DYANANDASAGAR COLLEGE OF ENGINEERING
DEPARTMENT
OF
ELECTRONICS & COMMUNICATION ENGINEERING

INTEGRATED SIGNALS AND SYSTEMS LABORATORY MANUAL

IV Semester (21EC42)

Autonomous Course

Lab Manual Prepared by

Prof. Kavita Guddad, Prof. Shahla Sohail,
Prof. Kumar P, Prof. Srividya L,
Dr. T. C. Manjunath

HOD: Dr. T.C. Manjunath

Name of the Student	:	
Semester /Section	:	
USN	:	
Batch	:	

DYANANDASAGAR COLLEGE OF ENGINEERING
DEPARTMENT
OF
ELECTRONICS & COMMUNICATION ENGINEERING

Name of the Laboratory	**:** Integrated Signals and Systems Lab
Semester/Academic Year	**:** IV / 2022-23
No. of Students/Batch	**:** 20-23
No. of Systems	**:** 23
Area in square meters	**:** 68 Sq Mts
Lab In charge/s	**:** Prof. Kavita Guddad, Prof. Shahla Sohail, Prof. Kumar P, Prof. Srividya L, Dr. T. C. Manjunath
Instructors	**:** Mrs. Veena H. S./Mrs. Gayathri H.
HOD	**:** Dr. T.C. Manjunath, Ph.D. (IIT Bombay)

About the College & the Department:

The Dayananda Sagar College of Engineering was established in 1979, was founded by Sri R. Dayananda Sagar and is run by the Mahatma Gandhi Vidya Peetha Trust (MGVP). The college offers undergraduate, post-graduates and doctoral programmes under Visvesvaraya Technological University & is currently autonomous institution. MGVP Trust is an educational trust and was promoted by Late. Shri. R. Dayananda Sagar in 1960. The Trust manages 28 educational institutions in the name of “Dayananda Sagar Institutions” (DSI) and multi – Specialty hospitals in the name of Sagar Hospitals - Bangalore, India. Dayananda Sagar College of Engineering is approved by All India Council for Technical Education (AICTE), Govt. of India and affiliated to Visvesvaraya Technological University. It has widest choice of engineering branches having 16 Under Graduate courses & 17 Post Graduate courses. In addition, it has 21 Research Centres in different branches of Engineering catering to research scholars for obtaining Ph.D under VTU. Various courses are accredited by NBA & the college has a NAAC with ISO certification. One of the vibrant & oldest dept is the ECE dept. & is the biggest in the DSI group with 70 staffs & 1200+ students with 10 Ph.D.’s & 30^{+} staffs pursuing their research in various universities. At present, the department runs a UG course (BE) with an intake of 240 & 2 PG courses (M.Tech.), viz., VLSI Design Embedded Systems & Digital Electronics & Communications with an intake of 18 students each. The department has got an excellent infrastructure of 10 sophisticated labs & dozen class room, R & D centre, etc…

Vision of the College:

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 College:

- ❖ 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.

VISION OF THE DEPATTMENT

To achieve continuous improvement in quality technical education for global competence with focus on industry, societal needs, research and professional success.

MISSION OF THE DEPARTMENT

- ❖ Offering quality education in Electronics and Communication Engineering with effective teaching learning process in multidisciplinary environment.
- ❖ Training the students to take-up projects in emerging technologies and work with team spirit.
- ❖ To imbibe professional ethics, development of skills and research culture for better placement opportunities.

PROGRAMME EDUCATIONAL OBJECTIVES [PEOS]

The Graduate students must be able to:

- ❖ **PEO 1 :** Successful in industry, academia, or entrepreneurship as a result of a strong teaching learning process, with keen interest in pursuing higher studies in various domains.
- ❖ **PEO 2 :** Capable of leading technological and managerial projects for serving industry and society with knowledge of Electronics and Communication Engineering.
- ❖ **PEO 3 :** Competent professional capable of adapting to changing technological scenarios and Societal needs, with expertise in relevant domains.

PROGRAMME SPECIFIC OUTCOMES [PSOS]

PSO-1: Design, develop and integrate electronic circuits and systems using current practices and Standards.

PSO-2: Apply knowledge of hardware and software in designing Embedded and Communication Systems.

Evaluation Process of the integrated course

<table>
<tr><th colspan="6">Assessment Evaluation pattern for Integrated Professional Core Courses</th></tr>
<tr><td colspan="6">CIE for the theory component of Integrated Professional Core Courses (IPCC)
(Bloom’s Taxonomy Levels: Remembering, Understanding, Applying, Analyzing, Evaluating and Creating)</td></tr>
<tr><td rowspan="5">Each Test will be conducted for50 Marks adding up to 150 Marks. Final test marks will be reduced to 30 Marks.</td><td rowspan="2">IAT</td><td colspan="4">Marks</td></tr>
<tr><td>Max. Marks</td><td>Reduced to 30 Marks</td><td>Average</td><td>IAT Final Marks</td></tr>
<tr><td>IAT-I</td><td>50</td><td>30(A)</td><td rowspan="3">(A+B+C)/3
=30 (D)</td><td rowspan="3">Total out of 30 marks</td></tr>
<tr><td>IAT-II</td><td>50</td><td>30(B)</td></tr>
<tr><td>IAT-III</td><td>50</td><td>30(C)</td></tr>
<tr><td rowspan="2">QUIZ
(One Quiz to be evaluated for 30 marks)</td><td colspan="3">Evaluated for 30 Marks</td><td colspan="2">Reduced to 10 Marks</td></tr>
<tr><td colspan="3">30</td><td colspan="2">10 (E)</td></tr>
<tr><td rowspan="2">Alternate Assessment Tool (AAT)</td><td colspan="5">Reflection Note on Guest Lecture/ Reflection note on Industrial Visit/ E-course certification/Building models/Group discussion/Seminar/Paper Presentation/Open Book Assignment</td></tr>
<tr><td colspan="5">10 Marks (F)</td></tr>
<tr><td>Total CIE Marks</td><td colspan="3">CIE (D) + QUIZ (E) + AAT(F)</td><td colspan="2">50 (G) Marks</td></tr>
</table>

<table>
<tr><td colspan="5">CIE for the practical component of Integrated Professional Core Courses (IPCC) (50 Marks)</td></tr>
<tr><td colspan="2">Conduction of Experiments</td><td rowspan="4">30 (H)</td><td rowspan="5">Total= H+I=50 (J)</td><td rowspan="5">Total out of 50 Marks</td></tr>
<tr><td>Performance of the Experiment (On completion of every experiment/program in the laboratory, the students shall be evaluated and marks shall be awarded on the same day. 20 marks are for conducting the experiment and calculations/observations/output)</td><td>20</td></tr>
<tr><td>Record</td><td>05</td></tr>
<tr><td>Evaluation of outcome/Viva</td><td>05</td></tr>
<tr><td>Final test/Case Study/Open Ended Experiment(if it is not test then a five page report stapled hasto be submitted)</td><td>50</td><td>Reduced to 20 (I)</td></tr>
</table>

Continuous Evaluation

Experiment. No.	Date	calculations/ observations/ output	Alternative program/ Assignment	Record	Evaluation of outcome/ Viva	Total (H)	Sign. Of Teacher with Date
		10	10	5	5	30	
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							
11							
12							
Final TestT (I) 50 Reduced to20							
Final Marks (H+I)							

LIST OF EXPERIMENTS

Experiment. No.	Contents of the Experiment	Hours
1	Generation of discrete and continuous time signals such as unit impulse, unit step, unit ramp, Sinusoidal, real exponential, random and complex exponential signals and plotting them in different manner (Using plot, stem and subplot).	02
2	Performing signal operations: folding, Shifting, Amplitude and time scaling, addition, multiplication of continuous and discrete time signal	02
3	a. Computation of energy and power of a signal. b. Checking for symmetry of a signal. c. Finding even and odd parts of a signal.	02
4	a. Convolution using time domain approach and verification using built in command b. Verification of properties of convolution.	02
5	To perform correlation operation between two signals and verify properties.	02
6	To verify different properties of a given system as linear or non-linear, causal or non-causal, stable or unstable etc.	02
7	Solving difference equation with initial conditions for given input.	02
8	Finding transfer function, frequency response, impulse response and system response of a system defined by difference equation. Also plotting poles and zeros. (Need to comment on system property)	02
9	Finding Fourier series of a signal and verification of properties	02
10	Finding Fourier transform of a signal and verification of properties	02
11	Finding DFT of a signal/combination of signals and plotting the spectra.	02
12	Perform sampling of a CT signal and analyze in both time and frequency domain.	02

DO's

- All the students should come to LAB on time with proper dress code and identity card.
- Keep your belongings in the book rack of laboratory.
- Students have to enter their name, USN, time-in/out and signature in the log register maintained in the laboratory.
- All the students should submit their records before the commencement of Laboratory experiments.
- Students should come to the lab well prepared for the experiments which are to be performed in that particular session.
- Students are asked to do the experiments on their own and should not waste their precious time by talking, roaming and sitting idle in the labs.
- Observation book and record book should be complete in all respects and it should be corrected by the staff member.
- Before leaving the laboratory students should arrange their chairs and leave in orderly manner after completion of their scheduled time.
- Prior permission to be taken, if for some reasons, they cannot attend lab.
- Immediately report any sparks/ accidents/ injuries/ any other untoward incident to the faculty /instructor.
- 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.
- Don't switch on power supply without prior permission from the concerned staff.
- Never leave the experiments while in progress.
- Do not leave the Laboratory without the signature of the concerned staff in observation book.

Experiment 1
Generation of basic signals

Aim: To generate and plot discrete time (DT) and continuous time (CT) and discrete time signals.

Objective:

To write a program in MATLAB to simulate different signals such as unit impulse, unit step, unit ramp, sinusoidal, real exponential, random and complex exponential signals and plot them

Theory:

If the amplitude of the signal is defined at every instant of time then it is called continuous time signal. If the amplitude of the signal is defined at only at some instants of time then it is called discrete time signal. If the signal repeats itself at regular intervals then it is called periodic signal. Otherwise they are called aperiodic signals.

ex: ramp, Impulse, unit step, sinc- are few examples of Aperiodic signals

square, sawtooth, triangular sinusoidal – are few examples of periodic signals.

Ramp signal: The ramp function is a unitary real function, easily computable as the mean of the independent variable and its absolute value. This function is applied in engineering. The name ramp function is derived from the appearance of its graph.

$$\mathbf{r\,(t)} = \begin{cases} \mathbf{t \ \ when\ t \geq 0} \\ \mathbf{0 \ \ else} \end{cases}$$

Unit impulse signal: One of the more useful functions in the study of linear systems is the "unit impulse function". An ideal impulse function is a function that is zero everywhere but at the origin, where it is infinitely high. However, the area of the impulse is finite

$$\begin{aligned} x(t) &= 1 \qquad \text{when } t=0 \\ &= 0 \qquad \text{else} \end{aligned}$$

Unit step signal: The unit step function and the impulse function are considered to be fundamental functions in engineering, and it is strongly recommended that the reader becomes very familiar with both of these functions.

$$x(t) = 1 \text{ for all } t > 0$$

$$0 \text{ for } \quad t < 0$$

Sinc signal: There is a particular form that appears so frequently in communications engineering, that we give it its own name. This function is called the "Sinc function".

The Sinc function is defined in the following manner:

$$sinc(x) = \frac{\sin \pi x}{\pi x} \quad \textbf{if } \ x \neq 0 \ \textbf{and} \ \ \textbf{sinc(0) =1}$$

The value of sinc(x) is defined as 1 at x = 0, since

$$\lim_{x \to 0} sinc(x) = 1$$

Steps:

1. Open MATLAB
2. Open new M-file and type the program
3. Save in current directory and Run the program
4. For the output see command window/ Figure window

Reference MATLAB Code:

```
%discrete unit impulse sequence generation
clc; close all;
n=-3:4;
x=[n==0];
subplot(4,4,1);stem(n,x);
xlabel('Time (sec)');
ylabel('Amplitude');
title('discrete unit impulse');
%continuous unit impulse signal generation
t=-3:.25:4; x=[t==0];
subplot(4,4,2);plot(t,x);
xlabel('Time (sec)');
ylabel('Amplitude');
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 time Periodic Square Wave');

% continuous sawtooth 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;
```

<table>
<tr>
<td>
% 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;

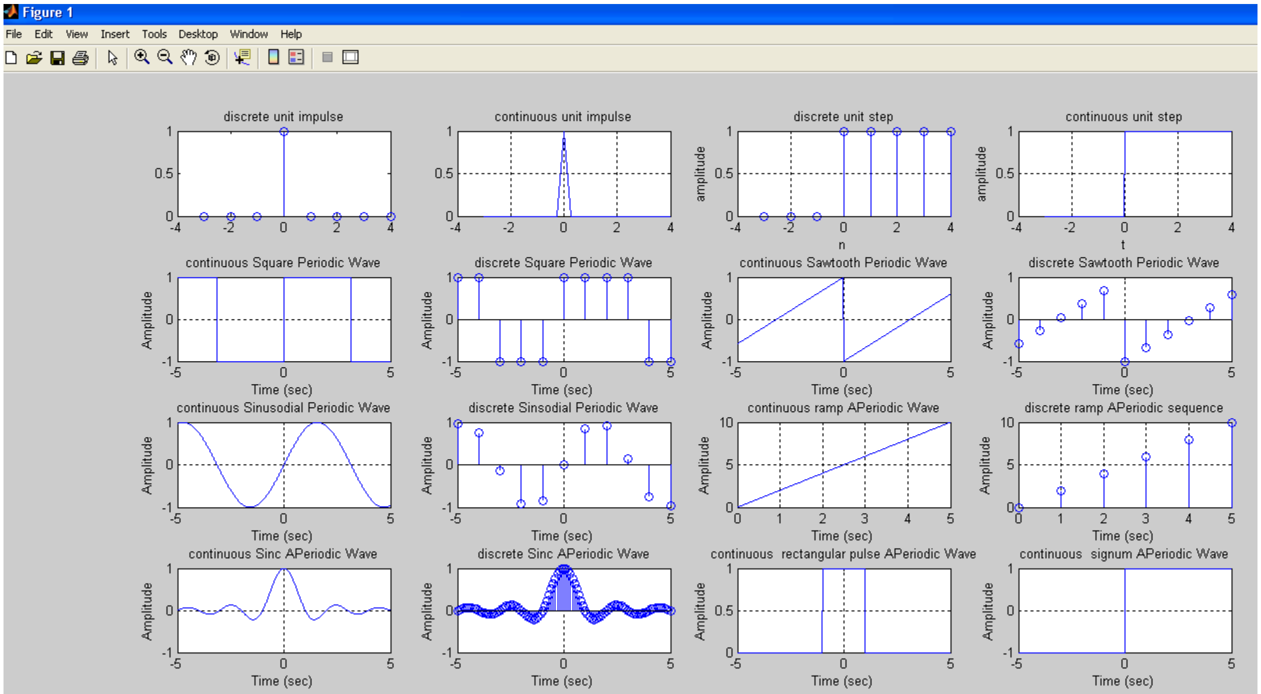
% discrete ramp sequence generator
n = 0:5;
x=2*n;
subplot(4,4,12);stem(n,x);
xlabel('Time (sec)');
ylabel('Amplitude');
title('discrete ramp Aperiodic sequence');
grid;
</td>
<td>
% continuous sinc signal generator
t = -5:.01:5;
x = sinc(t);
subplot(4,4,13);plot(t,x);
xlabel('Time (sec)');ylabel('Amplitude');
title('Continuous Aperiodic Sinc 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;
</td>
</tr>
</table>

Expected Output:



Assignment:

1. Generate a CT and DT exponential signal and a random signal and plot them in separate figure windows.
2. Generate a signal x which contains at least one cycle of sinusoidal signals of frequencies 50Hz and 100Hz.

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 2
Basic Operations on Signals

Aim: To study various basic operations that are performed on signals using MATLAB.

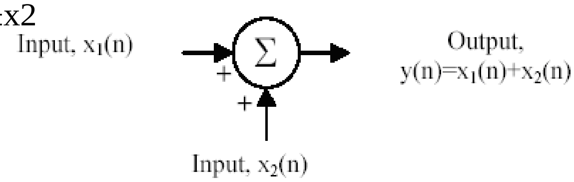
Objective:

To write a program in MATLAB to realize all basic operations performed on dependent variable and independent variable of a signal and verify.

Theory:

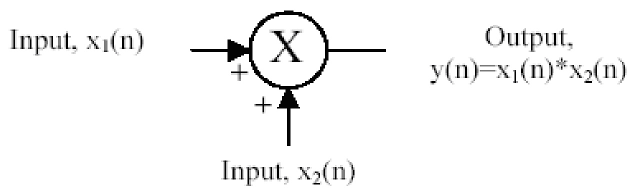
Operations on dependent variable

Signal Addition/Subtraction: Any two signals x1 and x2 can be added/subtracted to form a third signal, y=x1±x2



Signal Multiplication:

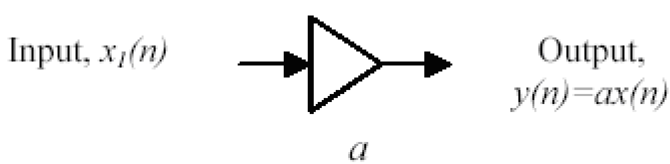
Multiplication of two signals x1 and x2 can be obtained by multiplying their values at every instants of time. y = x1 x2



Operations on independent variable

Time reversal/Folding: Time reversal of a signal x(n) can be obtained by folding the signal about =0. y(n)=y(-n)

Signal Amplification/Scaling: y(n)=ax(n) if a < 1 attenuation,
a >1 amplification



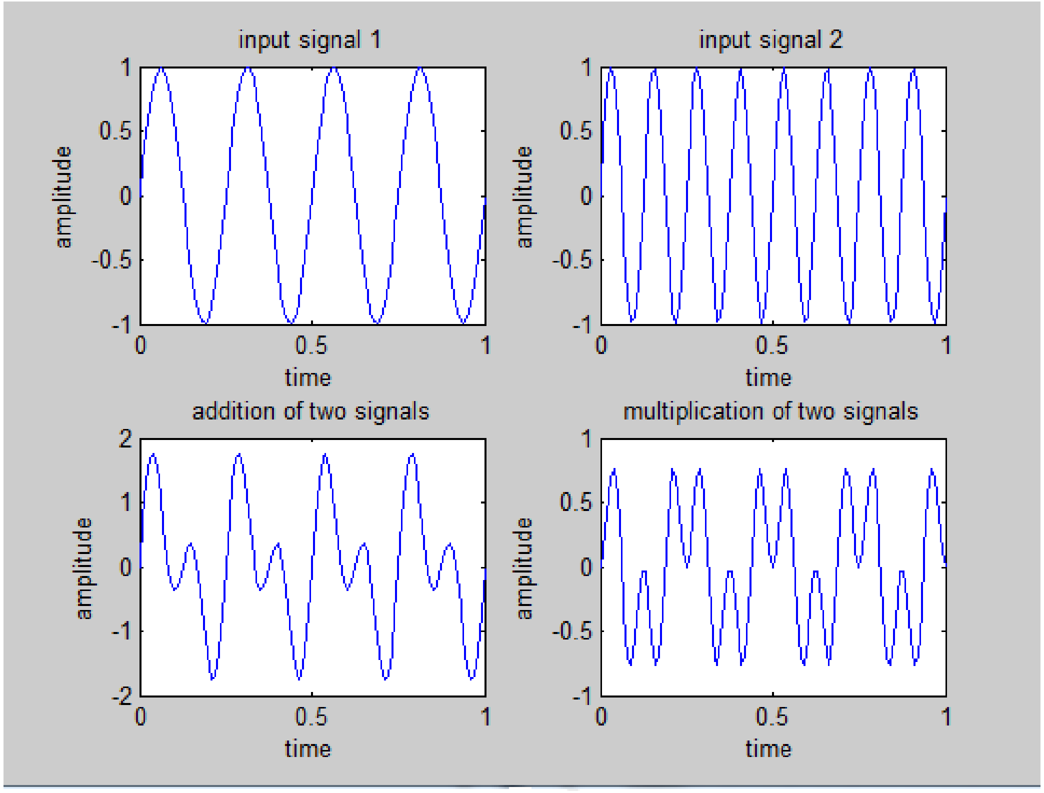
Time shifting: The time shifting of x(n) obtained by delay or advance the signal in time by using y(n)=x(n-k)

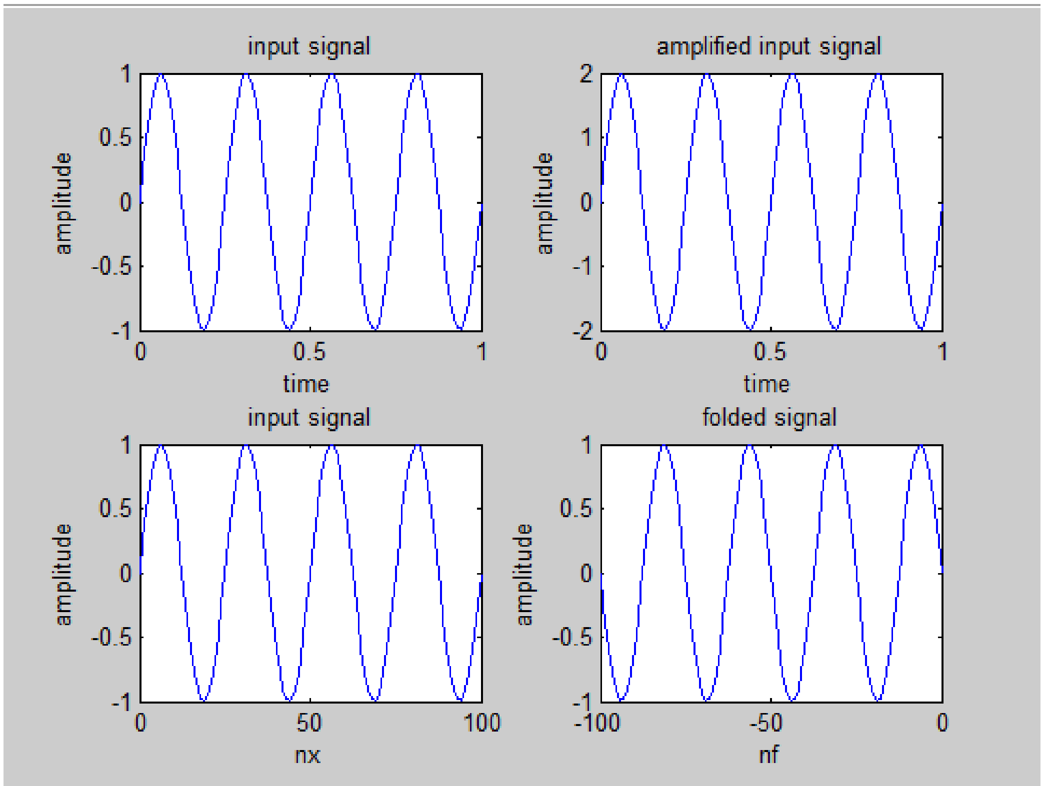
If k is a positive number, y(n) shifted to the right i.e. the shifting delays the signal
If k is a negative number, y(n) gets shifted left i.e. Signal Shifting advances the signal

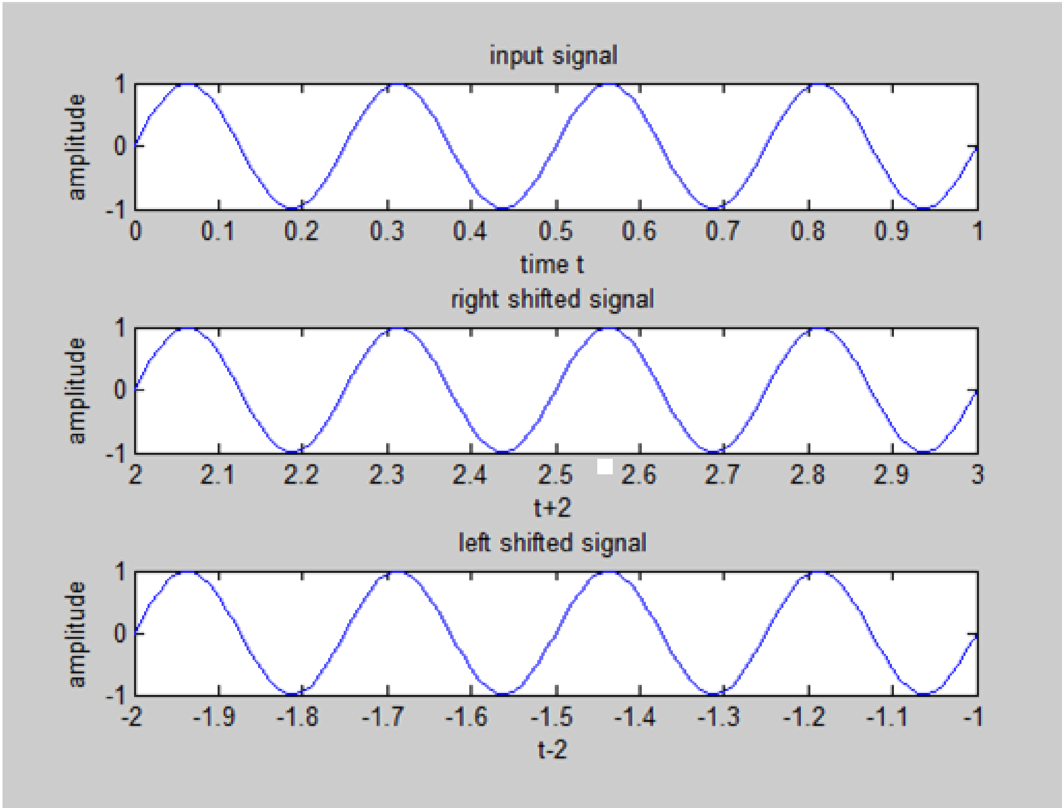
Reference MATLAB Code:

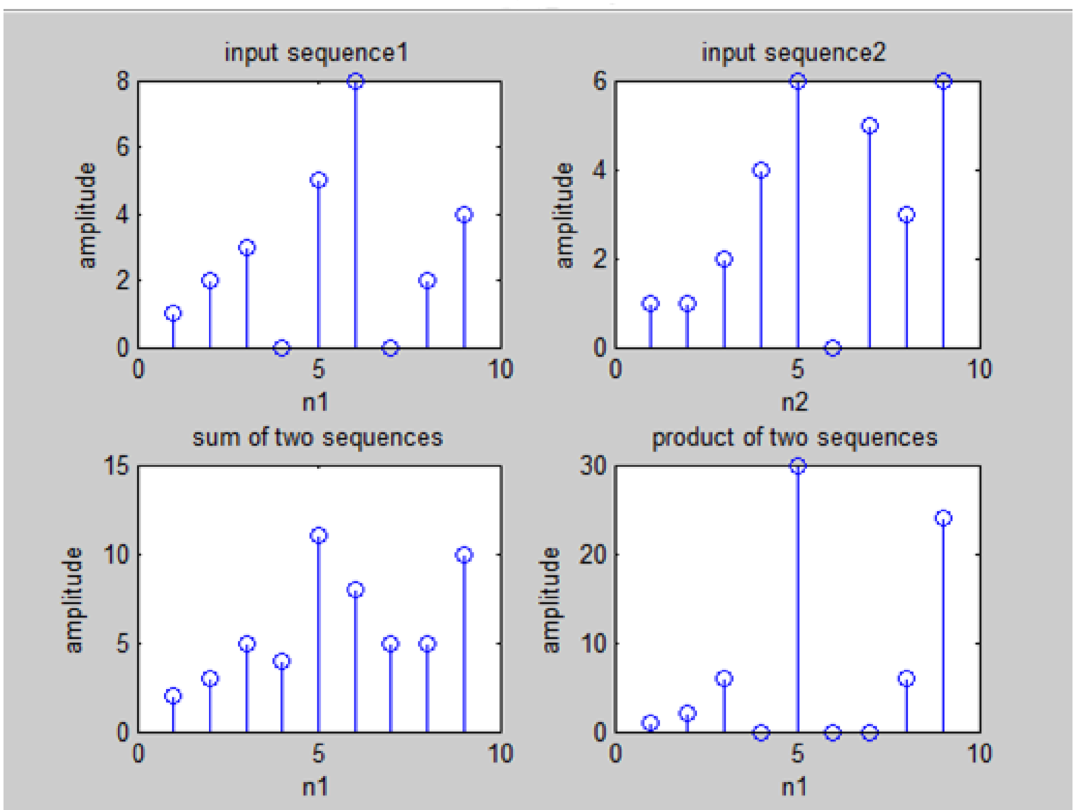
```
clc; clear all; close all;
% generating two input signals
t=0:.01:1;
x1=sin(2*pi*4*t);
x2=sin(2*pi*8*t);
subplot(2,2,1); plot(t,x1);
xlabel('time');
ylabel('amplitude');
title('input signal 1');
subplot(2,2,2); plot(t,x2);
xlabel('time');
ylabel('amplitude');
title('input signal 2');
% addition of signals
y1=x1+x2;
subplot(2,2,3); plot(t,y1);
xlabel('time');
ylabel('amplitude');
title('addition of two signals');
% multiplication of signals
y2=x1.*x2;
subplot(2,2,4);plot(t,y2);
xlabel('time');
ylabel('amplitude');
title('multiplication of two signals');
% scaling of a signal1
A=2;
y3=A*x1; figure;
subplot(2,2,1); plot(t,x1);
xlabel('time'); ylabel('amplitude'); title('input signal')
subplot(2,2,2);plot(t,y3);
xlabel('time'); ylabel('amplitude');
title('amplified input signal');
% folding of a signal1
h=length(x1);
nx=0:h-1;
subplot(2,2,3);plot(nx,x1);
xlabel('nx'); ylabel('amplitude'); title('input signal')
y4=fliplr(x1);
nf=-fliplr(nx);
subplot(2,2,4); plot(nf,y4);
xlabel('nf'); ylabel('amplitude'); title('folded signal');
%shifting of a signal 1
figure; subplot(3,1,1); plot(t,x1);
xlabel('time t'); ylabel('amplitude'); title('input signal');
 subplot(3,1,2); plot(t+2,x1);
xlabel('t+2'); ylabel('amplitude'); title('right shifted signal');
subplot(3,1,3);plot(t-2,x1);
xlabel('t-2'); ylabel('amplitude'); title('left shifted signal');
%operations on sequences
n1=1:1:9;
s1=[1 2 3 0 5 8 0 2 4];
figure; subplot(2,2,1); stem(n1,s1);
xlabel('n1'); ylabel('amplitude'); title('input sequence1');
s2=[1 1 2 4 6 0 5 3 6];
subplot(2,2,2);stem(n1,s2);
xlabel('n2'); ylabel('amplitude'); title('input sequence2');
% addition of sequences
s3=s1+s2;
subplot(2,2,3);stem(n1,s3);
xlabel('n1'); ylabel('amplitude'); title('sum of two sequences');
% multiplication of sequences
s4=s1.*s2;
subplot(2,2,4);stem(n1,s4);
xlabel('n1'); ylabel('amplitude');
title('product of two sequences');
```

Expected Output:






input signal
amplitude
time t
right shifted signal
t+2
left shifted signal
t-2


input sequence1
input sequence2
sum of two sequences
product of two sequences
amplitude
n1
n2

Manual Calculations/Observation:

Assignment:

a. Write MATLAB program for time scaling of a signal.
b. Perform signal operations with respect to dependent variable on x1(n) = a ramp signal and x2(n)= an exponential signal (May be done using MATLAB/Simulink)

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 3
Energy, Power and Symmetry of a signal

Aim: a. To compute energy and power of a signal.
b. To check for symmetry of a signal.
c. To find even and odd parts (components) of a signal.

Objectives:

a. To write a MATLAB program to find the energy and power of a given signal and verify.
b. To write a MATLAB program to check for symmetry of a signal and verify.
c. To write a MATLAB program to find odd and even parts of a signal and verify.

Theory:

a. Energy and Power of a signal:

The total Energy and average power of a given DT signal x(n) is given by E and P

E = $$\lim_{N\to\infty}\sum_{n=-N}^{N}|x[n]|^2 = \sum_{n=-\infty}^{\infty}|x[n]|^2$$

P = $$\lim_{N\to\infty}\frac{1}{2N+1}\sum_{n=-N}^{N}|x[n]|^2$$

Reference MATLAB code

```
%Only non-periodic and finite duration signals have finite energy
z1=input('enter the input sequence');
e1=sum(abs(z1).^2);
disp('energy of given sequence is');e1
% program for energy of a signal finite duration cosine signal
t=0:pi:2*pi;
z2=cos(2*pi*50*t).^2;
e2=sum(abs(z2).^2);
disp('energy of given signal is');e2
% program to find average power of a periodic signal of fundamental period 50 samples
N=50; n=0:N-1;
x=sin(2*pi*n/N);
stem(x);
P=sum(abs(x.^1))/N;
disp('power of given signal is');P
```

Expected Output :

```
enter the input sequence[1 3 2 4 1]
energy of given sequence is
 e1 =
    31
energy of given signal is
e2 =
   1.6775
power of given signal is
P =
   0.6358
```

b. Symmetry of a signal:

One of characteristics of signal is symmetry that may be useful for signal analysis. Even signals are symmetric around vertical axis, and Odd signals are symmetric about origin.

Even Signal: A signal is referred to as an even if it is identical to its time-reversed counterpart i.e. x(-n) = x(n) for all n

Odd Signal: A signal is odd if x(-n) = x(n) for all n

An odd signal must be 0 at t=0, in other words, odd signal passes the origin.

Using the definition of even and odd signal, any signal may be decomposed into a sum ofits even part, $x_e(t)$, and its odd part, $x_o(t)$, as follows

Reference MATLAB code	**Expected Output :**

clc;close all; clear all;
x=input('Enter the signal x ');
y=fliplr(x);
if(x==y)
disp('Signal is even symmetric');
elseif(x==-1*y)&&(x(length(x)/2)==0)
disp('Signal is odd symmetric');
else
disp('Signal is not symmetric');
end
	
	Enter the signal x [1 2 3 0 -3 -2 -1]
	Signal is odd symmetric
	
	Enter the signal x [1 2 3 2 1]
	Signal is even symmetric
	
	Enter the signal x [1 2 3 2 -3 -2 -1]
	Signal is not symmetric

c. Even and odd part of a signal:

Any signal x(t) can be expressed as sum of even and odd components of it

i.e .$\mathbf{x(t)=x_e(t)+x_o(t)}$

It may be shown that the even component $x_e(t)$ and the odd component $x_o(t)$can be expressed in terms of x(t) as

$$X_e(t) = \frac{x(t) + x(-t)}{2}$$

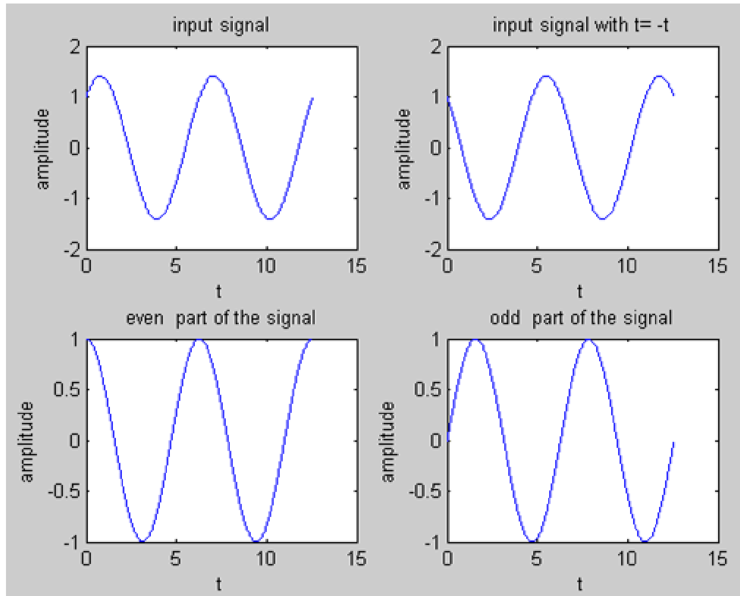
$$X_e(t) = \frac{x(t) + x(-t)}{2}$$

Reference MATLAB Code:

Program to find even and odd parts of a signal.

```
clc; close all; clear all;
%Even and odd parts of a signal
t=0:.001:4*pi;
x=sin(t)+cos(t);    %
x(t)=sint(t)+cos(t)
subplot(2,2,1);plot(t,x)
xlabel('t');ylabel('amplitude')
title('input signal');
y=sin(-t)+cos(-t); % y(t)=x(-t)
subplot(2,2,2);plot(t,y)
xlabel('t');
ylabel('amplitude');title('input
signal with t= -t');
even=(x+y)/2;
subplot(2,2,3); plot(t,even)
xlabel('t'); ylabel('amplitude');
title('even part of the signal') ;
odd=(x-y)/2;
subplot(2,2,4); plot(t,odd)
xlabel('t');  ylabel('amplitude');
title('odd part of the signal');
```

Expected Output:



Manual Calculations/Observation:

Assignment:

1. Find energy and power of another signal (To be specified by teacher)
2. Find even and odd components of another signal (To be specified by teacher)
3. In both of above cases verify whether original signal can be obtained by adding its even and odd components?

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 4
Study of Linear Convolution and its properties

Aim: To compute and simulate the Convolution of two discrete sequences and verify the output obtained using MATLAB.

Objectives:

a. To obtain the output of convolution operation on discrete signals and verify the same using MATLAB built-in function.
b. To verify the commutative, distributive and Associative properties of linear convolution.

Theory:

The output y[n] of a LTI (linear time invariant) system can be obtained by convolving the input x[n] with the system's impulse response h[n]. The linear convolution sum of two DT signals x(n) and h(n) is given by $y[n] = x[n] * h[n] = \sum_{k=-\infty}^{+\infty} x[k]h[n-k] = \sum_{k=-\infty}^{+\infty} x[n-k]h[k]$.

To compute this equation, the linear convolution involves the following operations:

- Folding-fold h[k] to get h[-k] (for y[0])
- Multiplication – $y_k[n] = x[k] \times h[n-k]$ (both sequences are multiplied sample by sample)
- Addition- Sum all the samples of the sequence $y_k[n]$ to obtain y[n]
- Shifting – the sequence h[-k] to get h[n-k] for the next n.

Reference MATLAB Code:

```
clc ;clear all ;close all;
x=input('Enter x: ') ; h=input('Enter h: ') ;
m=length(x); n=length(h);
X=[x,zeros(1,n)];
H=[h,zeros(1,m)];
for i=1:n+m-1
Y(i)=0;
for j=1:m
if(i-j+1>0)
Y(i)=Y(i)+X(j)*H(i-j+1);
end
end
end
Disp('Convolution output is y(n)= '); Disp('Y')
stem(Y); ylabel('Y[n]'); xlabel('----->n');
title('Convolution of Two Signals without using built in command');
```

Verification Using In-built Command:

>> y=conv(x,h);stem(y); ylabel('Y[n]'); xlabel('----->n'); title('Convolution of Two Signals without conv function');

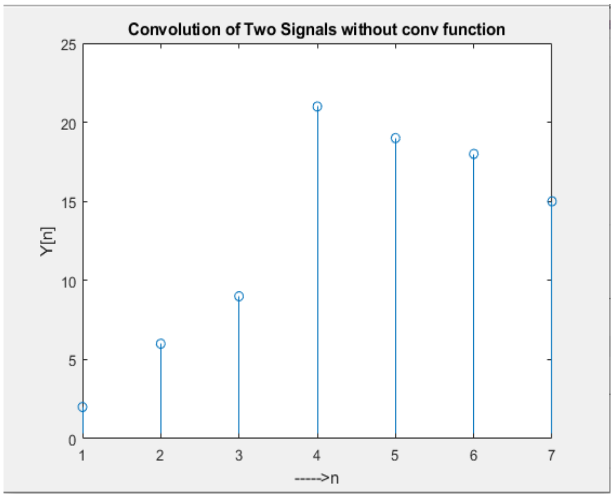
Expected Output:

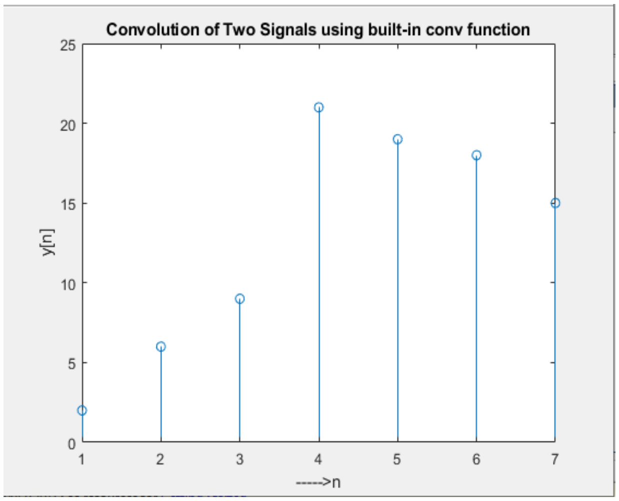
Enter x: [1 2 1 5]

Enter h: [2 2 3 3]

Convolution output is y(n)=

2 6 9 21 19 18 15





Manual Calculations/Observation:

Observation:

Verification of properties of convolution

1) Commutative property: x conv h= h conv x (a system property which is independent of which sequence is called the input or the impulse response)
2) Distributive property: x conv (h1 + h2) = x conv h1 + x conv h2 (result comes from parallelly connected systems)
3) Associative property: (x conv h1) conv h2 = x conv (h1 conv h2) (result comes from cascade Connection of systems)

Reference MATLAB Code:

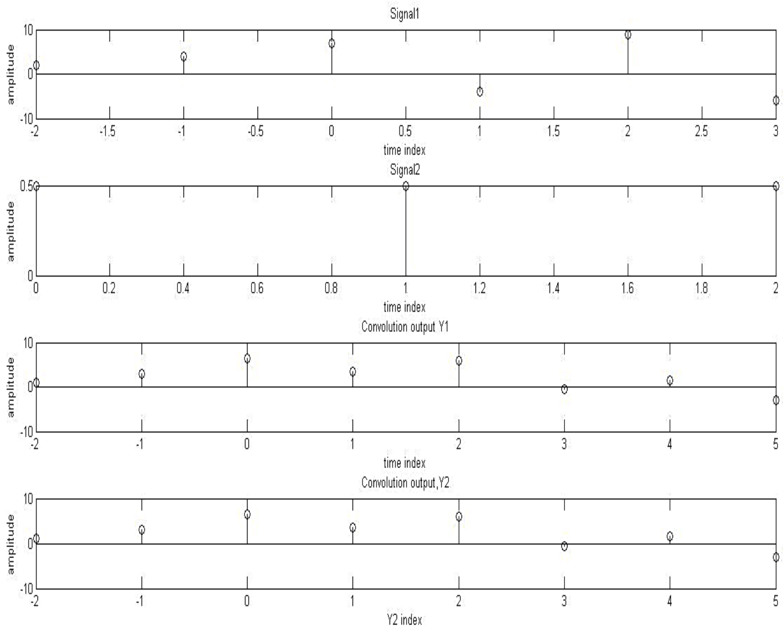
1) Commutative property: x conv h= h conv x

```
%This program uses non-causal signals
clc; clear all;close all;
x=input('Enter the 1st sequence:');
xL=input('Enter lower limits of x1:');
xH=input('Enter upper limits of x1:');
h=input('Enter the impulse response of system:');
hL=input('Enter lower limits of x2:');
hH=input('Enter upper limits of x2:');
Y1=conv(x,h);% output of LHS of commutative property
disp('Y1='); disp(Y1);
Y1L=xL+hL; Y1H=xH+hH;
subplot(4,1,1);stem(xL:xH,x); title('Signal1'); xlabel('time index');ylabel('amplitude');
subplot(4,1,2);stem(hL:hH,h); title('Signal2’); xlabel('time index');ylabel('amplitude');
subplot(4,1,3);stem(Y1L:Y1H,Y1); title('Convolution output Y1');xlabel('time
index');ylabel('amplitude');
Y2= conv(h,x); % output of RHS of commutative property
disp('Y2=');disp(Y2);
Y2L=hL+xL; Y2H=hH+xH;
subplot(4,1,4);stem(Y2L:Y2H,Y2); title('Convolution output,Y2');
xlabel('Y2 index');ylabel('amplitude');
```

Expected Output:

Enter the 1st sequence:[2 4 7 -4 9 -6]
Enter lower limits of x1:-2
Enter upper limits of x1:3
Enter the impulse response of system:[1/3 1/3 1/3]
Enter lower limits of x2:0
Enter upper limits of x2:2
Y1=
    0.6667   2.0000   4.3333   2.3333   4.0000   -0.3333   1.0000   -2.0000
Y2=

    0.6667   2.0000   4.3333   2.3333   4.0000   -0.3333   1.0000   -2.0000



Distributive Property: x conv (h1 + h2) **=** x conv h1 + x conv h2

```
clc; clear all; close all;
x=input('Enter the 1st sequence:');
h1=input('Enter the impulse response of 1st system:');
h2=input('Enter the impulse response of 2nd system:');
LHS= conv(x,(h1 + h2));
disp('LHS='); disp(LHS);
RHS1= conv(x,h1);
RHS2 = conv(x,h2);
RHS = RHS1 + RHS2 ;
disp('RHS='); disp(RHS);
```

```
if(LHS==RHS)
disp('Distributive Property is proved')
else
disp('Distributive Property is not proved');
end
```

Expected Output:

```
Enter the 1st sequence:[1 2 1]
Enter the impulse response of 1st system:[2 2 3 3]
Enter the impulse response of 2nd system:[1 2 1 2]
LHS=
    3    10    15    17    14     5
RHS=
    3    10    15    17    14     5
Distributive Property is proved
```

Associative property: (x conv h1) conv h2 = x conv (h1 conv h2)

```
clc; clear all; close all;
x=input('Enter the 1st sequence:');
h1=input('Enter the impulse response of 1st system:');
h2=input('Enter the impulse response of 2nd system:');
LHS1= conv(x,h1);
m=length(LHS1);
n=length(h2);
X=[LHS1,zeros(1,n)];  H=[h2,zeros(1,m)];
LHS=conv(X,H) ;
disp('LHS='); disp(LHS);
RHS1= conv(h1,h2);
m1=length(RHS1); n1=length(x);
X1=[LHS1,zeros(1,n1)]; H1=[h2,zeros(1,m1)];
RHS = conv(X1,H1);
disp('RHS='); disp(RHS);
if(LHS==RHS)
disp('Associative Property is proved')
else
disp('Associative Property is not proved');
end
```

Expected Output:

```
Enter the 1st sequence:[1 2 1]
Enter the impulse response of 1st system:[1 2 1 2]
Enter the impulse response of 2nd system:[2 2 3 3]
LHS=
    2   10   23   39   52   50   37   21    6    0    0    0    0    0    0    0    0    0    0
RHS=
    2   10   23   39   52   50   37   21    6    0    0    0    0    0    0    0    0    0    0
Associative Property is proved
```

Alternate Program/Assignment:

Perform convolution using user defined function/simulink.

Main program	**Function Program**

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 5
Verification of system properties

Aim: To verify various properties of a LTI system.

Objective:

To verify linearity, causality and stability properties of a system using MATLAB.

Theory:

Linearity: A system is said to be linear if it satisfies the superposition principal. Superposition principal state that Response to a weighted sum of input adequate to the corresponding weighted sum of the outputs of the system to every of the individual input signals.
If x(n) is input and y(n) is output of a system, then

y(n)=T[x(n)], y1(n) =T[x1(n)] and y2(n)=T[x2(n)]

if, x3 =[a*x1(n) + b *x2(n)] then y3(n) = T [x3(n)]

T [a*x1(n) + b*x2(n)] = a y1(n) + b y2(n)

Time-Invariant System: A system is named time-invariant if its input-output characteristics don't change with time.
Let x(t) be the input and y(t) is the output, x(t-k) be a delayed form of input by k seconds and y(t-k), delayed form of output by k seconds.

Then for y(t)=T[x(t)] if y(t-k)=T[x(t-k)] then system is time invariant system.

Static and Dynamic: A system is called static or memoryless if its output at any instant depends on the input at that instant but not on past or future values of input. Otherwise the system is said to be dynamic or with memory.

Ex. y(n) = nx(n) is a static system as its output depends only on present input value

y(n) = x(n-1) is a dynamic system as its output depends on past values of input.

Reference MATLAB Code:

Linearity Property:

% a) y(n)=nx(n) b) y=x^2(n)
clc; close all; clear all;
n=0:40;
a1=input('enter the scaling factor a1=');
a2=input('enter the scaling factor a2=');
x1=cos(2*pi*0.1*n); x2=cos(2*pi*0.4*n);
x3=a1*x1+a2*x2;
%y(n)=n.x(n);
y1=n.*x1; y2=n.*x2; y3=n.*x3;
yt=a1*y1+a2*y2;
yt=round(yt);
	y3=round(y3);
	if y3==yt
	disp('given system [y(n)=n.x(n)]is Linear');
	else
	disp('given system [y(n)=n.x(n)]is non Linear');
	end

%y(n)=x(n).^2
x1=[1 2 3 4 5]; x2=[1 4 7 6 4];
x3=a1*x1+a2*x2;
y1=x1.^2;
y2=x2.^2;
y3=x3.^2;
yt=a1*y1+a2*y2;
if y3==yt
disp('given system [y(n)=x(n).^2]is Linear');
else
disp('given system is [y(n)=x(n).^2]non Linear');
end
	Expected Output:
	Output will obtained as
	Enter the scaling factor a1=3
	Enter the scaling factor a2=5
	Given system [y(n) =n.x(n)]is Linear
	Given system is [y(n) =x(n).^2] non Linear

Time Invariant Property

% a)y=x^2(n) b) y(n)=nx(n)
clc; clear all; close all;
n=1:9;
x(n)=[2 1 4 3 6 5 8 7 9];
d=3; % Time delay
xd=[zeros(1,d),x(n)]; %x(n-k)
y(n)=x(n).^2;
yd=[zeros(1,d),y];%y(n-k)
disp('transformation of delay signal yd:'); disp(yd)
dy=xd.^2; % T[x(n-k)]
disp('delay of transformation signal dy:'); disp(dy)
if dy==yd
disp('given system is time invariant');
else
disp('given system is not time invariant');
end
y=n.*x;
yd=[zeros(1,d),y(n)];
disp('transformation of delay signal yd:');disp(yd);
n1=1:length(xd);
	dy=n1.*xd;
	disp('delay of transformation signal dy:');disp(dy);
	if yd==dy
	disp('given system [y(n)=nx(n)]is a time invariant');
	else
	disp('given system [y(n)=nx(n)]not a time invariant');
	end
	Expected Output:
	Transformation of delay signal yd:
	0 0 0 4 1 16 9 36 25 64 49 81
	Delay of transformation signal dy:
	0 0 0 4 1 16 9 36 25 64 49 81
	Given system [y(n)=x(n).^2]is time invariant
	Transformation of delay signal yd:
	0 0 0 2 2 12 12 30 30 56 56 81
	Delay of transformation signal dy:
	0 0 0 8 5 24 21 48 45 80 77 108
	Given system [y(n)=nx(n)]not a time invariant

Alternate Program/Assignment

Verify system properties for system (System to be given by the lab teacher)

Manual Calculations/Observation:

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 6
Auto and Cross Correlation

Aim: To perform auto correlation of a signal and cross correlation operation between two signals and verify properties.

Objectives:

a. To compute cross correlation of two DT signals and verify using MATLAB
b. To verify properties of cross correlation.
c. To compute auto correlation of a given DT signal and verify using MATLAB
d. To verify properties of auto correlation.

Theory:

Correlation of sequences:

It is a measure of the degree to which two sequences are similar. Given two real valued Sequences x(n) and y(n) of finite energy, correlation mathematically involves the following operations .

1. Shifting 2. Multiplication 3. Addition

There are two types of correlation operations: Auto-correlation and Cross-correlation.

Cross-correlation

In signal processing, cross-correlation is a measure of similarity of two waveforms as a function of a time-lag applied to one of them. This is also known as a sliding dot product or inner-product. It is commonly used to search a long duration signal for a shorter, known feature.

The cross correlation operation can be represented by a mathematical expression as follows:

$R_{xy}(l) = \sum_{n=-\infty}^{\infty} x(n)y(n-l)$ --- (1)

Where l is known as shifting parameter or lag index which indicates the time shift between the pair.

The properties of cross correlation are

1. The cross correlation sequence sample values are upper bounded by the inequality

$$R_{xy}(l) \leq \sqrt{E_x E_y} \text{ where } E_x \text{ and } E_y \text{ are energies of x and y}$$

2. The cross correlation of two sequences x[n] and y[n] = x[n-k] shows a peak at the value at k. Hence cross correlation is employed to compute the exact value of the delay k between the two signals. Used in radar and sonar applications, where the received signal reflected from the target is the delayed version of the transmitted signal (measure delay to determine the distance of the target).
3. The ordering of the subscripts xy specifies that x[n] is the reference sequence that remains fixed in time, whereas the sequence y[n] is shifted w.r.t x[n]. If y[n] is the reference sequence then $r_{yx}[l]$ is obtained by time reversing the sequence $r_{xy}[l]$ (i.e $R_{yx}[l] = R_{xy}[-l]$).

Note: When convolution process and correlation process are compared a close resemblance is observed.

Cross correlation: $R_{xy}(l)=x(l) * y(-l)$

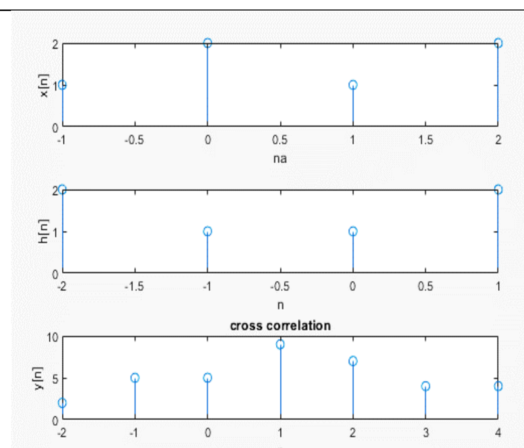
Auto correlation: $R_{xx}(l) = x(l) * x(-l)$

Cross correlation and verification of properties

Main program	**Function program** **Save the program with name 'convm'**

clc; close all; clear all;
x=input('Enter the first sequence x= ');
nx=input('Enter the time index for the first sequence nx= ');
h=input('Enter the second sequence h= ');
nh=input('Enter the time index for the second sequence nh= ');
[y,ny]=convm(x,nx,fliplr(h),-fliplr(nh))
%Verification of property1
Ex=sum(x.^2) ;
Eh=sum(h.^2) ;
Z=sqrt(Ex*Eh)
disp('Compare each element of y with Z to verify property1')
subplot(311);stem(nx,x);xlabel('nx');ylabel('x[n]');
subplot(312);stem(nh,h);xlabel('nh');ylabel('h[n]');
subplot(313);stem(ny,y);xlabel('ny');ylabel('y[n]');
title('cross correlation');
	function[y,ny]=convm(x,nx,h,nh)
	nyb=nx(1)+nh(1);
	nye=nx(length(x))+nh(length(h));
	ny=nyb:nye;
	y=conv(x,h);

Expected Output

Enter the first sequence x= [1 2 1 2] Enter the time index for the first sequence nx= -1:2 Enter the second sequence h= [2 1 1 2] Enter the time index for the second sequence nh= -2:1 y = 2 5 5 9 7 4 4 ny = -2 -1 0 1 2 3 4 Z = 10 **Compare each element of y with Z to verify property1**	

Auto correlation

It is correlation of signal with itself. Auto-correlation is a mathematical tool for finding repeating patterns, such as the presence of a periodic signal which has been buried under noise, or identifying the missing fundamental frequency in a signal implied by its harmonic frequencies. It is used frequently in signal processing for analyzing functions or series of values, such as time domain signals. Informally, it is the similarity between observations as a function of the time separation between them. More precisely, it is the cross-correlation of a signal with itself.

If x=y in above equation (1), it results in to auto correlation.

$$\text{i.e.} \quad R_{xx}(l) = \sum_{n=-\infty}^{\infty} x(n)x(n-l)$$

Some of the properties of autocorrelation are enumerated below

1. The autocorrelation sequence is an even function i.e., $r_{xx}[l] = r_{xx}[-l]$
2. At zero lag, i.e., at l =0, the sample value of the autocorrelation sequence has its maximum value equal to the total energy of the signal E_x

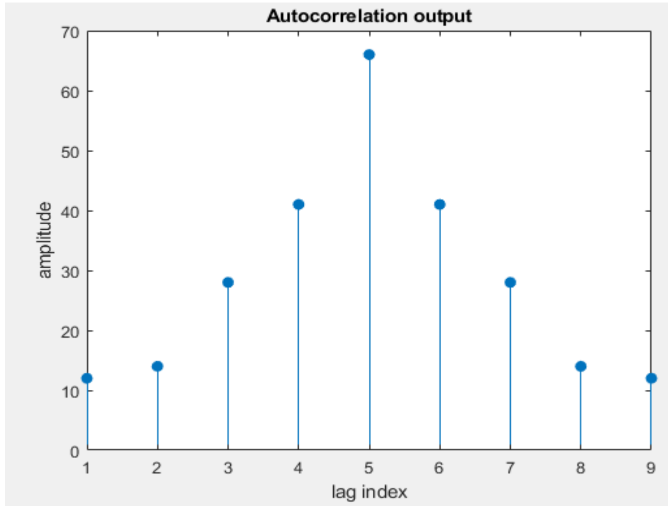
   $$\text{i.e} \quad R_{xx}[l] \le r_{xx}[0] = E_x = \sum_{n=-\infty}^{\infty} | x^2(n) | .$$

   A time shift of a signal does not change its autocorrelation sequence.
3. For example, let y[n]=x[n-k]; then $r_{yy}[l] = r_{xx}[l]$ i.e., the autocorrelation of x[n] and y[n] are the same regardless of the value of the time shift k. This can be verified with sine and cosine sequences of same amplitude and frequency will have identical autocorrelation functions.

Reference MATLAB Code:

Auto correlation and verification of properties:
clc;clear all;close all;
x=input ('Enter sequence x(n)=');
Rxx=xcorr(x,x);
%Rxx= conv(x,fliplr(x));instead of above line we can use this also
disp('Rxx=');
disp(Rxx);
figure(1);
stem(Rxx,'filled');   title('Autocorrelation output');
xlabel('lag index');   ylabel('amplitude');
	%verification of properties
	Energy=sum(x.^2)
	center_index=ceil(length(Rxx)/2)
	Rxx_0=Rxx(center_index)
	if Rxx_0==Energy
	disp('Rxx(0) gives energy - proved');
	else
	disp('Rxx(0) gives energy - not proved');
	end
	Rxx_right=Rxx(center_index:1:length(Rxx))
	Rxx_left=Rxx(center_index:-1:1)
	if Rxx_right == Rxx_left
	disp('Rxx is even');
	else
	disp('Rxx is odd');
	end

Expected Output:

Enter sequence x(n)=[2 1 3 4 6] rxx= 12 14 28 41 66 41 28 14 12 Energy = 66 center_index = 5 Rxx_0 = 66 Rxx(0) gives energy - proved Rxx_right = 66 41 28 14 12 Rxx_left = 66 41 28 14 12 Rxx is even	

Alternate Program/Assignment

1. Verify remaining cross correlation properties.
2. Perform correlation between two sinusoidal signals shifted by half a period.
3. Find the correlation between two sequences given by [0 0 0 0 1 0 0 1 0 1 1] and [0 0 1 0 0 1 0 1 1] and find and display the relation between them.

Manual Calculations/Observation:

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 7
Solving difference equation

Aim: To solve given difference equation with initial conditions which represents an LTI system and find the output of the system y(n) for a given input.

Objectives:

a. Solving given difference equation for given initial conditions and input.
b. Finding impulse response of the system described by a difference equation.

Theory:

A difference equation with constant coefficients describes an LTI system. For example the difference equation $y(n) + 0.8y(n-2) + 0.6y(n-3) = x(n) + 0.7x(n-1) + 0.5x(n-2)$ describes an LTI system of order 3. The coefficients 0.8, 0.7, etc are all constant i.e., they are not functions of time (n). The difference equation $y(n) + 0.3ny(n-1) = x(n)$ describes a time varying system as the coefficient $0.3n$ is not a constant.

The difference equation can be solved to obtain$y(n)$, the output for a given input $x(n)$ by rearranging as $y(n) = x(n) + 0.7x(n-1) + 0.5x(n-2) - 0.8y(n-2) - 0.6y(n-3)$.

The output depends on the input $x(n)$

With

- $x(n) = \delta(n)$ an impulse input, the computed output $y(n)$ is the impulse response.
- $x(n) = u(n)$ a step input, the computed output $y(n)$ is the step response response.
- $x(n) = \cos(\Omega n)$, a sinusoidal input, a steady state response is obtained (wherein $y(n)$ is of the same frequency as $x(n)$, with only an amplitude gain and phase shift).

Similarly for any arbitrary sequence of $x(n)$, the corresponding output response $y(n)$ is computed.

The difference equation containing past samples of output, i.e., $y(n-1)$, $y(n-2)$, etc., leads to a recursive system, whose impulse response is of infinite duration (IIR). For such systems the impulse response is computed for a large value of n, say n =100 (to approximate $n = \infty$. The MATLAB function $filter(b, a, x)$ is used to compute the impulse response/ step response/ response to any given $x(n)$ where b & a are the coefficients of $x(n)$ & $y(n)$ respectively of a difference equation defining a Discrete Time (DT) LTI system. Note: The filter function evaluates the convolution of an infinite sequence (IIR) and $x(n)$, which is not possible with $conv$ function (remember $conv(x, h)$ function requires both the sequences to be finite).

The difference equation having only $y(n)$ and present and past samples of input $x(n),\ x(n-k)$ represents a system whose impulse response is of finite duration (FIR). The response of FIR systems can be obtained by both the '$conv$' and '$filter$' functions. The filter function results in a response whose length is equal to that of the input $x(n)$, whereas the output sequence from '$conv$' function is of a longer length (x length + h length-1).

Reference MATLAB Code:

Solve the given difference equation for the given specifications.

$y(n) = x(n) + \frac{3}{2}y(n-1) - \frac{1}{2}y(n-2)$ where $x(n) = \left(\frac{1}{4}\right)^n u(n)$. The initial conditions are $y(-1) = 4, y(-2) = 10.$

```
clc; clear all; close all;
a = input ( 'Enter the co-efficient of y(n),y(n-1)......= ' );
b = input ( 'Enter the co-efficient of x(n),x(n-1)......= ' );
xi = input ( 'Enter the initial conditions x(-1),x(-2)... = ' );
yi = input ('Enter initial conditions y(-1),y(-2)....= ');
N = input ( 'Enter the length of the response required…’);
zi = filtic ( b, a, yi, xi );
n =  0 : N-1;    %For N output samples using filter function, the i/p should also be of size N
x = (1/4).^n     % input sequence
y = filter ( b, a, x, zi )   % output response
subplot ( 2, 1, 1 );stem ( n, x );
title ( 'input' ); xlabel ( 'n' ); ylabel ('x(n)' );
subplot (2, 1, 2 );stem ( n, y );
title ( 'Res of DE with ICs  ' );xlabel ( 'n' );ylabel ( 'y(n)' );
```

Expected Output:

Enter the co-efficient of y(n), y(n-1)......= [1   -3/2   1/2]

Enter the co-efficient of x(n), x(n-1)......= [1]

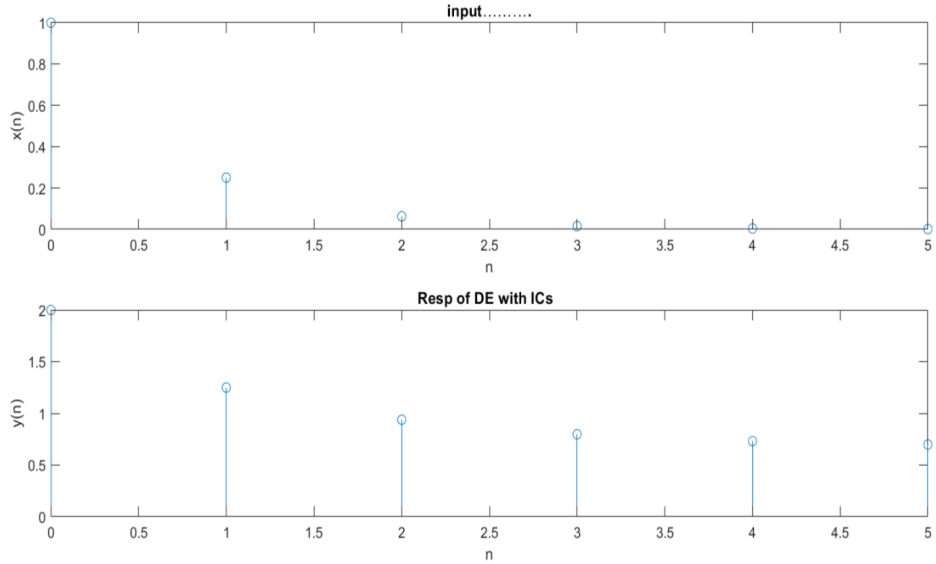
Enter the initial conditions x(-1), x(-2)... = [ ]

Enter initial conditions y(-1), y(-2)....= [4 10]

Enter the length of the response required…6

x =   1.0000   0.2500   0.0625   0.0156   0.0039   0.0010

y =   2.0000   1.2500   0.9375   0.7969   0.7305   0.6982



Alternate Program/Assignment

1. Find the impulse response, step response of a relaxed system described by the difference equation $y(n) = \frac{1}{3}[x(n) + x(n-1) + x(n-2)] + 0.95\, y(n-1) + 0.9025\, y(n-2)$, $n \geq 0$.
2. Solve the difference equation for the given specification.
   $y(n) = \frac{1}{3}[x(n) + x(n-1) + x(n-2)] + 0.95\, y(n-1) + 0.9025\, y(n-2), \quad n \geq 0$
   for the input $x(n) = \cos\left(\frac{\pi n}{3}\right) u(n)$ and $y(-1) = -2 \;\; y(-2) = -3; \;\; x(-1) = 1, x(-2) = 1$.

Manual calculation/Observation

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 8
System Analysis

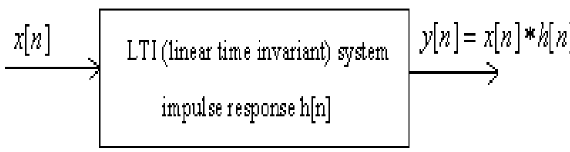
Aim: To find transfer function, frequency response, impulse response and system response of a system defined by difference equation and to plot poles and zeros.

Objectives:

Given an LTI system in the form of LCCDE

a. Find impulse response and transfer function of the system and to plot poles and zeros.
b. Frequency response and response of the system for given input.

Theory:



A complete characterization of any LTI system can be represented in terms of its response to a unit impulse, which is referred to as *Impulse response* of the system. Alternatively, the impulse response is the output of an LTI system due to an impulse input applied at $t = 0$, or $n = 0$.
A discrete time LTI system (also called digital filters) is represented by

- A linear constant coefficient difference equation, for example,
$$y(n) + a_1 y(n-1) + a_2 y(n-2) = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2)$$
- A system function $H(z)$ (obtained by applying Z transform to the difference equation) given as
$$H(z) = \frac{Y(z)}{X(z)} == \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots}$$

Given the difference equation or transfer function $H(z)$, the impulse response of the LTI system is found using '*filter*' or '*impz*' MATLAB functions.

Reference MATLAB Code:

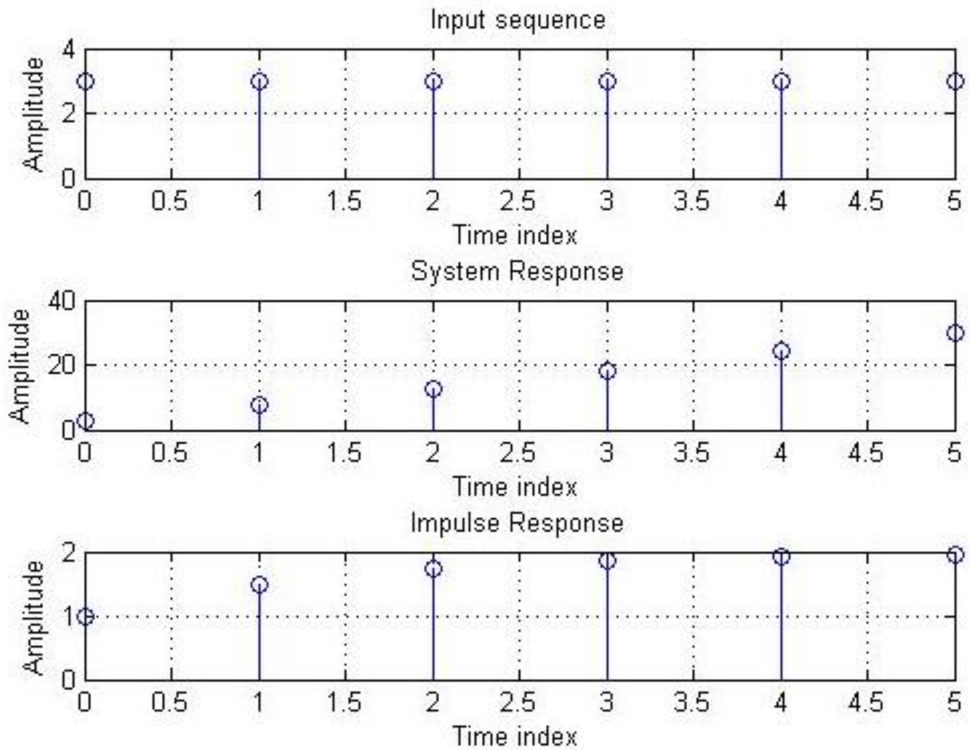
Compute and plot the impulse response, system response, frequency response and pole-zero for the system described by .

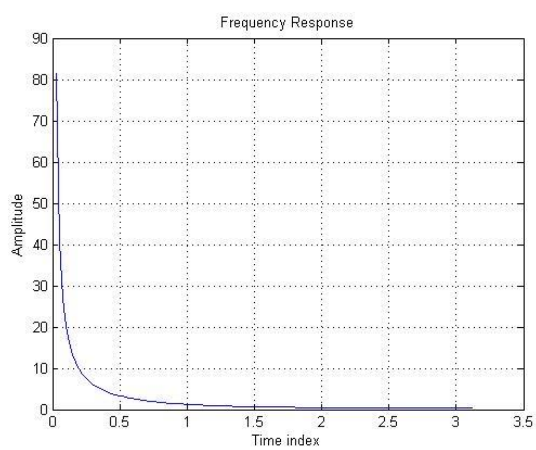
$$y(n) - \frac{3}{4}y(n-1) + \frac{1}{8}y(n-2) = x(n) - \frac{1}{3}x(n-1) \quad \text{where } x(n) = 3u(n)$$

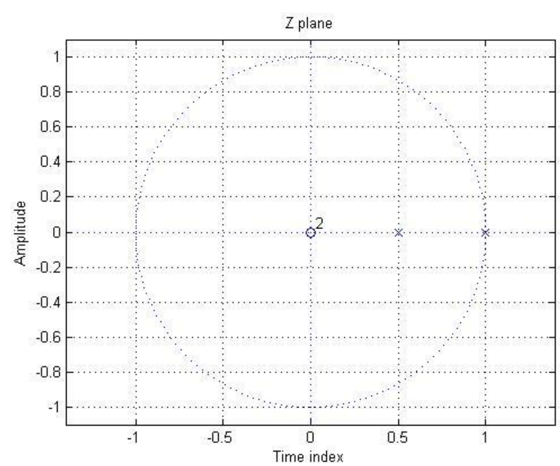
```
clc; clear all; close all;
N=input('Enter the length of impulse response =');
b=input('Enter the numerator coeff = ');
a=input('Enter the denominator coeff = ');
n=0:N-1;
x=3*ones(1,N);                  % input sequence
y=filter(b,a,x)                 % system response
[h,t]=impz(b,a,N)               % impulse response
[r,p,k]=residuez(b,a)           % computing poles and zeroes
[H,w]=freqz(b,a,128);           % frequency response
H_mag=abs(H);
figure(1);subplot(3,1,1);stem(n,x);grid;
xlabel('Time index');ylabel ('Amplitude');title('Input sequence');
subplot(3,1,2);stem(n,y);grid;
xlabel('Time index');ylabel('Amplitude');title('System Response');
subplot(3,1,3);stem(t,h);grid;
xlabel('Time index');ylabel('Amplitude');title('Impulse Response');
figure(2);zplane(b,a);grid;
xlabel('Time index');ylabel('Amplitude');title('Z plane');
figure(3); plot(w, H_mag); grid;
xlabel('Time index'); ylabel('Amplitude');title('Frequency Response');
```

Output

Enter the length of impulse response =6	t =
Enter the numerator coeff = [1]	0
Enter the denominator coeff = [1 -3/2 1/2]	1
Enter the length of impulse response =6	2
Enter the numerator coeff = 1	3
Enter the denominator coeff = [1 -3/2 1/2]	4
y =	5
3.0000 7.5000 12.7500 18.3750 24.1875	r =
30.0938	2
h =	-1
1.0000	p =
1.5000	1.0000
1.7500	0.5000
1.8750	k =
1.9375	[]
1.9688	







Alternate Program/Assignment:

Analyze the following systems

- $\boldsymbol{y(n) - \frac{3}{4}y(n-1) + \frac{1}{8}y(n-2) = x(n) - \frac{1}{3}x(n-1)}$
- $\boldsymbol{y(n) = 0.5x(n) + 0.5x(n-1)}$

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 9
Fourier series Analysis

Aim: Finding Fourier series (FS) of a signal and verification of its properties.

Objectives:

a. To find Fourier series of a DT signal and plot the spectra
b. To verify the properties of Fourier series.

Theory

The Fourier Series:

The FS is a method to analyze CTP signals in frequency domain. The FS of a finite power signal x(t) with fundamental period T is represented as X(k) or Ck and is given by

$$\mathrm{X(k)} = \frac{1}{T}\int_0^T x(t)e^{-jk\omega_o t}dt$$

FS is also discrete in frequency and non-periodic and hence the signal x(t) may be obtained from its FS using the inverse FS given by

$$x(t) = \sum_{k=-\infty}^{\infty} X(k)\, e^{jk\omega_o t}$$

Here $\omega_o$is angular frequency of the signal $\omega_o = 2\pi/T$

The Discrete Time Fourier Series (DTFS)

The FS is a method to analyze DTP signals in frequency domain. The DTFS of a finite power signal x(n) with fundamental period N is represented as X(k) or Ck and is given by

$$\mathrm{X(k)} = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\, e^{-jk\Omega_o n}$$

DTFS is also discrete in frequency and periodic with fundamental period N and hence the signal x(n) may be obtained from its DTFS using the inverse DTFS given by

$$x(n) = \sum_{k=0}^{N-1} X(k)\, e^{jk\Omega_o n}$$

Here $\Omega_o$is angular frequency of the signal $\Omega_o = 2\pi/N$

Reference MATLAB Code:

```
clc; clear all; close all;
N=input('Enter the fundamental period of the signal (N)');
M=input('Enter how many points of Fourier series to compute(M=N or M=multiples of N)  ')
n=0:N-1;
x=[sin(2*pi*n/6)];
for k=0:M-1;
Sum=0;
   for n=0:N-1;
      Ck(k+1)=x(n+1)*exp(-j*2*pi*k*n/N);
      Sum=Sum+Ck(k+1);
   end
   Ck(k+1)=Sum/N;
end
Mag=abs(Ck);Phase_ang=angle(Ck);
disp(Ck);disp(Mag);disp(Phase_ang);
subplot(311);stem([0:length(x)-1],x);xlabel('n-->');ylabel('x(n)');title('Input signal')
subplot(312);stem([0:length(Ck)-1],abs(Ck));xlabel('k');ylabel('Magnitude');title('Magnitude
Spectrum');
subplot(313);stem([0:length(Ck)-1],Phase_ang); xlabel('k');ylabel('Phase angle');title('Phase
Spectrum')
```

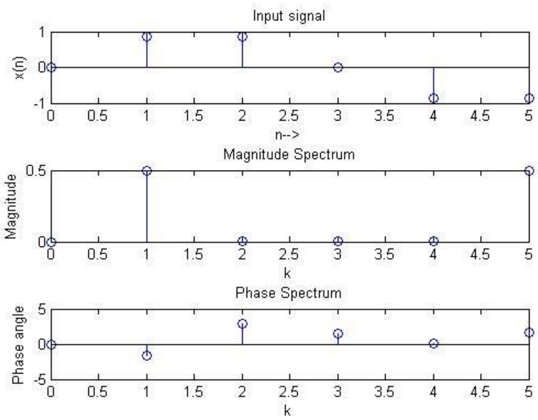
Expected Output

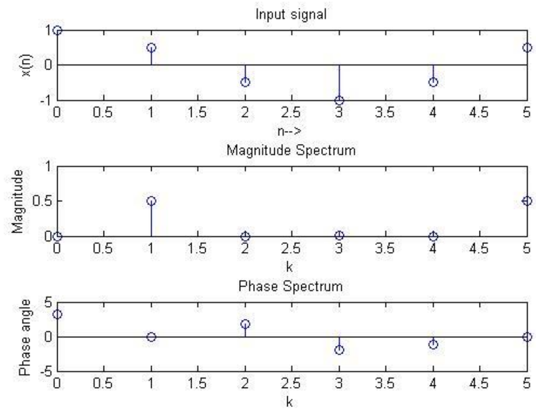
For x(n)=sin(2(pi)n/N)

```
Enter the fundamental period of the signal (N)6
'Enter how many points of Fourier series to compute (M=N or M=multiples of N)  6
M =     6
0.0000 + 0.0000i    0.0000 - 0.5000i   -0.0000 + 0.0000i    0.0000 + 0.0000i    0.0000 + 0.0000i
0.0000 + 0.5000i
0.0000    0.5000    0.0000    0.0000    0.0000    0.5000
0   -1.5708    2.9229    1.5075    0.0588    1.5708
```

For x(n)=cos(2(pi)n/N)

```
Enter the fundamental period of the signal (N)6
Enter how many points of Fourier series to compute (m>N and m=multiples of N)  6
M =     6
-0.0000 + 0.0000i   0.5000 - 0.0000i  -0.0000 + 0.0000i  -0.0000 - 0.0000i   0.0000 - 0.0000i
0.5000 + 0.0000i
0.0000    0.5000    0.0000    0.0000    0.0000    0.5000
3.1416   -0.0000    1.7943   -1.9871   -1.1071    0.0000
```





Assignment:

1. Verify the periodicity of DTFS by giving M>N.
2. Verify the DTFS for signals with multiple frequencies.
3. Write MATLAB code to find inverse DTFS
4. Verify the time shifting property of DTFS

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 10
Fourier Transformation

Aim: Study of Fourier transform representation of a signal.

Objectives:

a. To perform computation of Fourier Transform of a signal
b. To perform computation of Discrete Time Fourier Transform of a signal.
c. Verification of properties of DTFT.

Theory

The Fourier Transform (FT)

The FT is a method to analyze CTNP signals in frequency domain. The FT of a finite energy signal x(t) is represented as X(Ω) given by

$$\mathrm{X}(\Omega) = \int_{-\infty}^{\infty} x(t) e^{-j\Omega t} dt$$

FT is also continuous in frequency and non-periodic and hence the signal x(t) may be obtained from its FT using the inverse FT given by

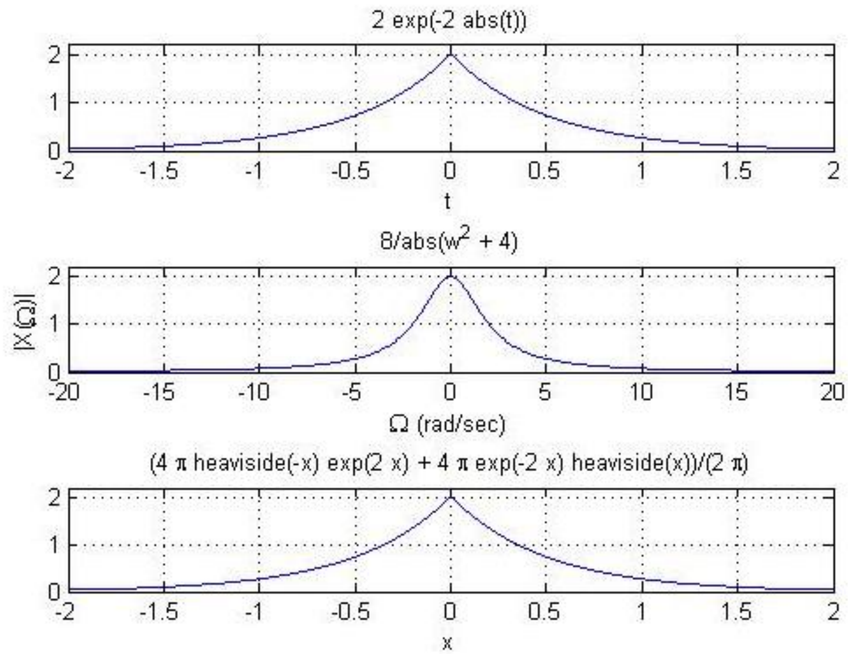
$$x(t) = \int_{-\infty}^{\infty} \mathrm{X}(\Omega) e^{j\Omega t} d\Omega$$

Reference MATLAB code

```
%To find Fourier transform of a signal and get back the time domain signal from it
clc; clear all;close all;
syms t w x; %creating symbolic variables
x=2*exp(-2*abs(t))
X=fourier(x)
figure(1);
subplot(311);ezplot(x,[-2,2]); grid; axis([-2 2 0 2.2])
subplot(312); ezplot(abs(X),[-30,30]);grid;axis([-20 20 0 2.2]);
xlabel('\Omega (rad/sec)'); ylabel('|X(\Omega)|')
x1=ifourier(X)
subplot(313);ezplot(x1,[-2,2]); grid; axis([-2 2 0 2.2])
```

Expected output

```
x =
 2*exp(-2*abs(t))
 X =
 8/(w^2 + 4)
 x1 =
 (4*pi*heaviside(-x)*exp(2*x) + 4*pi*exp(-2*x)*heaviside(x))/(2*pi)
```



The Discrete-Time Fourier Transform (DTFT)

If x(n) is absolutely summable signal then its discrete time Fourier transform is given by

$$X(e^{j\omega}) \triangleq \mathcal{F}[x(n)] = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n}$$

DTFT is periodic with fundamental period 2π radian. Hence x(n) may be obtained from DTFT using inverse discrete-time Fourier transform (IDTFT) of $X(e^{j\omega})$ which is given by

$$x(n) \triangleq \mathcal{F}^{-1}[X(e^{j\omega})] = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}d\omega$$

Two important properties of DTFT are

1. **Periodicity:** The discrete-time Fourier transform X(ejω) is periodic in ω with period 2π. $X(e^{j\omega}) = X(ej[\omega+2\pi])$.
   Implication: We need only one period of $X(e^{j\omega})$ (i.e., $\omega \in [0, 2\pi]$, or $[-\pi, \pi]$, etc.) for analysis and not the whole domain $-\infty < \omega < \infty$.
2. **Symmetry:** For real-valued x(n), $X(e^{j\omega})$ is conjugate symmetric i.e.

$$X(e^{-j\omega}) = X^*(e^{j\omega})$$

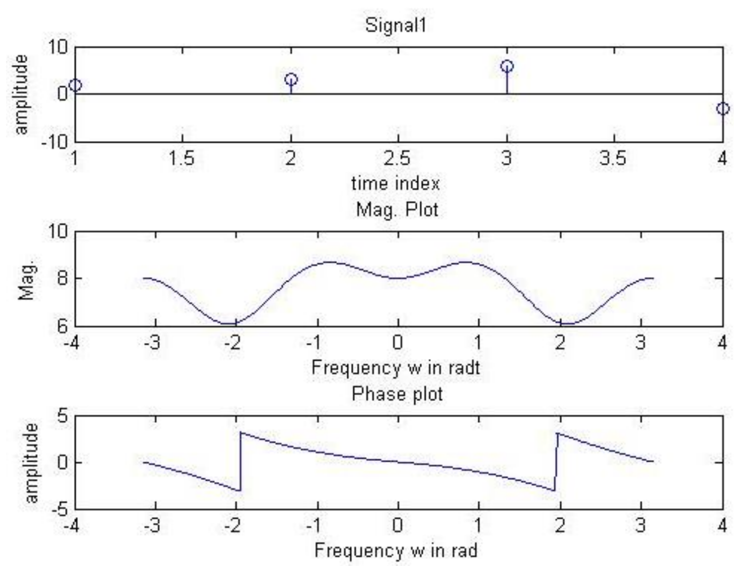
$$\begin{aligned}
\mathrm{Re}[X(e^{-j\omega})] &= \mathrm{Re}[X(e^{j\omega})] \quad \text{(even symmetry)} \\
\mathrm{Im}[X(e^{-j\omega})] &= -\,\mathrm{Im}[X(e^{j\omega})] \quad \text{(odd symmetry)} \\
|X(e^{-j\omega})| &= |X(e^{j\omega})| \quad \text{(even symmetry)} \\
\angle X(e^{-j\omega}) &= -\angle X(e^{j\omega}) \quad \text{(odd symmetry)}
\end{aligned}$$

Implication: To plot $X(e^{j\omega})$, we now need to consider only a half period of $X(e^{j\omega})$. Generally, in practice this period is chosen to be $\omega \in [0, \pi]$.

Reference MATLAB code

Main Program	**Function program**
clc; clear all; close all;	function [X] = dtft(x,w)
x=[2 3 6 -3];	% Computes Discrete-time Fourier Transform
n=0:length(x)-1;	% X = DTFT values computed at w frequencies
w=-pi:0.01:pi;	% x = finite duration sequence over n
[X] = dtft(x, w);	% w = frequency location vector
X_mag=abs(X); X_angle= angle(X);	for i=0:length(w)-1
subplot(311);stem(n,x);title('Signal');	X(i)=0;
xlabel('time index');ylabel('amplitude');	for k=0:length(x)-1
subplot(312);plot(w, X_mag);	X(i+1) =X(i+1)+ exp(-1i*w(i)*k)* x(k+1);
title("Mag. Plot"); xlabel('Frequency w in rad');	end
ylabel('Mag.');	end
subplot(313);plot(w, X_angle);	end
title('Phase plot');xlabel('Frequency w in rad');	
ylabel('Phase angle');	

Expected Output



Assignment:

1. **Periodicity property:** Change w vector from –3_pi to 3* pi and run the program and observe output.
2. **Conjugate Symmetry Property:** Verify whether the magnitude spectrum is even symmetric and phase spectrum is odd symmetric?
3. Verify time shifting property of DTFT.

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 11
N-point DFT

Aim: Study of Discrete Fourier Transform and use it to analyze a signal.

Objectives:

a. To compute Discrete Fourier Transform of a signal without using built in command and plot magnitude and phase spectra.
b. To compute Discrete Fourier Transform of a signal consisting of multiple frequencies, using built-in command and analyzing the signal.
c. To analyze a noisy signal using Discrete Fourier Transform.

Theory:

- Discrete Fourier Transform (DFT) is used for performing frequency analysis of discrete time signals. DFT gives a discrete frequency domain representation whereas the other transforms are continuous in frequency domain.
- The N point DFT of discrete time signal x[n] is given by the equation

$$X(k) = \sum_{n=0}^{N-1} x[n] e^{\frac{-j2\pi kn}{N}}; \quad k = 0,1,2,....N-1$$

Where N is chosen such that $N \geq L$, where L=length of x[n].

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \quad 0 \leq k \leq N-1$$

- The inverse DFT allows us to recover the sequence x[n] from the frequency samples.

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} x[n] e^{\frac{j2\pi kn}{N}}; \quad n = 0,1,2,....N-1$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-kn}, \quad 0 \leq n \leq N-1$$

- X(k) is a complex number (e^{jw}=cosw + jsinw). It has both magnitude and phase which are plotted versus k. These plots are magnitude and phase spectrum of x[n]. The 'k' gives us the frequency information.
- Here k=N in the frequency domain corresponds to sampling frequency (fs). Increasing N, increases the frequency resolution, i.e., it improves the spectral characteristics of the sequence. For example if fs=8kHz and N=8 point DFT, then in the resulting spectrum, k=1 corresponds to 1kHz frequency. For the same fs and x[n], if N=80 point DFT is computed,

then in the resulting spectrum, k=1 corresponds to 100Hz frequency. Hence, the resolution in frequency is increased.

Since $N \geq L$, increasing N to 80 from 8 for the same x[n] implies x[n] is still the same sequence (<8), the rest of x[n] is padded with zeros. This implies that there is no further information in time domain, but the resulting spectrum has higher frequency resolution. This spectrum is known as 'high density spectrum' (resulting from zero padding x[n]). Instead of zero padding, for higher N, if more number of points of x[n] are taken (more data in time domain), then the resulting spectrum is called a "high resolution spectrum".

Reference MATLAB Codes:

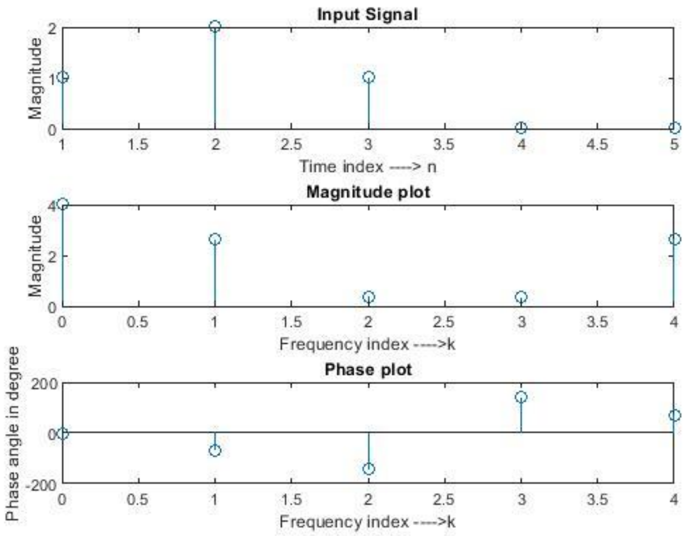
To compute Discrete Fourier Transform of a signal without using built in command and plot magnitude and phase spectra.

```
clc; close all; clear all;
xn = input('Enter the sequence for which DFT to be calculated = ');
N = input('Enter the the value of N for N-Point DFT = ');
n=[0:1:N-1];             % row vector for n
k=[0:1:N-1];             % row vecor for k
WN=exp(-j*2*pi/N);       % Wn factor
nk=n'*k;                 % creates a N by N matrix of nk values
WNnk=WN.^nk;             % DFT matrix
Xk=xn*WNnk               % row vector for DFT coefficients
% To verify IDFT
WNnkI= WN.^(-nk);        % IDFT matrix
xnI=(Xk*WNnkI)/N         % row vector for IDFT values
MagX=abs(Xk)             % Magnitude of calculated DFT
PhaseX=angle(Xk)*180/pi % Phase of the calculated DFT
%plotting the signals
subplot(3,1,1); stem(xn); xlabel('Time index ----> n');ylabel('Magnitude');title('Input Signal');
subplot(3,1,2); stem(k,MagX);
xlabel('Frequency index ---->k'); ylabel('Magnitude');title('Magnitude plot');
subplot(3,1,3); stem(k,PhaseX);
xlabel('Frequency index ---->k');ylabel('Phase angle in degree');title('Phase plot');
```

Expected Output

```
Enter the sequence for which DFT to be calculated = [1 2 1 0 0]
Enter the the value of N for N-Point DFT = 5
Xk =   4.0000    0.8090 - 2.4899i   -0.3090 - 0.2245i   -0.3090 + 0.2245i    0.8090 + 2.4899i
xnI =   1.0000 - 0.0000i   2.0000 - 0.0000i   1.0000 - 0.0000i   0 + 0.0000i    0 + 0.0000i
MagX =    4.0000          2.6180          0.3820          0.3820          2.6180
```

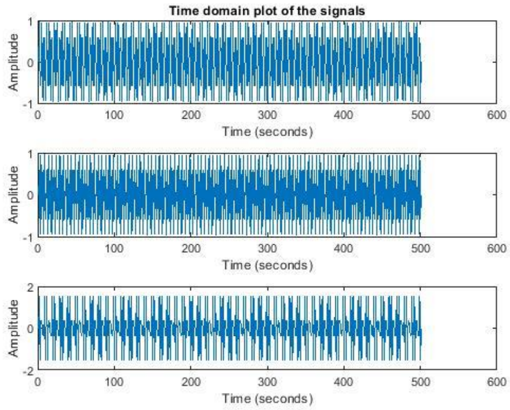
PhaseX = 0 -72.0000 -144.0000 144.0000 72.0000

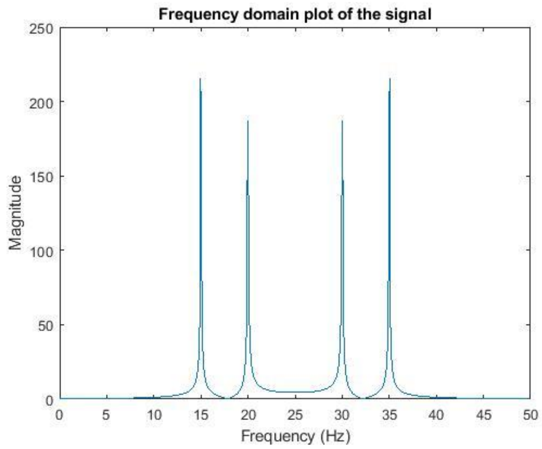


To compute Discrete Fourier Transform of a signal consisting of multiple frequencies, using built-in command and analyzing the signal.

```
clc;clear all;close all;
Fs=100; %Define a sampling frequency
Ts = 1/Fs;
t = 0:Ts:1-Ts;
f1=15;f2=20;%Define signal frequencies
%Generate a multi frequency signal
x1=sin(2*pi*15*t);x2=sin(2*pi*20*t);
x = x1+x2;
%Plot the signal
subplot(411); plot(x1);
xlabel('Time (seconds)');ylabel('Amplitude');title('Time domain plot of the signal1')
subplot(412);plot(x2);
xlabel('Time (seconds)');ylabel('Amplitude'); title('Time domain plot of the signal2')
subplot(413);plot(x);
xlabel('Time (seconds)');ylabel('Amplitude'); title('Time domain plot of the signal x')
%Find DFT of the signal
y = fft(x);
f = (0:length(y)-1)*Fs/length(y);
%Plot the spectrum of the signal
subplot(414);plot(f,abs(y));xlabel('Frequency (Hz)');ylabel('Magnitude');
title('Frequency domain plot of the signal');
```

Expected Output





Alternate Program/Assignment:

1. Observe, record and comment on values of n, k, WN, nk, WNnk in first program.

2. Analyse a noisy signal (by adding a random noise to a signal) and comment on the result

MATLAB program

Result/Outcome of the Experiment:

Signature of the faculty with date:

Experiment 12
Sampling Process

Aim: Study of sampling process and analysis in time and frequency domain.

Objectives:

a. To perform sampling of a CT signal and plot the signal and its sampled version.
b. To perform under sampling, nyquist sampling and over sampling of the CT signal and plot.
c. To analyze the frequency domain representation of the sampled signal.

Theory:

- Sampling is a process of converting a continuous time signal (analog signal) x(t) into a discrete time signal x[n], which is represented as a sequence of numbers. (A/D converter)
- Converting back x[n] into analog (resulting in $\hat{x}(t)$) is the process of reconstruction. (D/A converter)
- Techniques for reconstruction-(i) ZOH (zero order hold) interpolation results in a staircase waveform, is implemented by MATLAB plotting function ***stairs(n,x),*** (ii) FOH (first order hold) where the adjacent samples are joined by straight lines is implemented by MATLAB plotting function ***plot(n,x),***(iii) spline interpolation, etc.
- For $\hat{x}(t)$ to be exactly the same as x(t), sampling theorem in the generation of x(n) from x(t) is used. The sampling frequency fs determines the spacing between samples.
- Aliasing-A high frequency signal is converted to a lower frequency, results due to under sampling. Though it is undesirable in ADCs, it finds practical applications in stroboscope and sampling oscilloscopes.

MATLAB Implementation:

Step 1: MATLAB can generate only discrete time signals. For an approximate analog signal xt, choose the spacing between the samples to be very small (≈0), say 50µs = 0.00005. Next choose the time duration, **tfinal** for how long the signal xt exists.(for ex. tfinal= 0.02 seconds means xt is generated for 0.02 seconds if the input frequency is 50 Hz i.e. time period is 0.02 sec, then there will be one complete cycle in the plots) Generate the time vector vector **t** that represents the time base i.e. **t = 0:0.00005: tfinal;**
Given t, the analog signal xt of frequency fd is generated using **xt=sin(2*pi*fd*t)** pi is recognized as 3.14 by MATLAB.
Step 2: To illustrate oversampling condition, choose sampling frequency fso=2.2*fd. For this sampling rate T0=1/fso, generate the time vector as **n1 = 0:T0:0.05;** & over sampled discrete time signal x1 is given by **x1=sin(2*pi*fd*n1);**
[Alternately let n1 be a fixed number of samples, say n=0:10; & x1=sin(2*pi*n*fd/fs0);]
Plot the original and oversamples signals using **plot** command. Subplot command is used to show multiple graphs in single figure window.

Step 3: Repeat step 2 for different sampling frequencies, i.e., fs=1.3*fd & fs=2*fd for under sampling and Nyquist sampling conditions respectively.
Step4: Also plot the frequency domain plot of all signals using **fft** command.

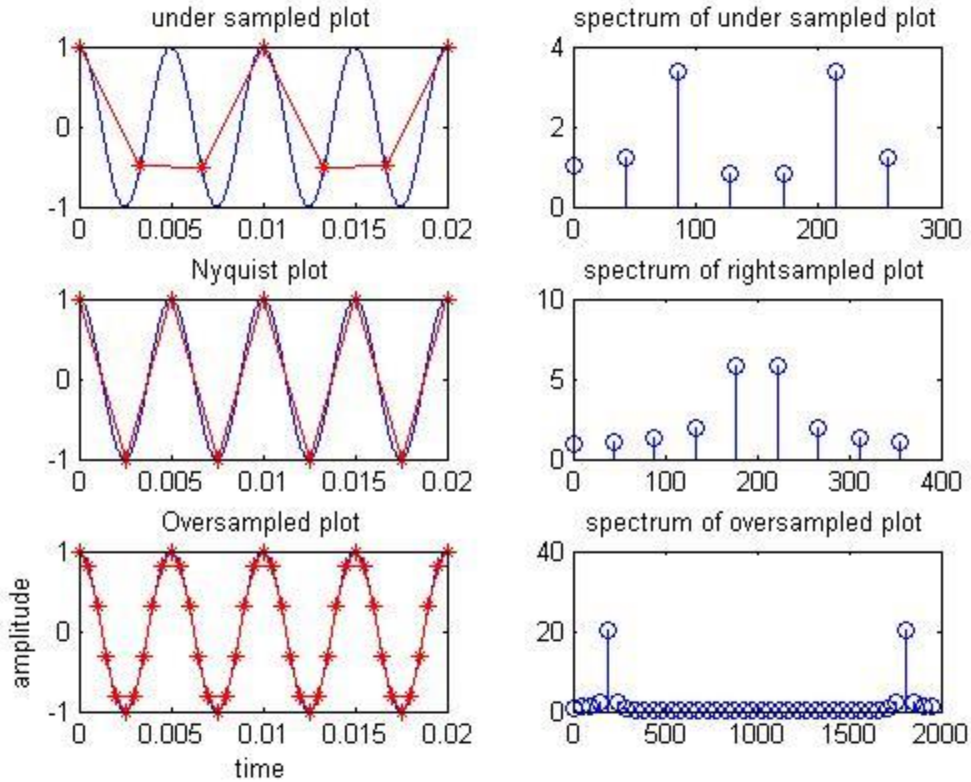
Reference MATLAB Code:

```
clc; clear all; close all;
%Generation of analog signal
tfinal=0.02;
t=0:0.00002: tfinal;
fd=input('Enter analog freuency ');
%define analog signal for comparison
xt= cos (2*pi*fd*t);
%Under sampling
%Condition for Nyquist sampling
fs1=1.5*fd; %simulate condition for under
%sampling i.e., fs1<2*fd
n1=0:1/fs1:tfinal; %define the time vector
%Generate the under sampled signal
xn1=cos(2*pi*n1*fd);
%plot the analog & under sampled signals
subplot(3,2,1); plot(t,xt,'b',n1,xn1,'r*-');
title('under sampled plot');
% plot the undersampled signal in frequency
%domain
Xk1=fft(xn1); %Conversion to frequency
domain
f1=(0:length(Xk1) -1)*fs1/length(Xk1); %
frequency index of spectrum plot
subplot(3,2,2);stem(f1,abs(Xk1));title('spect
rum of under sampled plot');
%Nyquist Sampling
%Condition for Nyquist sampling
fs2=2*fd;
n2=0:1/fs2:tfinal;
xn2=cos(2*pi*fd*n2);
%plot the analog & Nyquist sampled signals
subplot(3,2,3);
plot(t,xt,'b',n2,xn2,'r*-');title('Nyquist plot');
%plot the Nyquist sampled signal in frequency
$domain
Xk2=fft(xn2);
f2=(0:length(Xk2)-1)*fs2/length(Xk2);
subplot(3,2,4);stem(f2,abs(Xk2));
title('spectrum of rightsampled plot');
%Oversampling
%Condition for oversampling
fs3=10*fd;
n3=0:1/fs3:tfinal;
xn3=cos(2*pi*fd*n3);
%plot the analog & over sampled signals
subplot(3,2,5);plot(t,xt,'b',n3,xn3,'r*-');
title('Oversampled plot');
xlabel('time'); ylabel('amplitude');
%plot the over sampled signal in frequency
%domain
Xk3=fft(xn3);
f3=(0:length(Xk3) -1)*fs3/length(Xk3);
subplot(3,2,6);stem(f3,abs(Xk3));
title('spectrum of oversampled plot');
```

Note: For Sine waveform, start sampling n2 from (0.005/4). i.e. n2=0.00125:1/fs2:tfinal;
For Cos n2=0:1/fs2: tfinal;

Expected Output:

Enter analog frequency 200



Assignment :

1. Perform the sampling of the signal which is combination of signals of frequencies 30Hz,50Hz and observe the results
2. Use function program concept to implement under sampling, nyquist sampling and over sampling.

Result/Outcome of the Experiment:

Signature of the faculty with date: