Scilab Manual for
DIGITAL SIGNAL PROCESSING
by Prof Desai Siddhibahen Deveshbhai
Others
Laxmi Institute Of Technology,sarigam[1]

Solutions provided by
Prof Desai Siddhibahen Deveshbhai
Others
Laxmi Institute Of Technology,sarigam

July 16, 2024

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

Contents

List of Experiments

List of Figures

Experiment: 1

WRITE A SCILAB PROGRAM TO GENERATE COMMON DISCRETE TIME SIGNALS.

Scilab code Solution 1.01 PROGRAM TO GENERATE COMMON DISCRETE TIME SIGNALS

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
// CAPTION: PROGRAM TO GENERATE COMMON DISCRETE TIME
    SIGNALS

clc;
clear;
close;

//UNIT IMPULSE SIGNAL
L=input('Enter the Length='); //SET LIMIT
n=-L:L;
```

```
x=[zeros(1,L),ones(1,1),zeros(1,L)];
subplot(2,4,1);
a=gca();
a.y_location ="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3(n,x)
title ('Unit Impulse Signal' );
xlabel('Samples n');
ylabel('Amplitude');

// UNIT STEP SIGNAL
y=[zeros(1,L),ones(1,L+1)];
subplot(3,4,2);
b=gca();
b.y_location ="middle";
b.foreground = 5;
b.font_color = 5;
b.font_style = 5;
plot2d3(n,y)
title('Unit Step Signal' );
xlabel('Samples n');
ylabel('Amplitude');

//UNIT RAMP SIGNAL
z=[zeros(1,L),0:L],
subplot(2,4,3)
c=gca();
c.y_location ="middle";
c.foreground = 5;
c.font_color = 5;
c.font_style = 5;
plot2d3(n,z)
title('Unit Ramp Signal' );
xlabel('Samples n');
ylabel('Amplitude');

```

```
// EXPONENTIALLY INCREASING SEQUENCE

n=0:1:10;
x=exp(n);
subplot(2,4,4);
d=gca();
d.x_location="origin";
d.y_location ="origin";
d.foreground = 5;
d.font_color = 5;
d.font_style = 5;
plot2d3(n,x)
title('Exponentially Increasing Sequence');
xlabel('Samples n');
ylabel('Amplitude');

// EXPONENTIALLY DECREASING SEQUENCE
n=0:1:10;
x=exp(-n);
subplot(2,4,5);
d=gca();
d.x_location="origin";
d.y_location ="origin";
d.foreground = 5;
d.font_color = 5;
d.font_style = 5;
plot2d3(n,x)
title('Exponentially Decreasing Sequence');
xlabel('Samples n');
ylabel('Amplitude');

// SINE WAVE
t=0:0.04:1;
x=sin(2*%pi*t);
subplot(2,4,6);
a=gca();
a.foreground = 5;
a.font_color = 5;
```

```
a.font_style = 5;
plot2d3(t,x);
title('Sine Wave')
xlabel('Samples n');
ylabel('Amplitude');

// COSINE WAVE
t=0:0.04:1;
x=cos(2*%pi*t);
subplot(2,4,7);
b=gca();
b.foreground = 5;
b.font_color = 5;
b.font_style = 5;
plot2d3(t,x);
title('Cosine Wave')
xlabel('Samples n');
ylabel('Amplitude');

//INPUT:
//Enter the Length=5
```

Experiment: 2

WRITE A SCILAB PROGRAM TO OBSERVE THE EFFECTS OF LOWER SAMPLING RATE AND HIGHER SAMPLING RATE ON C.T. SIGNAL(SAMPLING THM.)

Scilab code Solution 2.02 PROGRAM TO OBSERVE THE EFFECTS OF LOWER SAMPLING RATE AND HIGHER SAMPLING RATE ON CONTINUOUS TIME SIGNAL

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM TO OBSERVE THE EFFECTS OF LOWER
   SAMPLING RATE AND
```

```
// HIGHER SAMPLING RATE ON C.T. SIGNAL(SAMPLING THM
    .)

clc;
clear;
close;
f=input('Enter continuous time signal frequency=');
    // f=0.1
a=input('Enter continuous time signal amplitude=');
    //a=1
t=0:0.01:100;
x=a*sin(2*%pi*f*t);
subplot(2,2,1);
b=gca();
b.x_location="origin";
b.y_location="origin";
b.foreground = 5;
b.font_color = 5;
b.font_style = 5;
plot(t,x);
title('Original Continuous time Signal');
xlabel('Time t');
ylabel('Amplitude');

// Sampling Rate=2*max frequency
fs1=input('Enter sampling frequency equal to 2*fs=')
    ;//fs1=0.2
fd=f/fs1;
n=0:0.01:100;
x1n=a*sin(2*%pi*f*n/fs1);
subplot(2,2,2);
b=gca();
b.x_location="origin";
b.y_location="origin";
b.foreground = 5;
b.font_color = 5;
b.font_style = 5;
plot2d3('gnn',n,x1n);
```

```
title('Reconstructed signal with sampling frequency equal 2*fs');
xlabel('Time t');
ylabel('Amplitude');


// Sampling Rate<2*max frequency= Aliasing Effect
fs2=input('Enter sampling frequency less than 2*fs=');//fs2=0.1
x2n=a*sin(2*%pi*f*n/fs2);
subplot(2,2,3);
b=gca();
b.x_location="origin";
b.y_location="origin";
b.foreground = 5;
b.font_color = 5;
b.font_style = 5;
plot2d3('gnn',n,x2n);
title('Reconstructed signal with sampling frequency Less than 2*fs=Aliasing effect');
xlabel('Time t');
ylabel('Amplitude');

// Sampling Rate>2*max frequency=Perfect Reconstuection
fs3=input('Enter sampling frequency Greater than 2*fs=');//fs3=1
x3n=a*sin(2*%pi*f*n*fs3);
subplot(2,2,4);
b=gca();
b.x_location="origin";
b.y_location="origin";
b.foreground = 5;
b.font_color = 5;
b.font_style = 5;
plot2d3('gnn',n,x3n);
title('Perfect Reconstructed signal with sampling frequency greater than 2*fs');
```

```
xlabel('Time t');
ylabel('Amplitude');

//INPUT:
//Enter continuous time signal frequency=0.1
//Enter continuous time signal amplitude=1
//Enter sampling frequency equal to 2*fs=0.2
//Enter sampling frequency less than 2*fs=0.1
//Enter sampling frequency Greater than 2*fs=1
```

Experiment: 3

WRITE A SCILAB PROGRAM TO COMPUTE LINEARITY PROPERTY OF A GIVEN SIGNAL.

Scilab code Solution 3.03 PROGRAM TO COMPUTE LINEARITY PROPERTY OF GIVEN SIGNAL

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
// CAPTION: PROGRAM TO COMPUTE LINEARITY PROPERTY OF
    A GIVEN SIGNAL
//Input Sequence:    x1(n)=sin(2%pi0.1n) and x2(n)=
    cos(2%pi0.5n) and system y(n)=0.5x(n)
clc;
clear;
close;
n=0:10;
a=2;
b=3;
```

```
x1=sin(2*%pi*0.1*n); // input sequence 1
x2=cos(2*%pi*0.5*n); //input sequence 2
x=a.*x1+b.*x2;// homogenity and superposition --op1
y=0.5*x;// system y=0.5*x
y1=0.5*x1;
y2=0.5*x2;
yt=a.*y1+b.*y2;// homogenity and superposition --op2
d=y-yt;// op2-op1=difference
disp('Output of System y(n)=0.5x(n) is:');
if(d==0)
    disp('System is linear');
else
    disp('System is Non-linear');
end

//Input Sequence:    x1(n)=sin(2%pi0.1n) and x2(n)=
    cos(2%pi0.5n) and system y(n)=sqrt(x(n))
n=0:10;
a=2;
b=3;
x1=sin(2*%pi*0.1*n); //input sequence 1
x2=cos(2*%pi*0.5*n);//input sequence 2
x=a.*x1+b.*x2;// homogenity and superposition--op1
y=sqrt(x);// system y=sqrt(x)
y1=sqrt(x1);
y2=sqrt(x2);
yt=a.*y1+b.*y2; // homogenity and superposition--op2
d=y-yt; // op2-op1=difference
disp('Output of System y(n)=sqrt(x(n)) is:');
if(d==0)
    disp('System is linear');
else
    disp('System is Non-linear');
end


//OUTPUT:
 //Output of System y(n)=0.5x(n) is:
```

```
 //System is linear

 //Output of System y(n)=sqrt(x(n)) is:

// System is Non-linear
```

Experiment: 4

WRITE A SCILAB PROGRAM TO COMPUTE LINEAR CONVOLUTION OF TWO SEQUENCES USING BASIC EQUATION.

Scilab code Solution 4.04 PROGRAM TO COMPUTE LINEAR CONVOLUTION OF TWO SEQUENCES USING BASIC EQUATION

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM TO COMPUTE LINEAR CONVOLUTION OF
    TWO SEQUENCES USING BASIC EQUATION
clc;
clear;
close;
//input sequences
x=input('Enter the Input Sequence x(n)=') //x=[1 2 3
    1]
```

```
m=length(x);
xl=input('Enter the lower index of Input Sequence=') //0
xh=xl+m-1;
n=xl:1:xh;
subplot(3,1,1);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',n,x);
title('Input Sequence x[n]');
xlabel('Samples n');
ylabel('Amplitude');

h=input('Enter the Impulse response Sequence h(n)=');//h=[1 2 1 -1]
l=length(h);
hl=input('Enter the lower index of Impulse response Sequence=');//-1
hh=hl+l-1;
g=hl:1:hh;
subplot(3,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',n,h);
title('Impulse Response Sequence h[n]');
xlabel('Samples n');
ylabel('Amplitude');

nx=xl+hl; // range of k
nh=xh+hh;// range of n
```

```
x=[x,zeros(1,l)];
h=[h,zeros(1,m)];
y=zeros(1,m+l-1)// n1+n2-1=length of linear convolution
for i=1:m+l-1
    y(i)=0;
    for j=1:m+l-1
        if(j<i+1)
            y(i)=y(i)+x(j)*h(i-j+1);
            end
    end
end
disp('Linear Convolution using Equation is y(n):')
disp(y);
r=nx:nh;
subplot(3,1,3);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',r,y);
title('Output Response Sequence of Linear Convolution using Equation y[n]');
xlabel('Samples n');
ylabel('Amplitude');
//INPUT:
//Enter the Input Sequence x(n)=[1 2 3 1]
//Enter the lower index of Input Sequence=0
//Enter the Impulse response Sequence h(n)=[1 2 1 -1]
//Enter the lower index of Impulse response Sequence =-1

 //OUTPUT:
 //Linear Convolution using Equation is y(n):

```

```
// 1.      4.      8.      8.      3.   − 2.   − 1.
```

Experiment: 5

WRITE A SCILAB PROGRAM TO FIND AUTOCORRELATION AND CROSS CORRELATION OF THE GIVEN SEQUENCES.

Scilab code Solution 5.05 PROGRAM TO FIND AUTO CORRELATION AND CROSS CORRELATION OF THE GIVEN SEQUENCES

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM TO FIND AUTOCORRELATION AND
   CROSSCORRELATION OF THE GIVEN SEQUENCES
clc;
clear;
close;
x=input('Enter the Input Sequence=') //x=[1 2 3 1]
m=length(x);
xl=input('Enter the lower index of Input Sequence=')
```

```
    //0
xh=xl+m-1;
n=xl:1:xh;
subplot(2,2,1);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',n,x);
title('Input Sequence x[n]');
xlabel('Samples n');
ylabel('Amplitude');

h=input('Enter the Impulse response Sequence=');//h
    =[1 2 1 1]
l=length(h);
hl=input('Enter the lower index of impulse response
    Sequence=');//0
hh=hl+l-1;
g=hl:1:hh;
subplot(2,2,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',g,h);
title('Impulse Response Sequence h[n]');
xlabel('Samples n');
ylabel('Amplitude');

//Autocorrelation
y=xcorr(x,x);
disp('Auto Correlation Of given Sequence y(n)=');
disp(y);
```

```
nx=xl+xl;
nh=xh+xh;
r=nx:nh;
subplot(2,2,3);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',r,y);
title('Output of Auto Correlation of Sequence');
xlabel('Samples n');
ylabel('Amplitude');

//Cross correlation
z=xcorr(x,h);
disp('Cross Correlation of Sequence y(n)=');
disp(z);
subplot(2,2,4);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',r,z);
title('Output of Cross correlation of Sequence');
xlabel('Samples n');
ylabel('Amplitude');

//INPUT:
//Enter the Input Sequence=[1 2 3 1]
//Enter the lower index of Input Sequence=0
//Enter the Impulse response Sequence=[1 2 1 1]
//Enter the lower index of impulse response Sequence=0

```

```
//OUTPUT:
//Auto Correlation Of given Sequence y(n)=

  // 1.    5.    11.    15.    11.    5.    1.

//Cross Correlation of Sequence y(n)=

  // 1.    3.    7.    9.    9.    5.    1.
```

Experiment: 6

WRITE A SCILAB PROGRAM TO FIND N-POINT DFT OF THE GIVEN SEQUENCE.

Scilab code Solution 6.06 PROGRAM TO FIND N POINT DFT OF THE GIVEN SEQUENCE

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
// CAPTION: PROGRAM TO FIND N-POINT DFT OF THE GIVEN
    SEQUENCE
clc;
clear;
close;
N=input('Enter the value of N='); //n-point
x=input('Enter the input sequence x(n)=');
subplot(3,2,1);
a=gca();
a.foreground = 5;
```

```
a.font_color = 5;
a.font_style = 5;
plot2d3(x);
title('Input sequence');
xlabel('Samples n');
ylabel('Amplitude');

//calculation of DFT
for k=1:N
    y(k)=0;
    for n=1:N
        y(k)=y(k)+x(n).*exp(-%i*2*%pi*(k-1)*(n-1)/N)
            ;
        A=real(y);
        B=imag(y);
    end
end
mag=abs(y);
x1=atan(imag(y),real(y));
phase=x1*(180/%pi);
disp('The output DFT sequence is:');
disp(y);
subplot(3,2,2);
a=gca();
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3(y);
title('Output DFT sequence');
xlabel('Samples n');
ylabel('Amplitude');

//REAL VALUE
disp('The resultant real value is:');
disp(A);
subplot(3,2,3);
a=gca();
a.foreground = 5;
```

```
a.font_color = 5;
a.font_style = 5;
plot2d3(A);
title('Real Value');
xlabel('Samples n');
ylabel('Amplitude');

//IMAGINARY VALUE
disp('The resultant imaginary value is:');
disp(B);
subplot(3,2,4);
a=gca();
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3(B);
title('Imaginary Value');
xlabel('Samples n');
ylabel('Amplitude');

//MAGNITUDE RESPONSE
disp('The Magnitude response is:');
disp(mag);
subplot(3,2,5);
a=gca();
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3(mag);
title('Magnitude Response');
xlabel('Samples n');
ylabel('Amplitude');

//PHASE RESPONSE
disp('The Phase response is:');
disp(phase);
subplot(3,2,6);
a=gca();
```

```
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3(phase);
title('Phase Response');
xlabel('Samples n');
ylabel('Phase');



//INPUT:
//Enter the value of N=4
//Enter the input sequence x(n)=[1 2 3 4]

//OUTPUT:
//The output DFT sequence is:

   // 10.
  //- 2. + 2.i
  //- 2. - 9.797D-16i
  //- 2. - 2.i

 //The resultant real value is:

   // 10.
  //- 2.
  //- 2.
  //- 2.

 //The resultant imaginary value is:

   // 0.
   // 2.
  //- 9.797D-16
  //- 2.

 //The Magnitude response is:

```

```
      //  10.
      //2.8284271
      //2.
      //2.8284271

//The Phase response is:

    //   0.
  //   135.
  //- 180.
// - 135.
```

Experiment: 7

WRITE A SCILAB PROGRAM TO COMPUTE LINEAR CONVOLUTION OF TWO SEQUENCES USING DFT BASED APPROACH.

Scilab code Solution 7.07 PROGRAM TO COMPUTE LINEAR CONVOLUTION OF TWO SEQUENCES USING DFT BASED APPROACH

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM TO COMPUTE LINEAR CONVOLUTION OF
   TWO SEQUENCES USING DFT BASED APPROACH
clc;
clear;
close;
x=input('Enter the input sequence x(n)=') //x=[1 2 3
    1]
m=length(x);
```

```
xl=input('Enter the lower index of input sequence=') //0
xh=xl+m-1;
n=xl:1:xh;
subplot(3,1,1);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',n,x);
title('Input Sequence x[n]');
xlabel('Samples n');
ylabel('Amplitude');

h=input('Enter the impulse response sequence h(n)=') ;//h=[1 2 1 -1]
l=length(h);
hl=input('Enter the lower index of impulse response sequence=');//-1
hh=hl+l-1;
g=hl:1:hh;
subplot(3,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',n,h);
title('Impulse Response Sequence h[n]');
xlabel('Samples n');
ylabel('Amplitude');

nx=xl+hl; // range of k
nh=xh+hh;// range of n
p=m+l-1;
```

```
x=[x,zeros(1,p-1)];
h=[h,zeros(1,p-m)];

// dft-idft
XK=fft(x,-1); // DFT of x
HK=fft(h,-1); // DFT of h
YK=XK.*HK; // Multiplication of DFT of x and h
yn=fft(YK,1);  // Inverse DFT for obtaining yn
disp('Linear Convolution by DFT-IDFT Method is y(n):
    ');
disp(real(yn));
r=nx:nh;
subplot(3,1,3);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',r,yn);
title('Output Response Sequence of Linear
    Convolution by DFT-IDFT Method y[n]');
xlabel('Samples n');
ylabel('Amplitude');

//INPUT:
//Enter the input sequence x(n)=[1 2 3 1]
//Enter the lower index of input sequence=0
//Enter the impulse response sequence h(n)=[1 2 1
    -1]
//Enter the lower index of impulse response sequence
    =-1

 //OUTPUT:
 //Linear Convolution by DFT-IDFT Method is y(n):

    //1.    4.    8.    8.    3.  - 2.  - 1.
```

Experiment: 8

WRITE A SCILAB PROGRAM TO COMPUTE CIRCULAR CONVOLUTION OF TWO SEQUECNES USING BASIC EQUATION.

Scilab code Solution 8.08 PROGRAM TO COMPUTE CIRCULAR CONVOLUTION OF TWO SEQUENCES USING BASIC EQUATION

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM TO COMPUTE CIRCULAR CONVOLUTION
   OF TWO SEQUENCES USING BASIC EQUATION
clc;
clear;
close;
x=input('Enter the input sequence=') //x=[1 1 2 2]
m=length(x);
xl=input('Enter the lower index of input sequence=')
```

```
    //0
xh=xl+m-1;
n=xl:1:xh;
subplot(3,1,1);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',n,x);
title('Input Sequence x[n]');
xlabel('Samples n');
ylabel('Amplitude');

h=input('Enter the impulse response sequence=');//h
    =[1 2 3 4]
l=length(h);
hl=input('Enter the lower index of impulse response
    sequence=');//0
hh=hl+l-1;
g=hl:1:hh;
subplot(3,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',g,h);
title('Impulse Response Sequence h[n]');
xlabel('Samples n');
ylabel('Amplitude');


// for making length of both signals equal
N=max(m,l);
p=m-l;
```

```
if(p>=0) then
 h=[h,zeros(1,p)];
else
 x=[x,zeros(1,-p)];
end
for i=1:N
    y(i)=0;
    for j=1:N
        k=i-j+1;
        if(k<=0)
            k=N+k;
        end
        y(i)=y(i)+x(j)*h(k);
    end
end
disp(' Circular convolution by equation is y[n]:');
disp(y);
nx=xl+hl;
r=nx:length(y)-1;
subplot(3,1,3);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',r,y);
title('Output Response Sequence of Circular
   Convolution y[n] using Basic Equation');
xlabel('Samples n');
ylabel('Amplitude');

//INPUT:
//Enter the input sequence=[1 1 2 2]
//Enter the lower index of input sequence=0
//Enter the impulse response sequence=[1 2 3 4]
//Enter the lower index of impulse response sequence
   =0
```

```
 //OUTPUT:
  //Circular convolution by equation is y[n]:

//      15.
  //   17.
   // 15.
    //13.
```

Experiment: 9

WRITE A SCILAB PROGRAM TO COMPUTE CIRCULAR CONVOLUTION OF THE TWO SEQUENCES USING DFT BASED APPROACH.

Scilab code Solution 9.09 PROGRAM TO COMPUTE CIRCULAR CONVOLUTION OF THE TWO SEQUENCES USING DFT BASED APPROACH

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION:PROGRAM TO COMPUTE CIRCULAR CONVOLUTION OF
    THE TWO SEQUENCES USING DFT BASED APPROACH
clc;
clear;
close;
x=input('Enter the Input sequence=') //x=[1 1 2 2]
```

```
m=length(x);
xl=input('Enter the lower index of input sequence=')
    //0
xh=xl+m-1;
n=xl:1:xh;
subplot(3,1,1);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',n,x);
title('Input Sequence x[n]');
xlabel('Samples n');
ylabel('Amplitude');

h=input('Enter the Impulse response sequence=');//h
    =[1 2 3 4]
l=length(h);
hl=input('Enter the lower index of impulse response
    sequence=');//0
hh=hl+l-1;
g=hl:1:hh;
subplot(3,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',g,h);
title('Impulse Response Sequence h[n]');
xlabel('Samples n');
ylabel('Amplitude');

// for making length of both signals equal
N=max(m,l);
```

```
p=m-l;
if(p>=0) then
 h=[h,zeros(1,p)];
else
 x=[x,zeros(1,-p)];
end
XK=fft(x,-1);
HK=fft(h,-1);
YK=XK.*HK;
y=ifft(YK);
disp(' Circular convolution by DFT is y(n):');
disp(real(y));
nx=xl+hl;
r=nx:length(y)-1;
subplot(3,1,3);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',r,y);
title('Output Response Sequence of Circular Convolution y[n] using DFT');
xlabel('Samples n');
ylabel('Amplitude');

//INPUT:
//Enter the Input sequence=[1 1 2 2]
//Enter the lower index of input sequence=0
//Enter the Impulse response sequence=[1 2 3 4]
//Enter the lower index of impulse response sequence =0

 //OUTPUT:
  //Circular convolution by DFT is y(n):

   //  15.     17.     15.     13.
```

Experiment: 10

WRITE A SCILAB PROGRAM TO COMPUTE BLOCK CONVOLUTION USING OVERLAP ADD METHOD

Scilab code Solution 10.10 PROGRAM TO COMPUTE BLOCK CONVOLUTION USING OVERLAP ADD METHOD

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM TO COMPUTE BLOCK CONVOLUTION
    USING OVERLAP ADD METHOD
clc;
clear;
close;
x=input('Enter the input sequence=') //x=[1 2 -1 2 3
     -2 -3 -1 1 1 2 -1]
m=length(x);
```

```
xl=input('Enter the lower index of input sequence=')
    //0
xh=xl+m-1;
n=xl:1:xh;
subplot(3,1,1);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',n,x);
title('Input Sequence x[n]');
xlabel('Samples n');
ylabel('Amplitude');

h=input('Enter the impulse response sequence=');//h
    =[1 2 3 -1]
l=length(h);
hl=input('Enter the lower index of impulse response
    sequence=');//0
hh=hl+l-1;
g=hl:1:hh;
subplot(3,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',g,h);
title('Impulse Response Sequence h[n]');
xlabel('Samples n');
ylabel('Amplitude');

N=m+l-1;
h1=[h zeros(1,l-1)];
n3=length(h1);
```

```
y=zeros(1,N);

H=fft(h1);
for i=1:l:m
    if i<=(m+l-1) then
        x1=[x(i:i+n3-l) zeros(1,n3-l)];
    else
        x1=[x(i:m) zeros(1,n3-l)];
    end
    x2=fft(x1);
    x3=x2.*H;
    x4=round(ifft(x3));
    if(i==1)
    y(1:n3)=x4(1:n3);
else
    y(i:i+n3-1)=y(i:i+n3-1)+x4(1:n3);
end
end

disp('Output sequence using Overlap add method y(n):
   ');
disp(y(1:N));
nx=xl+hl;
r=nx:length(y)-1;
subplot(3,1,3);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',r,y);
title('Output sequence using Overlap add method y[n]
   ');
xlabel('Samples n');
ylabel('Amplitude');

//INPUT:
```

```
//Enter the input sequence=[1 2 -1 2 3 -2 -3 -1 1 1
   2 -1]
//Enter the lower index of input sequence=0
//Enter the impulse response sequence=[1 2 3 -1]
//Enter the lower index of impulse response sequence
   =0

 //OUTPUT:
 //Output sequence using Overlap add method y(n):


   //        column  1 to 12

   // 1.     4.     6.     5.     2.     11.    0.   - 16.
        - 8.     3.     8.     5.

 //          column 13 to 15

   // 3.   - 5.     1.
```

Experiment: 11

WRITE A SCILAB PROGRAM RO COMPUTE BLOCK CONVOLUTION USING OVERLAP SAVE METHOD.

Scilab code Solution 11.11 PROGRAM TO COMPUTE BLOCK CONVOLUTION USING OVERLAP SAVE METHOD

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM TO COMPUTE BLOCK CONVOLUTION
   USINF OVERLAP SAVE METHOD
clc;
clear;
close;
x=input('Enter the input sequence=') //x=[1 2 -1 2 3
    -2 -3 -1 1 1 2 -1]
m=length(x);
```

```
xl=input('Enter the lower index of input sequence=')//0
xh=xl+m-1;
n=xl:1:xh;
subplot(3,1,1);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',n,x);
title('Input Sequence x[n]');
xlabel('Samples n');
ylabel('Amplitude');

h=input('Enter the impulse response sequence=');//h=[1 2 3 -1]
l=length(h);
hl=input('Enter the lower index of impulse response sequence=');//0
hh=hl+l-1;
g=hl:1:hh;
subplot(3,1,2);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',g,h);
title('Impulse Response Sequence h[n]');
xlabel('Samples n');
ylabel('Amplitude');

N=m+l-1;
h1=[h zeros(1,N-m)];
n3=length(h1);
```

```
y=zeros(1,N);
x1=[zeros(1,n3-l) x zeros(1,n3)];
H=fft(h1);
for i=1:l:N
    y1=x1(i:i+(2*(n3-l)));
    y2=fft(y1);
    y3=y2.*H;
    y4=round(ifft(y3));
    y(i:(i+n3-l))=y4(l:n3);
end
disp('Output sequence using overlap save method Y(n):');
disp(y(1:N));
nx=xl+hl;
r=nx:length(y)-1;
subplot(3,1,3);
a=gca();
a.x_location="origin";
a.y_location="origin";
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot2d3('gnn',r,y);
title('Output sequence using Overlap save method y[n]');
xlabel('Samples n');
ylabel('Amplitude');

//INPUT:
//Enter the input sequence=[1 2 -1 2 3 -2 -3 -1 1 1 2 -1]
//Enter the lower index of input sequence=0
//Enter the impulse response sequence=[1 2 3 -1]
//Enter the lower index of impulse response sequence=0

 //OUTPUT:
 //Output sequence using overlap save method Y(n):
```

```
//          column   1  to  12

//  1.        4.        6.        5.        2.        11.        0.   - 16.
      - 8.        3.        8.        5.

   //       column  13  to  15

//  3.   - 5.        1.
```

Experiment: 12

WRITE A SCILAB PROGRAM TO FIND FFT USING DECIMATION IN TIME(DIT) METHOD.

Scilab code Solution 12.12 PROGRAM TO FIND FFT USING DECIMATION IN TIME METHOD

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
// CAPTION: PROGRAM TO FIND FFT USING DECIMATION IN
   TIME(DIT) METHOD
clc;
clear;
close;
x=input('Enter Input sequence='); // x=[0 1 2 3 4 5
   6 7]  or x= [0 1 2 3]
N=length(x);

s=log2(N); // computing array size

//for sequence size 8
```

```
if (N==8) then
stage1=1;

x=[x(1) x(5) x(3) x(7) x(2) x(6) x(4) x(8)]; // stage1
for stage=1:s
    for index=0:(2^stage):(N-1);//series of butterfly for each stage
      for n=0:(stage1-1);    // create butterlfy and save result
        pos=n+index+1; // index of data sample
        pow=(2^(s-stage))*n; // part of power of complex multiplier
        w=exp((-1*%i)*(2*%pi)*pow/N); // complex multiplier
        a=x(pos)+x(pos+stage1).*w; // 1 st part of butterfly creating operations
        b=x(pos)-x(pos+stage1).*w;// 2nd part of butterfly creating operation
        x(pos)=a; // saving computation of 1st half
        x(pos+stage1)=b; // saving computation of second part

  end
end
stage1=2*stage1; // computing next stage
end

y=x;
disp('FFT of the given input sequence is y(n)=');
disp(y);

// for sequence size -4
 else
    stage1=1;
x=[x(1) x(3) x(2) x(4)]; //  stage1
for stage=1:s
```

```
    for index=0:(2^stage):(N-1);//series of butterfly for each stage
        for n=0:(stage1-1);   // create butterlfy and save result
            pos=n+index+1; // index of data sample
            pow=(2^(s-stage))*n; // part of power of complex multiplier
            w=exp((-1*%i)*(2*%pi)*pow/N); // complex multiplier
            a=x(pos)+x(pos+stage1).*w; // 1 st part of butterfly creating operations
            b=x(pos)-x(pos+stage1).*w;// 2nd part of butterfly creating operation
            x(pos)=a; // saving computation of 1st half
            x(pos+stage1)=b; // saving computation of second part

  end
end
stage1=2*stage1; // computing next stage
end

y=x;
disp('FFT of the given input sequence y(n)=');
disp(y);
end

//INPUT:
//Enter Input sequence=[0 1 2 3 4 5 6 7]

 //OUTPUT:
// FFT of the given input sequence is y(n)=


 //          column 1 to 3

  //   28.   - 4. + 9.6568542i   - 4. + 4.i
```

```
    //       column 4 to 5

//  - 4. + 1.6568542i  - 4.

  //         column 6 to 7

//  - 4. - 1.6568542i  - 4. - 4.i

  //         column 8

 // - 4. - 9.6568542i

 // INPUT:
 //Enter Input sequence=[0 1 2 3]

 //OUTPUT:
 //FFT of the given input sequence y(n)=

//     6.   - 2. + 2.i  - 2.   - 2. - 2.i
```

Experiment: 13

WRITE A SCILAB PROGRAM FOR DESIGNING OF FIR FILTER FOR LOW PASS, HIGH PASS, BANDPASS AND BAND REJECT RESPONSES.

Scilab code Solution 13.13 PROGRAM FOR DESIGNING OF FIR FILTER FOR LOW PASS HIGH PASS BAND PASS AND BAND REJECT RESPONSES

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM FOR DESIGNING OF FIR LOW PASS,
   HIGH PASS, BAND PASS AND BAND REJECT RESPONSES

// Filter length=5 , Order of filter=4 , Window=
   Rectangular
```

```
clc;
clear;
close;
xdel(winsid());
fc1=input('Enter the analog cutoff Frequency in Hz='
    ); // 250
fc2=input('Enter analog higher cutoff frequency in
    Hz=');//600
fs=input('Enter the analog sampling Frequency in Hz=
    ');//2000
M=input('Enter order of filter=');//4
w1=(2*%pi)*(fc1/fs);
w2=(2*%pi)*(fc2/fs);

// FIR LOW PASS FILTER
disp('DESIGNING OF FIR LOW PASS FILTER:');
disp(w1, 'Digital Cutoff Frequency in radians.cycles
    /samples');
wc1=w1/%pi;
disp(wc1,'Normalized digital cutoff frequency in
    cycles/samples');
[wft,wfm,fr]=wfir('lp',M+1,[wc1/2,0],'re',[0,0]);
disp(wft,'Impulse Response of LPF FIR filter:h(n)=')
    ;
//plotting magnitude response of LPF filter
subplot(2,4,1);
a=gca();
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot(2*fr,wfm);
title('Magnitude Response of FIR LPF');
xlabel('Normalized Digital Frequency w');
ylabel('Magnitude |H(w)|');
xgrid(1)
subplot(2,4,2);
a=gca();
```

```
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot(fr*fs,wfm);
title('Magnitude Response of FIR LPF');
xlabel('Analog frequency in Hz f');
ylabel('Magnitude |H(w)|');
xgrid(1)

//FIR HIGH PASS FILTER
disp('DESIGNING OF FIR HIGH PASS FILTER:')
disp(w1, 'Digital Cutoff Frequency in radians.cycles
   /samples');
wc1=w1/%pi;
disp(wc1,'Normalized digital cutoff frequency in
   cycles/samples');
[wft,wfm,fr]=wfir('hp',M+1,[wc1/2,0],'re',[0,0]);
disp(wft,'Impulse Response of HPF FIR filter:h(n)=')
   ;
//plotting magnitude response of HPF filter
subplot(2,4,3);
a=gca();
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot(2*fr,wfm);
title('Magnitude Response of FIR HPF');
xlabel('Normalized Digital Frequency w');
ylabel('Magnitude |H(w)|');
xgrid(1)
subplot(2,4,4);
a=gca();
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
```

```
plot(fr*fs,wfm);
title('Magnitude Response of FIR HPF');
xlabel('Analog frequency in Hz f');
ylabel('Magnitude |H(w)|');
xgrid(1)

// FIR BAND PASS FILTER
disp('DESIGNING OF FIR BAND PASS FILTER:')
disp(w1,'Digital lower cutoff frequency in radians cycles/samples');
disp(w2,'Digital higher cutoff frequency in radians cycles/samples');
wc1=w1/%pi;
wc2=w2/%pi;
disp(wc1,'Normalized digital lower cutoff frequency in cycles/samples');
disp(wc2,'Normalized digital higher cutoff frequency in cycles/samples');
[wft,wfm,fr]=wfir('bp', M+1,[wc1/2,wc2/2],'re',[0,0]);
disp(wft,'Impulse response of Bandpass Filter FIR filter:h(n)=');
//plotting the magnitude Response of HPF FIR filter
subplot(2,4,5);
a=gca();
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot(2*fr, wfm);
xlabel('Normalized Digital Frequency w');
ylabel('Magnitude |H(W)|');
title('Magnitude Response of FIR BPF');
xgrid(1);
subplot(2,4,6);
a=gca();
a.thickness=2;
a.foreground = 5;
```

```
a.font_color = 5;
a.font_style = 5;
plot(fr*fs,wfm);
xlabel('Analog Frequency in Hz f');
ylabel('Magnitude |H(w)|');
title('Magnitude response of FIR BPF');
xgrid(1);

// FIR BAND REJECT FILTER
disp('DESIGNING OF FIR BAND REJECT FILTER:')
disp(w1,'Digital lower cutoff frequency in radians cycles/samples');
disp(w2,'Digital higher cutoff frequency in radians cycles/samples');
wc1=w1/%pi;
wc2=w2/%pi;
disp(wc1,'Normalized digital lower cutoff frequency in cycles/samples');
disp(wc2,'Normalized digital higher cutoff frequency in cycles/samples');
[wft,wfm,fr]=wfir('sb', M+1,[wc1/2,wc2/2],'re',[0,0]);
disp(wft,'Impulse response of Bandreject Filter FIR filter:h(n)=');
//plotting the magnitude Response of HPF FIR filter
subplot(2,4,7);
a=gca();
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot(2*fr, wfm);
xlabel('Normalized Digital Frequency w');
ylabel('Magnitude |H(W)|');
title('Magnitude Response of FIR BRF');
xgrid(1);
subplot(2,4,8);
a=gca();
```

```
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot(fr*fs,wfm);
xlabel('Analog Frequency in Hz f');
ylabel('Magnitude |H(w)|');
title('Magnitude response of FIR BRF');
xgrid(1);

//INPUT:
//Enter the analog cutoff Frequency in Hz=250
//Enter analog higher cutoff frequency in Hz=600
//Enter the analog sampling Frequency in Hz=2000
//Enter order of filter=4

 //OUTPUT:
// DESIGNING OF FIR LOW PASS FILTER:

// Digital Cutoff Frequency in radians.cycles/
   samples

 //    0.7853982

// Normalized digital cutoff frequency in cycles/
   samples

  //   0.25

 //Impulse Response of LPF FIR filter:h(n)=

   // 0.1591549    0.2250791    0.25    0.2250791
        0.1591549

 //DESIGNING OF FIR HIGH PASS FILTER:

// Digital Cutoff Frequency in radians.cycles/
   samples
```

```
 //   0.7853982

//Normalized digital cutoff frequency in cycles/samples

  // 0.25

//Impulse Response of HPF FIR filter:h(n)=

// - 0.1591549  - 0.2250791     0.75  - 0.2250791  - 0.1591549

//DESIGNING OF FIR BAND PASS FILTER:

//Digital lower cutoff frequency in radians cycles/samples

  // 0.7853982

//Digital higher cutoff frequency in radians cycles/samples

  // 1.8849556

//Normalized digital lower cutoff frequency in cycles/samples

  // 0.25

//Normalized digital higher cutoff frequency in cycles/samples

  // 0.6

//Impulse response of Bandpass Filter FIR filter:h(n)=

```

```
 //- 0.2527039     0.0776516     0.35     0.0776516   -
    0.2527039

//DESIGNING OF FIR BAND REJECT FILTER:

//Digital lower cutoff frequency in radians cycles/
   samples

  // 0.7853982

//Digital higher cutoff frequency in radians cycles
   /samples

  // 1.8849556

//Normalized digital lower cutoff frequency in
   cycles/samples

  // 0.25

//Normalized digital higher cutoff frequency in
   cycles/samples

  // 0.6

//Impulse response of Bandreject Filter FIR filter:
   h(n)=

   //0.2527039  - 0.0776516     0.65  - 0.0776516
         0.2527039
```

Experiment: 14

WRITE A SCILAB PROGRAM TO DESIGN DIGITAL IIR BUTTERWORTH LOW PASS FILTER.

Scilab code Solution 14.14 PROGRAM TO DESIGN DIGITAL IIR BUTTERWORTH LOW PASS FILTER

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM TO DESIGN BUTTERWORTH LOW PASS
   FILTER
clc;
clear;
close;
xdel(winsid());
fc=input('Enter cutoff freq in Hz fc='); //1000
fs=input('Enter sampling freq in Hz fs='); //10000
```

```
N=input('Enter order of Butterworth filter N=');//4
Fp=2*fc/fs;
[Hz]=iir(N,'lp','butt',[Fp/2,0],[0,0]);
[Hw,w]=frmag(Hz,256);
subplot(2,1,1);
a=gca();
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot(2*w,abs(Hw));
title('Magnitude Response of IIR LPF');
xlabel('Normalized Digital frequency w');
ylabel('Magnitude |H(w)|');
xgrid(1);
subplot(2,1,2);
a=gca();
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot(2*w*fs,abs(Hw));
title ('Magnitude Response of IIR LPF');
xlabel('Analog Frequency in Hz f');
ylabel('Magnitude |H(w)|');
xgrid(1);

//INPUT:
//Enter cutoff freq in Hz fc=1000
//Enter sampling freq in Hz fs=10000
//Enter order of Butterworth filter N=4
```

Experiment: 15

WRITE A SCILAB PROGRAM TO DESIGN DIGITAL IIR CHEBYSHEW FILTER.

Scilab code Solution 15.15 PROGRAM TO DESIGN DIGITAL IIR CHEBYSHEW FILTER

```
// VERSION: Scilab: 5.4.1
// OS: windows 7
//CAPTION: PROGRAM TO DESIGN DIGITAL CHEBYSHEV IIR
   FILTER
clc;
clear;
close;
wp=input('Enter the Digital Pass Band edge Frequency
   ='); //0.2*%pi
ws=input('Enter the Digital Stop Band edge Frequency
   =');//0.6*%pi
t=input('Sampling Interval=');//1
```

```
del1=input('Enter the Pass Band Ripple=');//0.8
del2=input('Enter the Stop Band Ripple=');//0.2
disp(wp,'Wp=');
disp(ws,'Ws=');
del=sqrt(((1/del2)^2)-1);
disp(del,'Delta=');
epsilon=sqrt(((1/del1)^2)-1);
disp(epsilon,'Epsilon=');
N=(acosh(del/epsilon))/(acosh(ws/wp));
N=ceil(N);
disp(N,'N=');
wc=wp/((((1/del1)^2)-1)^(1/(2*N)));
[pols,gn]=zpch1(N,epsilon,wp);
hs=poly(gn,'s','coeff')/real(poly(pols,'s'));
z=poly(0,'z');
hz=horner(hs,((2/t)*((z-1)/(z+1))));
hw=frmag(hz(2),hz(3),512); // freq. response for 512
    points
w=0:%pi/511:%pi;
a=gca();
a.thickness=2;
a.foreground = 5;
a.font_color = 5;
a.font_style = 5;
plot(w/%pi,abs(hw));
xgrid(1);
title('Magnitude Response of Digital Chebyshew LPF
    IIR Filter');
xlabel('Normalized digital Frequency');
ylabel('Magnitude in db');

//INPUT:
//Enter the Digital Pass Band edge Frequency=0.2*%pi
//Enter the Digital Stop Band edge Frequency=0.6*%pi
//Sampling Interval=1
//Enter the Pass Band Ripple=0.8
//Enter the Stop Band Ripple=0.2

```

```
//OUTPUT:
//Wp=

   // 0.6283185

//Ws=

   // 1.8849556

//Delta=

   // 4.8989795

// Epsilon=

  //   0.75

//N=

  2.
```