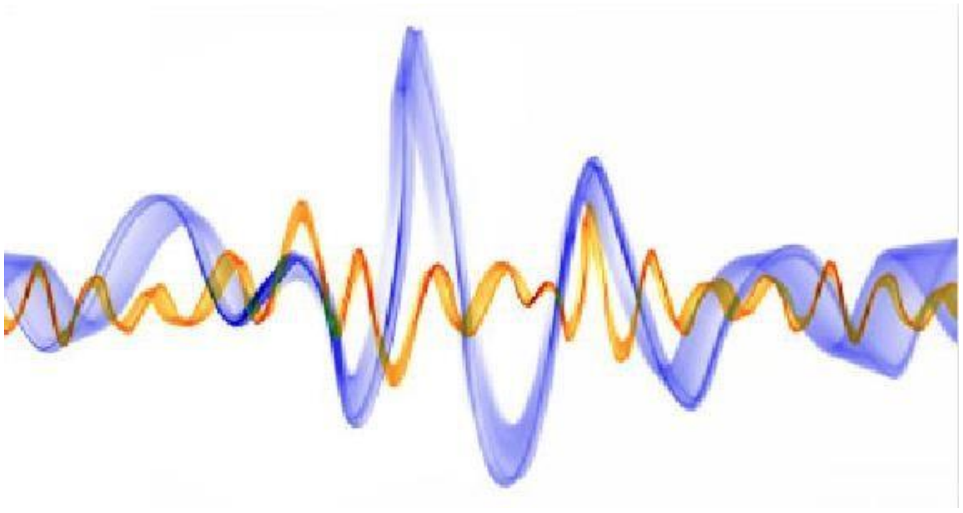
Basic Signal Processing Laboratory Manual

III Semester (21ET33)



DAYANANDA SAGAR COLLEGE OF ENGINEERING

Accredited by National Assessment & Accreditation Council (NAAC) with 'A' Grade

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi & ISO 9001:2008 Certified)

Shavige Malleswara Hills, Kumaraswamy Layout Bengaluru-560078

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATON ENGINEERING

Vision of the Institute

To impart quality technical education with a focus on Research and Innovation emphasizing on Development of Sustainable and Inclusive Technology for the benefit of society.

Mission of the Institute

- To provide an environment that enhances creativity and Innovation in pursuit of Excellence.

- To nurture teamwork in order to transform individuals as responsible leaders and entrepreneurs.

- To train the students to the changing technical scenario and make them to understand the importance of Sustainable and Inclusive technologies.

DAYANANDA SAGAR COLLEGE OFENGINEERING

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING, BENGALURU-560078

Vision of the Department

To disseminate quality technical education for achieving Academic Excellence through focused research and innovation encompassing contemporary technologies trends in electronic devices, telecommunication systems and standards for societal needful applications, of country in specific and global at broader perspective.

MISSION

1. ***By adopting quality curriculum design, associated with standard teaching learning process and assessment techniques; on par with standards for portability of graduates globally.***
2. ***By educating students towards contemporary technologies through consistent interaction with the leading industries based on the current socio-economic conditions of the society.***
3. ***By inculcating research attitude among graduates through lifelong learning in pursuit of academic excellence and career enrichment.***

Program Educational Objectives (PEOs)

1. **PEO1:** The graduate envisages the vision of the department, on engineering needs of the country at large and industry in specific and thereby contribute services with perspective of self-sustenance through higher studies, and research in the program domain.

2. **PEO2:** The graduate enduring the optimally designed curriculum, demonstrates ability to analyze Scientific, Mathematical perspectives of Electronics and Telecommunication Engineering and apply its knowledge with multidisciplinary approach for solving real life problems for a successful career.

3. **PEO3:** The graduate possesses professional attitude for career building in the society, ability to work and lead a team with highest integrity and ethical values for managing large corporate projects through suitable research and development.

4. **PEO4:** The graduate demonstrates lifelong learning to upgrade knowledge of emerging technologies through training, interaction with professional bodies for finding sustainable solution to changing scenarios of the technology frontiers, emphasizing on security and safety concern of individual and global environment.

Program Specific Outcomes (PSO's)

PSO1: To apply fundamental concepts of electrical, electronic devices and circuits to communication theory and systems on par with the electronics and telecommunication industry domain.

PSO2: To apply the core merits of embedded systems, signal processing and network protocols for electronics and telecommunication standards.

PSO3: To design secured cognitive collaborative solutions related to electronics and telecommunication systems.

DAYANANDA SAGAR COLLEGE OF ENGINEERING
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING, BENGALURU-560078

BASIC SIGNAL PROCESSING LAB
(SYLLABUS)
III SEMESTER B. E (ETE)

Sub Code : 21ET33 (IPCC)	**CIE: 50**

Course Objectives:

1. Preparation: To prepare students with fundamental knowledge/ overview in the field of
   Signal Processing and Familiarization with the concept of Vector spaces and qualitative insight into applications in communications.
2. Core Competence:

a) To equip students with a strong foundation of Signal Processing:
b) by delivering the basics of quantitative parameters for Matrices & Linear Transformations,
c) the mathematical description of discrete time signals and systems,
d) analyzing the signals in time domain using convolution techniques,
e) classifying signals into different categories based on their properties,
f) analyzing Linear Time Invariant (LTI) systems in time and transform domains

Syllabus:

Practical Component of the IPCC	
Sl. No.	**Experiments**
1.	a. To create and modify a vector (array). b. To create and modify a matrix.
2.	Perform basic operations on matrix.
3.	Generation of various signals and sequences
4.	Operations on signals and sequences(Dependent and independent Variable)
5.	Verification of Sampling Theorem
6.	Solving a given difference equation.
7.	Auto Correlation and Cross Correlation
8.	To perform convolution of two given sequences.
9.	a. To perform verification of commutative property of linear convolution. b. To perform verification of distributive property of linear convolution. c. To perform verification of associative property of linear convolution.
10.	To compute impulse response.
11.	To find Z-transform and inverse Z-transform of a sequence.
12.	To Plot Poles and Zeros on Z-transform domain

Course outcomes (Course Skill Set) at the end of the course the student will be able to:

1. Able to understand and analyze various types of signals, systems and perform different operations on signals.
2. Able to analyze LTI system by convolution technique.
3. Able to represent input- output relationships for Linear Time Invariant systems by solving difference equations.
4. Able to model and distinguish different implementation techniques of Continuous Time and Discrete Time systems
5. Able to analyze Continuous Time and Discrete Time systems by applying Z transforms and its properties
6. Able to demonstrate the Usage of open source tools for analysis and interpretation of signals and systems

DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING, BENGALURU-560078

Basic Signal Processing (Integrated Lab)

LABORATORY

1. To create and modify a vector (array) and to create and modify a matrix.

2. Perform basic operations on matrix.

3. Generation of various signals and sequences

4. Operations on signals and sequences (Dependent and independent Variable)

5. Verification of Sampling Theorem

6. Solving a given difference equation.

7. Auto Correlation and Cross Correlation

8. To perform convolution of two given sequences.

9. a. To perform verification of commutative property of linear convolution.

b. To perform verification of distributive property of linear convolution.

c. To perform verification of associative property of linear convolution.

10. To compute impulse response.

11. To find Z-transform and inverse Z-transform of a sequence.

12. To Plot Poles and Zeros on Z-transform domain

DAYANANDA SAGAR COLLEGE OF ENGINEERING
DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION ENGINEERING, BENGALURU-560078

DO's

- Adhere and follow timings, proper dress code with appropriate footwear.
- Bags, and other personal items must be stored in designated place.
- Come prepare with the viva, procedure, and other details of the experiment.
- Secure long hair, loose clothing & know safety and emergency procedures.
- Do check for the correct ranges/rating and carry one meter/instrument at a time
- Inspect all equipment/meters for damage prior to use
- Conduct the experiments accurately as directed by the teacher.
- Immediately report any sparks/accidents/injuries/any other untoward incident to the faculty/instructor.
- Handle the apparatus/meters/computers gently and with care
- In case of an emergency or accident, follow the safety procedure.
- Switch OFF the power supply after completion of experiment

DONT's

- The use of mobile/ any other personal electronic gadgets is prohibited in the laboratory.
- Do not make noise in the Laboratory & do not sit on experiment table.
- Do not make loose connections and avoid overlapping of wires
- Don't switch on power supply without prior permission from the concerned staff.
- Never point/touch the CRO/Monitor screen with the tip of the open pen/pencil/any other sharp object.
- Never leave the experiments while in progress.
- Do not insert/use pen drive/any other storage devices into the CPU
- Do not leave the Laboratory without the signature of the concerned staff in observation book

EXPERIMENT NO 1: To create and modify a vector array and a matrix

Aim: **a.** To create and modify a vector array
b. To create and modify a matrix.

Algorithm:

- MATLAB treats all variables as matrices. For our purposes a matrix can be thought of as an array, in fact, that is how it is stored.
- Vectors are special forms of matrices and contain only one row OR one column.
- Scalars are matrices with only one row AND one column. A matrix with only one row AND one column is a scalar.
- A scalar can be treated in MATLAB as follows:
- A matrix with only one row is called a row vector. A matrix with only one column is called a column vector.

MATLAB Program:

```
clc;
Clear all;
Close all;

% A row vector can be created in MATLAB as follows:
 a = [12, 14, 63];
 b= [13, 12,-6];

% A column vector can be created in MATLAB as follows:
 a1= [14; -32; 19];
b1 = [13; 45; -2];

%Addition of two vectors
 y=a+b;
y1=a1+b1;
disp('the addition value is');
disp(y);
disp(y1);

%Subtraction of two vectors
x=a-b;
x1=a1-b1;
disp('the subtraction value is');
disp(x);
disp(x1);
```

```
%Multiplication of two vectors
z=a*b1;
z1=a1*b;
disp('the multiplication value is');
disp(z);
disp(z1);


%Division of a vector by a scalar value
r=a/2;
disp('the matrix division by a value 2 is');
disp(r);

%Transpose of a vector
p=a';
disp('the transpose of matrix a is');
disp(p);
```

Results:

```
the addition value is
    25    26    57

    27
    13
    17

the subtraction value is
    -1     2    69

     1
   -77
    21

the multiplication value is
   660

   182   168   -84
  -416  -384   192
   247   228  -114
```

```
the matrix division by a value 2 is
    6.0000    7.0000   31.5000

the transpose of matrix a is
    12
    14
    63
```

EXPERIMENT NO 2: Create a matrix and perform basic operations on matrix.

Aim: Perform basic operations on matrix.

Algorithm:

- MATLAB treats all variables as matrices. For our purposes a matrix can be thought of as an array, in fact, that is how it is stored.
- Vectors are special forms of matrices and contain only one row OR one column.
- Scalars are matrices with only one row AND one column. A matrix with only one row AND one column is a scalar.

MATLAB Program:

```
clc;
clear all;
close all;
 a=input ('enter the first matrix a');
b=input ('enter the second matrix b');

%Addition of two matrices
 y=a+b;
disp('the addition value is');
disp(y);

%Subtraction of two matrices
x=a-b;
disp('the subtraction value is');
disp(x);

%Multiplication of two matrices
z=a*b;
disp('the multiplication value is');
disp(z);

%Multiplication of two matrices element by element
z=a.*b;
disp('the multiplication value is');
disp(z);
```

```
%Division of a matrix by a scalar value
r=a/2;
disp('the matrix division by a value 2 is');
disp(r);

%Transpose of a matrix
p=a';
disp('the transpose of matrix a is');
disp(p);

%Inverse of matrix
q=inv(a);
disp('the inverse of matrix a is');
disp(q);

%Determinant of a matrix
r=det(a);
disp('the determinant of matrix a is');
disp(r);

% Creating a matrix of size 3X3
A1=[1,2,3;4,5,6;7,8,9]

%Here we extract column 2 of the matrix and make a column vector:
col_two=A1 (:, 2)

%Here we extract row 2 of the matrix and make a row vector. Note that the 2:2
specifies the second row and the 1:3 specifies which columns of the row.
Rowvec=A1 (2: 2, 1:3)
```

Results:

```
enter the first matrix a
[1 2 3; 4 5 6; 7 8 6]
enter the second matrix b
[2 4 8; 3 4 6; 7 6 2]
the addition value is
     3     6    11
     7     9    12
    14    14     8
```

```
the subtraction value is
    -1    -2    -5
     1     1     0
     0     2     4

the multiplication value is
    29    30    26
    65    72    74
    80    96   116

the multiplication value is
     2     8    24
    12    20    36
    49    48    12

the matrix division by a value 2 is
    0.5000    1.0000    1.5000
    2.0000    2.5000    3.0000
    3.5000    4.0000    3.0000

the transpose of matrix a is
     1     4     7
     2     5     8
     3     6     6

the inverse of matrix a is
   -2.0000    1.3333   -0.3333
    2.0000   -1.6667    0.6667
   -0.3333    0.6667   -0.3333
Rowvec=A1(2: 2, 1:3)
A1 =

     1     2     3
     4     5     6
     7     8     9

col_two =

     2
     5
     8

Rowvec =

     4     5     6
```

EXPERIMENT NO 3: Generation of various signals and sequences

Aim: Generation On Various Signals and Sequences (Periodic and Aperiodic), Such As Unit Impulse, Unit Step, Square, Saw tooth, Triangular, Sinusoidal, Ramp, Sinc

MATLAB Program:

```
 %discrete unit impulse sequence generation
 clc;
close all;
n=- 3:4;
x=[n==0];
 subplot (4,4,1),stem(n,x);
title ('discrete unit impulse');

%continuous unit impulse signal generation
t=-3:.25:4;
x=[t==0];
subplot (4, 4, 2), plot (t,x);
title('continuous unit impulse');
grid;

% discrete unit step sequence generation
n=-3:4;
y=[n>=0];
subplot(4,4,3),stem(n,y);
 xlabel('n')
ylabel('amplitude');
title('discrete unit step');grid;



% continuous unit step signal generation
t=- 3:.025:4;
y=[t>=0];
 subplot(4,4,4),
plot(t,y);
xlabel('t');
ylabel('amplitude');
title('continuous unit step');
grid;

% continuous square wave generator
t = -5:.01:5;
x = square(t);
```

```
subplot(4,4,5),plot(t,x);
xlabel('Time (sec)');
ylabel('Amplitude');
title('continuous Square Periodic Wave');
grid;

% discrete square wave generator
 n = - 5:5;
x = square(n);
subplot(4,4,6),stem(n,x);
xlabel('Time (sec)');
ylabel('Amplitude');
title('discrete Square Periodic Wave');grid;

% continuous saw tooth wave generator
t = - 5:.01:5;
x = sawtooth(t);
subplot(4,4,7),plot(t,x);
 xlabel('Time (sec)');
ylabel('Amplitude');
 title('continuous Saw tooth Periodic Wave');grid;

% discrete saw tooth sequence generator
n = -5:5;
 x = sawtooth(n);
subplot(4,4,8),stem(n,x);
xlabel('Time (sec)');
ylabel('Amplitude');
title('discrete Saw tooth Periodic Wave');grid;

% continuous sinusoidal signal generator
t = - 5:.01:5;
x = sin(t);
subplot(4,4,9),plot(t,x);
 xlabel('Time (sec)');
ylabel('Amplitude');
title('continuous Sinusoidal Periodic Wave');
grid;
```

```
% discrete sinusoidal sequence generator
n = - 5:5;
x = sin(n);
subplot(4,4,10),stem(n,x);
xlabel('Time (sec)');
ylabel('Amplitude');
title('discrete Sinusoidal Periodic Wave');
grid


% continuous ramp signal generator
t = 0:.01:5;
x=2*t;
subplot(4,4,11),plot(t,x);
xlabel('Time (sec)');
ylabel('Amplitude');
 title('continuous ramp Aperiodic Wave');
grid;

% discrete ramp sequence generator
n = 0:5;
x=2*n;
subplot(4,4,12),stem(n,x);
 xlabel('Time (sec)');
ylabel('Amplitude');
title('discrete ramp APeriodic sequence');
grid

% continuous sinc signal generator
t = -5:.01:5;
x = sinc(t);
subplot(4,4,13),plot(t,x);
xlabel('Time (sec)');
ylabel('Amplitude');
title('continuous Sinc APeriodic Wave');
grid
```

```
% discrete sinc sequence generator
n = - 5:.1:5;
x = sinc(n);
subplot(4,4,14),stem(n,x);
xlabel('Time (sec)');
ylabel('Amplitude');
title('discrete Sinc APeriodic Wave');
grid;

%To generate a Aperiodic rectangular pulse
 t=- 5:0.01:5;
pulse = rectpuls(t,2);
%pulse of width 2 time units
 subplot(4,4,15),plot(t,pulse);
 xlabel('Time (sec)');
ylabel('Amplitude');
title('continuous rectangular pulse APeriodic Wave');
grid;

%To generate a Aperiodic signum function
 t=- 5:0.01:5;
pulse = sign(t);
%pulse of width 2 time units
subplot(4,4,16),plot(t,pulse);
xlabel('Time (sec)');
ylabel('Amplitude');
title ('continuous signum APeriodic Wave');
grid;
```

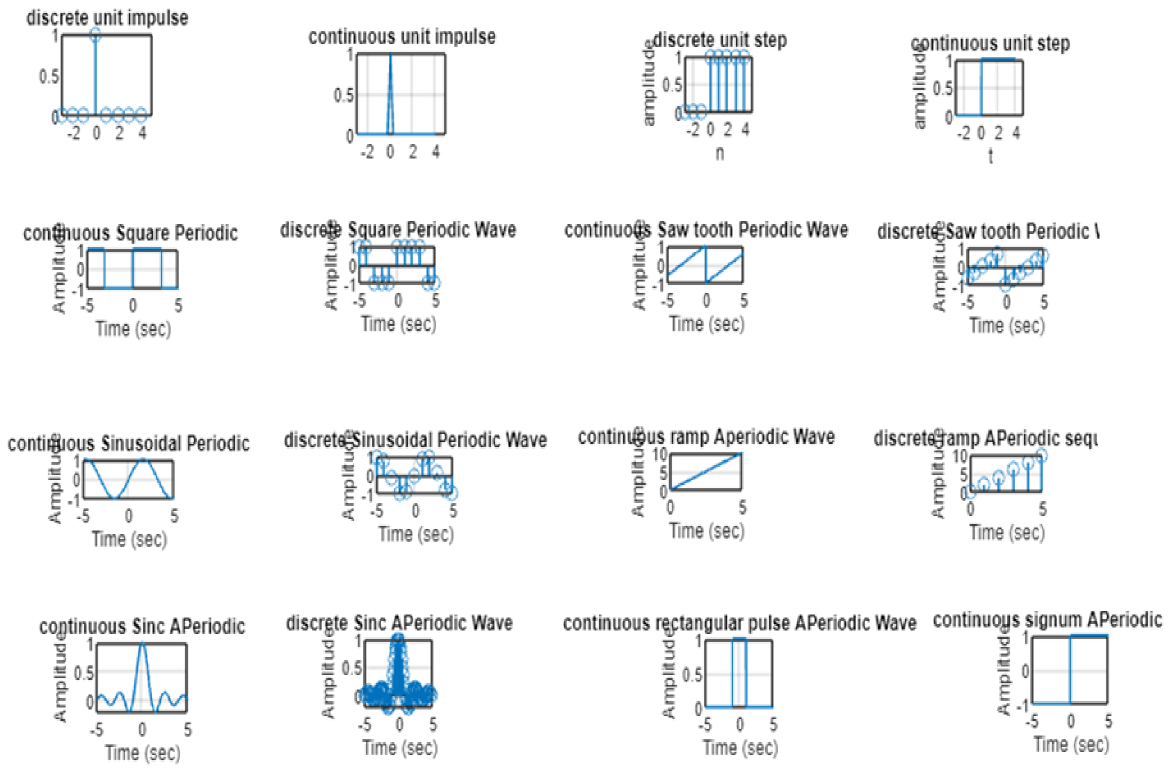


Figure 3.1: Generation of Various types of signals

EXPERIMENT NO 4: Operations on signals and sequences (Dependent and independent Variable)

Aim: To performs functions on signals and sequences such as addition, multiplication, scaling, shifting, folding, computation of energy and average power.

Algorithm:

MATLAB Program:

Operations on signals such as addition, subtraction, multiplication, amplitude scaling

```
clc;
clear all;
close all;
%plot 2hz sinusodial signal
t=0:.01:1;
f1=2;
A=8;
s1=A*sin(2*pi*f1*t);
figure, subplot(2,1,1),plot(t,s1,'r');
grid;
xlabel('---time');
ylabel('amplitude');
title('signal 1');

%plot 10hz sinusodial signal
t=0:.01:1;
f2=10;
A=6;
s2=A*sin(2*pi*f2*t);
subplot(2,1,2),plot(t,s2,'g');
grid;
xlabel('---time');
ylabel('amplitude');
 title('signal 2');


%signal
y=s1+s2;
figure, subplot(2,1,1),
plot(t,y);
grid;
xlabel('-->time');
ylabel('amplitude');
title('the summed signal s1+s2');

%signal subtraction
 y=s1-s2;
subplot(2,1,2),plot(t,y,'r');
grid;
xlabel('-->time');
ylabel('amplitude');
title('the subtracted signal s1-s2');

%signal multiplication element by element
y=s1.*s2;
figure, plot(t,y,'g') ;
xlabel('-->time');
 ylabel('amplitude') ;
title('the multiplied signal s1.*s2');

%Amplitude scaling of signal s1
alpha1= 3;
y=alpha1*s1;
 figure, subplot(2,1,1),plot(t,y);
xlabel('-->time');
ylabel('amplitude') ;
title('the signal s1 amplitude is multiplied by alpa');

%amplitude scaling of signal s2
 alpha2=1/2;
y=alpha2*s2;
subplot(2,1,2),plot(t,y) ;
 xlabel('-->time');
ylabel('amplitude') ;
 title ('the signal s2 is amplitude scaled by 1/alpha');
```

Result:

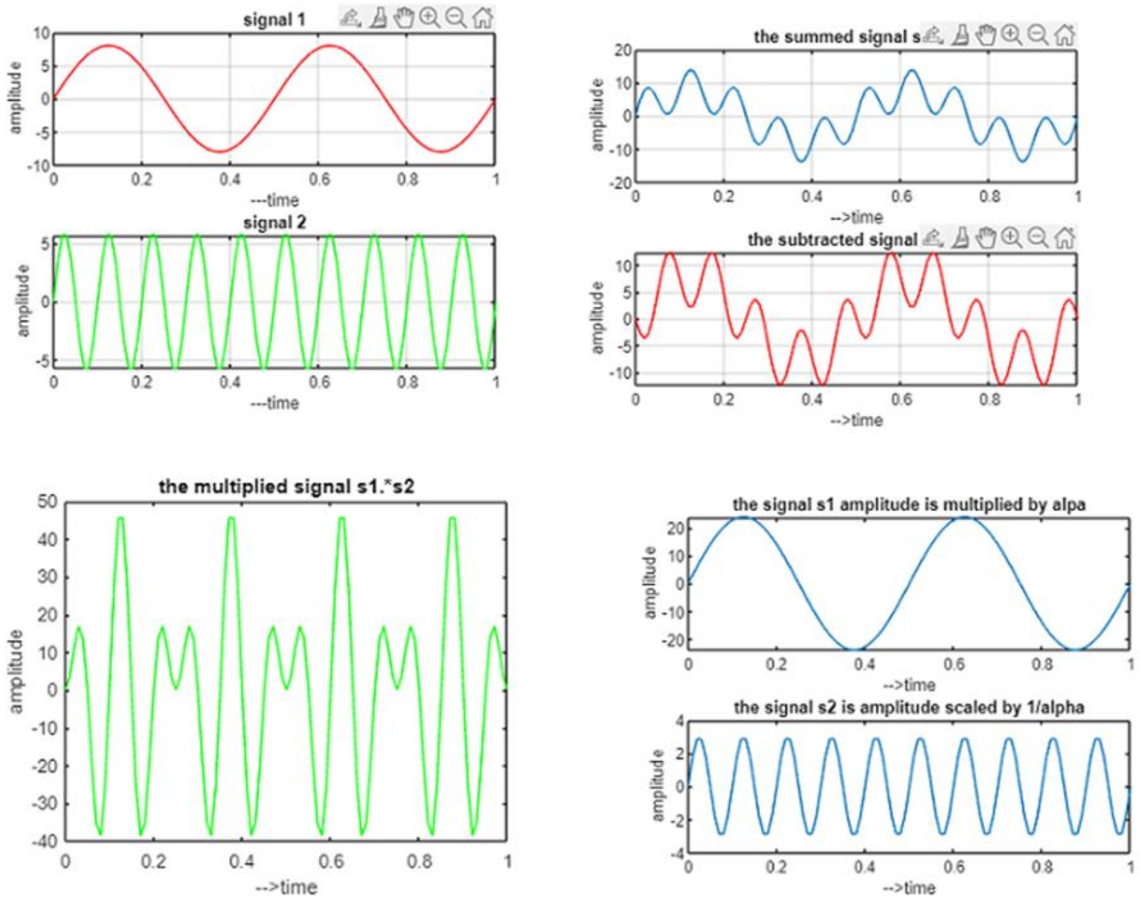


Figure 4.1: Addition, Subtraction, Multiplication, Amplitude scaling of signals

```
clc;
clear all;
close all;
%plot ramp signal
t=[0:0.01:5];
x=1*t;
subplot(2,2,1),plot(t,x);grid;axis([0 5 0 5]);
title('ramp signal');xlabel('t');ylabel('amplitude');

%plot advanced ramp signal
t= [-3:0.01:2]
x=1*(t+3); subplot (2, 2, 2), plot (t,x);grid;
axis([-3 2 0 5]);
title('advanced ramp signal');xlabel('t');ylabel('amplitude');

%plot delayed ramp signal
```

```
t= [3:0.01:8];
x=1*(t-3); subplot (2, 2, 3), plot (t,x);grid;axis([3 8 0 5]);
title('delayed ramp signal');xlabel('t');ylabel('amplitude');

%plot folding or time reversal operation t= [-5:.01:0];
x=1*-t;
subplot (2, 2,4),plot(t,x);grid;
title ('time reversal operation');xlabel('t');ylabel('amplitude');

%plot folding or time reversal operation t= [-5:.01:0];
x=1*-t;
subplot(2,2,4),plot(t,x);grid;
title ('time reversal operation');xlabel('t');ylabel('amplitude');
```

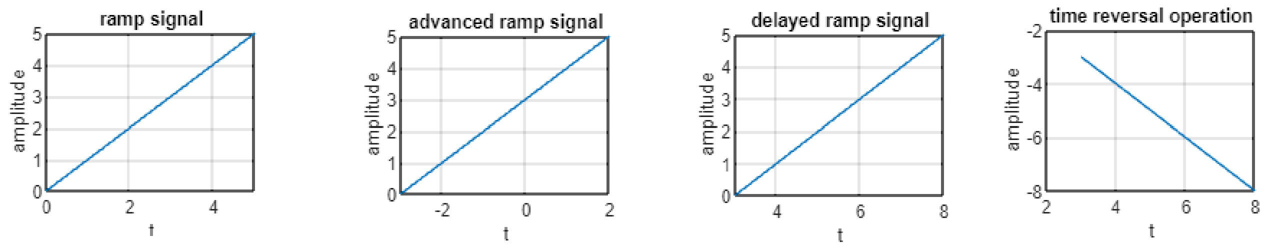


Figure 4.2: Time shifting and Reversal operations

```
%plot time scaling
t=0:0.01:1;
f1=2;
A=8;
s1=A*sin(2*pi*f1*t);
figure;
subplot (3,1,1),plot(t,s1);grid;
title('original signal');xlabel('t');ylabel('amplitude');


%plot compressed signal
t=0:0.01:1;
f1=2;
A=8;
s1=A*sin(2*pi*f1*(2*t));
subplot(3,1,2),plot(t,s1,'g');
grid; title('compressed signal');
xlabel('t');
ylabel('amplitude');
```

```
 %plot('expanded signal
t=0:0.01:1;
f1=2;
A=8;
s1=A*sin(2*pi*f1*(0.5*t));
subplot(3,1,3),plot(t,s1,'r');
grid;
title('expanded signal');
xlabel('t');
ylabel('amplitude');
```

Results:

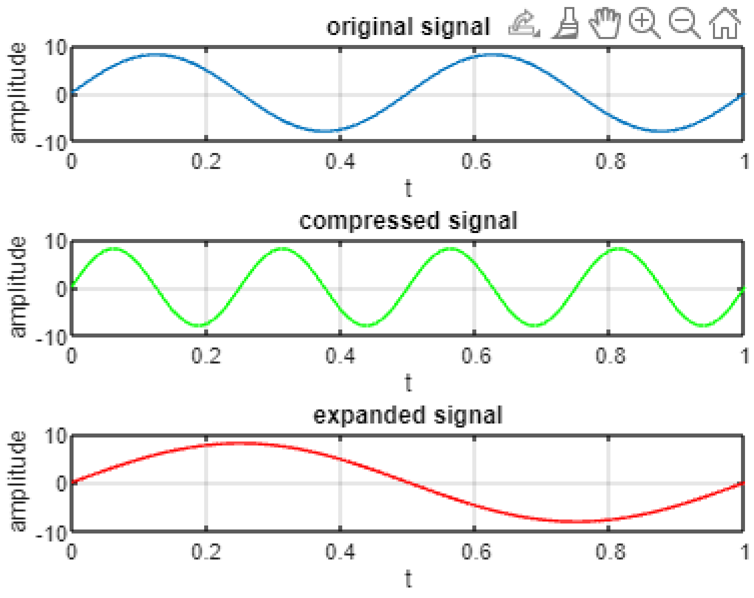


Figure 4.3: Expansion and Compression of signals

EXPERIMENT NO 5: To verify Sampling theorem for a signal of given frequency.

Algorithm:

1. **Input the desired frequency fd (for which sampling theorem is to be verified).**
2. **Generate an analog signal xt of frequency fd for comparison.**
3. **Generate oversampled, Nyquist & under sampled discrete time signals.**
4. **Plot the waveforms and hence prove sampling theorem.**

MATLAB Program:

```
tfinal=0.05;
t=0:0.00005: tfinal;
fd=input('Enter analog frequency');
%define analog signal for comparison
xt=sin(2*pi*fd*t);
%simulate condition for under sampling i.e., fs1<2*fd
fs1=1.3*fd;
%define the time vector
n1=0:1/fs1: tfinal;
%Generate the under sampled signal
xn=sin(2*pi*n1*fd);
%plot the analog & sampled signals
 subplot(3,1,1);
plot(t,xt,'b',n1,xn,'r*-');
title('under sampling plot');
%condition for Nyquist plot
fs2=2*fd;
n2=0:1/fs2:tfinal;
xn=sin(2*pi*fd*n2);
subplot(3,1,2);
 plot(t,xt,'b',n2,xn,'r*-');
title('Nyquist plot');
%condition for oversampling
fs3=5*fd;
n3=0:1/fs3:tfinal;
xn=sin(2*pi*fd*n3);
subplot(3,1,3);
plot(t,xt,'b',n3,xn,'r*-');
title('Oversampling plot');
xlabel('time');
ylabel('amplitude');
legend(‘analog’, ‘discrete')
```

Result:

```
Enter analog frequency 200
```

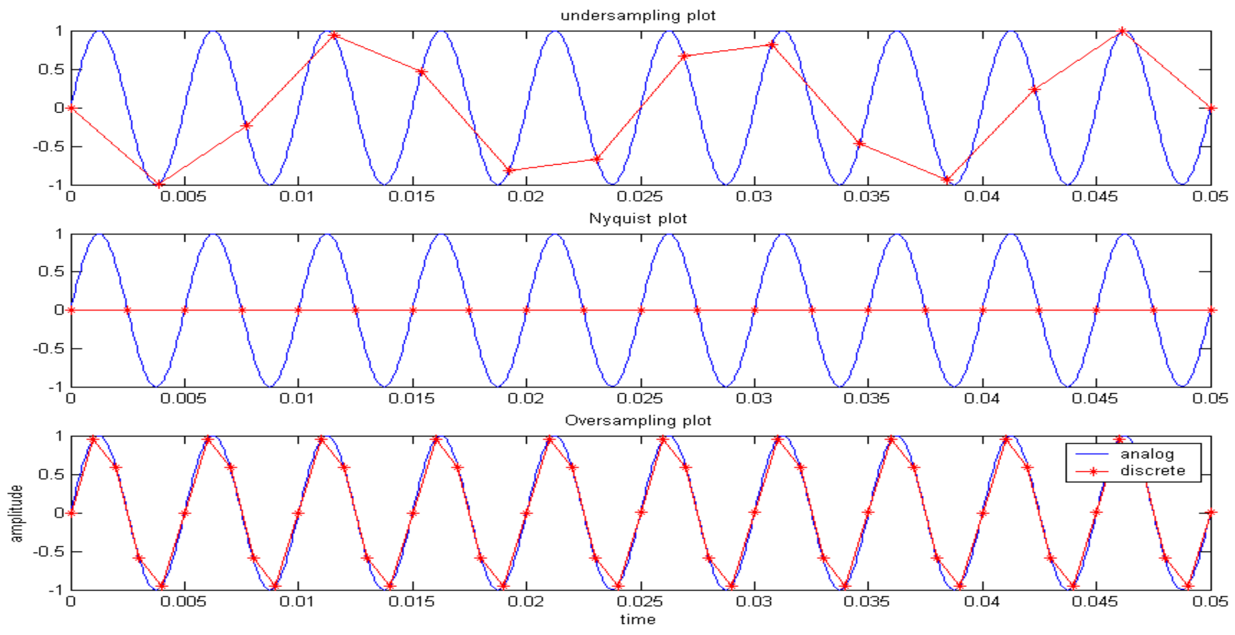


Figure 5.1: Plots of sampled Sine Wave of Frequency 200Hz

Sampling at the Nyquist rate results in samples sin(πn) which are identically zero, i.e., we are sampling at the zero crossing points and hence the signal component is completely missed. This can be avoided by adding a small phase shift to the sinusoid. The above problem is not seen in cosine waveforms (except cos(90n)). A simple remedy is to sample the analog signal at a rate higher than the Nyquist rate.

The Fig5.2 shows the result due to a cosine signal (x1=cos(2*pi*fd*n1);

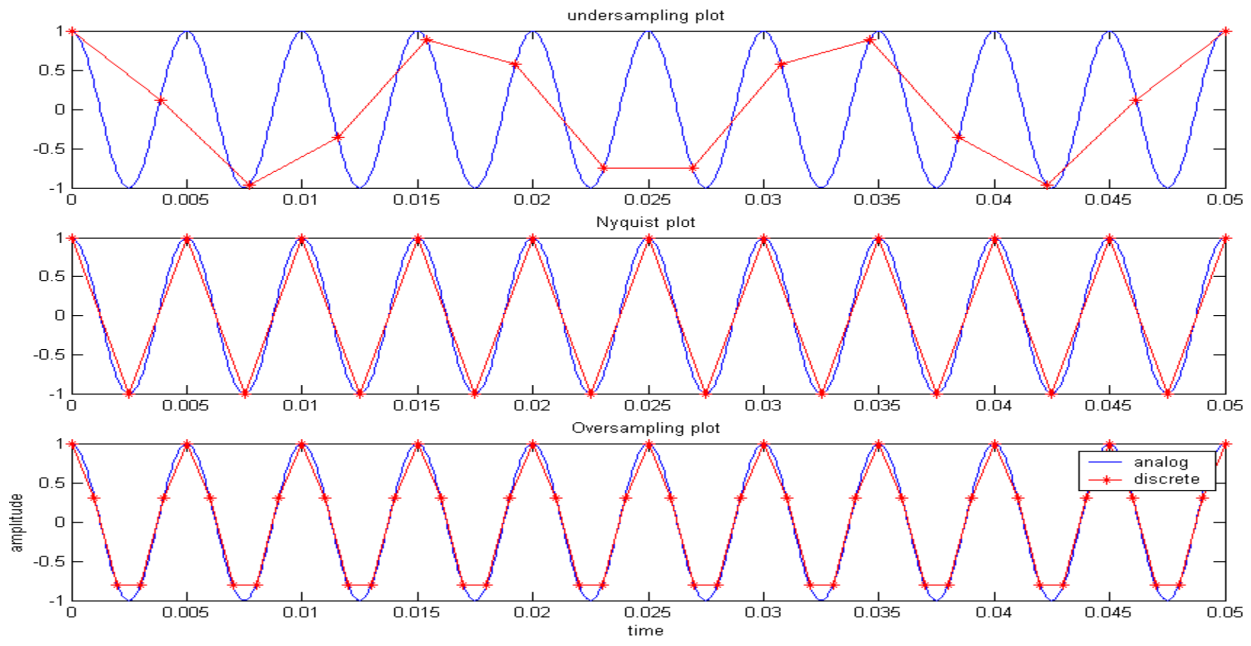


Figure 5.2: Plots of sampled Cosine Wave of frequency 200Hz

EXPERIMENT NO 6: Solving a given difference equation

Aim: To obtain the impulse response/step response of a system described by the given difference equation

Algorithm:

1. **Input the two sequences as a and b representing the coefficients of y and x.**
2. **If IIR response, then input the length of the response required (say 100, which can be made constant).**
3. **Compute the output response using the “filter” command.**
4. **Plot the input sequence & impulse response (and also step response, etc if required).**

Given Problem

1) **Difference equation y(n) - y(n-1) + 0.9y(n-2) =x(n);**
   Calculate impulse response h(n) and also step response at n=0,…..,100

MATLAB Program:

```
%To find Impulse Response
N=input('Length of response required=');
b=[1];  %x[n] coefficient
a=[1,-1,0.9];   %y coefficients
%impulse input
x=[1,zeros(1,N-1)];
%time vector for plotting
 n=0:1:N-1;
%impulse response
h=filter(b,a,x);

%plot the waveforms
subplot(2,1,1);
stem(n,x);
title('impulse input');
xlabel('----->   n');
ylabel('---->  x(n)');
subplot(2,1,2);
stem (n,h);
title ('impulse response'); xlabel('--->   n');
ylabel('---> h(n)');
```

Result: 'Length of response required = 100

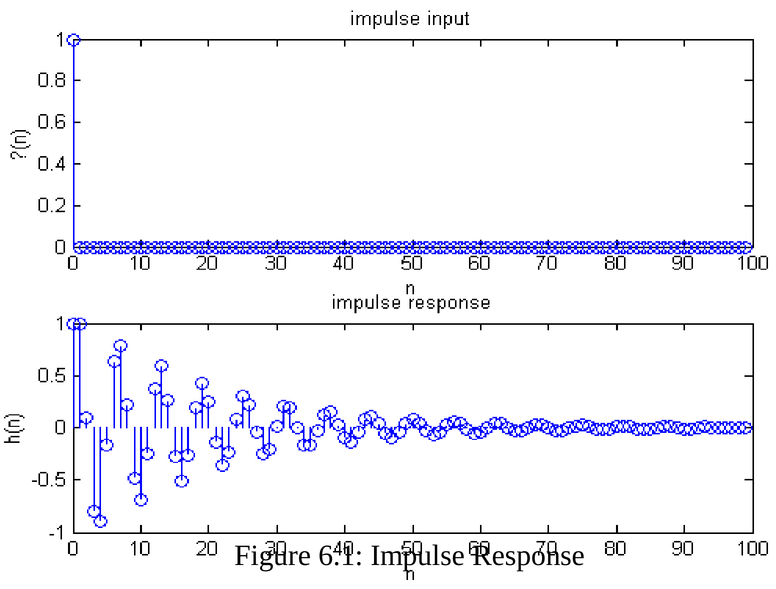


Figure 6.1: Impulse Response

```
%To find step response
clc;
clear;

 N=input('Length of response required=');
 b=[1];                 %x[n] coefficient
a=[1,-1,0.9];       %y coefficients
x=[ones(1,N)]; %step input
 n=0:1:N-1;             %time vector for plotting
y=filter(b,a,x);        %step response
 %plot the waveforms
subplot(2,1,1);
stem(n,x);
title('step input');
xlabel('n');
ylabel('u(n)');
 subplot(2,1,2);
stem(n,y);
title('step response');
xlabel('n');
ylabel('y(n)');
```

Result:
Length=100

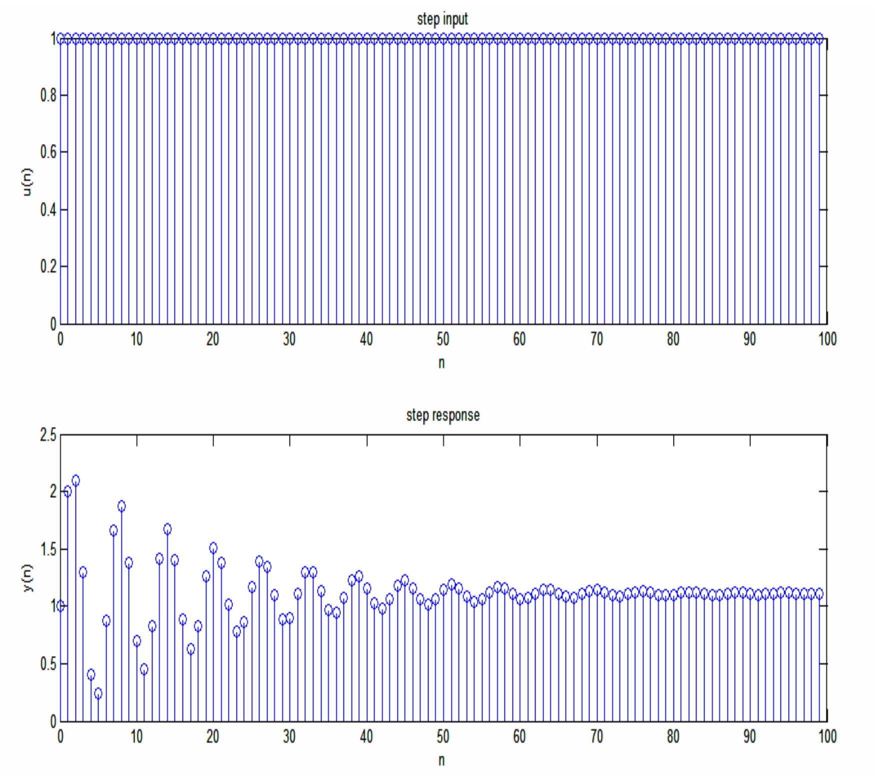


Figure 6.2: Step Response

EXPERIMENT NO 7A: Autocorrelation of a given sequence and verification of its properties.
Aim: To obtain autocorrelation of the given sequence and verify its properties. Algorithm:

1. **Input the sequence.**
2. **Use the “xcorr” function to get auto correlated output.**
3. **Plot the sequences.**

MATLAB Programs

```
clc
clear all;

xn = input('Enter the sequence x(n) =  ');
n = 0:length(xn)-1;
[rxx,l] = xcorr(xn);

disp('auto correlation of given sequence: ');
disp(rxx);

subplot(2,1,1);
stem(n,xn);
xlabel('----> n');
ylabel('----> x(n)');
title('Input sequence');

subplot(2,1,2);
stem(l,rxx);
xlabel('----> lag');
ylabel('----> rxx(l)');
title('Auto correlation');

%Verification of 1st Property
r1xx=fliplr(rxx);
if(r1xx==rxx)
disp('rxx is Symmetric : First property is verified');
else
disp('rxx is Not symmetric : First property is not verified');
end


%Verification of 2nd Property
m=max(rxx);
N = length(xn);
if(m==(rxx(N)))
disp('Maximum at origin 2nd property is verified');
else
disp('Maximum is not at origin 2nd  property is not verified');
end
```

```
%Verification of 3rd property
en=0.0;
for i=1:N
y=(xn(i)*xn(i));
en=en+y;
end
disp('energy of the sequence-');
disp(en);
disp(rxx(N));
if(en==int8(rxx(N)))
disp('Energy property is verified')
else
disp('energy property is not verified')
end
```

Result:

```
Enter the sequence x(n) =  [1 2 2 1]
Auto correlation of given sequence:
    1.0000    4.0000    8.0000   10.0000    8.0000    4.0000    1.0000

rxx is Symmetric : First property is verified
Maximum at origin 2nd property is verified
Energy of the sequence-
 10
 10.0000
Energy property is verified
```

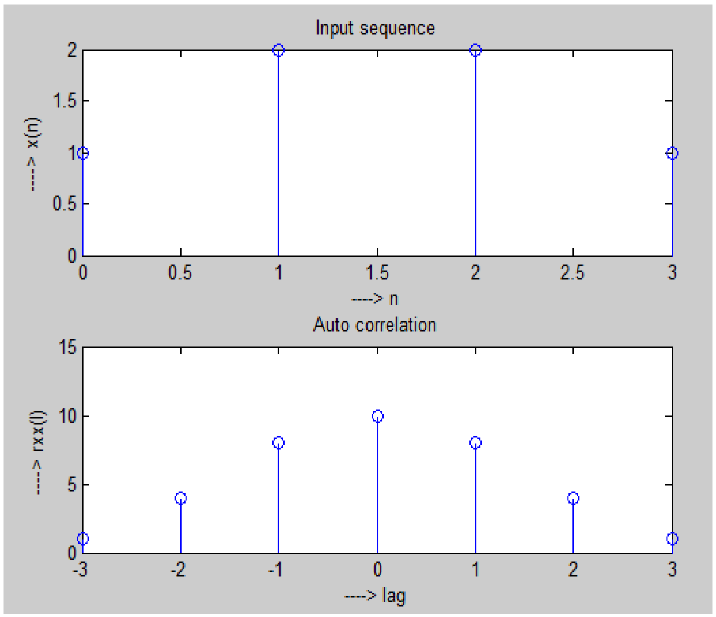


Figure 7.1: Autocorrelation of two signals

EXPERIMENT NO 7B: Cross correlation of a given sequence and verification of its properties.
Aim: To obtain cross correlation of the given sequence and verify its properties.
1.Rxy(1)=Ryx(-1)   2.Rxy(1)<=1/2[|Rxx(0)+Ryy(0)|]

MATLAB Programs

```
% Computation of Cross-correlation Sequence using folded sequence and convolution

x = input('Type in the reference sequence = ');
y = input('Type in the second sequence =     ');

% Compute the correlation sequence
[Rxy ,l] = xcorr(x,y);
disp('Cross correlation output is=');
 disp(Rxy);

stem(1,Rxy);
xlabel('---- > lag');
ylabel('----- > Rxy');
title('cross correlation');

%Verification of 1st property
Ryx=xcorr(y,x);
If(Rxy=fliplr(Ryx))
disp('1st Property is verified');
else
disp('1st property is not verified');
end

%Verification of 2nd property
Rxx=xcorr(x,x);
Ryy=xcorr(y,y);
a=0.5*(abs(max(Rxx)+max(Ryy)))
if(max(Rxy)<=a)
disp(' 2nd  Property is verified');
else
disp('2nd Property is  not verified');
end
```

Result:

```
Type in the reference sequence = [1 2 1 2]
Type in the second sequence =     [2 3 2 1]
Cross correlation output is=
    1.0000    4.0000    8.0000   12.0000   11.0000    8.0000    4.0000

1st Property is verified
2nd Property is verified
```

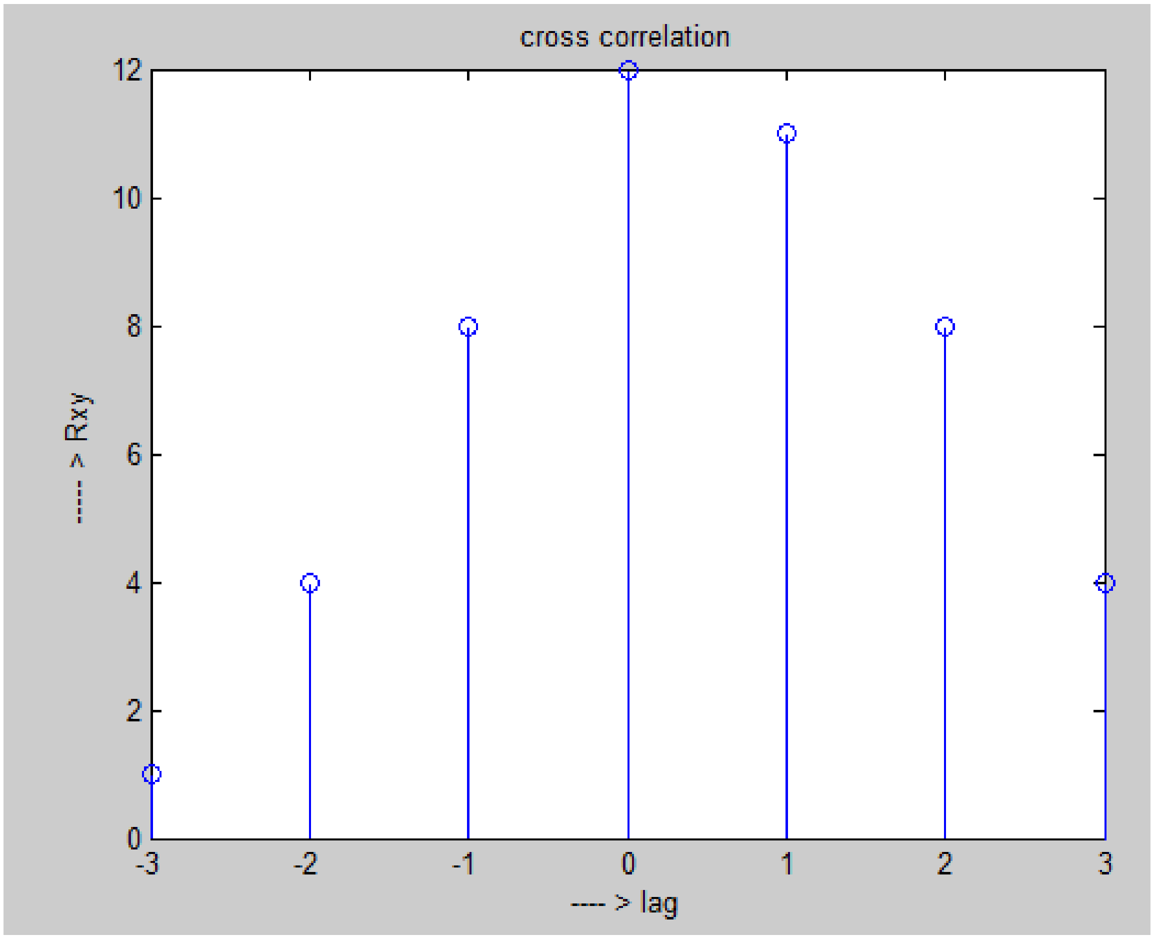


Figure 7.2: Cross Correlation of two Signals

EXPERIMENT NO 8: **Linear convolution of two given sequences**

Aim: To obtain convolution of two finite duration sequences. Algorithm:

1. Input the two sequences as x1,x2
2. Convolve both to get output y.
3. Plot the sequences.

MATLAB Program:

Level 1 Program (both sequences are right sided)

```
%main part of computation
x1=input ('Enter first sequence')
x2=input ('Enter second sequence')
y=conv(x1, x2);
disp('linear con of x1 & x2 is y=');
disp(y);
%graphical display part
subplot (2, 1, 1);
stem(y);
xlabel('time index n');
ylabel('amplitude ');
title('convolution output');
subplot(2,2,3);
stem(x1);
xlabel('time indexn');
ylabel('amplitude ');
title('plot of x1');
subplot(2,2,4)
stem(x2);
xlabel('time indexn');
ylabel('amplitude ');
title('plot ofx2');
```

Result

```
Enter first sequence [1 2 3 4]
x1 =1 2 3 4
Enter second sequence [ 2 3 2 4]
x2 = 2 3 2 4
linear con of x1 & x2  is
y={2, 7, 14, 25, 26, 20, 16}
```

Fig. 8.1 shows the plots

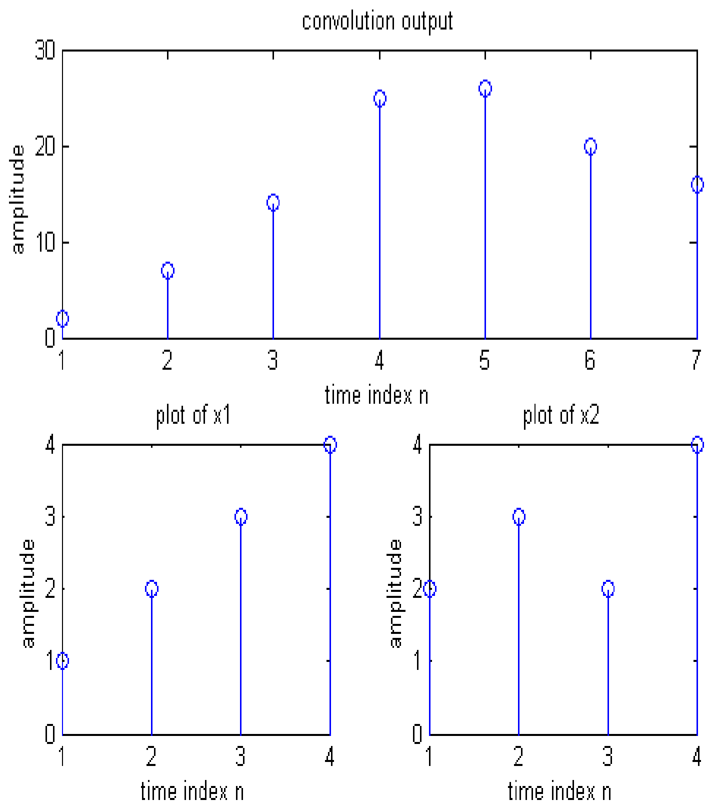
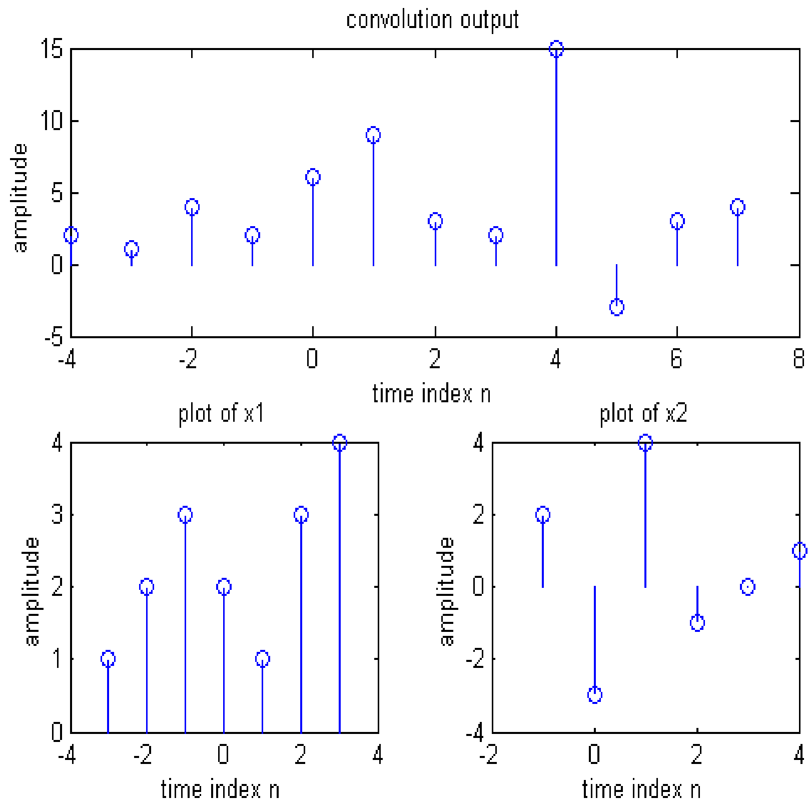


Figure 8.1: Convolution output for right sided sequences

Level 2 Program (Two sided sequences)

```
x1=[1 2 3 2 1 3 4]
 %first sequence
n1=-3:3 %
time vector of first seq
x2=[ 2 -3 4 -1 0 1]
 %second seq
n2=-1:4
%time vector of second seq
%add first elements of time vector
 ybegin=n1(1)+n2(1);
%add last elements of time vector
yend=n1(length(x1))+n2(length(x2));
 ny=[ybegin:yend];
y=conv(x1,x2) ;
disp('linear con of x1 & x2 is y=');
disp(y);
```

```
% graphical display
subplot (2,1,1);
stem(ny,y);
xlabel('time index n');
ylabel('amplitude ');
title('convolution output');
subplot(2,2,3);
stem(n1,x1);
xlabel('time index n');
ylabel('amplitude ');
 title('plot of x1');
subplot(2,2,4);
stem(n2,x2);
 xlabel('time index n');
ylabel('amplitude ');
title('plot of x2');
```



Result

```
x1 =[1, 2, 3, 2, 1, 3, 4]
n1 =-3 -2 -1 0 1 2 3
x2 =[2,-3, 4, -1, 0, 1]
n2 =-1 0 1 2 3 4
Linear con of x1 & x2 is y = [2, 1, 4, 2, 6, 9, 3, 2, 15, -3, 3, 4]
```

Experiment No. 9

a. To perform verification of commutative property of linear convolution.

b. To perform verification of distributive property of linear convolution.

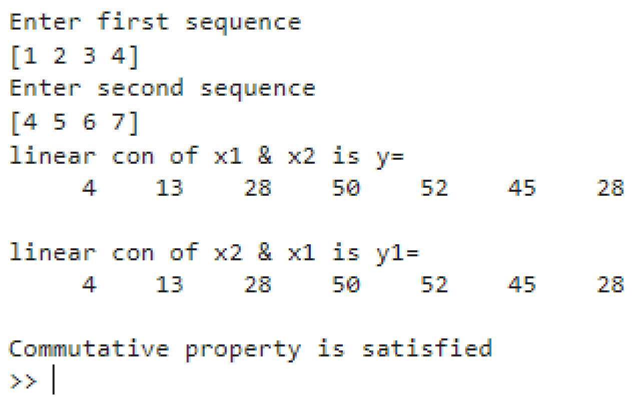
c. To perform verification of associative property of linear convolution.

MATLAB Program:

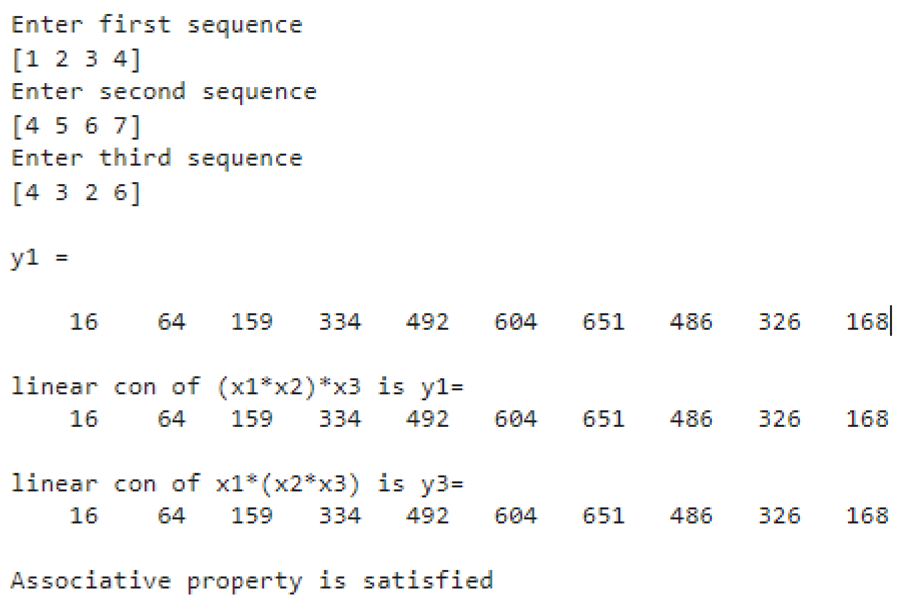
```
% Commutative Property
x1=input ('Enter first sequence');
x2=input ('Enter second sequence');
y=conv(x1, x2);
disp('linear con of x1 & x2 is y=');
disp(y);
y1=conv(x2, x1);
disp('linear con of x2 & x1 is y1=');
 disp(y1);
if(y==y1)
disp('Commutative property is satisfied');
else
disp('Commutative property not satisfied');
end

% Associative Property
x1=input ('Enter first sequence');
x2=input ('Enter second sequence');
x3=input('Enter third sequence');
y=conv(x1, x2);
y1=conv(y, x3)
disp('linear con of (x1*x2)*x3 is y1=');
disp(y1);
y2=conv(x2, x3);
y3=conv(x1, y2);
disp('linear con of x1*(x2*x3) is y3=');
 disp(y3);
if(y1==y3)
disp('Associative property is satisfied');
else
```

```
disp('Associative property not satisfied');
end

x1=input ('Enter first sequence');
x2=input ('Enter second sequence');
x3=input('Enter third sequence');
y=x1+ x2;
y1=conv(x3, y);
disp('linear con of x3*(x1+x2) is y1=');
disp(y1);
y2=conv(x3, x1);
y3=conv(x3, x2);
y4=y2+y3;
disp('linear con of x3*x1+ x3*x2 is y4=');
 disp(y4);
if(y1==y4)
disp('Distributive property is satisfied');
else
disp('Distributive property not satisfied');
end
```

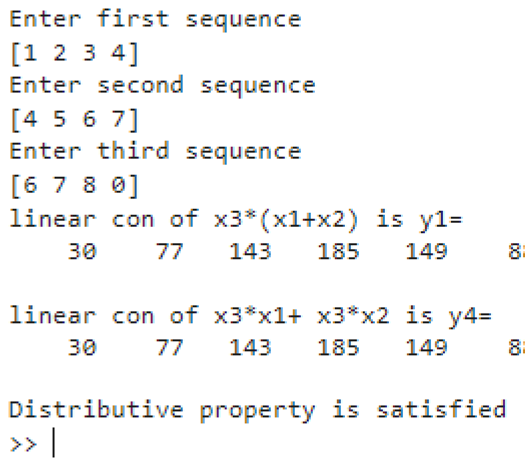
Results:



```
Enter first sequence
[1 2 3 4]
Enter second sequence
[4 5 6 7]
linear con of x1 & x2 is y=
     4    13    28    50    52    45    28

linear con of x2 & x1 is y1=
     4    13    28    50    52    45    28

Commutative property is satisfied
>> |
```



```
Enter first sequence
[1 2 3 4]
Enter second sequence
[4 5 6 7]
Enter third sequence
[4 3 2 6]

y1 =

    16    64   159   334   492   604   651   486   326   168

linear con of (x1*x2)*x3 is y1=
    16    64   159   334   492   604   651   486   326   168

linear con of x1*(x2*x3) is y3=
    16    64   159   334   492   604   651   486   326   168

Associative property is satisfied
```



```
Enter first sequence
[1 2 3 4]
Enter second sequence
[4 5 6 7]
Enter third sequence
[6 7 8 0]
linear con of x3*(x1+x2) is y1=
    30    77   143   185   149    88     0

linear con of x3*x1+ x3*x2 is y4=
    30    77   143   185   149    88     0

Distributive property is satisfied
>> 
```

Experiment No. 10: Impulse response of a given system

Aim: To find the impulse response h (n) of the given LTI system whose response y (n) to an input x (n) is given.

Algorithm for Problem1:

1. Input the two sequences as x andy.
2. Use deconv to get impulse responseh.
3. Plot thesequences.
4. Verify using conv(x,h) to get yback.

MATLAB Programs:

```
y= input (‘the output sequence y (n) of the system=');
x=input ('the input sequence of the system=');
h=deconv(y,x);
disp ('the impulse response of the system is=');
disp (h);

%graphical display part N=length (h);
n=0:1:N-1;
stem(n,h);
 xlabel('Time index n');
 ylabel('Amplitude');
title('impulse response of a system')

%Verification yv=conv(x,h);
disp('the verified output sequence is');
disp(yv)
```

Fig. 10.1 Impulse Response

Result:

```
The output sequence y(n) of the system=[1   2   1   1    0  -1]
the input sequence of the system= [1 1 1 1]
the impulse response of the system is=[1 1 -1]
the verified output sequence is      [1   2   1  1 0 -1]
```

The plots are shown in Fig.10.1

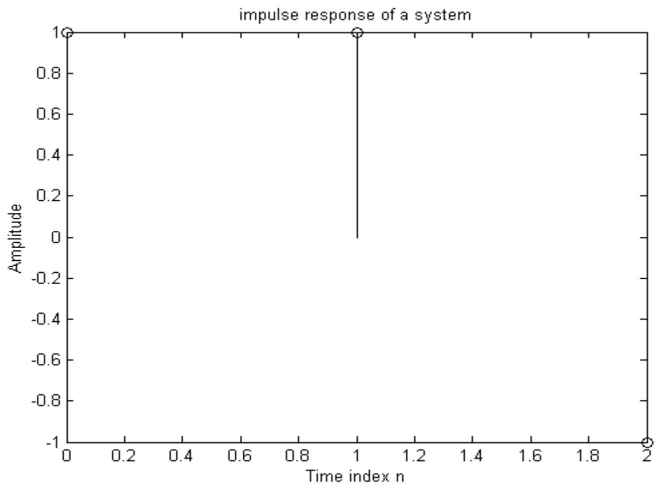


Figure 10.1: Impulse Response of a system

Experiment No. 11: To find Z-transform and inverse Z-transform of a sequence.

```
clc;
clear all;
 close all;
x=[1 2 3 4];
n=length(x);
b=0;
 y=sym('z');
 for i=1:n
b=b+x(i)*y^(i-1);
end
disp(b);

% To compute the Inverse Z transform
syms z;
F=(z^3+2*z^2+3*z+4)/(z^3);
f=iztrans(F);
disp('X(Z)');
disp(f);
```

Results;
The Z transform for x[n] = [1 2 3 4] is
4*z^3 + 3*z^2 + 2*z + 1

The Inverse Z-transform of F= (z^3+2*z^2+3*z+4)/ (z^3) is
X(Z)=2*kroneckerDelta(n - 1, 0) + 3*kroneckerDelta(n - 2, 0) + 4*kroneckerDelta(n - 3, 0) + kroneckerDelta(n, 0)

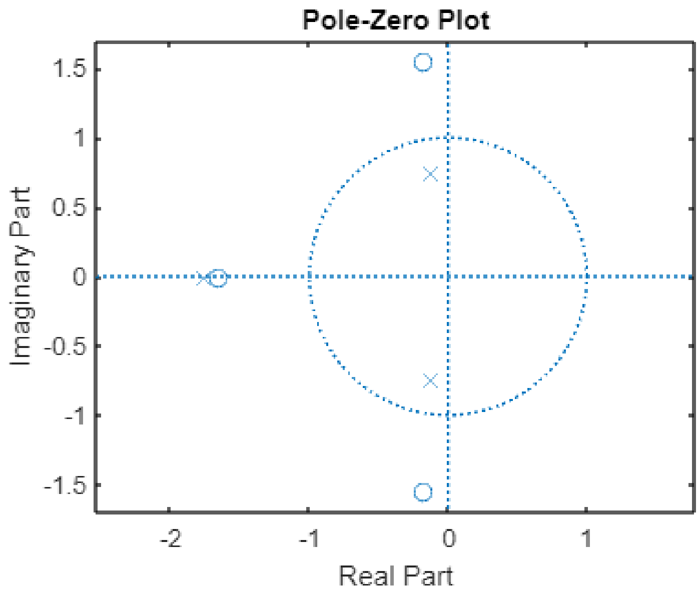
Note: kronecker delta is $\delta_{ij} = \begin{cases} 1, when i = j \\ 0, when i \neq j \end{cases}$

Experiment No. 12: To Plot Poles and Zeros on Z-transform domain

Z-transforms the Z-transform converts a discrete time-domain signal, which is a sequence of real or complex numbers, into a complex frequency-domain representation. The Z-transform, like many other integral transforms, can be defined as either a one-sided or two-sided transform.

MATLAB Program

```
clc;
close all;
clear all;
%b= input('enter the numerator coefficients')
%a= input('enter the denominator coefficients')
b=[1 2 3 4]
a=[1 2 1 1 ]
zplane(b,a);
```



Basic Signal Processing Lab VIVA Questions

1. What is MATLAB?
2. What are the applications of MATLAB?
3. State sampling theorem.
4. What is meant by Nyquist rate and Nyquist criteria?
5. Explain scaling and superposition properties of a system.
6. What is meant by linearity of a system and how it is related to scaling and superposition?
7. What is impulse function?
8. What is meant by impulse response?
9. What is energy signal? How to calculate energy of a signal?
10. What is power signal? How to calculate power of a signal?
11. Differentiate between even and odd signals.
12. Explain time invariance property of a system with an example.
13. What is memory less system?
14. When a system is said to have memory?
15. What is meant by causality?
16. Explain linear convolution and circular convolution.
17. What is the length of linear and circular convolutions if the two sequences are having the length n1 andn2?
18. What are Fourier series and Fourier transform?
19. What are the advantages and special applications of Fourier transform, Fourier series, Z transform and Laplace transform?
20. Differentiate between DTFT and DFT. Why it is advantageous to use DFT in computers rather than DTFT?
21. How to perform linear convolution using circular convolution? What is meant by correlation?
22. What is auto-correlation?
23. What is cross-correlation?
24. What are the advantages of using autocorrelation and cross correlation properties in signal processing fields?
25. How auto-correlation can be used to detect the presence of noise?
26. What are difference equations and differential equations?
27. What is non real time processing?
28. What is meant by real time processing?