Group Members Name	Urwa Maryam, Umar Hayyat
Reg. #	2019-EE-352, 2019-EE-360
Marks	4. 6-10-21

Experiment #3

Linear Time Invariant (LTI) System and Representation

Introduction:

Linear, time-invariant (LTI) systems are the primary signal-processing tool for modeling the action of a physical phenomenon on a signal, such as propagation and measurement. LTI systems also are a very important tool for processing signals. For example, filters are almost always LTI systems. In this lesson you will learn the definition of a system and the important system properties of linearity, time-invariance, and causality.

Objectives:

To study the LTI system and its representation using MATLAB software.

Examples of Creating LTI Models
Building LTI models with Control System Toolbox is straightforward. The following sections show simple examples. Note that all LTI models, i.e. TF, ZPK and SS are also MATLAB objects.

Example of Creating Transfer Function Models
You can create transfer function (TF) models by specifying numerator and denominator coefficients. For example,

```
>>num = [1 0];
>>den = [1 2 1];
>>sys = tf(num,den)
```

```
Transfer function:
     s
-------------
s^2 + 2 s + 1
```

A useful trick is to create the Laplace variable, s. That way, you can specify polynomials using s as the polynomial variable.

```
>>s=tf('s');
>>sys= s/(s^2 + 2*s + 1)
```

```
Transfer function:
     s
-------------
s^2 + 2 s + 1
```

This is identical to the previous transfer function.

Example of Creating Zero-Pole-Gain Models

To create zero-pole-gain (ZPK) models, you must specify each of the three components in vector format. For example,

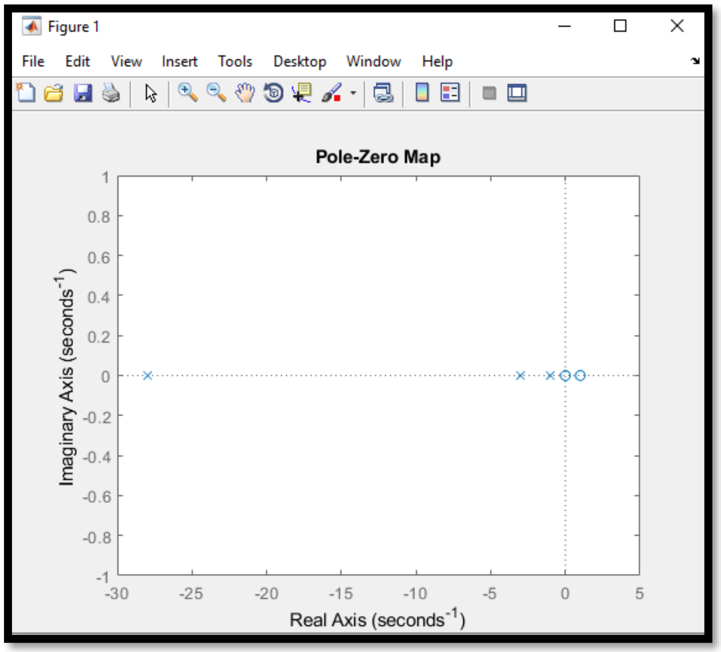
```
>>sys = zpk([0],[-1 -1],[1])

Zero/pole/gain:

 s
-------
(s+1)^2
```

Plotting poles and zeros of a system:
pzmap
Compute pole-zero map of LTI models

```
pzmap(sys)
pzmap(sys1,sys2,...,sysN)
[p,z] = pzmap(sys)
```

```
>> sys=zpk([1 0], [-1 -3 -28], [.776])

sys =

    0.776 s (s-1)
  ------------------
  (s+1) (s+3) (s+28)

Continuous-time zero/pole/gain model.

>> pzmap(sys)
```

When pzmap command is given output will be like that



```
>>H = tf([2 5 1],[1 2 3]);
>>pzmap(H)
```

Simulation of Linear systems to different inputs:

You can simulate the LTI systems to inputs like impulse, step and other standard inputs and see the plot of the response in the figure window. MATLAB command ‘impulse’ calculates the unit impulse response of the system, ‘step’ calculates the unit step response of the system and ‘lsim’ simulates the (time) response of continuous or discrete linear systems to arbitrary inputs.

To obtain an impulse response

```
>> H = tf([2 5 1],[1 2 3]);
>>impulse(H)
```

To obtain a step response type

```
>>step(H)
```

To contain the response of the system you can also specify the time interval to simulate the system to. For example,

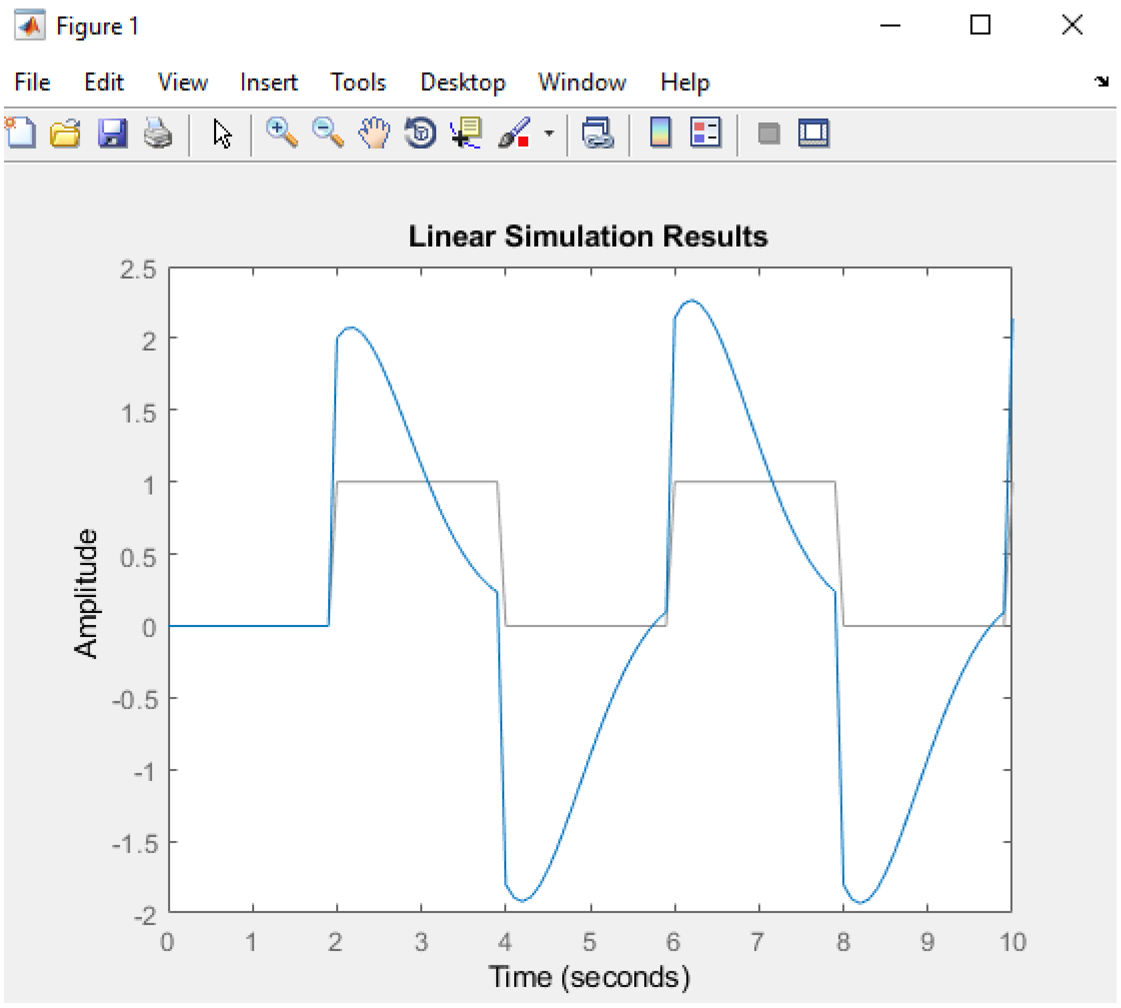
```
>> t = 0:0.01:10;
>> impulse(H,t)
```

Or

```
>> step(H,t)
```

T = 0:dt:Tfinal

```
>> H=tf([2 5 1], [1 2 3]);
>> t= 0:.1:10;
>> [u,t]=gensig ('square',4,10,.1);
>> lsim(H,u,t)
```



Exercise 1:

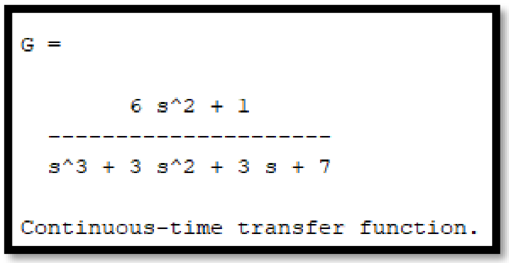
Consider the transfer function

$$G(s) = \frac{6s^2 + 1}{s^3 + 3s^2 + 3s + 7}$$

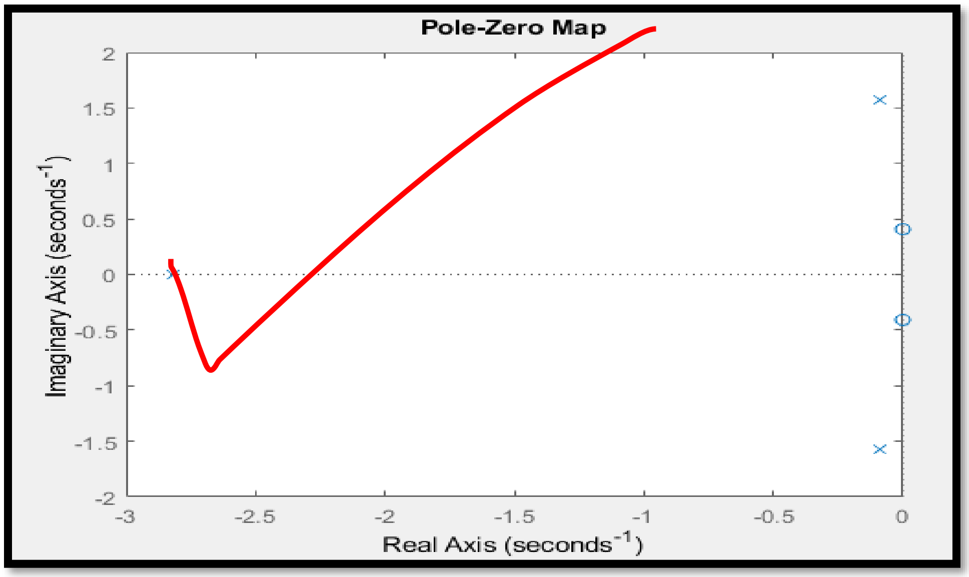
Using MATLAB plot the pole zero map of the above system

Code & Output:

```
%G(s) = Y(s)/X(s) = output / input
%G(s) = 6s^2 + 1 / s^3 + 3s^2 + 3s + 7
output = [6 0 1];
input = [1 3 3 7];
G = tf(output, input)
pzmap(G)
```



```
G =

        6 s^2 + 1
  ---------------------
  s^3 + 3 s^2 + 3 s + 7

Continuous-time transfer function.
```



Exercise 2:

a. Obtain the unit impulse response for the following system

$$\frac{B(s)}{A(s)} = \frac{1}{s^2 + 0.2s + 1}$$

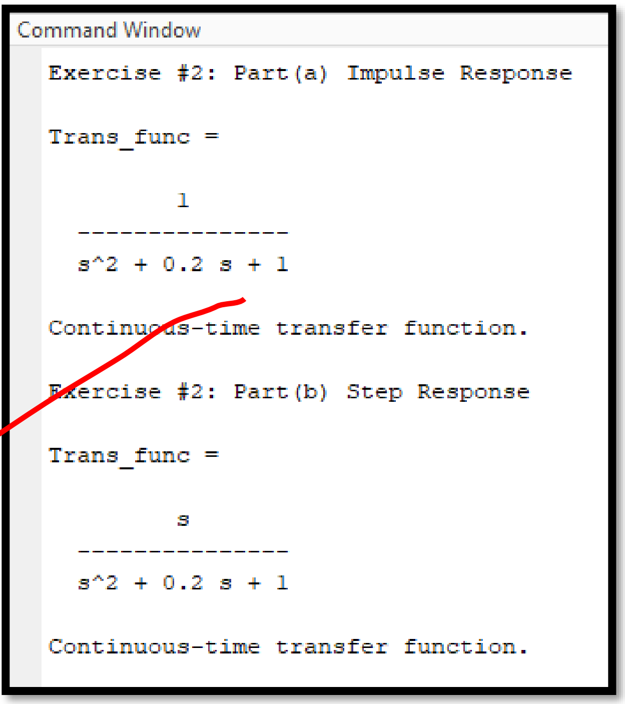
b. Obtain the unit step response for the following system

$$\frac{B(s)}{A(s)} = \frac{s}{s^2 + 0.2s + 1}$$

c. Explain why the results in a. and b. are same?

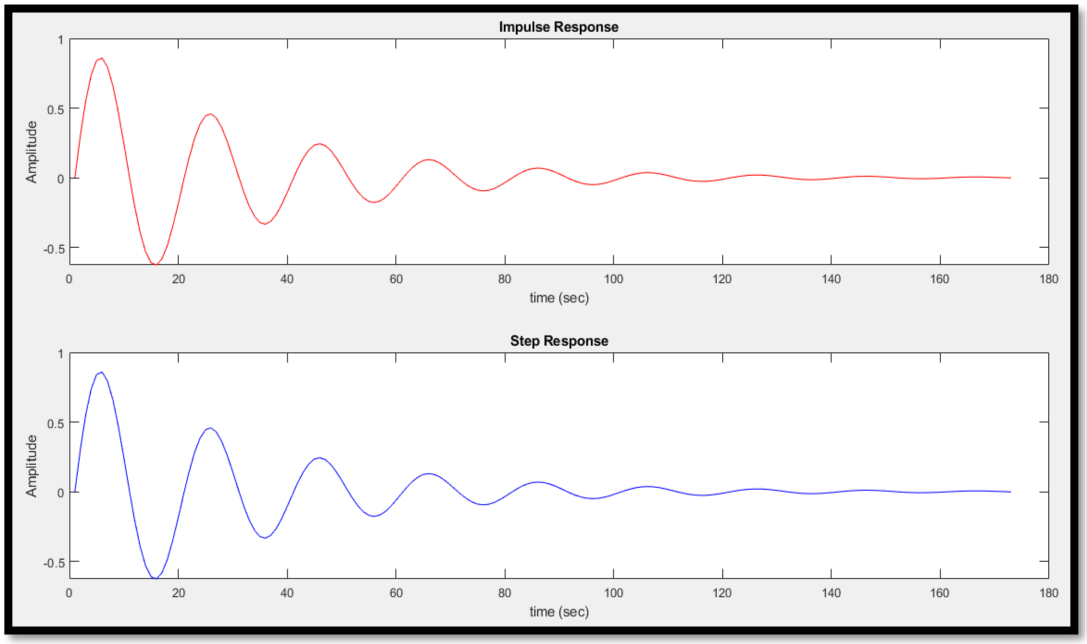
Code & Output:

```
B = [1];
A = [1 0.2 1];
disp('Exercise #2: Part(a) Impulse
Response')
Trans_func = tf(B,A)
imp = impulse(Trans_func);
subplot(2,1,1)
plot(imp,'r')
title('Impulse Response')
ylabel ('Amplitude')
xlabel('time (sec)')
disp('Exercise #2: Part(b) Step
Response')
B = [1 0];
Trans_func = tf(B,A)
subplot(2,1,2)
step = step(Trans_func);
plot(step, 'b')
title('Step Response')
ylabel ('Amplitude')
xlabel('time (sec)')
```



Command Window

```
Exercise #2: Part(a) Impulse Response

Trans_func =

         1
  ---------------
  s^2 + 0.2 s + 1

Continuous-time transfer function.

Exercise #2: Part(b) Step Response

Trans_func =

         s
  ---------------
  s^2 + 0.2 s + 1

Continuous-time transfer function.
```



(C) Why both Graphs are same?

Reason:

Output= (Transfer Function) (Input)

Laplace transform of impulse function is **1**.

Output= (1/(s^2+0.2*s+1)) * 1 = 1/(s^2+0.2*s+1)

Laplace transform of unit step is **1/s.**

Output=((s/s^2+0.2*s+1) * (1/s)) =1/s^2+0.2*s+1

Output in both cases is same so same plots are obtained.

Exercise 3:

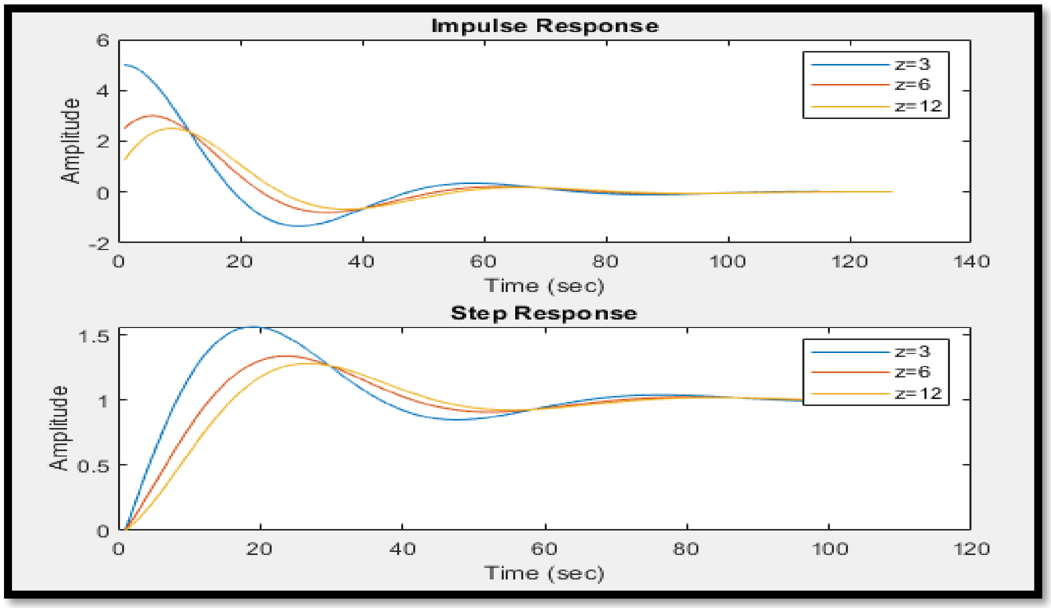
A system has a transfer function

$$\frac{X(s)}{R(s)} = \frac{(15/z)(s+z)}{s^2+3s+15}$$

Plot the response of the system when R(s) is a unit impulse and unit step for the parameter z=3, 6 and 12.

Code & Output:

```
z = 3;
while z<=12
    output=[15/z 15];
    input=[1 3 15];
    trans = tf(output, input);
    I = impulse(trans);
    Y = step(trans);
    z = z*2;
    subplot(2,1,1);
    plot(I)
    title('Impulse Response');
    ylabel('Amplitude');
    xlabel('Time (sec)');
    hold on;
    legend('z=3','z=6','z=12');
    subplot(2,1,2);
    plot(Y)
    title('Step Response');
    ylabel('Amplitude');
    xlabel('Time (sec)');
    hold on;
    legend('z=3','z=6','z=12');
end
```



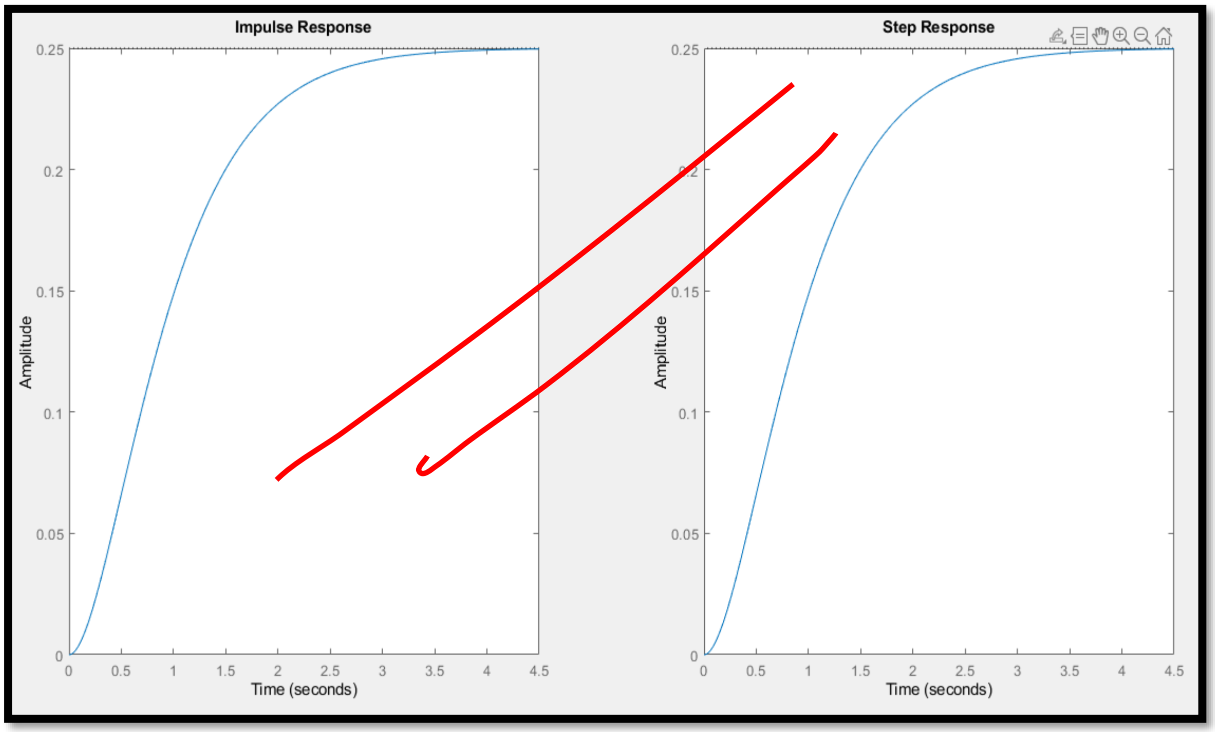
Exercise 4:

Consider the differential equation $\ddot{y} + 4\dot{y} + 4y = u$ where $y(0) = \dot{y}(0) = 0$ and $u(t)$ is a unit step. Determine the solution analytically and verify by co-plotting the analytical solution and the step response obtained with 'step' function.

Code & Output:

Calculations???

```
%y''+ 4y'+ 4y = u(t)
%s2Y(s)-sy(0)-y'(0) + 4sY(s)-y(0) + 4Y(s) = 1/s
%Given y'(0) = y(0) = 0
%Y(s)[s2+4s+4] =1/s
%Y(s)= (1/s) / (s2+4s+4)
%Y(s)=1/(s3+4s2+4s)
%Actual analytical solution we obtained
Y=tf([1],[1 4 4 0]);
subplot(1,2,1);
impulse(Y)    %here we calculated impulse response which is actually output
response
Y=tf([1],[1 4 4]);
subplot(1,2,2);
step(Y) %here we calculated its step response
```



Conclusion?