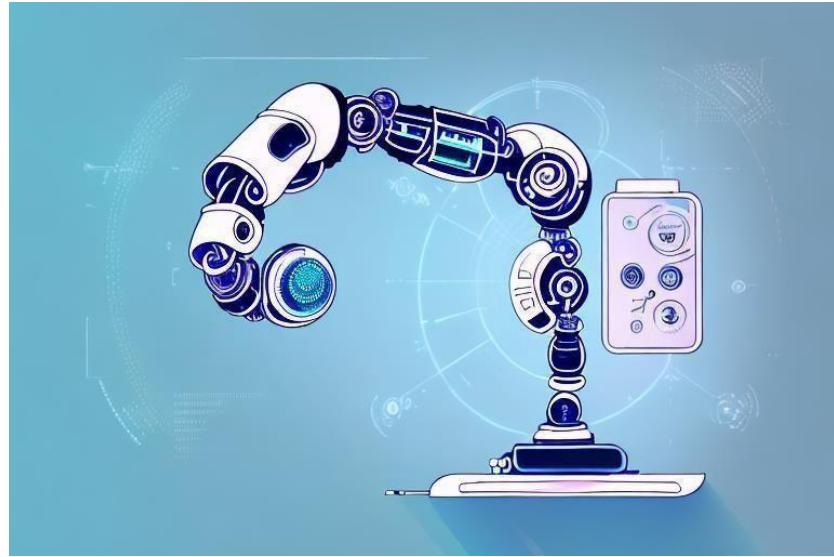




**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)**

DIGITAL CONTROL



Digital Control Systems

PRACTICAL
20ES-BATCH

Submitted To: Engr. Mehran M. Memon

Name: KARAN

Roll No.: 20ES062.

DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)DIGITAL CONTROL

CERTIFICATE

This is to certify that Mr/Ms. KARAN, bearing Roll Number 20ES062, studying in Fourth year, 7th semester in the year 2024 in Department of Electronic Engineering, MUET, Jamshoro, has completed practical course based on the course contents of subject Digital Control Systems-(ES-413) and given a satisfactory account of it in this labbook containing a record the laboratory work.

(Subject Teacher)
Engr. Mehran M. Memon

Dated: _____

INDEX

Sr. #	LIST OF EXPERIMENTS	CLO	Taxonomy Level	Obt. Marks
1	Introduction To MATLAB/Simulink Software	4,5	P3, A4	

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL**

2	Plotting Of Discrete Time Testing Signals	4,5	P3, A4	
3	To Analyze Z-Transforms and Inverse Z-Transforms	4,5	P3, A4	
4	Discrete Time Root Locus	4,5	P3, A4	
5	Mathematical Modelling of Electrical System and Dc Circuit Analysis Using MATLAB	4,5	P3, A4	
6	The Step Response of a Discrete Time System and Effect of Sampling Time on System Response	4,5	P3, A4	
7	The Sample Response Analysis and The Computer Simulation of a Sampled-Data System	4,5	P3, A4	
8	Discrete-Time State Space Representation of a System and Modelling of Magnetically Suspended Ball System	4,5	P3, A4	
9	The Controllability and Observability of a System in Discrete Time Domain	4,5	P3, A4	
10	Frequency Response and Construction of Bode Plot	4,5	P3, A4	
11	Performance Of Pid and Digital Pid Controller	4,5	P3, A4	
12	Tuning Of Digital Pid Controller	4,5	P3, A4	
13	Operation Of Compensator and Design of Lead Compensator	4,5	P3, A4	
14	Performance And Design of Digital Filter	4,5	P3, A4	
15	Low Pass, High Pass & Band Pass Digital Filters	4,5	P3, A4	
16	Open Ended Lab	4,5	P3, A4	
<i>Total Marks Obtained (Out of 15):</i>				

LAB # 01: INTRODUCTION TO MATLAB/SIMULINK SOFTWAREName: KARANRoll # 20ES062

Signature of Lab Tutor:

Date: _____

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL****OBJECTIVE(S):**

#	Topic	# Of Lectures	CLO	Taxonomy level
1	<i>The purpose of this lab is to introduce you to software tool namely MATLAB/SIMULINK</i>	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	
Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
			TOTAL	



DEPARTMENT OF ELECTRONIC ENGINEERING SYSTEMS (ES-413)

DIGITAL CONTROL

EQUIPMENTS REQUIRED:

- Personal Computer
- MATLAB R2013a or above

THEORY:

Millions of engineers and scientists worldwide use MATLAB® to analyze and design the systems and products transforming our world. The matrix-based MATLAB language is the world's most natural way to express computational mathematics. Built-in graphics make it easy to visualize and gain insights from data. The desktop environment invites experimentation, exploration, and discovery. These MATLAB tools and capabilities are all rigorously tested and designed to work together. MATLAB helps you take your ideas beyond the desktop. You can run your analyses on larger data sets and scale up to clusters and clouds. MATLAB code can be integrated with other languages, enabling you to deploy algorithms and applications within web, enterprise, and production systems. MATLAB Control System Toolbox™

Control System Toolbox of MATLAB provides algorithms and apps for systematically analyzing, designing, and tuning linear control systems. You can specify your system as a transfer function, state-space, zero-pole-gain, or frequency-response model. Apps and functions, such as step response plot and Bode plot, let you analyze and visualize system behavior in the time and frequency domain. You can tune compensator parameters using interactive techniques such as Bode loop shaping and the root locus method. The toolbox automatically tunes both SISO and MIMO compensators, including PID controllers. Compensators can include multiple tunable blocks spanning several feedback loops. You can tune gain-scheduled controllers and specify multiple tuning objectives, such as reference tracking, disturbance rejection, and stability margins. You can validate your design by verifying rise time, overshoot, settling time, gain and phase margins, and other requirements. **Getting Started with Control System Toolbox**

- Dynamic System Models: Represent simple and complex dynamic systems, discretize models, and reduce model order.

With Control System Toolbox™ software, you represent dynamic systems as model objects. Model objects are specialized data containers that encapsulate model data and other attributes in a structured way. Model objects enable you to manipulate linear systems as single entities rather than keeping track of multiple data vectors, matrices, or cell arrays. Once you have a linear model, you can perform time-domain or frequency-domain analysis of it, design a controller for it, and perform other design or analysis tasks. It includes following key features:

- Linear System Representation: Models of linear time-invariant systems
- Model Interconnection: Series, parallel, and feedback connections; block diagram building
- Model Transformation: Model type conversion, continuous-discrete conversion, order reduction.
- Model Reduction: Model order reduction, low-order approximation, pole-zero cancellation



DEPARTMENT OF ELECTRONIC ENGINEERING SYSTEMS (ES-413)

DIGITAL CONTROL

○ Linear Analysis: Time- and frequency-domain responses, stability margins, parameter sensitivity.

Control System Toolbox™ software lets you analyze the dynamics of linear systems. You can visualize system behavior in time domain and frequency domain. You can extract system characteristics such as rise time, overshoot, and settling time. You can also analyze system stability. Use these tools to analyze the behavior of plant models or validate the performance of a tuned control system. Key features include:

- Linear System Analyzer: Use this app to examine the time-domain and frequency- domain behavior of SISO and MIMO systems.
- Time and Frequency Domain Analysis: System responses such as Bode plots and step responses; system characteristics such as response time and overshoot; simulation
- Stability Analysis: Gain and phase margins, pole and zero locations
- Sensitivity Analysis: Robustness of control systems
- Passivity and Sector Bounds: Analyze systems for passivity and arbitrary conic- sector bounds
- Plot Customization: Customize analysis plot appearance and units

○ Control System Design and Tuning: Tune PID controllers and other control architectures automatically or interactively, design Kalman filters.

Control System Toolbox™ control design tools let you design and tune single-loop and multi loop control systems. Use these techniques and tools to:

- i. Automatically tune common control components such as PID controllers, lead-lag networks, LQG Controllers, and Kalman filters.
- ii. Graphically tune SISO compensators using classical tools such as root locus, Bode diagrams, and Nichols charts.
- iii. Automatically tune SISO or MIMO control systems to meet high-level design goals such as reference tracking, disturbance rejection, and stability margins, regardless of control system architecture.

The key features include:

- PID Controller Tuning: Automatic and interactive tuning of PID gains
- Classical Control Design: Design, tuning, and analysis of single-input, single-output (SISO) feedback systems
- State-Space Control Design and Estimation: Linear-Quadratic-Gaussian control, pole placement, Kalman estimators.
- Multiloop, Multiobjective Tuning: Automated tuning of control systems to meet design requirements
- Gain Scheduling: Tuning of gain-scheduled controllers for nonlinear plants

○ Matrix Computations: Controllability and observability, Lyapunov and Riccati equations



DEPARTMENT OF ELECTRONIC ENGINEERING SYSTEMS (ES-413)

DIGITAL CONTROL

Control System Toolbox™ employs various functions using which you can solve controllability and observability matrices, continuous-time and discrete-time Lyapunov equations, continuous-time and discrete-time Riccati equations.

Introduction to Simulink

Simulink is a software package for modeling, simulating, and analyzing dynamic systems. It supports linear and nonlinear systems, modeled in continuous time, sampled time, or a hybrid of the two. One can easily build, analyze and visualize models. Models are represented graphically in Simulink as block diagrams.

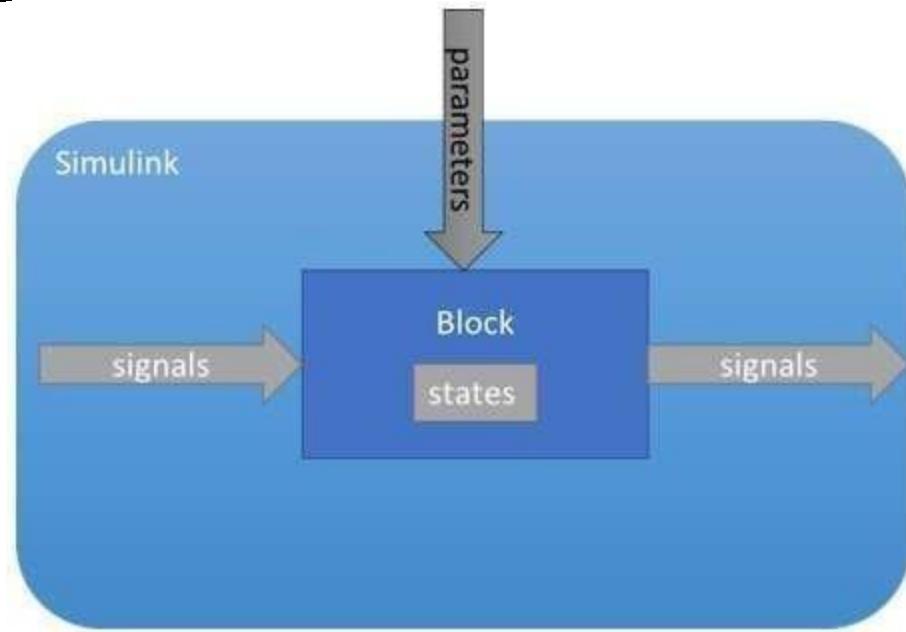
One of the primary advantages of employing Simulink (and simulation in general) for the analysis of dynamic systems is that it allows us to quickly analyze the response of complicated systems that may be prohibitively difficult to analyze analytically. Simulink is able to numerically approximate the solutions to mathematical models that we are unable to, or don't wish to, solve "by hand."

Simulink® is a block diagram environment for multi domain simulation and Model-Based Design. It supports system-level design, simulation, automatic code generation, and continuous test and verification of embedded systems. Simulink provides a graphical editor, customizable block libraries, and solvers for modeling and simulating dynamic systems. It is integrated with

MATLAB®, enabling you to incorporate MATLAB algorithms into models and export simulation results to MATLAB for further analysis.

Simulink handles data in three categories:

- Signals — Block inputs and outputs, computed during simulation
- States — Internal values, representing the dynamics of the block, computed during simulation
- Parameters — Values that affect the behavior of a block, controlled by the user

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL****Figure 1.1. Simulink Manipulation**

At each time step, Simulink computes new values for signals and states. By contrast, you specify parameters when you build the model and can occasionally change them while simulation is running.

1.1 Creating discrete time signal in MATLAB

1.1.1 Example1: The MATLAB function $hn = \text{dimpulse}(P, Q, n)$ returns the discrete impulse response consisting of n samples, applied to the discrete transfer function $H(z) = P(z)/Q(z)$.
Let

$$H(z) = \frac{2.24z^2 + 2.5z + 2.25}{z^2 + 0.5z + 0.68}$$

Create the script file `disc_imp` that returns the discrete response in the form of a table and a plot in the form of sequence of length $n=10$

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL****MATLAB Solution**

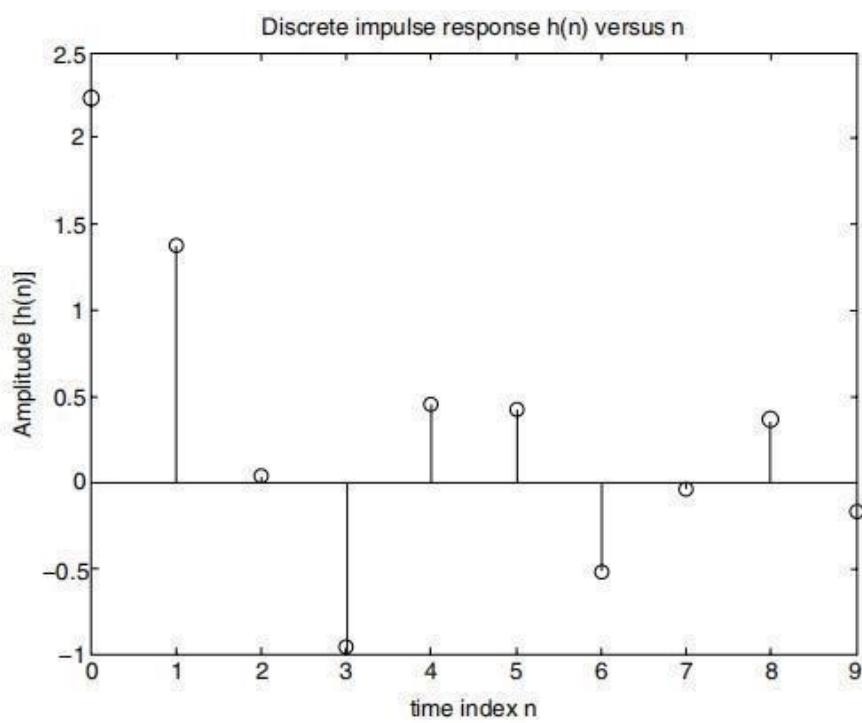
```
% Script file: disc_imp
clc; clf;
n =10;
P = [2.24 2.5 2.25];
Q = [1 .5 .68];
hn = dimpulse(P,Q,n);
nn = 0:1:9;
results = [ nn' hn];
disp('*****');
disp('The impulse response sequence h(n) for the first 10 samples is:');
disp('~~~~~');
disp(' n      h(n)');
disp('~~~~~');
disp(results);
disp('*****');
yzero = zeros(1,10);
stem(nn,hn); hold on; plot(nn,yzero)
title('Discrete impulse response h(n) vs n')
xlabel('time index n');
ylabel('Amplitude [h(n)]');
```

The script file *disc_imp* is executed and the results are shown as follows:

DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)DIGITAL CONTROL

The impulse response sequence $h(n)$ for the first 10 samples is:

n	$h(n)$
0	2.2400
1.0000	1.3800
2.0000	0.0368
3.0000	-0.9568
4.0000	0.4534
5.0000	0.4239
6.0000	-0.5203
7.0000	-0.0281
8.0000	0.3679
9.0000	-0.1648

Figure 1.2- Discrete impulse response $h(n)$

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL**

1.1.2 Example2: The MATLAB function $un = dstep(P,Q,n)$ returns the discrete step response with length of n of the system defined by the transfer function $H(Z) = P(Z)/Q(Z)$. Let

$$H(Z) = \frac{2.24z^2 + 2.5z + 2.25}{z^2 + 0.5z + 0.68}$$

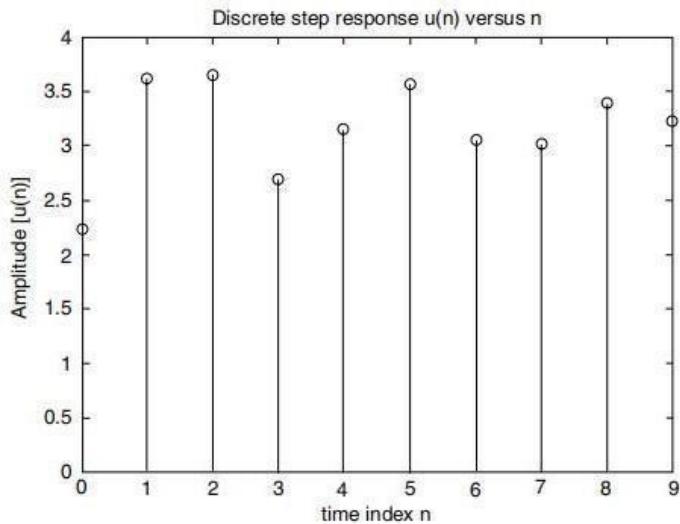
Create the script file disc_step that returns the discrete response in the form of a table and a plot for a sequence of length $n=10$

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL****MATLAB Solution**

```
% Script file: disc_step
clc; clf;
n = 10;
P = [2.24 2.5 2.25];
Q = [1 .5 .68];
un = dstep(P,Q,n);
nn = 0:1:9;
results = [nn' un'];
disp('*****');
disp('The discrete step sequence u(n) for the first 10 samples is:');
disp('~~~~~');
disp(' n      u(n)');
disp('~~~~~');
disp(results);
disp('*****');
yzero = zeros(1,10);
stem (nn,un); hold on; plot (nn,yzero)
title ('Discrete step response u(n) vs n')
xlabel ('time index n');
ylabel ('Amplitude[u(n)]');
```

```
*****
The discrete step sequence u(n) for the first 10 samples is:
~~~~~
n      u(n)
~~~~~
0      2.2400
1.0000  3.6200
2.0000  3.6568
3.0000  2.7000
4.0000  3.1534
5.0000  3.5773
6.0000  3.0570
7.0000  3.0289
8.0000  3.3968
9.0000  3.2320
*****
```

Figure 1.3 Discrete step response

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL**

1.1.1 Example3: Create the script file signal_noise that returns the plot of the following signals.

- Gaussian white noise
- The signal - Signal = $\cos(2\pi n/64)$
- The contaminated signal plus noise using $n=128$ elements. Let the white Gaussian signal power be $P=2.7$ watt.

MATLAB Solution

```
% Script file: signal_noise
n = 1:128; P = 2.7;
figure(1)
subplot(2,1,1)
white_noise = sqrt(P)*randn(1,128);
stem(n,white_noise); hold on; plot(n,white_noise);
xlabel('discrete time n');
ylabel('Amplitude'); title('Gaussian noise');
subplot(2,1,2)
signal = cos(2*pi*n/64);
stem(n,signal); hold on; plot(n,signal);
title('cosine wave with period=64')
xlabel('discrete time n');
ylabel('Amplitude');

figure(2)
signal_noise = white_noise+signal;
stem(n,signal_noise); hold on; plot(n,signal_noise);
title('cosine wave plus white Gaussian noise')
xlabel('discrete time n');
ylabel('Amplitude');
```

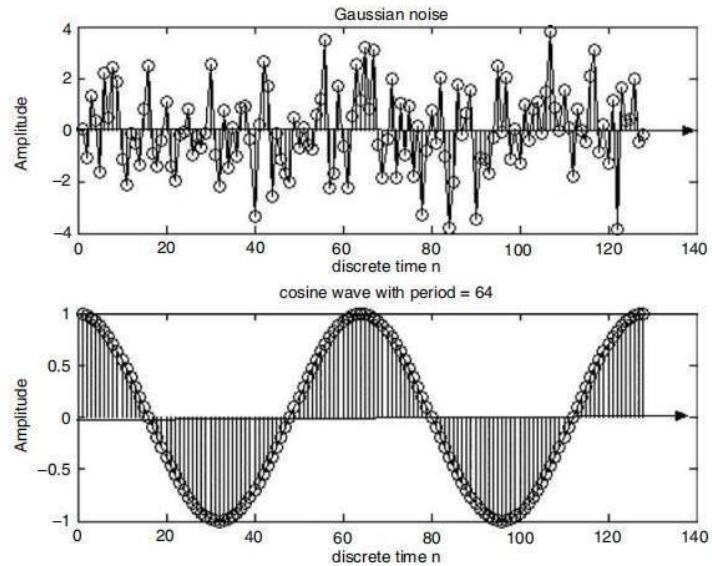
DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)DIGITAL CONTROL

Figure 1.4- Plots of part A and B

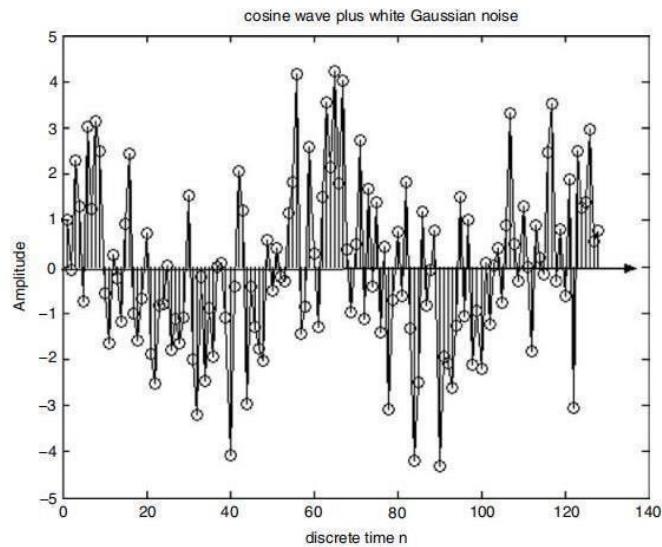


Figure 1.5- Plot of part C

1.1.2 Example4: Let the discrete sequence

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL**

$$f(n) = 4n0.8^n u(n)$$

be contaminated by a random noisy signal with a magnitude less than 1.

Create the script file *averages* that returns the following plots:

1. $f(n)$ versus n
2. noise signal [$n(t)$] versus n
3. $[f(n) + n(t)]$ versus n
4. $[f(n) + n(t)]$ moving average using two terms versus n
5. $[f(n) + n(t)]$ moving average using three terms versus n
6. $[f(n) + n(t)]$ moving average using four terms versus n

Estimate the best moving average approximation for $[f(n) + n(t)]$ versus n , by executing the script file at least three times and observing and recording the results

Where the L point moving average is defined by

$$\{\text{moving aver. } L\} = \frac{1}{L} \sum_{k=0}^{L-1} f(n - k)$$

for $L = 2, 3$, and 4 , over the range $0 \leq n \leq 20$.

```
MATLAB Solution
% Script file: averages
n = 0:1:20;
figure(1)
subplot(3,1,1)
signal = 4.*n.*(.8.^n); stem(n,signal); hold on;
plot(n,signal,n,signal,'o');
title('Signal vs. n')
ylabel('Amplitude'); xlabel('time index n');
subplot(3,1,2)
noise = 2.*rand(1,21)-1.0; y = zeros(1,21);
plot(n,y); title('Noise vs. n'); hold on;
stem(n,noise); hold on; plot(n,y,n,noise);
ylabel('Amplitude'); xlabel('time index n');
axis([0 20 -1.3 1.3]);
subplot(3,1,3);
signoi = signal+noise;
stem(n,signoi); title('[Signal + Noise] vs. n'); hold on;
plot(n,signoi); ylabel('Amplitude[ signal+noise]');
xlabel(' time index n')

figure(2)
subplot(3,1,1)
N = [.5 .5];
D = 1;
movave2 = filter(N,D,signoi);
plot(n,signal,n,signoi,'s--',n,movave2,'o-.');
legend('signal','signal+noise','moving ave/2term')
title('Various moving averages'); ylabel('magnitude')
subplot(3,1,2)
N=[.33 .33 .33];
D=1;
movave3=filter(N,D,signoi);
plot(n,signal,n,signoi,'s--',n,movave3,'o-'); ylabel('magnitude')
legend('signal','signal+noise','moving ave/3term')
subplot(3,1,3)
N = [.25 .25 .25 .25];
D = 1;
```

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL**

```
movave4 = filter(N,D,signoi);
plot(n,signal,n,signoi,'s--',n,movave4,'o-.');
legend('signal','signal+noise','moving ave/4term')
ylabel('magnitude'); xlabel(' time index n')

figure(3)
plot(n,signal,n,signoi,'ks--',n,movave2,'ko-.');
legend('signal','signal+noise','moving ave/2term');
title('Best approximation using moving averages');
ylabel('magnitude'); xlabel(' time index n')

figure(4)
err2 = signal-movave2;
stem(n,err2); hold on; plot(n,y,n,err2);
title('error=[mov.ave/2terms ] vs t');
ylabel('error'); xlabel(' time index n')
error1= sum(abs(signal-signoi)/21);
error2 = sum(abs(signal-movave2)/21);
error3 = sum(abs(signal-movave3)/21);
error4 = sum(abs(signal-movave4)/21);
disp('*****ANALYSIS OF ERROR*****')
disp('*****no ave 2 term ave 3 term ave 4 term ave*****')
disp(' no ave 2 term ave 3 term ave 4 term ave ');
disp([error1 error2 error3 error4])
disp('*****ANALYSIS OF ERROR*****')
disp('*****no ave 2 term ave 3 term ave 4 term ave*****')
```

Back in the command window the script file *averages* is executed three times, and the results are shown as follows and in the plots of Figures

```
averages
*****
*****ANALYSIS OF ERROR*****
***** no
    ave 2 term ave 3 term ave 4 term ave
    0.5389 0.5276      0.7011      0.9793
*****
>> averages
*****
*****ANALYSIS OF ERROR*****
*****no ave 2 term ave 3 term ave 4 term ave 0.5272
    0.4794 0.6574 0.9033
*****
>> averages
-----
*****ANALYSIS OF ERROR*****
*****no ave 2 term ave 3 term ave 4 term ave 0.4144
```

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL**

0.3750 0.5572 0.8223

Observe that the results obtained by executing the script file *averages* three times are not the same, but in each case the best results are achieved when the moving average uses two terms.

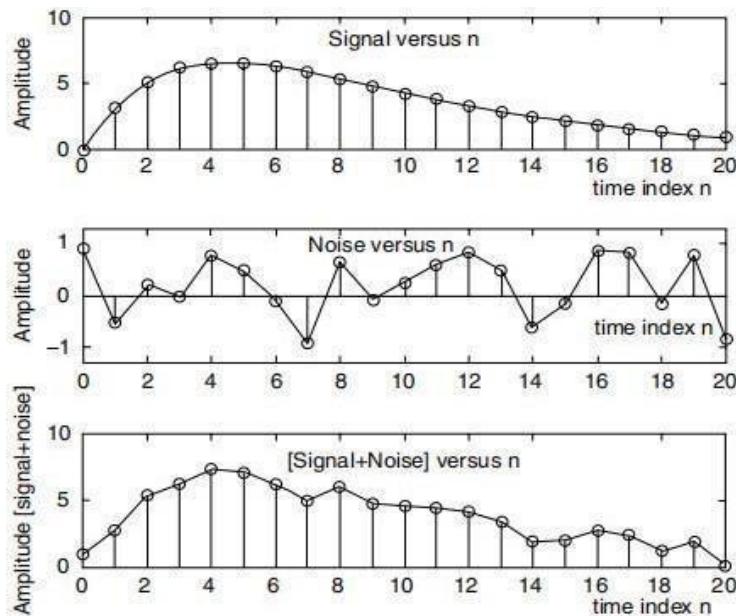
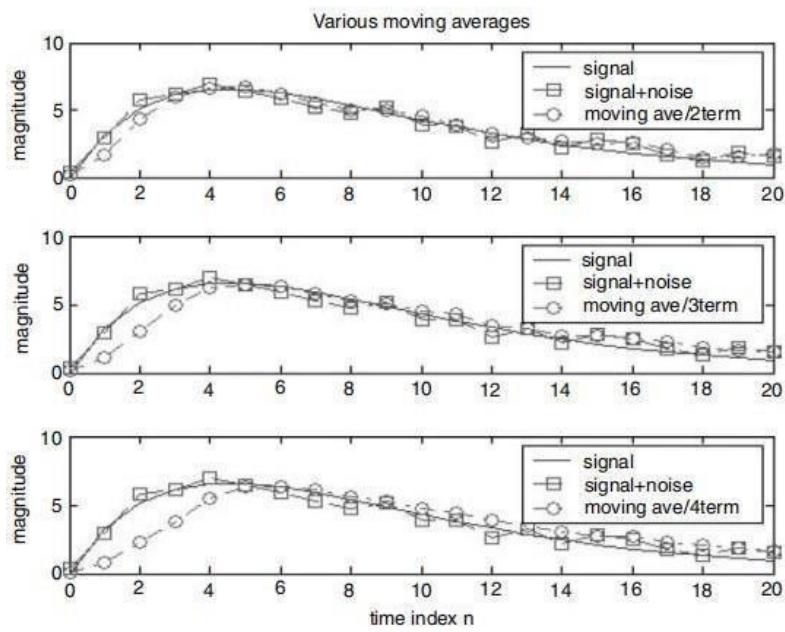
Figure 1.6- Plots of $f(n)$, noise signal [$n(t)$], and $[f(n) + n(t)]$ 

Figure 1.7- Plots of various moving averages

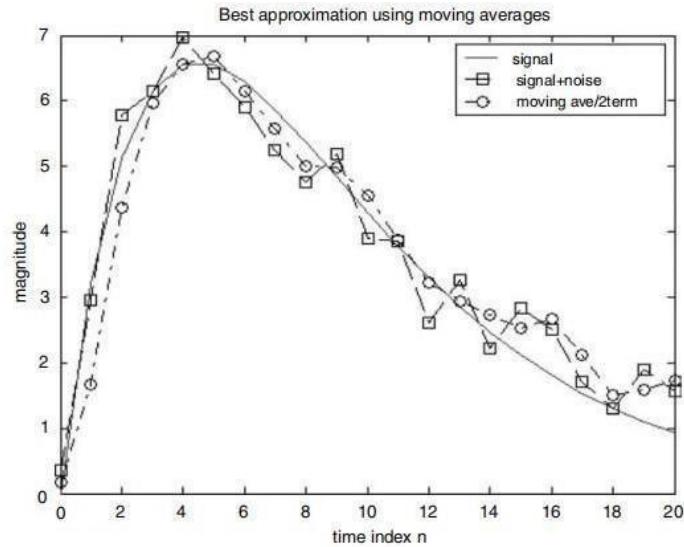
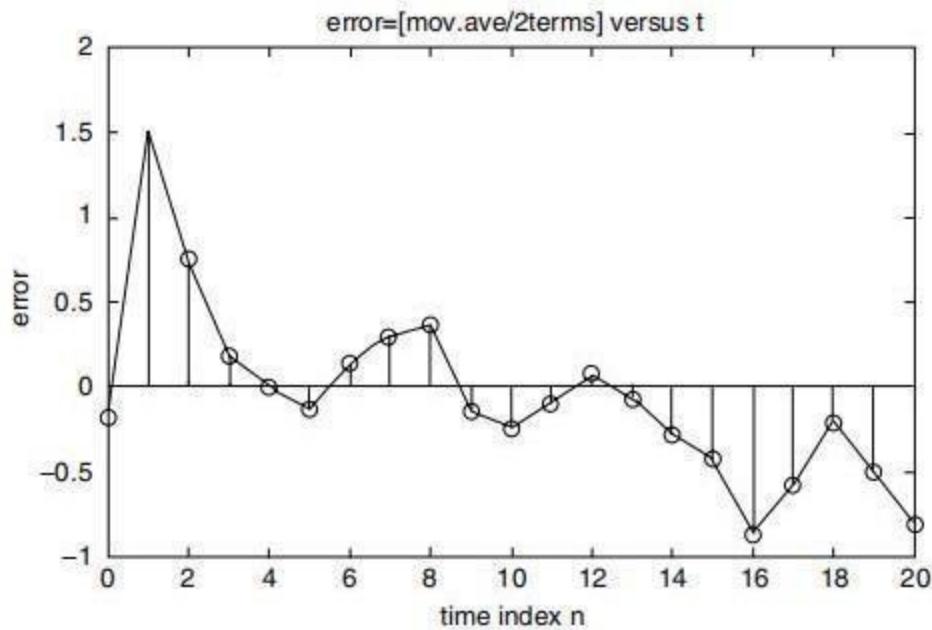
**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL**Figure 1.8- Plots of $f(n)$, $[f(n) + n(t)]$, and best moving average

Figure 1.9- Error plot of moving average using two terms

1.2 Review Exercise?

- a) Enlist the main features of MATLAB Control System Toolbox.

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL**

Transfer-function, state-space, zero-pole-gain, and frequency-response, Step response, Nyquist plot, and other time-domain and frequency-domain tools for analyzing stability and performance.
PID Controller Tuning, gain-scheduled, and arbitrary SISO and MIMO control systems.

b) For what purpose Simulink is used?

Simulink provides a graphical editor, customizable block libraries, and solvers for modeling and simulating dynamic systems. It is integrated with MATLAB, enabling you to incorporate MATLAB algorithms into models and export simulation results to MATLAB for further analysis.

c) What is real time application of digital control system?

Real-time applications of digital control systems are industrial automation, power systems, aerospace and defense, robotics, communication, automotive systems, renewable energy systems, Environmental control, and traffic control system.

d) What are examples of digital devices?

Examples of such devices are

Personal Computers (Desktops and Laptops),
Smartphones, and Tablets,
Calculators,
Digital Weighing Machine,
Accounting machines,
Digital clock,
ATM (Automated Teller Machine).

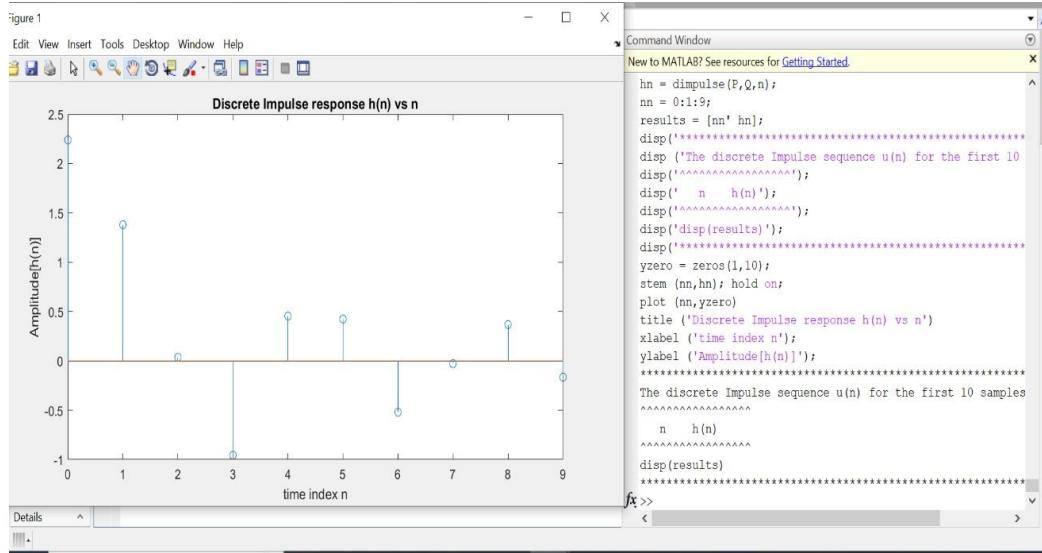
e) Attach the Plot of Example 1 to Example 4 and write a MATLAB program for the following system

Example-1:

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL**

```
n = 10;
P = [2.24 2.5 2.25];
Q = [1 .5 .68];
hn = dimpulse(P,Q,n);
nn = 0:1:9;
results = [nn' hn];
disp('*****');
disp('The discrete Impulse sequence u(n) for the first 10 samples is:');
disp('~~~~~');
disp(' n h(n)');
disp('~~~~~');
disp('disp(results)');
disp('*****');
yzero = zeros(1,10);
stem(nn, hn); hold on;
plot(nn,yzero)
title ('Discrete Impulse response h(n) vs n')
xlabel ('time index n');
ylabel ('Amplitude[h(n)]');
```

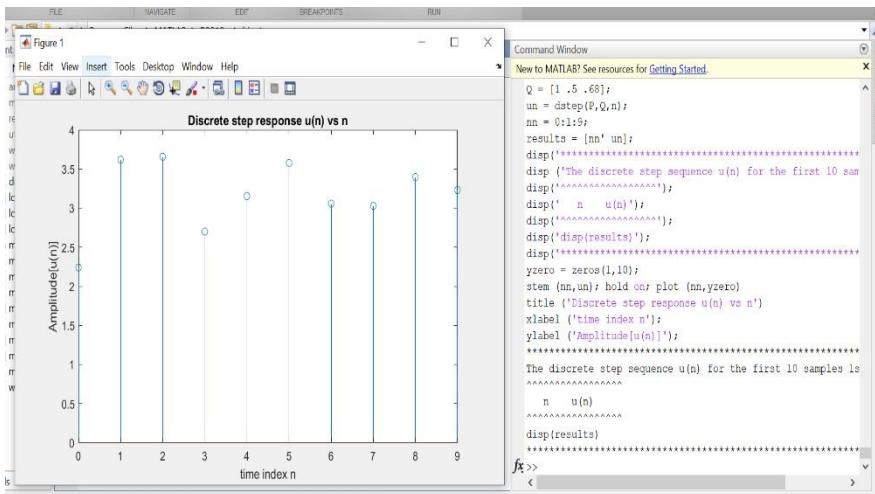
I	[0.8000;0.400...
n	10
nn	[0,1,2,3,4,5,6,...
P	[2.2400,2.500...
Q	[1,0.5000,0.6...
R	[-1,1,1;10,5,0...
result	[0.8000,0.400...
results	10x2 double
sys	1x1 tf
sysc	1x1 tf
syscz	1x1 tf
syscz2	1x1 tf
syscz3	1x1 tf
sysz	1x1 tf
sysz1	1x1 tf
sysz2	1x1 tf
tout	63x1 double
V	[0;10;10]
yzero	[0,0,0,0,0,0,0,...

**Example-2:**

DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)DIGITAL CONTROL

```
1 % Script file: disc_impulse
2
3 n = 10;
4 P = [2.24 2.5 2.25];
5 Q = [1 .5 .68];
6 un = dstep(P,Q,n);
7 nn = 0:1:9;
8 results = [nn' un'];
9 disp('*****');
10 disp('The discrete step sequence u(n) for the first 10 samples is:');
11 disp('*****');
12 disp(' n u(n)');
13 disp('*****');
14 disp('Step(results)');
15 disp('*****');
16 yzero = zeros(1,10);
17 stem(nn,un); hold on; plot(nn,yzero)
18 title ('Discrete step response u(n) vs n')
19 xlabel ('time index n');
20 ylabel ('Amplitude[u(n)]');
21
```

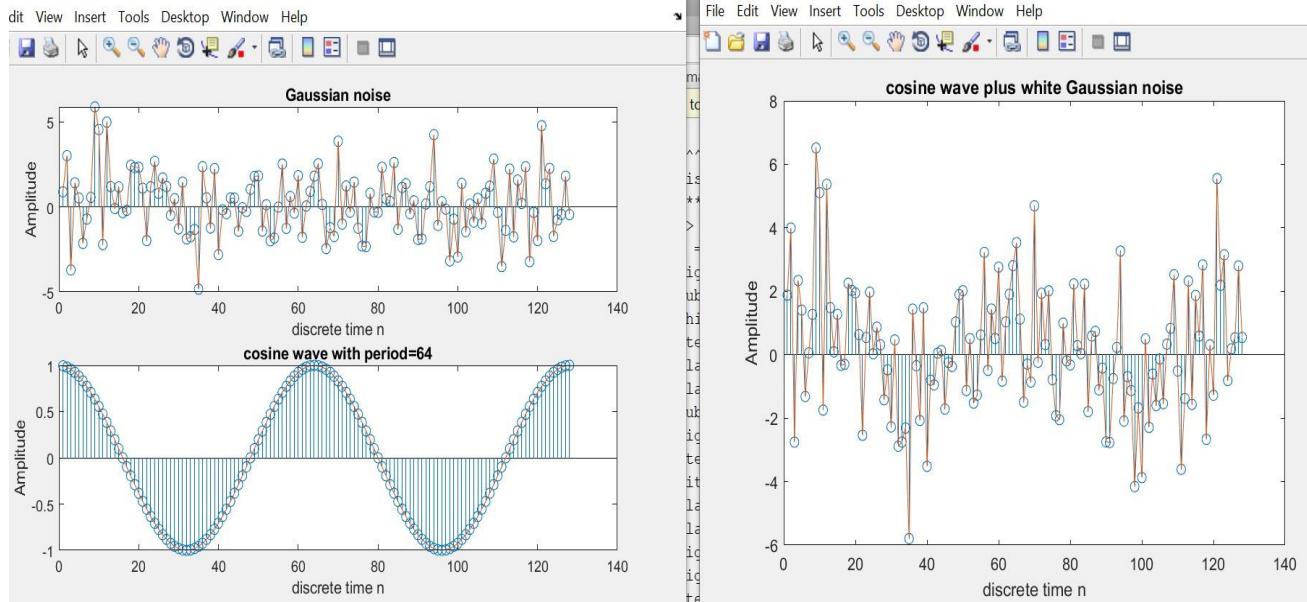
```
d [1.2.1]
Nn [2.2400:1.380...
I [0.8000:0.400...
m 10
nn [0.1.2.3.4.5.6...
P [2.2400:2.500...
Q [1.0.5000:0.6...
B [-1.1.10.5.0...
result [0.8000:0.400...
results 10x2 double
sys 2x1 tf
sysc 2x1 tf
sysca 2x1 tf
sysc2 2x1 tf
sysca2 2x1 tf
sysr 2x1 tf
sys1 2x1 tf
sys2 2x1 tf
tout 6x1 double
un [2.2400:3.620...
V [0.10.10...
yzero [0.00.0.0.0...
```



Example-3:

```
2 % Script file: signal_noise
3 n = 1:128; P = 2.7;
4 figure(1)
5 subplot (2,1,1)
6 white_noise = sqrt(P)*randn(1,128);
7 stem (n, white_noise); hold on; plot (n,white_noise);
8 xlabel('discrete time n');
9 ylabel('Amplitude'); title('Gaussian noise');
10 subplot (2,1,2)
11 signal = cos(2*pi*n/64);
12 stem (n,signal); hold on; plot (n,signal);
13 title( 'cosine wave with period=64');
14 xlabel('discrete time n');
15 ylabel ('Amplitude');
16 figure(2)
17 signal_noise = white_noise+signal;
18 stem(n,signal_noise);hold on;plot (n,signal_noise);
19 title('cosine wave plus white Gaussian noise')
20 xlabel ('discrete time n');
21 ylabel ('Amplitude');
22
23
```

```
~~~~~
disp(results)
*****
>> % Script file: signal_noise
n = 1:128; P = 2.7;
figure(1)
subplot (2,1,1)
white_noise = sqrt(P)*randn(1,128);
stem (n, white_noise); hold on; plot (n,white_noise);
xlabel('discrete time n');
ylabel('Amplitude'); title('Gaussian noise');
subplot (2,1,2)
signal = cos(2*pi*n/64);
stem (n,signal); hold on; plot (n,signal);
title( 'cosine wave with period=64');
xlabel('discrete time n');
ylabel ('Amplitude');
figure(2)
signal_noise = white_noise+signal;
stem(n,signal_noise);hold on;plot (n,signal_noise);
title('cosine wave plus white Gaussian noise')
xlabel ('discrete time n');
ylabel ('Amplitude');
fx>> |
```

DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)DIGITAL CONTROL

Example-4:

DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)DIGITAL CONTROL

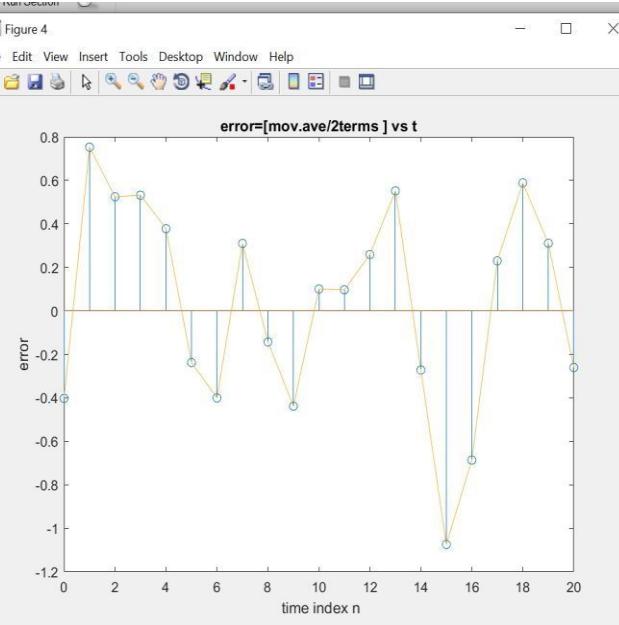
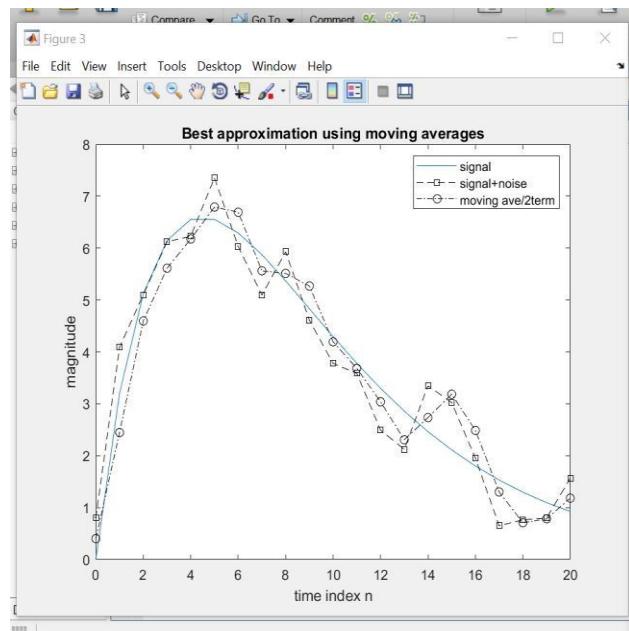
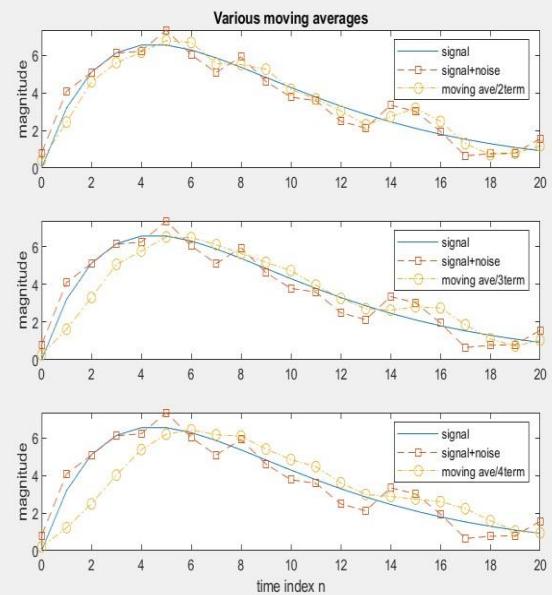
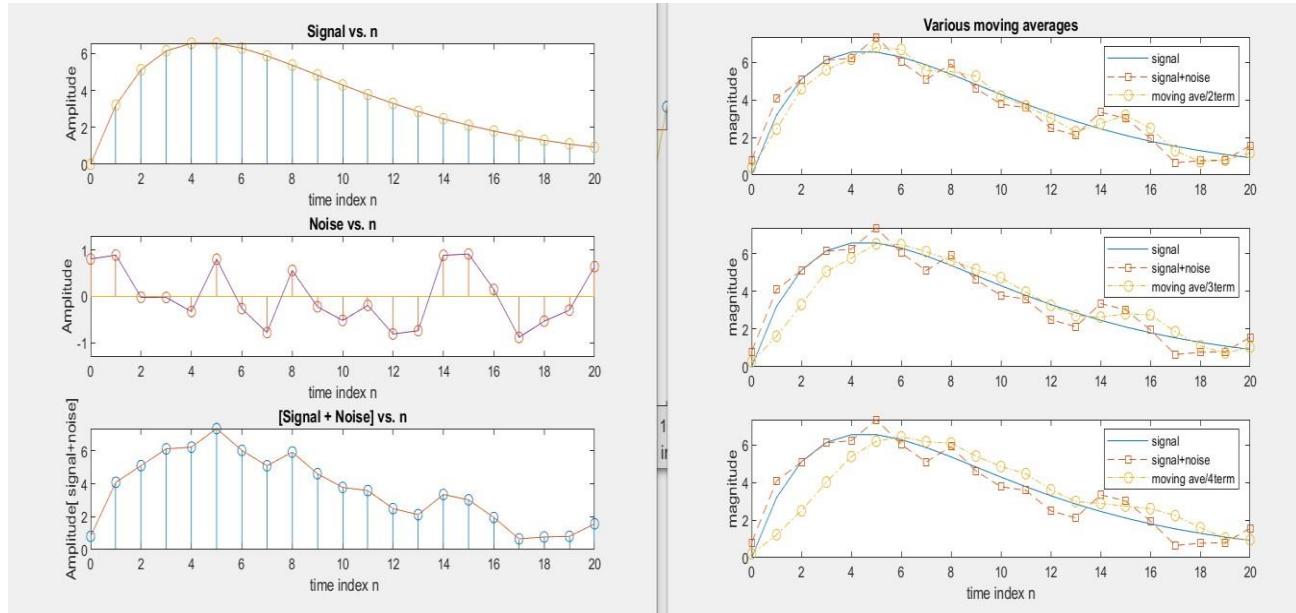
```
2 % Script file: averages
3 n = 0:1:20;
4 figure (1)
5 subplot (3,1,1)
6 signal = 4.*n.*(.8.^n);stem(n, signal);hold on;
7 plot (n,signal,n,signal, 'o');
8 title('Signal vs. n')
9 ylabel ('Amplitude'); xlabel('time index n');
10 subplot (3,1,2)
11 noise = 2.*rand(1,21)-1.0; y = zeros(1,21);
12 plot(n,y);title( 'Noise vs. n');hold on;
13 stem (n, noise);hold on;plot (n,y,n,noise);
14 ylabel ('Amplitude'); xlabel('time index n');
15 axis ([0 20 -1.3 1.3]);
16 subplot (3,1, 3);
17 signoi =signal+noise;
18 stem(n,signoi);title('[Signal + Noise] vs. n');hold on;
19 plot (n, signoi);ylabel('Amplitude[ signal+noise]');
20 xlabel(' time index n')
21 figure(2)
22 subplot (3,1,1)
23 N = [.5 .5];
24 D = 1;
25 movave2 = filter(N,D,signoi);

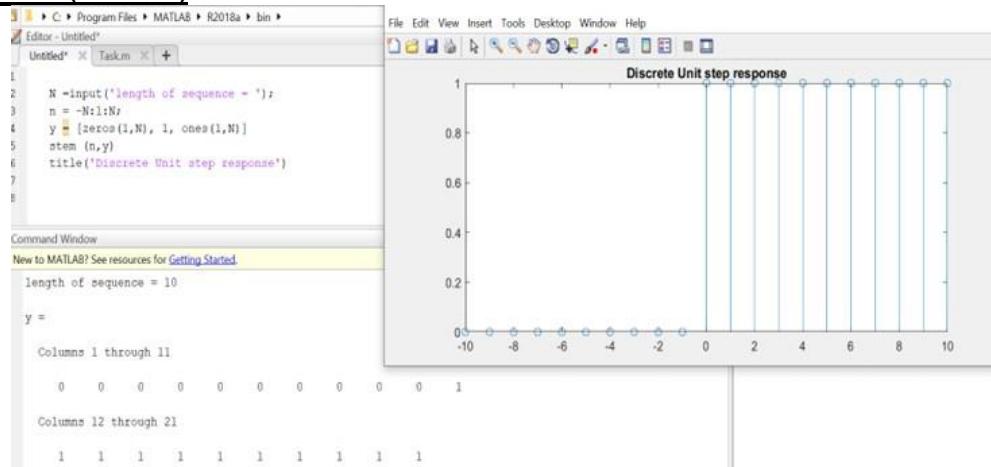
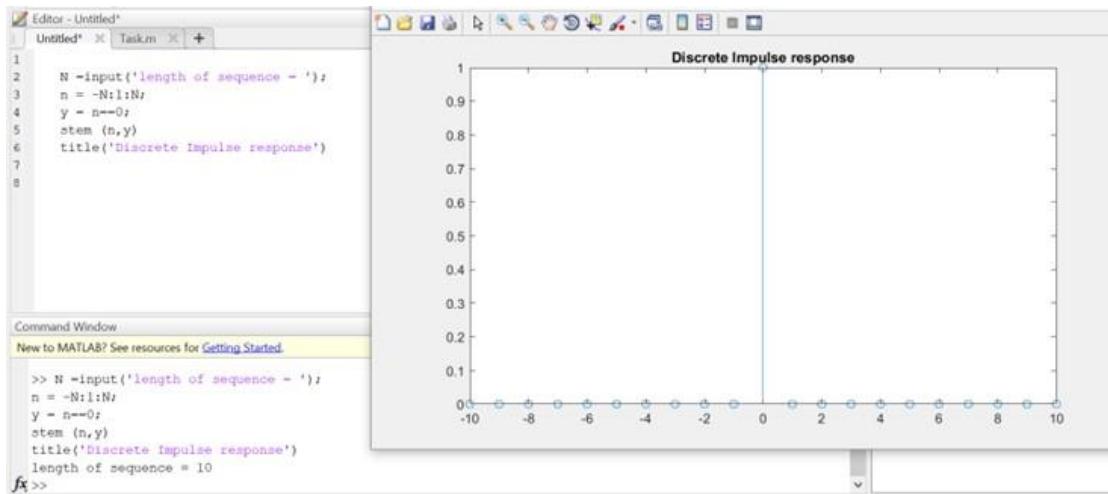
26 movave2 = filter(N,D,signoi);
27 plot (n, signal,n,signoi, 's--',n, movave2, 'o-.');
28 legend('signal', 'signal+noise', 'moving ave/2term')
29 title('Various moving averages');ylabel('magnitude')
30 subplot (3,1,2)
31 N= [.33 .33 .33];
32 D=1;
33 movave3=filter (N,D,signoi);
34 plot (n, signal,n,signoi, 's--',n, movave3, 'o-.');
35 ylabel ('magnitude')
36 legend('signal','signal+noise', 'moving ave/3term')
37 subplot (3,1,3)
38 N = [.25 .25 .25];
39 D = 1;
40 movave4 = filter (N,D,signoi);
41 plot (n, signal,n,signoi, 's--',n, movave4, 'o-.');
42 legend('signal', 'signal+noise', 'moving ave/4term')
43 xlabel ('magnitude');xlabel( 'time index n')
44 figure(3)
45 plot (n, signal,n,signoi, 'ks--', n, movave2, 'ko-.');
46 legend('signal', 'signal+noise', 'moving ave/2term');
47 title('Best approximation using moving averages');
48 ylabel ('magnitude');xlabel( 'time index n')
49 figure(4)
50 err2 = signal-movave2;

51 plot (n, signal,n,signoi, 'ks--', n, movave2, 'ko-.');
52 legend('signal', 'signal+noise', 'moving ave/2term');
53 title('Best approximation using moving averages');
54 ylabel ('magnitude'); xlabel( 'time index n')
55 figure(4)
56 err2 = signal-movave2;
57 stem (n,y,n,err2);hold on;plot (n,y,n,err2);
58 title('error=[mov.ave/2terms ] vs t');
59 ylabel('error'); xlabel(' time index n')
60 error1= sum(abs(signal-signoi))/21;
61 error2= sum(abs(signal-movave2))/21;
62 error3= sum(abs (signal-movave3))/21;
63 error4= sum(abs (signal-movave4))/21;
64 disp('*****ANALYSIS OF ERROR*****')
65 disp('*****no ave 2 term ave 3 term ave4 term ave ');
66 disp([error1 error2 error3 error4])
67 disp('*****')
68 disp('*****')

69
```

err2
error2
error3
error4
errorl
hn
I
movave2
movave3
movave4
n
sum
N
nn
noise
P
Q
R
result
results
signal
signal_nois
signoi
sys
sysc
SVSC

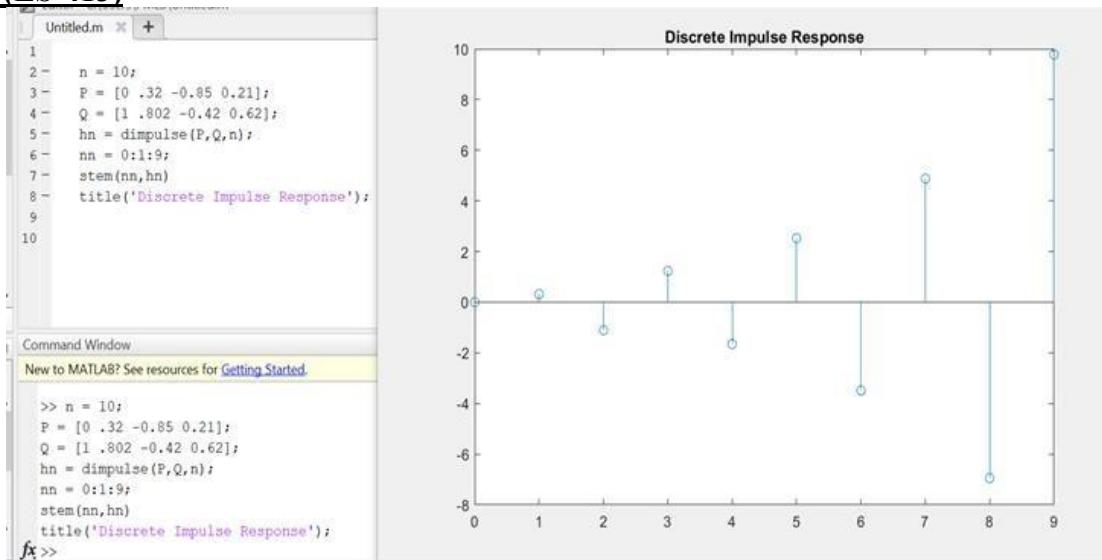
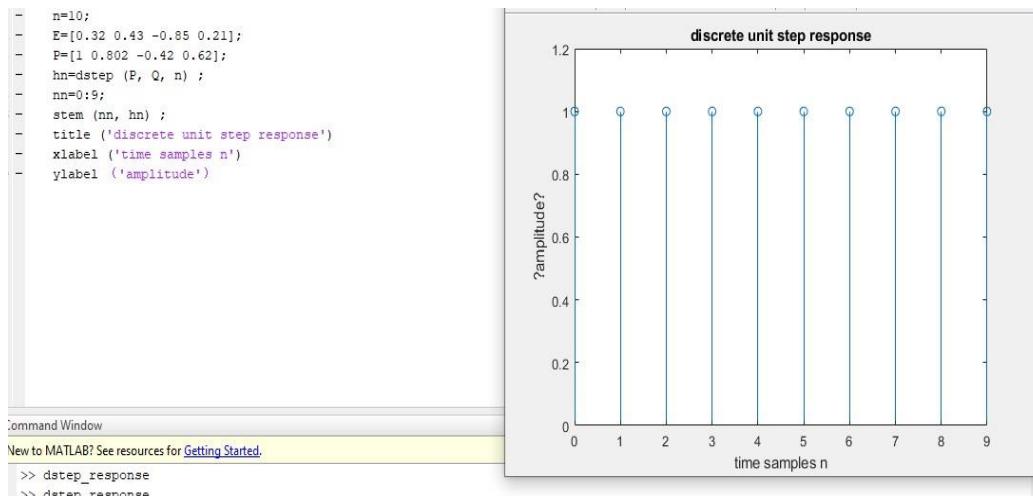
**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL****1. Discrete step and impulse response plots by creating an impulse and step sequence as input****Discrete Step response:**

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL****Discrete Impulse response:**

2 Repeat part a by using the discrete MATLAB functions dimpulse and dstep, and compare the result obtained with the result of part a1

$$H(z) = \frac{0.32 + 0.43z^{-1} - 0.85z^{-2} + 0.2z^{-3}}{1 + 0.802z^{-1} - 0.42z^{-2} + 0.62z^{-3}}$$

Discrete impulse response:

**DEPARTMENT OF ELECTRONIC ENGINEERING
SYSTEMS (ES-413)****DIGITAL CONTROL****Discrete step response:****REFERENCE(S)**

1. <https://www.mathworks.com>
2. Matlab: A Practical Introduction to Programming and Problem Solving by Stormy Attaway.

FINAL CHECK LIST

1. Before you leave, properly save your work and close the software tool.
2. Submit your completed laboratory work, before the next laboratory session.



JAMSHORO

MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY,

413)

DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-**LAB # 02: PLOTTING OF FISCRETE TIME TESTING SIGNALS**Name: KARANRoll # 20ES062

Signature of Lab Tutor:

Date: _____

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	To illustrate how Discrete Time simulation is performed in Simulink	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	
Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	



	TOTAL

Equipment Required: Computer with MATLAB software

Approximate Time Required: Two to three hours

Introduction: Control systems implemented on a digital computer are called digital control systems . Digital control systems differ from their analog counterparts in two important ways. Digital systems operate in discrete time, not continuous time. That is, control computations do not occur continuously (as they do using op-amps for analog systems), but occur at discrete instants in time. These instants are usually regular periodic times, separated by the sampling period T [sec]. The sampling operation introduces artifacts into the signal which are known as signal aliases. It also introduces delay into the closed-loop system which tends to destabilize the control.

Digital systems do not use in finite-precision mathematics. To store the value of a signal such as a control signal in computer memory, the value must be quantized. This means that the value is rounded to the nearest number which can be stored. The coefficients of the transfer function of the controller must also be quantized. These are two separate issues: signal quantization and coefficient quantization.

In this lab period you will implement and simulate a discrete-time system in Simulink.

Notation for Sampling

Consider a continuous-time signal $x(t)$ If we look at the signal at discrete points in time that are separated by a constant sampling period T , then we have the set of values $x(kT)$ where k is an integer. To simplify notation, we say that $x[k] = x(kT)$ where $x[k]$ is a discrete-time signal (denoted using square brackets) and $x(t)$ is a continuous-time signal (denoted by parentheses). When you read $x[k]$ you should understand that some sampling period T is implied, which must be somehow specified else where.

Linear Constant Coefficient Difference Equations

The continuous-time systems you have seen are defined by Linear Constant Coefficient .Ordinary Differential Equations (LCCODEs). Linear, time invariant, lumped discrete-time systems may be defined by Linear Constant Coefficient Difference Equations (LCCDEs). A controller would be implemented with a LCCDE such as:

$$\sum_{i=0}^n a_i u[k-i] = \sum_{i=0}^m b_i e[k-i],$$

with input $e[k]$ and output $u[k]$ and a_i and b_i which determine the transfer function of the controller.An example discrete-time system is a discrete-time integrator. If the sampling period is T , then the output of an integrator may be approximated by (rectangular rule)

$$y[k] = T \sum_{i=0}^k x[i],$$

where $y[k]$ is the output of the integral and $x[k]$ is the input signal. Note that

$$y[k-1] = T \sum_{i=0}^{k-1} x[i] \text{ so we can write}^2$$

$$y[k] = y[k-1] + T x[k].$$



Discrete-Time Simulation

When we consider implementing the above equation, we find that we need two quantities: $x[k]$ and $y[k-1]$. The input to the system is $x[k]$, and $y[k-1]$ is a delayed version of the system's output. Therefore, we can implement the above equation with a feedback loop which has a delay in it.

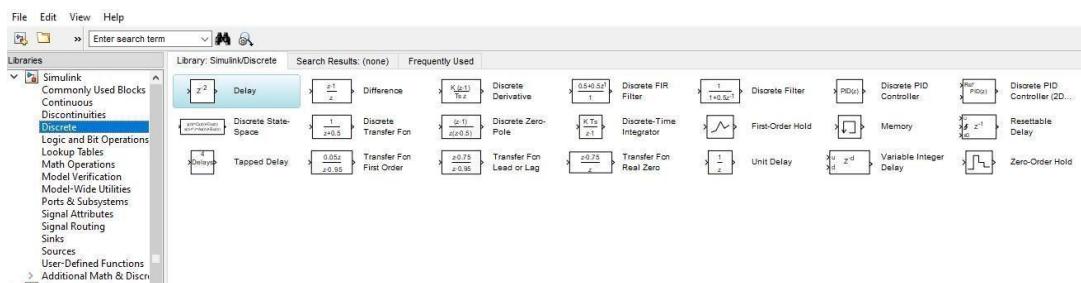


Figure 2.1- Simulink's discrete-time library.

Simulink has a number of blocks which will aid you in making discrete-time simulations. These are part of the Discrete-Time Simulink library, and are shown in Fig. 2.1. The two blocks of interest to us now are the Zero-Order Hold and Unit Delay blocks. Over the course of the semester, you will learn to use many of the other blocks as well. The reason for the label $1/z$ on the unitdelay will also become apparent.

To continue with the integrator example, consider the block diagram in Fig. 2.2. A source (in this case a sine wave) generates an input signal $x(t)$. This signal must be sampled to make $x[k]$. The Simulink block which does this function is the Zero-Order Hold block. The $x[k]$ signal is scaled by T to produce $Tx[k]$. The output of the integrator is $y[k]$ which is computed as $x[k] + y[k-1]$. The final remaining signal $y[k-1]$ is computed from $y[k]$ by passing it through a Unit Delay block. The zero-order hold, amplifier and unit delay blocks require the sampling period T . You will find that it is best to enter these as symbolic T and define the value of T in the MATLAB workspace. This allows easy and consistent change of sampling rate.

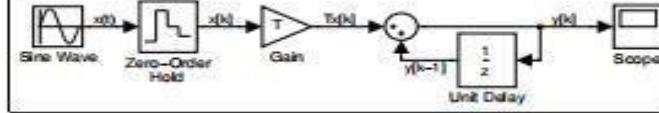


Figure 2.2- Discrete-time integrator implemented in Simulink

Laboratory Experiment

A leaky integrator is defined by the difference equation

$$y[k] = ay[k - 1] + Tx[k], \quad \text{where}$$

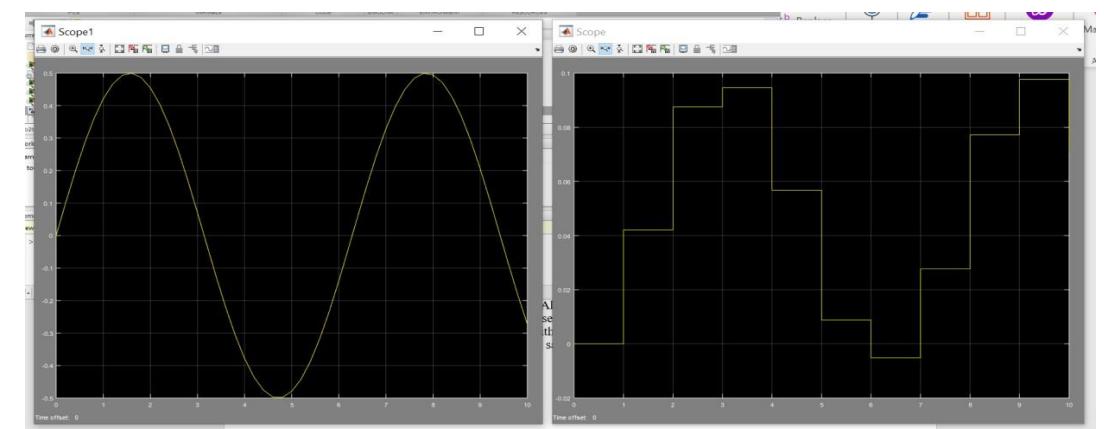
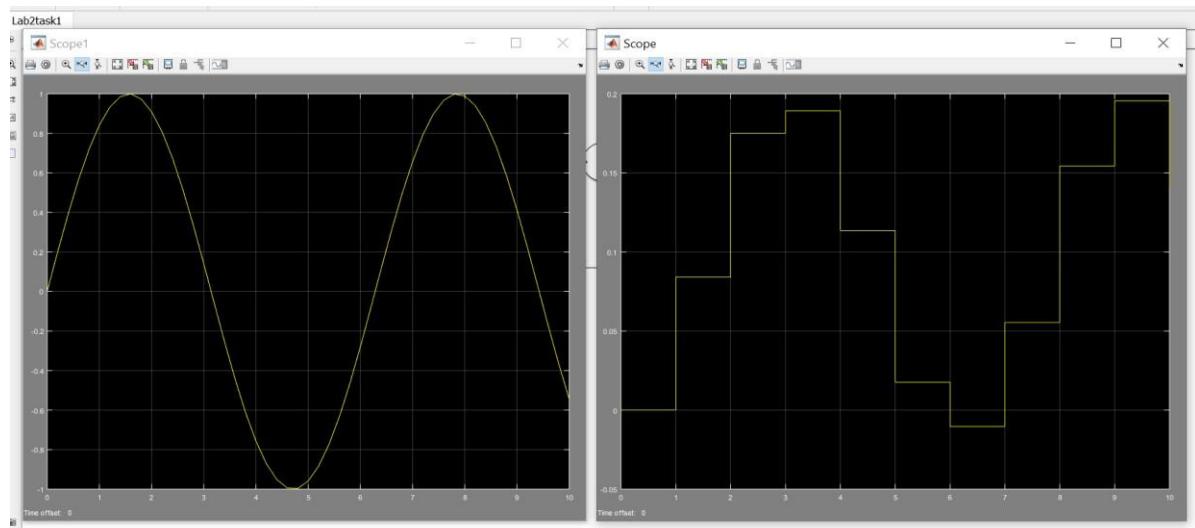
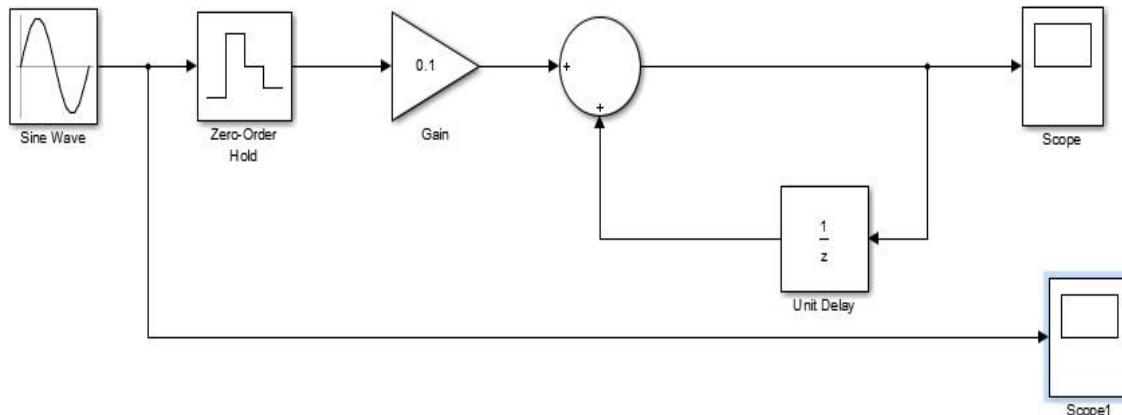
usually $0 < a < 1$. The system is stable, however, for $1 < a < 1$.

1. Implement a leaky integrator in Simulink. Plot the step response of the system for a number of values of $0 \leq a \leq 1$. Also plot the step response of the system for a number of values $-1 \leq a < 0$. Use $T = 0.1$ sec.
2. Plot the response of the system with the input being a sine wave and $a=0.9$. Use a sine wave of frequency 1 rad/sec. Use sampling periods $T = 0.1; 0.5; 1$ sec.

Review Exercise

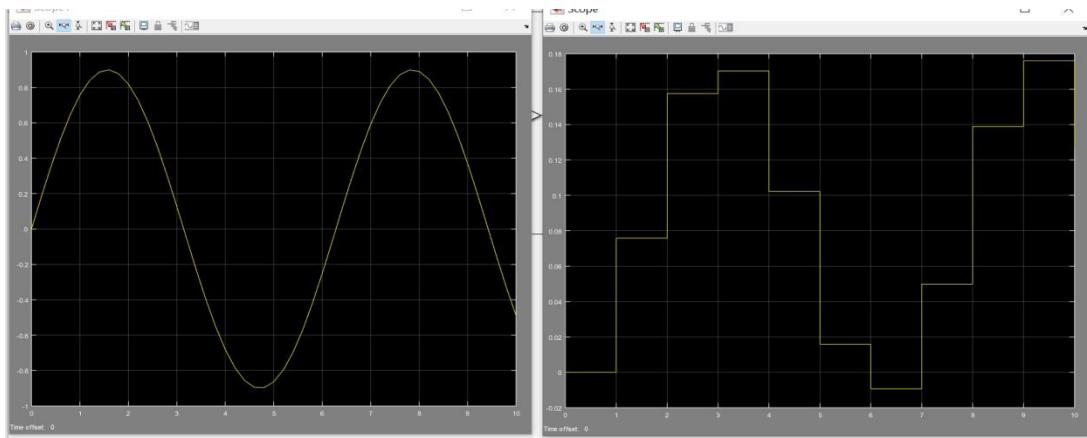


- a) Include your Simulink diagram and plots in your report. Comment on the shapes of the output curves. What prominent features do you see? How do you explain them? Your leaky integrator also has another name suggest what kind of discrete-time filter it is implementing (Hint: It is either a low-pass, band-pass or high-pass filter).

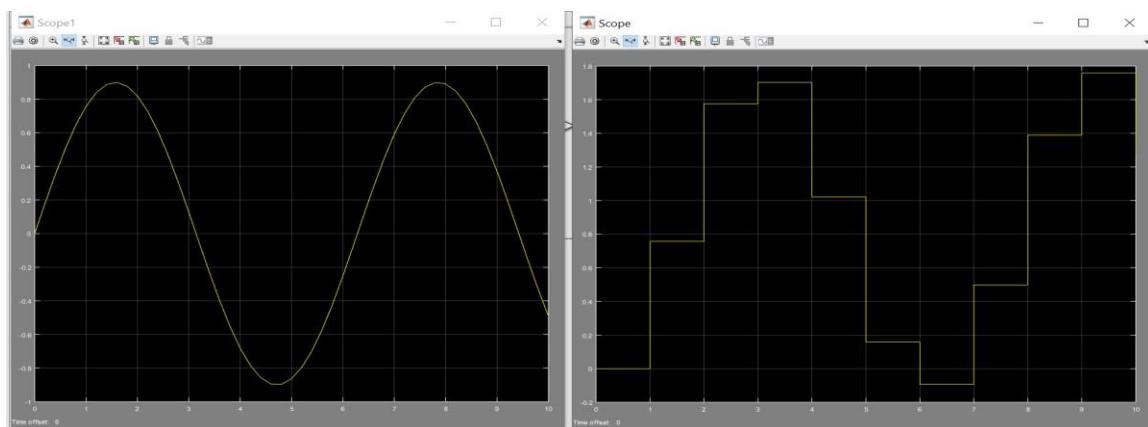




(a=0.5, t=0.1)



(a=0.9, t=0.1)



(a=0.9, t=1)

Comments:

Increasing the gain of a leaky integrator amplifies input signal response and accelerates system dynamics, potentially leading to instability and saturation if not carefully controlled.

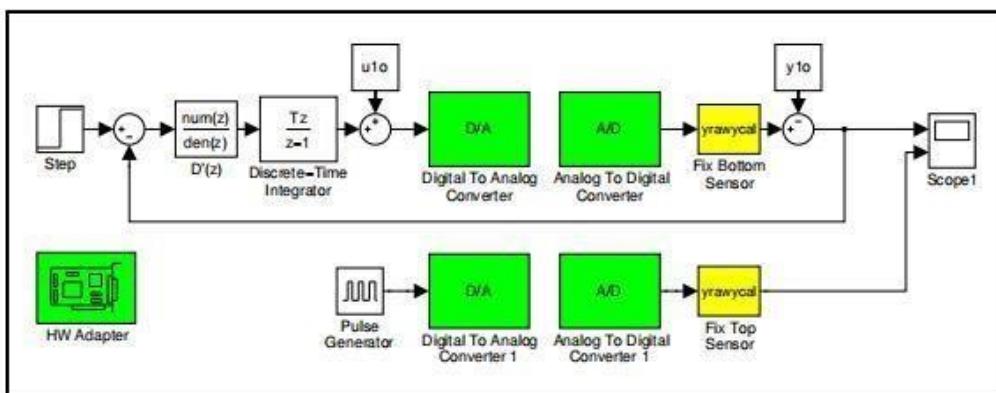
- b) Simulate the Single Input Multi Output (SIMO) system in simulink



DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-

413)





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)
LAB # 03: Z-Transform & Inverse Z-Transform

Name: KARAN

Roll #: 20ES062

Signature of Lab Tutor:

Date: _____

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	To analyze Z-Transform and inverse ZTransform practically using MATLAB	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

Equipment Required: Computer with MATLAB software

Approximate Time Required: Two to three hours

Z-transform

It converts a discrete time-domain signal, which is a sequence of real or complex numbers, into a complex frequency-domain representation. The z-transform, like many other integral transforms, can be defined as either a one-sided or two-sided transform. The bilateral or two-sided Z-transform of a discrete-time signal $x[n]$ is the function $X(z)T$ alternatively, in cases where $x[n]$ is defined only for $n \geq 0$, the single-sided or unilateral Z- transform this definition is used when the signal is causal. As analog filters are designed using the Laplace transform, recursive digital filters are developed with a parallel technique called the z-transform. The overall strategy of these two transforms is the same: probe the impulse response with sinusoids and exponentials to find the system's poles and zeros. The Laplace transforms deals with differential equations, the s-domain, and the s-plane. Correspondingly, the ztransform deals with difference equations, the z-domain, and the z-plane. However, the two techniques are not a mirror image of each other; the s-plane is arranged in a rectangular coordinate system, while the z-plane uses a polar format. Recursive digital filters are often designed by starting with one of the classic analog filters, such as the Butterworth, Chebyshev, or elliptic. A series of mathematical conversions are then used to obtain the desired digital filter. The Z transform of a discrete time system $X[n]$ is defined as Power Series.

ztrans **z-**
transform.

Syntax

$F = \text{ztrans}(f)$

Description

$F = \text{ztrans}(f)$ is the z-transform of the scalar symbol f with default independent variable n . The default return is a function of z .

Example 1: Find the z-transform of:



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

$$x(n) = \frac{4}{n} u(n)$$

Solution

syms z n

ztrans((1/4)^n) **Result**

ans =

$$z/(z - 1/4)$$

Z-Transfer Function

The z-transfer function, also known as the Pulse Transfer Function $H(z)$ is defined as the ratio of the ztransform of the output to the z-transform of the input when all initial conditions are zero.

Mathematically,

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \dots + b_M z^{-M}}{1 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_N z^{-N}}$$
 (1)

+
+ 1 +

The roots of the denominator of $H(z)$ are called “poles” and those of the numerator are called “zeros”. The above equation has M zeros and N poles

A system is said to be stable if all of its poles lie inside of a unit circle on the z-plane. The following example illustrate the use of MATLAB to compute zero and poles of a transfer function and to plot them onto the z-plane.

Example 2: Plot poles and zeros of the following pulse transfer function onto the z-plane and check its stability.

$$H(z) = \frac{2 - z^{-1}}{1 - z^{-1} - 0.02z^{-2}}$$

$$= \frac{2 - z^{-1}}{0.1z - 1}$$

Solution



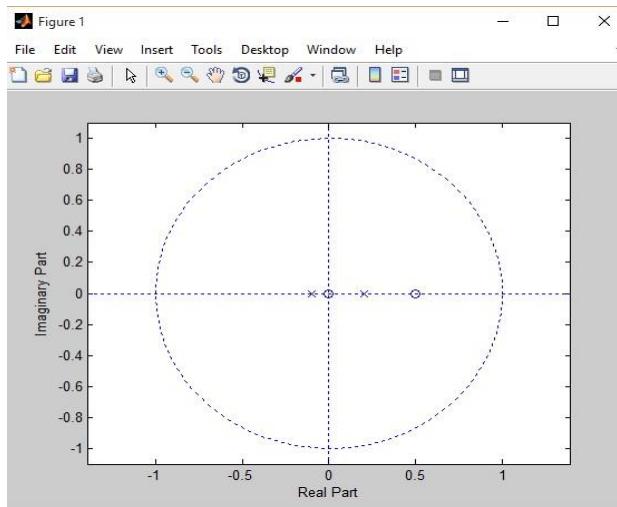
MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

num = [2 -1]; % numerator of H(z)

den = [1 -0.1 -0.02]; % denominator of

H(z) zplane(num,den)

Result



“x” indicate the poles and

“0” indicate the zeros.

Figure:3.1 Since all the poles lie inside the unit circle, the system is stable.

Computing Inverse Z-Transform using MATLAB function “iztrans”: Syntax

f = iztrans(F) **Description**

f = iztrans(F) is the inverse z-transform of the scalar symbolic object F with default independent variable z. The default return is a function of n.

Example 3: Find the inverse z-transform:

$$\frac{2z}{2z-1} x(z)=\underline{\hspace{2cm}}$$

Solution: syms z n

iztrans((2*z)/(2*z-1))

Result: ans

=

$$(1/2)^n$$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

As we know, there are two methods to find inverse z-transform: Partial fraction method and Power series method
MATLAB has built-in functions to find the inverse z-transform by using both of the above methods.

(a) Partial Fraction Method:

Example 4: Using MATLAB determine the partial fraction expansion of

$$18z^3$$

$$H(z) = \frac{18z^3 + 3z^2 - 4z - 1}{18z^3}$$

Solution:

Use the following MATLAB program:

```
num=[18 0 0 0]; den =[18 3 -4 -  
1];  
[r, p, k] = residuez(num,den); % students are advised to get on-help about "residuez" to  
% findout more about it.  
disp('Residues'); disp(r)  
disp('Poles'); disp(p) disp('Direct  
terms'); disp(k)
```

(b) Power Series Method

To find inverse z-transform by power series, one may use the built-in function "impz".

You may also use MATLAB function "filter" for the same purpose.

Example:5 Using MATLAB determine the power series expansion of $\frac{1}{1-1.5z^{-1}+0.5z^{-2}}$

Solution:

Use the following MATLAB program

```
L = 11; % Length of output vector  
num = [1 0 0]; den = [1 -1.5 0.5]; [y, t] =  
impz(num,den,L);  
disp('Coefficients of the power series expansion'); disp(y)
```

Example 6: Repeat example 5 with the built-in function "filter"

Solution: N = 11; num

```
= [1 0 0]; den =  
[1 -1.5 0.5]; x =[1
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

```
zeros(1,N-1)]; y =  
filter(num,den, x);  
disp('Coefficients of the power series expansion'); disp(y)
```

Frequency Response:

The Freqz function computes and display the frequency response of given z-Transform of the function **Freqz(b,a,Fs)**

b= Coeff. Of Numerator a= Coeff. Of Denominator Fs= Sampling Frequency

Example7: Create the script file that analyzes the discrete system transfer function given by

$$H(e^{jW}) = \frac{G(e^{jW})}{F(e^{jW})} = \frac{5 + 0.9e^{-jW} + e^{-2jW}}{1 - 0.7e^{-jW} + 0.3e^{-j2W}}$$

and returns the following:

1. The pole/zero plot of $H(e^{jW})$ (evaluates the system stability)
2. The plot of $\text{abs}\{H(e^{jW})\}$ versus W
3. The plot of angle $\{\text{angle}(H(e^{jW}))\}$ versus W
4. List all the poles, zeros, and stand alone term of the transfer function $H(z=e^{jW})$ and compare the results obtained with the results of part a
5. Determine the ROC of $H(z = e^{jW})$
6. The magnitude of the system poles (length from origin)
7. The partial fraction expansion of $H(z)$
8. $h(n)$ by using the symbolic MATLAB inverse ZT of $H(z)$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

MATLAB Solution

```
%Script file = analys
num = [5 0.9 1];
den = [1 -0.7 0.3];
figure(1)
zplane(num,den); title('Pole and zero plot of H(z)')
figure(2)
W = -pi:pi/99:pi;
[H,W] = freqz(num,den,W);

magH = abs(H);phaseH = angle(H);
subplot (2,1,1);
plot (W,magH);title ('abs[H(exp(-jW))] vs W');ylabel('Magnitude');
subplot (2,1,2);
plot (W,phaseH);title('angle[H(exp(-jW))] vs W');
ylabel ('Phase in rad.');//xlabel('frequency W in rad');
[z,p,k] = tf2zp(num,den);
m = abs(p);
disp('*****');
disp('***** R E S U L T S *****');
disp('*****');
disp('The zeros of H(z) are:');disp(z);
disp('The poles of H(z) are:');disp(p);
disp('The gain of H(z) is:');disp(k);
disp('The magnitude of the poles are:');
disp(m);disp('The ROC is outside the circle with radius:');
disp(max(m))
% partial fraction expansion
[r,pp,kk] = residuez (num,den);
disp('Partial fraction coefficients of H(z)')
disp('the residues are:')
disp(r);disp('the stand alone term is :');disp(kk);
syms z Fz;
disp('The partial fraction expansion is:')
Fz = r(1)/(z-pp(1))+r(2)/(z-pp(2))+ kk
hn = iztrans(Fz);disp('The impulse response h(n) is:');hn
disp('^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^')
```

The script file *analys* is executed, and the results are as follows:



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

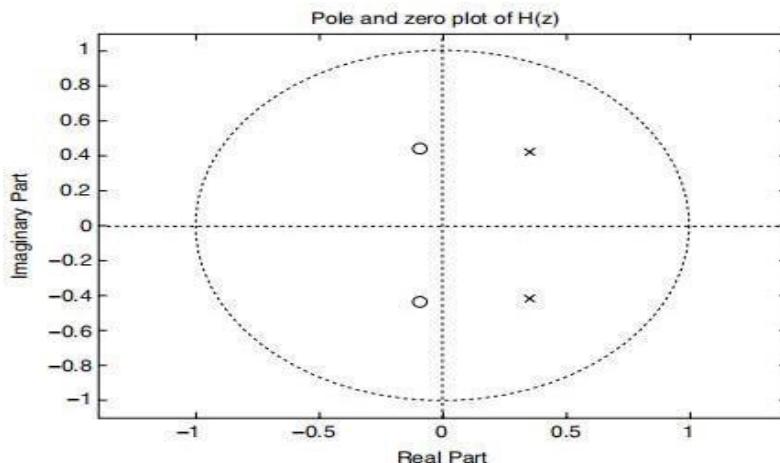
```
>> analysis
*****
***** R E S U L T S *****
*****
The zeros of H(z) are:
-0.0900 + 0.4381i
-0.0900 - 0.4381i
The poles of H(z) are:
0.3500 + 0.4213i
0.3500 - 0.4213i
The gain of H(z) is:
5
The magnitude of the poles are:
0.5477
0.5477
The ROC is outside the circle with radius:
0.5477
Partial fraction coefficients of H(z)
the residues are:
0.8333 - 4.5295i
0.8333 + 4.5295i
the stand alone term is:
kk =
3.3333
```

The partial fraction expansion is:

$$Fz = \frac{(5/6-1/426*i*3723311^{(1/2)})/(z-7/20-1/20*i*71^{(1/2)})+}{(5/6+1/426*i*3723311^{(1/2)})/(z-7/20+1/20*i*71^{(1/2)})+10/3}$$

The impulse response h(n) is:

$$hn = \frac{127/9*charfcn[0](n)-97/18*(-6/(i*71^{(1/2)}-7))^{n-1}-979/1278*i*(-6/(i*71^{(1/2)}-7))^{n-1}71^{(1/2)}97/18*6^n*(1/(7+i*71^{(1/2)}))^{n-1}+979/1278*i*6^n*(1/(7+i*71^{(1/2)}))^{n-1}71^{(1/2)}}{127/9*charfcn[0](n)-97/18*(-6/(i*71^{(1/2)}-7))^{n-1}-979/1278*i*(-6/(i*71^{(1/2)}-7))^{n-1}71^{(1/2)}97/18*6^n*(1/(7+i*71^{(1/2)}))^{n-1}+979/1278*i*6^n*(1/(7+i*71^{(1/2)}))^{n-1}71^{(1/2)}}$$





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

Figure 3.2- Pole/Zero plot of $H(e^{jW})$

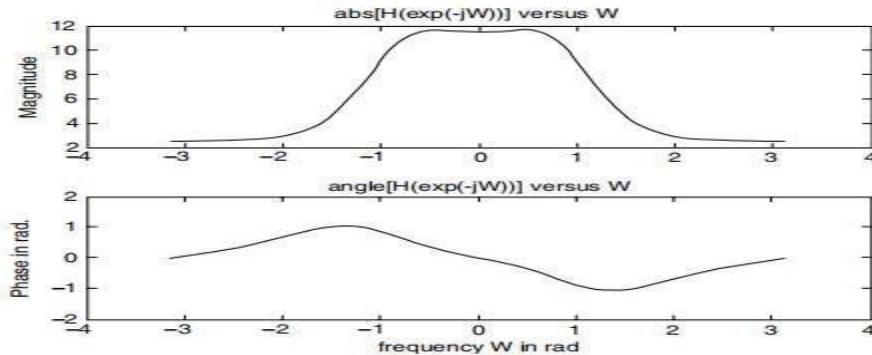


Figure 3.3- Plot of $\text{abs } H(e^{jW})$ and $\text{angle } H\{e^{jW}\}$ }

Example8: Create the script file pole_zeroes that returns the rational transfer function $H(z)$, the numerator and denominator coefficients of $H(z)$, and the first 11 coefficients of their system impulse response $h(n)$ given the following system's zeros, poles, and gain



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

$$\begin{aligned}z_1 &= 0.21 & p_1 &= -0.45 \\z_2 &= 3.14 & p_2 &= 0.67 \\z_3 &= -0.3 + j5 & p_3 &= 0.81 + j0.72 \\z_4 &= -0.3 - j5 & p_4 &= 0.81 - j0.72 \\k &= 2.3 & v_n &= -0.33\end{aligned}$$

MATLAB Solution

```
% Script file: pole_zeroes
z1 = 0.21 ; p1 = -0.45;
z2 = 3.14 ; p2 = 0.67;
z3 = -0.3 + 5j; p3 = 0.81+0.72j;
z4 = -0.3-5j; p4 = 0.81-0.72j;
k = 2.3; p5 = 0.33;
zr = [z1 z2 z3 z4];
p1 = [p1 p2 p3 p4 p5];
z = zr'; p=p1';
[num,den] = zp2tf(z,p,k);
n = 0:10;
disp ('*****')
disp ('***** R E S U L T S *****')
disp ('*****')
tf(num,den,1)
disp('*****')
disp ('Coefficients of the numerator polynomial of H(z)'); disp(num');
disp('*****')
disp ('Coefficients of the denominator polynomial of H(z)');
disp(den');L=11; [y,t] = impz(num,den,L);
disp('*****')
disp ('The first 11 coefficients of the impulse response h(n) are:');
disp('*****')
disp(' coef. num. h(n)')
disp('*****')
[n' y]
disp ('*****')
```

The script file *pole_zeroes* is executed, and the results are as follows:

```
>> pole_zeroes
*****
***** R E S U L T S *****
*****
Transfer function:
2.3 z^4 - 6.325 z^3 + 54.6 z^2 - 192.4 z + 38.05
-----
z^5 - 2.17 z^4 + 1.837 z^3 - 0.1757 z^2 - 0.43 z + 0.1169
Sampling time: 1
*****
Coefficients of the numerator polynomial of H(z)
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

```
0
2.3000
-6.3250
54.6006
-192.4085
38.0520
*****
Coefficients of the denominator polynomial of H(z)
1.0000
-2.1700
1.8366
-0.1757
-0.4300
0.1169
*****
The first 11 coefficients of the impulse response h(n) are:
*****
coef. num.      h(n)
*****
ans =
0          0
1.0000    2.3000
2.0000   -1.3340
3.0000    47.4817
4.0000   -86.5192
5.0000   -236.1448
6.0000   -346.0348
7.0000   -311.8158
8.0000   -125.3485
9.0000   148.4517
10.0000  376.3726
*****
```

Note that the system transfer function is given by

$$H(z) = \frac{2.3z^{-1} - 6.325z^{-2} + 54.6006z^{-3} - 192.4085z^{-4} + 38.0520z^{-5}}{1 - 2.17z^{-1} + 1.8366z^{-2} - 0.1757z^{-3} - 0.43z^{-4} + 0.1169z^{-5}}$$

Note also that the corresponding system impulse response $h(n) = Z^{-1}[H(z)]$ is given by $h(n) = 0 + 2.3\delta(n - 1) - 1.3403\delta(n - 2) + 47.4817\delta(n - 3) - 86.5192\delta(n - 4) - 236.1448\delta(n - 5) - 346.0348\delta(n - 6) - 311.8158\delta(n - 7) - 155.3485\delta(n - 8) + 148.4517\delta(n - 9) + 376.3726\delta(n - 10)$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Review exercise

- a) What is ROC? Repeat example 6 by following transfer function

$$H(Z) = \frac{1+0.3Z^{-1}+0.1Z^{-2}}{5+0.8Z^{-1}+0.4Z^{-2}}$$

The region of convergence, known as the ROC, is important to understand because it defines the region where the z-transform exists. The ROC for a given $x[n]$, is defined as the range of z for which the z-transform converges.

The screenshot shows the MATLAB R2018a interface. The Current Folder browser shows various MATLAB files. The Editor window displays the following code:

```
1 N = 11;
2 num = [1 0.3 0.1];
3 den = [5 0.8 0.4];
4 x =[1 zeros(1,N-1)];
5 y = filter(num,den, x);
6 disp("Coefficients of the power series expansion");
7 disp(y);
8
```

The Workspace browser shows the variables:

Name	Value
den	[5.0000,0.8000,0.4000]
N	11
num	[1,0.3000,0.1000]
x	1x11 double
y	1x11 double

The screenshot shows the MATLAB Command Window. The user has run the script from the previous screenshot. The command window shows the following output:

```
num = [1 0.3 0.1];
den = [5 0.8 0.4];
x =[1 zeros(1,N-1)];
y = filter(num,den, x);
disp("Coefficients of the power series expansion");
Coefficients of the power series expansion
Columns 1 through 10
0.2000 0.0280 -0.0005 -0.0022 0.0004 0.0001 -0.0000 -0.0000 0.0000 -0.0000
Column 11
-0.0000
fx >> |
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)
MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

b) Determine the partial fraction expansion of the z-transform $G(z)$ given by:

$$\frac{5z^2+3z}{14z^3+3z^2+8z^1+9z^0}$$

```
Command Window
New to MATLAB? See resources for Getting Started.
% findout more about it.

disp("Residues");
disp(r)
disp("Poles");
disp(p)
disp("Direct terms");
disp(k)

Residues
    0.0105 - 0.1824i
    0.0105 + 0.1824i
    -0.0210 + 0.0000i

Poles
    0.2442 + 0.9247i
    0.2442 - 0.9247i
    -0.7028 + 0.0000i

Direct terms
    0
```

Workspace variables:

Name	Value
den	[14,3,8,9]
k	0
N	11
num	[0,5,3,0]
p	[0.2442 + 0.9...
r	[0.0105 - 0.18...
x	1x11 double
y	1x11 double


```
Untitled.m
1 num=[0 5 3 0];
2 den =[14 3 8 9];
3 [r, p, k] = residue(num,den); % students are advised to get on-help about "residue"
4 % findout more about it.
5
6 disp("Residues");
7 disp(r);
8 disp("Poles");
9 disp(p);
10 disp("Direct terms");
11 disp(k);
12
```

Current Directory:

- arch
- m3registry
- registry
- util
- win32
- win64
- deploytoolbat
- lodata.xml
- lodataxsd
- lodata_utf8.xml
- matlab.exe
- mbuild.bat
- mcc.bat
- mex.bat
- mexpl
- mexent.bat
- mexsetup.pml
- mexus.pml
- mw_implevbat

a) What is difference between z-transform and Fourier transform?

The Fourier Transform is employed for continuous-time signals in the frequency domain, representing them as a sum of sinusoidal components, while the Z-Transform is designed for discrete-time signals,



operating in the complex z-plane and providing information about system behavior in terms of poles and zeros, making it particularly useful in digital signal processing applications.

LAB # 04: DISCRETE TIME ROOT LOCUS

Name: KARAN

Roll #: 20ES062

Signature of Lab Tutor:

Date: _____

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	<ul style="list-style-type: none">• Know about the Root locus technique.• Analyze the system performance in discrete-time domain using Root locus technique with the help of MATLAB.	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
--------------------	------------	---------------	------------	-------



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	
Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

Equipment Required: PC with MATLAB software

Approximate Time Required: Two to three hours

Theory:

The relative stability and the transient performance of a closed-loop control system are directly related to the location of the closed-loop roots of the characteristic equation in the z-plane. It is frequently necessary to adjust one or more system parameters in order to obtain suitable root locations. Therefore, it is worthwhile to determine how the roots of the characteristic equation of a given system migrate about the z-plane as the parameters are varied; that is, it is useful to determine the Locus of roots in the z-plane as a parameter is varied. The root locus technique is a graphical method for sketching the locus of roots in the z-plane as a system parameter is varied. The root locus method provides graphical information, and therefore an approximate sketch can be used to obtain qualitative information concerning the stability and performance of the system.

Discrete Root Locus

The root-locus is the locus of points where roots of the characteristic equation can be found as a single parameter is varied from zero to infinity. The characteristic equation of our unity-feedback system with simple proportional gain, K, is:

$$1 + KG'(z)H_{zoh}(z) = 0$$

where $G'(z)$ is the digital controller and $H_{zoh}(z)$ is the plant transfer function in z-domain (obtained by implementing a zero-order hold).

Commands used for Root Locus

- `rlocus(SYS)`: computes and plots the root locus of the single-input, single-output LTI model SYS.
- `rlocfind`: Finds root locus gain K for a given set of roots.
- `[K,POLES] = rlocfind(SYS)` is used for interactive gain selection from the root locus plot of the SISO system SYS generated by rlocus. rlocfind puts up a crosshair cursor in the graphics window which is used to select a pole location on an existing root locus. The root locus gain associated with this point is returned in K and all the system poles for this gain are returned in POLES.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

Example 1: Given, $G(z) = K[0.3678(z+0.7189) / (z-1)(z+0.24)]$ & $H(z) = 1$. Using root locus, determine the range of K for which system becomes stable, with the help of MATLAB.

Solution

Since, the characteristic equation is given by $1+G(z)H(z)=0$, this yields

$$1 + K[0.3678(z+0.7189) / (z-1)(z+0.24)] * (1) = 0 \text{ Comparing}$$

it with $1+KG'(z)H_{zoh}(z)=0$, we have

$$G'(z) = [0.3678(z+0.7189) / (z-1)(z+0.24)] \text{ Now,}$$

type the following MATLAB code:

```
close all n
=[0.3678 0.2644];
d= [1 -0.76 -0.24];
s=tf(n,d,0.1); rlocus(s) [k,poles]=rlocfind(s)
%verification figure num= k*[0.3678
0.2644]; den=[1 0.368*k-0.76
0.2644*k-0.24]; sys =
tf(num,den,0.1) pzmap(sys)
p = pole(sys)
selected_point = -0.4810 + 0.8727i
k = 4.6782 poles =
-0.4803 + 0.8753i
-0.4803 - 0.8753i
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

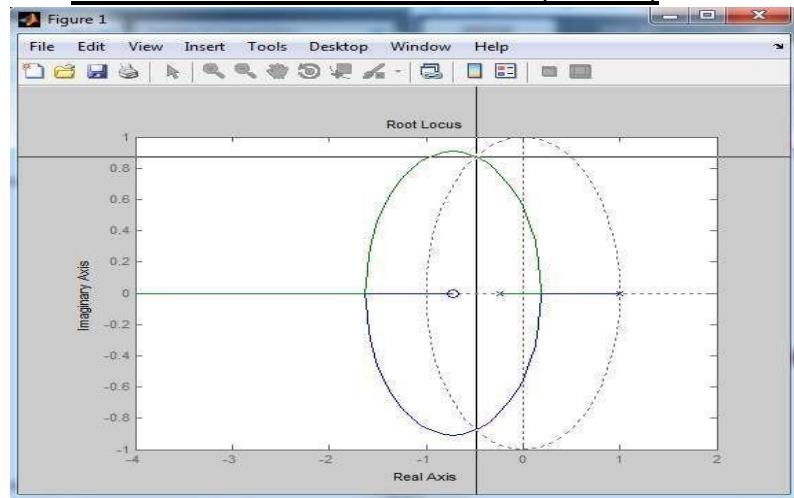


Figure-4.1 Root loci of given system

Verification

N sys

=

$$1.721 z + 1.237$$

$$z^2 + 0.9616 z + 0.9969$$

Sample time: 0.1 seconds Discrete-time transfer

function. p =



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

$$-0.4808 + 0.8751i$$

$$-0.4808 - 0.8751i$$

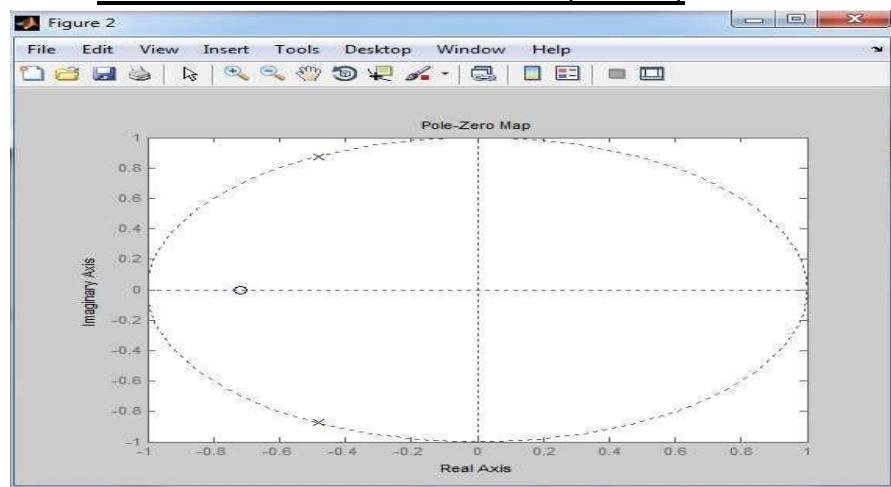


Figure-4.2 Location of poles and zeros in pzmap

Thus, the given system is stable for $0 < K < 4.6782$

Example 2: Plot the Root loci of following system

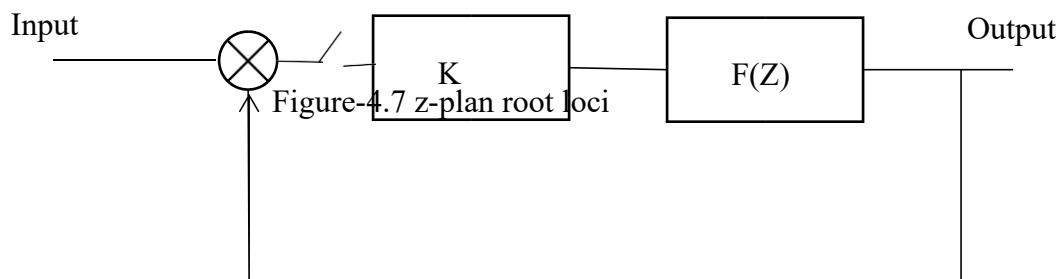


Figure-4.3 Block diagram of system

$$F(Z) = \frac{0.368(Z+0.717)}{(Z-1)(Z-0.368)}$$

MATLAB Code:

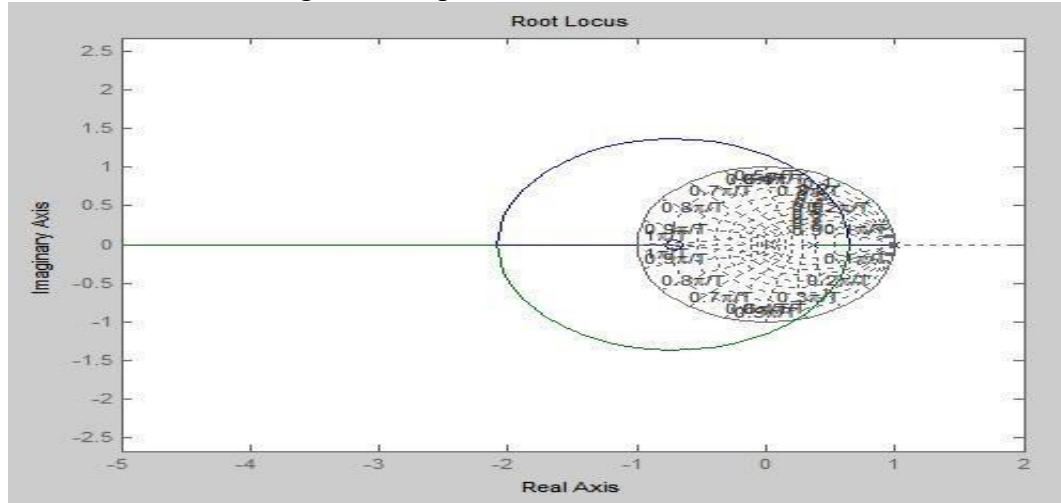
```
num=[0.368  
0.368*0.717]; den=[1 -  
1.368 0.368];  
sysd=tf(num,den,-1);  
rlocus(sysd); zgrid;  
axis('equal');
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

Figure-4.4 z-plan root loc



Stability and Transient Response:

For continuous systems, we know that certain behaviors result from different pole locations in the s- plane. For instance, a system is unstable when any pole is located to the right of the imaginary axis. For discrete systems, we can analyze the system behaviors from different pole

$$z = e^{sT}$$

locations in the z-plane. The characteristics in the z-plane can be related to those in the s-plane by the following expression.

- T = sampling time (sec/sample)
- s = location in the s-plane
- z = location in the z-plane

The figure below shows the mapping of lines of constant damping ratio (ζ) and natural frequency (ω_n) from the s-plane to the z-plane using the expression shown above



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

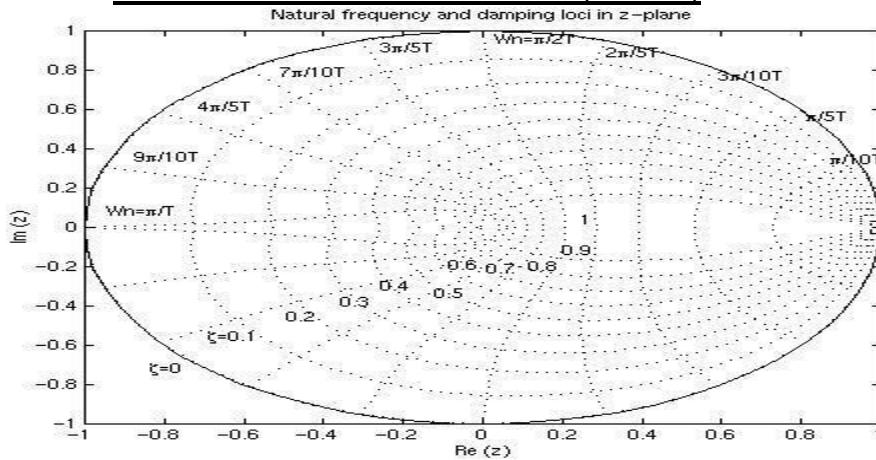


Figure-4.5 z-plane root loci

You may have noticed that in the z-plane the stability boundary is no longer the imaginary axis, but rather is the circle of radius one centered at the origin, $|z| = 1$, referred to as the **unit circle**. A discrete system is stable when all poles are located inside the unit circle and unstable when any pole is located outside the circle.

For analyzing the transient response from pole locations in the z-plane, the following three equations used in continuous system designs are still applicable.

where,

- ζ = damping ratio
- ω_n = natural frequency (rad/sec)
- T_s = 1% settling time
- T_r = 10-90% rise time
- M_p = maximum overshoot

$$\zeta \omega_n \geq \frac{4.6}{T_s}$$

$$\omega_n \geq \frac{1.8}{T_r}$$

$$\zeta = \frac{-\ln(M_p)}{\sqrt{\pi^2 + \ln^2(M_p)}}$$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

Example 3: Suppose we have the following discrete transfer function

$$\frac{Y(z)}{F(z)} = \frac{1}{z^2 - 0.3z + 0.5}$$

Create a new m-file and enter the following commands. Running this m-file in the command window gives you the following plot with the lines of constant damping ratio and natural frequency.

MATLAB Code:

```
numDz = 1; denDz =  
[1 -0.3 0.5];  
sys = tf(numDz,denDz,-1); % the -1 indicates that the sample time is  
undetermined  
pzmap(sys) axis([-1 1 -1 1])  
zgrid
```

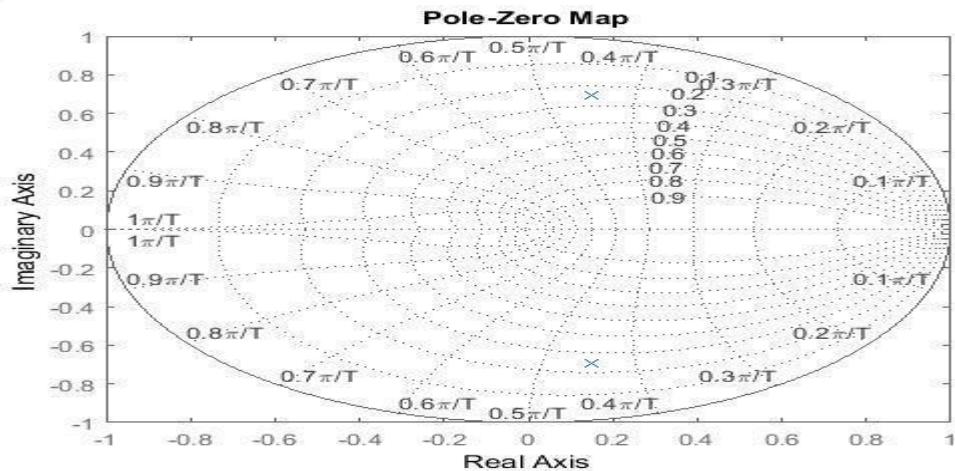


Figure-4.6 z-plan root loci

From this plot, we see the poles are located approximately at a natural frequency of $9\pi/20T$ (rad/sample) and a damping ratio of 0.25. Assuming that we have a sampling time of 1/20 sec (which leads to $\omega_n = 28.2$ rad/sec) we can determine that this system should have a rise time of 0.06 sec, a settling time of 0.65 sec, and a maximum percent overshoot of 45% (0.45 times more than the steady-state value). Let's obtain the step response and see if these are correct. Add the following commands to the above m-file and rerun it in the command window. You should obtain the following step response.

MATLAB Code:

```
sys = tf(numDz, denDz, 1/20);
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

```
step(sys,2.5);
```

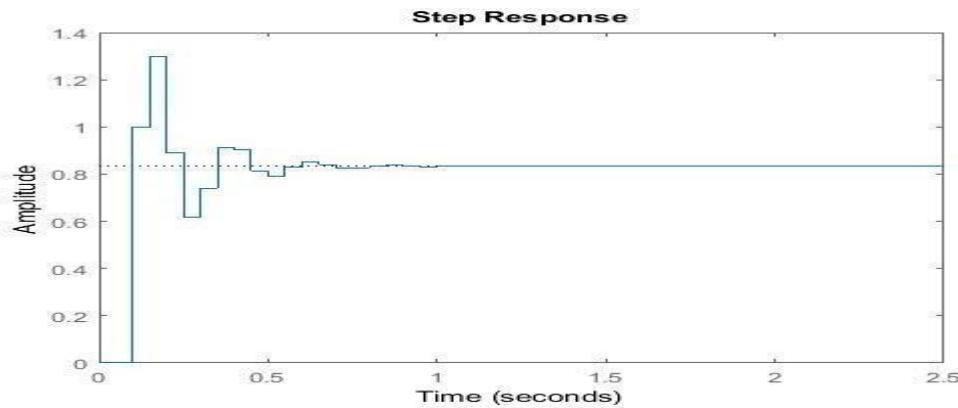


Figure-4.7 Step response of discrete root loci system

As we can see from the plot, the rise time, settling time and overshoot came out to be what we expected. This demonstrates how we can use the locations of poles and the above three equations to analyze the transient response of a discrete system

LAB TASK(S):

- Given, $G(z) = K[(z+1) / (z^2 - 1.2z + 0.2)]$ & $H(z) = 1$. Using root locus, determine the range of K for which system becomes stable, with the help of MATLAB. Note: Attach code & results with your handout.

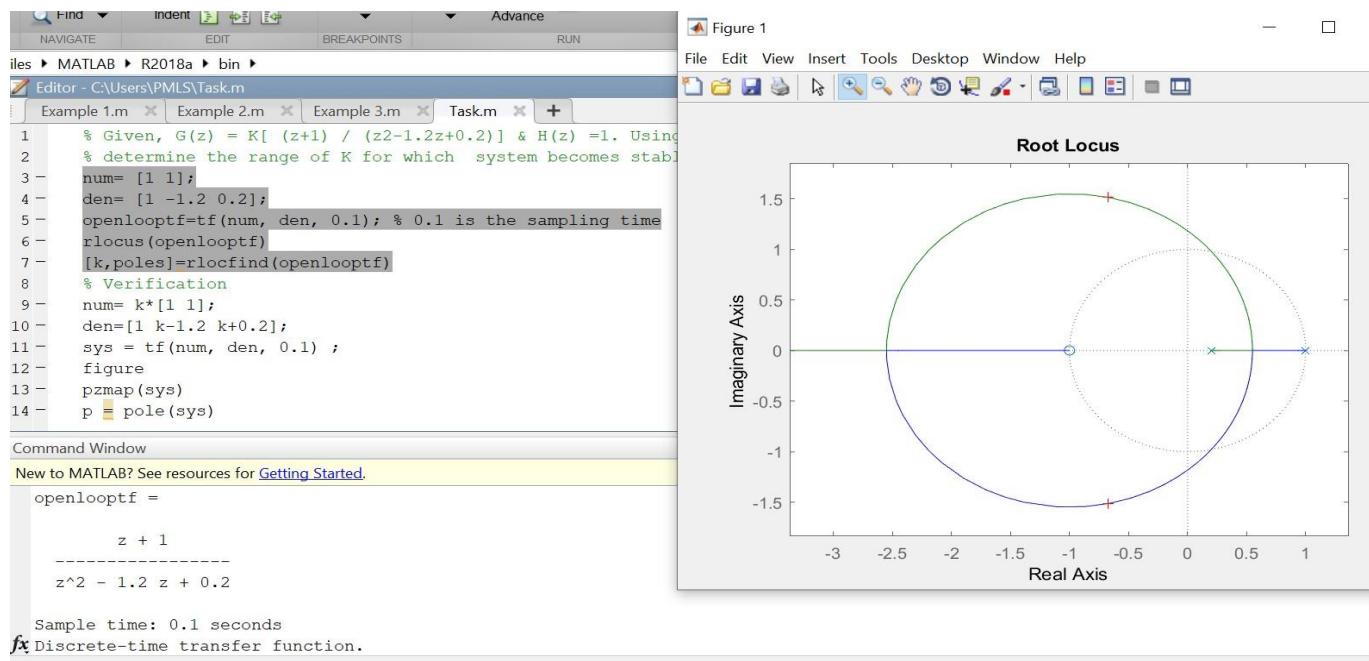
MATLAB Code:



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

```
1 % Given, G(z) = K[ (z+1) / (z^2-1.2z+0.2) ] & H(z) =1. Using root locus,
2 % determine the range of K for which system becomes stable, with the help of MATLAB.
3 num= [1 1];
4 den= [1 -1.2 0.2];
5 openlooptf=tf(num, den, 0.1); % 0.1 is the sampling time
6 rlocus(openlooptf)
7 [k,poles]=rlocfind(openlooptf)
8 % Verification
9 num= k*[1 1];
10 den=[1 k-1.2 k+0.2];
11 sys = tf(num,den,0.1)
12 figure;
13 pzmap(sys)
14 p = pole(sys)|
```





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

```
% determine the range of K for
num= [1 1];
den= [1 -1.2 0.2];
openloopf=tf(num, den, 0.1);
rlocus(openloopf)
[k,poles]=rlocfind(openloopf)
% Verification
num= k*[1 1];
den=[1 k-1.2 k+0.2];
sys = tf(num, den, 0.1) ;
figure
pzmap(sys)
p = pole(sys)

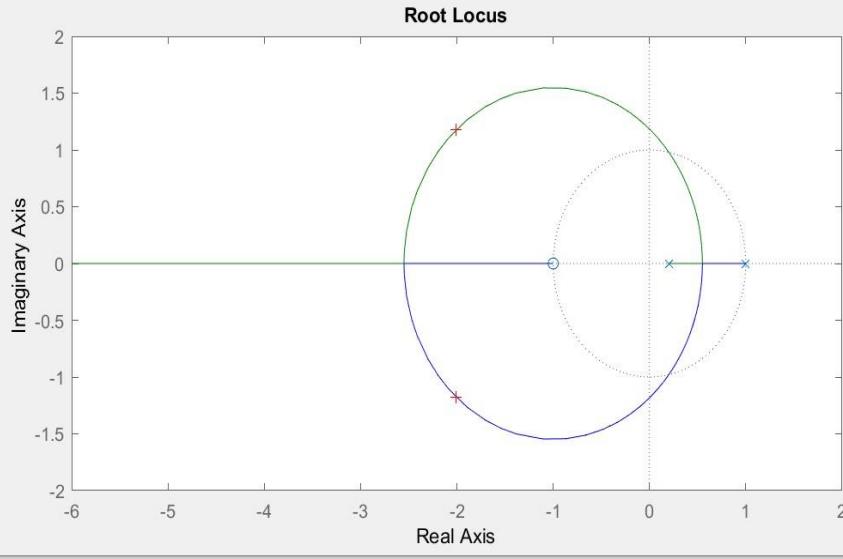
mmand Window
w to MATLAB? See resources for Getting Started.
```

k =

5.2119

poles =

-2.0060 + 1.1782i
-2.0060 - 1.1782i



cprint

In 7

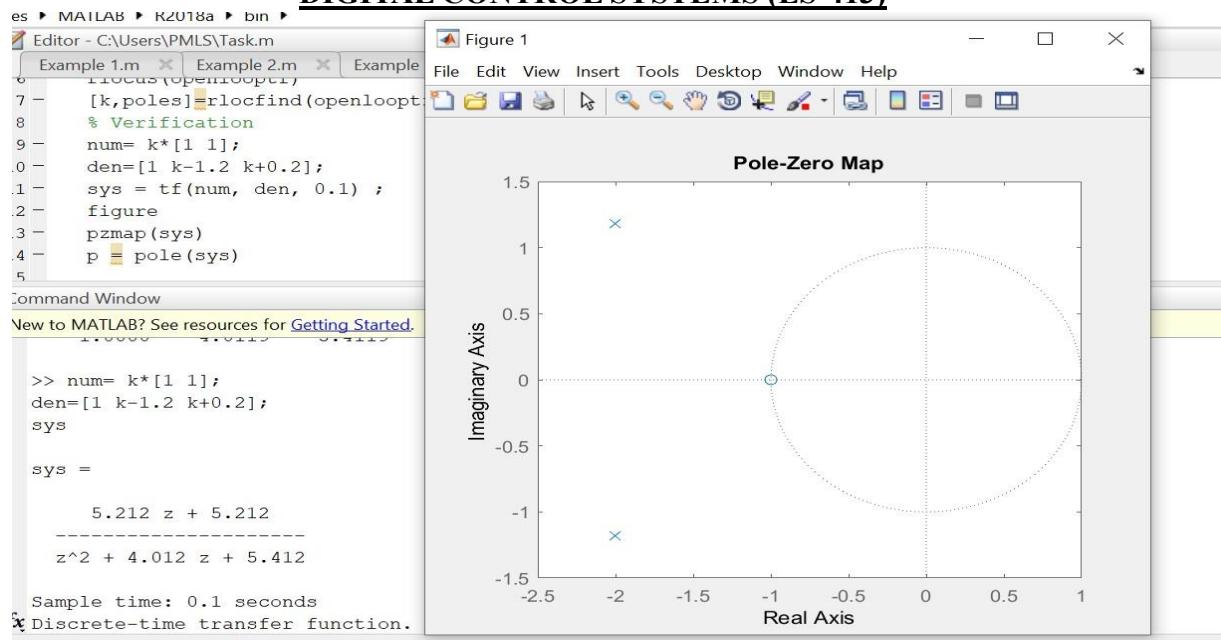
selected_point = -2.2275 + 1.5579i
k = 5.2119 poles = -2.0060 +
1.1782i
-2.0060 - 1.1782i

Verification:



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)



```
num= k*[1 1]; den=[1
k-1.2 k+0.2]; sys =
5.212 z + 5.212 -----
----- z^2 +
4.012 z + 5.412
```

Sample time: 0.1 seconds
Discrete-time transfer
function. p = -2.0060 +
1.1782i
-2.0060 - 1.1782i

From Code Execution k = 5.2119
Thus the given system is stable between 0 < K < 5.2119



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING

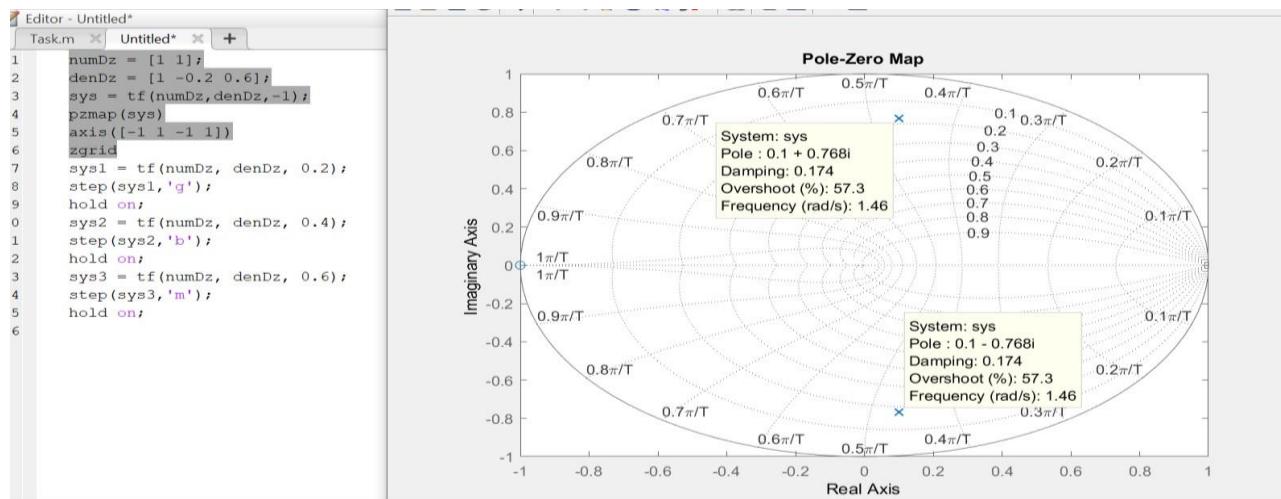
DIGITAL CONTROL SYSTEMS (ES-413)

2. Given, $G(z) = K[(z+1) / (z^2 - 0.2z + 0.6)]$ & $H(z) = 1$. Plot pz-map, also step response with different time samples, with the help of MATLAB.

Matlab code:

```
1 numDz = [1 1];
2 denDz = [1 -0.2 0.6];
3 sys = tf(numDz,denDz,-1);
4 pzmap(sys)
5 axis([-1 1 -1 1])
6 zgrid
7 sys1 = tf(numDz, denDz, 0.2);
8 step(sys1,'g');
9 hold on;
10 sys2 = tf(numDz, denDz, 0.4);
11 step(sys2,'b');
12 hold on;
13 sys3 = tf(numDz, denDz, 0.6);
14 step(sys3,'m');
15 hold on;
16
```

denDz	[1,-0.2000,0.6...
numDz	[1,1]
sys	1x1 tf
sys1	1x1 tf
sys2	1x1 tf
sys3	1x1 tf
sys4	1x1 tf

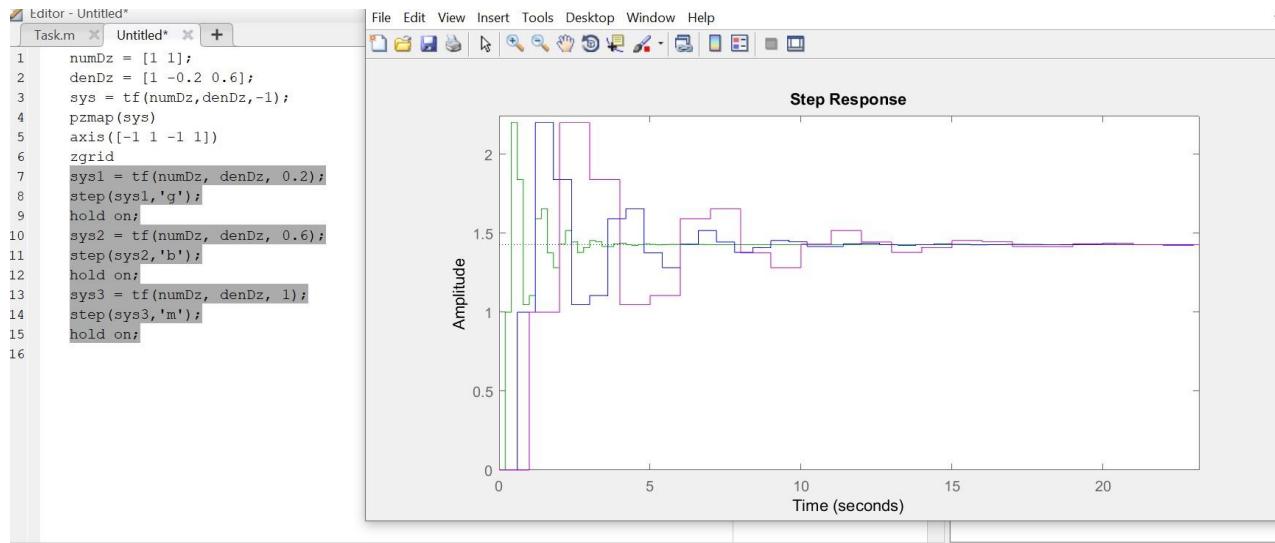




MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

From the pz-map, it is shown that damping ratio is 0.17, and a maximum percent overshoot of 57% (0.57 times more than the steady-state value).



From plot of step responses at different time samples, it is observed that by decreasing sampling time, system comes at stability quickly.

REFERENCE(S)

1. <https://www.mathworks.com>
2. Matlab: A Practical Introduction to Programming and Problem Solving by Stormy Attaway.

FINAL CHECK LIST

1. Before you leave, properly save your work and close the software tool.
2. Submit your completed laboratory work, before the next laboratory session.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO

DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

LAB # 05: MATHEMATICALLY MODELING OF ELECTRICAL SYSTEM AND DC CIRCUIT ANALYSIS USING MATLAB

Name: KARAN

Roll # 20ES062

Signature of Lab Tutor:

Date: _____

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	Mathematical modeling of electrical system and DC circuit analysis using MATLAB	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

Equipment Required: PC with MATLAB software

Approximate Time Required: Two to three hours

Introduction

There are various types of physical systems, namely we have:

1. Mechanical systems
2. Electrical systems
3. Electronic systems
4. Thermal systems
5. Hydraulic systems
6. Chemical systems

First we need to understand why do we need to model these systems in the first place? Mathematical modeling of a control system is the process of drawing the block diagrams for these types of systems in order to determine their performance and transfer functions.

In an electrical type of system we have three variables

1. Voltage which is represented by 'V'.
2. Current which is represented by 'I'.
3. Charge which is represented by 'Q'.

And also we have three parameters which are active and passive components:

1. Resistance which is represented by 'R'.
2. Capacitance which is represented by 'C'.
3. Inductance which is represented by 'L'.

System response

The natural overdamped RLC series circuit response occurs for the condition $\alpha^2 > w_o^2$.

Then the roots s_1 and s_2 are real, unequal, and negative numbers

The natural critical-damped RLC series circuit response occurs for the condition $\alpha^2 = w_o^2$. Then the roots s_1 and s_2 are real ($-\alpha$) and repeated, resulting in a solution of the form $i(t) = A_1 e^{-\alpha t} + A_2 t e^{-\alpha t}$, where the constants A_1 and A_2 can be evaluated from the system's initial conditions.

The natural underdamped RLC series circuit response occurs for the condition $\alpha^2 < w_o^2$. Then the roots s_1 and s_2 are a complex conjugate pair, resulting in a solution of the form



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

The transient analysis of a circuit that contains more than one loop can be described by a set of differential equations. The set of differential equations may consist of either loop or node equations where the unknowns may be either the loop cur-rents or the node voltages. **Example1:**

Using two loop equations and one node equation, and the matrix operations $I = \text{inv}(R) * V$ (where R is the [equivalent] impedance matrix of the network and V the voltage vector), solve for the three branch currents— I_1 , I_2 , and I_3 —shown in the circuit diagram of figure1. The node and the two loop equations are shown as follows:

$$\text{Node A: } -I_1 + I_2 + I_3 = 0$$

$$\text{Loop\# 1: } 10 * I_1 + 5 * I_2 = 10$$

$$\text{Loop\# 2: } -5 * I_2 + (10 + 20) * I_3 = 0$$

The preceding equations in matrix form is given by

$$\begin{bmatrix} -1 & 1 & 1 \\ 10 & 5 & 0 \\ 0 & -5 & 30 \end{bmatrix} * \begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 10 \\ 0 \end{bmatrix}$$

Then the 3×3 matrix becomes the resistance matrix R , and the voltage V is given by the column vector $[0 \ 10 \ 0]^T$ as illustrated by the following matrix equation:

$$[R] * [I] [V]$$

then

$$I = \text{inv}(R) * V$$

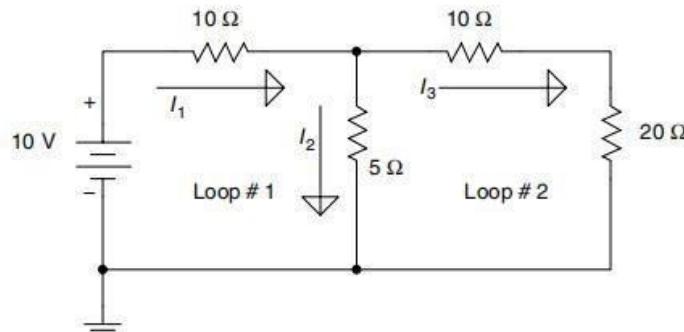


Figure5.1:Network Example



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

MATLAB Solution

```
>> R = [-1 1 1;10 5 0;0 -5 30];
>> V = [0;10;0];
>> I = inv(R)*V; % Solves for the loop currents
>> Result = [I(1) I(2) I(3)];
>> % Results are printed
>> disp('*****R E S U L T S *****');
>> disp('*****');
>> disp('The currents I1, I2, I3 are:');
>> disp(Result)
>> disp('    amp.    amp.    Amp.');
>> disp('*****');
```

*****R E S U L T S*****

The currents I1, I2, I3 are:
0.7000 0.6000 0.1000
amp. amp. amp.

Example2: For the circuit diagram shown in Figure 2.60

7. Write the three system mesh (loop) equations 2. Arrange the result of part 1 into a matrix equation
3. Using MATLAB, solve for the three loop currents (I_1 , I_2 , and I_3)

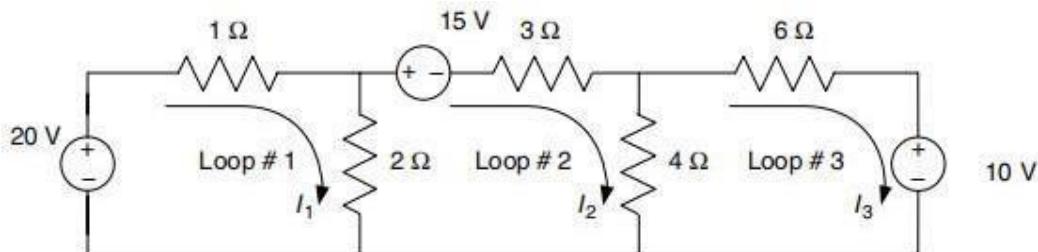


Figure5.2: Network Example

ANALYTICAL Solution

The loop equations are

$$\begin{aligned} \text{Loop 1: } & 20 = 3I_1 - 2I_2 + 0I_3 \\ \text{Loop 2: } & -15 = -2I_1 + 9I_2 - 4I_3 \\ \text{Loop 3: } & -10 = 0I_1 - 4I_2 + 10I_3 \end{aligned}$$

The resulting matrix equation is



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

$$\begin{bmatrix} 3 & -2 & 0 \\ -2 & 9 & -4 \\ 0 & -4 & 10 \end{bmatrix} * \begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} 20 \\ -15 \\ -10 \end{bmatrix}$$

MATLAB Solution

```
>> R = [3 -2 0; -2 9 -4; 0 -4 10];
>> V = [20; -15; -10];
>> I = inv(R)*V;
>> Result = [I(1) I(2) I(3)];
>> disp('*****')
>> disp('The loop currents I1, I2 and I3(in amp) are:');
>> disp(Result);
>> disp('*****')

*****
The loop currents I1, I2 and I3 (in amp) are:
6.0440
-0.9341
-1.3736
*****
```

Example3: The switch shown in the circuit diagram in Figure:5.3 has been in position a for a long time. At $t = 0$, the switch is moved to position b where it remains for 2 s and then moves back to position a, where it remains indefinitely.

Obtain analytical expressions for $v_c(t)$ and $i_c(t)$ for all $t \geq 0$. Use MATLAB to obtain plots over the range $0 \leq t \leq 10$ s of

- A. The voltage $v_c(t)$ versus t
- B. The current $i_c(t)$ versus t



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

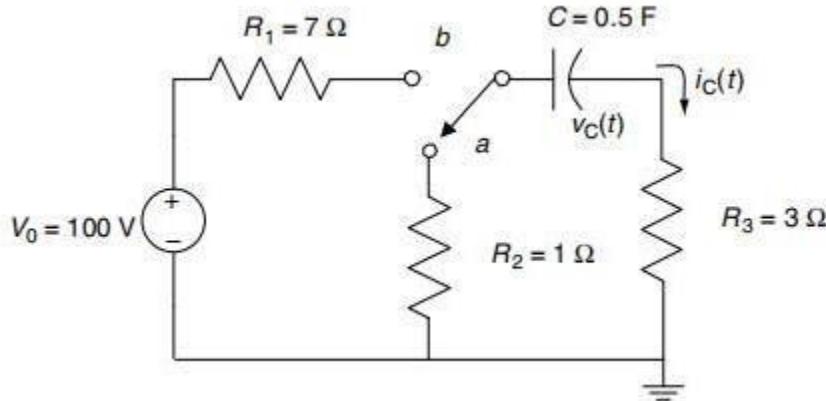


Figure 5.3-Network example

ANALYTICAL Solution

For $t < 0$, the switch is in position a . Therefore,

$$v_c(t) = 0$$

and

$$i_c(t) = 0$$

For $0 < t < (t_1 = 2)$ seconds, the switch is in position b . Then,

$$v_c(t) = V_0(1 - e^{-t/\tau_1})$$

and

$$i_c(t) = \frac{V_0 - v_c(t)}{R_1 + R_3}$$

where $\tau_1 = (R_1 + R_3) * C = 10 \Omega * 0.5 F = 5 s$.

At time $t = t_1 = 2$ s, the voltage across the capacitor is

$$v_c(t_1) = V_{c\max} = V_0(1 - e^{-2/\tau_1})$$

For $t > (t_1 = 2$ s), the switch is back in position a . Then



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

$$v_C(t) = V_{C_{\max}} * e^{-(t-t_1)/\tau_2}$$

and

$$i_C(t) = \frac{-v_C(t)}{R_3 + R_2}$$

where $\tau_2 = (R_2 + R_3) * C = 4 \Omega * 0.5 F = 2 s$.

MATLAB Solution

```
>> V = 100; R1 = 7; R2 = 1; R3=3; C = 0.5; % circuit elements
>> tau1 = (R1+R3)*C % time constant #1
tau1 =
    5
>> tau2 = (R2+R3)*C % time constant #2
tau2 =
    2
>> for k =1:40
    t(k) = k/20; % 0 < t < 2
    v(k) = V*(1-exp(-t(k)/tau1));
    i(k) = (V-v(k))/(R1+R2);
end
>> Vmax = v(40)
Vmax =
    32.9680
>> for k=41:200
    t(k)=k/20; % 2 < t < 10
    v(k)=Vmax*exp(-(t(k)-t(40))/tau2);
    t(40)=40/20=2sec=t1
    i(k)= -v(k)/(R2+R3);
end
% plot the Voltage VC(t)
>> subplot(2,1,1)
>> plot(t,v)
>> axis([0 10 0 40]);
>> title ('Transients in the circuit of Fig 2.62')
>> xlabel('time in sec'), ylabel('Voltage vc(t)')
>> grid on
% plot the current IC (t)
>> subplot(2,1,2)
>> plot(t,i)
>> axis([0 10 -13 15]);
>> xlabel('time in sec'), ylabel('Current ic(t)')
>> grid on
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

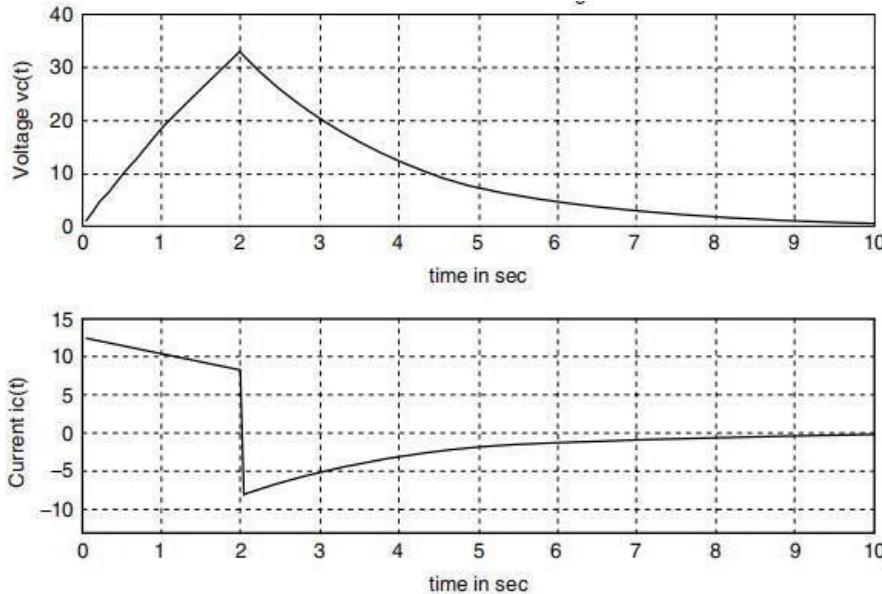


Figure 5.4: Transient plots

Example 4: Steady-state conditions exist in the network shown in Figure

5.5 at $t = 0$ when the $V_1 = 120$ V source is connected to the RCL parallel circuit. At $t = 0$, the switch moves downward, and the source $V_1 = 120$ V and resistor $R = 10 \Omega$ are disconnected from the parallel (RLC) structure.

Analyze the transient response ($t > 0$) of the source-free parallel RLC circuit, for each of the following values of R , $R = 3, 9$, and 72Ω .

1. Determine the analytical response $v_C(t)$ for each value of R , for $t \geq 0$
2. Create the script file `transient_RLC_parallel` that returns the MATLAB solutions of part 1 and its corresponding voltage plots
4. Compare the MATLAB solutions of part 2 with the analytical solutions of part 1

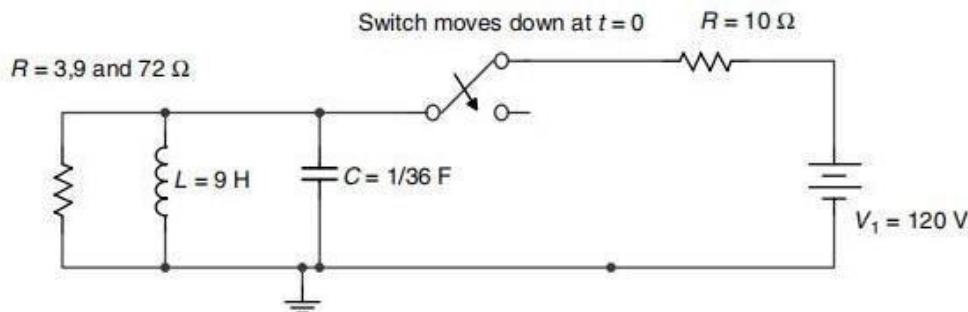


Figure 5.5: Network of example



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

ANALYTICAL Solution

From the circuit diagram of Figure 5.5 for $t \leq 0$, the initial conditions are $v_C(0) = 0$ and $i_L(0) = 120/10 = 12 \text{ A}$, and

$$C \frac{dv_C(t)}{dt} \Big|_{t=0} = i_C(t=0) = i_C(0) = i_L(0) + i_R(0)$$

then

$$\frac{dv_C(t)}{dt} \Big|_{t=0} = 36i_C(0) = 36(12 + 0) = 432 \text{ V/s}$$

Recall that the node equation is

$$\frac{d^2v_C(t)}{dt^2} + \frac{1}{RC} \frac{dv_C(t)}{dt} + \frac{1}{CL} v_C(t) = 0 \quad \text{for } t \geq 0$$

For $R = 3 \Omega$, the resonant frequency is

$$w_0 = \frac{1}{\sqrt{LC}} = 2 \text{ rad/s}$$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

The neper frequency is

$$\alpha = \frac{1}{2RC} = 6 \text{ Hz}$$

The complex frequencies are s_1 and s_2 , given by

$$s_{1,2} = -\alpha \pm \sqrt{\alpha^2 - w_0^2}$$

then $0 > \sqrt{\alpha^2 - w_0^2}$ and is clearly the overdamped case.

Then the solution $v_C(t)$ is of the form $v_C(t) = A_1 e^{-s_1 t} + A_2 e^{-s_2 t}$
For $R = 9 \Omega$, the resonant frequency is

$$w_0 = \frac{1}{\sqrt{LC}} = 2 \text{ rad/s}$$

The neper frequency is

$$\alpha = \frac{1}{2RC} = 2 \text{ Hz}$$

Then s_1 and s_2 are negative and repeated frequencies, which are given by



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

$$s_{1,2} = -\alpha \pm \sqrt{\alpha^2 - w_0^2} = -\alpha$$

Since $0 = \sqrt{\alpha^2 - w_0^2}$, the case is critical damped.

Then the solution $v_C(t)$ is of the form $v_C(t) = A_1 e^{-\alpha t} + A_2 t e^{-\alpha t}$

For $R = 72 \Omega$, the resonant frequency is

$$w_0 = \frac{1}{\sqrt{LC}} = 2 \text{ rad/s}$$

The neper frequency is

$$\alpha = \frac{1}{2RC} = \frac{1}{4} \text{ Hz}$$

The complex frequencies s_1 and s_2 are given by $s_{1,2} = -\alpha \pm j\sqrt{w_0^2 - \alpha^2}$ (complex conjugate), and is clearly the underdamped case. Then the solution $v_C(t)$ is of the form

$$v(t) = e^{-\alpha t} [A_1 \cos(w_d t) + A_2 \sin(w_d t)]$$

where $w_d = \sqrt{w_0^2 - \alpha^2}$.

Let us use MATLAB to obtain and plot the solutions just presented, and compare them with the analytical results.

```
MATLAB Solution
% Script file : transient_RLC_parallel
% source free RLC parallel circuit analysis
disp('*****')
disp('*****R E S U L T *****')
disp('Source free parallel RLC circuit')
disp('*****')
disp('Solution of diff. equation for R=3 ohms')
overdamped = dsolve('(1/36)*D2y+(1/3)*Dy+(1/9)*y=0','y(0) = 0','Dy(0)= 32','t')
figure(1)
ezplot(overdamped)
xlabel('time (sec)')
ylabel('voltage(volts)')
title('v(t) vs t (overdamped case; R=3 ohms)')
axis([0 3 0 40]);
disp('*****')
figure(2)
disp('Solution of diff. equation for R=9 ohms')
criticaldamped = dsolve('(1/36)*D2y+(1/9)*Dy+(1/9)*y=0','y(0)=0','Dy(0)=432','t')
simple_crit=simple(criticaldamped);
ezplot(criticaldamped)
xlabel('time (sec)')
ylabel('voltage(volts)')
title('v(t) vs t (criticaldamped case; R=9 ohms)')
axis([0 4 0 90]);
disp('*****')
disp('Solution of diff. equation for R=72 ohms')
underdamped = dsolve('(1/36)*D2y+(1/72)*Dy+(1/9)*y=0','y(0)=0','Dy(0)=432','t')
figure(3)
ezplot(underdamped)
xlabel('time (sec)'); ylabel('voltage(volts)')
title('v(t) vs t (underdamped case; R=72 ohms)')
axis([0 6 -150 200]);
disp('*****')
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Results are:

```
>> transient _ RLC _ parallel
*****
***** R E S U L T *****
Source free parallel RLC circuit
*****
Solution of diff. equation for R=3 ohms overdamped =
27*2^(1/2)*exp(2*(-3+2*2^(1/2))*t)-27*2^(1/2)*exp(-2*(
3+2*2^(1/2))*t) ***** Solution of
diff. equation for R=9 ohms criticaldamped =432*exp(-
2*t)*t*****
***** Solution of diff. equation for R=72 ohms
underdamped =
576/7*7^(1/2)*exp(-1/4*t)*sin(3/4*7^(1/2)*t)
*****
```

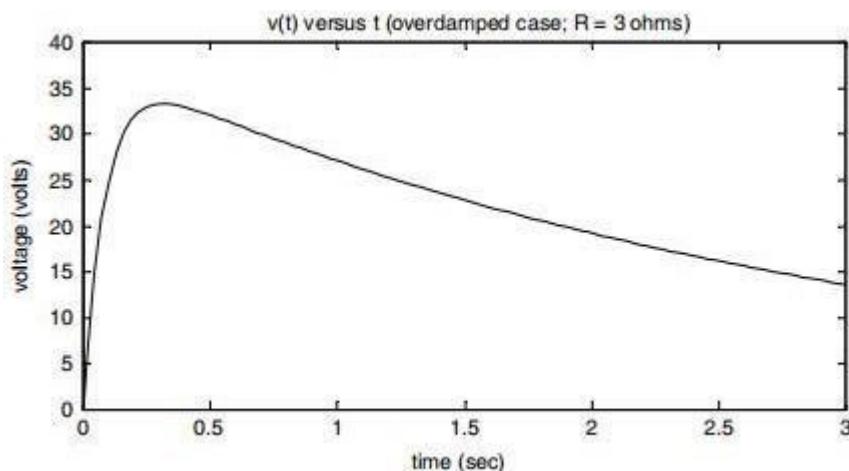


Figure5.6: Transient over damped plot ($t > 0$) of the source-free parallel RLC



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

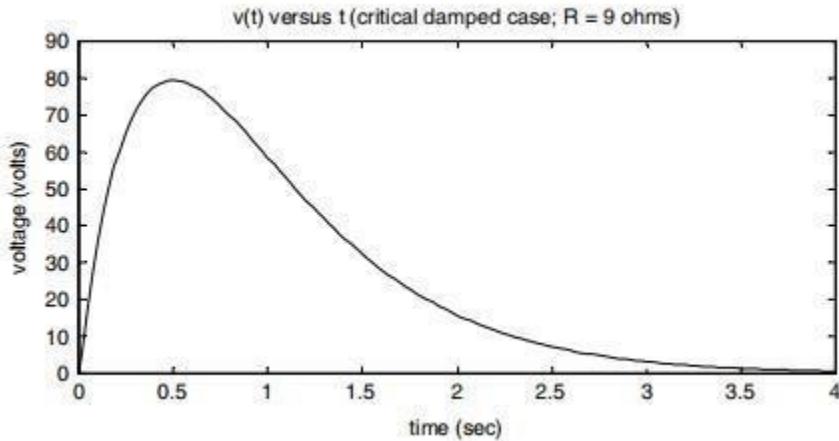


Figure 5.7: Transient critical damped plot ($t > 0$) of the source-free parallel RLC

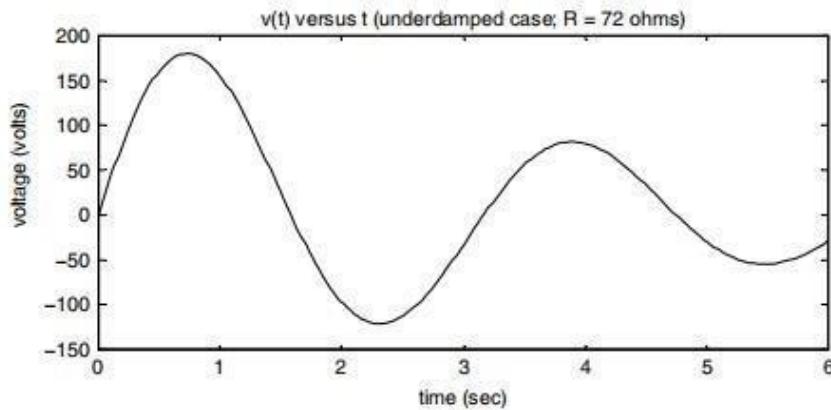


Figure 5.8: Transient under damped plot ($t > 0$) of the source-free parallel RLC

Observe that by changing the value of R ($R = 3, 9$, and 72Ω), the three solutions for the secondorder differential equation—over, critical, and under damped—are obtained. Observe that the analytical solutions completely agree with the MATLAB solutions

Example5: Steady-state conditions exist in the circuit of Figure 5.9 for $t \leq 0$, while the voltage source $V_1 = 6$ V is connected to the network. At $t = 0^+$, the switch moves downward disconnecting the source while connecting R_1 to the rest of the circuit.

1. Obtain analytically the loop differential equation set and the initial conditions
2. Using the MATLAB symbolic solver, create the script file transient_2loops that returns the transient currents for each loop, and their respective plots, for $t \geq 0$
3. Also obtain simplify and *pretty* expressions for each of the transient loop currents of part 2.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

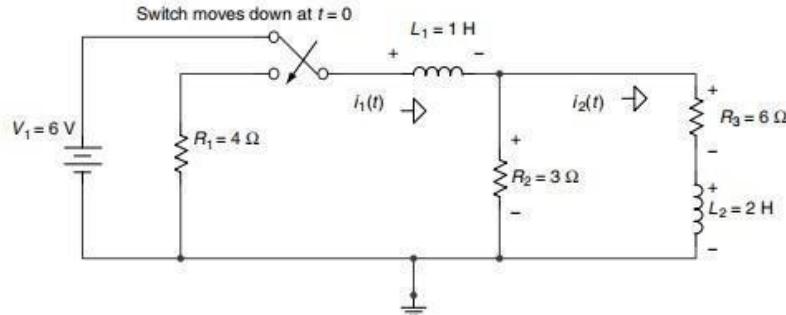


Figure 5.9: Network of example

ANALYTICAL Solution

The conditions at $t = 0$ are $i_1(0) = 3 \text{ A}$ and $i_2(0) = 2 \text{ A}$. Applying KVL to the two-mesh network of Figure 5.9 for $t \geq 0$, results in the following set of simultaneous differential equations:



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

$$(R_1 + R_2)i_1(t) + L_1 \frac{di_1(t)}{dt} - R_2 i_2(t) = 0$$

$$-R_2 i_1(t) + (R_3 + R_2)i_2(t) + L_2 \frac{di_2(t)}{dt} = 0$$

Replacing the elements by their values yields the following set of differential equations:

$$7i_1(t) + \frac{di_1(t)}{dt} - 3i_2(t) = 0$$

or

$$\frac{di_1(t)}{dt} = -7i_1(t) + 3i_2(t)$$

and

$$-3i_1(t) + 9i_2(t) + 2 \frac{di_2(t)}{dt} = 0$$

or

$$2 \frac{di_2(t)}{dt} = 3i_1(t) - 9i_2(t)$$

MATLAB Solution

```
% Script file : transient_2loops
% transient solutions for two loop network
disp('*****R E S U L T S*****')
[y1 y2] = dsolve('Dy1=-7*y1+3*y2,2*Dy2=3*y1- 9*y2','y1(0)=3','y2(0)=2','t');
figure(1)
ezplot (y1,[0 3])
axis([0 1.5 0 3.5])
xlabel ('time (sec)')
ylabel ('current i1(t) in amps')
title ('i1(t) vs t ')
figure(2)
ezplot(y2,[0 3])
axis([0 1.5 0 2.3])
xlabel('time (sec)')
ylabel('current i2(t) in amps')
title('i2(t) vs t ')
disp('The currents i1(t) and i2(t), in amps are :')
[y1 y2]
disp('*****')
disp('The simplify solution for i1(t) is:')
simplify(y1)
disp('*****')
disp('The simplify solution for i2(t) is :')
simplify(y2)
disp('*****')
disp('The pretty solution for i1(t) is :')
pretty(y1)
disp('*****')
disp('The pretty solution for i2(t) is :')
pretty(y2)
disp('*****')
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO

DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL CONTROL SYSTEMS (ES-413)

```
figure(3)
ezplot(y1)
hold on
ezplot(y2)
xlabel('time (sec)')
axis([0 1.0 0 3.5])
ylabel('currents [i1(t)&i2(t)] in amps')
title('i1(t) & i2(t) vs t ')

The script file transient_2loops is executed, and the results and plots (Figures 2.106 through 2.108) are as follows:

>> transient_2loops

*****
*****R E S U L T S*****
*****
Solution of the two diff. loop equations for t>0
*****
The currents i1(t) and i2(t), in amps are :
ans =
[ conj(-9/194*97^(1/2)*exp(-1/4*(23+97^(1/2))*t)+9/194*97^(1/2)*exp(1/4*(-23+97^(1/2))*t)+3/2*exp(1/4*(-23+97^(1/2))*t)+3/2*exp(-1/4*(23+97^(1/2))*t))]
[ conj(exp(1/4*(-23+97^(1/2))*t)-14/97*97^(1/2)*exp(-1/4*(23+97^(1/2))*t)+14/97*97^(1/2)*exp(1/4*(-23+97^(1/2))*t)+exp(-1/4*(23+97^(1/2))*t))
*****
The simplify solution for i1(t) is:
ans =
-9/194*97^(1/2)*exp(-1/4*(23+97^(1/2))*t)+9/194*97^(1/2)*exp(1/4*(-23+97^(1/2))*t)+3/2*exp(1/4*(-23+97^(1/2))*t)+3/2*exp(-1/4*(23+97^(1/2))*t)
*****
The simplify solution for i2(t) is :
ans =
exp(1/4*(-23+97^(1/2))*t)-14/97*97^(1/2)*exp(-1/4*(23+97^(1/2))*t)+14/97*97^(1/2)*exp(1/4*(-23+97^(1/2))*t)+exp(-1/4*(23+97^(1/2))*t)
*****
The pretty solution for i1(t) is :

$$-\frac{9}{194} \cdot 97^{\frac{1}{2}} \exp\left(-\frac{1}{4}(23 + 97^{\frac{1}{2}}) t\right) + \frac{9}{194} \cdot 97^{\frac{1}{2}} \exp\left(\frac{1}{4}(-23 + 97^{\frac{1}{2}}) t\right) + \frac{3}{2} \exp\left(\frac{1}{4}(-23 + 97^{\frac{1}{2}}) t\right) + \frac{3}{2} \exp\left(-\frac{1}{4}(23 + 97^{\frac{1}{2}}) t\right)$$

*****
The pretty solution for i2(t) is :

$$\exp\left(\frac{1}{4}(-23 + 97^{\frac{1}{2}}) t\right) - \frac{14}{97} \cdot 97^{\frac{1}{2}} \exp\left(-\frac{1}{4}(23 + 97^{\frac{1}{2}}) t\right) + \frac{14}{97} \cdot 97^{\frac{1}{2}} \exp\left(\frac{1}{4}(-23 + 97^{\frac{1}{2}}) t\right) + \exp\left(-\frac{1}{4}(23 + 97^{\frac{1}{2}}) t\right)$$

*****
```

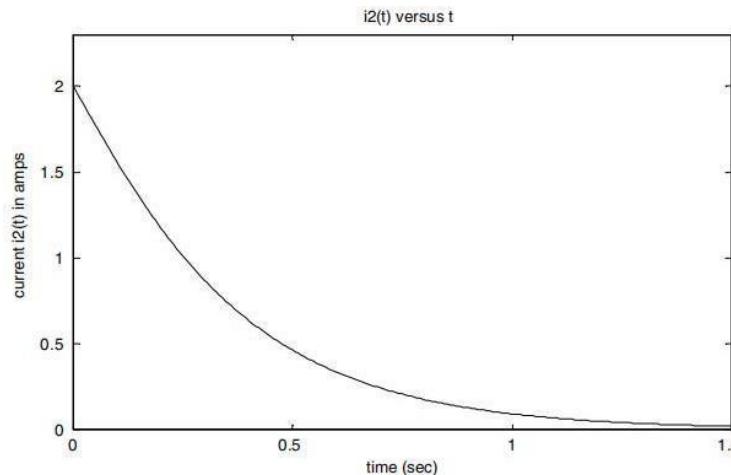


Figure 5.10: Plot of I1(t)



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO

DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

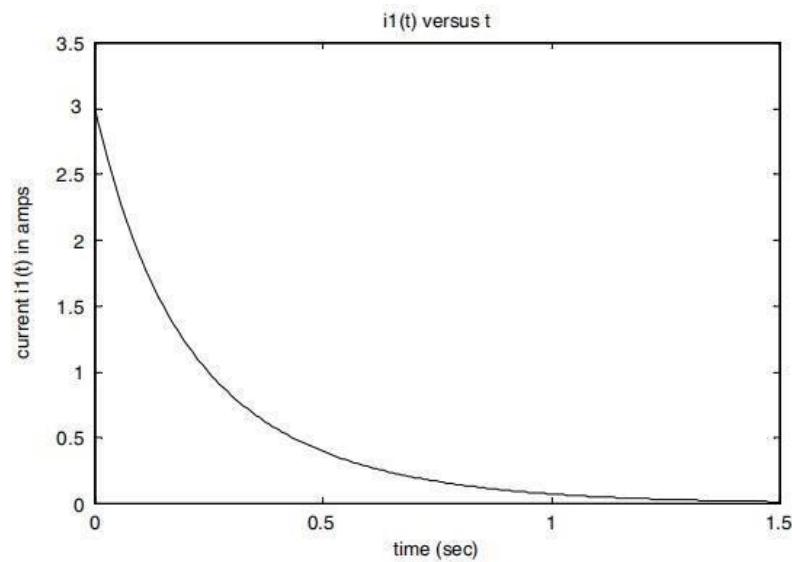


Figure 5.11: Plot of I2(t)



**DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**

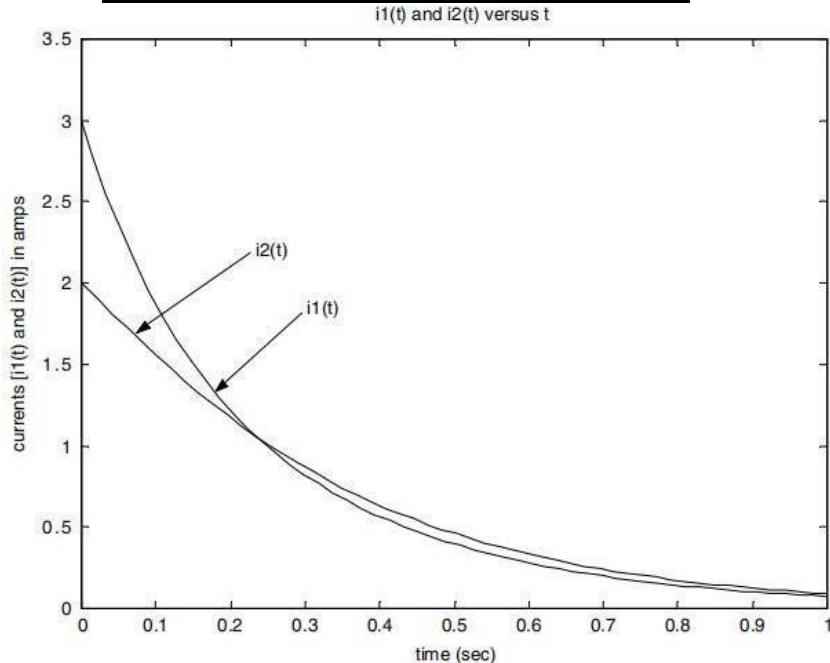


Figure 5.12: Plot of i₁(t) and i₂(t)

Review Exercise?

a) **What is Mathematical Modelling of any type of system?**

Mathematical modeling of a control system is the process of drawing the block diagrams for these types of systems in order to determine their performance and transfer functions.

b) **What is transient and steady state response of system?**

The transient response is the circuit's temporary response that will die out with time. The steady state response is the behavior of the circuit a long time after an external excitation is applied.

c) **Why do you get delay in system output response?**

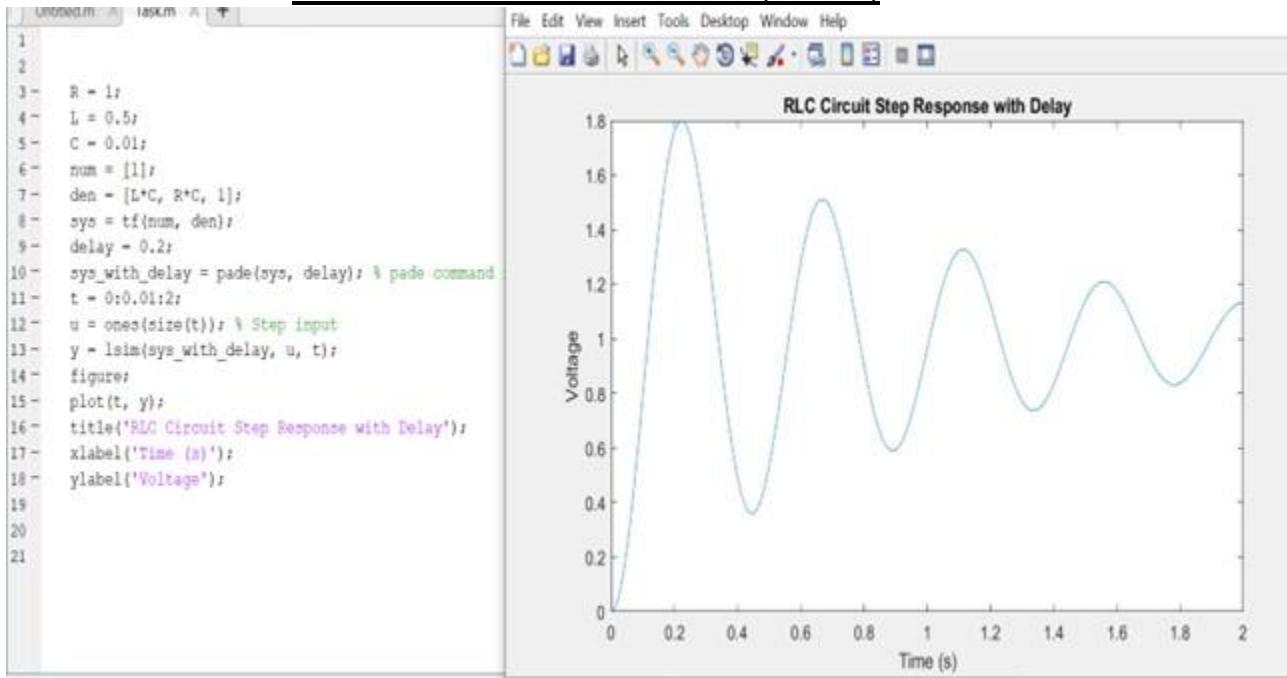
Delays can be attributed to acquiring information to make a decision, creating a control decision and/or executing the decisions. For example in the control system of an aircraft delays can be caused by actuators and sensors.

d) **Take any RLC circuit by your choice and write the program for that circuit in MATLAB.**

Include delay in system response and explain the behavior of system?



**DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**



e) Attach the screenshots of example1 to example5 (including codes and Output plots/results)

Example: 1



DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

The screenshot shows a MATLAB interface. On the left is the code editor window titled "Untitled2.m" containing the following MATLAB script:

```
1 R=[-1 1 1;10 5 0;0 -5 30];
2 V=[0;10;0];
3 I=inv(R)*V;
4 Result=[I(1) I(2) I(3)];
5 disp('THE CURRENT I1,I2,I3 ARE:');
6 disp(Result)
7 disp('amp. amp. Amp.');


```

To the right is the command window showing the output of the script:

```
>> Untitled2
THE CURRENT I1,I2,I3 ARE:
    0.7000      0.6000      0.1000
amp. amp. Amp.
>> Untitled2
THE CURRENT I1,I2,I3 ARE:
    0.7000      0.6000      0.1000
amp. amp. Amp.
fx >>
```

Example: 2

The screenshot shows a MATLAB interface. On the left is the code editor window titled "Untitled.m" containing the following MATLAB script:

```
1 R = [3 -2 0; -2 9 -4; 0 -4 10];
2 V = [20; -15; -10];
3 I = inv (R)*V;
4 Result = [I(1) I(2) I(3)];
5 disp('The loop currents I1, I2 and I3(in amp) are:');
6 disp(Result');
7
8
9
```

To the right is the command window showing the output of the script:

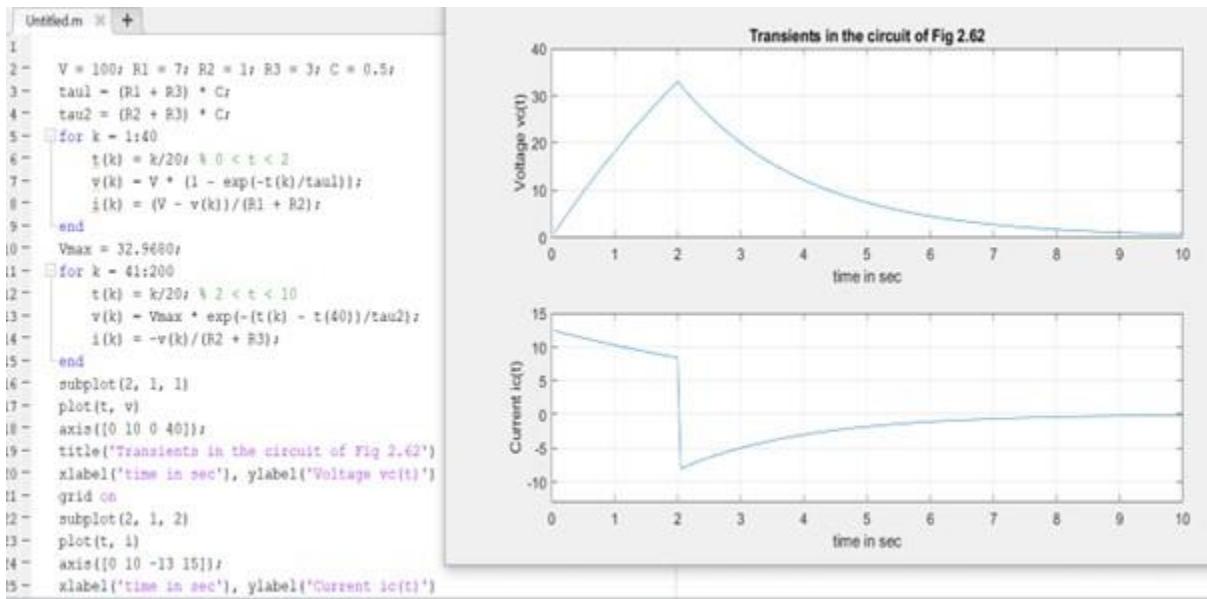
```
Command Window
New to MATLAB? See resources for Getting Started.
>> R = [3 -2 0; -2 9 -4; 0 -4 10];
V = [20; -15; -10];
I = inv (R)*V;
Result = [I(1) I(2) I(3)];
disp('The loop currents I1, I2 and I3(in amp) are:');
disp(Result');
The loop currents I1, I2 and I3(in amp) are:
    6.0440
   -0.9341
   -1.3736
fx >>
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO

DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL CONTROL SYSTEMS (ES-413)

Example:3



Example: 4



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO

DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL CONTROL SYSTEMS (ES-413)

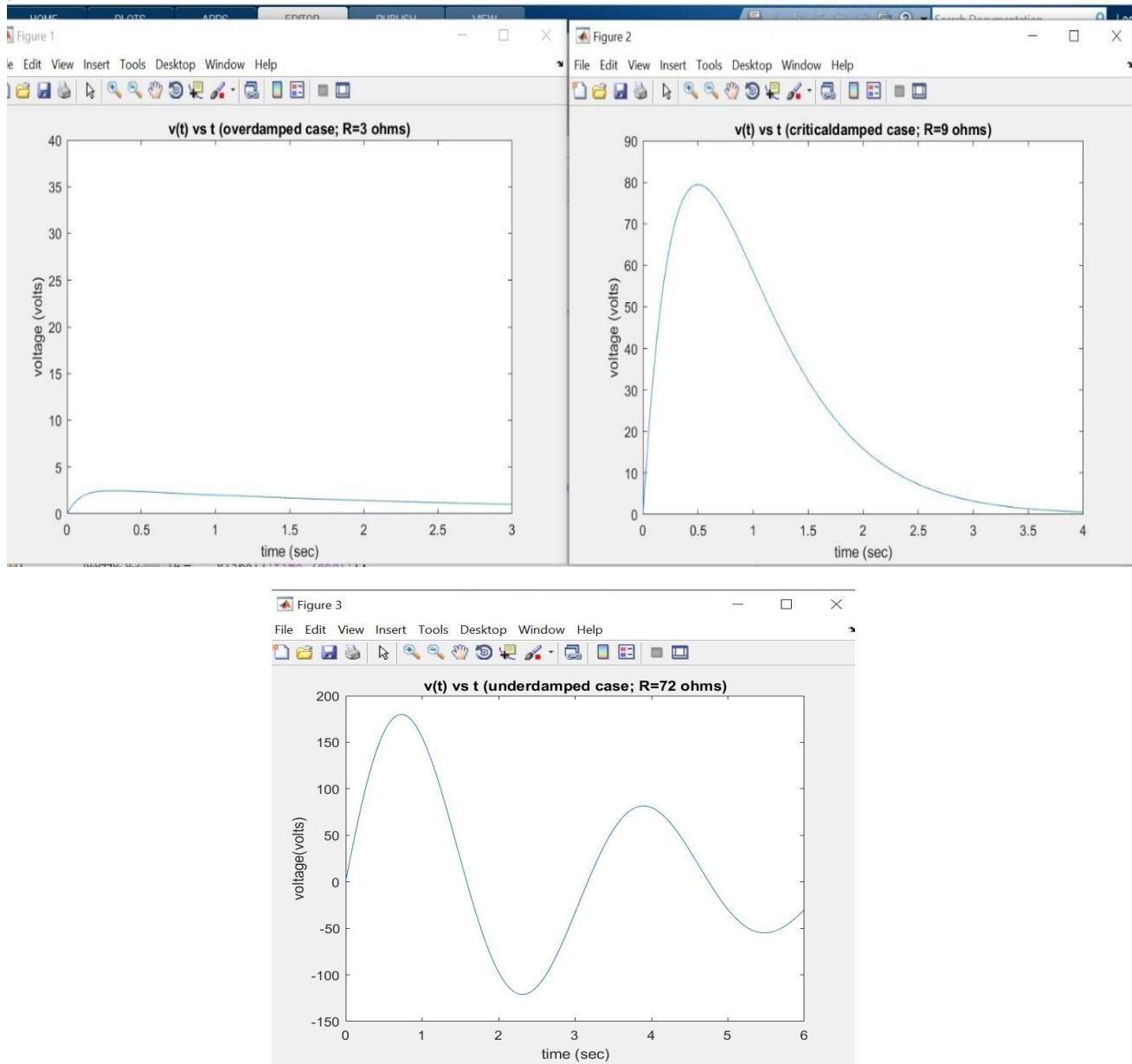
```
2 - disp('*****RESULTS*****')
3 - % Overdamped case (R = 3 ohms)
4 - disp('Source free parallel RLC circuit')
5 - disp('Solution of diff. equation for R=3 ohms')
6 - overdamped = dsolve('(1/36)*D2y+(1/3)*Dy+(1/9)*y=0', 'y(0)=0', 'Dy(0)=32', 't');
7 - figure(1)
8 - ezplot(overdamped)
9 - xlabel('time (sec)');
10 - ylabel('voltage (volts)');
11 - title('v(t) vs t (overdamped case; R=3 ohms)')
12 - axis([0 3 0 40]);
13 - % Critically damped case (R = 9 ohms)
14 - disp('Solution of diff. equation for R=9 ohms')
15 - criticaldamped = dsolve('(1/36)*D2y+(1/9)*Dy+(1/9)*y=0', 'y(0)=0', 'Dy(0)=432', 't');
16 - simple_crit = simplify(criticaldamped);
17 - figure(2)
18 - ezplot(criticaldamped)
19 - xlabel('time (sec)');
20 - ylabel('voltage (volts)');
21 - title('v(t) vs t (criticallydamped case; R=9 ohms)')
22 - axis([0 4 0 90]);
23 - % Underdamped case (R = 72 ohms)
24 - disp('Solution of diff. equation for R=72 ohms')
25 - underdamped = dsolve('(1/36)*D2y+(1/72)*Dy+(1/9)*y=0', 'y(0)=0', 'Dy(0)=432', 't');
26 - figure(3)
27 - ezplot(underdamped)
28 - xlabel('time (sec)');
29 - ylabel('voltage(volts)')
30 - title('v(t) vs t (underdamped case; R=72 ohms)')
31 - axis([0 6 -150 200]);
32
33
```

Command Window

script



DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

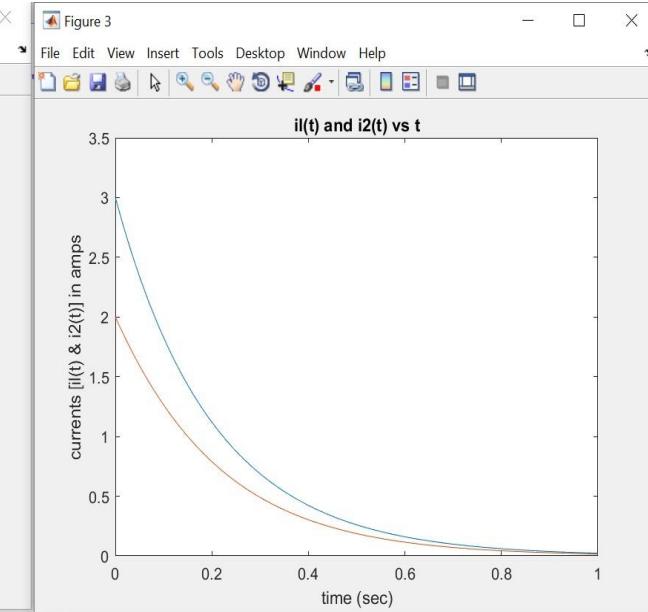
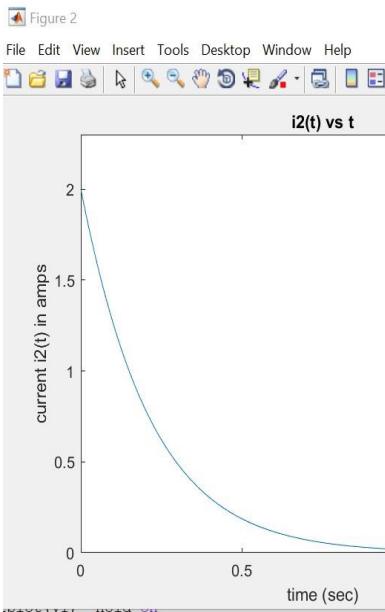
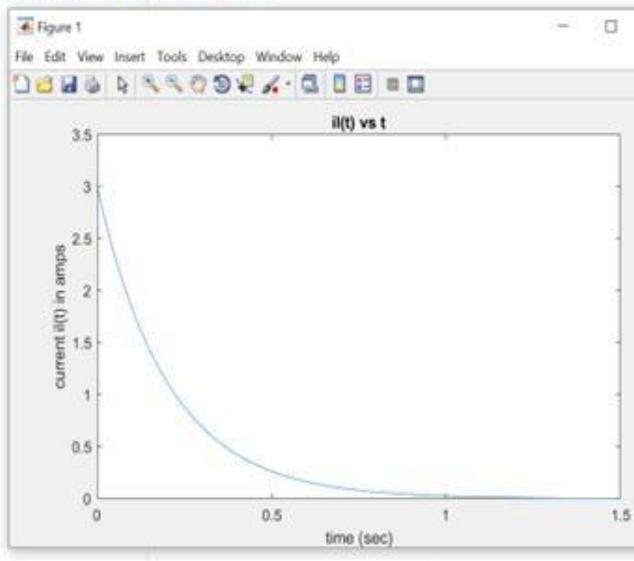


Example: 5



**DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**

```
2 - disp('**RESULTS**')
3 - [y1, y2] = dsolve('Dy1 = -7*y1 + 3*y2', 'Dy2 = 3*y1 - 7*y2', 'y1(0) = 3', 'y2(0) = 2', 't');
4 - figure(1)
5 - ezplot(y1, [0 3])
6 - axis([0 1.5 0 3.5])
7 - xlabel('time (sec)')
8 - ylabel('current i1(t) in amps')
9 - title('i1(t) vs t')
10 - figure(2)
11 - ezplot(y2, [0 3])
12 - axis([0 1.5 0 2.3])
13 - xlabel('time (sec)')
14 - ylabel('current i2(t) in amps')
15 - title('i2(t) vs t')
16 - disp('The currents i1(t) and i2(t) in amps are:')
17 - [y1 y2];
18 - disp('The simplified solution for i1(t) is:')
19 - simplify(y1)
20 - disp('The simplified solution for i2(t) is:')
21 - simplify(y2)
22 - disp('The pretty solution for i1(t) is:')
23 - pretty(y1)
24 - disp('The pretty solution for i2(t) is:')
25 - pretty(y2)
26 - figure(3)
27 - ezplot(y1); hold on
28 - ezplot(y2)
29 - xlabel('time (sec)')
30 - axis([0 1.0 0 3.5])
```





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

**LAB # 06: THE STEP RESPONSE OF DISCRETE TIME SYSTEM AND EFFECT OF
SAMPLING TIME ON SYSTEM RESPONSE**

Name: <u>KARAN</u>	Roll # <u>20ES062</u>
Date: _____	
Signature of Lab Tutor:	

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	To analyze the step response of a discrete time system and effect of sampling time on system response	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

LAB RUBRIC(S):

Equipment/software Required: Computer with MATLAB

Approximate Time Required: Two to three hours

Introduction

Absolute stability is a basic requirement of all control systems. Apart from that, good relative stability and steady state accuracy are also required in any control system, whether continuous time or discrete time. Transient response corresponds to the system closed loop poles and steady state response corresponds to the excitation poles or poles of the input function. In many practical control systems, the desired performance characteristics are specified in terms of time domain quantities. Unit step input is most commonly used in analysis of a system since it is easy to generate and represent a sufficiently drastic change thus providing useful information on both transient and steady state responses. The transient response of a system depends on the initial conditions. It is a common practice to consider the system initially at rest. Consider the digital control system shown in Figure-5.1

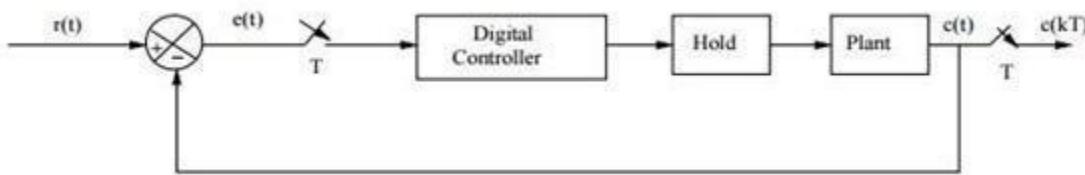


Figure-6.1: Block diagram of a digital control system

Similar to the continuous time case, transient response of a digital control system can also be characterized by the following.

1. Rise time (t_r): Time required for the unit step response to rise from 0% to 100% of its final value in case of under damped system or 10% to 90% of its final value in case of overdamped system.
2. Delay time (t_d): Time required for the unit step response to reach 50% of its final value.
3. Peak time (t_p): Time at which maximum peak occurs.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

4. Peak overshoots (M_p): The difference between the maximum peak and the steady state value of the unit step response.
5. Settling time (t_s): Time required for the unit step response to reach and stay within 2% or 5% of its steady state value.

However since the output response is discrete the calculated performance measures may be slightly different from the actual values as shown in Figure-5.2. The output has a maximum value c_{max} whereas the maximum value of the discrete output is c^*_{max} which is always less than or equal to c_{max} . If the sampling period is small enough compared to the oscillations of the response then this difference will be small otherwise c^*_{max} may be completely erroneous.

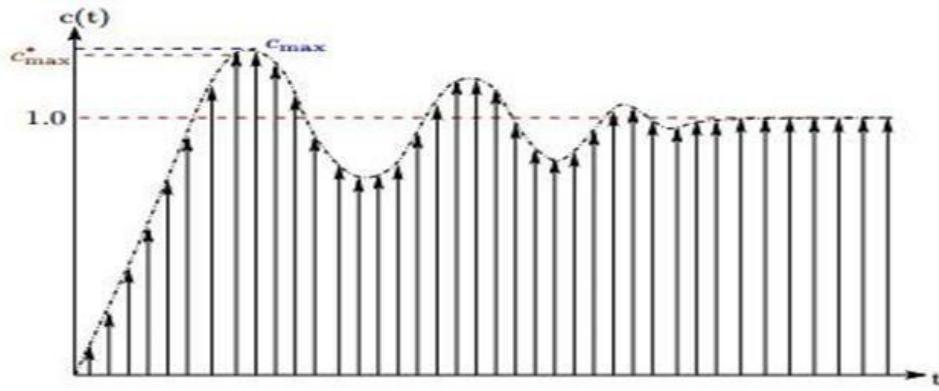


Figure-6.2: Unit step response of discrete time system

6.1 Example1: Consider a unity feedback control system having forward path transfer function $G(s)=$

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

Determine (i) step response in continuous and discrete domain.

(ii) effect of sampling time on system response

Matlab Code:

```
n=[1] d=[1 1 0]
sys=tf(n,d)
sysz=c2d(sys,1,'zoh')
sysc=feedback(sys,1)
syscz=feedback(sysz,1
)
step(syscz,'b') hold on
sysz1=c2d(sys,.5,'zoh')
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

```
syscz2=feedback(sysz1,1
```

```
)
```

```
step(syscz2,'g') hold
```

```
on
```

```
sysz2=c2d(sys,.10,'zoh')
```

```
syscz3=feedback(sysz2,1)
```

```
step(syscz3,'m') hold
```

```
on
```

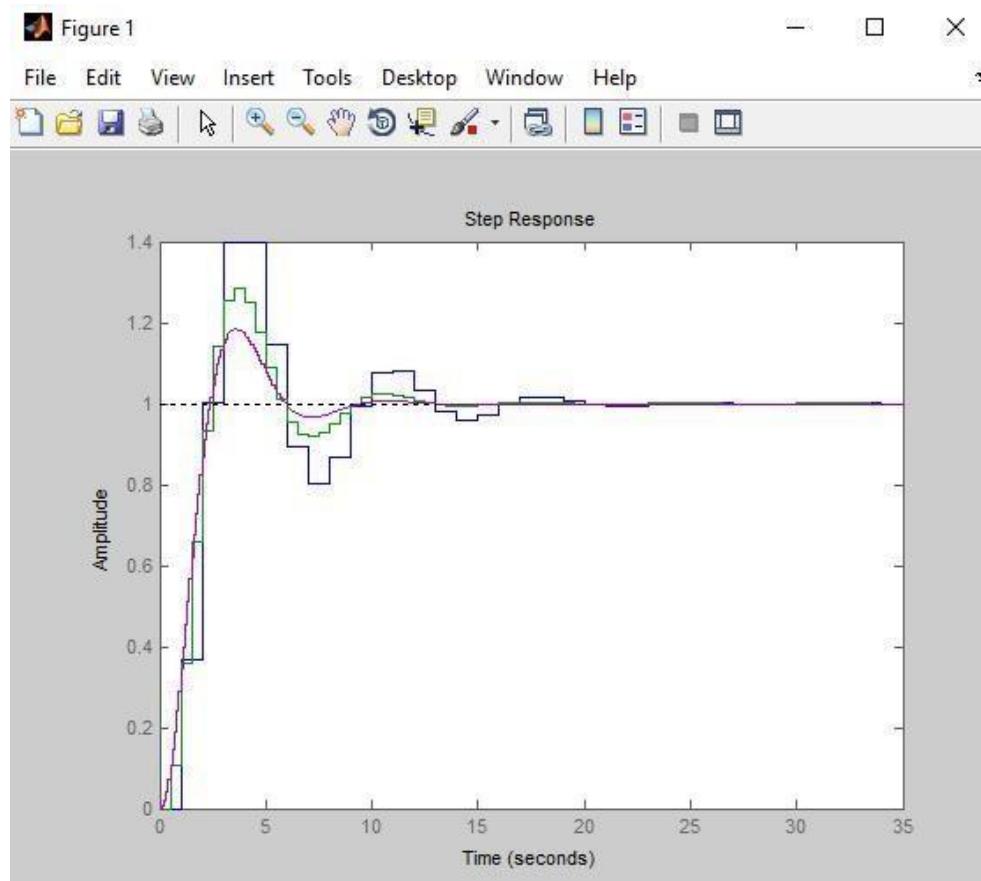


Figure-6.3: Sampling response of example1 at different sampling time



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

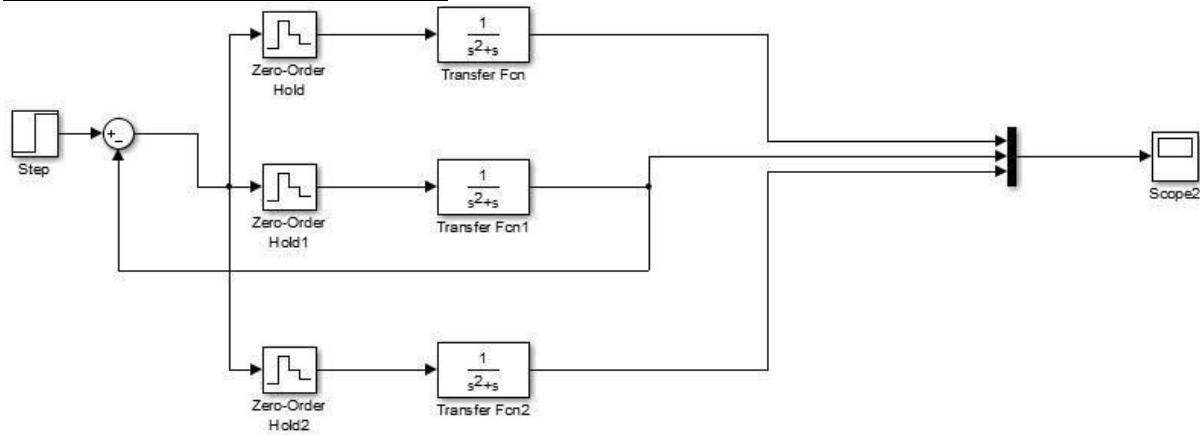


Figure-6.4: Simulink model of step response of system at different sampling time



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

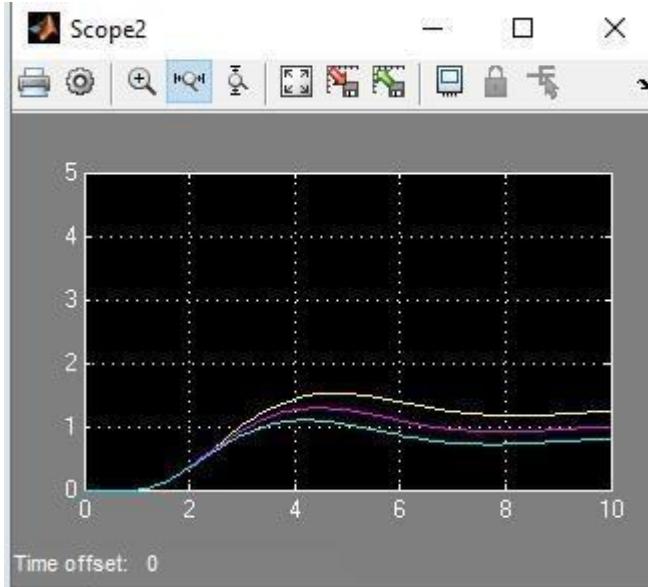


Figure-6.5: Output response of simulink model design

6.2 Example2: Consider a unity feedback control system having forward path transfer function

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

Determine (i) step response in continuous and discrete domain.

(ii) effect of sampling time on system response

Matlab Code:

```
n=[2]; d=[1 3 1];
sys=tf(n,d);
sysz=c2d(sys,1,'zoh');
sysc=feedback(sys,1);
syscz=feedback(sysz,1)
;
step(syscz,'b'); hold
on;
sysz1=c2d(sys,.5,'zoh');
syscz2=feedback(sysz1,1);
step(syscz2,'g'); hold
on;
sysz2=c2d(sys,.10,'zoh');
syscz3=feedback(sysz2,1);
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

```
step(syscz3,'m'); hold  
on;
```

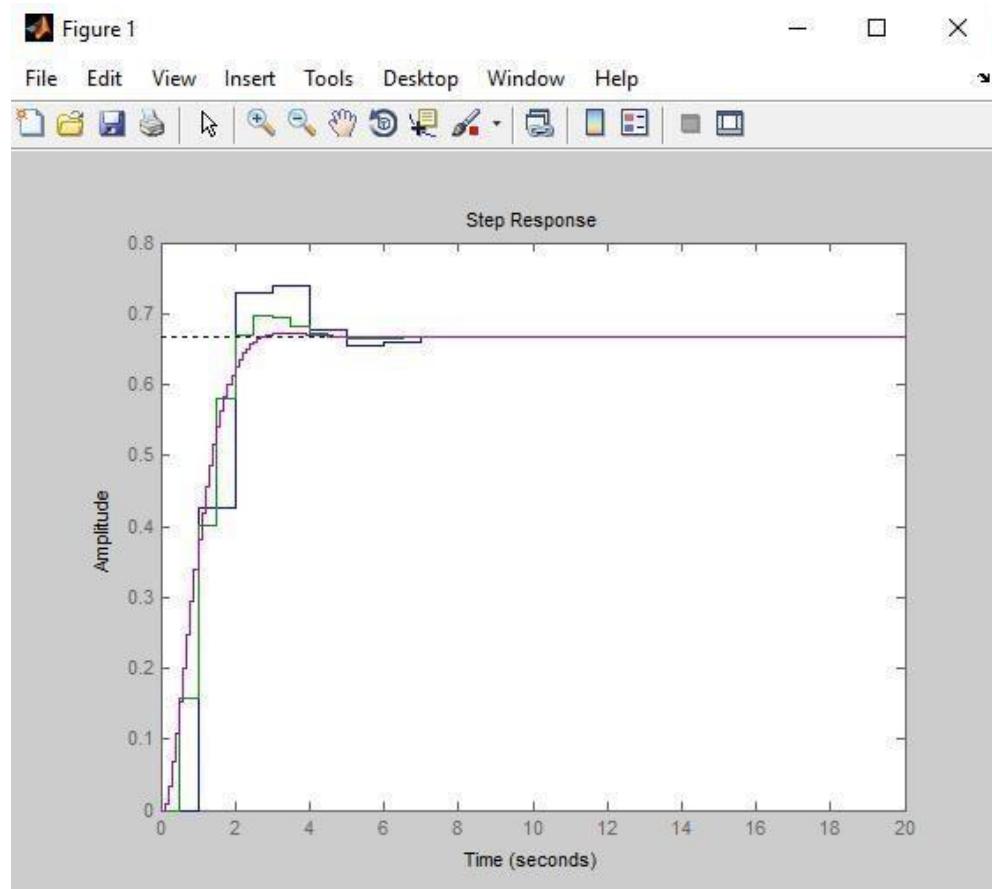


Figure-6.6: Sampling response of example2 at different sampling time



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

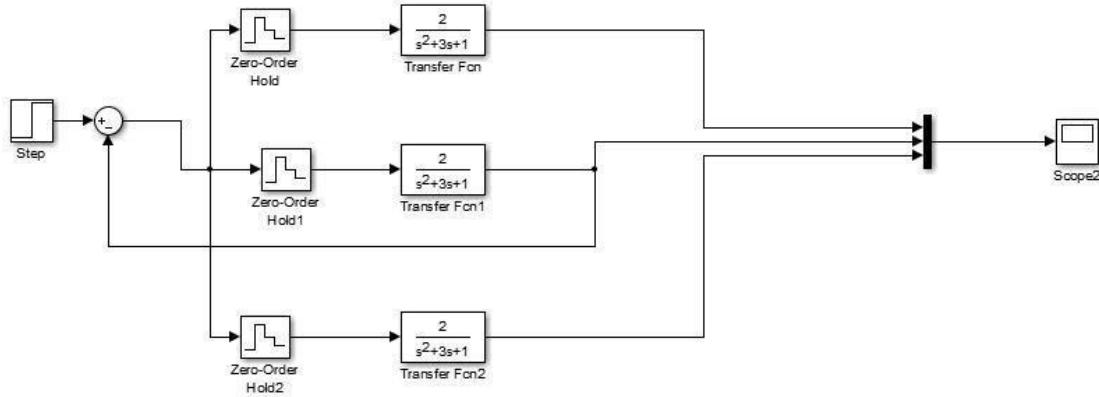


Figure-6.7: Simulink model of step response of system at different sampling time

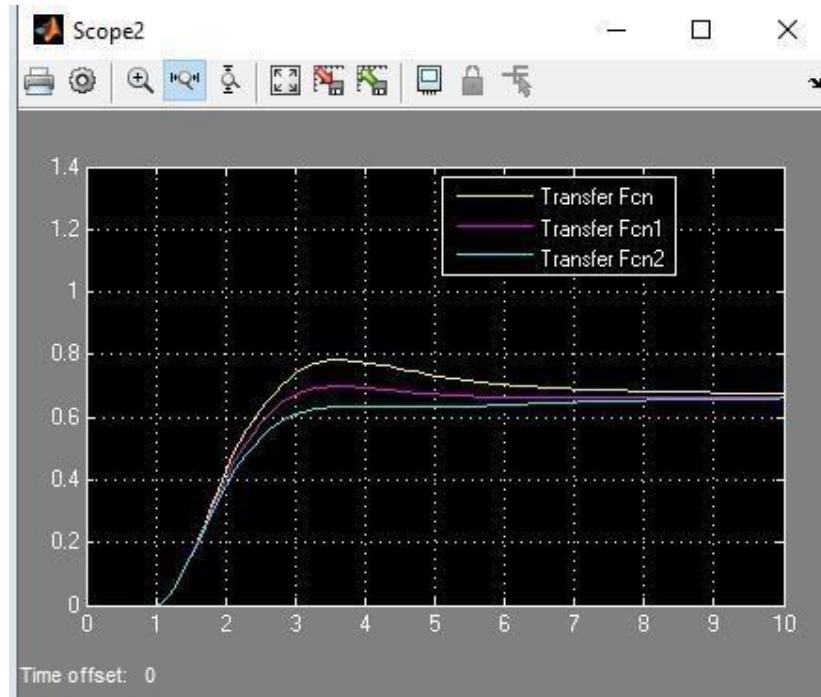


Figure-6.8: Output response of simulink model design

6.3 Review Exercise

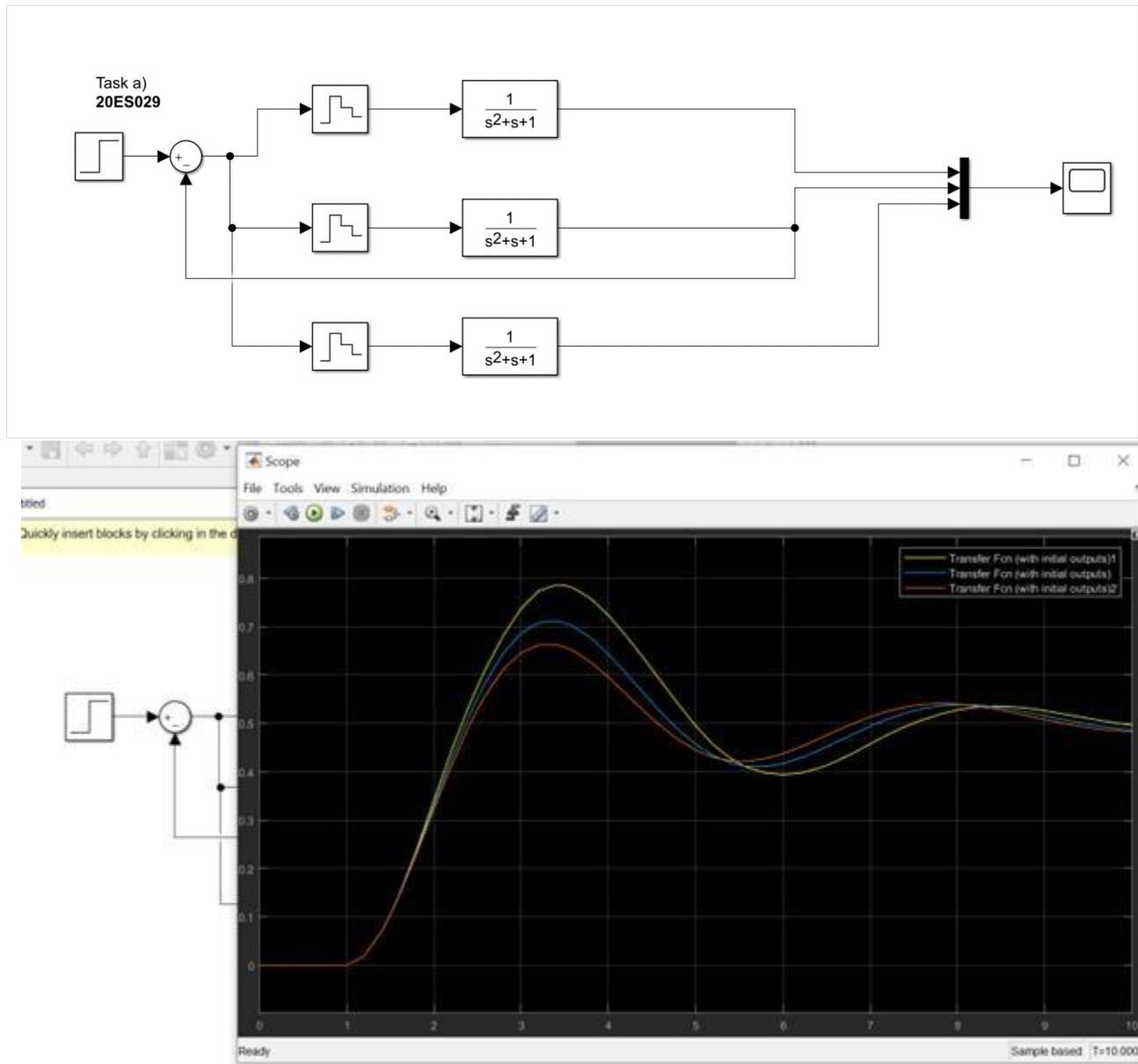


MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

- a) Build Simulink model to obtain step response of a unity feedback system whose closed loop transfer function is given by:

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

Also show effect of sampling time on time response specification parameters.

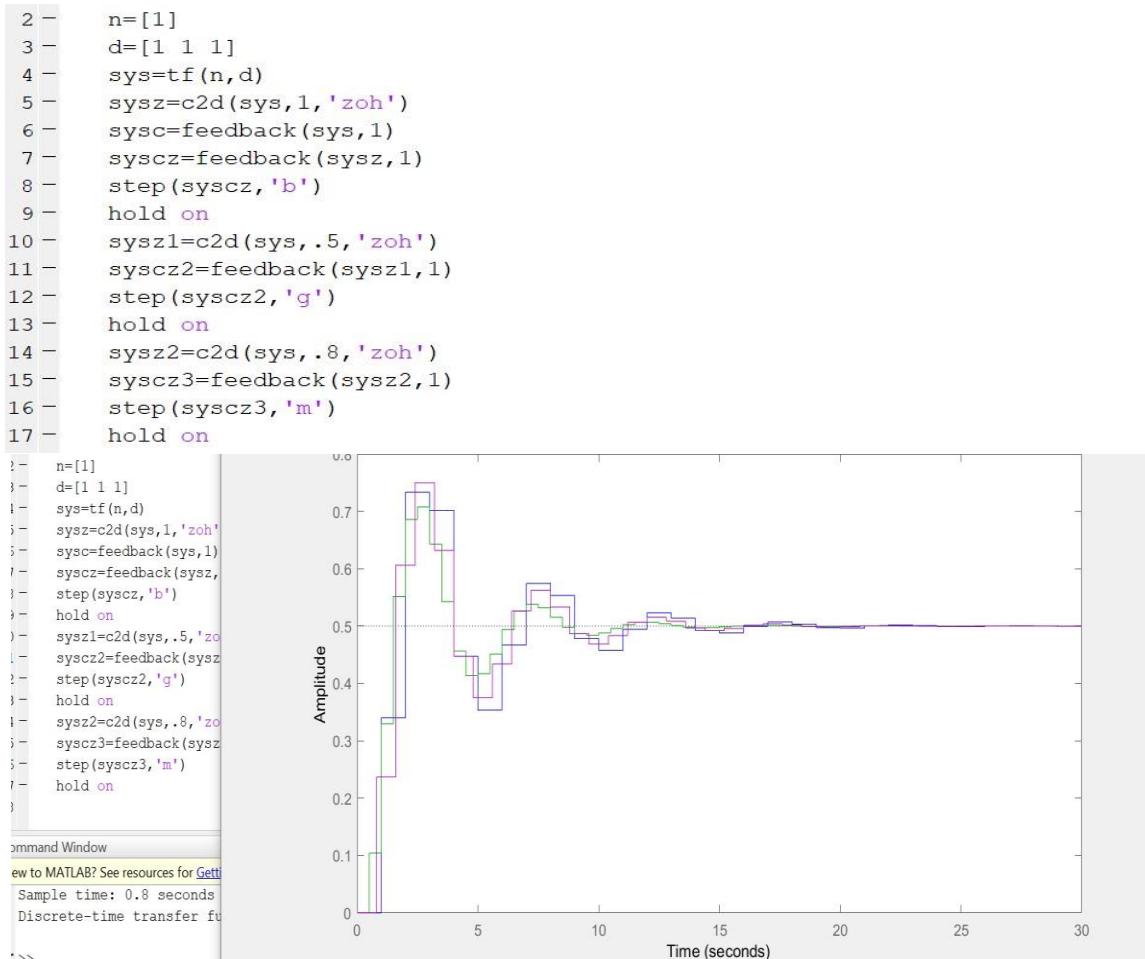




MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

- b) Write down MATLAB code to obtain step response of a unity feedback system having forward path transfer function

$$G_{ts}(s) = \frac{1}{s^2 + s + 1}$$



- c) Briefly illustrate the effect of sampling time on time response specification parameters with the aid of simulation results?

Shorter sampling times in a sample and hold circuit mean it can quickly grab information about the input signal. This helps in capturing the signal accurately and making the output settle faster. However, if the sampling time is too short, the circuit may become more sensitive to small timing errors, called jitter, which can affect accuracy. Also, longer sampling times can lead to the stored voltage (held value) dropping more quickly. In fast situations, short sampling times are needed for quick and accurate responses.

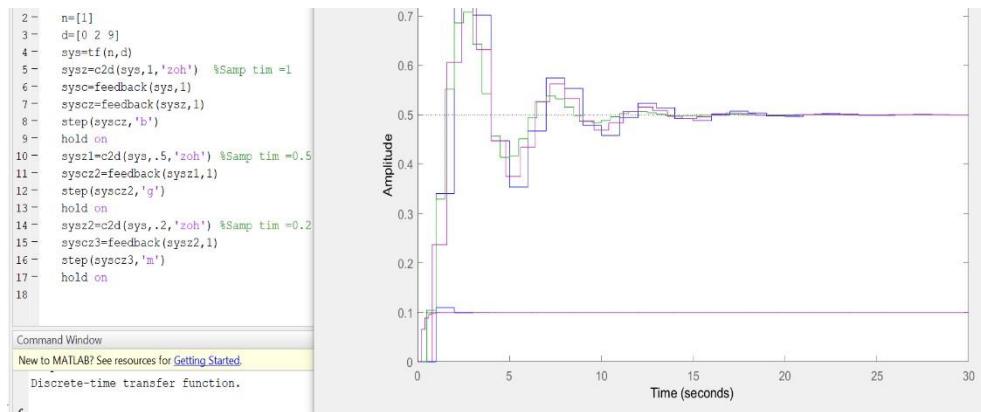


MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

- d) How practical sample and hold circuit works?

A practical sample and hold circuit samples an analog signal during a short period, holds that value, and repeats the process at regular intervals, providing a way to digitize continuous signals. The sample and hold circuit is often used in analog-to-digital converters (ADCs). It allows the conversion of continuous analog signals into discrete digital values by capturing and holding the analog voltage at specific points in time .

- e) Take different systems transfer function. Perform simulation of all systems and explain the behavior of all the systems. Include your simulations and resulting plots.





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

**LAB # 07: The sample response analysis and the computer simulation
of a sampled-data system**

Name: <u>KARAN</u>	Roll # <u>20ES062</u>
Signature of Lab Tutor:	Date:

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	To perform the sample response analysis and the computer simulation of a sampled-data system using MATLAB	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

Equipment/software Required: Computer with MATLAB software

Approximate Time Required: Two to three hours

Introduction

A typical sampled-data system is shown in Figure-7.1, where S and H represent the Sampler (A/D converter) and the Holder (D/A converter) respectively.

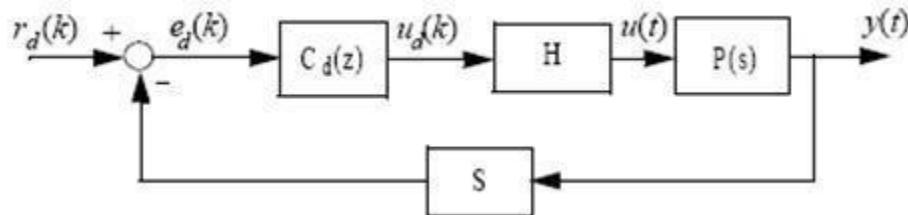


Figure-7.1 Sampled-Data System

Unlike a pure analog system (that has an analog controller and the continuous-time plant), or a pure discrete-time system (with a digital controller and the discretized model of the plant), the sampled-data system has both the digital controller and the continuous-time plant. The signals passing through the entire system is a mixture of sampled data and continuous time signals. Most control system simulation software packages such as MATLAB only have functions for continuous-time and discrete-time simulations, e.g., lsim, step, and dstep, dlsim, etc., but none for sampled data simulation. This lab is to write a general MATLAB program (function) to simulate the step response of a sampled-data (digital) control system.

Experiment Procedure

The analog plant $P(s)$ that we considered in this lab is a flexible beam system:



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

$$P(s) = \frac{1.6188s^2 - 0.1575s - 43.9425}{s^4 + 0.1736s^3 + 27.9001s^2 + 0.0186s}$$

Assume that you have designed an advanced digital controller for this complex system (at sampling period $T = 0.5$ sec), i.e. as in Figure-7.1:

$$C_d(z) = \frac{-0.1084z^5 - 0.01202z^4 + 0.1708z^3 + 0.08469z^2 - 0.09198z - 0.04313}{z^6 - 0.6528z^5 - 0.8377z^4 + 0.4495z^3 + 0.4709z^2 - 0.5820z + 0.1521}$$

Before we can implement this digital controller on-line and perform real-time control on the real flexible beam system, first of all, we need to test the control system performance in a computer environment. This can be achieved by sampled-data (digital) simulation. The following steps describe the procedure for simulating the step response of the sampled data system using existing MATLAB functions:

1. Define the models of the digital controller and the analog plant, e.g. **sysC d = tf(numC d, denC d, T)** %Create a discrete-time transfer function with sampling period T $\approx T = 0.5$
sysP = tf(numP,denP)
2. Discretize the continuous-time plant $P(s)$ to obtain the ZOH equivalent $P_d(z) = SP(s)H$ via the MATLAB function **c2d**

The discretized system is shown in Figure-7.2:

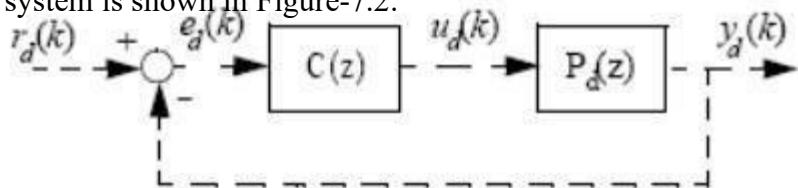


Figure-7.2: The Discretized System

Here r_d is the sampled input sequence, which is a discrete-time unit step. Figure-7.2 is the exact model of Figure-7.1 at sampling instants.

3. Obtain the control signal sequence u_d . First of all, compute the transfer function from r_d to u_d in Figure-7.2,



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

$$G_1(z) = \frac{U_d(z)}{R_d(z)} = \frac{C_d(z)}{1 + C_d(z)P_d(z)}$$

`G1 = feedback(sysC_d, sysP_d)` and then compute the discrete-time control sequence u_d via `dstep`, e.g.

`[numG1,denG1] = tfdata(G1, 'v')`

`U_d = dstep(numG1,denG1,N)`

Note: N is the number of samples to be simulated

4. Simulate zero-order-hold, i.e., compute the continuous-time control input u after the ZOH in Figure-8.1 ($u = Hu_d$). Suppose we would like to compute n inter-sample points for every sampling period (of length T), the time increment (step size) in u is then T/n . Thus the vector u is obtained by holding each value of (u_d for n time, i.e.

$$u = [u_d(1)u_d(1)...u_d(1)u_d(2)...u_d(2)....u_d(N)...u_d(N)]$$

5. With input u to P_{discrete} obtained from above, the output $y_{\text{discrete}}(t)$ of P_{discrete} in Figure-7.1 can finally be computed by continuous-time simulation, e.g. `lsim`.

(The obtained $y_{\text{discrete}}(t)$ is in fact the computer approximation of the actual continuous-time output in the real system. It should be a good approximation because 10 points of the inter-sample response are calculated during each sampling interval T.)
Finally, the above steps can be integrated into a general function named `sdstep` (or choose your own preferred name) in MATLAB:

`[y,u,t] = sdstep(numC_d,denC_d,numP,denP,T,N,n)`

For this function, the user can define the model of the digital controller in `numC_d` and `denC_d`, and the model of the continuous-time plant in `numP` and `denP`. T and N are sampling period and number of samples to be simulated, respectively; and the integer n is the ratio of sampling period T to the time increment $T1$ used when calculating the sample response. The returned variables of the above function are the output from continuous-time plant P_{discrete} , $y_{\text{discrete}}(t)$ (see Figure-7.1, and the corresponding input $u_{\text{discrete}}(t)$ and the time *vector* t .

Practical Analysis of sampling and reconstruction

Example1: Let $f(t) = te^{-1000t} u(t)$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

Create the script file sample_data that returns the following plots:

1. $f(t)$ versus t , over the range $0 \leq t \leq 5$ ms
2. $f(nT)$ versus nt , for $T = 0.2$ ms, over the range $1 \leq n \leq 26$
3. Reconstruct $f(t)$ from $f(nT)$ using
4. Summations of sinc functions (emulating a low-pass filter)
5. Stair function
6. Spine function
7. $\text{error}(t)$ versus t , for the reconstruction process when a low-pass filter is used, where

$$\text{error}(t) = \left| f(t) - \sum_{n=1}^{N=26} f_n(nT_s) \text{sinc}[F_s(t - nT_s)] \right|, \quad \text{for } F_s = \frac{1}{T_s}$$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

MATLAB Solution

```
% Script file: sample_data
t = 0:0.0005:0.005; % 26 time samples
fa = t.*exp(-1000.*t);

figure(1)
subplot(2,1,1)
plot(t,fa);title('f(t) vs t')
ylabel('Amplitude'); xlabel('time t (msec)');
subplot(2,1,2)
Ts = 0.0002; n = 0:1:25; Fs=1/Ts;
nTs = n*Ts;
fd = nTs.*exp(-1000*nTs);
stem(nTs,fd);hold on;
plot(nTs,fd);
title('f(t) sample with Ts=.2msec.');
ylabel('Amplitude'); xlabel('time index n (msec)')
```

```
figure(2) % reconstructions
f25 = 0;
for k = 1:1:26;
    fr25 = fd(k)*sinc(Fs*(t-k*Ts));
    f25 = f25+fr25;
end
subplot(3,1,1)
plot(t,f25,'ko-',t,fa,'ks-.'); legend('sinc-reconst','f(t)');
title('Reconstruction of f(t) ')
ylabel('Amplitude')
subplot(3,1,2)
stem(nTs,fd);hold on;
stairs(nTs,fd); ylabel('Amplitude'); legend('stairs')
subplot(3,1,3)
y = spline(nTs,fd,t);
plot(nTs(1:2.5:26.5),y); legend('spline');
ylabel('Amplitude')
xlabel('time (sec)')

figure(3)
error=abs(fa-f25);
plot(t,error);ylabel('magnitude')
title('error(t) = abs [f(t) - Reconstruction (Sums(sinc))]')
xlabel('time(sec)')
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

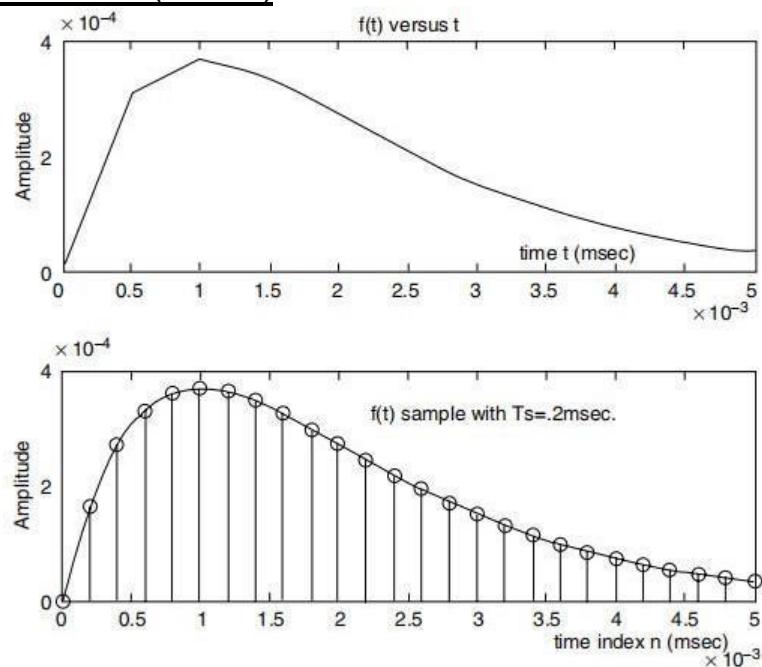


Figure-7.3: Plot of $F(t)$ and $F(nT)$

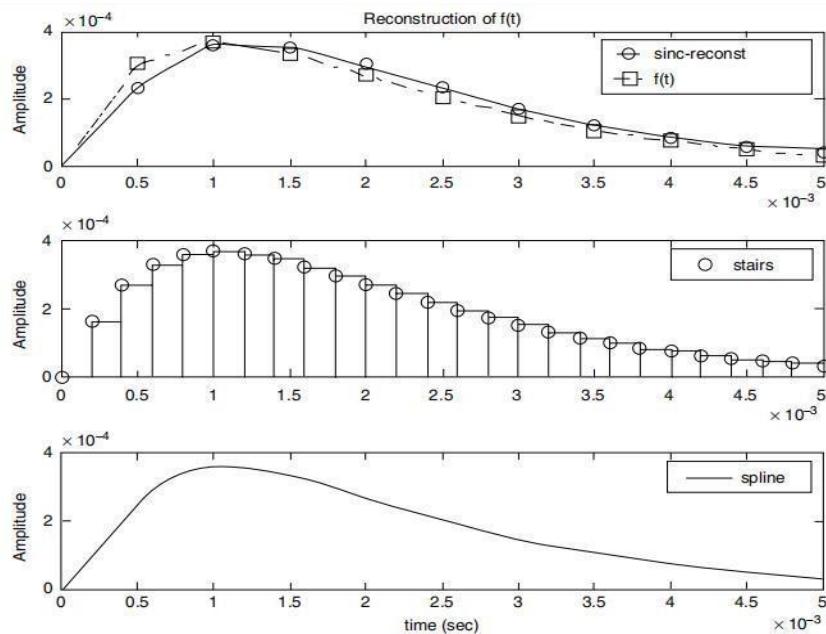


Figure-7.4: Reconstruction plot for $F(t)$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

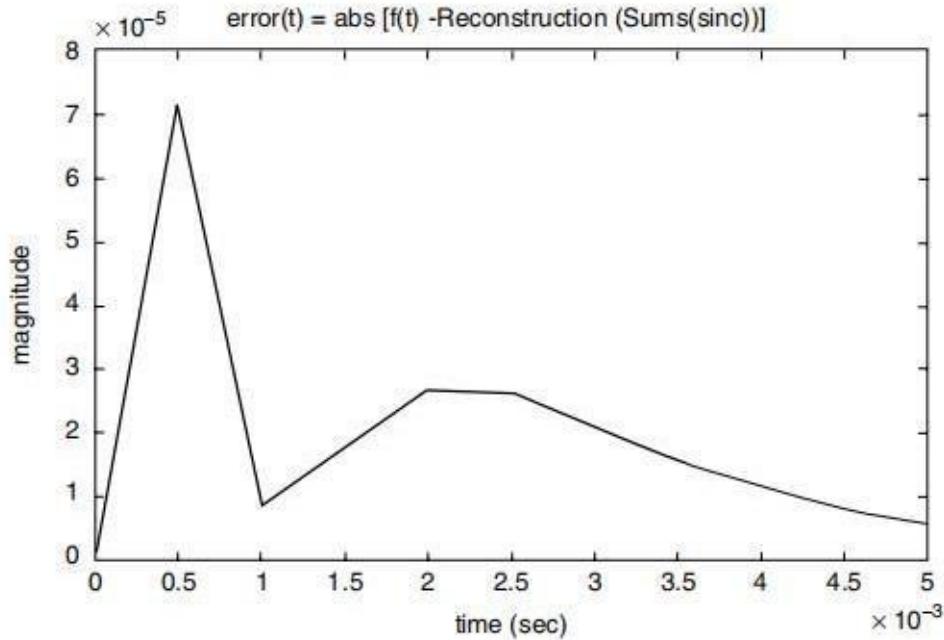


Figure-7.5: Reconstruction plot error for $F(t)$

Example2: Create the script file up_down_samples that returns the approximation plots of the following discrete time function:

$$f(n) = \cos(2 \cdot 0.05n) + 2 \sin(2 \cdot 0.03n)$$

for a length of $N = 100$ samples, for the following cases:

1. Down-sample or decimate the sequence $f(n)$ with the integer factors of $M = 2, 4$, and 10 .
2. Up-sample or interpolate the sequence $f(n)$ with the integer factors of $L = 2, 4$, and 10 .
3. Resample $f(n)$ by the ratio of the two integers given by L/M , for the following cases: $L = 3, M = 2$, and $L = 2, M = 3$.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

```
MATLAB Solution
% Script file: up_down_samples
n = 0:1:99;
f = cos(2*pi*.05*n)+2*sin(2*pi*.03*n);
yzero = zeros(1,500);

figure(1); % down-sample with M=2, 4, 10
subplot(2,2,1)
stem(n,f);hold on; plot(n,f,n,yzero(1:100))
title('f(n) vs n'); ylabel('Amplitude');
xlabel('time index n');
subplot(2,2,2)
gm2 = decimate(f,2,'fir');
nm2 = 0:100/2-1;
stem(nm2,gm2(1:50)); hold on; plot(nm2,yzero(1:50));
title('f(n) is decimated with M=2'); ylabel('Amplitude');
xlabel('time index n');
subplot(2,2,3)
gm4 = decimate(f,4,'fir');
nm4 = 0:100/4-1;
stem(nm4,gm4(1:25));hold on;
plot(nm4,gm4(1:25),nm4,yzero(1:25));
ylabel('Amplitude'); xlabel('time index n');
title('f(n) is decimated with M=4')
subplot(2,2,4)
gm10 = decimate(f,10,'fir');
nm10=0:100/10-1;
stem(nm10,gm10(1:10));hold on;
plot(nm10,gm10(1:10),nm10,yzero(1:10));
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
CONTROL SYSTEMS (ES-413)

DIGITAL

```
ylabel('Amplitude'); xlabel('time index n')
title('f(n) is decimated with M=10')

figure(2);
% up-sample with L = 2, 4, 10
subplot(2,2,1)
stem(n,f); hold on; plot(n,f,n,yzero(1:100))
title('f(n) vs n'); ylabel('Amplitude');
xlabel('time index n');
subplot(2,2,2)
gL2 = interp(f,2);
nL2 = 0:100*2-1;
stem(nL2,gL2(1:200)); hold on; plot(nL2,yzero(1:200));
title('f(n) is upsampled with L=2'); ylabel('Amplitude');
xlabel('time index n');
subplot(2,2,3)
gL4 = interp(f,4);
nL4 = 0:100*4-1;
stem(nL4,gL4(1:400)); hold on;
plot(nL4,yzero(1:400));
ylabel('Amplitude'); xlabel('time index n');
title('f(n) is upsampled with L=4');
subplot(2,2,4)
gL10= interp(f,10);
nL10 = 0:100*10-1;
stem(nL10,gL10(1:1000)); hold on;
ylabel('Amplitude'); xlabel('time index n')
title('f(n) is upsampled with L=10')

figure(3)

% re-sample by ratio of L=3/M=2 & L=2/M=3
subplot(3,1,1)
stem(n,f); hold on; plot(n,f,n,yzero(1:100))
title('f(n) vs n');
ylabel('Amplitude'); xlabel('time index n');
subplot(3,1,2)
gr32= resample(f,3,2);
nr32 = 0:100*3/2-1; yzeros = zeros(1,length(nr32));
stem(nr32,gr32(1:100*3/2)); hold on; plot(nr32,yzeros);
ylabel('Amplitude'); xlabel('time index n');
subplot(3,1,3)
gr23 = resample(f,2,3);
nr23 = 0:100*2/3-1;
yzero = zeros(1,length(nr23));
stem(nr23,gr23(1:100*2/3));
hold on; plot(nr23,yzero);
axis([0 65 -5 5]);
ylabel('Amplitude'); xlabel('time index n')
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

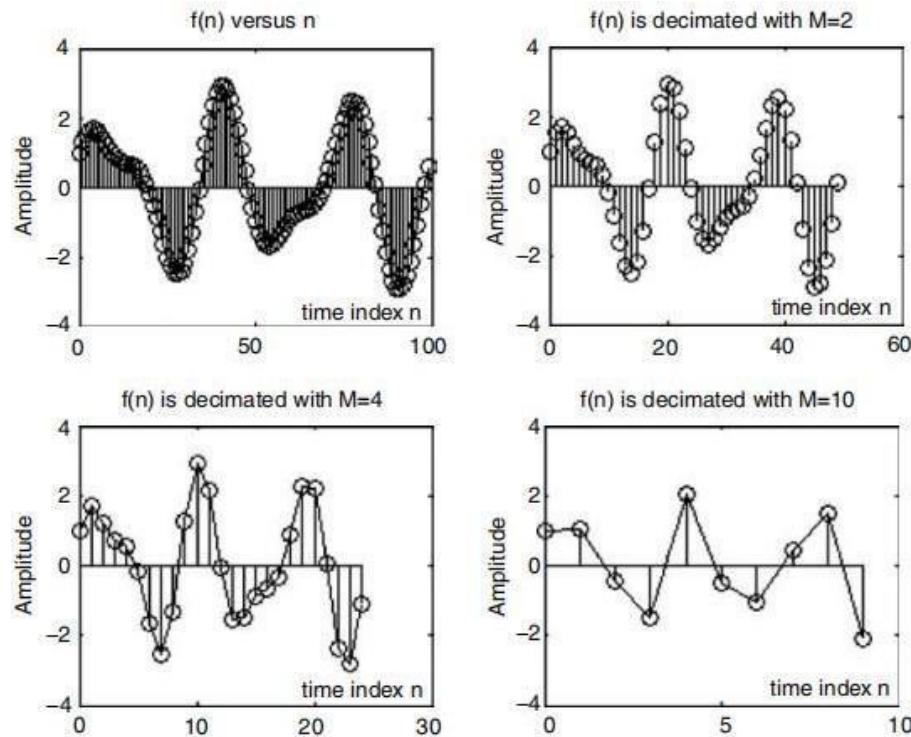


Figure-7.6: Decimation plot of $F(t)$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

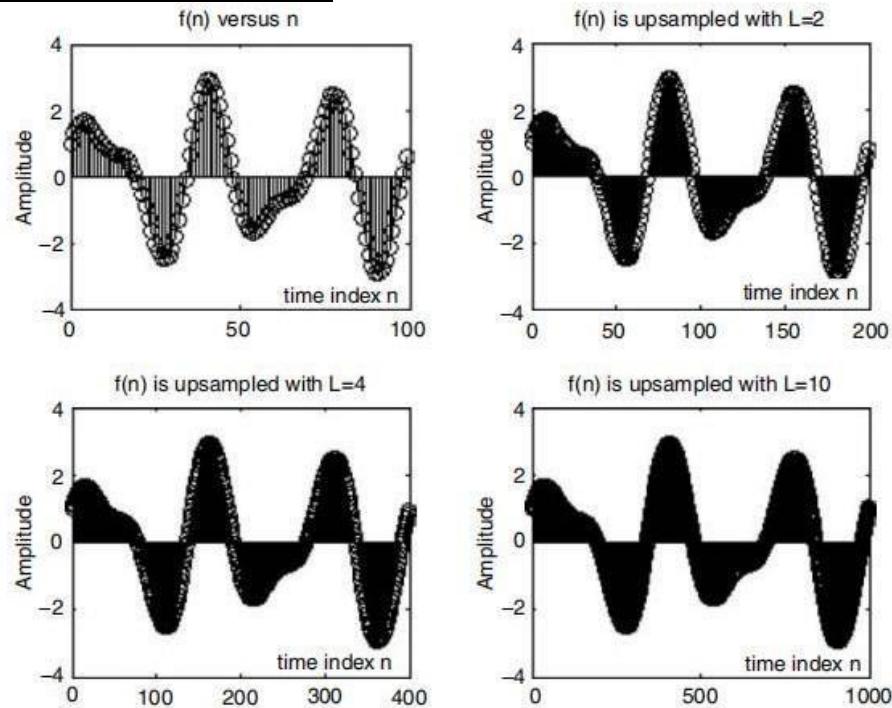


Figure-7.8: Interpolation plots of $F(t)$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

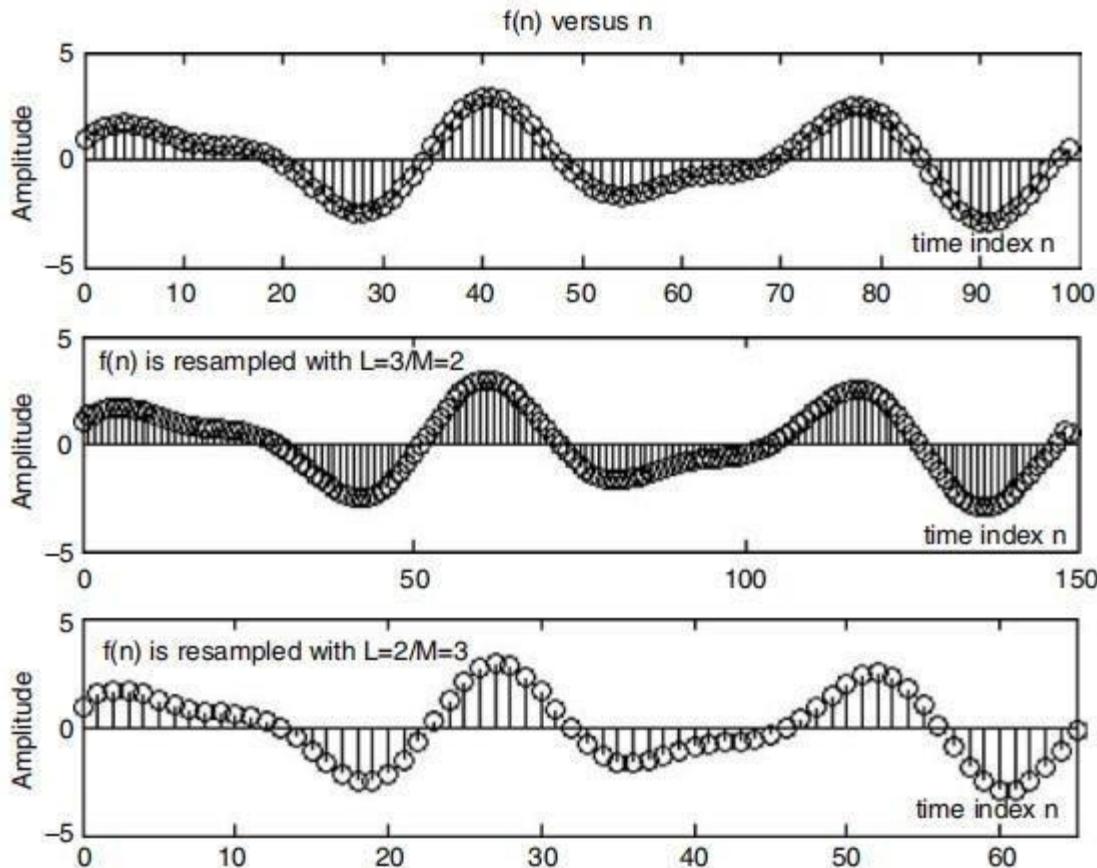


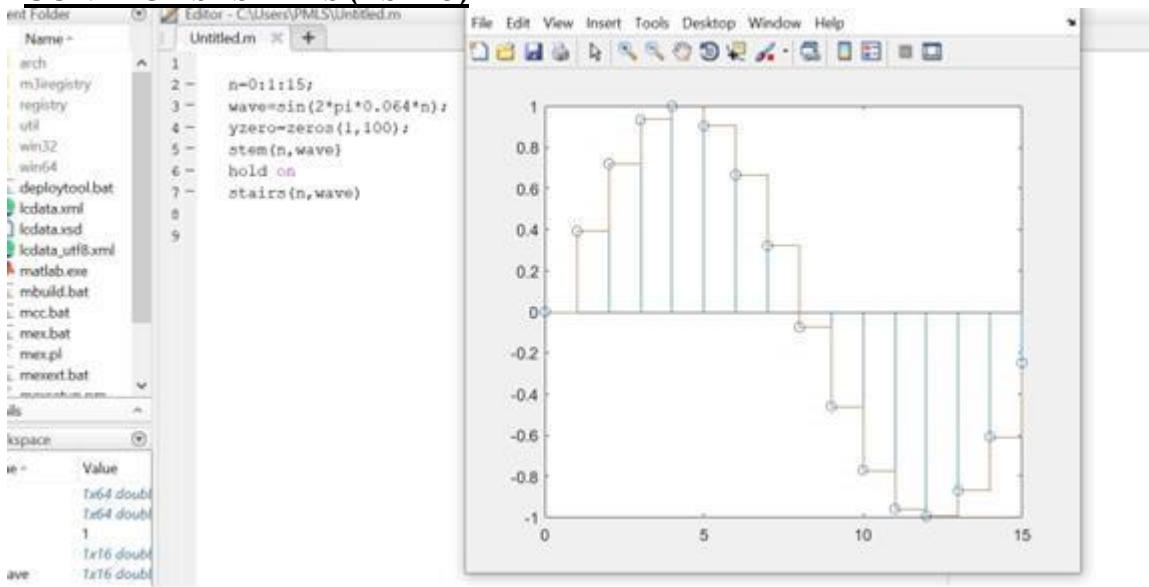
Figure-7.9: Plots of $f(t)$ and resampled $f(t)$ with notes given by L/M

Review Exercise?

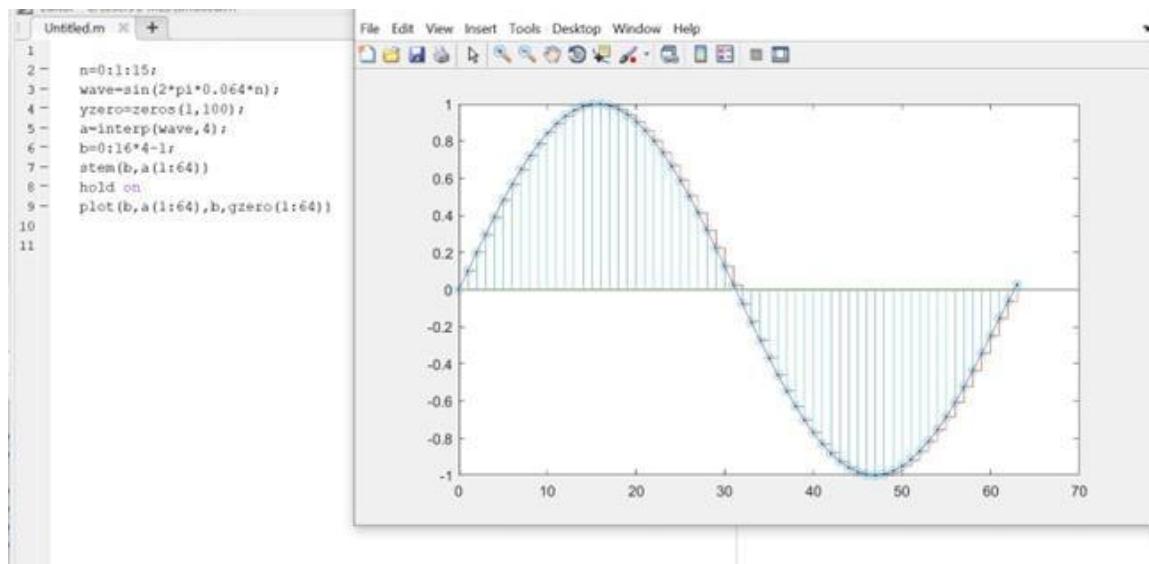
- Construct a sinusoidal signal. Specify a sample rate such that 16 samples corresponds to exactly one signal period. Draw a stem plot of the signal. Overlay a stair step graph for sample and hold visualization.?



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)



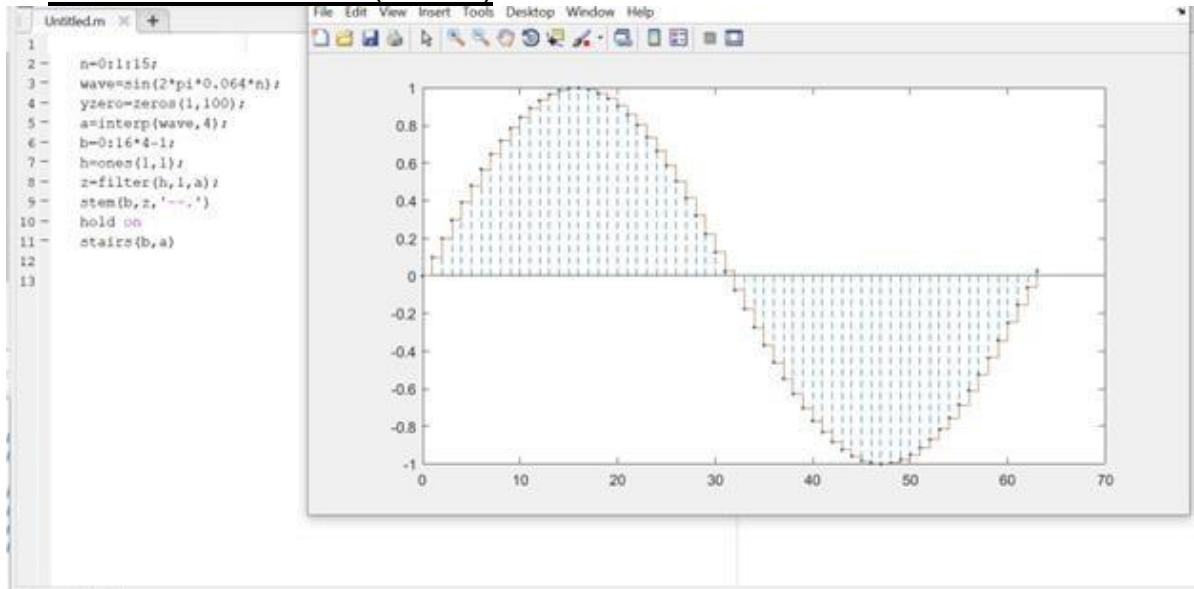
- b) Up sample the signal by a factor of four. Plot the result along with the original signal. Up sample increases the sample rate of the signal by adding zeroes between the existing samples?



- c) Filter with moving average FIR filter to fil in the zeros with sample and hold values?



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

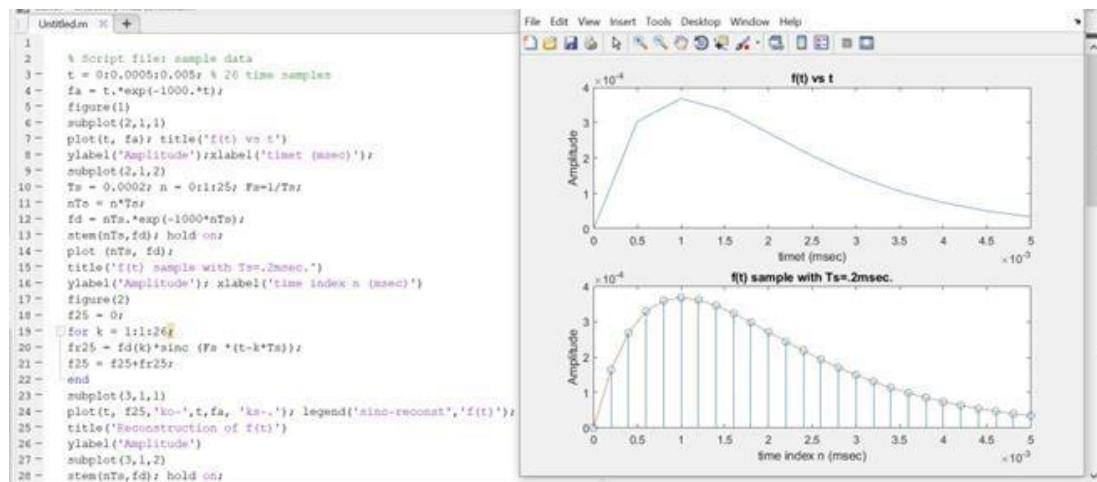


d) What is the effect of Zero-order-hold?

The zero-order hold is a method which is widely used to reconstruct the signals in real time. In the zero-order hold reconstruction method, the continuous signal is reconstructed from its samples by holding the given sample for an interval until the next sample is received. Therefore, the zero-order hold generates the step approximations.

e) Attach the screenshots of MATLAB code and plots of example1 and example2 in your report

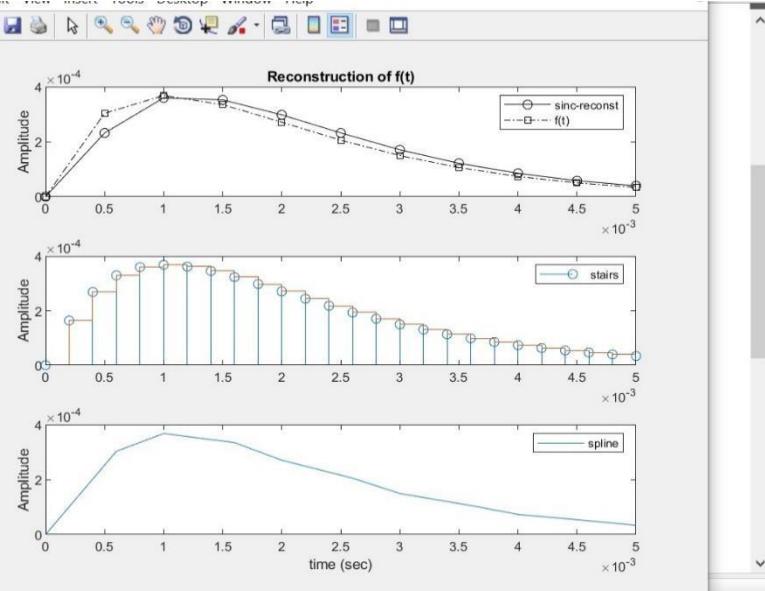
Example 1



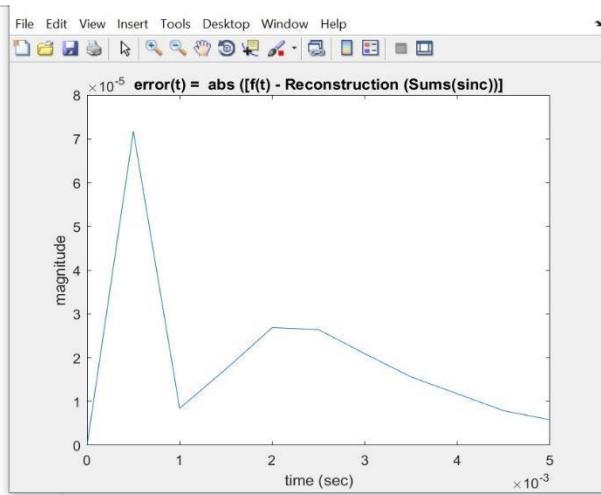


MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

```
f25 = f25+fr25;
end
subplot(3,1,1)
plot(t, f25,'ko- ',t,fa, 'ks- .'); legend('sinc-reconst','f(t)');
title('Reconstruction of f(t)')
ylabel('Amplitude')
subplot(3,1,2)
stem(nTs,fd); hold on;
stairs (nTs, fd); ylabel('Amplitude'); legend('stairs')
subplot(3,1,3)
y = spline(nTs,fd,t);
plot(nTs(1:2.5:26.5),y); legend('spline');
ylabel('Amplitude')
xlabel('time (sec)')
figure(3)
error=abs (fa-f25);
plot(t, error); ylabel('magnitude')
title('error(t) = abs ([f(t) - Reconstruction (Sums(sinc))])')
xlabel('time (sec)')
t = 0:0.0005:0.005; % 26 time samples
fa = t.*exp(-1000.*t);
figure(1)
subplot(2,1,1)
plot(t, fa); title('f(t) vs t')
ylabel('Amplitude'); xlabel('timet (msec)')
subplot(2,1,2)
Ts = 0.0002; n = 0:1:25; Fs=1/Ts;
nTs = n*Ts;
```



```
49 - fd = nTs.*exp(-1000*nTs);
50 - stem(nTs,fd); hold on;
51 - plot(nTs, fd);
52 - title('f(t) sample with Ts=.2msec.')
53 - ylabel('Amplitude'); xlabel('time index n (msec)')
54 - figure(2)
55 - f25 = 0;
56 - for k = 1:1:26
57 - fr25 = fd(k)*sinc (Fs *(t-k*Ts));
58 - f25 = f25+fr25;
59 - end
60 - subplot(3,1,1)
61 - plot(t, f25,'ko- ',t,fa, 'ks- .'); legend('sinc-reconst','f(t)');
62 - title('Reconstruction of f(t)')
63 - ylabel('Amplitude')
64 - subplot(3,1,2)
65 - stem(nTs,fd); hold on;
66 - stairs (nTs, fd); ylabel('Amplitude'); legend('stairs')
67 - subplot(3,1,3)
68 - y = spline (nTs,fd,t);
69 - plot(nTs(1:2.5:26.5),y); legend('spline');
70 - ylabel('Amplitude')
71 - xlabel('time (sec)')
72 - figure(3)
73 - error=abs (fa-f25);
74 - plot(t, error); ylabel('magnitude')
75 - title('error(t) = abs ([f(t) - Reconstruction (Sums(sinc))])')
76 - xlabel('time (sec)')
```

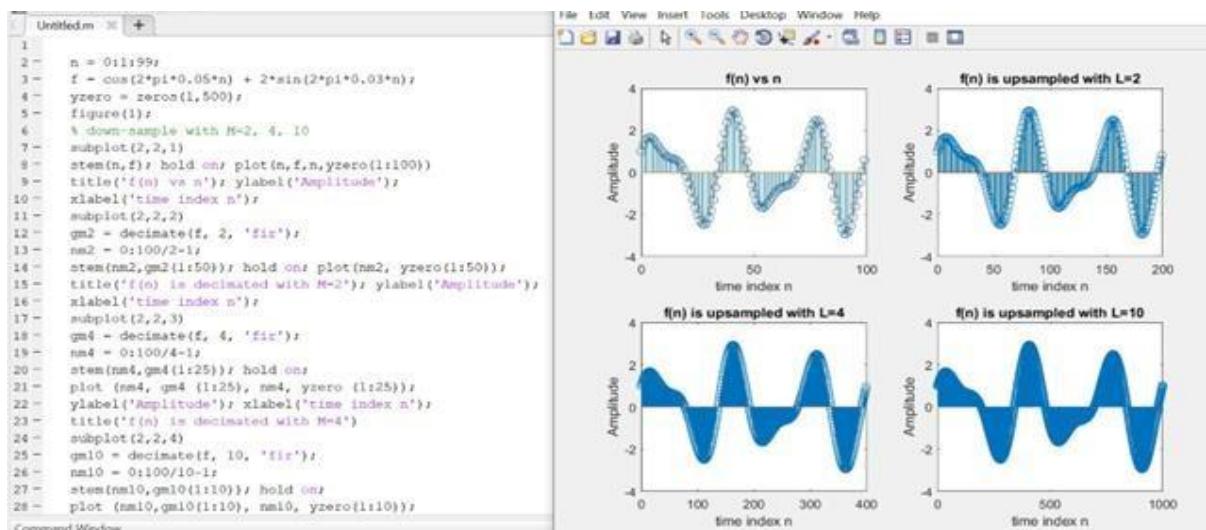
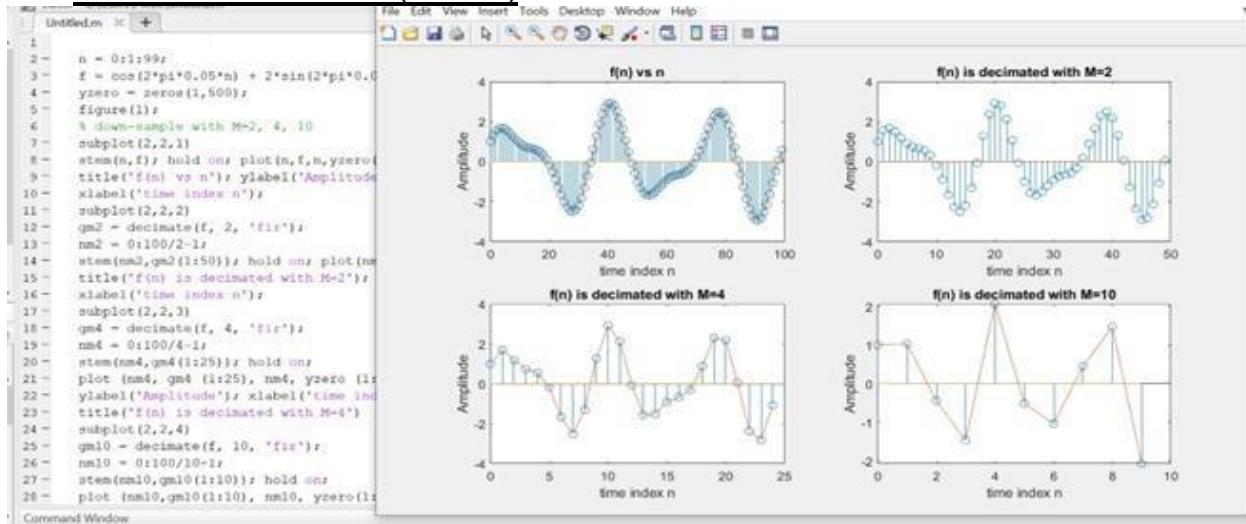


Example 2:



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
CONTROL SYSTEMS (ES-413)

DIGITAL



f) Write conclusion/observation of this laboratory?

An analog signal can only be effectively utilized after conversion to digital form if it is sampled accurately. Otherwise, the resulting digital representation may not faithfully capture the original signal, rendering it unusable or unreliable for further processing or analysis.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)



**LAB # 08: THE DISCRETE TIME STATE SPACE REPRESENTATION OF A SYSTEM AND
MODELING OF MAGNETICAL SUSPENDED BALL SYSTEM**

Name: KARAN Roll # 20ES062
Date: _____

Signature of Lab Tutor:

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	To illustrate the Discrete-Time State Space representation of a System and modeling of magnetically suspended ball system using MATLAB	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)



Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	
Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

Equipment Required: Computer with MATLAB

Approximate Time Required: Two to three hours.

Introduction: The Transfer function approach is applicable only to linear time-invariant systems having a single-input and single output. Thus, the state-space methods for the analysis and synthesis of control systems are best suited for dealing with linear/non linear time variant, time invariant and multiple-inputmultiple-output systems along with linear time-invariant and single-input-single- output systems.

Concept of State-Space Method: State-space method is based on the description of system in terms of n first-order difference equations (DTS) or differential equations (CTS) which can be represented in vector-matrix notation to simplify the mathematical representation of system of equations.

State & State Variables: The *state* of a system is the smallest set of variables called *state variables*, such that the knowledge of these variables at $t=0$, together with the knowledge of input for $t \geq 0$, completely determines the behaviour of the system for any time $t \geq 0$. In state-space analysis, we are concerned with three types of variables; input variables, output variables and state variables.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

For time-varying (linear or nonlinear) discrete-time systems, the state equation may be written as

$$\mathbf{x}(k+1) = \mathbf{f}[\mathbf{x}(k), \mathbf{u}(k), k]$$

and the output equation as

$$\mathbf{y}(k) = \mathbf{g}[\mathbf{x}(k), \mathbf{u}(k), k]$$

For linear time-varying discrete-time systems, the state equation and output equation may be simplified to

$$\mathbf{x}(k+1) = \mathbf{G}(k)\mathbf{x}(k) + \mathbf{H}(k)\mathbf{u}(k)$$

$$\mathbf{y}(k) = \mathbf{C}(k)\mathbf{x}(k) + \mathbf{D}(k)\mathbf{u}(k)$$

where

$$\mathbf{x}(k) = n\text{-vector} \quad (\text{state vector})$$

$$\mathbf{y}(k) = m\text{-vector} \quad (\text{output vector})$$

$$\mathbf{u}(k) = r\text{-vector} \quad (\text{input vector})$$

$$\mathbf{G}(k) = n \times n \text{ matrix} \quad (\text{state matrix})$$

$$\mathbf{H}(k) = n \times r \text{ matrix} \quad (\text{input matrix})$$

$$\mathbf{C}(k) = m \times n \text{ matrix} \quad (\text{output matrix})$$

$$\mathbf{D}(k) = m \times r \text{ matrix} \quad (\text{direct transmission matrix})$$

The appearance of the variable k in the arguments of matrices $\mathbf{G}(k)$, $\mathbf{H}(k)$, $\mathbf{C}(k)$, and $\mathbf{D}(k)$ implies that these matrices are time varying. If the variable k does not appear explicitly in the matrices, they are assumed to be time invariant, or constant. That is, if the system is time invariant, then the last two equations can be simplified to

$$\mathbf{x}(k+1) = \mathbf{G}\mathbf{x}(k) + \mathbf{H}\mathbf{u}(k)$$

$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k)$$

Note: The notation $x(k)$ simply means the vector $x(t)$ at $t=kT$, where $k=0,1,2,\dots$ & T is the sampling period.



In matrix form,

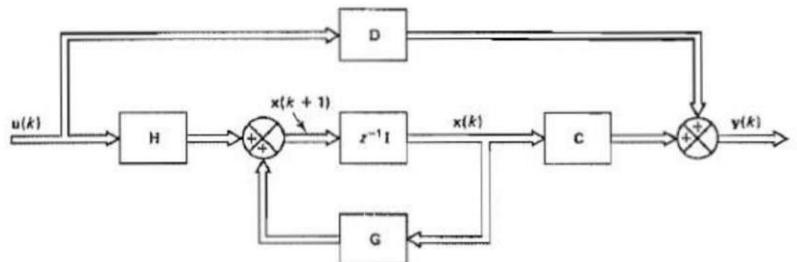
Figure-8.1: Block Diagram Representation of Discrete-Time State Equations

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \\ \vdots \\ x_{n-1}(k+1) \\ x_n(k+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & -a_1 \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_{n-1}(k) \\ x_n(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} u(k)$$

$$y(k) = [b_n - a_n b_0; b_{n-1} - a_{n-1} b_0; \dots; b_1 - a_1 b_0] \begin{bmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_n(k) \end{bmatrix} + b_0 u(k)$$

State Equations of Sampled Data Systems

Let us assume that the following continuous time system is subject to sampling process with an interval of T



$$x(t) = Ax(t) + Bu(t) \quad : \text{state}$$

$$\text{equat}^{\circ} \text{ on } , y(t) = Cx(t) + Du(t)$$

:Output equat^o o

We know that the solution to above state equation is:

$$()_t = \varphi(t \text{ t to } x) + \int_{t_0}^t \varphi(t \text{ t } T) Bu(t) dt \quad 10)$$

Since the inputs are constants in between two sampling instants, one can write

$$\leq t \leq (k+1) \quad UT = u \frac{dT}{dt} kT \text{ for } hT \quad 11)$$

$$kT \quad () \quad () \quad + \int_{t_0}^t \varphi(t \text{ t } T) Bu(t) dt \quad \text{We consider } t_0 = kT$$

$$x(t) = \int_{kT}^{(k+1)T} \varphi(t \text{ t } T) Bu(t) dt \quad t = \varphi(t \text{ t } kT) x(kT) \quad 13)$$

$$\theta(T) = \int_{kT}^{(k+1)T} \theta(t) dt \quad kT dT$$

$$T \int t du$$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)
MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

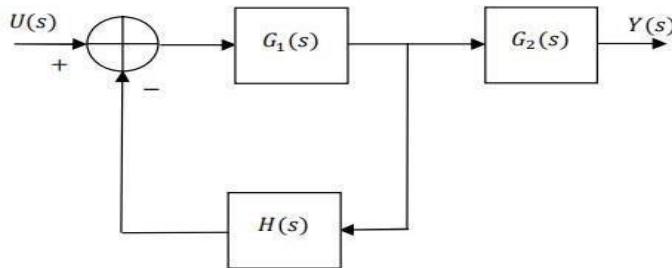
When a discrete system is composed of all digital signals, the state and output equations can be described by

$$x[k+1] = Ax[k] + Bu[k] \quad (1)$$

$$y[k] = Cx[k] + Du[k]$$

$$G(z) = \frac{Y(z)}{U(z)} = CzI + A^{-1}B + D \quad (14)$$

Example 1: Example. Using MATLAB, obtain the transfer function for the control system in the figure below.



where

$$\mathbf{G}_1 = \begin{pmatrix} 1 & 3 \\ 2 & 5 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u; \quad \mathbf{C}_1 = \begin{pmatrix} 1 & 2 \end{pmatrix}$$

$$\mathbf{G}_2 = \begin{pmatrix} -4 & 0 \\ 0 & 3 \end{pmatrix} x + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u; \quad \mathbf{C}_2 = \begin{pmatrix} 3 & 0 \end{pmatrix}$$

$$\mathbf{H} = \begin{pmatrix} 3 & 2 \\ 1 & 0 \end{pmatrix} x + \begin{pmatrix} 1 \\ 1 \end{pmatrix} u; \quad \mathbf{C}_3 = \begin{pmatrix} 1 & 1 \end{pmatrix}$$

MATLAB Solution:

```
a1 = [1 3; 2 5]; b1 =  
[0; 1]; c1 = [1 2]; d1  
= 0; a2 = [-4 0; 0 3];  
b2 = [1; 0]; c2 = [3 0];  
d2 = 0; a3 = [3 2; 1 0];  
b3 = [1; 1]; c3 = [1 1]; d3  
= 0;
```

The following MATLAB code will convert the system from state-space to transfer function:

```
[numg1, deng1] = ss2tf(a1,b1,c1,d1);  
[numg2, deng2] = ss2tf(a2,b2,c2,d2);
```



```
[numh, denh] = ss2tf(a3,b3,c3,d3);
[ numf,           denf] = feedback(numg1,deng1,numh,denh); [nums, dens] =
series(numf,denf,numh,denh);
printsys(nums,dens)

num/den =
```

$$4 s^4 - 10 s^3 - 14 s^2 - 4 s$$

$$-----$$
$$s^6 - 12 s^5 + 44 s^4 - 22 s^3 - 87 s^2 - 40 s - 4$$

Example 2: Discretize the following continuous-time transfer function

$$H(s) = e^{-0.3s} \frac{s-1}{s^2+4s+5}.$$

MATLAB Solution:

```
H = tf([1 -1],[1 4 5],'InputDelay', 0.3);
Hd = c2d(H,0.1,'foh'); step(H,'-',Hd,'--')
```

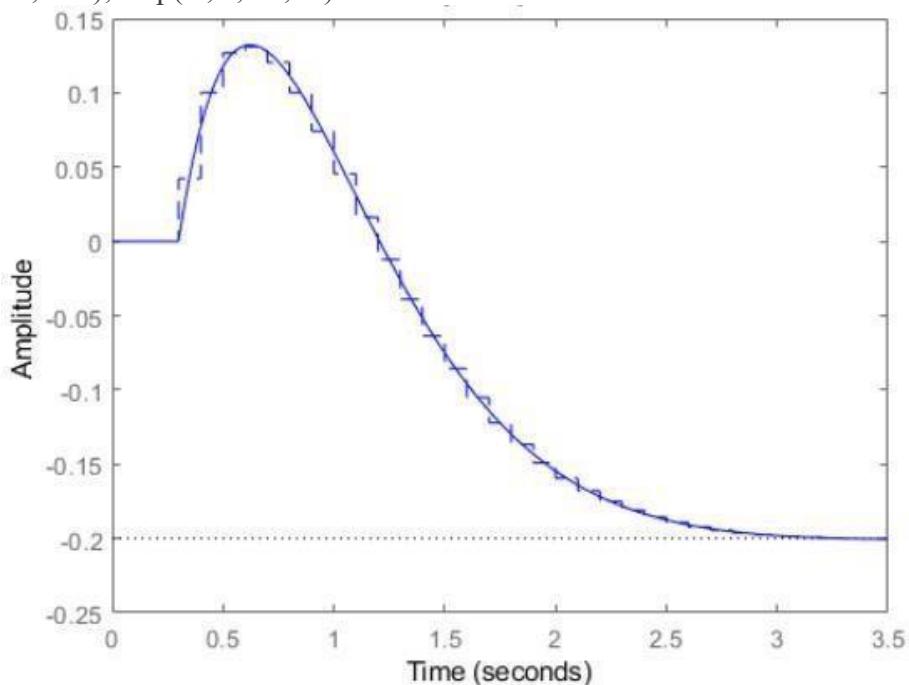


Figure-8.2: output response of transfer function



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO

DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO

DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

Example 3: Discretize the following delayed transfer function using zero-order hold on the input, and a 10-Hz sampling rate.

$$H(s) = e^{-0.25s} \frac{10}{s^2 + 3s + 10}.$$

MATLAB Solution:

```
h = tf(10,[1 3 10],'IODelay',0.25);
hd = c2d(h,0.1) step(h,'--',hd,'-')
```

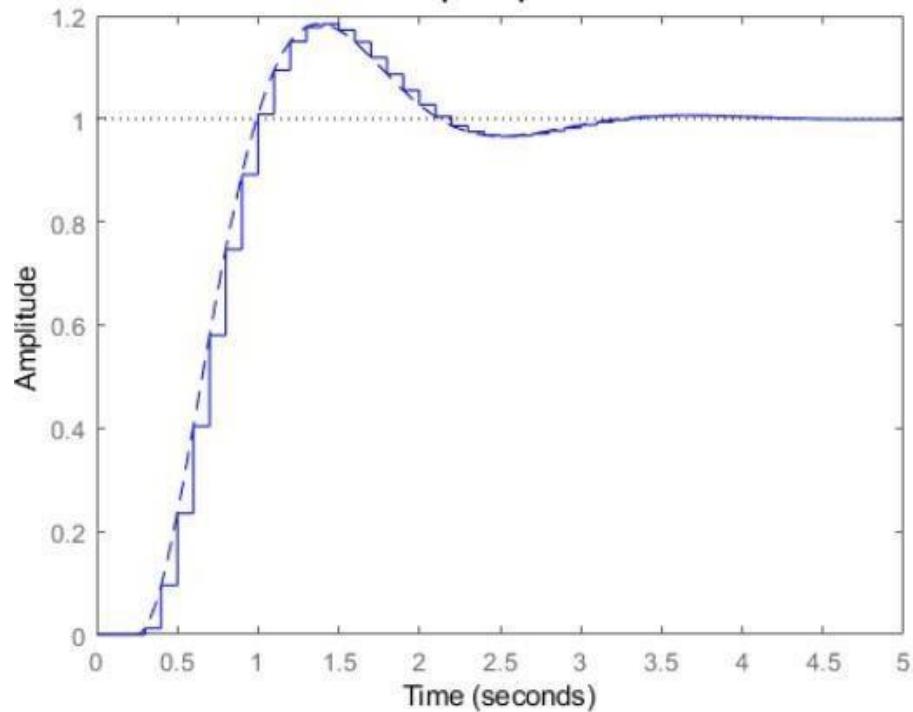


Figure-8.3: output response of transfer function

Example 4: Given a continuous-time state-space model with the state-space matrices:

$$A = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C = [1 \ 1] \quad D = [0]$$



Convert it into discrete-time state-space model, using MATLAB.

Take sample time of 1 second.

MATLAB Solution:

```
a=[0 1; -1 -2];
```

```
b=[0;1]; c=[1  
1]; d=0;
```

```
sys_c=ss(a,b,c,d) % creates/stores continuous-time state-space model ts=1;
```

```
sys_d=c2d(sys_c,ts,'zoh') % gives discrete-time state-space model
```

Result

```
sys_c = a  
= x1 x2  
x1 0 1 x2 -  
1 -2
```

$b = u_1$

$x_1 \quad 0 \quad x_2 \quad 1$

$c = x_1$

x_2

$y_1 \quad 1 \quad 1$

$d = u_1$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

y1 0

Continuous-time state-space model.

sys_d =

a =

x1 x2

x1 0.7358 0.3679 x2 -

0.3679 1.92e-17

b =

u1

x1 0.2642

x2 0.3679

c =

x1 x2

y1 1 1 d =

u1

y1 0 Sample time: 1

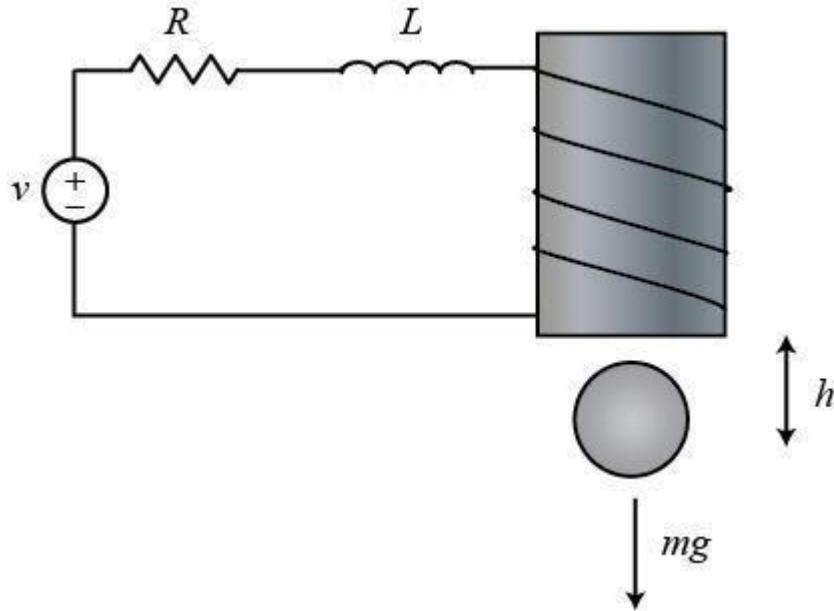
seconds

Discrete-time state-space model.

8.1 Example 5: To introduce the state-space control design method, we will use the magnetically suspended ball as an example. The current through the coils induces a magnetic force which can balance the force of gravity and cause the ball (which is made of a magnetic material) to be



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)



suspended in mid-air.

Figure-8.4: Magnetically suspended ball

We linearize the equations about the point $h = 0.01$ m (where the nominal current is about 7 Amps) and obtain the linear state-space equations:

$$\frac{d\mathbf{x}}{dt} = A\mathbf{x} + Bu$$

$$y = C\mathbf{x} + Du$$

$$\mathbf{x} = \begin{bmatrix} \Delta h \\ \Delta \dot{h} \\ \Delta i \end{bmatrix}$$

Here the values of $A = [0 \quad 1 \quad 0 \\ 980 \quad 0 \quad -2.8 \\ 0 \quad 0 \quad -100]$
 $B = [0 \\ 0 \\ 100]$
 $C = [1 \ 0 \ 0]$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

MATLAB Solution:

```
A = [ 0 1 0 ; 980 0 -2.8 ; 0 0 -100 ];  
B = [ 0; 0 ;100 ];  
C = [ 1 0 0 ];
```

8.1.1 Stability Analysis

One of the first things we want to do is analyze whether the open-loop system (without any control) is stable. As discussed in the Introduction: System Analysis section, the eigenvalues of the system matrix, , (equal to the poles of the transfer function) determine stability. The eigenvalues of the matrix are the

values of that are solutions of $\det(sI - A) = 0$.

```
Poles= eig(A) poles
```

```
=
```

```
31.3050
```

```
-31.3050
```

```
-100.0000
```

From inspection, it can be seen that one of the poles is in the right-half plane (i.e. has positive real part), which means that the open-loop system is unstable.

To observe what happens to this unstable system when there is a non-zero initial condition, add the following lines to your m-file and run it again:

```
t = 0:0.01:2; u =  
zeros(size(t)); x0  
= [0.01 0 0]; sys  
= ss(A,B,C,0);  
[y,t,x] = lsim(sys,u,t,x0);  
plot(t,y)  
title('Open-Loop Response to Non-Zero Initial Condition')  
xlabel('Time (sec)')  
ylabel('Ball Position (m)')
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

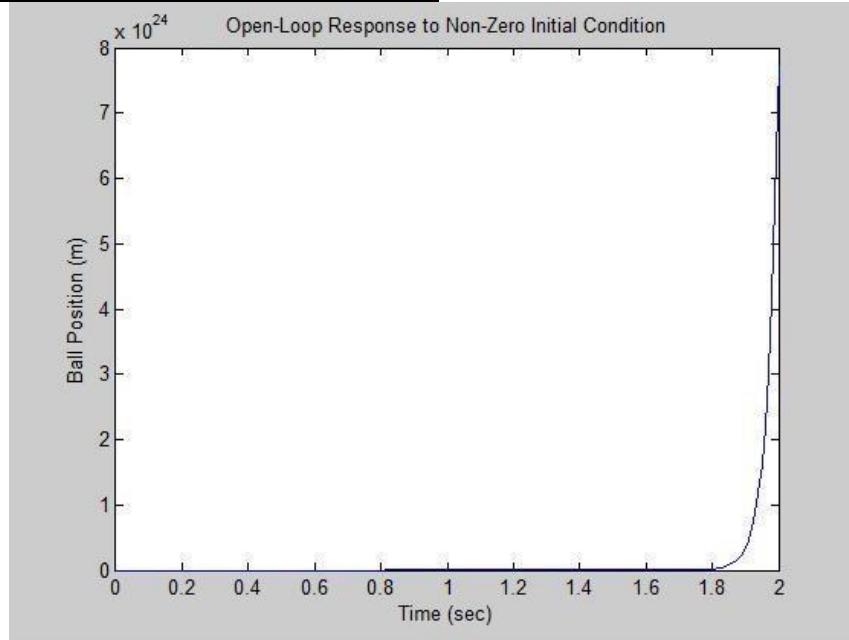


Figure-8.5: output response

It looks like the distance between the ball and the electromagnet will go to infinity, but probably the ball hits the table or the floor first (and also probably goes out of the range where our linearization is valid).

8.2 Review Exercise?

a) What is state space representation of Discrete-Time system?

The dynamics of a linear time (shift) invariant discrete-time system may be expressed in terms state (plant) equation and output(observation or measurement) equation as follows: $x(k+1) = Ax(k)+Bu(k)$ and $y(k) = Cx(k)+Du(k)$ where $x(k)$ an n dimensional slate rector a time $t=kT$ an r-dimensional control (input) vector and $y(k)$ an mdimensional output vector respectively are represented as $x(k)=[x_1(k), x_2(k), x_3(k), \dots, x_n(k)]^T$, $u(k)=[u_1(k), u_2(k), u_3(k), u_4(k), \dots, u_r(k)]^T$, $y(k)=[y_1(k), y_2(k), y_3(k), y_4(k), \dots, y_m(k)]^T$.

b) What is a state model and what are state variables?

State model: The state of a dynamic system is the smallest set of variables and the knowledge of these variables at $t=10$ together with inputs for $t \geq 10$ completely determines the behaviour of the system at $t \geq 10$. A compact and concise representation of the past history of the system can be termed as the state of the system.

State Variable: The smallest set of variables that determine the state of the system are known as state variable.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

c) What are the advantages of state space techniques?

This technique can be used for linear and non-linear time-variant and time invariant systems.

- It is easier to apply where Laplace transforms cannot be applied.
- The nth order differential equation can be expressed as 'n' equation of first order.
- It is time domain method.

d) A continuous-time state-space model of Mass-Spring-Damper is given as follows:

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} F(t)$$

$$y = [1 \ 0] \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$$

Where $m = 1$; $b = 10$; $k = 20$.

Using MATLAB, obtain its discrete-time state-space model. Take sample time of 1/100 seconds.

Solution:

```
4 - m = 1;
5 - b = 10;
6 - k = 20;
7 - a=[0 1; -k/m -b/m];
8 - b=[0;1/m];
9 - c=[1 0];
10 - d=[0];
11 - sys_c=ss(a,b,c,d) % creates/stores continuous-time state-space model
12 - ts=1/100;
13 - sys_d=c2d(sys_c,ts,'zoh') % gives discrete-time state-space model
14 -
15 |
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Result:

FILE NAVIGATE EDIT BREAKPOINTS

Command Window

New to MATLAB? See resources for [Getting Started](#).

```
ts=1/100;
sys_d=c2d(sys_c,ts,'zoh') % gives discrete-time state-space model

sys_c =
 
A =
    x1    x2
x1    0    1
x2   -20   -10

B =
    u1
x1    0
x2    1

C =
    x1    x2
y1    1    0

D =
    u1
y1    0

Continuous-time state-space model.
```

fx

FILE NAVIGATE EDIT BREAKPOINTS

Command Window

New to MATLAB? See resources for [Getting Started](#).

```
sys_d =
 
A =
    x1      x2
x1    0.999  0.009513
x2   -0.1903  0.9039

B =
    u1
x1  4.837e-05
x2   0.009513

C =
    x1    x2
y1    1    0

D =
    u1
y1    0

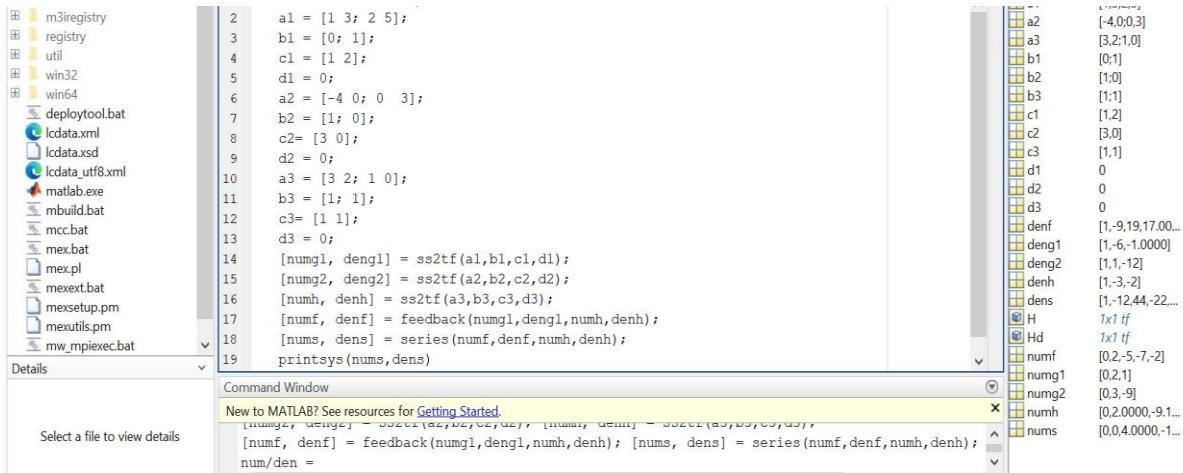
Sample time: 0.01 seconds
Discrete-time state-space model.
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

- e) Attach the plots and results of example1 to example5. Also explain example 5 briefly

Example 1



The screenshot shows the MATLAB workspace and command window. The workspace contains the following variables and their values:

Variable	Value
a2	[-4,0,3]
a3	[3,2;1,0]
b1	[0,1]
b2	[1,0]
b3	[1,1]
c1	[1,2]
c2	[3,0]
c3	[1,1]
d1	0
d2	0
d3	0
denf	[1,-9,19,17,00...]
deng1	[1,-6,-1,0000]
deng2	[1,1,-12]
den1	[1,-3,-2]
dens	[1,-12,44,-22,...]
H	1x1 tf
Hd	1x1 tf
numf	[0,2,-5,-7,-2]
numg1	[0,2,1]
numg2	[0,3,-9]
numh	[0,2,0000,-9,1...
nums	[0,0,4,0000,-1...

The command window displays the following MATLAB code:

```
a1 = [1 3; 2 5];
b1 = [0; 1];
c1 = [1 2];
d1 = 0;
a2 = [-4 0; 0 3];
b2 = [1; 0];
c2= [3 0];
d2 = 0;
a3 = [3 2; 1 0];
b3 = [1; 1];
c3=[ 1 1];
d3 = 0;
[numg1, deng1] = ss2tf(a1,b1,c1,d1);
[numg2, deng2] = ss2tf(a2,b2,c2,d2);
[numh, denh] = ss2tf(a3,b3,c3,d3);
[numf, denf] = feedback(numg1,deng1,numh,denh);
[nums, dens] = series(numf,denf,numh,denh);
printsys(nums,dens)
```

Example 2



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

The screenshot shows the MATLAB Command Window with the following code and output:

```
8 c2= [3 0];
9 d2 = 0;
10 a3 = [3 2; 1 0];
11 b3 = [1; 1];
12 c3= [1 1];
13 d3 = 0;
14 [numg1, denq1] = ss2tf(a1,b1,c1,d1);
15 [numg2, denq2] = ss2tf(a2,b2,c2,d2);
16 [numh, denh] = ss2tf(a3,b3,c3,d3);
17 [numf, denf] = feedback(numg1,deng1,numh,denh);
18 [nums, dens] = series(numf,denf,numh,denh);
19 printsys(nums,dens)

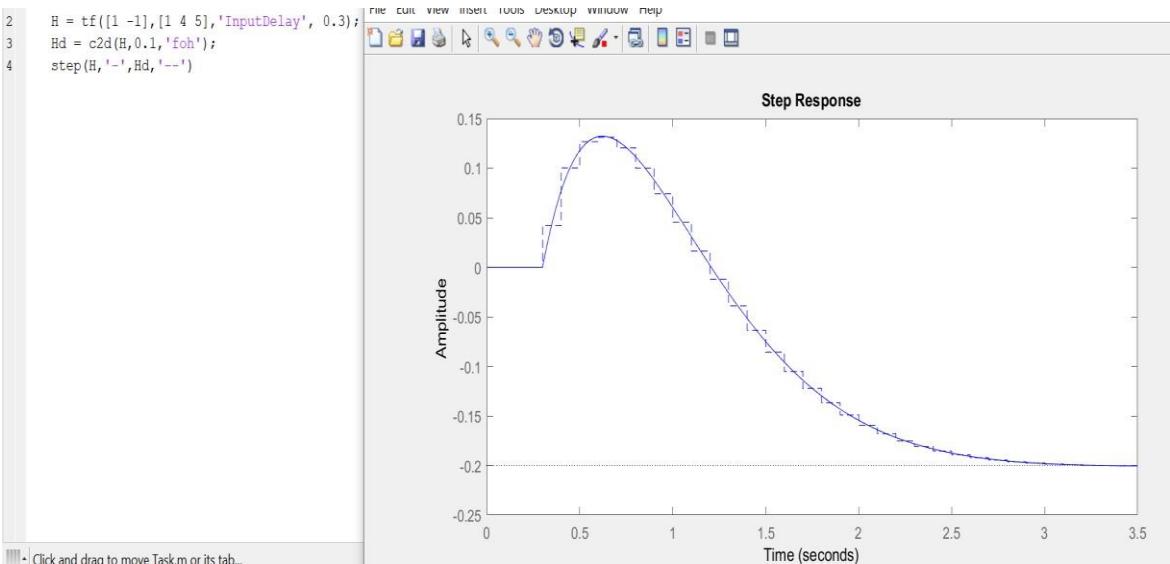
Command Window
New to MATLAB? See resources for Getting Started.
[numr, denr] = feedback(numg1,deng1,numh,denh);
[nums, dens] = series(numf,denf,numh,denh);
printsys(nums,dens)

num/den =
4 s^4 - 10 s^3 - 14 s^2 - 4 s + 1.8395e-16
-----
s^6 - 12 s^5 + 44 s^4 - 22 s^3 - 87 s^2 - 40 s - 4
fx>> H = tf([1 -1],[1 4 5], 'InputDelay', 0.3);
Hd = c2d(H,0.1,'foh');
step(H, '--', Hd, '--')
```

The right side of the window shows the workspace variables and their values:

a1	[1,3;2,5]
a2	[-4:0;0,3]
a3	[3,2;1,0]
b1	[0;1]
b2	[1;0]
b3	[1;1]
c1	[1,2]
c2	[3,0]
c3	[1,1]
d1	0
d2	0
d3	0
denf	[1,-9,19,17,00...]
deng1	[1,-6,-1.0000]
deng2	[1,1,-12]
denh	[1,-3,-2]
dens	[1,-12,44,-22...]
H	1x1 tf
Hd	1x1 tf
numf	[0,2,-5,-7,-2]
numg1	[0,2,1]
numg2	[0,3,-9]
numh	[0,2,0000,-9,1...
nums	[0,0,4,0000,-1...

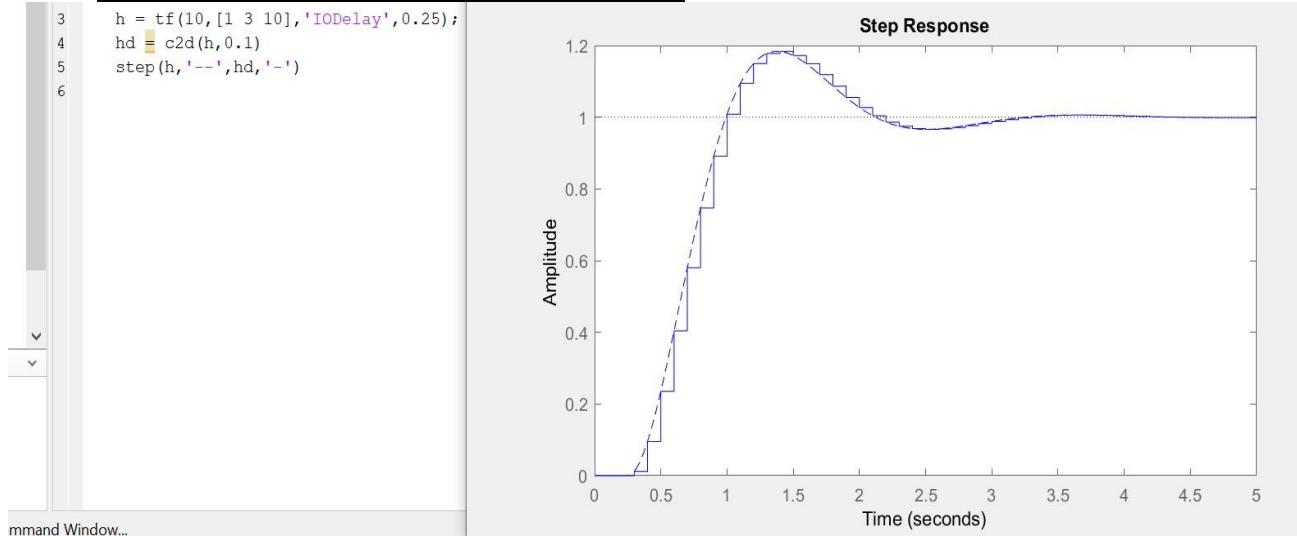
Example 3



Example 4



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)



```
3 a=[0 1; -1 -2];
4 b=[0;1];
5 c=[1 1];
6 d=0;
7 sys_c=ss(a,b,c,d) % creates/stores continuous-time state-space model
8 ts=1;
9 sys_d=c2d(sys_c,ts,'zoh') % gives discrete-time state-space model
10
```

Command Window

New to MATLAB? See resources for [Getting Started](#).

```
>> a=[0 1; -1 -2];
b=[0;1];
c=[1 1];
d=0;
sys_c=ss(a,b,c,d) % creates/stores continuous-time state-space model
ts=1;
sys_d=c2d(sys_c,ts,'zoh') % gives discrete-time state-space model

sys_c =

$$A = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}$$


```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Command Window

New to MATLAB? See resources for [Getting Started](#).

```
sys_c =  
  
A =  
    x1   x2  
x1    0    1  
x2   -1   -2  
  
B =  
    u1  
x1    0  
x2    1  
  
C =  
    x1   x2  
y1    1    1  
  
D =  
    u1  
y1    0  
  
Continuous-time state-space model.
```

Command Window

New to MATLAB? See resources for [Getting Started](#).

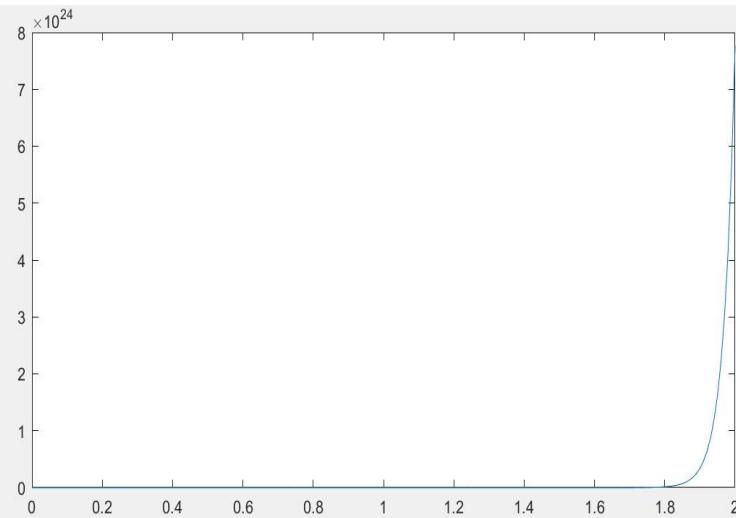
```
sys_d =  
  
A =  
    x1      x2  
x1    0.7358    0.3679  
x2   -0.3679     0  
  
B =  
    u1  
x1  0.2642  
x2  0.3679  
  
C =  
    x1   x2  
y1    1    1  
  
D =  
    u1  
y1    0  
  
Sample time: 1 seconds  
Discrete-time state-space model.  
fx ...
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Example 5

```
3 A = [ 0 1 0 ; 980 0 -2.8 ; 0 0 -100 ];
4 B = [ 0; 0 ;100 ];
5 C = [ 1 0 0 ];
6 Poles=eig(A)
7 poles =
8 -31.3050
9 100.0000
10 t = 0:0.01:2;
11 u = zeros(size(t));
12 x0 = [0.01 0 0];
13 [y,t,x] = lsim(sys,u,t,x0);
14 plot(t,y)
```



It is shown that the distance between the ball and the electromagnet goes to infinity, but probably the ball hits the table or the floor first (and also probably goes out of the range where our linearization is valid).



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)



**LAB # 09 DETERMINING THE CONTROLLABILITY AND OBSERVABILITY OF A SYSTEM
IN DISCRETE TIME DOMAIN**

Name: _____ KARAN	Roll # _____ 20ES062
Date: _____	
Signature of Lab Tutor:	

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	Determining the Controllability and Observability of a system in Discrete-Time domain.	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)



Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

Equipment Required: PC with MATLAB software

Approximate Time Required: Two to three hours

Introduction

Controllability and observability are two important properties of state models which are to be studied prior to designing a controller. Controllability deals with the possibility of forcing the system to a particular state by application of a control input. If a state is uncontrollable then no input will be able to control that state. On the other hand whether or not the initial states can be observed from the output is determined using observability property. Thus if a state is not observable then the controller will not be able to determine its behavior from the system output and hence not be able to use that state to stabilize the system.

Controllability is an important property of a control system, and the Controllability property plays a crucial role in many control problems, such as stabilization of unstable systems by feedback, or optimal control. Controllability and observability are dual aspects of the same problem.

Controllability:

A system is completely controllable if there exists an unconstrained control $u(t)$ that can transfer any initial state $x(t_0)$ to any other desired location $x(t)$ in a finite time, $t_0 \leq t \leq T$.

Observability:

A system is completely observable if and only if there exists a finite time T such that the initial state $x(0)$ can be determined from the observation history $y(t)$ given the control $u(t)$, $0 \leq t \leq T$.

Controllability

To determine Controllability of a system, consider a dynamical system given as

$$\begin{aligned} \mathbf{x}(k+1) &= A\mathbf{x}(k) + B\mathbf{u}(k) \\ \mathbf{y}(k) &= C\mathbf{x}(k) + D\mathbf{u}(k) \end{aligned} \tag{9.1}$$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

The state equation (9.1) or the pair (A, B) is controllable if and only if the $n \times nm$ Controllability matrix:

$$S = [B \ AB \ A^2B \ \dots \ A^{n-1}B]$$

has rank n , i.e., full row rank.

In MATLAB we can obtain the Controllability matrix S by using built-in function `ctrb`.

In order to check whether the given system is controllable or not, we compare the rank of Controllability matrix S with the length of state matrix A . The system is said to be controllable if

$$\text{rank}(S) = \text{length}(A)$$

Observability

The state equation (9.1) or the pair (A, C) is observable if and only if the $np \times n$ observability matrix:

$$V = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

has rank n , i.e., full column rank.

In MATLAB we can obtain the observability matrix V by using built-in function `obsv`.

In order to check whether the given system is observable or not, we compare the rank of observability matrix V with the length of state matrix A . The system is said to be observable if

$$\text{rank}(V) = \text{length}(A)$$

Determining System Controllability & Observability in discrete-time domain using MATLAB

Example 1: Given the state-space matrices of a discrete-time state-space model with a sample time of 0.25 seconds:

$$A = \begin{bmatrix} 0 & 1 \\ -5 & -2 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 3 \end{bmatrix} \quad C = \begin{bmatrix} 0 & 1 \end{bmatrix} \quad D = [0]$$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)



Determine the system Controllability & observability in discrete-time domain, using MATLAB.

MATLAB Code

```
A = [0 1;-5 -2];
B = [0;3];
C=[0 1];
D=0; Ts = 0.25; sys = ss(A,B,C,D,Ts)%creates/stores discrete-
time state-space
%model LengthofA=
length(sys.A) s=ctrb(sys)%gives
controllability matrix RankofS= rank(s)
v=obsv(sys)%gives observability matrix
RankofV= rank(v)
```

Resul

```
t sys =
a =
x1 x2
x1 0 1 x2 -
5 -2
```

```
b=u1 x1 0
x2 3
```

```
c =
x1 x2
y1 0 1
```

```
d=u1
y1 0
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Sample time: 0.25 seconds

Discrete-time state-space model.

LengthofA = 2 s =

0 3 3 -6

Rankof

S = 2 v

=

0 1

-5 -2

RankofV = 2

Hence, the given system is controllable as well as observable in discrete-time domain.

Example 2: Given the state-space matrices of a continuous-time state-space model:

$$A = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C = [1 \ 1] \quad D = [0]$$

Determine the system Controllability & observability in discrete-time domain, using MATLAB.

Take sample time of 1 second.

MATLAB Code

```
a=[0 1; -1 -  
2]; b=[0;1];  
c=[1 1]; d=0;  
sys_c=ss(a,b,c,d) % creates/stores continuous-time state-space model ts=1;  
sys_d=c2d(sys_c,ts,'zoh') % gives discrete-time state-space model  
lengthofA=length(sys_d.a) s=ctrb(sys_d)  
%gives controllability matrix
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

rankofS=rank(s) v=obsv(sys_d) %gives

observability matrix

rankofV=rank(v)

Result

sys_c = a

= x1 x2

 x1 0 1

 x2 -1 -2

b = u1

x1 0 x2 1

c = x1

 x2

y1 1 1

d = u1

y1 0

Continuous-time state-space model.

sys_d =

a =

 x1 x2

 x1 0.7358 0.3679 x2 -

 0.3679 1.92e-17

b =

 u1

 x1 0.2642

 x2 0.3679

c = x1 x2 y1 1 1 d = u1



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

y1

0

Sample time: 1 seconds

Discrete-time state-space

model. lengthofA = 2 s =

0.2642 0.3298

0.3679 -0.0972

rankofS = 2 v

=

1.0000 1.0000

0.3679 0.3679 rankofV =

1

Hence, the given system is controllable but not observable in discrete-time domain.

Review Exercise?

a. What is Controllability?

A system is completely controllable if there exists an unconstrained control $u(t)$ that can transfer any initial state $x(0)$ to any other desired location $x(t)$ in a finite time, $0 \leq t \leq T$. Controllability is an important property of a control system, and the Controllability property plays a crucial role in many control problems, such as stabilization of unstable systems by feedback, or optimal control. Controllability and observability are dual aspects of the same problem.

b. What is observability?

A system is completely observable if and only if there exists a finite time T such that the initial state $x(0)$ can be determined from the observation history $y(t)$ given the control $u(t)$, $0 \leq t \leq T$.

c. Given the state-space matrices of a continuous-time state-space model:

$$A = \begin{bmatrix} -5 & -1 \\ 3 & -1 \end{bmatrix} \quad B = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad C = [1 \ 2] \quad D = [0]$$



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Determine the system Controllability & observability in discrete-time domain, using MATLAB.
Take sample time of 1/100 seconds. **Program:**

```
A = [-5 -1;3 -1];
B = [1;0];
C=[1 2]; D=0; Ts
= 1/100;
sys = ss(A,B,C,D,Ts)%creates/stores discrete-time state-space
%model
LengthofA= length(sys.A)
s=cctrb(sys)%gives controllability matrix  RankofS=
rank(s)
v=obsv(sys)%gives observability matrix
RankofV= rank(v)
3- A = [-5 -1;3 -1];
4- B = [1;0];
5- C=[1 2]; D=0;
6- Ts = 1/100;
7- sys = ss(A,B,C,D,Ts)%creates/stores discrete-time state-space
8- %model
9- LengthofA= length(sys.A)
10- s=cctrb(sys)%gives controllability matrix
11- RankofS= rank(s)
12- v=obsv(sys)%gives observability matrix
13- RankofV= rank(v)
14-
15-
16-
```

```
RankofV =
2
```

```
LengthofA =
2

s =
1 -5
0 3

RankofS =
2

v =
1 2
1 -3

RankofV =
2
```

```
sys =
A =
x1 x2
x1 -5 -1
x2 3 -1

B =
u1
x1 1
x2 0

C =
x1 x2
y1 1 2

D =
u1
y1 0

Sample time: 0.01 seconds
Discrete-time state-space model.
```



DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

LAB # 10: FREQUENCY RESPONSE AND CONSTRUCTION OF BODE PLOT

Name: <u>KARAN</u>	Roll #: <u>20ES062</u>
Date: _____	
Signature of Lab Tutor:	

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	<i>To analyse the frequency response and construction of bode plot using MATLAB</i>	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	



DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	
Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

Equipment Required: Computer with MATLAB

Approximate Time Required: Two to three hours.

Introduction

Bode plot is a graph of the frequency response of a system. It is usually a combination of a Bode magnitude plot, expressing the magnitude of the frequency response and a Bode phase plot, expressing the phase shift.

We are interested in the frequency response of an LTI system. The transfer function can be written like this:

$$H(s) = K \frac{(s - z_1)(s - z_2)\dots}{(s - p_1)(s - p_2)(s - p_3)\dots}$$

such that when we plug in $j\omega$ for s, we get

$$H(j\omega) = K \frac{(j\omega - z_1)(j\omega - z_2)\dots}{(j\omega - p_1)(j\omega - p_2)(j\omega - p_3)\dots}$$

That's a product (or quotient) of a bunch of complex numbers. Using polar form, we can say that the angle of the product (quotient) is the sum of the angles of each term (except for division we subtract, so it's the sum of the angles for the top terms, minus the sum of the angles for the terms in the denominator). Similarly, the magnitude is the product of the magnitude of all the terms. Summing terms is easy to do graphically; products are harder. However, on a log scale (e.g., dB), the product turns into a sum. Thus, if we plot the behavior of each term, we can then simply add the plots to find the total behavior.

For the poles, we could either plot the behavior of $(s - p)$ and subtract it, or plot the behavior of $1/(s - p)$ and add that behavior. We'll plot the behavior of $1/(s - p)$, such that we only need to add terms.



DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

The general plan for how to sketch a Bode plot by hand is, then, to first gain an understanding of what individual poles and individual zeros do, and then add the responses together. It is easiest to understand complex poles and zeros by looking at the response of a complex conjugate pair, rather than trying to look at the complex poles or zeros individually

Effect of Constant Terms:

Constant terms such as K contribute a straight horizontal line of magnitude $20 \log_{10}(K)$

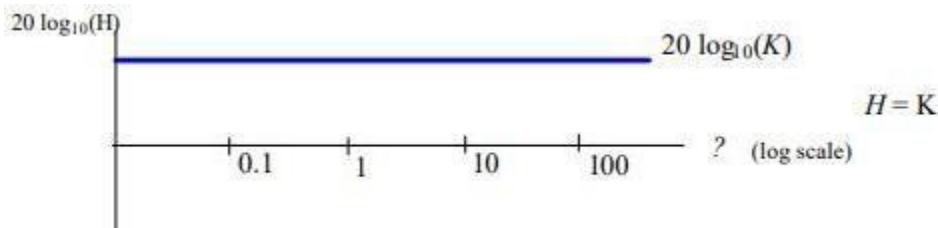


Figure-10.1: Effect of Constant Term

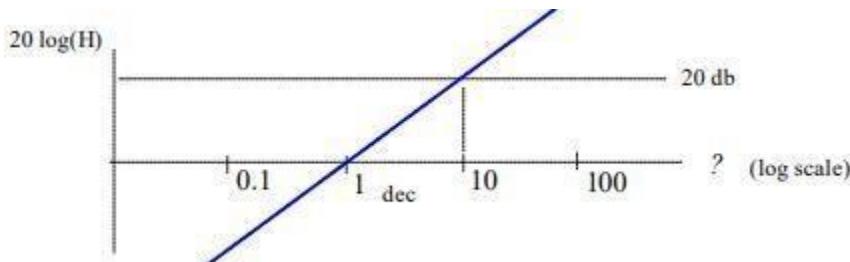
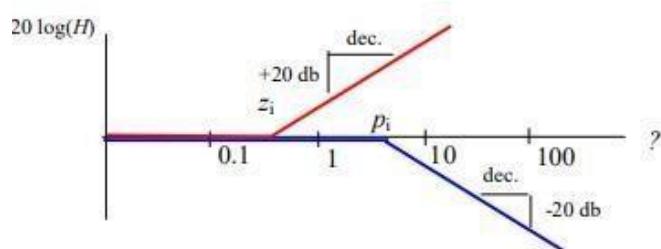


Figure-10.2: effect of pole and zero at origin

Effect of Individual Zeros and Poles Not at the Origin Zeros and Poles not at the origin are indicated by the $(T + \frac{1}{z_i s})$ and $(T + \frac{1}{p_i s})$. The values z_i and p_i in each of these expression is called a critical frequency (or break frequency). Below their critical frequency these terms do not contribute to the log magnitude of the overall plot. Above the critical frequency, they represent a ramp function of 20 db per decade. Zeros give a positive slope. Poles produce a negative slope.





DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

Figure-10.3 Effect of Pole and Zero at origin

To complete the log magnitude vs. frequency plot of a Bode diagram, we superposition all the lines of the different terms on the same plot.

10.1 Technique to get started:

- 1) Draw the line of each individual term on the graph
- 2) Follow the combined pole-zero at the origin line back to the left side of the graph.
- 3) Add the constant offset, $20 \log_{10}(K)$, to the value where the pole/zero at the origin line intersects the left side of the graph.
- 4) Apply the effect of the poles/zeros not at the origin. working from left (low values) to right (higher values) of the poles/zeros

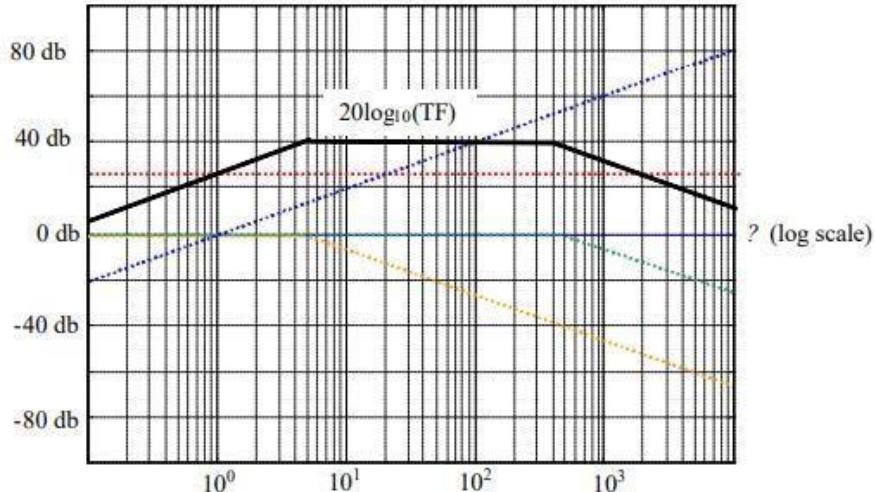


Figure-10.4: bode log plot

Example: Find the bode plot of following system using MATLAB



DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

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

Matlab code

```
% Numerator num =
[2]; % Denominator
den = [1 1]; %
Transfer Function
G = tf(num, den)
% Plot Frequency Response bode(G),
grid
```

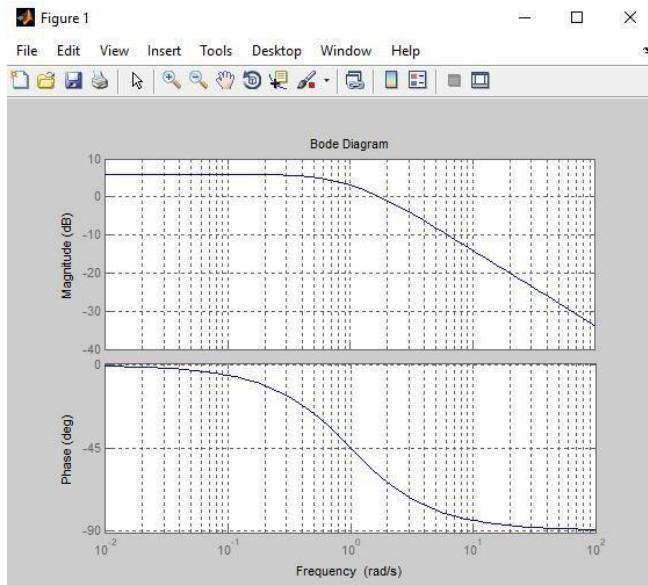


Figure-

10.5: Output waveform

Bode Plot of dynamic system

Create a Bode plot of the following continuous-time SISO dynamic system.

MATLAB Code:

```
H = tf([1 0.1 7.5],[1 0.12 9 0
0]);
bode(H)
```



DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

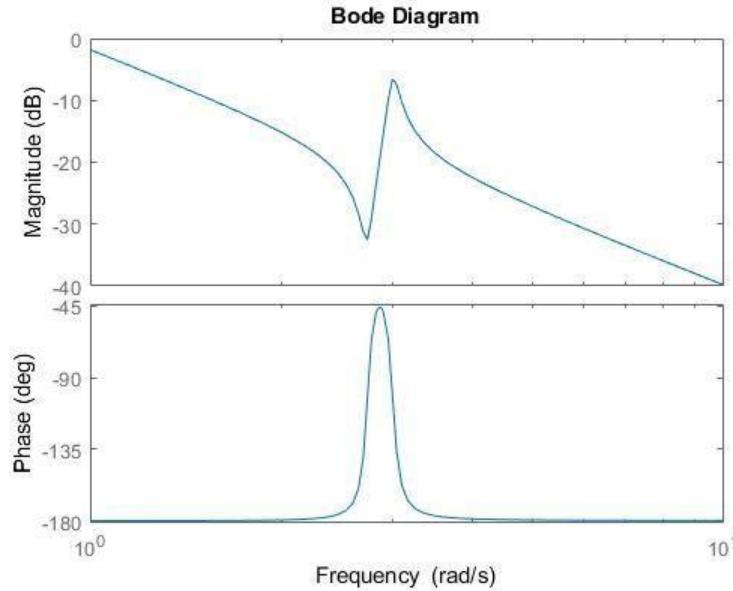


Figure-10.6: Bode Plot of above system

Bode Plot at Specified Frequencies

Create a Bode plot over a specified frequency range. Use this approach when you want to focus on the dynamics in a particular range of frequencies

MATLAB Code:

```
H = tf([-0.1,-2.4,-181,-1950],[1,3.3,990,2600]);  
bode(H,{1,100})  
grid on
```



DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

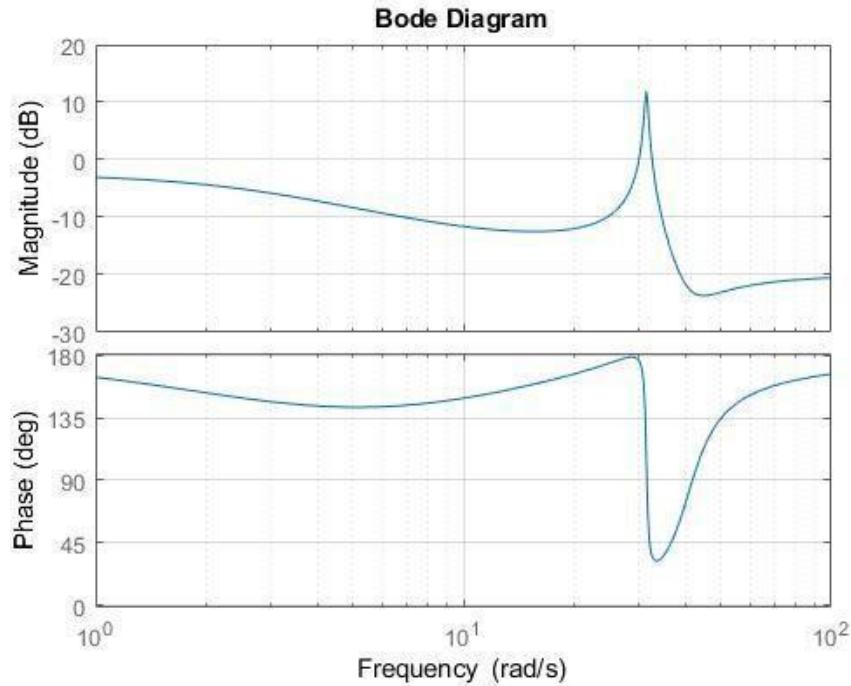


Figure-10.7: Output plot

The cell array `{1,100}` specifies the minimum and maximum frequency values in the Bode plot. When you provide frequency bounds in this way, the function selects intermediate points for frequency response data.

Alternatively, specify a vector of frequency points to use for evaluating and plotting the frequency response.

Create the bode plot at specified frequency points

MATLAB Code: `w = [1 5 10 15 20 23 31 40
44 50 85 100]; bode(H,w,'.-') grid on`



DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

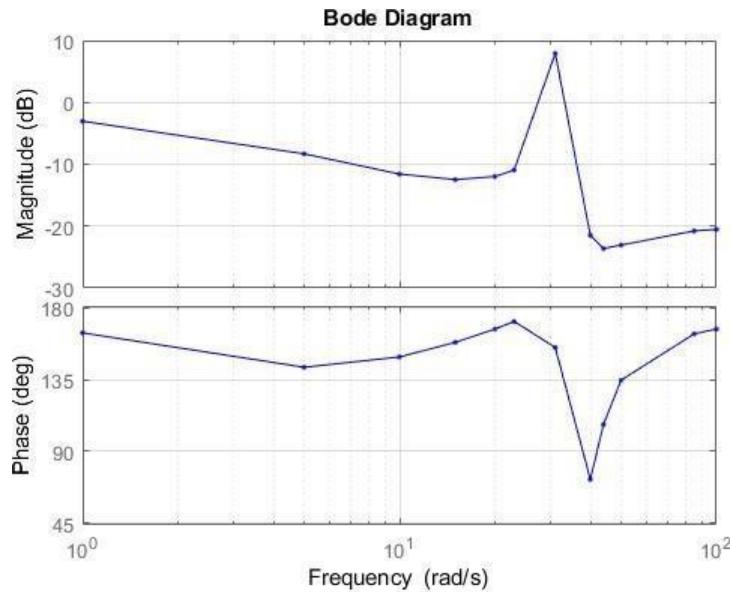


Figure-10.8: Bode Plots the frequency response at the specified frequencies

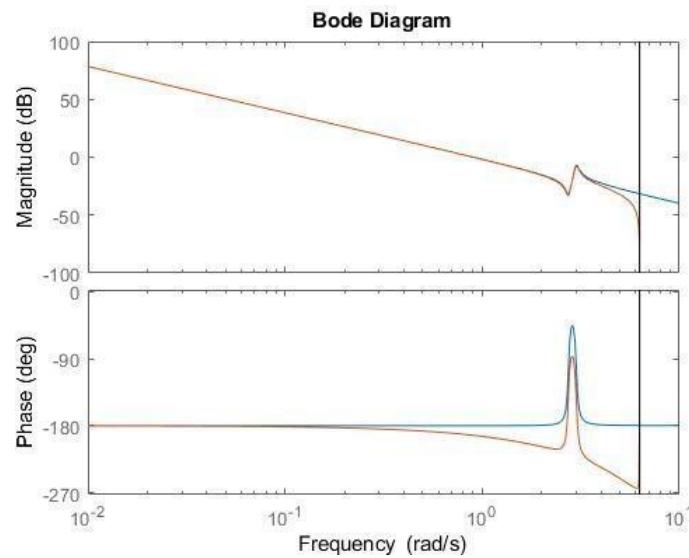
Compare Bode Plots of Several Dynamic Systems

Compare the frequency response of a continuous-time system to an equivalent discretized system on the same Bode plot.

Create continuous-time and discrete-time dynamic systems.

MATLAB Code:

```
H = tf([1 0.1 7.5],[1 0.12 9 0 0]);  
Hd = c2d(H,0.5, 'zoh'); bode(H,Hd)
```





DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

Figure-10.9: Output plot

The Bode plot of a discrete-time system includes a vertical line marking the Nyquist frequency of the system.

Bode Plot with Specified Line Attributes

Specify the line style, color, or marker for each system in a Bode plot using the LineSpec input argument.

MATLAB Code:

```
H = tf([1 0.1 7.5],[1 0.12 9 0 0]);  
Hd = c2d(H,0.5, 'zoh');  
bode(H, 'r', Hd, 'b--')
```

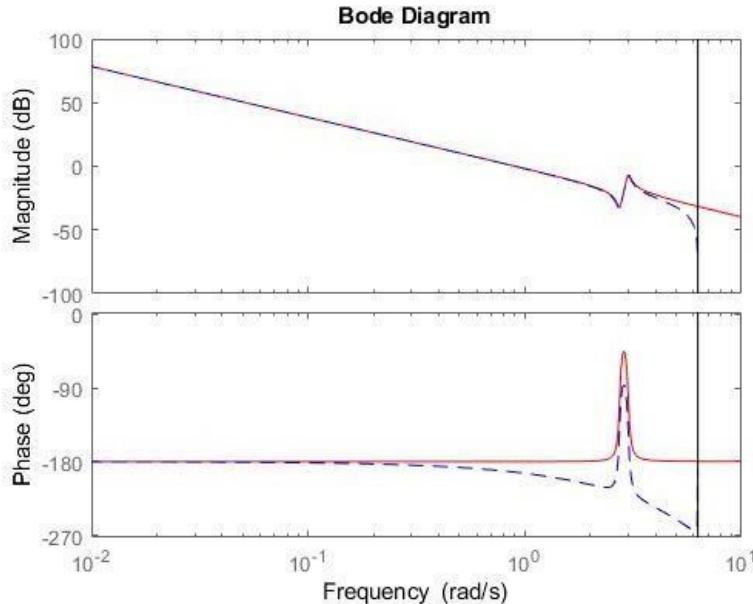


Figure-10.10: Output plot

The first LineSpec, 'r', specifies a solid red line for the response of H. The second LineSpec, 'b--', specifies a dashed blue line for the response of Hd.

Getting Magnitude and Phase Data for a System

The Bode plot of a system, G, shows the magnitude, $G(jw)$ (in dB), and phase $\angle G(jw)$ (degrees) over a range of frequencies. Often it is necessary to access this frequency response data directly, which is accomplished easily using the following commands:

MATLAB Code:

```
win = logspace(-2,2,5)
```



DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

```
[mag,phase,wout] =  
bode(G,win); size(wout)  
size(mag) size(phase)  
Results: win = 0.0100 0.1000 1.0000 10.0000  
100.0000 ans = 5        1 ans = 1        1  
5 ans = 1        1        5
```

where win is the input frequency vector in rad/s, mag is the magnitude in absolute units (i.e. not in dB), phase is the phase in degrees, and wout is the output frequencies at which the response was evaluated, useful for instance if you let MATLAB automatically choose the frequency vector by specifying only the range, $\text{win} = \{\text{wmin}, \text{wmax}\}$.

You'll notice that though wout is a standard ($n \times 1$) column vector, the magnitude and phase are actually ($1 \times 1 \times n$). This is because the `bode` command accepts transfer function matrices in which case the (i,j,k) element of magnitude and phase corresponds to the j th input and i th output. For the SISO systems we most often encounter, $i = j = 1$, these singleton dimensions are an inconvenience for matrix operations, etc. Fortunately MATLAB offers the function, `squeeze`, to fix just this problem:

```
mag = squeeze(mag)  
phase = squeeze(phase)
```

Results:

```
mag = 1.0000 0.9950 0.7071 0.0995 0.0100 phase =  
-0.5729 -5.7106 -45.0000 -84.2894 -89.4271
```

Review exercise?

a) What is bode plot?

Bode plot is a graph of the frequency response of a system. It is usually a combination of a Bode magnitude plot, expressing the magnitude of the frequency response and a Bode phase plot, expressing the phase shift.

b) Why Bode plot is used?

Bode plot is like a map that shows how a system responds to different frequencies. It helps engineers understand and control things like stability, amplification, and filtering in systems like electronic circuits. By looking at a Bode plot, they can make sure a system works well at various frequencies and avoid problems like distortion or instability. It's a useful tool for designing and tweaking systems to do what they're supposed to do.

c) Define phase crossover frequency and gain cross over frequency in Bode plot?



DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

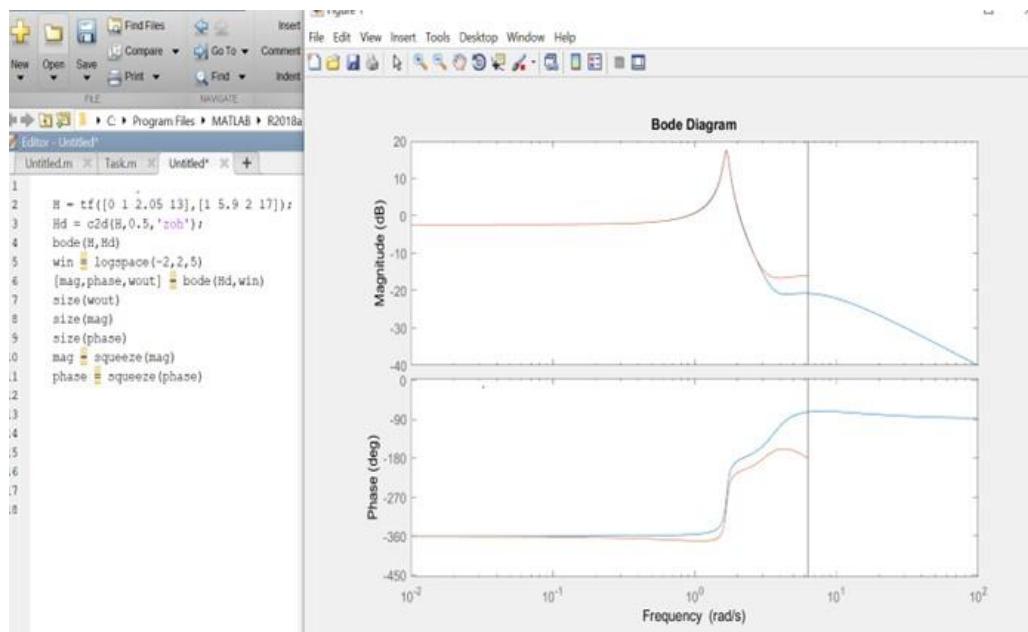
The phase crossover frequency tells us when the phase of the system's response becomes zero degrees, and the gain crossover frequency indicates when the system's gain is unity. These frequencies are key parameters in Bode plots, aiding engineers in analyzing and designing systems, especially in the context of control theory and stability analysis.

- d) Write the MATLAB Code to create the bode plot of continuous time and discrete time system?

$$S^2 + 2.05S + 13 \quad S^3 + 5.9S^2 + 2S + 17$$

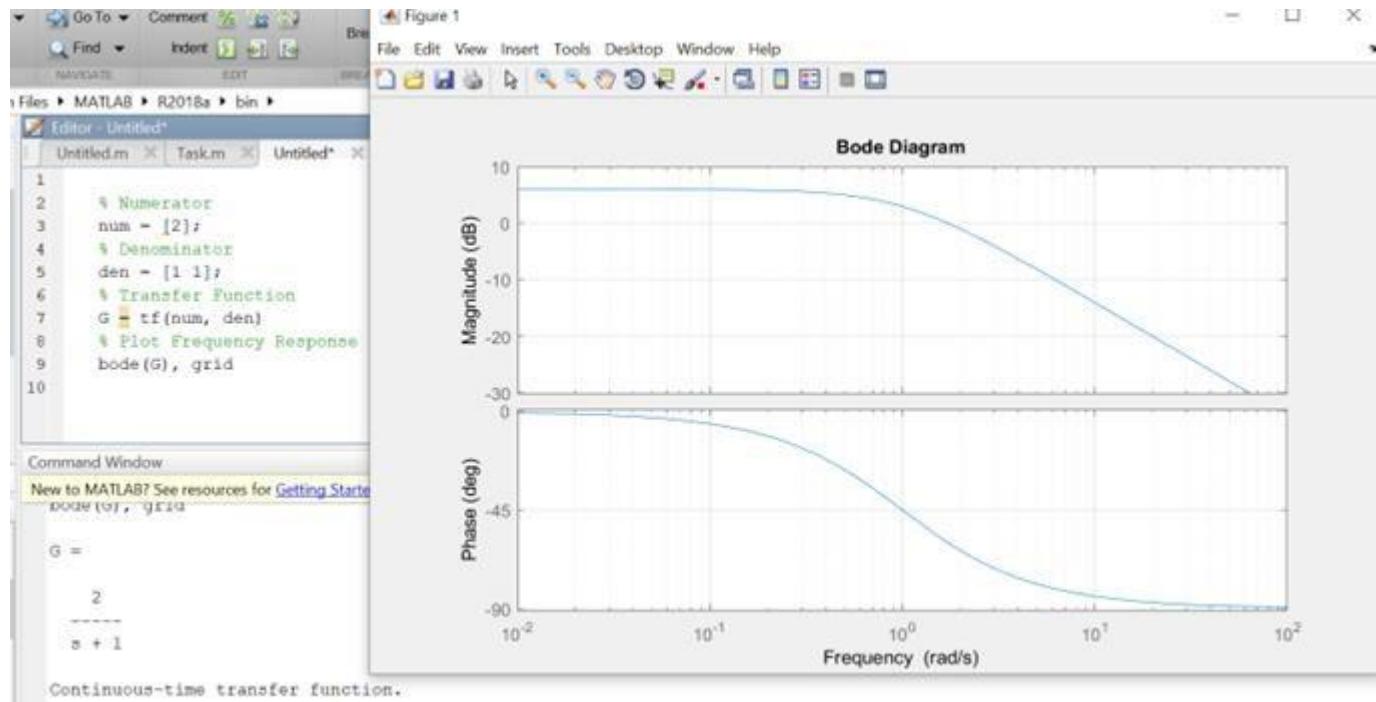
$G(S) = \underline{\hspace{10em}}$

```
H = tf([0 1 2.05 13],[1 5.9 2 17]);
Hd = c2d(H,0.5,'zoh');
bode(H,Hd)
win = logspace(-2,2,5)
[mag,phase,wout] =
bode(Hd,win) size(wout)
size(mag) size(phase) mag =
squeeze(mag) phase =
squeeze(phase)
```

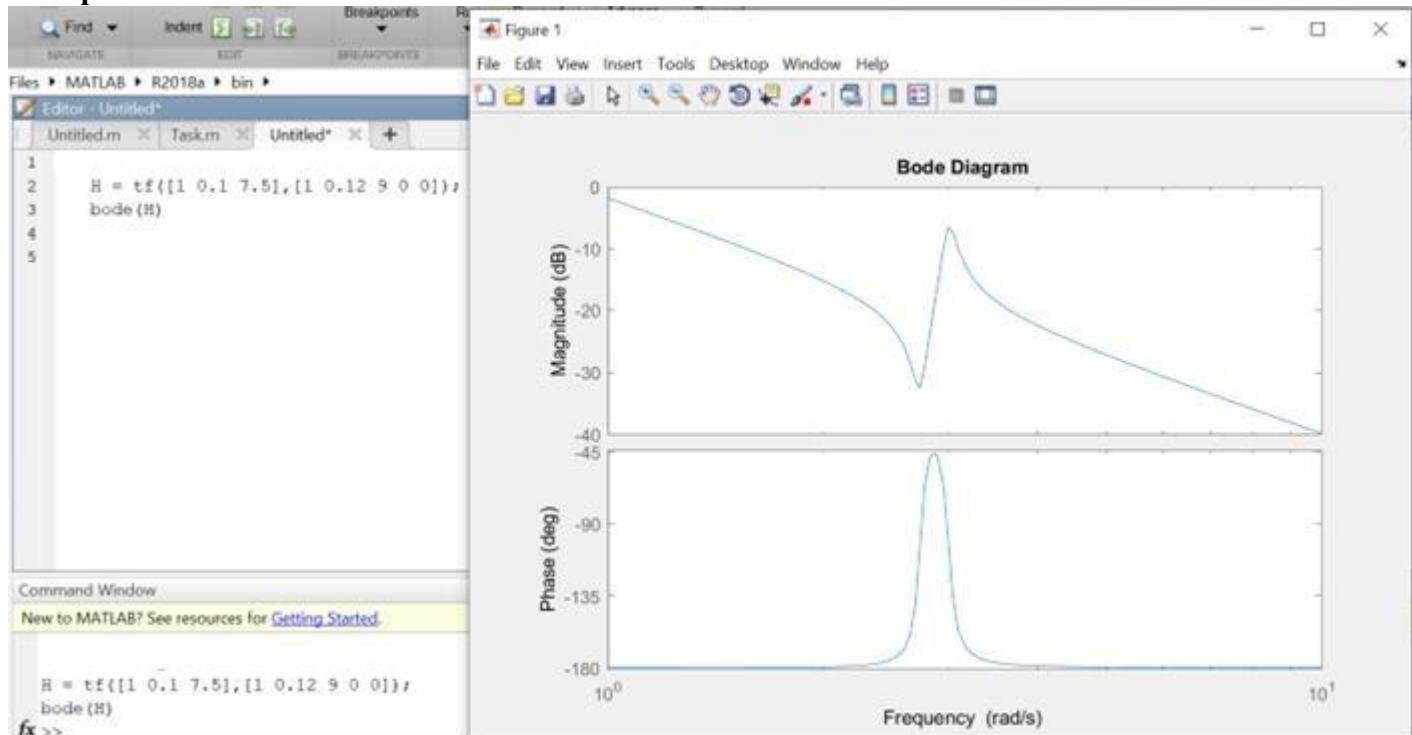




e) Attach the plots and results of all examples done in this laboratory **Example:1**



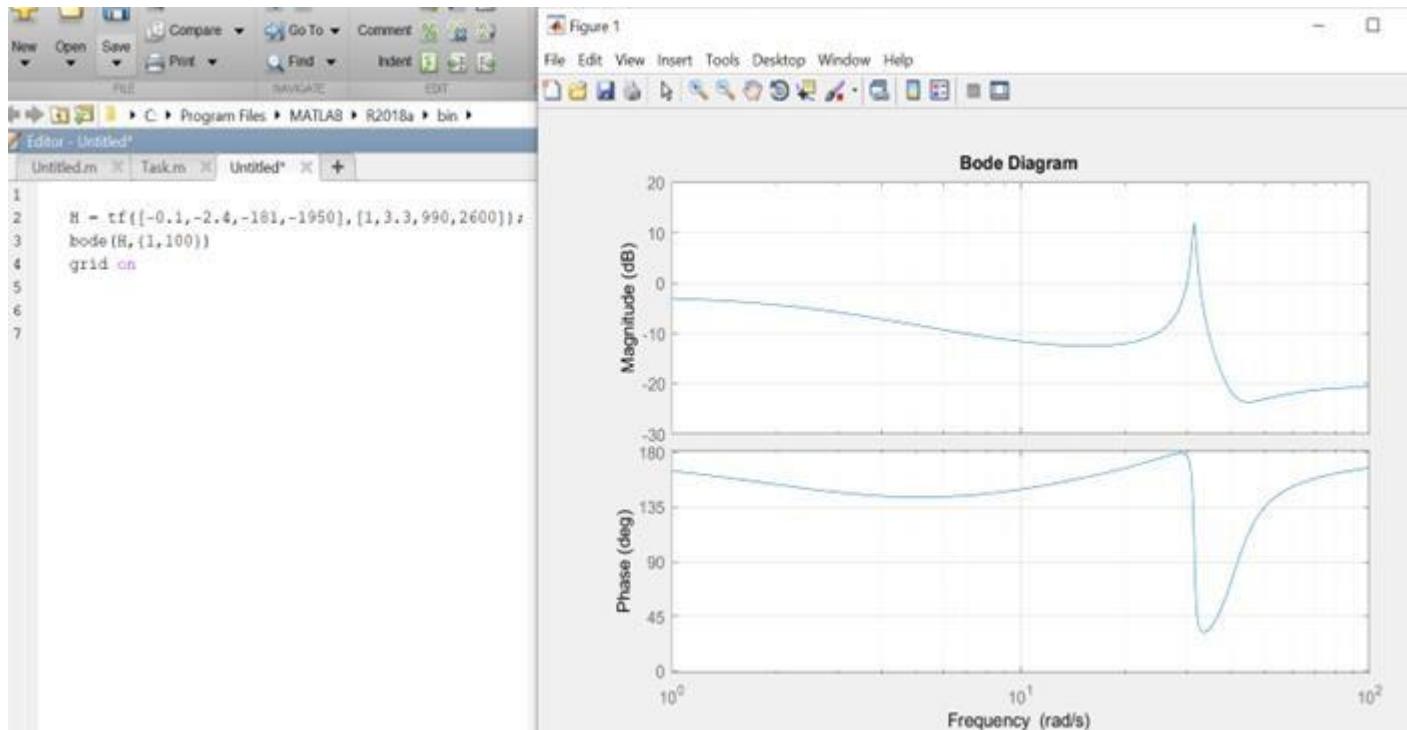
Example: 2



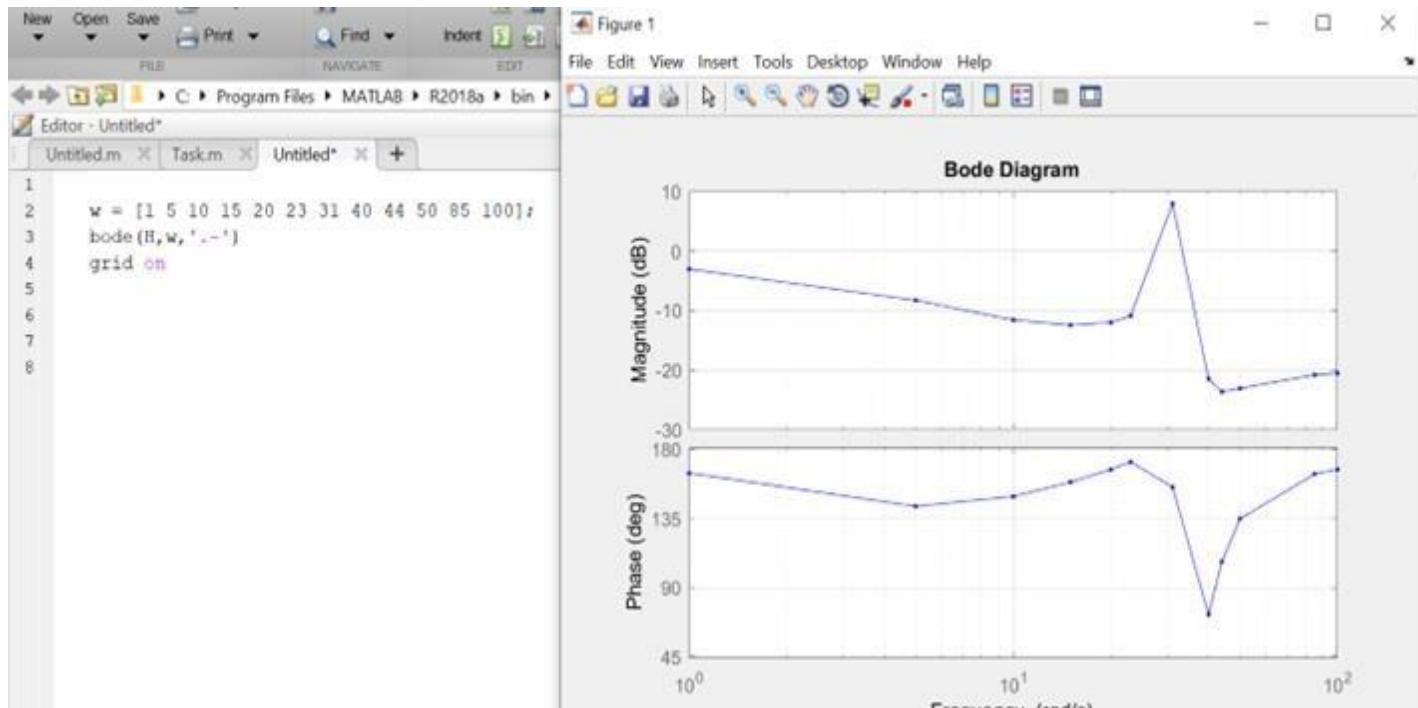
Example: 3



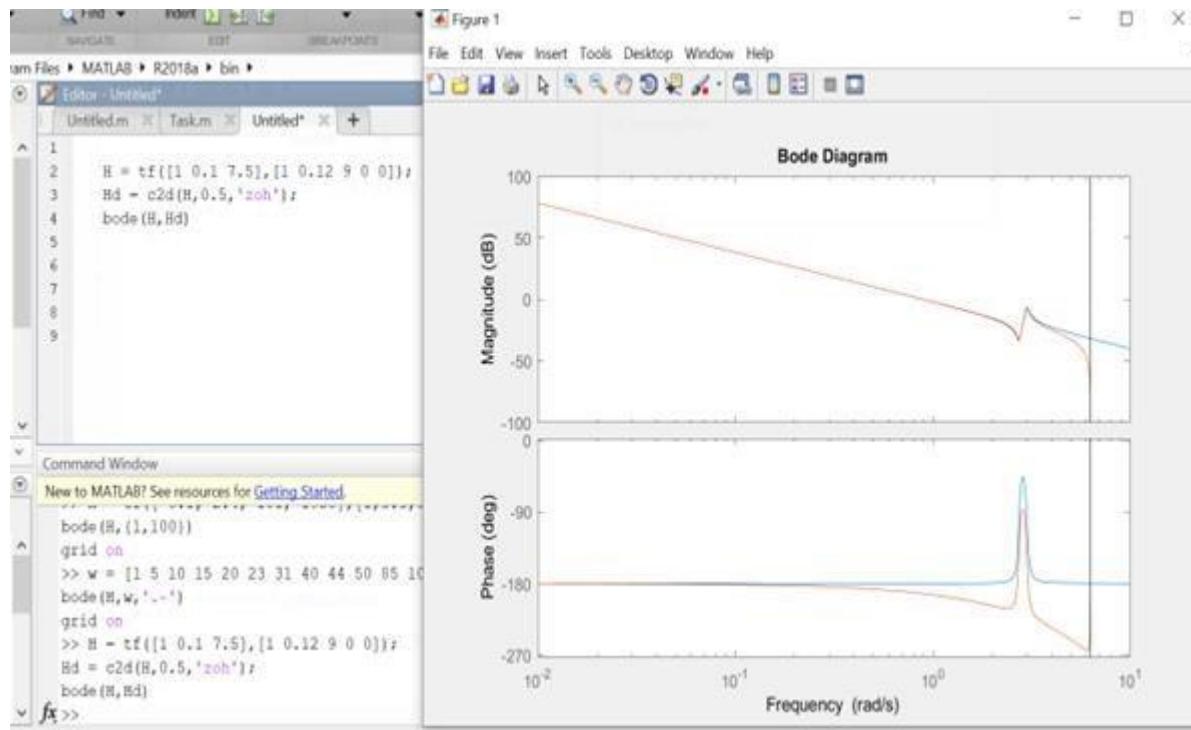
413)



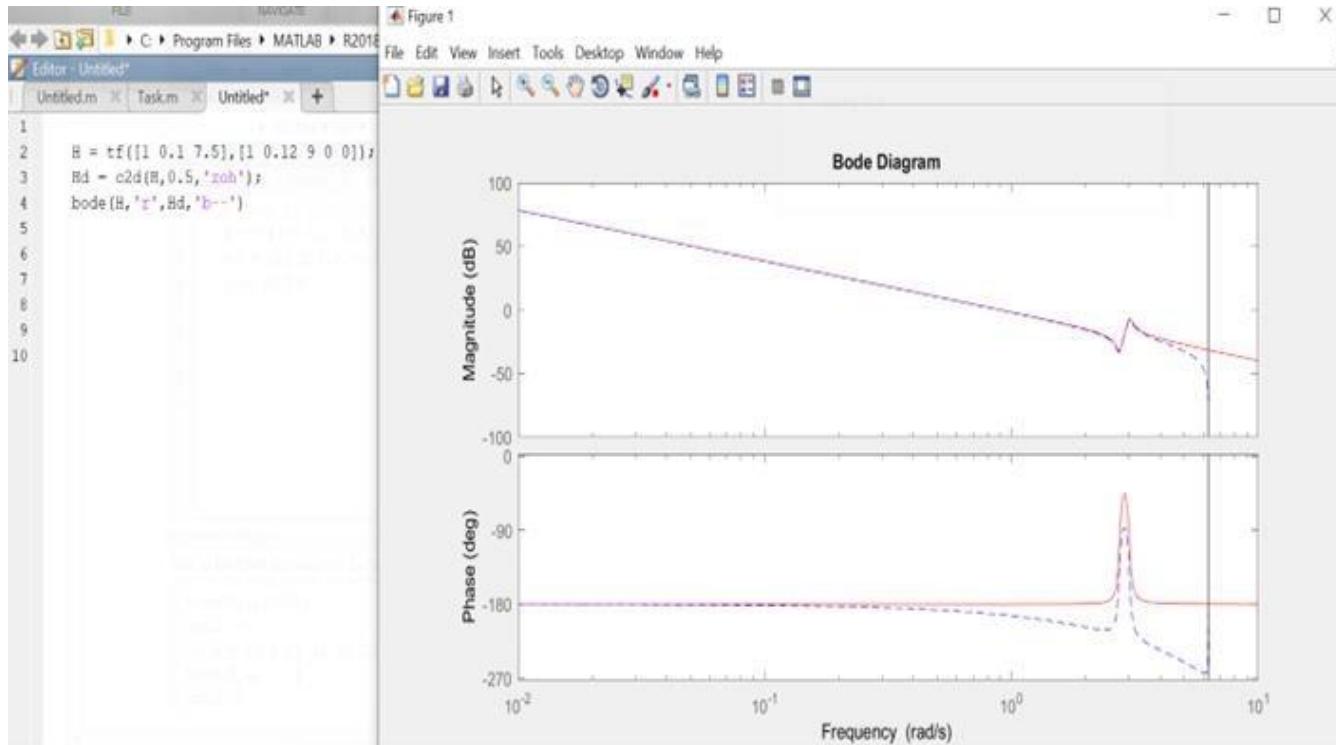
Example: 4



Example: 5



Example: 6





Example: 7

```
2     win = logspace(-2,2,5)
3     [mag,phase,wout] = bode(G,win);
4     size(wout)
5     size(mag)
6     size(phase)
7
8
9
10
11
12
```

Command Window

New to MATLAB? See resources for [Getting Started](#).

```
win = logspace(-2,2,5)
[mag,phase,wout] = bode(G,win);
size(wout)
size(mag)
size(phase)
```

Command Window

New to MATLAB? See resources for [Getting Started](#).

```
[mag,phase,wout] = bode(G,win);
size(wout)
size(mag)
size(phase)

win =
0.0100    0.1000    1.0000   10.0000   100.0000

ans =
5      1

ans =
1      1      5

ans =
1      1      5
```

fx >> |



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC

ENGINEERING



DIGITAL CONTROL SYSTEMS (ES-

413)



DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

LAB # 11: PERFORMANCE OF PID AND DIGITAL PID CONTROLLER

Name: <u>KARAN</u>	Roll #: <u>20ES062</u>
Date: _____	
Signature of Lab Tutor:	

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	<i>To illustrate the performance of PID and digital PID controller using MATLAB/Simulink</i>	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	



Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO



ENGINEERING

DEPARTMENT OF ELECTRONIC

DIGITAL CONTROL SYSTEMS (ES-413)

Equipment Required: Computer with MATLAB

Approximate Time Required: Two to three hours

Introduction

Digital PID Controller

A digital PID Controller works on the same principle as an analog PID. For a pulse transfer function a digital PID Controller is given as follows:

$$G_D(z) = K_p + K_i \cdot z / (z - 1) + K_d \cdot (z - 1) / z$$

Advantages of Digital PID Controllers

There are many useful options in digital controllers that are unavailable in analog controllers, such as variable tuning; selectable PID structure [P and D on process variable (PV) or error] and form; gap control; error squared; tuning algorithms; accurate tuning settings; and others. However, for loops that require a very fast response, the analog controller is faster than most digital controllers.

The characteristics of P, I, and D controllers

The proportional controller (K_p) will have the effect of reducing the rise time and will reduce, but never eliminate, the steady state error.

An integral controller (K_i) will have the effect of eliminating the steady state error, but it may make the transient response worse.



**MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY,
JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**

A derivative control (K_D) will have the effect of increasing the stability of the system, reducing the overshoot and improving the transient response.

Effect of each controller K_P , K_I and K_D on the closed-loop system are summarized below:

CL Response	Rise Time	Overshoot	Settling Time	S-S Error
K_P	Decrease	Increase	Small Change	Decrease
K_I	Decrease	Increase	Increases	Eliminate
K_D	Small Change	Decreases	Decreases	Small Change

Table -11.1 Characteristics of K_P , K_I & K_D

MATLAB Code to design Digital PID Controller for plant given by

Example 1: $G(s) = 1/s(s+1)$ & $H(s) = 1$

```
close all s=tf('s'); csys=1/(s*(s+1));
csys_closed=feedback(csys,1)
step(csys_closed)%cont: time step
response hold on dsys=c2d(csys,1,'zoh');
dsys_closed=feedback(dsys,1)
step(dsys_closed,'y')%discrete time step response without pid
stepinfo(dsys_closed)%discrete time transient response ch.
%without pid hold on z=tf('z',1); kp=1;ki=0.2;kd=0.2;
dpid=((kp+ki+kd)*z^2-(kp+2*kd)*z+kd)/(z^2-z) %pulse t/f of
%digital pid
dsys_closed_pid=feedback(dpid*dsys,1)
step(dsys_closed_pid,'g')%discrete time step response with pid
stepinfo(dsys_closed_pid)%discrete time transient response ch.
%with pid
```

Result $csys_closed$

=
1

$$s^2 + s + 1$$

Continuous-time transfer function.

$$dsys_closed = 0.3679$$
$$z + 0.2642$$

$z^2 - z + 0.6321$ Sample

time: 1 seconds



**MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY,
JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**

Discrete-time transfer function.

ans = RiseTime: 1

SettlingTime: 16

SettlingMin: 0.8015

SettlingMax: 1.3996

Overshoot: 39.9576

Undershoot: 0

Peak: 1.3996

PeakTime: 3 dpid

=

$$1.4 z^2 - 1.4 z + 0.2$$

$$z^2 - z$$

Sample time: 1 seconds Discrete-time transfer function.

dsys_closed_pid =

$$0.515 z^3 - 0.1451 z^2 - 0.2964 z + 0.05285$$

$$z^4 - 1.853 z^3 + 1.591 z^2 - 0.6642 z + 0.05285$$

Sample time: 1 seconds Discrete-time transfer function. ans = RiseTime: 1

SettlingTime: 18

SettlingMin: 0.7669

SettlingMax: 1.7079

Overshoot: 70.7898

Undershoot: 0



**MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY,
JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**

Peak: 1.7079

PeakTime: 3

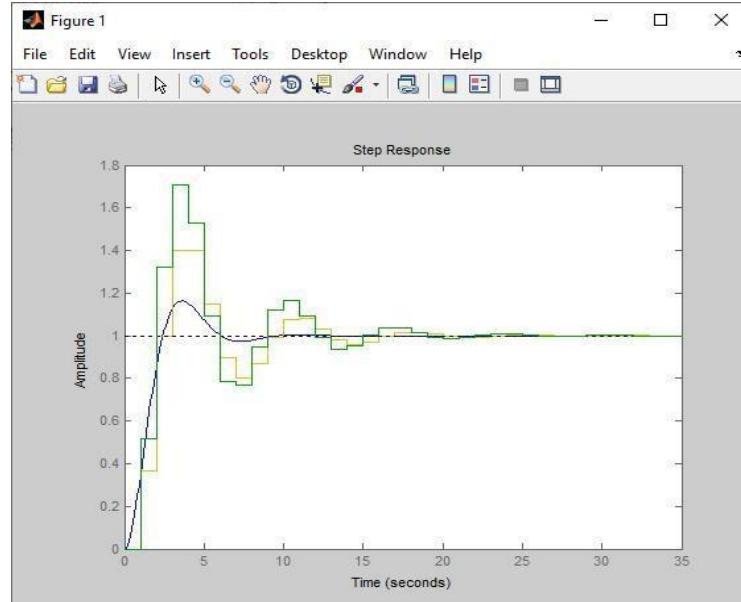


Figure-11.1: Output waveform

There are two ways to design a digital controller or a discrete-time control system. In first way design a continuous-time controller and then discretize it using ZOH or bilinear transformation or any discretization technique to obtain an equivalent digital controller as shown in Figure-11.2. Alternatively, one could discretize the plant first to obtain a sampled-data system or discrete-time system and then apply digital control system design techniques to design a digital controller as shown in Figure-11.3.

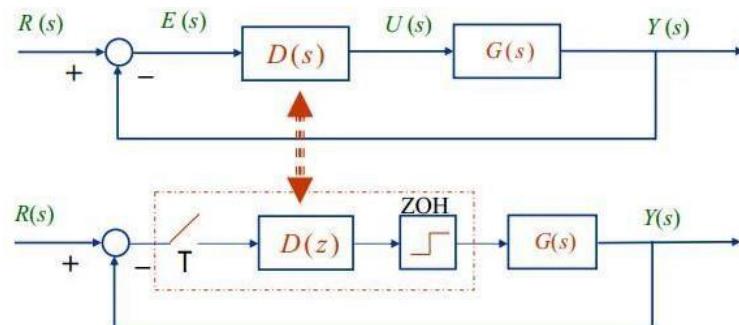


Figure-11.2: Digital controller using continuous-time controller



**MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY,
JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**

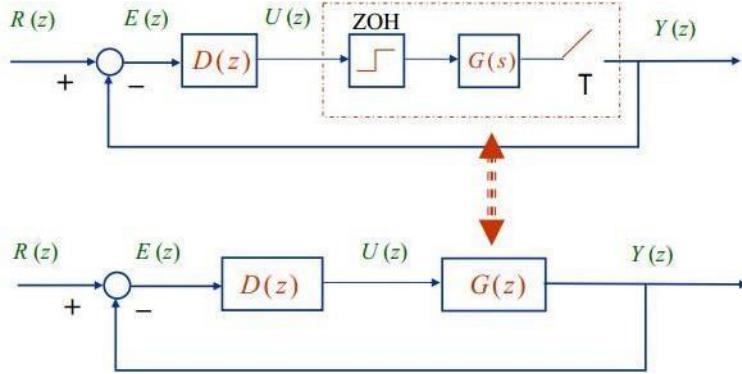


Figure-11.3 Digital controller using sampled-data system

The three term Proportional-Integral-Derivative (PID) control is still widely used in industries because of its simplicity. No need for a plant model and no design to be performed. The user just installs a controller and adjusts three gains to get the best achievable performance. PID regulation is the most mature and the most widely used technology of continuous system. The substance of its regulation is based on the deviation of the input value, a function of the proportional, integral and differential operator and the result of calculation for the output to control. In practical applications, depending on the circumstances, the structure of the PID control can be flexibly changed. Most PID controllers nowadays are digital.

For the continuous-time PID, start with the so-called parallel form

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{d}{dt} e(t) \quad (1)$$

with its Laplace transform

$$G_s = K_p + K_i \frac{1}{s} + K_d s \quad (2)$$

$$(G_s) = K_p + K_i \frac{1}{T_i s} + K_d s = K_p \left[1 + \frac{1}{T_i s} + T_d s \right] \quad (3)$$

Where K_p : proportional gain, K_i : integral gain, K_d : derivative gain, T_i : integral time constant and T_d : derivative time constant.



**MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY,
JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**

In MATLAB, the script code of parallel form may be represented by: s

= tf('s'); %% PID Parallel form

Kp=10;

Td=0.1;

Ti=0.1;

$$G = K_p \frac{1}{s} + \frac{1}{s} + T_d s$$

The control parameters are:

- The proportional term: providing an overall control action proportional to the error signal through the constant gain factor.
- The integral term: the action is to reduce steady-state errors through low-frequency compensation by an integrator.
- The derivative term: improves transient response through high-frequency compensation by a differentiator.

The very same system may be designed at SIMULINK Toolbox, represented in Figure-8.3

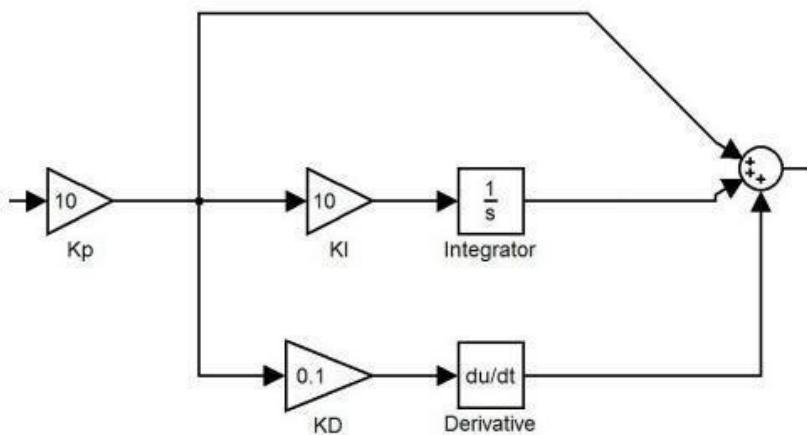


Figure-11.4 Simulink PID Control

11.1 Discrete-time PID Algorithm

For digital implementation, z-transform of equation-1 becomes:

$$U(z) = \left[K_p + \frac{K_i}{1-z^{-1}} + K_d(1-z^{-1}) \right] E(z)$$

11.1.1 Example: 2



**MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY,
JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**

Consider a unity feedback system with forward path transfer function:

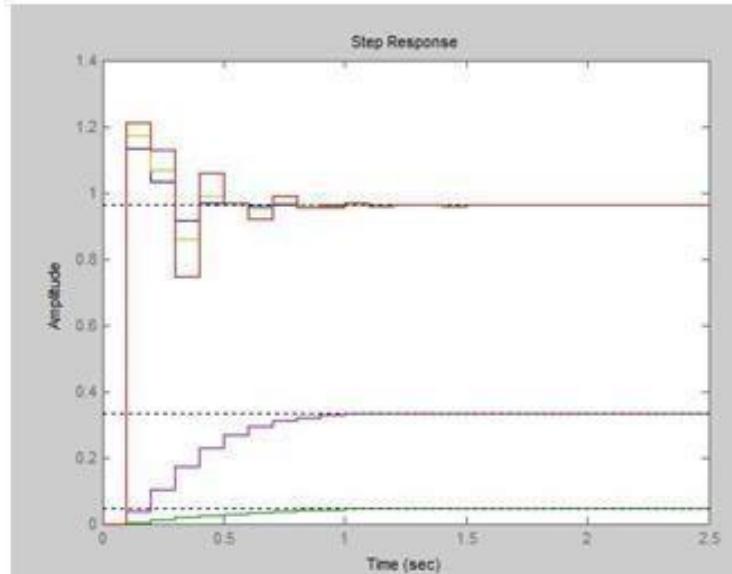
$$G(s) = \frac{1}{s^2 + 10s + 20}$$

Convert this system in z-domain with $T = 0.1$ sec. Show the effect of addition of a PD controller on the system performance. Eliminate overshoots in the controller



**MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY,
JAMSHORO DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)**

```
num=1; den=[1 10
20]; g1=tf
(num,den)
t1=feedback(g1,1)
d1=c2d(t1,.1,'zoh')
step(d1,'g')
hold on
num1=10;
den1=[1 10 20];
g2=tf (num1,den1)
t2=feedback(g2,1)
d2=c2d(t2,.1, 'zoh')
step(d2,'m')
hold on
Kp=500; Kd=10; numc=[Kd Kp];
numo=conv(numc,num) deno=den
g3=tf(numo,deno) t3=feedback(g3,1)
d3= c2d(t3,.1,'zoh') step(d3,'b') hold
on Kp=500; Kd=.01; numc=[Kd Kp];
numo=conv(numc,num) deno=den
g3=tf(numo,deno) t3=feedback(g3,1)
d3= c2d(t3,.1,'zoh') step(d3,'r') hold
on
```





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

11.2 Review Exercise

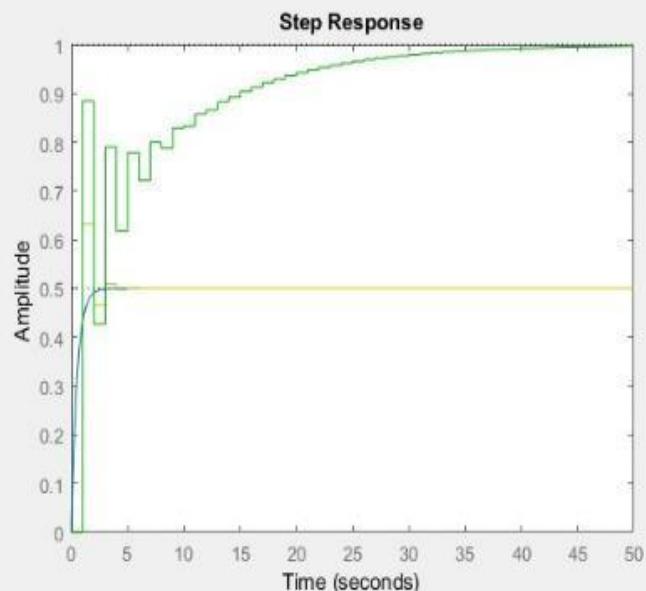
- a) What is digital PID controller?

A Digital Proportional-Integral-Derivative (PID) controller is a type of feedback control system commonly used in industrial processes and various control applications. PID controllers are designed to regulate a system's output by continuously adjusting the input based on the difference between the desired setpoint and the actual process variable. A digital PID Controller works on the same principle as an analog PID. For a pulse transfer function a digital PID Controller is given as follows:

$$GD(z) = K_p + K_i \cdot z / (z-1) + K_d \cdot (z-1) / z$$

- b) Write a MATLAB code to design Digital PID Controller for plant $G(s) = 1/(s+1)$ & $H(s)=1$. Choosing an appropriate sampling time, tune K_p , K_i & K_d to get an optimized step response.

```
2 - close all
3 - s=tf('s');
4 - csys=1/(1*(s+1));
5 - csys_closed=feedback(csys, 1)
6 - step(csys_closed) %cont: time step response
7 - hold on
8 - dsys=c2d(csys,1,'zoh');
9 - dsys_closed=feedback(dsys, 1)
10 - step(dsys_closed, 'y') %discrete time step response without pid
11 - stepinfo(dsys_closed) %discrete time transient response ch.
12 - %without pid
13 - hold on
14 - z=tf('z', 1);
15 - kp=1;ki=0.2;kd=0.2;
16 - dpid=((kp+ki+kd)*z^2-(kp+2*kd)*z+kd)/(z^2-z) %pulse t/f of
17 - %digital pid
18 - dsys_closed_pid=feedback(dpid*dsys, 1)
19 - step (dsys_closed_pid, 'y')%discrete time step response with pid
20 - stepinfo(dsys_closed_pid)%discrete time transient response ch.
21 - %with pid
```





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

der
ledtabiii...
led5555...
led3155...
led333.m
led4han...
led4.m
led4.asv
led3OPEN...
led3OPEN...
led3.m
led2ope...
led2.m
led2.asv

Editor - D:\Program Files\MATLAB\R2016a\bin\Untitled3OPEN.m

Command Window

```
>> Untitled3OPEN

csys_closed =
1
-----
s + 2

Continuous-time transfer function.

dsys_closed =
0.6321
-----
z + 0.2642

Sample time: 1 seconds
Discrete-time transfer function.

ans =
RiseTime: 0
SettlingTime: 3
```

dtabiii...
d5555...
d3155...
d333.m
d4han...
d4.m
d4.asv
d3OPEN...
d3OPEN...
d3.m
d2ope...
d2.m
d2.asv

Editor - D:\Program Files\MATLAB\R2016a\bin\Untitled3OPEN.m

Command Window

```
RiseTime: 0
SettlingTime: 3
SettlingMin: 0.4651
SettlingMax: 0.6321
Overshoot: 26.4241
Undershoot: 0
Peak: 0.6321
PeakTime: 1

dpid =
1.4 z^2 - 1.4 z + 0.2
-----
z^2 - z

Sample time: 1 seconds
Discrete-time transfer function.

dsys_closed_pid =
0.885 z^2 - 0.885 z + 0.1264
```

Name

- Untitleddtabiii...
Untitled5555...
Untitled3155...
Untitled333.m
Untitled4han...
Untitled4.m
Untitled4.asv
Untitled3OPEN...
Untitled3OPEN...
Untitled3.m
Untitled2ope...
Untitled2.m
Untitled2.asv

Details

Workspace

Name	Type
a	
A	
a1	
a2	
a3	
ans	
b	
n	

Command Window

```
Discrete-time transfer function.

dsys_closed_pid =
0.885 z^2 - 0.885 z + 0.1264
-----
z^3 - 0.4829 z^2 - 0.5171 z + 0.1264

Sample time: 1 seconds
Discrete-time transfer function.

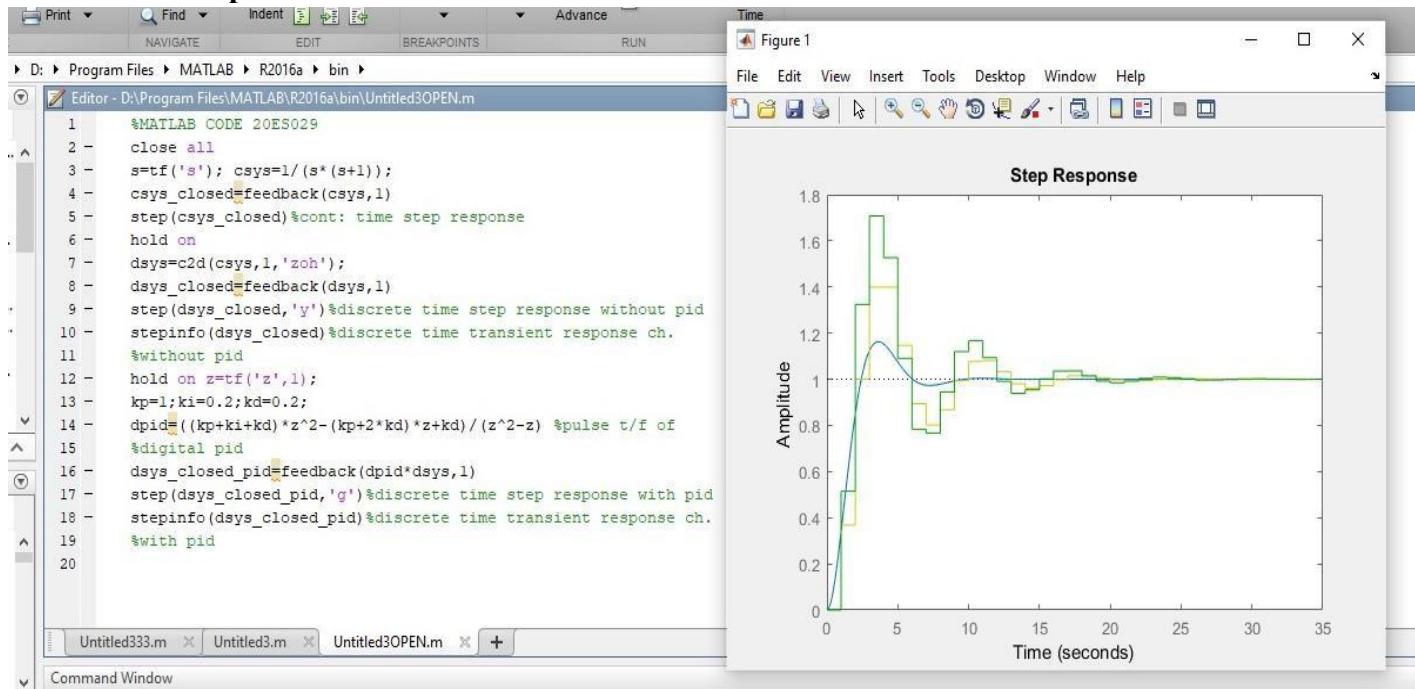
ans =
RiseTime: 14
SettlingTime: 31
SettlingMin: 0.9045
SettlingMax: 1.0000
Overshoot: 0
Undershoot: 0
Peak: 1.0000
PeakTime: 89
```



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING **DIGITAL**
CONTROL SYSTEMS (ES-413)

c) Attach the code and results of example1 and example2.

Example:1





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)

Editor - D:\Program Files\MATLAB\R2016a\bin\Untitled3OPEN.m

Command Window

```
>> Untitled3OPEN
csys_closed =
    1
-----
s^2 + s + 1

Continuous-time transfer function.
```

days_closed =
$$\frac{0.3679 z + 0.2642}{z^2 - z + 0.6321}$$

Sample time: 1 seconds
Discrete-time transfer function.

ans =
$$\frac{\text{RiseTime: 1}}{z^2 - z}$$

Editor - D:\Program Files\MATLAB\R2016a\bin\Untitled3OPEN.m

Command Window

```
ans =
    RiseTime: 1
SettlingTime: 16
SettlingMin: 0.8015
SettlingMax: 1.3996
Overshoot: 39.9576
Undershoot: 0
Peak: 1.3996
PeakTime: 3

dpid =

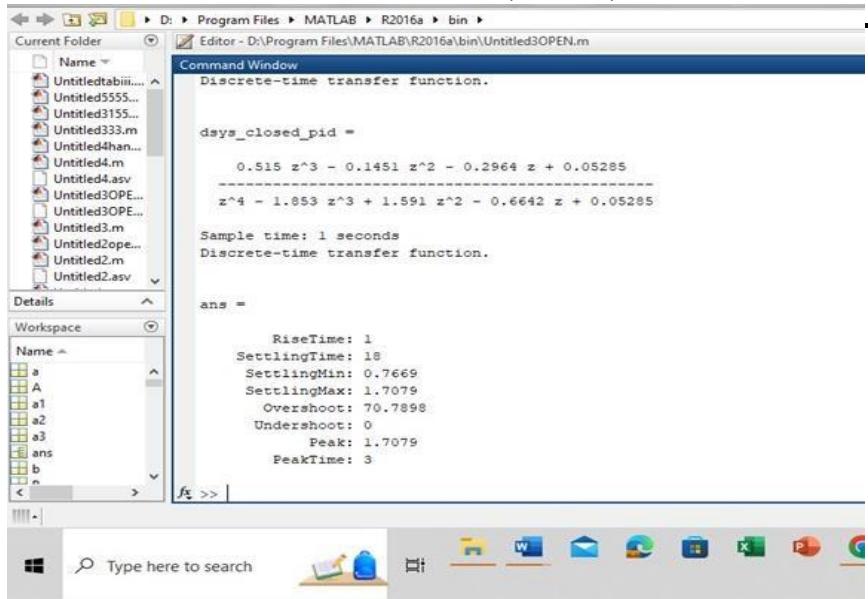
$$\frac{1.4 z^2 - 1.4 z + 0.2}{z^2 - z}$$


Sample time: 1 seconds
Discrete-time transfer function.
```

days_closed_pid =
$$0.515 z^3 - 0.1451 z^2 - 0.2964 z + 0.05285$$

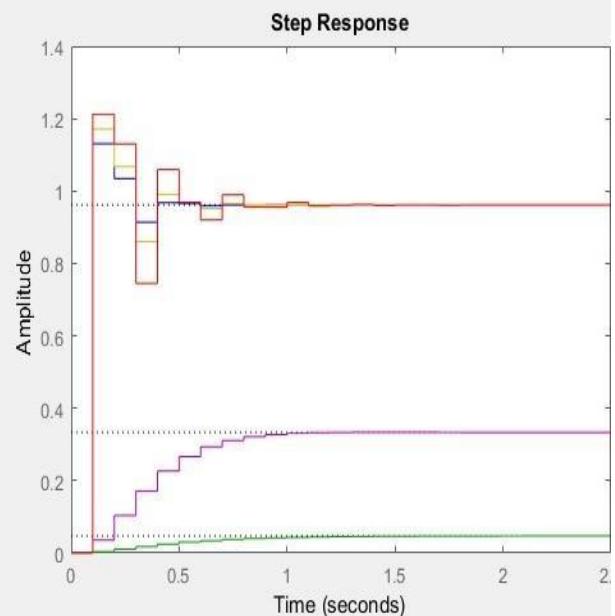


MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING DIGITAL
CONTROL SYSTEMS (ES-413)



Example:2

```
2 - num=1;
3 - den=[1 10 20];g1=tf (num,den)
4 - tl=feedback(g1,1) d1=c2d(tl,.1,'zoh')step(d1,'g')
5 - hold on
6 - num1=10;den1=[1 10 20];g2=tf (num1,den1) t2=feedback(g2,1)
7 - d2=c2d(t2,.1, 'zoh')step(d2,'m')
8 - hold on
9 - Kp=500; Kd=10;numc=[Kd Kp];
10 - numo=conv(numc,num) deno=den
11 - g3=tf(numo,deno) t3=feedback(g3,1)
12 - d3= c2d(t3,.1,'zoh')step(d3,'b')
13 - hold on
14 - Kp=500; Kd=5;numc=[Kd Kp]; numo=conv(numc,num) deno=den
15 - g3=tf(numo,deno) t3=feedback(g3,1)
16 - d3= c2d(t3,.1,'zoh')step(d3,'y')
17 - hold on
18 - Kp=500; Kd=.01;numc=[Kd Kp]; numo=conv(numc,num)
19 - deno=den g3=tf(numo,deno) t3=feedback(g3,1)
20 - d3= c2d(t3,.1,'zoh')
21 - step(d3,'r')
22 - hold on
```





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

LAB # 12: TUNING OF DIGITAL PID CONTROLLER USING SIMULINK

Name: <u>KARAN</u>	Roll #: <u>20ES062</u>
Signature of Lab Tutor:	Date: _____

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	Tuning of digital PID controller using simulink	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
--------------------	------------	---------------	------------	-------



Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	
Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

Equipment Required: PC with MATLAB

Approximate Time Required: Two to three hours

MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Introduction

Tuning of PID Controller

PID tuning is the process of finding the values of proportional, integral, and derivative gains of a PID controller to achieve desired performance and meet design requirements. PID controller tuning appears easy, but finding the set of gains that ensures the best performance of your control system is a complex task.

Traditionally, PID controllers are tuned either manually or using rule-based methods. Manual tuning methods are iterative and time-consuming, and if used on hardware, they can cause



damage. Rule-based methods also have serious limitations: they do not support certain types of plant models, such as unstable plants, high-order plants, or plants with little or no time delay.

PID tuning using Simulink

The Simulink Control Design™ tools let you tune single-loop control systems

Example: Consider a plant given by

$$G(z) = (0.3679z+0.2642)/(z^2-1.368z+0.3679) \text{ & } H(z) = 1$$

First, the closed-loop step response of the plant can be observed using the following Simulink model:

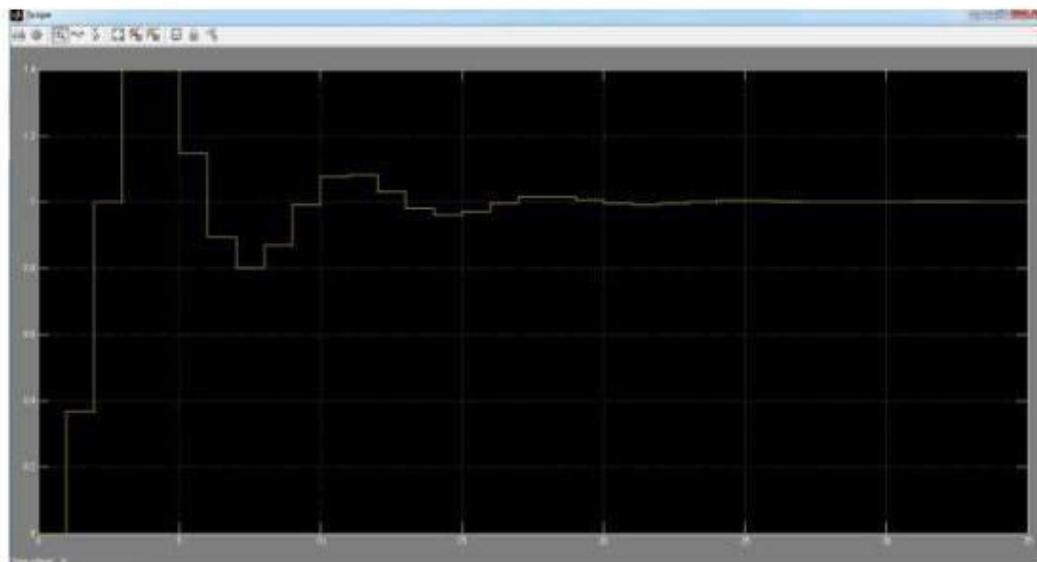
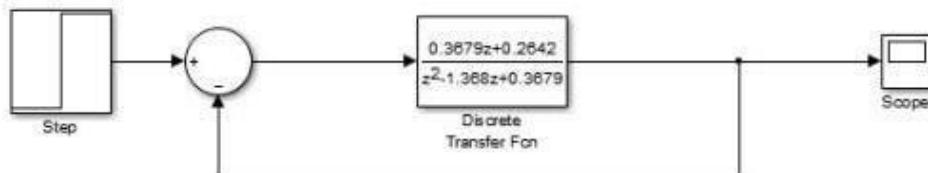


Figure-12.1 Closed Loop Response of plant $G(z)$

MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

Then, keeping in view the pulse transfer function of a Digital PID Controller, i.e.
 $G_D(z) = K_p + K_i * z / (z - 1) + K_d * (z - 1) / z$

a Simulink model for digital PID controller can be designed using Simulink blocks such as Discrete-Transfer Fcn Block(s), Gain Block(s), a source block such as Step Block and a sink block such as Scope Block.

Simulink Design of a digital PID controller for the plant $G(z)$ specified before, is shown as follows.



Here $K_p=1$, $K_i= 0.2$ and $K_d= 0.2$.

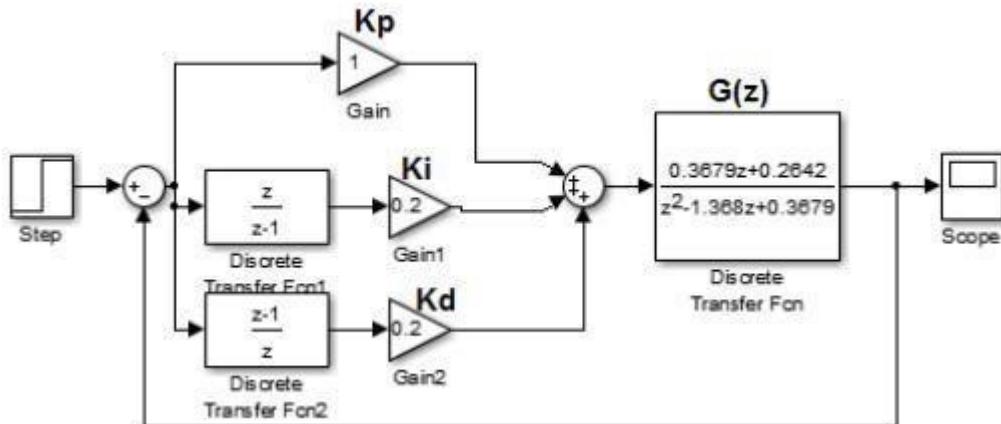


Figure 12.2- Simulink diagram

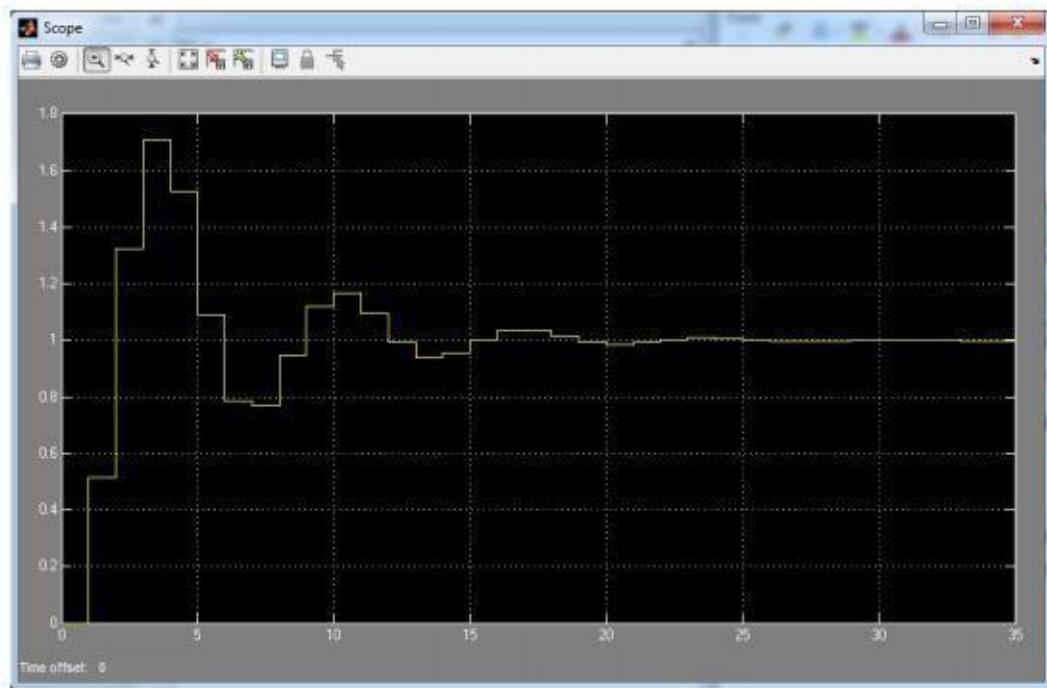


Figure -12.3 Simulink Design of digital PID controller for the plant $G(z)$

MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

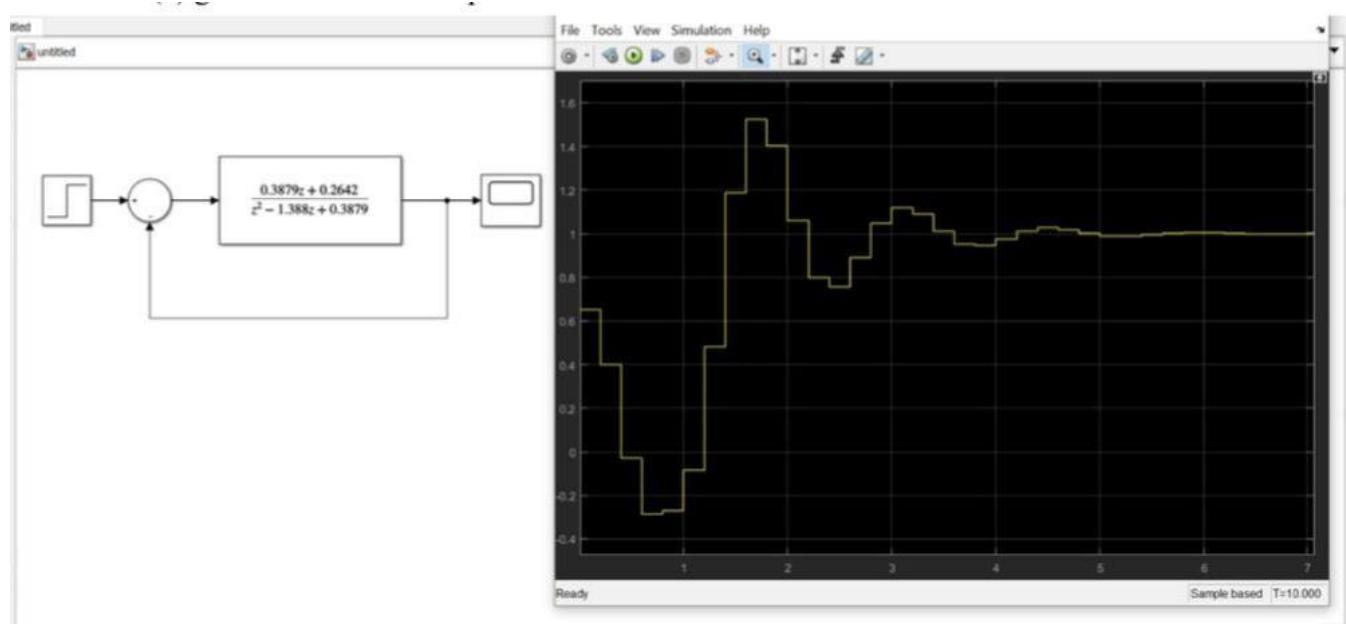
These values of K_p , K_i and K_d are tuned on hit-&-trial basis in order to obtain desired system response.

Likewise, you can design digital PI- Controller, PD-Controller in Simulink



Review Exercise?

- a) Using Simulink, perform PID tuning to get an optimized step response of the plant $G(z)$ given in the lab example





DIGITAL CONTROL SYSTEMS (ES-413)

LAB # 13: OPERATION OF COMPENSATOR & DESIGN OF LEAD COMPENSATOR

Name: KARAN

Roll # 20ES062

Signature of Lab Tutor: _____

Date: _____

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	To illustrate the operation of compensator and design of Lead compensator	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S)

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	
Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	



DIGITAL CONTROL SYSTEMS (ES-413)

TOTAL	
-------	--

Equipment Required: PC with MATLAB

Approximate Time Required: Two to three hours

Introduction

A feedback control system that provides in optimum performance without any necessary adjustment is rare, in building a control system the proper modification of plant dynamics may be a simple way to meet the performance specifications, this however may not be possible in many practical situations because the plant may be fixed and not modifiable then we must adjust parameters other than those in the fixed plant.

It is then required to reconsider the structure of the system and redesign the system, the design problems therefore become those of improving system performance by insertion of compensator

Compensator

Compensator is an additional component or circuit that is inserted into a control system to equalize or compensate for a deficient performance Compensation schemes commonly used for feedback control systems are

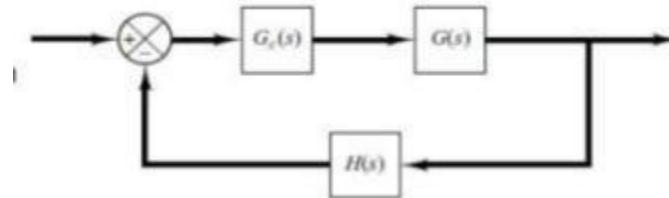


Figure-13.1: Series compensator

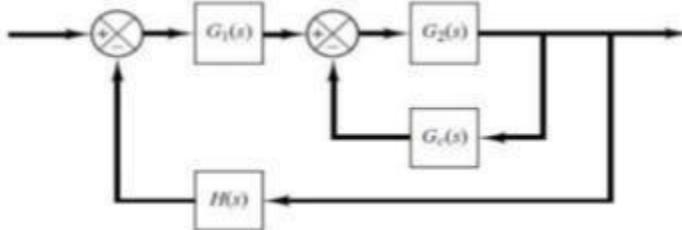


Figure-13.2: Parallel Compensator

Phase Lead or Lag Compensator



DIGITAL CONTROL SYSTEMS (ES-413)

A discrete time phase lead /lag compensator is designed by using the bilinear transformation that maps the inside of the unit circle in Z-Plan to the entire the LHP of w-plane. The phase lead lag compensator is designed in the w-domain utilizing the techniques available for continuous time compensator design, then map it back to the z-domain by the bilinear transformation. The first order w-domain compensator is given by analogy to s-domain as

$$D(W) = a_0 \frac{\frac{W}{1+\frac{W}{W_{wp}}}}{1+\frac{W}{W_{w0}}} \quad 1)$$

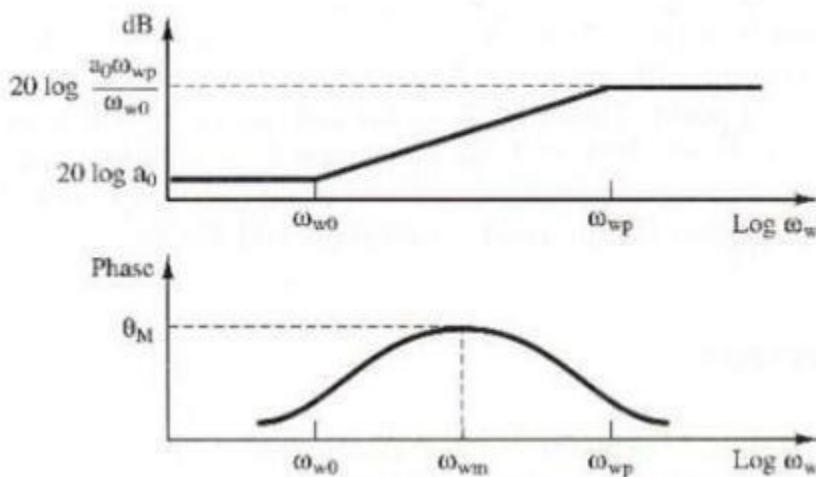


Figure-13.3: a Pole at $w = -Wwp$ and a zero is at $w = -Wwo$

Phase lead compensator: $Wwp > Wwo$

Substituting $\frac{2z-1}{Tz+1}$ for w in the compensator transfer function

$$D(z) = \frac{\frac{2z-1}{Tz+1}}{1 + \frac{\omega_o}{\omega_{wp}} \frac{2z-1}{Tz+1}} \quad 2)$$

$$= a_0 \frac{\omega_{wo} \omega T (z+1) + 2 \omega_{wp} (z-1)}{\omega_{wo} \omega T (z+1) + 2 \omega_{wo} (z-1)} \quad 3)$$

Digital Compensator Design

1 Frequency Response Method

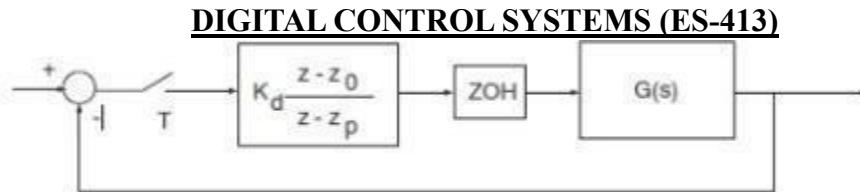


Figure-13.4: Digital compensator design block diagram Design

a phase Lead or Phase lag compensator

$$D(z) = \frac{kd(z - z_o)}{z - z_p}$$

Such that the (open loop system satisfies a given phase margin of $\varphi m \approx (100\epsilon)$) so that the close loop system has damping ratio of ϵ since the frequency response method based on phase margin and gain margin established in the Laplace domain (s-domain) is used here the frequency response of $D(z)$ combined with a plant into *w-domain* by using the bilinear transformation because stable region of the z-domain plan that is the inside of a unit circle is mapped into the entire *LHP (Left half plan)* of the w- plane. The parameters k, d, z_c, z_p of the first order phase lead.lag compensator are determined in terms of $\alpha_0, \omega_{wo}, \omega_{wp}$ defined in the

w-domain

Design a phase lag compensator that achieve a phase margin of

$$\varphi m = 55^\circ \text{ for } \alpha_0 = 1$$

$$\angle G(j\omega_1) = -180^\circ + 55^\circ + 5^\circ = -120^\circ$$

By the bode diagram $\omega\omega_1 \approx 0.36$.

$$G(j\omega_1) = 2.57 = 8.2 \text{ DB} \quad \text{at this frequency}$$

therefore

$$\omega\omega_0 = 0.1\omega\omega_1 = 0.036$$

$$\omega\omega\rho = \frac{0.1\omega\omega_1}{\alpha_0[G(j\omega_1)]} = 0.036$$

Using these values of α_0, ω_{wo} and ω_{wp}

$$D(z) = 0.3891 \frac{z - 0.9982}{z - 0.9993}$$

The frequency response after compensator is shown in the frequency axis of $\omega\omega$



DIGITAL CONTROL SYSTEMS (ES-413)

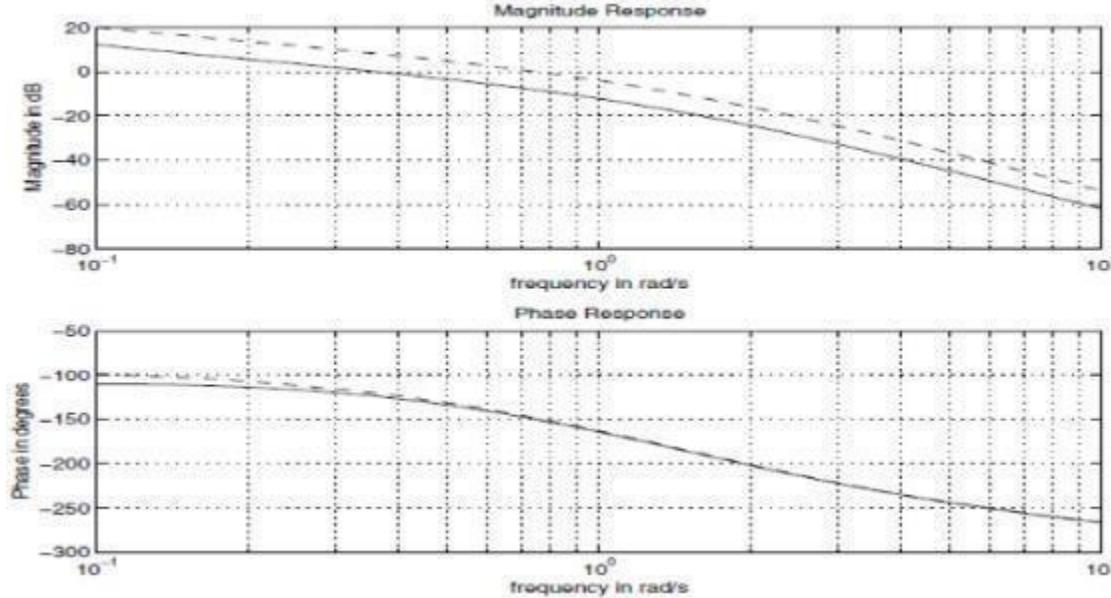
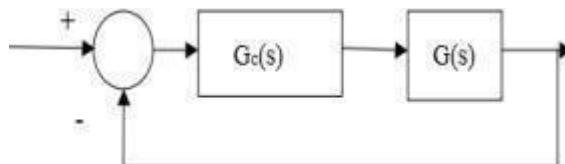


Figure-13.5: compensator output response

Matlab Code For Phase Lead Compensation



```
wm = 4.5;
alpha = 0.3; T =
1/(wm*sqrt(alpha)); k= 10;
gnum = [k];
gden = [1 1 0]; uncompensated = tf(gnum,gden)
cnum = [T 1];
cden = [T*alpha 1]; compensator = tf(cnum,cden) numo =
conv(cnum,gnum); deno
= conv(cden,gden); compensated = tf(numo,deno)
bode(uncompensated,'r',compensated , 'g')
uncomtr=feedback(uncompensated,1)
comtr=feedback(compensated,1) step(uncomtr,'y') hold
on step(comtr,'b')
```



DIGITAL CONTROL SYSTEMS (ES-413)

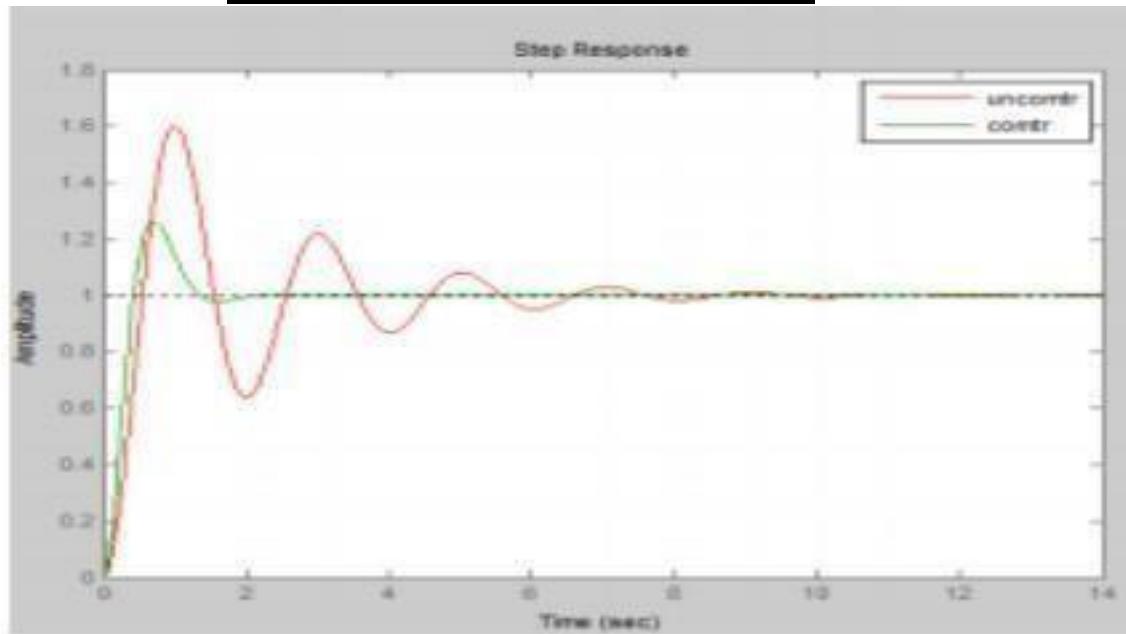


Figure-13.6: Output response of Phase Lead compensator

Design of lead Compensator :

Example: Design a lead compensator for the following system. The system achieve the following requirement. $M_p \leq 20\%$, $T_r \leq 0.3$ sec

$$G(S) = \frac{1}{s(-s+1)}$$

MATLAB Code:

```
G=tf(1, [1 1 0]) z= 8.0501  
k= 4.472 C=tf(k*[1  
z],[1])  
T=feedback(C*G,1  
)  
step(T) Sisotool(G)
```



DIGITAL CONTROL SYSTEMS (ES-413)

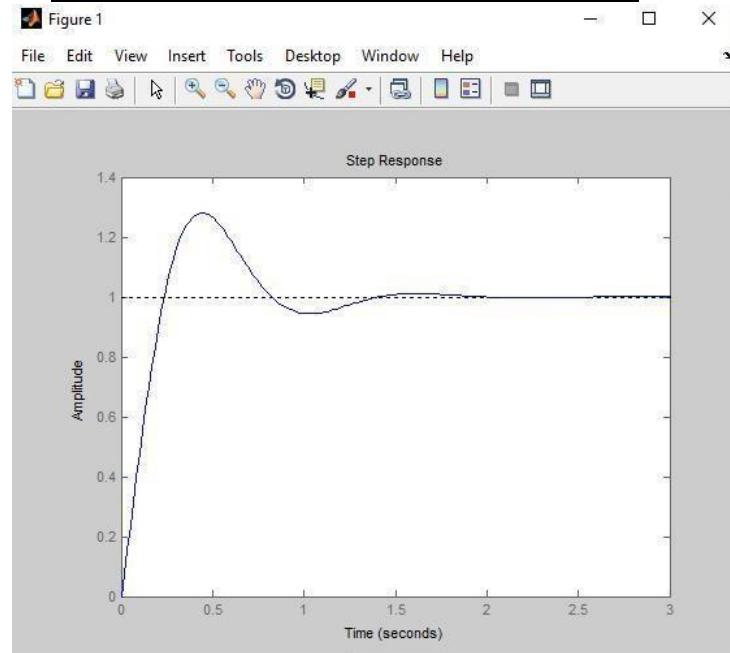


Figure-13.7: Output response of Phase Lead

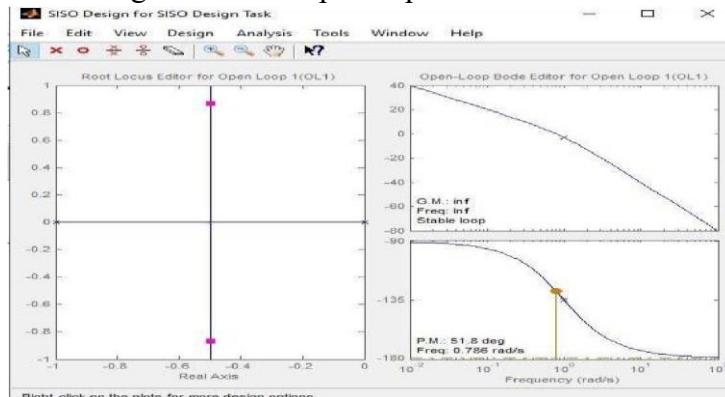


Figure-13.8: Output response using SISO tool

Example2:

```
Gv = tf(0.02,[4 1]); G1 = tf(70,[50 1]); H = tf(1,[12 1]);  
Gc = tf(1); % dummy static gain
```

```
Gol = Gc*Gv*G1; % Openloop: I'm assuming Gp = G1  
Gcl = feedback(Gol,H); % closed loop  
step(Gcl) % look at the response stepinfo(Gcl) %
```



DIGITAL CONTROL SYSTEMS (ES-413)

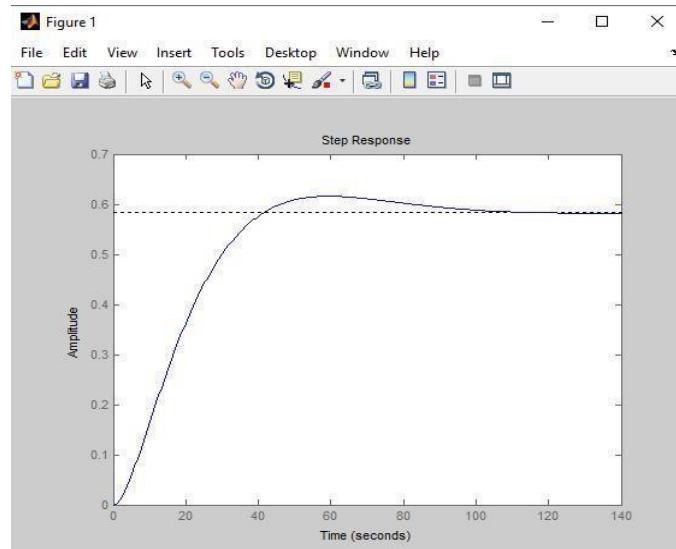


Figure-13.9: Output response of compensator

ans =

```
RiseTime: 27.5867
SettlingTime: 88.7066
SettlingMin: 0.5274
SettlingMax: 0.6159 Overshoot:
5.5879
Undershoot: 0
Peak: 0.6159
PeakTime: 59.4622
```

Example3:



DIGITAL CONTROL SYSTEMS (ES-413)

```
% ----- Unit ramp response -----  
  
% ***** Unit-ramp responses of compensated system and  
% uncompensated system *****  
  
% ***** Unit-ramp response will be obtained as the unit-step  
% response of C(s)/[sR(s)] *****  
  
% ***** Enter the numerators and denominators of C1(s)/[sR(s)]  
% and C2(s)/[sR(s)], where C1(s) and C2(s) are Laplace  
% transforms of the outputs of the compensated and un-  
% compensated systems, respectively. *****  
  
numc = [0 0 0 0 1.0235 0.0512];  
denc = [1 3.005 2.015 1.0335 0.0512 0];  
num = [0 0 0 0 1.06];  
den = [1 3 2 1.06 0];  
  
% ***** Specify the time range (such as t = 0:0.1:50) and enter  
% step command and plot command. *****  
  
t = 0:0.1:50;  
[c1,x1,t] = step(numc,denc,t);  
[c2,x2,t] = step(num,den,t);  
plot(t,c1,'-',t,c2,'.',t,t,'-')  
grid  
text(2.2,27,'Compensated system');  
text(26,21.3,'Uncompensated system')  
title('Unit-Ramp Responses of Compensated and Uncompensated Systems')  
xlabel(' Sec');  
ylabel('Outputs c1 and c2')
```

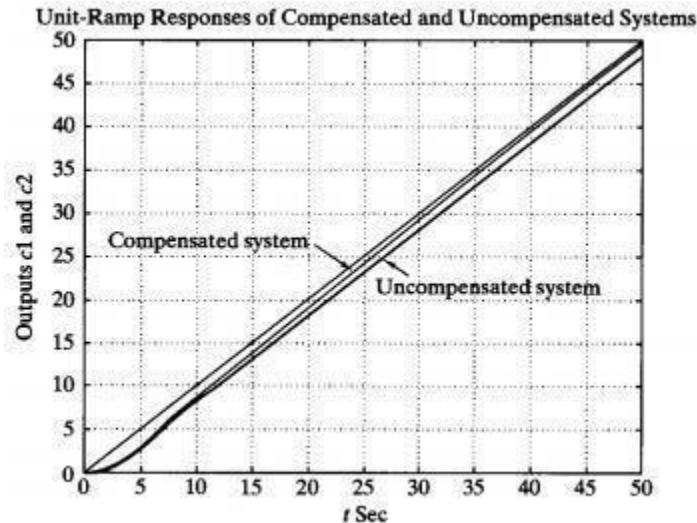


Figure-13.10: Compensated and uncompensated output response of a system



DIGITAL CONTROL SYSTEMS (ES-413)

Example4:

```
% ----- Unit-step response -----  
  
% ***** Unit-step responses of compensated system and  
% uncompensated system *****  
  
% ***** Enter the numerators and denominators of the  
% compensated and uncompensated systems *****  
  
numc = [0 0 0 1.0235 0.0512];  
denc = [1 3.005 2.015 1.0335 0.0512];  
num = [0 0 0 1.06];  
den = [1 3 2 1.06];  
  
% ***** Specify the time range (such as t = 0:0.1:40) and enter  
% step command and plot command. *****  
  
t = 0:0.1:40;  
[c1,x1,t] = step(numc,denc,t);  
[c2,x2,t] = step(num,den,t);  
plot(t,c1,'-',t,c2,'-')  
grid  
text(13,1.12,'Compensated system')  
text(13.6,0.88,'Uncompensated system')  
title('Unit-Step Responses of Compensated and Uncompensated Systems')  
xlabel('t Sec')  
ylabel('Outputs c1 and c2')
```

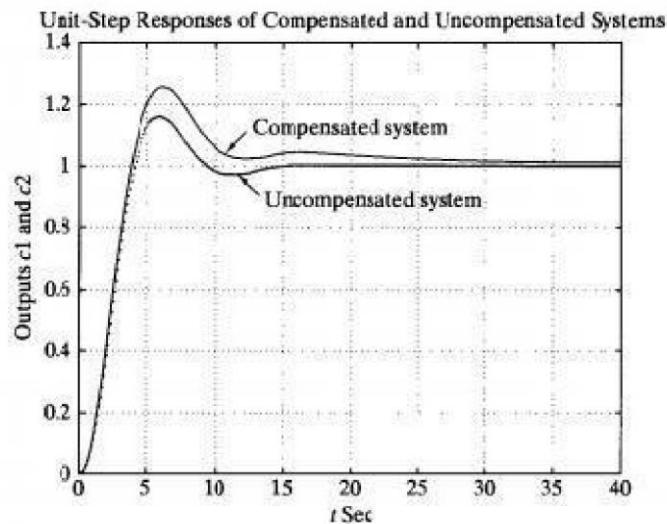


Figure-13.11: Compensated and uncompensated output response of a system

lag-compensated system exhibits a larger maximum overshoot and slower response than the original uncompensated system. Notice that a pair of the pole at $s = -0.0549$ and zero at $s = -0.05$ generates a long tail of small amplitude in the transient response. If a larger maximum overshoot and a slower response are not desired, we need to use a lag-lead compensator as presented in Section 7-5.



Review Questions

a) What is lead compensator in control system?

A lead compensator is a control system element that introduces phase advance, improving stability and transient response by boosting the phase of the system's open-loop transfer function.

b) What is the difference between controller and compensator?

Both controllers and compensators are used to influence the behavior of control systems, controllers directly adjust the system's input based on feedback, while compensators modify the system's dynamics to achieve desired performance characteristics.

c) What is the effects of the phase lead compensator on gain cross over frequency and the bandwidth?

The phase lead compensator increases the gain crossover frequency and broadens the bandwidth of the control system, leading to improved stability, faster response, and enhanced control performance.

d) Attach screenshots of all tasks done in this laboratory.

EXAMPLE 1:

```

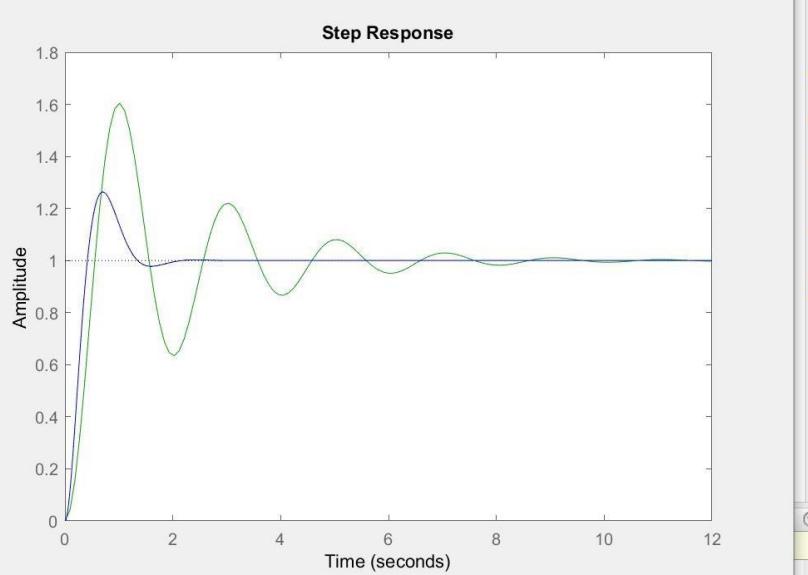
3 - 
4 - 
5 - 
6 - 
7 - 
8 - uncompensated = tf(gnum,gden)
9 - cnum = [T 1]; cden = [T*alpha 1];
10 - compensator = tf(cnum,cden)
11 - numo = conv(cnum,gnum);
12 - deno = conv(cden,gden);
13 - compensated = tf(numo,deno)
14 - bode(uncompensated,'r',compensated , 'g')
15 - uncomtr=feedback(uncompensated,1)
16 - comtr=feedback(compensated,1)
17 - step(uncomtr,'g')
18 - hold on
19 - step(comtr,'b')
20 - 
21 -

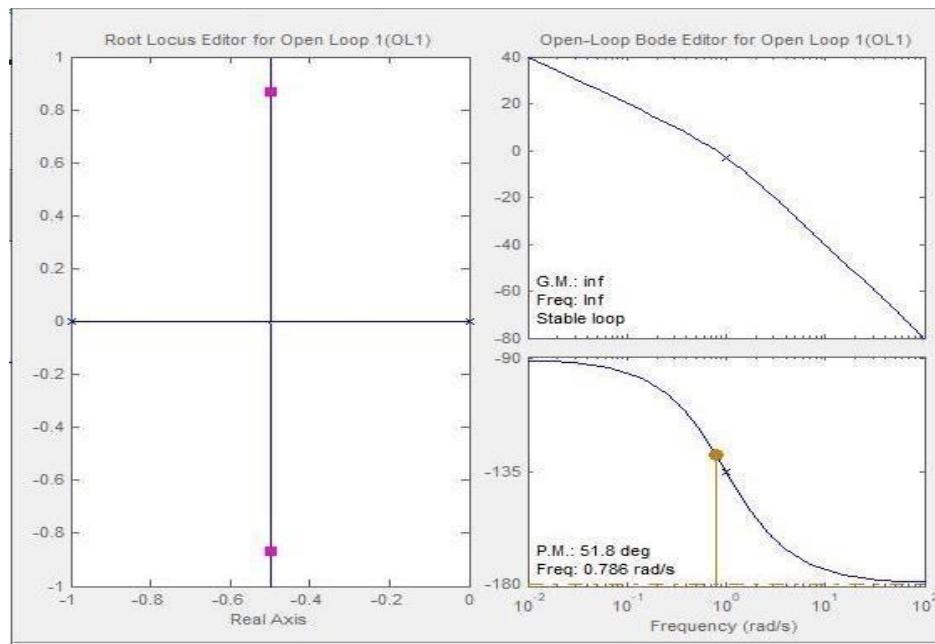
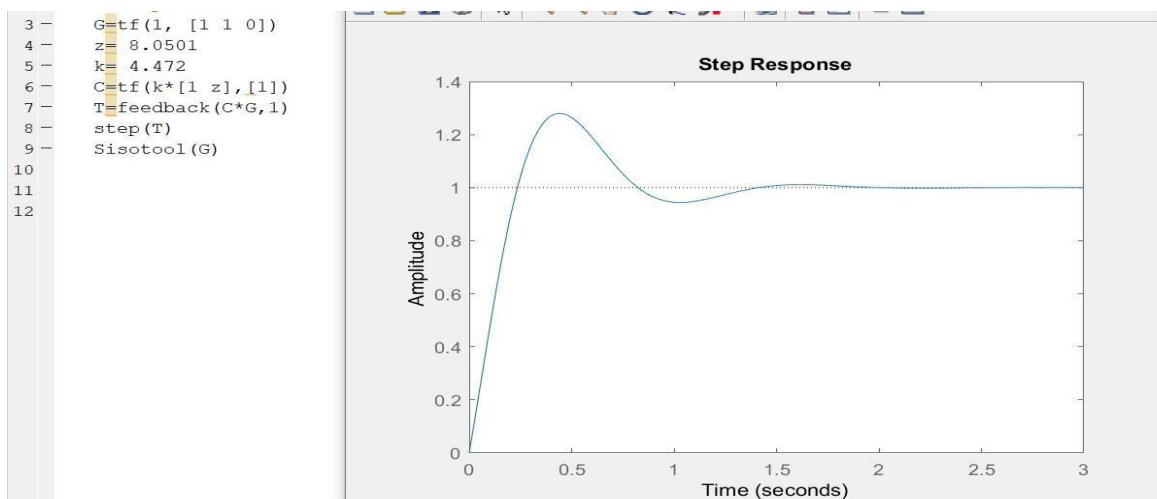
```

Command Window

New to MATLAB? See resources for [Getting Started](#).

Continuous-time transfer function.



413)DIGITAL CONTROL SYSTEMS (ES-**EXAMPLE 2:**



DIGITAL CONTROL SYSTEMS (ES-

EXAMPLE 3:

```
3 - Gv = tf(0.02,[4 1]);
4 - G1 = tf(70,[50 1]);
5 - H = tf(1,[12 1]);
6 - Gc = tf(1); % dummy static gain
7 - Gol = Gc*Gv*G1; % Openloop: I'm assuming Gp = G1
8 - Gcl = feedback(Gol,H); % closed loop
9 - step(Gcl) % look at the response
10 - stepinfo(Gcl) % extract key data for your design
11
12
```

Command Window

New to MATLAB? See resources for [Getting Started](#).

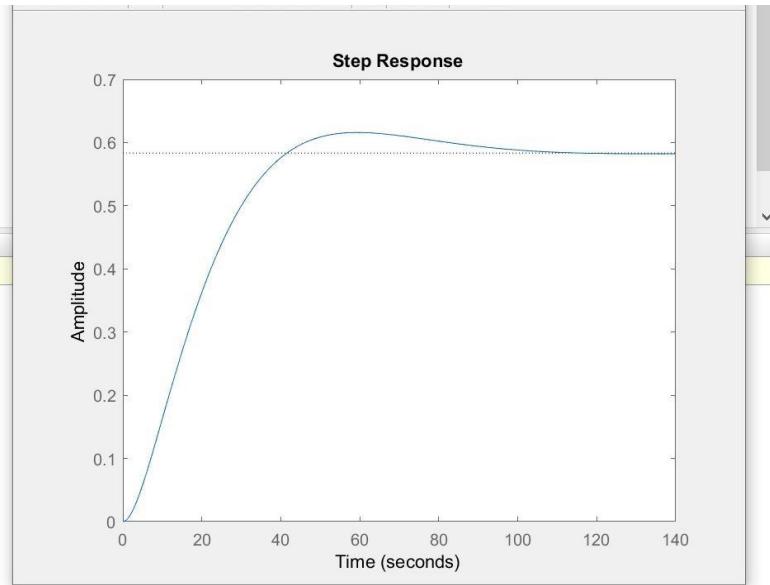
```
stepinfo(Gcl) % extract key data for your design
```

ans =

struct with fields:

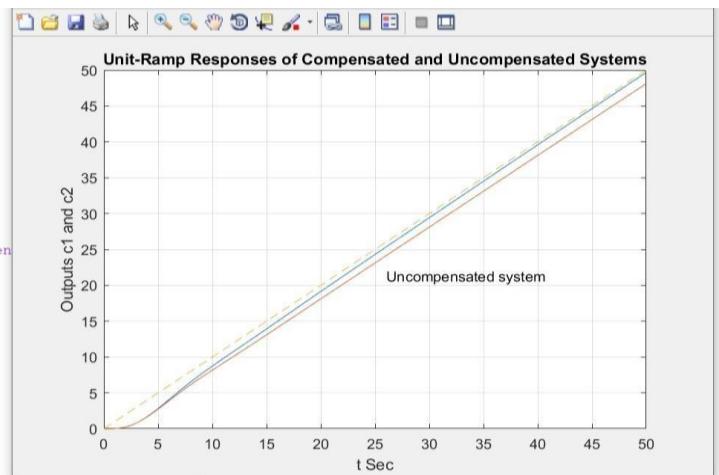
```
RiseTime: 27.5867
SettlingTime: 88.7066
SettlingMin: 0.5274
SettlingMax: 0.6159
Overshoot: 5.5879
Undershoot: 0
Peak: 0.6159
PeakTime: 59.4622
```

fx >>



EXAMPLE 4:

```
3 - numc = [0 0 0 1.0235 0.0512];
4 - denc =[1 3.005 2.015 1.0335 0.0512 0];
5 - num = [0 0 0 1.06];
6 - den = [1 3 2 1.06 0];
7 - t = 0:0.1:50;
8 - [c1,x1,t] = step(numc, denc,t);
9 - [c2,x2,t] = step(num, den,t);
10 - plot(t,c1,'-',t,c2,t,'--')
11 - grid
12 - text(2,2,27, 'Compensated system');
13 - text(26,21,3,'Uncompensated system')
14 - xlabel('t Sec');
15 - title('Unit-Ramp Responses of Compensated and Uncompensated Systems')
16 - ylabel('Outputs c1 and c2')
17
18
19
20
```

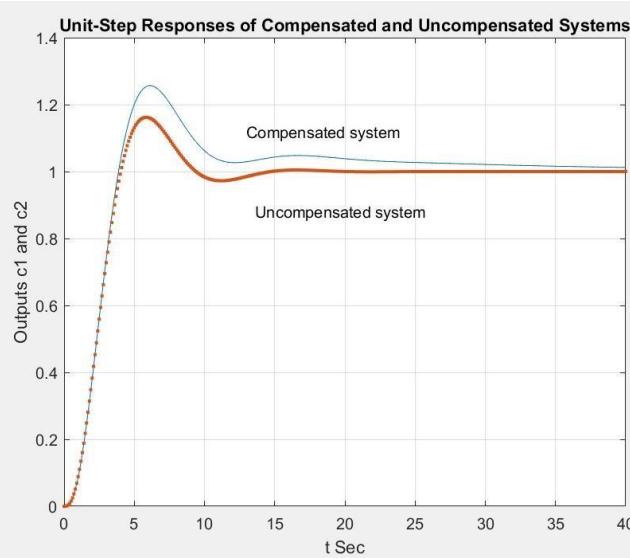




DIGITAL CONTROL SYSTEMS (ES-

EXAMPLE 5:

```
3 - numc=[0 0 0 1.0235 0.0512];
4 - denc= [1 3.005 2.015 1.0335 0.0512];
5 - num = [0 0 0 1.06];
6 - den= [1 3 2 1.06];
7 - t = 0:0.1:40;
8 - [c1,x1,t] = step(numc,denc,t);
9 - [c2,x2,t] = step(num,den,t);
10 - plot(t,c1,'-',t,c2,'.')
11 - grid
12 - text(13,1.12,'Compensated system')
13 - text(13.6,0.88,'Uncompensated system')
14 - xlabel('t Sec')
15 - title('Unit-Step Responses of Compensated and Uncom
16 - ylabel('Outputs c1 and c2')
17
18
19
```





DEPARTMENT OF ELECTRONIC

ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)LAB # 14: PERFORMANCE AND DESIGN OF DIGITAL FILTERName: KARANRoll #: 20ES062

Signature of Lab Tutor:

Date: _____

OBJECTIVE(S):

#	Topic	# Of Lectures	CLO	Taxonomy level
1	<i>To illustrate the performance of digital filter and design of digital filter using MATLAB</i>	3	4,5	P3, A4

OUTCOME(S):

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRIC(S):

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information. Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	



DEPARTMENT OF ELECTRONIC

ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized, lack key aspects.	
TOTAL				

Equipment Required: PC with MATLAB

Approximate Time Required: Two to three hours

Introduction

A digital filter uses a digital processor to perform numerical calculations on sampled values of the signal. The processor may be a general-purpose computer such as a PC, or a specialized DSP. The analog input signal must first be sampled and digitized using an ADC (analog to digital converter). The resulting binary numbers, representing successive sampled values of the input signal, are transferred to the processor, which carries out numerical calculations on them. These calculations typically involve multiplying the input values by constants and adding the products together. If necessary, the results of these calculations, which now represent sampled values of the filtered signal, are output through a DAC (digital to analog converter) to convert the signal back to analog form. In a digital filter, the signal is represented by a sequence of numbers, rather than a voltage or current.

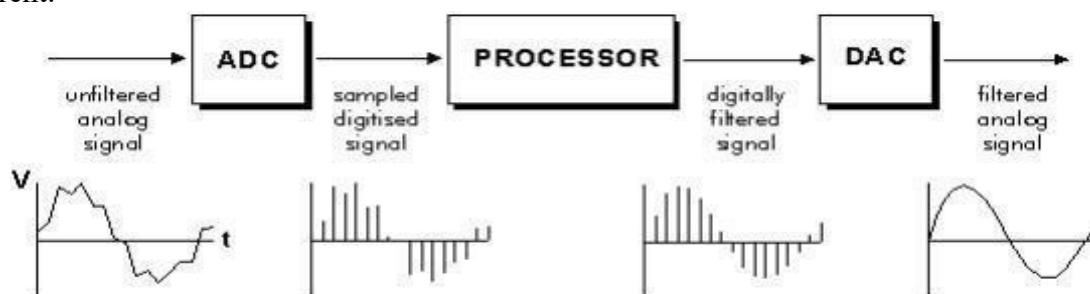


Figure-14.1: Basic setup of Digital filter

A digital filter is programmable, i.e. its operation is determined by a program stored in the processor's memory. This means the digital filter can easily be changed without affecting the circuitry (hardware). An analog filter can only be changed by redesigning the filter circuit.

Digital filters are easily designed, tested and implemented on a general-purpose computer or workstation.



ENGINEERING

DEPARTMENT OF ELECTRONIC

DIGITAL CONTROL SYSTEMS (ES-413)

The characteristics of analog filter circuits (particularly those containing active components) are subject to drift and are dependent on temperature. Digital filters do not suffer from these problems, and so are extremely stable with respect both to time and temperature.

Unlike their analog counterparts, digital filters can handle low frequency signals accurately. As the speed of DSP technology continues to increase, digital filters are being applied to high frequency signals in the RF (radio frequency) domain, which in the past was the exclusive preserve of analog technology.

Digital filters are very much more versatile in their ability to process signals in a variety of ways; this includes the ability of some types of digital filter to adapt to changes in the characteristics of the signal.

Fast DSP processors can handle complex combinations of filters in parallel or cascade (series), making the hardware requirements relatively simple and compact in comparison with the equivalent analog circuitry

Operation of digital filters

Basic theory of the operation of digital filters. This is essential to an understanding of how digital filters are designed and used,

Suppose the "raw" signal which is to be digitally filtered is in the form of a voltage waveform described by the function

$$V = x^t \quad 1)$$

where t is time. This signal is sampled at time intervals h (the sampling interval). The sampled value at time

$$\begin{aligned} t &= it && \text{is} \\ Xt &= x^it \end{aligned} \quad 2)$$

Thus the digital values transferred from the ADC to the processor can be represented by the sequence

$$x, x, x, x, \dots 0 \ t \ 2 \ 3$$

corresponding to the values of the signal waveform at

$$t = 0, t, 2t, 3t, \dots \text{and } t = 0$$

is the instant at which sampling begins.

At time

$$t = nt$$

(where n is some positive integer), the values available to the processor, stored in memory, are x

$$x, x, x, \dots x \ 0 \ t \ 2 \ 3$$

Note that the sampled values

$$xn \ h \ t, xn \ h \ 2 \quad 3)$$

etc. are not available, as they haven't happened yet! The digital output from the processor to the



ENGINEERING

DEPARTMENT OF ELECTRONIC

DIGITAL CONTROL SYSTEMS (ES-413)

DAC consists of the sequence of values

$$y_0, y_1, y_2, y_3, \dots, y_n$$

In general, the value of y_n is calculated from the values

$$x_0, x_1, x_2, x_3, \dots, x_n$$

The way in which the y 's are calculated from the x 's determines the filtering action of the digital filter. In the next section, we will look at some examples of simple digital filters.

14.6 Unity gain filter: $y = x$

Each output value y_n is exactly the same as the corresponding input value

$$\begin{aligned} y_0 &= x_0 \\ y_1 &= x_1 \\ y_2 &= x_2 \\ y_3 &= x_3 \end{aligned}$$

14.6.1 Simple gain filter:

$$y_n = Kx_n \text{ where } K = \text{constant.}$$

This simply applies a gain factor K to each input value. $K > 1$ makes the filter an amplifier, while $0 < K < 1$ makes it an attenuator. $K < 0$ corresponds to an inverting amplifier.

14.6.2 Digital filter coefficients

All of the digital filter examples given above can be written in the following general forms: Zero order:

$$y_n = a_0 x_n$$

First order:

$$y_n = a_0 x_n + a_1 x_{n-1} \quad 4)$$

Second order:

$$y_n = a_0 x_n + a_1 x_{n-1} + a_2 x_{n-2} \quad 5)$$

14.7 Example1: Design the following function using matlab

$$H(z) = \frac{b(1)}{a(1) + a(2)z^{-1}} = \frac{1}{1 - 0.2z^{-1}}$$

MATLAB Code:

```
rng default %initialize random number generator x
= rand(2,15); b = 1; a = [1 -0.2]; y =
filter(b,a,x,[],2); t = 0:length(x)-1; %index
vector plot(t,x(1,:)) hold on plot(t,y(1,:))
legend('Input Data','Filtered Data')
title('First Row')
```



ENGINEERING

DEPARTMENT OF ELECTRONIC

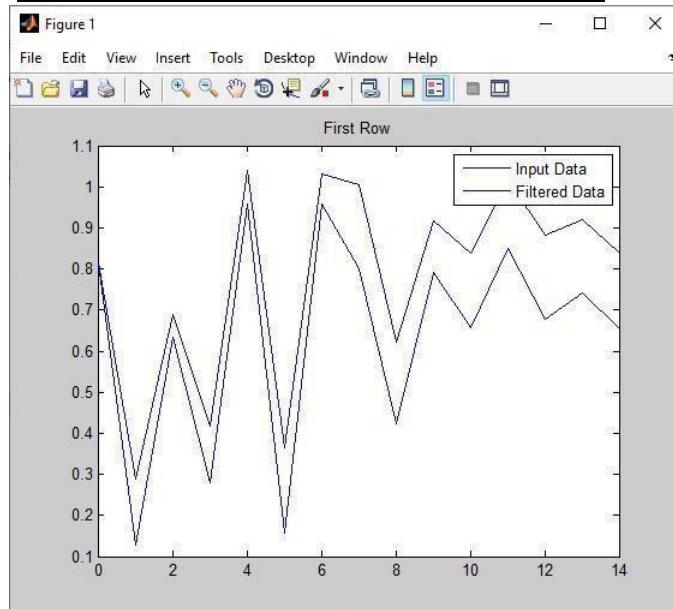
DIGITAL CONTROL SYSTEMS (ES-413)

Figure-14.2:

Figure first row

1

Example2: Plot the second row of input data against the filtered data.

MATLAB Code:

```
figure
plot(t,x(2,:)) hold on
plot(t,y(2,:))
legend('Input Data','Filtered Data') title('Second
Row')
```



ENGINEERING

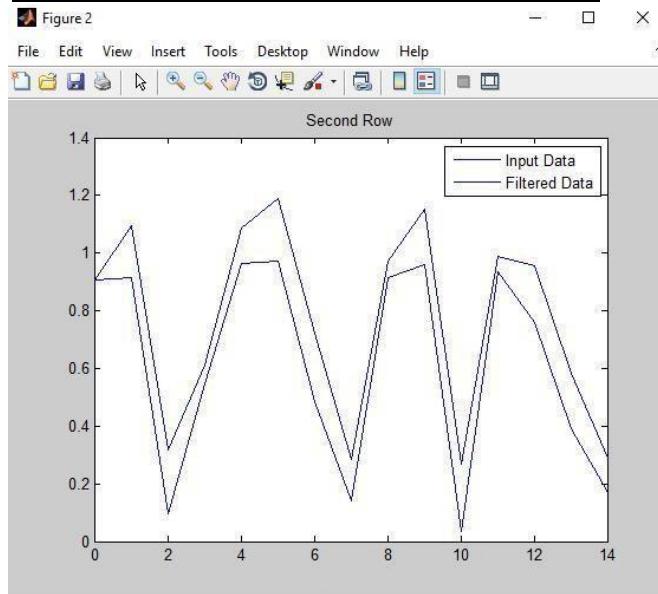
DIGITAL CONTROL SYSTEMS (ES-413)

Figure-14.3: Second row

Example3: Create the script file direc_buttdisig that solves this by directly obtaining the magnitude and phase plots of the discrete transfer function $H(z)$ (LPF third order, with $T = 0.3$ and a cutoff frequency of $w_c = 3 \text{ rad/s}$).

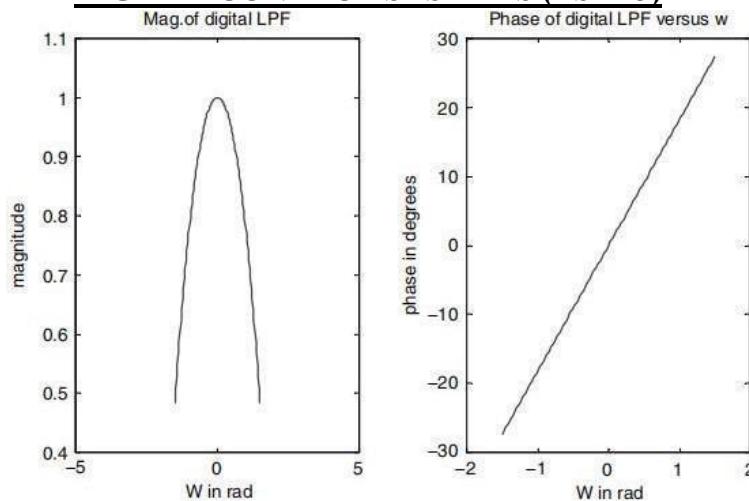
MATLAB Code:

```
% Script file : direc_buttdisig
n =3;
T=.03;wc=3;w =-5:.1:5;
wz = w*T;
[Pz,Qz] = butter(n,wc*T/pi);
[magz,phasez]=freqz(Pz,Qz,wz/pi);
subplot(1,2,1)
plot(wz,magz);
title('Mag. of digital LPF');axis([-5 5 0.4 1.1]);
xlabel('w in rad')
ylabel('magnitude')
subplot(1,2,2)
plot(wz,phasez*180/pi);
title('Phase of digital LPF vs w')
xlabel('w in rad')
ylabel('phase in degrees')
```



ENGINEERING

DEPARTMENT OF ELECTRONIC

DIGITAL CONTROL SYSTEMS (ES-413)Figure 14.4: Magnitude and phase plots of $H(z)$

Example4: Create the script file cheby_HPF that returns the coefficients of the digital transfer function and the magnitude and phase plots of a third-order Chebyshev type-1 and -2 digital HPFs with the following specs:

1. $w_c = 0.5 \text{ rad/s}$
2. $T=0.3$
3. $R_p = 2 \text{ dB}$
4. $R_s = 3 \text{ dB}$

MATLAB Code:

```
% Script file: cheby_HPF
N =3; % order
R_p = 2; R_s = 3; % ripples
T = .3; % sampling period
Wc = .5; % cut off frequency
Wz = -2*pi:.01*pi:3*pi; % normalized dig. frequency
wzc = Wc*T/pi;
[Pz1, Qz1] = cheby1(N, R_p, wzc, 'high');
disp('*****R E S U L T S *****')
disp('For the transfer function H(z) of the Cheby/type1 HPF')
disp('the numerators coefficients are:')
disp(Pz1)
disp('The denominator coefficients are:')
disp(Qz1)
[Pz2, Qz2] = cheby2(N, R_s, wzc, 'high');
disp(' For the transfer function H(z) of the Cheby/type2 HPF')
disp('The numerators coefficients are:')
```



ENGINEERING

DEPARTMENT OF ELECTRONIC

DIGITAL CONTROL SYSTEMS (ES-413)

```
disp(Pz2)
disp('The denominator coefficients are:'); disp(Qz2)
disp('*****')
[mag1, phase1] = freqz (Pz1,Qz1,Wz);
[mag2, phase2] = freqz (Pz2,Qz2,Wz);
subplot (2,2,1)
plot (Wz, mag1)
title ('Mag of Cheby. type/1, HPF');
axis([0 5 0 1.1]);
xlabel (' W in rad')
ylabel ('Magnitude')
subplot (2,2,2)
plot (Wz, phase1* 180/pi)
title ('Phase of Cheby. type/1,HPF ')
ylabel ('Angle in degrees')
xlabel (' W in rad')
subplot (2,2,3)
plot (Wz, mag2)
title ('Mag.of Cheby. type/2, HPF'); axis([0 1 0 1.5]);
xlabel ('W in rad'); ylabel ('Magnitude')
subplot (2,2,4)
plot (Wz, phase2 * 180/ pi);
xlabel ('W in rad')
ylabel ('Angle in degrees')
title('Phase. of Cheby. type/2, HPF')
disp('***** FILTERS TRANSFER FUNCTIONS*****')
disp('*****')
disp('Chebyshev type 1')
tf(Pz2,Qz2,T)
disp('^^^^^^^^^^^^^^^^^')
disp('Chebyshev type 2')
tf(Pz1,Qz1,T)
disp('*****')
```

The script file *cheby_HPF* is executed and the results are shown as follows

```
>> cheby_HPF
```

```
*****
***** R E S U L T S *****
*****
```

For the transfer function H(z) of the Cheby/type1 HPF the numerators coefficients are:

0.8006 -2.4019 2.4019 -0.8006

DIGITAL CONTROL SYSTEMS (ES-413)

The denominator coefficients are:

1.0000 -2.5767 2.2067 -0.6217

For the transfer function $H(z)$ of the Cheby/type2 HPF The numerators coefficients are:

0.9564 -2.8530 2.8530 -0.9564

The denominator coefficients are:

1.0000 -2.8943 2.8098 -0.9147

```
*****
*****FILTERS TRANSFER FUNCTIONS *****
*****
Chebyshev type 1
```

Transfer function:

0.8006 z^3 - 2.402 z^2 + 2.402 z - 0.8006

z^3 - 2.577 z^2 + 2.207 z - 0.6217

Sampling time: 0.3
^^^^^^^^^^^^^^^^^

Chebyshev type 2

Transfer function:

0.9564 z^3 - 2.853 z^2 + 2.853 z - 0.9564

z^3 - 2.894 z^2 + 2.81 z - 0.9147

Sampling time: 0.3

.....



ENGINEERING

DEPARTMENT OF ELECTRONIC

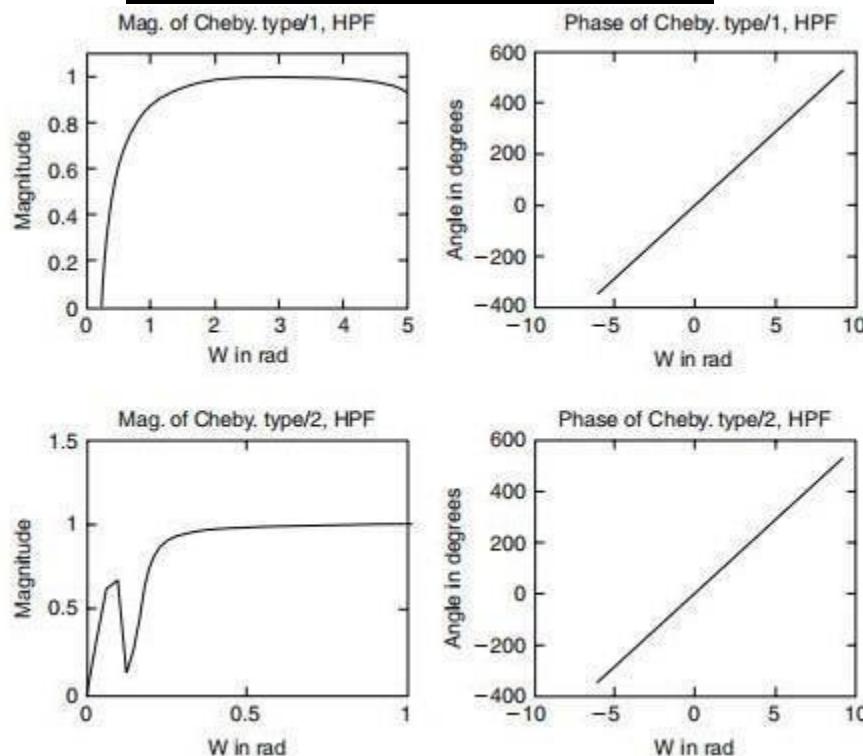
DIGITAL CONTROL SYSTEMS (ES-413)

Figure-14.5: Magnitude and phase plots of Chebyshev type-1 and -2 digital HPFs

Example5: Test the Butterworth LP digital filter by applying as an input the signal $x(t) = 2 + \cos(0.2 * t) + \cos(4 * t)$ sampled every $T = 0.3$ s.

Create the script file `inp_out` that returns the following plots:

1. $x(t)$ versus t
2. $y(t)$ versus t
3. $x(nT)$ versus nT
4. $y(nT)$ versus nT where y denotes the filter's output.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)



MATLAB Code:

```
% Script file: inp_out
Pz = [0.0380 0.1141 0.1141 0.0380];
Qz = [1.0000 -1.3445 0.8215 -0.172];
n = 0:1:100; %Discrete time sequence
T = 0.3;
input = 2 + cos(0.2*T.*n)+cos(4*T.*n);
output= filter(Pz,Qz,input);
t = 0:.05:8; % time range
inputa = 2 + cos(0.2.*t) + cos(4.*t);
subplot(2,2,1)
plot(t,inputa)
xlabel('t in sec')
ylabel('Amplitude')
title('x(t) = 2 + cos(0.2.*t) + cos (4.*t)] vs. t')
axis([0 5 0 5])
subplot(2,2,2)
plot(n*T,output)
xlabel('t in sec')
ylabel('Amplitude')
title('y(t) vs. t');axis([0 8 0 4])
subplot(2,2,3)
stem(n*T,input)
xlabel('time index nT')
ylabel('Amplitude')
title('x(nT) vs. nT');
axis([0 5 0 5])
subplot(2,2,4)
stem(n*T,output)
xlabel('time index nT')
ylabel('Amplitude')
title('y(nT) vs. nT');
axis([0 8 0 4])
```

The script file *inp_out* is executed and the results are shown in Figure 6.60.

```
>> inp_out
```

Note that the filter's output presents a high DC component plus the low frequency of the input with some traces of the high frequency.



MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)

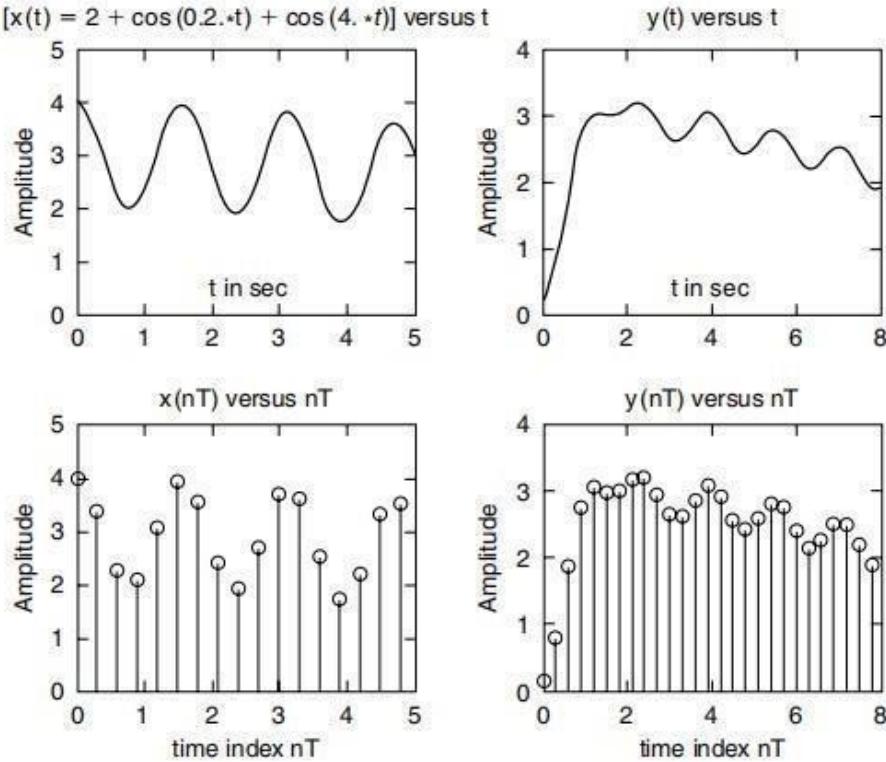


Figure-14.6: Analog and discrete filter's input and output

REVIEW EXERCISE?

a) **Why do we use digital filters in digital control system?**

Digital filters play a crucial role in digital control systems by providing signal processing capabilities, flexibility, adaptability, ease of implementation, precision, and integration with control algorithms, ultimately improving system performance and reliability.

b) **What is FIR digital filter?**

A Finite Impulse Response (FIR) digital filter is a type of digital filter used in digital signal processing and digital control systems. It is characterized by having a finite impulse response, meaning that its output response to an input signal is of finite duration. In simpler terms, an FIR filter takes a sequence of input samples and produces a sequence of output samples by convolving the input samples with a set of coefficients called the filter taps.

c) **What are types of digital filters?**



DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

FIR (Finite Impulse Response) Filter: A digital filter with a finite impulse response, meaning the output is determined by a finite number of input samples and filter coefficients.

IIR (Infinite Impulse Response) Filter: A digital filter with an impulse response that extends infinitely, typically involving feedback loops for recursive calculations.

Butterworth Filter: An IIR filter with a maximally flat frequency response in the passband and a gradual roll-off in the stopband.

Chebyshev Filter: An IIR filter providing steeper roll-off in the stopband (Type I) or passband (Type II) at the expense of ripple in the opposite region.

Elliptic Filter (Cauer Filter): An IIR filter with the steepest roll-off among common types while allowing specified ripple levels in both passband and stopband.

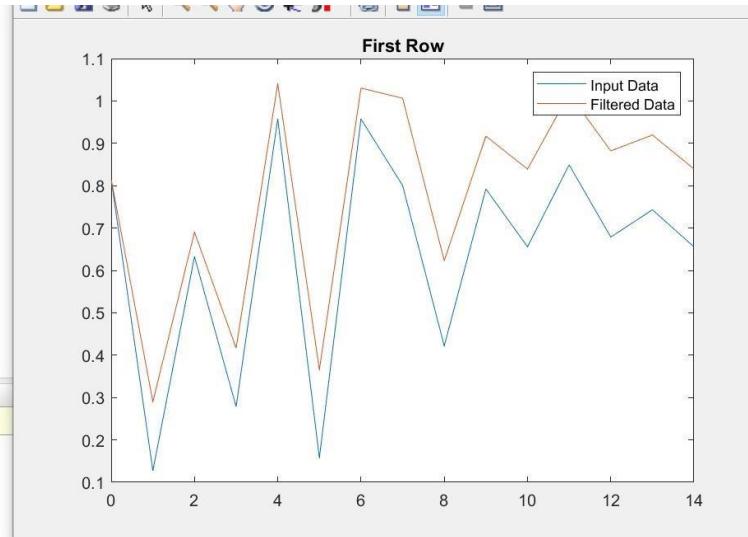
d) Attach screenshots of all examples done in this laboratory. Example:1

```
2 - rng default %initialize random number generator
3 - x = rand(2,15);
4 - b = 1;
5 - a = [1 -0.2];
6 - y = filter(b,a,x,[],2);
7 - t = 0:length(x)-1; %index vector
8 - plot(t,x(1,:))
9 - hold on
10 - plot(t,y(1,:))
11 - legend('Input Data','Filtered Data')
12 - title('First Row')
13
14
```

Command Window

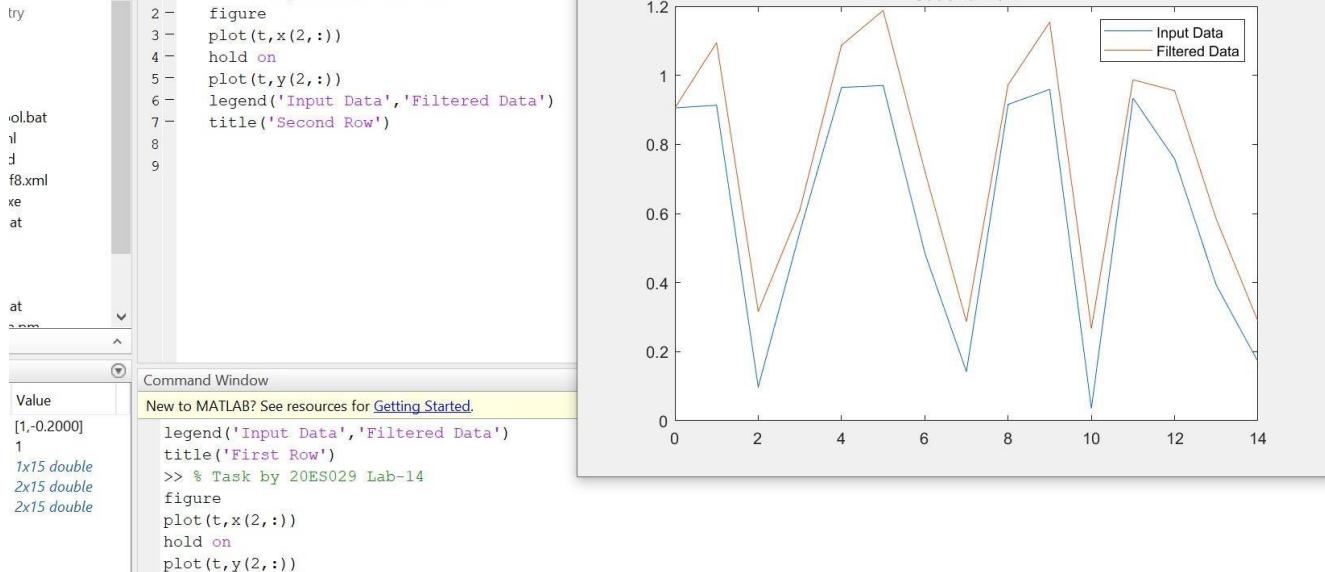
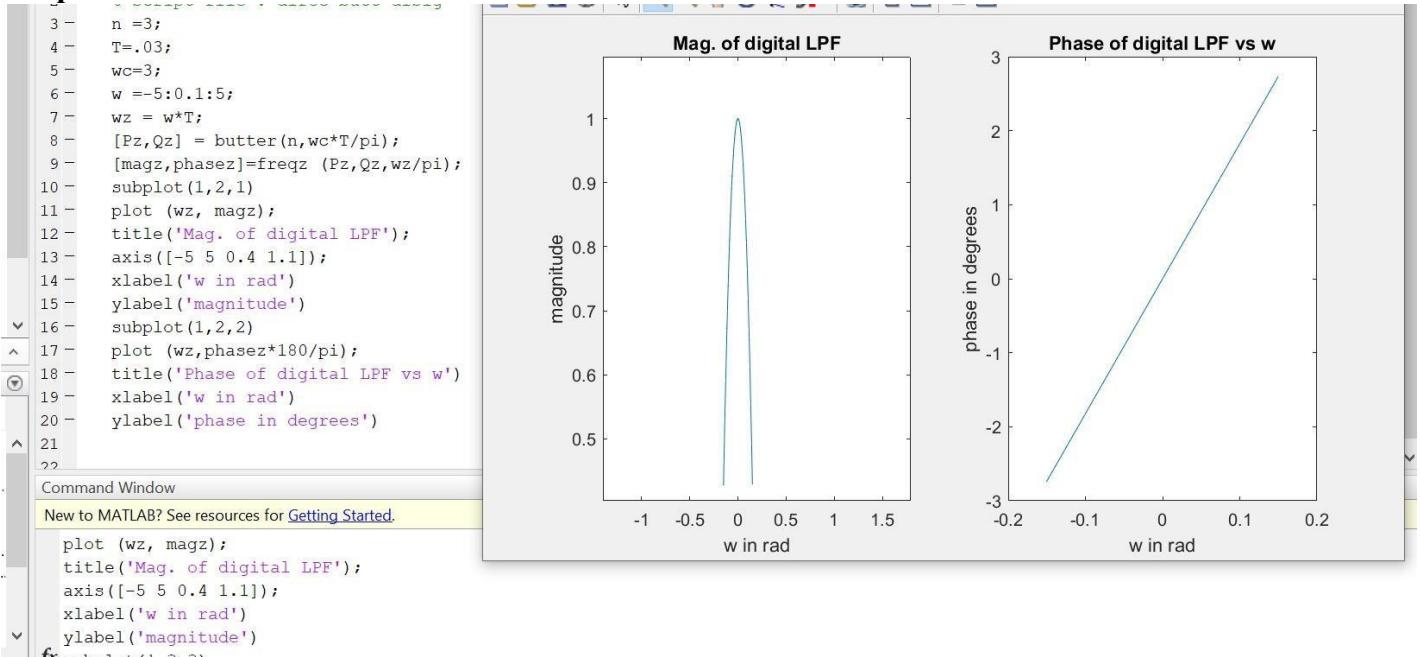
New to MATLAB? See resources for [Getting Started](#).

```
b = 1;
a = [1 -0.2];
y = filter(b,a,x,[],2);
t = 0:length(x)-1; %index vector
plot(t,x(1,:))
hold on
```

**Example:2**



DEPARTMENT OF ELECTRONIC ENGINEERING

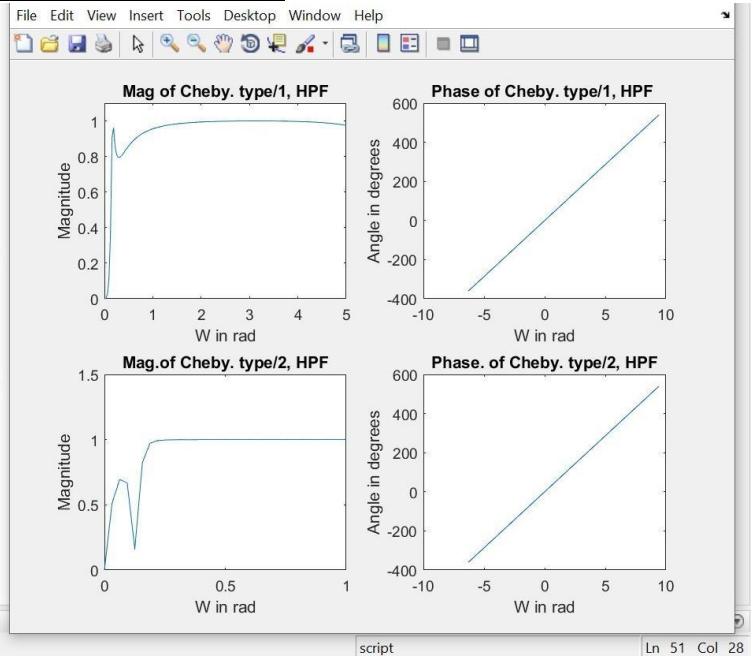
DIGITAL CONTROL SYSTEMS (ES-413)**Example:3****Example:4**



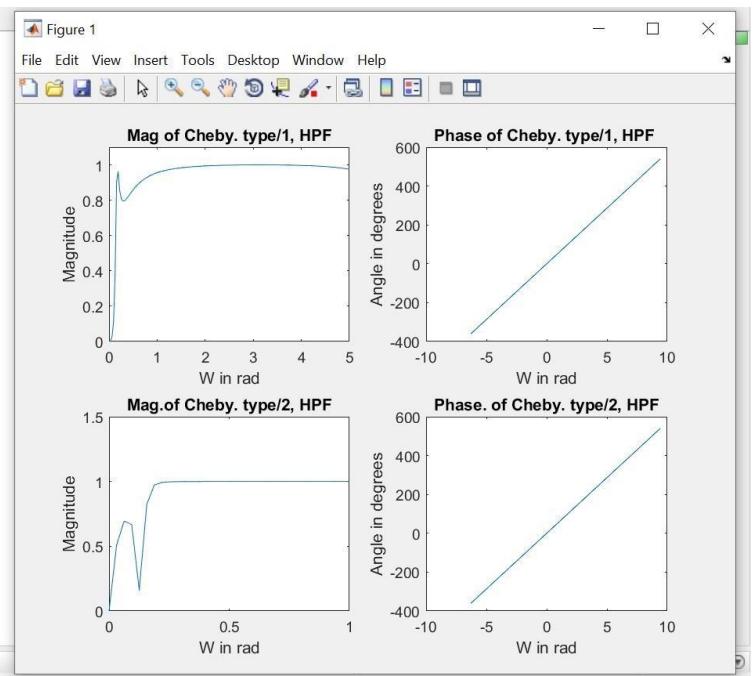
DEPARTMENT OF ELECTRONIC ENGINEERING

DIGITAL CONTROL SYSTEMS (ES-413)

```
2 - N = 3;
3 - Rp = 2;
4 - R = 3;
5 - T = 0.3;
6 - Wc = 0.5;
7 - Wz = -2*pi:0.01*pi:3*pi;
8 - wzc = Wc*T/pi;
9 - [Pz1, Qz1] = cheby1(N, Rp, wzc, 'high');
10 - disp('*****RESULTS *****')
11 - disp('*****RESULTS *****')
12 - disp('*****RESULTS *****')
13 - disp('For the transfer function H(z) of the Cheby/type1 HPF')
14 - disp('the numerators coefficients are:')
15 - disp(Pz1)
16 - disp('The denominator coefficients are:')
17 - disp(Qz1)
18 - [Pz2, Qz2] = cheby2(N, R, wzc, 'high');
19 - disp('For the transfer function H(z) of the Cheby/type2 HPF')
20 - disp('The numerators coefficients are:')
21 - disp(Pz2)
22 - disp('The denominator coefficients are:')
23 - disp(Qz2)
24 - disp('*****RESULTS *****')
25 - [mag1, phase1] = freqz(Pz1, Qz1, Wz);
26 - [mag2, phase2] = freqz(Pz2, Qz2, Wz);
27 - subplot(2,2,1)
28 - plot(Wz, abs(mag1))
29 - title('Mag of Cheby. type/1, HPF');
```



```
Untitled.m
27 - subplot(2,2,1)
28 - plot(Wz, abs(mag1))
29 - title('Mag of Cheby. type/1, HPF');
30 - axis([0 5 0 1.1]);
31 - xlabel('W in rad')
32 - ylabel('Magnitude')
33 - subplot(2,2,2)
34 - plot(Wz, phase1 * 180/pi)
35 - title('Phase of Cheby. type/1, HPF ')
36 - ylabel('Angle in degrees')
37 - xlabel('W in rad')
38 - subplot(2,2,3)
39 - plot(Wz, abs(mag2))
40 - title('Mag.of Cheby. type/2, HPF');
41 - axis([0 1 0 1.5]); xlabel('W in rad'); ylabel('Magnitude')
42 - subplot(2,2,4)
43 - plot(Wz, phase2 * 180/pi);
44 - xlabel('W in rad')
45 - ylabel('Angle in degrees')
46 - title('Phase. of Cheby. type/2, HPF')
47 - disp('** FILTERS TRANSFER FUNCTIONS**')
48 - disp('*****RESULTS *****')
49 - disp('Chebyshev type 1')
50 - tf(Pz1,Qz1,T)
51 - disp('~~~~~')
52 - disp('Chebyshev type 2')
53 - tf(Pz2,Qz2,T)
54 - disp('*****RESULTS *****')
55 -
```





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)



Editor - C:\Users\PMLS\Untitled.m

New to MATLAB? See resources for [Getting Started](#).

```
*****
*****RESULTS *****
*****
For the transfer function H(z) of the Cheby/type1 HPF
the numerators coefficients are:
    0.8006
    -2.4019
    2.4019
    -0.8006

The denominator coefficients are:
    1.0000
    -2.5767
    2.2067
    -0.6217

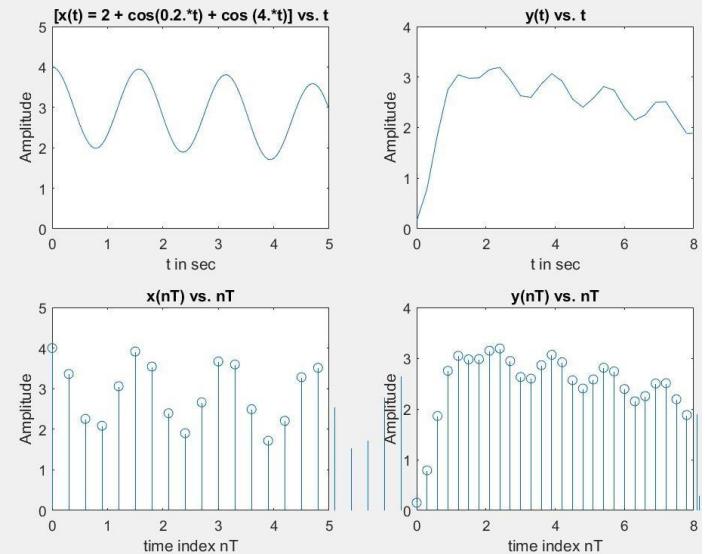
For the transfer function H(z) of the Cheby/type2 HPF
The numerators coefficients are:
    0.9564
    -2.8530
    2.8530
    -0.9564

The denominator coefficients are:
    1.0000
    -2.8943
    2.8098
    -0.9147
```

Example: 5

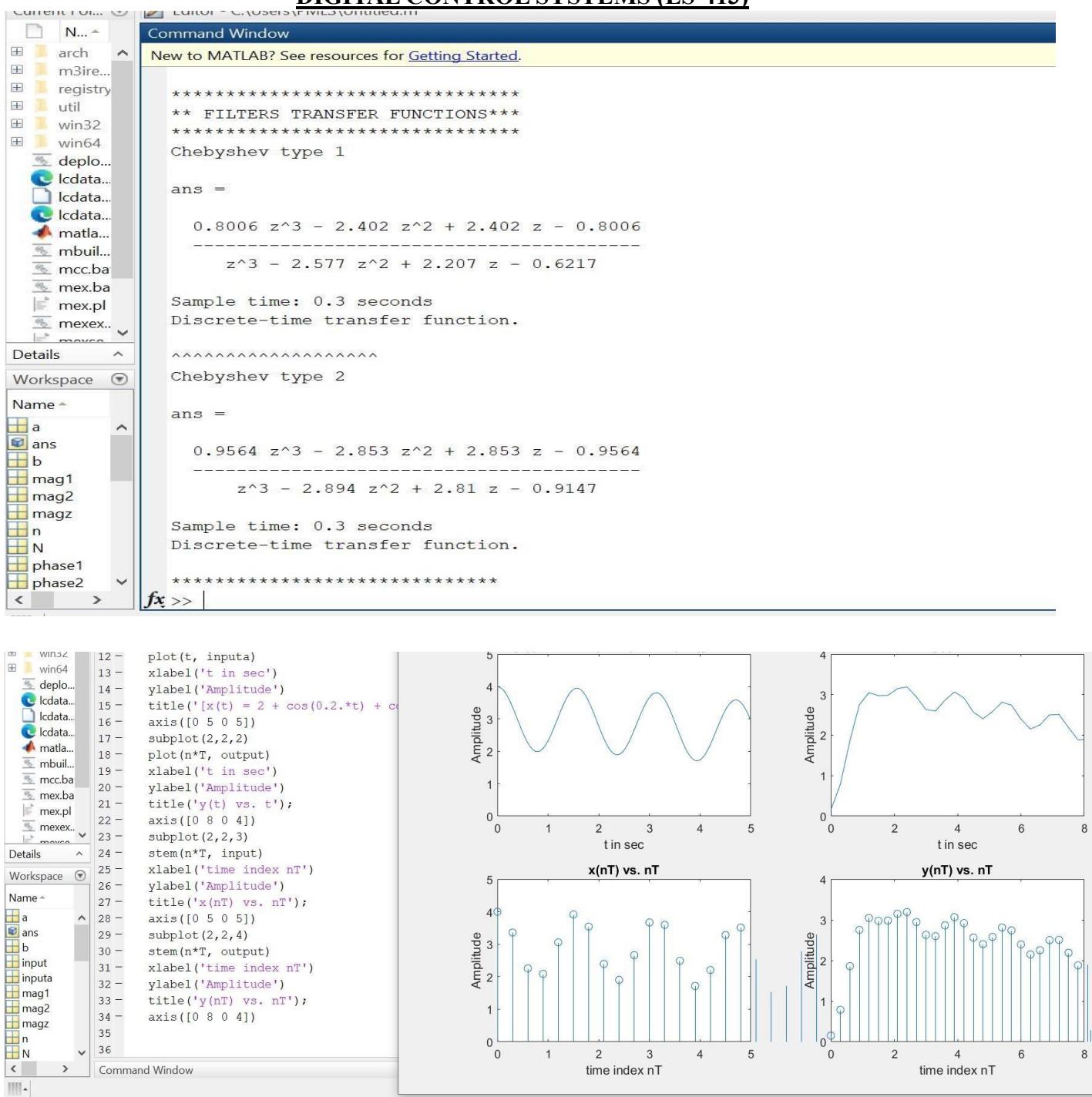
Script file: inp_out

```
2 % Script file: inp_out
3 Pz = [0.0380 0.1141 0.1141 0.0380];
4 Qz = [1.0000 -1.3445 0.8215 -0.172];
5 n = 0:1:100; % Discrete time sequence
6 T = 0.3;
7 input = 2 + cos(0.2*T.*n) + cos(4*T.*n);
8 output = filter(Pz, Qz, input);
9 t = 0:0.05:8;
10 inputs = 2 + cos(0.2.*t) + cos(4.*t);
11 subplot(2,2,1)
12 plot(t, input)
13 xlabel('t in sec')
14 ylabel('Amplitude')
15 title('[x(t) = 2 + cos(0.2.*t) + cos (4.*t)] vs. t')
16 axis([0 5 0 5])
17 subplot(2,2,2)
18 plot(n*T, output)
19 xlabel('t in sec')
20 ylabel('Amplitude')
21 title('y(t) vs. t')
22 axis([0 8 0 4])
23 subplot(2,2,3)
24 stem(n*T, input)
25 xlabel('time index nT')
26 ylabel('Amplitude')
27 title('x(nT) vs. nT');
28 axis([0 5 0 5])
29 subplot(2,2,4)
30 stem(n*T, output)
31 xlabel('time index nT')
32 ylabel('Amplitude')
33 title('y(nT) vs. nT')
34 axis([0 8 0 4])
```





MEHRAN UNIVERSITY OF ENGINEERING & TECHNOLOGY, JAMSHORO
DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL CONTROL SYSTEMS (ES-413)



**LAB # 15: LOW PASS,HIGH PASS & BAND PASS DIGITAL FILTERS**

Name: KARAN	Roll # 20ES062
Signature of Lab Tutor	Date

OBJECTIVE(S)

#	Topic	# Of Lectures	CLO	Taxonomy level
1	To illustrate Low Pass, High Pass an Bandpass Digital filters in MATLAB using Simulink modeling	3	4,5	P3, A4

OUTCOME(S)

a. An ability to use the techniques, skills, and modern engineering tools necessary for engineering practice.	PLO5: Modern Tool Usage
b. An ability to communicate effectively (written/oral)	PLO10: Communication

LAB RUBRICS:

Performance Metric	Good (5-4)	Average (3-2)	Poor (1-0)	Score
Use of modern engineering software [PLO5]	Demonstrates knowledge and application of modern engineering software through accurate development and interpretation of computer programs to solve problems.	Demonstrates awareness of modern engineering software through mostly correct development and interpretation of computer programs to solve problems, but may contain minor mistakes or syntax errors	Unable to use modern engineering software to develop or interpret computer programs to solve problems.	
Level of understanding of the learned skill [PLO5]	Demonstration of full knowledge of the handout with explanations and elaboration.	At ease with content and able to elaborate and explain to some degree.	No grasp of information Clearly no knowledge of subject matter. No questions are answered. No interpretation made.	
Responsiveness to Questions/ Accuracy [PLO10]	Responds well, quick and very accurate all the time.	Generally responsive and accurate most of the times.	Non-responsive at all or the candidate giving only one correct response of viva voce questions.	
Documentation: contents and organization [PLO10]	Report well organized, Appropriately sectioned, uses diagram/description when appropriate, important issues clearly stated.	Report reasonably well documented. May lack some minor aspects.	Report not well organized lack key aspects.	
TOTAL				

**TOOLS:**

- MATLABR-2016a or higher version

BACKGROUND:

Filters are used to eliminate unwanted signal components such as noise and distortion, or to emphasize or correct certain signals (equalizers in stereo systems are generally just a bank of several bandpass filters). For example, a lowpass filter is used to suppress high frequency components of a signal. An anti-aliasing filter (an analog lowpass filter) is used prior to sampling of a continuous signal in order to get rid of high frequency components of the signal. This is essential in order to maintain the integrity of the low frequency components during sampling. Figure 1 shows ideal filter characteristics for lowpass, bandpass, highpass, and bandstop filters.

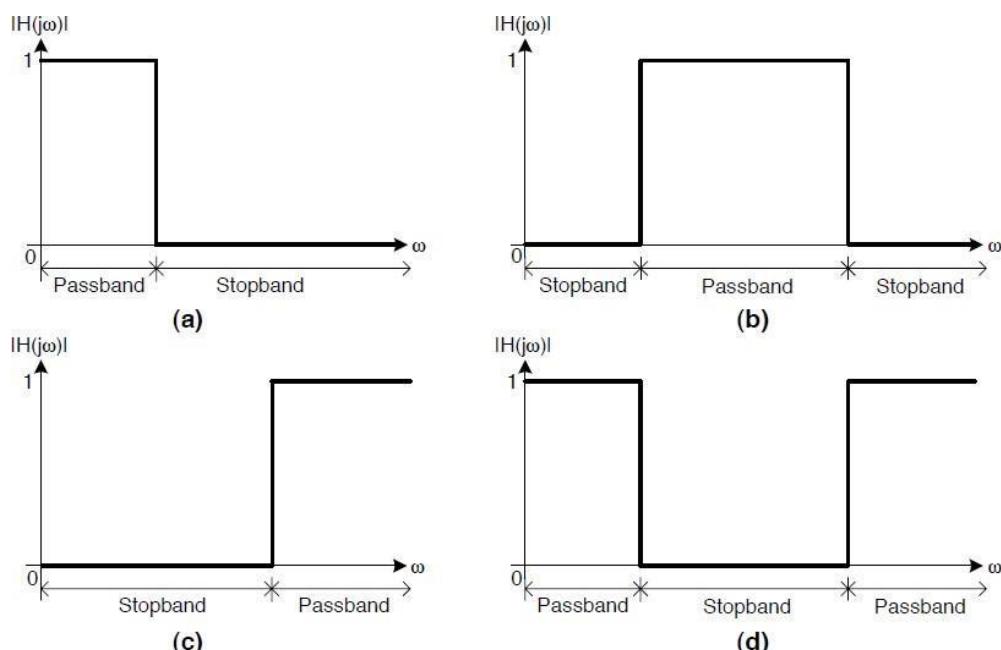


Figure 1: Idea lfilter characteristics offilters

.(a)Lowpassfilter.(b)Bandpassfilter.(c)Highpass filter. (d) Bandstop filter.

As can be seen from Figure 1, an ideal filter has a passband (unity gain) and a stopband (zero gain) with a sudden transition from the passband to the stopband. There is no transition band. However, for practical filters (see Figure 2), the transition takes place over a finite band of frequencies. Moreover, for realizable filters, the gain cannot be zero over a finite band (Paley-Wiener condition).

We therefore define a stopband to be a band over which the gain is below some small number G_s (maximum stopband gain), as illustrated in Figure 2. Similarly, we define a passband to be a band over which the gain is between 1 and some number G_p (minimum passband gain). In our design procedure, G_p , G_s , w_p , and w_s will be specified for lowpass and highpass filters. For bandpass and

DEPARTMENT OF ELECTRONIC ENGINEERING
DIGITAL
CONTROL SYSTEMS (ES-413)

bandstop filters, G_p , G_s , ω_p , ω_s , ω_{p1} , ω_{s1} , ω_{p2} , and ω_{s2} will be specified. In this lab, the gains (G_p and G_s) are specified in decibels (dB), a logarithmic scale, and the frequencies are specified in radians.

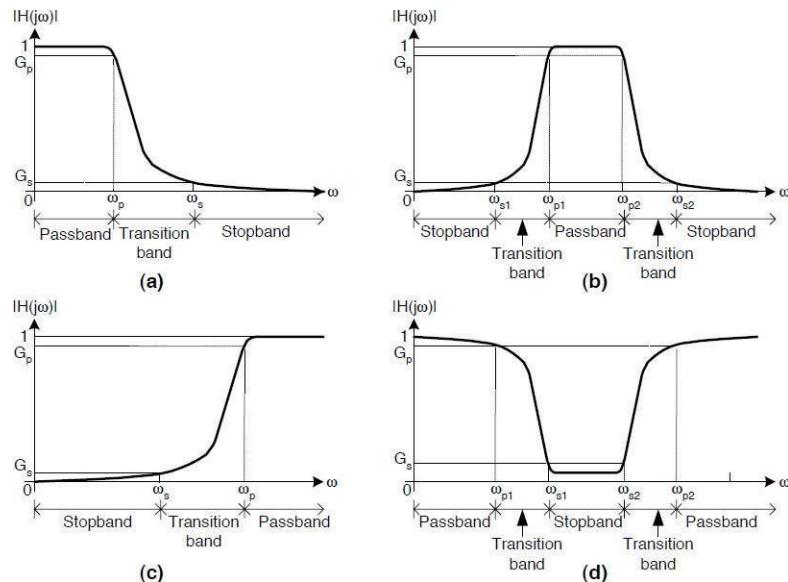
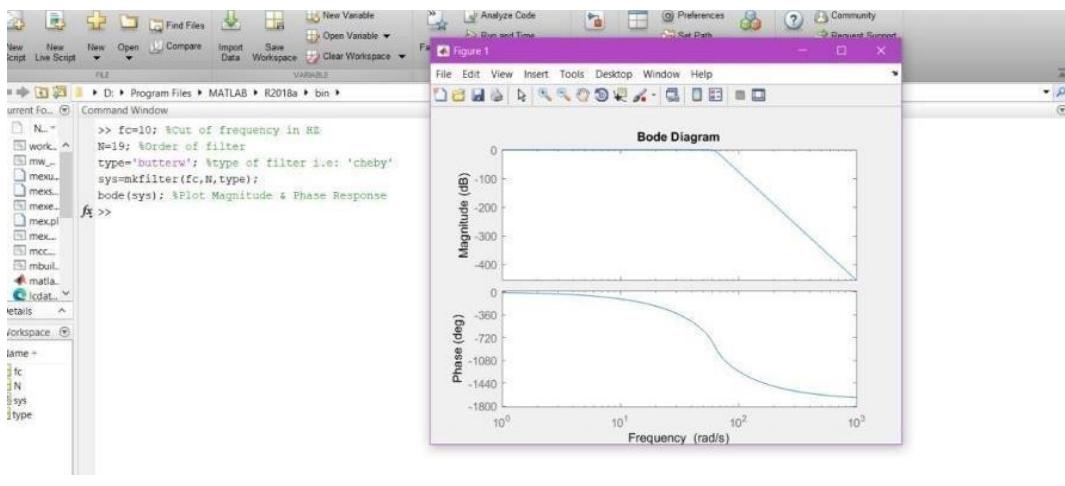
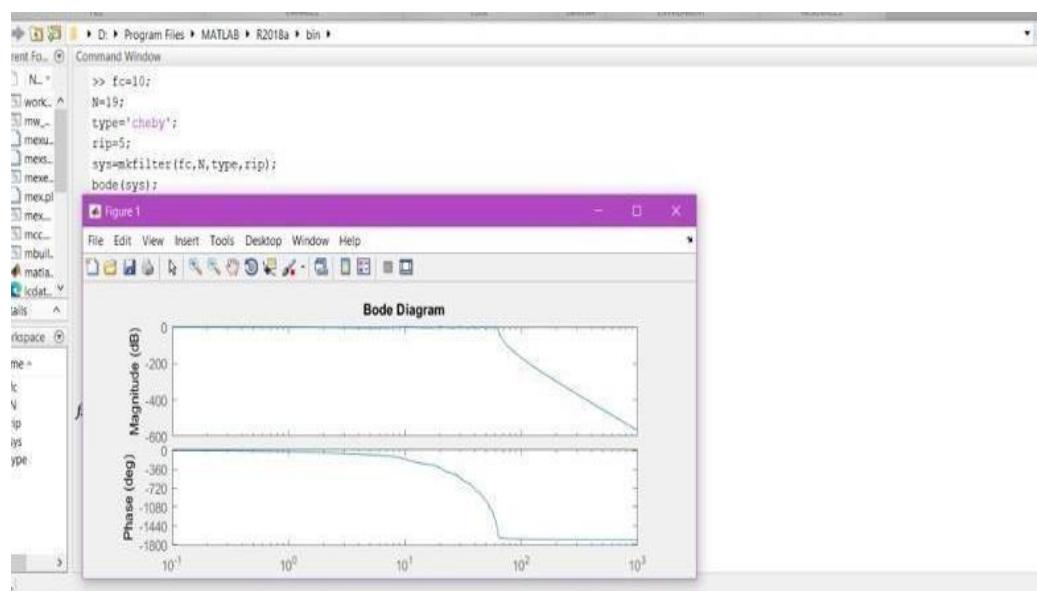


Figure 2: Passband, stopband, and transition band in various types of filters. (a) Lowpass filter. (b) Bandpass filter. (c) Highpass filter. (d) Bandstop filter.

PROCEDURE:**• Designing Continuous Time Filters**

Continuous time Lowpass Butterworth filter

```
fc=10;%Cutoff frequency in Hz N=19; %Order of filter
type='butterw';%type of filter i.e: 'cheby' sys=mkfiltter(fc,N,type);
bode(sys);%Plot Magnitude & Phase Response
```

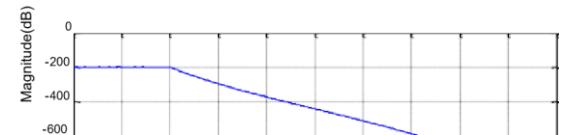
DEPARTMENT OF ELECTRONIC ENGINEERING
CONTROL SYSTEMS (ES-413)DIGITAL**Continuous time Lowpass Chebyshev filter****Butterworth Lowpass Digital Filter with $F_s=40$ & $F_c=4$ Hz** $t = -3 : .025 : 3 ; \text{ % } 40 \text{ Samples / Sec THEREFORE } f = fs / 2 = 20 \text{ hz}$

DEPARTMENT OF ELECTRONIC ENGINEERING
CONTROL SYSTEMS (ES-413)DIGITAL

```
[ba]=butter(25,.2); %Lowpassfilterwithfc=.2x20=4hz figure, freqz(b,a);
```

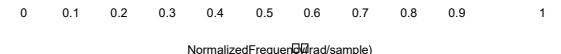
```
%generatingaSinwavewithf=2HZ&passingitthroughFilter x1=sin(2*pi*3*t);
```

```
figure subplot(2,1,1) plot(t,x1);  
axis([-33-22]);
```



```
y1=filter(b,a,x1);  
subplot(2,1,2)
```

```
plot(t,y1) %Outputoffilteratx1 axis([-3 3 -2 2]);
```

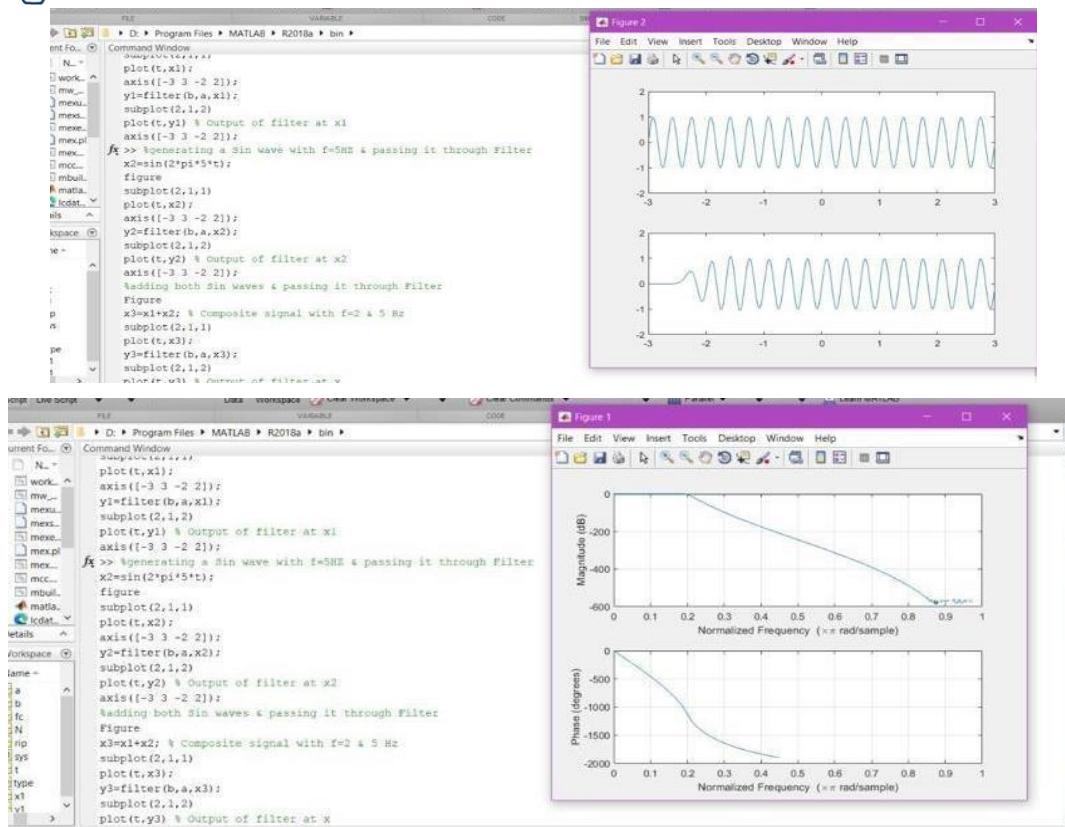


NormalizedFrequency(rad/sample)

```
%generatingaSinwavewithf=5HZ&passingitthroughFilter
```

```
x2=sin(2*pi*5*t); figure  
subplot(2,1,1) plot(t,x2);  
axis([-33-22]);  
y2=filter(b,a,x2);  
subplot(2,1,2)  
plot(t,y2)%Outputoffilteratx2  
axis([-3 3 -2 2]);
```

```
%addingbothSinwaves&passingitthroughFilter Figure  
x3=x1+x2;%Compositesignalwithf=2&5Hz  
subplot(2,1,1) plot(t,x3); y3=filter(b,a,x3);  
subplot(2,1,2)  
plot(t,y3)%Outputoffilteratx1
```

DEPARTMENT OF ELECTRONIC ENGINEERING
CONTROL SYSTEMS (ES-413)DIGITAL



MEHRAN UNIVERSITY OF ENGINEERING AND TECHNOLOGY, JAMSHORO
DIGITAL CONTROL SYSTEMS (ES-413)

