

Experiment 1**Determination of Pole, Zero and Gain for the Given Transfer Function using MATLAB****1.1 AIM:**

To obtain

- I. Pole, zero, gain values from a given transfer function
- II. Transfer function model from pole, zero, gain values
- III. Pole, zero plot of a transfer function

1.2 APPARATUS REQUIRED

Sl.No	Equipment	Quantity
1.	Personal Computer with Windows operating system	01
2.	MATLAB Software.	01

1.3 THEORETICAL CALCULATION

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 S 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 S 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' that is 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:

$$G(s) = \frac{N(s)}{D(s)} = K \frac{(s-z_1)(s-z_2)\dots(s-z_n)}{(s-p_1)(s-p_2)\dots(s-p_m)}$$

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

The values of s for which N(S) = 0, are known as zeros of the system. i.e; at $s = z_1, z_2, \dots, z_n$.

The values of s for which D(S) = 0, are known as poles of the system. i.e; at $s = p_1, p_2, \dots, p_n$.

1.4 PROCEDURE

1. Open MATLAB
2. Type the program in Editor Window / Draw Simulink diagram
3. Save the program and Run the program.
4. If error occurs troubleshoot it

Program 1: Obtain pole, zero & gain values of a transfer function $G(S) = \frac{s^2+4s+3}{(s+5)(3s^2+4s+7)}$. Also obtain pole zero plot.

MATLAB Code :

```
num = [1 4 3]
den= conv ([1 5], [3 4 7])
g = tf (num,den)
[z,p,k] = tf2zp (num,den)
pzmap (g)
```

Output

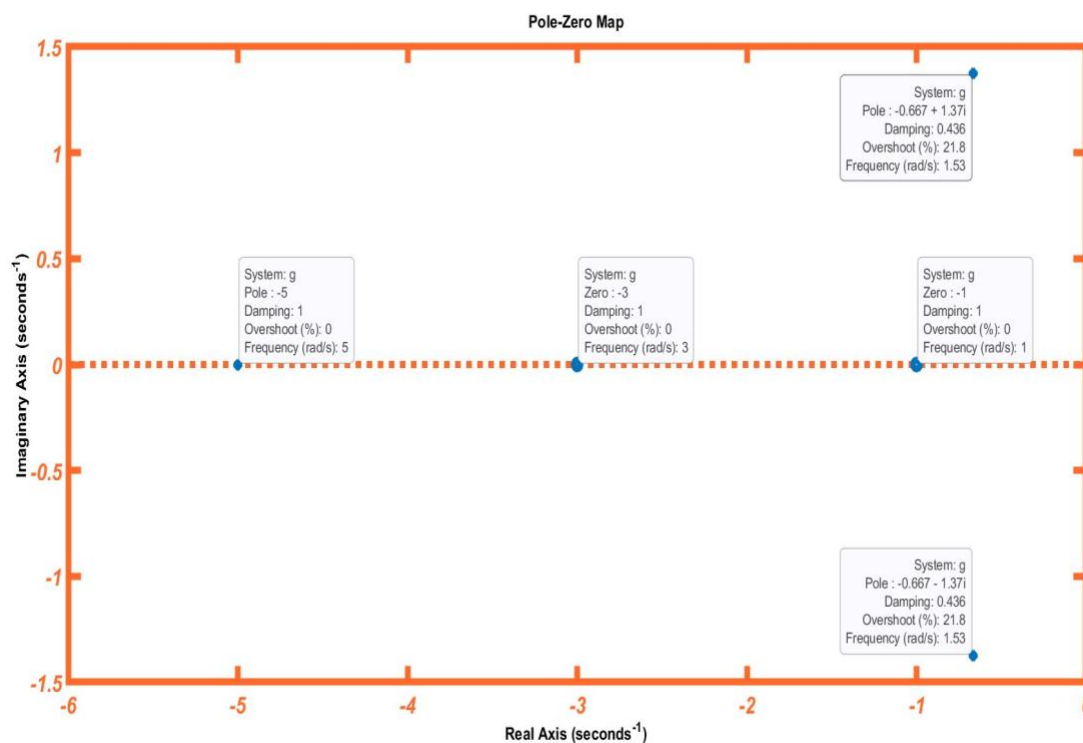
```
num =      1      4      3
den =      3     19     27     35
g =
      s^2 + 4 s + 3
-----
  3 s^3 + 19 s^2 + 27 s + 35
```

Continuous-time transfer function.

Model Properties

```
z =      -3      -1
p =  -5.0000 + 0.0000i   -0.6667 + 1.3744i   -0.6667 - 1.3744i
k =      0.3333
```

Pole – Zero Map



Program 2: Write MATLAB code to obtain transfer function of a system from its pole ,zero, gain values. Assume pole locations are -2, 1, zero at -1 and gain is 7.

MATLAB Code :

```
p= [-2 1]
z= [-1]
k=7
[num,den]= zp2tf(z,p,k)
g=tf(num,den)
```

```

p =      -2      1
z =      -1
k =      7
num =      0      7      7
den =      1      1      -2
g =

```

$$\frac{7s + 7}{s^2 + s - 2}$$

Assignments:

1. Obtain Pole, zero, gain values of the transfer functions given below. Also verify your result theoretically.

$$(i) G(S) = \frac{1}{(S^2+S+4)} \quad (ii) G(S) = \frac{5}{S^2+9} \quad 3. G(S) = \frac{1}{(S^2+3S+5)(S+3)(S+5)}$$

2. Obtain Transfer function of the systems (Theoretically & Practically).

1. Poles = -1+i, -1-i, -4. Zeros = -2, -5, gain = 1

2. Poles = -1+4i, -1-4i, -5. Zeros = -8, -5, gain = .75

RESULT:

Pole, Zero and Gain values of the given transfer function obtained and verified using MATLAB software.

Experiment 2

Mathematical modeling of physical systems using MATLAB programming and SIMULINK Tool Box

2.1 AIM:

Calculate the transfer function for the given control system using D'Alembert principle. Obtain the step response for the physical system MATLAB (program /Simulink) Software.

2.2 APPARATUS REQUIRED

Sl.No	Equipment	Quantity
1.	Personal Computer with Windows operating system	01
2.	MATLAB Software.	01

2.3 Transfer Function of Mechanical System

The Basic Mechanical System shown in figure 2.1 consists of Mass(M), Damper(B) and Spring(K). The force applied to the system is $F(t)$. A force $F(t)$ applied to the mass produces an acceleration of the mass. The reaction force f_m is equal to the product of mass and acceleration and is opposite in direction to the applied force in terms of displacement (x), a velocity v , and acceleration a .

Viscous friction(B): it is experienced when a body is in motion and the force is directly proportional to the velocity of the body. Spring is connected between fixed end and free end. When a force is applied, based on Hooke's law the reaction force (f_k) developed due to compression and elongation of the spring is equal to the product of the stiffness (k) and amount of deformation of the spring.

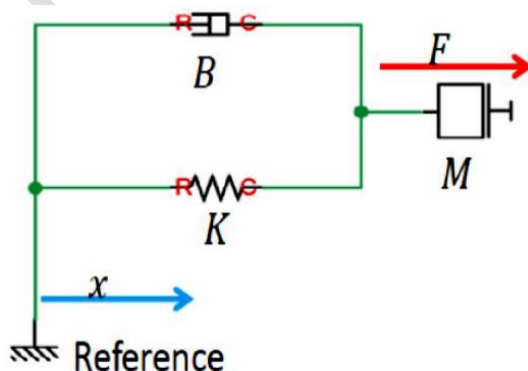


Figure 2.1 : Basic Mechanical System.

Given System:

$$F(t) = M \frac{d^2}{dt^2} x(t) + B \frac{d}{dt} x(t) + k x(t)$$

Taking Laplace Transform & Considering all initial conditions are zero

$$F(s) = M L\left[\frac{d^2}{dt^2} x(t)\right] + B L\left[\frac{d}{dt} x(t)\right] + K L[x(t)]$$

$$F(s) = Ms^2 x(s) + Bs x(s) + k x(s)$$

$$F(s) = x(s) [Ms^2 + Bs + k]$$

$$\frac{x(s)}{F(s)} = \frac{1}{Ms^2 + Bs + k}$$

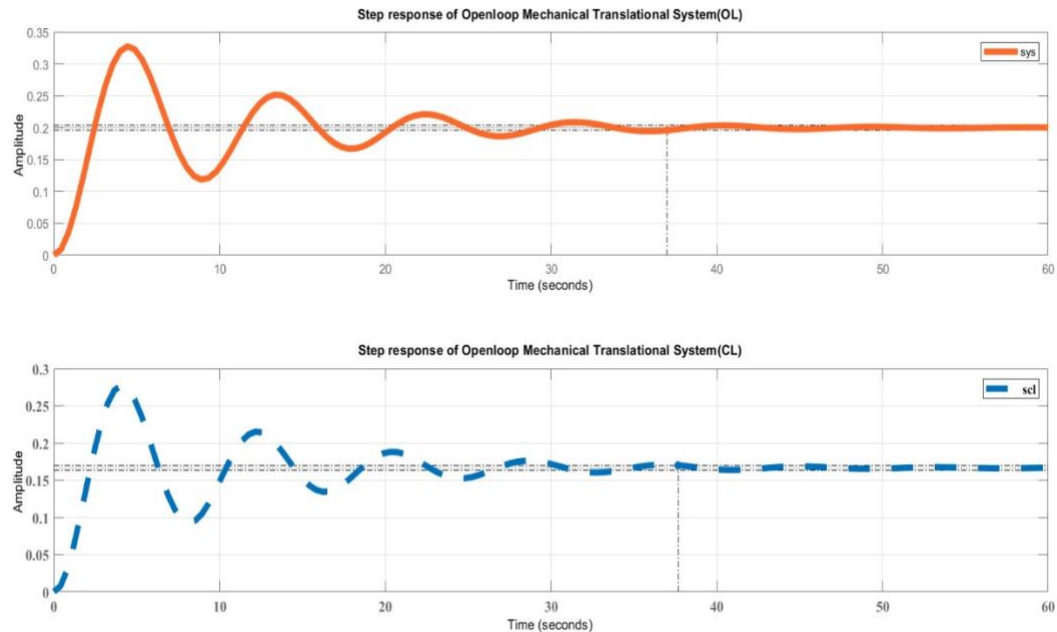
2.4 PROCEDURE

1. Open MATLAB
2. Type the program in Editor Window / Draw Simulink diagram
3. Save the program and run the program.
4. If error occurs troubleshoot it

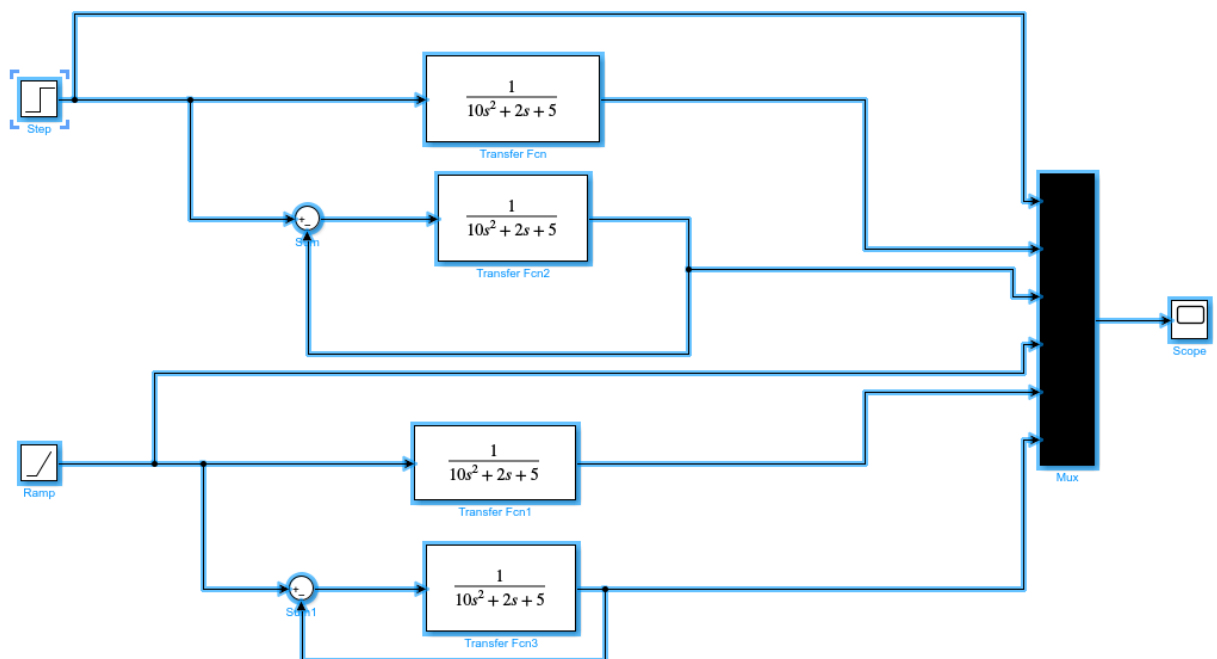
MATLAB Program

```
m=10;
b=2;
k=5;
num=[1];
den=[m b k];
figure(1);
subplot(2,1,1)
sys=tf(num,den)
step(sys)
title('Step response of Openloop Mechanical Translational
System(OL)');
figure(1);
subplot(2,1,2);
scl=feedback(sys,1)
step(scl)
title('Step response of Openloop Mechanical Translational
System(CL)')
```

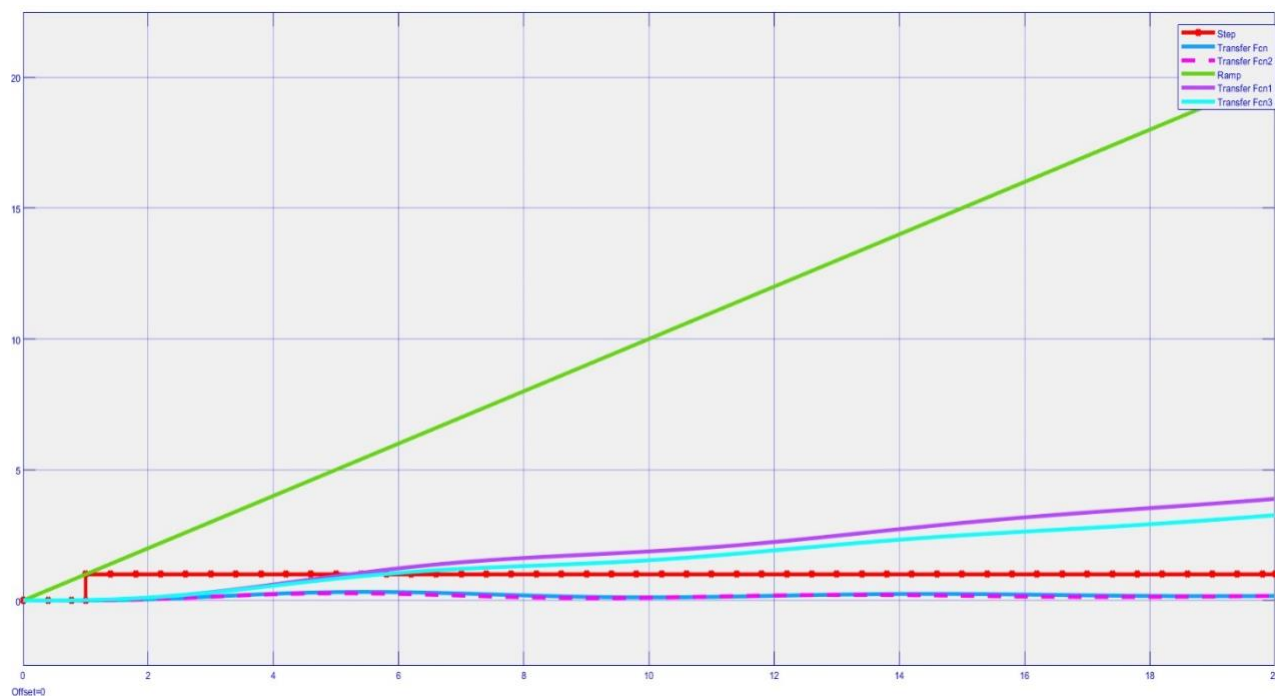
Output Response (OL/CL) of Mechanical Translational Mechanical Systems subjected to step input



Simulink diagram of Mechanical Translational Systems



Output Response (OL/CL) of Mechanical Translational Mechanical Systems subjected to step input in Simulink Environment



RESULT:

Step response for the Physical (Mechanical Translational) system is obtained using MATLAB program and Simulink.

Experiment 3 Generation of Standard Test Signals using MATLAB

3.1 AIM:

Generate standard test signals in a MATLAB Environment.

3.2 APPARATUS REQUIRED

Sl.No	Equipment	Quantity
1.	Personal Computer with Windows operating system	01
2.	MATLAB Software.	01

3.3 Theory

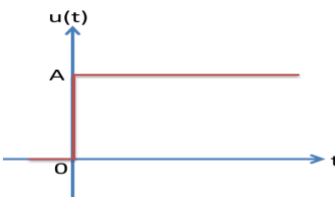
The standard test signals are step, ramp, impulse, parabolic and sinusoidal. The signals which are most commonly used as reference inputs are called as standard test signals. They are used to predetermine the system performance through mathematical & experimental analysis.

They resembles the characteristics of input signals (sudden change, constant velocity, shock, constant acceleration). So if it satisfy the system performance we can implement the system for various inputs.

1. Step Signal

The step signal imitates the sudden change characteristic of actual input signal. If $A=1$, the step signal is called unit step signal

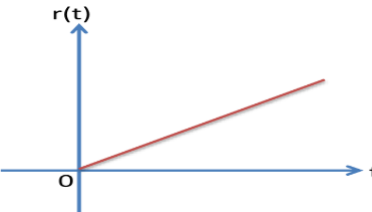
$$u(t) = \begin{cases} A & t \geq 0 \\ 0 & t < 0 \end{cases}$$

$$L\{u(t)\} = U(s) = \frac{A}{s}$$


2.Ramp Signal

The ramp signal imitates the constant velocity characteristic of actual input signal. If $A=1$, the ramp signal is called unit ramp signal.

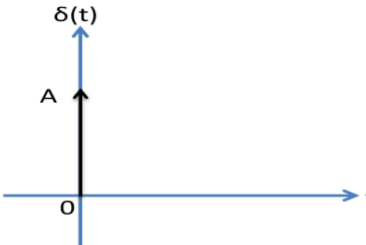
$$r(t) = \begin{cases} At & t \geq 0 \\ 0 & t < 0 \end{cases}$$

$$L\{r(t)\} = R(s) = \frac{A}{s^2}$$


2.Impulse Signal

The impulse signal imitates the sudden shock characteristic of actual input signal. If $A=1$, the impulse signal is called unit impulse signal.

$$\delta(t) = \begin{cases} A & t = 0 \\ 0 & t \neq 0 \end{cases}$$

$$L\{\delta(t)\} = \delta(s) = A$$


3.4 PROCEDURE

1. Open MATLAB
2. Type the program in Editor Window / Draw Simulink diagram
3. Save the program and run the program.
4. If error occurs troubleshoot it

3.5 MATLAB Program

Program 1: Discrete Signal Generation in MATLAB

%GENERATION OF STEP SIGNAL

```
N=20;  
x=ones(1,N);  
n=0:1:N-1;  
subplot(4,2,1);  
stem(n,x);  
xlabel('n');  
ylabel('X(n)');  
title('Unit Step Signal');
```

%GENERATION OF RAMP SIGNAL

```
N=20;  
x=n;  
n=0:1:N-1;  
subplot(4,2,2);  
stem(n,x);  
xlabel('n');  
ylabel('X(n)');  
title('Unit Ramp Signal');
```

%GENERATION OF UNIT IMPULSE SIGNAL

```
N=20;  
x=[ones(1,1),zeros(1,(N-1))];  
n=0:1:N-1;  
subplot(4,2,3);  
stem(n,x);  
xlabel('n');  
ylabel('X(n)');  
title('Unit Impulse Signal');
```

%GENERATION OF EXPONENTAIL SIGNAL

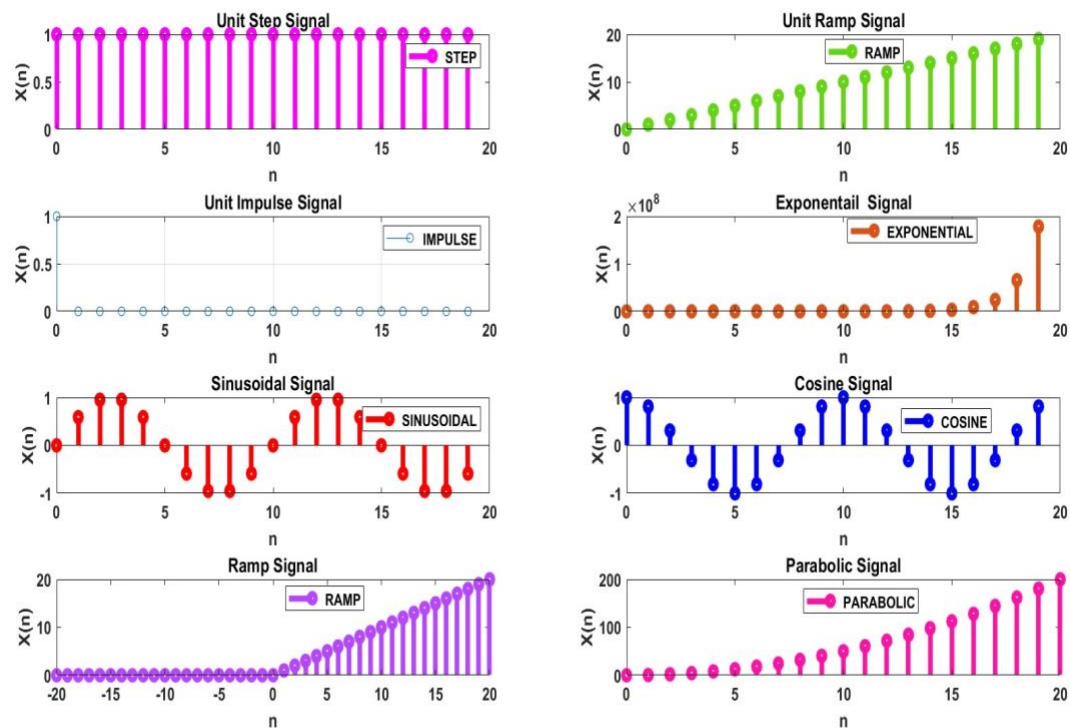
```
N=20;  
x=exp(n);  
n=0:1:N-1;  
subplot(4,2,4);  
stem(n,x);  
xlabel('n');  
ylabel('X(n)');  
title('Exponentail Signal');
```

%GENERATE SINUOIDAL SIGNAL

```
N=20;  
x=sin(0.2*pi*n);  
n=0:1:N-1;  
subplot(4,2,5);  
stem(n,x);  
xlabel('n');  
ylabel('X(n)');  
title('Sinusoidal Signal');
```

%GENERATE AN COSINE SIGNAL

```
N=20;  
x=cos(0.2*pi*n);  
n=0:1:N-1;  
subplot(4,2,6);  
stem(n,x);  
xlabel('n');  
ylabel('X(n)');  
title('Cosine Signal');  
N=-20:20;  
ramp_N=(N>=0).*N;  
subplot(4,2,7);  
stem(N,ramp_N);  
xlabel('n');  
ylabel('X(n)');  
title('Ramp Signal');  
N=0:1:20;  
parabola=0.5*(N.^2);  
subplot(4,2,8);  
stem(N,parabola);  
xlabel('n');  
ylabel('X(n)');  
title('Parabolic Signal');
```

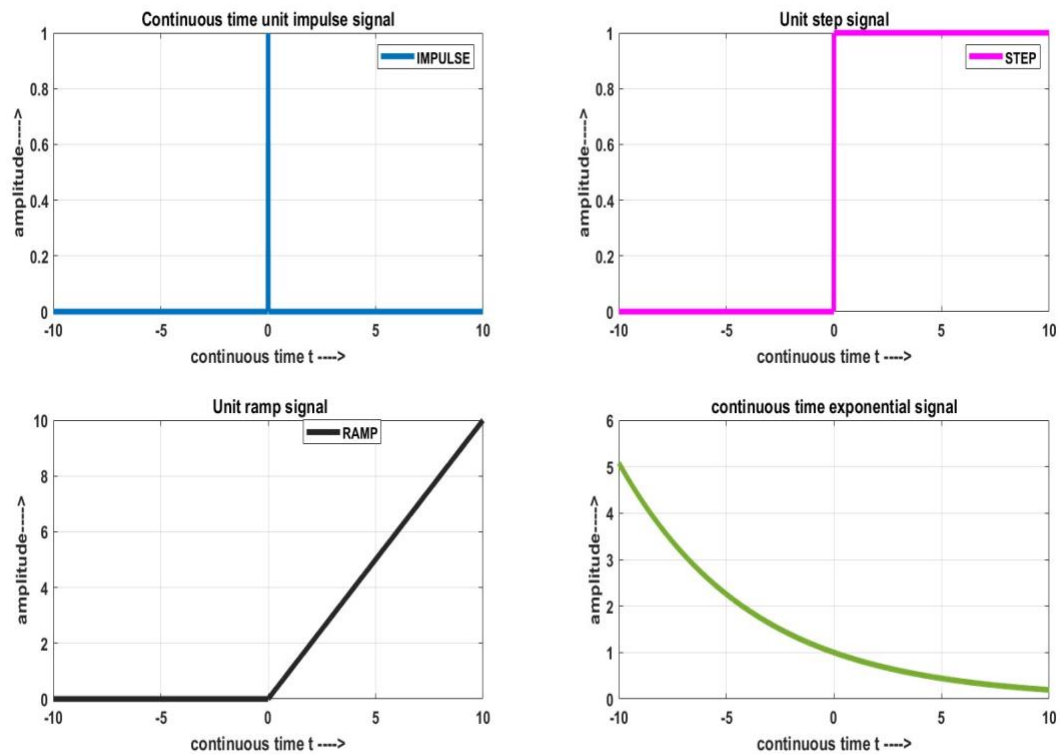
MATLAB OUTPUT**Figure :3.1 Discrete Standard Signals****Program 2: Continuous Signal Generation in MATLAB (using Loop Structures)**

```

clc;
clear all;
close all;
t=-10:0.01:10;
L=length(t);
for i=1:L
% to generate a continuous time impulse function
if t(i)==0
x1(i)=1;
else
x1(i)=0;
end;
% to generate a continuous time unit step signal
if t(i)>=0
x2(i)=1;
% to generate a continuous time ramp signal
x3(i)=t(i);

```

```
else
x2(i)=0;
x3(i)=0;
end;
end;
% to generate a continuous time exponential signal
a=0.85;
x4=a.^t;
figure;
subplot(2,2,1);
plot(t,x1);
grid on;
xlabel('continuous time t ---->');
ylabel('amplitude---->');
title('Continuous time unit impulse signal');
subplot(2,2,2);
plot(t,x2);
grid on;
xlabel('continuous time t ---->');
ylabel('amplitude---->');
title('Unit step signal');
subplot(2,2,3);
plot(t,x3);
grid on;
xlabel('continuous time t ---->');
ylabel('amplitude---->');
title('Unit ramp signal');
subplot(2,2,4);
plot(t,x4);
xlabel('continuous time t ---->');
grid on;
ylabel('amplitude---->');
title('continuous time exponential signal');
```

MATLAB OUTPUT**Figure :3.2 Continuous Standard Signals****RESULT:**

Standard test signals are generated using MATLAB program.

Experiment 4 Block Diagram Reduction using MATLAB

4.1 AIM:

To reduce the block diagram and obtain transfer function in a MATLAB Environment.

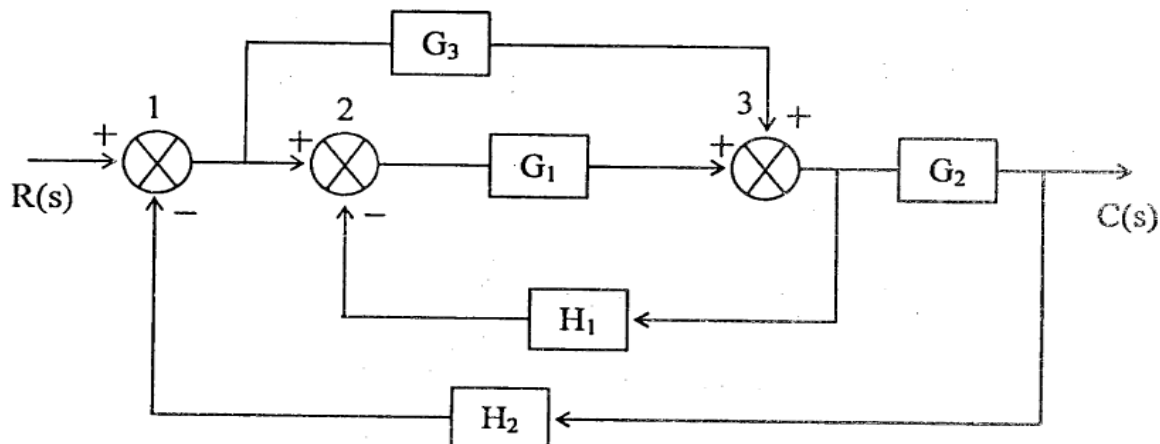
4.2 APPARATUS REQUIRED

Sl.No	Equipment	Quantity
1.	Personal Computer with Windows operating system	01
2.	MATLAB Software.	01

4.3 Block Diagram Reduction

Reduce the following block using block diagram reduction rules and obtain transfer function of same.

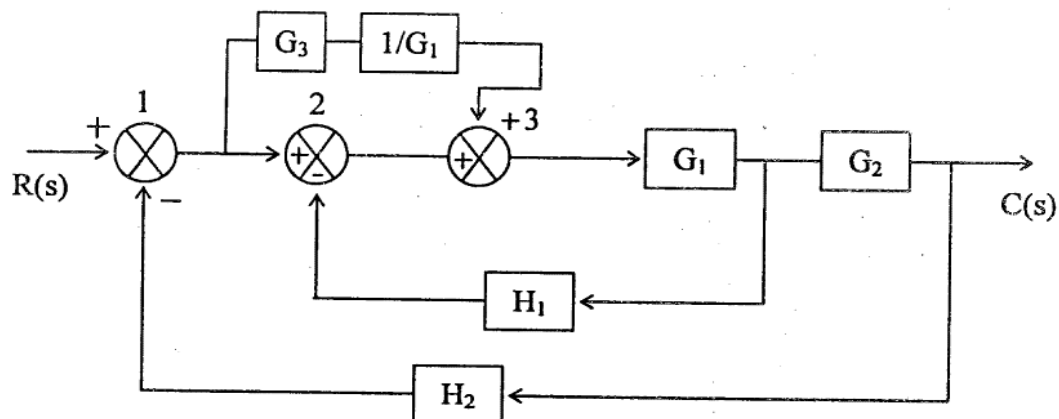
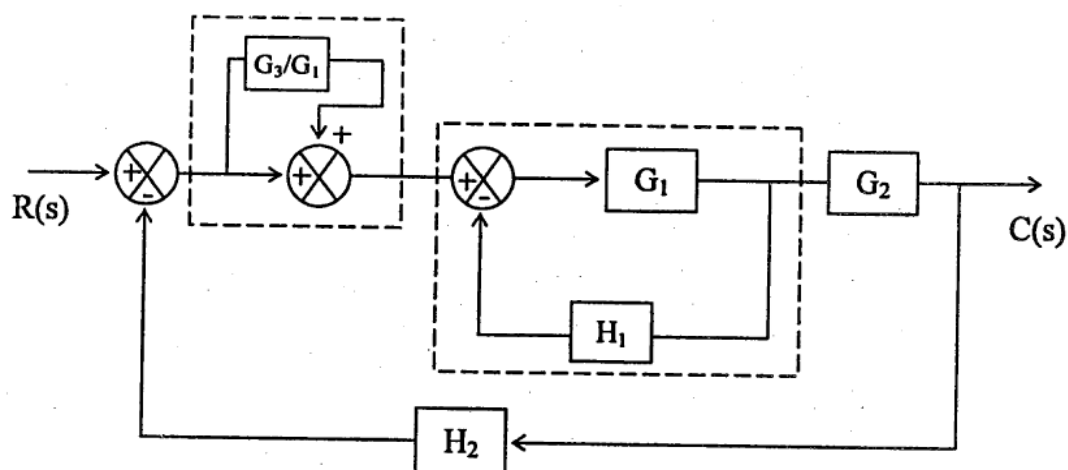
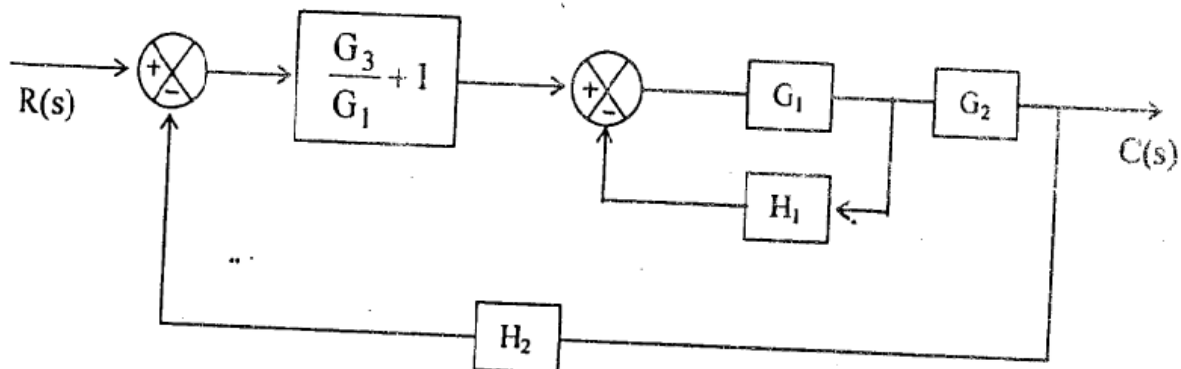
Given

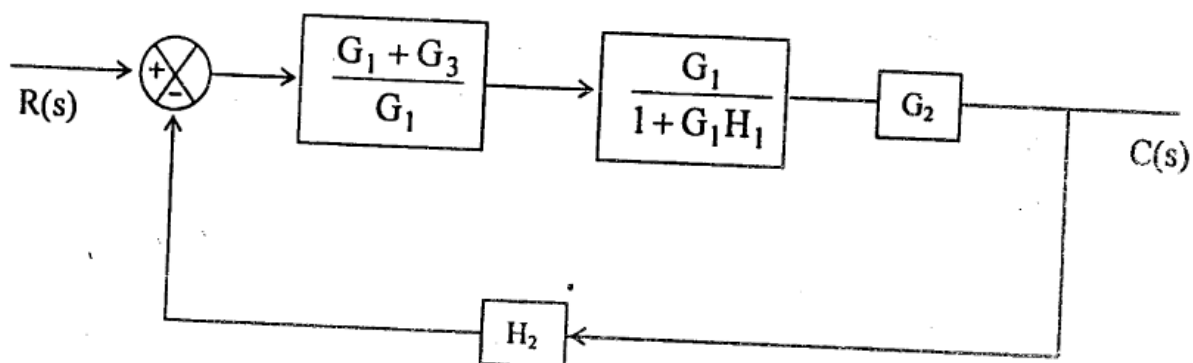
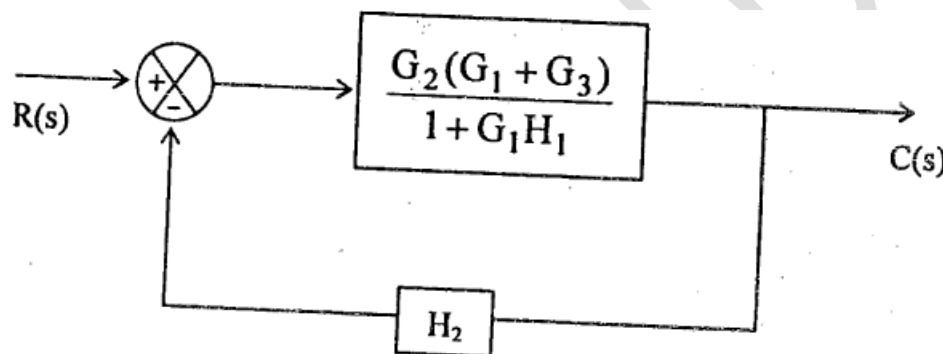


$$G_1 = \frac{1}{s+10}, G_2 = \frac{1}{s+1}, G_3 = \frac{s^2+1}{s^2+4s+4}, H_1 = \frac{s+1}{s+2}, H_2 = 2$$

Step 1

Move summing point (3) before G_1 .

**Step 2** Interchanging summing Points 2 and 3**Step 3** Parallel Elimination and Feedback Elimination

Step4: Cascading the blocks**Step5: Feed Back Elimination**

$$\frac{C(S)}{R(S)} = \frac{G_2(G_1 + G_3)}{1 + G_1H_1 + G_2H_2(G_1 + G_3)}$$

4.4 PROCEDURE

1. Open MATLAB
2. Type the program in Editor Window / Draw Simulink diagram
3. Save the program and run the program.
4. If error occurs troubleshoot it

MATLAB Program for block diagram reduction

```

num1=[1];
den1=[1 10];
G1=tf(num1,den1)
num2=[1];
den2=[1 1];
G2=tf(num2,den2)
num3=[1 0 1];
den3=[1 4 4];
G3=tf(num3,den3)
num4=[1 1];
den4=[1 2];
H1=tf(num4,den4)
num5=[2];
den5=[1];
H2=tf(num5,den5)
S1=G3/G1
S2=parallel(S1,1)
S3=feedback(G1,H1)
S4=series(S2,S3)
S5=series(G2,S4)
sys=feedback(S5,H2)

```

MATLAB OUTPUT

```

G1 =
      1
-----
s + 10

```

Continuous-time transfer function.

Model Properties

```

G2 =
      1
-----
s + 1

```

Continuous-time transfer function.

Model Properties

```

G3 =
      s^2 + 1
-----
s^2 + 4 s + 4

```

Continuous-time transfer function.

Model Properties

H1 =

$$\frac{s + 1}{s + 2}$$

Continuous-time transfer function.

Model Properties

H2 =

2

Static gain.

Model Properties

S1 =

$$\frac{s^3 + 10s^2 + s + 10}{s^2 + 4s + 4}$$

Continuous-time transfer function.

Model Properties

S2 =

$$\frac{s^3 + 11s^2 + 5s + 14}{s^2 + 4s + 4}$$

Continuous-time transfer function.

Model Properties

S3 =

$$\frac{s + 2}{s^2 + 13s + 21}$$

Continuous-time transfer function.

Model Properties

S4 =

$$\frac{s^4 + 13s^3 + 27s^2 + 24s + 28}{s^4 + 17s^3 + 77s^2 + 136s + 84}$$

Continuous-time transfer function.

Model Properties

S5 =

$$\frac{s^4 + 13s^3 + 27s^2 + 24s + 28}{s^5 + 18s^4 + 94s^3 + 213s^2 + 220s + 84}$$

Continuous-time transfer function.

Model Properties

$$\text{sys} = \frac{s^4 + 13s^3 + 27s^2 + 24s + 28}{s^5 + 20s^4 + 120s^3 + 267s^2 + 268s + 140}$$

Continuous-time transfer function.
Model Properties

RESULT:

Transfer function of the given block diagram was computed and verified using MATLAB

Experiment 5
OPEN LOOP AND CLOSED LOOP TIME RESPONSE OF FIRST ORDER SYSTEM
WHEN IT IS SUBJECTED TO STEP, RAMP AND IMPULSE, PARABOLIC INPUTS

5.1 AIM:

The First order system has the transfer function $G(s) = \frac{1}{s+1}$. When the system (Open/Closed loop) is subjected to step, ramp, impulse, parabolic inputs and check the results using MATLAB Software.

5.2 APPARATUS REQUIRED

SLNo	Equipment	Quantity
1.	Personal Computer with Windows operating system	01
2.	MATLAB Software.	01

5.3 Time Response of First order system subjected to various inputs**Case1: Step Input**

$$\frac{C(S)}{R(S)} = \frac{1}{S+1}$$

$$C(S) = R(S) \frac{1}{s+1}$$

For step input $R(s) = \frac{1}{s}$ $C(S) = \frac{1}{s} \frac{1}{s+1}$

$$L^{-1}[C(s)] = L^{-1}\left[\frac{1}{s(s+1)}\right]$$

Partial Fraction

$$\frac{1}{s(s+1)} = \frac{A}{s} + \frac{B}{(s+1)}$$

$$1 = A(s+1) + BS$$

$$S=0 \quad A=1$$

$$S=-1 \quad B=-1$$

$$L^{-1}[C(s)] = L^{-1}\left[\frac{1}{s} + \frac{-1}{(s+1)}\right]$$

$$C(t) = 1 - e^{-t}$$

First order system response subjected to step input

Time	Response $C(t) = 1 - e^{-t}$
1 sec	$C(t) = 1 - e^{-1} = 0.6321$
2 sec	$C(t) = 1 - e^{-2} = 0.8646$
3 sec	$C(t) = 1 - e^{-3} = 0.9502$
4 sec	$C(t) = 1 - e^{-4} = 0.9816$
5 sec	$C(t) = 1 - e^{-5} = 0.9932$
$t = \infty$	$C(t) = 1 - e^{-\infty} = 1$

Case2: Ramp Input

$$\frac{C(S)}{R(S)} = \frac{1}{S+1}$$

$$C(S) = R(S) \frac{1}{S+1}$$

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

$$C(S) = \frac{1}{s^2(s+1)}$$

$$L^{-1}[C(s)] = L^{-1}\left[\frac{1}{s^2(s+1)}\right]$$

Partial Fraction

$$\frac{1}{s^2(s+1)} = \frac{A}{s} + \frac{B}{s^2} + \frac{C}{(s+1)}$$

$$1 = AS(s+1) + B(S+1) + CS^2$$

$$S = -1 \quad C = 1$$

$$S = 0 \quad B = 1$$

Equating S^2 coefficients $0 = A+C$ Then $A = -1$

$$L^{-1}[C(s)] = L^{-1}\left[\frac{-1}{s} + \frac{1}{s^2} + \frac{1}{(s+1)}\right]$$

$$C(t) = t - 1 + e^{-t}$$

Case3: Parabolic Input

$$\frac{C(S)}{R(S)} = \frac{1}{S+1}$$

$$C(S) = R(S) \frac{1}{S+1}$$

For parabolic input $R(s) = \frac{1}{s^3}$

$$C(S) = \frac{1}{s^3(s+1)}$$

$$L^{-1}[C(s)] = L^{-1}\left[\frac{1}{s^3(s+1)}\right]$$

Partial Fraction

$$\frac{1}{s^3(s+1)} = \frac{A}{s} + \frac{B}{s^2} + \frac{C}{s^3} + \frac{D}{(s+1)}$$

$$1 = AS^2(S+1) + BS(S+1) + C(S+1) + DS^3$$

$$S = 0 \quad C = 1$$

$$S = -1 \quad D = -1$$

Equating S^3 coefficients $0 = A+D \quad A = 1$

Equating S coefficients $0 = B+C$ Then $B = -1$

$$L^{-1}[C(s)] = L^{-1}\left[\frac{1}{s} + \frac{-1}{s^2} + \frac{1}{s^3} + \frac{-1}{(s+1)}\right]$$

$$C(t) = 1 - t + \frac{t^2}{2} + e^{-t}$$

Case4: Impulse Input

$$\frac{C(S)}{R(S)} = \frac{1}{S+1}$$

For Impulse input $R(s) = 1$

$$C(S) = R(S) \frac{1}{S+1}$$

$$L^{-1}[C(s)] = L^{-1}\left[\frac{1}{s+1}\right]$$

$$C(t) = e^{-t}$$

5.4 PROCEDURE

1. Open MATLAB
2. Type the program in Editor Window / Draw Simulink diagram
3. Save the program and run the program.
4. If error occurs troubleshoot it

First Order System [OL/CL] Subjected to Unit Step, Ramp, Parabolic and Impulse Inputs using MATLAB Program

```
%RESPONSE OF FIREST ORDER SYSTEM[OL] SUBJECTED TO STEP INPUT
num1=[1];
den1=[1 1];
t=0:0.1:10
subplot(2,4,1)
step(num1,den1,t);
title( 'FOS[OL] WITH STEP INPUT')
xlabel( 'TIME IN SECONDS')
ylabel( 'STEP INPUT')
grid
%RESPONSE OF FIREST ORDER SYSTEM[OL] SUBJECTED TO RAMP INPUT
r=t
t=0:0.1:10
y=lsim(num1,den1,r,t);
subplot(2,4,2)
plot(t,r, '-',t,y, '*')
title( 'FOS[OL] WITH RAMP INPUT');
xlabel( 'TIME IN SECONDS');
ylabel( 'RAMP INPUT AND SYSTEM OUTPUT');
grid
%RESPONSE OF FIREST ORDER SYSTEM[OL] SUBJECTED TO PARABOLIC INPUT
r=((t.^2)/2)
t=0:0.1:10
y=lsim(num1,den1,r,t);
subplot(2,4,3)
plot(t,r, '-',t,y, '*')
title( 'FOS[OL] WITH PARABOLIC INPUT');
xlabel( 'TIME IN SECONDS');
ylabel( 'PARABOLIC INPUT AND SYSTEM OUTPUT');
grid
%RESPONSE OF FIREST ORDER SYSTEM[OL] SUBJECTED TO IMPULSE INPUT
t=0:0.1:10
subplot(2,4,4)
impz(num1,den1,t);
title( 'FOS[OL] WITH IMPULSE INPUT');
```

```
xlabel( 'TIME IN SECONDS');
ylabel( 'SYSTEM OUTPUT');
grid
%RESPONSE OF FIREST ORDER SYSTEM[CL] SUBJECTED TO STEP INPUT
sys=tf(num1,den1);
sys_c1=feedback(sys,1);
subplot(2,4,5)
step(sys_c1,t);
title( 'FOS[CL] WITH STEP INPUT')
xlabel( 'TIME IN SECONDS')
ylabel( 'STEP INPUT')
grid
%RESPONSE OF FIREST ORDER SYSTEM[CL] SUBJECTED TO RAMP INPUT
r=t
t=0:0.1:10
y=lsim(sys_c1,r,t);
subplot(2,4,6)
plot(t,r, '- ',t,y, '*')
title( 'FOS[CL] WITH RAMP INPUT');
xlabel( 'TIME IN SECONDS');
ylabel( 'RAMP INPUT AND SYSTEM OUTPUT');
grid
%RESPONSE OF FIREST ORDER SYSTEM[CL] SUBJECTED TO PARABOLIC INPUT
r=((t.^2)/2)
t=0:0.1:10
y=lsim(sys_c1,r,t);
subplot(2,4,7)
plot(t,r, '- ',t,y, '*')
title( 'FOS[CL] WITH PARABOLIC INPUT');
xlabel( 'TIME IN SECONDS');
ylabel( 'PARABOLIC INPUT AND SYSTEM OUTPUT');
grid
%RESPONSE OF FIREST ORDER SYSTEM[CL] SUBJECTED TO IMPULSE INPUT
t=0:0.1:10
subplot(2,4,8)
impz(sys_c1,t);
title( 'FOS[CL] WITH IMPULSE INPUT');
xlabel( 'TIME IN SECONDS');
ylabel( 'SYSTEM OUTPUT');
grid
```

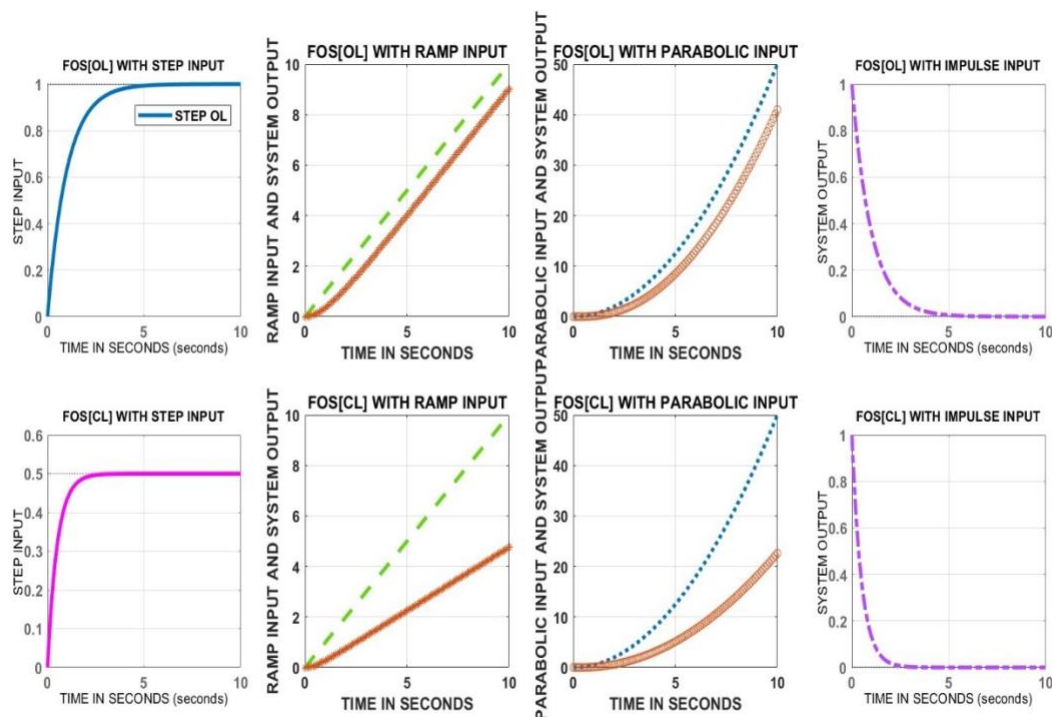
MATLAB OUTPUT

Figure 5.1 Output response of First Order System [OL/CL] Subjected to Unit Step, Ramp, Parabolic and Impulse Inputs using MATLAB Program.

First Order System [OL/CL] Subjected to Unit Step, Ramp, Parabolic and Impulse Inputs using SIMULINK

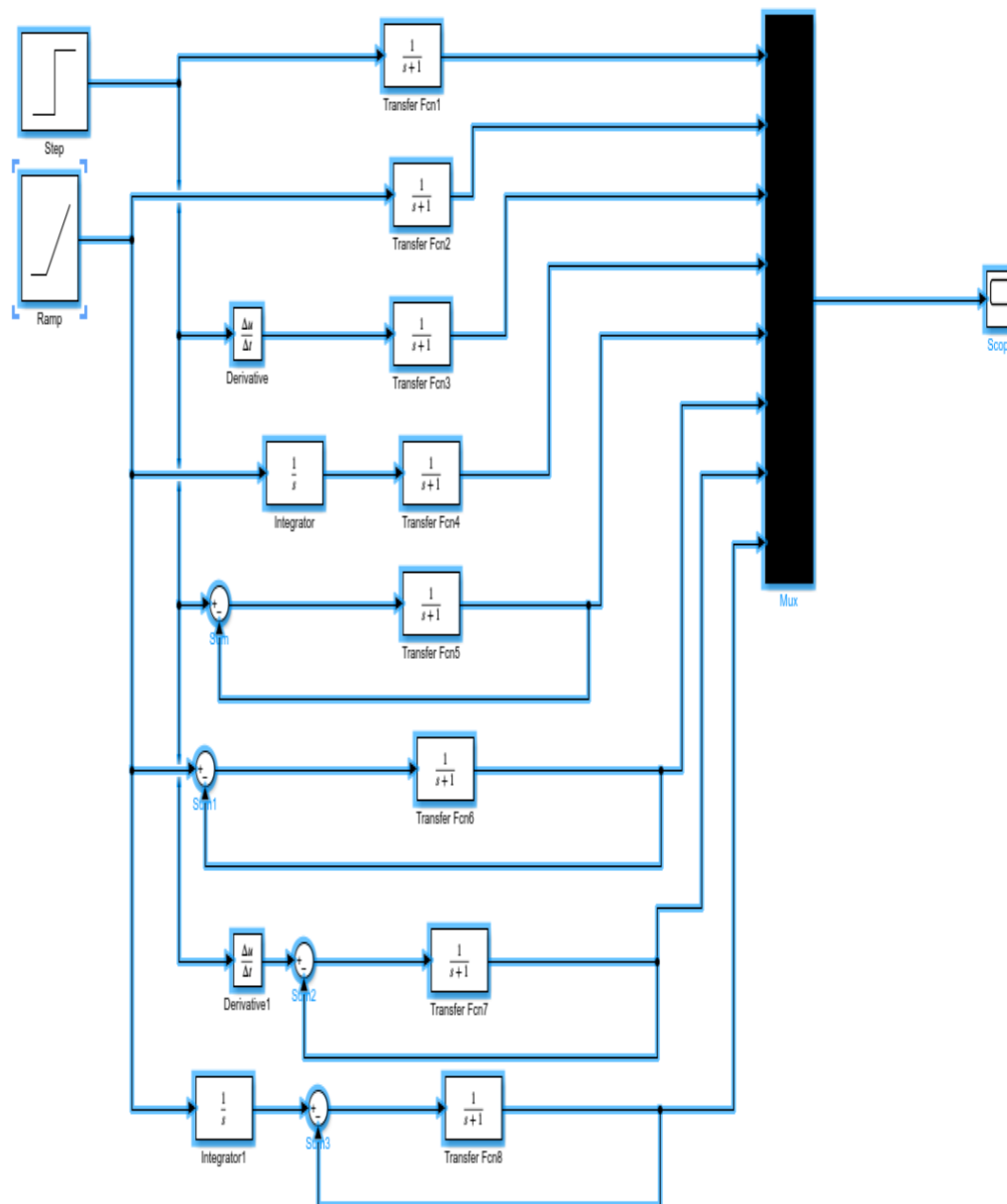


Figure 5.2 SIMULINK Model for first Order System [OL/CL] Subjected to Unit Step, Ramp, Parabolic and Impulse Inputs.



Figure 5.3 Output response of First Order System [OL/CL] Subjected to Unit Step, Ramp, Parabolic and Impulse Inputs using SIMULINK.

RESULT:

Output response of First Order System [OL/CL] Subjected to Unit Step, Ramp, Parabolic and Impulse Inputs are obtained using MATLAB Program and SIMULINK Model.

Experiment 6 TIME DOMAIN SPECIFICATIONS OF A SECOND ORDER SYSTEM

6.1 AIM:

Obtain the time domain specifications of given second order system using MATLAB.

6.2 APPARATUS REQUIRED

Sl.No	Equipment	Quantity
1.	Personal Computer with Windows operating system	01
2.	MATLAB Software.	01

6.3 Time Domain Specifications of Second Order System

Transfer Function of Second Order System

$$\frac{C(S)}{R(S)} = \frac{\omega_n^2}{S^2 + 2\zeta\omega_n S + \omega_n^2}$$

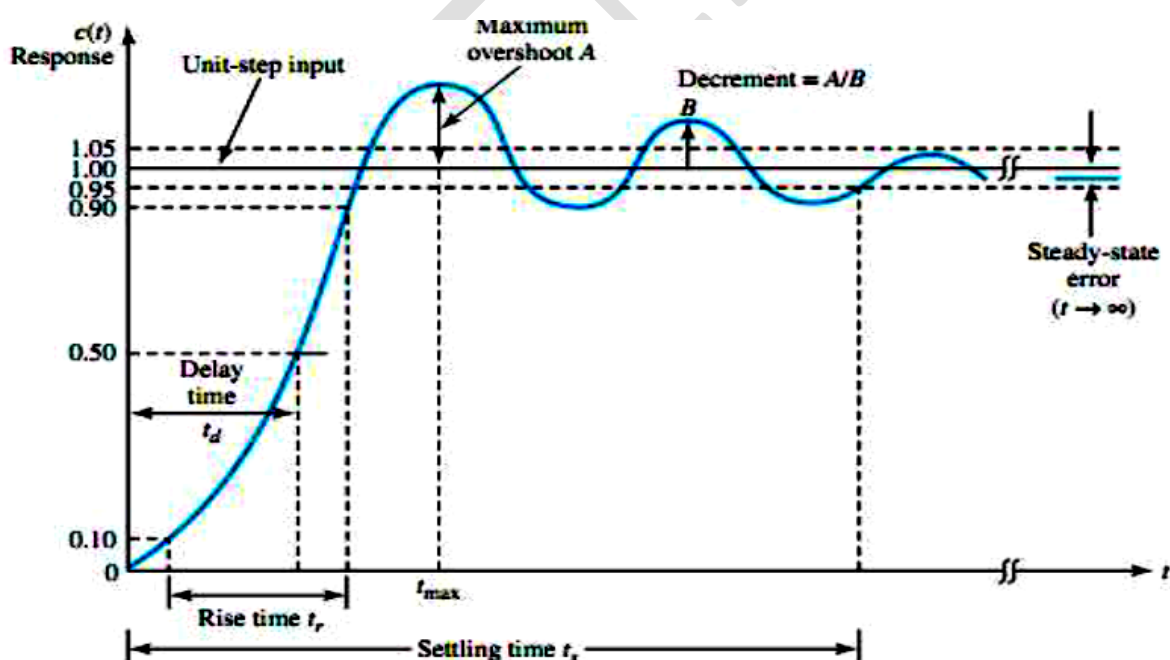


Figure 6.1 Time Domain Specifications of Under Damped Second Order ($0 < \zeta < 1$) System Subjected to Step input

Definitions**Time Response Specifications :****1. Delay time (td) :**

It is time required for the response to reach 50% of the final value in first attempt.

$$t_d = \frac{1 + 0.7 \xi}{\omega_n} \text{ second}$$

2. Rise time (tr) :

It is time required for the response to rise from 10% to 90% of the final value for overdamped systems and 0 to 100% of the final value for underdamped system.

$$t_r = \frac{\pi - \theta}{\omega_d} \text{ second where } \theta \text{ must be in radians.}$$

3. Peak time (tp) :

It is time required for response to reach its peak value.

$$t_p = \frac{\pi}{\omega_d} \text{ second.}$$

4. Peak overshoot (MP) :

It indicates the normalized difference between the time response peak and steady state output and is given by :

$$MP\% = C(t) |_{t=T_P} - 1$$

$$MP\% = e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}} \times 100$$

5. Settling time (Ts) :

It is time required for the response to reach and stay within a specified tolerance band [usually 2% or 5%] of its final value.

$T_s = 4 T$ for 2 % tolerance band

$$= 3 T \text{ for 5\% tolerance band where } T = \frac{1}{\xi\omega_n}.$$

Transient Response Specifications * Formulae *

$$1. C(t) = 1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}} \sin(\omega_d t + \theta)$$

where $\omega_d = \omega_n \sqrt{1-\xi^2}$ and $\theta = \tan^{-1} \frac{\sqrt{1-\xi^2}}{\xi}$.

$$2. \text{ delay time (td)} = \frac{1+0.7\xi}{\omega_n} \text{ second}$$

$$3. \text{ Rise time (tr)} = \frac{\pi - \theta}{\omega_d} \text{ second}$$

$$4. \text{ Peak time (tp)} = \frac{\pi}{\omega_d} \text{ second}$$

$$5. \text{ Mp \%} = e^{-\pi\xi/\sqrt{1-\xi^2}} \times 100$$

$$6. \text{ Ts} = 3T \quad \text{for 5 \% tolerance}$$

$$= 4T \quad \text{for 2 \% tolerance} \quad \text{where } T = \frac{1}{\xi\omega_n}.$$

Problem 1 :

A second order system is given by :

$$\frac{C(s)}{R(s)} = \frac{25}{s^2 + 6s + 25}$$

Find its rise time, peak time, peak overshoot and settling time if subjected to unit step input. Also calculate expression for its output response.

Solution :

Comparing $\frac{C(s)}{R(s)} = \frac{25}{s^2 + 6s + 25}$ with $\frac{C(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}$.

* $\omega_n^2 = 25$

$\omega_n = 5 \text{ rad/sec.}$

* $2\xi\omega_n = 6$

$\xi = \frac{6}{5 \times 2} = \frac{6}{10} \Rightarrow \xi = 0.6.$

* $\theta = \tan^{-1} \frac{\sqrt{1-\xi^2}}{\xi} = 0.9272 \text{ radians.}$

* $\omega_d = \omega_n \sqrt{1-\xi^2} = 5 \left[\sqrt{1-(0.6)^2} \right]$

$\omega_d = 4 \text{ rad/sec.}$

* $T_r = \frac{\pi - \theta}{\omega_d} = \frac{\pi - 0.9272}{4}$

$T_r = 0.5535 \text{ second}$

$$* \quad T_p = \frac{\pi}{\omega_d} = \frac{\pi}{4} = 0.785 \text{ second}$$

$$\boxed{T_p = 0.785} \text{ second.}$$

$$* \quad \% M_p = e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}} \times 100$$

$$\boxed{M_p\% = 9.48\%}$$

$$* \quad T_s = \frac{4}{\xi\omega_n} = 1.33 \text{ sec.}$$

$$\boxed{T_s = 1.33 \text{ sec.}}$$

for 2 % tolerance.

$$* \quad C(t) = 1 - \frac{e^{-\xi\omega_n t}}{\sqrt{1-\xi^2}} \sin[\omega_d t + \theta]$$

$$= 1 - \frac{e^{-3t}}{\sqrt{1-(0.6)^2}} \sin[4t + 0.9272]$$

$$\boxed{C(t) = 1 - 1.5625 e^{-3t} \sin[4t + 0.9272]}$$

Type I Method

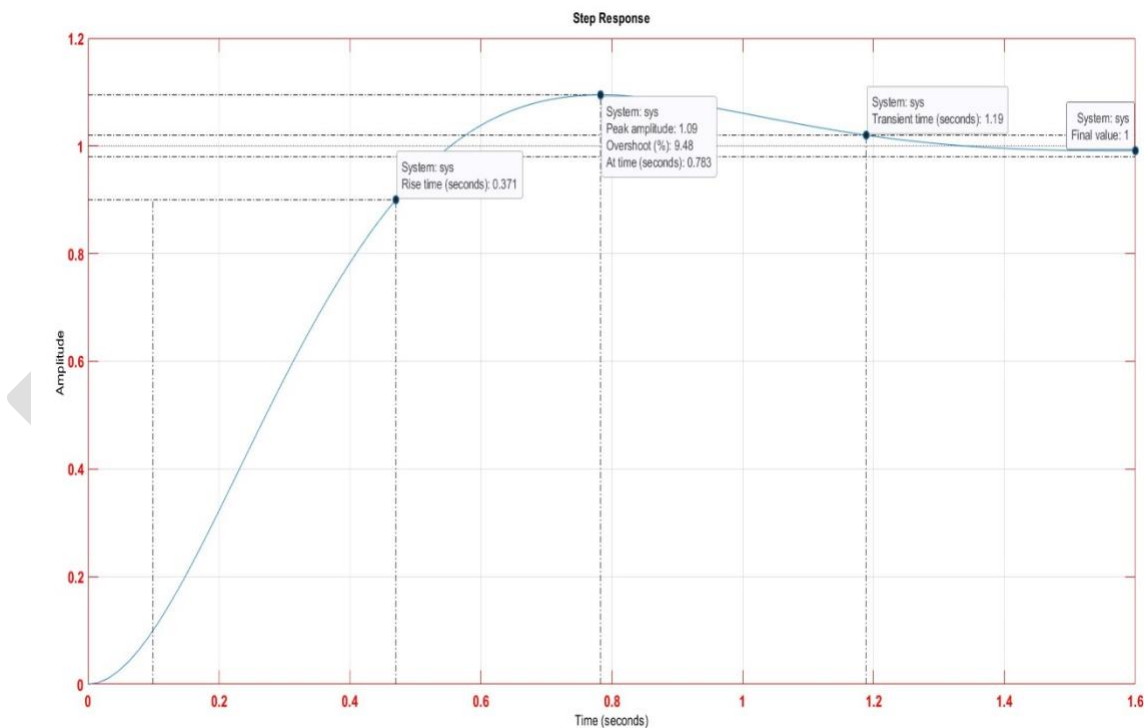
```
clc;
clear;
num=[25];
den=[1 6 25];
sys=tf(num,den)
R=roots(den)
s=stepinfo(sys,'RiseTimeLimits',[0.00,1.00])
ltiview(sys)
```

MATLAB OUTPUT**sys =****25**

$$\frac{25}{s^2 + 6s + 25}$$

Continuous-time transfer function.

Model Properties

R =**-3.0000 + 4.0000i****-3.0000 - 4.0000i****s = struct with fields:****RiseTime: 0.5536****TransientTime: 1.1887****SettlingTime: 1.1887****SettlingMin: 0.9910****SettlingMax: 1.0948****Overshoot: 9.4773****Undershoot: 0****Peak: 1.0948****PeakTime: 0.7829****Figure 6.1 Time domain specifications of given second order system using LTIVIEW**

Type II Method

```
clc;
clear;
num=[25];% Define Numerator
den=[1 6 25];%Define Denominator
sys=tf(num,den)%Transfer Function Generation
disp('The natural frequency is');
wn=sqrt(25)% Calculate Natural frequency
disp('The damping ratio is');
dr=6/(2*wn)% Calculate Damping ratio
disp('The damping frequency is');
wd=wn*sqrt(1-(dr^2))% Calculate Damping Frequency
disp('The Theta value is');
teta=atan(sqrt(1-(dr^2))/dr)%Calculate Theta Value
disp('The Delay time is');
td=((1+0.7*dr)/wn)%Delay Time Calculation
disp('The Rise time is');
tr=((pi-teta)/wd) % Rise time Calculation
disp('The peak time is');
tp=pi/wd %Peak time Calculation
disp('The maximum peak overshoot is');
POS=100*exp((-pi*dr)/(sqrt(1-(dr^2))))% Maximum Overshoot Calculation
disp('The settling time is');
ts=4/(dr*wn)% Settling time at 2%
ts=3/(dr*wn)% Settling time at 5%
```

MATLAB OUTPUT

```
sys =  
      25  
-----  
s^2 + 6 s + 25
```

Continuous-time transfer function.

Model Properties

```
The natural frequency is wn      = 5  
The damping ratio is dr          = 0.6000  
The damping frequency is wd      = 4  
The Theta value is teta         = 0.9273  
The Delay time is td             = 0.2840  
The Rise time is tr              = 0.5536  
The peak time is tp              = 0.7854  
The maximum peak overshoot is POS = 9.4780  
The settling time is ts          = 1.3333 ; ts = 1
```

RESULT:

The Time Domain Specifications for the given second order system is verified using MATLAB

Experiment 7
RESPONSE OF THE GIVEN TRANSFER FUNCTION WITH RESPECT TO CHANGE
IN DAMPING FACTOR

7.1 AIM:

Obtain the response of the given transfer function with respect to change in damping factor.

7.2 APPARATUS REQUIRED

SLNo	Equipment	Quantity
1.	Personal Computer with Windows operating system	01
2.	MATLAB Software.	01

7.3 Time Domain Specifications of Second Order System

Transfer Function of Second Order System

$$\frac{C(S)}{R(S)} = \frac{\omega_n^2}{S^2 + 2\zeta\omega_n S + \omega_n^2}$$

Damping is an effect created in an oscillatory system that reduces, restricts or prevents the oscillations in the system

Systems can be classified as follows depending on damping effect:

1. **Overdamped systems:** Transients in the system exponentially decays to steady state without any oscillations
2. **Critically damped systems:** Transients in the system decay to steady state without any oscillations in shortest possible time
3. **Underdamped systems:** System transients oscillate with the amplitude of oscillation gradually decreasing to zero
4. **Undamped systems:** System keeps on oscillating at its natural frequency without any decay in amplitude

Application of Damped Systems

Overdamped systems:

1. Push button water tap shut-off valves
2. Automatic door closers (can be critically damped also)

Critically damped systems:

1. Elevator mechanism
2. Gun mechanism (returns to neutral position in shortest possible time)

Underdamped systems:

1. All string instruments, bells are underdamped to make sound appealing
2. Analog electrical or mechanical measuring instruments

Table 7.1 Second Order System Classification based on various damping ratio values

Sr. No.	Range of ξ	Types of closed loop poles	Nature of response	System classification
1	$1 < \xi < \infty$	Real, unequal and negative	Purely exponential	Overdamped
2	$\xi = 1$	Real, equal and negative	Critically pure exponential	Critically damped
3	$0 < \xi < 1$	Complex conjugate with negative real part.	Damped oscillations	Underdamped
4	$\xi = 0$	Purely imaginary	Oscillations with constant frequency and amplitude	Underdamped

Given transfer function

$$\frac{C(S)}{R(S)} = \frac{400}{S^2 + 20S + 400}$$

(i) For Under damped system ($0 < \zeta < 1$)

$$\text{From above transfer function } \omega_n^2 = 400 \quad \omega_n = 20$$

$$2\zeta\omega_n = 20$$

$$2 \times \zeta \times 20 = 20$$

$$\zeta = 0.5$$

(ii) For Critically damped system ($\zeta=1$)

Given transfer function

$$\frac{C(S)}{R(S)} = \frac{400}{S^2 + 40S + 400}$$

From above transfer function $\omega_n^2 = 400$ $\omega_n = 20$

$$2\zeta\omega_n = 40$$

$$2 \times \zeta \times 20 = 40$$

$$\zeta = 1.0$$

(iii) For Over damped system ($\zeta>1$)

Given transfer function

$$\frac{C(S)}{R(S)} = \frac{400}{S^2 + 60S + 400}$$

From above transfer function $\omega_n^2 = 400$ $\omega_n = 20$

$$2\zeta\omega_n = 60$$

$$2 \times \zeta \times 20 = 60$$

$$\zeta = 1.5$$

(i) For Undamped system ($\zeta=0$)

Given transfer function

$$\frac{C(S)}{R(S)} = \frac{400}{S^2 + 400}$$

From above transfer function $\omega_n^2 = 400$ $\omega_n = 20$

$$2\zeta\omega_n = 0$$

$$2 \times \zeta \times 20 = 0$$

$$\zeta = 0$$

MATLAB Program

```
t=0:0.1:5
num1=[400];
den1=[1 20 400];
figure(1)
s1=step(num1,den1,t);
plot(s1,'b');
title('Step response for different Zeta values systems');
xlabel('time(s)');
ylabel('amplitude');
hold on
den2=[1 40 400];
s2=step(num1,den2,t);
plot(s2,'g');
hold on
den3=[1 60 400];
s3=step(num1,den3,t);
plot(s3,'o');
hold on
den4=[1 0 400];
s4=step(num1,den4,t);
plot(s4,'r')
```

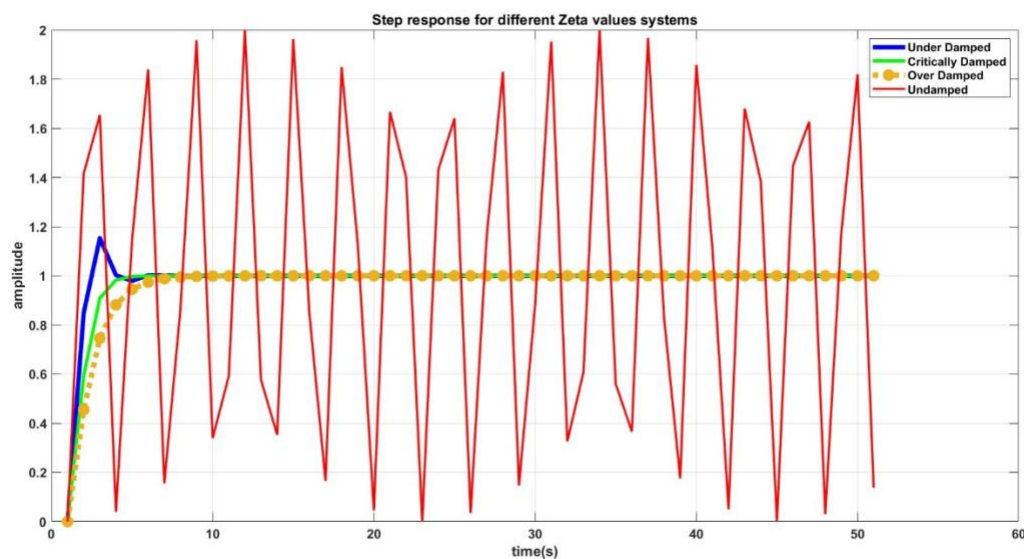
MATLAB OUTPUT

Figure 7.1 Response of a Second order System for various Zeta Values

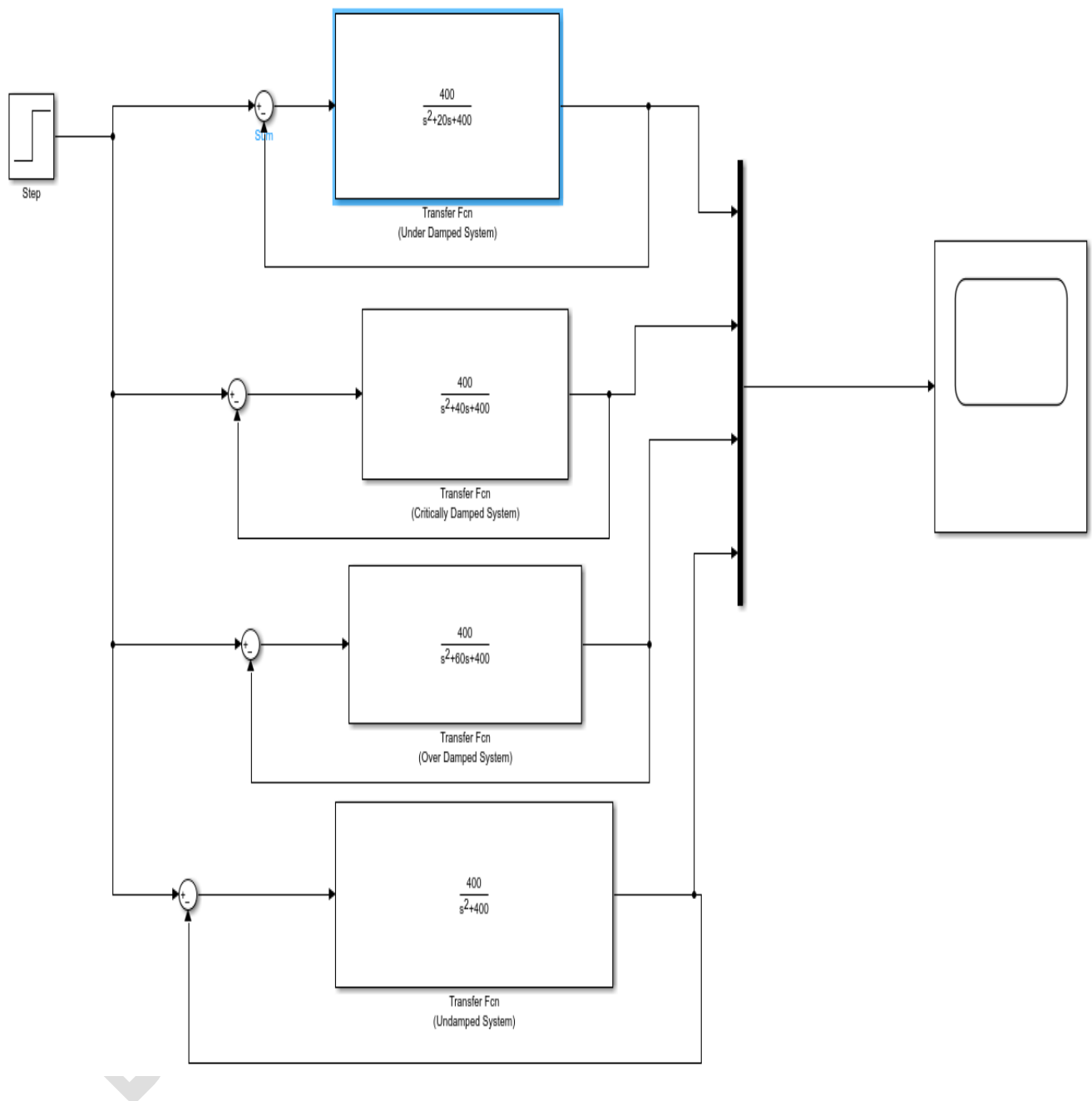


Figure 7.2 SIMULINK MODEL of a Second order System for various Zeta Values

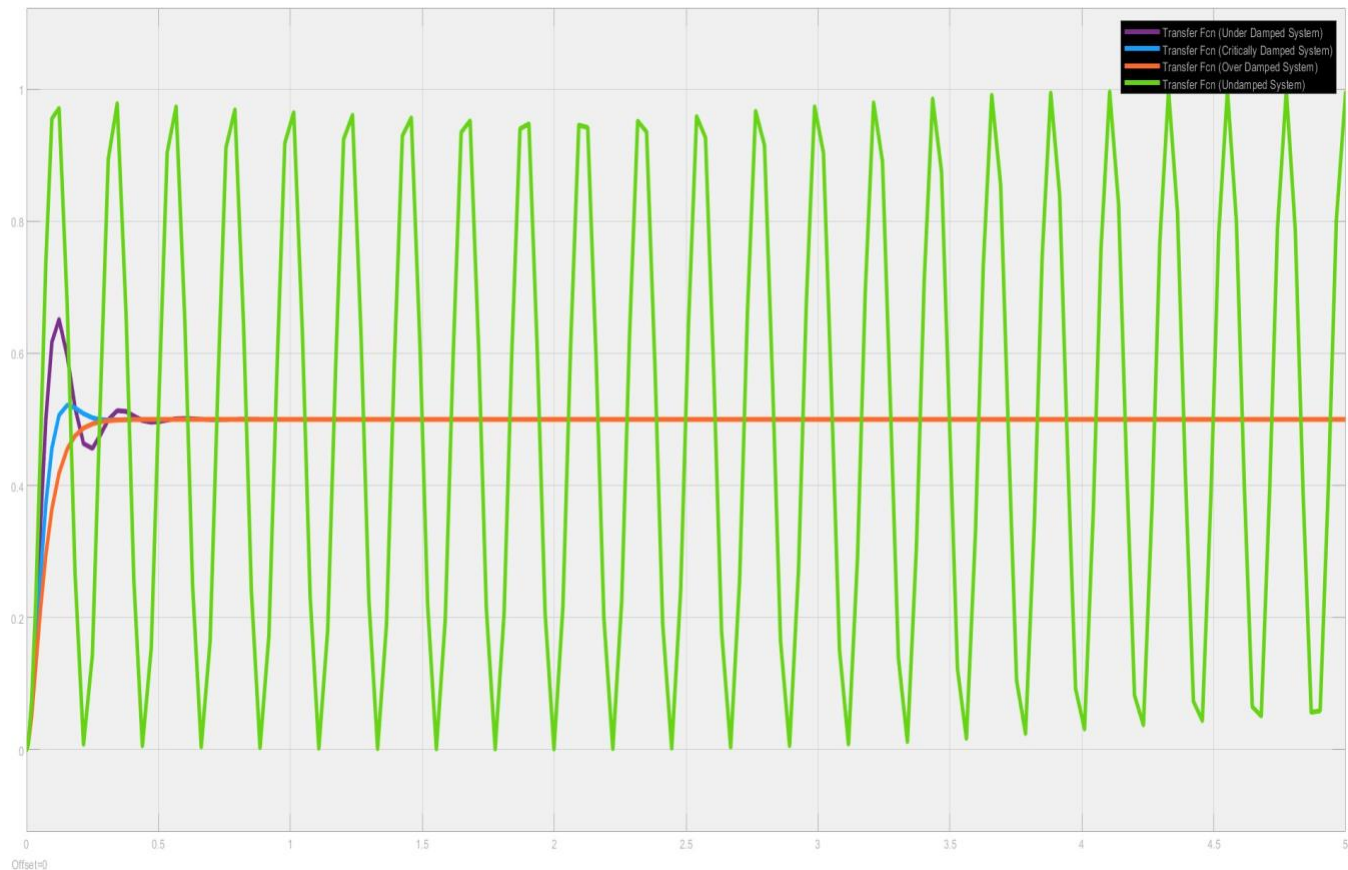
MATLAB OUTPUT

Figure 7.3 SIMULINK MODEL response of a Second order System for various Zeta Values

RESULT:

The step response for various zeta values of second order system is verified using MATLAB.

Experiment 8 STABILITY ANALYSIS USING R-H METHOD IN MATLAB ENVIRONMENT

8.1 AIM:

Obtain the stability of the given control system using R-H method in MATLAB environment.

8.2 APPARATUS REQUIRED

Sl.No	Equipment	Quantity
1.	Personal Computer with Windows operating system	01
2.	MATLAB Software.	01

8.3 Determine the stability of following system using R-H method

The Characteristic Equation of the given system is

$$P(S) = S^5 + 3S^4 + 5S^3 + 4S^2 + S + 3$$

Table 8.1 R-H for given characteristic equation

s^5	1	5	1
s^4	3	4	3
s^3	3.667	0	0
s^2	4	3	0
s^1	-2.75	0	0
s^0	3	0	0

2 rhp; 3 lhp

There are two poles on RHS of S plane. Hence system is unstable.

MATLAB Program

```
%% Routh-Hurwitz stability criterion
% The Routh-Hurwitz stability criterion is a necessary (and
frequently
% sufficient) method to establish the stability of a single-input,
% single-output(SISO), linear time invariant (LTI) control system.
% More generally, given a polynomial, some calculations using only
the
% coefficients of that polynomial can lead us to the conclusion that
it
% is not stable.% in this program you must give your system
coefficients and the
% Routh-Hurwitz table would be shown % Initialization
clear ;
close all;
clc
% Taking coefficients vector and organizing the first two rows
coeffVector = input('input vector of your system coefficients: \n i.e.
[an an-1 an-2 ... a0] = ');
ceoffLength = length(coeffVector);
rhTableColumn = round(ceoffLength/2);
% Initialize Routh-Hurwitz table with empty zero array
rhTable = zeros(ceoffLength,rhTableColumn);
% Compute first row of the table
rhTable(1,:) = coeffVector(1,1:2:ceoffLength);
% Check if length of coefficients vector is even or odd
if (rem(ceoffLength,2) ~= 0)
    % if odd, second row of table will be
    rhTable(2,1:rhTableColumn - 1) = coeffVector(1,2:2:ceoffLength);
else
    % if even, second row of table will be
    rhTable(2,:) = coeffVector(1,2:2:ceoffLength);
end
%% Calculate Routh-Hurwitz table's rows
% Set epss as a small value
epss = 0.01;
% Calculate other elements of the table
for i = 3:ceoffLength
    % special case: row of all zeros
    if rhTable(i-1,:) == 0
        order = (ceoffLength - i);
        cnt1 = 0;
        cnt2 = 1;
```

```
    for j = 1:rhTableColumn - 1
        rhTable(i-1,j) = (order - cnt1) * rhTable(i-2,cnt2);
        cnt2 = cnt2 + 1;
        cnt1 = cnt1 + 2;
    end
end

for j = 1:rhTableColumn - 1
    % first element of upper row
    firstElemUpperRow = rhTable(i-1,1);

    % compute each element of the table
    rhTable(i,j) = ((rhTable(i-1,1) * rhTable(i-2,j+1)) - ....
        (rhTable(i-2,1) * rhTable(i-1, j+1))) / firstElemUpperRow;
end
% special case: zero in the first column
if rhTable(i,1) == 0
    rhTable(i,1) = epss;
end
end

%% Compute number of right hand side poles(unstable poles)
% Initialize unstable poles with zero
unstablePoles = 0;
% Check change in signs
for i = 1:coeffLength - 1
    if sign(rhTable(i,1)) * sign(rhTable(i+1,1)) == -1
        unstablePoles = unstablePoles + 1;
    end
end
% Print calculated data on screen
fprintf('\n Routh-Hurwitz Table:\n')
rhTable
% Print the stability result on screen
if unstablePoles == 0
    fprintf('~~~~~> it is a stable system! <~~~~~\n')
else
    fprintf('~~~~~> it is an unstable system! <~~~~~\n')
end
fprintf('\n Number of right hand side poles =%2.0f\n',unstablePoles)
reply = input('Do you want roots of system be shown? Y/N ', 's');
if reply == 'y' || reply == 'Y'
    sysRoots = roots(coeffVector);
    fprintf('\n Given polynomial coefficients roots :\n')
    sysRoots
```

end

MATLAB OUTPUT

input vector of your system coefficients:

i.e. $[a_n \ a_{n-1} \ a_{n-2} \ \dots \ a_0] = [1 \ 3 \ 5 \ 4 \ 1 \ 3]$

Routh-Hurwitz Table:

rhTable =

1.0000	5.0000	1.0000
3.0000	4.0000	3.0000
3.6667	0	0
4.0000	3.0000	0
-2.7500	0	0
3.0000	0	0

~~~~~> it is an unstable system! <~~~~~

Number of right hand side poles = 2

Do you want roots of system be shown? Y/N Y

Given polynomial coefficients roots :

sysRoots =

-1.6313 + 0.0000i  
-0.9407 + 1.5042i  
-0.9407 - 1.5042i  
0.2563 + 0.7201i  
0.2563 - 0.7201i

### RESULT:

The stability of given system is obtained using R-H method and verified using MATLAB.

## Experiment 9.1

## STABILITY ANALYSIS USING ROOT LOCUS

**9.1 AIM:**

Sketch the root locus for the unity feedback system whose OLTF is

$$G(s)H(s) = \frac{k(s + 5)}{(s^2 + 4s + 20)}$$

**9.2 APPARATUS REQUIRED**

| Sl.No | Equipment                                       | Quantity |
|-------|-------------------------------------------------|----------|
| 1.    | Personal Computer with Windows operating system | 01       |
| 2.    | MATLAB Software.                                | 01       |

**9.3 Theory**

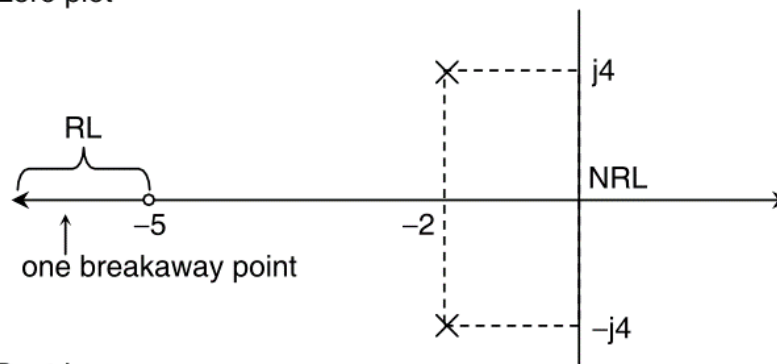
*Solution :*

Step 1 : Number of poles  $P = 2$ ,  $Z = 1$ ,  $N = P$ . One branch has to terminate at finite zeros  $s = -5$  while  $P - Z = 1$  branch has to terminate at  $\infty$ .

Starting points of branches are

$$\frac{-4 \pm \sqrt{16 - 80}}{2} = -2 \pm j4$$

Step 2 : Pole - Zero plot



NRL → No Root Locus

RL → Root Locus

## Step 3 : Angles of asymptotes

One branch approaches to  $\infty$  so one asymptote is required.

$$\theta = \frac{(2q+1)180^\circ}{P-Z}, \quad q=0$$

$$\therefore \theta = 180^\circ$$

Branch approaches to  $\infty$  along  $+180^\circ$  i.e. negative real axis.

## Step 4 : Centroid

As there is only one branch approaching to  $\infty$  and one asymptote exists, centroid is not required.

## Step 5 : Breakaway points :

characteristic equation :  $1 + G(s)H(s) = 0$

$$1 + \frac{k(s+5)}{(s^2 + 4s + 20)} = 0$$

$$\therefore s^2 + 4s + 20 + ks + 5k = 0$$

$$\therefore s^2 + 4s + 20 + k(s+5) = 0$$

$$\therefore k = \frac{-s^2 - 4s - 20}{(s+5)}$$

$$\text{Now, } \frac{dk}{ds} = \frac{vu' - uv'}{v^2} = 0$$

$$= (s+5)(-2s-4) - (-s^2 - 4s - 20)(1) = 0$$

$$\text{i.e. } -s^2 - 10s = 0$$

$$\therefore -s(s+10) = 0$$



$s = 0$  and  $s = -10$  are breakaway points. But  $s = 0$  cannot be breakaway point as for  $s = 0$ ,  $k = -4$ .

$$\text{For } s = -10, \quad k = \frac{-100 + 40 - 20}{-10 + 5} = 16$$

Hence  $s = -10$  is valid breakaway point.

Step 6 : Intersection with imaginary axis characteristic equation.

$$s^2 + 4s + 20 + ks + 5k = 0$$

$$s^2 + s(k + 4) + (20 + 5k) = 0$$

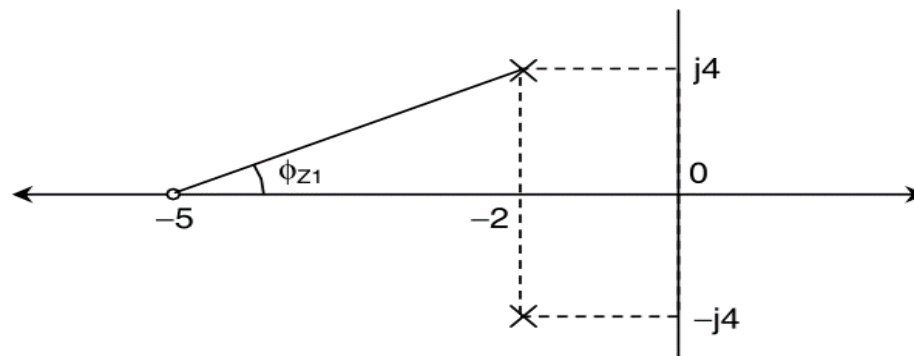
$k_{\max} = -4$  makes  $s$  row as row of zeros.

But as it is negative, there is no intersection of root locus with imaginary axis.

|       |           |           |
|-------|-----------|-----------|
| $s^2$ | 1         | $20 + 5k$ |
| $s^1$ | $k + 4$   | 0         |
| $s^0$ | $20 + 5k$ |           |

Step 7 : Angle of departure

Consider  $-2 + j4$  join remaining pole and zero to it.



$$\phi_{p1} = 90^\circ$$

$$\Sigma \phi_p = 90^\circ$$

$$\phi_{Z1} = \tan^{-1} \frac{4}{3} = 53.13^\circ$$

$$\Sigma \phi_Z = 53.13^\circ$$

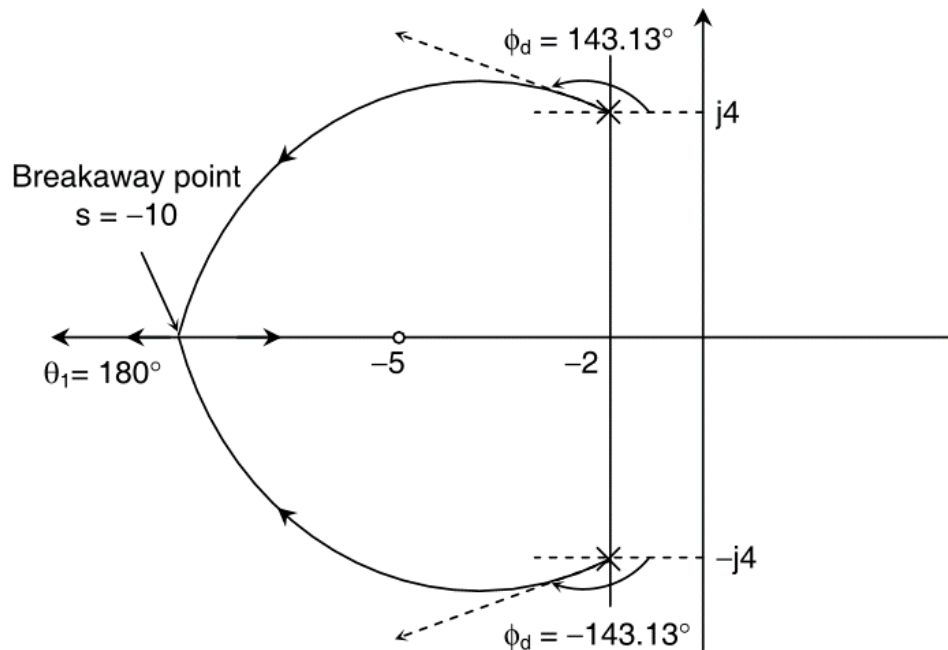
$$\therefore \phi = \Sigma \phi_p - \Sigma \phi_Z = 36.86^\circ$$

$$\phi_d = 180^\circ - \phi$$

$$= 143.13^\circ \quad \text{at } -2 + j4 \text{ pole}$$

$$= -143.13^\circ \quad \text{at } -2 - j4 \text{ pole}$$

Step 8 : Complete Root Locus is using following figure.



Step 9 : Prediction of Stability

For all ranges of  $k$  i.e.  $0 < k < \infty$ , both the roots are always in left half of  $s$ -plane.  
So system is inherently stable.

#### 9.4 PROCEDURE

1. Open MATLAB
2. Go Home menu; Open New Script
3. Type the program in editor window
4. Save the program
5. Run the program and trouble shoot the error if it occurs
6. Obtain the output from command prompt
7. Verify the output with theoretical calculations

#### 9.5 MATLAB PROGRAM

*%find the stability of linear systems using root locus*

*%g(s)=K(S+5)/(s^2+4s+20)*

clc;

clear all;

num=[1 5];

den=[1 4 20];

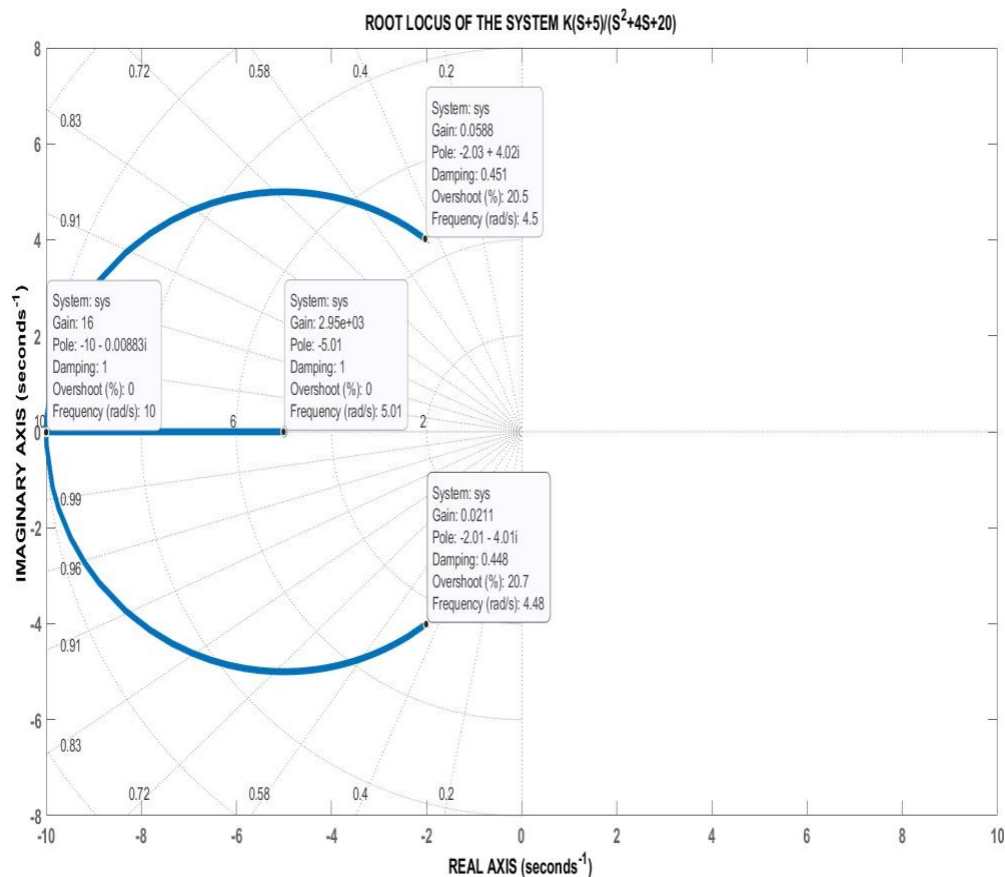
rlocus(num,den)

v=[-10,10,-8,8];

axis(v);

axis('SQUARE')

```
xlabel('REAL AXIS')  
ylabel('IMAGINARY AXIS')  
title('ROOT LOCUS OF THE SYSTEM  $K(S+5)/(S^2+4S+20)$ ')
```

**MATLAB OUTPUT**

**Figure 9.1 Root Locus for the given transfer function**

**RESULT:**

**Root Locus for the given transfer function is simulated in MATLAB Environment.**

## Experiment 9.2

## STABILITY ANALYSIS USING BODE PLOT

**9.2.1 AIM:**

The OLTF of a UBF is  $G(s) = \frac{25000}{s(s+5)(s+50)}$ . Draw the bode plot and find

- i. Gain Margin (GM)
- ii. Phase Margin (PM)
- iii. Gain crossover frequency ( $\omega_{gc}$ )
- iv. Phase crossover frequency ( $\omega_{pc}$ )

**9.2.2 APPARATUS REQUIRED**

| Sl.No | Equipment                                       | Quantity |
|-------|-------------------------------------------------|----------|
| 1.    | Personal Computer with Windows operating system | 01       |
| 2.    | MATLAB Software.                                | 01       |

**9.2.3 Design**

$$G(s) = \frac{25000}{s(s+5)(s+50)}$$

**Step1:**

$$\text{Bode form (Time constant form)} = \frac{25000}{s5\left(\frac{s}{5}+1\right)50\left(\frac{s}{50}+1\right)}$$

$$G(s) = \frac{100}{s(1+0.2s)(1+0.02s)}$$

**Step 2: Corner frequency: -**

$$\frac{1}{(1+0.2s)} = \frac{1}{0.2} = 5 \text{ rad/sec} = \omega_{F1}$$

$$\frac{1}{(1+0.02s)} = \frac{1}{0.02} = 50 \text{ rad/sec} = \omega_{F2}$$

Sinusoidal transfer function  $G(j\omega) = \frac{100}{j\omega(1+0.2j\omega)(1+0.02j\omega)}$

Step 3: Magnitude plot

| Factor                | CF in rad/sec       | Slope<br>dB/dec | Slope at any instant<br>PS+PS (previous slope<br>+present slope) |
|-----------------------|---------------------|-----------------|------------------------------------------------------------------|
| $\frac{100}{s}$       | -                   | -20             | 0-20=-20(slope starting<br>point)                                |
| $\frac{1}{1+0.2s}$    | $\omega_{c_1} = 5$  | -20             | -20-20=-40( $\omega_{c_1}$ to $\omega_{c_2}$ )                   |
| $\frac{1}{(1+0.02s)}$ | $\omega_{c_2} = 50$ | -20             | -20-40=-60(net slope from<br>$\omega_{c_2}$ to $\omega_h$ )      |

$$\omega_h = 100 \text{ rad/sec}$$

$$\omega_L = 0.1 \text{ rad/sec}$$

$$\text{At } \omega = 0.1 \quad A = 20 \log \left| \frac{100}{\omega} \right|_{\omega=0.1} = 60 \text{ dB}$$

$$\text{At } \omega = 5 \quad A = 20 \log \left| \frac{100}{\omega} \right|_{\omega=5} = 26 \text{ dB}$$

$$\text{At } \omega = 50 \quad A = \{SC \text{ from } \omega_{c_1} \text{ to } \omega_{c_2}\} X \log \frac{\omega_{c_2}}{\omega_{c_1}} + A \Big|_{\omega=5} = -14 \text{ dB}$$

$$= -40 X \log \frac{50}{5} + 26 = -14 \text{ dB}$$

$$\text{At } \omega_h = 100$$

$$A = \{SC \text{ from } \omega_{c_2} \text{ to } \omega_h\} X \log \frac{\omega_h}{\omega_{c_2}} + A \Big|_{\omega=50}$$

$$= -60 X \log \frac{100}{50} - 14$$

$$A = -32 \text{ dB}$$

## Magnitude plot points

|          |     |    |     |     |
|----------|-----|----|-----|-----|
| $\omega$ | 0.1 | 5  | 50  | 100 |
| M        | 60  | 26 | -14 | -32 |

## Phase plot

$$\angle G(j\omega) = -90^\circ - \tan^{-1}(0.2\omega) - \tan^{-1}(0.02\omega)$$

|      |       |       |       |       |
|------|-------|-------|-------|-------|
| 0.1  | 5     | 10    | 50    | 100   |
| -91° | -141° | -165° | -219° | -241° |

$$\angle G(j\omega) = -90^\circ - \tan^{-1}(0.2\omega) - \tan^{-1}(0.02\omega)$$

At 0.1  $\angle G(j\omega) = -90^\circ - 1.145^\circ - 0.1145^\circ = -91.25^\circ \approx -91^\circ$

At 5  $\angle G(j\omega) = -90^\circ - \tan^{-1} 0.2 \times 5 - \tan^{-1} 0.02 \times 5 = -90^\circ - 45^\circ - 5.71^\circ$   
 $= -140.7^\circ \approx -141^\circ$

At 50  $\angle G(j\omega) = -90^\circ - \tan^{-1} 0.2 \times 50 - \tan^{-1} 0.02 \times 50 = -90^\circ - 84.28^\circ - 45^\circ$   
 $= -219.28^\circ \approx -219^\circ$

At 100  $\angle G(j\omega) = -90^\circ - \tan^{-1} 0.2 \times 100 - \tan^{-1} 0.02 \times 100$   
 $= -90^\circ - 87.137^\circ - 63.43^\circ$   
 $= -240.57^\circ \approx -241^\circ$

At 10  $\angle G(j\omega) = -90^\circ - \tan^{-1} 0.2 \times 10 - \tan^{-1} 0.02 \times 10$   
 $= -90^\circ - 63.43^\circ - 11.30^\circ$   
 $= -164.73^\circ \approx -165^\circ$





### **9.2.4 PROCEDURE**

1. Open MATLAB
2. Go Home menu; Open New Script
3. Type the program in editor window
4. Save the program
5. Run the program and trouble shoot the error if it occurs
6. Obtain the output from command prompt
7. Verify the output with theoretical calculations

### **9.2.5 MATLAB PROGRAM**

```
num=[25000 ];  
den=[1 55 250 0];  
w=logspace(-1,3,100);  
figure(1);  
bode(num,den,w);  
title('BODE PLOT FOR THE GIVEN TRANSFER FUNCTION  
G(s)=25000/s(s+5)(s+50)')  
grid;  
[Gm Pm wcg wcp] =margin(num,den);  
Gain_Margin_dB=20*log10(Gm)  
Phase_Margin=Pm  
Gaincrossover_Frequency=wcg  
Phasecrossover_Frequency=wcp
```



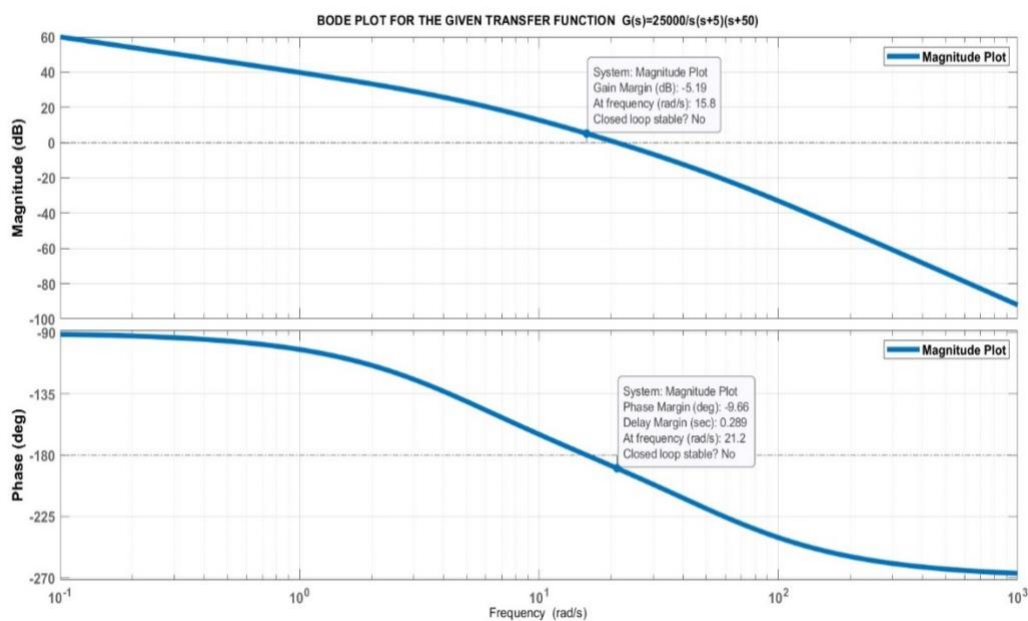
**MATLAB OUTPUT**

Figure 9.2.1 Bode Plot for the transfer function  $G(s) = \frac{25000}{s(s+5)(s+50)}$

**RESULT:**

Bode Plot for the given transfer function is simulated in MATLAB Environment.

| S.No                                                   | Frequency Domain Values        | Theoretical (Calculated) Values | MATLAB Output |
|--------------------------------------------------------|--------------------------------|---------------------------------|---------------|
| 1.                                                     | Gain Margin(Gm)                | - 3 dB                          | -5.1927 dB    |
| 2.                                                     | Phase Margin(Pm)               | - 8 °                           | -9.6566 °     |
| 3.                                                     | Gain Crossover frequency(Wcg)  | 23 rad/sec                      | 21.1685       |
| 4.                                                     | Phase Crossover frequency(Wcp) | 19 rad/sec                      | 15.8114       |
| Both GM,PM are Negative, Hence the system is Unstable. |                                |                                 |               |

### Experiment 9.3 STABILITY ANALYSIS USING NYQUIST PLOT

#### 9.3.1 AIM:

The OLTF of a UBF is  $G(s) = \frac{K}{s(s+1)(s+2)}$ . Draw the Nyquist plot and find

- i. Gain Margin (GM)
- ii. Phase Margin (PM)
- iii. Gain crossover frequency ( $\omega_{gc}$ )
- iv. Phase crossover frequency ( $\omega_{pc}$ )

#### 9.3.2 APPARATUS REQUIRED

| Sl.No | Equipment                                       | Quantity |
|-------|-------------------------------------------------|----------|
| 1.    | Personal Computer with Windows operating system | 01       |
| 2.    | MATLAB Software.                                | 01       |

#### 9.3.3 Design

Assume  $K=1$

Given

$$G(s)H(s) = \frac{k}{s(s+1)(s+2)}$$

$$G(j\omega)H(j\omega) = \frac{k}{(j\omega)(j\omega+1)(j\omega+2)}$$

**Magnitude**

$$\begin{aligned}
 M &= |G(j\omega)H(j\omega)| \\
 &= \frac{\sqrt{k^2 + 0^2}}{\sqrt{0^2 + \omega^2} \sqrt{\omega^2 + 1^2} \sqrt{\omega^2 + 2^2}}
 \end{aligned}$$

$$M = \frac{k}{\omega\sqrt{\omega^2+1}\sqrt{\omega^2+4}}$$

**Phase Angle**

$$PA = \phi = \text{ang} (G(j\omega)H(j\omega))$$

$$G(j\omega)H(j\omega) = \frac{k}{(j\omega)(j\omega+1)(j\omega+2)}$$

$$\text{ang}G(j\omega)H(j\omega) = 0 - \tan^{-1}\left(\frac{\omega}{0}\right) - \tan^{-1}\left(\frac{\omega}{1}\right) - \tan^{-1}\left(\frac{\omega}{2}\right)$$

$$\text{ang}G(j\omega)H(j\omega) = -90^\circ - \tan^{-1}\left(\frac{\omega}{1}\right) - \tan^{-1}\left(\frac{\omega}{2}\right)$$

(i) **Phase Cross Over Frequency( $\omega_{pc}$ )**

To calculate the  $\omega_{pc}$  equate the  $\text{ang} G(j\omega)H(j\omega)$  to  $-180^\circ$

$$-90^\circ - \tan^{-1}\left(\frac{\omega}{1}\right) - \tan^{-1}\left(\frac{\omega}{2}\right) = -180^\circ$$

$$-\tan^{-1}\left(\frac{\omega}{1}\right) - \tan^{-1}\left(\frac{\omega}{2}\right) = -180^\circ + 90^\circ$$

$$\tan^{-1}\left(\frac{\omega}{1}\right) - \tan^{-1}\left(\frac{\omega}{2}\right) = 90^\circ$$

**WKT**

$$\tan(A+B) = \frac{\tan A + \tan B}{1 - \tan A \tan B}$$

$$\tan^{-1}\left[\frac{\omega + \omega/2}{1 - \left(\omega \times \frac{\omega}{2}\right)}\right] = 90^\circ$$

X by tan on both sides.

$$\frac{\omega + \frac{\omega}{2}}{1 - \frac{\omega^2}{2}} = \tan 90^\circ = \infty$$

$$1 - \frac{\omega^2}{2} = 0$$

$$2 - \omega^2 = 0$$

$$\omega^2 = 2$$

$$\omega = \sqrt{2}$$

$$\omega_{pc} = 1.414 \frac{\text{rad}}{\text{sec}}$$

(ii) **Gain Crossover Frequency:** As per definition the frequency at which the magnitude becomes unity

$$|G(j\omega)H(j\omega)| = 1$$

$$\frac{1}{\omega\sqrt{\omega^2 + 1}\sqrt{\omega^2 + 4}} = 1$$

Trial & Error

$$\omega = 0.5$$

$$\frac{1}{\omega\sqrt{\omega^2 + 1}\sqrt{\omega^2 + 4}} = \frac{1}{0.5\sqrt{(0.5)^2 + 1}\sqrt{(0.5)^2 + 4}} = 0.87$$

$$\omega = 0.45$$

$$\frac{1}{0.45\sqrt{(0.45)^2 + 1}\sqrt{(0.45)^2 + 4}} = 0.98$$

Hence Gain Crossover Frequency

$$\omega_{gc} = 0.45 \text{ rad/sec}$$

(iii) **Gain margin**

As per definition The gain margin =  $\frac{1}{|G(j\omega)H(j\omega)|_{\omega=\omega_{pc}}}$

$$\begin{aligned} &= \frac{1}{\frac{1}{\omega\sqrt{\omega^2 + 1}\sqrt{\omega^2 + 4}}} \\ &= \omega\sqrt{\omega^2 + 1}\sqrt{\omega^2 + 4} \end{aligned}$$

$$\begin{aligned}
 &= 1.414\sqrt{(\sqrt{2})^2 + 1}\sqrt{(\sqrt{2})^2 + 4} \\
 &= 1.414\sqrt{2 + 1}\sqrt{2 + 4} \\
 &= 1.414 \times 1.732 \times 2.44
 \end{aligned}$$

**Gain Margin = 6**

**(iv) Phase Margin( $\gamma$ ) :**

Phase Margin is obtained by substituting  $\omega_{gc}$  in Phase angle equation

$$\begin{aligned}
 |G(j\omega)H(j\omega)|_{\omega=\omega_{gc}} + 180^\circ &= -90^\circ - \tan^{-1} \omega - \tan^{-1} \frac{\omega}{2} \\
 &= -90^\circ - \tan^{-1}(0.45) - \tan^{-1} \frac{0.45}{2} + 180^\circ \\
 &= -90^\circ - 24.22^\circ - 12.68^\circ + 180^\circ \\
 &= 53.1^\circ \cong 53^\circ = PM
 \end{aligned}$$

### **9.3.4 PROCEDURE**

1. Open MATLAB
2. Go Home menu; Open New Script
3. Type the program in editor window
4. Save the program
5. Run the program and trouble shoot the error if it occurs
6. Obtain the output from command prompt
7. Verify the output with theoretical calculations

### **9.3.5 MATLAB PROGRAM**

```

num=[1];
den=[1 3 2 0];
figure(1);
nyquist(num,den)
title('NYQUIST PLOT FOR THE TRANSFER FUNCTION G(s)=1/S(S+1)(S+2)')
[Gm,Pm,Wcg,Wcp] =margin(num,den);
Gain_Margin=Gm
Phase_Margin=Pm
PhaseCrossover_Frequency=Wcg
GainCrossover_Frequency=Wcp

```

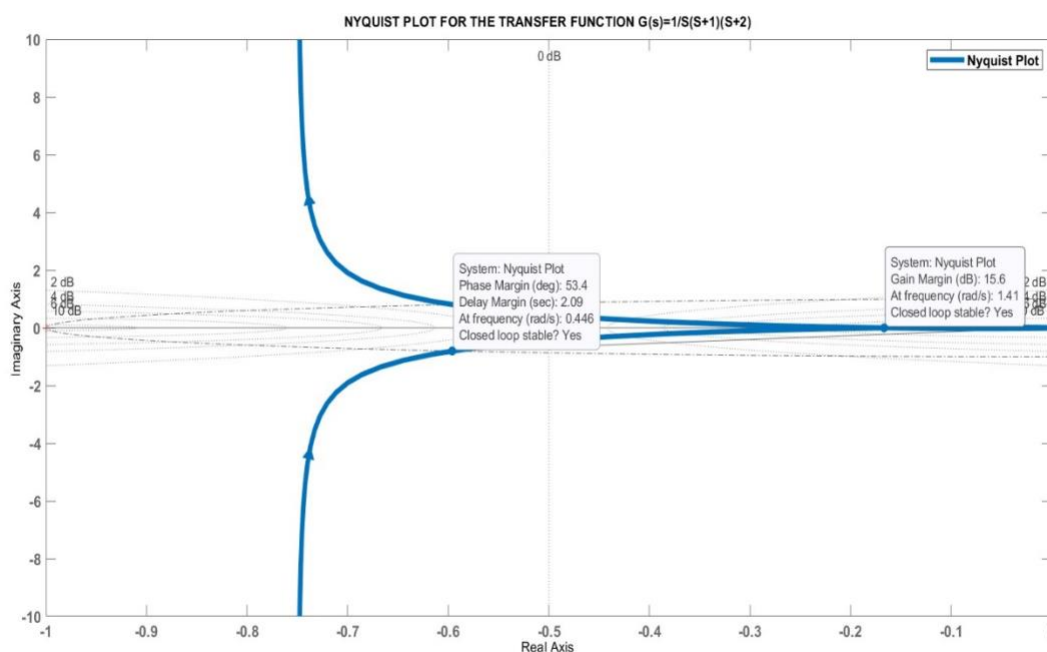
**MATLAB OUTPUT**

Figure 9.3.1 Nyquist Plot for the transfer function  $G(s) = \frac{K}{s(s+1)(s+2)}$

**RESULTS**

Nyquist Plot for the given transfer function is simulated in MATLAB environment

| S.No | Frequency Domain Values                               | Theoretical (Calculated) Values | MATLAB Output  |
|------|-------------------------------------------------------|---------------------------------|----------------|
| 1.   | Gain Margin(Gm)                                       | 15.56 dB                        | 6.0 dB         |
| 2.   | Phase Margin(Pm)                                      | 53.9 °                          | 53.41 °        |
| 3.   | Gain Crossover frequency(Wcg)                         | 0.45 rad/sec                    | 0.4457         |
| 4.   | Phase Crossover frequency(Wcp)                        | 1.414 rad/sec                   | 1.4142 rad/sec |
|      | Both GM,PM are Negative, Hence the system is Unstable |                                 |                |

## Experiment 10 PERFORMANCE CHARACTERISTICS OF P, PD, PI, PID CONTROLLERS

### 10.1 AIM:

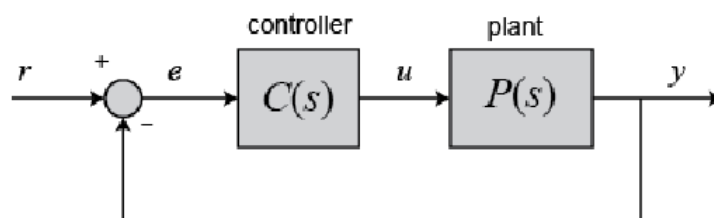
To determine the Performance characteristics of P,PD,PI,PID control

### 10.2 APPARATUS REQUIRED

| Sl.No | Equipment                                       | Quantity |
|-------|-------------------------------------------------|----------|
| 1.    | Personal Computer with Windows operating system | 01       |
| 2.    | MATLAB Software.                                | 01       |

### 10.3 THEORY

Consider the following unity feedback system:



The output of a PID controller, equal to the control input to the plant, in the time-domain is as follows:

$$u(t) = K_p e(t) + K_i \int e(t) dt + K_d \frac{de(t)}{dt} \dots\dots\dots \text{Eq (1)}$$

First, let's take a look at how the PID controller works in a closed-loop system using the schematic shown above. The variable (e) represents the tracking error, the difference between the desired input value (r) and the actual output (y). This error signal (e) will be sent to the PID controller, and the controller computes both the derivative and the integral of this error signal. The control signal (u) to the plant is equal to the proportional gain ( $K_p$ ) times the magnitude of the error plus the integral gain ( $K_i$ ) times the integral of the error plus the derivative gain ( $K_d$ ) times the derivative of the error.

This control signal ( $u$ ) is sent to the plant, and the new output ( $y$ ) is obtained. The new output ( $y$ ) is then fed back and compared to the reference to find the new error signal ( $e$ ). The controller takes this new error signal and computes its derivative and its integral again, ad infinitum.

The transfer function of a PID controller is found by taking the Laplace transform of Eq.(1).

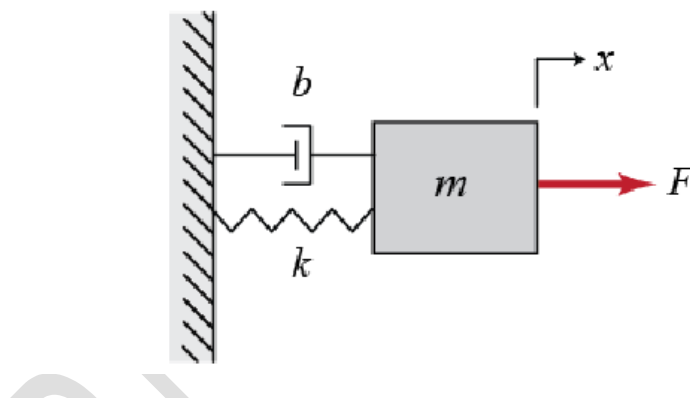
$$K_P + \frac{K_I}{s} + K_D s = \frac{K_D s^2 + K_P s + K_I}{s}$$

$K_P$  = Proportional gain

$K_I$  = Integral gain

$K_D$  = Derivative gain

Suppose we have a simple mass, spring, and damper problem.



The modeling equation of this system is  $M\ddot{x} + b\dot{x} + kx = F$

Taking the Laplace transform of the modeling equation, we get

$$Ms^2X(s) + bsX(s) + kX(s) = F(s)$$

The transfer function between the displacement  $X(s)$  and the input  $F(s)$  then becomes

$$\frac{X(s)}{F(s)} = \frac{1}{Ms^2 + bs + k}$$



**Design Procedure:****Proportional Control**

The closed-loop transfer function of the above system with a proportional controller is:

$$\frac{X(s)}{F(s)} = \frac{K_p}{s^2 + 10s + (20 + K_p)}$$

**Proportional-Derivative Control**

The PD control and the derivative controller ( $K_d$ ) reduces both the overshoot and the settling time. The closed-loop transfer function of the given system with a PD controller is:

$$\frac{X(s)}{F(s)} = \frac{K_d s + K_p}{s^2 + (K_d + 10)s + (20 + K_p)}$$

**Proportional-Integral Control**

Before going into a PID control, we see that an integral controller ( $K_i$ ) decreases the rise time, increases both the overshoot and the settling time, and eliminates the steady-state error. For the given system, the closed-loop transfer function with a PI control is:

$$\frac{X(s)}{F(s)} = \frac{K_p s + K_i}{s^3 + 10s^2 + (20 + K_p)s + K_i}$$

**Proportional-Integral-Derivative Control**

The closed-loop transfer function of the given system with a PID controller is:

$$\frac{X(s)}{F(s)} = \frac{K_d s^2 + K_p s + K_i}{s^3 + (10 + K_d)s^2 + (20 + K_p)s + K_i}$$

Let

$$M = 1 \text{ kg}; \quad b = 10 \text{ N s/m}; \quad k = 20 \text{ N/m}; \quad F = 1 \text{ N}$$

Plug these values into the above transfer function

$$\frac{X(s)}{F(s)} = \frac{1}{s^2 + 10s + 20}$$

### 10.4 PROCEDURE

1. Open MATLAB
2. Go Home menu; Open New Script
3. Type the program in editor window
4. Save the program
5. Run the program and trouble shoot the error if it occurs
6. Obtain the output from command prompt
7. Verify the output with theoretical calculations

### 10.5 MATLAB PROGRAM

```
clc;
clear all;
close all;
%Open-Loop Step Response
a=figure(1);
s = tf('s');
P = 1/(s^2 + 10*s + 20);
step(P);
title('Open loop step response');
grid on;
%Proportional Control
a=figure(2);
Kp = 300;
C = Kp/(s^2+10*s+20+Kp)
T = feedback(C,1);
title('Step response with respective to Different Controllers');
step(T)
title('Response for Different Controllers');
hold on
%Proportional-Derivative Control
Kp = 300;
Kd = 10;
C = (Kd*s+Kp)/(s^2+(Kd+10)*s+20+Kp)
T = feedback(C,1);
step(T)
%Proportional-Integral Control
Kp = 30;
Ki = 70;
C = (Kp*s+Ki)/(s^3+10*s^2+(20+Kp)*s+Ki)
T = feedback(C,1);
step(T)
%Proportional-Integral-Derivative Control
Kp = 350;
Ki = 300;
```

```
Kd = 50;  
C = (Kd*s^2+Kp*s+Ki)/(s^3+(10+Kd)*s^2+(20+Kp)*s+Ki)  
T = feedback(C,1);  
step(T)  
legend('P', 'PD', 'PI', 'PID');  
grid on;  
hold off;
```

### MATLAB OUTPUT

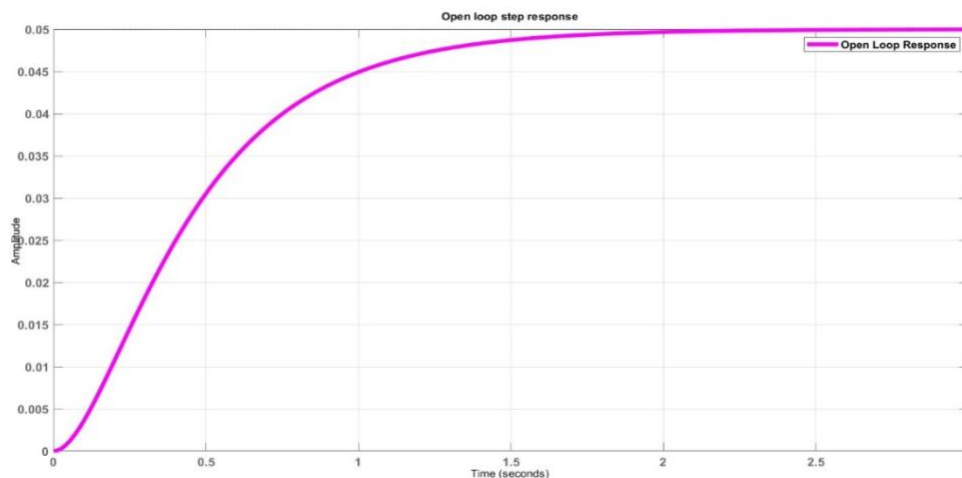


Figure 10.1 Open Loop Step Response for the given transfer function

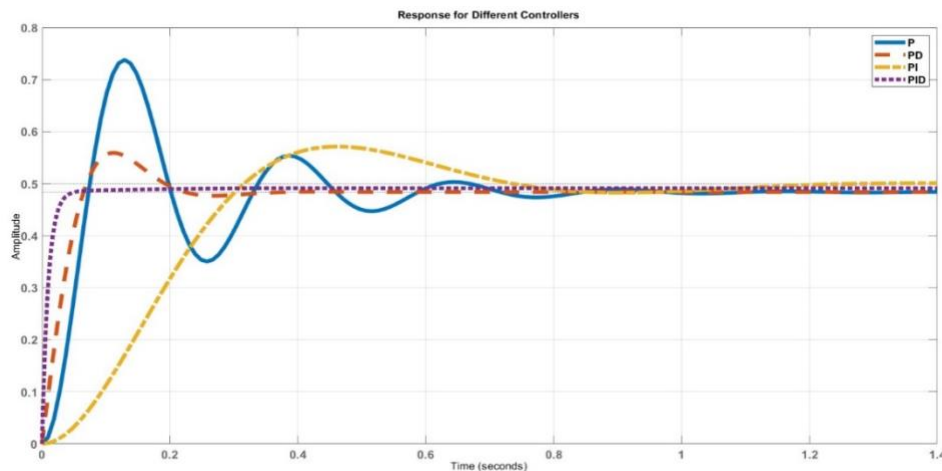
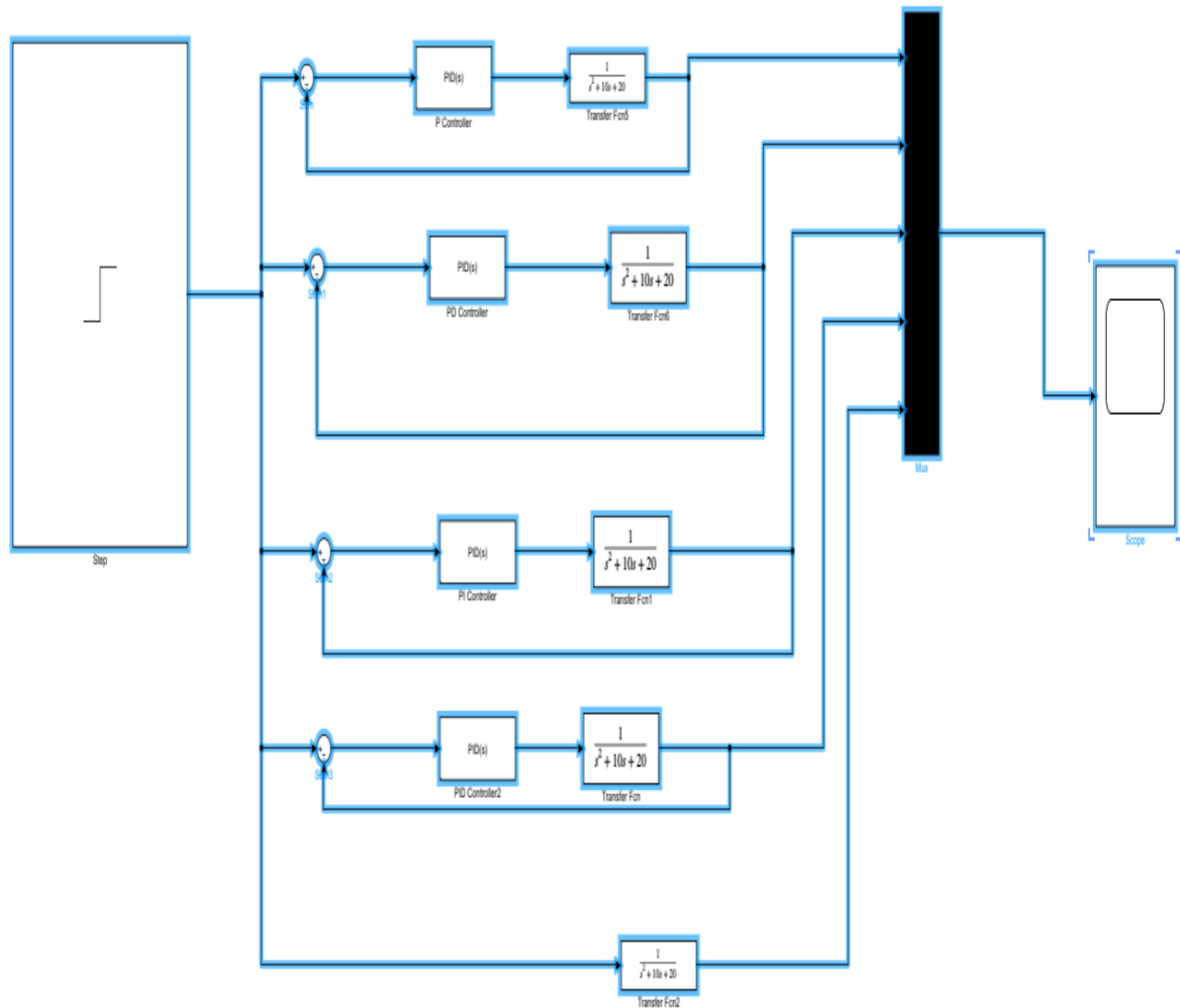
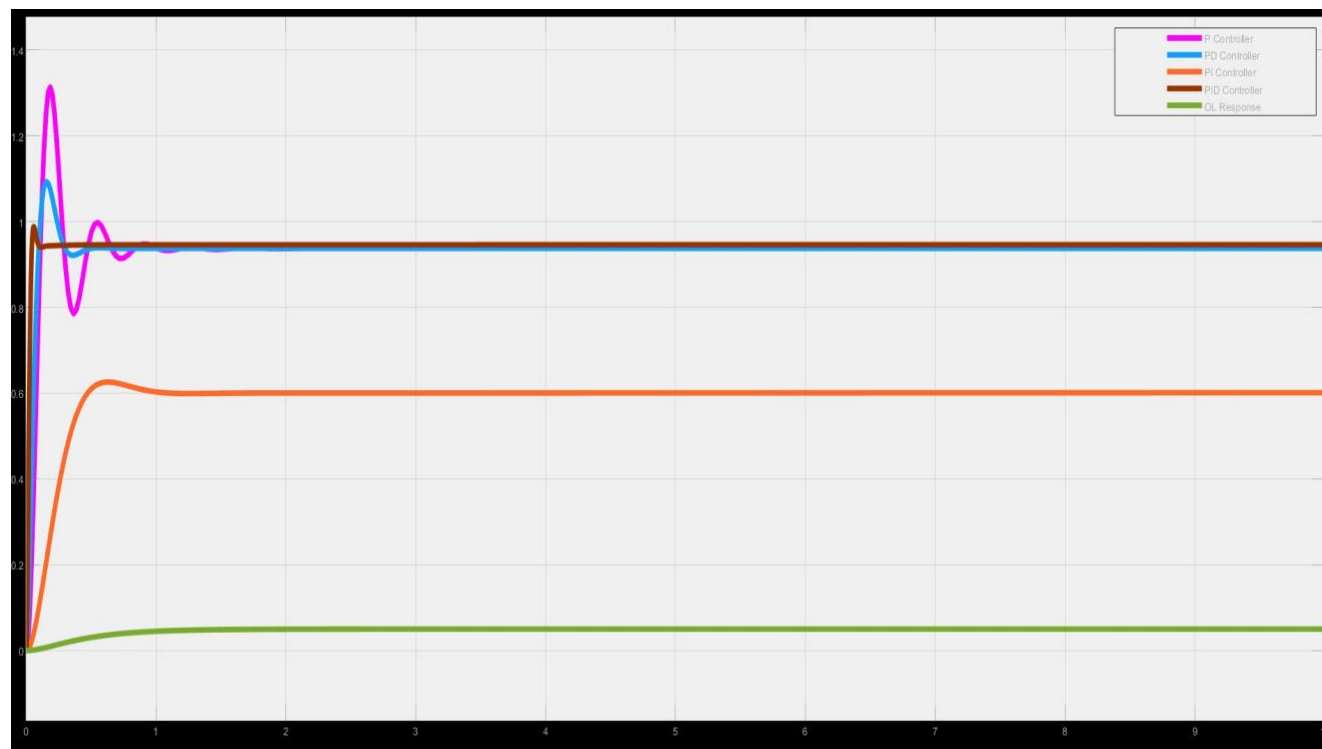


Figure 10.2 Various Controller Response for the given transfer function subjected to step i/p

**10.6 SIMULINK MODEL FOR CONTROLLERS**

**Figure 10.3 Simulink Model for various Controllers Response for the given transfer function subjected to step i/p**

**SIMULINK MODEL OUTPUT**

**Figure 10.4 Simulink Model Output for various Controllers Response for the given transfer function subjected to step i/p**

**RESULTS**

The characteristics various controller response for the given transfer function subjected to step i/p is verified using MATLAB.