Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRICAL ENGINEERING

EEE3001

CONTROL SYSTEMS

LABORATORY PRACTICE

FALL 2021 -2022

Name :
Reg.No

Table of Contents

Understanding Systems with MATLAB

Ex. No: 1 **Date:**

AIM:

The objective of this session is to in

1. Visualizing discrete time signals
2. Understanding Sampling Theorem
3. Finding the solution of differential equation
4. Find the impulse and step response

PRACTICE:

Code, Execute and obtain the plots for each of the following as shown in an example.

1. Write a code in Matlab to plot a 3 cycles of a sine and cosine wave of 50 Hz frequency

2. Plot the signals
   a. $x(t) = e^{5t}\cos(100\pi t)$
   b. $x(t) = e^{-5t}\cos(100\pi t)$,
   c. $x(t) = e^{-j100\pi t}$
   d. $x(t) = \frac{sin(100\pi t)}{(100\pi t)}$

3. Two continuous time signals of frequency fc1 and fc2 are sampled with a sampling frequency of fs=1 kHz. Find the frequencies for which the discrete time signals that will be identical. Plot the signals for the same time scale.

a. $\frac{dy}{dx} + 10y = 0$

a. $\frac{dy}{dx} + 10y = 0$ tial equations

a. $\frac{dy}{dx} + 10y = 0$

b. $\frac{d^2y}{dx^2} + 10\frac{dy}{dx} + 25y = 0$

c. $\frac{d^2y}{dx^2} + 25y = 0$

5. Find the step and impulse response of the system with transfer functions

a. $G(s) = \frac{1}{s+5}$

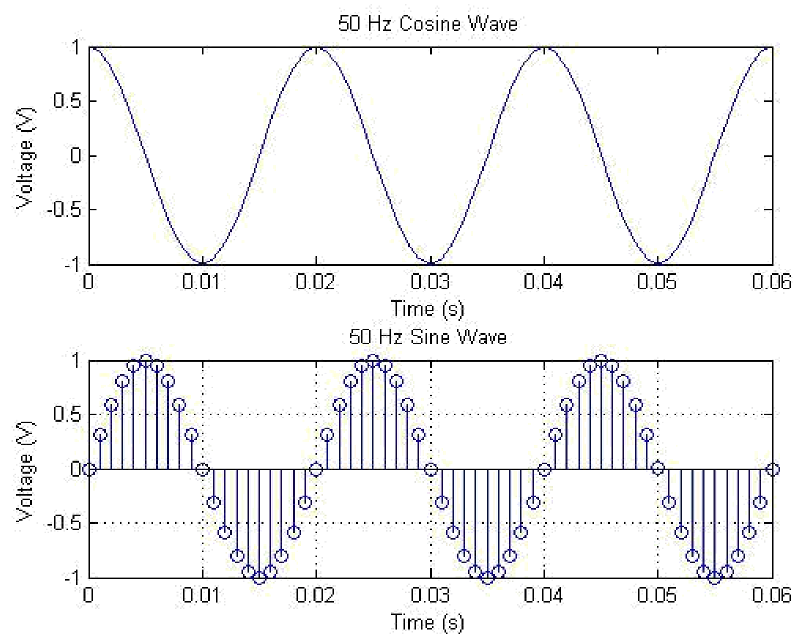
b. $G(s) = \frac{1}{s^2+10s+25}$

c. $G(s) = \frac{1}{s^2+25}$

PROGRAMS:

```
%create a cosine/sine wave with f = 50 Hz

clc
clear all

f=50; tp=1/f;          % signal frequency in continuous time
fs=1000; ts=1/fs ;     % time period
                       % sampling frequency
                       % sampling Time

t=0:ts:3*tp;
x=cos(2*pi*f*t)
subplot(2,1,1)
plot(t,x)
title('50 Hz Cosine Wave')
xlabel('Time (s)');
ylabel('Voltage (V)');

t=0:ts:3*tp;
x=sin(2*pi*f*t)
title('50 Hz Sine Wave')
subplot(2,1,2)
stem(t,x)
xlabel('Time (s)');
ylabel('Voltage (V)');
grid on
```

Output



```
%  identical discrete time signals from
%  continuous time signals of frequency 50 and 550 Hz
%  sampled with a sampling frequency of 1kHz
clc
clear all

%sampling theorem (fs>2fc) satisfied

f=40; tp=1/f;          % signal frequency in continuous time
fs=100;                % time period
ts=1/fs ;              % sampling frequency

                       % sampling Time

t=0:ts:10*tp;
x=sin(2*pi*f*t)

subplot(2,1,1)
stem(t,x)
title('fc = 40 Hz and fs = 1 kHz')
xlabel('Time (s)');
ylabel('Voltage (V)');


%sampling theorem not satisfied

f=140; tp=1/f;         % signal frequency in continuous time
fs=100; ts=1/fs ;      % time period
                       % sampling frequency
                       % sampling Time

t=0:ts:30*tp;
x=sin(2*pi*f*t)

subplot(2,1,2)
stem(t,x)
title('fc = 440 Hz and fs = 1 kHz')
xlabel('Time (s)');
ylabel('Voltage (V)');
grid on


%  % x(t) = e^at cos wt clc
clear all f=50; tp=1/f; samples=20;
t=0:tp/samples:5*tp; a=10
x=exp(a*t).*cos(2*pi*f*t)
subplot(2,1,1) plot(t,x)
title('Growing Exponential') xlabel('Time
(s)'); ylabel('Voltage (V)');
```

```
x=exp(-a*t).*cos(2*pi*f*t)
subplot(2,1,2)
stem(t,x)
title('Decaying Exponential')
xlabel('Time (s)');
ylabel('Voltage (V)');


% complex exponential figure;
f=50;
tp=1/f;
samples=20;
t=0:tp/samples:5*tp;
a=10
x=cos(2*pi*f*t)
y=sin(2*pi*f*t)
plot3(t,x,y)
title('Complex Exponential') xlabel('Time
(s)'); ylabel('Cos'); zlabel('Sin');
grid on
% sinc function

figure;
f=50;
tp=1/f;
samples=20;
t=-5*tp:tp/samples:5*tp;
arg=2*pi*f*t
x=sin(arg)./arg
subplot(2,1,1)
plot(t,x)
title('Sinc Function')
xlabel('Time (s)');
ylabel('Cos');
subplot(2,1,2)
plot(t,sinc(arg))
title('Sinc Function Inbuilt')
xlabel('Time (s)');
ylabel('Cos');

%differential equation
clc
clear all
x=dsolve('Dx=5*x')
x=dsolve('Dx=-5*x')
x=dsolve('D2x=-25*x', 'x(0)=5','Dx(0)=0')
x=dsolve('D2x=-10*Dx-25*x', 'x(0)=5','Dx(0)=0')
x=dsolve('D2x=-5*Dx-25*x', 'x(0)=5','Dx(0)=0')

%transfer function, step and impulse response

%first order system
gtf=tf([1],[1 25])
```

```
subplot(2,1,1)
impulse(gtf)
subplot(2,1,2)
step(gtf)
%second order system
gtf=tf([1],[1 10 25])
subplot(2,1,1)
impulse(gtf)
subplot(2,1,2)
step(gtf)
%second order system
gtf=tf([1],[1 0 25])
subplot(2,1,1)
impulse(gtf)
subplot(2,1,2)
step(gtf)
```

Understanding the Dynamics of Second Order Systems

Ex. No: 2 **Date:**

AIM:

The objective of this exercise is

1. To know the general form of transfer function of a second order system
2. Effect of variation in damping ratio on the response of the second order system
3. Pole positioning of the system due to change in the damping ratio
4. Identify a first order and second order system that have identical step response.

PRACTICE:

The general form of the second order system transfer function

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

$where,$

ζ is the damping ration of the system

ω_n is the natural frequency of the system

Consider a second order system with $\omega_n = 5\, rad/s$. Find the transfer function and plot the impulse response, step response and the pole-zero map of the system using matlab functions for the following cases

1. Damping ratio $\zeta = 0$
2. Damping ratio $0 < \zeta < 1$
3. Damping ratio $\zeta = 1$
4. Damping ratio $\zeta > 1$

Comment on the type of system based on damping.

Assignment:

Draw the locus of roots of the system as the damping ratio changes increases. Write a matlab code to do plot the locus.

Understanding the Dynamics of Second Order Systems

PROGRAMS:

```
%effect on damping on the system response and poles %second order system
clear all
clc

% undamped system
sys = tf([25],[1 0 25])
subplot(5,3,1)
impulse(sys)
subplot(5,3,2)
step(sys)
subplot(5,3,3)
pzmap(sys)

%  underdamped system sys =
tf([25],[1 5 25]) subplot(5,3,4)
impulse(sys) subplot(5,3,5)
step(sys) subplot(5,3,6)
pzmap(sys)
%critically damped
sys = tf([25],[1 10 25])
subplot(5,3,7)
impulse(sys)
subplot(5,3,8)
step(sys)
subplot(5,3,9)
pzmap(sys)

%overdamped
sys = tf([25],[1 100 25])
subplot(5,3,10)
impulse(sys)
subplot(5,3,11)
step(sys)
subplot(5,3,12)
pzmap(sys)

%first order system
sys = tf([0.2506],[1 0.2506])
subplot(5,3,13)
impulse(sys)
subplot(5,3,14)
step(sys)
subplot(5,3,15)
pzmap(sys)
```

Time Domain Specifications of a Second Order System

Ex. No: 3 **Date:**

AIM:

The objective of this exercise is to

1. Understand the step response (transient and steady state) of a second order system
2. Know the time domain specifications like peak overshoot, rise time, settling time and steady state error.
3. Analyze the effect of additional poles and zeros on a second order system

PRACTICE:

Commands Used

```
#sys1= tf ([num],[den1 coefficients])
#ltiview(sys)
#pole(sys)
#zero(sys)
#r=roots(sys)
#poly(-a1; r)
#sys2= tf ([num],[den2 coefficients])
#sys3= tf ([num],[den3 coefficients])
#ltiview(sys1,sys2,sys3)
```

System #1

1. $G(s) = \frac{25}{s^2+5s+25}$
2. Add a real pole near the dominant pole of G(s)
3. Add a real pole away the dominant pole of G(s)

Procedure

1. Find the step response of the system.
2. Using LTI viewer find the time domain specification of the system. Tabulate the results.
3. Add a pole (-p1) near to the dominant pole of the system. Using LTI view find the steady state value of the system (K), normalize the gain of G(s) by scaling the gain of the numerator as shown,

   $$G1(s) = \frac{25/K}{(s+p1)(s^2+5s+25)}$$

   Plot the step response and tabulate the time domain specifications of the system.
4. Repeat the same with the pole away from the dominant pole of the system
5. Give the plot with all three system response in LTI viewer and tabulate the results of transient response as follows.

Time Domain Specifications of a Second Order System

Effect of adding a pole to the system

System Specification	Given System	System with a pole near the dominant pole	System with a pole away from the dominant pole
STEP RESPONSE			
Transfer Function			
Poles			
Zeros			
Steady State Value	1		
Normalized Gain (K)	1		
STEP RESPONSE WITH NORMALIZED GAIN			
Peak Response			
Rise Time			
Settling Time			
Steady State Value			
Inference			

Repeat the same procedure for addition of zero in the system

$$G2(s) = \frac{\frac{25}{K}(s + z1)}{(s^2 + 5s + 25)}$$

Effect of adding a zero to the system

System Specification	Given System	System with a zero near the dominant pole	System with a zero away from the dominant pole
STEP RESPONSE			
Transfer Function			
Steady State Value	1		
Normalized Gain (K)	1		
STEP RESPONSE WITH NORMALIZED GAIN			
Peak Response			
Rise Time			
Settling Time			
Steady State Value			
Inference			

Comment on the change in the time domain specifications of the system due to addition of a pole and zero as the Inference

Study of Armature Controlled DC Motor

Ex. No: 4 **Date:**

AIM:

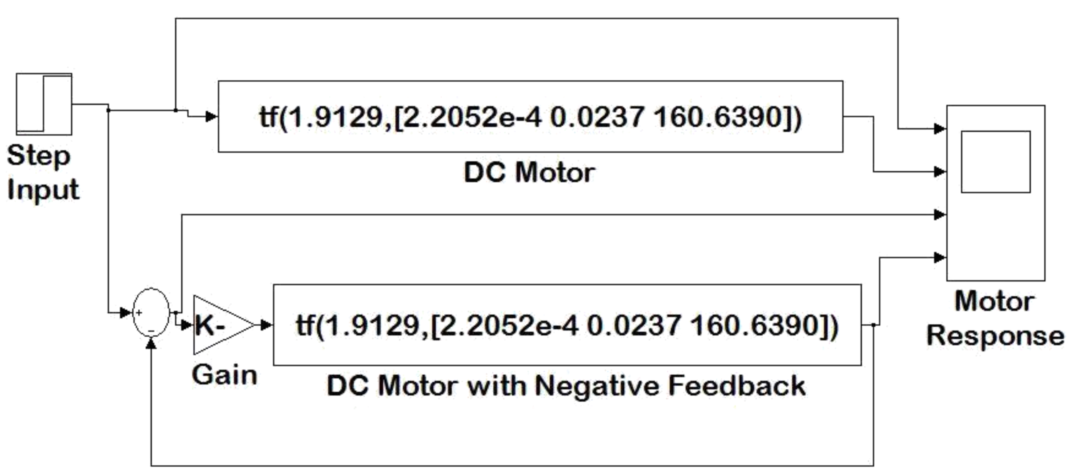
The objective of this exercise is to

1. Model armature controlled dc motor and obtain the transfer function
2. Simulate the same in Matlab Simulink and understand feedback compensation
3. Learn block diagram reduction in Matlab

PRACTICE:

Steady State Error Analysis

1. Simulate the transfer function model of dc motor in Matlab- Simulink
2. Obtain the step response of the system without feedback.
3. Repeat the same with a negative feedback and a gain in the forward path
4. Observe the step response by increasing the gain linearly
5. Observe and tabulate the steady state error in the system



Effect of Gain Proportional to the Error in a Feedback System

Test Case	Expected Output	Actual Output	Steady State Error
Open Loop System			
Negative Feedback (K=1)			
Negative Feedback (K=10)			
Negative Feedback (K=100)			
Negative Feedback (K=500)			
Negative Feedback (K=1000)			

Comment on the effect of increasing gain with negative feedback compensation on the steady state error of the system.

Block Diagram Reduction

Ex. No: 5 **Date:**

AIM:

The objective of this exercise is to

1. Learn block diagram reduction in Matlab

THEORY :

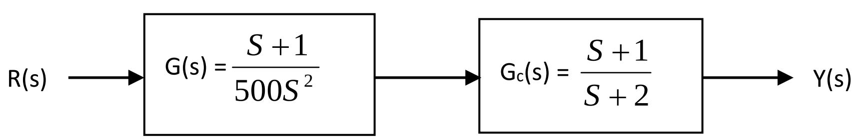
It is a representation of the control system giving the inter-relation between the transfer function of various components. The block diagram is obtained after obtaining the differential equation & Transfer function of all components of a control system. The arrow head pointing towards the block indicates the i/p & pointing away from the block indicates the o/p. After obtaining the block diagram for each & every component, all blocks are combined to obtain a complete representation. It is then reduced to a simple form with the help of block diagram algebra.

<u>Blocks connected in Series</u>

Let the process represented by the transfer function G(s) be

G(s) = $\frac{S+1}{500S^2}$ .and let the controller represented by the transfer function

G $_c$(s) be G $_c$(s) = $\frac{S+1}{S+2}$.Compute the series transfer function G $_c$(s)G(s)




```
»numg=[1 1]; deng = [500 0 0];
»numh=[1 1]; denh = [1 2]
»[num, den]= series(numg,deng,numh,denh);
»printsys (num,den)
```

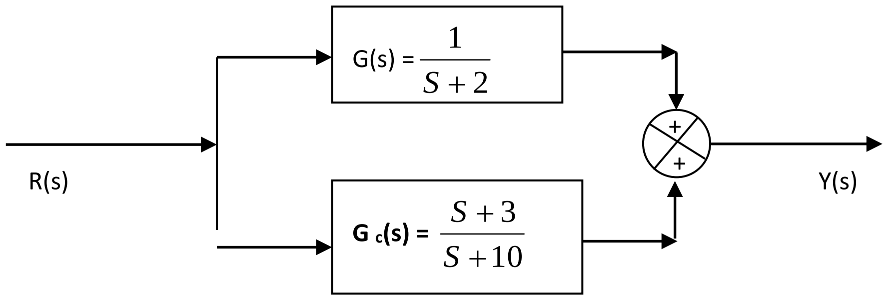
The output of this program is

num/den =

```
    s^2 + 2 s + 1
  ------------------
  500 s^3 + 1000 s^2
```

Blocks connected in parallel

A simple open – loop control system can be obtained by interconnecting a plant and a controller in parallel.



The transfer function in this case is $G_c(s)+G(s)$.Let the process represented by the transfer function G(s) be G(s) = $\frac{1}{S+2}$ .and let the controller represented by the transfer function $G_c(s)$ be $G_c(s)$ = $\frac{S+3}{S+10}$**.** Compute the combined transfer function of the blocks.

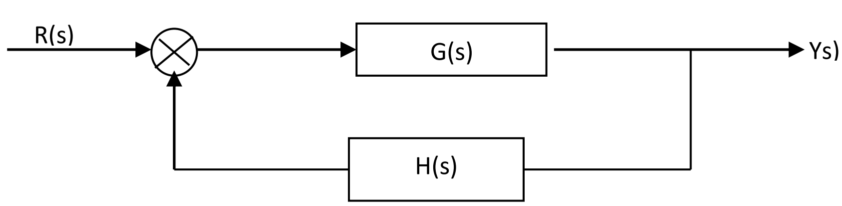
```
»num1 = 1;
»den1 =[1 2];
»num2 =[1 3];
»den2 =[1 10];
» [nump, denp] = parallel(num1, den1, num2, den2);
» printsys(nump, denp);
```

Execution of the above yields the result

num/den =
s^2 + 6s +16
s^2 + 12s + 20

Feedback function

This function reduces the process of computing the closed loop transfer function for single or multiple loops.



T(s) $=\frac{Y(S)}{G(S)}=\frac{num}{den}$; G(s)= $\frac{num1}{den1}$;H(s) = $\frac{num2}{den2}$

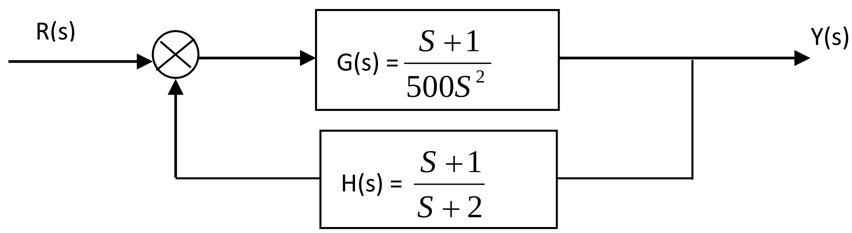
+1 – positive feed back

–1 – negative feedback

command : [num, den] = feedback(num1, den1, num2, den2, sign)

Single loop

Example :1




```
» numg = [1 1] ; deng = [500  0  0];
» numh = [1 1] ;  denh = [ 1  2 ];
» [num, den] = feedback(numg,deng,numh,denh,-1);
» printsys(num, den)
```

The output of the program will be

```
Transfer function
         s^2 + 3 s + 2
  ----------------------------
  500 s^3 + 1001 s^2 + 2 s + 1
```

PROCEDURE:

1. Enter the command window of the MATLAB.
2. Create a new M – file by selecting File – New – M – File.
3. Type and save the program.
4. Execute the program by either pressing F5 or Debug – Run.
5. View the results.
6. Analysis the stability of the system.

For reducing very complex systems, blkbuild and connect commands are used as illustrated below; BLKBUILD Builds a block-diagonal state-space structure from a block diagram of transfer functions and state models. This is a SCRIPT FILE.
Each block is numbered as shown. In the first seven lines of the program, the above blocks are completely defined. The numbers of blocks are then defined n blocks. The command blkbuild uses the variable nblocks to begin building the system. It converts all the transfer functions to state space models and assembles them into one large block state space model called a, b, c, d.
The next step is to create the matrix q that describes the interconnections of various blocks. Each row corresponds to a block and the first element in the row is the block number and the remaining elements indicate the blocks whose outputs are connected to the input of the current block.
After defining q, the block 1 that receives the system input and block 4 that produces the system output are defined by the variables ‘input’ and ‘output’. The connect command makes the

connections and reduces the system into a single state space model which is then converted back into transfer function.

INPUTS:

n is the number of blocks in the diagram.

n_i and d_i are the numerator and denominator polynomials for the i th block if it is a transfer function.

a_i,b_i,c_i,d_i are the state matrices for the I th block if it is a state model.

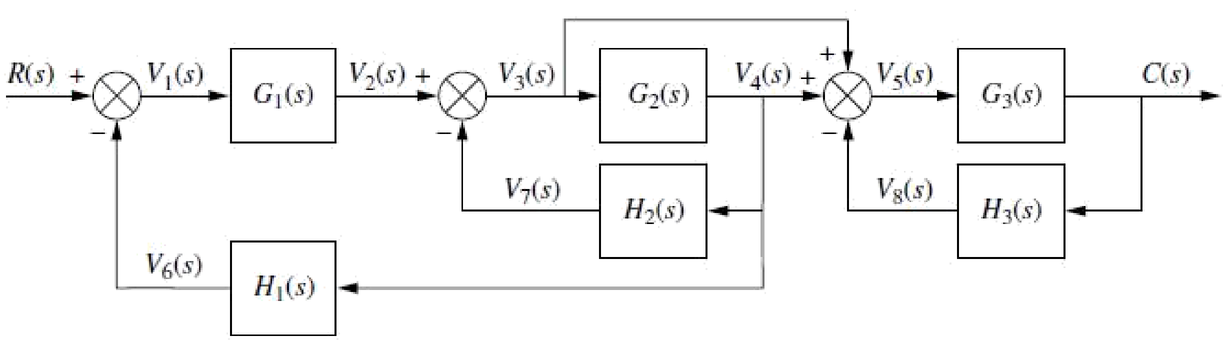
OUTPUTS:

a,b,c,d is the resulting state space structure. The matrices are built up by progressive APPENDs of the state

Block Diagram Reduction - Matlab Commands Used

#tf(G1, G2)
#series(G1, G2)
#parallel(G1, G2)
#feedback(G1,G2)
#append(G1, G2)
#connect(sys, Q, input, output)
#tf2ss (num,den)

Write a code to reduce the block diagram and obtain the transfer function.



Let $G_1(s) = 1$ $G_2(s) = 1/(s+1)$,

G3(s) = 1/(s+2)G4(s) = 1/(s+3),

H1(s) = 4 H2(s) = 8, H3(s) = 12

%TF OF COMPLETE BLOCK DIAGRAM 1

```
»n1=1;d1=1;
»n2=1;d2=[1 1];
»n3=1;d3=[1 2];
»n4=1;d4=[1 3];
»n5=4;d5=1;
```

```
»n6=8;d6=1;
»n7=12;d7=1;
»nblocks=7;
»blkbuild
»q = [  1  0  0  0  0
        2  1 -5  0  0
    3  2 -6  0  0
    4  2 -6  3  -7
    5  3  0  0  0
    6  3  0  0  0
    7  4  0  0  0 ]
»iu = 1;
»iy = 4;
» [A,B,C,D]=connect(a,b,c,d,q,iu,iy);
» sys=ss(A,B,C,D);
»sys=tf (sys)
```

System Response:

State model [a,b,c,d] of the block diagrams has 7 inputs and 7 outputs

Comment on the effect of block diagram reduction and find the final transfer function and verify with the given below.

num/den =

$$\frac{1s + 3}{s^3 + 26 s^2 + 179s + 210}$$

Static Error Constants and Steady State Error

Ex. No: 6 **Date:**

AIM:

The objective of this exercise is to

1. Introduce type of a system
2. Understand steady state error in systems of different types
3. Define static error constants and estimate steady state error from error constants

PRACTICE:

Identify the type of systems given below with

1. Poles at (-8, -9, -12), zeros at (-2, -3) and gain of 100
2. Poles at (0, -8, -9, -12), zeros at (-2, -3) and gain of 100
3. Poles at (0,-8, -9, -12), zeros at (-2, -3) and gain of 100

Find the static error constants, the steady state error and plot the response of the systems for step, ramp and parabolic inputs

Sample Code

```
%steady state error
clc
clear all

%  type 0 system
%step input
Gp0=zpk([-2 -5], [ -8 -10 -12], 500) %G(s) - step input Kp0=dcgain(Gp0)
ssr_step0=1/(1+Kp0) %ramp
input
Gv0=zpk([0 -2 -5], [ -8 -10 -12], 500) %sG(s) - ramp input Kv0=dcgain(Gv0)
ssr_ramp0=1/(Kv0)
%parabolic input
Ga0=zpk([0 0 -2 -5], [ -8 -10 -12], 500) %s^2G(s) - parabolic input Ka0=dcgain(Ga0)
ssr_parabolic0=1/(Ka0)


%plot the response
G0=zpk([-2        -5], [ -8 -10 -12], 500)
G1=zpk([-2        -5], [0 -8 -10 -12], 500)
G2=zpk([-2        -5], [0 0 -8 -10 -12], 500)

%figure
sys0=feedback(tf(G0),1);
subplot(3,3,1);
t = 0.0001:0.01:5;   u = t./t; u(1:100)=0;                          lsim(sys0,u,t) ;
```

```
subplot(3,3,2);
t = 0.0001:0.01:5; u = t; u(1:100)=0; lsim(sys0,u,t); subplot(3,3,3);
t = 0.0001:0.01:5;                u = t.*t/2; u(1:100)=0;                lsim(sys0,u,t) ;
```

Output

Give the result as a table and the response plot

SIMULATION OF FIRST & SECOND ORDER SYSTEMS

Expt. No.7 **Date:**

AIM:

To digitally simulate the time response characteristics of a linear system without non- lineraties and to verify it manually.

THEORY:

The time response characteristics of control systems are specified in terms of time domain specifications. Systems with energy storage elements cannot respond instantaneously and will exhibit transient responses, whenever they are subjected to inputs or disturbances.

The desired performance characteristics of a system of any order may be specified in terms of transient response to a unit step input signal. The transient response characteristics of a control system to a unit step input is specified in terms of the following time domain specifications

- Delay time t_d
- Rise time tr
- Peak time t_p
- Maximum peak overshoot M_p
- Settling time t_s

Some practical examples of first order systems are RL and RC circuits.

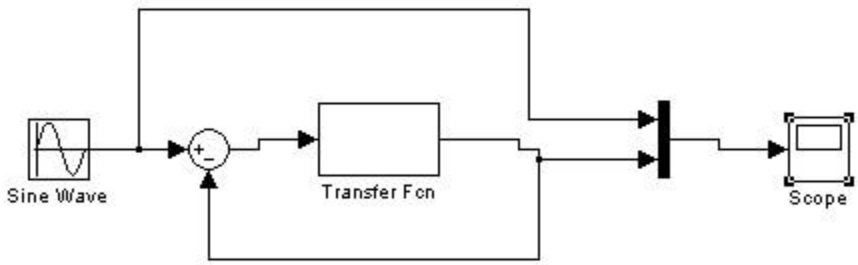
PROCEDURE:

1. Derive the transfer function of a RL series circuit.
2. Assume R= 1 Ohms L = 0. 1 H. Find the step response theoretically and plot it on a graph sheet.
3. To build a SIMULINK model to obtain step response / sine response of a first order system, the following procedure is followed:
    1. In MATLAB software open a new model in SIMULINK library browser.
    2. From the continuous block in the library drag the transfer function block.
    3. From the source block in the library drag the step input/ sine input.
    4. From the sink block in the library drag the scope.
    5. From the math operations block in the library drag the summing point.
    6. Connect all to form a system and give unity feedback to the system.
    7. For changing the parameters of the blocks connected double click the respective block.

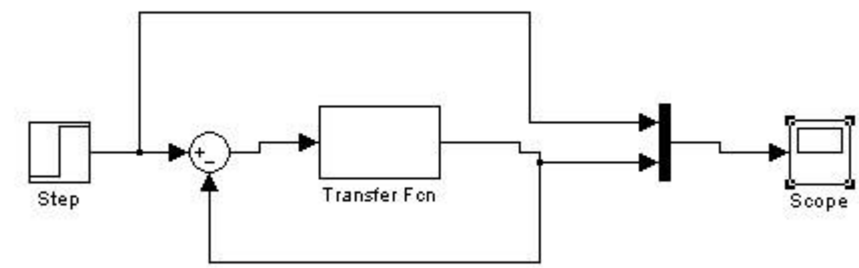
8. Start simulation and observe the results in scope. (Use a mux from the signal routing block to view more than one graph in the scope)
9. Compare the simulated and theoretical results.

BLOCK DIAGRAM:

Sine response of a first order system:



Step response of a first order system:



MATLAB (m-file) program to obtain the step response and impuse response

```
% MATLAB program to find the step response

num=[          ];
den=[          ];
sys = tf (num,den);
step (sys);
grid


% MATLAB program to find the impulse response

num=[              ];
```

```
den=[          ];
sys = tf (num,den);
impulse (sys);
grid
```

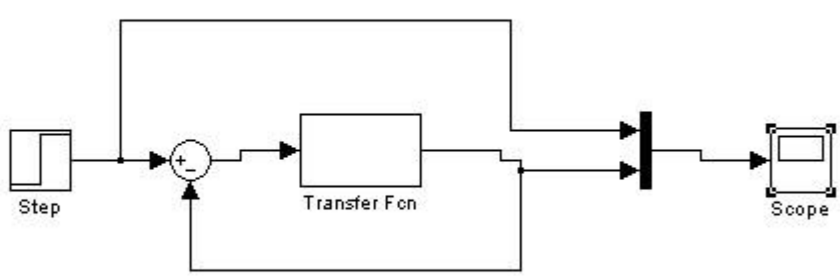
SIMULATION OF SECOND ORDER SYSTEMS

PROCEDURE:

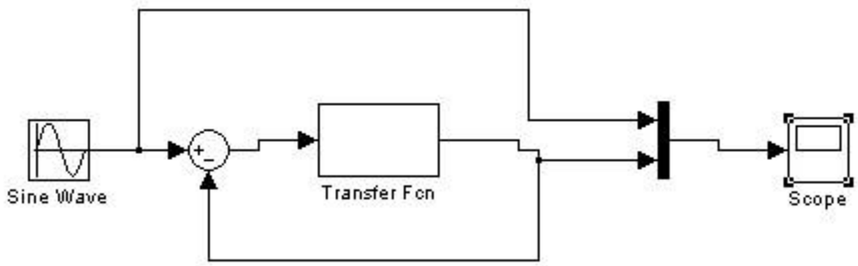
1. Derive the transfer function of a RLC series circuit.
2. Assume R= 1 Ohms,   L = 0. 1 H and C = 1 micro Farad. Find the step response theoretically and plot it on a graph sheet.
3. To build a SIMULINK model to obtain step response / sine response of a second order system, the following procedure is followed:
    1. In MATLAB software open a new model in SIMULINK library browser.
    2. From the continuous block in the library drag the transfer function block.
    3. From the source block in the library drag the step input/ sine input.
    4. From the sink block in the library drag the scope.
    5. From the math operations block in the library drag the summing point.
    6. Connect all to form a system and give unity feedback to the system.
    7. For changing the parameters of the blocks connected double click the respective block.
    8. Start simulation and observe the results in scope. (Use a mux from the signal routing block to view more than one graph in the scope)
    9. From the step response obtained note down the rise time, peak time, peak overshoot and settling time.
    10. Compare the simulated and theoretical results.

BLOCK DIAGRAM:

Step response of a second order system:



Sine response of a second order system:



Comments:

The time response characteristics of the given first and second order system is simulated digitally and verified manually. Mention those comments.

STABILITY ANALYSIS OF LINEAR SYSTEMS

Ex. No: 8 **Date:**

AIM:

To find the stability of the system in the form of transfer function using MATLAB

$$G(s) = \frac{25}{s^2 + 10s + 25}$$

Given poles are -3.2+j7.8,-3.2-j7.8,-4.1+j5.9,-4.1-j5.9,-8 and the zeroes are -0.8+j0.43,-0.8-

$$G(s) = \frac{25}{s^2 + 10s + 25}$$

Sample Code

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

THEORITICAL CALCULATIONS:

Enter zeros Z =

Enter poles P =

Enter gain K=

Result :

Stability analysis of linear systems using Root locus method

Ex. No: 9 **Date:**

AIM:

To obtain the Root locus plot and to verify the stability of the system

PRACTICE:

$$G(s) = \frac{25}{s^2 + 10s + 25}$$

d to verify the stability of the system with transfer function,

Sample Code

```
%root locus with fixed 'w' and variable damping ratio
%ROOT LOCUS OF THE SYSTEM%
num=[                    ]
den=[                    ]
sys=tf(num,den)
rlocus(sys)
v=[-10,10,-8,8];
axis(v)
xlabel('Real Axis')
ylabel('Imaginary Axis')
title('Root Locus of the system')
title('Root Locus Plot of the system
```

MANUAL CALCULATIONS:

OUTPUT (from manual calculation)

OUTPUT (from program):

RESULT:

The Root locus plot is drawn for the given transfer function, G(s)= ____________________ using MATLAB and the range of gain K for stability is______________.

Frequency Response – Bode Plot

Ex. No: 10 **Date:**

AIM:

The objective of this exercise is to learn frequency response with bode plot

$$G(s) = \frac{25}{s^2 + 10s + 25}$$

Sample Code

```
%root locus with fixed 'w' and variable damping ratio
w=5;
num=[w*w]
for e=0:0.3:1.2
        den=[1 2*e*w w*w]
        sys = tf(num,den)
%           p=pole(sys)
%           plot(real(p),imag(p),'*')
        bode(sys)
        hold on
        grid on
end


%bode plot
num = [25];
den = [1 0 25];
sys = tf(num,den)
bode(sys)
hold on
num = [10*25];
den = [1 5 25];
sys = tf(num,den)
bode(sys)
hold on
num = [100*25];
den = [1 10 25];
sys = tf(num,den)
bode(sys)
polar(sys)

num = [1 10];
den = [1];
sys = tf(num,den)
bode(sys)

num = [1];
den = [1 10];
```

```
sys = tf(num,den)
bode(sys)

num = [1];
den = [1 0];
sys = tf(num,den)
bode(sys)

num = [1 10 25];
den = [1];
sys = tf(num,den)
bode(sys)

num = [1];
den = [1 10 25];
sys = tf(num,den)
bode(sys)

sys=zpk([], [-5,-20,-50],10000)
bode(sys)
[Gm, Pm]=margin(sys)
hold on
sys=zpk([], [-5,-20,-50],100)
bode(sys)
[Gm, Pm]=margin(sys)
hold on
sys=zpk([], [-5,-20,-50],10)
bode(sys)
[Gm, Pm]=margin(sys)
hold on
```

State Space Stability and Controllability

Ex. No: 11 **Date:**

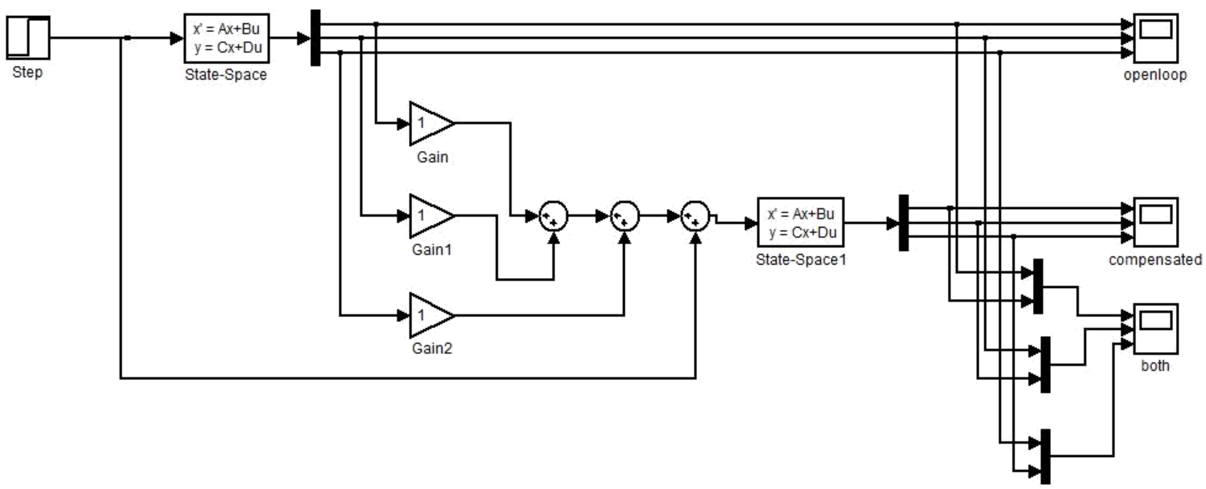
AIM:

The aim of this lab exercise is to design state feedback control for a system.

State Space Model

A=[-5 1 0; 0 -2 1; 0 0 -1];
B=[0; 0; 1];
C=[-1 1 0];
D=0;

Simulation



Sample Code

```
%state space design
clf
A=[-5 1 0; 0 -2 1; 0 0 -1];
B=[0; 0; 1]
C=[-1 1 0];
D=0;
Told=ss(A,B,C,D);
pos=20.8/100
Ts=4
zeta=(-log(pos/100))/(sqrt(pi^2+log(pos/100)^2))
wn=4/(zeta*Ts)
[num,den]=ord2(wn,zeta)
r=roots(den);
poles=[r(1) r(2) -4]
K=acker(A,B,poles)
Anew=A-B*K
Tss=ss(Anew,B,C,D)
T=tf(Tss)
T=minreal(T)
new_poles=pole(T)
```

```
subplot(2,2,1)
pzmap(Told)
subplot(2,2,2)
step(Told)
subplot(2,2,3)
pzmap(Tss)
subplot(2,2,4)
step(Tss)
```

Ex. No: 12 **Date:**

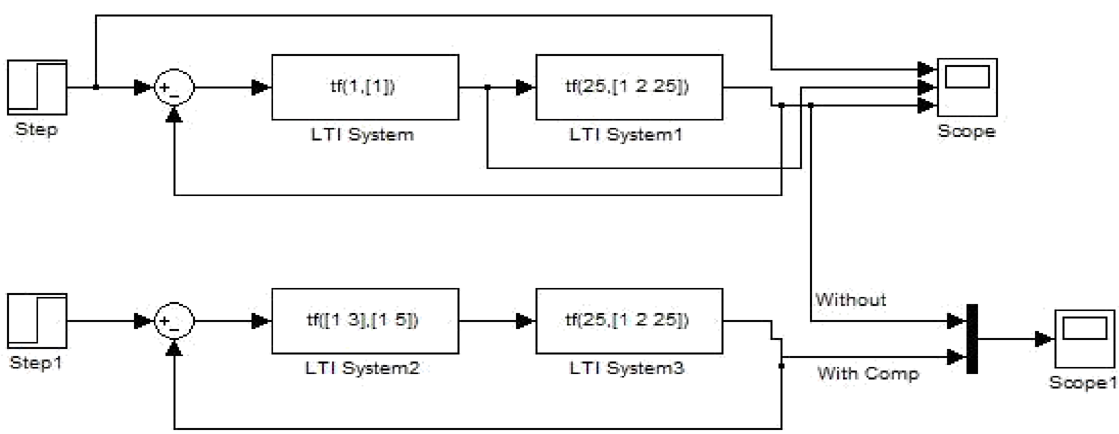
AIM:

The aim of the lab exercise is to design a controller and compensator for a second order system.

System

$$G(s) = \frac{25}{s^2 + 10s + 25} \qquad G(s) = \frac{25}{s^2 + 2s + 25}$$

Simulation



Sample Code

```
%pi controller
% unity feedback second order system with K=1

G=tf([13],[1 6 13])
T=feedback(G,1)
pole(T)
subplot(2,2,1)
step(T)
subplot(2,2,2)
rlocus(T)
%  unity feedback second order system with K=100 G=tf([10*13],[1 6
13])
T=feedback(G,1)
pole(T)
subplot(2,2,3)
step(T)
subplot(2,2,4)
rlocus(T)
```

Sample VIVA questions

1. Define transfer function ?
2. What is the importance of transfer function ?
3. What assumption is made concerning initial conditions when dealing with transfer functions ?
4. Why do transfer functions for mechanical networks look identical to transfer functions for electrical networks ?
5. To what classifications of systems can be transfer function be best applied?
6. Do the zeros of a system change with a change in gain ?
7. where are the zeros of the closed loop transfer function ?
8. Illustrate the meaning of each of the following : Direct transfer function, Loop transfer function and closed loop transfer function
9. State an advantage of the transfer function approach over the state space approach.
10. Since the motor's transfer function relates armature displacement to armature voltage, how can the transfer function that relates load displacement and armature voltage be determined.
11. What do you mean by control system ?
12. Define transfer function?
13. What is DC servo motor? What are the main parts?
14. What is servo mechanism?
15. Is this a closed loop or open loop system .Explain?
16. What is back EMF?
17. What are the main parts of an DC servomotor?
18. What are the advantages and disadvantages of an DC servo motor?
19. Give the applications of DC servomotor?
20. What do you mean by servo mechanism?
21. What are the characteristics of an DC servomotor?