Contents

Introduction to SCILAB

Scilab is a free and open-source, cross-platform numerical computational package and a high-level, numerically oriented programming language. It can be used for signal processing, statistical analysis, image enhancement, fluid dynamics simulations, numerical optimization and modeling, simulation of explicit and implicit dynamical systems and symbolic manipulations.

Generation of signals

Important functions : plot(), plot2d(), plot2d3(),ones(), zeros(), figure(), xtitle(), subplot(), xlabel(), ylabel(),........

Scilab 5.5.2 , OS: Ubuntu 14.04

i) Unit Sample Sequence

```
clear ;clc ;close ;
L = 4;                        // U p p e r l i m i t
n = -L:L;                     // index of the sequence
x = [zeros(1,L),1,zeros(1,L)];
figure (1);
plot2d3(n,x);                 //For discrete time signal
xtitle(' Unit Sample Sequence','n','x1[n]') ;
```

ii) Unit step function

```
clear ;clc ;close ;
n=0:5
x=[ones(1,6)];
figure(1); plot2d3(n,x);
xtitle('Discrete Unit Step Sequence','n','x[n]') ;
```

iii) Unit ramp function

```
clear ;clc ;close ;
L = 4;
n = -L : L;
x = [zeros(1,L ),0:L ];
```

```
plot2d3(n,x);
xtitle(' Discrete Unit Ramp Sequence','n','x[n]') ;
```

iv) Discrete time Exponential signal

```
clear ;clc ;close ;
a =0.5;   //For decreasing a<1 and For increasing exponential a>1
n = 0:10;
x = (a).^n ;
plot2d3(n,x);xtitle('Exponentially Decreasing Signal ','n','x[n]');
```

v) Complex valued signals

```
clc;clear;
n= [-10:1:10];
a= -0.1+0.3*%i;
x=exp(a*n);
subplot(221), plot2d3(n,real(x));xtitle('Real part','n');
subplot(223), plot2d3(n,imag(x));xtitle('Imaginary','n');
subplot(222), plot2d3(n,abs(x));xtitle('Magnitude part','n');
theta=(180/%pi)*atan(imag(x),real(x));
subplot(224), plot2d3(n,theta);xtitle('Phase part','n');
```

vi) Sinusoidal signal

```
clc;clear;
fm=input('Enter the input signal frequency:');
k=input('Enter the number of Cycles of input signal:');
A=input('Enter the amplitude of input signal:');
tm=0:1/(fm*fm):k/fm;
x=A*cos(2*%pi*fm*tm);
figure(1);plot2d3(tm,x);
title('Graphical Representation of Sinusoidal Signal');
xlabel('Time');ylabel('Amplitude');
xgrid(1)
```

vii) Square wave

```
clc;clear;
t=(0:0.1:4*%pi)';
plot2d3(4*%pi*squarewave(t));
xtitle('square wave','time','amplitude');
```

viii) Triangular wave

```
clear;clc;
A=input('enter the amplitude:');
K=input('enter number of cycles:');
x1 = [0:A  A-1:-1:1];
x=x1;
for i=1:K-1
x=[x x1];
end
n=0:length(x)-1;     // Index of the sequence
plot2d3(n,x);xtitle('Triangular wave','time','amplitude');
```

ix) Sawtooth wave

```
clc;clear;
A=input('enter the amplitude:');
K=input('enter number of cycles:');
x1 = [0:A];
x=x1;
for i=1:K-1
   x=[x x1];
end
n=0:length(x)-1;
plot2d3(n,x);xtitle('Sawtooth wave','time','amplitude');
```

x) Writing your own function

Syntax: function [output variables] = function_name(input variables);

```
clear all;clc
function [Ex] = energy(x)
Ex= sum(x.*conj(x));
end function
x=[1 2 3 4 5];
n=0:length(x);
E=energy(x)
```

Problems:

1. Decompose a real sequence x[n] into even and odd components.

1. Compute the Frequency response of a system characterized by the impulse response h[n] and plot its magnitude and phase spectrum. Verify the result using SCILAB.

Design steps:

1. Use $X(e^{j\omega}) = \sum x[n]\, e^{-j\omega n}$ or the matrix notation X =Wx
   where $W = [e^{(-j2\pi k n/M)}]$, k=0,1,...M -1 and n=0,1,..N-1
   and $x= [x[0]\ x[1]...]^T$ to compute DTFT
2. Plot the magnitude and phase response
3. Ensure that the frequency axis is from [0, π] and verify the periodicity and conjugate symmetry property.

Design example: Plot the frequency response of the sequence $h[n] = (0.9)^n$, $0 \le n \le 9$.
Using matrix notation $X^T= x^T [\ e^{(-j2\pi\, nTk/M)]}$

```
clear;clc;close;
h=input('enter the input sequence');      //h=[1 1 1 1 1]
n=0:length(h)-1;
w=-%pi:%pi/4:%pi;                         //To evaluate X(ejw) at equi-spaced frequencies
wn= n'*w;
minus_jwn= -%i*n'*w
H=h*exp(minus_jwn);
magH=abs(H);
angH=atan(imag(H),real(H));
figure();
subplot(311),plot2d3(n,h);
xtitle('Impulse Response','time index n');
subplot(312),plot2d(w,magH);
xtitle('Magnitude Response', 'frequency in rad', 'Magnitude');
subplot(313),plot2d(w,angH);
xtitle('Phase Response', 'frequency in rad', 'Phase');

//Expected Result
//H= [1, 0.4142136i, 1, 2.4142136i, 5,- 2.4142136i, 1, - 0.4142136i,1]
```

Problems:

1. $x[n] = \begin{cases} 2 - (\frac{1}{2})^n & ;|n| \le 4 \\ 0 & ;otherwise \end{cases}$

2. x[n] = $(0.5)^n$ {u(n+3)-u(n-2)}

3. x[n]= $(0.25)^n$ {u(n+4)-u(n-4)}

2. Determine and Plot the frequency response of a discrete-time system represented by difference equation y[n]=b0x[n]+b1x[n-1]+a1y[n-1] +a2y[n-2]. Verify the result using SCILAB.

Design steps: Given $a_0\, y[n] = -a_2\, y[n-2] - a_1\, y[n-1] + b_0\, x[n] + b_1\, x[n-1] + b_2\, x[n-2]$

1. System function $H(z) = b_0 + b_1\, z^{-1} + b_2\, z^{-2} / 1 + a_1\, z^{-1} + a_2\, z^{-2}$
2. Put $z= e^{(jw)}$ to get the frequency response

Design example: Plot the magnitude and phase response of the system represented by 6y[n]+5y[n-1]+y[n-2]=18x[n] + 8x[n-1]

```
clear;clc;close;
b=[18, 8];
a=[6 5 1];
m= 0: length(b)-1; p=0:length(a)-1;
w=-2*%pi:%pi/100:2*%pi;
num = b* exp(-%i*m'*w);
den = a*exp(-%i*p'*w);
H= num./den;
magH = abs(H); angH= atan(imag(H),real(H));
figure;
subplot(211), plot( w, magH);
xtitle('Magnitude response','Frequency in rad','Magnitude');
subplot(212),plot(w, angH);
xtitle('Phase Response','Frequency in rad','Phase');
```

Problems:

1. y[n]= 0.5 y[n-1] – 0.06 y[n-2] + 0.2 x[n] -0.5 x[n-1]
2. y[n] = x[n] – x[n-1] -0.5 y[n-1]
3. y[n] = x[n] -x[n-1] + x[n-2] +0.95 y[n-1] -0.9025 y[n-2]
4. y[n] = x[n-1] + (1/9) y[n-2] with y[-1] = 1, y[-2]=0 & x[n]=u[n]

3. For a given signal x[n]= A*cos(2*pi*fm*t), using SCILAB

i) Plot x(n) and its spectrum for fs=fm, 2fm & 4fm. and N= ….. .

ii) Demonstrate the effect of zero padding given x(n)=…... 0≤n≤3.

iii) Demonstrate effect of zero interpolation.

```
clc;
close;
clear;
fm=input('Enter the input signal frequency:');
fs=input('Enter the sampling frequency:');
k=input('Enter the number of Cycles of input signal:');
tm=0.0001:1/fs:k/fm;
x=cos(2*%pi*fm*tm);
No_of_samples=k*fs/fm;
x_fft=fft(x);
x_mag=abs(x_fft);
N=length(x);
freq_axis=(0:N-1)*fs/N;
plot2d3(freq_axis,x_mag);
title('Magnitude spectrum of the signal');
xlabel('Frequency (Hz)');
ylabel('Amplitude');
//verify the following cases (with and without aliasing for smpling theorem)
```

1. fm=10;fs=100;m=2; (fs>2fm—No aliasing)
2. fm=10;fs=20, m=2; (fs=2fm—No aliasing)
3. fm=10;fs=15, m=2; (fs<2fm—aliasing)

//verify the following cases (examples without spectral leakage)

1. fm=10;fs=100;m=2;
2. fm=10;fs=125;m=2;
3. fm=75;fs=250;m=3;

//verify the following cases (examples with spectral leakage)

1. fm=10;fs=125;m=1;
2. fm=200;fs=100000;m=2.5;

% Effect of Zero padding

```
clc;
close;
clear;
x=input('Enter a sequence:');
k=input('Number of zeros to be appended:');
N=length(x);
x_appended=[x, zeros(1,k)];
N_appended=length(x_appended);
bin_no=0:N-1;
bin_no_appended=0:N_appended-1;
x_dft=abs(fft(x));
x_appended_dft=abs(fft(x_appended));
figure(1);
subplot(211),plot2d3(bin_no,x_dft);
xtitle('DFT of the original sequence','Bin no.','Amplitude');
subplot(212),plot2d3(bin_no_appended,x_appended_dft);
xtitle('DFT of the zero appended sequence','Bin no.','Amplitude');
```

Examples

1. x[n]=[1 2 3 4] before and after appending 4 zeros
2. x[n]=[1 1 1 1] before and after appending 4 zeros

% Effect of inserting zeros in between samples (Interpolation)

```
clc;
close;
clear;
x=input('Enter a sequence:');
k=input('Enter the number of zeros to be inserted after every sample:');
x_mod=[];
N=length(x);
for i=1:N
x_mod=[x_mod, x(i), zeros(1,k)];
end
N_mod=length(x_mod);
x_dft=abs(fft(x));
x_mod_dft=abs(fft(x_mod));
bin_no=0:N-1;
bin_no_mod=0:N_mod-1;
figure(1);
subplot(211),plot2d3(bin_no,x_dft);
xtitle('DFT of the original sequence','Bin no.','Amplitude');
subplot(212),plot2d3(bin_no_mod,x_mod_dft);
xtitle('DFT of the zero inserted sequence','Bin no.','Amplitude');
```

Examples

1. x[n]=[1 2 3 4] before and after inserting zeros
2. x[n]=[1 1 1 1] before and after inserting zeros

% Effect of repetition in time domain

```
clc;
close;
clear;
x=input('Enter a sequence:');
k=input('Enter number of times the sequnece has to be repeated:');
x_mod=x;
N=length(x);
for i=1:k
x_mod=[x_mod, x];
end
N_mod=length(x_mod);
x_dft=abs(fft(x));
x_mod_dft=abs(fft(x_mod));
bin_no=0:N-1;
bin_no_mod=0:N_mod-1;
figure(1);
subplot(211),plot2d3(bin_no,x_dft);
xtitle('DFT of the original sequence','Bin no.','Amplitude');
subplot(212),plot2d3(bin_no_mod,x_mod_dft);
xtitle('DFT of the repeated sequence','Bin no.','Amplitude');
```

Examples

1. x[n]=[1 2 3 4] before and after repetition
2. x[n]=[1 1 1 1] before and after repetition

4. Plot the frequency response of a sequence x[n] using N point DFT. Verify the result using SCILAB. OR Plot the magnitude and phase spectrum of the given sequence using DIT-FFT /DIF-FFT algorithm. Verify the result using SCILAB.

Design steps

1. Using signal flow graph or DFT formula compute the DFT of the sequence x(n).
2. Compute the magnitude response |X(k)|.
3. Compute the phase response of X(k).
4. Compute the N-point IDFT of X (k) and verify that it is same as x (n).

Design example: Compute 8 point DFT of the sequence x(n)={1,1,1,1,0,0,0,0} and hence plot its magnitude and phase response.

Solution:

1 Using signal flow graph (either DFT or DIF FFT) compute 8 point DFT of x (n).

X (k) = [4, 1-2.412j, 0, 1-0.4142j, 0, 1+0.4142j, 0, 1+2.4142j]

2. Represent X (k) in polar form

|X (k)|= [4, 2.6131, 0, 1.0824, 0, 1.0824, 0, 2.6131]

$\angle$ X(k) in(radians) = [0,-1.1781, 0, -0.3927, 0, 0.3927, 0, 1.1781]

3. Plot magnitude and phase spectrum as a function of w=2Π k / N

4. compute 8-point IDFT of X (k) and verify that IDFT [X (k)] =x (n)

```
clc;clear;close;
x=input ('enter the input sequence values x (n)= ');   //[1 0 1 0 1 0 1 0]
N=input('enter the number of frequency samples = ');


//To find FFT
for k=1:N
   X(k)=0;
  for n=1:length(x)
      X(k)=X(k)+x(n).*exp((-%i).*2.*%pi.*(n-1).*(k-1)./N);
  end
end


X_mag=abs(X);                                        //Magnitude spectrum
X_ph=atan(imag(X),real(X));                         //Phase spectrum
```

//To plot magnitude and phase response of X(k).

```
k=0:N-1;
subplot(211);plot2d3(k,X_mag);xtitle('magnitude response', 'k','|X(k)|')
subplot(212);plot2d3(k,X_ph);xtitle('Phase response', 'k','phase of X(k)')
```

//To find IFFT

```
for n=1:length(x)
    x_cap(n)=0;
   for k=1:N
        Y(k)=X(k).*exp((%i).*2.*%pi.*(n-1).*(k-1)./N)
       x_cap(n)=x_cap(n)+(1/N).*Y(k);
    end
end
disp(real(x_cap),'Reconstructed signal is:')
```

```
//Expected result
//FFT: X=[4 0 0 0 4 0 0 0]
//IFFT x=[ 1 0 1 0 1 0 1 0]
```

Problems: (1) $x(n) = (-1)^n$, $0<n<3$
(2) $x(n) = (2)^{-n} u(n) – (2)^{-n} u(n-4)$

5. Obtain the response of a system with impulse response h[n] for an input sequence x[n] using DFT relations. Verify the result using SCILAB. Given input Sequence x[n] = ………………………… impulse Response h[n] = ………………………

Solution: Calculate the output sequence y[n] = x[n] *h[n] either by using formula or by graphical method or matrix method.

Example: x[n] = {-2,-1,3,2,8}

h[n] = {5,-4,3,2}

```
clear;
clc;
close;
x=[1 -2 3 4];
h=[1 -1];
L1=length(x);
L2=length(h);
L3=L1+L2-1;
for n=0:L3-1
   y(n+1)=0;
   for k=0:n
      if (((n-k)<L2)&(k<L1))
         y(n+1)=y(n+1)+x(k+1)*h(n-k+1);
end
end
end

disp(y);

 //Method 2 Using In built Function
f = convol(x,h)
disp(f, ' Convolution Sum Result using Inbuilt Function = ')

//Method 3 Using frequency Domain multiplication
```

```
N = L1 +L2 -1;                          //Linear convolution output length
x = [ x zeros(1 ,N - L1 ) ];
h = [ h zeros(1 ,N - L2 ) ];
f1 = fft(x)
f2 = fft(h)
f3 = f1.* f2 ;                          // Multiplication in frequency domain
f4 = ifft(f3)
disp (f4 , 'Convolution Sum Result DFT and IDFT method = ')

//To plot input, impulse and output signals.
subplot (5,1,1) ;plot2d3(x);xtitle('Input signal x ','n','x[n]');
subplot(5,1,2) ;plot2d3(h);xtitle('Impulse signal h','n','h[n]');
subplot(5,1,3) ;plot2d3(y);xtitle('Output signal ','n','y[n]');
subplot(5,1,4) ;plot2d3(f);xtitle('Output signal ','n','f[n]');
subplot(5,1,5) ;plot2d3(f4);xtitle('Output signal ','n','f4[n]');
```

6. Compute Circular Convolution of sequences x[n] and h[n] using DFT relations. Verify the result using SCILAB. Given Input Sequence x[n] = ………………… Impulse Response h[n] = …………………………..N=……

```
clear;
clc;
close;
x=[1 2 0 1];
h=[2 -1 1 -1];
L1=length(x);
L2=length(h);
N=max(L1,L2);
if (L1>L2)
   h=[h zeros(1,L1-L2)];
else
   x=[x zeros(1,L2-L1)];
end
for n=0:N-1
   y(n+1)=0;
   for k=0:N-1
      n_minus_k=n-k;
      if n_minus_k<0
         n_minus_k=n_minus_k+N;
         end
         y(n+1)=y(n+1)+x(k+1)*h(n_minus_k+1);
end
end
disp(y);
```

// Frequency domain

```
X1=fft(x1);
X2=fft(x2);
Y=X1.*X2;
```

```
y=ifft(Y);
disp (y , 'Circular Convolution y = ')
```

Problems:

2. x(n) = δ(n) -2 δ(n-1) + 3 δ(n-2), h(n) = u(n) – u(n-3)
3. x(n) = cos(nπ/4), 0<n<3, h(n) = sin(nπ/4), 0<n<3
4. x [n] = $\{1,2,\underset{\uparrow}{2},1\}$ and h[n] = $\{2, 3, \underset{\uparrow}{1}, 1\}$
5. x[n]= {1, 2, 3, 4} and h[n] = {4, 2, 3, 3}

7. **Design a digital FIR low pass filter with following specifications.**

i) Pass band cut-off frequency : wp=_________rad

ii) Pass band ripple : rp=__________dB

iii) Stop band cut-off frequency : ws=__________rad

iv) Stop band attenuation : rs=__________dB. Choose an appropriate window function and determine impulse response and provide a plot of frequency response of the designed filter. Verify the result using SCILAB.

Design steps:

1. Choose an appropriate window function from the stop band attenuation using the following table.

Sl. No	Transistion width	Window function	Stop band attenuation
1	4π / N	Rectangular window	21 dB
2	8π / N	Barlett (triangular) window	27 dB
3	8π / N	Hanning window	44 dB
4	8π / N	Hamming window	53 dB
5	12π/ N	Blackman window	75 dB

II. Determine the order of the FIR filter (number of coefficients) using the relation
$N \geq 2\pi k / (ws-wp)$
where k=2 for rectangular window and k=4 for Barlett, Hanning and Hamming windows.

<u>Note:</u> Choose N as odd. If N is even select next odd integer.

III. Choose the cut-off frequency of the filter as wc=wp.

IV. Choose $\alpha = (N-1)/2$ so that linear phase is ensured.

Barlett (triangular) window	: 1 - 2 (n – [(N-1) / 2]) / (N-1)
Hanning window	: 0.5 – 0.5cos (2πn) / (N-1)
Hamming window	: 0.54 – 0.46cos (2πn) / (N-1)
Blackman window	: 0.54 – [0.42cos (2πn) / (N-1)] + 0.8cos (4πn) / (N-1)

V. The impulse response of the filter is given by

$$h(n) = \frac{\sin(wc(n-\alpha))}{} * w(n)$$

where w(n) is window function

$$\pi\,(n-\alpha) \qquad\qquad \text{for} 0<n<N-1 \ \&\ n \neq \alpha$$

$$= \frac{wc}{\pi} * w(n) \qquad \text{for } n = \alpha$$

$H(e^{jw}) = e^{-jw(\alpha)} * [\sum 2h(n)\cos(w(n-\alpha))+h((N-1)/2)]$

Design example:

Design a digital FIR low pass filter with following specifications.

a) Pass band cut-off frequency	:0.3π rad
b) Pass band ripple	:0.25 dB
c) Stop band cut-off frequency	:0.45π rad
a) Stop band attenuation	: 50 dB

I step: Choose an appropriate window function from the stop band attenuation specification.

As rs = 50 dB select Hamming window.

(Note: Though Blackman window can be used it results in higher transition bandwidth)

II step: Compute the order of the FIR filter (number of coefficients) using the relation

$N\geq\ 2\pi k/(wp-ws)$[k=4 for Hamming window]

$N\geq\ 53.5$ (choose N=55)

III step: Choose the cut-off frequency of the filter as wc=wp= 0.3* pi rad.

IV step: Choose $\alpha = (N-1)/2$ so that linear phase is ensured.

V step: The impulse response of the filter is given by

$$h(n-\alpha)= \frac{\sin(wc(n-\alpha))}{\pi\,(n-\alpha)} * w(n) \quad \text{where } w(n) \text{ is window function}$$

$$\text{for} 0<n<N-1 \ \&\ n \neq \alpha$$

h(n- α)=sin(0.3*pi(n-27))/(pi*(n-27))*[0.54-0.46*cos(2*pi*n/(N-1))]

for0<n<N-1 & n ≠ α

For ex : The impulse response coefficients are

h(0)=0.00029114 = h(54)

h(1)= -5.9806*10-4 = h(53)……..

Note: when n=α

i.e. h(27) = wc/pi*w(27) =0.3*1=0.3

VI. Frequency response can be obtained using equation

$H(e^{jw}) = e^{-jw(\alpha)} * [\sum 2h(n)\cos(w(n-\alpha))+h((N-1)/2)]$

Note: The symmetry of the impulse response must be verified and the frequency response must satisfy the specifications.

SCILAB Program

```
clc;
clear;
close;

wp=0.3*%pi;
ws=0.45*%pi;
rs=50;
freq_points=1024;
k=4;
trw=ws-wp;
N=(k*2*%pi/trw);
N=ceil(N);
if pmodulo(N,2)==0
N=N+1;
end
wc=wp;
aph=(N-1)/2;
for n=0:N -1
if n== aph
hdn_minusalph (n+1)=wc/ %pi ;
else
hdn_minusalph (n+1)= sin(wc .*(n-aph)) ./( %pi .*(n-aph ));
end
end
```

```
for n=0:N-1
if n== aph
wndw(n+1)=1;
else
wndw(n+1)=0.54-(0.46*cos((2*%pi*n)./(N-1)))
end
end
hn=hdn_minusalph.*wndw;
n=0:N-1;
figure(1); subplot(311);plot(n,wndw);xlabel('n');ylabel('wndw');
subplot(312);plot(n,hdn_minusalph);xlabel('n');ylabel('hdn_minusalph');
subplot(313);plot(n,hn);xlabel('n');ylabel('hn');

h=[hn' zeros(1,freq_points-length(hn))];
H= fft(h);
mag_H=20*log10(abs(H));
ph= atan(imag (H),real(H));
phase_H=unwrap(ph);
k=0: freq_points-1;
omega=k*2*%pi/( freq_points);
figure(2); subplot (211) ,plot2d (omega(1: freq_points/2) , mag_H (1: freq_points/2) );
xtitle ( 'Magnitude response' , 'w ( rad ) ' , 'Magnitude (dB)' );
subplot (212) ,plot2d (omega(1: freq_points/2) , phase_H (1: freq_points/2) );
xtitle ( ' Phase Response ' , 'w ( rad ) ' , ' Phase ( rad ) ' );
```

Problems:

1. Design a digital FIR low pass filter with following specifications.

a) Pass band cut-off frequency	:0.4π rad
b) Pass band ripple	:0.25 dB
c) Stop band cut-off frequency	:0.6π rad
a) Stop band attenuation	: 44 dB

2. Design a digital FIR low pass filter with following specifications.

a) Pass band cut-off frequency	:0.25π rad
b) Pass band ripple	:0.25 dB
c) Stop band cut-off frequency	:0.3π rad
a) Stop band attenuation	: 50 dB

8. Design a digital FIR low pass filter using Kaiser window with following specifications.

i) Pass band cut-off frequency :wp=_________rad

ii) Pass band ripple :rp=__________dB

iii) Stop band cut-off frequency: ws=__________rad

iv) Stop band attenuation :rs=__________dB. Determine impulse response and provide a plot of frequency response of the designed filter. Verify the result using SCILAB.

Design : Kaiser window is an adjustable window function which provides flexible transition bandwidths. The window function is given by

$w(n) = I_0[\ \beta \sqrt{}\ (1 - (1 - 2n / (M-1)^2\ ] / I_0[\beta]$

where I_0 is the modified zero-order Bessel function given by

$I_0(x) = 1 + \sum [\ (x/2)^k / k\ !]^2$ which is positive for all real values of x.

The parameter β controls the minimum stopband attenuation and can be chosen to yield different transition bandwidths for the same order.

- If β = 5.658, then transition width is equal to 7.8 pi /N, and the minimum stopband attenuation is equal to 60 dB.
- If β = 4.538, then the transition width is equal to 5.8 pi/N, and the minimum stopband attenuation is equal to 50 dB.

Empirical design equations:

Given wp, ws, rp and As, the parameters N and β are given by

transition width = Δw = ws-wp

Filter length N ~ (A – 7.95 / 2.285 Δw) + 1

$$\text{Parameter } \beta = \begin{cases} 0.1102(A-8.7), & A \geq 50 \\ 0.5842(A-21)^{0.4} + 0.07886\,(A-21), & 21 < A < 50 \end{cases}$$

Design example:

Design a digital FIR lowpass filter with the following specifications:

wp= 0.2π,

ws= 0.3π,

Rp= 0.25

dB, As=50 dB

Determine the impulse response and plot the frequency response.

Steps:

1. Find δp from $A_p = -20\log_{10}\left(\frac{1+\delta_p}{1-\delta_p}\right)$, δp=-0.0144
2. Find δs from $A_s = -20\log_{10}\delta_s$, δs=0.0032
3. $\delta = \min\left(\left|\delta_p\right|, \delta_s\right)$, δ=0.0032
4. $A = -20\log_{10}\delta$, A=49.897 dB
5. Choose the order of the filter using

   N = (A – 7.95 / 2.285 Δw) = 58.4 ≈59
6. Determine β based on the value of N

   β = 0.1102(A-8.7), As, A ≥ 50, β = 4.539
7. Obtain the Kaiser window function using

   w_kaiser = (kaiser(N, beta))';

8. The impulse response of the filter is given by

   h = hd .* w_kaiser;

SCILAB PROGRAM

```
clc;
clear;
close;

wp=0.2*%pi;
ws=0.3*%pi;
kp=0.25
ks=50;
freq_points=1024;
trw=ws-wp;
dp = ((10.^(-kp/20))-1)./((10.^(-kp/20))+1);
ds = (10.^(-ks/20));
d = min(abs(dp),ds);
A =-20*log10(d);
N = ceil((A-7.95)/(2.285*trw));
```

```
N=ceil(N);
if pmodulo(N,2)==0
N=N+1;
end
wc=wp;
aph=(N-1)/2;
for n=0:N -1
if n== aph
hdn_minusalph (n+1)=wc/ %pi ;
else
hdn_minusalph (n+1)= sin(wc .*(n-aph)) ./( %pi .*(n-aph ));
end
end

n=0:1:N-1;
beta=0.1102.*(A-8.7);
wndw=window('kr',N,beta);

hn=hdn_minusalph.*wndw';
n=0:N-1;
figure(); subplot(311);plot(n,wndw);xlabel('n');ylabel('wndw');
subplot(312);plot(n,hdn_minusalph);xlabel('n');ylabel('hdn_minusalph');
subplot(313);plot(n,hn);xlabel('n');ylabel('hn');

h=[hn' zeros(1,freq_points-length(hn))];
H= fft(h);
mag_H=20*log10(abs(H));
ph= atan(imag (H),real(H));
phase_H=unwrap(ph);
k=0: freq_points-1;
omega=k*2*%pi/( freq_points);
figure(); subplot (211) ,plot2d (omega(1: freq_points/2) , mag_H (1: freq_points/2) );
xtitle ( 'Magnitude response' , 'w ( rad ) ' , 'Magnitude (dB)' );
subplot (212) ,plot2d (omega(1: freq_points/2) , phase_H (1: freq_points/2) );
xtitle ( ' Phase Response ' , 'w ( rad ) ' , ' Phase ( rad ) ' );
```

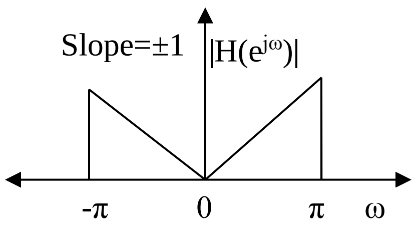
9. Design a Hilbert transformer of length _____ using a _______ window. Verify the result using SCILAB.

The frequency response of a linear-phase ideal differentiator is given by

$$H_d(e^{jw}) = \begin{cases} j\,\omega, & 0 < \omega < \pi \\ -jw, & -\pi < \omega < 0 \end{cases}$$

The ideal impulse response of a digital differentiator shifted by α with linear phase is given by

$$h_d(n-\alpha) = \begin{cases} \cos\pi(n-\alpha)/(n-\alpha), & n \neq \alpha \\ 0, & n = \alpha \end{cases}$$



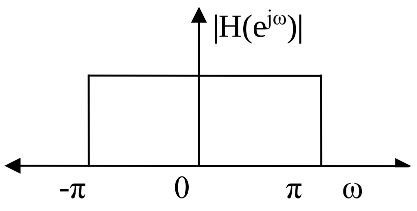
Scilab Program:

```
clc;clear;close;
N = input (" enter the window length ");
freq_points=1024;
windowfn =window('hm',N);
n = 0:N-1;
aph = (N-1)/2;
for n=0:N-1
   if n==aph
      hd(n+1)=0;
      else
    hd(n+1)= cos(%pi*(n-aph))./(n-aph);
    end
end
n=0:N-1;
hn = hd.*windowfn';
omega=-%pi:2*%pi/(freq_points-1):%pi;
hn_append=[hn' zeros(1,freq_points-length(hn))];
H=abs(fft(hn_append,-1));
figure();
plot(omega,fftshift(H));
xtitle('Magnitude Response of FIR differentiator', 'Frequency (rad)','Magnitude');
```

10. Design a differentiator of length_____ using a _______ window. Verify the result using SCILAB.

The ideal frequency response of a linear phase Hilbert transformer is given by

$$H_d(e^{jw}) = \begin{cases} -j\, e^{(-j w\alpha)}, & 0 < w < pi \\ j\, e^{(-j w\alpha)}, & -pi < w < 0 \end{cases}$$



The ideal impulse response is given by

$$h_d(n-\alpha) = \begin{cases} 2/pi \quad (\sin^2 pi(n-\alpha)/2) / (n-\alpha), & n \neq \alpha \\ 0, \; n = \alpha \end{cases}$$

SCILAB Program

```
clc;
clear;
close;

N = input("enter the window length");
freq_points=1024;
windowfn =window('hn',N);
n = 0:N-1;
aph = (N-1)/2;
for n=0:N-1
   if n==aph
      hd(n+1)=0;

 else
    hd(n+1)=(2/%pi)*((sin((%pi/2)*(n-aph)).^2)./(n-aph));

end
end
n=0:N-1;
 hn = hd.*windowfn';
 omega=-%pi:2*%pi/(freq_points-1):%pi;

hn_append=[hn' zeros(1,freq_points-length(hn))];
H=abs(fft(hn_append,-1));

figure();
plot(omega,fftshift(H));
xtitle('Magnitude Response of FIR hilbert transformer', 'Frequency (rad)','Magnitude');
```

11. Design an IIR digital Butterworth filter for the following specifications. Use bilinear transformation. Verify the result using SCILAB.

Design specifications:

a) rp=passband attenuation(dB)= -1dB
b) rs=stopband attenuation(dB)= -3 dB
c) f1=Passband edge(Hz)=1500 Hz
d) f2=Stopband edge(Hz)=2000 Hz
e) Fs=1/Ts=Sampling rate(samples/sec)=8000 Hz

Design steps:

I. Obtain the equivalent digital filter specifications:

i) w1=2*pi*f1*Ts rad.

ii) w2=2*pi*f2*Ts rad.

II. Prewarp the frequencies w1 & w2:

i) o1=2/Ts*tan(w1/2)

ii) o2=2/Ts*tan(w2/2)

III. Design an analog butterworth filter specifications rp,rs,o1,o2:

i) order of the filter=$n=\log_{10}[(10^{-k1/10}-1)/(10^{-k2/10}-1)]/2*\log_{10}(o1/o2)$

ii) cut-off frequency = $oc=o2/(10^{-k2/10}-1)^{1/2n}$

iii) from the butterworth polynomial table write down the transfer function of the normalized analog butterworth filter Hn(s).

iv) unnormalise the filter using substitution H(s)=Hn(s) with s=s/oc.

IV. Using blt obtain equivalent digital filter system function:

$H(z) = H(s)$, where $s=2/Ts*(1-z^{-1})/ (1+z^{-1})$

V. Verify the frequency response as to whether the filter satisfies the given specifications(w1,w2,rp,rs)

VI. For analog filter frequency response, x-axis is in Hz.

That is pass band edge after prewarping in Hz=o1/2*pi.

& stop band edge after prewarping in Hz=o2/2*pi.

For digital filter frequency response, x-axis is in rad. & it is normalized.

That is pass band edge=w1/pi.

And stop band edge=w2/pi.

Design example: Specifications:

a) pass band edge=f1=1500Hz
b) stop band edge=f2=2000 Hz
c) sampling rate =Fs=8000 Hz = 1/Ts
d) passband attenuation = -1db
e) stop attenuation = -3 db

Solution:

I. Equivalent digital specifications:

w_1=(2*pi*1500)/8000=1.178 rad,

w_2=(2*pi*2000)/8000=1.5708 rad

II. Prewarping:

o1 = 2/Ts * tan(w1/2)=10689.73 rad/sec

o2 = 2/Ts * tan(w2/2)=16000 rad/sec

III. Design of analog filter

a) order $n=\log_{10}[(10^{-k1/10}-1)/(10^{-k2/10} -1)]/2*\log_{10}(o1/o2)\cong 2$
b) cut-off frequency $oc=o2/ (10^{-k2/10}-1)^{1/2n}$ = 16000 rad/sec
c) $Hn(s) = 1/(s^2 +1.414 s + 1)$
d) $H(s) = H(s) \mid s = s/ oc =256 * 10^6 /(s^2 +22654.29s+256 * 10^6)$

IV. Digital filter using Bilinear transformation

$H(z) = H(s) \mid s = 2/Ts * (1- z^{-1}) / (1+z^{-1})$

$$H(z) = \frac{0.2932z^2+0.586z+0.2932}{z2+0.0014z+0.1716}$$

%PROGRAM:

```
clear all;clc;close;
f1=input('Enter the pass band edge(Hz)= ');
f2=input('Enter the stop band edge(Hz)= ');
k1=input('Enter the pass band attenuation(dB)= ');
k2=input('Enter the stop band attenuation(dB)= ');
fs=input('Enter the sampling rate(Hz)= ');


//Digital filter specifications(rad)
w1=2*%pi*f1*1/Fs;
w2=2*%pi*f2*1/Fs;


//Pre warping
o1=2*Fs*tan(w1/2);
o2=2*Fs*tan(w2/2);

//Design of analog filter
n=log10(((10.^(-k1/10))-1)/((10.^(-k2/10))-1))./(2*log10(o1/o2));
n=ceil(n);
oc= o2./((10.^(-k2/10)-1).^(1/(2*n)));
wc=2*atan(oc/(2*Fs));
hs=analpf(n,'butt',[],oc);

//Design the digital filter
hz=iir(n,'lp','butt',wc/(2*%pi),[]);
[hzm,fr]=frmag(hz,256);
magz=20*log10(hzm)';
figure();
plot2d(fr*(2*%pi),magz);xtitle('Digital IIR filter: lowpass','frequency(rad)', 'gain(dB)',' ');
```

Problems:

1. Realize an IIR Butterworth filter with pass band edge at 1200 Hz and stop band edge at 2000 Hz for a sampling frequency of 8 kHz. Variations of gain within pass band is 0.5 dB and stop band attenuation of 40 dB. Use bilinear transformation.

2. Realize an IIR Butterworth filter with pass band edge at 1000 Hz and stop band edge at 3000 Hz for a sampling frequency of 8 kHz. Variations of gain within pass band is 2 dB and stop band attenuation of 20 dB. Use bilinear transformation.

12. Design an IIR digital Chebyshev filter for the following specifications. Use bilinear transformation. Verify the result using SCILAB.

Design specifications:

a) rp=passband ripple(dB) = + 1 dB
b) rs=stopband attenuation(dB) = - 3 dB
c) f1=passband edge(Hz) = 1500 Hz
d) f2=stopband edge(Hz) = 2000 Hz
e) fs=1/Ts=sampling rate(samples/sec) = 8000 Hz

Design steps:

I. Obtain the equivalent digital filter specifications:

i) w1=2*pi*f1*Ts rad.
ii) w2=2*PI*f2*Ts rad.

II. Prewarp the frequencies w1 & w2:

i) o1=2/Ts*tan(w1/2)
ii) o2=2/Ts*tan(w2/2)

III. Design an analog chebyshev filter for specifications rp,rs,o1,o2:

Determine A and ε from the specifications:

$$-10 \log10(1+\varepsilon^2) = rp$$

$$-10 \log10(A^2) = rs$$

Determine $g = ((A^2-1)^{1/2}/ \varepsilon)$.

Obtain the stop band edge for the normalized filter or = o2/01

Order of the filter = $n = \log_{10}[g+(g^2-1)^{1/2}] / \log_{10}[or +(or^2 -1)^{1/2}]$

Cut-off frequency = oc= o1

From the chebyshev polynomial table write down the transfer function of the normalized analog Chebyshev Filter Hn(s)=k/ Chebyshev polynomial

Where $k=b_o$ for n odd, $b_o /(1+\varepsilon^2)^{1/2}$ for n even.

Un-normalized the filter using substitution H(s) = Hn(s) with s = s/oc.

Using BLT, obtain equivalent digital filter system function:

$H(z) = H(s)(s=2/Ts*(1-z^{-1})/(1+z^{-1}))$

Verify the frequency response as to whether the filter satisfies the given specifications(w1,w2,rp,rs)

For analog filter frequency response, x-axis is in Hz.

That is pass band edge after prewarping in Hz=o1/2*pi.

& stop band edge after prewarping in Hz=o2/2*pi.

For digital filter frequency response, x-axis is in rad. & it is normalized.

That is pass band edge=w1/pi.

And stop band edge=w2/pi.

Design Example:

a) Pass band edge =f1=1500 Hz
b) Stop band edge=f2 = 2000 Hz
c) Sampling rate=Fs = 8000 Hz
d) Passband attenuation = + 1 dB
e) Stopband attenuation = - 3 dB

Solution:

Equivalent digital specifications:

w1=(2*pi*1500)/8000= 1.178 rad,

w2=(2*pi*2000)/8000= 1.5708 rad;

Prewarping:

o1=2/Ts*tan(w1/2)=10689.73 rad/sec

o2=2/Ts*tan(w2/2)=16000 rad/sec

or=o2/o1=1.496

Design of analog filter

$-10\log(1+\varepsilon^2)= -1.$ $\varepsilon = 0.5088$

$-10\log(A^2) = -3$ $A=1.413$

$g=((A^2-1))^{1/2}/\ \varepsilon) = 1.962$

order $n=\log10[\ g+ (g^2-1)^{1/2}\ ] / \log10[or+(or^2-1)^{1/2}\ ]=2$

cut-off frequency wc=o1=10689.73 rad/sec

$Hn(s)= 0.975/(s^2 +1.095\ s +1.1\ )$

$H(s)= Hn(s)\ |\quad s=s/o1$

$$H(s)=\frac{112.3*10^6}{s^2+11735.72s+126.01*10^6}$$

Digital Filter using BLT

$$H(z)= H(s)|_{s = 2/Ts*(1-z^{-1})/ (1+z^{-1})}$$

$$H(z) = \frac{0.197z^2+0.3942z+0.197}{z^2-0.45627z+0.3409}$$

PROGRAM:

```
clear;clc;close;
f1=input('Enter the pass band edge(Hz)= ');
f2=input('Enter the stop band edge(Hz)= ');
rp=input('Enter the pass band ripple(dB)= ');
rs=input('Enter the stop band attenuation(dB)= ');
Fs=input('Enter the sampling rate(Hz)= ');

rp_ratio=10^(rp/20);

//Digital filter specifications(rad)
w1=2*%pi*f1*1/Fs
w2=2*%pi*f2*1/Fs

//Pre warping
o1=2*Fs*tan(w1/2)
o2=2*Fs*tan(w2/2)
or=o2/o1;//Stop-band edge of normalized lowpass filter


A2 =10.^(-rs/10);
A=sqrt(A2);
epsilon2 = (10.^(-rp/10)-1);
epsilon=sqrt(epsilon2)
g=((A2-1).^0.5./epsilon)

n= (acosh(g))/(acosh(or))
n = ceil(n)
oc=o1;
wc=2*atan(oc/(2*Fs));

hs=analpf(n,'cheb1',[1-rp_ratio],oc);

//Design of digital filter
hz=iir(n,'lp','cheb1',wc/(2*%pi),[1-rp_ratio 1]);
[hzm,fr]=frmag(hz,256);
magz=20*log10(hzm)';
```

```
figure();
plot2d(fr*(2*%pi),magz);xtitle('Digital IIR filter: lowpass','frequency(rad)', 'gain(dB)',' ');
```

% Note: Use zoom/axis commands to verify the design specifications.
%Note: If there is a precision problem to verify the filter co-efficients then int32 ( ) can be used.

Problems:

1. Design an IIR Chebyshev filter with pass band edge at 3000 Hz and stop band edge at 4000 Hz for a sampling rate of 9000 Hz. Variation of gain within pass ripple is 1 dB and stop band attenuation of -40 dB. Use bilinear transformation.

2. Design an normalized IIR Chebyshev Type –I filter with pass band edge at 0.6 rad/sec and stop band edge at 1 rad/sec. Variation of gain within pass ripple is 1 dB and stop band attenuation of -35 dB.

13. Design a simple notch filter to stop a disturbance with frequency F_0=1.25 kHz and a sampling frequency F_s=10 kHz. Verify the result using SCILAB.

```
clear;
close;
clc;

//Read the given ECG signal
fs=360;
[fd,SST,Sheetname,Sheetpos]=xls_open('F:\UG\sem odd 2020-21\Data for
Assignment\ECG.xls');
[signal,index]=xls_read(fd,Sheetpos(1));
mclose(fd);

//Plot the signal in the frequency domain and observe the power line interefnce of 60
Hz and its harmonics at 120 Hz and 180 Hz.
signal_fft=fft(signal);
k=0:length(signal_fft)-1;
f=k*fs/length(signal_fft);
figure;
plot(f,20*log10(abs(signal_fft)));
xtitle('Frequency Domain Input Signal','Frequency in Hz','Gain in dB');

//Design notch filter to eliminate power line interference of 60 Hz.
//Place the poles
w=2*%pi*60/fs;
radius=0.98;

//Filter Coefficients
num=[1 -2*cos(w) 1];
den=[1 -2*radius*cos(w) radius^2];

//Plot the magnuitude response of the filter
```

```
[hz]=frmag(num,den,length(f)/2);
figure;
plot(f(1:length(f)/2),hz);
xtitle('Magnitude Response of the Notch Filter','Frequency in Hz','Gain');

//Filter the signal and observe the absence of 60 Hz frequency
filter_op=filter(num,den,signal);
filter_fft=fft(filter_op);

figure;
plot(f,20*log10(abs(filter_fft)));
xtitle('Frequency domain of the output signal','Frequency in Hz','Gain in dB');
```

14. Design a digital resonator that resonates at 1000 Hz. Assume Fs=8000 Hz. Verify the result using SCILAB.

```
clear;clc;close;
fr=200;Fs=1000;
wr=2*%pi*fr/Fs;
r=0.98;
freq_points=1024;

num=[1];
den=[1 -2*r*cos(wr) r^2];
hz=frmag(num,den,freq_points/2);
figure;
k=0:freq_points-1;
f=k*Fs/freq_points;

figure;
plot(f(1:length(f)/2),hz);
xtitle('Magnitude Response of the Resonator','Frequency in Hz','Gain');
```

15. Design a comb filter that suppresses 60 Hz AC frequency component and its harmonics from the given ECG signal. Assume Fs=360 Hz. Verify the result using SCILAB.

```
clear;clc;close;

//Read the given ECG signal
fs=360;
[fd,SST,Sheetname,Sheetpos]=xls_open('F:\UG\sem odd 2020-21\Data for
Assignment\ECG.xls');
[signal,index]=xls_read(fd,Sheetpos(1));
mclose(fd);
figure;
plot(signal);
xtitle('Time Domain Input');

//Plot the signal in the frequency domain and observe the power line interefnce of 60
Hz and its harmonics at 120 Hz and 180 Hz.
signal_fft=fft(signal);
k=0:length(signal_fft)-1;
f=k*fs/length(signal_fft);

figure;
plot(f,20*log10(abs(signal_fft)));
xtitle('Frequency Domain Input Signal','Frequency in Hz','Gain in dB');

//Moving average filter of length M
freq_points=1024;
fundamental=60;
M=(fs/fundamental);
w=2*%pi*f/fs;
num=ones(1,M);
den=[M];
```

```
hz=frmag(num,den,freq_points/2);
figure;
k=0:freq_points-1;
f=k*fs/freq_points;


figure;
plot(f(1:length(f)/2),hz);
xtitle('Magnitude Response of the Notch Filter','Frequency in Hz','Gain');

//Filter the signal and observe the absence of 60 Hz frequency and its harmonics
filter_op=filter(num,den,signal);
filter_fft=fft(filter_op);
filter_fft(find(filter_fft==0))=0.001; //To ensure that none of the fft output points are
zero to take log

figure;
plot(filter_op);
xtitle('Time Domain output');
k=0:length(signal_fft)-1;
f=k*fs/length(signal_fft);
figure;
plot(f,20*log10(abs(filter_fft)));
xtitle('Frequency domain of the output signal','Frequency in Hz','Gain in dB');
```

16. Generate a DTMF signal given frequency F1 and F2. Using Goertzel algorithm, decode the DTMF signal. Verify the result using SCILAB.

```
clear;
clc;
close;

// DTMF Generation
row_number=input('Row Number');
column_number=input('Column Number');
fr=[697 770 852 941];
fc=[1209 1336 1477 1633];
fs=8000;
N=0.04*8000;//40 ms duration
f1=fr(row_number);
f2=fc(column_number);
tm=0:1/fs:(N-1)/fs;
x1=cos(2*%pi*f1*tm);
x2=cos(2*%pi*f2*tm);

x=x1+x2;

x_dft=abs(fft(x));
freq_axis=(0:N-1)*fs/N;
figure;
plot(freq_axis,x_dft);

//DTMF Detector
fr=[697 770 852 941];
fc=[1209 1336 1477 1633];

// Calculate the bin numbers for these frequencies
kr=ceil(fr*N/fs);
kc=ceil(fc*N/fs);

//To find WN_k (coefficient for first order Goertzel filter)
WN_kr=exp(%i*2*%pi*kr/N);
WN_kc=exp(%i*2*%pi*kc/N);

//Filteing Operation

for i=1:length(kr)
```

```
    yk_n(1)=x(1);
 for n=2:length(x)
    yk_n(n)=x(n)+WN_kr(i)*(yk_n(n-1));
 end
geortzel_op_kr(i)=abs(WN_kr(i)*yk_n(N));
end

for i=1:length(kc)
    yk_n(1)=x(1);
 for n=2:length(x)
    yk_n(n)=x(n)+WN_kc(i)*(yk_n(n-1));
 end
geortzel_op_kc(i)=abs(WN_kc(i)*yk_n(N));
end

row=find(geortzel_op_kr==max(geortzel_op_kr));
clm=find(geortzel_op_kc==max(geortzel_op_kc));

disp([row clm]);
```

DSP using C-language

17. Write a C-program to compute N-point DFT of a sequence x[n].

```
#include <stdio.h>
#include <math.h>
float x[8]={1,1,1,1,0,0,0,0};
        float y[16];
        float z[8]={0,0,0,0,0,0,0,0};                // To store  the magnitude of DFT
main(){
        float w;
        int n,k, N=8,xlen=4,i;
        for(k=0,i=0;i<N;i++,k=k+2){
                y[k]=0; y[k+1]=0;                    //initialize real & imag parts
                for(n=0;n<xlen;n++){
                        w=-2*3.1416*i*n/N;           //careful about minus sign
                        y[k]=y[k]+x[n]*cos(w);
                        y[k+1]=y[k+1]+x[n]*sin(w);
                }
                z[i]=sqrt(y[k]*y[k]+y[k+1]*y[k+1]);
                printf(" y(%d)= %f+j(%f) \n\n |y(%d)|=  %f\n\n",i,y[k],y[k+1],i,z[i]);
          }
}
```

18. Write a C-program to compute N-point linear convolution of sequences x[n] and h[n].

```
    #include<stdio.h>
    #include<math.h>
    main(){
        float h[4]={2,2,2,2};float x[4]={1,2,3,4};
        float y[10]; int xlen=4; int hlen=4;
        int len;
        int N=xlen+hlen-1;                          //Length of linear convolution
        int k,n;
        for(n=0;n<N;n++){
                y[n]=0;
                for(k=0;k<=n;k++){
                        if(((n-k)<hlen)&(k<xlen))
                        y[n]=y[n]+x[k]*h[n-k];
                }
                printf("%f\n",y[n]);
        }
}
```

NOTE:

Convolution formula

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k)$$

19. Write a C-program to compute N-point circular convolution of sequences x[n] and h[n].

```
# include<stdio.h>
# include<math.h>
# define N 5
float x1[N]={1,2,3,4,5};
float x2[N]={5,4,3,2,1};
float x3[N];
main(){
        int k,n,i;
        for(n=0;n<N;n++)
        {                                   //outer loop for y[n] array
        x3[n]=0;
        for(k=0;k<N;k++)
       {                                    //inner loop for computing each y[n] point
        i=(n-k)%N;                          //compute the index modulo N
        if(i<0) i=i+N;                      //if index is <0, say x[-1], then convert to
                                            //x[N-1]
        x3[n]=x3[n]+x2[k]*x1[i];             //compute output
        }                                   //end of inner for loop
                printf("%f\t",x3[n]);
        }
                                            //end of outer for loop
}                                           //end of main
```

NOTE:

Circular convolution formula

$$x3(n) = \sum_{k=0}^{N-1} x1((n-k))_N \, x2(k)$$

20. Write a C-program to design IIR Butterworth low pass filter and demonstrate the filtering action of a synthesized signal.

```
//This program demonstrates removal of 4kHz Noise from the composite signal
(1kHz+4kHz)

// STEPS:
// Design of LPBW digital IIR filter with the specification
// fp = 1.5 kHz, fs = 2 kHz. Sampling freq= 10 kHz. kp = -1 dB, k2 = -3 dB
// Take these filter co-eff(numz and denz) and obtain difference equation
// Solve this diff.eqn. with input as combination of Signal and Noise
// Observe the filter response( solution to diff. eqn ) both in Time domain and
// in frequency domain
#include <stdio.h>
#include <math.h>
# define fp     1000.0                          // message in PassBand
# define fn     4000.0                          // Noise in StopBand
# define fs     10000.0
# define pi     3.1416
# define N      50                              // 5 Cycles of input
float b0= 0.2069, b1=0.4137, b2=0.2069;         // numz obtained from SCILAB
float a1= -0.3682, a2=0.1956;                   // denz obtained from SCILAB
int i,j;
float  x1[60],x2[60],x[60],y[70],t,z;           // allot memory for i/p & o/p
main( ){
        int k=0;
for(t=.0001,k=0; t<=5*(1/fp);k++,t=t+(1/fs) ){
                x1[k] = 0.55*sin(2*pi*t*fp);
                x2[k] = 0.5*sin(2*pi*t*fn);
                x[k]= x1[k]+x2[k];              // Combination of Signal & Noise
        }
        for(j=0;j<N;j++){                       // Solution to Diff. equation
                y[j]=b0*x[j];
                if(j>0) y[j]=y[j]+b1*x[j-1]-a1*y[j-1];
                if ((j-1)>0) y[j]=y[j]+b2*x[j-2]-a2*y[j-2];
                printf("%f \t",y[j]);
        }
    }
```

PART – C
Open Ended Experiments

This is a group assignment. Each group has to solve one question. Use Scilab.

1. **Speaker Recognition**: Record vowel ‘a’ as in bat from one male and one female speaker. Use sampling rate of 8 kHz. Use 30 ms of each utterance and plot the spectrum of both using 512 point DFT. Comment on the two spectra. If a spectrum of vowel is given, will you be able to identify the gender of the speaker with that?

2. **Understanding the difference between DFT and DCT in terms of energy compactness**: Read the vowel given ‘fa.wav’. Find the sampling rate of the signal. Select 30 ms of the signal from mid of the vowel. How many samples are present in this 30 ms signal? Compute N point DFT of these selected samples where N-is the length of the selected signal. Also, compute N point DCT of these selected samples. Compute N-point IDFT and IDCT of these with first N1 points where (i) N1=10, (ii) N1=50, (iii), N1=100 and N1=200. Plot the original selected 30 ms signal, IDFT and IDCT obtained in each of these cases and compare. What is the inference? Does this illustrate high energy compactness of DCT compared to DFT?

3. **Application related to overlap and save**: Read the given wav file, “machali.wav” using “wavread” function. Find the sampling rate of the signal. Generate a sinusoid of frequency 3000 Hz of length same as that of the given wave file. Add this sinusoid as noise to the signal. Listen to the original and its noisy version. Let the impulse response of filter be h=[1 -2cos(2π3000/fs) 1]. Use overlap and save method with block length 100 to perform linear filtering and listen to original signal, noisy version and filtered signal. Also, plot these three signals and compare.

4. **Application related to overlap and add**: Read the given wav file, “machali.wav” using “wavread” function. Find the sampling rate of the signal. Generate a sinusoid of frequency 3000 Hz of length same as that of the given wave file. Add this sinusoid as noise to the signal. Listen to the original and its noisy version. Let the impulse response of filter be h=[1 -2cos(2π3000/fs) 1]. Use overlap and add method with block length 100 to

perform linear filtering and listen to original signal, noisy version and filtered signal. Also, plot these three signals and compare.

5. **Effect of passing a signal through a moving average filter**: Generate 8 single tones of frequencies varying from 1 to 8 kHz in step of 1 kHz. Use a sampling rate of 8 kHz. Add all 8 signals. Plot the resultant time domain signal and its spectrum. Assume 8000 frequency points. Pass the signal through moving average filters of length-3 and length-5. Plot the output signals and their spectra. Compare the three time domain signals and the three spectra and comment.

6. **Understanding the application of chirp-z transform**: Chirp-Z transform computes z-transform on a spiral contour. One of the applications of chirp-Z transform is spectral zooming. Generate two sinusoids of 2000 Hz and 2060 Hz at a sampling rate of 10 kHz for duration of 0.01 second add them. Let this sequence be x(n). Compute 512 point DFT of the resulting signal. Now compute 512 point DFT of the sequence, $x(n)r^{-n}$ where r=0.96 (circle of radius 0.96). Compare the two plots and comment.

7. **Design of Graphic equalizer**: Design a 3-band graphic equalizer with the following specifications.

   Band1-Low frequency components up to 500 Hz (bass),
   Band2-Band of frequencies from 500 Hz to 1.5 kHz (vocal),
   Band3-High frequency components from 1.5 kHz to 20 kHz (treble).
   Assume sampling rate of 44.1 kHz.
   Verify the design by passing an audio sampled at 44.1 kHz through the equalizer designed.

8. **Compute fundamental frequency of a speaker using autocorrelation**: Record vowel 'a' as in bat from one speaker. Use sampling rate of 8 kHz. Extract 30 ms of the signal from mid of the vowel. Compute its autocorrelation. Measure the difference between two peaks of the autocorrelation function. Measure fundamental frequency of the speaker as reciprocal of that difference.

9. **Sketch the spectrum of an all pass filter**: Plot the magnitude response of an all pass filter and verify H(z)=(a+z)/(1+az) where 0<a<1.

10. **Echo Generation**: Design an echo generator that generates a single echo.