CONTROL SYSTEMS

LAB

II YEAR II SEM

Department of Electrical and Electronics Engineering

Gokaraju Rangaraju Institute of Engineering & Technology
Bachupally,Kukatpally,Hyderabad

TelanganaState - 500090

GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Autonomous Institute under JNTU Hyderabad)

CERTIFICATE

This is to certify that it is a bonafide record of practical work done in the Control systems Laboratory in II semester of II year during the year 2023-2024

Name:
Roll No:
Branch:

Signature of staff member

Contents:

1. TRANSFER FUNCTION FROM ZEROS AND POLES AND VICEVERSA

AIM:

To obtain a transfer function from given poles and zeros and zeros and poles from given transfer function.

APPARATUS:

Software: MATLAB

THEORY:

A transfer function is also known as the network function is a mathematical representation, in terms of spatial or temporal frequency, of the relation between the input and output of a (linear time invariant) system. The transfer function is the ratio of the output Laplace Transform to the input Laplace Transform assuming zero initial conditions. Many important characteristics of dynamic or control systems can be determined from the transfer function.

The transfer function is commonly used in the analysis of single-input single-output electronic system, for instance. It is mainly used in signal processing, communication theory, and control theory. The term is often used exclusively to refer to linear time-invariant systems (LTI). In its simplest form for continuous time input signal x(t) and output y(t), the transfer function is the linear mapping of the Laplace transform of the input, X(s), to the output Y(s).

Zeros are the value(s) for z where the numerator of the transfer function equals zero. The complex frequencies that make the overall gain of the filter transfer function zero. Poles are the value(s) for z where the denominator of the transfer function equals zero. The complex frequencies that make the overall gain of the filter transfer function infinite.

The general procedure to find the transfer function of a linear differential equation from input to output is to take the Laplace Transforms of both sides assuming zero conditions, and to solve for the ratio of the output Laplace over the input Laplace.

The transfer function provides a basis for determining important system

response characteristics without solving the complete differential equation. As defined, the

transfer function is a rational function in the complex variable $s = \sigma + j\omega$, that is

$$H(s) = \frac{b_m s_m + b_{m-1} s_{m-1} + \cdots\cdots\cdots\cdots\cdots + b_0}{a_n s_n + a_{n-1} s_{n-1} + \cdots\cdots\cdots\cdots\cdots + a_0}$$

It is often convenient to factor the polynomials in the numerator and the

denominator, and to write the transfer function in terms of those factors:

$$H(s) = \frac{N(s)}{D(s)} = K\frac{(s-z_1)(s-z_2)\cdots\cdots\cdots\cdots(s-z_{m-1})(s-z_m)}{(s-p_1)(s-p_2)\cdots\cdots\cdots\cdots(s-p_{m-1})(s-p_m)}$$

where, the numerator and denominator polynomials, N(s) and

D(s), have real coefficients defined by the system's differential equation and $K = \frac{b_m}{a_n}$.

As written in the above equation,

the $z_i's$ are the roots of the equation $N(s) = 0$ and are defined to be the system zeros

the $p_i's$ are the roots of the equation $D(s) = 0$ and are defined to be the system poles.

MATLAB PROGRAM:

Transfer function from zeros and poles:

```
z=input('enter zeroes')
p=input('enter poles')
k=input('enter gain')
[num,den]=zp2tf(z,p,k)
tf(num,den)
```

zeros and poles from Transfer function :

```
num = input('enter the numerator of the transfer function')
den = input('enter the denominator of the transfer function')
[z,p,k] = tf2zp(num,den)
```

PROCEDURE:

- Write MATLAB program in the MATLAB editor document.
- Then save and run the program.
- Give the required input.
- The syntax "zp2tf(z,p,k)" and "tf(num,den)" solves the given input poles and zeros and gives the transfer function.
- zp2tf forms transfer function polynomials from the zeros, poles, and gains of a system in factored form.
- tf2zp converts the transfer function filter parameters to pole-zero-gain form.

- [z,p,k] = tf2zp(b,a) finds the matrix of zeros z, the vector of poles p, and the associated vector of gains k from the transfer function parameters b and a:
- The numerator polynomials are represented as columns of the matrix b.
- The denominator polynomial is represented in the vector a.
- Note down the output of the program that is zeros, poles and gain obtained in MATLAB.

Now find the output theoretically and compare it with the output obtained practically

EXAMPLES

THEORITICAL CALCULATIONS:

(1)

Enter zeros

Z =

Enter poles

P =

Enter gain

K=

num =

den =

Transfer function=

Simulation Output

(2)

Enter the numerator of the transfer function

num =

Enter the denominator of the transfer function

den =

z =

p =

k=

Simulation Output

RESULT:

2.STEP RESPONSE OF A TRANSFER FUNCTION

AIM:

To obtain the step response of a transfer function of the given system using MATLAB.

APPARATUS:

Software: MATLAB

THEORY:

A step signal is a signal whose value changes from one level to another level in zero time. Mathematically, the step signal is represented as given below:

$r(t) = u(t)$, where

$u(t) = 1; t > 0$

$= 0; t < 0$

In the Laplace transform form,

$$R(s) = \frac{1}{s}$$

The step response of the given transfer function is obtained as follows:

$$T(s) = \frac{C(s)}{R(s)}$$

So, $C(s) = R(s)T(s)$

$$C(s) = \frac{T(s)}{s}$$

The output is given by,

$c(t) = L^{-1}[C(s)]$

MATLAB PROGRAM:

num = input('enter the numerator of the transfer function')

den = input('enter the denominator of the transfer function')

step(num,den)

EXAMPLE:

Obtain the step response of the transfer function given below:

PROCEDURE:

- Type the program in MATLAB editor that is in M-file.
- Save and run the program.
- Give the required inputs in the command window of MATLAB in matrix format.
- 'step' function calculates the unit step response of a linear system.
- Zero initial state is assumed in state-space case.
- When invoked with no output arguments, this function plots the step response on the screen.
- step (sys) plots the response of an arbitrary LTI system.
- This model can be continuous or discrete, and SISO or MIMO.
- The step response of multi-input systems is the collection of step responses for each input channel.
- The duration of simulation is determined automatically based on the system poles and zeroes.
- Note down the response of the transfer function obtained in MATLAB.
- The response of the transfer function is also obtained theoretically.
- Both the responses are compared.

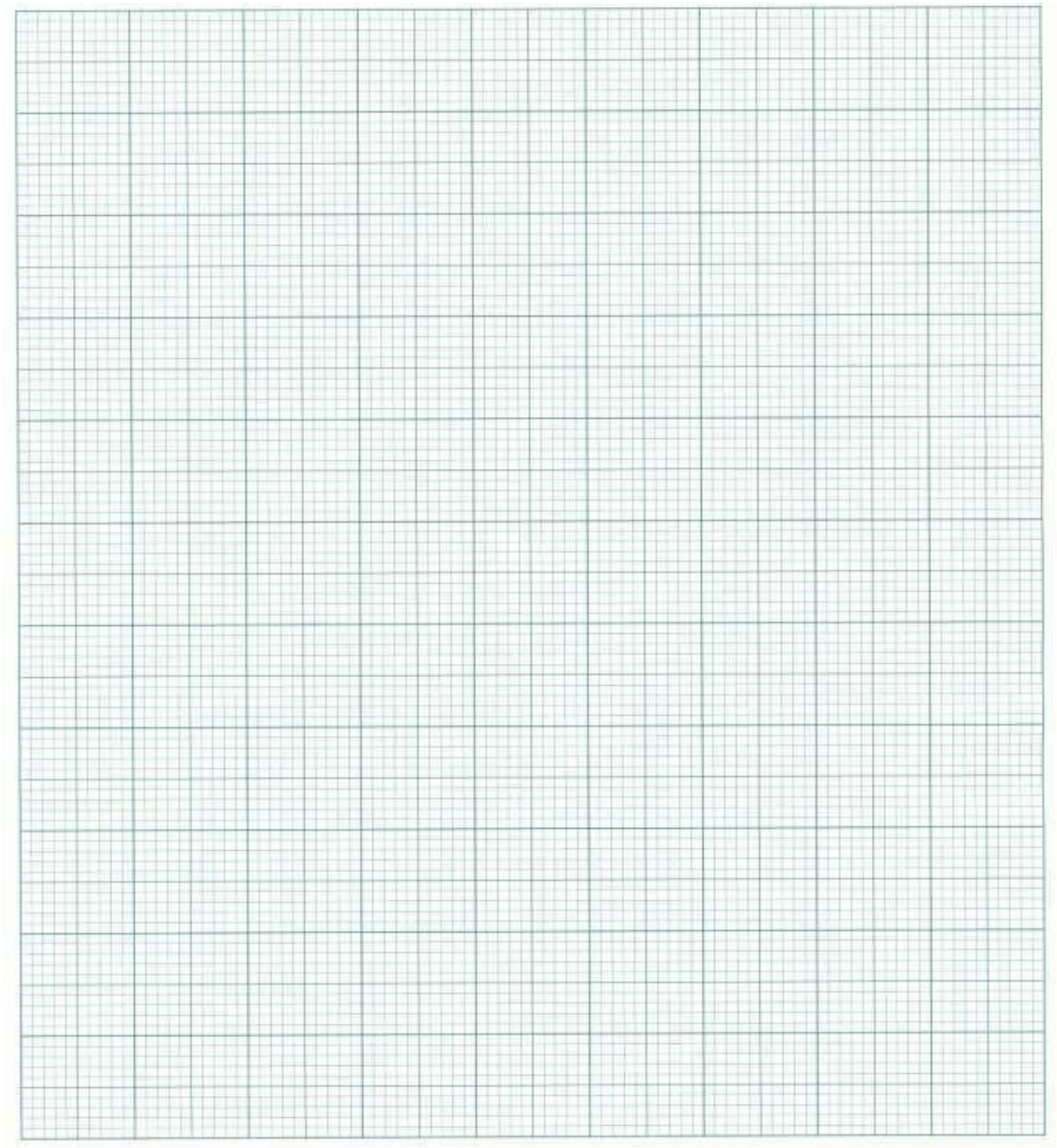
THEORETICAL CALCULATIONS:

TABULAR FORM

t	C(t)
0	
1	
2	
3	
4	
5	
6	

Simulation Output

GRAPH:



RESULT:

3.RAMP RESPONSE OF A TRANSFER FUNCTION

AIM:

To obtain the ramp response of a transfer function of the given system using MATLAB.

APPARATUS:

Software: MATLAB

THEORY:

A ramp signal is a signal which changes with time gradually in a linear fashion. Mathematically, the unit ramp signal is represented as given below:

$r(t) = t; t > 0$

$= 0; t < 0$

In the Laplace transform form,

$R(s) = \frac{1}{s^2}$

The step response of the given transfer function is obtained as follows:

$$T(s) = \frac{C(s)}{R(s)}$$

So,

$C(s) = R(s)T(s)$

$C(s) = \frac{1}{s^2} \times T(s)$

$C(s) = \frac{T(s)}{s^2}$

The output is given by,

$c(t) = L^{-1}[C(s)]$

$c(t) = L^{-1}\left[\frac{T(s)}{s^2}\right]$

MATLAB PROGRAM:

```
t = 0:0.01:10;

u=t;

num = input('enter the numerator of the transfer function')

den = input('enter the denominator of the transfer function')

lsim(num,den,u,t)
```

EXAMPLE:

Obtain the ramp response of the transfer function given below:

PROCEDURE:

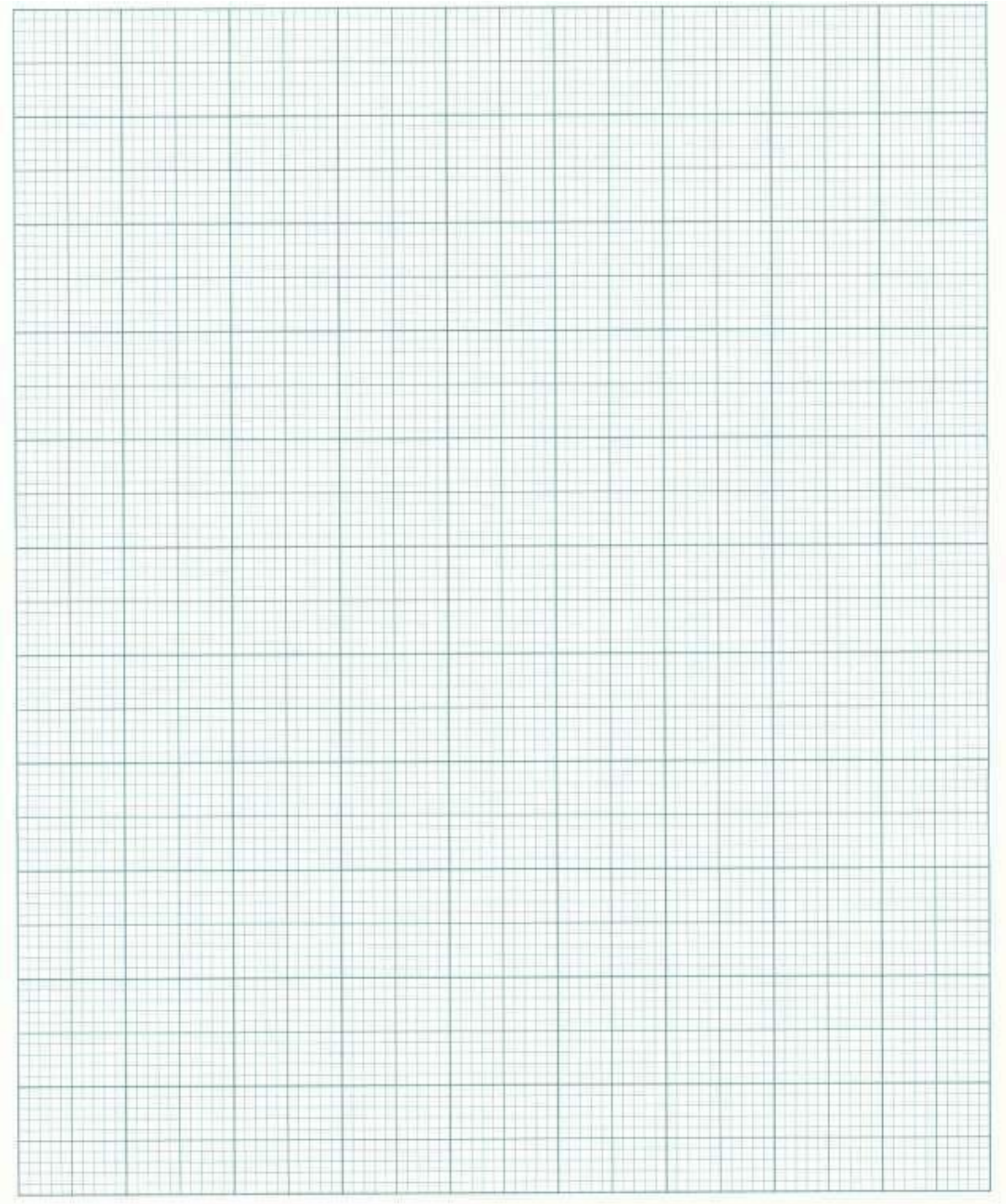
- Type the program in the MATLAB editor that is in M-file.
- Save and run the program.
- Give the required inputs in the command window of MATLAB in matrix format.
- lsim simulates the (time) response of continuous or discrete linear systems to arbitrary inputs.
- When invoked without left-hand arguments, lsim plots the response on the screen.
- lsim(sys,u,t) produces a plot of the time response of the LTI model sys to the input time history t,u.
- The vector t specifies the time samples for the simulation and consists of regularly spaced time samples.
- t = 0:dt:Tfinal
- The matrix u must have as many rows as time samples (length(t)) and as many columns as system inputs.
- Each row u(i,:) specifies the input value(s) at the time sample t(i).
- Note down the response of the transfer function obtained in MATLAB.
- The response of the transfer function is also obtained theoretically.
- Both the responses are compared.

THEORETICAL CALCULATIONS:

TABULAR FORM

t	C(t)
0	
1	
2	
3	
4	
5	
6	

GRAPH:



RESULT:

4.IMPULSE RESPONSE OF A TRANSFER FUNCTION

AIM:

To obtain the impulse response of a transfer function of the given system using MATLAB.

APPARATUS:

Software: MATLAB

THEORY:

An impulse signal is a signal whose value changes from zero to infinity in zero time. Mathematically, the unit impulse signal is represented as given below:

$r(t) = \delta(t)$,

where: $\delta(t) = 1; t = 0$

$= 0; t \neq 0$

In the Laplace transform form,

$R(s) = 1$

The impulse response of the given transfer function is obtained as follows:

$T(s) = \frac{C(s)}{R(s)}$

So, $C(s) = R(s)T(s)$

$C(s) = 1 \times R(s)$

$C(s) = R(s)$

The output is given by,

$c(t) = L^{-1}[C(s)]$

MATLAB PROGRAM:

num = input('enter the numerator of the transfer function')

den = input('enter the denominator of the transfer function')

impulse(num,den)

EXAMPLE:

Obtain the impulse response of the transfer function given below:

PROCEDURE:

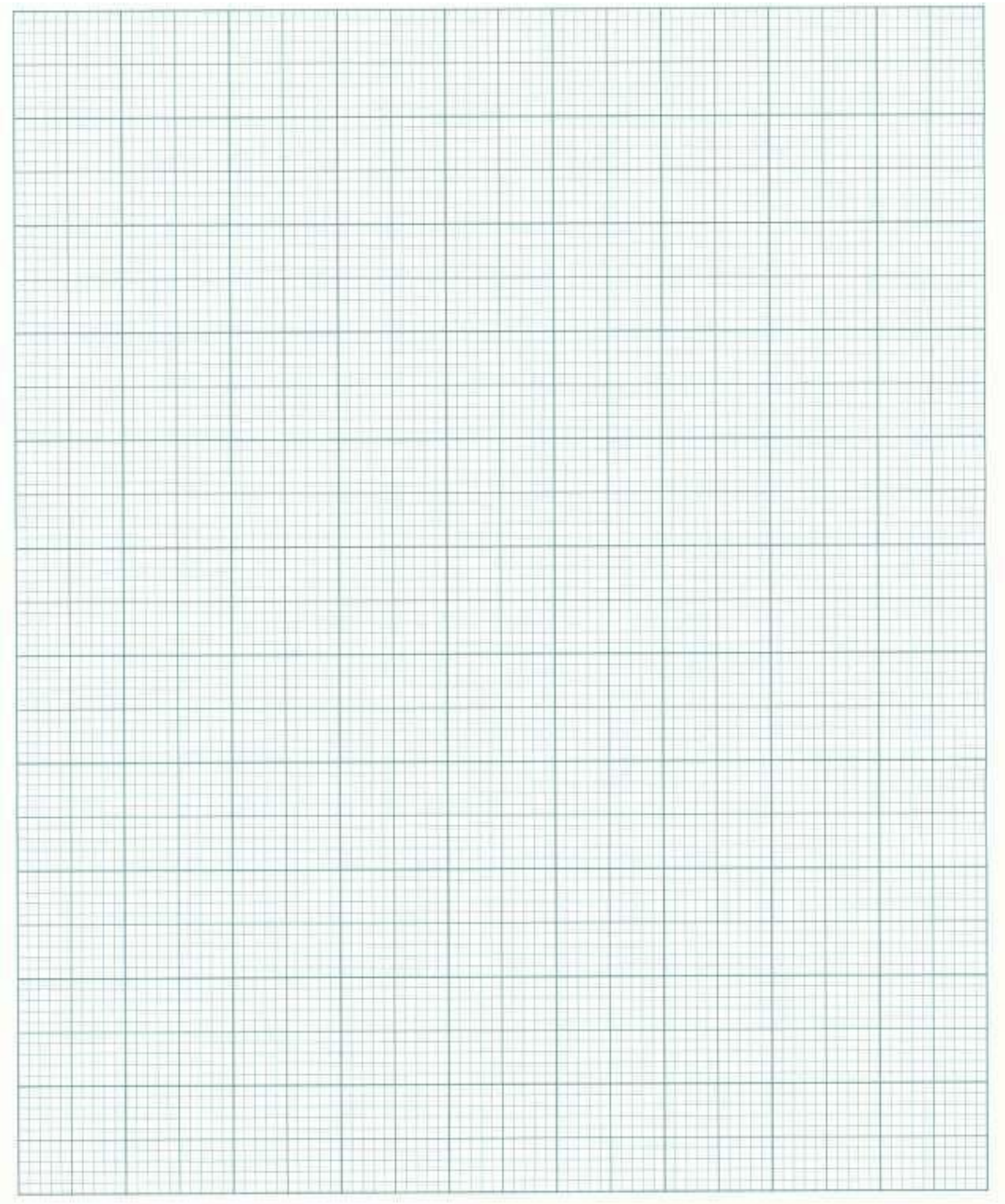
- Type the program in the MATLAB editor that is in M-file.
- Save and run the program.
- Give the required inputs in the command window of MATLAB in matrix format.
- 'impulse' calculates the impulse response of a linear system.
- The impulse response is the response to the Dirac input, δ (t) for continuous time systems and to a unit pulse at $t = 0$ for discrete time systems.
- Zero initial state is assumed in the state space case.
- When invoked without left hand arguments, this function plots the impulse response on the screen.
- 'impulse(sys)' plots the impulse response of an arbitrary LTI model sys.
- This model can be continuous or discrete, SISO or MIMO.
- The impulse response of multi-input systems is the collection of impulse responses for each input channel.
- The duration of simulation is determined automatically to display the transient behavior of the response.
- Note down the response of the given transfer function obtained in MATLAB.
- The response of the transfer function is also obtained theoretically.
- Both the responses are compared.

THEORETICAL CALCULATIONS:

TABULAR FORM

t	C(t)
0	
1	
2	
3	
4	
5	
6	

GRAPH:



RESULT:

5.ROOT LOCUS FROM A TRANSFER FUNCTION

AIM:

To plot the root locus for a given transfer function of the system using MATLAB.

APPARATUS:

Software: MATLAB

THEORY:

rlocus computes the Evans root locus of a SISO open-loop model. The root locus gives the closed-loop pole trajectories as a function of the feedback gain k (assuming negative feedback). Root loci are used to study the effects of varying feedback gains on closed-loop pole locations. In turn, these locations provide indirect information on the time and frequency responses.

rlocus(sys) calculates and plots the root locus of the open-loop SISO model sys. This function can be applied to any of the following feedback loops by setting sys appropriately.

If sys has transfer function

$h(s) = \frac{n(s)}{d(s)}$

The closed-loop poles are the roots of

$d(s) + k*n(s)=0$

MATLAB PROGRAM:

```
num=input('enter the numerator of the transfer function')
den=input('enter the denominator of the transfer function')
h=tf(num,den)
rlocus(h)
```

PROCEDURE:

- Write MATLAB program in the MATLAB specified documents.
- Then save the program to run it.
- The input is to be mentioned.

- The syntax “h=tf(num,den)” gives the transfer function and is represented as h.
- The syntax “rlocus(h)” plots the rootlocus of the transfer function h.
- Generally the syntax is of the form

  rlocus(sys)

  rlocus(sys,k)

  rlocus(sys1, sys2, ….)

  [r,k] = rlocus(sys)

  r = rlocus(sys,k)

- rlocus(sys) calculates and plots the root locus of the open loop SISO model sys.
- Now we have to solve it theoretically.
- Now we have to compare the practical and theoretical ouputs to verify each other correctly.

EXAMPLE:

Transfer function =

THEORETICAL CALCULATIONS:

enter the numerator of the transfer function

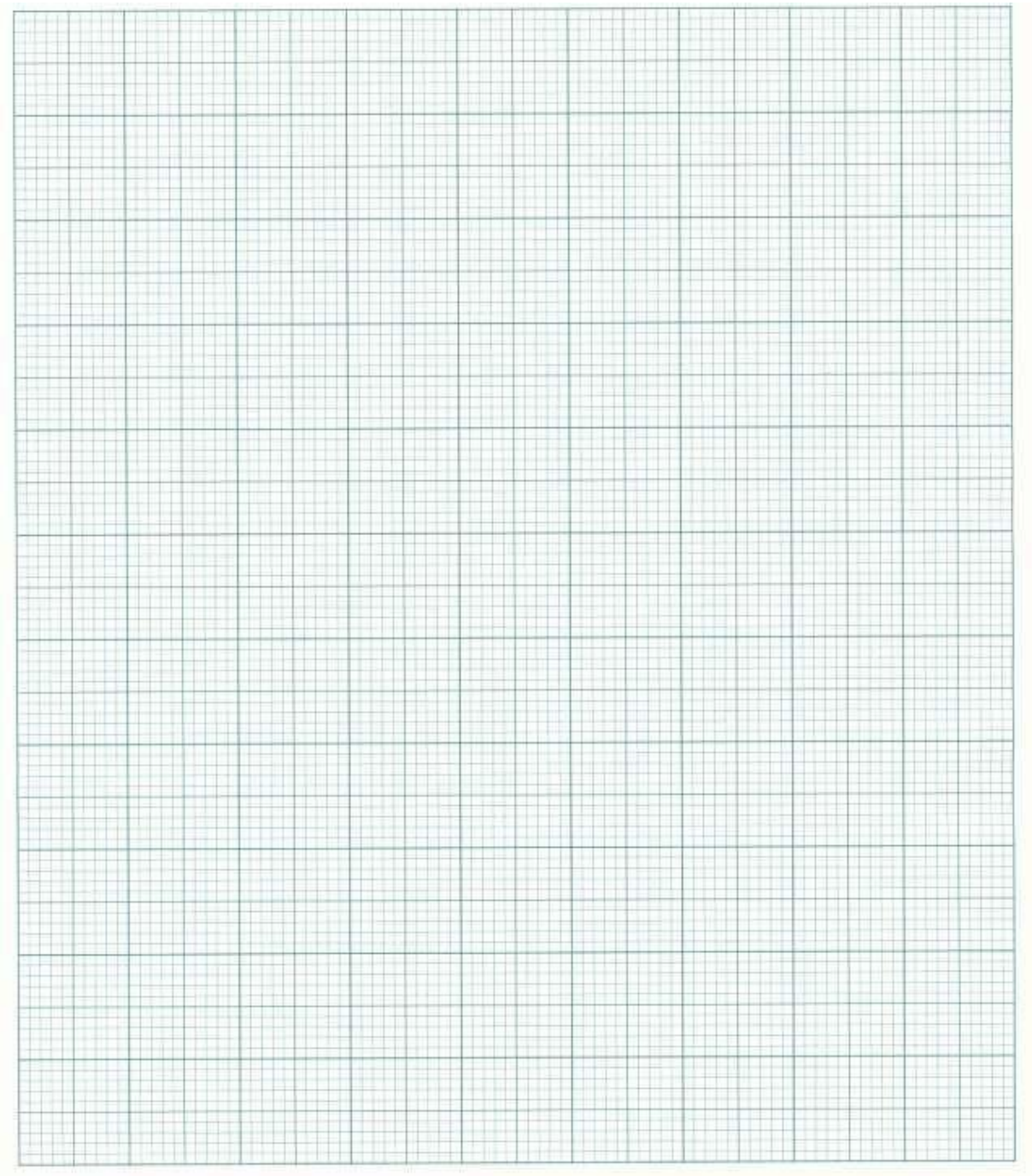
num=

enter the denominator of the transfer function

den=

Transfer function :

GRAPH:



RESULT:

6.BODE PLOT FROM A TRANSFER FUNCTION

AIM:

To obtain bode plot for a given transfer function of the system using MATLAB.

APPARATUS:

Software: MATLAB

THEORY:

Bode computes the magnitude and phase of the frequency response of LTI models. When invoked without left-side arguments, bode produces a Bode plot on the screen. The magnitude is plotted in decibels (dB), and the phase in degrees. The decibel calculation for mag is computed as 20log10(|H(jw)|), where H(jw) is the system's frequency response. Bode plots are used to analyze system properties such as the gain margin, phase margin, DC gain, bandwidth, disturbance rejection, and stability.

Bode Calculations Gain

The magnitude of the transfer function T is defined as:

$$|T(j\omega)| = \sqrt{R^2 + X^2}$$

However, it is frequently difficult to transition a function that is in "numerator/denominator" form to "real+imaginary" form. Luckily, our decibel calculation comes in handy. Let's say we have a frequency response defined as a fraction with numerator and denominator polynomials defined as:

$$T(j\omega) = \frac{\prod_n |j\omega + z_n|}{\prod_m |j\omega + p_m|}$$

If we convert both sides to decibels, the logarithms from the decibel calculations convert multiplication of the arguments into additions, and the divisions into subtractions:

$$Gain = \sum_n 20\log(j\omega + z_n) - \sum_m 20\log(j\omega + p_m)$$

bode(sys) plots the Bode response of an arbitrary LTI model sys. This model can be continuous or discrete, and SISO or MIMO. In the MIMO case, bode produces an array of Bode plots, each plot showing the Bode response of one particular I/O channel. The frequency range is determined automatically based on the system poles and zeros.

bode(sys,w) explicitly specifies the frequency range or frequency points to be used for the plot. To focus on a particular frequency interval [wmin,wmax], set w = {wmin,wmax}. To use particular frequency points, set w to the vector of desired frequencies. Use logspace to generate logarithmically spaced frequency vectors. All frequencies should be specified in radians/sec.

bode(sys1,sys2,...,sysN) or bode(sys1,sys2,...,sysN,w) plots the Bode responses of several LTI models on a single figure. All systems must have the same number of inputs and outputs, but may otherwise be a mix of continuous and discrete systems. This syntax is useful to compare the Bode responses of multiple systems.

bode(sys1,'PlotStyle1',...,sysN,'PlotStyleN') specifies which color, linestyle, and/or marker should be used to plot each system. For example,

bode(sys1,'r--',sys2,'gx')

uses red dashed lines for the first system sys1 and green 'x' markers for the second system sys2.

When invoked with left-side arguments

[mag,phase,w] = bode(sys)

[mag,phase] = bode(sys,w)

return the magnitude and phase (in degrees) of the frequency response at the frequencies w (in rad/sec). The outputs mag and phase are 3-D arrays with the frequency as the last dimension (see "Arguments" below for details). You can convert the magnitude to decibels by

magdb = 20*log10(mag)

MATLAB PROGRAM:

```
num=input('enter the numerator of the transfer function')

den=input('enter the denominator of the transfer function')

h=tf(num,den)

[gm pm wcp wcg]=margin(h)

bode(h)
```

EXAMPLE:

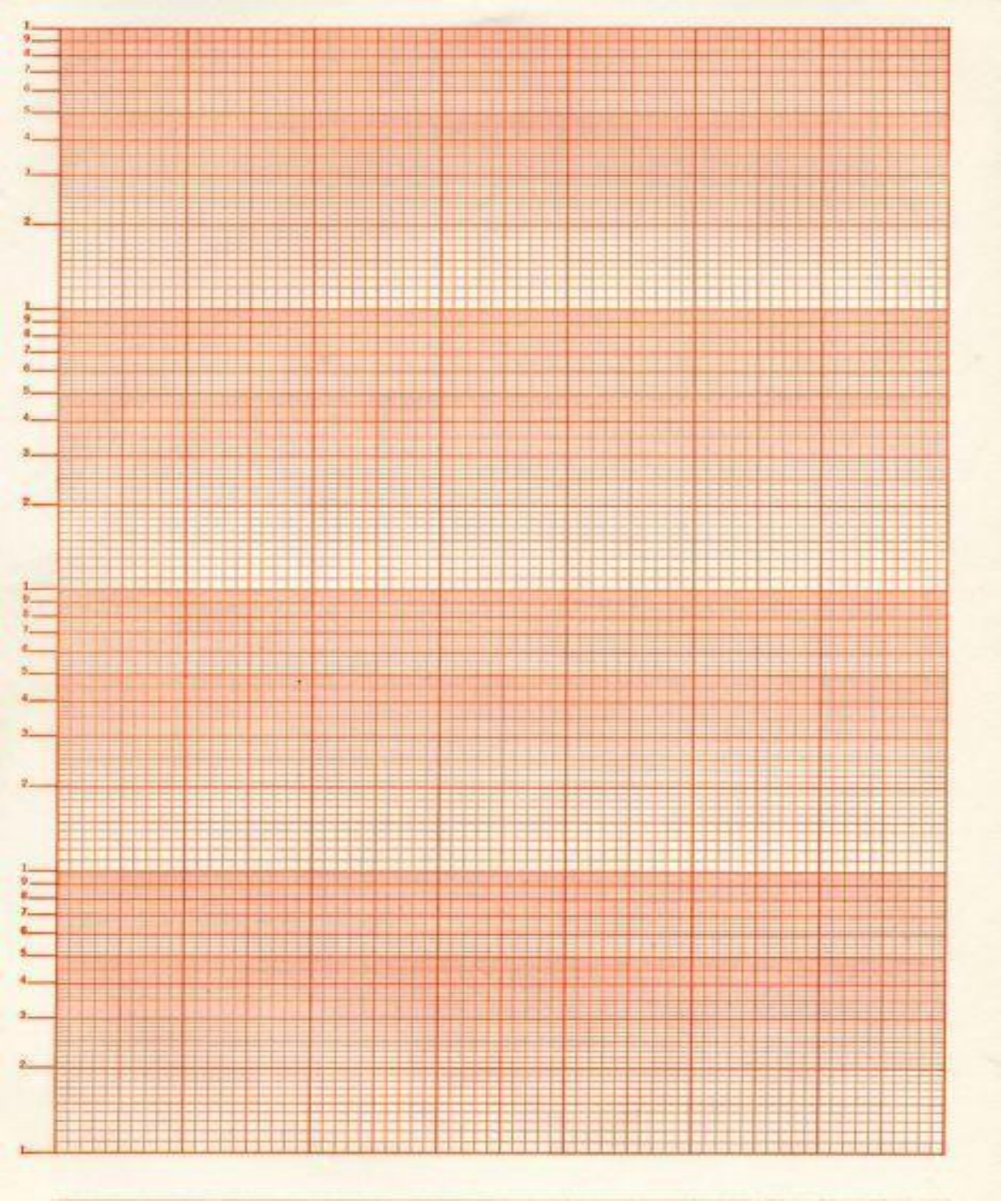
Transfer function=

PROCEDURE:

- Write the MATLAB program in the MATLAB editor.
- Then save and run the program.
- Give the required inputs.
- The syntax "bode(h)" solves the given input transfer function and gives the bode plot,
- where num,den are the numerator and denominator of the transfer function.
- Now plot the bode plot theoretically for the given transfer function and compare it with the plot obtained practically.

THEORETICAL CALCULATIONS:

GRAPH:



RESULT:

7. STATE MODEL FROM TRANSFER FUNCTION

AIM:

To obtain the state model from the given transfer function.

APPARATUS:

Software: MATLAB

THEORY:

There are three methods for obtaining state model from transfer function:

1. Phase variable method
2. Physical variable method
3. Canonical variable method

Out of three methods given above canonical form is probably the most straightforward method for converting from the transfer function of a system to a state space model is to generate a model in "controllable canonical form." This term comes from Control Theory but its exact meaning is not important to us. To see how this method of generating a state space model works, consider the third order differential transfer function:

$$H(s) = \frac{Y(s)}{U(s)} = \frac{b_0 s^2 + b_1 s + b_2}{s^3 + a_1 s^2 + a_2 s + a_3}$$

We start by multiplying by Z(s)/Z(s) and then solving for Y(s) and U(s) in terms of Z(s). We also convert back to a differential equation.

$$Y(s) = \left(b_0 s^2 + b_1 s + b_2\right) Z(s) \qquad y = b_0 \ddot{z} + b_1 \dot{z} + b_2 z$$

$$U(s) = \left(s^3 + a_1 s^2 + a_2 s + a_3\right) Z(s) \qquad u = \dddot{z} + a_1 \ddot{z} + a_2 \dot{z} + a_3 z$$

We can now choose z and its first two derivatives as our state variables

$$q_1 = z \qquad \dot{q}_1 = \dot{z} = q_2$$

$$q_2 = \dot{z} \qquad \dot{q}_2 = \ddot{z} = q_3$$

$$q_3 = \ddot{z} \qquad \dot{q}_3 = \dddot{z} = u - a_1 \ddot{z} - a_2 \dot{z} - a_3 z$$

$$= u - a_1 q_3 - a_2 q_2 - a_3 q_1$$

Now we just need to form the output

$$y = b_0\ddot{z} + b_1\dot{z} + b_2 z$$
$$= b_0 q_3 + b_1 q_2 + b_2 q_1$$

From these results we can easily form the state space model:

$$\dot{\mathbf{q}} = \mathbf{A}\mathbf{q} + \mathbf{B}u = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_3 & -a_2 & -a_1 \end{bmatrix}\mathbf{q} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$
$$y = \mathbf{C}\mathbf{q} + Du = \begin{bmatrix} b_2 & b_1 & b_0 \end{bmatrix}\mathbf{q} + 0 \cdot u$$

In this case, the order of the numerator of the transfer function was less than that of the denominator. If they are equal, the process is somewhat more complex. A result that works in all cases is given below; the details are here. For a general n^{th} order transfer function:

$$H(s) = \frac{Y(s)}{U(s)} = \frac{b_0 s^n + b_1 s^{n-1} + \cdots + b_{n-1} s + b_n}{s^n + a_1 s^{n-1} + \cdots + a_{n-1} s + a_n}$$

the controllable canonical state space model form is

$$\dot{\mathbf{q}} = \mathbf{A}\mathbf{q} + \mathbf{B}u; \quad \mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \cdots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & -a_1 \end{bmatrix}; \qquad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$y = \mathbf{C}\mathbf{q} + Du \quad \mathbf{C} = \begin{bmatrix} b_n - a_n b_0 & b_{n-1} - a_{n-1} b_0 & \cdots & b_2 - a_2 b_0 & b_1 - a_1 b_0 \end{bmatrix} \quad D = b_0$$

MATLAB PROGRAM:

num=input('enter the numerator of the transfer function')

den=input('enter the denominator of the transfer function')

ss(tf(num,den))

EXAMPLE:

Obtain the state model from the transfer function given below:

$T(s) =$

PROCEDURE:

- Type the program in the MATLAB editor that is in M-file.
- Save and run the program.
- Give the required inputs in the command window of MATLAB in matrix format.
- The command ss(tf(num,den)) converts the given transfer function into a state model.
- Note down the output obtained in MATLAB.
- The state model is also obtained theoretically.
- Both the state models are compared.

THEORETICAL CALCULATIONS

Enter the transfer function

A =

B =

C =

D =

RESULT:

8. ZEROS AND POLES FROM STATE MODEL

AIM:

To obtain poles and zeros from a given state model using MATLAB

APPARATUS:

Software: MATLAB

THEORY:

Let's say we have a transfer function defined as a ratio of two polynomials:

$H(s)=\frac{N(s)}{D(s)}$

Where N(s) and D(s) are simple polynomials.

Zeroes are the roots of N(s) (the numerator of the transfer function)obtained by setting N(s)=0 and solving for s. Poles are the roots of D(s) (the denominator of the transfer function),obtained by setting D(s)=0 and solving for s.

The state space model represents a physical system as n first order coupled differential equations. This form is better suited for computer simulation than an n^{th} order input-output differential equation.

The general vector-matrix form of state space model is:

$$\dot{X} = AX + BU$$

Where,

X = state vector

U = input vector

A = n x n matrix

B = n x 1 matrix

The output equation for the above system is,

$$Y = CX + DU$$

MATLAB PROGRAM:

```
A=input('enter matrix A')

B=input('enter matrix B')

C=input('enter matrix C')

D=input('enter matrix D')

[z,p,k]=ss2zp(A,B,C,D)
```

EXAMPLE:

A=

B=

C=

D=

PROCEDURE:

- the MATLAB window and open a new MATLAB editor.
- Write the MATLAB program in the MATLAB editor.
- Open Save and run the MATLAB program.
- The state model is given as input and entered in matrix format.
- The syntax "[z,p,k]=ss2zp(A,B,C,D)" solves the given state model entered in matrix format as input and gives the output in the form of poles, zeros and gain.
- This syntax transforms the given state model into poles, zeros and gain.
- Note down the output zeros, poles and gain obtained practically by using the syntax "[z,p,k]=ss2zp(A,B,C,D)".
- Now find the poles, zeros and gain theoretically for the given state model

- Compare the theoretically obtained poles, zeros and gain from the given state model with the one obtained practically. Write the result based on the comparison between theoretical and practical result.

THEORETICAL CALCULATIONS:

RESULT:

9. TRANSFER FUNCTION OF A DC MOTOR/GENERATOR

(a) TRANSFER FUNCTION OF A DC MOTOR

AIM:

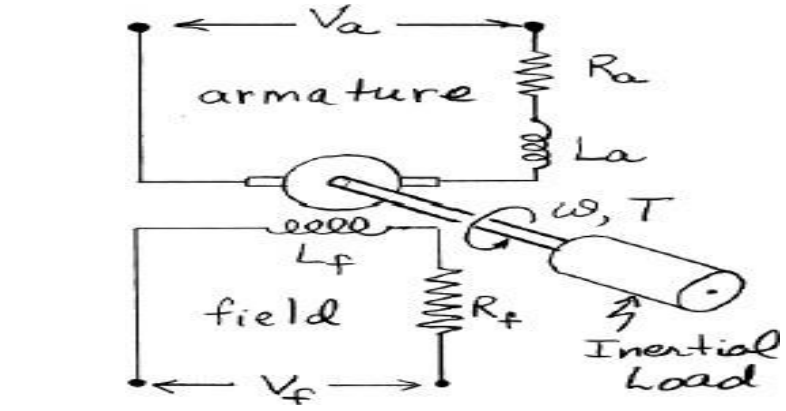
To determine the transfer function of a DC Motor.

APPARATUS:

S.No	Name of Equipment
1	Experimental Kit.
2	Connecting Probes.

THEORY:

The transfer function of a DC Motor is studied (in general)



Emf equation is

$$e_a = E_b + I_a(R_a) + L_a \frac{di_a}{dt}$$

Using Laplace Transform,

$$E_a(s) = E_b(s) + I_a(s)[R_a + sL_a]$$

$$\Rightarrow I_a(s) = \frac{E_a(s) - E_b(s)}{R_a + sL_a}$$

$T_m = K_m' K_f I_f I_a$ Becomes

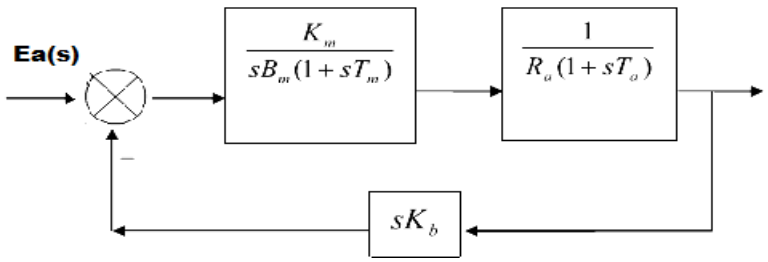
$$T_m = K_m' K_f I_f [\frac{E_b - K_b s\theta_m(s)}{R_a + sL_a}] \longrightarrow (1)$$

$$\Rightarrow T_m = (J_m s^2 + sB_m)\theta_m(s) \longrightarrow (2)$$

$$H(s) = sK_b$$

$$\therefore G(s) = \frac{K_m}{sR_a B_m (1 + sT_m)(1 + sT_a)}$$

Block Diagram:



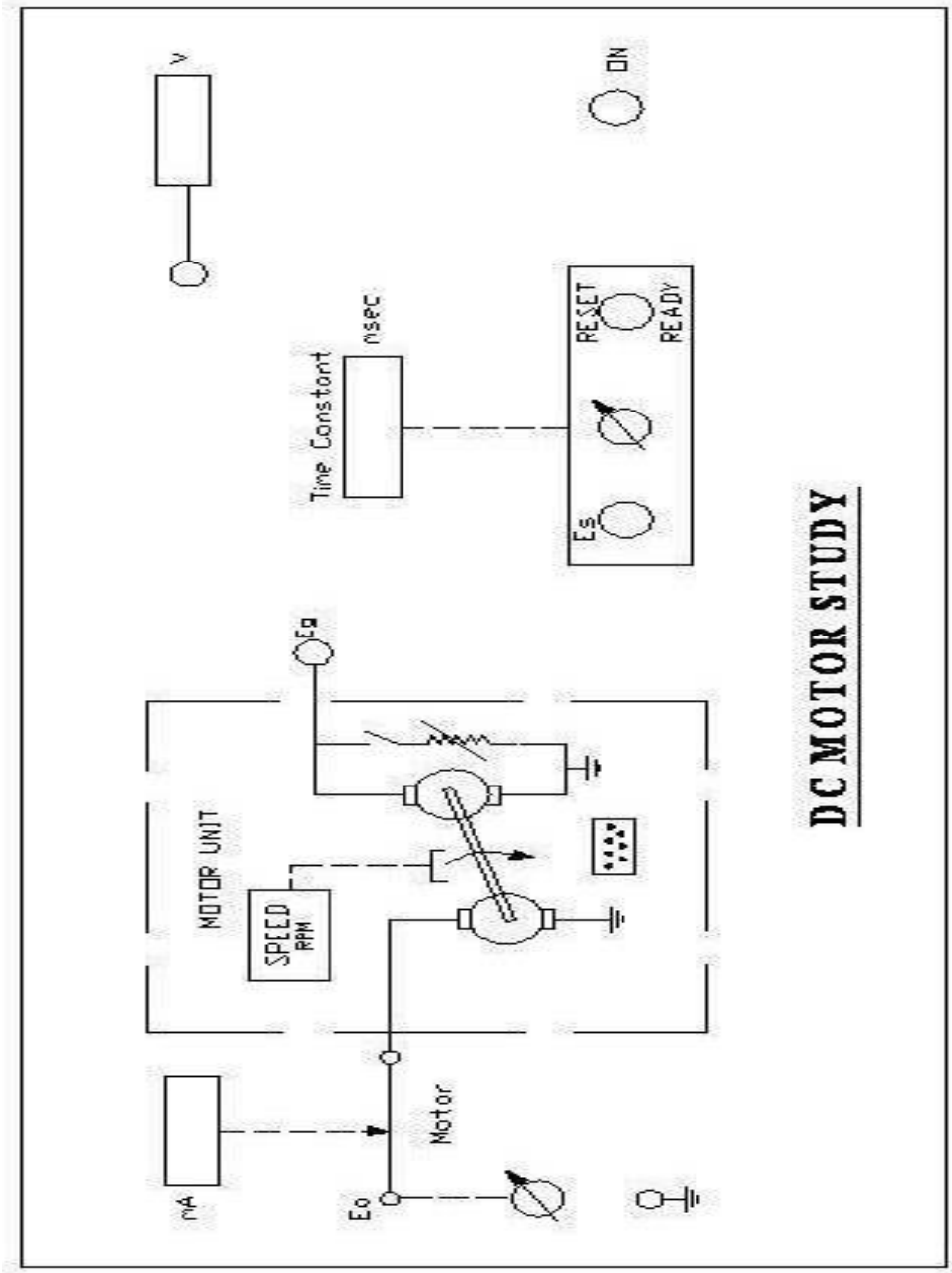
PROCEDURE:

- Type the program in the MATLAB editor that is in M-file.
- Save and run the program.
- Give the required inputs in the command window of MATLAB .

MATLAB PROGRAM:

```
J=0.01;
B=0.1;
K=0.01;
R=1;
L=0.5;
S=tf('s');
P_motor=K/((J*S+B)*(L*S+R)+K^2);
zpk(P_motor);
```

HARDWARE UNIT:



PROCEDURE:

1. Connections are made as per circuit diagram. Rheostat of Generator Field circuit is kept in minimum output position and that of Motor Field in minimum resistance position.

2. Set "MOTOR" switch to onset "RESET" to "RESET" set "LOAD" switch to "O" position.Vary Ea in small steps and take readings of Ia & Speed.

3. Set Es to 63.2% of Eg measured above. This is the Generator Voltage at which the counter will stop counting.

4. Switch 'OFF' the Motor. Set 'RESET' switch to 'READY'. Now switch the Motor 'ON'. Record the counter reading as time constant milli seconds.

5. Repeat the above procedure with Ea = 10V & 12V and tabulate the readings.Different constant values are calculated and also calculate the transfer function.

OBSERVATIONS:

Ra = 3Ω J = 0.0456 Kg-m^2/rad

S.No	Ea (V)	Ia (mA)	N(RPM)	Eg (V)	Es = 0.632 x Eg (V)	Ta (mS)	ω = 2πN/60 (rad/sec)	Eb = Ea-IaRa (V)	T = 9.55xEbIa/N (N-m)	B = T/ω (N-m)/ (rad/sec)	Kt = T/Ia (N-m)/A	Kb = Eb/ω	Tm = J/B

PRECAUTIONS:

1. Avoid loose connections.
2. Take the readings without any parallax errors.

RESULT:

(b) TRANSFER FUNCTION OF A DC GENERATOR

AIM: To determine the transfer function of DC. Generator.

APPARATUS:

S.No	Name of the Equipment	Type	Range	Quantity
1	Voltmeter	MC	(0-250)V	1
2	Voltmeter	MI	(0-150)V	1
3	Ammeter	MI	(0-2)A	1
4	Field Excitation			
5	Rheostat		400Ω,1.7A	1
6	Rheostat		298Ω,1.5A	1
7	Variac		240/(0-270)v,5A	1
8	Tachometer	Digital		1
9	DMM			1

CIRCUIT DIAGRAM:

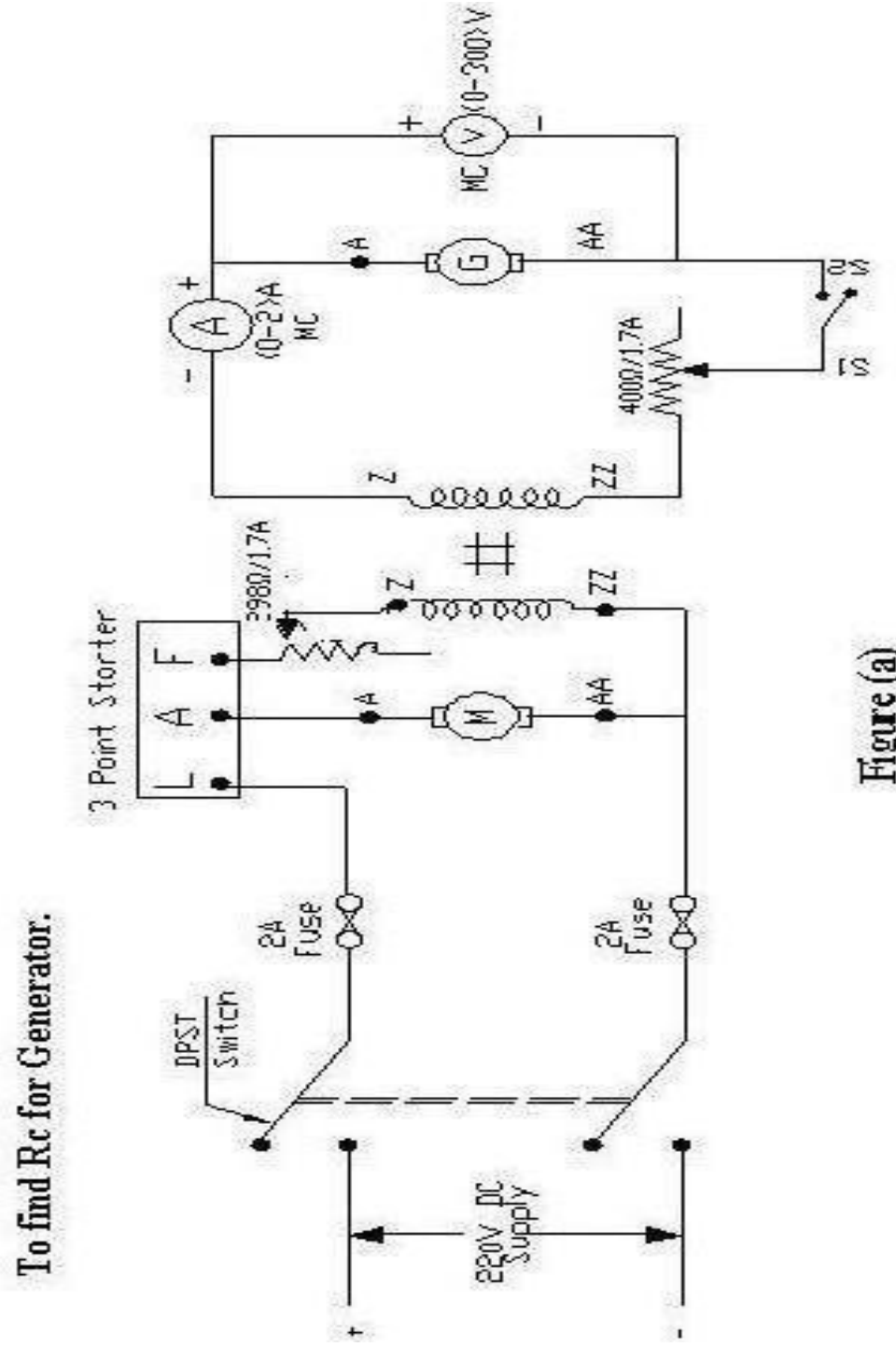


Figure (a)

CIRCUIT DIAGRAMS:

To Find Rf for Generator.

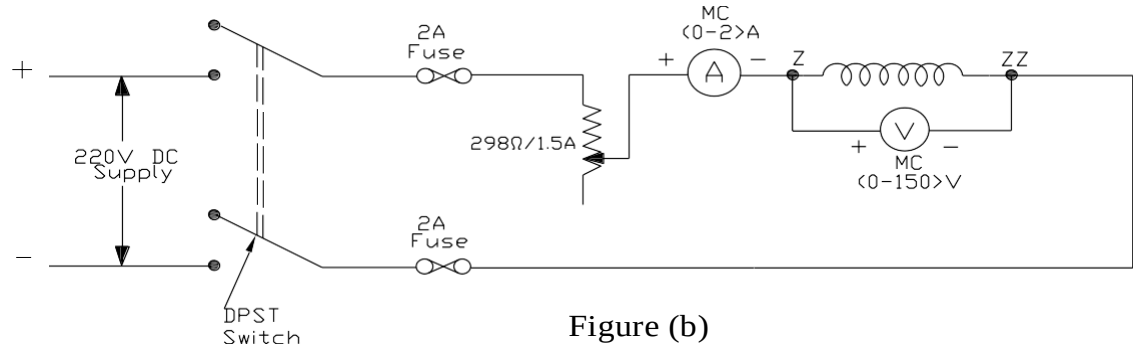


Figure (b)

To Find Lf for Generator.

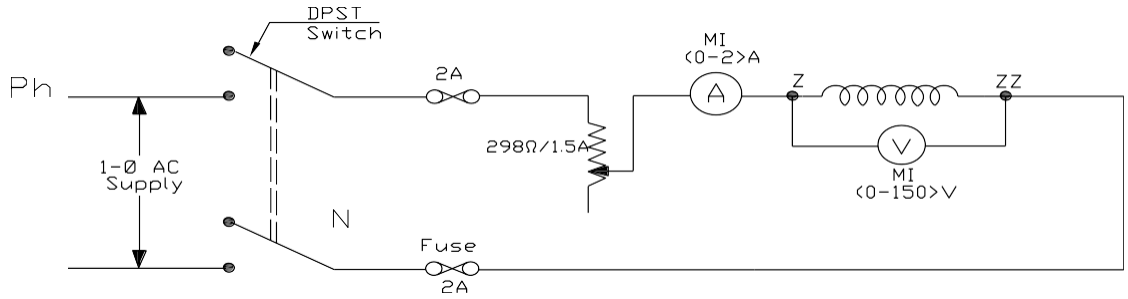


Figure (c)

PROCEDURE:

1. Give connections as per the circuit diagram fig – a.
2. Bring the motor to rated speed by exciting armature resistance and increasing field resistance.
3. By using field excitation apply it at steps of 0.05 Amps and note down the value of V & I.

4. Draw a graph between V_o and If and find Critical Resistance R_c =OB/OA = K_g
5. Now connect the circuit as per fig – b.
6. By varying the rheostat at different steps note down the values of V and I_f hence R_f = V/I_f.
7. Now connect the circuit as per circuit diagram fig – c.
8. By varying auto transformer note the values of V and I_f which gives Z_f = V/I_f and

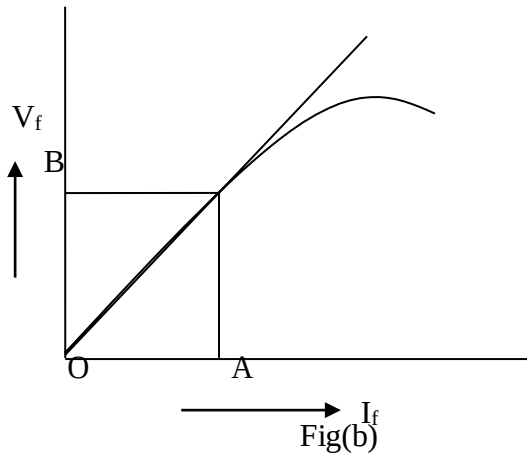
   Find X_f = sqrt(Z_f^2-R_f^2), hence L_f = $X_f/2\pi f$.

9. Then write the transfer function as T.F = **K_G/(1+ST)**

PRECAUTIONS:

- Avoid loose and wrong connections.
- The shunt field Rheostat & armature Rheostat of motor is kept as min. and max. position before starting the experiment.
- While doing OCC experiment maintain speed as constant.
- Readings should be taken without any parallax errors.

MODEL GRAPH:



Fig(b)

OBSERVATIONS:

For Fig – (a)

S. No.	V(volts)	I(amp)

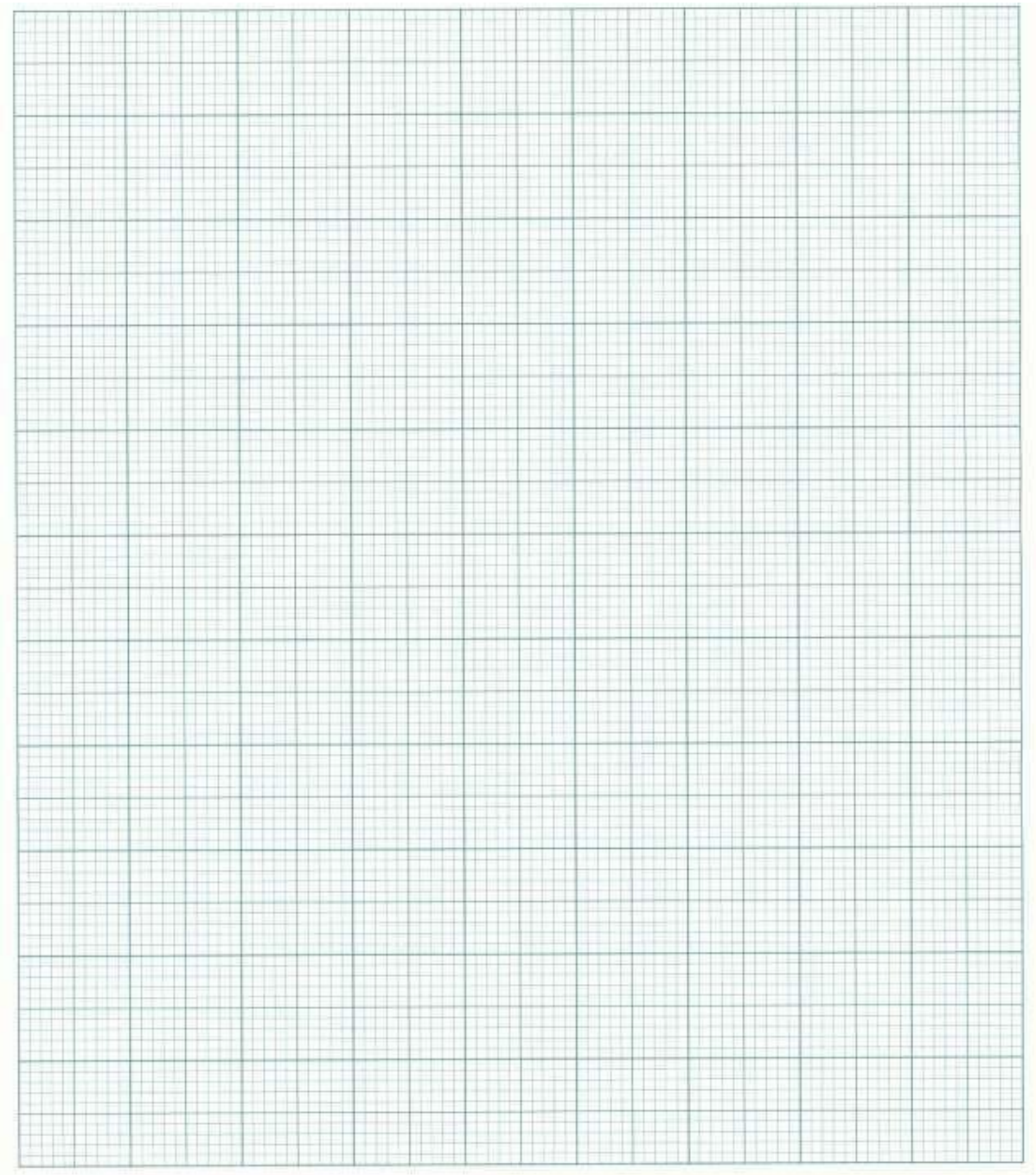
For Fig – (b)

S. No.	V_f(volts)	I_f(amp)	R_f(Ohms)

For Fig – (c)

S. No.	V_f(volts)	I_f(amp)	Z_f(Ohms)

GRAPH:



RESULT:

10. TIME RESPONSE OF SECOND ORDER SYSTEM

AIM:

To obtain the time response of a given second order system.

APPARATUS:

Software: MATLAB

THEORY:

The time response has utmost importance for the design and analysis of control systems because these are inherently time domain systems where time is independent variable. During the analysis of response, the variation of output with respect to time can be studied and it is known as time response. To obtain satisfactory performance of the system with respect to time must be within the specified limits. From time response analysis and corresponding results, the stability of system, accuracy of system and complete evaluation can be studied easily.

Due to the application of an excitation to a system, the response of the system is known as time response and it is a function of time. The two parts of response of any system:

(i) Transient response

(ii) Steady-state response.

Transient response: The part of the time response which goes to zero after large interval of time is known as transient response.

Steady state response: The part of response that means even after the transients have died out is said to be steady state response.

The total response of a system is sum of transient response and steady state response:

$C(t)=C_{tr}(t)+C_{ss}(t)$

Time response of second order control system:

A second order control system is one wherein the highest power of 's' in the denominator of its transfer function equals 2.

Transfer function is given by:

$$TF = \frac{\omega^2_n}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

ω_n—is called natural frequency of oscillations.

$\omega_d=\omega_n\sqrt{1-\delta^2}$ is called damping frequency oscillations.

δ −Affects damping and called damping ratio.

$\delta\omega n$ – is called damping factor or actual damping or damping coefficient.

MATLAB PROGRAM:

```
wn=input('enter value of undamped natural frequency')

z=input('enter value of damping ratio')

n=[wn*wn]

p=sqrt(1-z^2)

wd=wn*p

h=[p/z]

k=atan(h)

m=pi-k;

tr=[m/wd]

tp=[pi/wd]

q=z*wn

ts=[h/q]

r=z*pi

f=[r/p]

mp=exp(-f)

num=[0 0 n]

den=[1 2*z*wn n]

s=tf(num,den)

hold on

step(s)

impulse(s)

hold off
```

PROCEDURE:

1.Time response of the system is being found when we give the values of natural undamped frequency and damping ratio.

2. When we give these values first rise time, peak time, peak overshoot, transfer function are being calculated.

3. Then “step(s)” And “impulse(s)” generates time response of the system.

5. The hold function determines whether new graphics object are added to the graph or replaces objects in the graph.

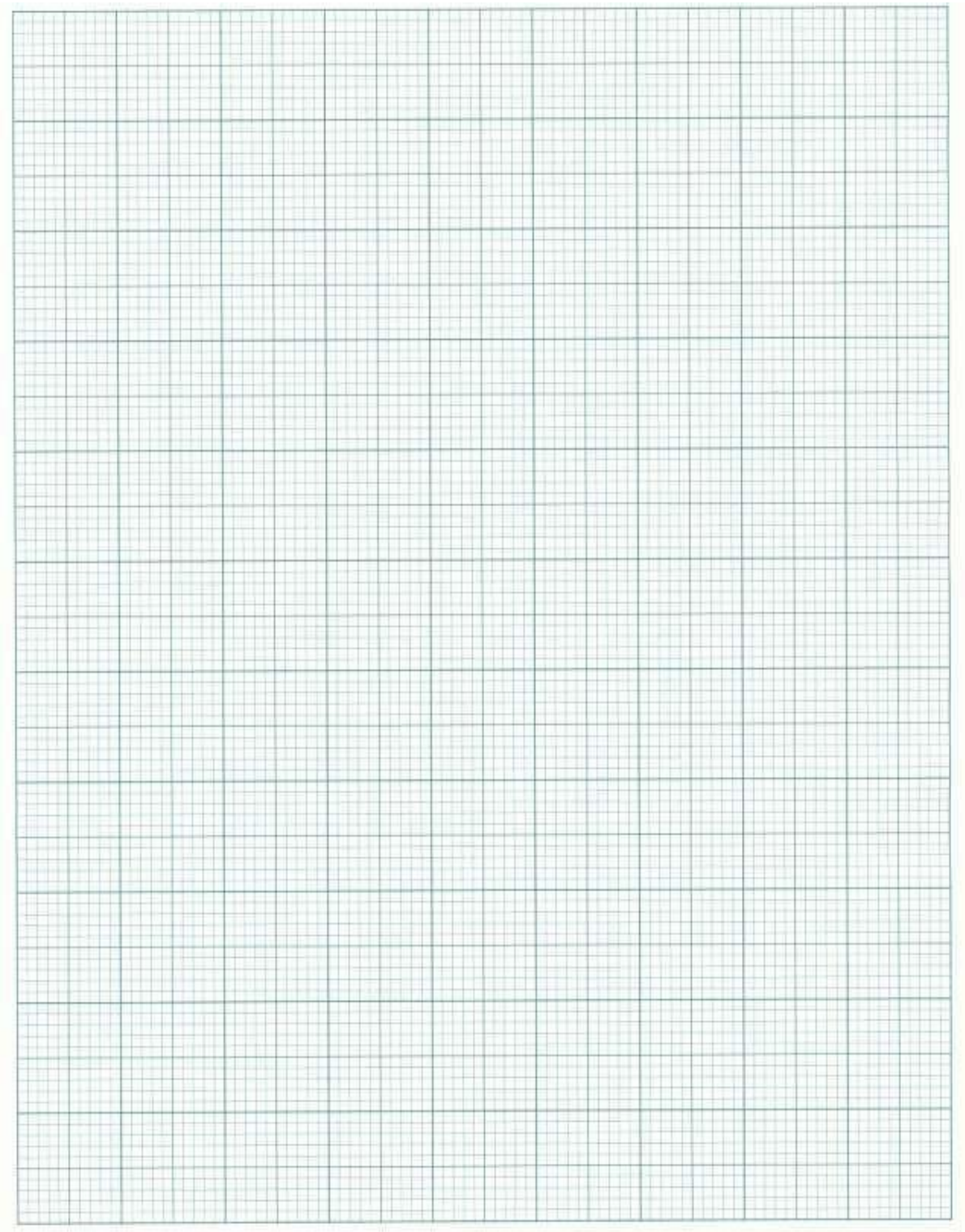
6. Hold on retains the current plot and certain axes properties so that subsequent graphing command adds to the existing graph.

7. Hold off resets axes properties to their defaults before drawing new plots. Hold off is the default.

EXAMPLE:

THEORETICAL CALCULATIONS:

GRAPH:



Hardware:

APPARATUS:

S.No	Name of Equipment
1	Linear Simulator Kit.
2	CRO and Connecting Probes.

PROCEDURE:

1. Patch the connections in front panel as shown on figure.
2. Switch on the supply to the unit.
3. Square wave of amplitude Vpp is given as Step input to the system at low frequency.
4. Adjust the value of proportional band such that there is a sustained oscillation for each step input.
5. At this instant note down the values of delay time t_d, rise time t_r, peak time t_p, stalling time t_s and peak overshoot Mp.
6. Draw the response of System on graph.

PRECAUTIONS:

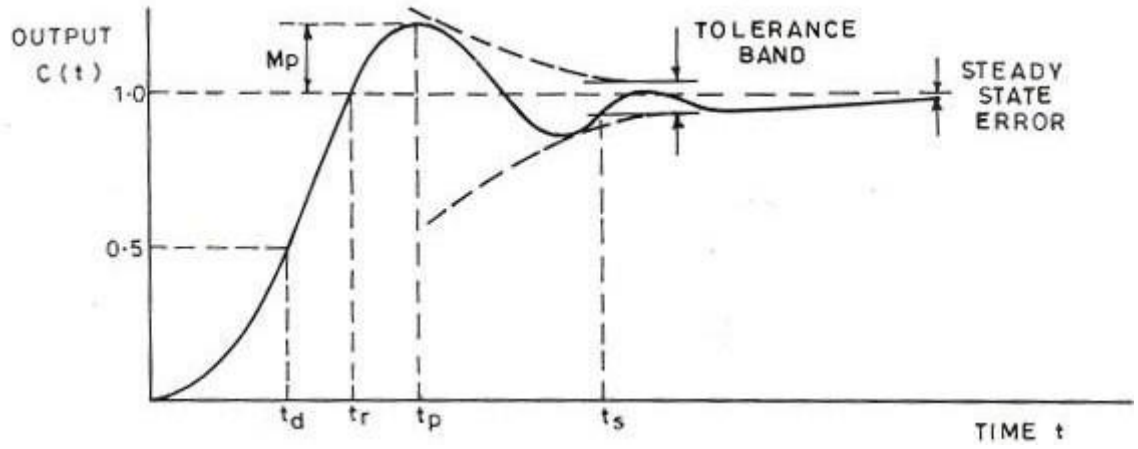
1. Avoid loose connections.
2. Note down the readings with out any parallax errors.

OBSERVATIONS:

S.No	Mp(ns)	Tr(ms)	Tp(ms)	Ts(ms)	ξ	ωn

Model graph:-

Second order system



RESULT:

11.DC SERVO MOTOR.

AIM:

To determine the Speed – Torque Characteristics of DC Servo motor.

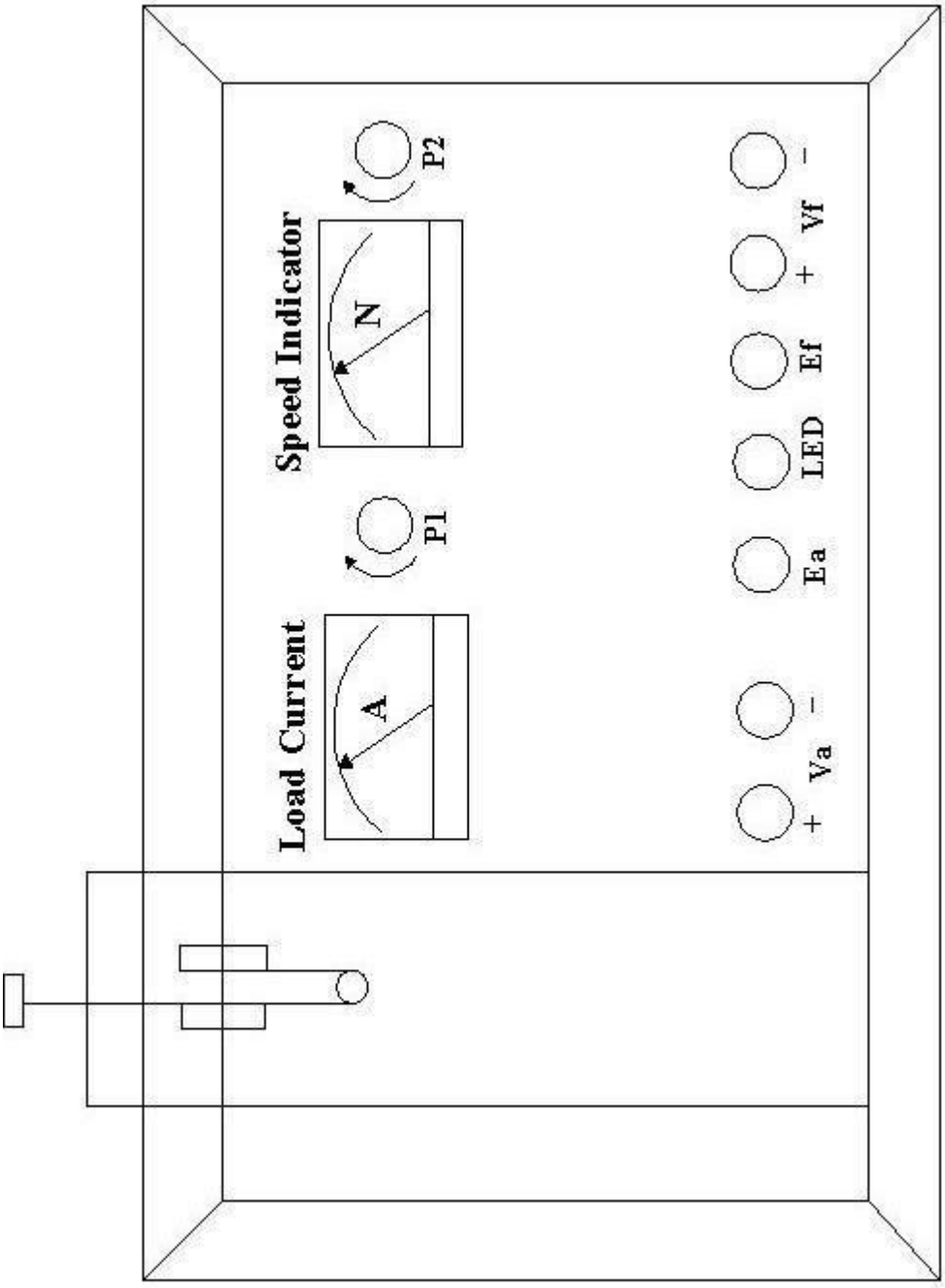
APPARATUS:

S.No	Name of Equipment	Range	Type	Quantity
1	Experimental Kit			1
2	Digital Multi Meter	(0-25)V		1
3	Patch Cords			

PROCEDURE:

1. Before supply is given to the kit short the terminals located near the supply switch with the help of patch cords.
2. Ensure the pot P1 and P2 are in maximum, in full anti clock wise direction.
3. Switch ON the power supply.
4. Connect a Digital Multi Meter (0 – 25) V across the terminals marked as armature to measure the armature voltage for a fixed value of a field voltage V_f say at 20V.
5. Adjust the pot P1 so that V_a = 15V and pot P2 so that V_f = 20V.
6. Calculate the torque using (T1 – T2) x 3.5 gm – cm.
7. Now for V_a =15V, 17.5V & 20V for a fixed V_f = 20V repeat the above steps.
8. From the data obtained, plot the Speed – Torque characteristics by taking Speed on X - axis and Torque on Y- axis.

BLOCK DIAGRAM:

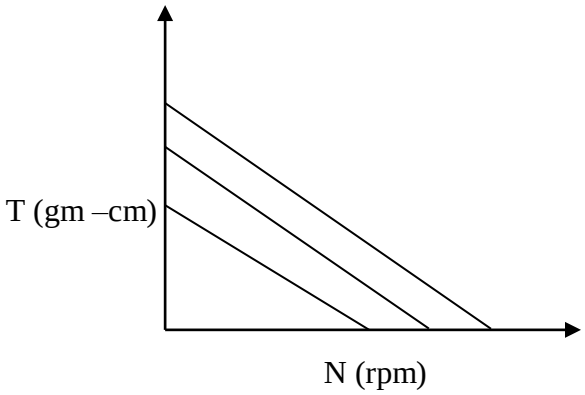


OBSERVATIONS:

At V_F = 20V

S.No	N(RPM)	T1(gm)	T2(gm)	T =(T1-T2)3.5 (gm-cm)	I_a (A)	V_a (V)

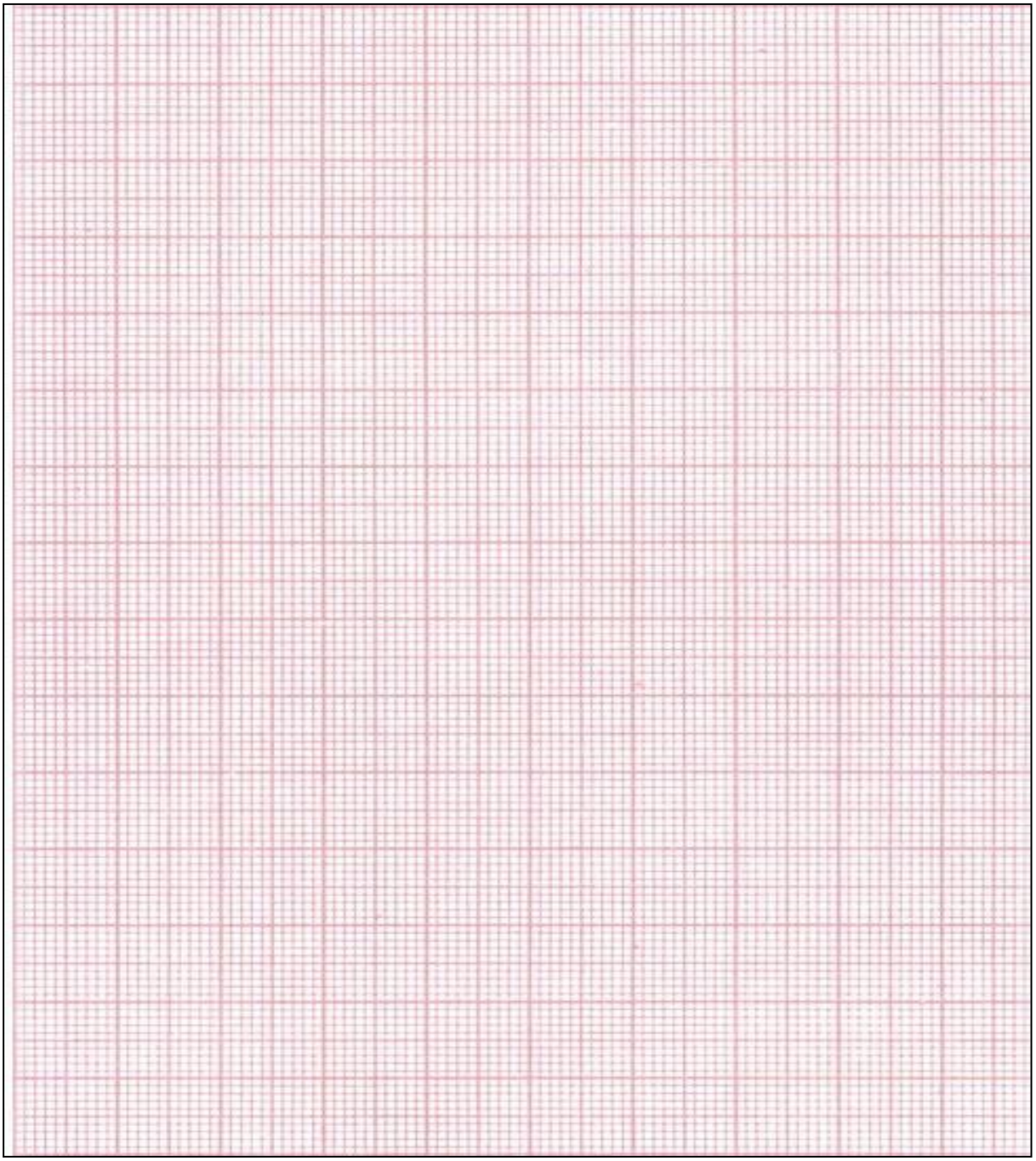
MODEL GRAPHS:



PRECAUTIONS:

1. Before switch ON the supply the side terminals must be shorted.
2. Avoid the over loading on the Servomotor.
3. If the Servomotor is not started when supply is given, then slowly vary the pot.

OUTPUT WAVE FORMS



RESULT:

12. PID CONTROLLER

AIM:

To control the closed loop system using PID controller.

APPARATUS:

Software: MATLAB

THEORY:

PID controllers are commercially successful and widely used as controllers in industries. For example, in a typical paper mill there may be about 1500 controllers and out of these 90 percent would be PID controllers. The PID controller consists of a proportional mode, an Integral mode and a Derivative mode. The first letters of these modes make up the name PID controller. Depending upon the application one or more combinations of these modes are used. For example, in a liquid control system where we want zero steady state error, a PI controller can be used and in a temperature control system where zero stead state error is not specified, a simple P controller can be used.

The equation of a PID controller in time-domain is given by

$$m(t) = k_c e(t) + \frac{k_c}{T_i}\int_0^t e(t)dt + k_c T_d \frac{de(t)}{dt}$$

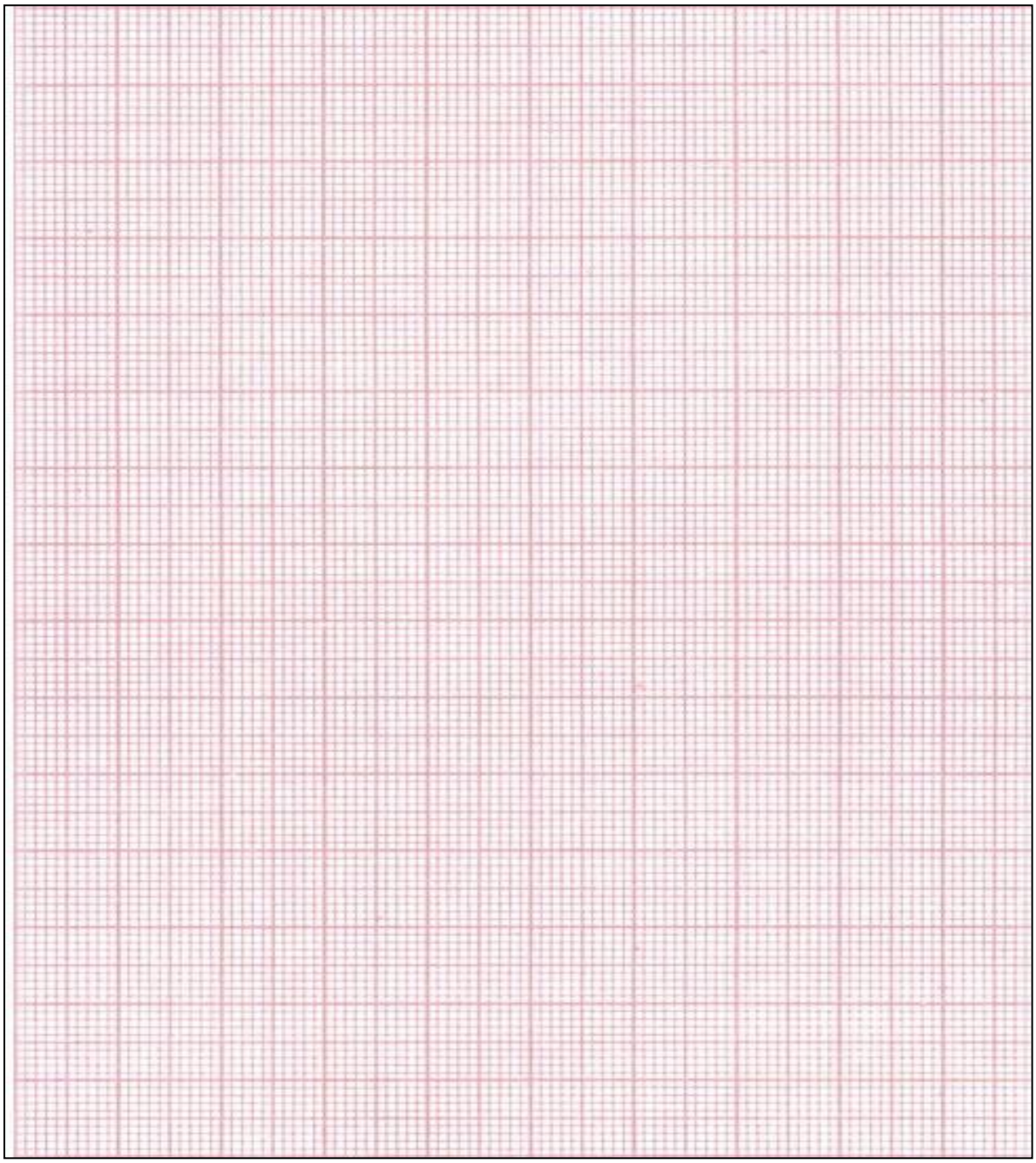
Where k_c is the proportional gain, $T_i$is the integral reset time and T_d is the derivative time of the PID controller, m(t) is the output of the controller and e(t) is the error signal given by e(t)=r(t)-c(t).The controller used here is a PID controller represented by a block PID and the system or plant is represented by G(s). R(s) and D(s) are reference signal and disturbance signal respectively. Y(s), E(s) and M(s) are the output, error and controller output of the system respectively. For the purpose of good control, we require the system output Y(s) to track any reference signal F(s) and at the same time reject or suppress deviation due to the disturbance signal D(s). Hence the PID controller can realize this objective.

EXAMPLE:

MATLAB PROGRAM:

PROCEDURE:

THEORITICAL CALCULATIONS:



RESULT:

13. CHARACTERISTICS OF SYNCHROS

AIM: -

To study of the operation of Synchro transmitter and Synchro receiver pair.

APPARATUS: -

S.No	Name of Equipment
1	Experimental Kit
2	Connecting Probes.

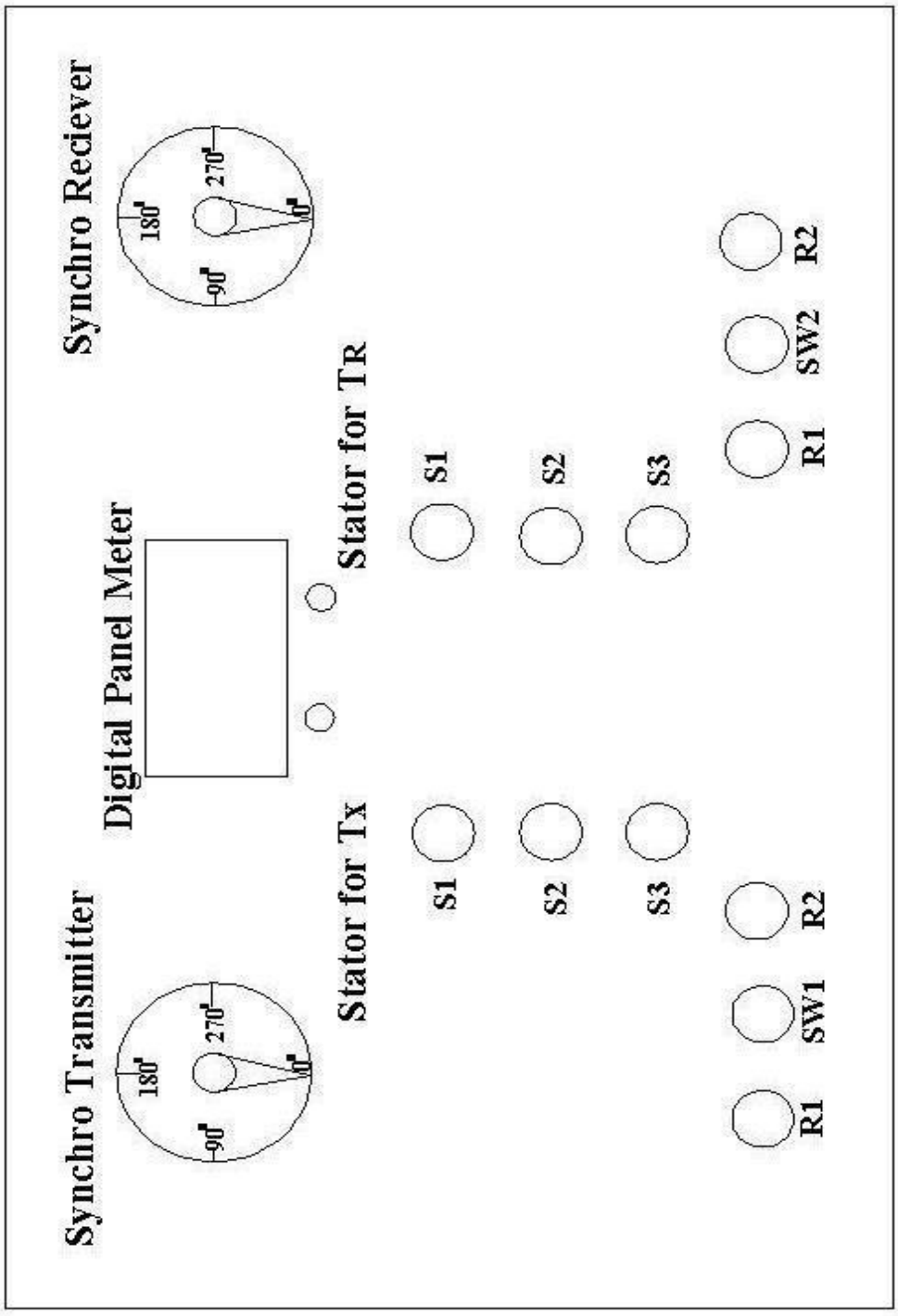
PROCEDURE:

1. Connect the circuit as per circuit diagram.
2. Short R1 and R2 terminals of transmitter and receiver.
3. Short (S1 – S3), (S3 – S2), (S3 – S2) of receiver and transmitter with the help of patch cords.
4. Switch ON the power supply.
5. As the power is switched ON transmitter will be at 0^o and receiver will be at 120^o.
6. Vary the transmitter angle & observe the corresponding changes in receiver angle.
7. Repeat the above steps for various degrees of phase shift.
8. Draw the graphs for transmitter and receiver separately.

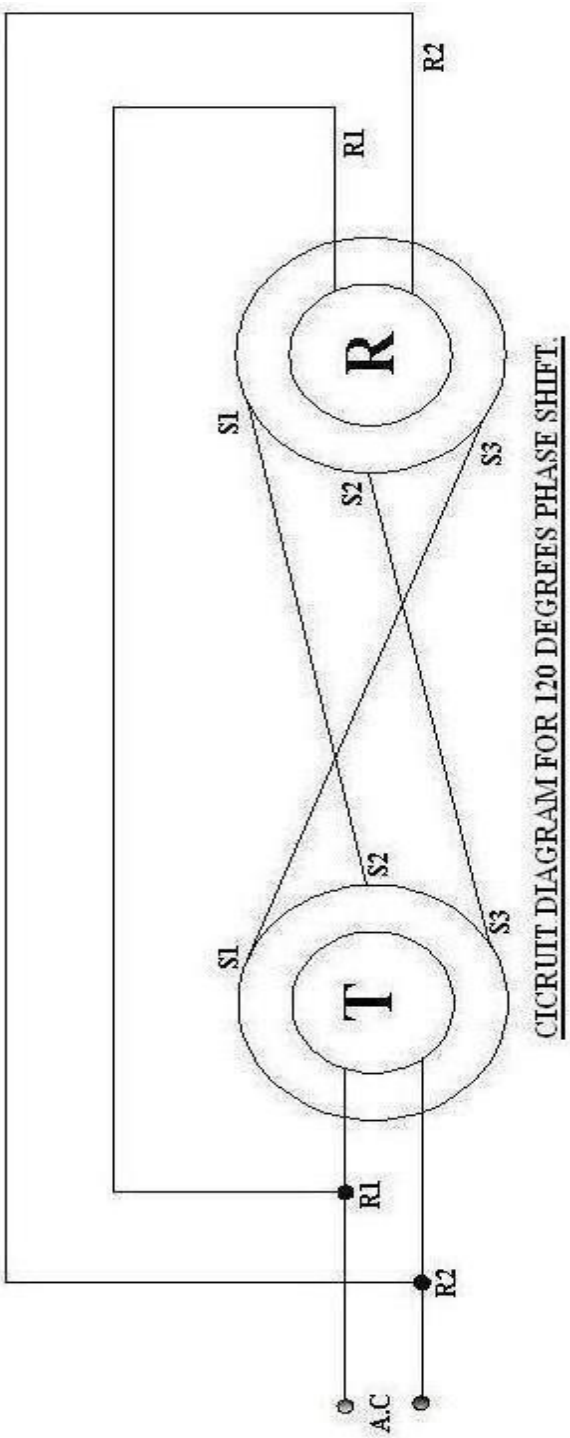
PRECUTIONS:

1. Take readings without any parallax errors.
2. Avoid loose connections.

BLOCK DIAGRAM:

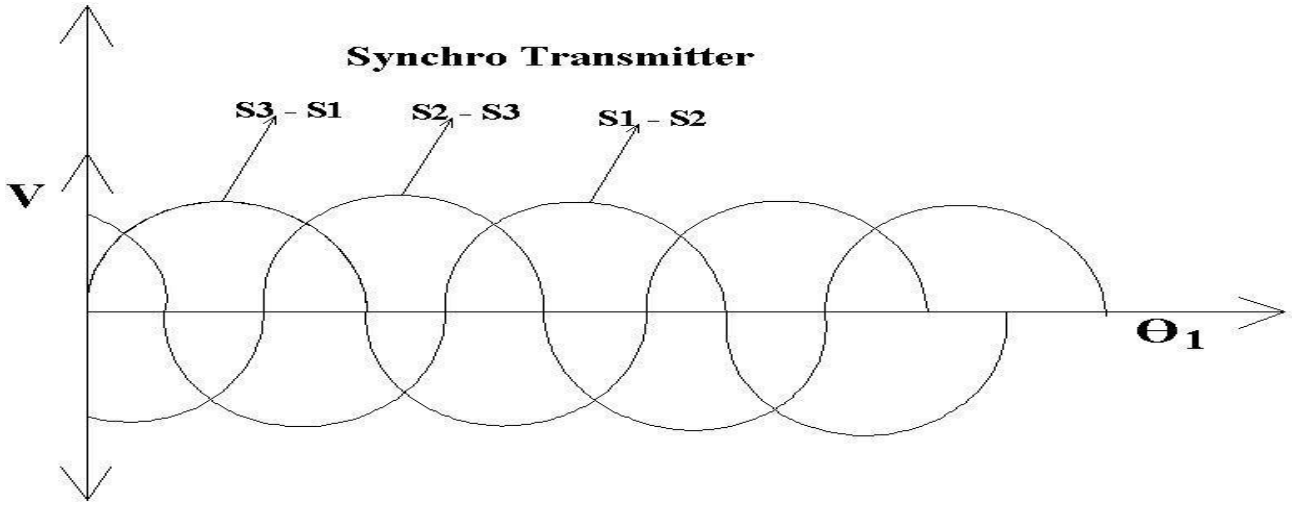


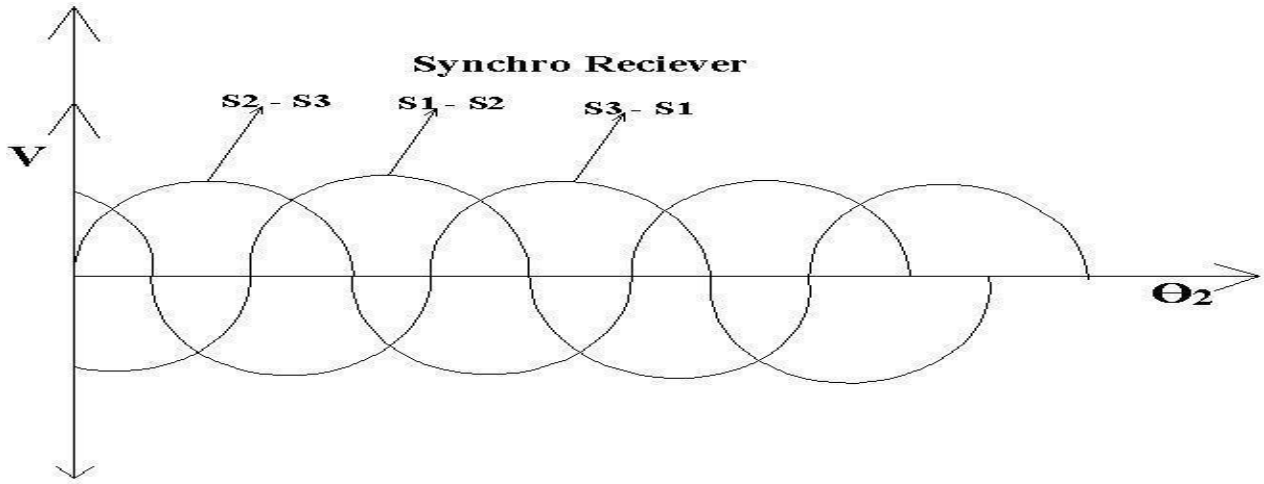
CIRCUIT DIAGRAM:

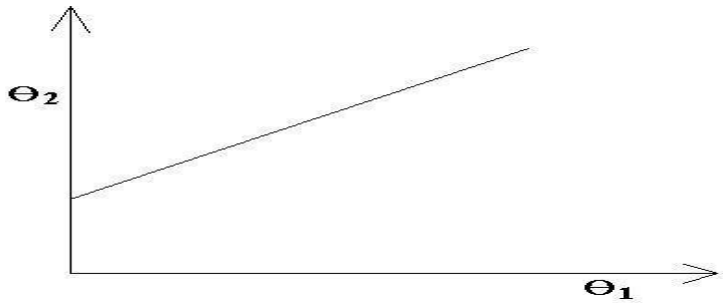


CICRUIT DIAGRAM FOR 120 DEGREES PHASE SHIFT.

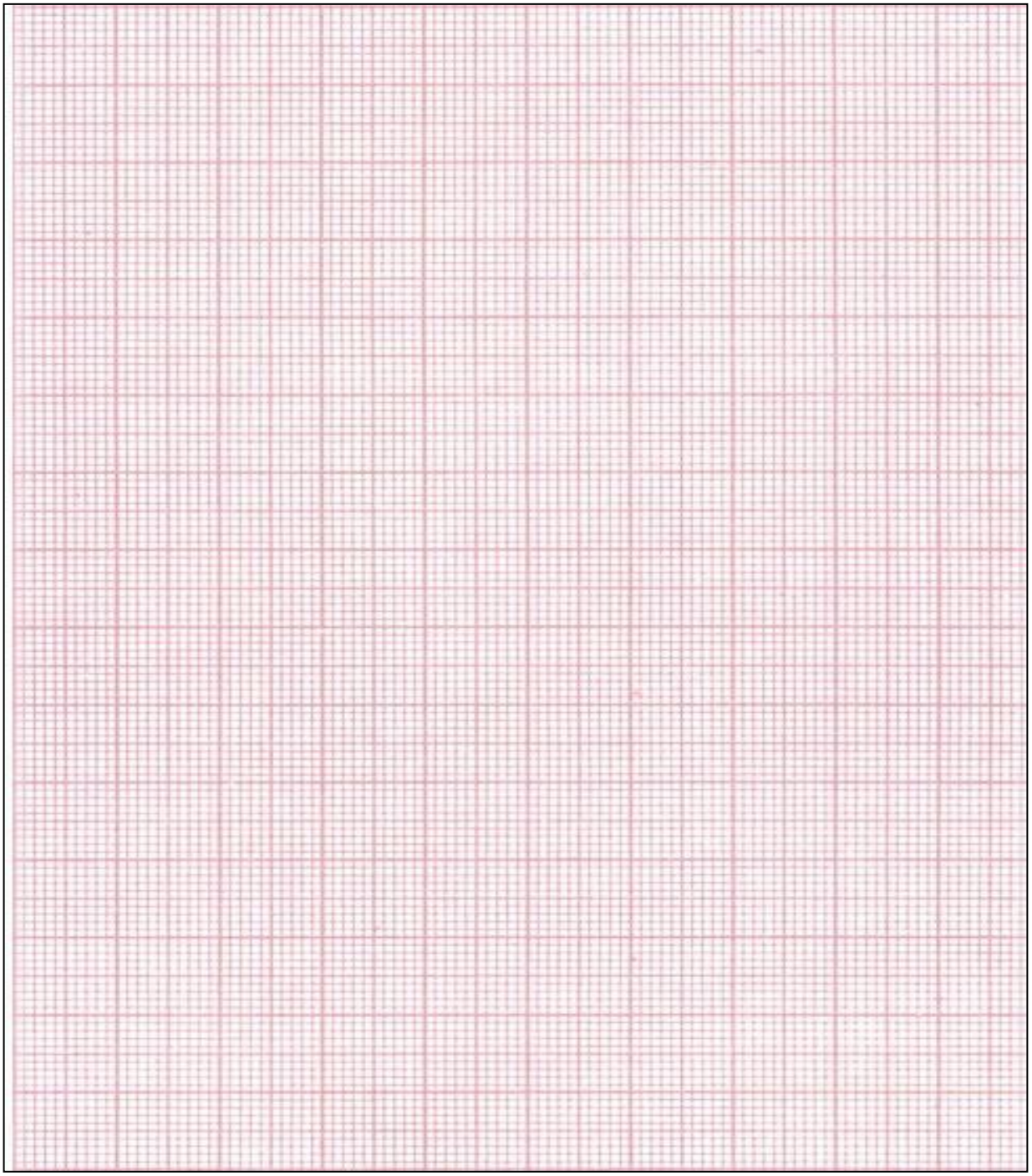
MODEL GRAPHS:







OUTPUT WAVE FORMS:



RESULT:

14. LAG- LEAD COMPENSATOR

AIM:

To design lag-lead compensator using closed loop system.

APPARATUS:

Software: MATLAB

Hardware:

S.No	Name of Equipment	Type	Range	Quantity
1	Lag and Lead Network Kit			1
2	CRO			1
3	Connecting Probes			

THEORY:

A **lead–lag compensator** is a component in a control system that improves an undesirable frequency response in a feedback and control system. It is a fundamental building block in classical control theory.

lagts delays a financial time series object by a specified time step.

newfts = lagts(oldfts) delays the data series in oldfts by one time series date entry and returns the result in the object newfts. The end will be padded with zeros, by default.

newfts = lagts(oldfts, lagperiod) shifts time series values to the right on an increasing time scale. lagts delays the data series to happen at a later time. lagperiod is the number of lag periods expressed in the frequency of the time series object oldfts. For example, if oldfts is a daily time series, lagperiod is specified in days. lagts pads the data with zeros (default).

newfts = lagts(oldfts, lagperiod, padmode) lets you pad the data with an arbitrary value, NaN, or Inf rather than zeros by setting padmode to the desired value.

leadts advances a financial time series object by a specified time step.

newfts = leadts(oldfts) advances the data series in oldfts by one time series date entry and returns the result in the object newfts. The end will be padded with zeros, by default.

newfts = leadts(oldfts, leadperiod) shifts time series values to the left on an increasing time scale. leadts advances the data series to happen at an earlier time. leadperiod is the number of lead periods expressed in the frequency of the time series object oldfts. For example, if oldfts is a daily time series, leadperiod is specified in days. leadts pads the data with zeros (default).

newfts = leadts(oldfts, leadperiod, padmode) lets you pad the data with an arbitrary value, NaN, or Inf rather than zeros by setting padmode to the desired value.

MATLAB PROGRAAM:

```
num=input('enter the numerator')
den=input('enter the denominator')
h=tf(num,den)
kv=input('enter velocity error')
phm=input('enter the phase margin')
h1=tf([1 0],[1])
m=dcgain(h1*h)
k=kv/m
g=k*h
[mag phase w]=bode(g)
[gm pm wcg wcp]=margin(g)
e=input('enter margin of safety')
bode(g)
theta=phm+e;
bm=theta-180;
wcm=('enter wcm corresponding to bm')
a=input('enter gain corresponding to wcm')
beta=10^(a/20)
w21g=wcm/4
tou=1/w21g
w11g=1/(beta*tou)
g1=(h1+w21g)/(h1+w11g)
theta1d=theta1d*(pi/180);
alpha=(1-sin(theta1d)/(1+sin(theta1d)))
a=-20*log10(1/sqrt(alpha))
wcm1d=input('enter the value of wcmld corresponding to gain a1')
w11d=wcm1d*sqrt(alpha)
w21d=wcm1d/sqrt(alpha)
g3=tf([1/w11d1],[1/w21d 1])
g4=g3*g1*g
[mag3 phase3 w3]=bode(g2)
bode(g)
hold
bode(g4)
```

PROCEDURE:

- Numerator of the given transfer function is assigned to num
- Denominator of the transfer function is assigned to den
- The value of the static velocity constant is assigned to the kv
- Margin of the safty is assigned to e
- Plot is obtained by in-build function bode()

- Wcm values assigned to wcm which is obtained from above bode plot
- The gain corresponding to wcm is assigned to a
- If wcg1>wcg orr wcp1>wcp, the network is compensated otherwise it is not compensated.

THEORETICAL CALCULATIONS:

Transfer function

For complex w,

Gain =

MAGNITUDE PLOT:

Factor	Corner frequency	Log magnitude in db =-20log(factor)

PHASE PLOT: Φ=

W(rad/sec)	Φ

New phase margin required is =

Frequency corresponding to

PROCEDURE:

Lag Network:

1. Connect the circuit as per circuit diagram.
2. A Sinusoidal input of say 2V p-p is applied with a frequency of 10 KHz.
3. The amplitude of output wave form is observed with the help of CRO.
4. Connect the output signal of Lag Network to the second channel of CRO.
5. Read the input and output signal voltages by slowly varying the input frequency in steps keeping the input amplitude constant.
6. Also record the Phase difference by comparing the two wave forms on CRO in each step.
7. Plot the output Voltage vs Frequency and Phase difference vs frequency curves.

Lag Network:

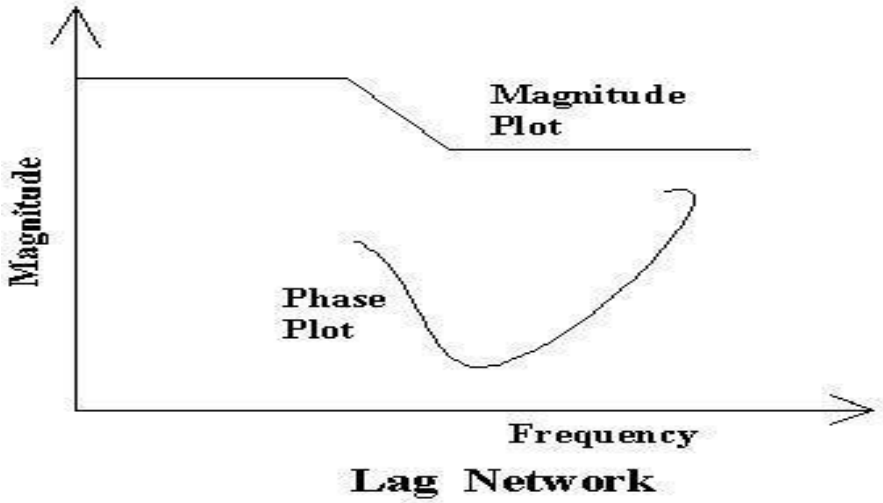
1. Connect the circuit as per circuit diagram.
2. A Sinusoidal input of say 2V p-p is applied with a frequency of 10 KHz.
3. The amplitude of output wave form is observed with the help of CRO.
4. Connect the output signal of Lead Network to the second channel of CRO.
5. Read the input and output signal voltages by slowly varying the input frequency in steps keeping the input amplitude constant.
6. Also record the Phase difference by comparing the two wave forms on CRO in each step.
7. Plot the output Voltage vs Frequency and Phase difference vs frequency curves.

OBSERVATIONS:

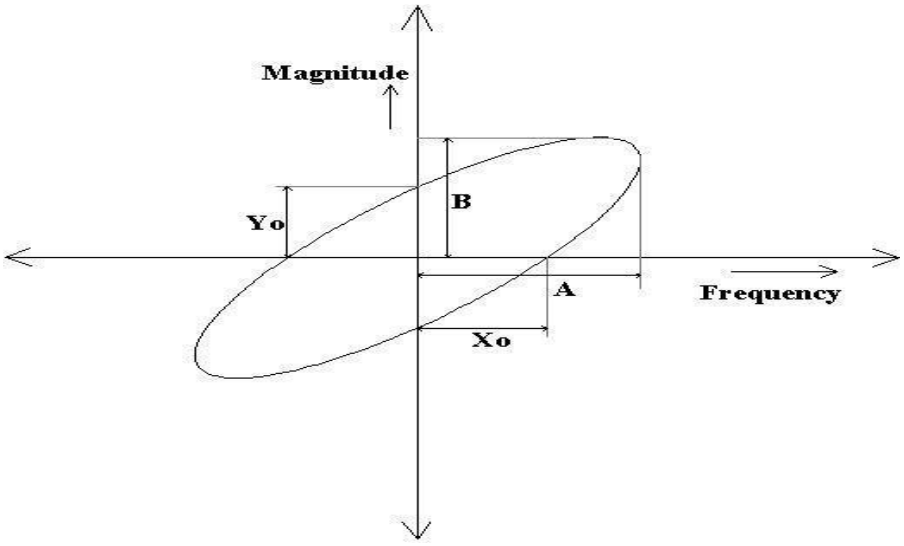
Lag Network:

S.No	Frequency	A	B	X_O	Y_O	Gain	Phase Angle

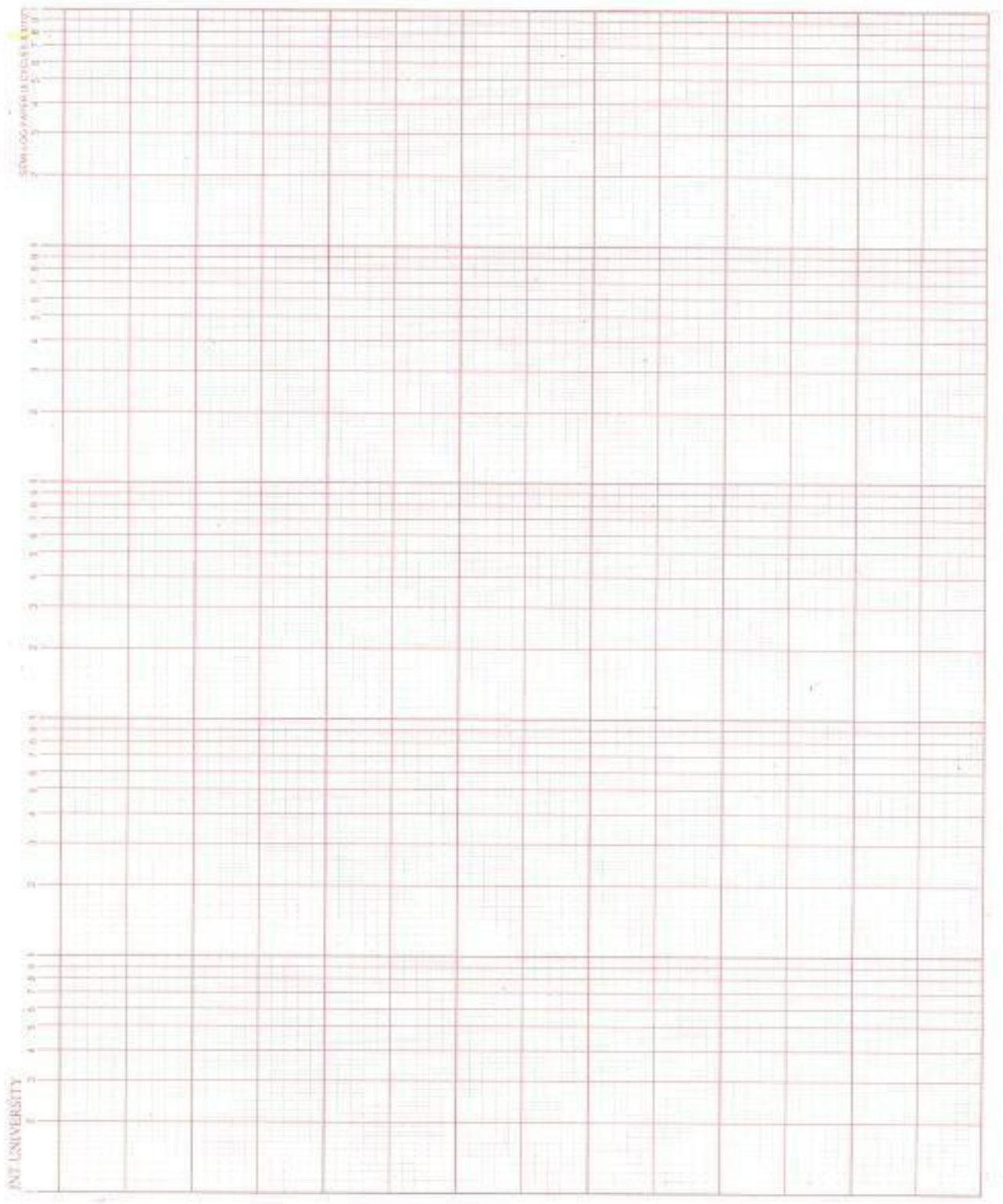
MODEL GRAPHS:



Lag Network



OUTPUT WAVEFORMS

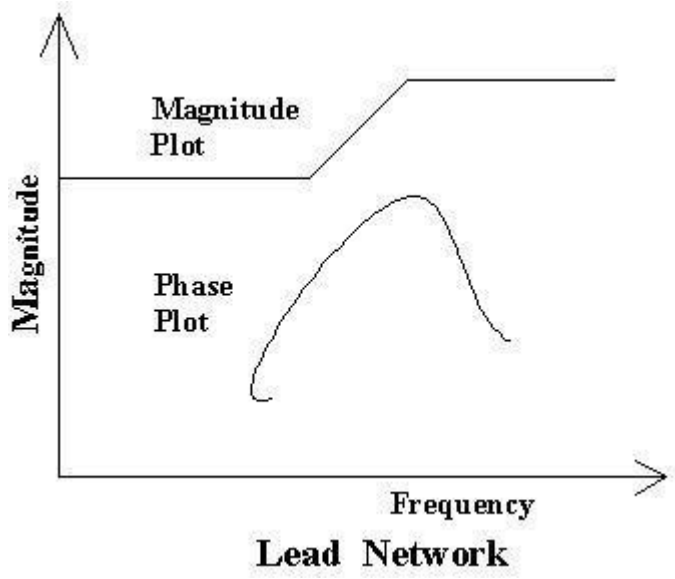


OBSERVATIONS:

Lead Network:

S.No	Frequency	A	B	X_O	Y_O	Gain	Phase Angle

MODEL GRAPHS:

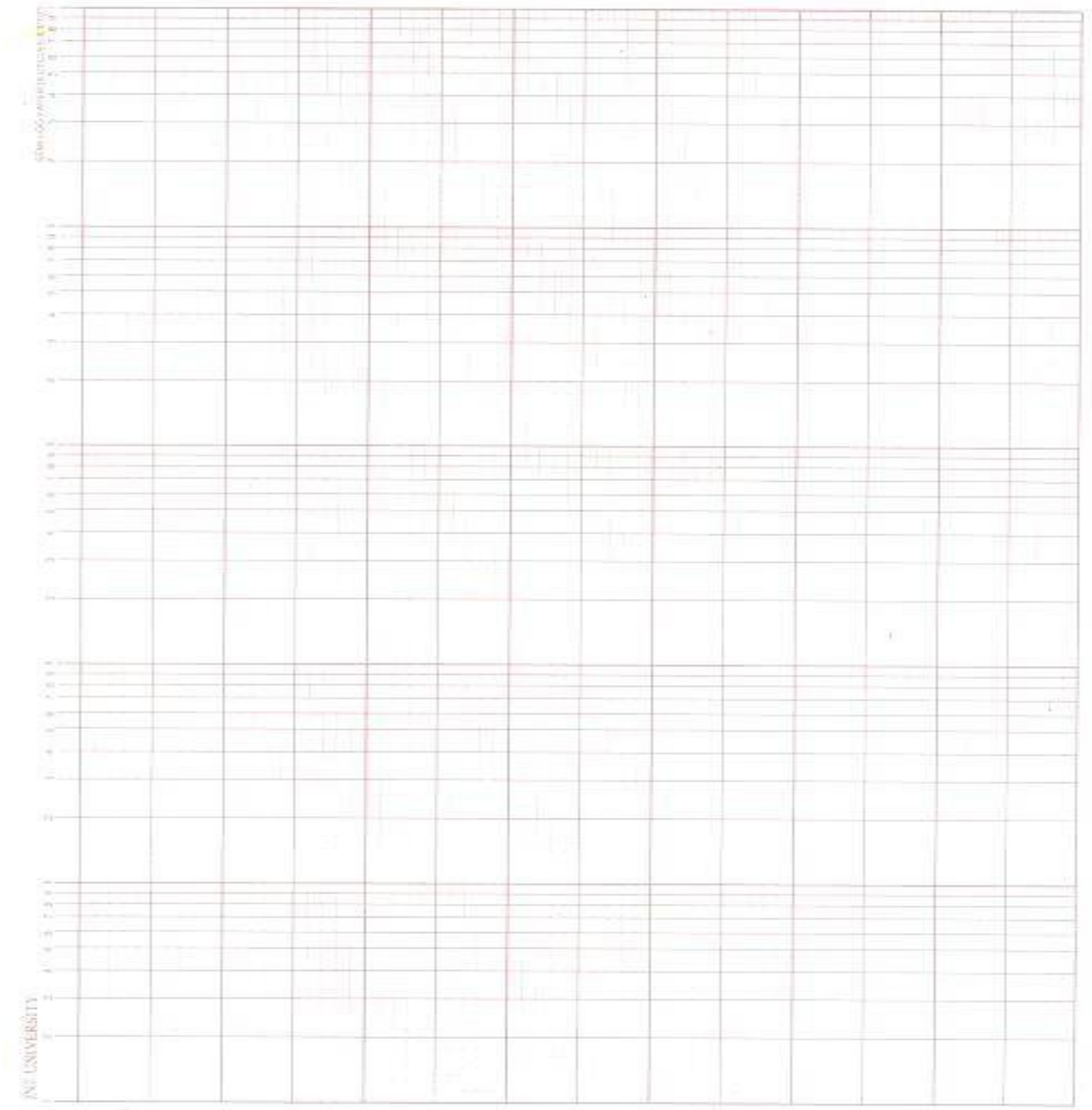


Lead Network

PRECAUTIONS:

1. Avoid lose connections.
2. Observe the response with out any parallax errors

OUTPUT WAVEFORMS:



RESULT: