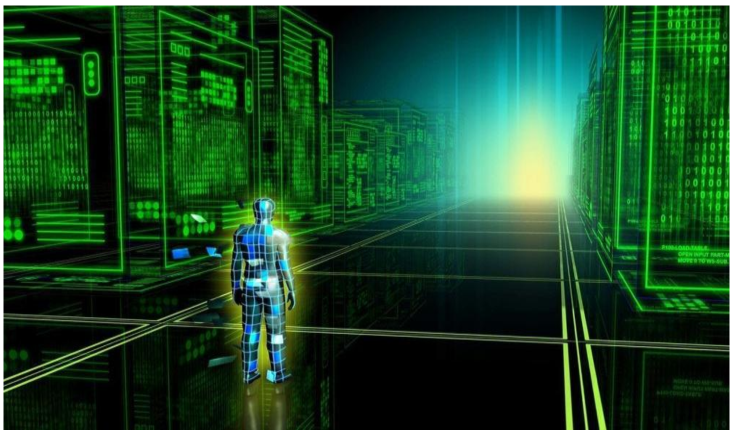
MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

Department of Electronics & Communication Engineering

MODELLING SIMULATION
LABORATORY MANUAL

MLRS-R25



II B.TECH(ECE) - I Semester

Academic Year: 2026-2027

Prepared by
Mr.A.Anil Kumar
Asst. Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that this manual is a bonafide record of practical work carried out in the Modelling Simulation lab for the **B.Tech (Electronics and Communication Engineering) III Semester** Programme during the academic year **2026–2027.**

This manual has been prepared by **Mr. A.Anil Kumar (Assistant Professor)** Department of Electronics and Communication Engineering, with my/our own efforts and to the best of our knowledge.

Signature of Lab Faculty **Signature of HOD**

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

PREFACE

This laboratory lays the foundation for the Electronics and Communication Engineering students during Fourth year of their course.

Modelling Simulation lab, students will know how to write Verilog programming of digital circuits and simulate using Xilinx software tool.

In cycle -II, students will design the circuits of digital system and implement them in Cadence tool. After performing all the experiments included in this Laboratory, it is hoped the student receives good training to handle any electronic equipment available in electronics field.

By,
Mr. A.Anil Kumar

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

ACKNOWLEDGEMENT

It was really a good experience, working at Modelling Simulation Lab. First, I would like to thank Dr. N. Srinivas, Professor, Department of Electronics and Communication Engineering, Marri Laxman Reddy Institute of technology & Management for giving the technical support in preparing the document.

I express my sincere thanks to Dr. N. Srinivas, Head of the Department of Electronics and Communication Engineering, Marri Laxman Reddy Institute of technology & Management, for his concern towards me and gave me opportunity to prepare Concrete Technology laboratory manual.

I am deeply indebted and gratefully acknowledge the constant support and valuable patronage of Dr.Ravi Prasad, Dean Academics, Marri Laxman Reddy Institute of technology & Management. I am unboundedly grateful to him for timely corrections and scholarly guidance.

I express my heartful thanks to Dr.P.Sridhar, Director, and Dr. R.Murali Prasad, Principal, Marri Laxman Reddy Institute of technology & Management, for giving me this wonderful opportunity for preparing the Concrete Technology laboratory manual.

At last, but not the least I would like to thank the entire ECE Department faculties those who had inspired and helped me to achieve my goal.

By,
Mr.A.Anil kumar
Department of Electronics and Communication Engineering

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

GENERAL INSTRUCTIONS

1. Students are instructed to come to MODELLING SIMULATION LABORATORY on time. Late comers are not entertained in the lab.

2. Students should be punctual to the lab. If not, conducted experiments will not be repeated.

3. Students are expected to come prepared at home with the experiments which are going to performed.

4. Students are instructed to display their identity cards and apron before entering into the lab.

5. Students are instructed not to bring mobile phones to the lab.

6. The equipment's and other accessories used in Modelling Simulationlab should be handled with care and responsibility.

7. Any damage to the equipment's during the lab session is student's responsibility and penalty or fine will be collected from the student.

8. Students should update the records and lab observation books session wise. Before leaving the lab, the student should get his lab observation book signed by the faculty.

9. Students should submit the lab records 2/3 days in advance to the concerned faculty members in the staffroom for their correction and return.

10. Students should not move around the lab during the lab session.

11. If any emergency arises, the student should take the permission from faculty member concerned in written format.

12. The faculty members may suspend any student from the lab session on disciplinary grounds.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

SAFETY MEASURES

1. While working in the laboratory suitable precautions should be observed to prevent accidents.
2. Always follow the experimental instructions strictly.
3. Use the first aid box in case of any accident/mishap.
4. Never work in the laboratory unless a demonstrator or teaching assistant in present.
5. When the experiment is completed, students should disconnect the setup made by them, and should return all the components/instruments taken for the purpose.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

VISION & MISSION OF THE INSTITUTE

Vision of the Institute

To be a globally recognized institution that fosters innovation, excellence, and leadership in education, research, and technology development, empowering students to create sustainable solutions for the advancement of society.

Mission of the Institute

To foster a transformative learning environment that empowers students to excel in engineering, innovation, and leadership.

To produce skilled, ethical, and socially responsible engineers who contribute to sustainable technological advancements and address global challenges.

To develop future leaders through innovative research, strong industry collaboration, and meaningful community engagement..

Quality Policy

The management is committed in assuring quality service to all its stakeholders, students, parents, alumni, employees, employers, and the community.

Our commitment and dedication are built into our policy of continual quality improvement by establishing and implementing mechanisms and modalities ensuring accountability at all levels, transparency in procedures, and access to information and actions.

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

Vision of the Department

To provide quality technical education in Electronics and Communication Engineering through research, innovation, striving for global recognition in specified domain, leadership, and sustainable societal solutions.

Mission of the Department

- To create a transformative learning environment that empowers students in electronics and communication engineering, fostering excellence in technical skills and leadership.
- To drive innovation through research, deliver a transformative education grounded in ethical principles, and nurture the development of professionals
- To cultivate strong industry partnerships, and engaging actively with the community for societal and technological progress.

Program educational Objectives (PEOs)

PEO 1: Have Successful career in Industry

Graduates will excel in the Electronics and Communication industry with a strong foundation in technical expertise, continuous learning, and innovation.

PEO 2: Show Excellence in higher studies/Research

Graduates will excel in higher studies and research in Electronics and Communication Engineering (ECE) through a combination of rigorous academic dedication, cutting-edge innovation, and a deep understanding of emerging technologies.

PEO 3: Show Good Competency towards Entrepreneurship

Graduates will have to show good competency towards entrepreneurship in the field of Electronics and Communication Engineering, one must demonstrate an in-depth understanding of emerging technologies, market trends, and the ability to innovate within this rapidly evolving industry.

Program Outcomes (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes (PSOs)

1. Analyze and design analog & digital circuits or systems for a given specification and function.
2. Implement functional blocks of hardware-software co-designs for signal processing and communication applications.

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF ELECTRONICS AND COMMUNICATION
MODELLING SIMULATION LABORATORY

Virtual lab details

Name of the Virtual Lab:
Virtual Lab Host Institute:
URL/Link to Lab
Academic Year
Semester

List of Experiments Available in Virtual Lab

1. Write the code / script for generating various standard viz: Periodic and Aperiodic, Unit Impulse, Unit Step, Square, Saw tooth, Triangular, Sinusoidal, Ramp, Sinc and Nonstandard Signals and Sequences generated from these standard signals /sequences using Waveform synthesis. Also perform different operations viz: Addition, Multiplication, Scaling, Shifting, Folding, Computation of Energy and Average Power on them.
2. . Write the code / script for finding the Even and Odd parts of a signal/sequence Real and Imaginary parts of a signal
3. Write the code / script for finding the output of a System for a given input and pulse Response and finding Auto Correlation and Cross Correlation of Signals / sequences.
4. Write the code / script for Verifying whether a given Continuous/Discrete System is Linear, Time Invariant, Stable and Physically Realizable
5. . Write the code / script for obtaining Sinusoidal response and Impulse response of a given Continuous / Discrete LTI System.
   a) Plot the Real and Imaginary part and
   b) Magnitude and Phase Plot of the response
6. Write the code / script for finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Fourier Transform by using the properties where ever required
7. Write the code / script for finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Laplace Transform by using the properties where ever required. Also plot pole-zero diagram in S-plane
8. Write the code/ script for finding and plotting the Magnitude and Phase Spectrum of any given Sequence by finding its Z-Transform by using the properties wherever required. Also plot pole —zero diagram in Z-plane
9. Design a Simulink or equivalent model for
10. Gibbs Phenomenon and waveform synthesis

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF Electronics and Communication ENGINEERING

MODELLING SIMULATION LABORATORY

LAB PLANNER

S.No	Experiment	CO	Virtual Lab Availabilty	Date planned	Date conducted
1	Write the code / script for generating various standard viz: Periodic and Aperiodic, Unit Impulse, Unit Step, Square, Saw tooth, Triangular, Sinusoidal, Ramp, Sinc and Nonstandard Signals and Sequences generated from these standard signals /sequences using Waveform synthesis. Also perform different operations viz: Addition, Multiplication, Scaling, Shifting, Folding, Computation of Energy and Average Power on them..	CO1		03.12	04)12
2	Write the code / script for finding the Even and Odd parts of a signal/sequence Real and Imaginary parts of a signal.	CO1		10.12	
3	Write the code / script for finding the output of a System for a given input and pulse Response and finding Auto Correlation and Cross Correlation of Signals / sequences.	CO2		17.12	
4	Write the code / script for Verifying whether a given Continuous/Discrete System is Linear, Time Invariant, Stable and Physically Realizable	CO2		24.12	
5	Write the code / script for obtaining Sinusoidal response and Impulse response of a given	CO2		31.12	
6	Write the code / script for finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Fourier Transform by using the properties where ever required	CO3		07.01	
7	LAB INTERNAL-1			21.01	

8	Write the code / script for finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Laplace Transform by using the properties where ever required. Also plot pole-zero diagram in S-plane	CO3		28.01	
9	Write the code/ script for finding and plotting the Magnitude and Phase Spectrum of any given Sequence by finding its Z-Transform by using the properties wherever required. Also plot pole —zero diagram in Z-plane	CO3		11.02	
10	Design a Simulink or equivalent model for	CO4		18.02	
11	Gibbs Phenomenon and waveform synthesis	CO4		25.02	
12		CO5		11.03	
13	Writhe the code for generating various Random variables with different CDFs/PDFs.	CO5		18.03	
14	MID-II			25.03	

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

MODELLING SIMULATION LABORATORY

LAB PLANNER

Date planed	**10/7**			**17/7**			**24/7**																									
Date conducted																																
Roll Number	Exp No	CO	VL*	Exp No	CO	VL*	Exp No	CO	VL*	Exp No	CO	VL*	Exp No	CO	VL*		Exp No	CO	VL*	Exp No	CO	VL*	Exp No	CO	VL*	Exp No	CO	VL*	Exp No	CO	VL*	
237Y1A 0401	1	1	Y	2	1	N	3	2	Y	4	2	N	5	3	Y		6	4	Y	7	4	Y	8	4	Y	9	5	N	10	5	Y	
237Y1A 0402	2	1	N	3	2	Y	4	2	N	5	3	Y	6	4	Y	M	7	4	Y	8	4	Y	9	5	N	10	5	Y	1	1	Y	**M**
237Y1A 0403	3	2	Y	4	2	N	5	3	Y	6	4	Y	7	4	Y	I	8	4	Y	9	5	N	10	5	Y	1	1	Y	2	1	N	**I**
237Y1A 0104	4	2	N	5	3	Y	6	4	Y	7	4	Y	8	4	Y	D	9	5	N	10	5	Y	1	1	Y	2	1	N	3	2	Y	D
237Y1A 0405	5	3	Y	6	4	Y	7	4	Y	8	4	Y	9	5	N	-	10	5	Y	1	1	Y	2	1	N	3	2	Y	4	2	N	**-**
237Y1A 0406	6	4	Y	7	4	Y	8	4	Y	9	5	N	10	5	Y	I	1	1	Y	2	1	N	3	2	Y	4	2	N	5	3	Y	**I I**

Note:VL*-Virtual Lab Availabilty

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
MODELLING SIMULATION LABORATORY

RUBRICS USED TO ASSESS LEARNINGS IN LABORATORIES

1. RUBRICS FOR DAY TO DAY EVALUATION

Parameter	Max Marks	Level-1 (Very Poor)	Level-2 (Poor)	Level-3 (Average)	Level-4 (Good)	Level-5 (Excellent)
Observation Book	05	No observations or irrelevant data. (0–1)	Incomplete or incorrect data. (2)	Basic values with some errors. (3)	Mostly correct with good format. (4)	Fully correct, clear, and well-formatted. (5)
Record Writing	05	Not submitted. (0-1)	Submitted but mostly incomplete. (2)	Submitted with some missing/wrong parts. (3)	Submitted with minor issues. (4)	Fully complete, correct algorithm & flowchart. (5)
Result	05	No result or major errors. (0–1)	Result partially obtained. (2)	Acceptable result with limited error. (3)	Near-correct result and reasonable error. (4)	Accurate result. (5)
Viva-Voce	05	Did not answer any questions. (1)	Answered very few questions. (2)	Answered some questions with help. (3)	Answered most questions correctly. (4)	Answered all questions accurately. (5)

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
MODELLING SIMULATION LABORATORY

2.RUBRICS FOR INTERNAL EVALUATION

Criterion	**Max Marks**	**Level-1** ***(Very Poor)***	**Level-2** ***(Poor)***	**Level-3** ***(Average)***	**Level-4** ***(Good)***	**Level-5** ***(Excellent)***
Design/Tool/Apparatus Selection	2 Marks	Incorrect tool/design and no reasoning. **(0)**	Tool/design selection attempted with unclear logic. **(0.5)**	Satisfactory selection with partial justification. **(1)**	Correct selection and proper analysis with few errors. **(1.5)**	Smart selection with accurate, relevant analysis. **(2)**
Execution (Code/Debug/Run) /Analysis/Method Used	4 Marks	Did not attempt or completely failed to execute. **(0)**	Attempted but unable to proceed or with major errors. **(1)**	Partial execution with some logic/syntax errors. **(2)**	Mostly correct execution with minimal help. **(3)**	Fully correct and independently executed program. **(4)**
Results& Documentation	2 Marks	Incomplete or poorly presented. **(0)**	Basic structure but lacks clarity or formatting. **(0.5)**	Complete but generic or with formatting issues. **(1)**	Well-structured and mostly clear. **(1.5)**	Well-organized, professional, and engaging documentation. **(2)**
Viva-Voce (Understanding of Concepts)	2 Marks	No understanding; could not answer questions. **(0)**	Answered a few with difficulty. **(0.5)**	Answered half the questions with basic clarity. **(1)**	Good understanding with confident answers. **(1.5)**	Answered all questions with clarity and depth. **(2)**

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
MODELLING SIMULATION LABORATORY

3. RUBRICS FOR SEMESTER END EXAMINATIONS

Criterion	**Max Marks**	**Level-1 (Very Poor *(0–2 marks)***	**Level-2 (Poor) *(3–4 marks)***	**Level-3 (Average) *(5–6 marks)***	**Level-4 (Good) *(7–9 marks)***	**Level-5 (Excellent) *(10–12 marks)***
Preparedness for the Experiment	12 marks	No clarity on objective or procedure. Unable to explain basics.	Limited idea of the objective/procedure. Needed prompting.	Has basic understanding; minor gaps in concept or preparation.	Well-prepared, with clear understanding of steps and background.	Fully prepared with strong conceptual clarity and confident explanation.
Performance in the Laboratory	12 marks	Unable to perform experiment. Relied entirely on examiner's help.	Performed with multiple errors and constant support.	Performed with some errors; required occasional help.	Performed mostly independently with minimal support.	Performed independently, efficiently, and with precision.
Calculations & Graphs	12 marks	No or incorrect calculations. Graphs missing or irrelevant.	Multiple calculation errors. Graphs/plots inaccurate or poorly labeled.	Calculations partially correct. Graphs present but with some flaws.	Correct calculations and graphs with minor errors.	Accurate calculations and well-labeled graphs with proper interpretation.
Results & Error Analysis	12 marks	No result or invalid result. No error analysis attempted.	Incorrect result with vague or no error discussion.	Acceptable result. Error analysis attempted but limited.	Correct result with sound error discussion.	Accurate result with detailed and relevant error analysis.
Viva-Voce (Subject Knowledge)	12 marks	Unable to answer any questions. No conceptual understanding.	Answered few questions with poor logic.	Answered half of the questions with average understanding.	Answered most questions with clarity and confidence.	Answered all questions with depth, clarity, and reasoning.

EXPERMENT NO: 10

Gibbs Phenomenon and waveform synthesis

AIM: -

To demonstrate Gibbs Phenomenon using MATLAB.

SOFTWARE REQURIED:-

1.MATLAB R2025a.
2.Windows XP2SP2.

THEORY:-

Gibbs Phenomenon:

The peculiar manner in which the Fourier series of a piecewise continuously differentiable periodic function behaves at a jump discontinuity: the nth partial sum of the Fourier series has large oscillations near the jump, which might increase the maximum of the partial sum above that of the function itself. The overshoot does not die out as the frequency increases, but approaches a finite limit.

The Gibbs phenomenon involves both the fact that Fourier sums overshoot at a jump discontinuity, and that this overshoot does not die out as the frequency increases.

The best known version of the Gibbs phenomenon is the overshoot that arises when a discontinuous function is represented by a truncated set of Fourier expansion terms. The situation is similar if the truncated Fourier expansion is instead obtained by means of interpolation on an equispaced grid.

PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
%Gibbs phenomenon%
clc;
clear all;
close all;
t=0:0.01:1;
N=input('enter no of harmonics');
x=square(2*pi*t);
plot(t,x);
hold on;
gp=0;
for n=1:2:N;
   gp=gp+(4/(n*pi))*sin(2*pi*n*t);
end;
plot(t,gp);
hold off;
xlabel('time');
ylabel('amplitude');
title('gibbs phenomenon');
```

Validation of the Results :

VIVA QUESTIONS

S.No	Question	CO	Blooms Taxonomy
1	Define Signal	CO1	Understand
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			

Note: Each experiment should contain Minimum 20 Viva Questions

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

Department of Electronics & Communication Engineering

COURSE STRUCTURE, OBJECTIVES

COURSE STRUCTURE

Laboratory subjects – Internal and external evaluation– Details of marks

MODELLING SIMULATION LABORATORY will have a continuous evaluation during 6th semester for 40 sessional marks and 60 end semester examination marks.

Out of the 40 marks for internal evaluation, day-to-day work in the laboratory shall be evaluated for 20 marks and internal practical examination shall be evaluated for 20 marks conducted by the laboratory teacher concerned.

The end examination will be evaluated for a maximum of 60 marks. The end semester examination shall be conducted with an external examiner and internal examiner. The external examiner shall be appointed by the principal / Chief Controller of examinations

COURSE OBJECTIVES

- Generation and manipulation of standard and nonstandard continuous/discrete-time signals
- Analytical skills for understanding system properties, convolution and transform techniques
- The Fourier, Laplace, and Z-Transform methods for spectral analysis, waveform synthesis, and pole-zero analysis in time and frequency domains
- Concepts of stochastic processes and probability theory, including random variable generation
- Modelling and simulation of control systems using Simulink, including solving differential equations, analysing RLC circuits, and implementing PID controllers

Department of Electronics & Communication Engineering

Course Outcomes:

Modelling Simulation Processing Lab	• Generate various standard and nonstandard signals, analyze their properties, and perform signal operations and synthesis in both time and frequency domains • Capable of evaluating system responses using convolution, testing system characteristics like linearity and time-invariance, and analyzing LTI systems through impulse and sinusoidal inputs • Perform spectral analysis using Fourier, Laplace, and Z-transforms, and interpret magnitude-phase spectra and pole-zero plots in s- and z-domains • Demonstrate the ability to simulate random processes, analyze statistical properties of Gaussian noise, and apply correlation techniques for noise removal and signal extraction • Proficient in modeling and simulating dynamic systems in Simulink, including RLC circuit behavior and PID control, as well as assessing controllability and observability of systems

Course Outcomes (CO's)–Program Outcomes(PO's)Mapping

CO's \ PO's	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	3	3	1	3	3	-	-	-	-	1	-	-	2	2
CO2	3	3	1	3	3	-	-	-	-	1	-	-	2	-
CO3	3	3	1	3	3	-	-	-	-	1	-	-	2	-
CO4	3	3	1	3	3	-	-	-	-	1	-	-	2	2
CO5	3	3	1	3	3	-	-	-	-	1	-	-	2	-

Simple-1 **Moderate-2** **High-3**

History and fundamental background of the lab:

During the past half-century simulation has advanced as a tool of choice for operational systems analysis. The advances in technology have stimulated new products and new environments without software standards or methodological commonality. Each new simulation language or product offers its own unique set of features and capabilities. Yet these simulation products are the evolution of research, development, and application. In this paper we interpret the historical development of simulation modeling. In our view simulation modeling is that part of the simulation problem-solving process that focuses on the development of the model. It is the interpretation of a real production (or service) problem in terms of a simulation language capable of performing a simulation of that real-world process. While “interpretation” is in the “eyes of the beholder” (namely us) there are some historical viewpoints and methods that influence the design of the simulation model.

Introduction

Models vs. Modeling

Formal mathematical and statistical models of operations have been developed over several decades of research. However, when a model is applied beyond its assumptions, it leaves open the question of its applicability and the modeler is faced with adapting the model or developing a new model. The adaption or creation of a new analytic model is a significant task and that challenges the analytical skills of the developer, and may require considerably more time and effort, perhaps without tangible results.

The value of a model is its use. Since individual models have limitations and their extension can be daunting, some researchers and practitioners prefer a more flexible modeling environment. Simulation has long been viewed as modeling tool that has a very broad development environment and does not require mathematical or statistical sophistication for model development and use. Furthermore, to ease the burden of the development of simulation models, a variety of simulation languages have been developed. Each simulation language offers its own modeling constructs within which a simulation model can be constructed, simulated, and analyzed. While the languages have changed over time, the modeling approach is inextricably linked to its simulation execution perspective.

Modeling

The purpose of this paper is to provide a personalized (and therefore limited) history of simulation modelling as it has been largely popularized through the Winter Simulation Conference. Using simulation in a problem-solving context requires several proficiencies:

- Defining the problem correctly and setting the objects.
- Using modeling concepts for abstracting the essential features of the system into a model.
- Collecting and compiling data and input for the model.
- Converting the model into computer readable code that is capable of simulating the system.
- Instructing the computer to perform the simulation correctly and efficiently for various scenarios.
- Summarizing and analyzing the simulation output into performance measures.

Kiviat (1967) was among the first to describe the modeling process and the proficiencies needed. A user with in-depth knowledge of almost any general purpose programming language can provide most of these proficiencies. However, the burden of achieving a final simulation is formidable as the programmer needs to provide software to accomplish all the needed components including random number generation, random variate generation, status updating and time advance, statistical collection and display, etc. To accomplish all this for a single model is generally prohibitive and there are simulation languages and simulation libraries to ease the burden of creating simulations.

While many simulation languages have come and gone, the fundamental simulation approaches are similar. A simulation language executes a model of the system to dynamically act out the behavior of the real system over time. This is done by changing the value of state variables over simulated time. Simulation languages may be generally categorized into two broad families: discrete and continuous. Discrete tools model systems where the state of the system changes in discrete units at specific event times. Continuous tools model systems where the state of the system changes continuously over portions of time. The focus here is on discrete systems, although continuous systems is considered. Nowadays, most simulation languages are multi-modal and support several modelling paradigms which usually mix discrete and continuous capabilities.

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

Department of Electronics & Communication Engineering

MODELLING SIMULATION LABORATORY

List of Experiments:

1. Write the code / script for generating various standard viz: Periodic and Aperiodic, Unit Impulse, Unit Step, Square, Saw tooth, Triangular, Sinusoidal, Ramp, Sinc and Nonstandard Signals and Sequences generated from these standard signals /sequences using Waveform synthesis. Also perform different operations viz: Addition, Multiplication, Scaling, Shifting, Folding, Computation of Energy and Average Power on them.
2. . Write the code / script for finding the Even and Odd parts of a signal/sequence Real and Imaginary parts of a signal
3. Write the code / script for finding the output of a System for a given input and pulse Response and finding Auto Correlation and Cross Correlation of Signals / sequences.
4. Write the code / script for Verifying whether a given Continuous/Discrete System is Linear, Time Invariant, Stable and Physically Realizable
5. . Write the code / script for obtaining Sinusoidal response and Impulse response of a given

Continuous / Discrete LTI System.
a) Plot the Real and Imaginary part and
b) Magnitude and Phase Plot of the response

6. Write the code / script for finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Fourier Transform by using the properties where ever required
7. Write the code / script for finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Laplace Transform by using the properties where ever required. Also plot pole-zero diagram in S-plane
8. Write the code/ script for finding and plotting the Magnitude and Phase Spectrum of any given Sequence by finding its Z-Transform by using the properties wherever required. Also plot pole —zero diagram in Z-plane
9. Design a Simulink or equivalent model for
10. Gibbs Phenomenon and waveform synthesis
11. Writhe the code for generating various Random variables with different CDFs/PDFs.
12. Write the code for generating Gaussian noise and for finding the mean ,skewness,Kurtosis,PDF and PSD
13. Write the code/script for verifying sampling theorem for different sampling rates sampling types and duty cycles and for plotting the sampled and reconstructed signals
14. Write the script/code for Removal of noise from the signal using cross relation
15. Write the code/script for Extraction of periodic signal masked by noise using auto correlation
16. 16.Build and simulate a DC Motor using Simulink
17. Implementation of a PID Controller from equations using Simulink
18. Controllability and Observability.

10.

NOTE: Minimum of 12 experiments to be conducted

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

Department of Electronics & Communication Engineering

INSTRUCTIONS TO THE STUDENT

1. Students are required to attend all labs.
2. Students will work individually in hardware laboratories and in computer laboratories.
3. While coming to the lab bring the lab manual cum observation book, record etc.
4. Take only the lab manual, calculator (if needed) and a pen or pencil to the work area.
5. Before coming to the lab, prepare the pre-lab questions. Read through the lab experiment to familiarize yourself with the components and assembly sequence.
6. Utilize 3 hours time properly to perform the experiment (both in software and hardware) and note down the readings properly. Do the calculations, draw the graph and take signature from the instructor.
7. If the experiment is not completed in the prescribed time, the pending work has to be done in the leisure hour or extended hours.
8. You have to submit the completed record book according to the deadlines set up by your instructor.
9. For practical subjects there shall be a continuous evaluation during the semester for 25 sessional marks and 50 end examination marks.
10. Of the 25 marks for internal, 15 marks shall be awarded for day-to-day work and 10 marks to be awarded by conducting an internal laboratory test.

INDEX

13	.Write the code/script for verifying sampling theorem for different sampling rates sampling types and duty cycles and for plotting the sampled and reconstructed signals	
14	Write the script/code for Removal of noise from the signal using cross relation.	
15	Write the code/script for Extraction of periodic signal masked by noise using auto correlation	
16	Build and simulate a DC Motor using Simulink	
17	Implementation of a PID Controller from equations using Simulink	
18.	Controllability and Observability.	
19	Writhe the code for generating various Random variables with different CDFs/PDFs.	

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

2530472: MODELLING AND SIMULATION LABORATORY

II Year B.Tech. ECE I – Sem.

L	T	P	C
0	0	2	1

Couse Overview:

This comprehensive set of experiments offers hands-on exposure to signal processing, systems analysis, probability theory, stochastic processes, and control systems. Students will generate and manipulate standard and nonstandard signals, analyze system behaviors through transforms and correlation techniques, and explore real-world applications such as noise reduction, waveform synthesis, and PID control. Additionally, practical simulations using Simulink and Python equip students with critical skills in modeling, analysis, and system design.

Course Objectives:

The students will try to learn

Generation and manipulation of standard and nonstandard continuous/discrete-time signals

analysis,

- of stochastic processes and theory, including random variable
- Analytical skills for understanding system properties, convolution and transform techniques.
- The Fourier, Laplace, and Z-Transform methods for spectral and of waveform synthesis, and pole-zero analysis in time and frequency domains.
- Concepts probability generation.
- Modeling simulation control systems using Simulink, including differential equations, analyzing RLC circuits, and implementing PID controllers.

- Analytical skills for understanding system properties, convolution and transform techniques.
- The Fourier, Laplace, and Z-Transform methods for spectral waveform synthesis, and pole-zero analysis in time and frequency domains.
- Concepts probability generation.
- Modeling simulation control systems using Simulink, including differential equations, analyzing RLC circuits, and implementing PID controllers.

Write the code / script for generating various standard viz: Periodic and Aperiodic, Unit Impulse, Unit Step, Square, Saw tooth, Triangular, Sinusoidal, Ramp, Sinc and Nonstandard Signals and Sequences generated from these standard signals /sequences using Waveform synthesis. Also for perform different operations viz: Addition, Multiplication, Scaling, Shifting, Folding, Computation of Energy and Average Power on them.

Generate various standard and nonstandard signals, analyze their properties, and perform signal operations and synthesis in both time and frequency domains

Capable of evaluating system responses using convolution, testing system characteristics like linearity and time-invariance, and analyzing LTI systems through impulse and sinusoidal inputs.

2. Perform spectral analysis using Fourier, Laplace, and Z-transforms, and interpret magnitude-phase spectra and pole-zero plots in s- and z-domains.

3. the ability to simulate random processes, analyze statistical properties of Gaussian noise, and apply correlation techniques for noise removal and signal extraction.

4. Proficient in modeling and simulating dynamic systems in Simulink, including RLC circuit behavior and PID control, as well as assessing controllability and observability of systems.

5.

Note: • All the experiments are to be simulated using MATLAB or equivalent software • Minimum of 12 experiments are to be completed / simulated.

List of Experiments:

6.

1.
7.

8.

9.

10.

Probability Theory and Stochastic Processes (Minimum 3 Experiments)

11. Write the code / script for generating various Random Variables with different CDFs/ PDFs
12. Write the code / script for generating Gaussian noise and for finding its mean, Skewness, Kurtosis, PDF and PSD.
13. Write the code / script for Verifying Sampling theorem for different sampling rates, Sampling types and Duty Cycles and for plotting the sampled and reconstructed Signals.
14. Write the code / script for Removal of noise from the signal using Cross correlation.
15. Write the code / script for Extraction of Periodic Signal masked by noise using Auto
16. Correlation

Control Systems (Minimum 2 Experiments)

17. Build and Simulate a DC Motor using Simulink
18. Implementation of a PID Controller from equations using Simulink
19. Controllability and Observability

Note: For the experiments with code/scripts written in MATLAB or equivalent (1-8, 11-15), the student can design a user interface or app using MATLAB App Designer or equivalent.

Application on Real Time signals

1. Application of Autocorrelation: GPS Synchronization Satellite communication toolbox is required for this experiment.
   Generate the GPS signal.
   Visualize the GPS signal. Plot of autocorrelation of C/A code and visualize the spectrum of GPS signals. For exact steps, go through the following page:
https://www.mathworks.com/help/satcom/ug/gps-waveform-generation.html

2. Sampling of Speech Signals

6

Record and play speech in MATLAB. For steps, go through the following page:
https://in.mathworks.com/help/matlab/import_export/record-and-play-audio.html
Change the sampling rate of the recorded speech signal and play back to see the effect of aliasing. For steps, go through the following page:
https://in.mathworks.com/help/signal/ug/changing-signal-sample-rate.html

EXPERMENT NO: 1

GENERATION OF VARIOUS SIGNALS & SEQUENCES

AIM: - Write the code / script for generating various standard viz: Periodic and Aperiodic, Unit Impulse, Unit Step, Square, Saw tooth, Triangular, Sinusoidal, Ramp, Sinc and Nonstandard Signals and Sequences generated from these standard signals /sequences using Waveform synthesis. Also perform different operations viz: Addition, Multiplication, Scaling, Shifting, Folding, Computation of Energy and Average Power on them.

SOFTWARE REQURIED: -

1. MATLAB R2025a.
2. Windows XP SP2.

THEORY: -

1. Periodic and Aperiodic Signals

- **Periodic signal**: Repeats after a fixed interval (T)
  [
  x(t) = x(t+T)
  ]
  Example: Sinusoidal wave.
- **Aperiodic signal**: Does **not repeat** over time.
  Example: Unit step, ramp.

2. Standard Signals

- **Unit Impulse (\delta(t))**

- Infinitely narrow, area = 1
- Sifting property:
  [
  \int x(t)\delta(t-t_0)dt = x(t_0)
  ]

u(t))

[
u(t) =
\begin{cases}
1, & t \ge 0 \
0, & t < 0
\end{cases}
]

al (r(t))

[
r(t) = t \cdot u(t)
]

Signal

[
x(t) = A\sin(\omega t + \phi)
]

- Periodic, fundamental in communication systems.

ave

- Alternates between two levels (±A)
- Periodic, used in digital systems.

Wave

- Linear rise and fall.
- Continuous but non-sinusoidal.

Wave

- Linearly increases then drops sharp ly.

Function

[
\text{sinc}(t) = \frac{\sin(\pi t)}{\pi t}
]

- Important in signal reconstruction.

◇ 3. Nonstandard Signals (Waveform Synthesis)

- Created by combining standard signals.
- Example:
  - Pulse = difference of two step signals
    [
    x(t) = u(t) - u(t-T)
    ]

◇ 4. Operations on Signals

Addition
y(t) = x_1(t) + x_2(t)
]

[
y(t) = x_1(t)\cdot x_2(t)
]

- **Amplitude scaling**: (y(t) = A x(t))
- **Time scaling**:
  [
  y(t) = x(at)
  ]
  - (a>1): compression
  - (a<1): expansion

- **Time shift**:
  [
  y(t) = x(t - t_0)
  ]
  - Right shift: delay
  - Left shift: advance

(Time Reversal)

[
y(t) = x(-t)
]

5. Energy and Power Energy Signal

[
E = \int_{-\infty}^{\infty} |x(t)|^2 dt
]

- Finite energy, zero average power.

Power Signal

[
P = \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} |x(t)|^2 dt
]

- Finite power, infinite energy (e.g., periodic signals).

- **Periodic → finite power**, **Aperiodic → finite energy**
- Standard signals act as **building blocks**
- Operations like shifting, scaling, folding help **analyze and modify signals**
- Waveform synthesis creates complex signals from simple ones

PROGRAM: -

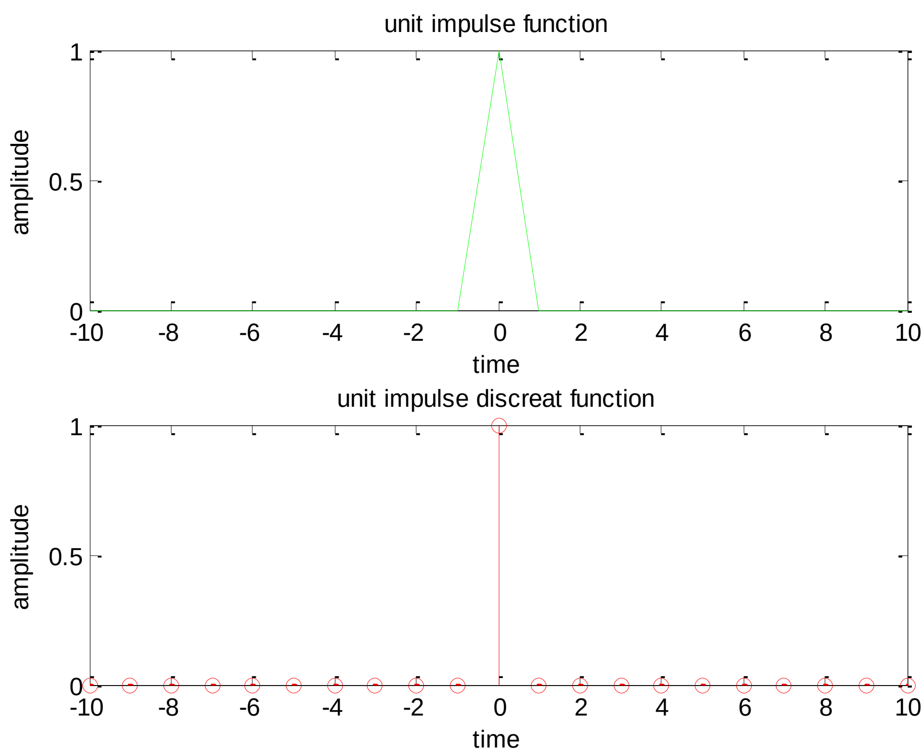
```
% MATLAB Script: Signal Generation and Operations
% Author: ChatGPT
% This script generates various standard signals and performs operations

clc; clear; close all;

%% Time Vector
fs = 1000;            % Sampling frequency
t = -1:1/fs:1;        % Time axis
n = -20:20;           % Discrete index

%% 1. Unit Impulse (Discrete)
delta = (n==0);

%% 2. Unit Step
u = double(t >= 0);

%% 3. Ramp
ramp = t .* (t >= 0);

%% 4. Sinusoidal
f = 5;                % Frequency
sin_wave = sin(2*pi*f*t);

%% 5. Square Wave
square_wave = square(2*pi*f*t);

%% 6. Sawtooth Wave
saw_wave = sawtooth(2*pi*f*t);

%% 7. Triangular Wave
tri_wave = sawtooth(2*pi*f*t, 0.5);

%% 8. Sinc Function
sinc_wave = sinc(t);

%% 9. Periodic Signal
periodic_signal = sin(2*pi*2*t);

%% 10. Aperiodic Signal
aperiodic_signal = exp(-abs(t));

%% Plot Signals
figure('Name','Standard Signals');
subplot(5,2,1); stem(n,delta); title('Unit Impulse');
subplot(5,2,2); plot(t,u); title('Unit Step');
subplot(5,2,3); plot(t,ramp); title('Ramp');
subplot(5,2,4); plot(t,sin_wave); title('Sinusoidal');
subplot(5,2,5); plot(t,square_wave); title('Square Wave');
subplot(5,2,6); plot(t,saw_wave); title('Sawtooth');
subplot(5,2,7); plot(t,tri_wave); title('Triangular');
subplot(5,2,8); plot(t,sinc_wave); title('Sinc');
subplot(5,2,9); plot(t,periodic_signal); title('Periodic');
```

```
subplot(5,2,10); plot(t,aperiodic_signal); title('Aperiodic');

%% Waveform Synthesis (Example)
synth_signal = sin_wave + 0.5*square_wave + 0.3*saw_wave;

figure;
plot(t, synth_signal);
title('Synthesized Signal');

%% Signal Operations

% Addition
add_signal = sin_wave + square_wave;

% Multiplication
mult_signal = sin_wave .* square_wave;

% Scaling
scaled_signal = 2 * sin_wave;

% Time Shifting
shifted_signal = sin(2*pi*f*(t - 0.2));

% Folding (Time Reversal)
folded_signal = fliplr(sin_wave);

%% Plot Operations
figure('Name','Signal Operations');
subplot(3,2,1); plot(t,add_signal); title('Addition');
subplot(3,2,2); plot(t,mult_signal); title('Multiplication');
subplot(3,2,3); plot(t,scaled_signal); title('Scaling');
subplot(3,2,4); plot(t,shifted_signal); title('Time Shifting');
subplot(3,2,5); plot(t,folded_signal); title('Folding');

%% Energy and Power Computation

% Energy
energy = sum(abs(sin_wave).^2) * (1/fs);

% Average Power
power = mean(abs(sin_wave).^2);

fprintf('Energy of sinusoidal signal = %.4f\n', energy);
fprintf('Average Power of sinusoidal signal = %.4f\n', power);

%% Non-standard Signal (Combination)
nonstandard = (t>=0).*sin(2*pi*3*t).*exp(-2*t);

figure;
plot(t, nonstandard);
title('Non-Standard Signal');

%% Discrete Signal Operations
```

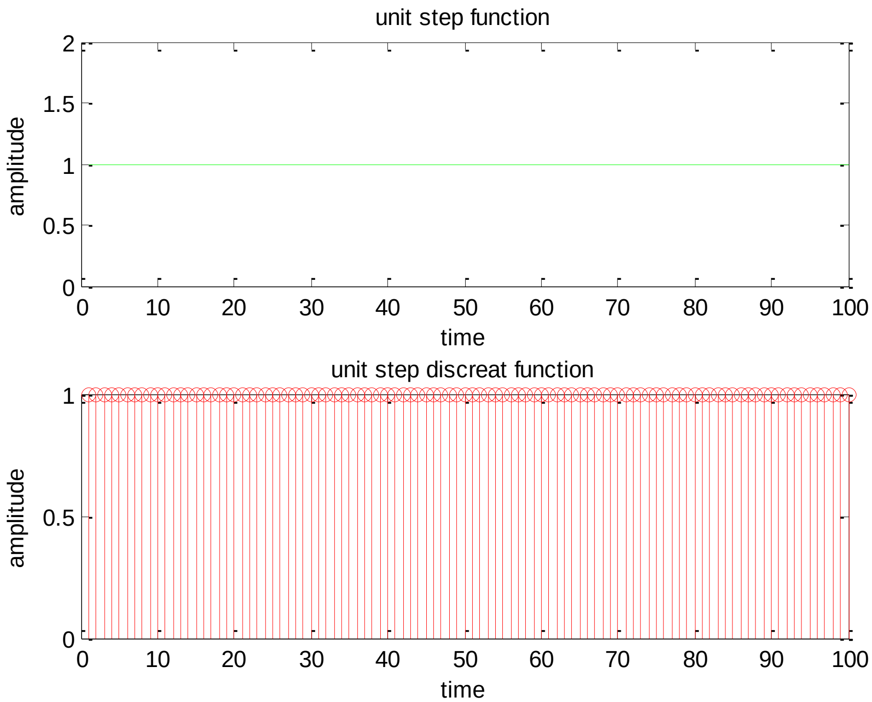
```
x = sin(0.2*pi*n);
y = cos(0.2*pi*n);

add_d = x + y;
mult_d = x .* y;
shift_d = sin(0.2*pi*(n-2));
fold_d = fliplr(x);

figure('Name','Discrete Signals');
subplot(2,2,1); stem(n,add_d); title('Addition');
subplot(2,2,2); stem(
```

OUTPUT:-

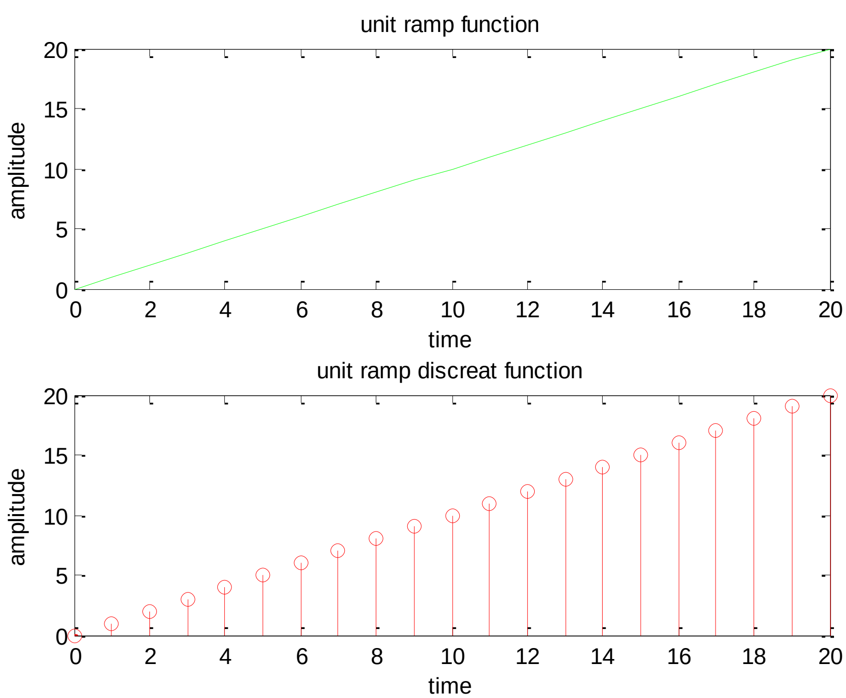
unit impulse function



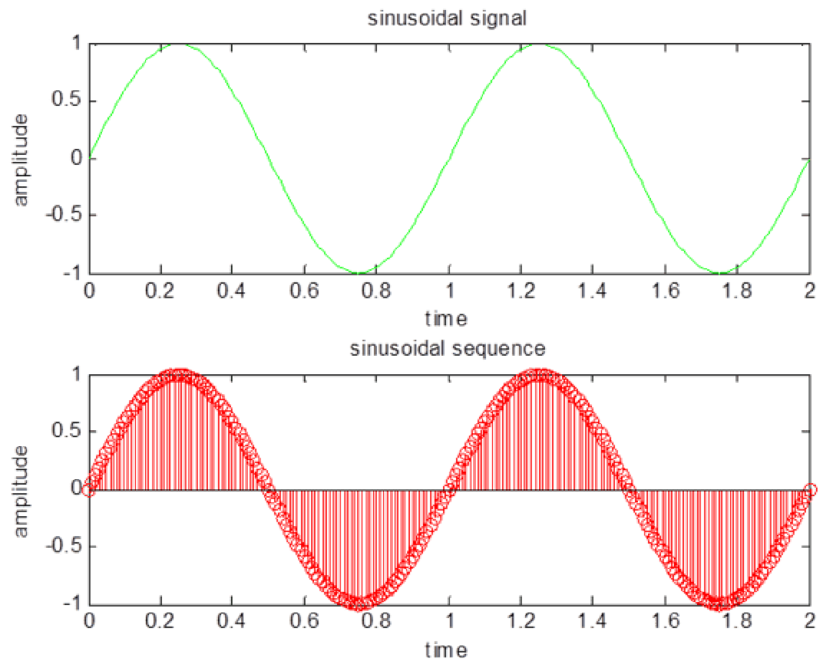
unit step function



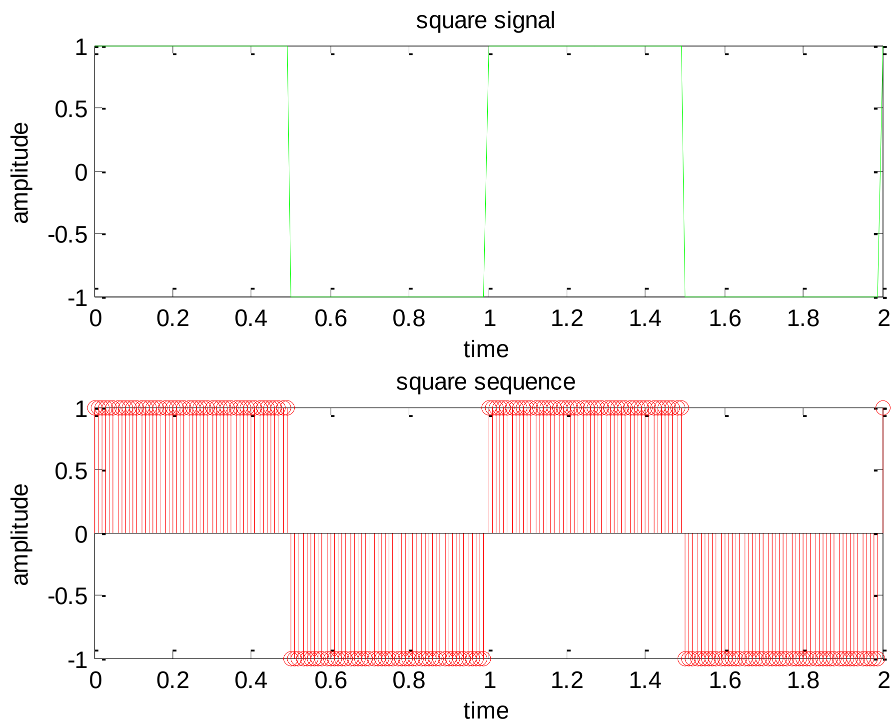
unit ramp function



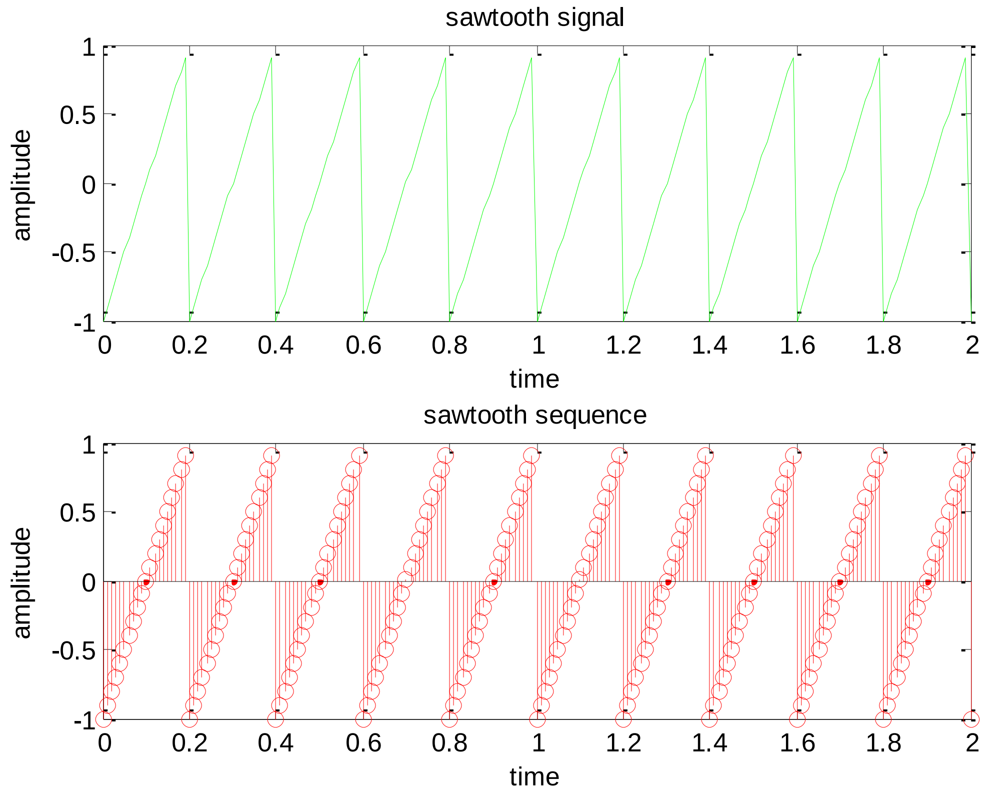
sinusoidal function



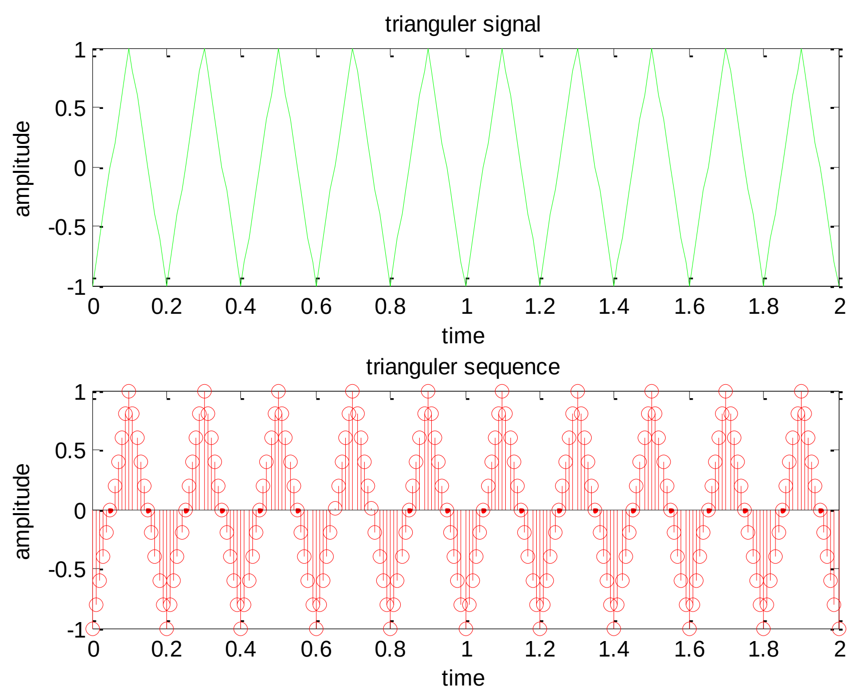
square function



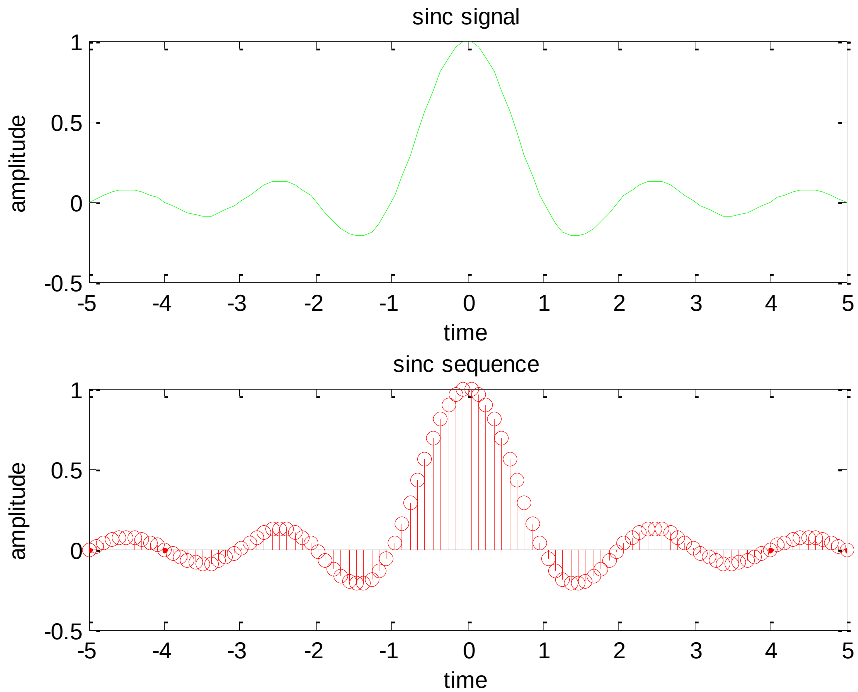
SAWTOOTH function



triangular function



sinc function



RESULT: Thus, the Generation of continuous time signals like unit step, sawtooth, triangular, sinusoidal, ramp and sinc functions are successfully completed by using MATLAB .

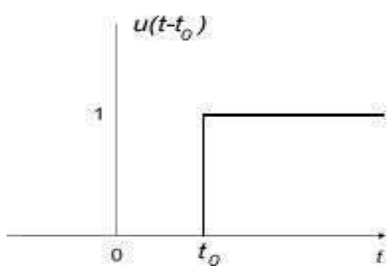
VIVA QUESTIONS:-

1. Define Symmetric and Anti-Symmetric Signals?
2. Define Continuous and Discrete Time Signals?
3. What are the Different types of representation of discrete time signals?
4. What are the Different types of Operation performed on signals?
5. What is System?
6.Explain Even Signals?
7.Explain Odd signals?
8.Define Periodic signals?
9.Define Aperiodic signals?
10.Differentiate Energy and Power signals with examples?
11.What is Energy Signals?
12.What is Power signals?
13.Define Systems?
14..How to determine time period of Periodic signals?
15.Define Frequency of signals?
16.Write MATLAB command for subtraction of two signals?
17.Write MATLAB command for addition of two signals?
18.Write MATLAB command for multiplication of two signals?
19.Write MATLAB command amplitude scaling of a sequence?
20.Write MATLAB command time scaling of a sequence?
21.Write MATLAB command time shifting of a sequence?
22.Write MATLAB command right shifting of a sequence?
23.Write MATLAB command left shifting of a sequence?
24.How to calculate even part of a signal?
25.How to calculate odd part of a signal?
26.Write condition for physical realization system?

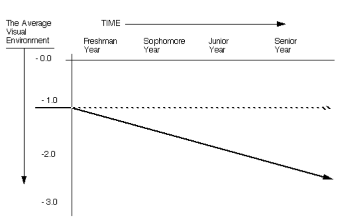
27.What is paley wiener criterion?
28.Write MATLAB command for division of two signals?
29.Write MATLAB command for left shift the signal to 8 times of the original signal
30Write MATLAB command for right shift the signal to 8 times of the original signal
31.Write MATLAB command for folding of a sequence?
32.Write MATLAB command for shift a positive time line signal to negative timeline signal
33.Write MATLAB command for shift a negative time line signal to positive timeline signal
34.Write MATLAB command for right shifting of a sequence to 8 times of the original signal
35.Write MATLAB command for left shifting of a sequence to 8 times of the original signal
36.Define complex exponential signal?
37.Define signum function?
38.Sketch the impulse signal?
39.Sketch the Double-sided exponential signal?
40.Define real exponential signal?
41.Define step function?
42.What is the sinusoidal signal?
43.Bring out the analogy between vectors and signals?
44.What is orthonormal vector and orthonormal set of vectors?
45.Define orthogonal signal space?
46.Define mean square error?
47.Explain orthogonal function?
48.What is basis vectors?
49.The relationship between unit step function and signum function?
50.Define rectangular function

EXERCISE QUESTIONS

1. Generate rectangular pulse function by using MATLAB.
2. Write a MATLAB program to get the above output where $t_0 = 2$

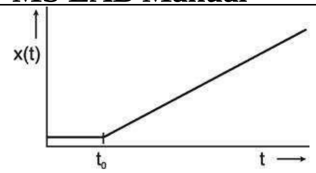


3. Write a program to get the result in signal
   r(t) = u(t) – 2*u(t+1)
4. Write a program to get the following output t = -1:-5



THE EYE'S VISUAL ENVIRONMENT IS APPROXIMATED BY A LINEAR RAMP
FIGURE 4

5. Write mat lab program to get the above output
   X(t) = 1; (0<t<2)
   = 10;(2<t<15) Note: Add ramp and unit



6. Write a MATLAB program to generate a sine wave with amplitude = 3, frequency 20Hz.
7. Write a MATLAB program to generate a cos wave with amplitude = 3, frequency 20Hz.
8. Write a MATLAB program to generate a triangular wave with amplitude = 8, frequency 10Hz.
9. Write a MATLAB program to generate a square wave with amplitude = 2, frequency 10kHz.
10. Write a MATLAB program to generate the signum function.
11. Write a MATLAB program to generate exponential growing signal.
12. Write a MATLAB program to generate exponential decaying signal.
13. Write a MATLAB program to generate a triangular wave with amplitude = 6, frequency 1Hz.
14. Write a MATLAB program to generate a square wave with amplitude = 5, frequency 5kHz.
15. Write a MATLAB program to generate a SAWTOOTH wave with amplitude = 8, frequency5Khz.
16. Write a MATLAB program to generate a sine wave with amplitude = 5, frequency 5Hz.
17. Write a MATLAB program to generate a cos wave with amplitude = 4, frequency 10Hz.
18. Write a MATLAB program to generate a triangular wave with amplitude = 5, frequency 4Hz.
19. Write a MATLAB program to generate a square wave with amplitude = 1, frequency 20kHz.
20. Write a MATLAB program to generate the signum function.
21. Write a MATLAB program to generate a square wave with amplitude = 4, frequency 5kHz.
22. Write a MATLAB program to generate a triangular wave with amplitude = 6, frequency 6Hz.
23. Write a MATLAB program to generate a triangular wave with amplitude = 8, frequency 2Hz.
24. Write a MATLAB program to generate a square wave with amplitude = 7, frequency 5kHz.
25. Write a MATLAB program to generate a SAWTOOTH wave with amplitude =10, frequency 10Khz.

Real Time Applications:

- Industrial control and automation (Control the velocity or position of an object)
- Examples: Controlling the position of a valve or shaft of a motor

EXPERMENT NO: 2

Even and Odd parts of a signal/sequence Real and Imaginary parts of a signal

AIM: -

Write the code / script for finding the Even and Odd parts of a signal/sequence Real and Imaginary parts of a signal

SOFTWARE REQURIED: -

1. MATLAB R2025a.
2. Windows XP SP2.

THEORY: -

1. Even and Odd Parts of a Signal

A signal (x(t)) (continuous) or (x[n]) (discrete) can always be decomposed into **even** and **odd** components.

- **Even part**: Symmetric around the origin.
  [
  x_e(t) = \frac{x(t) + x(-t)}{2} \quad \text{or} \quad x_e[n] = \frac{x[n] + x[-n]}{2}
  ]
- **Odd part**: Anti-symmetric around the origin.
  [
  x_o(t) = \frac{x(t) - x(-t)}{2} \quad \text{or} \quad x_o[n] = \frac{x[n] - x[-n]}{2}
  ]

1. (x(t) = x_e(t) + x_o(t)) (signal is sum of its even and odd parts)
2. (x_e(-t) = x_e(t)) (even symmetry)
3. (x_o(-t) = -x_o(t)) (odd symmetry)
4. The even part is like "mirror symmetric," the odd part flips sign across the origin.

Example (Continuous):
If (x(t) = e^{-t}),
[
x_e(t) = \frac{e^{-t} + e^t}{2} = \cosh(t), \quad x_o(t) = \frac{e^{-t} - e^t}{2} = -\sinh(t)
]

Example (Discrete):
If (x[n] = {1, 2, 3}) with (n = 0,1,2),
[
x_e[n] = \frac{x[n] + x[-n]}{2}, \quad x_o[n] = \frac{x[n] - x[-n]}{2}
]

2. Real and Imaginary Parts of a Signal

A signal (x(t)) or (x[n]) can also be **complex**, and any complex signal can be written as:

[
x(t) = \text{Re}{x(t)} + j , \text{Im}{x(t)}
]

Where:

- (\text{Re}{x(t)}) = real part of (x(t))
- (\text{Im}{x(t)}) = imaginary part of (x(t))
- (j = \sqrt{-1})

Example:
If (x(t) = 3 + 4j),
[
\text{Re}{x(t)} = 3, \quad \text{Im}{x(t)} = 4
]

to Even/Odd Parts

- **Real even**: A real signal that is symmetric.
- **Real odd**: A real signal that is anti-symmetric.
- **Imaginary even**: Imaginary part that is symmetric.
- **Imaginary odd**: Imaginary part that is anti-symmetric.

You can **decompose a complex signal** into four parts:

[
x(t) = \underbrace{\text{Re}{x(t)}\text{even}}\text{real even} + \underbrace{\text{Re}{x(t)}\text{odd}}\text{real odd} + j\underbrace{\text{Im}{x(t)}\text{even}}\text{imag even} + j\underbrace{\text{Im}{x(t)}\text{odd}}\text{imag odd}
]

PROCEDURE: -

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM: -

```
%Addition and multiplication of two signals%

% Define a sample complex signal
t = 0:0.1:10; % Time vector
x = sin(t) + 1i*cos(t); % Complex signal, using 1i for the imaginary unit

% Extract real and imaginary parts
real part = real(x);
imag_part = imag(x);
```

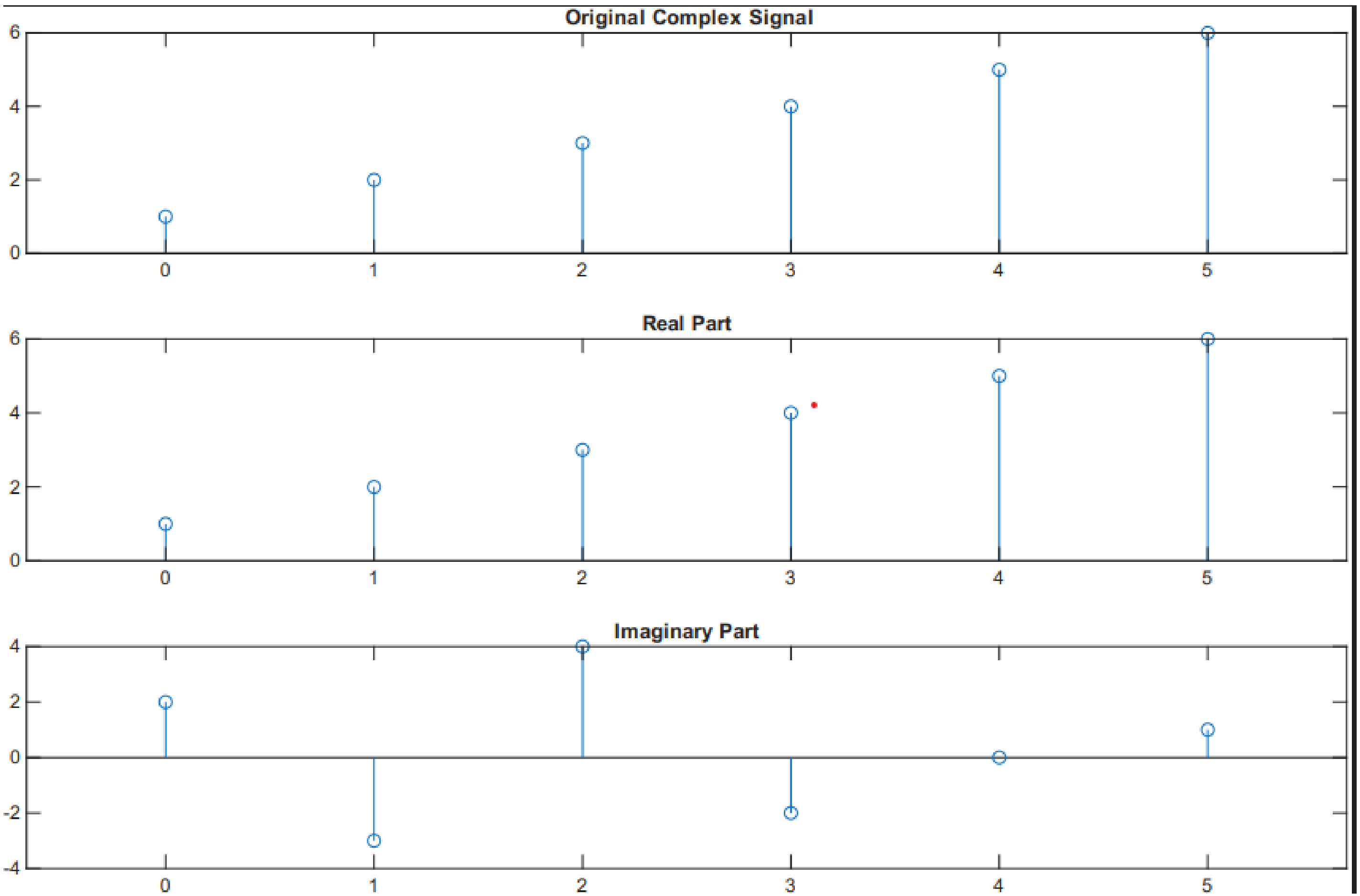
```
% Display or plot the results (optional)
figure;
subplot(3,1,1);
plot(t, abs(x));
title('Original Signal Magnitude');
xlabel('Time');
ylabel('Magnitude');
grid on;

subplot(3,1,2);
plot(t, real_part);
title('Real Part of the Signal');
xlabel('Time');
ylabel('Amplitude');
grid on;

subplot(3,1,3);
plot(t, imag_part);
title('Imaginary Part of the Signal');
xlabel('Time');
ylabel('Amplitude');
grid on;
```

```
% Define a sample discrete sequence and its index vector
n = -5:5; % Index vector (must be symmetric around 0 for standard decomposition)
x = [1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1]; % The signal sequence

% Ensure x and n have the same length
if length(n) ~= length(x)
    error('Error: Time vector n and signal vector x must have the same length.');
end

% Compute the time-reversed signal x(-n)
x_reversed = fliplr(x);

% Calculate the even and odd parts
even_part = 0.5 * (x + x_reversed);
odd_part = 0.5 * (x - x_reversed);

figure;
subplot(3,1,1);
stem(n, x, 'r');
title('Original Sequence');
xlabel('n');
ylabel('Amplitude');
grid on;

subplot(3,1,2);
stem(n, even_part, 'b');
title('Even Part of the Sequence');
xlabel('n');
ylabel('Amplitude');
grid on;
```

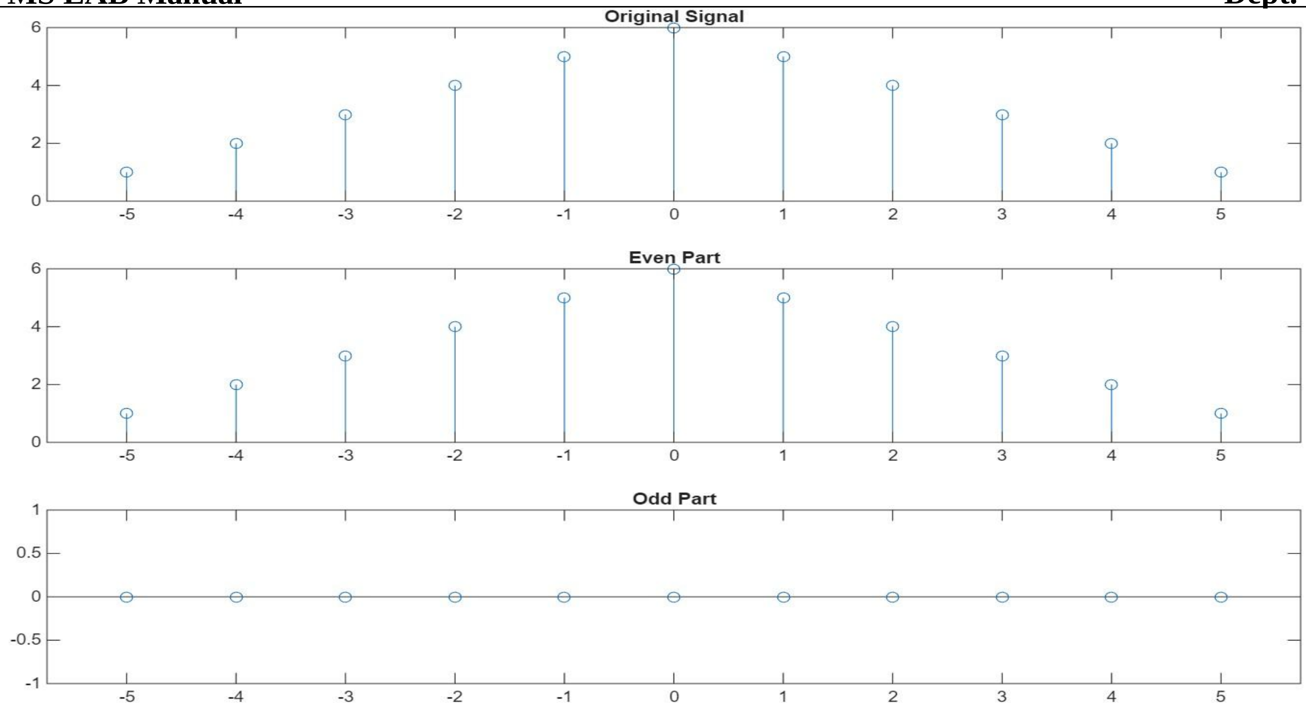
```
subplot(3,1,3);
stem(n, odd_part, 'g');
title('Odd Part of the Sequence');
xlabel('n');
ylabel('Amplitude');
grid on;
```

OUTPUT:-

Real and imaginary of two signals



Even and Odd of two signals



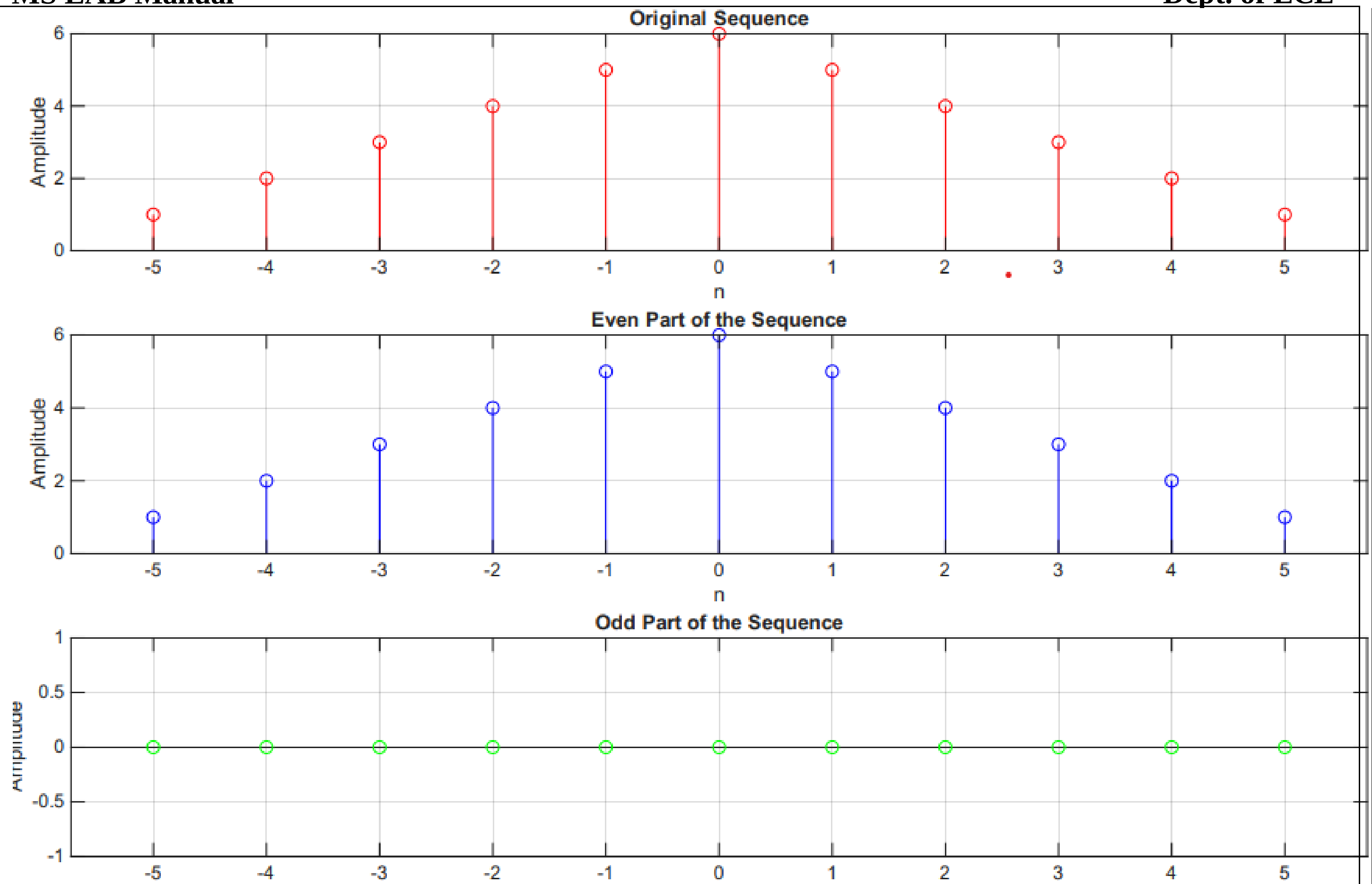
Amplitude scaling for signals

Time scaling for signals

Time shifting of a signal

Time folding of a signal

OUTPUT:-



RESULT:-

VIVA QUESTIONS:-

1. Define an **even signal**. Give an example.
2. Define an **odd signal**. Give an example.
3. How can any signal be expressed in terms of its **even and odd components**?
4. Write the formula for the **even part** of a signal (x(t)).
5. Write the formula for the **odd part** of a signal (x(t)).
6. Can a signal be both even and odd? Explain with an example.
7. What is the **even part** of (x(t) = t^3 + t^2)?
8. What is the **odd part** of (x(t) = t^3 + t^2)?
9. Explain why the **sum of the even and odd parts** equals the original signal.
10. If (x(t)) is **real-valued**, are its even and odd parts also real? Explain.

11. Define the **real part** of a complex signal (x(t)).
12. Define the **imaginary part** of a complex signal (x(t)).
13. Express a complex signal (x(t) = x_r(t) + j x_i(t)) in terms of its **real and imaginary parts**.
14. How can the **real and imaginary parts** of a signal be extracted mathematically?
15. Give an example of a signal with **purely real part**.
16. Give an example of a signal with **purely imaginary part**.
17. Can a signal be **both purely real and purely imaginary**? Explain.
18. What is the relationship between the **even/odd decomposition** and **real/imaginary decomposition**?
19. Explain why the **real part** of a signal is always even in Fourier Transform symmetry.
20. How does knowing the **real and imaginary parts** of a signal help in **signal processing applications** like filtering or modulation?

EXERCISE PROGRAMS

1. Write a MATLAB program to generate amplitude scaling of a sequence.
2. Write a MATLAB program to subtract two sinusoidal signals.
3. Write a MATLAB program to subtract and multiply two sinusoidal signals.
4. Write a MATLAB program to right shift the signal to 5 times of the original signal.
5. Write a MATLAB program to left shift the signal to 8 times of the original signal.
6. Write a MATLAB program to add two different signals with 2 <t<5
7. Write a MATLAB program to shift a positive time line signal to negative timeline signal.
8. Write a MATLAB program to subtract co-sinusoidal signals.
9. Write a MATLAB program to subtract two sinusoidal signals
10. Write a MATLAB program to division and multiply two co-sinusoidal signals.
11. Write a MATLAB program to generate time scaling of a sequence.
12. Write a MATLAB program to generate time shifting of a sequence.
13. Write a MATLAB program to generate time folding of a sequence.
14. Write a MATLAB program to generate amplitude scaling of a sequence with amplitude 5.
15. Write a MATLAB program to generate time scaling of a sequence with time 2sec.
16. Write a MATLAB program to add two different signals with 4 <t<8
17. Write a MATLAB program to shift a negative time line signal to positive timeline signal.
18. Write a MATLAB program to subtract sinusoidal signals.
19. Write a MATLAB program to subtract and divide two sinusoidal signals
20. Write a MATLAB program to add and multiply two co-sinusoidal signals.

REAL TIME APPLICATIONS:

- Signal Processing in Communications
- Block processing
- Vector processing

EXPERMENT NO: 3

CONVOLUTION AND AUTOCORRELATION AND CROSS CORRELATION BETWEEN SIGNALS AND SEQUENCES.

AIM: -
To find the output with linear convolution operation and compute auto correlation and cross correlation between signals and Sequences using MATLAB Software
SOFTWARE REQURIED:-
1.MATLAB R2025a.
2.Windows XP SP2.
THEORY:-
Linear Convolution involves the following operations.
1. Folding
2. Multiplication
3. Addition
4. Shifting
These operations can be represented by a Mathematical Expression as follows:

$$y[n] = \sum_{k=-\infty} x[k]h[n-k]$$

x[n]= Input signal Samples
h[n-k]= Impulse response co-efficient.
y[n]= Convolution output.
n = No. of Input samples
h = No. of Impulse response co-efficient.
Example : X(n)={1 2 -1 0 1}, h(n)={ 1,2,3,-1}.

In Signal processing, When the autocorrelation function is normalized by mean and variance, it is sometimes referred to as the autocorrelation coefficient. Given a signal f(t), the continuous auto correlation Rff(T) is most often defined as the continuous cross-correlation integral of f(t)with itself, at lag T

$$R_{ff}(\tau) = (f(t) * \overline{f}(-t))(\tau) = \int_{-\infty}^{\infty} f(t+\tau)\overline{f}(t)\,\mathrm{d}t = \int_{-\infty}^{\infty} f(t)\overline{f}(t-\tau)\,\mathrm{d}t.$$

The discrete autocorrelation Rat lag j for a discrete signal x(n) is

$$R_{xx}(j) = \sum_n x_n\,\overline{x}_{n-j}.$$

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 sliding inner-product. It is commonly used for searching a long signal for a shorter, known feature. It has applications in pattern recognition, single particle analysis, electron tomographic averaging, cryptanalysis, and neurophysiology.

For continuous functions f and g, the cross-correlation is defined as:

$$(f \star g)(\tau) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f^*(t)\ g(t+\tau)\,dt,$$

where f* denotes the complex conjugate of f and t is the time lag.

Similarly, for discrete functions, the cross-correlation is defined as**:**

$$(f \star g)[n] \stackrel{\text{def}}{=} \sum_{m=-\infty}^{\infty} f^*[m]\ g[m+n].$$

PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
%Convolution of two signals%
clc;
clear all;
 close all;
t=0:0.001:10;
x=sin(t);
h=square(t);
subplot(3,1,1);
plot(t,x,'g');
xlabel('time');
ylabel('amplitude');
title('sinusoidal signal');
subplot(3,1,2);
plot(t,h,'r');
xlabel('time');
ylabel('amplitude');
title('square function');
y=conv(x,h);
subplot(3,1,3);
plot(y);
xlabel('time');
ylabel('amplitude');
title('convolution signal');



%Convolution of two sequences%clc;
clear all;
```

```
close all;
L=input('enter the length of 1st sequence');
M=input('enter the length of 2nd sequence');
x=input('enter the first sequence:x(n)=');
h=input('enter the second sequence:y(n)=');
N=0:(L+M-1);
y=conv(x,h);
subplot(3,1,1);
stem(x,'g');
xlabel('discrete time');
ylabel('x(n)');
title('1st sequence');
subplot(3,1,2);
stem(h,'r');
xlabel('discrete time');
ylabel('h(n)');
title('second sequence');
subplot(3,1,3);
stem(y);
xlabel('discrete time');
ylabel('y(n)');
title('convolution of two sequences');
```

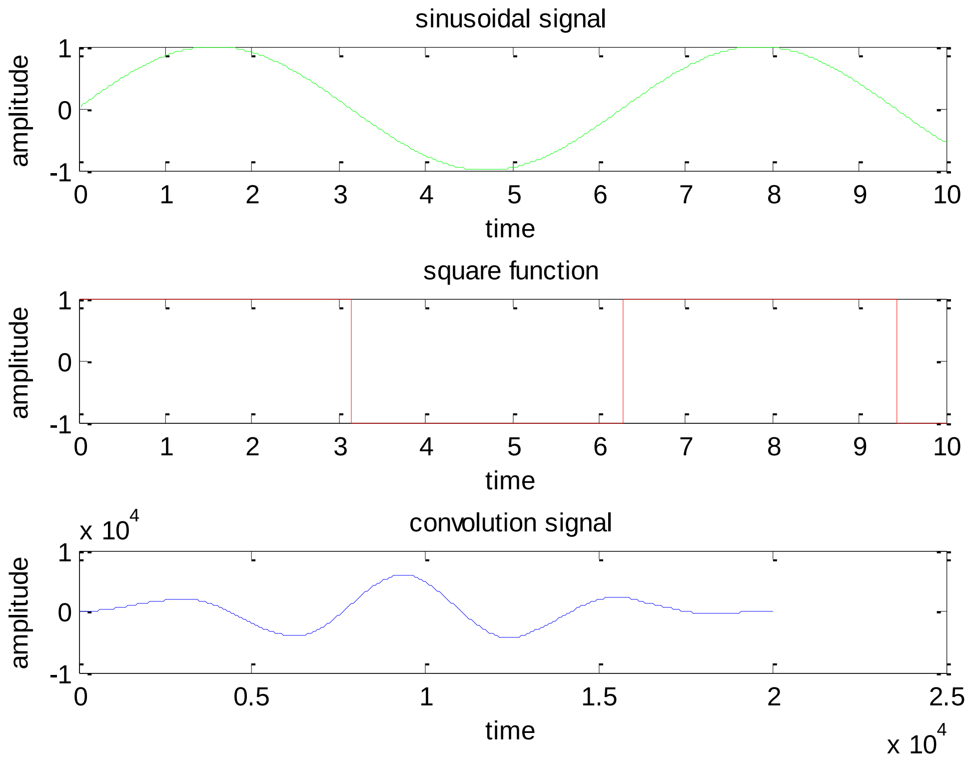
PROGRAM:-

```
%Auto correlation for a signal%
clc;
clear all;
close all;
t=0:0.001:1;
x=cos(2*pi*3*t);
a=xcorr(x);
subplot(2,1,1);
plot(t,x,'g');
xlabel('time');
ylabel('amplitude');
title('input signal');
subplot(2,1,2);
plot(a,'r');
xlabel('time');
ylabel('amplitude');
title('auto correlation signal');
 %cross correlation for a signal%
clc;
clear all;
close all;
t=0:0.001:1;
x=cos(2*pi*3*t);
y=cos(2*pi*5*t);
a=xcorr(x,y);
subplot(3,1,1);
plot(t,x,'g');
xlabel('time');
```

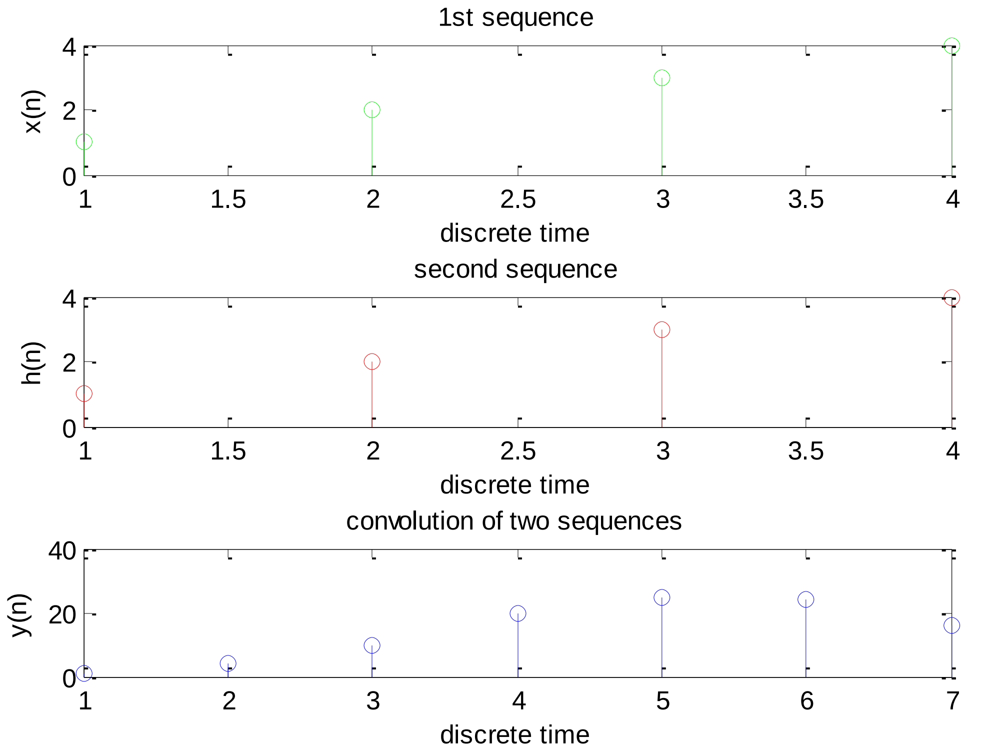
```
ylabel('amplitude');
title('1st signal');
subplot(3,1,2);
plot(t,y,'r');
xlabel('time');
ylabel('amplitude');
title('2nd signal');
subplot(3,1,3);
plot(a,'r');
xlabel('time');
ylabel('amplitude');
title('cross correlation signal');\
 %Auto correlation for a sequence%
clc;
clear all;
close all;
x=input('enter sample values')
a=xcorr(x);
subplot(2,1,1);
stem(x,'g');
xlabel('time');
ylabel('amplitude');
title('input sequence');
subplot(2,1,2);
stem(a,'r');
xlabel('time');
ylabel('amplitude');
title('auto correlation sequence');
%cross correlation for asequence%
clc;
clear all;
close all;
x=input('enter first sequence');
y=input('enter second sequence');
a=xcorr(x,y);
subplot(3,1,1);
stem(x,'g');
xlabel('time');
ylabel('amplitude');
title('1st sequence');
subplot(3,1,2);
stem(y,'r');
xlabel('time');
ylabel('amplitude');
title('2nd sequence');
subplot(3,1,3);
stem(a,'r');
xlabel('time');
ylabel('amplitude');
title('cross correlation sequence');
```

OUTPUT:-

Convolution of two signals

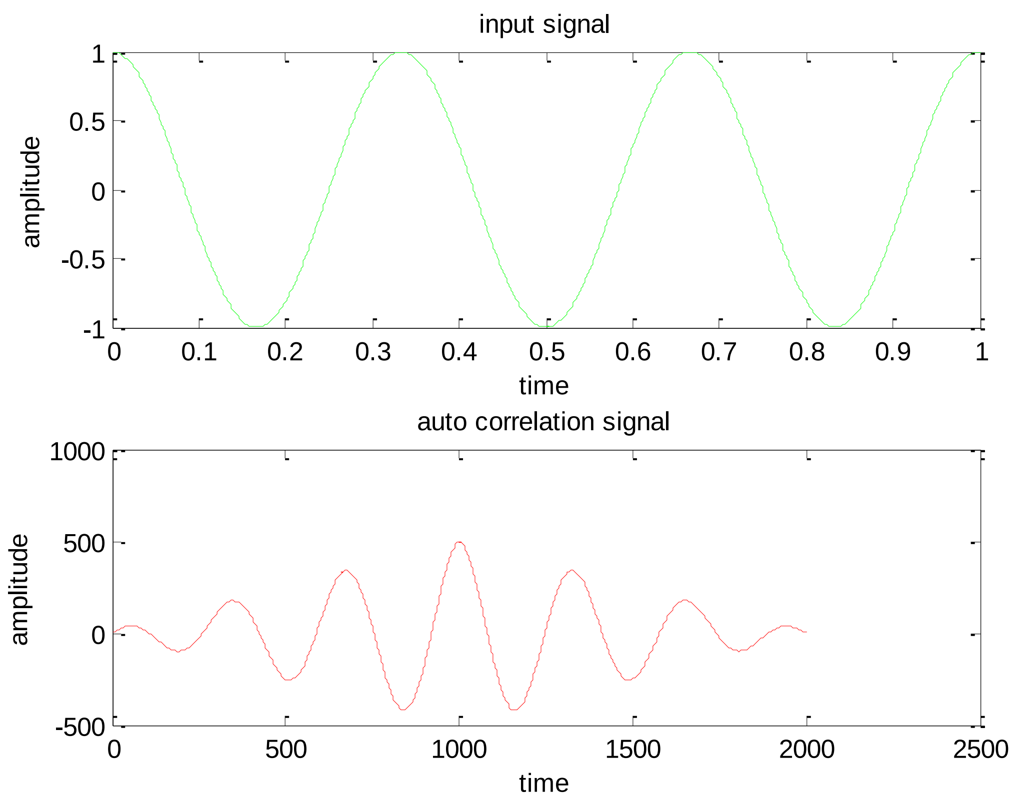


Convolution of two sequences
enter the length of 1st sequence4
enter the length of 2nd sequence4
enter the first sequence:x(n)=[1 2 3 4]
enter the second sequence:y(n)=[1 2 3 4]

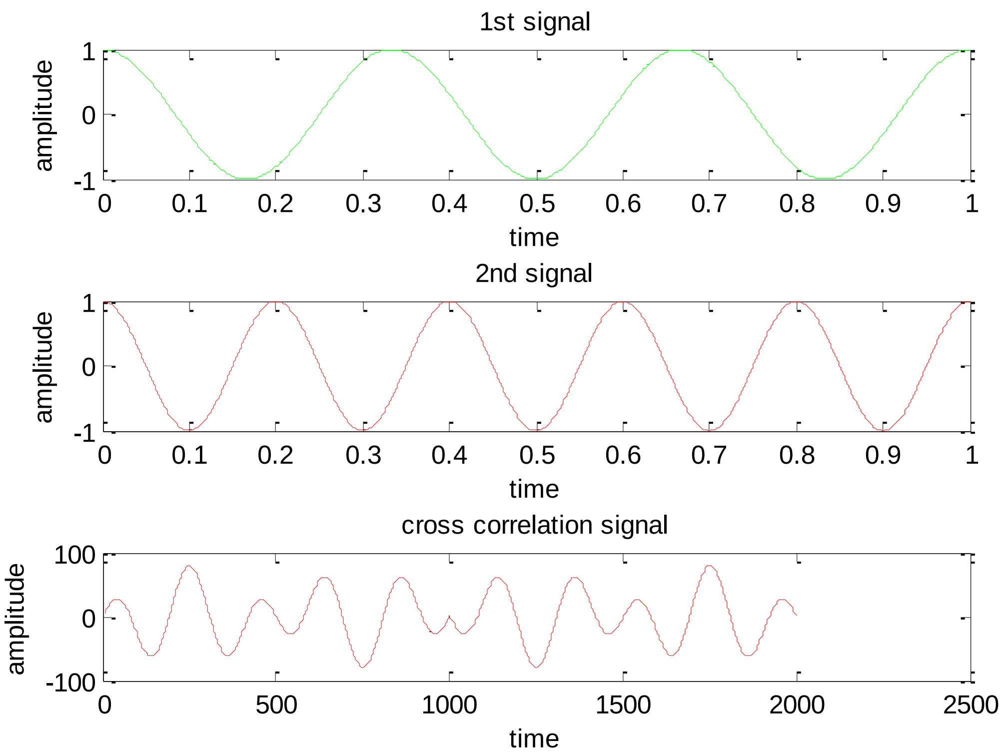


OUTPUT:-

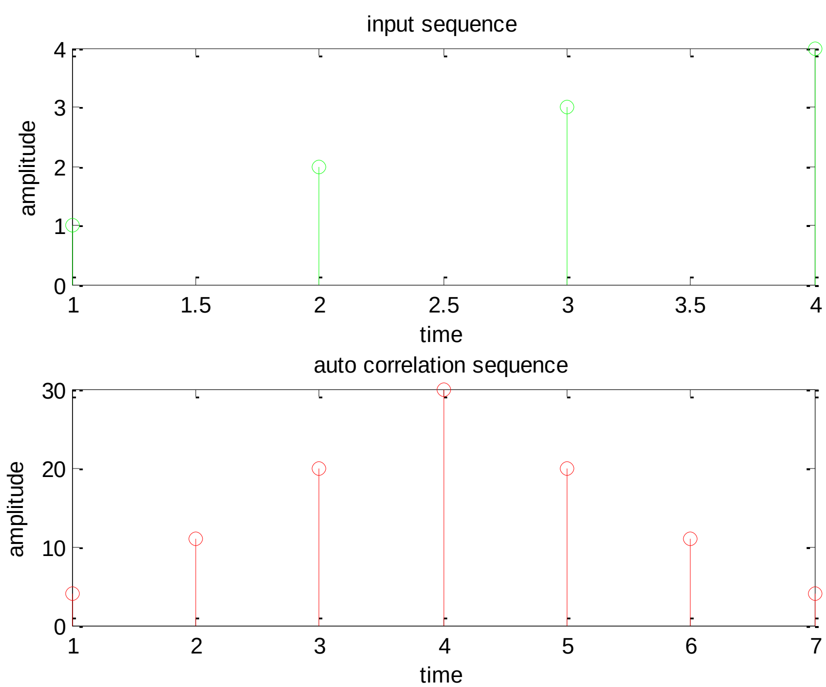
Auto correlation for a signal



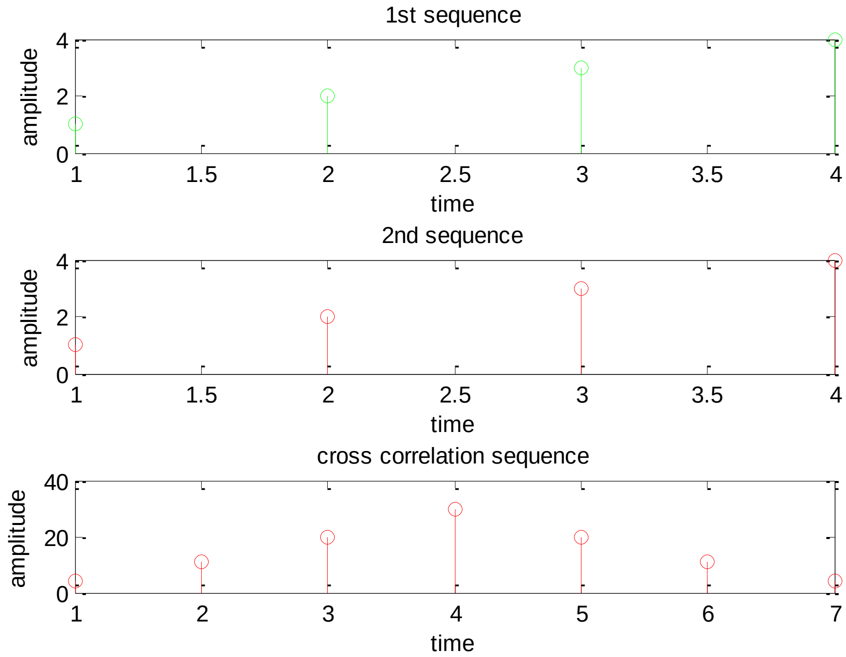
cross correlation for a signal



Auto correlation for a sequence
enter sample values[1 2 3 4]
x = 1 2 3 4



cross correlation for asequence
enter first sequence[1 2 3 4]
enter second sequence[1 2 3 4]



RESULT:-
In this experiment output with linear convolution operation and compute auto correlation and cross correlation between signals and Sequences have been performed Using MATLAB.

VIVA QUESTIONS: -

1. Define Convolution?
2. Define Properties of Convolution?
3. What is the Difference Between Convolution& Correlation?
4. What are Dirchlet Conditions of Fourier Series?
5. What is Half Wave Symmetry?
6. Define Linear Convolution?
7. Define Properties of Convolution in time domain?
8. What is the Difference Between Linear Convolution &Circular Correlation?
9. What are Dirchlet Conditions for Fourier Transform?
10. What is Full Wave Symmetry?
11. What is Correlation?
12. What is the importance of correlation?
13. What is the difference b/w correlation and convolution?
14. What is Auto Correlation?
15. What is Cross Correlation?
16. Relation between convolution and correlation?
17. State the commutative property of convolution?
18. State the associative property of convolution?
19. State the distributive property of convolution?
20. what is convolution integral?
21. What is quarter wave symmetry?
22. What is impulse response?
23. What is step response?
24. What is ramp response?
25. What is parabolic response?
26. How to calculate DC component of periodic signal?
27. How to calculate a_n component of periodic signal?
28. How to calculate b_n component of periodic signal?
29. Relation between energy and correlation?

30.Defne amplitude and phase spectrum?
31.Explain hidden symmetry?
32.Explain Gibbs phenomenon?
33.What are the Dirichlet conditions?
34.Write the parsevals theorem?
35.What is rotation symmetry?
36.What is power spectrum?
37.What is compact form fourier series?
38.What is fourier transform of impulse signal?
39.What is fourier transform of step signal?
40.What is fourier transform of ramp signal?
41.What is fourier transform of signum function?
42.What is fourier transform of SINC function?
43.When a periodic is said to have a half wave symmetry?
44.When a periodic is said to have a quater wave symmetry?
45.When a periodic is said to have a odd symmetry?
46.When a periodic is said to have a even symmetry?
47.What is the relationship between cosine and trigonometric representation?
48.What is the condition for half wave symmetry?
49.What is the condition for quater wave symmetry?
50.What is the condition for odd symmetry?

EXERCISE PROGRAMS

1.Write the MATLAB program to perform convolution between the following sequences
X(n)=[1 -1 4], h(n) = [-1 2 -3 1].
2. Write a mat lab program to perform the convolution between sinusoidal and ramp function and see how mat lab reacts to it.
3. Write a MATLAB program to perform convolution between square and step signal and see how mat lab reacts to it.
4. Write a MATLAB program to perform convolution between sinusoidal and ramp signal and see how mat lab reacts to it.
5. Write a MATLAB program to perform the convolution between X (n) = [1 2 3 5] and y (n) = [- 1 – 2] and see how MATLAB reacts to it.
6. Write a MATLAB program to perform the convolution between X (n) = [1 -3 5] and y (n) = [1 2 3 4] and see how MATLAB reacts to it.
7. Write a MATLAB program to perform the convolution between X (n) = [1 0 1 1] and y (n) = [1 0 0 0 0] and see how MATLAB reacts to it.
8.Write a MATLAB program to perform the convolution between X (n) = [1 1 1 1 0 0 0 0] and y (n) = [1 0 1 0 1 0 1 0] and see how MATLAB reacts to it.
9.Write a MATLAB program to perform the convolution between X (n) = [1 1 0 0] and y (n) = [1 1 1 1] and see how MATLAB reacts to it.
10.Write a MATLAB program to perform the convolution between X (n) = [6 7 8 9 10] and y (n) = [5 4 3 2 1] and see how MATLAB reacts to it.
11.Write the MATLAB program to perform convolution between the following sequences
X(n)=[2 -232, h(n) = [-24 -52].
12. Write a mat lab program to perform the convolution between step and ramp function and see how mat lab reacts to it.
13. Write a MATLAB program to perform convolution between square and step signal and see how mat lab

reacts to it.

14. Write a MATLAB program to perform convolution between sinusoidal and square signal and see how mat lab reacts to it.

15. Write a MATLAB program to perform the convolution between X (n) = [234 5] and y (n) = [-2 –4] and see how MATLAB reacts to it.

16. Write a MATLAB program to perform the convolution between X (n) = [3 -46] and y (n) = [2686] and see how MATLAB reacts to it.

17. Write a MATLAB program to perform the convolution between X (n) = [3120] and y (n) = [2434 5] and see how MATLAB reacts to it.

18.Write a MATLAB program to perform the convolution between X (n) = [22 1 0211 0] and y (n) = [1 1 1 2 1 0 1 0] and see how MATLAB reacts to it.

19.Write a MATLAB program to perform the convolution between X (n) = [2 1 21] and y (n) = [1 2 1 2] and see how MATLAB reacts to it.

20.Write a MATLAB program to perform the convolution between X (n) = [56987] and y (n) = [12 3 2 1] and see how MATLAB reacts to it.

21.Write the MATLAB program to perform convolution between the following sequences
X(n)=[4 -651], h(n) = [-13 -43].

22. Write a mat lab program to perform the convolution between ramp and step function and see how mat lab reacts to it.

23. Write a MATLAB program to perform convolution between square and step signal and see how mat lab reacts to it.

24. Write a MATLAB program to perform convolution between sinusoidal and step signal and see how mat lab reacts to it.

25. Write a MATLAB program to perform the convolution between X (n) = [1 5 6 5] and y (n) = [-3 –6] and see how MATLAB reacts to it.

Real Time Applications:

1.The convolution is to determine the response y[n] of a system of a known impulse response h[n] for a given input signal x[n] to obtain y[n].

2. Correlation is used to extract second (and higher) order statistics from any random signal.

EXPERMENT NO: 4

VERIFICATION OF LINEARITY AND TIME INVARIANCE PROPERTIES OF A GIVEN CONTINUOUS /DISCRETE SYSTEM

AIM: -
To compute linearity and time invariance properties of a given continuous /discrete
System.

SOFTWARE REQURIED: -
1.MATLAB R2025a.
2.Windows XP SP2.

THEORY: -

1. Linearity

A system is **linear** if it satisfies **two main principles**:

1. **Additivity**:
   If input (x_1(t)) gives output (y_1(t)) and input (x_2(t)) gives output (y_2(t)), then the input (x_1(t) + x_2(t)) should give output (y_1(t) + y_2(t)).

   Mathematically:
   [
   x_1(t) \rightarrow y_1(t), \quad x_2(t) \rightarrow y_2(t) \implies x_1(t)+x_2(t) \rightarrow y_1(t)+y_2(t)
   ]

2. **Homogeneity (Scaling)**:
   If input (x(t)) gives output (y(t)), then input (\alpha x(t)) should give output (\alpha y(t)), where (\alpha) is any scalar.

   Mathematically:
   [
   x(t) \rightarrow y(t) \implies \alpha x(t) \rightarrow \alpha y(t)
   ]

Linearity Test:
Combine the two:
[
\text{If } x_1(t) \rightarrow y_1(t) \text{ and } x_2(t) \rightarrow y_2(t), \text{ then } \alpha x_1(t) + \beta x_2(t) \rightarrow \alpha y_1(t) + \beta y_2(t)
]
If this holds for all inputs and scalars, the system is **linear**.

Example (Continuous-Time System)

System: (y(t) = 3x(t))

- Test additivity: (x_1 + x_2 \rightarrow 3(x_1+x_2) = 3x_1 + 3x_2) ☑
- Test homogeneity: (\alpha x \rightarrow 3(\alpha x) = \alpha (3x)) ☑

Example (Discrete-Time System)

System: $(y[n] = x[n]^2)$

- Test additivity: $((x_1 + x_2)^2 \neq x_1^2 + x_2^2)$

.

2. Time Invariance (TI)

A system is **time-invariant** if a **time shift in input produces the same time shift in output**.

- If $(x(t) \rightarrow y(t))$
- Then $(x(t-t_0) \rightarrow y(t-t_0))$

For **discrete-time**:

- If $(x[n] \rightarrow y[n])$
- Then $(x[n-n_0] \rightarrow y[n-n_0])$

Time Invariance Test:
Shift the input in time and see if the output shifts **exactly the same way**.

(Continuous-Time System)

System: $(y(t) = x(t-2))$

- Input shifted: $(x(t-t_0))$ → Output: $(y(t) = x((t-t_0)-2) = x(t-(t_0+2)))$
- TI check passes: output shifts by same amount? Yes, with consistent delay.

System: $(y(t) = t \cdot x(t))$

- Shift input: $(x(t-t_0) \rightarrow y(t) = t \cdot x(t-t_0))$
- But desired TI output: $(y(t-t_0) = (t-t_0) \cdot x(t-t_0) \neq t \cdot x(t-t_0))$

Conclusion: Not time-invariant.

Example (Discrete-Time System)

System: $(y[n] = x[n] + n)$

- Shift input: $(x[n-n_0] \rightarrow y[n] = x[n-n_0] + n)$
- Desired: $(y[n-n_0] = x[n-n_0] + (n-n_0))$ ✗

Conclusion: Not time-invariant.

3. Summary Table

Property	Continuous-Time Example	Discrete-Time Example	Linear?	Time-Invariant?
Linear, TI	(y(t)=3x(t))	(y[n]=2x[n])		
Nonlinear	(y(t)=x^2(t))	(y[n]=x[n]^2)		
Time-variant	(y(t)=t \cdot x(t))	(y[n]=x[n]+n)		

PROCEDURE: -

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
%Program1:%

clc;
clear;
close all;
%% Define system (EDIT THIS PART)
% Example systems:
% Continuous: y(t) = 2*x(t) + 3
% Discrete: y[n] = x[n] + x[n-1]

system_type = 'discrete'; % 'continuous' or 'discrete'

% Define system as function handle
% Change this according to your system

if strcmp(system_type,'continuous')
   sys = @(t,x) 2*x + 3;   % Example: NON-LINEAR (due to +3)
else
   sys = @(x) filter([1 1],1,x); % Example: y[n]=x[n]+x[n-1]
end

%% Generate input signals
t = 0:0.01:10;       % continuous time
n = 0:50;            % discrete time

x1 = sin(2*pi*0.5*t);
x2 = cos(2*pi*0.5*t);

x1d = sin(0.2*pi*n);
x2d = cos(0.2*pi*n);

a = 2; b = -1;       % constants for linearity test

%% ---------------- LINEARITY TEST ----------------
disp('--- LINEARITY TEST ---');

if strcmp(system_type,'continuous')
   y1 = sys(t,x1);
   y2 = sys(t,x2);
   y_combined = sys(t, a*x1 + b*x2);
```

```
    y_linear = a*y1 + b*y2;
else
    y1 = sys(x1d);
    y2 = sys(x2d);
    y_combined = sys(a*x1d + b*x2d);
    y_linear = a*y1 + b*y2;
end

error_lin = norm(y_combined - y_linear);

if error_lin < 1e-3
    disp('System is LINEAR');
else
    disp('System is NON-LINEAR');
end

%% ---------------- TIME INVARIANCE TEST ----------------
disp('--- TIME INVARIANCE TEST ---');

shift = 5;

if strcmp(system_type,'continuous')
x_shift = sin(2*pi*0.5*(t-shift));
y_shift_input = sys(t,x_shift);

y_original = sys(t,x1);
y_shift_output = interp1(t,y_original,t-shift,'linear',0);
else
x_shift = [zeros(1,shift) x1d(1:end-shift)];
y_shift_input = sys(x_shift);

y_original = sys(x1d);
y_shift_output = [zeros(1,shift) y_original(1:end-shift)];
end

error_ti = norm(y_shift_input - y_shift_output);

if error_ti < 1e-3
disp('System is TIME INVARIANT');
else
disp('System is TIME VARIANT');
end

%% ---------------- STABILITY TEST ----------------
disp('--- STABILITY TEST (BIBO) ---');

% bounded input
if strcmp(system_type,'continuous')
x_test = sin(t);
y_test = sys(t,x_test);
else
x_test = sin(n);
y_test = sys(x_test);
```

```
end

if max(abs(y_test)) < 100   % arbitrary threshold
disp('System is STABLE (BIBO)');
else
disp('System is UNSTABLE');
end

%% ---------------- CAUSALITY TEST ----------------
disp('--- CAUSALITY TEST ---');

if strcmp(system_type,'discrete')
% check if output depends on future values
x_future = x1d;
x_future(10) = 100; % disturb future

y1 = sys(x1d);
y2 = sys(x_future);

   if abs(y1(1) - y2(1)) < 1e-3
      disp('System is CAUSAL (Physically Realizable)');
   else
      disp('System is NON-CAUSAL');
   end
else
   disp('Causality test for continuous systems requires symbolic analysis');
end

disp('--- ANALYSIS COMPLETE ---');
```

OUTPUT:-

Program1:

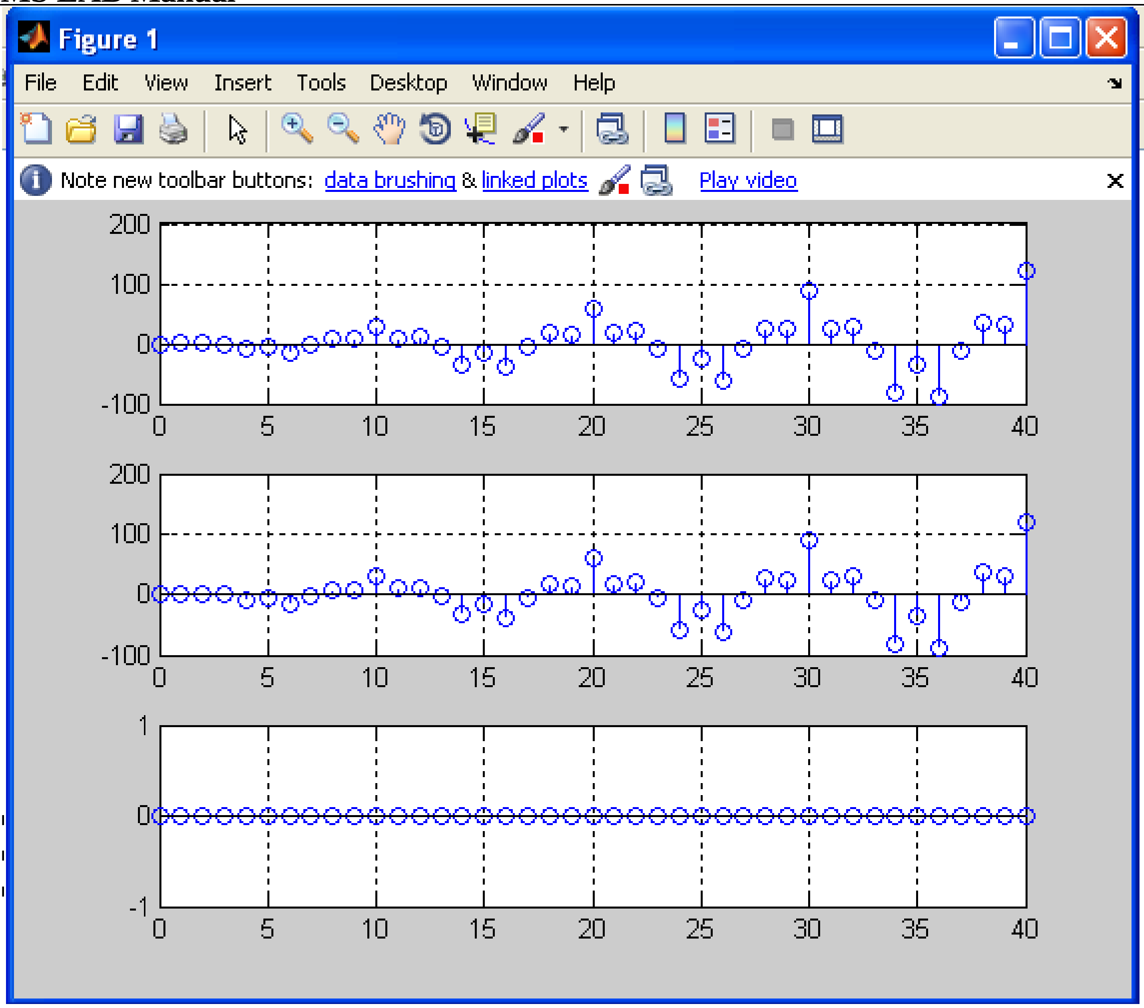
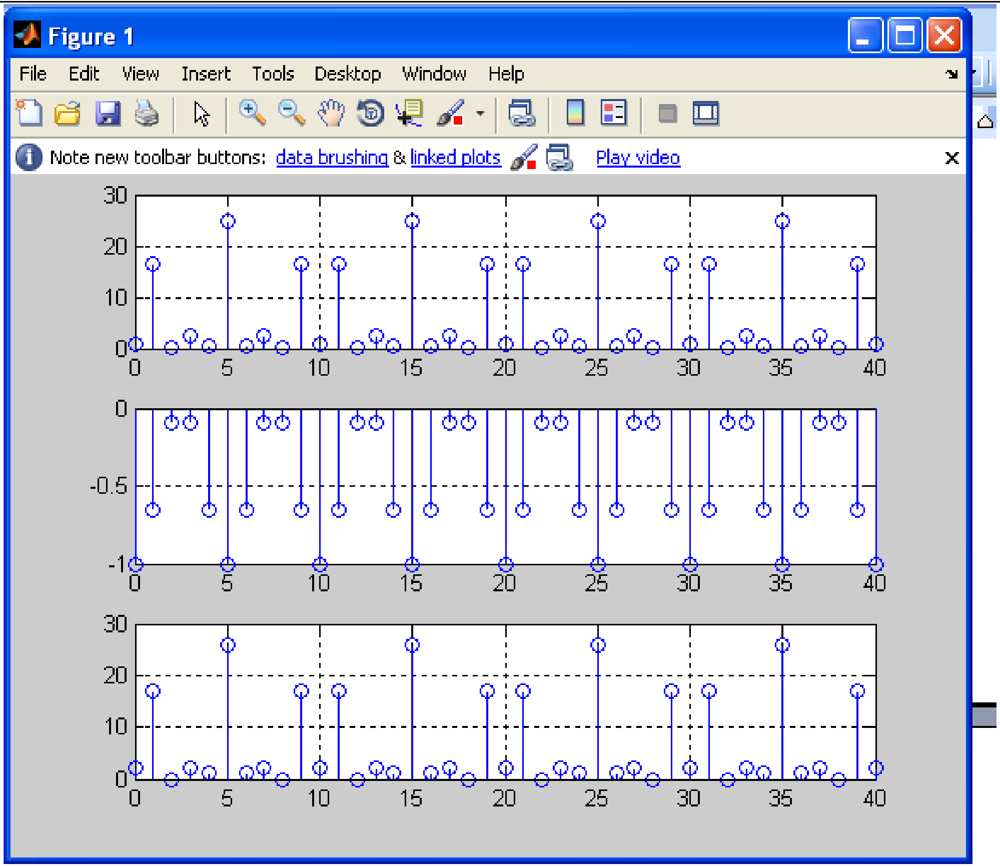
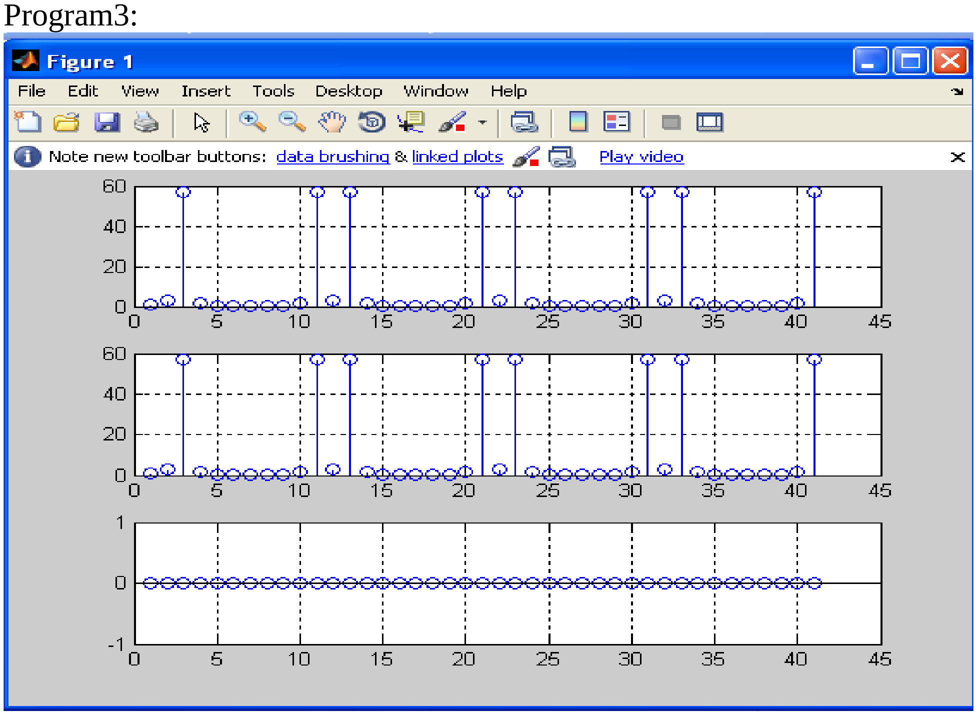


Figure 1
File Edit View Insert Tools Desktop Window Help
Note new toolbar buttons: data brushing & linked plots Play video
200
100
0
-100
0 5 10 15 20 25 30 35 40
200
100
0
-100
0 5 10 15 20 25 30 35 40
1
0
-1
0 5 10 15 20 25 30 35 40

Program2:



Program3:



VIVA QUESTIONS :

1. Define Systems?
2. What is LTI Systems?
3. Describe LTV Systems?
4.Summarize importance of Linear Systems?
5 Differentiate b/w Linear and Non-Linear Systems?
6. Summarize Properties of LTI Systems?

7.Explain two Properties of LTV Systems?
8.Appraise Superposition Principal?
9. Compare between CT and DT systems?
10.Memorize Causality of LTI Systems?
11.State Properties Of Convolution?
12.List the Applications Of Correlation?
13.Locate DFT ?
14.List classification of Continuous Time Signals? Name Them?
15Revise Examples of Causal Signal?
16.Apraise Amplitude Scaling And Time Scaling?
17.Classify Discrete Time Signal?
18.What is meant by Step Response Of The Dt System?
19.Define Impulse Response of a Dt System?
20.State The Significance of Difference Equations?
21.Write The Difference Equation For Discrete Time System?
22.What are the Properties of Convolution?
23.State the Commutative Properties of Convolution?
24.State the Associative Properties of Convolution?
25.Memorize Causal Lti Dt System?
26.What are the properties of continous Time fourier series.
27.State laplace transform.
28.Report the condition for convergance of L.T?
29. State region of convergance ?
30.State shifting property of L.T?
31.Memorize transfer function?
32.Memorize convolution property of L.T?
33.Define Time Variant And Time Invariant System?
34.State The Methods For Evaluating Inverse Z-transform.?
35.How to obtain The output Sequence of Linear Convolution through Circular Convolution?
36.What is Zero Padding?what Are Its Uses?
37.Define Sectional Convolution?
38.Why FFT Is Needed?
39.Distinguish Between Linear Convolution And Circular Convolution Of Two Sequences?
40.Justify Differences And Similarities Between Dif And Dit Algorithms?
41.What are The Different Types Of Filters Based On Impulse Response?
42.What are The Different Types Of Filters Based On Frequency Response?
43.What is Power signals?
44.Define Systems?
45.How to determine time period of Periodic signals?
46.Define Frequency of signals?
47.What is the difference b/w stem & plot?
48.What is Energy Signals?
49.What is Power signals?
50.Define Systems?

EXCERCISE QUESTIONS

1.Write a MATLAB program to verify the linearity property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the linearity property or not.

2.Write a MATLAB program to verify the linearity property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the linearity property or not

3.Write a MATLAB program to verify the linearity property of the following sequency x1= sin(2*pi*0.1*n); cos(2*pi*0.3*n),and chech whether it satisfies the linearity property or not

4.Write a MATLAB program to verify the time invariance property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the time invariance property or not.

5.Write a MATLAB program to verify the time invariance property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the time invariance property or not

6.Write a MATLAB program to verify the time invariance property of the following sequency x1= sin(2*pi*0.1*n); cos(2*pi*0.3*n),and chech whether it satisfies the time invariance property or not.

7.Write a MATLAB program to verify the time variance property of the following sequency x1= sin(2*pi*0.1*n); cos(2*pi*0.3*n),and chech whether it satisfies the time invariance property or not.

8.Write a MATLAB program to verify the non linearity property of the following sequency x1= sin(2*pi*0.1*n); cos(2*pi*0.3*n),and chech whether it satisfies the linearity property or not

9.Write a MATLAB program to verify the time variance property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the time invariance property or not.

10.Write a MATLAB program to verify the non linearity property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the linearity property or not.

11.Write a MATLAB program to verify the linearity property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the linearity property or not.

12.Write a MATLAB program to verify the linearity property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the linearity property or not

13.Write a MATLAB program to verify the linearity property of the following sequency x1= sin(2*pi*0.1*n); cos(2*pi*0.3*n),and chech whether it satisfies the linearity property or not

14.Write a MATLAB program to verify the time invariance property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the time invariance property or not.

15.Write a MATLAB program to verify the time invariance property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the time invariance property or not

16.Write a MATLAB program to verify the time invariance property of the following sequency x1= sin(2*pi*0.1*n); cos(2*pi*0.3*n),and chech whether it satisfies the time invariance property or not.

17.Write a MATLAB program to verify the time variance property of the following sequency x1= sin(2*pi*0.1*n); cos(2*pi*0.3*n),and chech whether it satisfies the time invariance property or not.

18.Write a MATLAB program to verify the non linearity property of the following sequency x1= sin(2*pi*0.1*n); cos(2*pi*0.3*n),and chech whether it satisfies the linearity property or not

19.Write a MATLAB program to verify the time variance property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the time invariance property or not.

20.Write a MATLAB program to verify the non linearity property of the following sequency x1= sin(2*pi*1*n);x2= sin(2*pi*2*n), and chech whether it satisfies the linearity property or not

Real Time Applications:

1. Wave Propagation such as sound and electromagnetic waves.
2. Electrical circuits composed of resistors, capacitors and inductors.

EXPERMENT NO: 5

Sinusoidal response and Impulse response of a given Continuous / Discrete LTI System

AIM: -

Write the MATLAB code / script for obtaining Sinusoidal response and Impulse response of a given Continuous / Discrete LTI System. a) Plot the Real and Imaginary part and b) Magnitude and Phase Plot of the response

SOFTWARE REQURIED:-

1.MATLAB R2025a.

2.Windows XP SP2.

THEORY:-

1. Impulse Response

The **impulse response** is the fundamental characteristic of a Linear Time-Invariant (LTI) system. It describes how the system reacts to a **delta function input**.

- **Continuous-time (CT) LTI system**:

If the system is defined by h(t)h(t)h(t), the impulse response:

y(t)=x(t)∗h(t)=∫−∞∞x(τ)h(t−τ)dτy(t) = x(t) * h(t) = \int_{-\infty}^{\infty} x(\tau) h(t-\tau) d\tauy(t)=x(t)∗h(t)=∫−∞∞ x(τ)h(t−τ)dτ

- **Discrete-time (DT) LTI system**:

If the system is defined by h[n]h[n]h[n], the impulse response:

y[n]=x[n]∗h[n]=∑k=−∞∞x[k]h[n−k]y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k] h[n-k]y[n]=x[n]∗h[n]=k=−∞∑∞x[k]h[n−k]

2. Sinusoidal Response

One remarkable property of LTI systems is that **sinusoids remain sinusoids at the output**, only changing in **amplitude and phase**.

Continuous-Time

For a continuous-time LTI system:

- Input: x(t)=Acos(ωt+ϕ)x(t) = A \cos(\omega t + \phi)x(t)=Acos(ωt+ϕ)
- Output: y(t)=∣H(jω)∣Acos(ωt+ϕ+∠H(jω))y(t) = |H(j\omega)| A \cos(\omega t + \phi + \angle H(j\omega))y(t)=∣H(jω)∣Acos(ωt+ϕ+∠H(jω))

Where:

- H(jω)H(j\omega)H(jω) is the **frequency response** of the system, obtained from the Fourier transform of h(t)h(t)h(t):

H(jω)=∫−∞∞h(t)e−jωtdtH(j\omega) = \int_{-\infty}^{\infty} h(t) e^{-j\omega t} dtH(jω)=∫−∞∞h(t)e−jωtdt

- ∣H(jω)∣|H(j\omega)|∣H(jω)∣ is the **gain** at frequency ω\omegaω
- ∠H(jω)\angle H(j\omega)∠H(jω) is the **phase shift** at frequency ω\omegaω

ne

For a discrete-time LTI system:

- Input: x[n]=Acos⁡(ω0n+ϕ)x[n] = A \cos(\omega_0 n + \phi)x[n]=Acos(ω0n+ϕ)
- Output: y[n]=∣H(ejω0)∣Acos⁡(ω0n+ϕ+∠H(ejω0))y[n] = |H(e^{j\omega_0})| A \cos(\omega_0 n + \phi + \angle H(e^{j\omega_0}))y[n]=∣H(ejω0)∣Acos(ω0n+ϕ+∠H(ejω0))

Where:

- H(ejω)H(e^{j\omega})H(ejω) is the **DTFT** of h[n]h[n]h[n]:

H(ejω)=∑n=−∞∞h[n]e−jωnH(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h[n] e^{-j\omega n}H(ejω)=n=−∞∑∞ h[n]e−jωn

3. Relationship Between Impulse Response and Sinusoidal Response

- The **impulse response** completely defines the LTI system.
- The **sinusoidal response** is derived from the **frequency response**, which is the Fourier transform of the impulse response.
  - Continuous: H(jω)=F{h(t)}H(j\omega) = \mathcal{F}\{h(t)\}H(jω)=F{h(t)}
  - Discrete: H(ejω)=DTFT{h[n]}H(e^{j\omega}) = \text{DTFT}\{h[n]\}H(ejω)=DTFT{h[n]}

PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

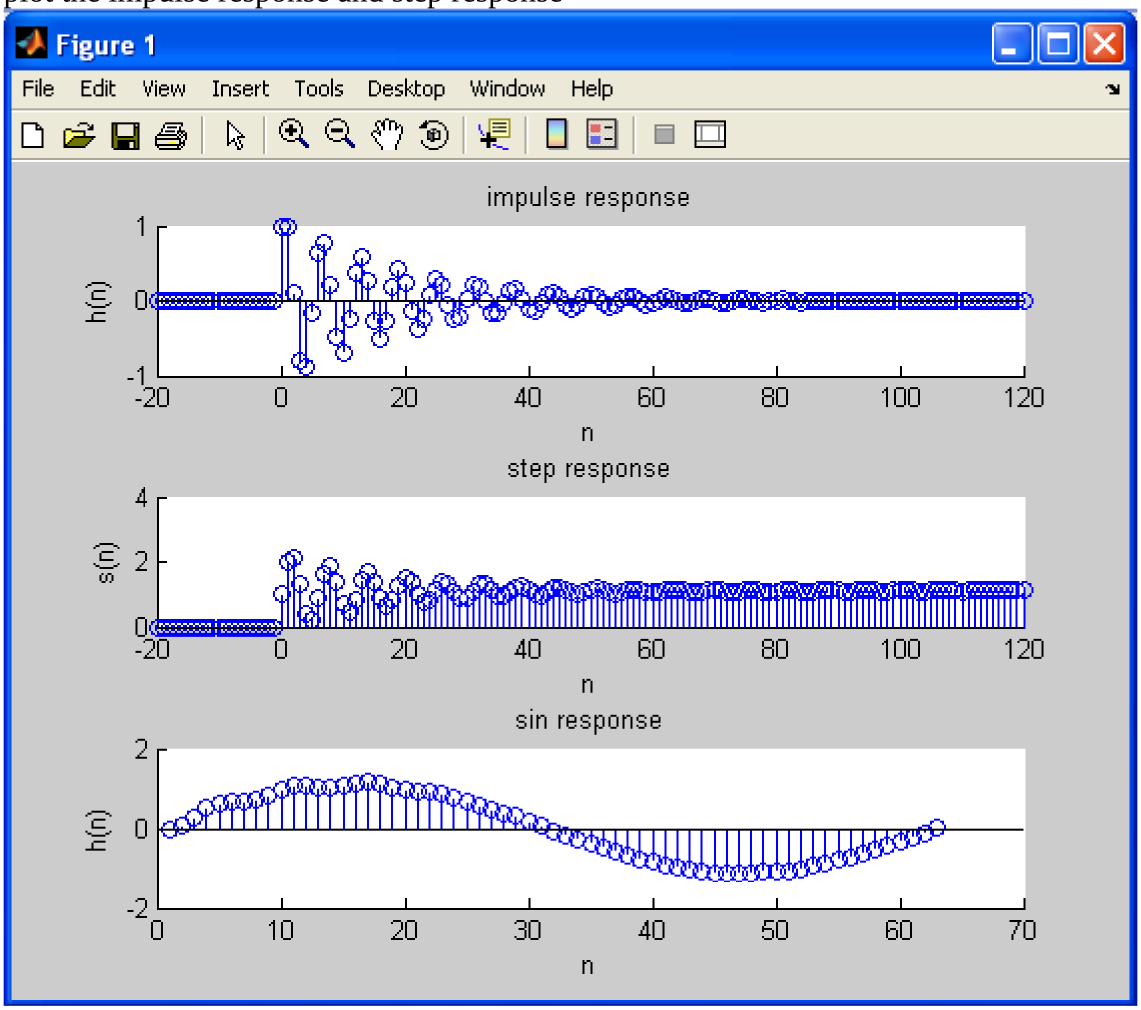
```
clc;
clear;
close all;

%% Define system (edit this)
% Example system: H(s) = 1 / (s^2 + 2s + 2)
num = [1];
den = [1 2 2];

%% Choose system type
isContinuous = true;   % true → Continuous, false → Discrete

if isContinuous
    sys = tf(num, den);     % Continuous system
else
    Ts = 1;                 % Sampling time
    sys = tf(num, den, Ts); % Discrete system
end

%% Time vector
t = 0:0.01:10;

%% -------------------------------
```

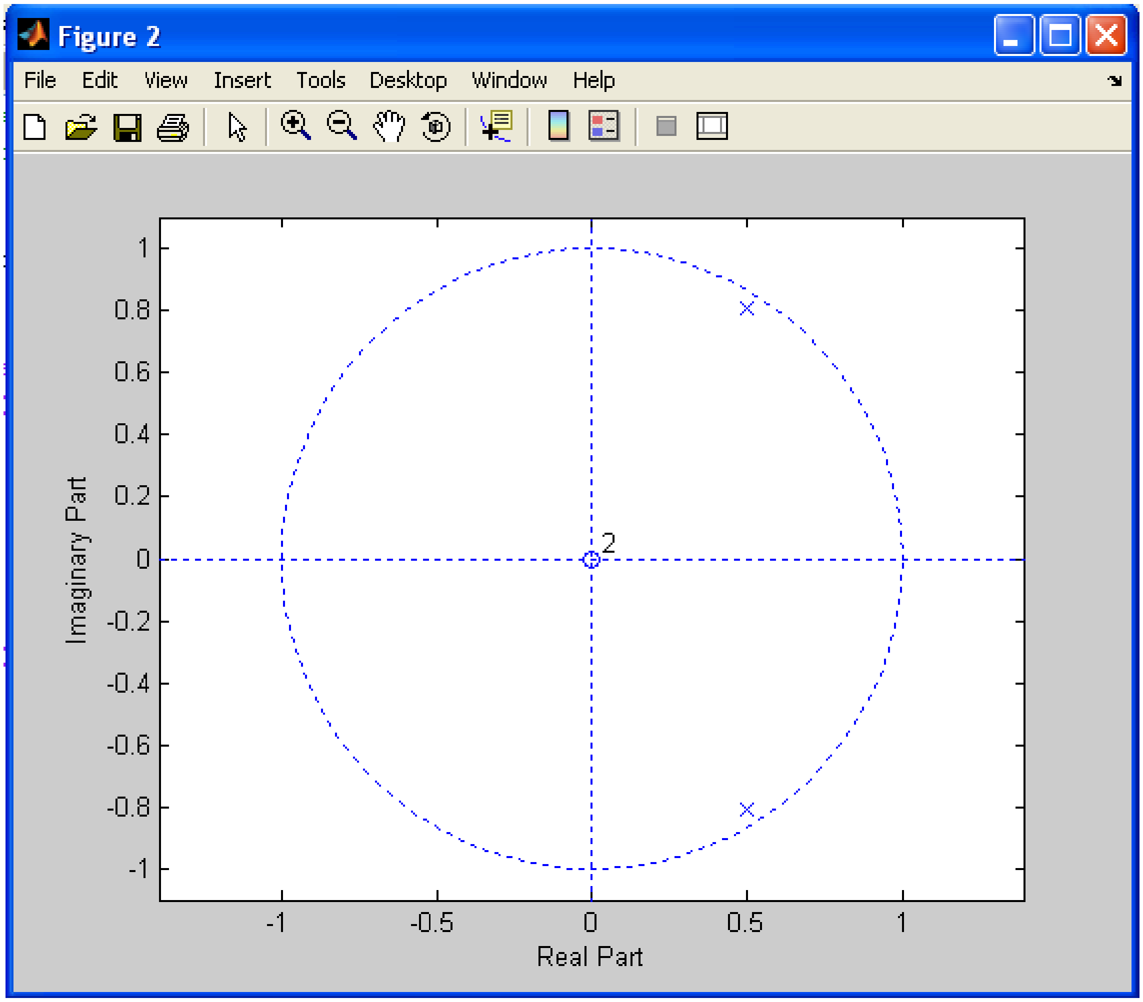
```
%% 1. Impulse Response
%% -------------------------------
figure;
impulse(sys, t);
title('Impulse Response');
grid on;

%% -------------------------------
%% 2. Sinusoidal Input Response
%% -------------------------------
A = 1;            % Amplitude
f = 1;            % Frequency (Hz)
w = 2*pi*f;       % Angular frequency

u = A * sin(w*t);   % Sinusoidal input

[y, t_out] = lsim(sys, u, t);

figure;
plot(t_out, y, 'b', 'LineWidth', 1.5);
title('Sinusoidal Response');
xlabel('Time');
ylabel('Output');
grid on;

%% -------------------------------
%% 3. Frequency Response
%% -------------------------------
w = logspace(-1, 2, 500);  % Frequency range

[H, wout] = freqresp(sys, w);
H = squeeze(H);

real_part = real(H);
imag_part = imag(H);
magnitude = abs(H);
phase = angle(H);

%% -------------------------------
%% (a) Real and Imaginary Plot
%% -------------------------------
figure;
subplot(2,1,1);
semilogx(wout, real_part, 'r', 'LineWidth', 1.5);
title('Real Part of Frequency Response');
xlabel('Frequency (rad/s)');
ylabel('Real');
grid on;

subplot(2,1,2);
semilogx(wout, imag_part, 'b', 'LineWidth', 1.5);
title('Imaginary Part of Frequency Response');
xlabel('Frequency (rad/s)');
```

```
ylabel('Imaginary');
grid on;

%% ------------------------------
%% (b) Magnitude and Phase Plot
%% ------------------------------
figure;
subplot(2,1,1);
semilogx(wout, magnitude, 'k', 'LineWidth', 1.5);
title('Magnitude Response');
xlabel('Frequency (rad/s)');
ylabel('|H(j\omega)|');
grid on;

subplot(2,1,2);
semilogx(wout, phase, 'm', 'LineWidth', 1.5);
title('Phase Response');
xlabel('Frequency (rad/s)');
ylabel('Phase (radians)');
grid on;
```

OUTPUT:-

plot the impulse response and step response





RESULT:-
In this experiment computation of unit sample, unit step andsinusoidal response of the given LTI system and verifying its physical realizabilityandstability properties Using MATLAB .

VIVA QUESTIONS:-

1. What is the sinusoidal response of an LTI system?
2. How does an LTI system respond to a sinusoidal input?
3. What is the significance of amplitude and phase in sinusoidal response?
4. What is steady-state response in sinusoidal analysis?
5. How does frequency affect the output of an LTI system?
6. Explain the difference between transient and steady-state sinusoidal response.
7. What is the transfer function approach to find sinusoidal response?
8. How is Bode plot related to sinusoidal response?
9. How do you determine the magnitude and phase response of a system?
10. Explain resonance in sinusoidal response.

11. What is the impulse response of an LTI system?
12. Why is the impulse response important?
13. How is impulse response related to convolution?
14. Explain the difference between continuous-time and discrete-time impulse responses.
15. How can the impulse response be used to determine system stability?
16. What is the relationship between impulse response and transfer function?
17. How does a system respond to a unit impulse?
18. How can impulse response be obtained from differential/difference equations?
19. Explain causal and non-causal impulse response.
20. How does discrete-time impulse response differ from continuous-time impulse response?

EXERCISE PROGRAM:-

1. Write a MATLAB program for generating u(n)-u(n-1).
2. Write a MATLAB program for generating delayed unit step response
3.Write a MATLAB program for generating delayed impulse response
4. Write a MATLAB program for generating u(n)+u(n-1) and verify how MATLAB reacts to it.
5. write a MATLAB program to find the energy & power of the signal x(t)=10sin(10t).
6. write a MATLAB program to find the even & odd of the signal x(t)=e^2t.
7. write a MATLAB program to find the even & odd of the signal x(t)=sin(t).
8. write a MATLAB program to find the even & odd of the signal x(t)=cos(t).
9. write a MATLAB program to find the energy & power of the signal x(t)=sin(t).
10. write a MATLAB program to find the energy & power of the signal x(t)=e^2t.
11. Write a MATLAB program for generating u(n-2)-u(n-1).
12. Write a MATLAB program for generating delayed step response with magnitude 2
13.Write a MATLAB program for generating delayed impulse response with magnitude
14. Write a MATLAB program for generating u(n+1)+u(n-1) and verify how MATLAB reacts to it.
15. Write a MATLAB program to find the energy & power of the signal x(t)=20sin(5t).
16. Write a MATLAB program to find the even & odd of the signal x(t)=e^3t.
17. Write a MATLAB program to find the even & odd of the signal x(t)=sin(t)+cos(t).
18. Write a MATLAB program to find the even & odd of the signal x(t)=cos(t)-sin(t).
19. Write a MATLAB program to find the energy & power of the signal x(t)=sin(t)-cos(t).
20. Write a MATLAB program to find the energy & power of the signal x(t)=e^4t.
21. Write a MATLAB program for generating u(n+2)-u(n-1).
22. Write a MATLAB program for generating delayed unit step response
23.Write a MATLAB program for generating delayed impulse response
24. Write a MATLAB program for generating u(n-3)+u(n-1) and verify how MATLAB reacts to it.
25. Write a MATLAB program to find the energy & power of the signal x(t)=5sin(20t).

Real Time Applications:

1. Electronic Circuits such as amplifiers and filters.
2. Mechanical motion from the interaction of masses, springs and dashpots (dampers).

EXPERMENT NO: 6

Finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Fourier Transform by using the properties where ever required

AIM: -

Write the code / script for finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Fourier Transform by using the properties where ever required

SOFTWARE REQURIED:-

1.MATLAB R2025a.
2.Windows XP2SP2.

THEORY:-

1. Fourier Transform Basics

For a continuous-time signal ($x(t)$), its Fourier Transform is:

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j 2 \pi f t} dt$$

Or in angular frequency ($\omega = 2 \pi f$):

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

- **Magnitude Spectrum:** ($|X(f)|$)
- **Phase Spectrum:** ($\angle X(f)$)

2. Using Fourier Transform Properties

Sometimes directly integrating is difficult. We can simplify using properties:

1. **Linearity:**
$$a x_1(t) + b x_2(t) \xrightarrow{\mathcal{F}} a X_1(f) + b X_2(f)$$
2. **Time Shifting:**
$$x(t-t_0) \xrightarrow{\mathcal{F}} X(f) e^{-j 2 \pi f t_0}$$
3. **Frequency Shifting:**
$$x(t) e^{j 2 \pi f_0 t} \xrightarrow{\mathcal{F}} X(f-f_0)$$

4. **Time Scaling:**
$$x(a t) \xrightarrow{\mathcal{F}} \frac{1}{|a|} X\left(\frac{f}{a}\right)$$
5. **Conjugation:**
$$x^(t) \xrightarrow{\mathcal{F}} X^(-f)$$
6. **Differentiation in Time:**
$$\frac{d}{dt} x(t) \xrightarrow{\mathcal{F}} j 2 \pi f X(f)$$
7. **Integration in Time:**
$$\int x(t) dt \xrightarrow{\mathcal{F}} \frac{X(f)}{j 2 \pi f} + \pi X(0) \delta(f)$$

---

3. Procedure to Find Magnitude and Phase Spectrum

1. **Identify the signal** ($x(t)$).
2. **Check if it matches a known FT pair** (like rect, sinc, delta, exponential, etc.).
3. **Apply FT properties** to simplify computation.
4. **Compute ($X(f)$) or ($X(\omega)$)** if needed.
5. **Magnitude Spectrum:**
$$|X(f)| = \sqrt{\text{Re}{X(f)}^2 + \text{Im}{X(f)}^2}$$
6. **Phase Spectrum:**
$$\angle X(f) = \arctan\left(\frac{\text{Im}{X(f)}}{\text{Re}{X(f)}}\right)$$

4. Example Using Properties

Let’s take:

$$x(t) = e^{-2 t} u(t)$$

where ($u(t)$) is the unit step function.

Step 1: Known FT

$$e^{-a t} u(t) \xrightarrow{\mathcal{F}} \frac{1}{a + j 2 \pi f}, \quad \text{Re}{a} > 0$$

Here ($a = 2$). So:

[
X(f) = \frac{1}{2 + j 2 \pi f}
]

Step 2: Magnitude Spectrum

[
|X(f)| = \frac{1}{\sqrt{2^2 + (2 \pi f)^2}} = \frac{1}{\sqrt{4 + (2 \pi f)^2}}
]

Step 3: Phase Spectrum

[
\angle X(f) = -\arctan\left(\frac{2 \pi f}{2}\right) = -\arctan(\pi f)
]

PROCEDURE:-

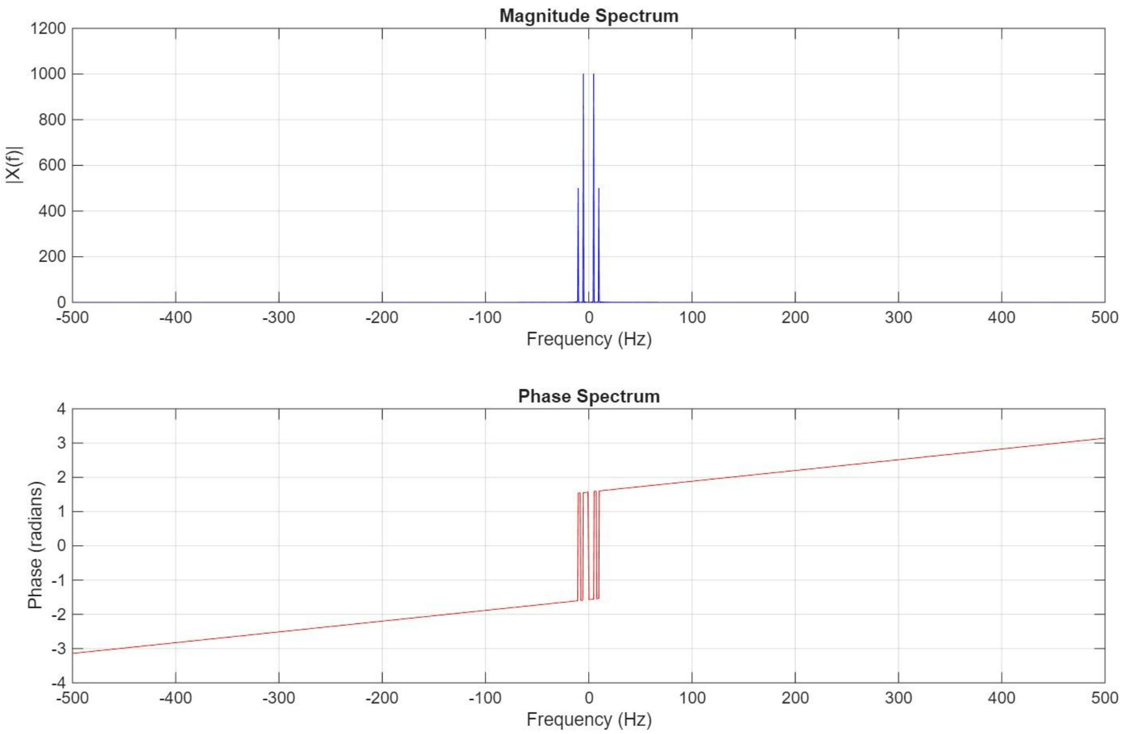
- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
clc;
clear;
close all;

% Define time axis
t = -1:0.001:1;   % Time vector

% Define signal (you can change this)
x = sin(2*pi*5*t) + 0.5*sin(2*pi*10*t);

% Number of samples
N = length(x);

% Compute Fourier Transform
X = fft(x);

% Shift zero frequency to center
X_shifted = fftshift(X);

% Frequency axis
fs = 1 / (t(2) - t(1));          % Sampling frequency
f = (-N/2:N/2-1)*(fs/N);          % Frequency vector

% Magnitude Spectrum
magnitude = abs(X_shifted);
```

```
% Phase Spectrum
phase = angle(X_shifted);

% Plotting
figure;

% Magnitude plot
subplot(2,1,1);
plot(f, magnitude, 'b');
title('Magnitude Spectrum');
xlabel('Frequency (Hz)');
ylabel('|X(f)|');
grid on;

% Phase plot
subplot(2,1,2);
plot(f, phase, 'r');
title('Phase Spectrum');
xlabel('Frequency (Hz)');
ylabel('Phase (radians)');
grid on;
```

OUTPUT:-



RESULT:-

In this experiment Finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Fourier Transform by using the properties have been demonstrated Using MATLAB .

VIVA QUESTIONS

1. What is the Fourier Transform of a signal, and why is it important in signal analysis?
2. Explain the difference between **magnitude spectrum** and **phase spectrum**.
3. What information does the magnitude spectrum provide about a signal?
4. What information does the phase spectrum provide about a signal?
5. Why do we often need both magnitude and phase to fully reconstruct a signal?

6. State and explain the **linearity property** of the Fourier Transform.
7. How does the **time-shifting property** affect the phase spectrum?
8. Explain the **frequency-shifting property** and its effect on the spectrum.
9. What is the **scaling property** of the Fourier Transform, and how does it affect magnitude and phase?
10. How does the **conjugation property** relate to real and imaginary parts of the Fourier Transform?
11. Explain the **duality property** in the context of Fourier analysis.

12. How would you find the Fourier Transform of a **sinusoidal signal**? Show the steps.
13. How do you find the magnitude and phase spectrum of a **rectangular pulse**?
14. For a **triangular signal**, explain how you would compute and plot the spectra.
15. How does the Fourier Transform of a **unit step function** differ from that of an impulse function?
16. Explain how **even and odd signals** affect the magnitude and phase spectra.

17. How do you plot the magnitude spectrum of a discrete-time signal versus a continuous-time signal?
18. How do you determine the **phase spectrum** when the Fourier Transform has negative and positive frequencies?
19. What are some common challenges or pitfalls when plotting the phase spectrum?
20. How can Fourier Transform properties be used to **simplify the computation** of spectra for complex signals?

EXERCISE PROGRAM:-

1. Generate a discrete-time sinusoidal signal x[n]=sin(2π5n/50)x[n] = \sin(2 \pi 5 n / 50)x[n]=sin(2π5n/50). Find and plot its magnitude and phase spectrum using the FFT.
2. Generate a cosine signal x[n]=cos(2π10n/100)x[n] = \cos(2 \pi 10 n / 100)x[n]=cos(2π10n/100). Compute its Fourier Transform and plot the magnitude and phase spectra.
3. Consider a unit step signal u[n]u[n]u[n] of length 50. Find and plot its magnitude and phase spectrum.
4. Generate an exponential signal x[n]=(0.9)nu[n]x[n] = (0.9)^n u[n]x[n]=(0.9)nu[n]. Find its Fourier Transform and plot magnitude and phase.
5. Generate a rectangular pulse of length 50. Plot its magnitude and phase spectrum.

6. Take the signal x[n]=cos(0.2πn)x[n] = \cos(0.2 \pi n)x[n]=cos(0.2πn) and compute the Fourier Transform of its time-shifted version x[n−5]x[n-5]x[n−5]. Plot the spectra and observe the phase changes.

7. For x[n]=sin(0.1πn)x[n] = \sin(0.1 \pi n)x[n]=sin(0.1πn), find the Fourier Transform of 2x[n]2 x[n]2x[n] and plot magnitude and phase. Discuss scaling property.
8. Generate x[n]=e−0.1nu[n]x[n] = e^{-0.1 n} u[n]x[n]=e−0.1nu[n]. Compute Fourier Transform of x[−n]x[-n]x[−n] and plot magnitude and phase spectra.
9. Create a signal x[n]=cos(πn/4)x[n] = \cos(\pi n / 4)x[n]=cos(πn/4). Compute the Fourier Transform of x[2n]x[2n]x[2n] and plot. Discuss time-scaling property effects.
10. Consider x[n]=sin(0.3πn)x[n] = \sin(0.3 \pi n)x[n]=sin(0.3πn). Compute and plot Fourier Transform of (−1)nx[n](-1)^n x[n](−1)nx[n] and explain frequency shift property.

11. Generate x[n]=cos(0.1πn)+cos(0.2πn)x[n] = \cos(0.1 \pi n) + \cos(0.2 \pi n)x[n]=cos(0.1πn)+cos(0.2πn). Plot its magnitude and phase spectrum.
12. Take x[n]=e−0.2nu[n]x[n] = e^{-0.2 n} u[n]x[n]=e−0.2nu[n] and multiply it by cos(0.2πn)\cos(0.2 \pi n)cos(0.2πn). Find Fourier Transform and plot spectra.
13. Generate x[n]=rectpuls(n−25,10)x[n] = \text{rectpuls}(n-25,10)x[n]=rectpuls(n−25,10). Modulate it with ej0.2πne^{j 0.2 \pi n}ej0.2πn and plot magnitude and phase.
14. Consider x[n]=u[n]−u[n−20]x[n] = u[n] - u[n-20]x[n]=u[n]−u[n−20]. Plot magnitude and phase spectrum using FFT.
15. Take x[n]=sin(0.1πn)+sin(0.3πn)x[n] = \sin(0.1 \pi n) + \sin(0.3 \pi n)x[n]=sin(0.1πn)+sin(0.3πn). Plot magnitude and phase spectra after time reversal x[−n]x[-n]x[−n].

16. Create x[n]=cos(0.05πn)x[n] = \cos(0.05 \pi n)x[n]=cos(0.05πn). Plot its spectrum after shifting the signal by 10 samples.
17. Generate x[n]=e−0.1nu[n]x[n] = e^{-0.1 n} u[n]x[n]=e−0.1nu[n]. Plot the spectrum after multiplying the signal by (−1)n(-1)^n(−1)n. Explain the effect.
18. For x[n]=triang(50)x[n] = \text{triang}(50)x[n]=triang(50), compute Fourier Transform and plot magnitude and phase.
19. Generate a combination x[n]=cos(0.1πn)+e−0.05nu[n]x[n] = \cos(0.1 \pi n) + e^{-0.05 n} u[n]x[n]=cos(0.1πn)+e−0.05nu[n]. Plot the magnitude and phase spectrum.
20. Create x[n]=sin(0.2πn)x[n] = \sin(0.2 \pi n)x[n]=sin(0.2πn) for n=0:49n = 0:49n=0:49. Compute FFT, normalize, and plot both magnitude and phase. Compare with theoretical DTFT.

EXPERMENT NO: 7

Finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Laplace Transform by using the properties where ever required. Also plot pole-zero diagram in S-plane

AIM: -
Write the matlab code / script for finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Laplace Transform by using the properties where ever required. Also plot pole-zero diagram in S-plane.

SOFTWARE REQURIED:-
1.MATLAB R2025a.
2.Windows XP SP2.

THEORY:-

The **Laplace Transform** of a signal (x(t)) is defined as:

[
X(s) = \mathcal{L}{x(t)} = \int_{0}^{\infty} x(t) e^{-st} dt
]

where (s = \sigma + j\omega).

Step 1: Identify the signal

Suppose we have a signal:

[
x(t) = e^{-at} u(t)
]

where (u(t)) is the unit step function and (a > 0).

Step 2: Apply Laplace Transform

[
X(s) = \int_0^\infty e^{-at} e^{-st} dt = \int_0^\infty e^{-(s+a)t} dt
]

[
X(s) = \frac{1}{s + a}, \quad \text{Re}(s) > -a
]

Properties you may use:

- **Linearity**: (\mathcal{L}{ax(t) + by(t)} = aX(s) + bY(s))
- **Time Shifting**: (\mathcal{L}{x(t-t_0)u(t-t_0)} = e^{-s t_0} X(s))
- **Differentiation**: (\mathcal{L}{\frac{dx}{dt}} = s X(s) - x(0^-))
- **Integration**: (\mathcal{L}{\int_0^t x(\tau)d\tau} = \frac{X(s)}{s})

2. Magnitude and Phase Spectrum

Once you have (X(s)), replace (s) with (j\omega) to get the **frequency response**:

[
X(j\omega) = X(s)|_{s = j\omega}
]

For (X(s) = \frac{1}{s + a}):

[
X(j\omega) = \frac{1}{j\omega + a}
]

[
|X(j\omega)| = \frac{1}{\sqrt{\omega^2 + a^2}}
]

[
\angle X(j\omega) = -\tan^{-1}\left(\frac{\omega}{a}\right)
]

You can **plot** these using any tool like Python/Matlab.

3. Pole-Zero Plot in S-Plane

- **Poles**: Values of (s) where (X(s) \to \infty) → Denominator = 0
- **Zeros**: Values of (s) where (X(s) = 0) → Numerator = 0

For (X(s) = \frac{1}{s + a}):

- Zero at (\infty) (no finite zeros)
- Pole at (s = -a)

Pole-zero diagram:

```
   Im(s)
    |
    |
    |       x  ← pole at s = -a
----+---------------- Re(s)
    |
```

- The **x** marks the pole.
- Zeros are usually marked with **o**.

PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

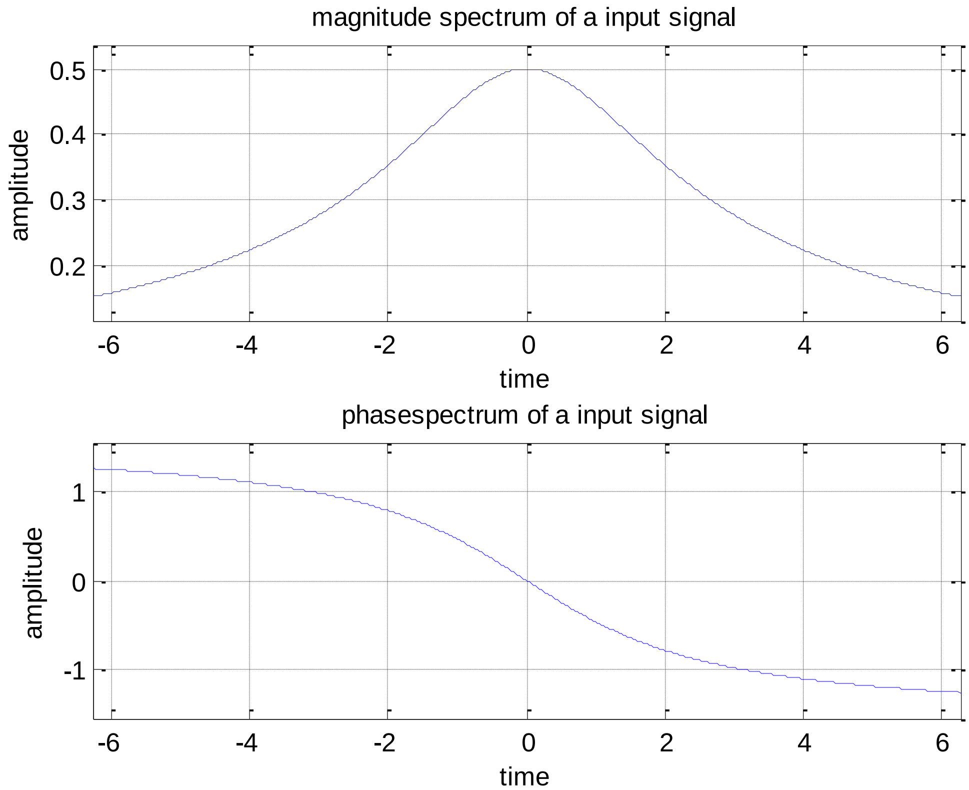
PROGRAM:-

```
clc;
clear;
close all;

%% Define symbolic variables
syms t s w real

%% Define the signal (you can modify this)
% Example: x(t) = e^(-2t) * u(t)
x = exp(-2*t) * heaviside(t);

%% Compute Laplace Transform
X_s = laplace(x, t, s);
disp('Laplace Transform X(s):');
pretty(X_s)

%% Substitute s = j*w to get frequency response
X_jw = subs(X_s, s, 1i*w);

%% Convert to MATLAB function for plotting
X_func = matlabFunction(X_jw, 'Vars', w);

%% Frequency range
w_vals = linspace(-20, 20, 1000);

%% Evaluate magnitude and phase
X_vals = X_func(w_vals);

magnitude = abs(X_vals);
phase = angle(X_vals);

%% Plot Magnitude Spectrum
figure;
subplot(2,1,1);
plot(w_vals, magnitude, 'LineWidth', 1.5);
grid on;
title('Magnitude Spectrum |X(j\omega)|');
xlabel('\omega');
ylabel('Magnitude');

%% Plot Phase Spectrum
subplot(2,1,2);
plot(w_vals, phase, 'LineWidth', 1.5);
```

```
grid on;
title('Phase Spectrum ∠X(j\omega)');
xlabel('\omega');
ylabel('Phase (radians)');

%% Pole-Zero Plot
% Extract numerator and denominator
[num, den] = numden(X_s);

% Convert symbolic polynomials to numeric coefficients
num_coeff = sym2poly(num);
den_coeff = sym2poly(den);

% Plot pole-zero diagram
figure;
pzmap(tf(num_coeff, den_coeff));
title('Pole-Zero Plot in S-plane');
grid on;
```

OUTPUT:-

```
Fourier Transform
Fourier Transform of input signal
Y=
1/(2+w*i)
Inverse Fourier Transform of input signal
Z=
Heaviside(x)/exp(2*x)
```



RESULT:-

In this experiment the Finding and plotting the Magnitude and Phase Spectrum of any given Signal by finding its Laplace Transform by using the properties where ever required. Also plot pole-zero diagram in S-plane have
been demonstrated using MATLAB .

VIVA QUESTIONS:-

1. What is the Laplace Transform of a signal, and why is it used for analyzing signals in the s-domain?
2. Explain the difference between the **magnitude spectrum** and **phase spectrum** of a signal.
3. What is the **region of convergence (ROC)** in Laplace Transform, and how does it affect stability?
4. How do poles and zeros in the s-plane relate to the frequency response of a system?
5. Explain why we use properties like **linearity, time-shifting, and differentiation** in Laplace Transform.
6. What is the physical significance of the **phase angle** in a signal's frequency spectrum?
7. How does the **frequency response** relate to evaluating (H(s)) along the (s = j\omega) axis?

8. Why do we plot the **pole-zero diagram** before plotting the magnitude and phase spectrum?
9. How do the locations of poles on the jω-axis affect the **resonance** of a system?
10. How do zeros in the s-plane influence the frequency response of a signal?

11. Find the Laplace Transform of (x(t) = e^{-2t}u(t)) and sketch its pole-zero diagram.
12. Determine the magnitude and phase spectrum of (X(s) = \frac{1}{s+3}).
13. Plot the pole-zero diagram for (X(s) = \frac{s+1}{s^2 + 4s + 5}) and discuss the stability.
14. Using Laplace Transform properties, find the transform of (x(t) = t \cdot e^{-t} u(t)).
15. Explain how to obtain the **frequency response** from the Laplace Transform (X(s)) for (s = j\omega).
16. Given (X(s) = \frac{10}{s(s+2)}), find and plot its **magnitude and phase spectrum**.
17. Sketch the magnitude and phase spectrum for (X(s) = \frac{s+2}{s^2 + 2s + 2}).
18. How do you handle **complex conjugate poles** when plotting magnitude and phase?
19. If a system has a **zero at the origin**, how does it affect the low-frequency magnitude?
20. For (X(s) = \frac{s+3}{s^2 + 2s + 10}), plot the **pole-zero diagram**, identify the type of system, and discuss the resonance effect on the magnitude spectrum.

EXERCISE PROGRAMS

1. Find the Laplace Transform of x(t)=e−2tu(t)x(t) = e^{-2t}u(t)x(t)=e−2tu(t) and plot its magnitude and phase spectrum. Plot its pole-zero diagram.
2. For x(t)=sin(3t)u(t)x(t) = \sin(3t)u(t)x(t)=sin(3t)u(t), compute the Laplace Transform using properties and plot magnitude and phase. Show the S-plane poles and zeros.
3. Determine the Laplace Transform of x(t)=cos(4t)u(t)x(t) = \cos(4t)u(t)x(t)=cos(4t)u(t) and plot the magnitude and phase spectrum along with the pole-zero diagram.
4. Given x(t)=e−tcos(2t)u(t)x(t) = e^{-t}\cos(2t)u(t)x(t)=e−tcos(2t)u(t), find its Laplace Transform and plot the spectra. Plot the poles and zeros in the S-plane.
5. For x(t)=te−3tu(t)x(t) = t e^{-3t}u(t)x(t)=te−3tu(t), compute the Laplace Transform and plot its magnitude, phase spectrum, and pole-zero diagram.

6. Find the Laplace Transform of x(t)=e−2tsin(5t)u(t)x(t) = e^{-2t}\sin(5t)u(t)x(t)=e−2tsin(5t)u(t) using Laplace properties. Plot magnitude, phase, and poles/zeros.
7. For x(t)=t2e−tu(t)x(t) = t^2 e^{-t}u(t)x(t)=t2e−tu(t), compute the Laplace Transform and plot the spectra and pole-zero diagram.
8. Determine the Laplace Transform of x(t)=e−t(cos2t+sin2t)u(t)x(t) = e^{-t} (\cos 2t + \sin 2t) u(t)x(t)=e−t(cos2t+sin2t)u(t) and plot magnitude, phase, and S-plane poles/zeros.
9. Given x(t)=e−0.5tcos(3t)u(t)x(t) = e^{-0.5t} \cos(3t) u(t)x(t)=e−0.5tcos(3t)u(t), compute the Laplace Transform and plot the magnitude & phase spectrum and pole-zero diagram.
10. For x(t)=te−2tsin(3t)u(t)x(t) = t e^{-2t} \sin(3t) u(t)x(t)=te−2tsin(3t)u(t), compute Laplace Transform using properties and plot the corresponding spectra and poles/zeros.

11. Find the Laplace Transform of the unit step response x(t)=u(t)x(t) = u(t)x(t)=u(t), plot magnitude, phase, and S-plane diagram.
12. Compute the Laplace Transform of x(t)=tu(t)x(t) = t u(t)x(t)=tu(t) and plot magnitude and phase spectra. Show poles and zeros.
13. For x(t)=t2u(t)x(t) = t^2 u(t)x(t)=t2u(t), determine its Laplace Transform and plot magnitude & phase spectrum. Draw the S-plane diagram.
14. Given x(t)=δ(t)+e−tu(t)x(t) = \delta(t) + e^{-t}u(t)x(t)=δ(t)+e−tu(t), compute its Laplace Transform, plot magnitude & phase, and pole-zero diagram.
15. Find Laplace Transform of x(t)=u(t)−u(t−2)x(t) = u(t) - u(t-2)x(t)=u(t)−u(t−2) (a rectangular pulse) and plot magnitude, phase, and pole-zero diagram.

16. Determine the Laplace Transform of x(t)=e−t+sin(2t)u(t)x(t) = e^{-t} + \sin(2t)u(t)x(t)=e−t+sin(2t)u(t), plot magnitude & phase spectrum, and S-plane diagram.
17. For x(t)=e−tcos(2t)+te−2tu(t)x(t) = e^{-t} \cos(2t) + t e^{-2t} u(t)x(t)=e−tcos(2t)+te−2tu(t), compute Laplace Transform and plot spectra & pole-zero diagram.

18. Compute Laplace Transform for x(t)=(1+t)e−tu(t)x(t) = (1 + t)e^{-t} u(t)x(t)=(1+t)e−tu(t) and plot magnitude, phase, and S-plane poles/zeros.
19. Given x(t)=e−t−e−3tx(t) = e^{-t} - e^{-3t}x(t)=e−t−e−3t, find Laplace Transform, magnitude & phase spectrum, and plot poles and zeros.
20. For x(t)=cos(3t)−sin(2t)e−tx(t) = \cos(3t) - \sin(2t) e^{-t}x(t)=cos(3t)−sin(2t)e−t, determine Laplace Transform, plot magnitude and phase spectrum, and S-plane poles/zeros.

Real Time Applications:

Fourier transforms having following applications

- Analysis of differential equations
- Fourier transform spectroscopy
- Quantum mechanics
- Signal processing applications etc

EXPERMENT NO: 8

LOCATING THE ZEROS AND POLES AND PLOTTING THE POLE ZERO MAPS IN S-PLANE AND Z-PLANE FOR THE GIVEN TRANSFER FUNCTION.

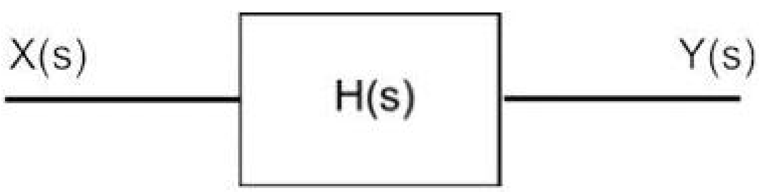
AIM: -
To locating the zeros and poles and plotting the pole zero maps in s-plane and z-plane for the given transfer function.

SOFTWARE REQURIED:-
1.MATLAB R2025a.
2.Windows XP SP2.

THEORY:-
A Transfer Function is the ratio of the output of a system to the input of a system, in the Laplace domain considering its initial conditions to be zero. If we have an input function of X(s), and an output function Y(s), we define the transfer function H(s) to be:

$$H(s) = \frac{Y(s)}{X(s)}$$

Transfer function is the Laplace transform of a system's impulse response.



Given a continuous-time transfer function in the Laplace domain, *H*(*s*) or a discrete-time one in the Z-domain, *H*(*z*), a zero is any value of *s* or *z* such that the transfer function is zero, and a pole is any value of *s* or *z* such that the transfer function is infinite.

***Zeros*:**1. The value(s) for *z* where the *numerator* of the transfer function equals zero
2. The complex frequencies that make the overall gain of the filter transfer function zero.

Poles: 1. The value(s) for *z* where the *denominator* of the transfer function equals zero
2. The complex frequencies that make the overall gain of the filter transfer functioninfinite.

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.

Bilateral Z-transform

The bilateral or two-sided Z-transform of a discrete-time signal x[n] is the function X(z) defined as

$$X(z) = \mathcal{Z}\{x[n]\} = \sum_{n=-\infty}^{\infty} x[n] z^{-n}$$

Unilateral Z-transform

Alternatively, in cases where x[n] is defined only for n ≥ 0, the single-sided or unilateral Z-transform is defined as

$$X(z) = \mathcal{Z}\{x[n]\} = \sum_{n=0}^{\infty} x[n] z^{-n}$$

In signal processing, this definition is used when the signal is causal.

where $z = r.e^{j\omega}$

$$X(z) = \frac{P(z)}{Q(z)}$$

The roots of the equation P(z) = 0 correspond to the 'zeros' of X(z)
The roots of the equation Q(z) = 0 correspond to the 'poles' of X(z)

PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

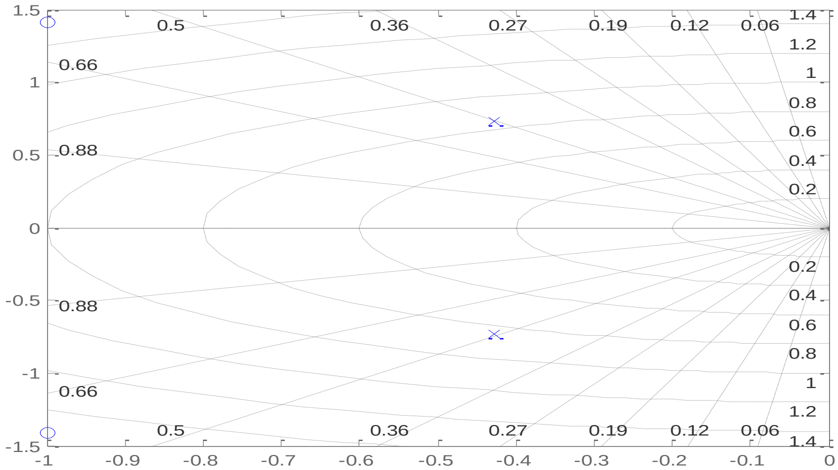
PROGRAM:-

```
%locating poles of zero on s-plane%
clc;
clear all;
close all;
num=input('enter numerator co-efficients');
den=input('enter denominator co-efficients');
h=tf(num,den);
poles=roots(den);
zeros=roots(num);
sgrid;
pzmap(h);
grid on;
title('locating poles of zeros on s-plane');

%locating poles &zeros on z-plane%

clc;
clear all;
close all;
num=input('enter numerator coefficient');
den=input('enter denominator coefficient');
p=roots(den);
z=roots(num);
```

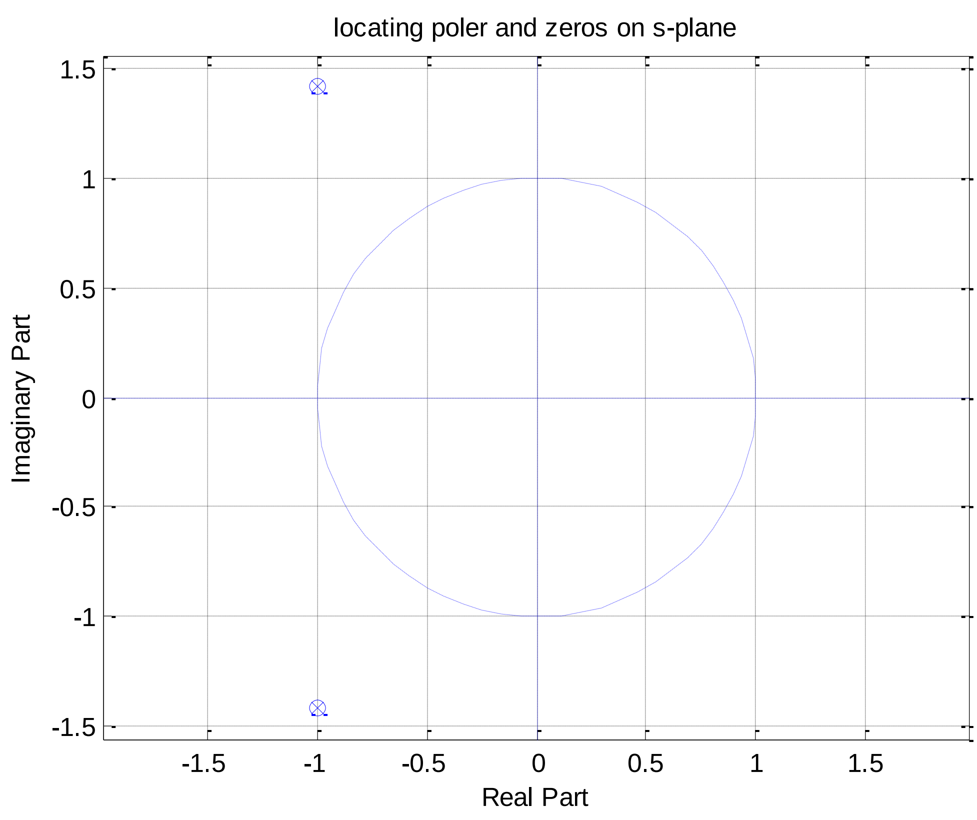
```
zplane(p,z);
grid;
title('locating poler and zeros on s-plane');
```

OUTPUT:-

locating poles of zero on s-plane
enter numerator coefficient[1 2 3]
enter denominator coefficient[7 6 5]



locating poles &zeros on z-plane
enter numerator coefficient[1 2 3]
enter denominator coefficient[1 2 3]



RESULT: -

In this experiment the zeros and poles and plotting the pole zero maps in s-plane and z-plane for the given transfer function using MATLAB.

VIVA QUESTIONS: -

1.study the details of ztrans() and iztrans() functions?
2.what are poles and zeros?
3.how you specify the stability based on poles and zeros?
4.define s-plane and z-plane?
5.what is the difference b/w s-plane and z-plane?
6.bilateral z-transform
7.z-transforms
8.unilateral z-transforms
9.what is pole?
10.what is zero?
11.what is splane?
12.what is zplane?
13.what is difference between ploe and zero?
14.what is subplot?
15.what is subplot(4,5,6)?
16.define s plane?
17.define z plane?
18.define transform function?
19.what is plot?
20.what is stem?
21.what is difference between plot and stem?
22.what is subplot(2 2 2)?
23.what is system bandwidth?
24what is signal bandwidth
25.what os signal?
26.what is orthogonality function?
27.what are the classifications of sgnals?
22.what are poles and zeros?
29.how you specify the stability based on poles and zeros?
30.Define s-plane and z-plane?
31.what is the difference b/w s-plane and z-plane?
32. what is subplot?
33.define transform function?
34.what is plot?
35.what is stem?
36.what is difference between plot and stem?
37.what is fourier transorm?
38.what is system?
39.what is signal?
40.what is trignometric series?
41.application of convolution?
42.applications of lti system?
43.what is stable system?
44.define laplace-transform?
45. what is the condition for convergence of the l.t?
46. what is the region of convergence(roc)?

47. state the shifting property of l.t?
48. state convolution property of l.t?
49. Define i.l.t?
50. applications of lti system

EXERCISE PROGRAMS

1. Write a MATLAB program to find the impulse response of the following difference equation3y(n)-5y(n-1)+4y(n-2)=x(n)-2x(n-1).
2. Write a MATLAB program to find the impulse response of the following difference equation 6y(n)-9y(n-1)-4y(n-2)=x(n)-5x(n-1).
3. Write a MATLAB program to find the impulse response of the following difference equation8y(n)-6y(n-1)+4y(n-2)=x(n)+2x(n-1).
4. Write a MATLAB program to find the impulse response of the following difference equation 3y(n)+5y(n-1)+9y(n-2)=x(n)-9x(n-1).
5. Write a MATLAB program to find the impulse response of the following difference equation6y(n)-5y(n-1)+4y(n-2)=x(n)-2x(n-1).
6. Write a MATLAB program to find the impulse response of the following difference equation7y(n)-5y(n-1)+4y(n-2)=x(n)-9x(n-1).
7. Write a MATLAB program to find the impulse response of the following difference equation3y(n)-5y(n-1)+4y(n-2)=x(n)-6x(n-1).
8. Write a MATLAB program to find the impulse response of the following difference equation 5y(n)-5y(n-1)+5y(n-2)=x(n)-2x(n-1).
9. Write a MATLAB program to find the impulse response of the following difference equation3y(n)-6 y(n-1)+4y(n-2)=x(n)-7x(n-1).
10. Write a MATLAB program to find the impulse response of the following difference equation2y(n)-5y(n-1)+4y(n-2)=x(n)-2x(n-1).
11. Write a MATLAB program to find the impulse response of the following difference equation 3y(n)-5y(n-1)+4y(n-2)=5x(n)-6x(n-1).
12. Write a MATLAB program to find the impulse response of the following difference equation6y(n)-8y(n-1)-4y(n-2)=7x(n)-5x(n-1).
13. Write a MATLAB program to find the impulse response of the following difference equation8y(n)-8y(n-1)+9y(n-2)=x(n)+5x(n-1).
14. Write a MATLAB program to find the impulse response of the following difference equation 3y(n)+9y(n-1)+9y(n-2)=9x(n)-5x(n-1).
15. Write a MATLAB program to find the impulse response of the following difference equation6y(n)-5y(n-1)+4y(n-2)=7x(n)-2x(n-1).
16. Write a MATLAB program to find the impulse response of the following difference equation 7y(n)-5y(n-1)+4y(n-2)=x(n)-7x(n-1).
17. Write a MATLAB program to find the impulse response of the following difference equation 3y(n)-5y(n-1)+4y(n-2)=x(n)-6x(n-1).
18. Write a MATLAB program to find the impulse response of the following difference equation 5y(n)-5y(n-1)+5y(n-2)=7x(n)-2x(n-1).
19. Write a MATLAB program to find the impulse response of the following difference equation3y(n)-6 y(n-1)+4y(n-2)=4x(n)-x(n-1).
20. Write a MATLAB program to find the impulse response of the following difference equation2y(n)-5y(n-1)+4y(n-2)=x(n)-2x(n-1).

Real Time Applications:

Z-Transforms having following applications

- Analysis of differential equations
- Z- transform spectroscopy
- Quantum mechanics
- Signal processing applications etc
- Conversion of analog signals to digital signals

EXPERMENT NO: 9

Design a Simulink or equivalent model

AIM: -
Design a Simulink or equivalent model CODE for MATLAB a) Solving Differential Equations b) Finding the response of any RLC Circuit with different initial Conditions for AC and DC inputs and plot the corresponding responses.

SOFTWARE REQURIED:-

1. MATLAB R2025a.
2.Windows XP SP2.

THEORY:-

- A **control system** (PID, state-space, etc.)
- A **mechanical system** (mass-spring-damper, robotics)
- An **electrical system** (RLC circuit, power electronics)
- A **chemical or process system** (reaction kinetics, temperature control)

Sampling Theorem:
The theorem shows that a band limited analog signal that has been sampled can be perfectly reconstructed from an infinite sequence of samples if the sampling rate exceeds 2B samples per second, where B is the highest frequency in the original signal.
If a signal contains a component at exactly B hertz, then samples spaced at exactly 1/(2B) seconds do not completely determine the signal, Shannon's statement notwithstanding.

Proof: Let g(t) be a bandlimited signal whose bandwidth is fm
(wm = 2πfm).

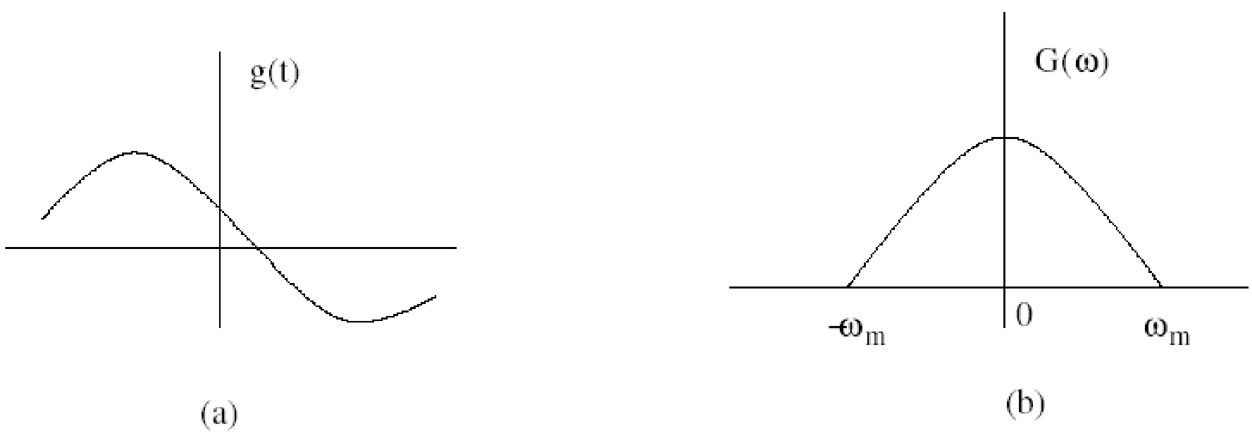


Figure 2: (a) Original signal g(t) (b) Spectrum G(w)
δ(t) is the sampling signal with fs = 1/T > 2fm.

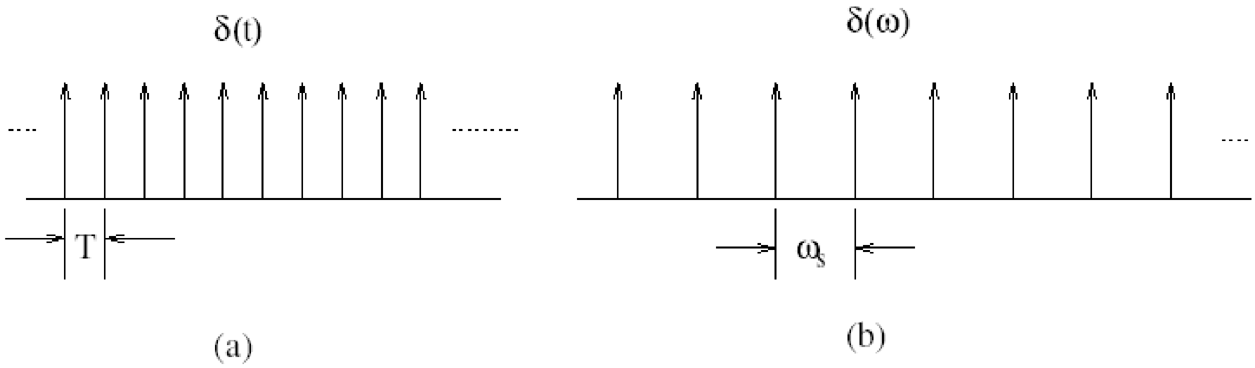


Figure 3: (a) sampling signal δ(t)) (b) Spectrum δ(w)

Let gs(t) be the sampled signal. Its Fourier Transform Gs(w) isgiven by

$$
\begin{aligned}
\mathcal{F}(g_s(t)) &= \mathcal{F}\left[g(t)\delta_T(t)\right] \\
&= \mathcal{F}\left[g(t)\sum_{n=-\infty}^{+\infty}\delta(t-nT)\right] \\
&= \frac{1}{2\pi}\left[G(\omega)*\omega_0\sum_{n=-\infty}^{+\infty}\delta(\omega-n\omega_0)\right] \\
G_s(\omega) &= \frac{1}{T}\sum_{n=-\infty}^{+\infty}G(\omega)*\delta(\omega-n\omega_0) \\
G_s(\omega) &= \mathcal{F}\left[g(t)+2g(t)\cos(\omega_0 t)+2g(t)\cos(2\omega_0 t)+\cdots\right] \\
G_s(\omega) &= \frac{1}{T}\sum_{n=-\infty}^{+\infty}G(\omega-n\omega_0)
\end{aligned}
$$

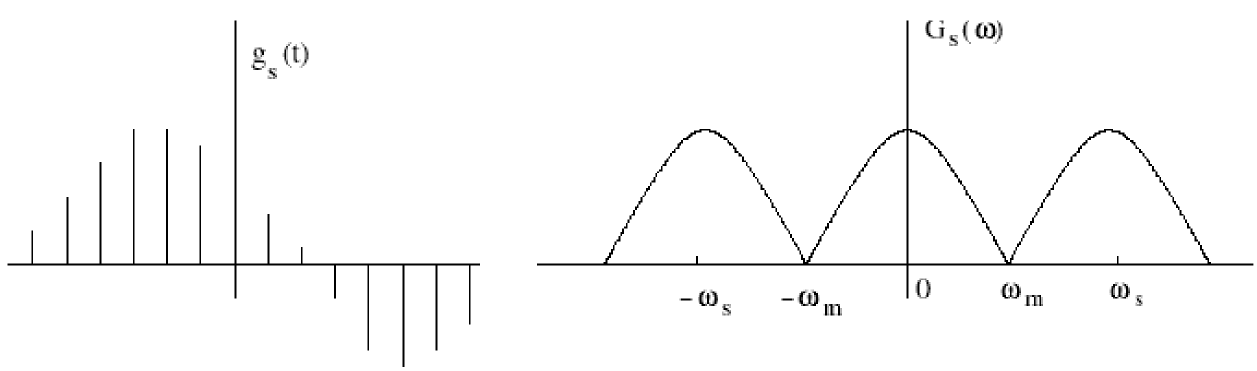


Figure 4: (a) sampled signal gs(t) (b) Spectrum Gs(w)

If $\omega_s = 2\omega_m$, i.e., $T = 1/2f_m$. Therefore, $G_s(\omega)$ is given by

$$G_s(\omega) = \frac{1}{T} \sum_{n=-\infty}^{+\infty} G(\omega - n\omega_m)$$

To recover the original signal G(w):
1. Filter with a Gate function, H2wm(w) of width 2wm
Scale it by T.

$$G(\omega) = TG_s(\omega)H_{2\omega_m}(\omega).$$

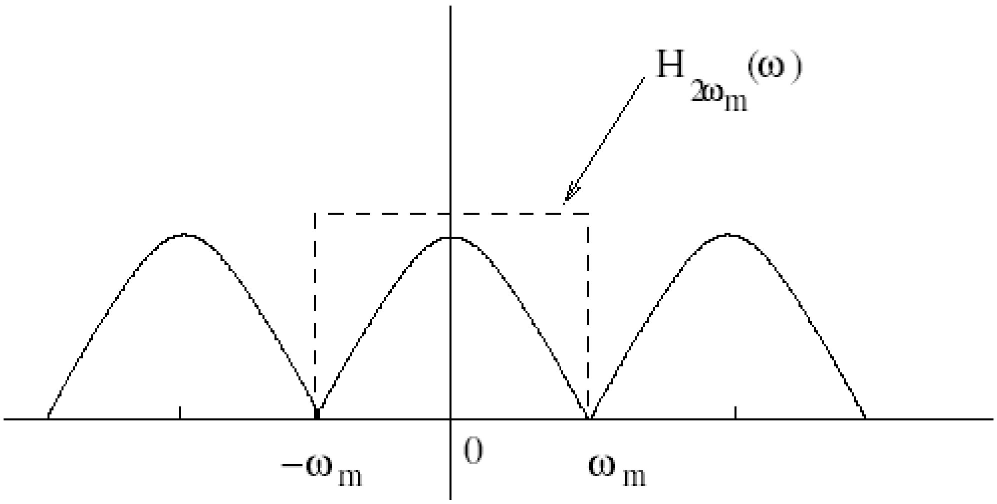


Figure 5: Recovery of signal by filtering with a fiter of width 2wm

Aliasing
{ Aliasing is a phenomenon where the high frequency components of the sampled signalinterfere with each other because of inadequate sampling ws < 2wm.

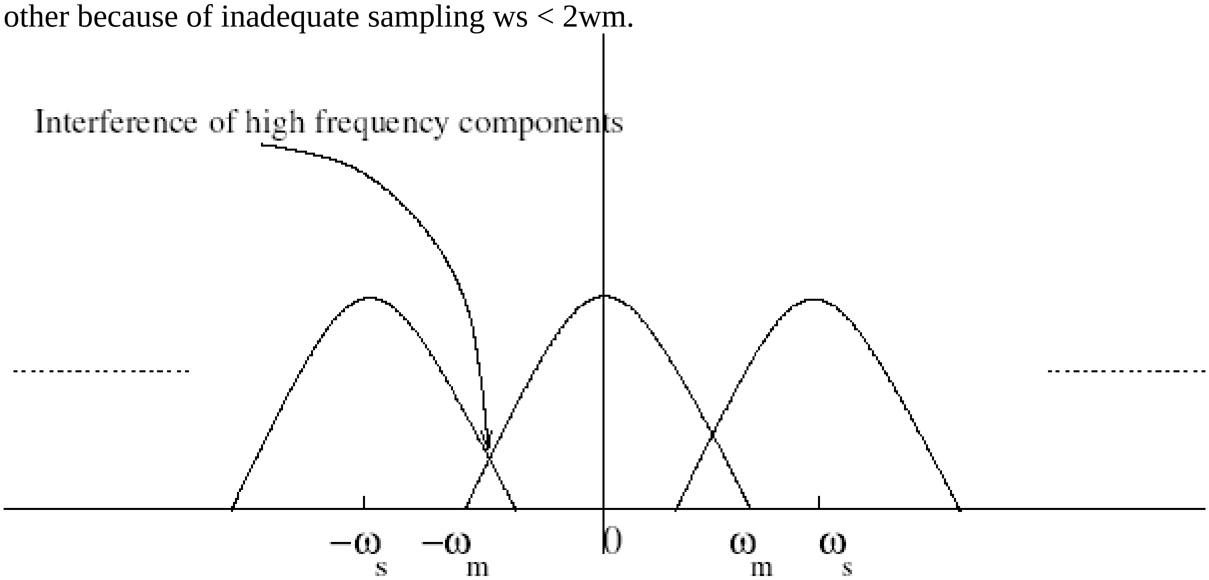


Figure 6: Aliasing due to inadequate sampling

Aliasing leads to distortion in recovered signal. This is the reason why sampling frequency should be at least twice the bandwidth of the signal.

PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

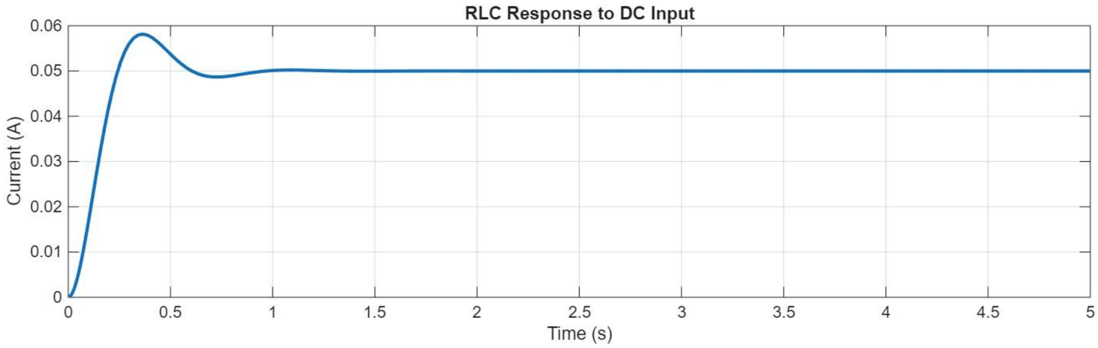
PROGRAM:-

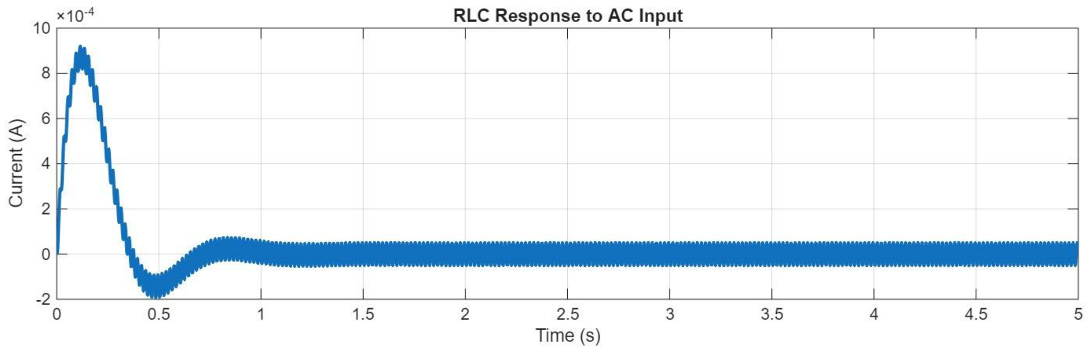
```
clc; clear; close all;

% Define the differential equation as a function
f = @(t,y) -2*y + 3;

% Time span
tspan = [0 10];

% Initial condition
y0 = 1;

% Solve using ode45
[t,y] = ode45(f, tspan, y0);

% Plot
figure;
plot(t,y,'LineWidth',2);
grid on;
xlabel('Time (t)');
ylabel('y(t)');
title('Solution of dy/dt = -2y + 3');
```

```
clc; clear; close all;

% Parameters
R = 10;     % Ohms
L = 1;      % Henry
C = 0.01;   % Farad

% Initial conditions
i0 = 0;     % initial current
di0 = 0;    % initial derivative

x0 = [i0; di0];

% Time span
tspan = [0 5];

% DC Input
```

```
Vdc = @(t) 5;

% AC Input
Vac = @(t) 5*sin(2*pi*50*t);

% RLC Model
rlc = @(t,x,V) [x(2);
            (1/L)*(V(t) - R*x(2) - (1/C)*x(1))];

% Solve for DC
[t1,x1] = ode45(@(t,x) rlc(t,x,Vdc), tspan, x0);

% Solve for AC
[t2,x2] = ode45(@(t,x) rlc(t,x,Vac), tspan, x0);

% Plot Results
figure;

subplot(2,1,1);
plot(t1, x1(:,1),'LineWidth',2);
grid on;
title('RLC Response to DC Input');
xlabel('Time (s)');
ylabel('Current (A)');

subplot(2,1,2);
plot(t2, x2(:,1),'LineWidth',2);
grid on;
title('RLC Response to AC Input');
xlabel('Time (s)');
ylabel('Current (A)');
```

OUTPUT:-





RESULT:-

In this experiment the sampling theorem have been verified using MATLAB .

VIVA QUESTIONS:-

1. What is Simulink, and how does it differ from MATLAB?
2. Explain the difference between a **continuous** and **discrete** system in Simulink.
3. What are the main types of blocks in Simulink?
4. What is a **Signal** in Simulink, and how is it different from a **Parameter**?
5. Explain the purpose of **Subsystem blocks** and when to use them.

6. How do you simulate a model in Simulink, and what are the steps involved?
7. What is a **Solver**, and why is it important in simulation?
8. Explain the difference between **fixed-step** and **variable-step** solvers.
9. How can you model a **feedback control system** in Simulink?
10. What is the use of **Scope blocks**, and how do they differ from **Display blocks**?

11. What are **Stateflow charts**, and when are they used?
12. How can you implement a **PID controller** in Simulink?
13. Explain the concept of **zero-crossing detection** in Simulink.
14. What is the difference between **Model Reference** and **Library Link blocks**?
15. How do you perform **parameter tuning** during simulation?

16. How can you import **real-world signals** into Simulink?
17. Describe the steps to generate **C code** from a Simulink model.
18. How would you model a **DC motor system** in Simulink?
19. How do you debug a Simulink model that is **not simulating correctly**?
20. What are some common **applications of Simulink in industry**?

EXERCISE PROGRAM:-

Absolutely! Here's a list of **20 MATLAB/Simulink exercise questions** that range from beginner to intermediate levels. They focus on modelling, simulation, and control systems. I've included a mix of tasks suitable for **Simulink** or MATLAB's **equivalent script-based modeling**.

1. Create a Simulink model of a **simple RC circuit** (Resistor-Capacitor) and plot the voltage across the capacitor for a step input.
2. Model a **mass-spring-damper system** in Simulink and simulate its response to a step force input.
3. Design a **first-order transfer function** in Simulink and simulate its response to a sine wave input.
4. Build a Simulink model to **add two signals** (sine and cosine waves) and display the result on a Scope.
5. Implement a **discrete-time integrator** using Simulink and compare its output with MATLAB's cumsum function.

6. Design a **PID controller** in Simulink for a DC motor and simulate its step response.
7. Create a **closed-loop feedback system** using a second-order plant (e.g., G(s) = 1/(s^2 + 3s + 2)) and simulate unit step response.
8. Model a **lead-lag compensator** in Simulink and test it with a step input.
9. Simulate the **root locus** of a transfer function in MATLAB and verify it by building the corresponding Simulink model.
10. Design a **state-space model** in Simulink for a 2-state system and simulate its response to an input vector.

11. Use Simulink to **generate a sinusoidal signal**, add Gaussian noise, and apply a low-pass filter.
12. Build a **discrete-time FIR filter** in Simulink and compare its output to MATLAB's filter() function.
13. Simulate **sampling and reconstruction** of a continuous-time signal in Simulink.
14. Create a model to **compute the FFT** of a signal and display its magnitude spectrum.
15. Model a **signal modulator/demodulator** (AM or FM) in Simulink.

16. Build a **multi-input multi-output (MIMO) system** in Simulink and simulate its step response.
17. Model a **buck or boost DC-DC converter** and simulate the output voltage under varying load conditions.
18. Implement a **state observer (Luenberger observer)** in Simulink and compare estimated states with actual states.
19. Simulate a **temperature control system** with feedback in Simulink.
20. Build a **vehicle cruise control system** in Simulink and test it for varying road slopes and set speeds.

REAL TIME APPLICATIONS:

- Industrial control and automation (Control the velocity or position of an object)
- Examples: Controlling the position of a valve or shaft of a motor
- Examples: Digital processing applications
- TV reception
- RADAR
- SONAR

EXPERMENT NO: 10

Gibbs Phenomenon and waveform synthesis

AIM: -

To demonstrate Gibbs Phenomenon using MATLAB.

SOFTWARE REQURIED:-

1.MATLAB R2025a.
2.Windows XP2SP2.

THEORY:-

Gibbs Phenomenon:

The peculiar manner in which the Fourier series of a piecewise continuously differentiable periodic function behaves at a jump discontinuity: the nth partial sum of the Fourier series has large oscillations near the jump, which might increase the maximum of the partial sum above that of the function itself. The overshoot does not die out as the frequency increases, but approaches a finite limit.

The Gibbs phenomenon involves both the fact that Fourier sums overshoot at a jump discontinuity, and that this overshoot does not die out as the frequency increases.

The best known version of the Gibbs phenomenon is the overshoot that arises when a discontinuous function is represented by a truncated set of Fourier expansion terms. The situation is similar if the truncated Fourier expansion is instead obtained by means of interpolation on an equispaced grid.

PROCEDURE: -

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

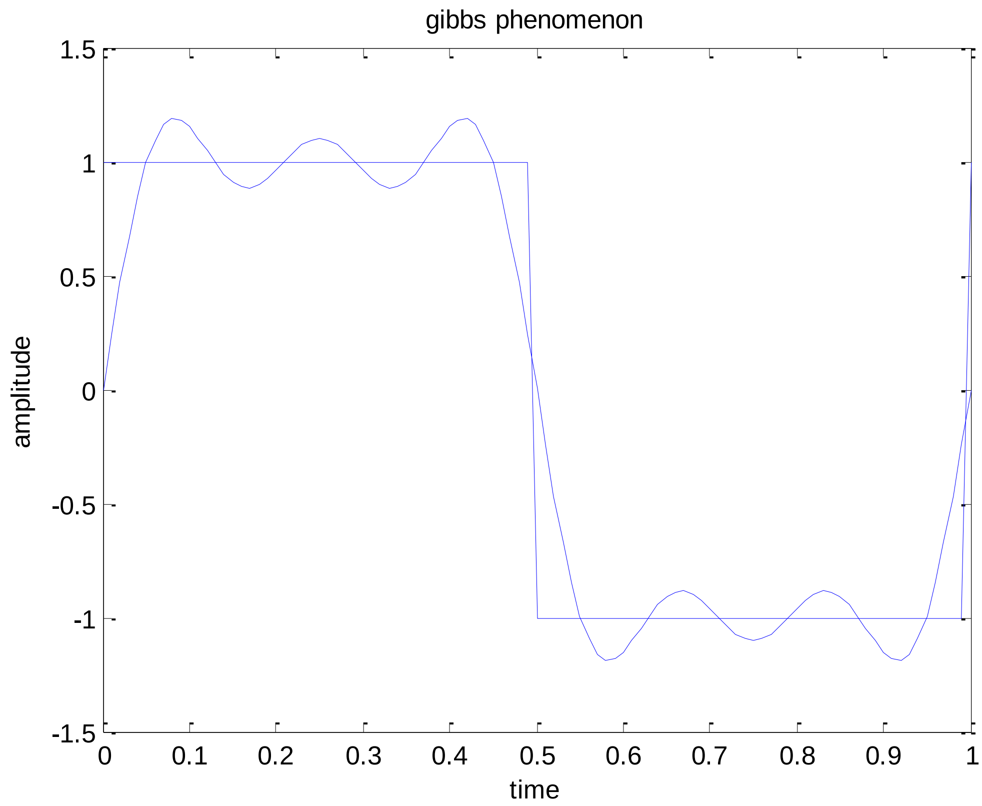
PROGRAM: -

```
%Gibbs phenomenon%
clc;
clear all;
close all;
t=0:0.01:1;
N=input('enter no of harmonics');
x=square(2*pi*t);
plot(t,x);
hold on;
gp=0;
for n=1:2:N;
   gp=gp+(4/(n*pi))*sin(2*pi*n*t);
end;
plot(t,gp);
hold off;
xlabel('time');
ylabel('amplitude');
title('gibbs phenomenon');
```

OUTPUT: -

Gibbs phenomenon

enter no of harmonics 5



RESULT: -

In this experiment Gibbs phenomenon have been demonstrated Using MATLAB .

EXPERIMENT – 11

Generate Random Variables with Different CDFs/PDFs

AIM: -
Write a Matlab code for Generate Random Variables with Different CDFs/PDFs
SOFTWARE REQURIED:-
1.MATLAB R2025a.
2.Windows XP SP2.
THEORY:-
1. Random Variables
A random variable (RV) is a function that assigns a real number to each outcome of a random experiment. It can be:
- Discrete: Takes countable values (e.g., number of heads in 3 coin flips).
- Continuous: Takes values in a continuum (e.g., time until a light bulb burns out).

2. CDF (Cumulative Distribution Function)
The CDF of a random variable XXX is defined as:
FX(x)=P(X≤x)F_X(x) = P(X \le x)FX(x)=P(X≤x)
Properties:
1. FX(x)F_X(x)FX(x) is non-decreasing.
2. lim x→−∞FX(x)=0\lim_{x \to -\infty} F_X(x) = 0limx→−∞FX(x)=0
3. lim x→+∞FX(x)=1\lim_{x \to +\infty} F_X(x) = 1limx→+∞FX(x)=1
4. For continuous RVs: FX(x)F_X(x)FX(x) is continuous, but for discrete RVs, it can have jumps.

3. PDF (Probability Density Function)
Only for continuous RVs, the PDF fX(x)f_X(x)fX(x) is defined as:
fX(x)=ddxFX(x)f_X(x) = \frac{d}{dx} F_X(x)fX(x)=dxdFX(x)
Properties:
1. fX(x)≥0f_X(x) \ge 0fX(x)≥0 for all xxx
2. ∫−∞∞fX(x) dx=1\int_{-\infty}^{\infty} f_X(x) \, dx = 1∫−∞∞fX(x)dx=1
3. FX(x)=∫−∞xfX(t) dtF_X(x) = \int_{-\infty}^{x} f_X(t) \, dtFX(x)=∫−∞xfX(t)dt

For discrete RVs, we talk about PMF (Probability Mass Function) instead of PDF:
P(X=xi)=piP(X=x_i) = p_iP(X=xi)=pi
4. Different RVs ⇒ Different CDFs and PDFs

If XXX and YYY are random variables and their distributions are different, then:
FX(x)≠FY(x)for some xF_X(x) \neq F_Y(x) \quad \text{for some } xFX(x)=FY(x)for some x
and/or
fX(x)≠fY(x)for continuous RVsf_X(x) \neq f_Y(x) \quad \text{for continuous RVs}fX(x)=fY(x)for continuous RVs

Examples:

Example 1: Continuous RVs
- X∼Uniform(0,1)X \sim \text{Uniform}(0,1)X∼Uniform(0,1)

fX(x)=1,0≤x≤1f_X(x) = 1, \quad 0 \le x \le 1fX(x)=1,0≤x≤1
- Y∼Exponential(λ=1)Y \sim \text{Exponential}(\lambda=1)Y∼Exponential(λ=1)

fY(y)=e−y,y≥0f_Y(y) = e^{-y}, \quad y \ge 0fY(y)=e−y,y≥0
PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
clc;
clear;
close all;
N = 10000;
% Uniform Distribution
u = rand(N,1);
% Normal Distribution
n = randn(N,1);
% Exponential Distribution
e = exprnd(1,N,1);
figure
subplot(3,2,1)
histogram(u,50,&#39;Normalization&#39;,&#39;pdf&#39;)
title(&#39;Uniform PDF&#39;)
subplot(3,2,2)
cdfplot(u)
title(&#39;Uniform CDF&#39;)
subplot(3,2,3)
histogram(n,50,&#39;Normalization&#39;,&#39;pdf&#39;)
title(&#39;Normal PDF&#39;)

subplot(3,2,4)
cdfplot(n)
title(&#39;Normal CDF&#39;)
subplot(3,2,5)
histogram(e,50,&#39;Normalization&#39;,&#39;pdf&#39;)
title(&#39;Exponential PDF&#39;)
subplot(3,2,6)
cdfplot(e)
title(&#39;Exponential CDF&#39;)
```

OUTPUT:-

RESULT:-
Hence the Generate Random Variables with Different CDFs/PDFssuccessfully completed by using MATLAB

VIVA QUESTIONS:-

1. Define a random variable. How are discrete and continuous random variables different?
2. What is a probability density function (PDF)? How does it differ from a cumulative distribution function (CDF)?
3. Explain the relationship between PDF and CDF for a continuous random variable.
4. Can a PDF be greater than 1? Justify your answer.

MS LAB Manual Dept.of ECE

95

5. Explain the properties that any CDF must satisfy.
6. Given a CDF ($F_X(x) = 1 - e^{-2x}, x \ge 0$), find the corresponding PDF.
7. Find the probability ($P(1 \le X \le 3)$) for a continuous random variable with PDF ($f_X(x) = 3x^2, 0 \le x \le 1$).
8. Determine the CDF for a uniform random variable (X) in the interval ($[a, b]$).
9. If a random variable has PDF ($f_X(x) = \lambda e^{-\lambda x}, x \ge 0$), identify the type of distribution and its mean.
10. For a discrete random variable (X) with PMF ($P(X=n) = (1/2)^n$), ($n = 1, 2, 3, \dots$), find the CDF.
11. A CDF jumps at certain points. What kind of random variable does it indicate?

12. Explain why the total area under a continuous PDF must equal 1.
13. How can you use the CDF to generate random numbers for simulation?
14. Compare the PDFs of exponential and uniform distributions. How do their CDFs reflect this difference?
15. How would you find the median of a continuous random variable using its CDF?
16. If (X) has PDF (f_X(x) = 2x, 0 \le x \le 1), find the PDF of (Y = X^2).
17. Derive the CDF of the sum of two independent uniform random variables (U(0,1)).
18. A continuous random variable has CDF (F_X(x) = x^2), (0 \le x \le 1). Find (P(0.25 \le X \le 0.75)).
19. Explain how a PDF can have multiple modes. Give an example of a bimodal distribution.
20. For a given CDF (F_X(x)), explain how to check if the corresponding PDF exists and is valid.

EXERCISE PROGRAMS

1. Write a MATLAB program to generate a uniform random variableX in [0,1] and plot its PDF and CDF.
2. Write a MATLAB program to generate a Gaussian random variableX ~ N(0,1) and plot its histogram, PDF, and CDF.
3. Write a MATLAB program to convolve two PDFs: X ~ Uniform(0,1) and Y ~ Uniform(0,1) to get the PDF of Z = X + Y.
4. Write a MATLAB program to simulate 50,000 samples of an exponential random variableλ = 2 and plot its empirical CDF.
5. Write a MATLAB program to generate a Poisson random variable with mean λ = 5 and plot its PMF.
6. Write a MATLAB program to compute the CDF of a triangular random variable with limits [0,1,2] and plot it.
7. Write a MATLAB program to simulate the sum of two independent Gaussian random variables and compare the empirical PDF with the theoretical PDF.
8. Write a MATLAB program to generate 1000 samples of a Bernoulli random variablep=0.3 and plot its PMF.
9. Write a MATLAB program to generate a discrete random variable with X = [1,2,3,4] and probabilities [0.1,0.2,0.3,0.4] and plot CDF.
10. Write a MATLAB program to compute the convolution of a uniform and an exponential PDF and plot the resulting PDF.
11. Write a MATLAB program to simulate a mixture of two Gaussian random variables and plot PDF and CDF.
12. Write a MATLAB program to calculate and plot the PDF of the maximum of two uniform random variables.
13. Write a MATLAB program to calculate and plot the PDF of the minimum of two independent exponential random variables.
14. Write a MATLAB program to generate a Rayleigh random variable and plot its PDF, CDF, and histogram.
15. Write a MATLAB program to generate a random variable from a Beta distributionBeta(2,5) and plot PDF and CDF.
16. Write a MATLAB program to simulate a random variable transformationY = sqrt(X) where X ~ Uniform(0,1) and plot PDF and CDF.
17. Write a MATLAB program to simulate 2D joint PDF of independent X ~ Uniform(0,1) and Y ~ Uniform(0,1) and plot 3D surface plot.
18. Write a MATLAB program to compute the PDF of the sum of a Gaussian and uniform random variable using convolution.
19. Write a MATLAB program to simulate a random walk of 1000 steps and plot the CDF of the final position.
20. Write a MATLAB program to estimate the expected value and variance of a random variable defined as X = 3*rand() + 2 using Monte Carlo simulation.

REAL TIME APPLICATIONS:

- Network packet arrivals
- Customer service
- Reliability engineering

EXPERIMENT-12

Generate Gaussian Noise and Find Mean, Skewness, Kurtosis, PDF and PSD

AIM: -

Write a Matlab code for Generate Gaussian Noise and Find Mean, Skewness, Kurtosis, PDF and PSD

SOFTWARE REQURIED: -

1.MATLAB R2025a.
2.Windows XP SP2.

THEORY: -

Definition:

Gaussian noise, also called **normal noise**, is a type of statistical noise with a probability density function (PDF) equal to that of the normal distribution. It is commonly used in communication systems, image processing, and signal processing to model random disturbances.

Characteristics:

- Random and unpredictable
- Amplitude follows a **Gaussian (normal) distribution**
- Completely described by its **mean (μ)** and **variance (σ²)**
- White Gaussian Noise (WGN) has constant power across all frequencies

Mathematical Representation:

A Gaussian random variable XXX has PDF:

fX(x)=12πσ2exp[−(x−μ)22σ2]f_X(x) = \frac{1}{\sqrt{2 \pi \sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]fX(x)=2πσ21exp[−2σ2(x−μ)2]

Where:

- μ\muμ = mean
- σ2\sigma^2σ2 = variance

2. Mean (Expectation)

The **mean** represents the average or expected value of the Gaussian noise:

μ=E[X]=∫−∞∞xfX(x)dx\mu = E[X] = \int_{-\infty}^{\infty} x f_X(x) dxμ=E[X]=∫−∞∞xfX(x)dx

For **Gaussian noise**, the mean is usually μ=0\mu = 0μ=0 (zero-mean) in signal processing applications.

3. Skewness

Skewness measures the asymmetry of the distribution:

Skewness=γ1=E[(X−μ)3]σ3\text{Skewness} = \gamma_1 = \frac{E[(X - \mu)^3]}{\sigma^3}Skewness=γ1 =σ3E[(X−μ)3]

- Gaussian noise has **zero skewness** because the distribution is symmetric about the mean.

4. Kurtosis

Kurtosis measures the "tailedness" of the distribution:

Kurtosis=γ2=E[(X−μ)4]σ4\text{Kurtosis} = \gamma_2 = \frac{E[(X - \mu)^4]}{\sigma^4}Kurtosis=γ2=σ4E[(X−μ)4]

- Gaussian noise has **kurtosis = 3** (mesokurtic).
- Sometimes **excess kurtosis** is used:

Excess Kurtosis=γ2−3=0\text{Excess Kurtosis} = \gamma_2 - 3 = 0Excess Kurtosis=γ2−3=0

5. Probability Density Function (PDF)

The **PDF** shows the likelihood of each amplitude value:

fX(x)=12πσ2exp[−(x−μ)22σ2]f_X(x) = \frac{1}{\sqrt{2 \pi \sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]fX(x)=2πσ21exp[−2σ2(x−μ)2]

Properties:

- Symmetric about mean μ\muμ
- Peak occurs at x=μx = \mux=μ
- Area under the curve = 1

6. Power Spectral Density (PSD)

The **PSD** shows how the power of a signal or noise is distributed over frequency.

For **white Gaussian noise (WGN):**

SX(f)=N02S_X(f) = \frac{N_0}{2}SX(f)=2N0

Where:

- N0N_0N0 = noise power spectral density (Watts/Hz)
- Flat PSD across all frequencies → “white” noise

For **colored Gaussian noise**, the PSD depends on frequency and is not constant.

Relation to autocorrelation:

SX(f)=F{RX(τ)}S_X(f) = \mathcal{F}\{R_X(\tau)\}SX(f)=F{RX(τ)}

Where RX(τ)R_X(\tau)RX(τ) is the autocorrelation function, and F\mathcal{F}F denotes Fourier transform.

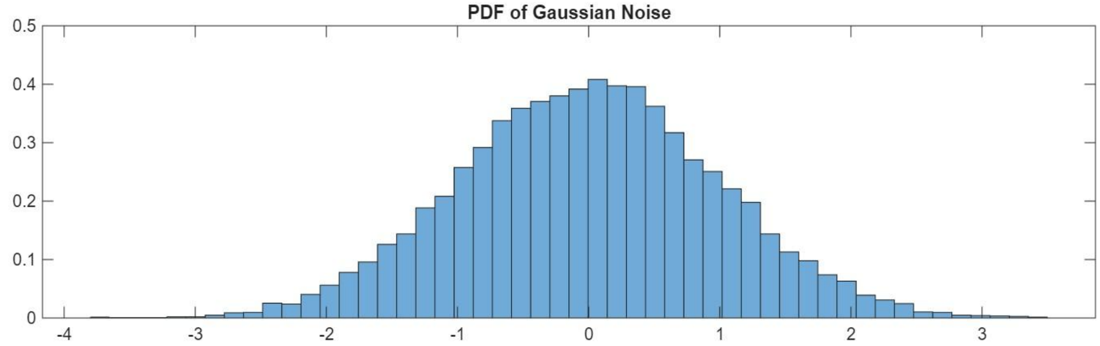
PROCEDURE:-

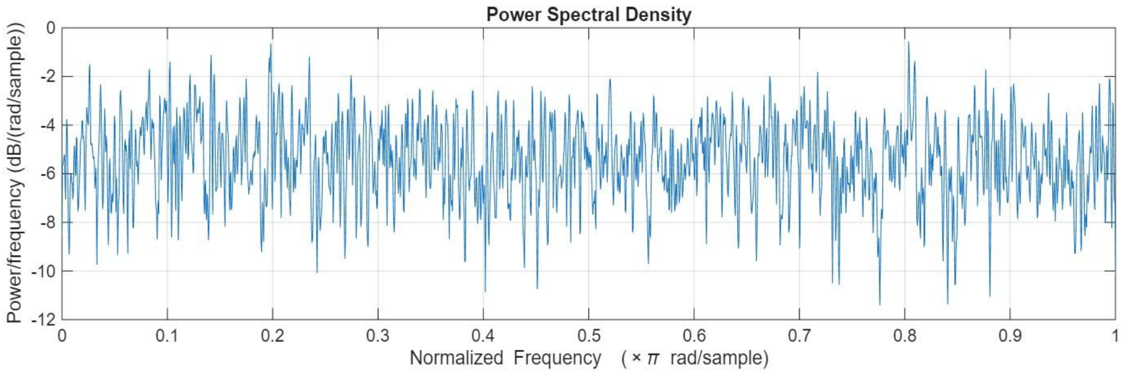
- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
clc;
clear;
close all;

N = 10000;

noise = randn(1,N);

mean_val = mean(noise);
skew_val = skewness(noise);
kurt_val = kurtosis(noise);

disp(['Mean = ', num2str(mean_val)])
disp(['Skewness = ', num2str(skew_val)])
disp(['Kurtosis = ', num2str(kurt_val)])

figure
subplot(2,1,1)
histogram(noise,50,'Normalization','pdf')
title('PDF of Gaussian Noise')

% PSD
subplot(2,1,2)
pwelch(noise)
title('Power Spectral Density')
```

OUTPUT:-





RESULT:-

Hence the Generate Gaussian Noise and Find Mean, Skewness, Kurtosis, PDF and PSD **successfully completed by using MATLAB**

VIVA QUESTIONS:-

1. What is Gaussian noise, and why is it called "Gaussian"?
2. What are the key properties of Gaussian noise?
3. How does Gaussian noise differ from uniform noise?
4. What is the probability density function (PDF) of Gaussian noise?
5. Why is Gaussian noise often assumed in communication systems and signal processing?
6. What is white Gaussian noise (WGN), and how does it differ from colored Gaussian noise?
7. How does Gaussian noise affect signals in time and frequency domains?

8. How do you calculate the **mean** of Gaussian noise, and what is its significance?
9. What is **skewness**, and what does a skewness of zero indicate for Gaussian noise?
10. Define **kurtosis**. What is the kurtosis of an ideal Gaussian distribution?
11. How can you interpret positive or negative kurtosis in a dataset?
12. Why are skewness and kurtosis useful in analyzing noise?

13. How do you estimate the PDF of Gaussian noise from sample data?
14. What does the shape of a Gaussian PDF tell you about the data?
15. How can you verify if a dataset follows Gaussian distribution using its PDF?

16. What is the Power Spectral Density (PSD) of Gaussian noise?

17. How is the PSD of white Gaussian noise different from colored noise?
18. Explain how you can estimate the PSD of Gaussian noise using FFT.
19. Why is PSD important in analyzing noise in communication systems?

20. How can Gaussian noise be generated in MATLAB or Python for simulation purposes?

that numerical set too?

EXERCISE PROGRAMS

1.Write a MATLAB program to generate a Gaussian noise sequence of length 1000 with zero mean and unit variance, then compute and display the **mean** and **variance** of the generated noise.

2.Write a MATLAB program to generate Gaussian noise of length 1000. Compute and display the **skewness** and **kurtosis** of the sequence. Comment on whether it is symmetric and mesokurtic.

3.Write a MATLAB program to generate Gaussian noise and plot its **Probability Density Function (PDF)** using histogram and normpdf. Verify whether it matches the theoretical Gaussian PDF.

4.Write a MATLAB program to generate Gaussian noise and compute its **Power Spectral Density (PSD)** using pwelch. Plot and interpret the flatness of the spectrum.

5.Write a MATLAB program to generate Gaussian noise sequences of lengths 100, 1000, and 10000. Compute the **mean, variance, skewness, and kurtosis** for each and comment on how sequence length affects statistical measures.

6.Write a MATLAB program to generate two Gaussian noise sequences, one with zero mean and one with mean Compute **mean, skewness, and kurtosis** for both sequences.

7.Write a MATLAB program to generate Gaussian noise sequences with standard deviations 1, 2, and 5. Plot the **PDF** of each sequence and comment on the spread of values.

8.Write a MATLAB program to generate Gaussian noise and plot the histogram with 20, 50, and 100 bins. Compute and display the **mean, skewness, and kurtosis** for each bin count.

9.Write a MATLAB program to generate Gaussian noise and artificially add a constant offset to the sequence. Compute the **skewness** before and after offset, and comment on symmetry.

10.Write a MATLAB program to generate Gaussian noise and add outliers to the sequence. Compute **kurtosis** before and after adding outliers. Comment on the effect of outliers on kurtosis.

11.Write a MATLAB program to generate Gaussian noise and plot **PDF using normpdf** and **histogram** on the

12.Write a MATLAB program to generate Gaussian noise and compute its **autocorrelation** using xcorr. Then, compute the **PSD** using fft and compare with pwelch.

13.Write a MATLAB program to generate two Gaussian noise sequences x1 and x2. Add them to form y = x1 + x2. Compute **mean, variance, skewness, and kurtosis** of x1, x2, and y, and comment on linearity of statistical measures.

14.Write a MATLAB program to generate Gaussian noise and pass it through a low-pass filter. Compute and plot the **PSD** before and after filtering.

15.Write a MATLAB program to generate Gaussian noise and normalize it to unit variance. Verify **mean, skewness, and kurtosis** after normalization.

16.Write a MATLAB program to generate Gaussian noise and uniform noise of the same length. Plot **histograms** and compute **mean, skewness, and kurtosis** for both sequences. Comment on the differences.

17.Write a MATLAB program to generate Gaussian noise of length 1,000,000. Compute **mean, skewness, kurtosis** and verify if they approximate the theoretical values of 0, 0, and 3.

18.Write a MATLAB program to generate 10 Gaussian noise sequences. Compute their **sum**, then calculate mean, skewness, and kurtosis. Comment on how the sum approaches a Gaussian distribution according to the central limit theorem.

19.Write a MATLAB program to generate 5 independent Gaussian noise sequences. Compute **PSD** for each and then compute the **average PSD**. Comment on the effect of averaging on spectral estimation.

20.Write a MATLAB program to generate Gaussian noise and scale it by 0.5, 2, and 5. Compute **mean, variance, skewness, and kurtosis** for each scaled sequence. Comment on which measures are affected by scaling.

REAL TIME APPLICATIONS:

- Network **packet arrivals**
- Customer **service**
- Reliability **engineering**

EXPERIMENT-13

Verify Sampling Theorem for Different Sampling Rates

AIM: -

Write a MATLAB code to Verify Sampling Theorem for Different Sampling Rates

SOFTWARE REQURIED:-

1.MATLAB R2025a.
2.Windows XP SP2.

THEORY:-

The **Sampling Theorem** states:

A continuous-time signal x(t)x(t)x(t) that is **bandlimited** to a maximum frequency fmf_mfm Hz can be perfectly reconstructed from its samples if it is sampled at a rate fsf_sfs greater than twice the maximum frequency:

fs>2fmf_s > 2 f_mfs>2fm

Here, fsf_sfs is the **sampling frequency**, and 2fm2 f_m2fm is called the **Nyquist rate**.

- **Bandlimited** means X(f)=0X(f) = 0X(f)=0 for ∣f∣>fm|f| > f_m∣f∣>fm
- **Nyquist rate:** fN=2fmf_N = 2 f_mfN=2fm

If fs≥2fmf_s \ge 2 f_mfs≥2fm, there is **no aliasing**, and the signal can be reconstructed perfectly using an **ideal low-pass filter**.

PROCEDURE:-

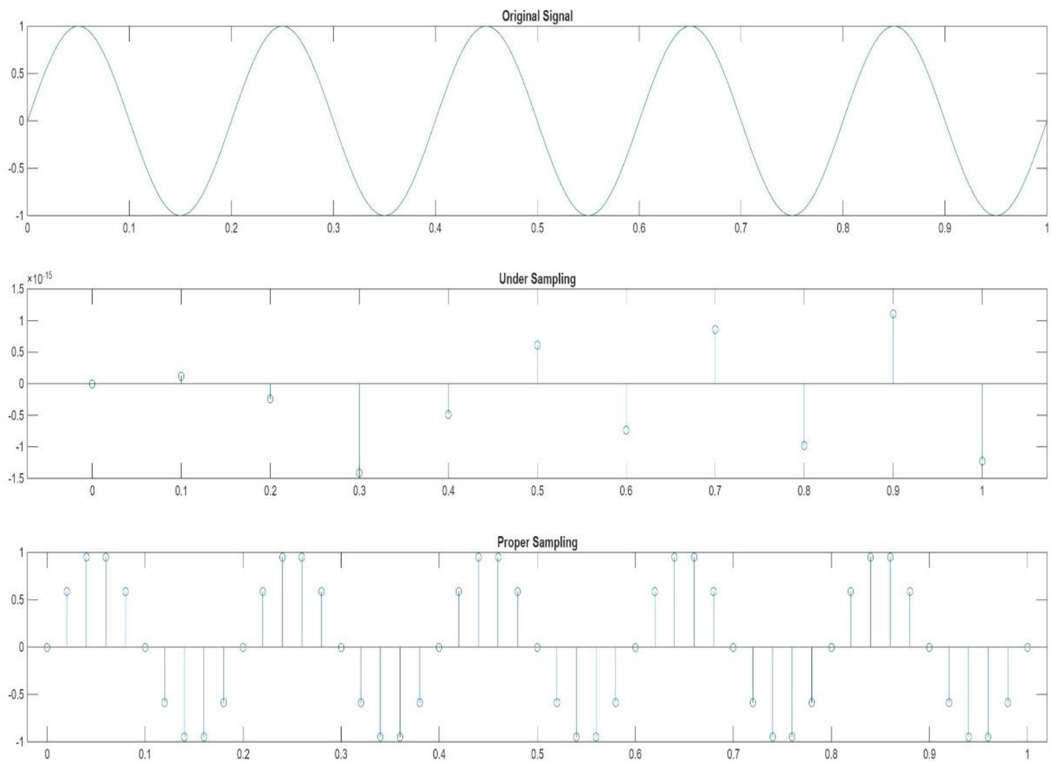
- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
clc;
clear;
close all;

t = 0:0.001:1;
f = 5;

x = sin(2*pi*f*t);

fs1 = 10;
ts1 = 0:1/fs1:1;
xs1 = sin(2*pi*f*ts1);
```

```
fs2 = 50;
ts2 = 0:1/fs2:1;
xs2 = sin(2*pi*f*ts2);

figure

subplot(3,1,1)
plot(t,x)
title('Original Signal')

subplot(3,1,2)
stem(ts1,xs1)
title('Under Sampling')

subplot(3,1,3)
stem(ts2,xs2)
title('Proper Sampling')
```

OUTPUT:-



RESULT:-

Hence the Verify Sampling Theorem for Different Sampling Rates **successfully completed by using MATLAB**

VIVA QUESTIONS:-

1. What is the Sampling Theorem (Nyquist-Shannon Theorem)?
2. What is meant by sampling of a signal?
3. Define Nyquist rate.
4. Define Nyquist interval.
5. What is the highest frequency component of a signal called?
6. Why is sampling required in digital signal processing?

7. What happens when a signal is sampled at the Nyquist rate?
8. What is oversampling?
9. What is undersampling?
10. What happens if sampling frequency is less than the Nyquist rate?
11. What are the advantages of oversampling?
12. Can a signal be perfectly reconstructed if sampled above Nyquist rate? ◇ Aliasing

13. What is aliasing?
14. Why does aliasing occur?
15. How can aliasing be prevented?
16. What is the role of an anti-aliasing filter?

17. State the condition for perfect reconstruction of a signal.
18. Express the relationship between sampling frequency and signal bandwidth.

19. What happens in frequency domain after sampling?
20. How does changing sampling rate affect the spectrum of the signal?

EXERCISE PROGRAMS

31. Write MATLAB command to generate a continuous-time sinusoidal signal for sampling analysis?
32. Write MATLAB command to sample a signal at a sampling rate equal to Nyquist rate?
33. Write MATLAB command to sample a signal at a rate higher than Nyquist rate (oversampling)?
34. Write MATLAB command to sample a signal at a rate lower than Nyquist rate (undersampling)?
35. Write MATLAB command to plot original and sampled signals on the same graph?
36. Write MATLAB command to demonstrate aliasing effect due to undersampling?
37. Write MATLAB command to reconstruct a signal using interpolation after sampling?
38. Write MATLAB command to calculate Nyquist rate for a given signal frequency?
39. Write MATLAB command to generate time vector for different sampling frequencies?
40. Write MATLAB command to compare signals sampled at two different rates?
41. Write MATLAB command to simulate sampling of a cosine signal with varying sampling rates?
42. Write MATLAB command to visualize frequency spectrum of sampled signal using FFT?
43. Write MATLAB command to show effect of sampling rate on signal reconstruction?
44. Write MATLAB command to generate a band-limited signal for sampling theorem verification?
45. Write MATLAB command to apply zero-order hold reconstruction to sampled signal?
46. Write MATLAB command to apply sinc interpolation for signal reconstruction?

47. Write MATLAB command to demonstrate folding frequency in undersampling case?
48. Write MATLAB command to vary sampling rate and observe aliasing graphically?
49. Write MATLAB command to generate discrete-time signal from continuous-time signal?
50. Write MATLAB command to compare original and reconstructed signal for different sampling rates?

.

REAL TIME APPLICATIONS:

- **Network packet arrivals**
- **Customer service**
- **Reliability engineering**

EXPERIMENT-14

Removal of Noise Using Cross Correlation

AIM: -

Write a Matlab code for Removal of Noise Using Cross Correlation

SOFTWARE REQURIED: -

1.MATLAB R2025a.
2.Windows XP SP2.

THEORY: -

Removal of Noise Using Cross-Correlation

Definition:
Cross-correlation is a mathematical technique used to measure the similarity between two signals as a function of the time-lag applied to one of them. It can help in detecting a known signal embedded in noise.

1. **Purpose**:
   - To extract a weak signal from a noisy background.
   - Commonly used in communication systems, radar, and biomedical signal processing.
2. **Principle**:
   - Suppose we have a received signal ($r(t) = s(t) + n(t)$), where ($s(t)$) is the desired signal and ($n(t)$) is noise.
   - Cross-correlate ($r(t)$) with a reference or known signal ($s(t)$).
   - The correlation function ($R_{rs}(\tau)$) will have a peak when the reference signal aligns with the desired signal in the noisy data.
3. **Mathematical Expression**:
   [
   $$R_{rs}(\tau) = \int r(t) \cdot s(t+\tau) , dt$$
   ]
   - ($R_{rs}(\tau)$) is the cross-correlation function.
   - (τ) is the time lag.
   - The peak in ($R_{rs}(\tau)$) indicates the presence of the signal ($s(t)$) in ($r(t)$).
4. **Advantages**:
   - Enhances signal detection in low signal-to-noise ratio (SNR) conditions.
   - Non-invasive and simple computationally.

Applications:

- Radar and sonar signal detection.
- ECG or EEG signal extraction.
- Digital communication systems for demodulation.

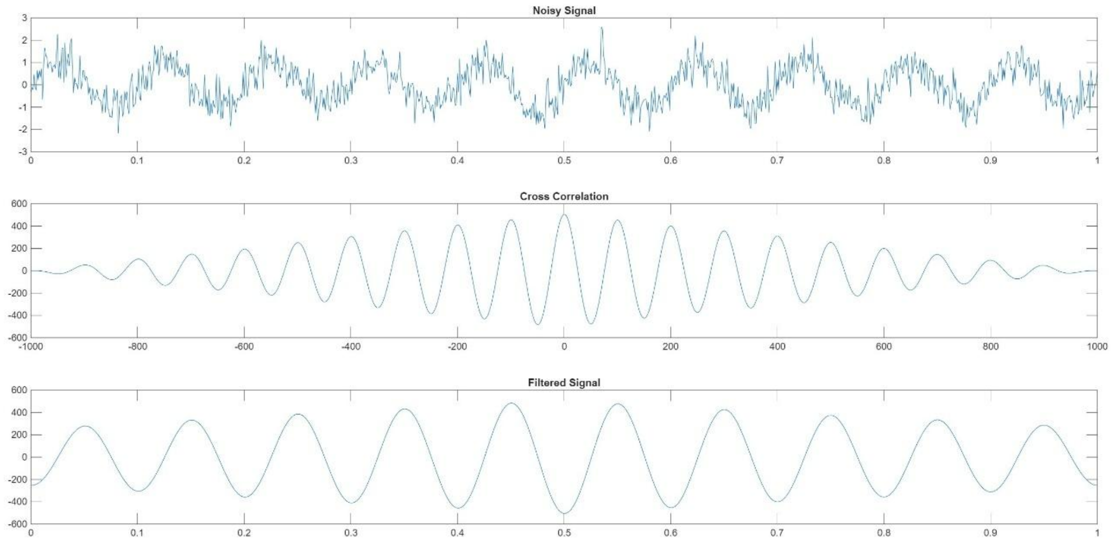
PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
clc;
clear;
close all;

t = 0:0.001:1;

signal = sin(2*pi*10*t);
noise = 0.5*randn(size(t));

noisy_signal = signal + noise;

[corr,lags] = xcorr(noisy_signal,signal);

filtered_signal = conv(noisy_signal,signal,'same');

figure

subplot(3,1,1)
plot(t,noisy_signal)
title('Noisy Signal')

subplot(3,1,2)
plot(lags,corr)
title('Cross Correlation')

subplot(3,1,3)
plot(t,filtered_signal)
title('Filtered Signal')
```

OUTPUT:-



RESULT:- **Hence the** Removal of Noise Using Cross Correlation **successfully completed by using MATLAB**

VIVA QUESTIONS:-

1. What is noise in signal processing?
2. What is cross-correlation?
3. How is cross-correlation different from convolution?
4. What is the purpose of using cross-correlation in noise removal?
5. Define auto-correlation and cross-correlation.

6. Write the mathematical expression for cross-correlation.
7. What does the peak in a cross-correlation function indicate?
8. How does cross-correlation help in detecting similar signals?
9. What assumptions are made about noise when using cross-correlation?
10. Explain the concept of lag in cross-correlation.

11. Why does cross-correlation reduce random noise?
12. What happens if noise is correlated instead of random?
13. Can cross-correlation completely remove noise? Why or why not?
14. How does signal-to-noise ratio (SNR) improve using cross-correlation?
15. What type of signals are best suited for this technique?

16. How is cross-correlation implemented in digital systems?
17. What role does sampling rate play in cross-correlation?
18. Where is cross-correlation-based noise removal used in real life?
19. What are the limitations of cross-correlation in noise reduction?
20. How does cross-correlation compare with filtering methods for noise removal?

EXERCISE PROGRAMS

1. Write a MATLAB program to generate a noisy sinusoidal signal and use cross-correlation with a clean reference signal to reduce noise.
2. Write a MATLAB program to add white Gaussian noise to a signal and estimate the original signal using cross-correlation.
3. Write a MATLAB program to detect a known signal buried in noise using cross-correlation.
4. Write a MATLAB program to compute the cross-correlation between a noisy signal and a template signal and identify the delay.
5. Write a MATLAB program to remove noise from a signal using normalized cross-correlation.
6. Write a MATLAB program to compare autocorrelation and cross-correlation for noise reduction.
7. Write a MATLAB program to estimate signal similarity in noisy conditions using cross-correlation.
8. Write a MATLAB program to filter noise from a periodic signal using cross-correlation with one period as reference.
9. Write a MATLAB program to simulate signal transmission with noise and recover it using cross-correlation.
10. Write a MATLAB program to detect the position of a signal segment within a noisy signal using cross-correlation.
11. Write a MATLAB program to analyze the effect of noise variance on cross-correlation-based signal recovery.
12. Write a MATLAB program to apply cross-correlation for speech signal denoising.
13. Write a MATLAB program to perform cross-correlation between two noisy signals and extract the common signal.
14. Write a MATLAB program to use cross-correlation for echo detection and noise removal.
15. Write a MATLAB program to compute cross-correlation in time and frequency domain for noise suppression.
16. Write a MATLAB program to remove impulse noise from a signal using cross-correlation with a clean template.
17. Write a MATLAB program to align two noisy signals using cross-correlation and reduce noise.
18. Write a MATLAB program to detect repeating patterns in a noisy signal using cross-correlation.
19. Write a MATLAB program to compare cross-correlation-based filtering with moving average filtering.
20. Write a MATLAB program to visualize the improvement in signal-to-noise ratio (SNR) after applying cross-correlation.

code solutions for any or all of these exercises.

REAL TIME APPLICATIONS:

- Network **packet arrivals**
- Customer **service**
- Reliability **engineering**

EXPERIMENT 15

Extraction of Periodic Signal Using Auto Correlation

AIM: -

Write a Matlab code for Extraction of Periodic Signal Using Auto Correlation

SOFTWARE REQURIED:-

1.MATLAB R2025a.
2.Windows XP SP2.

THEORY:-

Extraction of Periodic Signal Using Auto-Correlation

1. Definition of Auto-Correlation:

Auto-correlation measures the similarity of a signal with a delayed version of itself as a function of the delay (τ).

[
R_{xx}(\tau) = \int_{-\infty}^{\infty} x(t) \cdot x(t+\tau) , dt
]

- Peaks in the auto-correlation function indicate periodicity.

2. Principle:

- Periodic signals repeat after a fixed interval (T).
- When a noisy signal contains a periodic component, auto-correlation reinforces the periodic part while reducing random noise.
- The time delay corresponding to the first significant peak gives the **period** of the signal.

3. Steps for Extraction:

1. Obtain the noisy signal (x(t)).
2. Compute its auto-correlation (R_{xx}(\tau)).
3. Identify the peaks in (R_{xx}(\tau)).
4. Determine the period (T) from the lag between successive peaks.
5. Use this information to reconstruct or extract the periodic signal.

4. Advantages:

- Effective in noisy environments.
- No prior knowledge of the signal shape is needed.
- Simple computationally for digital signals.

5. Applications:

- Speech processing (pitch detection)
- Vibration analysis
- Communication systems (periodic signal detection)

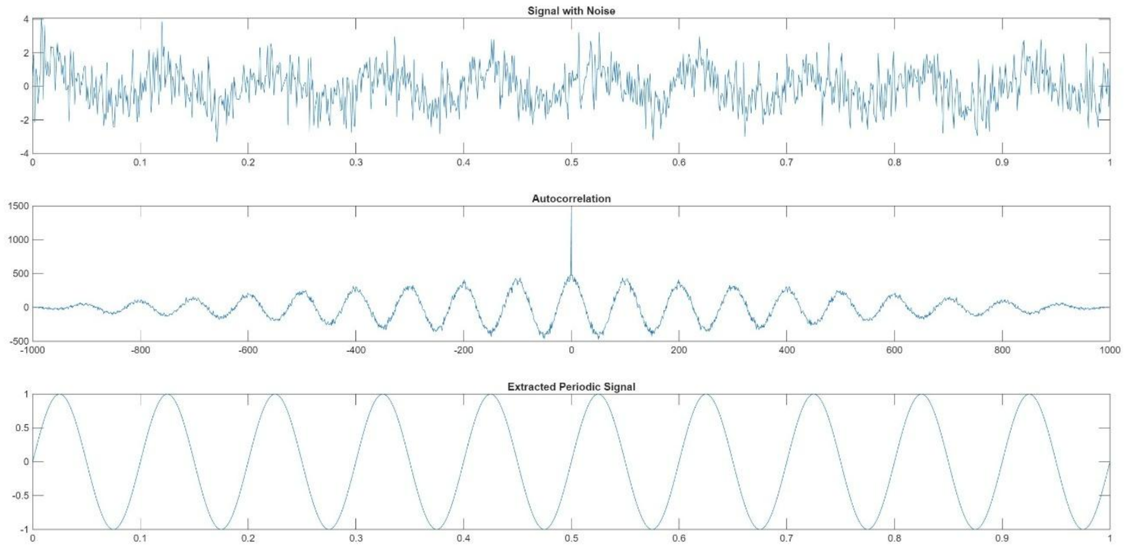
PROCEDURE: -

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM: -

```
clc;
clear;
close all;

t = 0:0.001:1;

signal = sin(2*pi*10*t);
noise = randn(size(t));

noisy = signal + noise;

[auto_corr,lags] = xcorr(noisy);

figure

subplot(3,1,1)
plot(t,noisy)
title('Signal with Noise')

subplot(3,1,2)
plot(lags,auto_corr)
title('Autocorrelation')

subplot(3,1,3)
plot(t,signal)
title('Extracted Periodic Signal')
```

OUTPUT:-



RESULT:-

Hence the Extraction of Periodic Signal Using Auto Correlation **successfully completed by using MATLAB**

VIVA QUESTIONS:-

1. What is auto-correlation of a signal?
2. Why is auto-correlation used in signal processing?
3. What is a periodic signal? Give an example.
4. How does auto-correlation help in detecting periodicity?
5. What is the mathematical definition of auto-correlation?
6. What is the difference between auto-correlation and cross-correlation?

7. How can you determine the period of a signal using auto-correlation?
8. Why does the auto-correlation function show peaks at multiples of the period?
9. What happens to auto-correlation for a non-periodic signal?
10. Can auto-correlation detect hidden periodic signals in noise? How?

11. What is the formula for discrete-time auto-correlation?
12. What is the significance of lag (shift) in auto-correlation?
13. What is meant by normalized auto-correlation?
14. How does noise affect the auto-correlation function?
15. Why is the peak at zero lag always maximum?

16. Where is auto-correlation used in real-world applications?
17. How is auto-correlation used in speech signal processing?
18. How can auto-correlation be used in pitch detection?
19. What are the limitations of using auto-correlation for periodic signal extraction?
20. How does sampling affect auto-correlation analysis?

EXERCISE PROGRAMS

1. Write a MATLAB program to generate a noisy sinusoidal signal and use auto-correlation to identify its fundamental period.
2. Write a MATLAB program to add white Gaussian noise to a periodic signal and recover its periodicity using auto-correlation.
3. Write a MATLAB program to detect a hidden periodic signal in noise using auto-correlation.
4. Write a MATLAB program to compute the auto-correlation of a noisy signal and estimate its dominant frequency.
5. Write a MATLAB program to extract a weak periodic signal from noise using normalized auto-correlation.
6. Write a MATLAB program to compare auto-correlation results of clean and noisy sinusoidal signals.
7. Write a MATLAB program to generate a square wave with noise and use auto-correlation to detect its period.
8. Write a MATLAB program to analyze the effect of noise variance on auto-correlation of a periodic signal.
9. Write a MATLAB program to compute biased and unbiased auto-correlation estimates of a signal.
10. Write a MATLAB program to detect repeating patterns in a signal using auto-correlation.
11. Write a MATLAB program to estimate the time delay between two periodic signals using cross-correlation.
12. Write a MATLAB program to generate two sinusoidal signals with phase shift and determine the shift using cross-correlation.
13. Write a MATLAB program to detect a known signal buried in noise using cross-correlation.
14. Write a MATLAB program to compute the cross-correlation between a noisy signal and a template signal and identify the delay.
15. Write a MATLAB program to extract a periodic component from a mixed signal using cross-correlation with a reference signal.
16. Write a MATLAB program to compare auto-correlation and cross-correlation for periodic signal detection.
17. Write a MATLAB program to estimate the frequency of a signal using peaks in its auto-correlation function.
18. Write a MATLAB program to remove noise from a periodic signal using averaging based on auto-correlation.
19. Write a MATLAB program to analyze the symmetry properties of auto-correlation for periodic signals.
20. Write a MATLAB program to visualize auto-correlation and cross-correlation of different periodic waveforms (sine, square, triangle).

REAL TIME APPLICATIONS:

- Network **packet arrivals**
- Customer **service**
- Reliability **engineering**

EXPERIMENT 16

Correlation Between Two Signals

AIM: -

Write a Matlab code for Correlation Between Two Signals

SOFTWARE REQURIED:-

1.MATLAB R2025a.
2.Windows XP SP2.

THEORY: -

Correlation Between Two Signals

1. Definition:

Correlation measures the similarity or relationship between two signals as a function of time-lag applied to one of them. It helps identify patterns, delays, or how one signal influences another.

2. Types of Correlation:

- **Cross-correlation:** Measures similarity between two different signals.
- **Auto-correlation:** Measures similarity of a signal with itself at different time lags.

3. Mathematical Expression:

For continuous-time signals ($x(t)$) and ($y(t)$):

$$R_{xy}(\tau) = \int_{-\infty}^{\infty} x(t) \cdot y(t+\tau) \, dt$$

For discrete-time signals ($x[n]$) and ($y[n]$):

$$R_{xy}[k] = \sum_{n=-\infty}^{\infty} x[n] \cdot y[n+k]$$

Where (τ) (or (k)) is the **time lag**.

4. Properties:

- **Symmetry:** ($R_{xy}(\tau) = R_{yx}(-\tau)$)
- **Maximum correlation:** Indicates the best alignment or similarity.
- **Unitless normalized correlation:** Often used to compare different signals:

  $$\rho_{xy} = \frac{R_{xy}}{\sqrt{R_{xx} R_{yy}}}$$

5. Applications:

- Signal detection & synchronization
- Time-delay estimation
- Pattern recognition
- Communications (finding similarity between transmitted and received signals)

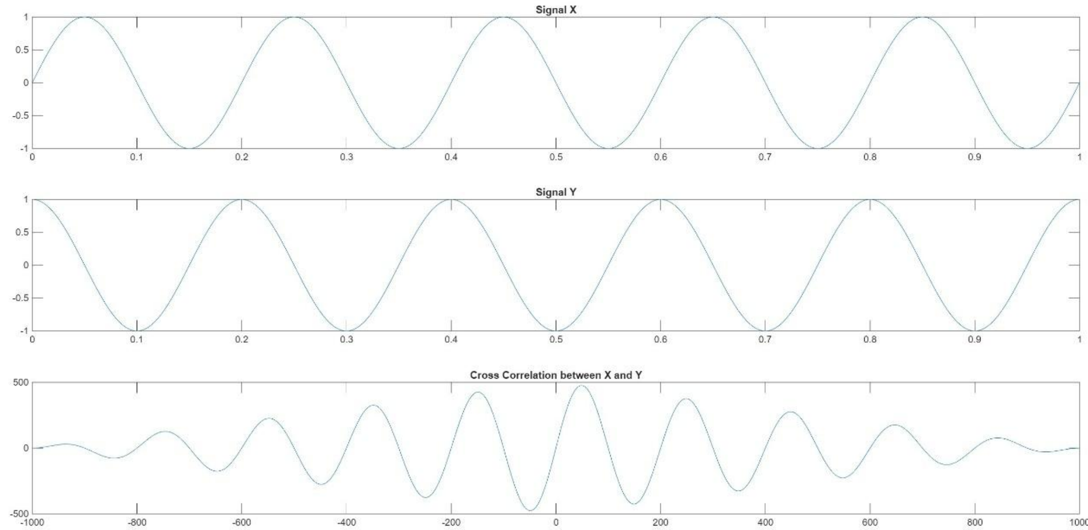
PROCEDURE: -

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM: -

```
clc;
clear;
close all;

t = 0:0.001:1;

x = sin(2*pi*5*t);
y = cos(2*pi*5*t);

[c,lags] = xcorr(x,y);

figure

subplot(3,1,1)
plot(t,x)
title('Signal X')

subplot(3,1,2)
plot(t,y)
title('Signal Y')

subplot(3,1,3)
plot(lags,c)
title('Cross Correlation between X and Y')
```

OUTPUT: -



RESULT:-

Hence the Correlation Between Two Signals **successfully completed by using MATLAB**

VIVA QUESTIONS:-

1. What is correlation between two signals?
2. What is the difference between correlation and convolution?
3. Define cross-correlation.
4. Define auto-correlation.
5. What does the value of correlation indicate about two signals?
6. What is the range of correlation coefficient?
7. What is meant by normalized correlation?

8. Write the mathematical expression for cross-correlation of two continuous signals.
9. Write the formula for discrete-time cross-correlation.
10. How is auto-correlation related to signal energy?
11. What happens to correlation when signals are orthogonal?

12. State the symmetry property of auto-correlation.
13. What is the maximum value of auto-correlation and when does it occur?
14. How does time shifting affect correlation?
15. What is the effect of scaling a signal on correlation?

16. How is correlation used in signal detection?
17. How is cross-correlation used in time delay estimation?
18. What is the role of correlation in pattern recognition?
19. How is correlation applied in communication systems?

20. What is the relationship between correlation and power spectral density (Wiener–Khinchin theorem)?

.

EXERCISE PROGRAMS

1. Write a MATLAB program to generate a noisy sinusoidal signal and use auto-correlation to identify its fundamental period.
2. Write a MATLAB program to add white Gaussian noise to a periodic signal and recover its periodicity using auto-correlation.
3. Write a MATLAB program to detect a hidden periodic signal in noise using auto-correlation.
4. Write a MATLAB program to compute the auto-correlation of a noisy signal and estimate its dominant frequency.
5. Write a MATLAB program to compute and plot the auto-correlation of a discrete-time signal.
6. Write a MATLAB program to compare auto-correlation of a clean signal and a noisy signal.
7. Write a MATLAB program to compute cross-correlation between two signals and determine their similarity.
8. Write a MATLAB program to find the time delay between two signals using cross-correlation.
9. Write a MATLAB program to align two signals using cross-correlation.
10. Write a MATLAB program to detect echo or repeated patterns in a signal using auto-correlation.
11. Write a MATLAB program to estimate signal periodicity using normalized auto-correlation.
12. Write a MATLAB program to compare cross-correlation results for identical and different signals.
13. Write a MATLAB program to compute correlation coefficient between two signals.
14. Write a MATLAB program to detect signal similarity in the presence of noise using cross-correlation.
15. Write a MATLAB program to identify a known signal pattern within a longer signal using cross-correlation.
16. Write a MATLAB program to analyze correlation between two random signals and interpret the result.
17. Write a MATLAB program to compute and visualize 2D correlation between two matrices (images).
18. Write a MATLAB program to use cross-correlation for template matching in signals.
19. Write a MATLAB program to compare biased and unbiased auto-correlation estimates.
20. Write a MATLAB program to compute correlation of two signals in the frequency domain using FFT.

REAL TIME APPLICATIONS:

- **Network packet arrivals**
- **Customer service**
- **Reliability engineering**

EXPERIMENT 17

Build and Simulate a DC Motor using MATLAB / Simulink
MATLAB Code (DC Motor Transfer Function Simulation)

AIM: -

Write a Matlab code for Build and Simulate a DC Motor using MATLAB / Simulink MATLAB Code (DC Motor Transfer Function Simulation)

SOFTWARE REQURIED:-

1.MATLAB R2025a.

2.Windows XP SP2.

THEORY: -

DC Motor Overview

A DC motor converts electrical energy into mechanical energy. Key components:

- **Armature (rotor)**: Rotates to produce mechanical output.
- **Field winding (stator)**: Produces magnetic field.
- **Commutation system**: Maintains unidirectional torque.

Equations:

1. **Electrical equation (armature circuit):**

Va(t)=Ldia(t)dt+Ria(t)+eb(t)V_a(t) = L \frac{di_a(t)}{dt} + R i_a(t) + e_b(t)Va(t)=Ldtdia(t)+Ria(t)+eb(t)

Where:

- VaV_aVa = applied voltage (V)
- iai_aia = armature current (A)
- R,LR, LR,L = armature resistance and inductance
- eb=Keω(t)e_b = K_e \omega(t)eb=Keω(t) = back EMF
- ω(t)\omega(t)ω(t) = angular speed (rad/s)

2. **Mechanical equation (rotor dynamics):**

Tm(t)=Jdω(t)dt+Bω(t)T_m(t) = J \frac{d\omega(t)}{dt} + B \omega(t)Tm(t)=Jdtdω(t)+Bω(t)

Where:

- Tm=Ktia(t)T_m = K_t i_a(t)Tm=Ktia(t) = motor torque
- JJJ = rotor inertia
- BBB = viscous friction coefficient

Transfer Function of DC Motor

Assuming armature-controlled motor:

1. Laplace transform electrical and mechanical equations:

Va(s)=(Ls+R)Ia(s)+KeΩ(s)V_a(s) = (Ls + R) I_a(s) + K_e \Omega(s)Va(s)=(Ls+R)Ia(s)+KeΩ(s)
KtIa(s)=(Js+B)Ω(s)K_t I_a(s) = (Js + B) \Omega(s)KtIa(s)=(Js+B)Ω(s)

2. Combine to get **speed transfer function**:

Ω(s)Va(s)=Kt(Ls+R)(Js+B)+KeKt\frac{\Omega(s)}{V_a(s)} = \frac{K_t}{(L s + R)(J s + B) + K_e K_t}Va(s)Ω(s) =(Ls+R)(Js+B)+KeKtKt

3. For **position transfer function**:

Θ(s)/Va(s)=Ω(s)sVa(s)\Theta(s)/V_a(s) = \frac{\Omega(s)}{s V_a(s)}Θ(s)/Va(s)=sVa(s)Ω(s)

Simulink Implementation

Steps to build and simulate:

1. Open **Simulink** → New Model.
2. Use these blocks:
   - **Voltage source** → input voltage VaV_aVa
   - **Gain blocks** → KtK_tKt, KeK_eKe
   - **Integrator blocks** → for ω\omegaω and θ\thetaθ
   - **Sum blocks** → for combining voltage, back EMF, and torque
   - **Scope** → to visualize speed/position
3. Connect the blocks according to the **DC motor transfer function**.

PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
clc;
clear;
close all;

% DC Motor Parameters
J = 0.01;      % Moment of inertia
b = 0.1;       % Friction coefficient
K = 0.01;      % Motor constant
R = 1;         % Resistance
L = 0.5;       % Inductance

% Transfer Function of DC Motor
num = K;
den = [J*L (J*R + b*L) (b*R + K^2)];

motor = tf(num, den);

disp('DC Motor Transfer Function:')
motor

% Step Response
```

```
figure
step(motor)
title('Step Response of DC Motor')
grid on
```

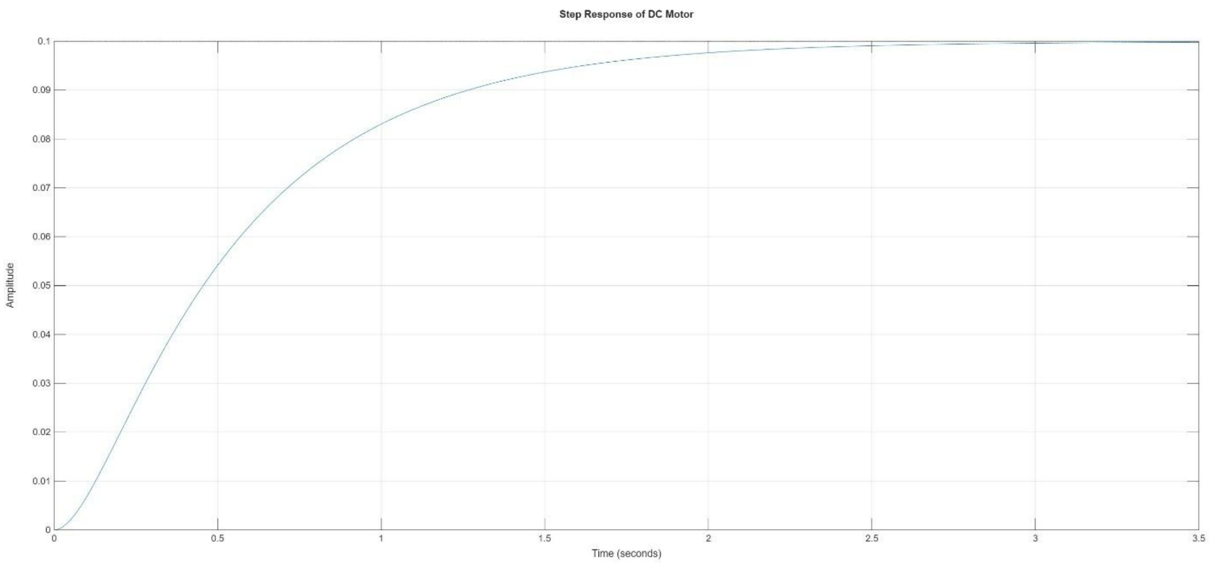
Simulink Blocks to Use

- Step Input
- Transfer Function
- Scope

Transfer Function parameters:

Numerator: [K]
Denominator: [J*L (J*R + b*L) (b*R + K^2)]

OUTPUT:-



RESULT:-

Thus the Build and Simulate a DC Motor using MATLAB / SimulinkMATLAB Code (DC Motor Transfer Function Simulation)**successfully completed by using MATLAB**

VIVA QUESTIONS:-

1. What is a DC motor and how does it work?
2. What are the main components of a DC motor?
3. Define back EMF in a DC motor.
4. Why is back EMF important in motor operation?
5. What are the types of DC motors?

6. Write the electrical equation of a DC motor.
7. Write the mechanical equation of a DC motor.
8. What assumptions are made while deriving the DC motor transfer function?
9. What is the transfer function of a DC motor?

10. What are the parameters involved in the transfer function (R, L, J, B, K)?
11. How do you represent a transfer function in MATLAB?
12. What is the purpose of the tf() function in MATLAB?
13. How do you simulate step response in MATLAB?
14. What does the step response of a DC motor indicate?
15. How can you analyze system stability using MATLAB?
16. What blocks are used to model a DC motor in Simulink?
17. How do you implement a transfer function in Simulink?
18. What is the role of the Scope block?
19. How can you change simulation parameters in Simulink?
20. What is the difference between MATLAB simulation and Simulink simulation?

EXERCISE PROGRAMS

1. **Write a MATLAB program to model a DC motor using its transfer function and plot the step response.**
2. **Write a MATLAB program to simulate a DC motor's speed response to a ramp input and plot the results.**
3. **Write a MATLAB program to generate a noisy voltage input to a DC motor and observe the effect on motor speed.**
4. **Write a MATLAB program to compute the DC motor's transfer function from its electrical and mechanical parameters.**
5. **Write a MATLAB program to simulate a DC motor under PID control and plot the closed-loop speed response.**
6. **Write a MATLAB program to add Gaussian noise to the motor input and use filtering to estimate the actual speed.**
7. **Write a MATLAB program to simulate a DC motor with a step disturbance torque and observe the transient response.**
8. **Write a MATLAB program to calculate the DC motor's frequency response and plot its Bode plot.**
9. **Write a MATLAB program to design a PID controller for a DC motor using the transfer function and simulate the closed-loop response.**
10. **Write a MATLAB program to simulate the DC motor's position response using a step input to the voltage and plot angular displacement.**
11. **Write a MATLAB program to model a DC motor using Simulink and simulate its response to a sinusoidal input voltage.**
12. **Write a MATLAB program to compute the DC motor's time constant from step response simulation.**
13. **Write a MATLAB program to simulate the effect of armature resistance variation on the speed response of a DC motor.**
14. **Write a MATLAB program to generate a noisy step input to the DC motor and estimate the speed using auto-correlation.**
15. **Write a MATLAB program to simulate a DC motor with load torque feedback and plot both speed and torque responses.**
16. **Write a MATLAB program to linearize a Simulink DC motor model and compare it to the transfer function model in MATLAB.**
17. **Write a MATLAB program to simulate the DC motor under a square wave input and analyze the overshoot and settling time.**

18. **Write a MATLAB program to design a lead-lag compensator for a DC motor and simulate the improvement in transient response.**
19. **Write a MATLAB program to simulate a DC motor with input noise and use auto-correlation to detect periodic speed fluctuations.**
20. **Write a MATLAB program to implement a closed-loop DC motor control with reference tracking and simulate both speed and position responses.**

REAL TIME APPLICATIONS:

- Network **packet arrivals**
- Customer **service**
- Reliability **engineering**

EXPERIMENT-18

Implementation of PID Controller using Simulink

MATLAB Code (PID Controller)

AIM: -

Write a Matlab code for Implementation of PID Controller using Simulink
MATLAB Code (PID Controller)

SOFTWARE REQURIED: -

1.MATLAB R2025a.
2.Windows XP SP2.

THEORY:-

PID Controller Overview

PID stands for **Proportional-Integral-Derivative** controller. It is widely used in control systems to maintain the desired output by minimizing the error between a setpoint and process variable.

The PID control law is:

u(t)=Kpe(t)+Ki∫0te(τ)dτ+Kdde(t)dtu(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}u(t)=Kpe(t)+Ki∫0t e(τ)dτ+Kddtde(t)

Where:

- u(t)u(t)u(t) = control signal
- e(t)=r(t)−y(t)e(t) = r(t) - y(t)e(t)=r(t)−y(t) = error (setpoint − output)
- KpK_pKp = proportional gain
- KiK_iKi = integral gain
- KdK_dKd = derivative gain

Functions of each term:

- **Proportional (P):** Reduces present error
- **Integral (I):** Eliminates steady-state error
- **Derivative (D):** Predicts future error, improves stability

Implementation in Simulink

1. **Open Simulink:**
   - MATLAB → simulink → Blank Model
2. **Add Blocks:**
   - **Step** (Setpoint)
   - **PID Controller** (from Simulink library: Simulink → Continuous → PID Controller)
   - **Transfer Function / Plant** (system to control)
   - **Scope** (to visualize output)
   - **Sum** block (for error calculation)
3. **Connect Blocks:**

- Step → (+) input of Sum
- System output → (-) input of Sum → PID Controller → Plant → Scope

4. **Set PID Parameters:**
   - Tune Kp,Ki,KdK_p, K_i, K_dKp,Ki,Kd manually or use PID Tuner in Simulink
5. **Simulate:**
   - Run the model and observe system response on Scope

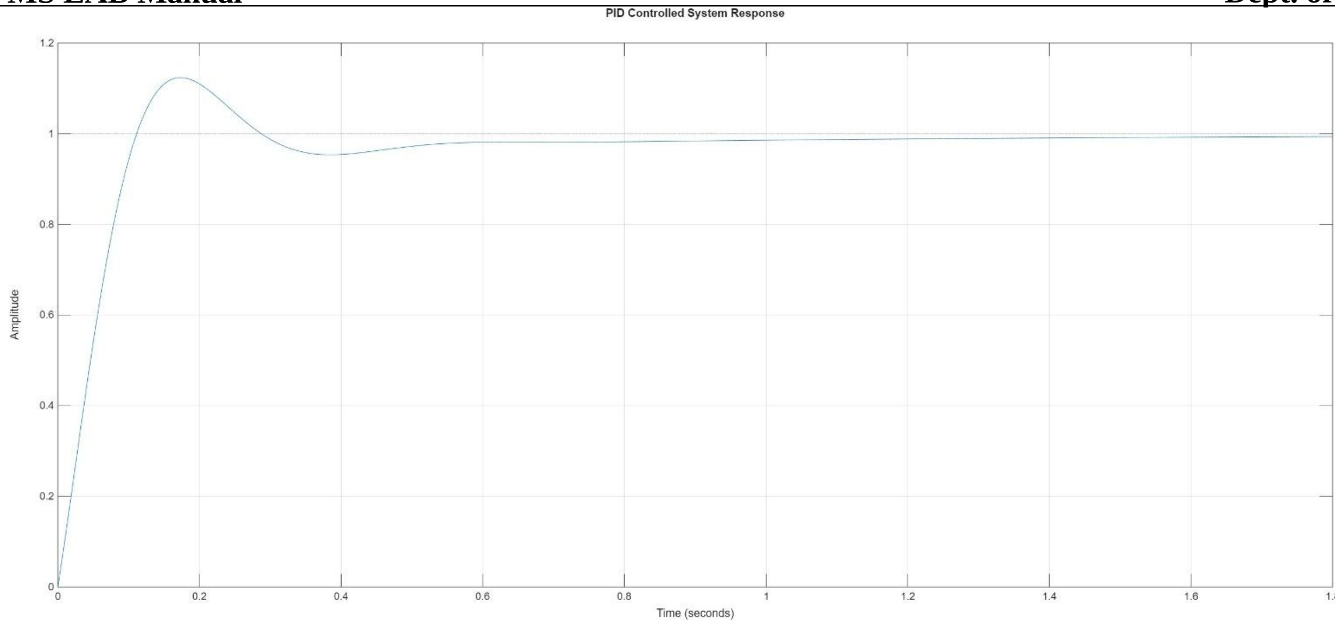
PROCEDURE:-

- Open MATLAB
- Open new M-file
- Type the program
- Save in current directory
- Compile and Run the program
- For the output see command window\ Figure window

PROGRAM:-

```
clc;
clear;
close all;

% Plant Transfer Function
num = 1;
den = [1 10 20];

plant = tf(num,den);

% PID Parameters
Kp = 300;
Ki = 300;
Kd = 10;

pid_controller = pid(Kp,Ki,Kd);

% Closed Loop System
closed_loop = feedback(pid_controller*plant,1);

figure
step(closed_loop)
title ('PID Controlled System Response')
grid on
```

OUTPUT: -



RESULT: - **Thus the** Implementation of PID Controller using Simulink
MATLAB Code (PID Controller
successfully completed by using MATLAB

VIVA QUESTIONS:-

1. What is a PID controller and what does PID stand for?
2. Explain the role of the Proportional, Integral, and Derivative terms in a PID controller.
3. How does the PID controller improve system performance?
4. What are the typical applications of PID controllers in industry?
5. What is the difference between P, PI, PD, and PID controllers?

6. How do you implement a PID controller in Simulink?
7. Which Simulink block is used to represent a PID controller?
8. How do you connect the PID block to a plant model in Simulink?
9. How can you simulate the response of a PID controller in Simulink?
10. How do you adjust the PID parameters in Simulink?

11. How can you define a PID controller using MATLAB code?
12. What is the MATLAB command to create a PID controller?
13. How do you tune the PID parameters in MATLAB programmatically?
14. How do you implement feedback control using a PID controller in MATLAB?
15. How can you plot the step response of a PID-controlled system in MATLAB?

16. What methods are used for tuning PID parameters?
17. How do you evaluate the performance of a PID controller? (mention rise time, overshoot, settling time)
18. What happens if the PID gains are too high or too low?
19. How can you handle steady-state error using a PID controller?
20. Can you explain how derivative action helps in damping system oscillations?

EXERCISE PROGRAMS

1. **Write a MATLAB program to model a DC motor using its transfer function and plot the step response.**
2. **Write a MATLAB program to simulate a DC motor's speed response to a ramp input and plot the results.**
3. **Write a MATLAB program to generate a noisy voltage input to a DC motor and observe the effect on motor speed.**
4. **Write a MATLAB program to compute the DC motor's transfer function from its electrical and mechanical parameters.**
5. **Write a MATLAB program to implement a proportional (P) controller for a DC motor and plot the step response.**
6. **Write a MATLAB program to implement a proportional-integral (PI) controller for a DC motor and compare it with the P controller.**
7. **Write a MATLAB program to implement a proportional-derivative (PD) controller for a DC motor and analyze the response.**
8. **Write a MATLAB program to implement a full PID controller for a DC motor and tune its parameters for minimal overshoot.**
9. **Write a MATLAB program to simulate a DC motor controlled by PID under a unit step input and plot the error signal.**
10. **Write a MATLAB program to plot the root locus of a DC motor with a PID controller and analyze system stability.**
11. **Write a MATLAB program to simulate a DC motor under PID control for a sinusoidal reference input and plot speed vs. time.**
12. **Write a MATLAB program to add sensor noise to the DC motor feedback and observe the PID controller's response.**
13. **Write a MATLAB program to design a PID controller using the Ziegler–Nichols tuning method and simulate the motor response.**
14. **Write a MATLAB program to implement anti-windup for the PID controller and compare it with the standard PID response.**
15. **Write a MATLAB program to plot the Bode diagram of a DC motor with PID control and analyze frequency response.**
16. **Write a MATLAB program to implement PID control in discrete-time using c2d for digital control simulation.**
17. **Write a MATLAB program to simulate PID control of a DC motor with saturation constraints on input voltage.**
18. **Write a Simulink model for a DC motor with PID control and plot the speed response to step and ramp inputs.**
19. **Write a Simulink model to simulate PID control of a DC motor under load disturbance and observe how the controller compensates.**
20. **Write a Simulink model to design a PID controller using the PID Tuner toolbox and automatically tune the parameters for desired performance.**

REAL TIME APPLICATIONS:

- Network **packet arrivals**
- Customer **service**
- Reliability **engineering**

EXPERIMENT 19
Controllability and Observability

MATLAB Code

AIM: -

Write a MATLAB code for Controllability and Observability
MATLAB Code

SOFTWARE REQURIED: -

1.MATLAB R2025a.
2.Windows XP SP2.

THEORY: -

1. Controllability

Definition:
A system is **controllable** if it is possible to move the system from any initial state (x(0)) to any final state (x(t_f)) in finite time using an appropriate input (u(t)).

State-space form:
[
\dot{x}(t) = A x(t) + B u(t)
]

- (x(t)) = state vector
- (u(t)) = input vector
- (A) = system matrix
- (B) = input matrix

Controllability Matrix:
[
\mathcal{C} = [B \ AB \ A^2B \ \dots \ A^{n-1}B]
]

- The system is **controllable** ⇨ (\text{rank}(\mathcal{C}) = n) (number of states).

Key Points:

- Only controllable states can be influenced by inputs.
- If a system is **not controllable**, some states cannot be moved arbitrarily.

2. Observability

Definition:
A system is **observable** if, for any initial state (x(0)), the output (y(t)) over a finite time interval uniquely determines (x(0)).

State-space form:
[
y(t) = C x(t) + D u(t)
]

- (y(t)) = output vector
- (C) = output matrix

Observability Matrix:
[
\mathcal{O} = \begin{bmatrix} C \ CA \ CA^2 \ \vdots \ CA^{n-1} \end{bmatrix}
]

- The system is **observable** ⇨ (\text{rank}(\mathcal{O}) = n).

Key Points:

- Only observable states can be inferred from outputs.
- Unobservable states cannot be reconstructed from outputs.

---

3. Duality

- Controllability and observability are **dual concepts**:
  [
  \text{Controllability of } (A, B) \leftrightarrow \text{Observability of } (A^T, C^T)
  ]

4. Summary Table

Property	Matrix	Condition
Controllability	(\mathcal{C} = [B \ AB \dots A^{n-1}B])	(\text{rank}(\mathcal{C}) = n)
Observability	(\mathcal{O} = \begin{bmatrix} C \ CA \ \vdots \ CA^{n-1} \end{bmatrix})	(\text{rank}(\mathcal{O}) = n)

PROCEDURE:-

```
clc;
clear;
close all;

% State Space System
A = [0 1;
    -2 -3];

B = [0;
```

```
    1];

C = [1 0];

D = 0;

sys = ss(A,B,C,D);

% Controllability Matrix
Co = ctrb(A,B);
rank_co = rank(Co);

disp('Controllability Matrix:')
disp(Co)

disp(['Rank of Controllability Matrix = ',num2str(rank_co)])

% Observability Matrix
Ob = obsv(A,C);
rank_ob = rank(Ob);

disp('Observability Matrix:')
disp(Ob)

disp(['Rank of Observability Matrix = ',num2str(rank_ob)])

% Checking conditions
n = size(A,1);

if rank_co == n
   disp('System is Controllable')
else
   disp('System is NOT Controllable')
end

if rank_ob == n
   disp('System is Observable')
else
   disp('System is NOT Observable')
end
```

OUTPUT:-

```
Controllability Matrix:
     0     1
     1    -3

Rank of Controllability Matrix = 2
Observability Matrix:
     1     0
     0     1

Rank of Observability Matrix = 2
System is Controllable
System is Observable
```

RESULT: **- Thus the** for Controllability and Observability is

MATLAB Code are successfully completed
by using MATLAB

VIVA QUESTIONS: -

1. What is **controllability** in a control system?
2. What is **observability** in a control system?
3. Explain the physical significance of controllability.
4. Explain the physical significance of observability.
5. Can a system be controllable but not observable? Give an example.
6. Can a system be observable but not controllable? Explain.

7. What is the **controllability matrix** and how is it constructed?
8. What is the **observability matrix** and how is it constructed?
9. State the **Kalman's controllability test**.
10. State the **Kalman's observability test**.
11. How do you determine if a system is **fully controllable**?
12. How do you determine if a system is **fully observable**?

13. What is the difference between **state controllability** and **output controllability**?
14. What is the difference between **state observability** and **output observability**?
15. How does **controllability affect pole placement**?
16. How does **observability affect state estimation**?
17. Can an unstable system be controllable? Can it be observable? Explain.

18. Give a real-world example where **controllability** is important.
19. Give a real-world example where **observability** is important.
20. How do **controllability and observability** affect the design of a **state feedback controller**?

EXERCISE PROGRAMS

1. **Write a MATLAB program to model a DC motor using its transfer function and plot the step response.**
2. **Write a MATLAB program to simulate a DC motor's speed response to a ramp input and plot the results.**
3. **Write a MATLAB program to generate a noisy voltage input to a DC motor and observe the effect on motor speed.**
4. **Write a MATLAB program to define a state-space model of a system and check its controllability.**
5. **Write a MATLAB program to define a state-space model of a system and check its observability.**
6. **Write a MATLAB program to design a state-feedback controller for a system and simulate the step response.**
7. **Write a MATLAB program to design a state observer for a system and compare the estimated and actual states.**
8. **Write a MATLAB program to simulate the controllability matrix of a multi-input system and verify rank.**
9. **Write a MATLAB program to simulate the observability matrix of a multi-output system and verify rank.**
10. **Write a MATLAB program to plot the pole-zero map of a system and discuss controllability implications.**
11. **Write a MATLAB program to convert a transfer function to a state-space model and test controllability.**

12. **Write a MATLAB program to convert a transfer function to a state-space model and test observability.**
13. **Write a MATLAB program to simulate the response of a controllable system with a step input.**
14. **Write a MATLAB program to simulate the response of an uncontrollable system with a step input and explain the result.**
15. **Write a MATLAB program to simulate the response of an observable system with output feedback.**
16. **Write a MATLAB program to simulate the response of an unobservable system with output feedback and explain the result.**
17. **Write a MATLAB program to design a full-order observer for a DC motor and simulate the speed estimation.**
18. **Write a MATLAB program to design a reduced-order observer and compare it with full-order observer results.**
19. **Write a MATLAB program to compute controllability and observability Gramians and visualize them.**
20. **Write a MATLAB program to simulate a system with both controllable and uncontrollable modes and analyse the response.**

REAL TIME APPLICATIONS:

- Network **packet arrivals**
- Customer **service**
- Reliability **engineering**