# Scilab Textbook Companion for Digital Signal Processing: A Computer Based Approach by S. K. Mitra[1]

Created by
Sanjeev Irny
B.Tech
Instrumentation Engineering
Shri Gurugobind Singhji Institute of Engg. & Tech.
College Teacher
Dr. B. M. Patre
Cross-Checked by

May 24, 2016

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

# Book Description

**Title:** Digital Signal Processing: A Computer Based Approach

**Author:** S. K. Mitra

**Publisher:** Tata McGraw - Hill Education

**Edition:** 3

**Year:** 2008

**ISBN:** 978-0-07-066756-3

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

# Chapter 2

# Discreet Time Signals and Systems

**Scilab code Exa 2.1** Ensemble Averaging

```
//EXAMPLE 2.1
//Ensemble averaging
clear ;
clc ;
n = 1:50;
clf();
figure(0)
a=gca();
a.x_location="origin";
a.y_location="origin";
   for i =1:length(n)
       s(i)=2*n(i)*((0.9)^n(i));
       d(i)=(-0.1)^n(i);       //arbitrary noise
          signal.
   end

M=length(n);

 for i =1:M
```

```
    d(i)=(-0.1)^i;
    S=sum(d);
 end
Eav=(s+S/M)';            //Ensemble average.
disp(Eav,'The output of Ensemble averaging is')
plot2d3(n,s)
plot(n,s,'r.')
xtitle('Ensemble averaging','n','Eav-s');
a.children.children.thickness=2;
a.children.children.foreground=2;
```

**Scilab code Exa 2.2** Basic operations

```
//EXAMPLE 2.2 , BASIC OPERATIONS.
clear;
clc;
c=[3.2 41 36 -9.5 0];
disp(c,'c = ');
d=[1.7 -0.5 0 0.8 1];
disp(d,'d = ');
w1=c.*d;                    //Multiplication
disp(w1,'The product of two input vectors is =');
w2=c+d;                     //addition
disp(w2,'The addition of two input vectors is =');
w3=3.5*c;
disp(w3,'The scaling of first input vector is =');
```

**Scilab code Exa 2.3** Unequal length sequence

```
//EXAMPLE 2.3, Basic ops on unequal length sequence
clear;
clc;
c=[3.2 41 36 -9.5 0];
```

```
disp(c,'c = ');
g=[-21 1.5 3];
disp(g,'g = ');
a=length(g);
b=length(c);
i=0;
    while(i<b-a)
        g(b-i)=0;
        i=i+1;
    end
w4=g.*c;
disp(w4,'The product of two sequences is =');
w5=c+g;
disp(w5,'The addition  of two sequences is =');
```

**Scilab code Exa 2.5** Generating symmetric parts

```
//EXAMPLE 2.5,Conjugate-Antisymmetric & Conjugate-symmetric parts of Sequence
clc;
clear;
g=[0, 1+%i*4, -2+%i*3, 4-%i*2, -5-%i*6, -%i*2, 3];
disp(g,'g = ')
g1=conj(g);             //Conjugate of g;
disp(g1,conj(g));
a=length(g);
for i=1:a
    g2(1,i)=g1(a-i+1);
end

gcs=(g+g2)/2          //Conjugate-Symmetric part
disp(gcs,'The Conjugate symmetric part is =');
gcas=(g-g2)/2;        //Conjugate-Antisymmetric part
disp(gcas,'The Conjugate antisymmetric part is =');
```

**Scilab code Exa 2.6** Energy Signal

```
//EXAMPLE 2.6, Energy Signal

clear ;
clc ;
n = -5:5;
for i =1:length ( n )
 if(n(i)>=1)
 h(i)=1/n(i);
 else
 h(i)=0;
 end
end

Sum=0;
N=1:10000;
  for i=1:length(N)
  h(i)=(1/N(i))^2;
  end

Energy = sum(h);

 if (Energy<%inf ) then
  disp ('Energy Signal') ;
  disp(Energy,'Energy of signal = ');
  else
  if  (Energy/length(N)<%inf ) then
  disp ('Power Signal') ;

  else
  disp ('Niether Energy nor Power Signal') ;
  end
 end
```

**Scilab code Exa 2.7** Power Signal

```
//EXAMPLE 2.7,Example of Power signal
clear ;
clc ;

Sum=0;
N=1:10000;
    for i=1:length(N)
     h1= 3*((-1)^i);
     h=h1^2;
    end

Energy = sum(h);
  if  (Energy/(2*(length(N))+1)<%inf ) then
  disp ('Power Signal') ;
  disp(Energy/2,'Power Signal = ');
  else
  disp ('Not a Power Signal') ;
  end
```

---

**Scilab code Exa 2.9** Square wave generation

```
//EXAMPLE 2.9, Generation of a Square wave sequence:
clc;
clear;
clf();
a=gca();
figure(0);
a.x_location="origin";
x=[0:1:80];
y1=sin(x*.05*%pi);
```

```
y2=sin(x*.15*%pi);
y3=sin(x*.25*%pi);
y4=y1+y2/3+y3/5;
plot2d3(x,y4,2)
plot(x,y4,'r.')
xtitle('Approximate Square wave','x','y4');
a.children.children.thickness=3;
```

**Scilab code Exa 2.16** Linearity of accumulator

```
//EXAMPLE 2.16,
clear;
clc;
//Given input sequence = [3 4 5]
x=[0 3 4 5 0];
disp([3 4 5],'Input sequence = ')
//determining median filter
//first sequence
for k=2:4
    if  x(k)>x(k-1) & x(k+1)>x(k-1) & x(k+1)>x(k)
         y(k-1)=x(k);
     else
         x(k-1)>x(k+1) & x(k)>x(k+1) & x(k)>x(k-1)
         y(k-1)=x(k-1);
    end
end
disp(y','The Median Filter of the given input is =')
   ;
```

**Scilab code Exa 2.20** Passive system

```
//EXAMPLE 2.20,Passive or lossless system.
clear;
```

```
clc;
a=input("any value of a less than or equal to one")
n=-10:1:10;
 x=n;
 y=a*n;
 S=0;
 for i=1:length(n)
     S=S+y^2;
 end

 if a<1 then
     disp('the system is passive')
 else

     a==1
     disp('the system is lossless')

 end
```

**Scilab code Exa 2.22** Impulse response of Accumulator

```
//EXAMPLE 2.22,impulse response of accumulator

clear;
clc;
d=[1];
t=-1:.01:1;
h=0;
clf();
figure(0);
a=gca();
a.x_location="origin";

for i=1:length(t)
    if t(i)<0
```

```
        h=0;
    else
        h=d;
        plot2d3(i-101,h)
        plot(i-101,h,'.r')
        xtitle('Impulse Response of accumulator','t'
           ,'Y');
        a.children.children.thickness=1;
        a.children.children.foreground=2;
    end
end
disp(h,'The impulse response of Accumulator is =')
```

**Scilab code Exa 2.26** Convolution

```
//EXAMPLE 2.26,convolution of x & h
x=[-2 0 1 -1 3];
disp(x,'x = ');
h=[1 2 0 -1];
disp(h,'h = ');
n=0:7;
y=convol(x,h);
disp(y,'The convolution of the two inputs is :')
```

**Scilab code Exa 2.27** Convolution

```
//EXAMPLE 2.27, convolution of an exponential
   sequence
clear;
clc;
n=0:.5:5
c=0.5;
b=0.4;
```

```
clf();
figure(0);
a=gca();
a.x_location="origin";
x = c^n;
subplot(2,2,1);
plot2d3(n,x,2);
plot(n,x,'.r');
xtitle('','n','x');
h = b^n ;
subplot(2,2,2);
plot2d3(n,h,2)
plot(n,h,'.r')
xtitle('','n','h');
N=0:.5:10;
y = convol (x , h );
subplot(2,2,3);
plot2d3(N,y,2)
plot(N,y,'.r')
xtitle('convol(x,h)' ,'n','y');
disp(y,'Convolution of the two exponential sequences
    is =')
```

---

**Scilab code Exa 2.28** Convolution

```
//EXAMPLE 2.28,graphical representation of
   convolution of x & h.
clear;
clc;
x=[-2 0 1 -1 3];
disp(x,'x');
h=[1 2 0 -1];
disp(h,'h');
n=0:7;
y=convol(x,h);
```

```
disp(y,'convolution = ');
clf();
figure(0);
a=gca();
a.x_location="origin";
a.y_location="origin";
plot2d3(n,y)
plot(n,y,'r.')
xtitle('convolution','n','Y');
a.children.children.thickness=2;
a.children.children.foreground=2;
```

**Scilab code Exa 2.29** Convolution

```
//Example 2.29,Convolution using Tabular method.
clear;
clc;
x=[-2 0 1 -1 3];
h=[1 2 0 -1];
q=length(x);
w=length(h);
z=q+w-1;
y0=0;
for i=1:z;
    y(i)=0;
    for k=1:i;
        if k>q
            x(k)=0;
        else
            if (i-k+1)>w
                h(i-k+1)=0;
            else
             y(i)= y(i) + x(k)*h(i-k+1);
            end
        end
```

```
    end
end
disp(y','The Convolution of the two sequences is =')
```

**Scilab code Exa 2.30** Convolution

```
//EXAMPLE 2.30
//Convolution of two sided sequences
clear;
clc;
g=[3 -2 4];//originating at n=-1
h=[4 2 -1];//originating at n=0
q=length(g);
w=length(h);
z=q+w-1;
y0=0;
for i=1:z;
    y(i)=0;
    for k=1:i;
        if k>q
            g(k)=0;
        else
            if (i-k+1)>w
                h(i-k+1)=0;
            else
             y(i)= y(i) + g(k)*h(i-k+1);
            end
        end
    end
end
n=-1:z-2;
disp(y,'The Convolution of the two sequences is =')
clf();
a=gca();
figure(0);
```

```
a.x_location="origin";
plot2d3(n,y,2);
plot(n,y,'r.');
xtitle('convolution','n','y');
```

**Scilab code Exa 2.31** Stabbility of causal system

```
//EXAMPLE 2.31, Stability for causal system.
//h[i]=impulse response of LTI system.
clear;
clc;
n= -5:0.001:5;
a=0.6;

for i=1:length(n)
    if (n(i)<0)
        h(i)=0;
    else
        h(i)=abs(a^n(i));

    end
end
S=sum(h);
if(S<%inf)
    disp('BIBO stable system');
else
    disp('BIBO unstable system');

end
```

**Scilab code Exa 2.32** Stability of Anti causal system

```
//EXAMPLE2.32 Stability for anti-Causal system.
```

```
//h[i]=impulse response of LTI system.
clear;
clc;
n= -5:1/1000:5;
a=5;
for i=1:length(n)
    if (n(i)>-1)
        h(i)=0;
    else
        h(i)=a^n(i);
        S=sum(h);
    end
end

if(S<%inf)
    disp('BIBO stable system');
else
    disp('BIBO unstable system');

end
```

**Scilab code Exa 2.33** Stability of a system

```
//EXAMPLE 2.33 ,stability of finite impulse response
    .
//h[i]=impulse response of LTI system.
clear;
clc;
n= -5:1/100:5;
a= input('value of a');
N1=input('lower limit');
N2=input('upper limit');
for i=1:length(a)
    if (n(i)<N1 & n(i)>N2)
        h(i)=0;
```

```
    else
        h(i)=a^n(i);
        S=sum(h);
    end
end

if(S<%inf)
    disp('BIBO stable system');        //as long as
       N1,N2!=%inf
else
    disp('BIBO unstable system');

end
```

**Scilab code Exa 2.46** Cross coreation computation

```
//EXAMPLE 2.46, Cross corelation Computation.
// Given two finite length sequence.x[n],y[n]:
clear;
clc;
x=[1 3 -2 1 2 -1 4 4 2];
disp(x,'x');
y=[2 -1 4 1 -2 3];
disp(y,'y');
//Cross corelation rxy[n]:

rxy=convol(x,mtlb_fliplr(y));
disp(rxy,'The Cross-Corelation Operation of the
   Inputs is =')
```

# Chapter 3

# Discreet TIme Fourier Transform

**Scilab code Exa 3.5** DTFT computation

```
//EXAMPLE 3.5
//DTFT of unit sample sequence
clc;
clear;
//a=0.5;
n=0:9;
x = [1,zeros(1,9)];
disp(x,'x[n] = ')

K = 4;
k = 0:4/1000:4;
W = k*2*%pi/K;
X = (x)*exp(%i*n'*W);
disp(X,'DTFT,x[n] --> ')
X_mag   = abs(X);
X_phase = phasemag(X);//no phase exists

figure(0);
plot2d3(mtlb_fliplr(W),X_mag);
```

```
xtitle('Magnitude plot','W --->','X_mag --->');
figure(1);
plot2d3(mtlb_fliplr(W),X_phase);
xtitle(' zero phase plot','W --->','X_phase --->');
```

**Scilab code Exa 3.6** DTFT computation

```
//EXAMPLE 3.6
//Determine DTFT of sequence
//PROGRAM REQUIRES MAXIMA SCILAB TOOLBOX

clc;
clear;
//Symbolic calculation
Syms n w a ;
x1=(a^n)*exp(-%i*n*w);
X1=nusum(x1,n,0,%inf);
disp(X1,'DFT,X = ');

//Given:
a=0.5;
n=0:9;
//x[n]=a^n*u[n]
for i = 0:9
    x(i+1) = a^i;
end
//The DTFT of the sequence
K = 4;
k = 0:4/1000:4;
W = k*6*%pi/K;
X = (x')*exp(%i*n'*W);
X_mag   = abs(X);
[X_phase,db] = phasemag(X);

clf();
```

```
a=gca();
figure(0);
//Note %pi ~ 3.14
plot2d3(mtlb_fliplr(W),X_mag);
xtitle('Magnitude response','W --->','Amplitude --->');
figure(1);
plot2d3(mtlb_fliplr(W),X_phase);
xtitle('Phase response','W --->','Phase in degrees --->');
```

**Scilab code Exa 3.7** Plotting real and imaginary part

```
//EXAMPLE 3.12
//x[n]=((-1)^n)*(a^n)*u[n].....given a=0.5;

clc;
clear;

a=0.5;
n=0:9;
for i = 0:9
    x(i+1) = (a*exp(-%i*%pi))^i;
end

//The DTFT of the sequence
K = 4;
k = 0:4/1000:4;
W = k*6*%pi/K;
X = (x')*exp(%i*n'*W);
X_mag   = abs(X);
X_phase = phasemag(X);

//PLOTTING GRAPHS FOR THE INTERVAL OF 0 TO 6*%pi
clf();
```

```
a=gca();
figure(0);
plot2d3(mtlb_fliplr(W),X_mag);
xtitle('Magnitude response','W','Amplitude');
figure(1);
plot2d3(mtlb_fliplr(W),X_phase);
xtitle('Phase response','W','X_phase, degrees');
```

**Scilab code Exa 3.10** DTFT of finite length exponential sequence

```
//EXAMPLE 3.10
// DTFT of a sequence
clc;
clear;
syms a n M w;
x=a^n;
X=nusum(x*(exp(-%i*w*n)),n,0,M-1)
disp(limit(X),'The DTFT of the given sequence, X = '
    )
```

**Scilab code Exa 3.12** Plotting DTFT of exponential sequence

```
//EXAMPLE 3.12
//x[n]=((-1)^n)*(a^n)*u[n].....given a=0.5;

clc;
clear;

a=0.5;
n=0:9;
for i = 0:9
    x(i+1) = (a*exp(-%i*%pi))^i;
end
```

```
//The DTFT of the sequence
K = 4;
k = 0:4/1000:4;
W = k*6*%pi/K;
X = (x')*exp(%i*n'*W);
X_mag   = abs(X);
X_phase = phasemag(X);

//PLOTTING GRAPHS FOR THE INTERVAL OF 0 TO 6*%pi
clf();
a=gca();
figure(0);
plot2d3(mtlb_fliplr(W),X_mag);
xtitle('Magnitude response','W','Amplitude');
figure(1);
plot2d3(mtlb_fliplr(W),X_phase);
xtitle('Phase response','W','X_phase,degrees');
```

**Scilab code Exa 3.13** DTFT computation

```
//EXAMPLE 3.13

clc;
clear;
a=0.5;
n=0:9;
for i = 0:9
    x(i+1) = a^i;
end
//The DTFT of the sequence
K  = 4;
k  = 0:4/1000:4;
W  = k*6*%pi/K;
X1 = (x')*exp(%i*n'*W);
```

```
X   = %i*diff(X1);
X   = [X,0] + X1;

X_mag    = abs(X);
[X_phase,db] = phasemag(X);

clf();
a=gca();
figure(0);
plot2d3(mtlb_fliplr(W),X_mag);
xtitle('Magnitude response','W','X_mag');
figure(1);
plot2d3(mtlb_fliplr(W),X_phase);
xtitle('Phase response','W','X_phase');
```

**Scilab code Exa 3.14** Energy of signal

```
//EXAMPLE 3.14
//ENERGY OF LP DISCREET TIME SIGNAL
//PROGRAM REQUIRES MAXIMA SCILAB TOOLBOX
clc;
clear;
syms n wc w;
 wc = input("the value of wc ( less than %pi)= ");
n = -5:0.05:5;

for i =0:length (n)
    hlp(i+1) = (wc/%pi)*sinc((wc*i)/%pi);
    E(i+1)=(abs(hlp(i+1)))^2;
end

Energy = sum(E);
 if (Energy<%inf ) then
  disp ('The filter is Energy Signal') ;
  else
```

```
  if  (Energy/length(N)<%inf ) then
  disp ('Power Signal') ;
  else
  disp ('Niether Energy nor Power Signal') ;
  end
 end
disp(Energy,'the energy is = ');
```

**Scilab code Exa 3.15** Energy of exponential sequence

```
//EXAMPLE 3.15
//ENERGY OF A SIGNAL x[n]=a^n*u[n]
clc;
clear;
a=0.5;
n=0:0.1:9.9;
//x[n]=a^n*u[n]
for i = 0:length(n)
    x(i+1) = a^i;
    E=(abs(x))^2;
end
Energy=sum(E);
disp(Energy,'Energy of the signal = ');
```

# Chapter 4

# Digital Processing of Continous TIme Systems

**Scilab code Exa 4.5** Passband and Stopband ripple computation

```
//EXAMPLE 4.5
//determine ripple values in db;
clc
clear;
ap = 0.01//Peak passband ripple in dB
as = 70//min. stopband atteuation in dB
dp = 1-10^-(ap/20);
ds = 10^-(as/20);
disp( dp,'dp = ');
disp( ds,'ds = ');
```

**Scilab code Exa 4.6** Order of Analog filter

```
//EXAMPLE 4.6
//Order of LP filter
clc;
```

```
clear;
ap = 1 //Peak passband ripple in dB
as = 40 //min. stopband atteuation in dB
wp = 1000 //Hz
ws = 5000 //Hz
k  = wp/ws;
disp(1/k,'1/k = ');
k1 = 1/(sqrt((10^(0.1* as)-1)/(10^(0.1*ap)-1)));
disp(1/k1,'1/k1 = ');
N=ceil(log10(sqrt((10^(0.1* as)-1)/(10^(0.1*ap)-1)))/log10(1/k));
disp(N,'order of the filter is :');
```

**Scilab code Exa 4.7** Order of Analog Chebyshev Filter

```
//EXAMPLE 4.7
//Determine the order of Analog Chebyshev LP filter.
clc;
clear;
ap = 1 //dB
as = 40 //dB
wp = 1000 //Hz
ws = 5000 //Hz
k  = wp/ws;
disp(1/k,'1/k = ');
k1 = 1/(sqrt(((10^(0.1* as))-1)/((10^(0.1*ap))-1)));
disp(1/k1,'1/k1 = ');
N = acosh(1/k1)/acosh(1/k);
disp(N,'N = ');
disp('Since order of the filter is always an integer, ');
disp(ceil(N),'Order of the filter is, N = ');
```

**Scilab code Exa 4.8** Order of Analog Lowpass Elliptic Filter

```
//EXAMPLE 4.8
//Determine the order of Analog Elliptic LP filter.
clc;
clear;
ap = 1 //dB
as = 40 //dB
Fp = 1000 //Hz
Fs = 5000 //Hz
wp = Fp*2*%pi;
ws = Fs*2*%pi;


k1 = 1/(sqrt((10^(0.1* as)-1)/(10^(0.1*ap)-1)));
disp(1/k1,'1/k1 = ');
k  = wp/ws;
k2 = sqrt(1 - (k*k));
disp(k2,"k2 = ");
po = (1 - sqrt(k2))/(2*(1 + sqrt(k2)));
disp(po,'po = ');
p  = po +2*po^5 + 15*po^9 + 150*po^13;
disp(p,'p = ');
N = (2*log10(4/k1))/log10(1/p);
disp(N,'N = ');
disp('Since order of the filter is always an integer
    , ');
disp(ceil(N),'Order of the filter is, N = ');
```

**Scilab code Exa 4.16** Design of Analof Butterworth HP Filter

```
//EXAMPLE 4.16
//Design analog butterworth High pass filter
clc;
clear;
```

```
wp=4000;
ws=1000;
ap=0.1;
as=40;

Ap=1; // assumption
As=(2*%pi*wp)*Ap/(2*%pi*ws);

N=ceil(log10(sqrt((10^(0.1* as)-1)/(10^(0.1*ap)-1)))
   /log10(As/Ap));
disp(N,'order of the filter is :');

Ac = As/((10^(0.1*as)-1)^(1/(N*2)));
disp(Ac,'cutoff frequency = ')

//[hs,pole,zero,gain]=analpf(N,'butt',Ac);

s=%s;
hs=1/((s + 1)*(s^2 + 0.61803*s + 1)*(s^2 + 1.61803*s
    + 1));
Hs=horner(hs,s/Ac);
H1 = numer(Hs)/0.0976514;
H2 = denom(Hs)/0.0976514;
disp(H1/H2,'the low pass transfer function is ,HLP(s)
    = ');
Hs=horner(hs,Ac/s);
H1 = numer(Hs);
H2 = denom(Hs);
disp(H1/H2,'the High pass transfer function is ,HHP(s
   ) = ');
```

# Chapter 5

# Finite Length Discreet Transform

**Scilab code Exa 5.1** DFT computation

```
//EXAMPLE 5.1
//DETERMINE DFT OF GIVEN SEQUENCE
clc;
clear;
N = input("length of sequence = ");
x = [1,zeros(1,N-1)];
disp(x,'The sequence is,x = ');
X = dft(x,0);
disp(X,'DFT of the sequence is X = ');
m = input("value of some intemediate (mth) point =
    ");
y = [zeros(1,m-1),1,zeros(1,N-m)];
Y = dft(y,0);
disp(Y,'DFT of the sequence is Y = ');
```

**Scilab code Exa 5.2** DFT of sinusoidal sequence

```
//EXAMPLE 5.2
//DFT of sinusoidal sequence
clc;
clear;

N = input("    input value of N    ");
r = input("    input r value    ") ;
n = 0:N-1;
x = cos(2*%pi*r*n/N)
X = dft(x,-1)
//X exisits only at n={r,N-r} where X = N/2
clf();
a=gca();
a.x_location = "origin";
a.y_location = 'origin';
plot2d3(n,X,2);
a.thickness=1;
plot(n,X,'r.');

xtitle('DFT','K -->','X[K] -->');
X = disp(X,'DFT of x--> ');
```

**Scilab code Exa 5.3** DFT computation

```
//EXAMPLE 5.3
//DETERMINE DFT OF GIVEN SEQUENCE

clc;
clear;
N = input("length of sequence,N  =  ");
M = input("M point DFT = ");
if M > N
    x = [ones(1,N),zeros(1,M-N)];
    disp(x,'the sequence is :');
    for n=0:M-1
```

```
        for k=0:M-1
            W(n+1,k+1) = exp(-(%i*2*%pi*k/M)*n);
        end
    end
    X = W*x';
    disp(X,'DFT is , X = ');
else
    disp('invalid computation');
end


n=0:M-1;
clf();
figure(0)
a = gca();
plot2d3(n,x,2) //  plotting the sequence
plot(n,x,'r.');
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 2.5;
xtitle('original sequence','n','x[n]');

figure(1)
a = gca();
plot2d3(n,abs(X),2)  //  plotting absolute value of
   DFT of sequence
plot(n,abs(X),'r.');
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 2.5;
xtitle('magnitude plot','M','Absolute value');
```

**Scilab code Exa 5.4** IDFT Computation

```
//EXAMPLE 5.4
//DETERMINE IDFT OF GIVEN SEQUENCE
clc;
clear;
K = input("    value of K    ");
disp('input M > K');
M = input("    value of M    ");
k1 = 0:K-1;
V1 = k1./K;//DFT
k=0:M-1;

N = length(V1);
V = [V1,zeros(1,M-N)];
v = dft(V,1);//IDFT

clf();
subplot(1,2,1)

a = gca();
plot2d3(k,real(v),2);
plot(k,real(v),'r.');
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 2;
xtitle('real part','N','v');

subplot(1,2,2)
a = gca();
plot2d3(k,imag(v),2)
plot(k,imag(v),'r.');
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 2;
xtitle('imaginary part','N','v');
v = disp(v);
```

**Scilab code Exa 5.5** DFT computation

```
//EXAMPLE 5.5
//DFT computation
clc;
clear;

N = 16 ;
r = 3 ;
n = 0:N-1;
x = cos(2*%pi*r*n/N)
X = fft(x,-1)//DFT of the sequence
clf();
a = gca();
plot2d3(n,X,2);
plot(n,X,'r.')
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 3;
xtitle('DFT','k','X');
X = disp(real(X),'  X = ');
```

**Scilab code Exa 5.7** Cicular convolution computation

```
//EXAMPLE 5.7
//Circular convolution
clear;
clc;
g = [1 2 0 1];
disp(g,'g[n] = ');
h = [2 2 1 1];
```

```
disp(h,'h[n] = ');
G = fft(g,-1);
H = fft(h,-1);
Y = G.*H;
yc = fft(Y,1);
n1 = 0:length(yc)-1;
yl = convol(g,h);
n2 = 0:length(yl)-1;

clf();
subplot(2,1,1)
a = gca();
plot2d3(n1,yc,2);
plot(n1,yc,'r.');
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 3;
xtitle('circular convolution','n','yc');

subplot(2,1,2)
a = gca();
plot2d3(n2,yl,2);
plot(n2,yl,'r.');
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 3;
xtitle('linear convolution','n','yl');

disp(real(yc)," circular convolution,yc = ");
disp(yl," linear convolution,yl = ");
```

**Scilab code Exa 5.8** Cicular convolution computation

```
//EXAMPLE 5.8
//Cicular convolution
clc;
clear;
g = [1 2 0 1];
disp(g,'g[n] = ');
h = [2 2 1 1];
disp(h,'h[n] = ');
G = fft(g,-1);
H = fft(h,-1);
Y = G.*H;
yc = fft(Y,1);//IDFT of Y
disp(yc,"circular convolution,yc = ")
n=0:3;
clf();
figure(0);
a = gca();
plot2d3(n,yc,2);
plot(n,yc,'r.');
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 3;
xtitle('Circular convolution','n','yc');
```

**Scilab code Exa 5.10** Generating symmetric parts

```
//EXAMPLE 5_10
//conjugate symmetric & anti-symmetric parts of
    complex sequence
clear;
clc;
un=[1+%i*4,-2+%i*3,4-%i*2,-5-%i*6];
disp(un,'u[n] = ');
u1=conj(un);
```

```
disp(u1,'u*[n] = ');
//modulo-4 circularly time reversed version:
disp(pmodulo(0,4),'u[<-0>4] = ');
disp(pmodulo(-1,4),'u[<-1>4] = ');
disp(pmodulo(-2,4),'u[<-2>4] = ');
disp(pmodulo(-3,4),'u[<-3>4] = ');
un1=[u1(pmodulo(-0,4)+1),u1(pmodulo(-1,4)+1),u1(
    pmodulo(-2,4)+1),u1(pmodulo(-3,4)+1)];
disp(un1,'u*[<-n>4] = ');
disp(0.5*(un+un1),'ucs[n] = ');
disp(0.5*(un-un1),'uca[n] = ');
```

**Scilab code Exa 5.11** Cicular convolution computation

```
//EXAMPLE 5.11
//Circular convolution using DFT
clc;
clear;
g = [1 2 0 1];
disp(g,'g[n] = ');
h = [2 2 1 1];
disp(h,'h[n] = ');
M=4;
for n=0:M-1
        for k=0:M-1
            W(n+1,k+1) = exp(-(%i*2*%pi*k/M)*n);
        end
    end
    G = W*g';
    H = W*h';
disp(G,'DFT is, G = ');
disp(H,'DFT is, H = ');

Y=G.*H;
y=(1/4)*conj(W)*(Y);
```

```
disp(real(y),'Circular convolution = ');
```

**Scilab code Exa 5.12** Linear Convolution using DFT

```
//EXAMPLE 5.12
//Linear convolution using Circular convolution
clc;
clear;
g = [1 2 0 1];
disp(g,'g[n] = ');
h = [2 2 1 1];
disp(h,'h[n] = ');

//linea convolution length = 4+4-1 = 7
//appending the two signals with zeros
g = [g,zeros(1,3)]
h = [h,zeros(1,3)]
G = fft(g,-1);
H = fft(h,-1);
Y = G.*H;      //element wise multiplication
y = fft(Y,1);//IDFT

//Plotting linear convolution
n=0:6;
figure(0);
clf();
a = gca();
a.x_location = 'origin';
a.y_location = 'origin';
plot2d3(n,y,2);
plot(n,y,'r.');
poly1 = a . children (1) . children (1) ;
poly1.thickness = 2;
xtitle('Linear convolution','n','y');
disp(y," linear convolution ,y = ");
```

**Scilab code Exa 5.14** DFT computationusing single DFT

```
//EXAMPLE 5.14
//DFT of two real sequences using one DFT
clear;
clc;
g = [1 2 0 1];
disp(g,'g[n] = ');
h = [2 2 1 1];
disp(h,'h[n] = ');
x = g + %i.*h;
disp(x,'x[n] = ');
X = fft(x,-1);
disp(X,'The DFT, X[k] = ');
X1 = conj(X);
disp(X1,'X*[k] = ');

for i=0:3;
    a(i+1)=pmodulo(-i,4);
    X2(i+1)=X1(a(i+1)+1);
end

X3 = conj(X2');
disp(X3,'X*[<4-k>4] = ');
disp(0.5*(X + X3),'G[k] = ');
disp((X - X3)/(2*%i),'H[k] = ');
```

**Scilab code Exa 5.15** DFT computationusing single DFT of shorter length

```
//EXAMPLE 5.15
//DFT computation using DFT of shorter length sequences
```

```
clc;
clear;
v = [1 2 2 2 0 1 1 1];
disp(v,'Length-8 real sequence v[n] = ')
for i=1:4
    g(i)=v(2*i-1);
    h(i)=v(2*i);
end
G = fft(g,-1);
H = fft(h,-1);
M=length(v);
//for n=0:M-1
        for k=0:M-1
            W(1,k+1) = exp(-(%i*2*%pi*k/M)*1);
        end
//end
G=[G(1) G(2) G(3) G(4) G(1) G(2) G(3) G(4)] ;
H=[H(1) H(2) H(3) H(4) H(1) H(2) H(3) H(4)] ;
V=G + W.*H;
disp(V,'DFt,V[k] = ');















//for k = 0:3
```

```
//      V1(k+1) = G(k+1) + (exp(-2*%pi*%i*k/8))*H(k+1);
//end

//for k = 4:7
//      V2(k) = G(k-3) + (exp(-2*%pi*%i*k/8))*H(k-3);
//end
//disp([V1,zeros(1,3)]+V2)
```

# Chapter 6

# z Transform

**Scilab code Exa 6.1** z Transform of causal exponential sequence

```
//EXAMPLE 6.1
//Z−Transform of causal sequence
clc;
clear;
syms n a z;
x = a^n;
X = nusum(x*(1/z)^n,n,0,%inf);
limit(X);
disp(' X = ',X);
disp(' ROC = |z|>|a| ')
```

**Scilab code Exa 6.2** z transform of anticausal sequence

```

clc;
clear;
syms n a z;
x = a^n;
```

```
X = nusum(-x*(1/z)^n,n,-%inf,-1);
limit(X);
disp(' X = ',X);
disp(' ROC = |z|<|a| ')
```

**Scilab code Exa 6.3** z Transform

```
//EXAMPLE 6.3
//Z-Transform
clc;
clear;
syms n a z M N;
x = a^n;
X = nusum(x*(1/z)^n,n,M,N-1);
limit(X);
disp(X,' X = ');
```

**Scilab code Exa 6.4** z Transform

```
//EXAMPLE 6.4
//Z-Transform
clc;
clear;
syms n z;
x = (-0.6)^n;
X = nusum(x*((1/z)^n),n,0,%inf);
limit(X);
disp(X,' X = ');
```

**Scilab code Exa 6.5** Z transform of causal sequence

```
//EXAMPLE 6.9
//Z-Transform of causal sequence
clc;
clear;
//z=%z;
syms n a z M N;
x = a^n;
X = nusum(x*(1/z)^n,n,-M,N);
limit(X);
disp(X,' X = ');
```

**Scilab code Exa 6.9** z Transform

```
//EXAMPLE 6.9
//Determination of ROC
clc;
clear;
z=%z;
a=2*z^4+16*z^3+44*z^2+56*z+32;
b=3*z^4+3*z^3-15*z^2+18*z-12;
[h1,g1]=factors(a);
[h2,g2]=factors(b);
disp(h1,'h1 = ');
disp(h2,'h2 = ');
c=a/b;
disp(c,'function is = ');
plzr(c);
```

**Scilab code Exa 6.10** Rational form of z Transform from its zero and pole locations

```
//EXAMPLE 6.10
//Z-transform from pole-zero locations
```

```
clc;
clear;
z=%z;
//using the pole & zero locations provided
num=(z-0.21)*(z-3.14)*(z-(-0.3+%i*0.5))*(z-(-0.3-%i
    *0.5));
den=(z+0.45)*(z-0.67)*(z-(0.81+%i*0.72))*(z-(0.81-%i
    *0.72));
k=2.2;
Gz=(num/den);
disp(k*Gz,'Gz = ');
```

**Scilab code Exa 6.11** Inverse z Transform

```
//EXAMPLE 6.11
//Inverse Z-transform
clc;
clear;
syms n z1;
z  = %z;
num = z;      //given |z|>1;
den = (z-1)^2;


//Power series expansion
x=ldiv(num,den,20);
disp(x,'x = ');
disp('x = n*u[n]');
```

**Scilab code Exa 6.12** Inverse z Transform

```
//EXAMPLE 6.12
//Inverse Z-transform
```

```
clear;
clc;
z=%z;
num = 0.5*z;
den = z^2 -z + 0.25;
func = num/den;
v = factors(den);
disp(v,'factors are = ');
h1=ldiv(num,den,10);
disp(h1,'h  = ');
//using the property of z-trasnform
disp('observing v(1) & v(2) we conclude          h=n
   *(0.5)^n ')
```

**Scilab code Exa 6.13** Proper fraction of Rational z Transform

```
//EXAMPLE 6.13
//Determining proper fraction
clc;
clear;
z=%z;
num = 2*z^3 + 0.8*z^2 + 0.5*z +0.3;
den = (z^3 + 0.8*z^2 + 0.2*z);
func = num/den;
disp(func,'the polynomial function is H = ')

  if degree(num)>=degree(den)
           disp('An  improper fraction');
      else disp('A proper fraction');
  end

disp('decomposing the fraction we get .....');

H1=func-(-3.5*z + 1.5)/z;
disp(H1,'H1 = ');
```

```
disp('H1 is a Proper fraction')
```

**Scilab code Exa 6.14** Inverse z Transform by partial fraction expansion

```
//EXAMPLE 6.14
//Inverse Z-transform
clear;
clc;
z = %z;
num=z*(z+2);
den=(z-0.2)*(z+0.6);
H=num/den;
elts=factors(den);
disp(elts);
//solving Partial Fractions, we get:
Hz = 2.75/(1-(0.2)/z) - 1.75/(1+(0.6)/z);
disp(Hz);
//disp(h = 2.75*(0.2^n) - 1.75*(0.6^n)*u(n));
h1= ldiv(2.75*z,(z-(0.2)),10)
disp(h1/2.75,'h1 = ');
h1= ldiv(1.75*z,(z+(0.6)),10)
disp(h1/1.75,'h2 = ');
disp('the inverse z-transform is :')
disp('h = 2.75*(0.2^n)*u(n) - 1.75*(-0.6^n)*u(n)')
```

**Scilab code Exa 6.15** residue computation using coefficient matching approach

```
//EXAMPLE 6.15
//solving for coefficients;
clear;
clc;
z = %z;
```

```
num=z*(z+2);
den=(z-0.2)*(z+0.6);
H=num/den;
disp('the factors are :');
elts=factors(den);
disp(elts);
//coeff are:
disp('The coefficients are p1,p2:');
p1 = horner((z+2)/(z+0.6),0.2);
disp(p1,'p1 = ') ;
p2 = horner((z+2)/(z-0.2),-0.6);
disp(p2,'p2 = ') ;
```

**Scilab code Exa 6.16** Inverse z Transform by power series expansion

```
//EXAMPLE 6.16
//Partial fraction expansion

clc;
clear;
z=%z;
num = z^3;
den = 18*z^3 + 3*z^2 - 4*z - 1;
elts=factors(den);
disp(elts,'the factors are :') ;
func = num/den;
//the partial fraction gives:
p1 = horner((1/(1+0.3333333/z)^2),0.5);
disp(p1,'p1 = ');
p2 = horner(1/((1-0.5/z)),-0.3333333);
disp(p2,'p2 = ');
p3 = horner(0.6/((1-0.5/z)),-0.3333333);
disp(p3,'p3 = ');
disp('partial fraction gives : ');
disp(p1*z/elts(1),'h1 = ');
```

```
disp(p3*z/elts(3),'h2 = ');
disp(p2*z^2/(elts(2)*elts(2)),'h3 = ');
```

**Scilab code Exa 6.17** Coefficients of rational form

```
//EXAMPLE 6.16
//Coefficients of Rational form

clc;
clear;
z=%z;
num = 18*z^3;
den = 18*z^3 + 3*z^2 - 4*z - 1;
disp(coeff(num)/18,'the Numerator polynomial
    coefficients are:');
disp(coeff(den)/18,'the denominator polynomial
    coefficients are:');
```

**Scilab code Exa 6.18** Inverse z Transform using long division

```
//EXAMPLE 6.18
//Inverse Z-transform using power series expansion
clc;
clear;
z=%z;
Xnum=z;
Xden=(z-1)^2;
xn=ldiv(Xnum,Xden,15);
disp(xn,'The function is = ');
disp(' Thus, xn = n*u(n)');
```

**Scilab code Exa 6.19** Inverse z Transform using long division

```
//EXAMPLE 6.19
//Inverse Z-transform using Long division method
clc;
clear;
z=%z;
Hnum=z^2 + 2*z;
Hden=z^2 + 0.4*z -0.12;
hn=ldiv(Hnum,Hden,20);
disp(hn,'The function is , hn = ');
```

**Scilab code Exa 6.20** Inverse z Transform

```
//EXAMPLE 6.20
//Inverse Z-transform using power series expansion
clc;
clear;
z=%z;
Hnum=z^2 + 2*z;
Hden=z^2 + 0.4*z -0.12;
hn=ldiv(Hnum,Hden,20);
disp(hn,'The impulse response is , hn = ');
```

**Scilab code Exa 6.22** z Transform

```
   //Example 6.22
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform of r n.cos(wo n)
clc ;
clear;
syms r wo n z ;
x1 =(r^n)*exp(%i*wo*n) ;
```

```
X1 = nusum(x1*(z^-n),n,0,%inf);
x2 =(r^n)*exp(-%i*wo*n) ;
X2 = nusum(x2*(z^-n),n,0,%inf) ;
X =(X1+X2)/2 ;
disp(X,'X(z)=') ;
disp('ROC : |z|>r');
```

**Scilab code Exa 6.23** z Transform

```
//Example 6.23
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform of w(n) =((-0.5)^(n-2) +(0.2)^(n-1))*
    u(n)

syms n z ;
w1 = 4*(-0.5)^n
W1 = nusum(w1,n,0,%inf);
disp(W1,'ROC = |z|> 0.5 ,W1 = ');
w2 = 5*(0.2)^n
W2 = nusum(w2,n,0,%inf);
disp(W2,' ROC = |z|> 0.2 , W2 = ');
disp(W1+W2,'The Z-Transform is = ');
disp('ROC = |z|>0.5');
```

**Scilab code Exa 6.24** sum of sequences of non overlapping ROC

```
//Example 6.24
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS PROGRAM
//Z transform of v(n) =(a)^(n)*u(n) - (b)^(n)*u(-n
    -1)
clc;
clear;
disp('assuming |b| > a');
```

```
syms a n;
x1 = a^n;
X1 = nusum(x1,n,0,%inf);
x1 = b^n;
X1 = nusum(x2,n,-%inf,-1);
Vz = X1 + X2;
disp(Vz,'The Z−transform is = ');
disp('ROC = |a|<|z|<|b|');
```

**Scilab code Exa 6.25** z Transform

```
//Example 6.25
//Z transform of Vz , d0∗v[n] + d1∗v[n−1] = p0∗d[n]
    + p1∗d[n−1];
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS EXAMPLE
clc;
clear;
syms p0 p1 d0 d1;
z= %z;
disp('given that v[n] −−> V(z).Using Time shifting
    property,we get : ')
disp(' d0∗Vz + d1∗Vz∗(1/z) = p0 + p1∗(1/z) ');
disp('Rearranging the terms ...');
Vz = (p0 + p1/z)/(d0 + d1/z);
disp(Vz,'Z−transform is Vz =');
```

**Scilab code Exa 6.26** z Transform

```
    // Example 6.26
//MAXIMA SCILAB TOOLBOX
//Z transform of (n+1)∗a^n∗u(n)

clear ;
```

```
clc ;
syms a n z ;
x1 =(a)^n ;
X1 = symsum(x1*(z^(-n)),n,0,%inf);
X2 = -z*(diff (X,z,1)) ;
X = X1 + X2;
disp (X , 'Z transform of(n+1)*a^n*u(n) is X = ');
```

**Scilab code Exa 6.27** Inverse z Transform

```
// Example 6.27
//inverse Z-transform of z^3/(z-0.5)*(z+1/3)^2;

clear ;
clc ;
z=%z;
Gnum = z^3;
Gden = (z-0.5)*(z+1/3)^2;
G = Gnum/Gden;
g1=ldiv(Gnum,Gden,10);
elts=factors(Gden);

//the partial fraction gives:
p1 = horner((1/(1+0.3333333/z)^2),0.5);
disp(p1,'p1 = ');
p2 = horner(1/((1-0.5/z)),-0.3333333);
disp(p2,'p2 = ');
p3 = horner(0.6/((1-0.5/z)),-0.3333333);
disp(p3,'p3 = ')
disp('partial fraction gives : ');
disp(p1*z/elts(1),'h1 = ');
disp(p3*z/elts(3),'h2 = ');
disp(p2*z^2/(elts(2)*elts(2)),'h3 = ');
disp('gn = 0.36*(0.5)^n + 0.24*(-1/3)^n + 0.4*(n+1)
    *(-1/3)^n')
```

```
disp(g1,'the first 10 samples of g[n] = ');
```

**Scilab code Exa 6.28** Enlargement of ROC by pole zero cancellation

```
// Example 6.28
//Enlargement of ROC by pole-zero cancellation
clc;
clear;
z=%z;
Gz = (2 + 1.2*(1/z))/(1 - 0.2*(1/z))
disp(Gz,'Gz = ');
disp('ROC = |z|>0.2');
Hz = 3/(1 + 0.6*(1/z));
disp(Hz,'Hz = ');
disp('ROC = |z|>0.6');
Xz = Gz*Hz;
disp(Xz,'Xz = ');
disp('ROC = |z|>0.2');
```

**Scilab code Exa 6.30** Convolution

```
//EXAMPLE 6.30
//PROGRAM REQUIRES MAXIMA SCILAB TOOLBOX
//USE Z-TRANSFORM TO EVALUATE CONVOLUTION OF TWO
    SEQUENCES:
clc;
clear;
syms n z;
x = [-2  0 1 -1 3];
h = [1 2 0 -1 0];

for n=0:4
    X(n+1) = x(n+1)*z^(-n);
```

```
    H(n+1) = h(n+1)*z^(-n);
end
disp(X','X = ');
disp(H','H = ');

for i=1:5
    U(i)=0;
    for j=1:5
        U(i)=U(i)+X(i)*H(j);
    end
end
Y=0;
for i=1:5;
    Y = Y + U(i);
end
disp(Y,'Y = ');

disp('y = [-2 -4 1 3 1 5 1 -3]')
```

**Scilab code Exa 6.31** Convolution

```
//EXAMPLE 6.31
//PROGRAM REQUIRES MAXIMA SCILAB TOOLBOX
//USE Z-TRANSFORM TO EVALUATE CONVOLUTION OF TWO
    SEQUENCES:
clc;
clear;
syms n z;
x = [3 -2 4];
h = [4 2 -1];

for n=-1:1
    X(n+2) = x(n+2)*(z^-n);
end
disp(X','X = ');
```

```

for n=0:2
    H(n+1) = h(n+1)*(z^-n);
end
disp(H','H = ');


for i=1:3
    U(i)=0;
    for j=1:3
        U(i)=U(i)+X(i)*H(j);
    end
end
Y=0;
for i=1:3;
    Y = Y + U(i);
end
disp(Y,'Y = ');

disp('y = [12 -2 9 10 -4]');
```

**Scilab code Exa 6.33** Transfer Function of Moving Average Filter

```
//EXAMPLE 6.33
//Transfer function of moving average filter
clear;
clc;
syms n z M;
x=z^(-n);
H1=nusum(x,n,0,M-1);
H=H1/M;
disp(H,'Transfer function , Hz = ');
```

**Scilab code Exa 6.34** Transfer function determination

```
//EXAMPLE 6.34
//y[n]=x[n-1] - 1.2*x[n-2] + x[n-3] + 1.3*y[n-1]
    -1.04*y[n-2] + 0.222*y[n-3]
//Transfer function determination

clc;
clear;
z=%z;
disp('Given the difference equation taking
    ztransform on both sides :')
Yz = z^2 -1.2*z +1;
Xz = z^3 -1.3*z^2 + 1.04*z -0.222;
Hz = Yz/Xz;
disp(Hz,'The transfer function is = ')
elts = factors(Xz);
disp(elts,'factors of Xz are = ')
plzr(Hz);
```

# Chapter 7

# LTI Discreet Time systems in the Transform Domain

**Scilab code Exa 7.1** Bounded real function

```
//EXAMPLE 7.1
//PROGRAM REQUIRES MAXIMA SCILAB TOOLBOX

clc;
clear;
syms K a z w;

hzden = (1-a*(z^-1)); //0<|a|<1;
Hz = K/hzden;
disp('|H(e^(jw))|^2 = K^2/((1+a)^2 - 2*cos(w)') ;
// considering a>0
disp('(at w = %pi),K^2/(1+a)^2 < |H|^2  < K^2/(1-a)^2,(at w = 0)');
//considering a<0
disp(' (at w = 0),K^2/(1+a)^2 < |H|^2  < K^2/(1-a)^2,(at w = %pi)');
disp('if K = +/-(1-a), observe....');
disp('|H(e^(jw))|  <= 1      Hence a Bounded real function.');
```

```
//w=0:%pi;
//[a,b]=freq(hznum,hzden,w);
disp(abs(Hz))
```

**Scilab code Exa 7.2** Transfer function determination

```
//EXAMPLE 7.2
//|H(e^(jw))|^2 = 4*((1.09 + 0.6*cosw)*(1.16-0.8*
    cosw))/((1.04-0.2*cosw)*(1.25+cosw))
//REPLACING cosw = (z + z(^-1))/2
clc;
clear;
z=%z;
H1=4*((1.09 + (0.3)*(z+1/z))*(1.16 - (0.4)*(z+1/z)))
    ;
H2=((1.04 - (0.2)*(z+1/z))*(1.25 + (0.5)*(z+1/z)));
H=H1/H2;
disp(H,'The transfer function is, H = ')
elts1=factors(numer(H));
disp(elts1,'The factors of numerator are :');
elts2=factors(denom(H));
disp(elts2,'The factors of denominator are :');
disp('The Four posible stable transfer function with
     same square magnitude function are :');
h1=2*((1+(0.3)/z)*(1- (0.4)/z))/((1-(0.2)/z)
    *(1+(0.5)/z));
disp(h1,'stable transfer function,h1 = ');
h2=2*((1+(0.3)/z)*((0.4)- (1)/z))/((1-(0.2)/z)
    *(1+(0.5)/z));
disp(h2,'stable transfer function,h2s = ');
h3=2*(((0.3)+1/z)*((1)- (0.4)/z))/((1-(0.2)/z)
    *(1+(0.5)/z));
disp(h3,'stable transfer function,h3 = ');
h4=2*(((0.3)+1/z)*((0.4)- (1)/z))/((1-(0.2)/z)
    *(1+(0.5)/z));
```

```
disp(h4,'stable transfer function,h4 = ');
```

**Scilab code Exa 7.6** FIR Transfer function

```
//EXAMPLE 7.6
//FIR Trasnfer functions with different Phase.
clc;
clear;
z  = %z;
W  = 0:(1/400):1;
z  = exp(%i*2*%pi*W);
for i=1:401
    H1z(i)= -1+ 2/z(i) - 3/(z(i)^2) + 6/(z(i)^3)
       -3/(z(i)^4) +2/(z(i)^5) -1/z(i)^(6);
end
H1z_phase = phasemag(H1z);

clf();
figure(0);
plot2d(W/(2*%pi),H1z_phase,1);
xtitle('phase response','W/(2*%pi)','H2z_phase in
   degrees');

for i=1:401
    H2z(i)= +1 - 2/z(i) + 3/(z(i)^2) - 6/(z(i)^3) +
       3/(z(i)^4) - 2/(z(i)^5) + 1/z(i)^(6);
end
H2z_phase = phasemag(H2z);

plot2d(W/(2*%pi),H2z_phase,2);
xtitle('phase response','W/(2*%pi)','H2z_phase in
   degrees');
```

# Chapter 8

# Digital Filter Structures

**Scilab code Exa 8.1** Analysis of Cascaded lattice digital filter structure

```
//EXAMPLE 8.1
//MAXIMA SCILAB TOOLBOX REQUIRED FOR THIS EXAMPLE
//Digital filter structure
clear;
clc;
syms W1 W2 W3 X Y a d B y E z;
//Equations obtained are as follows:
W1 = X - a*W3/z;
W2 = W1 - d*W2/z;
W3 = W2/z + E*W2;
Y  = B*W1 +y*W3/z;
//Solving the above equations:
Hz=(B + (B*d+y*E)/z + y/(z^2))/(1 + (d+a*E)/z + a/(z
   ^2))
disp(Hz,'Hz = ');
```

**Scilab code Exa 8.6** Factorization of FIR Transfer Function

```
//Example 8.6
//Factorization of FIR Transfer Function
clear;
clc;
z=%z;
Hz=50.4+28.02/z+13.89/z^2+7.42/z^3+6.09/z^4+3/z^5+1/
   z^6;
disp(factors(numer(Hz)),'The Factors of the FIR
   Transfer Function are = ');
```

**Scilab code Exa 8.7** Factorization of IIR Transfer Function

```
//Example 8.7
//Factorization of IIR Transfer Function
clear;
clc;
z=%z;
//Numerator of the transfer function
Numz=6+17.1/z+33.05/z^2+24.72/z^3+19.908/z^4-5.292/z
   ^5+18.144/z^6;
//Denominator of the transfer function
Denz=1+2.2/z+2.56/z^2+1.372/z^3+0.118/z^4-0.332/z
   ^5-0.168/z^6;
Fn=factors(numer(Numz));
disp(Fn,'Factors of the numerator of the Transfer
   Function = ');
Fd=factors(numer(Denz));
disp(Fd,'Factors of the denominator of the Transfer
   Function = ');
```

**Scilab code Exa 8.10** Cascaded lattice realization of IIR digital Transfer Function

```
//Example 8.10
//Cascaded lattice realization of IIR Transfer
  Function
clear;
clc;
z=%z;
P3z= -0.2 + 0.18/z + 0.4/(z^2) + 1/(z^3);
D3z=  1 + 0.4/z + 0.18/(z^2) - 0.2/(z^3);
A3z=P3z/D3z;
p1=coeff(numer(P3z));
p=mtlb_fliplr(p1);
disp(mtlb_fliplr(p),'The coefficients ofnumerator
  are = ');
d1=coeff(numer(D3z-1));
d=mtlb_fliplr(d1)
disp((d),'The coefficients ofnumerator are = ');
d1_1dash=(d(1)-d(3)*d(2))/(1-d(3)*d(3));
disp(d1_1dash,"d1_1dash = ");
d2_1dash=(d(2)-d(3)*d(1))/(1-d(3)*d(3));
disp(d2_1dash,"d2_1dash  ");
d1_2dash=(d1_1dash)/(1+d2_1dash);
disp(d1_2dash,"d1_2dash = ");
A1z=(d1_2dash + 1/z)/(1 + d1_2dash/z);
disp(A1z,'A1z = ');
A2z=(d2_1dash + d1_1dash*1/z + 1/z^2)/(1 + d1_1dash/
  z - d2_1dash/z^2);
disp(A2z,'A2z = ');
```

**Scilab code Exa 8.12** Gray Markel method of realization

```
//Example 8.12
//Gray Markel method of Realisation
clear;
clc;
z=%z;
```

```
P3z= 0 + 0.44/z + 0.362/(z^2) +0.02/(z^3);
D3z= 0.4/z + 0.18/(z^2) - .2/(z^3);
Hz=P3z/D3z;
p1=coeff(numer(P3z));
p=mtlb_fliplr(p1)
disp(mtlb_fliplr(p),'The coefficients ofnumerator are = ');
d1=coeff(numer(D3z));
d=mtlb_fliplr(d1)
disp(mtlb_fliplr(d),'The coefficients ofnumerator are = ');
d1_1dash=(d(1)-d(3)*d(2))/(1-d(3)*d(3));
disp(d1_1dash,"d1_1dash = ");
d2_1dash=(d(2)-d(3)*d(1))/(1-d(3)*d(3));
disp(d2_1dash,"d2_1dash  ");
d1_2dash=(d1_1dash)/(1+d2_1dash);
disp(d1_2dash,"d1_2dash = ");
a1=p(3);
disp(p(3),'a1 = ');
a2=p(2)-a1*d(1);
disp(p(2)-a1*d(1),'a2 = ');
a3=p(1)-a1*d(2)-a2*d1_1dash;
disp(p(1)-a1*d(2)-a2*d1_1dash,'a3 = ');
disp(0-a1*d(3)-a3*d1_2dash-a2*d2_1dash,'a4 = ');
```

**Scilab code Exa 8.18** Cascaded lattice realization

```
//Example 8.18
//Cascaded lattice realization of Power-symmetric FIR Transfer Function
clear;
clc;
z=%z;

H5z=(1 + 0.3/z + 0.2/z^2 - 0.376/z^3 - 0.06/z^4 +
```

```
    0.2/z^5);
disp(H5z,'FIR filter = ');
G5=horner(H5z,-1/z);
G5z=G5/z^5;
disp(G5z,'FIR filter = ');
k5=0.2;
H3z=(1/(1+k5^2))*(H5z - k5*G5z);
disp(H3z,'Synthesis eqn,H3z = ');
G3z=(1/(1+k5^2))*(k5*H5z + G5z);
disp(G3z,'Synthesis eqn,G3z = ');
k=coeff(numer(G3z));
disp(k(4),'k3 = ');
disp(k(2),'k1 = ');
```

# Chapter 9

# IIR digital filter design

**Scilab code Exa 9.1** Computating ripple values

```
//EXAMPLE 9.1
//pass band & stop band ripple
clc;
clear;
ap=0.1;//peak passband ripple in dB
as=35;//min. stopband attenuation in dB

//calculation of peak ripple values
dp=1-10^-(ap/20);
disp(dp,'dp = ');
ds=10^-(as/20);
disp(ds,'ds = ');
```

---

**Scilab code Exa 9.2** conversion of bandedged frequencies to Normalized digital frequencies

```
//EXAMPLE 9.2
//analog passband & stopband frequencies(in KHz) :
```

```
clc;
clear;
ap=7;
as=3;
//Sampling frequency (in KHz):
FT=25;
//digital frequencies:
wp=2*%pi*ap/FT;
disp(wp,'wp = ');
ws=2*%pi*as/FT;
disp(ws,'ws = ');
```

**Scilab code Exa 9.3** Design of HP Digital Filter

```
//Example 9.3
//Design of HP IIR filter
clc;
clear;
Fp=700//Hz
Fs=500//Hz
ap=1//dB
as=32//dB
FT=2000//Hz
//normalized angular edge frequencies in rad/sec
wp=2*%pi*Fp/FT;
ws=2*%pi*Fs/FT;
//prewarp the digital edge frequencies
Ap1=tan(wp/2);
As1=tan(ws/2);

Ap=1;//assuming
As=(2*%pi*Ap1)*Ap/(2*%pi*As1);
disp(As,'As = ')
//Order 'N' of the filter
k  = Ap/As;
```

```
disp(1/k,'1/k = ');
k1 = 1/(sqrt(((10^(0.1* as))-1)/((10^(0.1*ap))-1)));
disp(1/k1,'1/k1 = ');
N = ceil(acosh(1/k1)/acosh(1/k));
disp(N,'N = ');
disp(N,'Order of the filter is, N = ');

e=sqrt(10^(0.1*ap)-1);
u=1/e + sqrt(1+(1/(e*e)));
a=Ap*(u^(1/N) - u^(-1/N))/2;
b=Ap*(u^(1/N) + u^(-1/N))/2;

for i=1:N
    phi(i)= %pi/2 + (2*i -1)*(%pi)/(2*N);
    p(i)=a*cos(phi(i)) + %i*b*sin(phi(i));
end
s=%s;
z=%z;
H1=1;
//Numerator of H(s)
for i=1:N
    H1=H1*(s + p(i))
end
//Denominator of H(s)
H2=horner(H1,0);
//Transfer function
H=H2/H1;
disp(H,'H = ')
//Bilnear Transformaation, s=((z-1/(z+1));
Hz=horner(H,(z-1)/(z+1));
disp(Hz,'The digital HP filter is Hz = ');
```

**Scilab code Exa 9.6** Changing passband edge frequencies to LP IIR digital frequencies

```
//EXAMPLE 9.6
//LP TO LP Transformation

clc;
clear;
z=%z;
w=0:0.001*%pi:%pi;
Glz=(0.0662272*(1+1/z)^3)/((1-0.2593284/z)
   *(1-0.6762858/z+0.3917468/(z^2)))
wc=0.25*%pi;//Oringinal passband edge
Wc=0.35*%pi;//Required passband edge
l=sin((wc-Wc)/2)/sin((wc+Wc)/2)
disp(l,'lambda = ');
Gdz=horner(Glz,((1-l/z)/(1/z-l)));
disp(Gdz,'The transfer function is Gdz = ');
```

**Scilab code Exa 9.7** Design of HP IIR Digital Filter from LP Digital Filter

```
//EXAMPLE 9.7
//LP TO HP Transformation

clc;
clear;
z=%z;
Glz=(0.0662272*(1+1/z)^3)/((1-0.2593284/z)
   *(1-0.6762858/z+0.3917468/(z^2)));
wc=0.25*%pi;//Oringinal passband edge
Wc=0.55*%pi;//Required passband edge
l=-cos((wc+Wc)/2)/cos((wc-Wc)/2);
disp(l,'lambda = ');

w=0:0.001:1;
Ghz=horner(Glz,-((z + l)/(1 +l*z)));//LP TO HP
   Transformation formula
den=factors(denom(Ghz));
```

```
disp(Ghz,'The transfer function is Gdz = ');
disp(den,'the facors of the denominator are = ');
```

**Scilab code Exa 9.12** Minimum order of Type 2 Chebyshev HP IIR digital filter

```
//EXAMPLE 9.12
//Minimum order of type-2 Chebyshev highpass digital
    filter
clc;
clear;
ap = 1 //dB
as = 40 //dB
Fp = 1000 //Hz
Fs = 600 //Hz
Wp = Fp*2*%pi;
Ws = Fs*2*%pi;

F = 4000//Hz
T=1/F;

Ap=(2/T)*(tan(Wp*T/2))
As=(2/T)*(tan(Ws*T/2))


k  = Ap/As;
disp(1/k,'1/k = ');
k1 = 1/(sqrt(((10^(0.1* as))-1)/((10^(0.1*ap))-1)));
disp(1/k1,'1/k1 = ');
N = acosh(1/k1)/acosh(k);//order of the filter
disp(N,'N = ');
disp('Since order of the filter is always an integer
    , ');
disp(ceil(N),'Order of the filter is , N = ');
```

# Chapter 10

# FIR digital filter design

**Scilab code Exa 10.1** Kaiser formula

```
//Example 10.01
//Order estimation using Kaiser's formula
clear;
clc;
Fp=1800;//Passband edge freq. in Hz
Fs=2000;//stopband edge freq. in Hz
ap=0.1;//peak passband ripple in dB
as=35;//min. stopband attenuation in dB
FT=12000;//Sampling freq. in Hz

//calculation of peak ripple values
dp=1-10^-(ap/20);
disp(dp,'dp = ');
ds=10^-(as/20);
disp(ds,'ds = ');

//Order of the FIR filter
N=(-(20*log10(sqrt(ds*dp)) - 13)/((14.6)*(Fs-Fp)/FT
    );
disp(ceil(N),'Order of the filter is N = ')
```

**Scilab code Exa 10.2** Bellenger formula

```
//Example 10.01
// Order estimation using Bellanger's formula
clear;
clc;
Fp=1800;//Passband edge freq. in Hz
Fs=2000;//stopband edge freq. in Hz
ap=0.1;//peak passband ripple in dB
as=35;//min. stopband attenuation in dB
FT=12000;//Sampling freq. in Hz

//calculation of peak ripple values
dp=1-10^-(ap/20);
disp(dp,'dp = ');
ds=10^-(as/20);
disp(ds,'ds = ');

//Order of the FIR filter
N=((-2*log10(10*ds*dp)) /((3)*(Fs-Fp)/FT)) -1 ;
disp(ceil(N),'Order of the filter is N = ')
```

**Scilab code Exa 10.3** Hermann formula

```
//Example 10.03
//Order estimation using Hermann's formula
clear;
clc;
Fp=1800;//Passband edge freq. in Hz
Fs=2000;//stopband edge freq. in Hz
ap=0.1;//peak passband ripple in dB
as=35;//min. stopband attenuation in dB
```

```
FT=12000;//Sampling freq. in Hz

//calculation of peak ripple values
dp=1-10^-(ap/20);
disp(dp,'dp = ');
ds=10^-(as/20);
disp(ds,'ds = ');

a1=0.005309;
a2=0.07114;
a3=-0.4761;
a4=0.00266;
a5=0.5941;
a6=0.4278;
D_infi=((a1*(log10(dp)^2) + a2*log10(dp) + a3)*log10
   (ds))-(a4*(log10(dp))^2 + a5*(log10(dp)) + a6);
disp(D_infi,'D_infi = ');
b1=11.01217;
b2=0.51244;
F=b1 + b2*((log10(dp))-(log10(ds)));
disp(F,'F = ');

//Order of the FIR filter
N=(D_infi- F*((Fs-Fp)/FT)^2)/((Fs-Fp)/FT);
disp(ceil(N),'Order of the filter is N = ')
```

**Scilab code Exa 10.4** Order Estimation

```
//Example 10.04
//Kaiser's formula for bandpass filter
clear;
clc;
Fp1=300;//Passband edge freq. in Hz
Fs1=350;//stopband edge freq. in Hz
Fp2=1000;//Passband edge freq. in Hz
```

```
Fs2=1100;//stopband edge freq. in Hz

dp=0.004;//passband ripple in dB
ds=0.01;//stopband ripple in dB
FT=10000;//Sampling freq. in Hz

//Since (Fp1-Fs1)<(Fs2-Fp2),bandwith used is (Fp1-
   Fs1)

//Order of the FIR filter
N=(-(20*log10(sqrt(ds*dp)) - 13)/((14.6)*(Fs1-Fp1)/
   FT);
disp(ceil(N),'Order of the filter is N = ')
```

**Scilab code Exa 10.6** Filter length estimation for window based design

```
//EXAMPLE 10.6
//FIlter length  for window -based design
clear;
clc;
wp=0.3*%pi;//rad/sec
ws=0.5*%pi;//rad/sec
as=40;//dB

wc=(wp+ws)/2;//cutoff frequency
Bw=ws-wp;
disp(Bw,'Normalized transition bandwidth is = ')
//Hann window
M1=3.11*%pi/Bw;
disp(M1,'Value of M = ')
//Hamming window
M2=3.32*%pi/Bw;
disp(M2,'Value of M = ')
//Blackman window
M3=5.56*%pi/Bw;
```

```
disp(M3,'Value of M = ')
```

**Scilab code Exa 10.7** Order Estimation

```
//EXAMPLE 10.6
//Order estimation using Dolph−Cebyshev window
clear;
clc;
wp=0.3*%pi;//rad/sec
ws=0.5*%pi;//rad/sec
as=40;//dB

wc=(wp+ws)/2;//cutoff frequency
Bw=ws-wp;
disp(Bw,'Normalized transition bandwidth is = ')

//Order of the filter
N = ((2.056*as) - 16.4)/(2.285*Bw);
disp(ceil(N),'Order of the filter ,N =  ')
```

**Scilab code Exa 10.8** Kaiser window

```
//EXAMPLE 10.8
//Design of LP FIR filter using Kaiser window
clear;
clc;
wp=0.3*%pi;//rad/sec
ws=0.5*%pi;//rad/sec
as=40;//dB
wc=(wp+ws)/2;//cutoff frequency
Bw=ws-wp;
disp(Bw,'Normalized transition bandwidth is = ')

```

```
ds=10^(-as/20);
B = (0.5842*(as-21)^0.4) + 0.07886*(as-21);

N = ceil((as - 8)/(2.285*Bw));
disp(N,'Order of the filter ,N =  ');
M=(N-1)*0.5;
disp(M,'M = ');
w=window('kr',N,6);//Kaiser window
i=-M:1:M;

    hn=(wc/%pi)*sinc(wc*i'/(%pi));
    h=hn*w;

clf();
n=0:0.001:1;
[H,fr]=frmag(w,1001);

plot2d(2*fr,log10(H./max(H)),style=color('blue'))
set(gca(),'grid',[1 1]*color('gray'))
a = gca () ;
xlabel ( 'w/%pi' ) ;
ylabel ('Magnitude in dB') ;
title ( 'Gain Response of Kaiser Window' );
```

# Chapter 11

# DSP Algorithm implementation

**Scilab code Exa 11.3** Reconstruction of Transfer function from Impulse response coeeficients

```
//example 11.3
//Reconstruction of Transfer function from Impulse
   response coeff.
clear;
clc;
z=%z;
numz=2+6/z+3/(z^2);
denz=(1+1/z+2/z^2);
disp(numz/denz,'Hz = ');
d=coeff(numer(denz));
disp(d,'coefficients of the denominator are = ')
h1=ldiv(numer(numz),numer(denz),5);
disp(h1','The first five coeffcients are of H(z) = '
   );
for i=1:3
    for j=1:3
        if i>=j
            h(i,j)=h1(i-j+1)
        else
            h(i,j)=0;
```

```
            end
        end
end
disp(h,'h = ');
disp((h'*d')','coefficients of the numerator are = ');
```

**Scilab code Exa 11.11** Cascaded lattice Filter structure

```
//Example 11.11
//Simulation of IIR cascaded lattice filter structure
clear;
clc;
z=%z;
P3z= 0 + 0.44/z + 0.362/(z^2) +0.02/(z^3);
D3z= 1 + 0.4/z + 0.18/(z^2) - 0.2/(z^3);
Hz=P3z/D3z;
p1=coeff(numer(P3z));
p=mtlb_fliplr(p1)
disp(mtlb_fliplr(p),'The coefficients ofnumerator are = ');
d1=coeff(numer(D3z-1));
d=mtlb_fliplr(d1)
disp(mtlb_fliplr(d),'The coefficients ofnumerator are = ');
d1_1dash=(d(1)-d(3)*d(2))/(1-d(3)*d(3));
disp(d1_1dash,"d1_1dash = ");
d2_1dash=(d(2)-d(3)*d(1))/(1-d(3)*d(3));
disp(d2_1dash,"d2_1dash  ");
d1_2dash=(d1_1dash)/(1+d2_1dash);
disp(d1_2dash,"d1_2dash = ");
a1=p(3);
disp(p(3),'a1 = ');
a2=p(2)-a1*d(1);
```

```
disp(p(2)-a1*d(1),'a2 = ');
a3=p(1)-a1*d(2)-a2*d1_1dash;
disp(p(1)-a1*d(2)-a2*d1_1dash,'a3 = ');
disp(0-a1*d(3)-a3*d1_2dash-a2*d2_1dash,'a4 = ');
```

# Chapter 12

# Analysis of Finite Wordlength Effects

**Scilab code Exa 12.3** Signal to Quantisation Noise Ratio

```
//Example 12.3
//Signal-to-quantization Noise ratio
clear;
clc;
b=[7 9 11 13 15];//Given values of b
K=[4 6 8];//Given values of K
for i=1:5
    for j=1:3
        SNR(j,i)=6.02*b(i)+16.81-20*log10(K(j));
    end
end
disp(SNR,'SNR,A/D = ');
```

# Chapter 13

# Multirate DIgital Signal Processing Findamentals

**Scilab code Exa 13.1** Up sampling operation

```
//Example 13.1
//Upsampling Operation
clear;
clc;
clf();
a=gca();
figure(0);
n=[0:0.1:4.9];
a.x_location="origin";
x=sin(%pi*n);
plot2d3(n,x,2);
xtitle('The sine wave','n','sin(x)');
plot(n,x,'r.');
//Up sampling
//Up sampling value - user input
figure(1);
L=input(" The up sampling value  ");

a.x_location="origin";
```

```
x1=sin(%pi*n/L);
plot2d3(n,x1,5);
plot(n,x1,'r.');
xtitle('The sine wave','n','sin(x/L)');
```

**Scilab code Exa 13.2** Down sampling operation

```
//Example 13.2
//Downsampling Operation
clear;
clc;
clf();
a=gca();
figure(0);
n=[0:0.1:4.9];
a.x_location="origin";
x=sin(%pi*n);
plot2d3(n,x,2);
xtitle('The sine wave','n','sin(x)');
plot(n,x,'r.');
//Down sampling
//Down sampling - user input
figure(1);
M=input(" The down sampling factor  ");
a.x_location="origin";
x1=sin(%pi*n*M);
plot2d3(n,x1,1);
plot(n,x1,'r.');
xtitle('The sine wave','n','sin(x*M)');
```

**Scilab code Exa 13.6** Decimator Computation complexity

```
//Example 13.6
```

```
//Decimator computational complexity
clear;
clc;
//no. of multiplications/sec =Rm
FT = input("Sampling Frequency");
N  = input("The order of the FIR Hz");
Rm1 = N*FT;
disp(Rm1,'Rm,FIR = ');
//M = factor of Down sampler
M = input("The Down Sampling factor ");
disp(Rm1/M,'Rm,FIR-DEC = ');
K  = input("The order of the IIR Hz");
Rm2 = (2*K + 1)*FT;
disp(Rm2,'Rm,IIR = ');
disp( (K*FT + ((K+1)*FT/M)),'Rm,IIR-DEC = ');
```

# Chapter 14

# Applications of Digital Signal Processing

**Scilab code Exa 14.1** Effect of DFT length

```
//Example 14.1
//EFFECT OF DFT LENGTH ON SPECTRAL ANALYSIS
clear;
clc;
N=16;
n=0:N-1;
f1=0.22;
f2=0.34;

R = input("R point DFT(R E [16,128]) = ");          //Input f1 = 64
if R >= N
    x=0.5*(sin(2*%pi*f1*n')) + sin(sin(2*%pi*f2*n'));
    x=[x',zeros(1,R-length(n))];
    disp(x,'the sequence is :');
    for n=0:R-1
        for k=0:R-1
            W(n+1,k+1) = exp(-(%i*2*%pi*k/R)*n);
```

```
            end
        end
        X = W*x';
        disp(X,'DFT is , X = ')
else
        disp('invalid computation');
end
m=0:R-1;
clf();
figure(0)
a = gca();
plot2d3(m,abs(X),2) //  plotting DFT of sequence
plot(m,abs(X),'r.')
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 2.5;
xtitle('original sequence','n','x[n]');
```

**Scilab code Exa 14.2** Effect of DFT length

```
//Example 14.2
//EFFECT OF DFT LENGTH ON SPECTRAL ANALYSIS
clear;
clc;
N=16;
n=0:N-1;
f1=input("Enter f1 value between 0.28 to 0.31 = ");
        //Input f1 = 64
f2=0.34;

R = 128//
//DFT of the sequence x[n]
    x=0.5*(sin(2*%pi*f1*n')) + sin((2*%pi*f2*n'));
    x=[x',zeros(1,R-length(n))];
```

```
    disp(x,'the sequence is :');
    for n=0:R-1
        for k=0:R-1
            W(n+1,k+1) = exp(-(%i*2*%pi*k/R)*n);
        end
    end
    X = W*x';
    disp(X,'DFT is , X = ')
//plotting DFT of sequence
m=0:R-1;
clf();
figure(0)
a = gca();
plot2d3(m,abs(X),2)
plot(m,abs(X),'r.')
a.x_location = 'origin';
a.y_location = 'origin';
poly1 = a . children (1) . children (1) ;
poly1.thickness = 2.5;
xtitle('original sequence','n','x[n]');
```