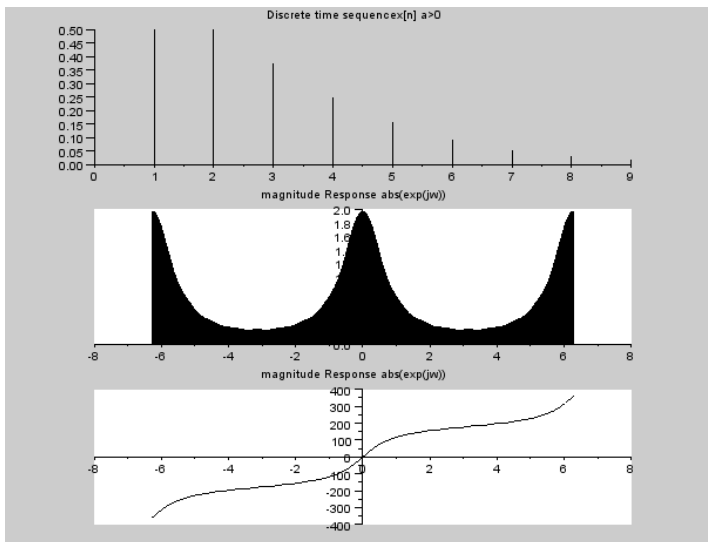


Figure 5.3: Some DTFT pairs using properties

```

60 subplot(3,1,3);
61 a=gca();
62 a.y_location="origin";
63 a.x_location="origin";
64 plot2d(W,XW2_phase);
65 title('phase Response<(X(jw))')

```

Scilab code Exa 5.3b Some DTFT pairs using properties

```

1 //DTFT of x[n]=n*(a)^n*u[n]
2 clear;
3 clc;close;
4 //DTS signal
5 a1=0.5;

```

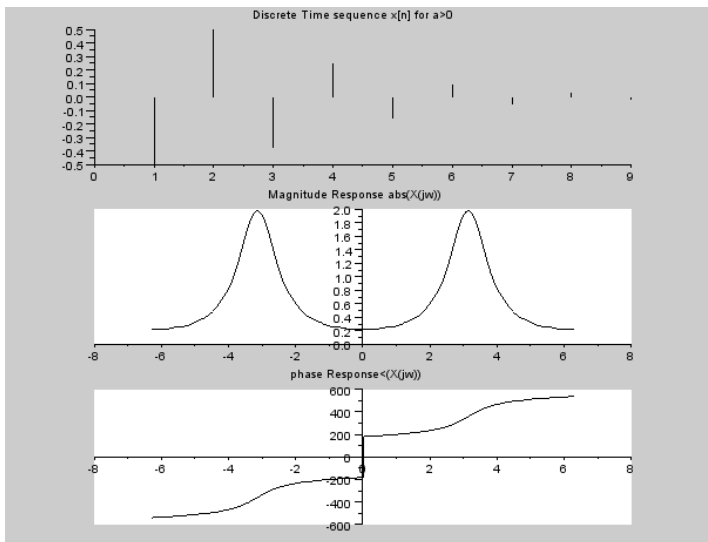


Figure 5.4: Some DTFT pairs using properties

```

6 a2=-0.5;
7 max_limit=10;
8 for n=0:max_limit-1
9     x1(n+1)=(n+1)*(a1^n);
10    x2(n+1)=(n+1)*(a2^n);
11 end
12 n=0:max_limit-1;
13 //discrete time fourier transform
14 wmax=2*%pi;
15 K=4;
16 k=0:(K/1000):K;
17 W=k*wmax/K;
18 x1=x1';
19 x2=x2';
20 XW1=x1*exp(%i*n'*W);
21 XW2=x2*exp(%i*n'*W);
22 XW1_Mag=abs(XW1);
23 XW2_Mag=abs(XW2);
24 W=[-mtlbfliplr(W),W(2:1001)]; //omega form
25 XW1_Mag=[mtlbfliplr(XW1_Mag),XW1_Mag(2:1001)];

```

```

26 XW2_Mag=[mtlbfliplr(XW2_Mag),XW2_Mag(2:1001)];
27 [XW1_phase,db]=phasemag(XW1);
28 [XW2_phase,db]=phasemag(XW2);
29 XW1_phase=[-mtlbfliplr(XW1_phase),XW1_phase(2:1001)
];
30 XW2_phase=[-mtlbfliplr(XW2_phase),XW2_phase(2:1001)
];
31
32 //plot for a>0
33 figure
34 subplot(3,1,1);
35 plot2d3('gmn',n,x1)
36 xtitle('Discrete time sequencex[n] a>0')
37 subplot(3,1,2);
38 a=gca();
39 a.y_location="origin";
40 a.x_location="origin";
41 plot2d3(W,XW1_Mag);
42 title('magnitude Response abs(exp(jw))')
43 subplot(3,1,3);
44 a=gca();
45 a.y_location="origin";
46 a.x_location="origin";
47 plot2d(W,XW1_phase);
48 title('magnitude Response abs(exp(jw))')
49 //plot for a<0
50 figure
51 subplot(3,1,1);
52 plot2d3('gmn',n,x2);
53 xtitle('Discrete Time sequence x[n] for a>0')
54 subplot(3,1,2);
55 a=gca();
56 a.y_location="origin";
57 a.x_location="origin";
58 plot2d(W,XW2_Mag);
59 title('Magnitude Response abs(X(jw))')
60 subplot(3,1,3);
61 a=gca();

```

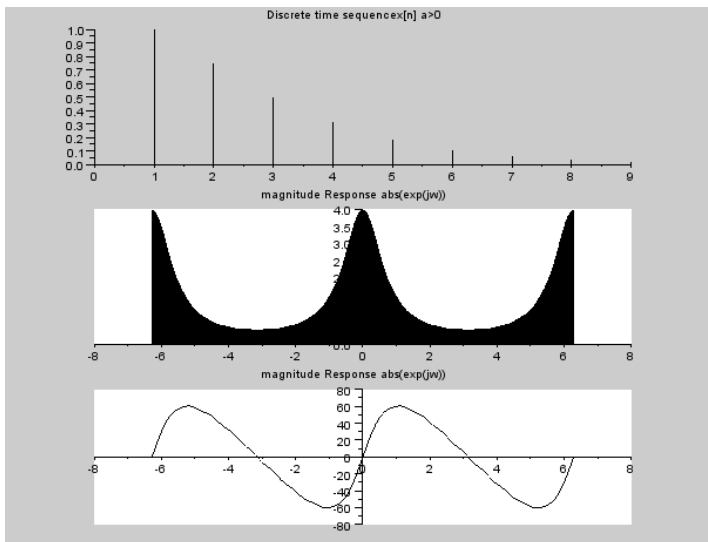


Figure 5.5: Some DTFT pairs using properties

```

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

```

Scilab code Exa 5.3d Some DTFT pairs using properties

```

1 //DTFT of x[n]=a^abs(n)
2 a=0.5;
3 n=-9:9;
4 x=a^abs(n);
5 //Discrete time Fourier Transform

```

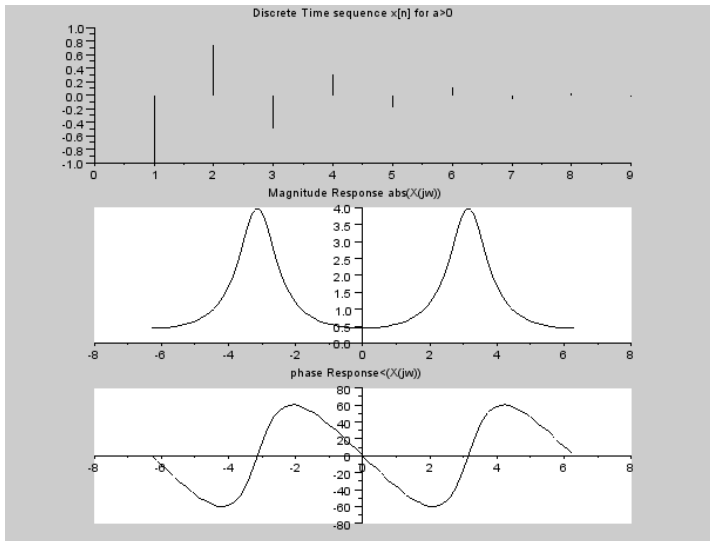



Figure 5.6: Some DTFT pairs using properties

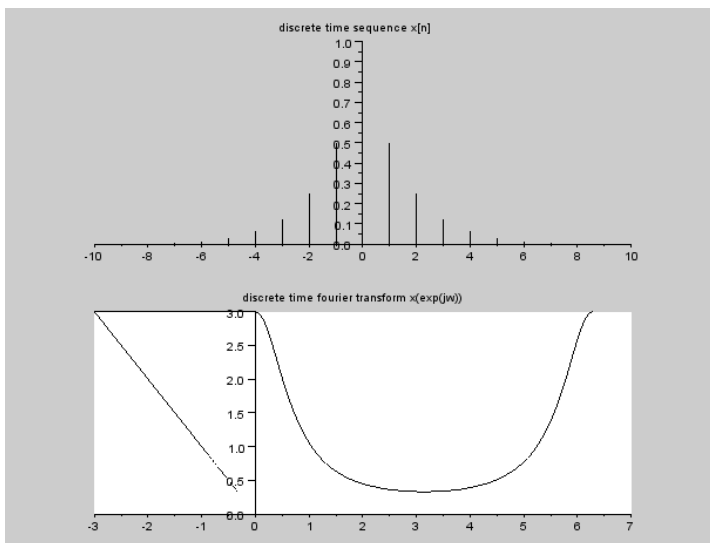


Figure 5.7: Some DTFT pairs using properties

```

6 k=0:(4/1000):4;
7 w=(2*%pi*k)/4;
8 xw=x*exp(%i*n'*w);
9 xw_mag=real(xw);
10 w=[-mtlbfliplr(xw_mag),w(2:1001)];
11 xw_mag=[mtlbfliplr(xw_mag),xw_mag(2:1001)];
12 figure
13 subplot(2,1,1);
14 a=gca();
15 a.x_location="origin";
16 a.y_location="origin";
17 plot2d3('gmn',n,x);
18 xtitle('discrete time sequence x[n]');
19 subplot(2,1,2);
20 a=gca();
21 a.x_location="origin";
22 a.y_location="origin";
23 plot2d(w,xw_mag);
24 title('discrete time fourier transform x(exp(jw))');

```

Scilab code Exa 5.3e Some DTFT pairs using properties

```

1 //DTFT of  $x[n]=n*(a)^n*u[n]$ 
2 clear;
3 clc;close;
4 //DTS signal
5 a1=0.5;
6 a2=-0.5;
7 max_limit=10;
8 for n=0:max_limit-1
9     x1(n+1)=4*(a1^(n+3));
10    x2(n+1)=4*(a2^(n+3));
11 end
12 n=0:max_limit-1;
13 //discrete time fourier transform

```

```

14 wmax=2*%pi;
15 K=4;
16 k=0:(K/1000):K;
17 W=k*wmax/K;
18 x1=x1';
19 x2=x2';
20 XW1=x1*exp(%i*n'*W);
21 XW2=x2*exp(%i*n'*W);
22 XW1_Mag=abs(XW1);
23 XW2_Mag=abs(XW2);
24 W=[-mtlbfliplr(W),W(2:1001)]; //omega form
25 XW1_Mag=[mtlbfliplr(XW1_Mag),XW1_Mag(2:1001)];
26 XW2_Mag=[mtlbfliplr(XW2_Mag),XW2_Mag(2:1001)];
27 [XW1_phase,db]=phasemag(XW1);
28 [XW2_phase,db]=phasemag(XW2);
29 XW1_phase=[-mtlbfliplr(XW1_phase),XW1_phase(2:1001)
    ];
30 XW2_phase=[-mtlbfliplr(XW2_phase),XW2_phase(2:1001)
    ];
31
32 //plot for a>0
33 figure
34 subplot(3,1,1);
35 plot2d3('gnn',n,x1)
36 xtitle('Discrete time sequencex[n] a>0')
37 subplot(3,1,2);
38 a=gca();
39 a.y_location="origin";
40 a.x_location="origin";
41 plot2d3(W,XW1_Mag);
42 title('magnitude Response abs(exp(jw))')
43 subplot(3,1,3);
44 a=gca();
45 a.y_location="origin";
46 a.x_location="origin";
47 plot2d(W,XW1_phase);
48 title('magnitude Response abs(exp(jw))')
49 //plot for a<0

```

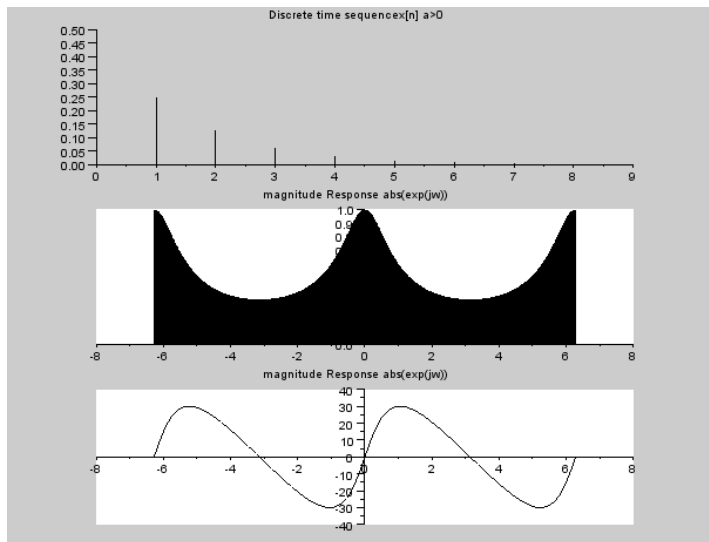


Figure 5.8: Some DTFT pairs using properties

```

50 figure
51 subplot(3,1,1);
52 plot2d3('ggn',n,x2);
53 xtitle('Discrete Time sequence x[n] for a>0')
54 subplot(3,1,2);
55 a=gca();
56 a.y_location="origin";
57 a.x_location="origin";
58 plot2d(W,XW2_Mag);
59 title('Magnitude Response abs(X(jw))')
60 subplot(3,1,3);
61 a=gca();
62 a.y_location="origin";
63 a.x_location="origin";
64 plot2d(W,XW2_phase);
65 title('phase Response <(X(jw))')

```

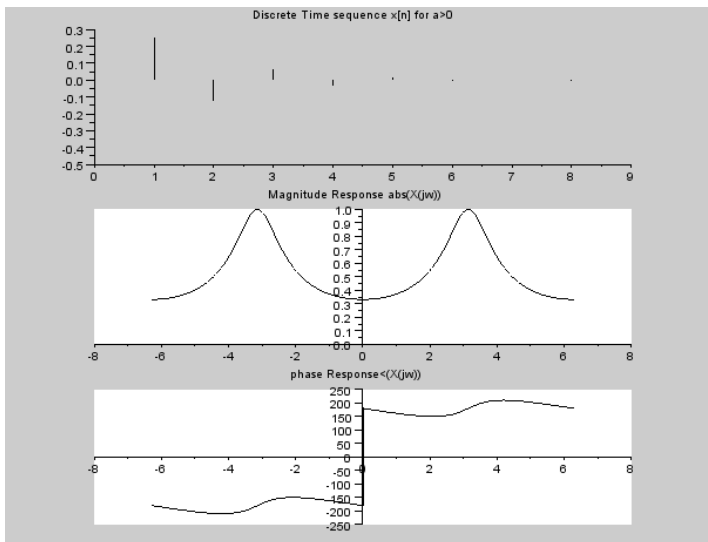


Figure 5.9: Some DTFT pairs using properties

Scilab code Exa 5.4 DTFT of periodic Signals

```

1 //DTfT of periodic signals
2 x=[3 2 1 2]; //one period of signal
3 n=0:3;
4 k=0:3;
5 x1=x*exp(%i*n'*2*k*%pi/4)
6 dtftx=abs(x1)
7 x=[3 2 1 2 3 2 1 2 3];
8 n=-4:4;
9 a=gca();
10 a.y_location="origin";
11 a.x_location="origin";
12 plot2d3('ggn',n,x);
13 xtitle('discrete periodic time signal');

```

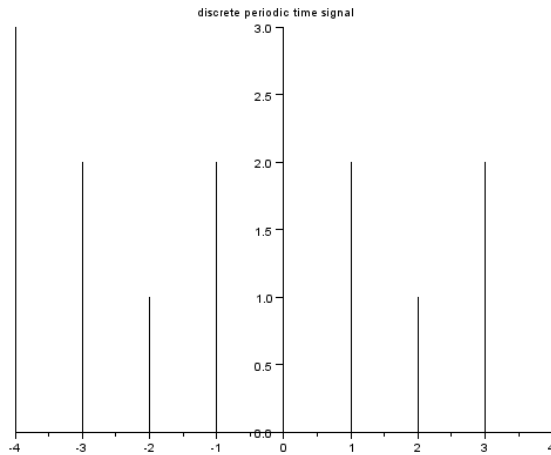


Figure 5.10: DTFT of periodic Signals

```

14 x2=[dtftx dtftx 8];
15 a=gca();
16 xset('window',1);
17 a.x_location="origin";
18 a.y_location="origin";
19 plot2d3('gmn',n,x2);
20 xtitle('DTFT of discrete periodic signal');

```

Scilab code Exa 5.5 The DFT,DFS and DTFT

```

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

```

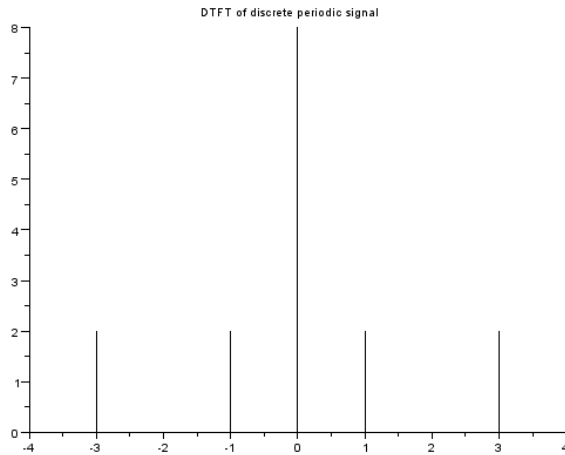


Figure 5.11: DTFT of periodic Signals

```

5 DTFTx=abs(x1)
6 DFT=fft(x,-1)
7 DFS=DFT/5

```

Scilab code Exa 5.7 Frequency Response of Recursive Filter

```

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

```

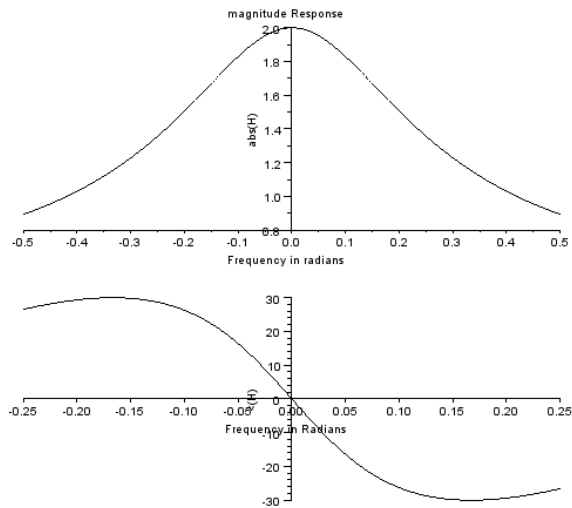


Figure 5.12: Frequency Response of Recursive Filter

```

11 H=abs(H);
12 a=gca();
13 subplot(2,1,1);
14 a.y_location="origin";
15 plot2d(w/%pi,H);
16 xlabel('Frequency in radians')
17 ylabel('abs(H)')
18 title('magnitude Response')
19 subplot(2,1,2);
20 a=gca();
21 a.x_location="origin";
22 a.y_location="origin";
23 plot2d(w/(2*%pi),phase_H)
24 xlabel('Frequency in Radians');
25 ylabel('<(H)')
26 title('Phase Response'))

```

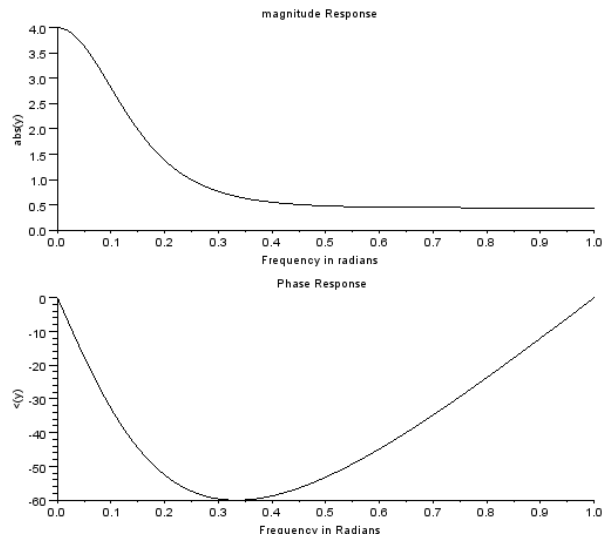


Figure 5.13: The DTFT in System Analysis

Scilab code Exa 5.8a The DTFT in System Analysis

```

1 //DTFT in system analysis
2 a=0.5;b=1;
3 n=0:50;
4 h=b*(a^n);
5 //Discrete-Time Fourier transform
6 K=500;
7 k=0:1:K;
8 w=%pi*k/K;
9 H=h*exp(-%i*n'*w);
10 //x[n] is given as (a)^n*u[n]
11 xw=h*exp(-%i*n'*w);
12 for i=1:501
13     y(i)=H(i)*xw(i);
14 end
15 [phase_y ,m]=phasemag(y);

```

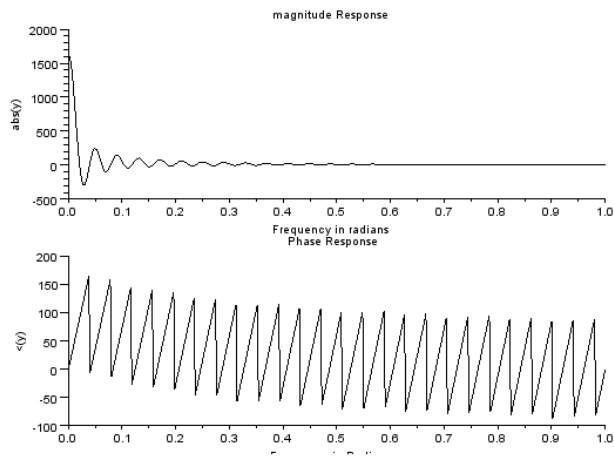


Figure 5.14: The DTFT in System Analysis

```

16 y=real(y);
17 subplot(2,1,1)
18 plot2d(w/%pi,y);
19 xlabel('Frequency in radians')
20 ylabel('abs(y)')
21 title('magnitude Response')
22 subplot(2,1,2)
23 plot2d(w/%pi,phase_y)
24 xlabel('Frequency in Radians');
25 ylabel('<y>')
26 title('Phase Response')

```

Scilab code Exa 5.8b The DTFT in System Analysis

```

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

```

```

3 h=4*(a^n);
4 //Discrete-Time Fourier transform
5 K=500;
6 k=0:1:K;
7 w=%pi*k/K;
8 H=h*exp(-%i*n'*w);
9 //x[n] is given as (a)^n*u[n]
10 x=4*[ones(1:51)];
11 xw=x*exp(%i*n'*w);
12 for i=1:501
13     y(i)=H(i)*xw(i);
14 end
15 [phase_y,m]=phasemag(y);
16 y=real(y);
17 subplot(2,1,1);
18 plot2d(w/%pi,y);
19 xlabel('Frequency in radians')
20 ylabel('abs(y)')
21 title('magnitude Response')
22 subplot(2,1,2)
23 plot2d(w/%pi,phase_y)
24 xlabel('Frequency in Radians');
25 ylabel('<(y)')
26 title('Phase Response')

```

Scilab code Exa 5.9a DTFT and steady state response

```

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

```

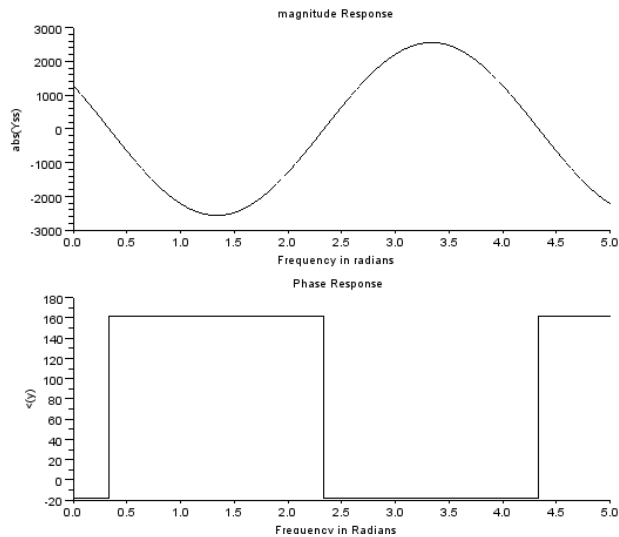


Figure 5.15: DTFT and steady state response

```

7 Yss=H*x;
8 [phase_Yss ,m]=phasemag(Yss);
9 Yss=real(Yss);
10 subplot(2,1,1)
11 plot2d(n,Yss);
12 xlabel('Frequency in radians')
13 ylabel('abs(Yss)')
14 title('magnitude Response')
15 subplot(2,1,2)
16 plot2d(n,phase_Yss)
17 xlabel('Frequency in Radians');
18 ylabel('<(y)')
19 title('Phase Response')

```

Scilab code Exa 5.9b DTFT and steady state response

```

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

```

```

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

```

Scilab code Exa 5.10a System Representation in various forms

```

1 //System Representation in various forms
2 a=0.8;b=2;
3 n=0:50;
4 h=b*(a^n);
5 //Discrete-Time Fourier transform
6 K=500;
7 k=0:1:K;
8 w=%pi*k/K;
9 H=h*exp(-%i*n'*w);
10 //caluculation of phase and magnitude of h(z)
11 [phase_H,m]=phasemag(H);
12 H=abs(H);
13 subplot(2,1,1);
14 plot2d(w/%pi,H);
15 xlabel('Frequency in radians')
16 ylabel('abs(H)')
17 title('magnitude Response')
18 subplot(2,1,2)
19 plot2d(w/%pi,phase_H)
20 xlabel('Frequency in Radians');
21 ylabel('<(H)')
22 title('Phase Response')

```

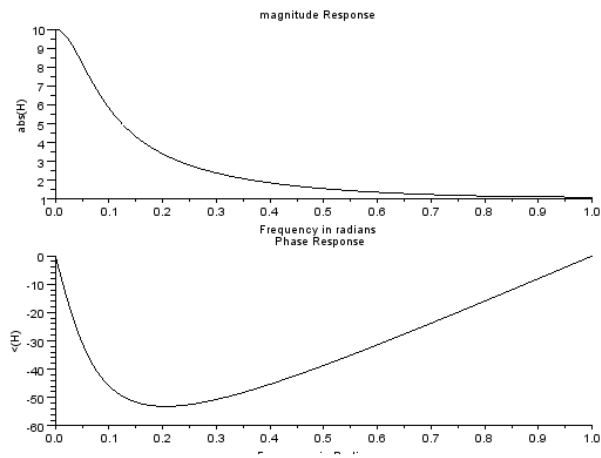


Figure 5.16: System Representation in various forms

Scilab code Exa 5.10b System Representation in various forms

```

1 //System Representation in various forms
2 a=0.6;b=1;
3 n=0:50;
4 h=b*(a^n);
5 //Discrete-Time Fourier transform
6 K=500;
7 k=0:1:K;
8 w=%pi*k/K;
9 H=h*exp(-%i*n'*w);
10 //caluculation of phase and magnitude of h(z)
11 [phase_H,m]=phasemag(H);
12 H=abs(H);
13 subplot(2,1,1);
14 plot2d(w/%pi,H);
15 xlabel('Frequency in radians')
16 ylabel('abs(H)')
17 title('magnitude Response')
18 subplot(2,1,2)

```

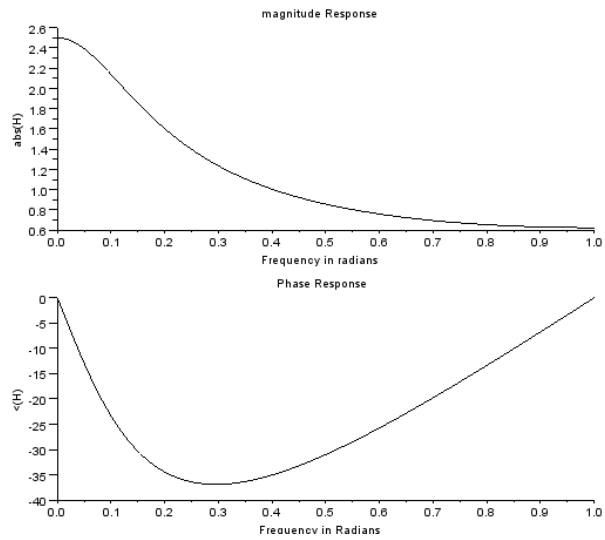


Figure 5.17: System Representation in various forms

```

19 plot2d(w/%pi,phase_H)
20 xlabel('Frequency in Radians');
21 ylabel('∠(H)')
22 title('Phase Response')

```

Chapter 6

Filter Concepts

Scilab code Exa 6.1 The Minimum Phase Concept

```
1 //The minimum phase concept
2 z=%z;
3 F=0:(0.5/400):0.5;
4 z=exp(%i*2*%pi*F);
5 for i=1:401
6 H1Z(i)=((z(i)-1/2)*(z(i)-1/4))/((z(i)-1/3)*(z(i)
   -1/5)));
7 end
8 for i=1:401
9 H2Z(i)=(((z(i)-1/2)*z(i)+1)*(z(i)-1/4))/((z(i)-1/3)*(z(i)
   )-1/5)));
10 end
11 for i=1:401
12 H3Z(i)=(((z(i)-1/2)*z(i)+1)*((z(i)-1/4)*z(i)+1))/((z(i)-1/3)
   *(z(i)-1/5)));
13 end
14 [phase_H1Z ,m]=phasemag(H1Z);
15 [phase_H2Z ,m]=phasemag(H2Z);
16 [phase_H3Z ,m]=phasemag(H3Z);
```

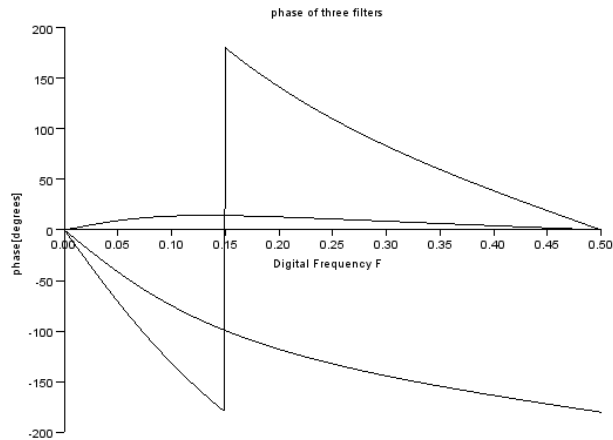



Figure 6.1: The Minimum Phase Concept

```

17 a=gca();
18 a.x_location="origin";
19 xlabel('Digital Frequency F');
20 ylabel('phase[degrees]');
21 xtitle('phase of three filters');
22 plot2d(F,phase_H1Z,rect=[0,-200,0.5,200]);
23 plot2d(F,phase_H2Z,rect=[0,-200,0.5,200]);
24 plot2d(F,phase_H3Z,rect=[0,-200,0.5,200]);

```

Scilab code Exa 6.4 Linear Phase Filters

```

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

```

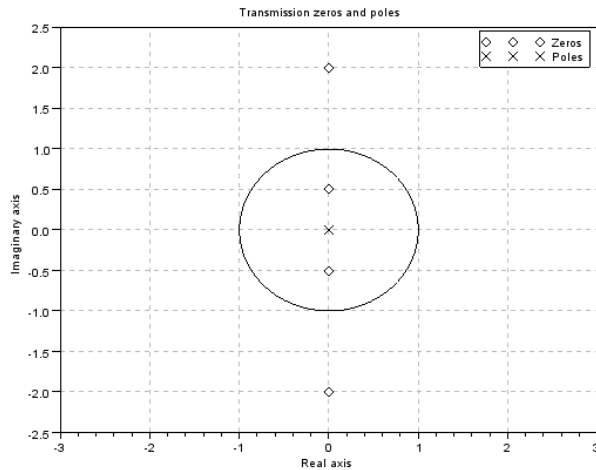


Figure 6.2: Linear Phase Filters

```

7 plzr(H2Z);
8 //from pole zero diagram and LPF
9 // characteristics its a linear phase filter
10 H3Z=((z^4+2.5*z^3-2.5*z-1)/(z^4));
11 xset('window',2);
12 plzr(H3Z);
13 //from pole zero diagram and LPF
14 // characteristics its a linear phase filter

```

Scilab code Exa 6.6 Frequency Response and Filter characteristics

```

1 //Frequency Response and filter characteristics

```

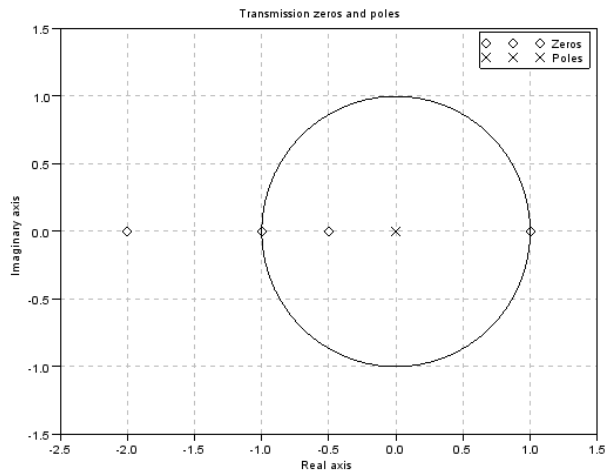


Figure 6.3: Linear Phase Filters

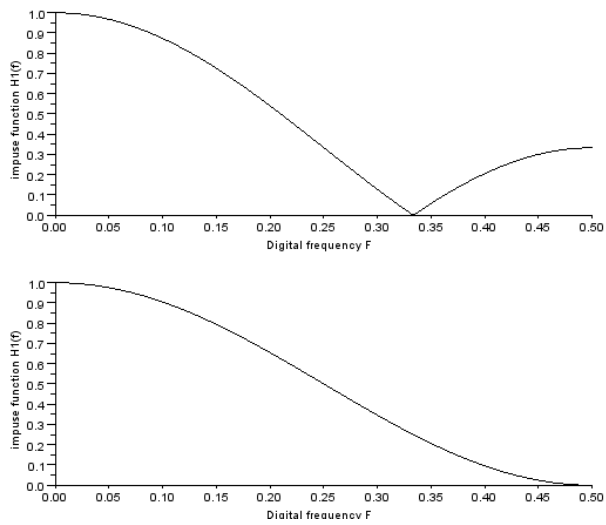


Figure 6.4: Frequency Response and Filter characteristics

```

2 z=%z;
3 F=0:(0.5/200):0.5;
4 z=exp(%i*2*%pi*F);
5 H1=(1/3)*(z+1+z^-1);
6 H2=(z/4)+(1/2)+(1/4)*(z^-1);
7 H1=abs(H1);
8 H2=abs(H2);
9 a=gca();
10 a.x_location="origin";
11 subplot(211);
12 plot2d(F,H1);
13 xlabel('Digital frequency F');
14 ylabel('impuse function H1(f)');
15 subplot(212);
16 plot2d(F,H2);
17 xlabel('Digital frequency F');
18 ylabel('impuse function H1(f)');

```

Scilab code Exa 6.7a Filters and Pole Zero Plots

```

1 z=%z;
2 s=%s;
3 F=0:(0.5/400):0.5;
4 s=exp(%i*2*%pi*F);
5 H1Z=(z^4+1)/(z^4+1.6982*z^2+0.7210);
6 for i=1:401
7     H1(i)=(s(i)^4+1)/(s(i)^4+1.6982*s(i)^2+0.7210);
8 end
9 H1=abs(H1);
10 plzr(H1Z);
11 a=gca();
12 xset('window',1);
13 a.x_location="origin";
14 a.y_location="origin";
15 plot2d(F,H1)

```

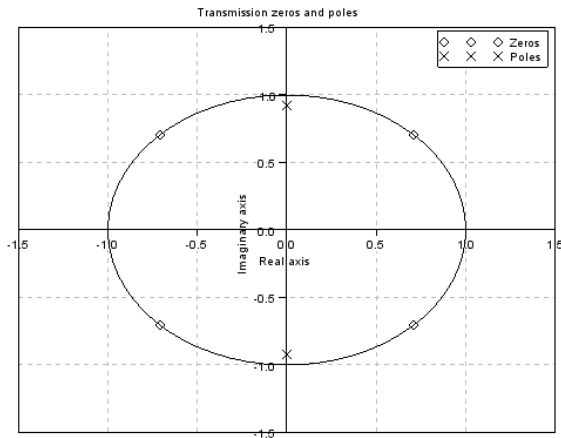


Figure 6.5: Filters and Pole Zero Plots

```

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

```

Scilab code Exa 6.7b Filters and Pole Zero Plots

```

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

```

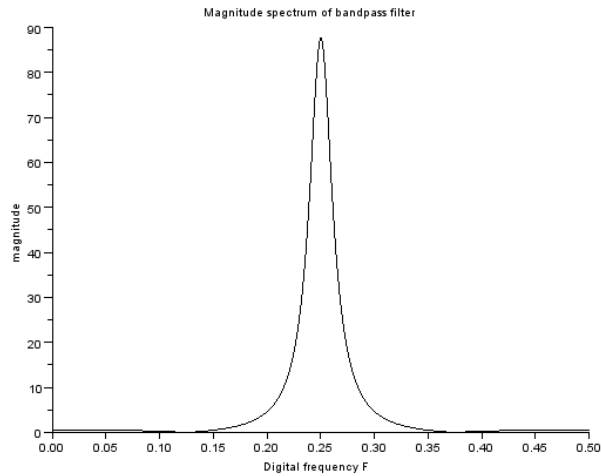


Figure 6.6: Filters and Pole Zero Plots

```

8 end
9 H1=abs(H1);
10 plzr(H1Z);
11 a=gca();
12 xset('window',1);
13 a.x_location="origin";
14 a.y_location="origin";
15 plot2d(F,H1)
16 xlabel('Digital frequency F');
17 ylabel('magnitude');
18 xtitle('Magnitude spectrum of bandpass filter');

```

Scilab code Exa 6.8 Digital resonator Design

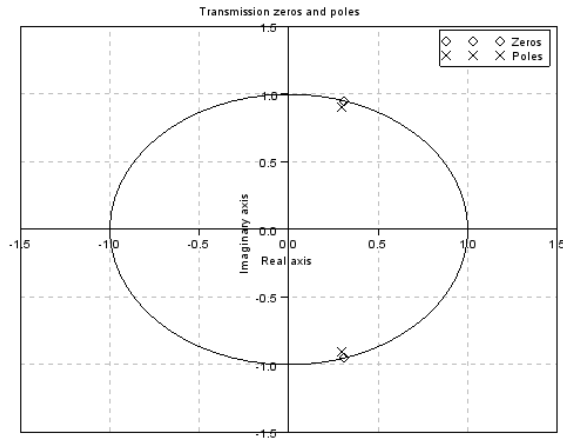


Figure 6.7: Filters and Pole Zero Plots

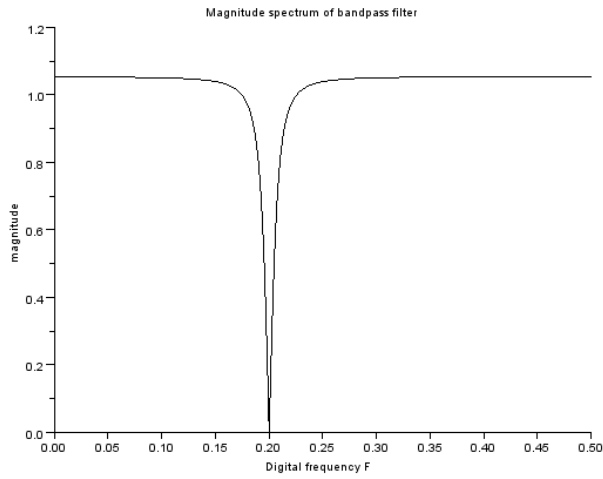


Figure 6.8: Filters and Pole Zero Plots

```

1 //Digital Resonator design with peak gain 50 HZ
2 //and 3 db bandwidth of 6HZ at sampling of 300 HZ
3 clf();
4 s=%s;
5 F=0:150;
6 f=F/300;
7 s=exp(%i*2*%pi*f);
8 for i=1:151
9     H1(i)=(0.1054*(s(i)^2))/(s(i)^2-0.9372*s(i)
        +0.8783);
10 end
11 H1=abs(H1);
12 H2=H1(40:60);
13 F1=40:60;
14 f1=F1/300;
15 a=gca();
16 a.x_location="origin";
17 a.y_location="origin";
18 plot2d(F,H1)
19 xlabel('Analog frequency F');
20 ylabel('magnitude');
21 xtitle('Magnitude spectrum of digital resonator with
        peak 50HZ');
22 xset('window',1);
23 a.x_location="origin";
24 a.y_location="origin";
25 plot2d(F1,H2)
26 xlabel('Analog frequency F');
27 ylabel('magnitude');
28 xtitle('passband detail');

```

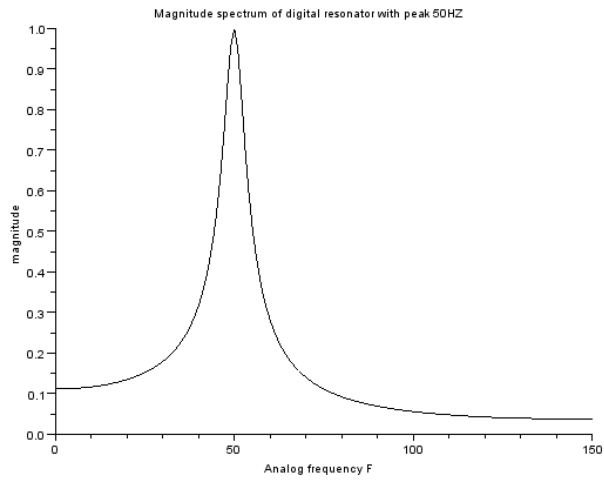


Figure 6.9: Digital resonator Design

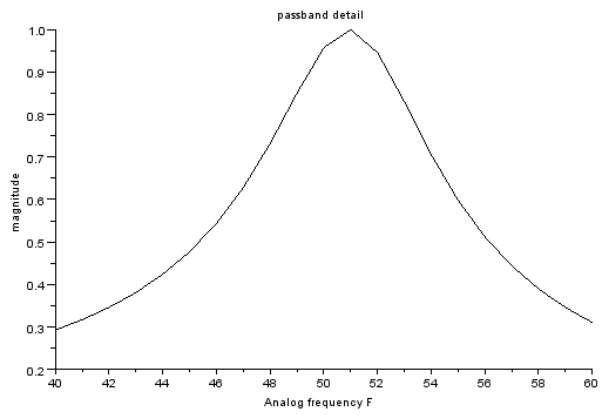


Figure 6.10: Digital resonator Design

Scilab code Exa 6.9 Periodic Notch Filter Design

```
1 //Periodic notch filter design at 60 HZ and sampling
   frequency 300HZ
2 z=%z;
3 f=0:(0.5/400):0.5;
4 z1=exp(%i*2*%pi*f);
5 for i=1:401
6     H1Z(i)=(z1(i)^5-1)/((z1(i)^5)-(0.9^5));
7     H2Z(i)=(z1(i)^5-1)/((z1(i)^5)-(0.99^5));
8 end
9 H1Z=abs(H1Z);
10 H2Z=abs(H2Z);
11 N1z=(1-z^-5)/(1-z^-1);
12 H3z=(N1z)/(horner(N1z,z/0.9));
13 H4z=(N1z)/(horner(N1z,z/0.99));
14 H3z=horner(H3z,z1);
15 H4z=horner(H4z,z1);
16 a=gca();
17 a.x_location="origin";
18 a.y_location="origin";
19 plot2d(f,H1Z);
20 plot2d(f,H2Z);
21 xlabel('Digital frequency f');
22 ylabel('magnitude');
23 xtitle('Periodic Notch Filter N=5,R=0.9,0.99');
24 xset('window',1);
25 plot2d(f,H3z);
26 plot2d(f,H4z);
27 xlabel('Digital frequency f');
28 ylabel('magnitude');
29 xtitle('Notch Filter that also passes DC N=5,R
   =0.9,0.99');
```

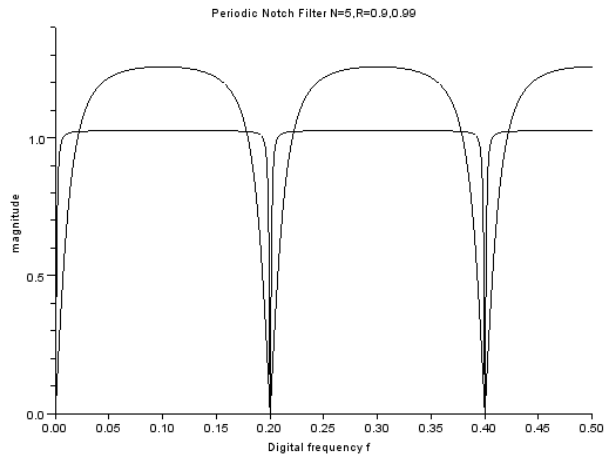


Figure 6.11: Periodic Notch Filter Design

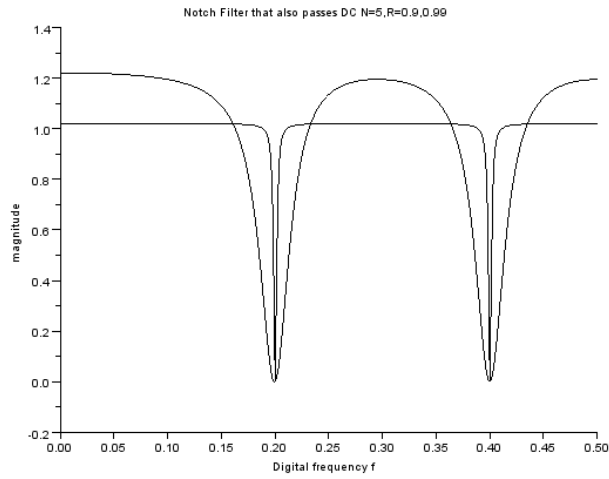


Figure 6.12: Periodic Notch Filter Design

Chapter 7

Digital Processing of Analog Signals

Scilab code Exa 7.3 Sampling oscilloscope

```
1 //Sampling Oscilloscope Concepts
2 fo=100;a=50;
3 s=(a-1)*fo/a;
4 B=100-s;
5 i=s/(2*B);
6 i=ceil(i);
7 disp(i,'The sampling frequency can at max divided by
      i');
8 disp(s,2*B,'range of sampling rate is between s and
      2*B');
9 fo1=100;
10 a=50;
11 s1=(a-1)*fo1/a;
12 B1=400-4*s1;
13 j=s1/(2*B1);
14 j=ceil(j);
15 disp(j,'The sampling frequency can at max divided by
      j');
16 disp(s1,2*B1,'range of sampling rate is between s1
```

and $2*B1'$);

Scilab code Exa 7.4 Sampling of Band pass signals

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

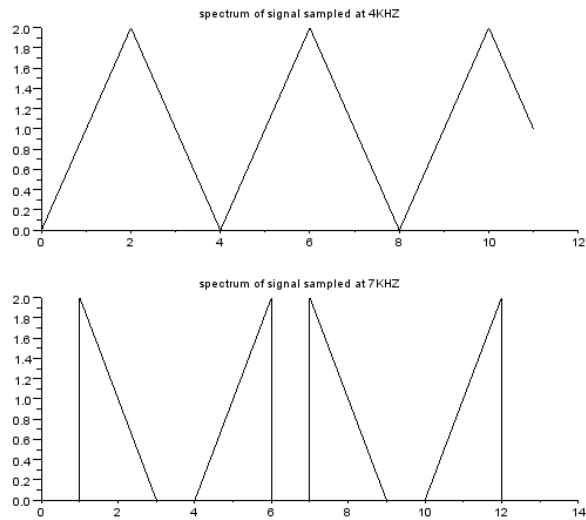


Figure 7.1: Sampling of Band pass signals

Scilab code Exa 7.6 Signal Reconstruction from Samples

```

1 //signal reconstruction from samples
2 //(a)By step interpolation method
3 x=[-1 2 3 2];
4 t=2.5;
5 ts=1;
6 t1=ceil(t);
7 t2=floor(t);
8 x1t=x(t2)
9 //(b)By linear interpolation method
10 x2t=(x(t1)+x(t2))/2
11 //(c)By sinc interpolation method

```

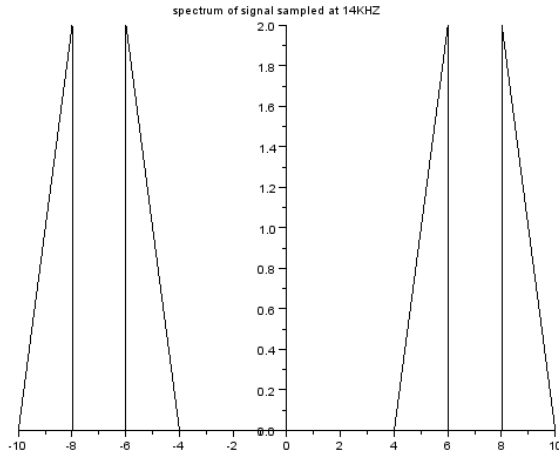


Figure 7.2: Sampling of Band pass signals

```

12 x3t=0;x1=[1 2 3 4];
13 for k=1:4
14     x3t=x3t+(x1(k)*sinc(%pi*(t-(k-1)))));
15 end
16 x3t//sinc interpolated value of x(2.5)
17 //(d)raised cosine interpolation method
18 x4t=0;
19 for k=1:4
20     p=(cos(0.5*%pi*(t-k+1))/(1-(t-k+1)^2));
21     xt=x1(k)*sinc(%pi*(t-k+1))*p;
22     x4t=x4t+xt;
23 end
24 x4t//raised cosine interpolated value of x(2.5)

```

Scilab code Exa 7.7 Zero Interpolation and Spectrum Replication

```

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

```

```

3 X1F=[XF XF XF 0];
4 YF=[X1F X1F];
5 DF=0.5*[XF XF 0];
6 GF=0.5*[XF 0 XF 0 XF 0];
7 f=-0.2:0.1:1;
8 f1=-0.1:0.05:1.15;
9 f2=-0.4:0.2:1.2;
10 f3=-0.2:0.1:1.2;
11 length(f3),length(GF)
12 a=gca();
13 a.y_location="origin";
14 subplot(211);
15 plot2d(f,X1F);
16 ylabel('X1F');
17 subplot(212);
18 a.y_location="origin";
19 plot2d(f1,YF);
20 ylabel('YF');
21 xset('window',1);
22 b=gca();
23 b.y_location="origin";
24 subplot(211);
25 plot2d(f2,DF);
26 ylabel('DF');
27 subplot(212);
28 b.y_location="origin";
29 plot2d(f3,GF);
30 ylabel('GF');

```

Scilab code Exa 7.8 Up Sampling and Filtering

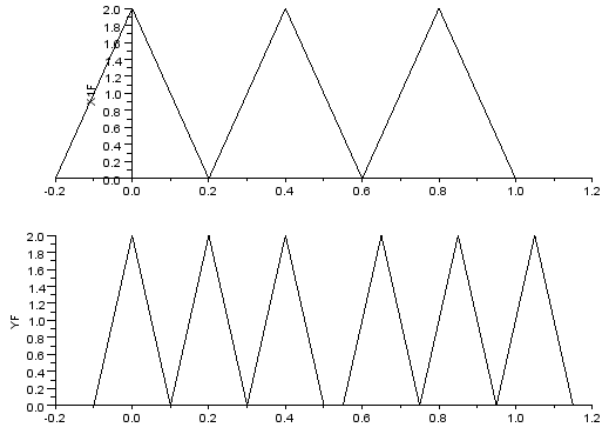


Figure 7.3: Zero Interpolation and Spectrum Replication

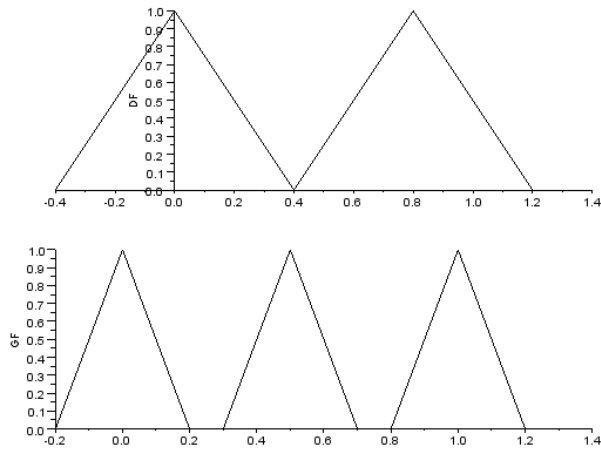


Figure 7.4: Zero Interpolation and Spectrum Replication

```

1  clf();
2  X=[0 0.5 1 0.5];
3  XF=[X 0];
4  WF=[X X X 0];
5  f=-0.5:0.25:0.5;
6  f1=-0.75:0.125:0.75;
7  HF=[0 1 1 1 0];
8  f2=[-0.126,-0.125:0.125:0.125,0.126];
9  for i=1:5
10     YF(i)=WF(i)*HF(i);
11 end
12 f3=[-0.126 -0.125 0 0.125 0.126];
13 a=gca();
14 a.y_location="origin";
15 subplot(211);
16 plot2d(f,XF);
17 xtitle('spectrum of XF');
18 a.y_location="origin";
19 subplot(212);
20 plot2d(f1,WF);
21 xtitle('spectrum of WF');
22 xset('window',1);
23 b=gca();
24 b.y_location="origin";
25 subplot(211);
26 plot2d(f2,HF);
27 xtitle('spectrum of HF');
28 b.y_location="origin";
29 subplot(212);
30 plot2d(f3,YF);
31 xtitle('spectrum of YF');

```

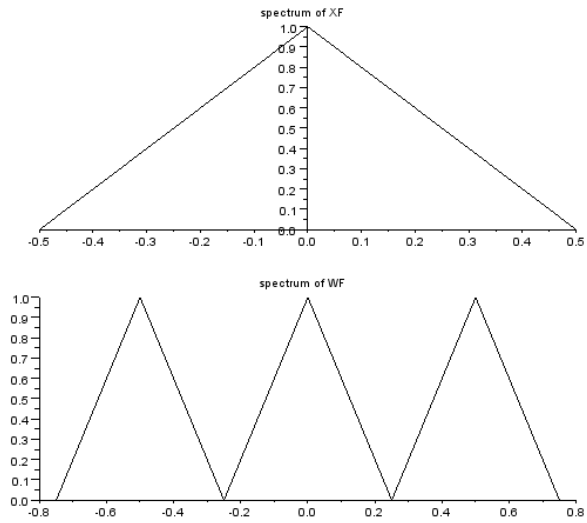


Figure 7.5: Up Sampling and Filtering

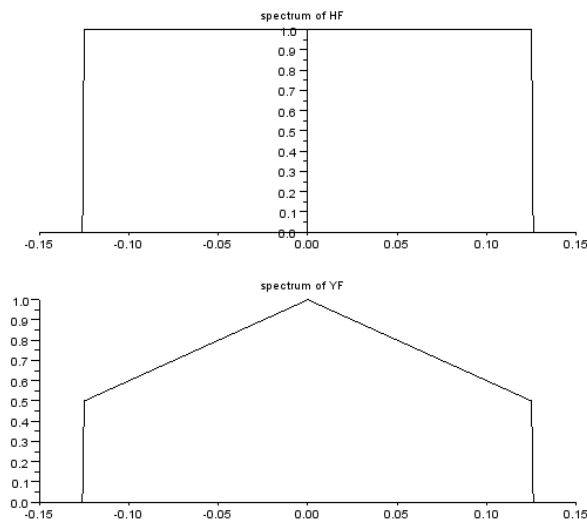


Figure 7.6: Up Sampling and Filtering

Scilab code Exa 7.9 Quantisation Effects

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

Scilab code Exa 7.10 ADC considerations

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

```

7 //Hence TA must satisfy  $TA \leq T_{Am}$  nano sec
8 //(b) conversion time of quantizer
9  $TA = 4 \times 10^{-9}$ ;
10  $TH = 10 \times 10^{-6}$ ; //hold time
11  $S = 30 \times 10^3$ ;
12  $TCm = 1/S - TA - TH$ ;
13  $TCm = TCm \times 10^6$ 
14 //Hence TC must satisfy  $TC \leq TCm$  micro sec
15 //(c) Holding capacitance C
16  $V_o = 10$ ;
17  $TH = 10 \times 10^{-6}$ ;
18  $B = 12$ ;
19  $R = 10^6$ ; //input resistance
20  $\text{delv} = V_o / (2^{(B+1)})$ ;
21  $Cm = (V_o * TH) / (R * \text{delv})$ ;
22  $Cm = Cm \times 10^9$ 
23 //Hence C must satisfy  $C \geq Cm$  nano farad

```

Scilab code Exa 7.11 Anti Aliasing Filter Considerations

```

1 //Anti Aliasing filter considerations
2 //minimum stop band attenuation As
3  $B = \text{input}('enter no. of bits')$ ; //no. of samples
4  $n = \text{input}('enter band width in KHZ')$ ;
5  $As = 20 * \log_{10}(2^{B * \text{sqrt}(6)})$ 
6 //nomalised frequency
7  $Vs = (10^{(0.1 * As)} - 1)^{(1/(2 * n))}$ 
8  $fp = 4$ ; //pass edge frequency
9  $fs = Vs * fp$  //stop band frquency
10  $S = 2 * fs$  //sampling frequency
11  $fa = S - fp$  //aliaed frequency
12  $Va = fa / fp$ ;
13 //Attenuation at aliased frequency
14  $Aa = 10 * \log_{10}(1 + Va^{(2 * n)})$ 

```

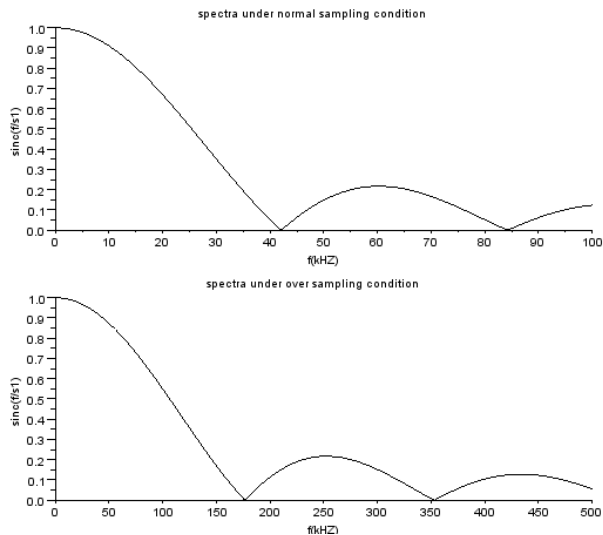


Figure 7.7: Anti Imaging Filter Considerations

Scilab code Exa 7.12 Anti Imaging Filter Considerations

```

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

```

```

14 Ap=0.316;
15 e2=sqrt(10^(0.1*Ap)-1);
16 n1=(log10(e1/e2))/(log(fs1/fp)); //new order of
    butworth filter
17 n1=ceil(n1)
18 f=0:100;
19 x=abs(sinc(f*%pi/S));
20 f1=0:500;
21 x1=abs(sinc(f1*%pi/S1));
22 a=gca();
23 subplot(211);
24 plot2d(f,x);
25 xtitle("spectra under normal sampling condition", "f(
    kHz)", "sinc(f/s1)");
26 subplot(212);
27 plot2d(f1,x1);
28 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

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

Scilab code Exa 8.2 The DFT and conjugate Symmetry

```
1 //The DTFT and conjugate symmetry
2 //8-point DFT
3 x=[1 1 0 0 0 0 0 0];
4 XDFT=fft(x,-1);
5 disp(XDFT,'The DFT of x is ');
6 disp('from conjugate symmetry we see XDFT[k]=XDFT[8-
    k]');
```

Scilab code Exa 8.3 Circular Shift and Flipping

```
1 //Circular shift and flipping
2 //(a)right circular shift
3 y=[1 2 3 4 5 0 0 6];
4 f=y;g=y;h=y;
5 for i=1:2
6     b=f(length(f));
7     for j=length(f):-1:2
8         f(j)=f(j-1);
9     end
10    f(1)=b;
11 end
12 disp(f,'By right circular shift y[n-2] is ');
13 //(b)left circular shift
14 for i=1:2
15     a=g(1);
16     for j=1:length(g)-1
17         g(j)=g(j+1);
18     end
19     g(length(g))=a;
20 end
21 disp(g,'By left circular shift y[n+2] is ');
22 //(c)flipping property
23 h=[h(1) h(length(h):-1:2)];
24 disp(h,'By flipping property y[-n] is ');
```

Scilab code Exa 8.4 Properties of DFT

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

```

4 y=x;
5 for i=1:2
6     a=y(1);
7     for j=1:length(y)-1
8         y(j)=y(j+1);
9     end
10    y(length(y))=a;
11 end
12 YDFT=fft(y,-1)
13 disp(YDFT,'By Time-Shift property DFT of x[n-2] is ')
    ;
14 //(b)flipping property
15 g=[x(1) x(length(x):-1:2)]
16 GDFT=fft(g,-1)
17 disp(GDFT,'By Time reversal property DFT of x[-n] is
    ');
18 //(c)conjugation property
19 p=XDFT;
20 PDFT=[p(1) p(4:-1:2)];
21 disp(YDFT,'BY conjugation property DFT of x*[n] is ')
    ;

```

Scilab code Exa 8.5a Properties of DFT

```

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

```

```

12     for j=1:3
13         HDFT3(i)=HDFT3(i)+HDFT2(j,i);
14     end
15 end
16 disp(HDFT3,'DFT of x[n]^2 is ');
17 //a2) periodic convolution
18 vn=convol(xn,xn);
19 vn=[vn,zeros(8:12)];
20 vn=[vn(1:4);vn(5:8);vn(9:12)];
21 vn1=[0 0 0 0];
22 for i=1:4
23     for j=1:3
24         vn1(i)=vn1(i)+vn(j,i);
25     end
26 end
27 VDFT=fft(vn1,-1);
28 VDFT1=XDFT.*XDFT;
29 disp(VDFT1,'DFT of x[n]*x[n] is ');
30 //a3) signal energy(parcewell's theorem)
31 xn2=xn.^2;
32 E=0;
33 for i=1:length(xn2)
34     E=E+abs(xn2(i));
35 end
36 XDFT2=XDFT.^2
37 E1=0;
38 for i=1:length(XDFT2)
39     E1=E1+abs(XDFT2(i));
40 end
41 E,(1/4)*E1;
42 disp(1/4*E1,'The energy of the signal is ');

```

Scilab code Exa 8.5b Properties of DFT

```

1 //b1) modulation

```

```

2 XDFT=[4 -2*i 0 2*i];
3 xn=dft(XDFT,1)
4 for i=1:length(xn)
5     zn(i)=xn(i)*%e^((%i*%pi*(i-1))/2);
6 end
7 disp(zn, 'The IDFT of XDFT[k-1] is ');
8 ZDFT=[2*i 4 -2*i 0];
9 zn1=dft(ZDFT,1)
10 //b2)periodic convolution
11 HDFT=(convol(XDFT,XDFT))
12 HDFT=[HDFT,zeros(8:12)];
13 HDFT=[HDFT(1:4);HDFT(5:8);HDFT(9:12)];
14 HDFT1=[0 0 0 0];
15 for i=1:4
16     for j=1:3
17         HDFT1(i)=HDFT1(i)+HDFT(j,i);
18     end
19 end
20 HDFT1;
21 hn=dft(HDFT1,1)
22 hn1=4*(xn.*xn);
23 disp(hn1, 'The IDFT of XDFT*XDFT is ');
24 //b3)product
25 WDFT=XDFT.*XDFT;
26 wn=dft(WDFT,1)
27 wn1=convol(xn,xn);
28 wn1=[wn1,zeros(8:12)];
29 wn1=[wn1(1:4);wn1(5:8);wn1(9:12)];
30 WN=[0 0 0 0];
31 for i=1:4
32     for j=1:3
33         WN(i)=WN(i)+wn1(j,i);
34     end
35 end
36 disp(WN, 'The IDFT of XDFT.XDFT is ');
37 //b4)Central ordinates and signal Energy
38 E=0;
39 for i=1:length(xn)

```

```

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

```

Scilab code Exa 8.5c Properties of DFT

```

1 //Regular convolution
2 xn=[1 2 1 0];
3 yn=[1 2 1 0 0 0];
4 YDFT=fft(yn,-1)
5 SDFT=YDFT.*YDFT
6 sn=fft(SDFT,1)
7 sn1=convol(xn,xn)

```

Scilab code Exa 8.6 Signal and Spectrum Replication

```

1 //Signal and spectrum replication
2 xn=[2 3 2 1];
3 XDFT=fft(xn,-1)
4 yn=[xn xn xn];
5 YDFT=fft(yn,-1)
6 YDFT1=3*[XDFT(1:1/3:length(XDFT))];
7 for i=2:3
8     YDFT1(i:3:length(YDFT1))=0;
9 end
10 YDFT1(12:-1:11)=0;
11 disp(YDFT1, 'the DFT of x[n/3] is ');
12 hn=[xn(1:1/3:length(xn))]
13 for i=2:3
14     hn(i:3:length(hn))=0;
15 end
16 hn(12:-1:11)=0;
17 hn

```

```

18 HDFT=fft(hn,-1)
19 HDFT1=[XDFT;XDFT;XDFT];
20 disp(HDFT1,'the DFT of y[n]=[x[n],x[n],x[n]] is ');

```

Scilab code Exa 8.7 Relating DFT and DTFT

```

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

```

Scilab code Exa 8.8 Relating DFT and DTFT

```

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

```

Scilab code Exa 8.9a The DFT and DFS of sinusoids

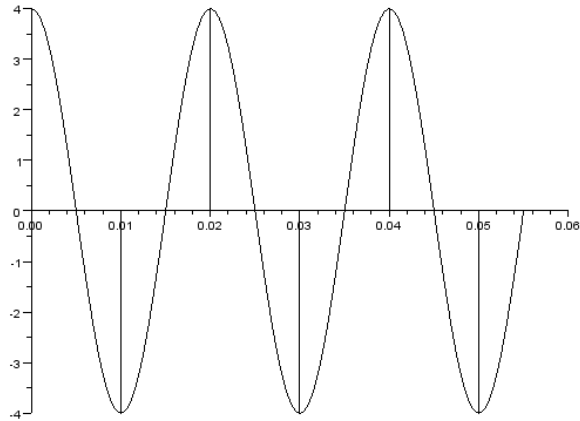


Figure 8.1: The DFT and DFS of sinusoids

```

1 //DFT and DFS of sinusoids
2 n2=0:0.5/1000:5.5/100;
3 xt=4*cos(100*pi*n2');
4 n=0:(0.5)/100:(5.5)/100; //F=3/12 hence N=12
5 xn=4*cos(100*pi*n');
6 XDFT=fft(xn,-1);
7 n1=0:11;
8 a=gca();
9 a.x_location="origin";
10 plot2d(n2,xt);
11 plot2d3('gnn',n,xn);
12 xset('window',1);
13 b=gca();
14 b.x_location="origin";
15 plot2d3('gnn',n1,XDFT);

```

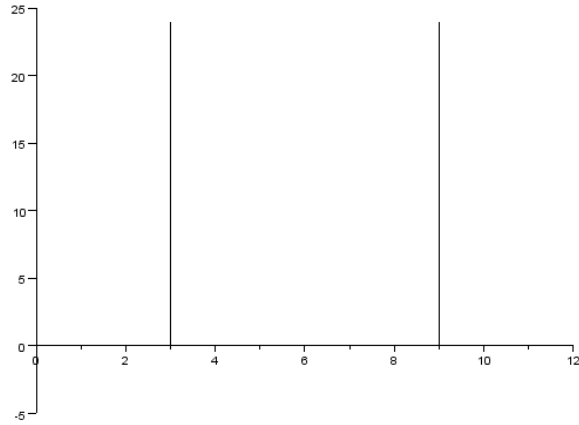


Figure 8.2: The DFT and DFS of sinusoids

Scilab code Exa 8.9b The DFT and DFS of sinusoids

```

1 //DFT and DFS of sinusoids
2 n2=0:1/1280:31/128;
3 xt=4*sin(72*%pi*n2');
4 n=0:1/128:31/128;//F=9/32 hence N=32
5 xn=4*sin(72*%pi*n');
6 XDFT=abs(fft(xn,-1));
7 n1=0:31;
8 a=gca();
9 a.x_location="origin";
10 plot2d(n2,xt);
11 plot2d3('gnn',n,xn);
12 xset('window',1);
13 b=gca();
14 b.x_location="origin";
15 plot2d3('gnn',n1,XDFT);

```

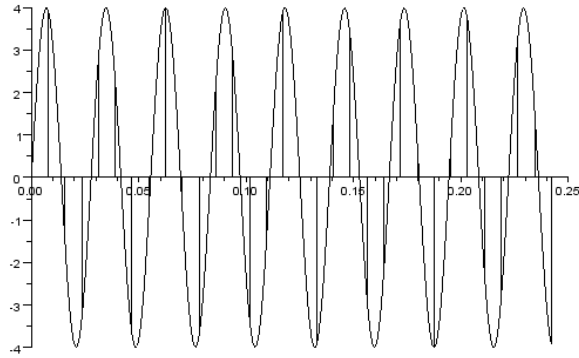



Figure 8.3: The DFT and DFS of sinusoids

Scilab code Exa 8.9c The DFT and DFS of sinusoids

```

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

```

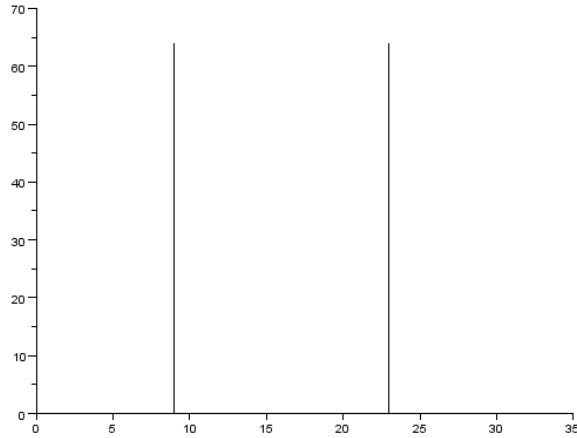


Figure 8.4: The DFT and DFS of sinusoids

```

12 xset('window',1);
13 b=gca();
14 b.x_location="origin";
15 plot2d3('gnn',n1,XDFT);

```

Scilab code Exa 8.9d The DFT and DFS of sinusoids

```

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

```

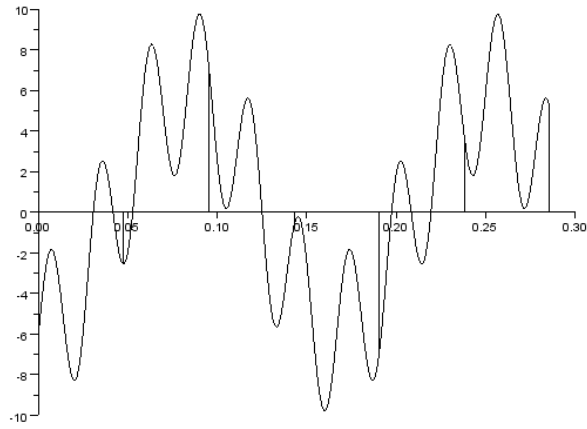


Figure 8.5: The DFT and DFS of sinusoids

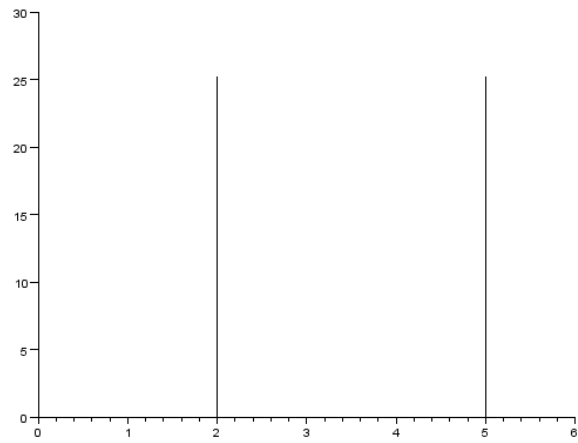


Figure 8.6: The DFT and DFS of sinusoids

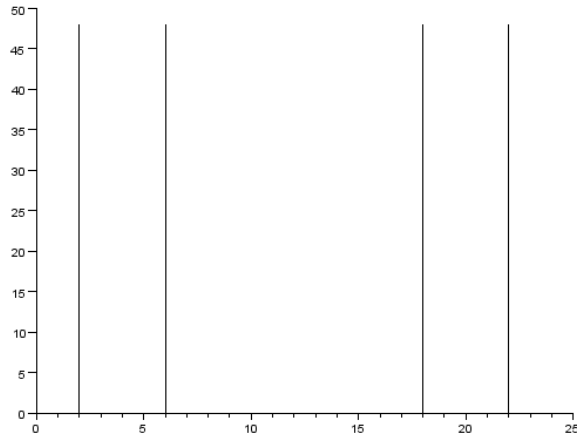


Figure 8.7: The DFT and DFS of sinusoids

```

8 a=gca();
9 a.x_location="origin";
10 plot2d(n2,xt);
11 plot2d3('gmn',n,xn);
12 xset('window',1);
13 b=gca();
14 b.x_location="origin";
15 plot2d3('gmn',n1,XDFT);

```

Scilab code Exa 8.10 DFS of sampled Periodic Signals

```

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

```

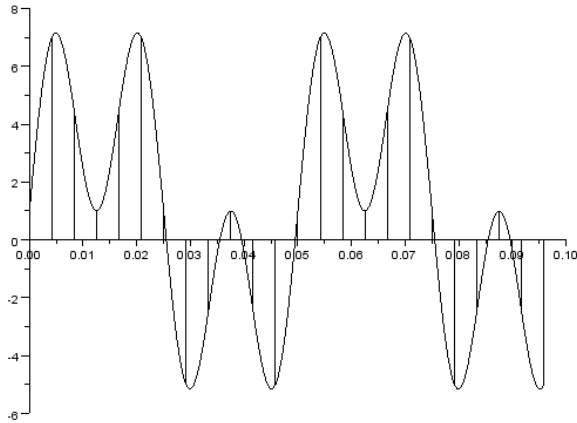


Figure 8.8: The DFT and DFS of sinusoids

```

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

```

Scilab code Exa 8.11 The effects of leakage

```

1 // Effects of leakage
2 n1=0:0.005:0.1;
3 n2=0:0.005:0.125;
4 n3=0:0.005:1.125;
5 xt1=(2*cos(20*pi*n1')+5*cos(100*pi*n1'));
6 xt2=(2*cos(20*pi*n2')+5*cos(100*pi*n2'));
7 xt3=(2*cos(20*pi*n3')+5*cos(100*pi*n3'));
8 XDFS1=abs(fft(xt1,-1))/20;
9 XDFS2=abs(fft(xt2,-1))/25;

```

```

10 XDFS3=abs(fft(xt3,-1))/225;
11 f1=0:5:100;
12 f2=0:4:100;
13 f3=0:100/225:100;
14 a=gca();
15 a.x_location="origin";
16 plot2d3('gnn',f1,XDFS1);
17 xlabel('analog frequency');
18 ylabel('Magnitude');
19 xset('window',1);
20 subplot(211);
21 plot2d3('gnn',f2,XDFS2);
22 xlabel('analog frequency');
23 ylabel('Magnitude');
24 subplot(212);
25 plot2d3('gnn',f3,XDFS3);
26 xlabel('analog frequency');
27 ylabel('Magnitude');

```

Scilab code Exa 8.15a Methods to find convolution

```

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

```

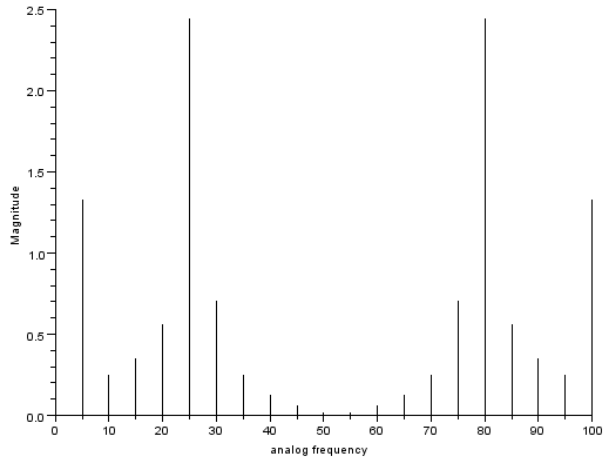


Figure 8.9: The effects of leakage

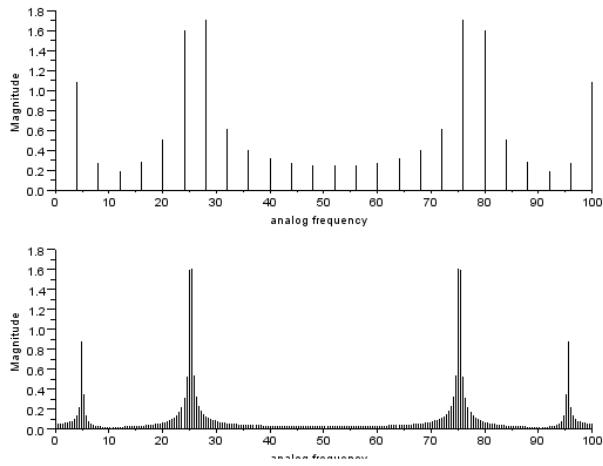


Figure 8.10: The effects of leakage

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

Scilab code Exa 8.15b Methods to find convolution

```
1 //(b)overlap-save method
2 xn=[1 2 3 3 4 5];
3 hn=[1 1 1];
4 xon=[0 0 1 2 3];
5 x1n=[2 3 3 4 5];
6 x2n=[4 5 0 0 0];
7 yon=convol(xon,hn);
8 y1n=convol(x1n,hn);
9 y2n=convol(x2n,hn);
10 yno=yon(3:5);
11 yn1=y1n(3:5);
12 yn2=y2n(3:5);
13 yn=[yno yn1 yn2]
14 YN=convol(xn,hn)
```

Scilab code Exa 8.16 Signal Interpolation using FFT

```
1 //signal interpolation using FFT
2 xn=[0 1 0 -1];
3 XDFT=fft(xn,-1)
4 ZT=[0 -2*i 0 zeros(1:27) 0 2*i];
5 xn1=fft(ZT,1);
6 t=0:1/length(xn1):1-(1/length(xn1));
7 a=gca();
8 a.x_location="origin";
```

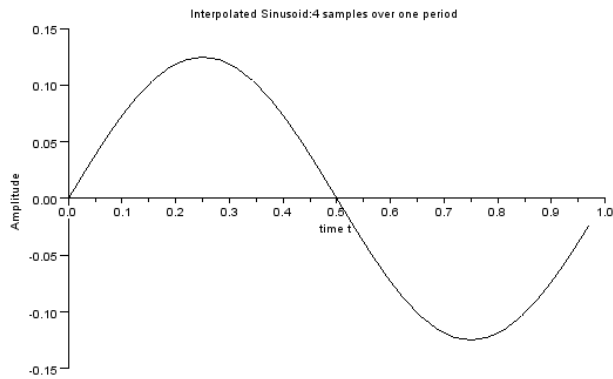



Figure 8.11: Signal Interpolation using FFT

```

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

```

Scilab code Exa 8.17 The Concept of Periodogram

```

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

```

Scilab code Exa 8.18 DFT from matrix formulation

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

Scilab code Exa 8.19 Using DFT to find IDFT

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

Scilab code Exa 8.20 Decimation in Frequency FFT algorithm

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

```

3 w=-%i;
4 xdft(1)=x(1)+x(3)+x(2)+x(4);
5 xdft(2)=x(1)-x(3)+w*(x(2)-x(4));
6 xdft(3)=x(1)+x(3)-x(2)-x(4);
7 xdft(4)=x(1)-x(3)-w*(x(2)-x(4));
8 XDFT=dft(x,-1);
9 xdft,XDFT

```

Scilab code Exa 8.21 Decimation in time FFT algorithm

```

1 //A four point decimation-in-time FFT algorithm
2 x=[1 2 1 0];
3 w=-%i;
4 xdft=[0 0 0 0];
5 for i=1:4
6     for j=1:4
7         xdft(i)=xdft(i)+x(j)*w^((i-1)*(j-1));
8     end
9 end
10 XDFT=dft(x,-1);
11 xdft,XDFT

```

Scilab code Exa 8.22 4 point DFT from 3 point sequence

```

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

```

Scilab code Exa 8.23 3 point IDFT from 4 point DFT

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

Scilab code Exa 8.24 The importance of Periodic Extension

```
1 //The importance of Periodic extension
2 //(a) For M=3
3 x=[1 2 1];
4 XDFT=fft(x,-1)';
5 w=exp(-%i*2*%pi/3);
6 for i=1:3
7     for j=1:3
8         WN(i,j)=w^((i-1)*(j-1));
9     end
10 end
11 WI=WN';
12 xn=1/3*WI*XDFT
13 //The result is periodic with M=3 & 1 period
    equals x[n]
14 //(b) For M=4
15 y=[1 2 1 0];
```

```
16 YDFT=fft(y,-1)'  
17 w=exp(-%i*%pi/2);  
18 for i=1:4  
19     for j=1:4  
20         WN(i,j)=w^((i-1)*(j-1));  
21     end  
22 end  
23 WI=WN';  
24 yn=1/4*WI*YDFT
```

Chapter 9

Design of IIR Filters

Scilab code Exa 9.1 Response Invariant Mappings

```
1 //Response in variant mappings
2 s=%s;z=%z;
3 HS=1/(s+1);
4 f=0:0.05:0.5;
5 HS1=horner(HS, (%i*%pi*2*f'));
6 ts=1;
7 HZ=z/(z-0.3679);
8 HZ1=horner(HZ, exp(%i*%pi*2*f'));
9 a=gca();
10 a.x_location="origin";
11 subplot(211)
12 plot2d(f, HS1);
13 plot2d(f, HZ1);
14 xlabel('Analog frequency f(Hz)');
15 ylabel('Magnitude');
16 xtitle('magnitude of H(s) and H(z)');
17 HZ1=HZ1-0.582; //magnitude after gain matching at dc
18 b=gca();
19 b.x_location="origin";
20 subplot(212);
21 plot2d(f, HS1);
```

```

22 plot2d(f,HZ1);
23 xlabel('Analog frequency f(Hz)');
24 ylabel('Magnitude');
25 xtitle('magnitude after gain matching at DC');
26 //Impulse response of analog and digital filter
27 t=0:0.01:6;
28 ht=exp(-t');
29 n=0:6;
30 hn=exp(-n');
31 xset('window',1)
32 c=gca();
33 subplot(211);
34 plot2d(t,ht);
35 plot2d3('gmn',n,hn);
36 xlabel('DT index n and time t=nts');
37 ylabel('Amplitude');
38 xtitle('Impulse response of analog and digital
        filter');
39 //Step response of analog and digital filter
40 t=0:0.01:6;
41 st=1-exp(-t');
42 n=0:6;
43 sn=(%e-%e^(-n'))/(%e-1);
44 c=gca();
45 subplot(212);
46 plot2d(t,st);
47 plot2d3('gmn',n,sn);
48 xlabel('DT index n and time t=nts');
49 ylabel('Amplitude');
50 xtitle('Step response of analog and digital filter')
    ;

```

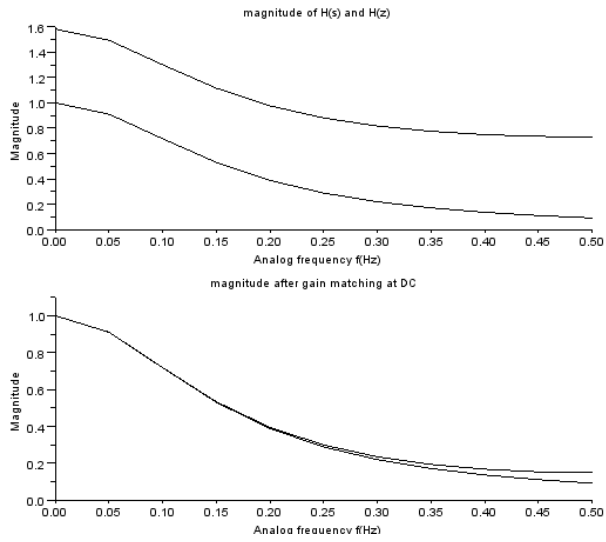


Figure 9.1: Response Invariant Mappings

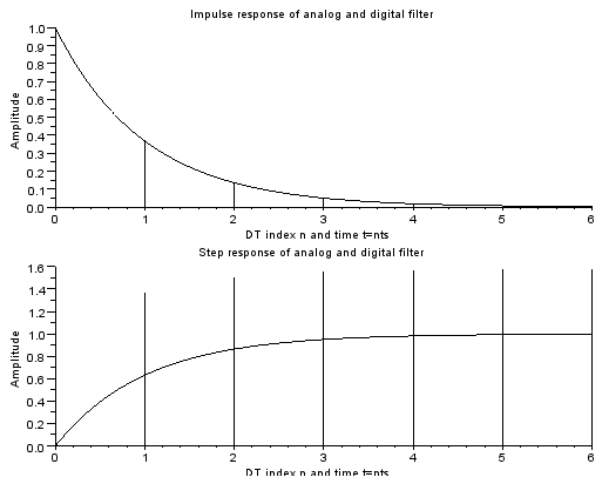


Figure 9.2: Response Invariant Mappings

Scilab code Exa 9.2 Impulse Invariant Mappings

```
1 //Impulse invariant mappings
2 //(a) converting  $H(s)=4s+7/s^2+5s+4$  to  $H(z)$  using
   impulse invariance
3 s=%s;
4 z=%z;
5 HS=(4*s+7)/(s^2+5*s+4);
6 pfss(HS)
7 ts=0.5;
8 HZ=3*z/(z-%e^(-4*ts))+z/(z-%e^(-ts))
9 //(b) converting  $H(s)=4s+7/s^2+5s+4$  to  $H(z)$  using
   impulse invariance
10 HS1=4/((s+1)*(s^2+4*s+5))
11 pfss(HS1);
12 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

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

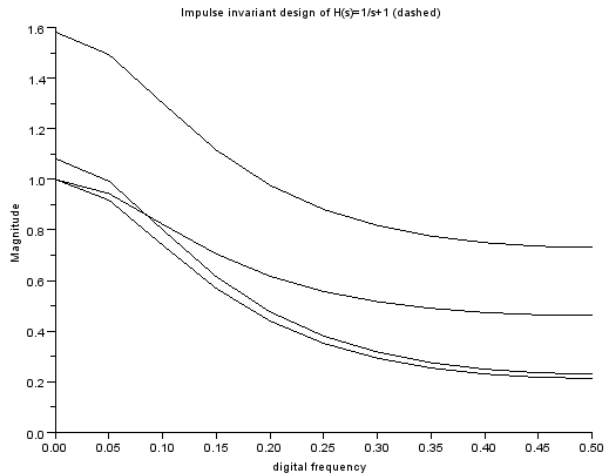


Figure 9.3: Modified Impulse Invariant Design

```

12 HZ1=horner(HZ,exp(%i*2*%pi*f'));
13 HMZ1=horner(HMZ,exp(%i*2*%pi*f'));
14 H1Z1=horner(H1Z,exp(%i*2*%pi*f'));
15 HM1Z1=horner(HM1Z,exp(%i*2*%pi*f'));
16 a=gca();
17 a.x_location="origin";
18 plot2d(f,HZ1);
19 plot2d(f,HMZ1);
20 plot2d(f,H1Z1);
21 plot2d(f,HM1Z1);
22 xlabel('digital frequency');
23 ylabel('Magnitude');
24 xtitle('Impulse invariant design of H(s)=1/s+1 (
    dashed)');

```

Scilab code Exa 9.3cd Modified Impulse Invariant Design

```

1 //modified Impulse invariant Design

```

```

2 // (c) H(s) = 4s + 7/s^2 + 5s + 4
3 s=%s; z=%z;
4 HS=(4*s+7)/(s^2+5*s+4);
5 [d1]=degree(numer(HS));
6 [d2]=degree(denom(HS));
7 HZ=((3*z)/(z-%e^-2))+(z/(z-%e^-0.5))
8 if (d2-d1==1) then
9     h=(4+7/%inf)/(1+5/%inf+4/%inf)
10    HMZ=HZ-0.5*h
11 else
12    HMZ=HZ
13 end
14 HS1=4/((s+1)*(s^2+4*s+5))
15 HZ1=(0.2146*z^2+0.093*z)/(z^3-1.2522*z^2+0.527*z
    -0.0821);
16 [d1]=degree(numer(HS1));
17 [d2]=degree(denom(HS1));
18 if (d2-d1==1) then
19    HMZ1=HZ1-0.5*h
20 else
21    HMZ1=HZ1
22 end

```

Scilab code Exa 9.5 Mappings from Difference Algorithms

```

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

```

```

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

```

Scilab code Exa 9.6 Mappings From Integration Algorithms

```

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

```

Scilab code Exa 9.7 DTFT of Numerical Algorithms

```

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

```

```

11 HFF=(%e^(%i*2*%pi*F'))-1;
12 HDF=1/(%i*2*%pi*F');
13 Hc=1+(%i*2*%pi*F')/2-(2*%pi*F')^2/6;
14 Hc=abs(Hc);
15 //for central difference operator
16 HCF=sin(2*%pi*F')./(%i*4*%pi*%pi*F'^2);
17 Hd=abs(sin(2*%pi*F')./(2*%pi*F'));
18 length(F),length(Ha)
19 a=gca();
20 a.x_location="origin";
21 plot2d(F,Ha,rect=[0,0.8,0.2,1.1]);
22 plot2d(F,Hb);
23 xtitle("Magnitude spectrum of Integration algorithms
",,"Digital Frequency F",,"Magnitude");
24 xset('window',1);
25 plot2d(F,Hc,rect=[0,0.8,0.2,1.1]);
26 plot2d(F,Hd);
27 xtitle("Magnitude spectrum of difference algorithms"
",,"Digital Frequency F",,"Magnitude");

```

Scilab code Exa 9.8a Bilinear Transformation

```

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

```

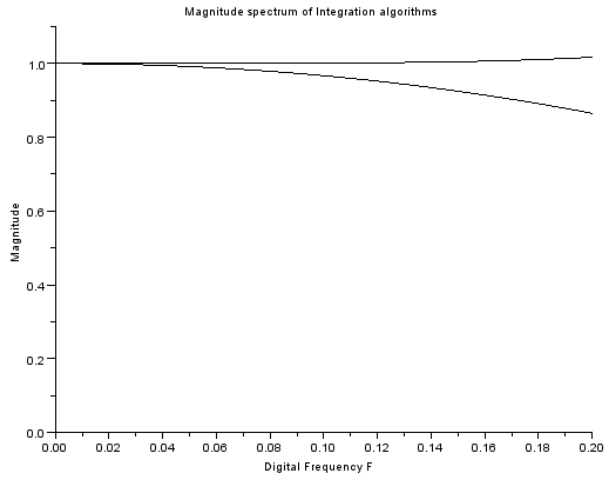


Figure 9.4: DTFT of Numerical Algorithms

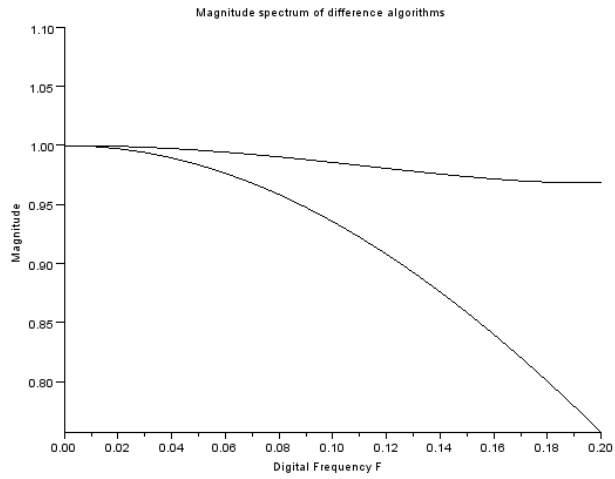


Figure 9.5: DTFT of Numerical Algorithms

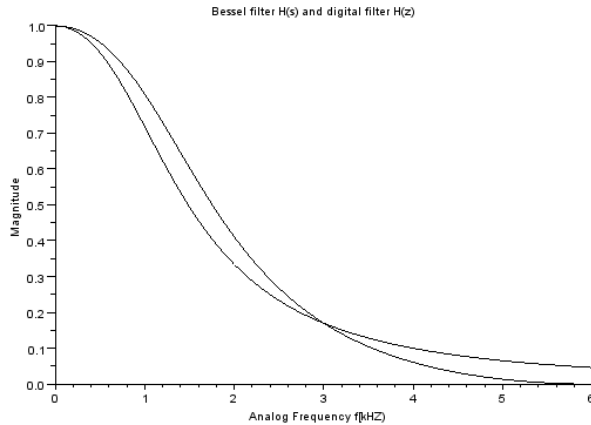


Figure 9.6: Bilinear Transformation

```

8 T=(2/Wa)*(tan(Wd/2));
9 HZ=horner(HS,(2/T)*(z-1)/(z+1))
10 f=0:0.1:6;
11 HS1=horner(HS,(%i*4*f'/3));
12 HS1=abs(HS1);
13 HZ1=horner(HZ,exp(-%i*%pi*f'/6));
14 HZ1=abs(HZ1);
15 a=gca();
16 a.x_location="origin";
17 plot2d(f,HS1);
18 plot2d(f,HZ1);
19 xlabel('Analog Frequency f[kHZ]');
20 ylabel('Magnitude');
21 xtitle('Bessel filter H(s) and digital filter H(z)')
    ;

```

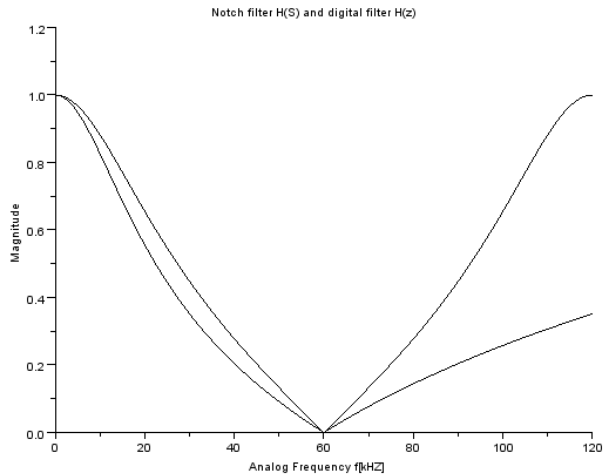


Figure 9.7: Bilinear Transformation

Scilab code Exa 9.8b Bilinear Transformation

```

1 //Bilinear transformation
2 //To convert twin-T notch analog filter to digital
  filter
3 s=%s;
4 z=%z;
5 HS=(s^2+1)/(s^2+4*s+1);
6 Wo=1;
7 S=240;f=60;//sampling and analog frequencies
8 W=0.5*%pi;//digital frequency
9 C=Wo/tan(0.5*W)
10 HZ=horner(HS,C*(z-1)/(z+1))
11 f=0:120;
12 HZ1=abs(horner(HZ,exp(-%i*%pi*f'/120)));
13 HS1=abs(horner(HS,(%i*f'/60)));
14 a=gca();
15 a.x_location="origin";
16 plot2d(f,HZ1);
17 plot2d(f,HS1);

```

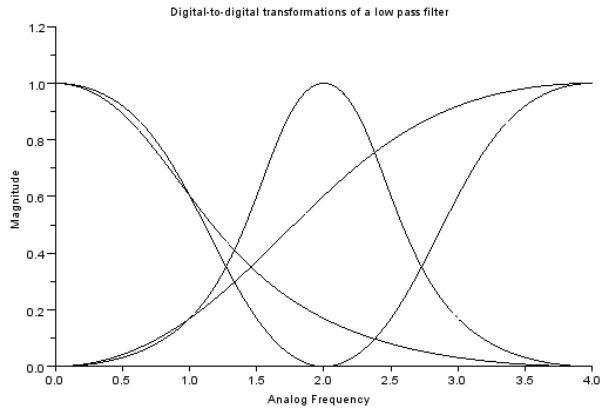



Figure 9.8: D2D transformations

```

18 xlabel('Analog Frequency f [kHz] ');
19 ylabel('Magnitude');
20 xtitle('Notch filter H(S) and digital filter H(z)');

```

Scilab code Exa 9.9 D2D transformations

```

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

```

```

10 W2=0.75*%pi;
11 K=tan(0.5*WD)/tan(0.5*(W2-W1))
12 a=-cos(0.5*(W1+W2))/cos(0.5*(W1-W2))//Mapping
    parameters k,a,A1,A2
13 A1=2*K*a/(K+1),A2=(K-1)/(K+1)
14 HBPZ=horner(HZ,-(z^2+A1*z+A2)/(A2*z^2+A1*z+1))
15 //(c)Lp 2 BS transformation
16 w1=3/8*%pi;w2=5/8*%pi;//band edges
17 K1=tan(0.5*WD)*tan(0.5*(w2-w1))
18 a1=-cos(0.5*(w1+w2))/cos(0.5*(w2-w1))//Mapping
    parameters k1,a1,A1c,A2c
19 A1c=2*a1/(K1+1),A2c=-(K1-1)/(K1+1)
20 HBSZ=horner(HZ,(z^2+A1c*z+A2c)/(A2c*z^2+A1c*z+1))
21 f=0:1/256:4;
22 for i=1:length(f)
23 HZ1(i)=horner(HZ,exp(-%i*%pi*f(i)/4));
24 end
25 HZ1=abs(HZ1);
26 HHPZ1=abs(horner(HHPZ,exp(-%i*%pi*f'/4)));
27 HBPZ1=abs(horner(HBPZ,exp(-%i*%pi*f'/4)));
28 HBSZ1=abs(horner(HBSZ,exp(-%i*%pi*f'/4)));
29 a=gca();
30 a.x_location="origin";
31 plot2d(f,HZ1);
32 plot2d(f,HHPZ1);
33 plot2d(f,HBPZ1);
34 plot2d(f,HBSZ1);
35 xlabel('Analog Frequency');
36 ylabel('Magnitude');
37 xtitle('Digital-to-digital transformations of a low
    pass filter');

```

Scilab code Exa 9.10a Bilinear Design of Second Order Filters

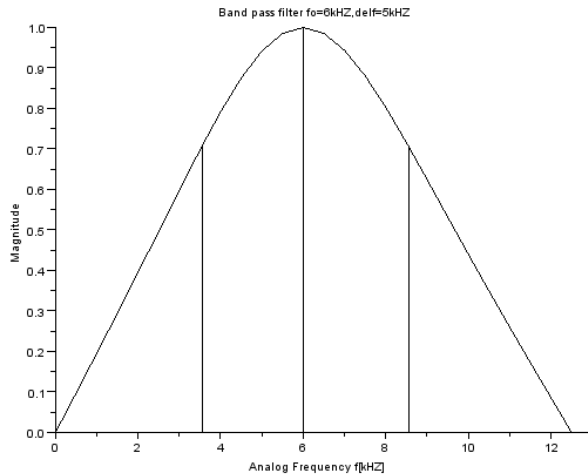


Figure 9.9: Bilinear Design of Second Order Filters

```

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

```

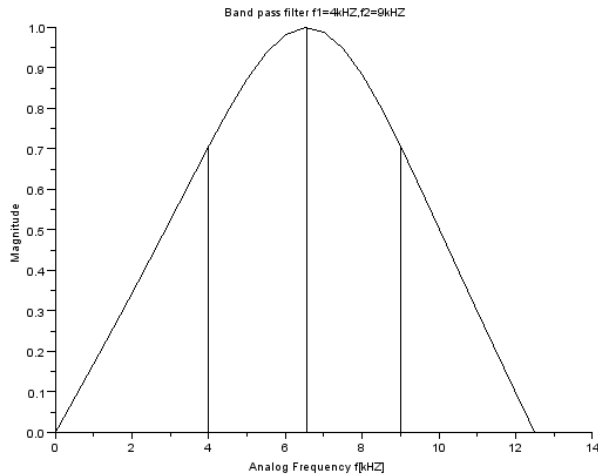


Figure 9.10: Bilinear Design of Second Order Filters

```

21 xlabel('Analog Frequency f [kHz] ');
22 ylabel('Magnitude ');
23 xtitle('Band pass filter fo=6kHz, delf=5kHz ');

```

Scilab code Exa 9.10b Bilinear Design of Second Order Filters

```

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

```

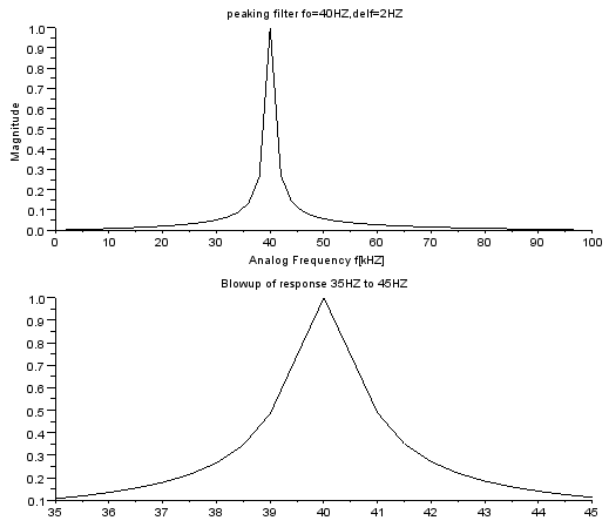


Figure 9.11: Bilinear Design of Second Order Filters

```

11 HZ1=abs(HZ1);
12 fo=S*acos(B)/(2*%pi)
13 f3=[f1 fo f2];
14 HZf=abs(horner(HZ,exp(-%i*2*%pi*f3'/25)));
15 a=gca();
16 a.x_location="origin";
17 plot2d(f,HZ1);
18 plot2d3('gmn',f3,HZf);
19 xlabel('Analog Frequency f [kHz]');
20 ylabel('Magnitude');
21 xtitle('Band pass filter f1=4kHz, f2=9kHz');

```

Scilab code Exa 9.10c Bilinear Design of Second Order Filters

```

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

```

```

3 fo=40;Wo=2*%pi*fo/200;
4 delf=2;S=25;
5 delW=2*%pi*delf/200;
6 B=cos(2*%pi*fo/200)
7 K=0.557;
8 C=K*tan(0.5*delW)
9 HS=1/(s+1);
10 HZ=horner(HS,(z^2-(2*B*z)+1)/(C*(z^2)-C))
11 f=0:2:100;
12 f1=35:0.5:45;
13 HZ1=horner(HZ,exp(%i*2*%pi*f'/200));
14 HZ2=horner(HZ,exp(%i*2*%pi*f1'/200));
15 HZ1=abs(HZ1);
16 HZ2=abs(HZ2);
17 a=gca();
18 a.x_location="origin";
19 subplot(211);
20 plot2d(f,HZ1);
21 xlabel('Analog Frequency f[kHZ]');
22 ylabel('Magnitude');
23 xtitle('peaking filter fo=40HZ,delf=2HZ');
24 subplot(212);
25 plot2d(f1,HZ2);
26 xtitle('Blowup of response 35HZ to 45HZ');

```

Scilab code Exa 9.11 Interference Rejection

```

1 //interference Rejection
2 //design oh high-Q and low-Q notch filters
3 s=%s;z=%z;
4 Q=50;
5 fo=60;S=300;
6 delf=fo/Q;

```

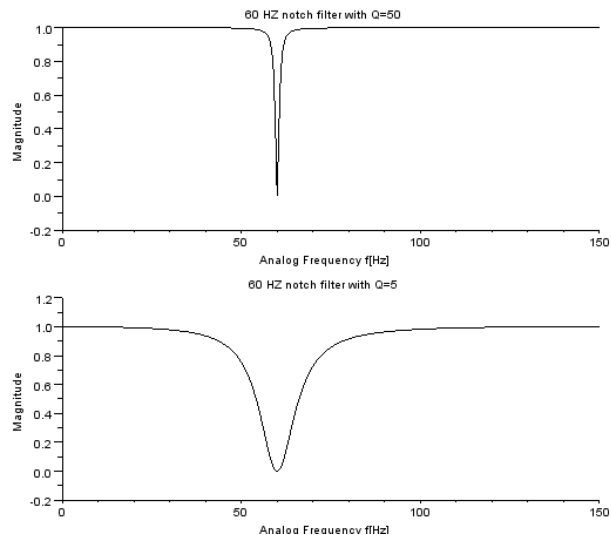


Figure 9.12: Interference Rejection

```

7  Wo=2*%pi*fo/S;
8  delW=2*%pi*delf/S;
9  C=tan(0.5*delW),B=cos(Wo)
10 HS=(s)/(s+1);
11 H1Z=horner(HS,(z^2-(2*B*z)+1)/(C*(z^2)-C))
12 Q1=5;delf1=fo/Q1;
13 delW1=2*%pi*delf1/S;
14 C1=tan(0.5*delW1),B1=cos(Wo)
15 H2Z=horner(HS,(z^2-(2*B1*z)+1)/(C1*(z^2)-C1))
16 f=0:0.5:150;
17 H1Z1=horner(H1Z,exp(%i*2*%pi*f'/S));
18 H2Z1=horner(H2Z,exp(%i*2*%pi*f'/S));
19 a=gca();
20 subplot(211);
21 plot2d(f,H1Z1);
22 xlabel('Analog Frequency f[Hz]');
23 ylabel('Magnitude');
24 xtitle('60 HZ notch filter with Q=50');
25 subplot(212);
26 plot2d(f,H2Z1);

```

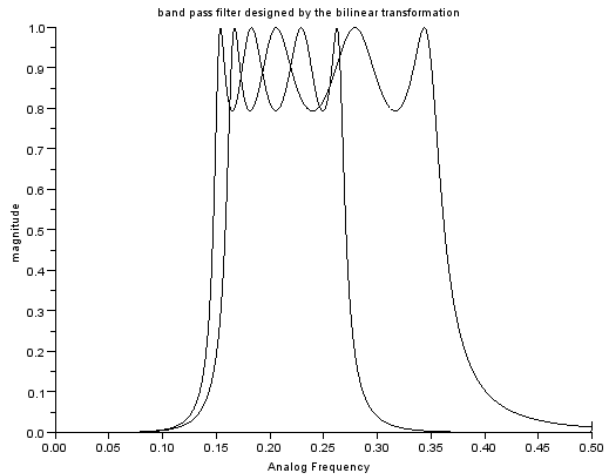


Figure 9.13: IIR Filter Design

```

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

```

Scilab code Exa 9.12 IIR Filter Design

```

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

```



```

9  omega=2*tan(0.5*W'); //prewarping each band edge
    frequency
10  epsilon=sqrt(10^(0.1*Ap)-1);
11  n=acosh(((10^(0.1*As)-1)/epsilon^2)^1/2)/(acosh(fs1/
    fp1));
12  n=ceil(n)
13  alpha=(1/n)*asinh(1/epsilon);
14  for i=1:n
15      B(i)=(2*i-1)*%pi/(2*n);
16  end
17  for i=1:n
18      p(i)=-sinh(alpha)*sin(B(i))+i*cosh(alpha)*cos(B
        (i));
19  end
20  Qs=1;
21  for i=1:n
22      Qs=Qs*(s-p(i))
23  end
24  Qo=0.1634;
25  HPS=Qo/Qs
26  HBPS=horner(HPS,(s^2+1.5045^2)/(s*1.202))
27  HZ=horner(HBPS,2*(z-1)/(z+1))
28  f=0:0.001:0.5;
29  HZF=abs(horner(HZ,exp(i*2*pi*f')));
30  HBPF=abs(horner(HBPS,i*2*pi*f'));
31  a=gca();
32  plot2d(f,HZF);
33  plot2d(f,HBPF);
34  xlabel('Analog Frequency');
35  ylabel('magnitude');
36  xtitle('band pass filter designed by the bilinear
    transformation');

```

Chapter 10

Design of FIR filters

Scilab code Exa 10.2 Truncation and Windowing

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

```

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

```

Scilab code Exa 10.3ab FIR lowpass Filter design

```

1 //FIR filter design using windows
2 //(a) Design of FIR filter to meet following
   specifications
3 fp=2; fs=4; Ap=2; As=40; S=20;
4 Fp=fp/S; Fs=fs/S;
5 Fc=0.15;
6 z=%z;
7 N1=3.21/(Fs-Fp);
8 N1=ceil(N1)
9 N2=5.71/(Fs-Fp);
10 N2=ceil(N2)
11 n1=-16:16;
12 n2=-28.5:1:28.5;
13 hn1=2*Fc*(sinc(2*Fc*n1'));
14 hn2=2*Fc*(sinc(2*Fc*n2'));
15 Wn1=0.5+0.5*(cos(2*pi*n1'/(N1-1))); //Vonhann window
16 Wn2=0.42+0.5*(cos(2*pi*n2'/(N2-1)))+0.08*(cos(4*pi
   *n2'/(N2-1))); //Blackman window
17 hwn1=abs(hn1.*Wn1);
18 hwn2=abs(hn2.*Wn2);
19 [hwn1F, fr1]=frmag(hwn1, 256);
20 [hwn2F, fr2]=frmag(hwn2, 256);
21 hwn1F1=20*log10(hwn1F);
22 hwn2F1=20*log10(hwn2F);
23 plot2d(fr1, hwn1F1);
24 plot2d(fr2(1:length(fr2)-2), hwn2F1(1:length(fr2)-2))
   ;
25 xlabel('Digital frequency');
26 ylabel('Magnitude [dB]');
27 title('Low pass filter using vonhann and Blackmann

```

```

        windows Fc=0.15,vonhann N=33,Blackman N=58');
28 // (b) Minimum length design
29 Fcv=0.1313;
30 Fcb=0.1278;
31 Nv=23; Nb=29;
32 nv=-11:11;
33 nb=-14:14;
34 hnv=2*Fcv*(sinc(2*Fcv*nv'));
35 hnb=2*Fcb*(sinc(2*Fcb*nb'));
36 Wnv=0.5+0.5*(cos(2*pi*nv'/(Nv-1))); // Vonhann window
37 Wnb=0.42+0.5*(cos(2*pi*nb'/(Nb-1)))+0.08*(cos(4*pi
    *nb'/(Nb-1))); // Blackman window
38 hwnv=abs(hnv.*Wnv);
39 hwnb=abs(hnb.*Wnb);
40 [hwnvF, frv]=frmag(hwnv,256);
41 [hwnbF, frb]=frmag(hwnb,256);
42 hwnvF=20*log10(hwnvF);
43 hwnbF=20*log10(hwnbF);
44 b=gca();
45 xset('window',2);
46 plot(frv,hwnvF);
47 plot(frb,hwnbF);
48 xlabel('Digital frequency');
49 ylabel('Magnitude [dB]');
50 title('Vonhann Fc=0.1313, Minimum N=23, Blackmann Fc
    =0.1278, Minimum N=29');

```

Scilab code Exa 10.3cd FIR filter Design

```

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

```

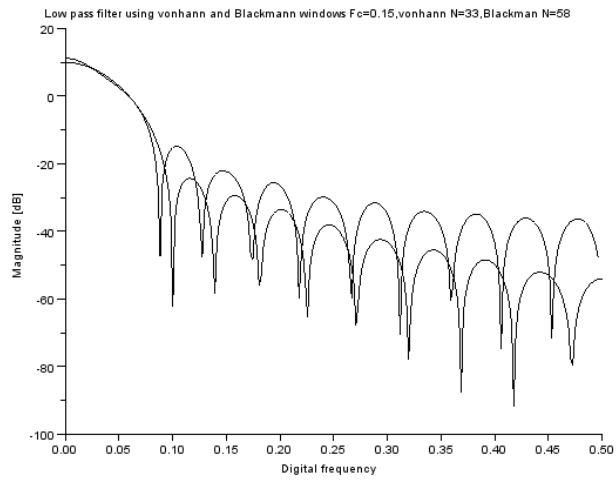


Figure 10.1: FIR lowpass Filter design

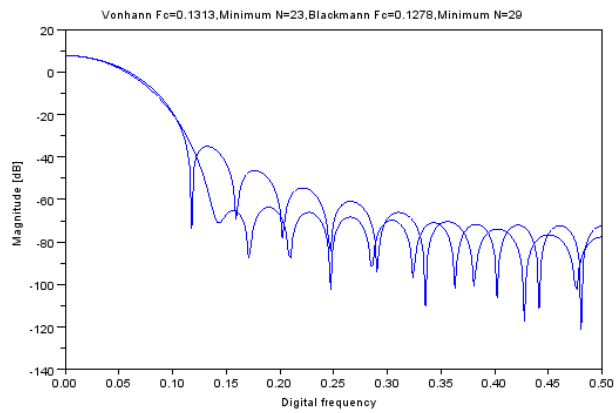


Figure 10.2: FIR lowpass Filter design

```

3 fp=2;fs=4;Ap=2;As=40;S=20;
4 Fp=fp/S;Fs=fs/S;
5 Ft=0.1;
6 Fc=0.15
7 N1=3.47/(Fs-Fp);//hamming
8 N1=int(N1)+1
9 N2=5.71/(Fs-Fp);//blackman
10 N2=int(N2)+1
11 [hn1]=eqfir(N1,[0 0.1;0.2 0.5],[0 1],[1 1]);
12 [HF1,fr1]=frmag(hn1,512);
13 Hf1=20*log10(HF1);
14 [hn2]=eqfir(58,[0 0.1;0.2 0.43],[0 1],[1 1]);
15 [HF2,fr2]=frmag(hn2,512);
16 Hf2=20*log10(HF2);
17 a=gca();
18 plot2d(fr1,Hf1,rect=[0 -120 0.5 4]);
19 plot2d(fr2(1:length(fr2)-5),Hf2(1:length(fr2)-5),
    rect=[0 -120 0.5 4]);
20 xlabel('Digital Frequency F');
21 ylabel('Magnitude [dB]');
22 xtitle('High pass filter using Hamming and Blackmann
    windows LPP Fc=0.35');
23 //Minimum Length Design
24 [hn3]=eqfir(22,[0 0.1;0.2 0.43],[0 1],[1 1]);
25 [HF3,fr3]=frmag(hn3,512);
26 Hf3=20*log10(HF3);
27 [hn4]=eqfir(29,[0 0.1;0.2 0.5],[0 1],[1 1]);
28 [HF4,fr4]=frmag(hn4,512);
29 Hf4=20*log10(HF4);
30 xset('window',1);
31 a=gca();
32 plot2d(fr3(1:length(fr3)-5),Hf3(1:length(fr3)-5),
    rect=[0 -120 0.5 4]);
33 plot2d(fr4,Hf4,rect=[0 -120 0.5 4]);
34 xlabel('Digital Frequency F');
35 ylabel('Magnitude [dB]');
36 xtitle('Hamming LPP Fc=0.3293 N=22;Blackmann LPP Fc
    =0.3277 N=29');

```

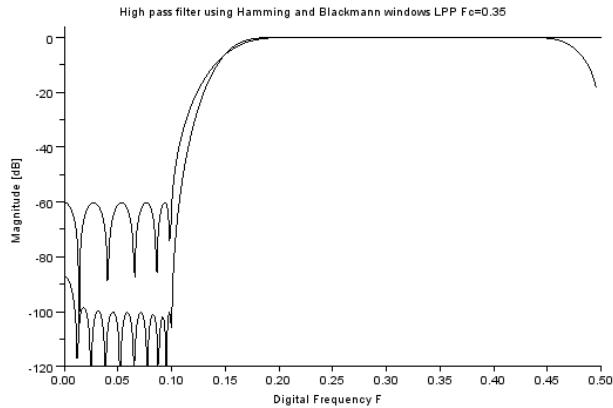


Figure 10.3: FIR filter Design

Scilab code Exa 10.4a Half Band lowpass FIR filter Design

```

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

```

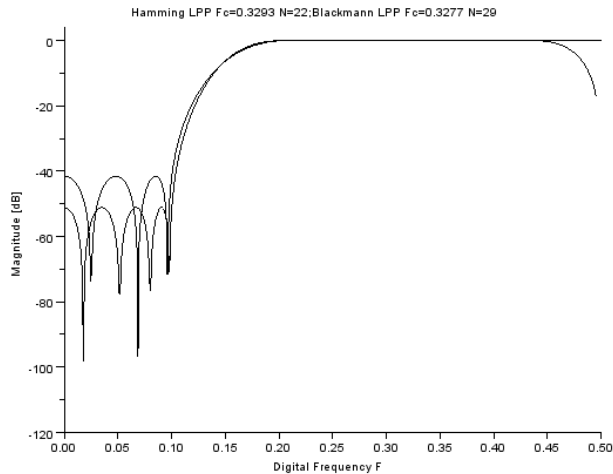


Figure 10.4: FIR filter Design

```

12 N1=int(N1)+1;
13 B=0.0351*(As0-8.7)
14 [hn1]=eqfir(19,[0 1/6;1/3 0.5],[1 0],[1 1]);
15 [HLPF1,fr1]=frmag(hn1,512);
16 HLPf1=20*log10(HLPF1);
17 a=gca();
18 plot2d(fr1,HLPf1);
19 xlabel('Digital Frequency');
20 ylabel('Magnitude in dB');
21 xtitle('Kaiser half band LPF:B=1.44;Fc=0.25');
22 [hn2]=eqfir(21,[0 1/6;1/3 0.5],[1 0],[1 1]);
23 [HLPF2,fr2]=frmag(hn2,512);
24 HLPf2=20*log10(HLPF2);
25 xset('window',1);
26 plot2d(fr2,HLPf2);
27 xlabel('Digital Frequency');
28 ylabel('Magnitude in dB');
29 xtitle('Hamming half-band LPF:N=21;Fc=0.25');

```

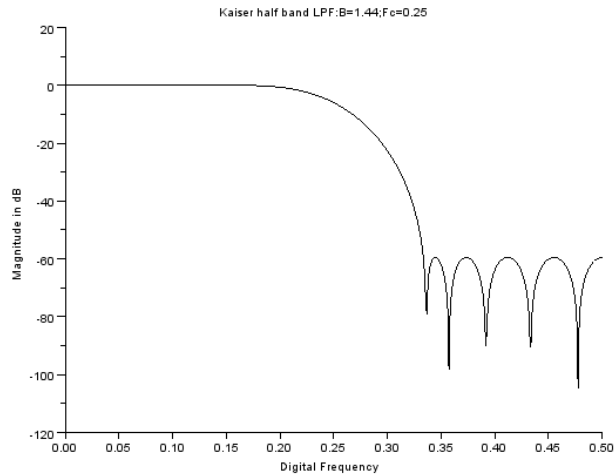


Figure 10.5: Half Band lowpass FIR filter Design

Scilab code Exa 10.4b Half Band bandstop FIR filter Design

```

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

```

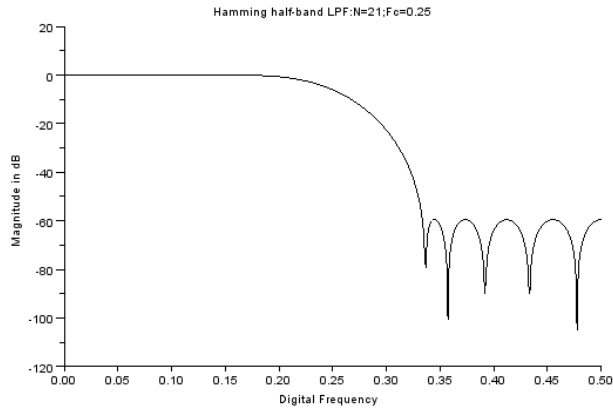


Figure 10.6: Half Band lowpass FIR filter Design

```

12 N1=(As0-7.95)/(14.36*(Fs-Fp))+1;
13 N1=ceil(N1);
14 B=0.0351*(As0-8.7)
15 [hn1]=eqfir(31,[0 0.1;0.2 0.3;0.4 0.5],[1 0 1],[1 1
    1]);
16 [HBSF1,fr1]=frmag(hn1,400);
17 HBSf1=20*log10(HBSF1);
18 a=gca();
19 plot2d(fr1,HBSf1);
20 xlabel('Digital Frequency');
21 ylabel('Magnitude in dB');
22 xtitle('Kaiser half band LPF:B=1.44;Fc=0.25');
23 [hn2]=eqfir(35,[0 0.1;0.2 0.3;0.4 0.5],[1 0 1],[1 1
    1]);
24 [HF2,fr2]=frmag(hn2,200);
25 HBSf2=20*log10(HF2);
26 xset('window',1);
27 plot2d(fr2,HBSf2);
28 xlabel('Digital Frequency');
29 ylabel('Magnitude in dB');

```

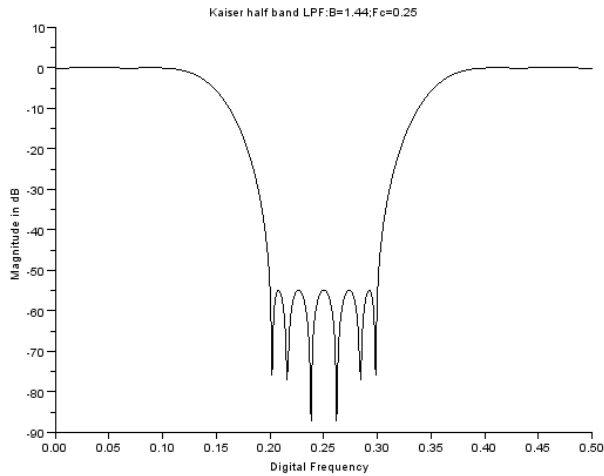


Figure 10.7: Half Band bandstop FIR filter Design

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

Scilab code Exa 10.5a Design by Frequency Sampling

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

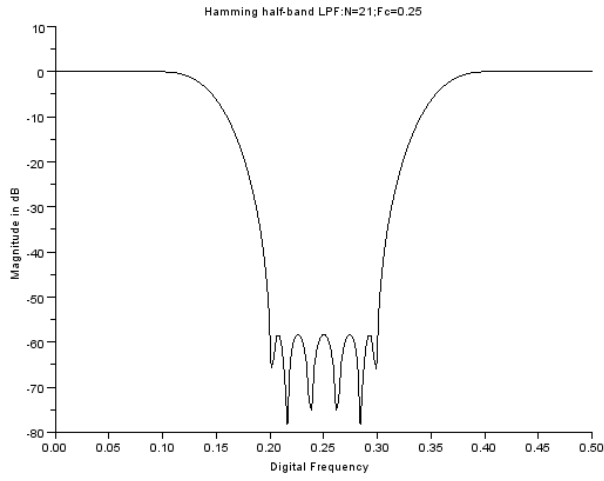


Figure 10.8: Half Band bandstop FIR filter Design

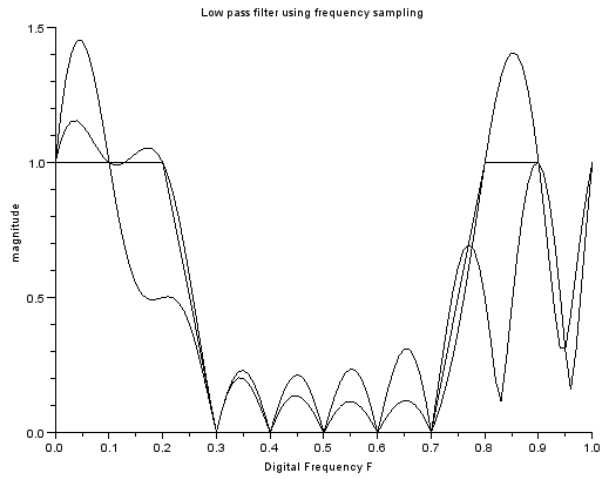


Figure 10.9: Design by Frequency Sampling

```

9  end
10 H1n=(fft(H1k,1));
11 H2k=H1k;
12 H2k(3)=0.5*%e^(-%i*1.8*%pi);
13 H2k(9)=0.5*%e^(%i*1.8*%pi);
14 H2n=(fft(H2k,1));
15 H1Z=0;H2Z=0;
16 for i=1:length(H1n)
17     H1Z=H1Z+H1n(i)*z^(-i);
18 end
19 for i=1:length(H2n)
20     H2Z=H2Z+H2n(i)*z^(-i);
21 end
22 F=0:0.01:1;
23 F1=0:0.1:0.9;
24 H1F=abs(horner(H1Z,exp(%i*2*%pi*F')));
25 H2F=abs(horner(H2Z,exp(%i*2*%pi*F')));
26 a=gca();
27 plot2d(F1,magHk);
28 plot2d(F,H2F);
29 plot2d(F,H1F);
30 xlabel('Digital Frequency F');
31 ylabel('magnitude');
32 xtitle('Low pass filter using frequency sampling');

```

Scilab code Exa 10.5b Design by Frequency Sampling

```

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

```

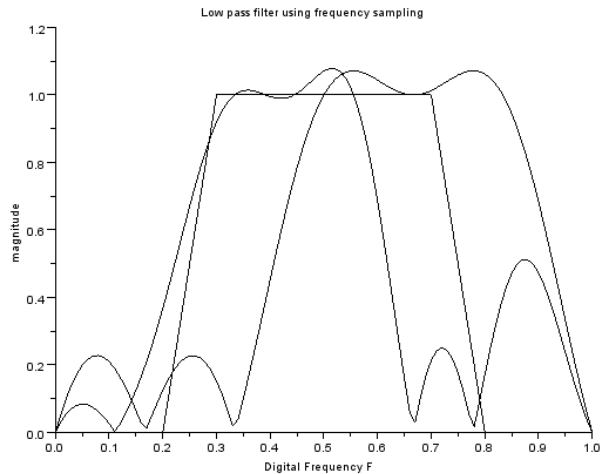


Figure 10.10: Design by Frequency Sampling

```

7 for i=1:length(fik)
8     H1k(i)=magHk(i)*exp(%i*fik(i));
9 end
10 H1n=(fft(H1k,1));
11 H2k=H1k;
12 H2k(3)=0.5*%e^(-%i*1.3*%pi);
13 H2k(9)=0.5*%e^(%i*1.3*%pi);
14 H2n=(fft(H2k,1));
15 H1Z=0;H2Z=0;
16 for i=1:length(H1n)
17     H1Z=H1Z+H1n(i)*z^(-i);
18 end
19 for i=1:length(H2n)
20     H2Z=H2Z+H2n(i)*z^(-i);
21 end
22 F=0:0.01:1;
23 F1=0:0.1:0.9;
24 H1F=abs(horner(H1Z,exp(%i*2*%pi*F')));
25 H2F=abs(horner(H2Z,exp(%i*2*%pi*F')));
26 a=gca();

```

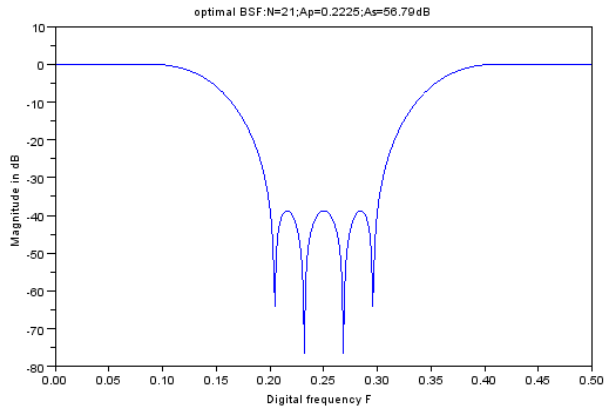


Figure 10.11: Optimal FIR Bandstop Filter Design

```

27 plot2d(F1, magHk);
28 plot2d(F, H2F);
29 plot2d(F, H1F);
30 xlabel('Digital Frequency F');
31 ylabel('magnitude');
32 xtitle('Low pass filter using frequency sampling');

```

Scilab code Exa 10.6a Optimal FIR Bandstop Filter Design

```

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

```

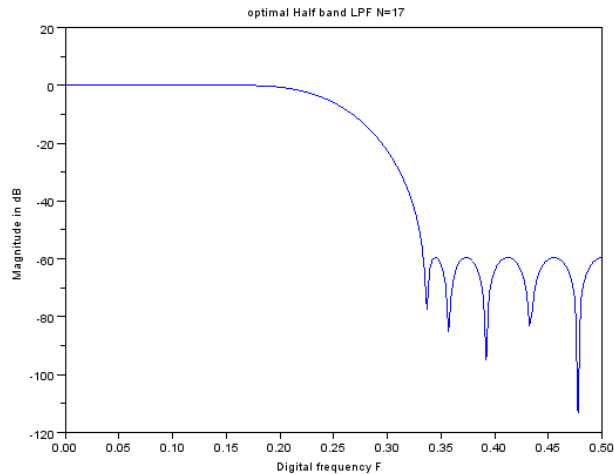


Figure 10.12: Optimal Half Band Filter Design

```

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

```

Scilab code Exa 10.6b Optimal Half Band Filter Design

```

1 //optimal Fir band pass filter design

```



```

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

```

Scilab code Exa 10.7 Multistage Interpolation

```

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

```

```

14 fs=Sin-fp;
15 L1=4*Sout./(fs-fp);
16 L=0;
17 for i=1:length(L1)
18     L=L+L1(i);
19 end
20 disp('By using 2 stage interpolator filter length is
      :')
21 ceil(L)
22 //(c) 3 stage interpolator with I1=2;I2=3;I3=2
23 Sin=[4 8 24];
24 I=[2 3 2];
25 Sout=[8 24 48];
26 fp=[1.8 1.8 1.8];
27 fs=Sin-fp;
28 L2=4*Sout./(fs-fp);L=0;
29 for i=1:length(L2)
30     L=L+L2(i);
31 end
32 disp('By using 3 stage interpolator filter length is
      :')
33 ceil(L)
34 //(d) 3 stage interpolator with I1=2;I2=3;I3=2
35 Sin=[4 12 24];
36 I=[3 2 2];
37 Sout=[12 24 48];
38 fp=[1.8 1.8 1.8];
39 fs=Sin-fp;
40 L3=4*Sout./(fs-fp);L=0;
41 for i=1:length(L3)
42     L=L+L3(i);
43 end
44 disp('By using 2 stage interpolator filter length is
      :')
45 ceil(L)

```

Scilab code Exa 10.8 Design of Interpolating Filters

```
1 //Design of interpolating filters
2 //(a)Design using a single stage interpolator
3 fp=1.8;Sout=48;Sin=4;
4 Ap=0.6;As=50;
5 fs=Sin-fp;
6 //finding ripple parameters
7 delp=(10^(Ap/20)-1)/(10^(Ap/20)+1);
8 dels=10^(-As/20);
9 N=Sout*(-10*log10(delp*dels)-13)/(14.6*(fs-fp))+1;
10 disp('By using single stage interpolator the filter
      design is:');
11 ceil(N)
12 //Design using 3-stage interpolator with I1=2;I2=3;
      I3=2
13 Ap=0.2;
14 Sin=[4 8 24];
15 Sout=[8 24 48];
16 fp=[1.8 1.8 1.8];
17 fs=Sin-fp;
18 delp=(10^(Ap/20)-1)/(10^(Ap/20)+1);
19 dels=10^(-As/20);
20 p=14.6*(fs-fp);
21 N1=(-10*log10(delp*dels)-13)./p);
22 N1=(Sout.*N1)+1;N=0;
23 for i=1:length(N1)
24     N=N+N1(i);
25 end
26 disp('By using single stage interpolator the filter
      design is:');
27 ceil(N)
```

Scilab code Exa 10.9 Multistage Decimation

```
1 //The concept of multistage Decimation
2 //(a)Single stage decimator
3 Sin=48;Sout=4;
4 fp=1.8;
5 fs=Sout-fp;
6 FT=(fs-fp)/Sin;
7 disp('By using single stage the total filter length
      is:')
8 L=4/FT
9 //(b)Two-stage decimator
10 Sin=[48 12];
11 D=[4 3];//decimating factors
12 Sout=[12 4];
13 fp=[1.8 1.8];
14 fs=Sout-fp;
15 L1=4*Sin./(fs-fp);
16 L=0;
17 for i=1:length(L1)
18     L=L+L1(i);
19 end
20 disp('By using 2 stage decimator filter length is:")
21 ceil(L)
22 //3 stage decimator with D1=2;D2=3;D3=2
23 Sin=[48 24 8];
24 D=[2 3 2];
25 Sout=[24 8 4];
26 fp=[1.8 1.8 1.8];
27 fs=Sout-fp;
28 L2=4*Sin./(fs-fp);L=0;
29 for i=1:length(L2)
30     L=L+L2(i);
31 end
```

```

32 disp('By using 3 stage decimator filter length is:')
33 ceil(L)
34 //3 stage decimator with I1=2;I2=3;I3=2
35 Sin=[48 24 12];
36 D=[2 2 3];
37 Sout=[24 12 4];
38 fp=[1.8 1.8 1.8];
39 fs=Sout-fp;
40 L3=4*Sin./(fs-fp);L=0;
41 for i=1:length(L3)
42     L=L+L3(i);
43 end
44 disp('By using 3 stage decimator filter length is:')
45 ceil(L)

```

Scilab code Exa 10.10 Maximally Flat FIR filter Design

```

1 //Maximally flat FIR filter design
2 Fp=0.2;
3 Fs=0.4;
4 Fc=0.3;
5 Ft=0.2;
6 N0=1+0.5/Ft^2;
7 N0=ceil(N0);
8 alpha=(cos(%pi*Fc))^2;
9 k=5;Mmin=14;
10 L=Mmin-k;
11 N=2*Mmin-1;
12 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

```
1 function [F]=aliasfrequency(f,s,s1)
2 if (s>2*f) then
3     disp(" alias has not occured")
4 else
5     disp(" alias has occured")
6 end
7 F=f/s;
8 for i=1:100
9     if (abs(F)>0.5)
10        F=F-i;
11    end
12 end
13 fa=F*s1;
14 disp(fa," frequency of reconstructed signal is")
15 endfunction
```
