ELECTRICAL

LAB MANUAL

DIGITAL SIGNAL AND PROCESSING LAB

Hamza Mobeen 2018-EE-394

Lab Instructor
Ma'am Munazza Sadaf

Department of Electrical and Electronics Engineering.

University of Engineering and Technology, Lahore.

Faisalabad Campus.

Rubrics of Digital Signal Processing Lab

Performance	Meets Expectations [51%-100%]	Does not meet Expectations [0%-50%]	Marks
Attendance	Shows attendance in lab and takes care of lab timings	Didn't care of lab timings	13
Online Lab performance	Used time well in lab, actively looks for the solutions and submitted lab tasks timely.	Have no idea about the experiment and, its operation. Submitted lab tasks after due time.	26
Lab Manual	Submit the complete lab (Including circuit diagram, procedure and observations/graphs) manual	Submit the lab manual after due time	26
Lab Quiz	Knows the complete theory and applications of the experiments	Doesn't know the complete theory and applications of the experiments	25
Open-Ended Lab			10
Total			100

Approved by OBE Committee

Convener/Chairman of the Department

Dr. Muhammad Akram

Committee Members

Ms. Mariam Mughees

Mr. Azeem Iqbal

Mr. M. Ali Raza

TABLE OF CONTENT

Reg.# 2018-EE-394

University of Engineering & Technology Lahore

Experiment # 1

Title: Starting with MATLAB/MATLAB Refresher

Equipment Required: Personal computer (PC) with windows operating system and MATLAB software

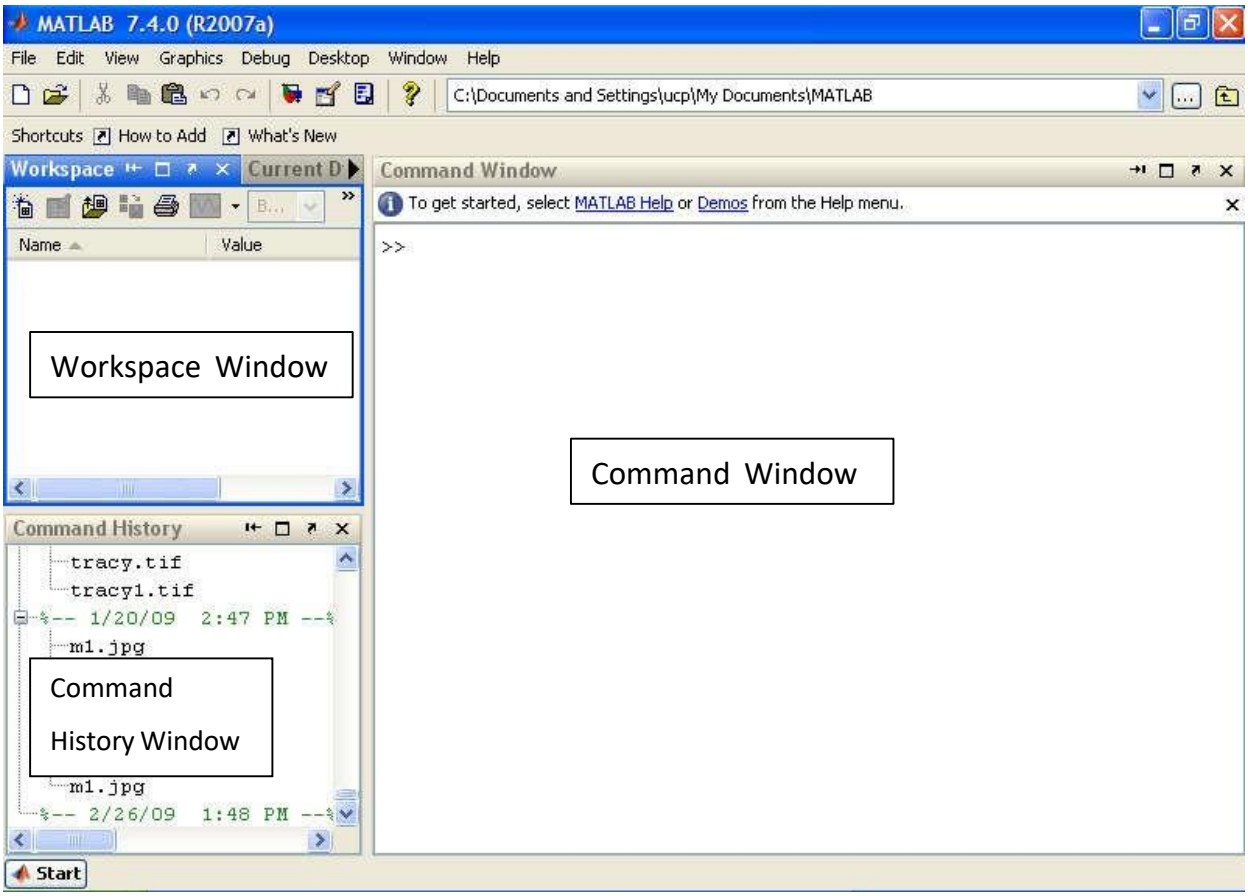
Theory:-

MATLAB is a powerful computing system for handling the calculations involved in scientific and engineering problems. The name MATLAB stands for *MATrix LABoratory,* because the system was designed to make matrix computations particularly easy.

One of the many things about MATLAB (and which distinguishes it from many other computer programming systems, such as C++ and Java) is that you can use it *interactively*. This means you type some commands at the special MATLAB prompt, and get the answers immediately. The problems solved in this way can be very simple, like finding a square root, or they can be much more complicated, like finding the solution to a system of differential equations. For many technical problems you have to enter only one or two commands, and you get the answers at once.

MATLAB WINDOWS:-

Window	Purpose
Command Window	Main window, enters variables, runs programs.
Figure Window	Contains output from graphic commands.
Editor Window	Creates and debugs script and function files.
Help Window	Provides help information.
Launch Pad Window	Provides access to tools, demos, and documentation.
Command History Window	Logs commands entered in the Command Window
Workspace Window	Provides information about the variables that are used.
Current Directory Window	Shows the files in the current directory



Procedure:-

1) To start MATLAB from Windows, double-click the MATLAB icon on your Windows desktop. When MATLAB starts, the MATLAB desktop opens as shown in Figure The window in the desktop that concerns us for this experiment is the Command Window, where the special >> prompt appears. This prompt means that MATLAB is waiting for a command. You can quit MATLAB at any time with one of the following:
   - > Select Exit MATLAB from the desktop File menu.
   - > Enter quit or exit at the Command Window prompt.
2) Once you have started MATLAB, try the following exercises in the Command Window.

   (a) Type 2+3 after the >> prompt, followed by Enter, i.e. press the Enter key, as indicated by <Enter>, below:

   >>2+3 <Enter>

   Commands are only carried out when you press Enter. The answer in this case is, of course, 5.

   (b) Next try the following:

>>3-2 <Enter>
>>2*3 <Enter>
>>1/2 <Enter>
>> 2ˆ3 <Enter>
>>2\1 <Enter>

symbols *, / and ˆ, are multiplication, division and exponentiation. The backslash means the denominator is to the left of the symbol and the numerator is to the right of the symbol; the result for the last command is 0.5. This operation is equivalent to 1/2.

3) Assign values to variables to do arithmetical operations with the variables.

(a) Enter the command a = 2, i.e. the MATLAB command line should look like this:

>>a = 2 <Enter>

The symbol a is called a *variable*. This statement *assigns* the value of 2 to a.

(b)Now enter the statement

>>b = 3; <Enter>

The semicolon (;) prevents the value of b from being displayed.

4) The output format is fixed-point with 4 decimal digits (called short), which is the default format for numerical values. The format can be changed with the formatcommand. Once the format command is entered, all the output that follows is displayed in the specified format. Several of the available formats are listed and described in Table below.

Display formats

Command	Description	Example
format short	Fixed-point with 4 decimal digits for: *0.001 ≤ number ≤*1000 Otherwise display format short e.	>> format short >> 290/7 ans = 41.4286
format long	Fixed-point with 14decimal digits.	>> format long >> 290/7 ans = 41.428571428571431
format short e	Scientific notation with 4 decimal digits.	>> format short e >> 290/7 ans = 4.1429e+001
format long e	Scientific notation with 15 decimal digits.	>> format long e >> 290/7 ans = 4.142857142857143e+001
format short g	Best of 5-digit fixed or floating point.	>> format short g >> 290/7

		ans = 41.429
format long g	Best of I5-digit fixed or floating point.	>> format long g >> 290/7 ans = 41.4285714285714
format bank	Two decimal digits.	>> format bank >> 290/7 ans = 41.43

5) ELEMENTARY MATH BUILT-IN FUNCTIONS

In addition to basic arithmetic operations, expressions in MATLAB can include functions. MATLAB has a very large library of built-in functions. A function has a name and an argument in parentheses. For example, the function that calculates the square root of a number is sqrt(x). Its name is sqrt, and the argument is x.

Elementary math functions

Function	**Description**	**Example**
sqrt(x)	Square root.	>> sqrt(81) ans = 9.00
exp(x)	Exponential *(e^x).*	>> exp(5) ans = 148.41
abs (x)	Absolute value.	>> abs (-24) ans = 24.00
log (x)	Natural logarithm. Base e logarithm (In).	>> log(1000) ans = 6.91
log10(x)	Base 10 logarithm.	>> log10(1000) ans = 3.00
factorial (x)	The factorial function *x!* *(x* must be a positive integer.)	>> factorial (5) ans = 120.00

Function	**Description**	**Example**
sin(x)	Sine of angle *x (x* in radians).	>> sin(pi/6) ans = 0.5000
cos(x)	Cosine of angle *x (x* in radians).	>> cos(pi/6) ans = 0.8660

tan (x)	Tangent of angle *x (x* in radians).	>> tan(pi/6) ans = 0.5774
cot (x)	Cotangent of angle *x (.x* in radians).	>> cot(pi/6) ans = 1.7321

Rounding functions

Function	**Description**	**Example**
round (x)	Round to the nearest integer.	>> round (17/5) ans = 3
fix(x)	Round towards zero.	>> fix (13/5) ans = 2
ceil(x)	Round towards infinity.	>> ceil (11/5) ans = 3
floor (x)	Round towards minus infinity.	>> floor(-9/4) ans = -3
rem(x,y)	Returns the remainder after *x* is divided by *y.*	>> rem(13,5) ans = 3
sign(x)	Signum function. Returns 1 if *x* > 0 , -1 if *x* < 0, and 0 if *x* = 0	>> sign(5) ans = 1

Rules About VariableNames

Variablenames:

- Can be up to 63 characters long
- Can contain letters, digits, and the underscore character
- Must begin With a letter.
- MATLAB is case sensitive; it distinguishes between uppercase and lowercase letters. For example, AA,Aa, aA, and aa are the names of four different variables.
- Avoid using the names of a built-in function for a variable (i.e. avoid using: cos, sin, exp, sqrt, etc.). Once a function name is used to define a variable, the function cannot be used.

Problems:-

Solve the following problems in the Command Window.

1) Calculate:

a) $\frac{35.7 * 64 - 7^3}{45 + 5^2}$

b) $\frac{5}{4} * 7 * 6^2 + \frac{3^7}{(9^3 - 652)}$

c) $(2+7)^3 + \frac{273^{2/3}}{2} + \frac{55^2}{3}$

d) $2^3 + 7^3 + \frac{273^3}{2} + 55^{3/2}$

e) $\frac{3^7 \log(76)}{7^3 + 546} + \sqrt[3]{910}$

f) $43 * \frac{\sqrt[4]{250}+23}{e^{(45-3^3)}}$

g) $\cos^{2\left(\frac{5\pi}{6}\right)} \sin\left(\frac{7\pi}{8}\right)^2 + \frac{\tan\left(\frac{\pi}{6}\ln(8)\right)}{\sqrt{7}}$

h) $\cos^{2\left(\frac{5\pi}{6}\right)} \sin\left(\frac{7\pi}{8}\right)^2 + \frac{\tan\left(\frac{\pi}{6}\ln(8)\right)}{\sqrt{7}}$

2) Define the variable x as x = 13.5, then evaluate:

a) $x^3 + 5x^2 - 26.7x - 52$

b) $\frac{\sqrt{14x^3}}{e^{3x}}$

c) $\log\left|x^2 - x^3\right|$

3) Define the variable x and z as x = 9.6, and z=8.1, then evaluate

a) $xz^2 - \left(\frac{2z}{3x}\right)^{\frac{3}{5}}$

b) $\frac{443z}{2x^3} + \frac{e^{-xz}}{(x+z)}$

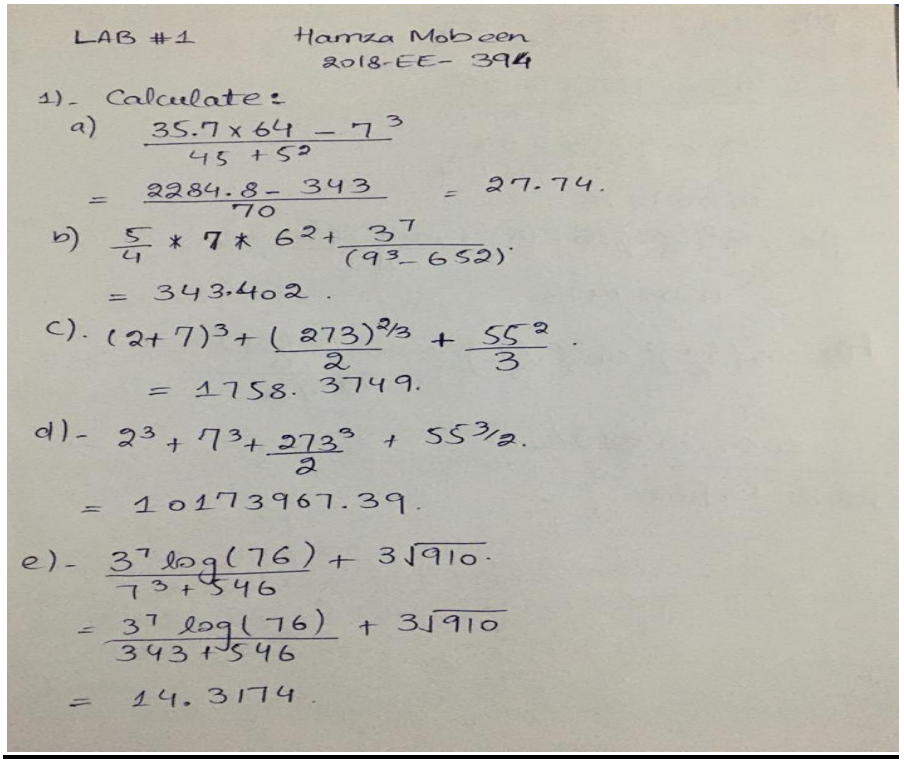
4) Define the variable a, b, c, and d as:
a = 15.62, b =-7.08, c = 62.5 and d = 0.5(ab-c)
evaluate:

a) $a + \frac{ab}{c} * \frac{(a+d)^2}{\sqrt{|ab|}}$

$\left(\frac{d}{-}\right) \quad \frac{ad+cd}{\frac{20}{-} + \frac{30}{-}}$

b) $de^{\left(\frac{-}{2}\right)} + \frac{a \quad b}{a+b+c+d}$

Problem 1:



LAB #1 Hamza Mobeen
2018-EE-394

1). Calculate:

a) $\frac{35.7 \times 64 - 7^3}{45 + 5^2}$

$= \frac{2284.8 - 343}{70} = 27.74.$

b) $\frac{5}{4} * 7 * 6^2 + \frac{3^7}{(9^3 - 652)}$

$= 343.402.$

c). $(2+7)^3 + \frac{(273)^{2/3}}{2} + \frac{55^2}{3}.$

$= 1758.3749.$

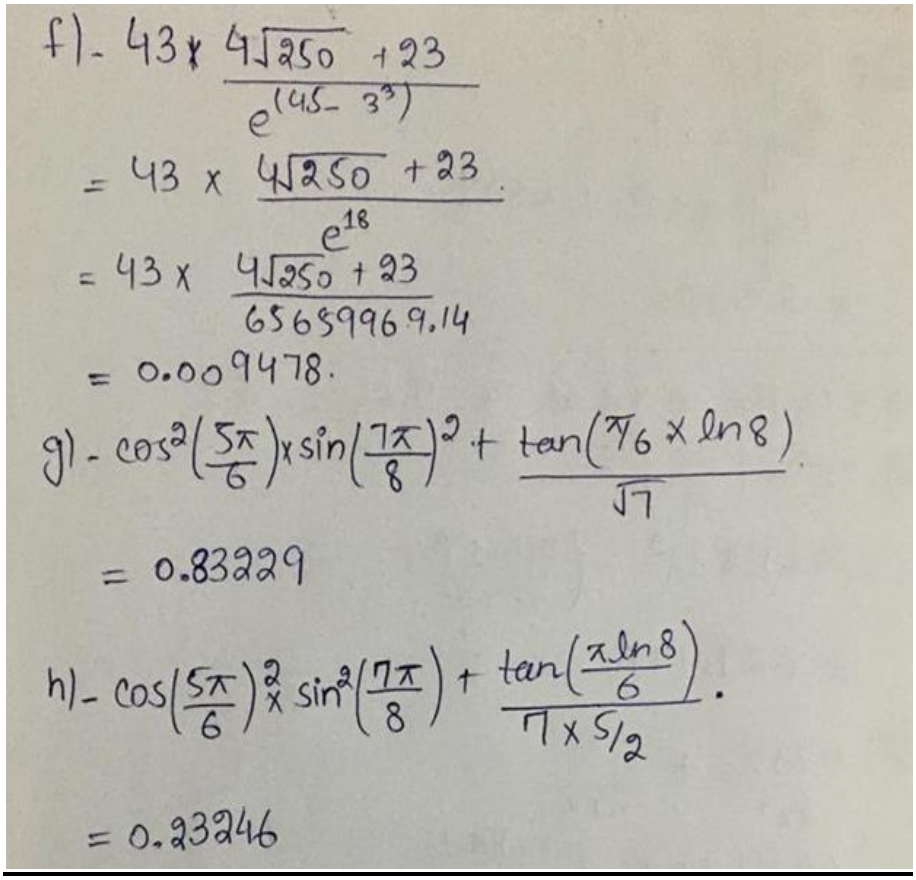
d)- $2^3 + 7^3 + \frac{273^3}{2} + 55^{3/2}.$

$= 10173967.39.$

e)- $\frac{3^7 \log(76)}{7^3 + 546} + 3\sqrt{910}.$

$= \frac{3^7 \log(76)}{343 + 546} + 3\sqrt{910}$

$= 14.3174.$



f)- $43 * \frac{4\sqrt{250} + 23}{e^{(45 - 3^3)}}$

$= 43 \times \frac{4\sqrt{250} + 23}{e^{18}}.$

$= 43 \times \frac{4\sqrt{250} + 23}{65659969.14}$

$= 0.009478.$

g)- $\cos^2\left(\frac{5\pi}{6}\right) \times \sin\left(\frac{7\pi}{8}\right)^2 + \frac{\tan\left(\frac{\pi}{6} \times \ln 8\right)}{\sqrt{7}}.$

$= 0.83229$

h)- $\cos\left(\frac{5\pi}{6}\right)^2 \times \sin^2\left(\frac{7\pi}{8}\right) + \frac{\tan\left(\frac{\pi \ln 8}{6}\right)}{7 \times 5/2}.$

$= 0.23246$

Code:

Part a,b,c:

lab1.m

```
%2018-EE-394
%lab 1
%task 1
a=(35.7*64-7^3)/(45+5^2)
b=(5/4)*7*6^2+((3^7)/((9^3)-652))
c=(2+7)^3+((273^(2/3))/2)+((55^2)/3)
```

Command Window

```
>> lab1

a =

   27.7400


b =

  343.4026


c =

   1.7584e+03
```

Part d,e,f:

lab1.m

```
%2018-EE-394
%lab 1
%task 1
d=2^3+7^3+((273^3)/2)+55^(3/2)
e=((3^7)*log10(76))/((7^3)+546)+(910^(1/3))
f=43*(((250^(1/4))+23)/exp(45-3^3))
```

Command Window

```
>> lab1

d =

   1.0174e+07


e =

   14.3174


f =

   1.7667e-05
```

Part g,h:

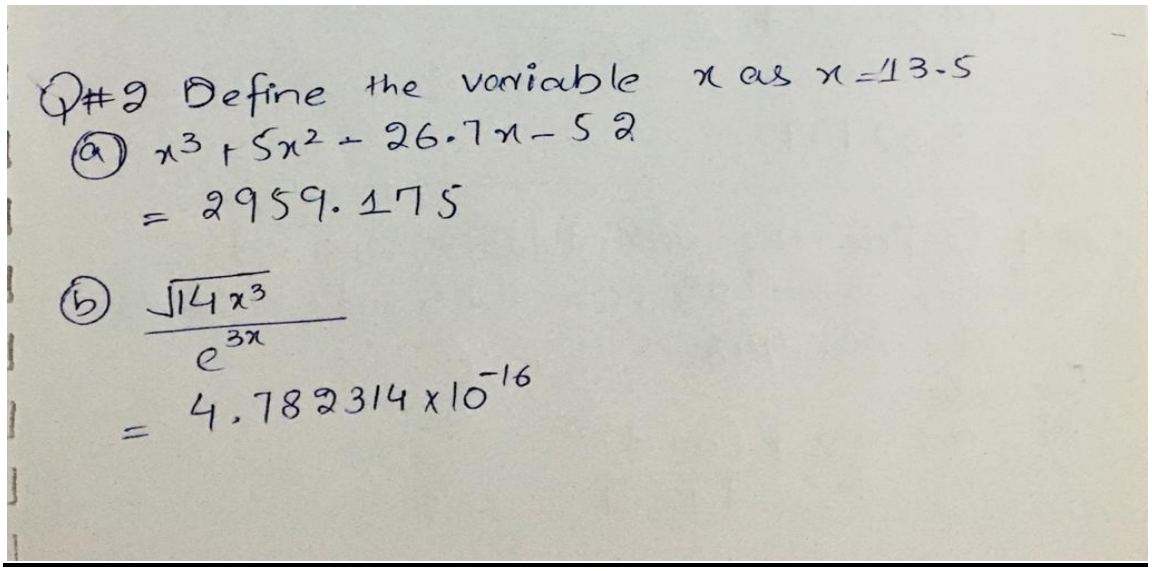
lab1.m

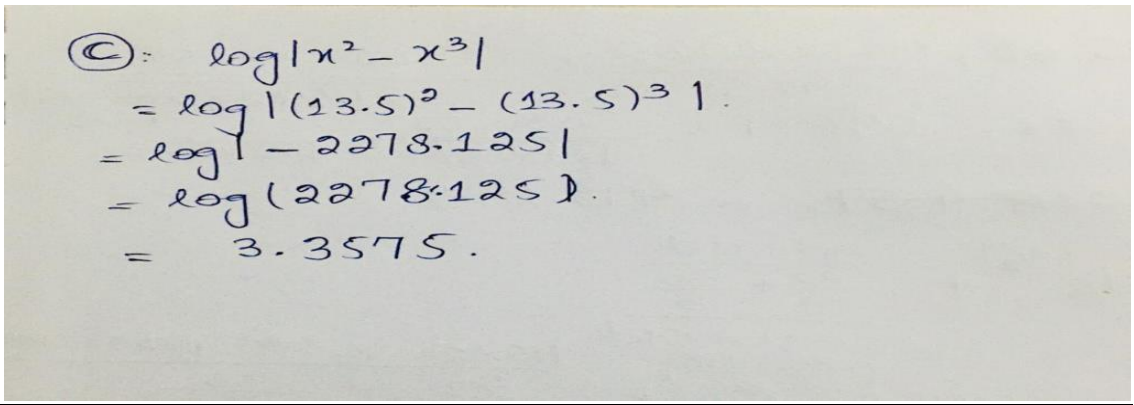
```
%2018-EE-394
%lab 1
%task 1
g=(cos((5*pi)/6))^2*(sin((7*pi/8))^2)+((tan(pi/6*log(8)))/sqrt(7))
h=(cos((5*pi)/6))^2*(sin((7*pi/8))^2)+((tan(pi/6*log(8)))/(7*5/2))
```

Command Window

```
>> lab1

g =

    0.8323


h =

    0.2191
```

Problem 2:



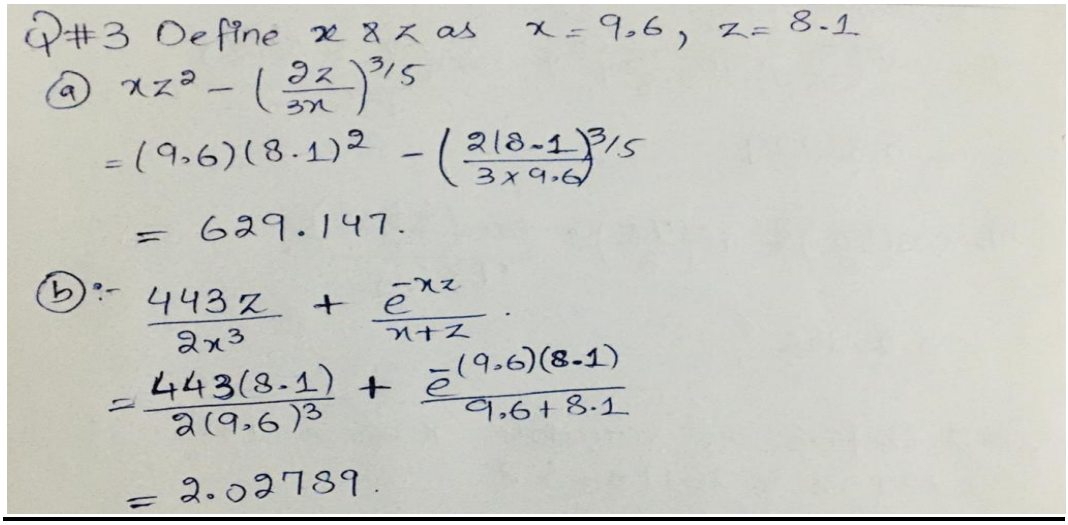


Code:

```
lab1.m
1       %2018-EE-394
2       %lab 1
3       %task 2
4 —     x=13.5;
5 —     a=x^3+5*x^2-26.7*x-52
6 —     b=sqrt(14*(x^3))/exp(3*x)
7 —     c=log10(abs(x^2-x^3))
8
```

```
Command Window
>> lab1

a =

    2.9592e+03


b =

    4.7823e-16


c =

     3.3576
```

Problem 3:



Code:

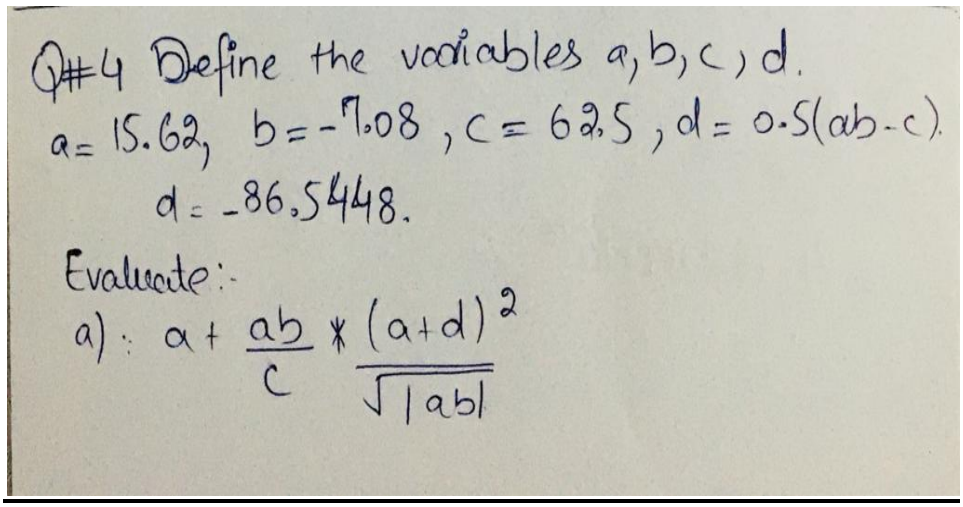
lab1.m

```
%2018-EE-394
%lab 1
%task 2
x=13.5;
a=x^3+5*x^2-26.7*x-52
b=sqrt(14*x)/exp(3*x)
c=log10(abs(x^2-x^3))
```

Command Window

```
>> lab1

a =

   2.9592e+03


b =

   3.5425e-17


c =

    3.3576
```

Problem 4:



$$= 15.62 + \frac{(15.62 \times (-7.08)}{62.5} \times \frac{(15.62 - 86.5448)^2}{\sqrt{|15.62 \times -7.08|}}$$

$$= 80.980738$$

(b) $$de^{(d/2)} + \frac{\frac{ad + cd}{\frac{20}{a} + \frac{30}{b}}}{a+b+c+d}$$

$$= (-86.5448) \times e^{\left(\frac{-86.5448}{2}\right)} + \frac{\frac{115.62 \times -86.5448) + 62.5 \times -86.5448}{\frac{20}{15.6} + 30/-7.08}}{15.62 - 7.08 + 62.5 - 86.5448}$$

$$= -86.5448 e^{-43.2724} - 147.43089.$$

$$= -147.470389.$$

Code:

Part a:

```
%2018-EE-394
%Lab 1
%task 4
a=15.62;
b=-7.08;
c=62.5;
d=0.5*(a*b-c)
A=a+((a+b)/c)*(((a+d)^2)/sqrt(abs(a*b)))
```

Command Window

```
>> lab1

d =

  -86.5448


A =

   80.9807
```

Part b:

lab1.m

```
%2018-EE-394
%lab 1
%task 4
a=15.62;
b=-7.08;
c=62.5;
d=0.5*(a*b-c)
A=(d*(exp(d/2)));
B=(((a*d)+(c*d))/((20/a)+(30/b)));
C=(a+b+c+d);
D=A+(B/C)
```

Command Window

```
>> lab1

d =

  -86.5448


D =

 -147.4700
```

Conclusion:

In this lab we have learnt about the basic commands in MATLAB. We have performed different function in MATLAB as well as using calculator , all of them are simple Mathematical functions and in the end we Came to this conclusion that using MATLAB we can solve difficult problems easily

Lab Task # 2

Submitted To:

Ms.Munazza Sadaf

Submitted By:

Hamza Mobeen

Registration No:

2018-EE-394

Section:

B

Course name:

Digital Signal and Processing Lab

Introduction of Arrays, Two dimensional plots and Elementary sequence.

Assignment 1:

Create a row vector that has the elements: 1, 5, 4, 8, 6, tan(pi/3) and cos(pi/3).

```
Command Window
New to MATLAB? See resources for Getting Started.
>> %2018-EE-394
>> %lab 2 task 1
>> A=[1 5 4 8 6 tan(pi/3) cos(pi/3)]

A =

    1.0000    5.0000    4.0000    8.0000    6.0000    1.7321    0.5000

fx >>
```

Assignment 2:

Creating a vector with constant spacing by specifying the first term, the spacing, and the last term

```
Command Window
>> %2018-EE-394
>> %lab 2 task2
>> A=[1:2:15]

A =

     1     3     5     7     9    11    13    15

fx >>
```

Assignment 3:

Create a row vector in which the first element is 100, the elements decrease with increments of -5 and the last element is 0.

```
Command Window
>> %2018-EE-394
>> %lab2 task 3
>> A=[100:-5:0]

A =

  Columns 1 through 16

   100    95    90    85    80    75    70    65    60    55    50    45    40    35    30    25

  Columns 17 through 21

    20    15    10     5     0

fx >>
```

Assignment 4:

Create a row vector with 10 equally spaced elements in which the first element is 7 and the last element is 40.

```
Command Window
>> %2018-EE-394
>> %lab 2 task 4
>> A=linspace(7,40,10)

A =

  Columns 1 through 9

    7.0000   10.6667   14.3333   18.0000   21.6667   25.3333   29.0000   32.6667   36.3333

  Column 10

   40.0000
```

Assignment 5:

Create a column vector with 12 equally spaced elements in which the first element is -1 and the last element is -15.

```
Command Window
>> %2018-EE-394
>> % lab2 task 5
>> A=[linspace(-1,-15,12)]'

A =

   -1.0000
   -2.2727
   -3.5455
   -4.8182
   -6.0909
   -7.3636
   -8.6364
   -9.9091
  -11.1818
  -12.4545
  -13.7273
  -15.0000

fx >>
```

Assignment 6:

Create the matrix shown below by using the vector notation for creating vectors with constant linspace command when entering the rows.

$$A = \begin{bmatrix} 20 & 30 & 40 \\ 10 & 20 & 30 \\ 0 & 0.5 & 1 \end{bmatrix}$$

```
Command Window
>> %2018-EE-394
>> %lab 2 task 6
>> A=[linspace(20,40,3);linspace(10,30,3);linspace(0,1,3)]

A =

   20.0000   30.0000   40.0000
   10.0000   20.0000   30.0000
         0    0.5000    1.0000

fx >>
```

Assignment 7:

Create the following matrix *A:*

$$A = \begin{bmatrix} 1 & 3 & 5 & 7 & 9 \\ 11 & 13 & 15 & 17 & 19 \\ 21 & 23 & 25 & 27 & 29 \end{bmatrix}$$

Use the matrix *A* to:

a) Create a five-element row vector named va that contains the elements of the second row of *A.*

```
Command Window
>> %2018-EE-394
>> %lab 2 task 7 a
>>  A=[1:2:9;11:2:19;21:2:29]

A =

     1     3     5     7     9
    11    13    15    17    19
    21    23    25    27    29

>> va=A(2,:)

va =

    11    13    15    17    19
```

b) Create a three-element row vector named vb that contains the elements of the fourth column of A.

```
Command Window
>> %2018-EE-394
>> % lab2 task 7b
>> A=[1:2:9;11:2:19;21:2:29]

A =

     1     3     5     7     9
    11    13    15    17    19
    21    23    25    27    29

>> vb=[A(:,4)]'

vb =

     7    17    27

fx >>
```

c) Create a ten-element row vector named vc that contains the elements of the first and second rows of *A.*

```
Command Window
>> %2018-EE-394
>> %lab2 task 7c
>> A=[1:2:9;11:2:19;21:2:29]

A =

     1     3     5     7     9
    11    13    15    17    19
    21    23    25    27    29

>> vc=[A(1,:),A(2,:)]

vc =

     1     3     5     7     9    11    13    15    17    19

fx >>
```

d) Create a six-element row vector named vd that contains the elements of the second and fifth columns of *A.*

```
Command Window
>> %2018-EE-394
>> %lab2 task 7d
>> A=[1:2:9;11:2:19;21:2:29]

A =

     1     3     5     7     9
    11    13    15    17    19
    21    23    25    27    29

>> vd=[A(:,2)',A(:,5)']

vd =

     3    13    23     9    19    29

fx >>
```

Assignment 8:

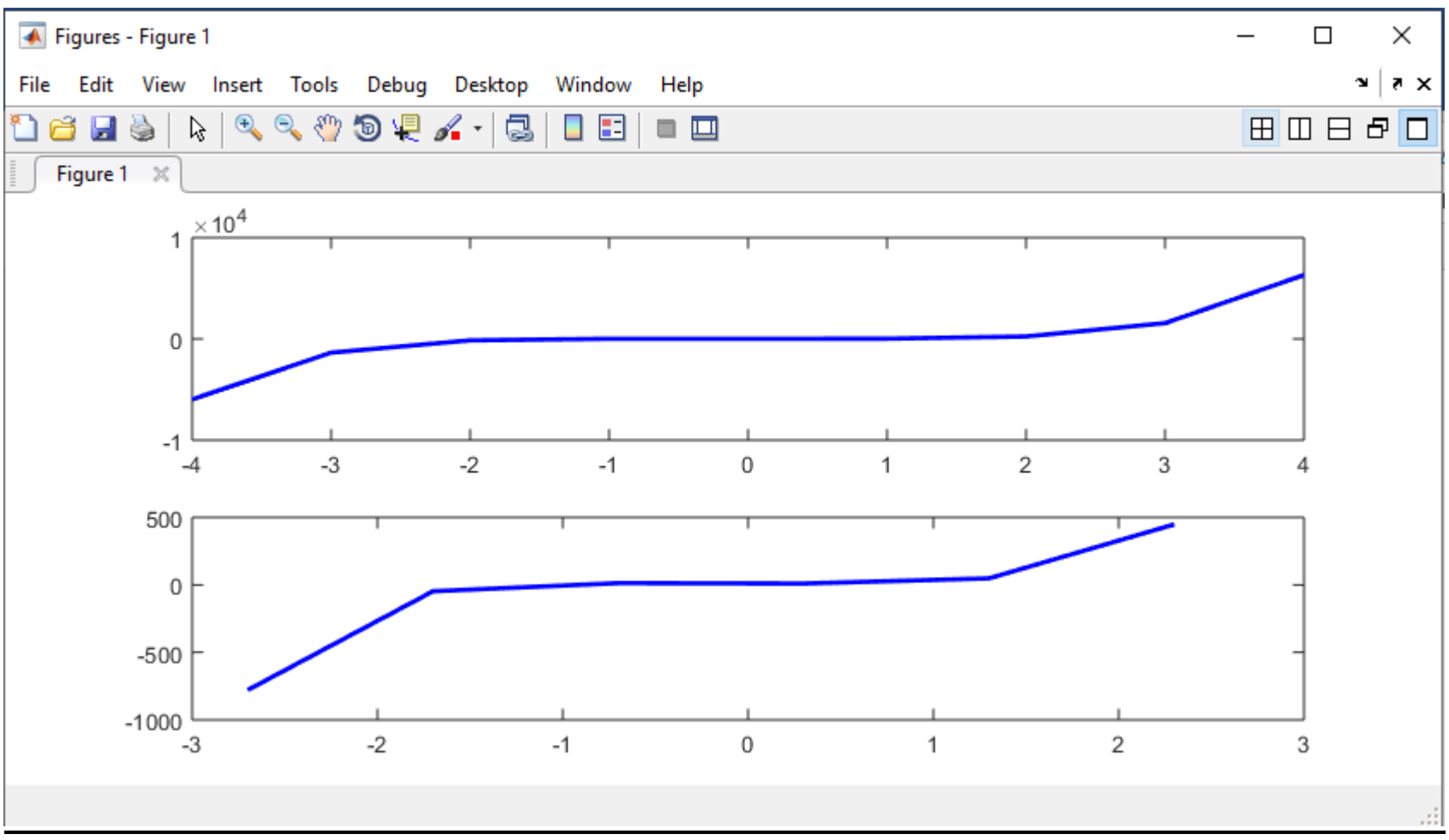
Solve the following problems in MATLAB Command Window.

1) Make two separate plots of the function $f(x) = 6x^5 + 10x^2 + 9$, one plot for -4≤x≤4, and one for -2.7≤x≤2.7.

Code:

```
Command Window
>> %2018-EE-394
>> %lab2 task 8a
>> subplot(2,1,1);
>> x=[-4:4];
>> fx=(6*x.^5)+(10*x.^2)+9;
>> plot(x,fx)
>> subplot(2,1,2);
>> y=[-2.7:2.7];
>> fy=(6*x.^5)+(10*x.^2)+9;
>> plot(y,fy)
```

Output:

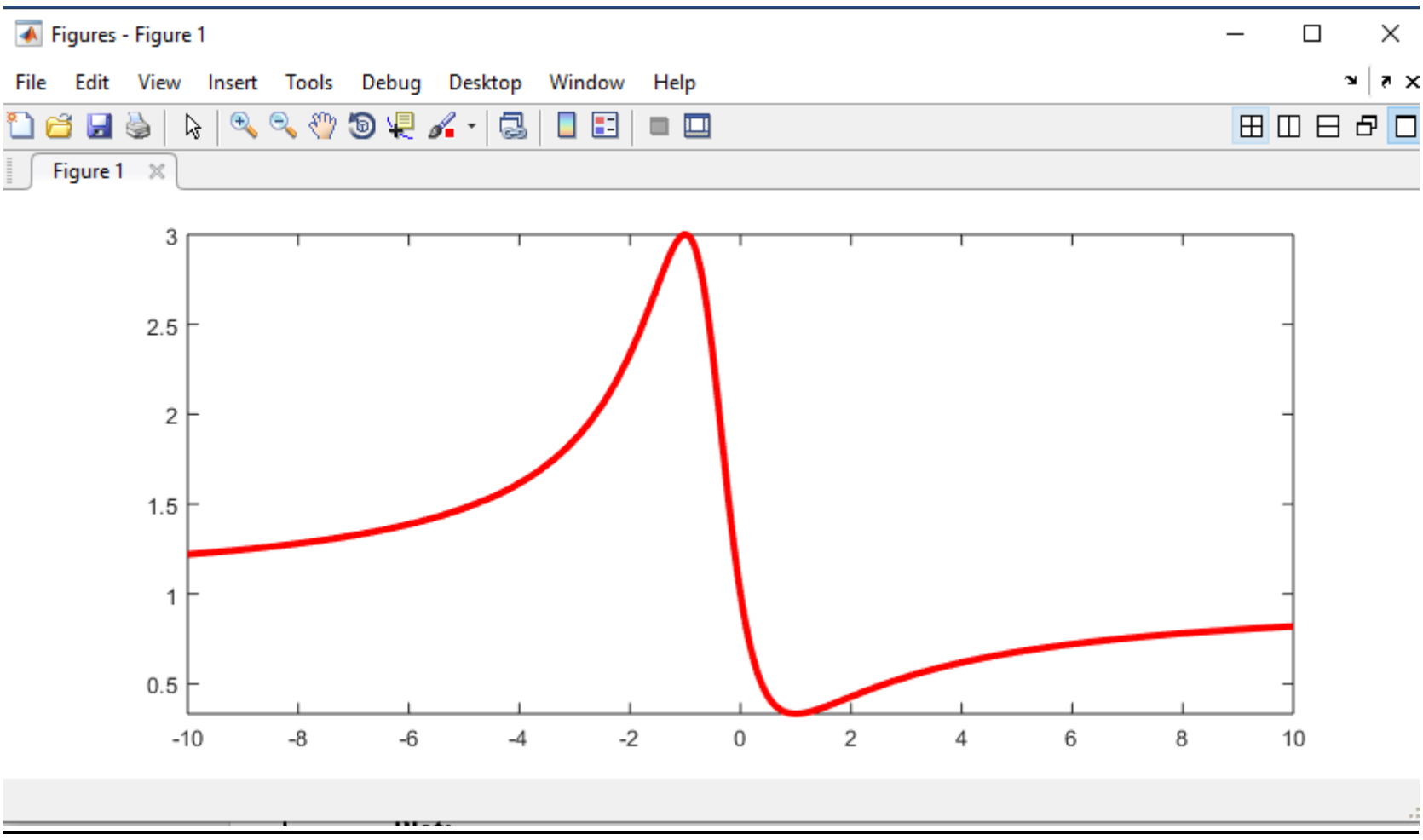


2) Plot the function $f(x) = \frac{x^2-x+1}{x^2+x+1}$ for -10 ≤ x ≤ 10

Code:

```
Command Window
>> %2018-EE-394
>> %lab 2 task 8b
>> fplot(@(x)(x.^2-x+1)/(x.^2+x+1),[-10,10])
```

Output:

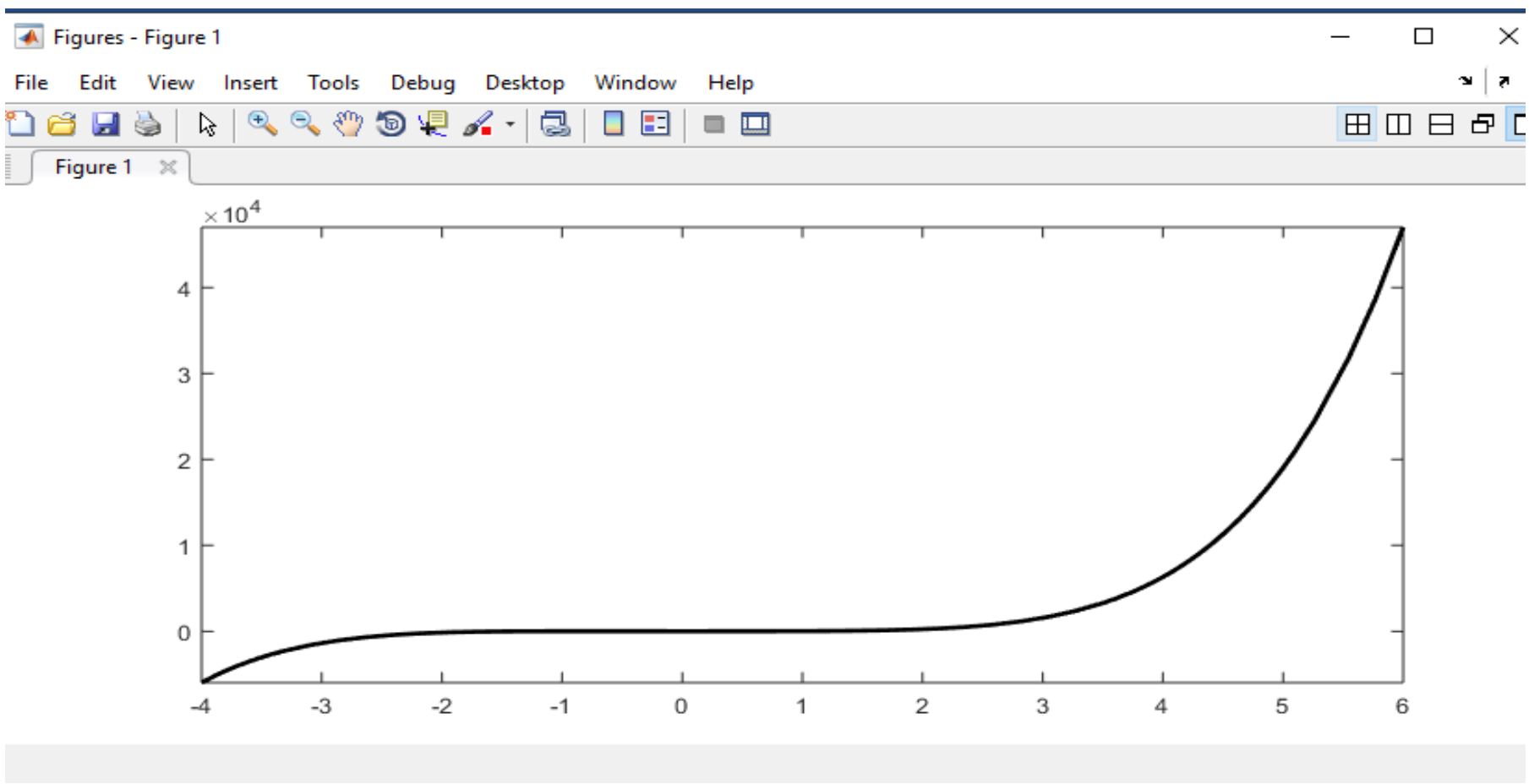


3) Use the fplot command to plot the function: $f(x) = 6x^5 + 10x^2 + 9$ In the domain -4 ≤ x ≤ 6

Code:

```
Command Window
>> %2018-EE-394
>> %lab 2 task 8c
>> fplot(@(x)(6*x.^5+10*x.^2+9),[-4,6])
fx >>
```

Output:



Assignment 9:

Plot the following data in MATLAB

Year	2000	2001	2002	2003	2004	2005
Minimum Marks	23	25	20	30	28	35
Maximum Marks	95	92	100	85	90	93

a) Label x Axis as year and y axis as marks

b) The color of the Minimum marks graph should be green and maximum marks graph should be red.

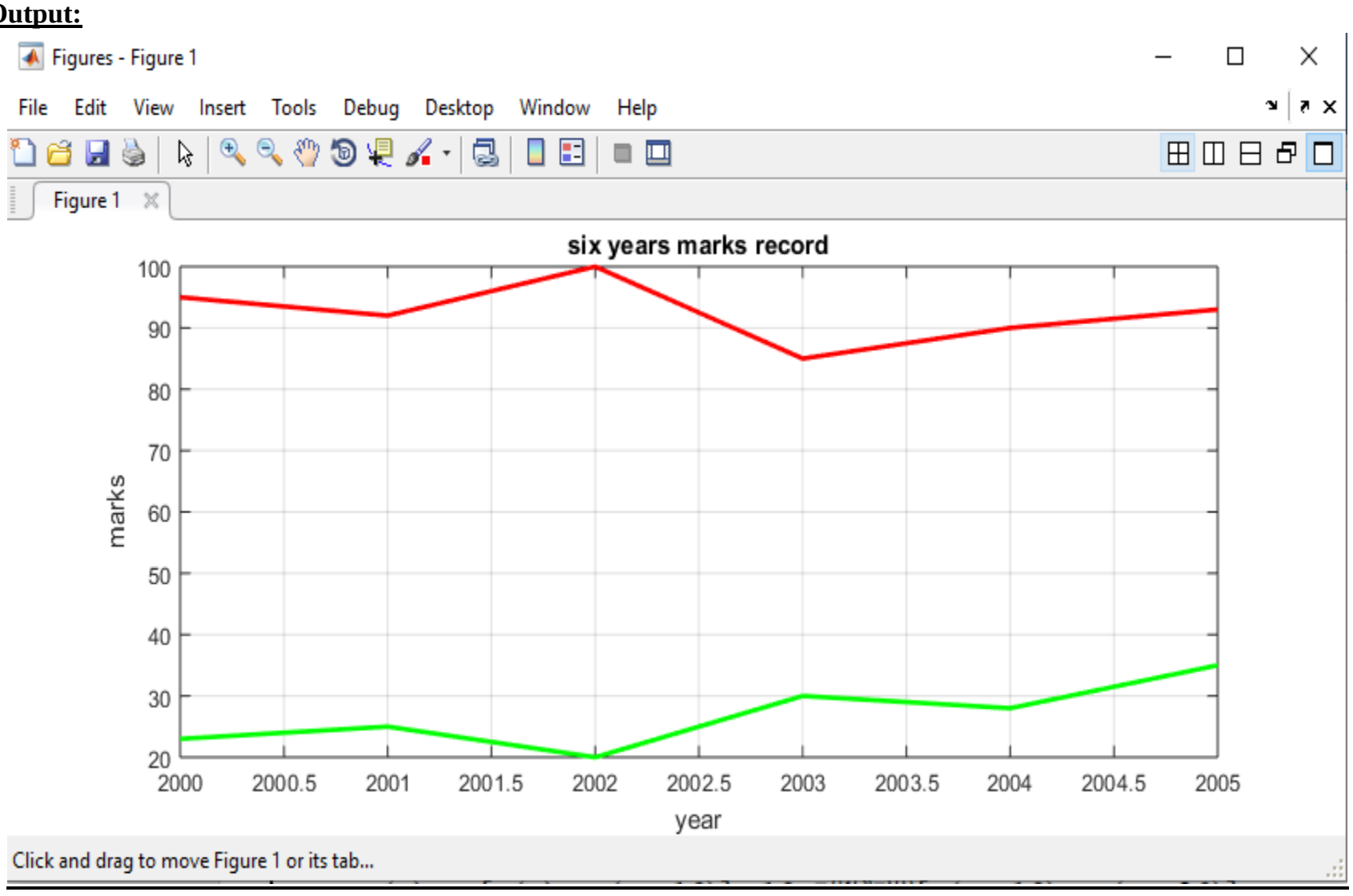
c) The range of x Axis should be 2000 to 2005 and y axis 20 to 100

d) The title of the graph should be “six years marks record”.

Code:

```
%2018-EE-394
%lab 2task 9
x=linspace(2000,2005,6);
minmarks=[23    25  20  30  28  35];
maxmarks=[95    92  100 85  90  93];
plot(x,minmarks,'green')
hold on
plot(x,maxmarks,'r')
axis([2000,2005,20,100])
xlabel('year')
ylabel('marks')
title('six years marks record')
grid on
hold off
```

Output:



Assignment 10:

Generate and plot the following sequence

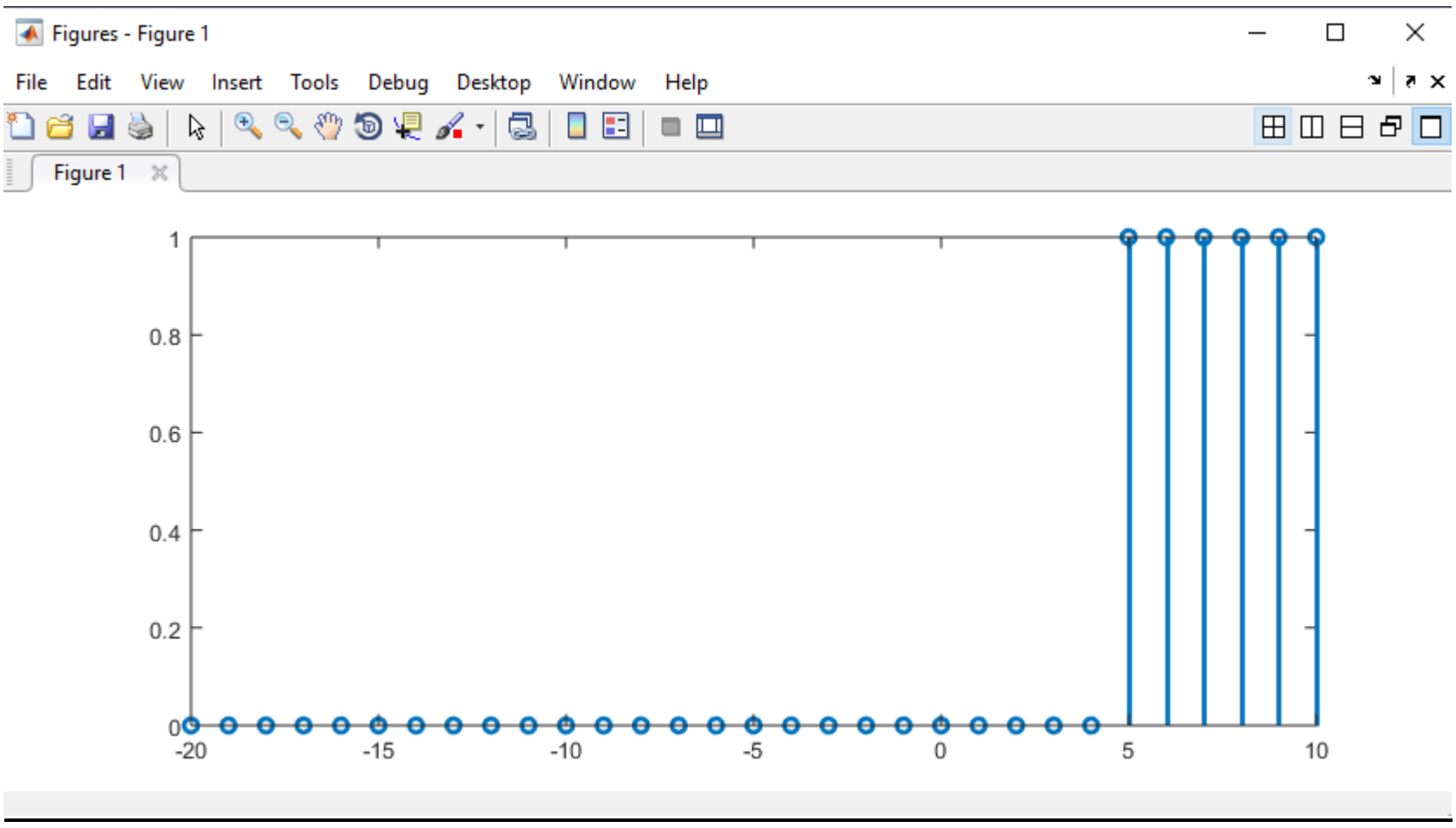
$u(n-5) \qquad -20 \le n \le 10$

Code:

```
%2018-EE-394
%lab 2 task 10a
n=[-20:10];
no=5;
x=([n-no]>=0)
stem(n,x)
```

```
Command Window
>> lab2_101

x =

  1×31 logical array

  Columns 1 through 24

   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0

  Columns 25 through 31

   0   1   1   1   1   1   1

fx >>
```

Output:



$$x(n) = (-10)^n \quad -10 \le n \le 10$$

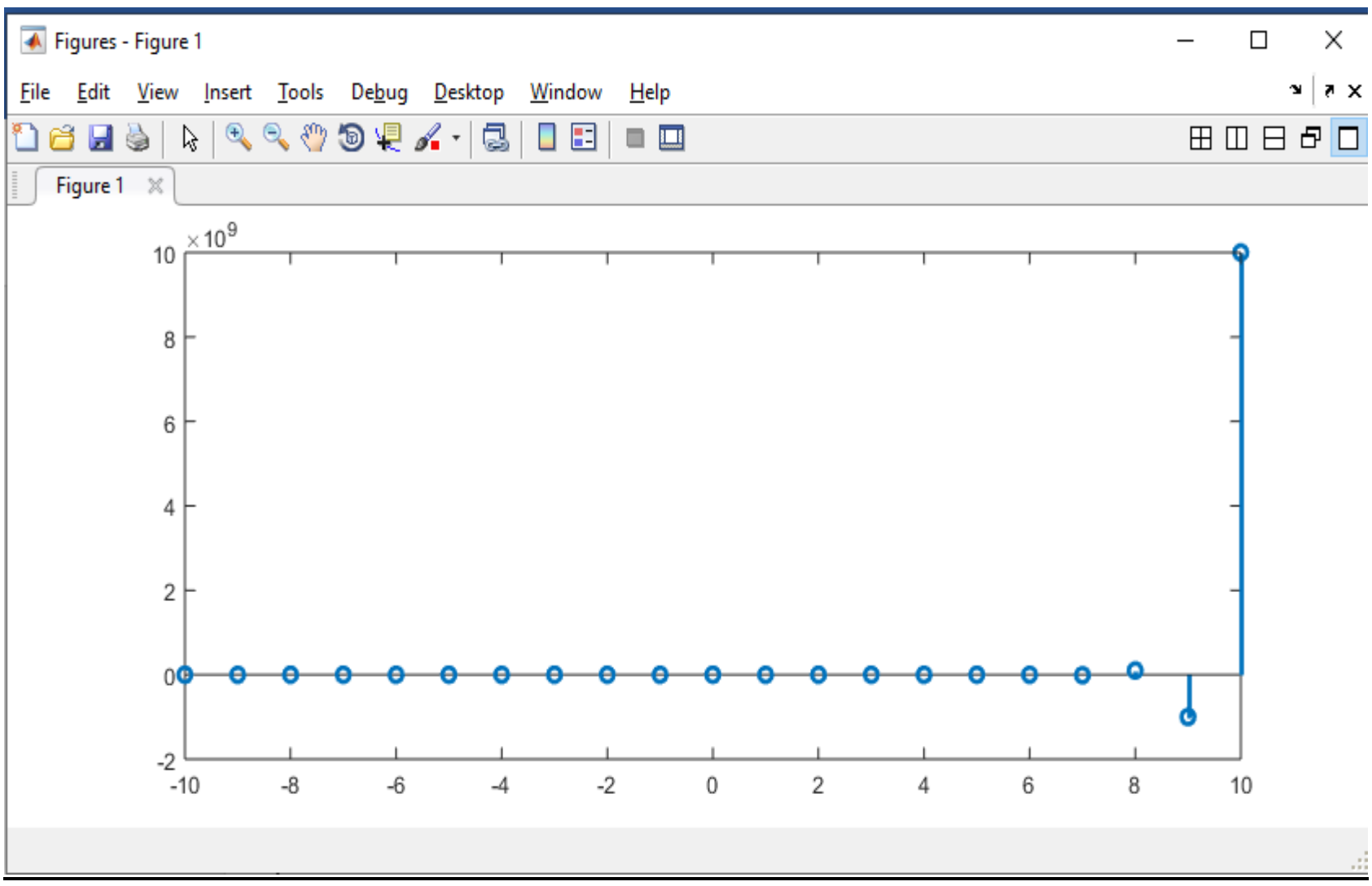
Code:

```
%2018-EE-394
%lab 2 task 10b
n=[-10:10];
x=(-10).^n
stem(n,x)
```

Command Window

```
Columns 10 through 18

   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0001   -0.0010

Columns 19 through 21

    0.0100   -0.1000    1.0000

fx >>
```

Output:



$x(n) = \exp[(2 + j3)\, n]$, $0 \le n \le 10$

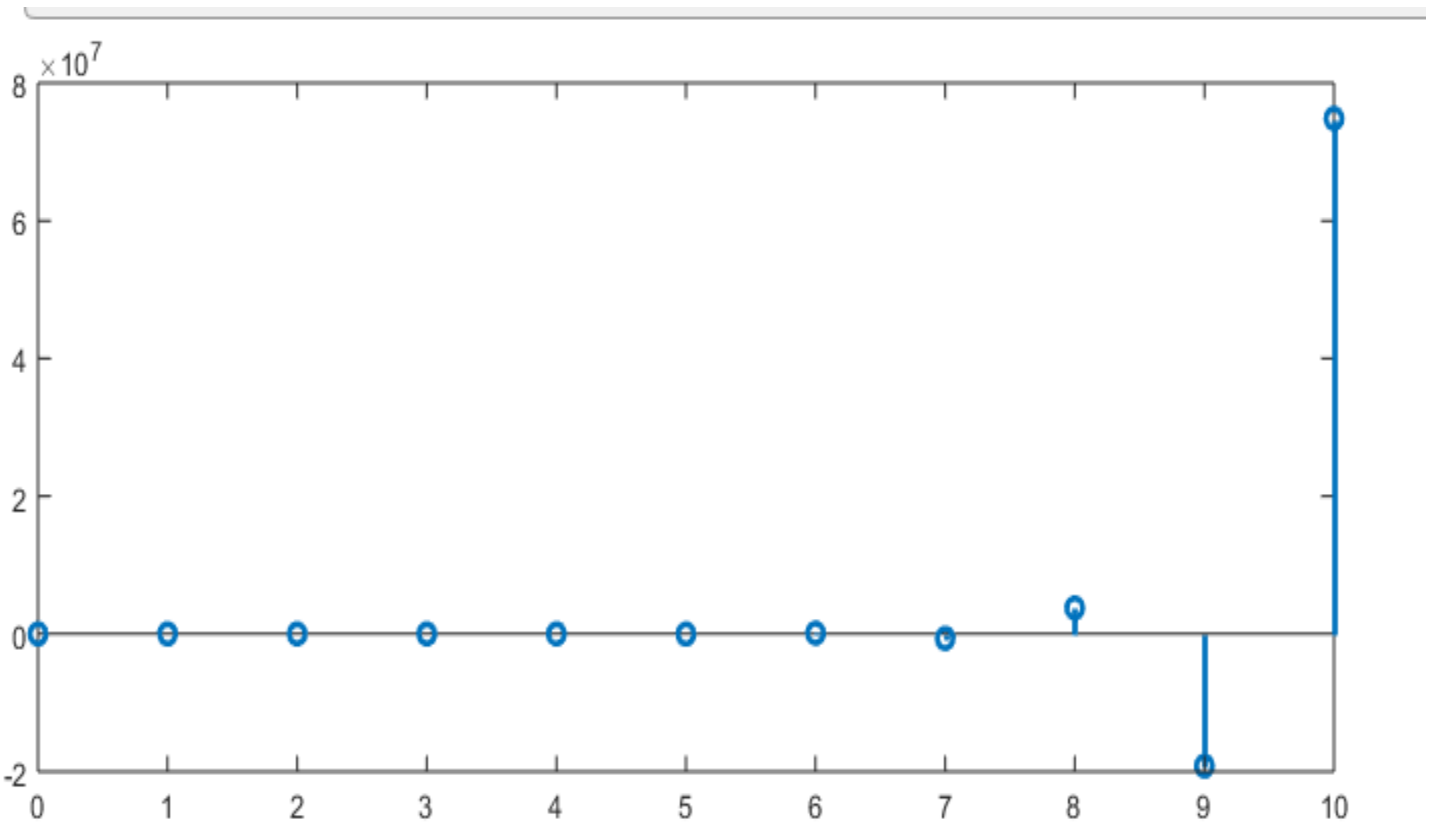
Code:

```
%2018-EE-394
%lab 2 task 10b
n=[0:10];
x=exp((2+j*3)*n)
stem(n,x)
```

Command Window

```
x =

   1.0e+08 *

Columns 1 through 5

   0.0000 + 0.0000i  -0.0000 + 0.0000i   0.0000 - 0.0000i  -0.0000 + 0.0000i   0.0000 - 0.0000i

Columns 6 through 10

  -0.0002 + 0.0001i   0.0011 - 0.0012i  -0.0066 + 0.0101i   0.0377 - 0.0805i  -0.1918 + 0.6280i

Column 11

   0.7484 - 4.7936i
```

Output:

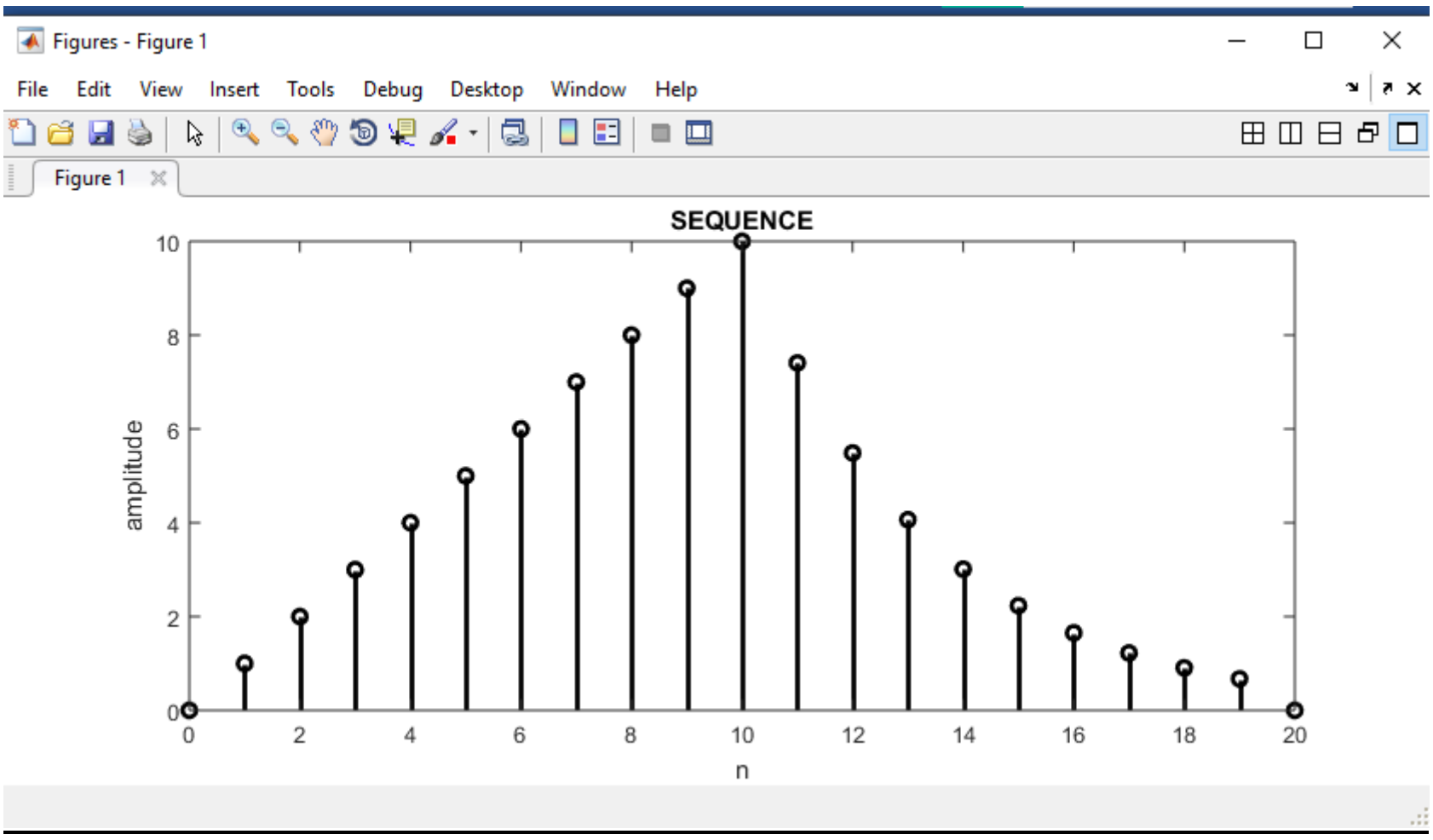


$$x(n) = n[u(n) - u(n-10)] + 10e^{-.03(n-10)}[u(n-10) - u(n-20)] \quad 0 \le n \le 20$$

Code:

```
%2018-EE-394
%lab 2 task 10d
n=[0:20];
x1=n.*(([n]>=0)-([n-10]>=0));
x2=10*exp(-0.3*(n-10)).*(([n-10]>=0)-([n-20]>=0));
x=x1+x2;
stem(n,x)
xlabel('n');
ylabel('amplitude')
title('SEQUENCE')
```

Output:



Conclusion:

In this lab we have learnt about the Array, Matrix and Plotting graphs of different functions .We have observed the row and column vectors. This lab also helps us to understand the different commands like **linspace** command, different representation of row and column vectors. We have use the **stem** command to display the graphs of discrete signals and plot command for continuous signals. Although this lab was all about basics commands but it was very important to understand .

Registration#____2018-EE-394_____

University of Engineering & Technology Lahore, FSD Campus

Experiment # 3

Title: **Introduction to Sampling**

Equipment Required: Personal computer (PC) with windows operating system and MATLAB software

Theory:

The course discusses discrete-time signal processing (DSP) in general that can be applied to any discrete-time sequences. One of the applications of DSP is in speech signal processing in which we process speech signals. In this lab, we will learn how to convert speech signal into discrete-time sequences using analog to-digital converter (ADC) available in the sound cards of every computer.

A sound card can be regarded as a combination of an analog-to-digital converter (ADC) and a digital-to-analog converter (DAC). Any sound card has at least the following inputs:

- Line-In
- Mic Input
- Speaker Output

Your sound card's ADC samples the signal coming from Line-In and Mic Inputs, whereas the DAC converts samples to an analog signal and sends it to playback devices such as your laptop's speaker. In this lab, we will learn how to sample a speech signal using MATLAB.

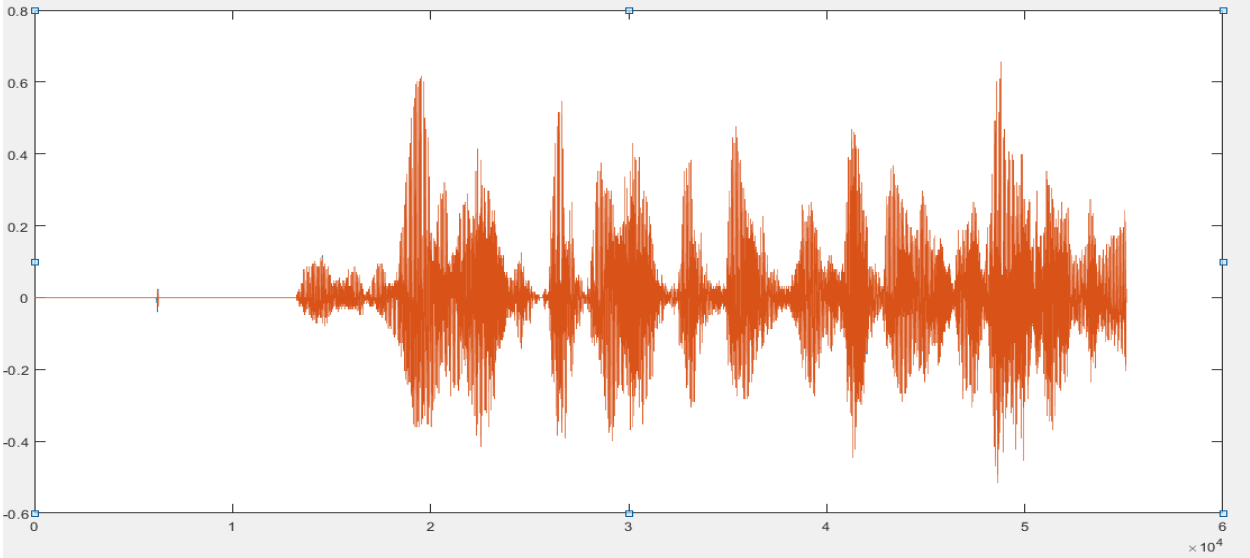
Task 1

Read the following webpage and write a MATLAB script to sample your own voice at 11025 samples per second and 8-bit width samples.

Code:

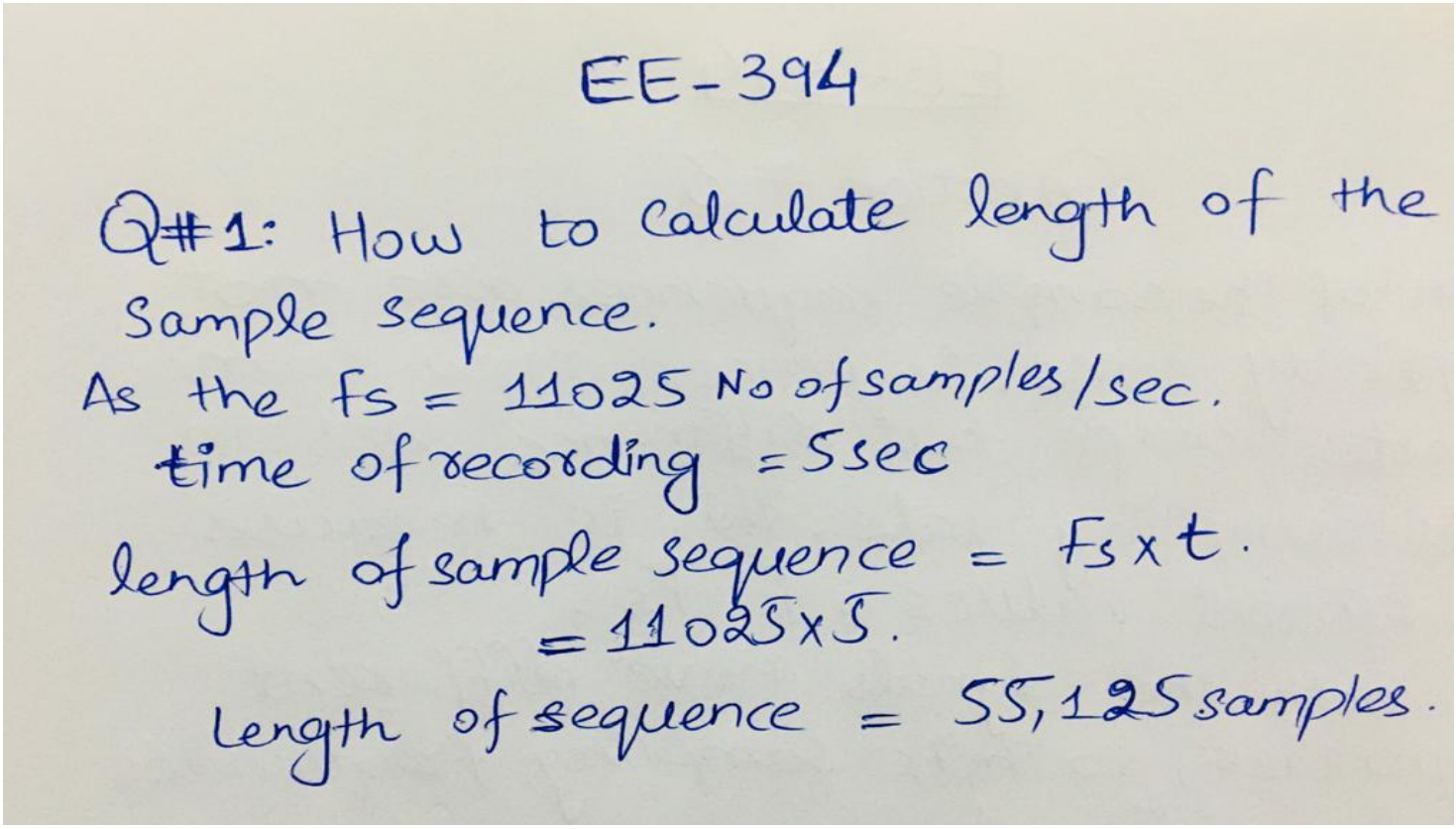
```
clear all;
recObj = audiorecorder;
Fs = 8000 ;
nBits = 8 ;
nChannels = 2 ;
ID = -1;
recObj = audiorecorder(Fs,nBits,nChannels,ID);
disp('Start speaking.')
recordblocking(recObj,5);
disp('End of Recording.');
play(recObj)
Audionote=getaudiodata(recObj);
plot(Audionote)
```

Output :



Questions

1. What is the length of the sampled sequence? How do you calculate this length on paper



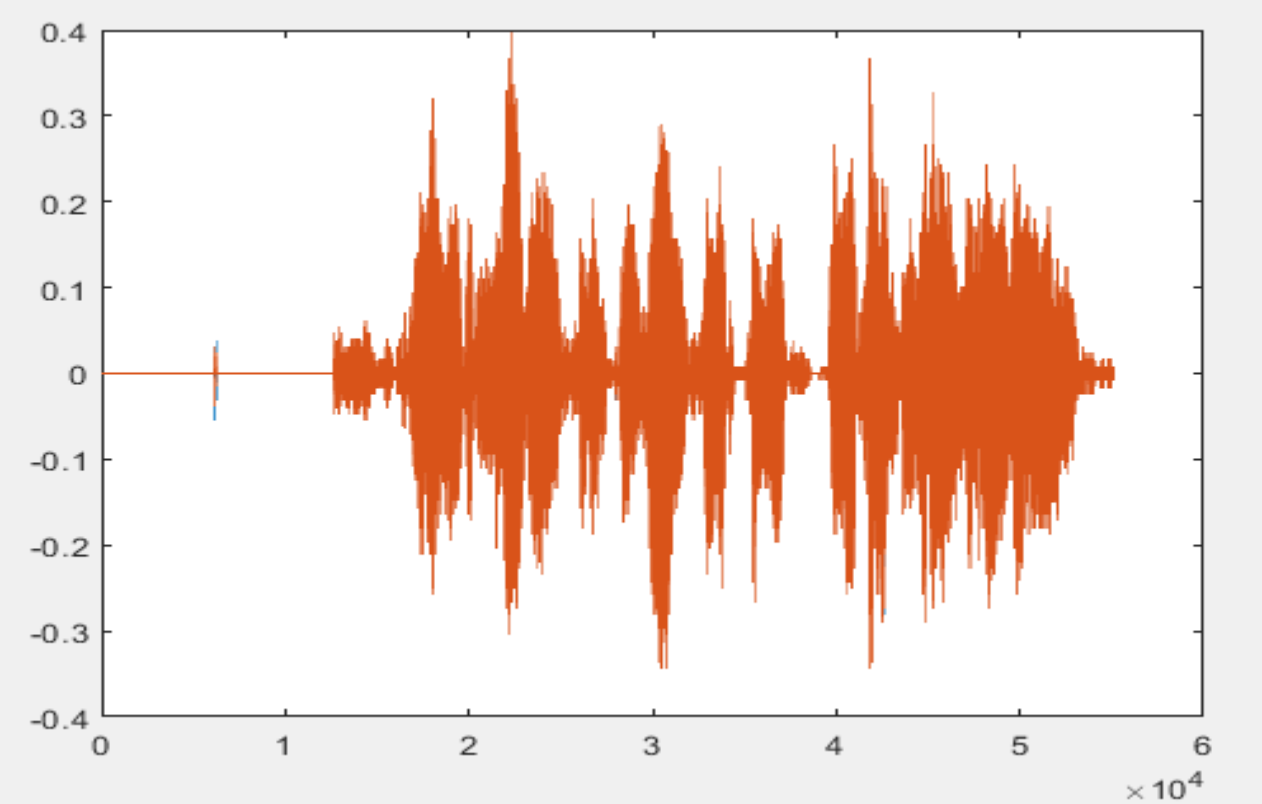
Task 2

Play the recorded voice on your computer's speakers using sound command in MATLAB at a DAC playback rate of 8000 samples per second with 8 bits per sample.

Code:

```
%2018-EE-394
%Task 2 lab 3
clear all;
recObj = audiorecorder;
Fs = 11025 ;
nBits = 8 ;
nChannels = 2 ;
ID = -1;
recObj = audiorecorder(Fs,nBits,nChannels,ID);
disp('Start speaking.')
recordblocking(recObj,5);
disp('End of Recording.');
Y=getaudiodata(recObj);
Fs=8000;
nBits = 8;
sound(Y,Fs,nBits);
plot(Y)
```

Output:



Questions

1. Do you hear the same voice as yours? If not, why the voice played back on the speaker is different?

EE-394

Question #2.

No, I didn't hear the same voice as it was recorded. As the recorded voice contains distortion and low pitch. This is because while recording the voice, distortion of noise is also there. while recording the sampling rate / frequency was 11025 and during listening the frequency was 8000 difference can be seen completely in both waveforms - Delay in sound was also noticed.

Task 3

In MATLAB, we can sample a few continuous-time signals without any hardware. For example, we can get samples from a continuous-time sinusoid.

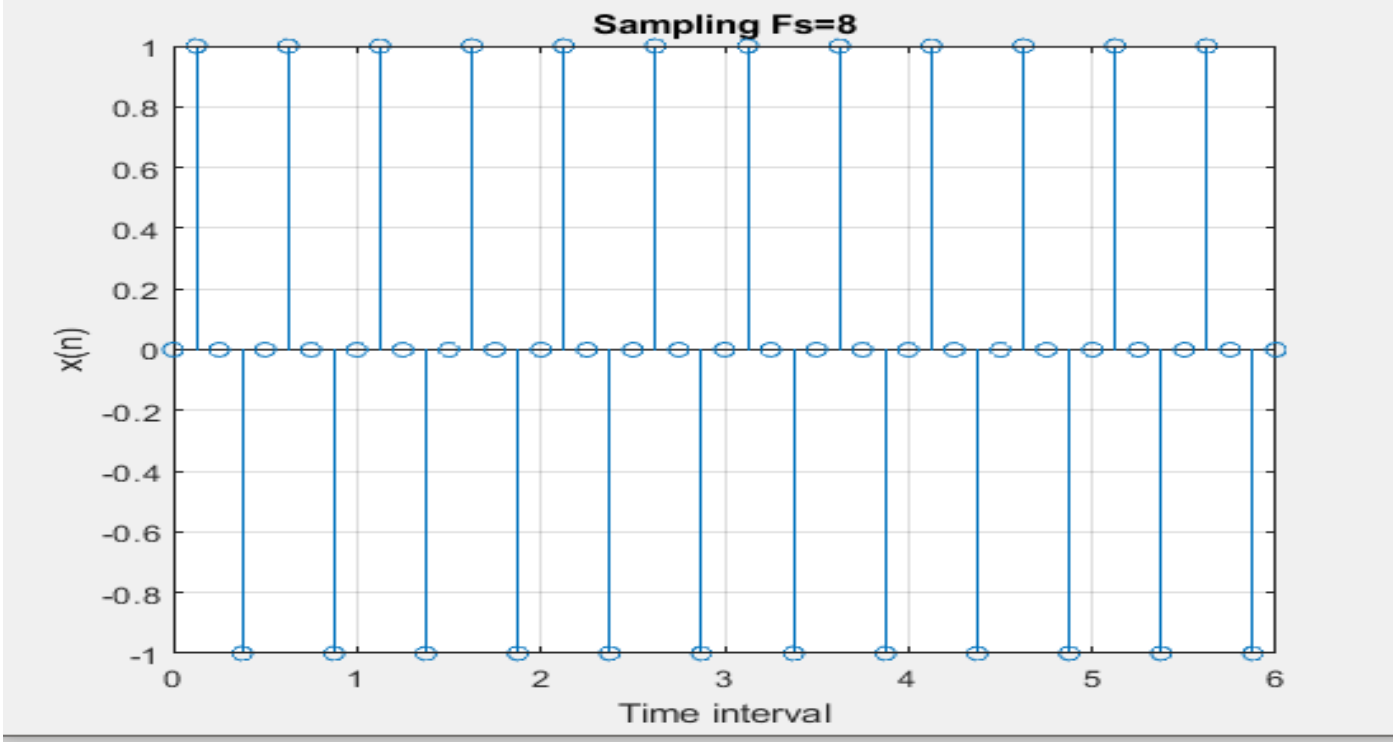
1. Sample a sinusoid of frequency 2 Hz at a sample rate of 8 samples per second in time from t = 0 seconds to t = 6 seconds. Plot the sampled values using stem.

2. Sample a sinusoid of frequency 6 Hz at a sample rate of 24 samples per second in time from t = 0 seconds to t = 2 seconds. Plot the sampled values using stem.

Code #1

```
%2018-EE-394
Fs=8;
t0=0:1/Fs:6;
x=cos(2*pi*2*t0);
stem(t0,x);
grid on
xlabel('Time interval')
ylabel('x(n)')
title('Sampling Fs=8');
```

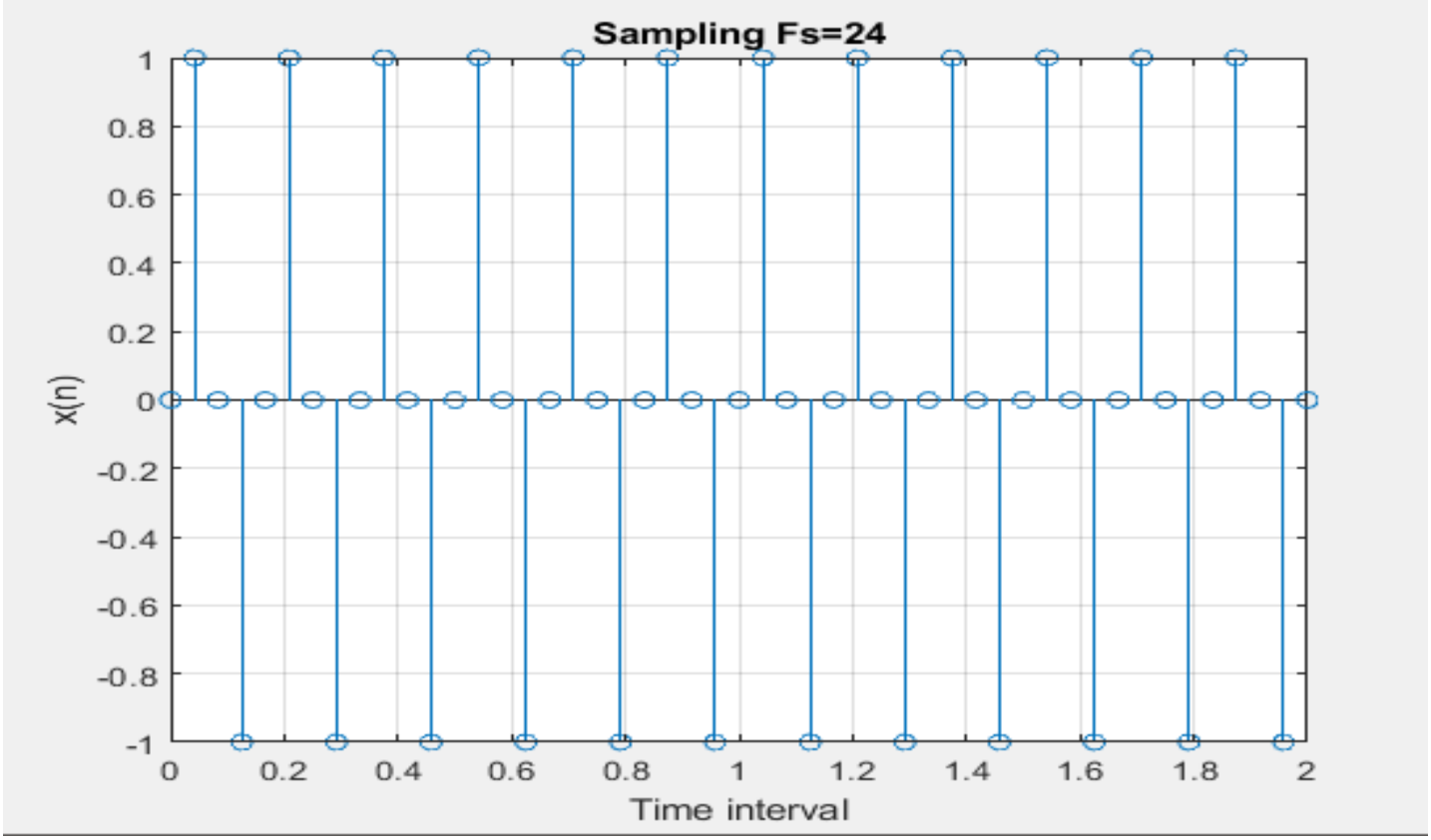
Output:



Code#2:

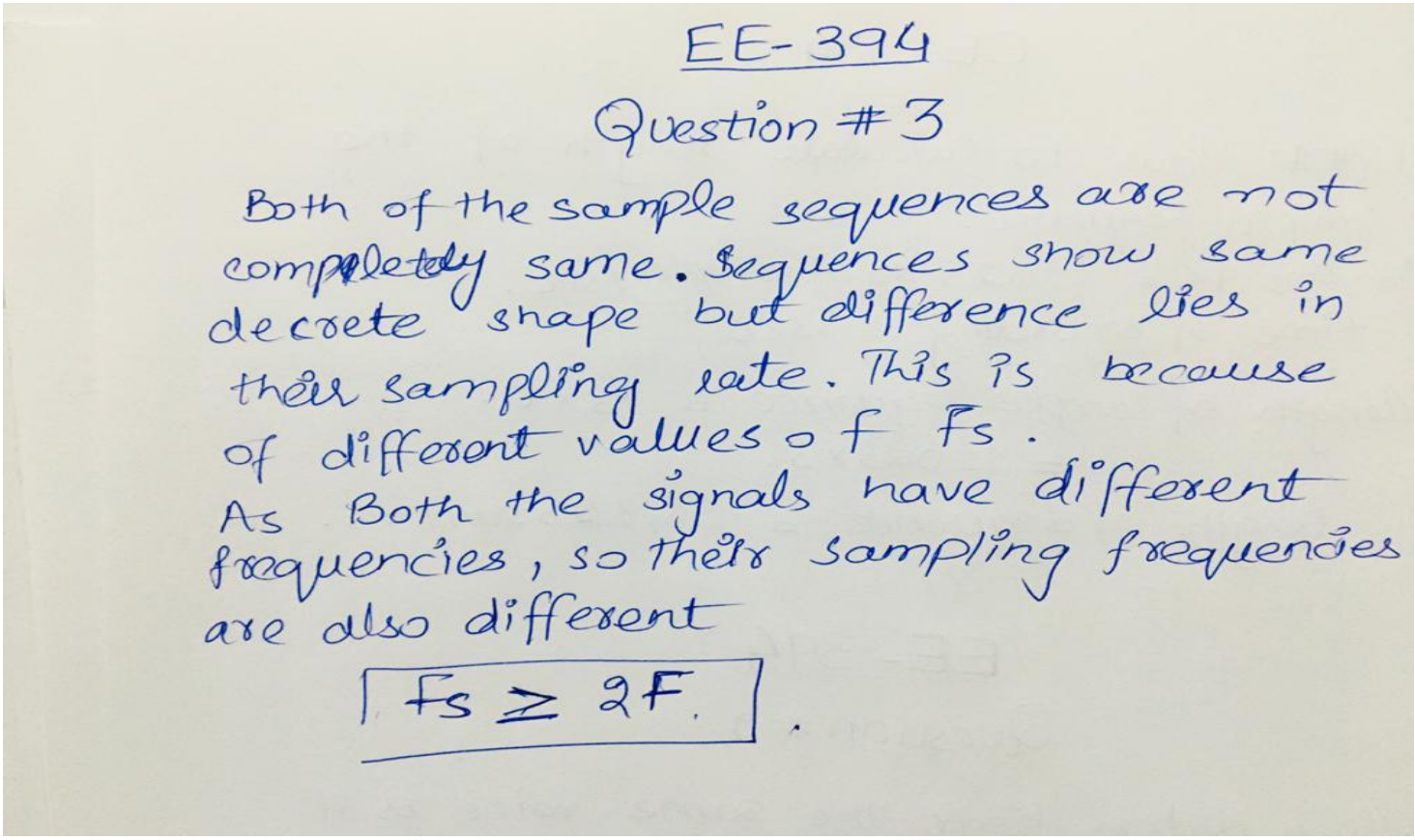
```
%2018-EE-394
Fs=24;
t0=0:1/Fs:2;
x=sin(2*pi*6*t0);
stem(t0,x);
grid on
xlabel('Time interval')
ylabel('x(n)')
title('Sampling Fs=24');
```

Output;



Questions

1. Are the two sampled sequences the same, different or partially similar. Why?



EE-394

Question # 3

Both of the sample sequences are not completely same. Sequences show same decrete shape but difference lies in their sampling rate. This is because of different values of Fs.

As Both the signals have different frequencies, so their sampling frequencies are also different

$$Fs \geq 2F.$$

Registration#_2018-EE-394_____

University of Engineering & Technology Lahore, FSD Campus

Experiment # 4

Title: **Linear Time-Invariant Systems**

Equipment Required: Personal computer (PC) with windows operating system and MATLAB software

Task 1

Let $y[n] - 0.1y[n-1] = x[n]$

with $\boldsymbol{y[-1] = c}$ be the difference equation describing a system.

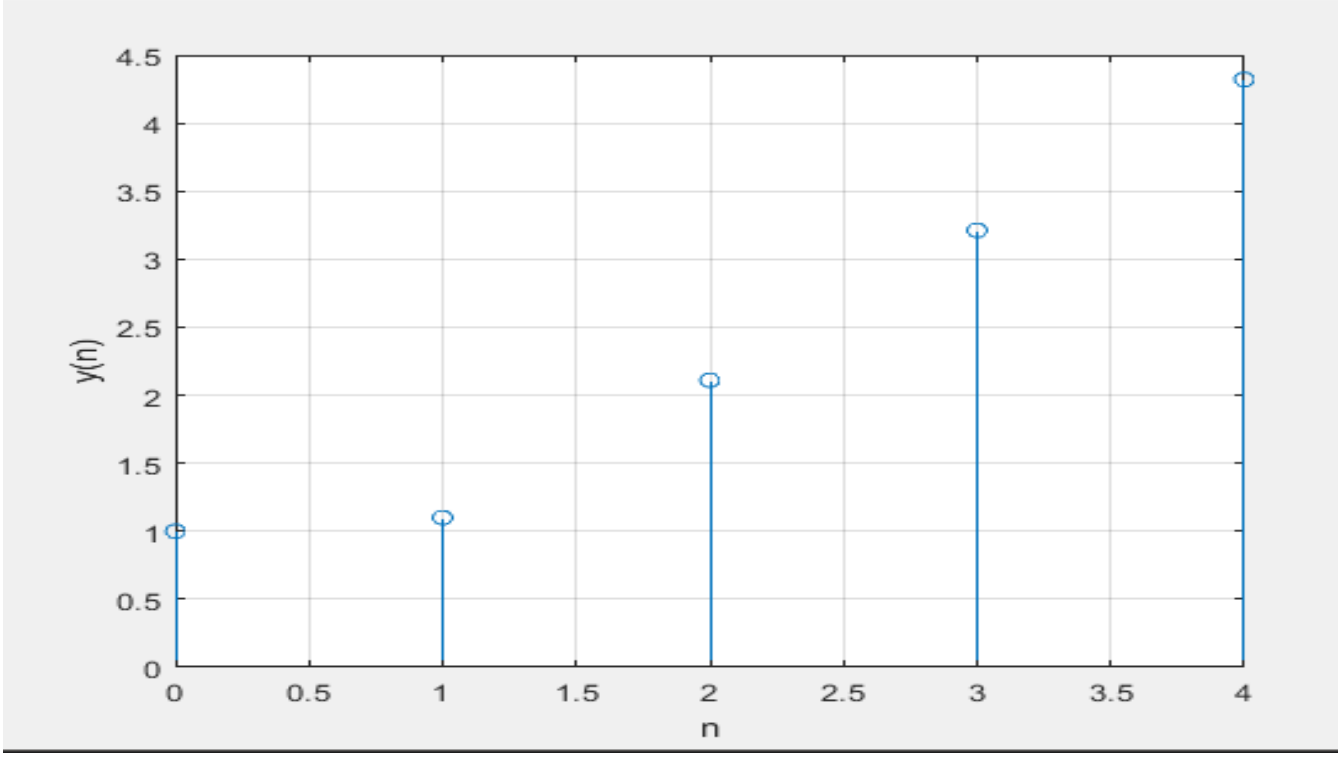
Write a MATLAB function y = *get_output_system*(x, c, n1, n2) to compute the output corresponding to input x[n].

MATLAB Code:

```
function y=get_output_system(x,c,n1,n2)
y(1)=c;
for n=n1:n2
    y(n)=x(n)+0.1.*y(n-1)
end
stem(x,y)
grid on
xlabel('n')
ylabel('y(n)')
end
```

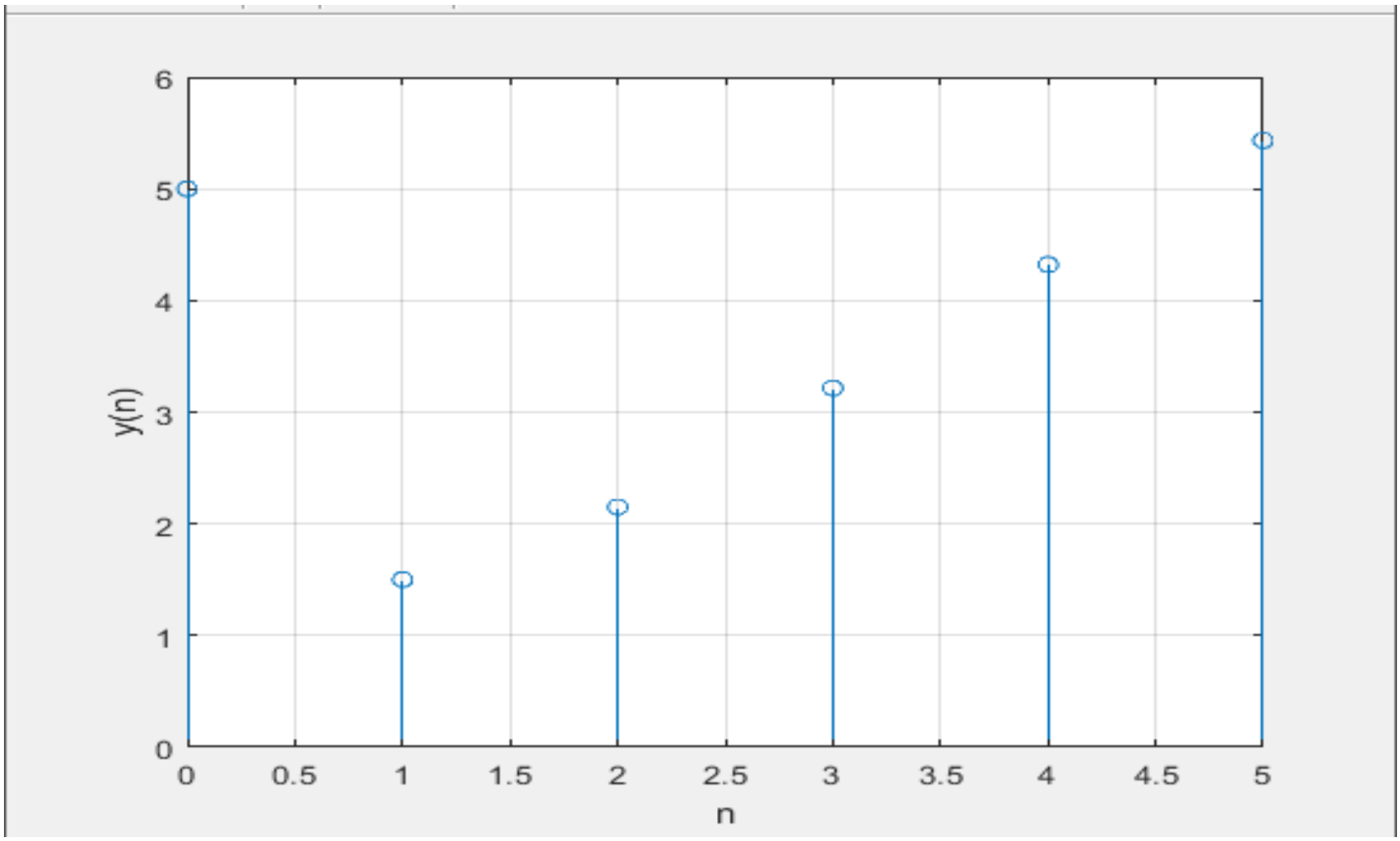
Output:

```
Command Window
>> get_output_system([0,1,2,3,4],1,2,5)
y =
    1.0000    1.1000
y =
    1.0000    1.1000    2.1100
y =
    1.0000    1.1000    2.1100    3.2110
y =
    1.0000    1.1000    2.1100    3.2110    4.3211
ans =
    1.0000    1.1000    2.1100    3.2110    4.3211
```



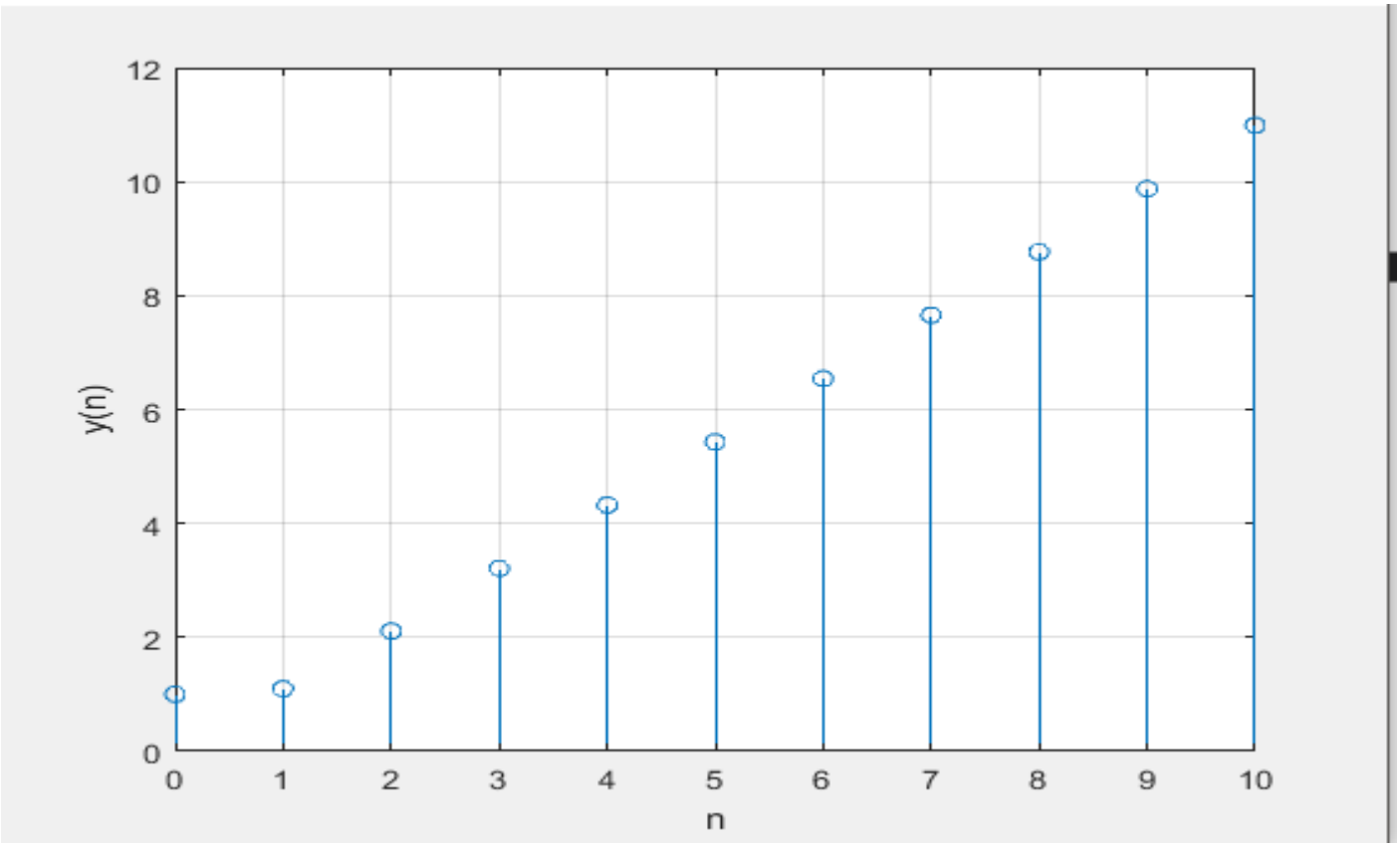
Linear system:

```
Command Window
>> get_output_system([0,1,2,3,4,5],5,2,6)
```



Causal system

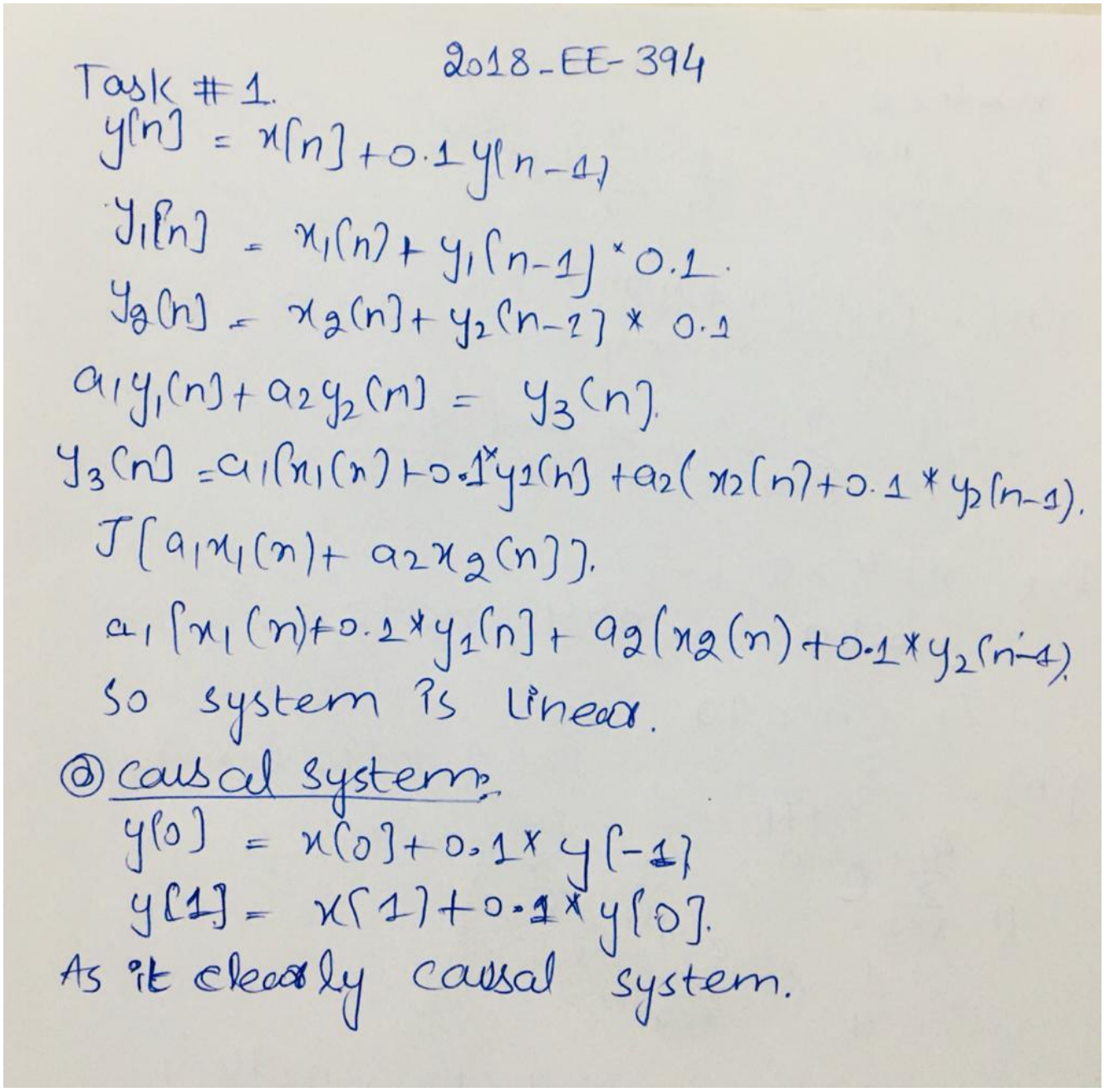
```
Command Window
>> get_output_system([0:10],1,2,11)

y =
```



Questions

1. Is this a linear system? Why or why not? Choose a suitable input and show the output using above MATLAB function to justify your answer.

2. Is this a causal system? Why or why not? Choose a suitable input and show the output using above MATLAB function to justify your answer.



Task 2

Consider the linear time-invariant system with impulse response

$$h[n] = \frac{C^n}{11}\,(u[n] - u[n-11])$$

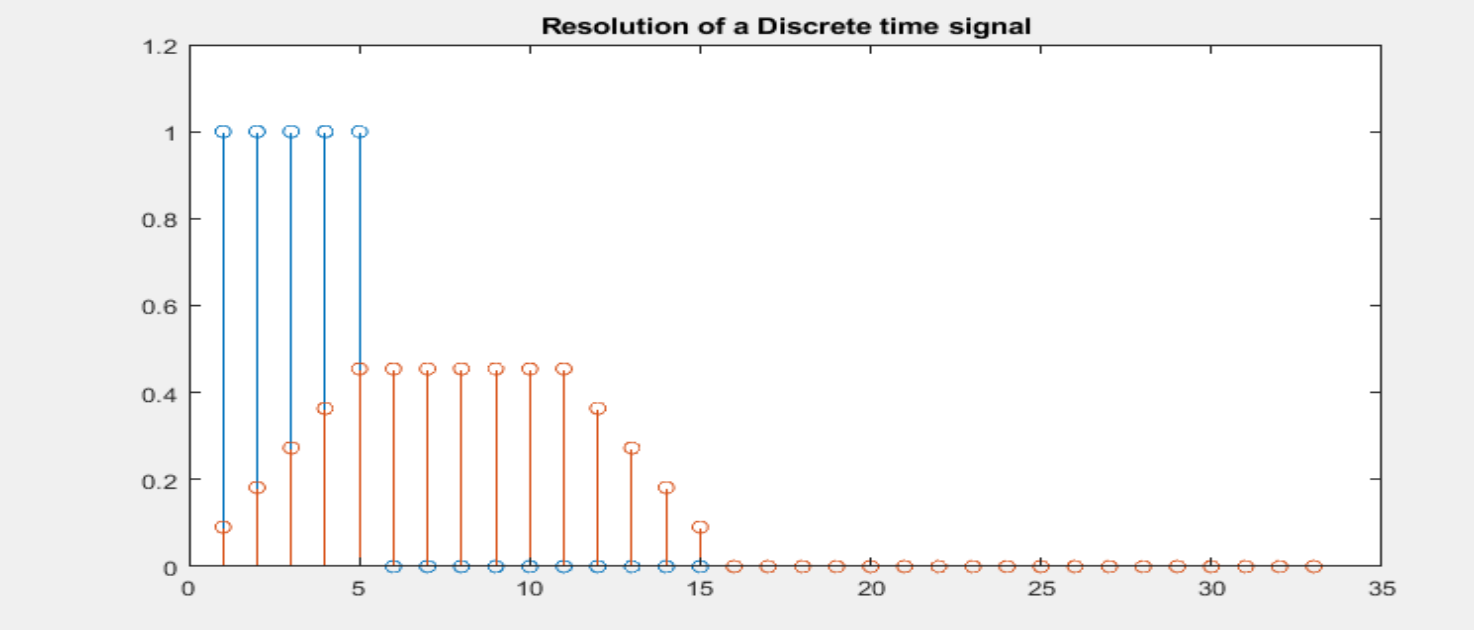
Find the output of the above-mentioned system to the input x[n] given below

$$\boldsymbol{x[n] = u[n] - u[n-5]}.$$

MATLAB Code(C=1);

```
%2018-EE-394
%lab#4 Task 2 C=1
C=1;
n=0:1:16;
U1=n>=0;
U2=n>=11;
U3=n>=5;
h=((C.^n)./11).*(U1-U2);
x=(U1-U3);
y=conv(x,h);
stem(x);
hold on;
stem(y);
hold off;
title('Resolution of a Discrete time signal');
```

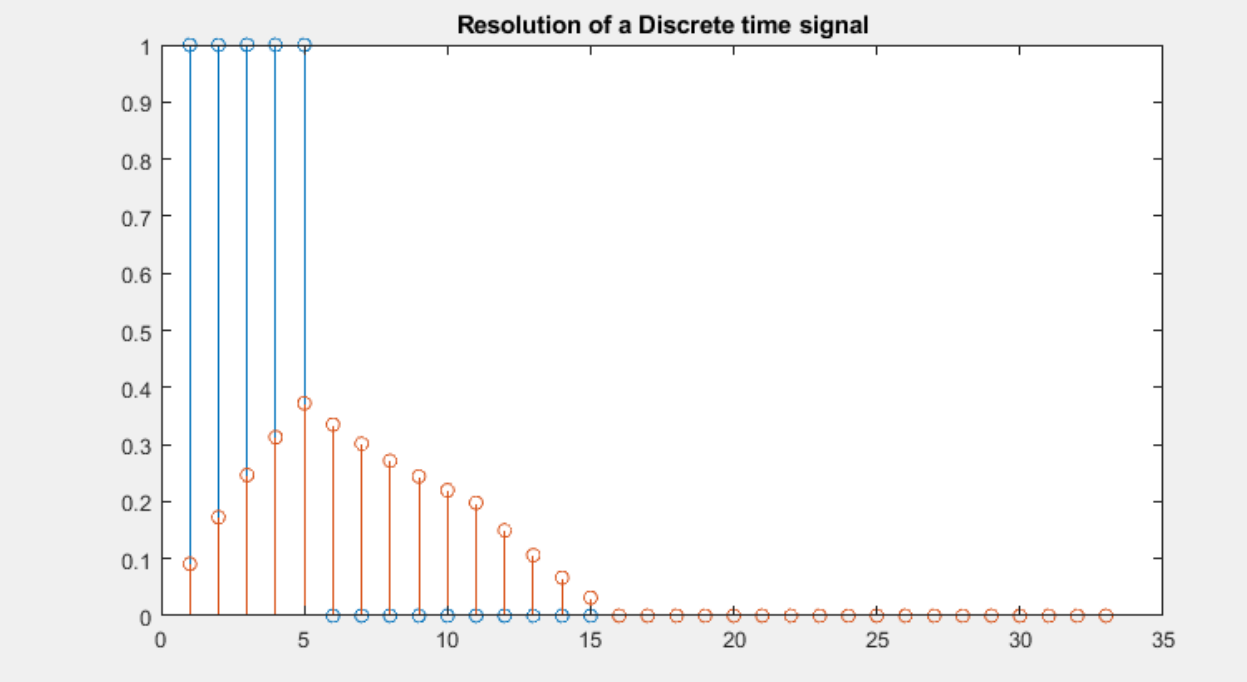
Output:



MATLAB Code(C=0.9):

```
%2018-EE-394
%lab#4 Task 2 C=0.9
C=0.9;
n=[0:1:16];
U1=n>=0;
U2=n>=11;
c=n>=5;
h=((C.^n)./11).*(U1-U2);
x=(U1-c);
y=conv(x,h);
stem(x);
hold on;
stem(y);
hold off;
title('Resolution of a Discrete time signal');
```

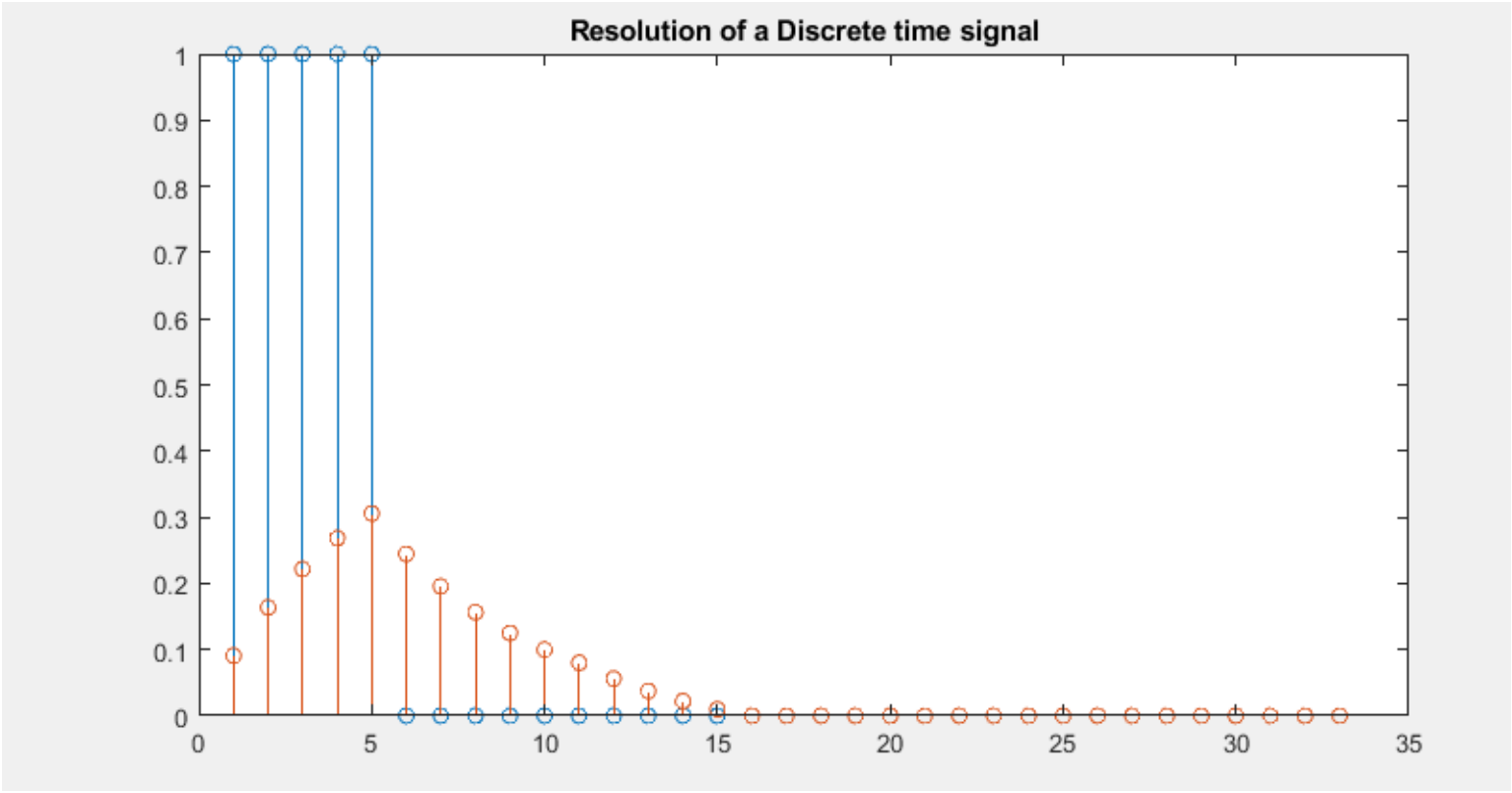
Output:



MATLAB Code(C=0.8):

```
%2018-EE-394
%lab#4 Task 2 C=1
C=0.8;
n=[0:1:16];
U1=n>=0;
U2=n>=11;
U3=n>=5;
h=((C.^n)./11).*(U1-U2);
x=(U1-U3);
y=conv(x,h);
stem(x);
hold on;
stem(y);
hold off;
title('Resolution of a Discrete time signal');
```

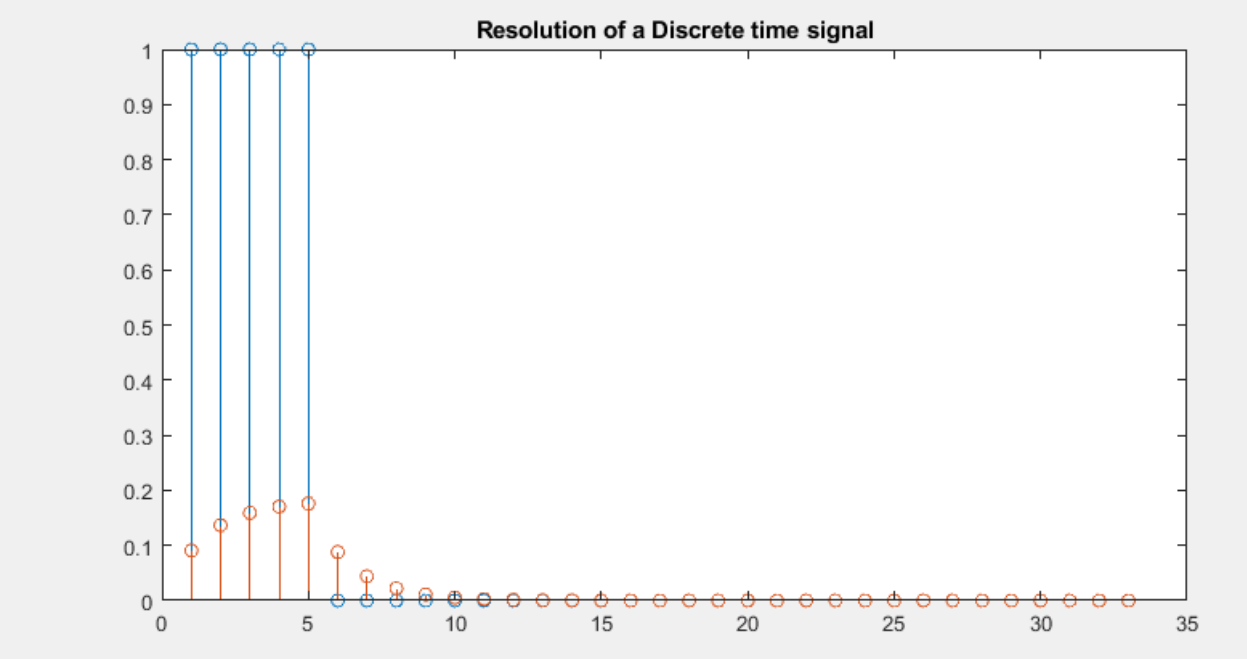
Output:



MATLAB Code(C=0.5):

```
%2018-EE-394
%lab#4 Task 2 C=1
C=0.5;
n=[0:1:16];
U1=n>=0;
U2=n>=11;
U3=n>=5;
h=((C.^n)./11).*(U1-U2);
x=(U1-U3);
y=conv(x,h);
stem(x);
hold on;
stem(y);
hold off;
title('Resolution of a Discrete time signal');
```

Output:



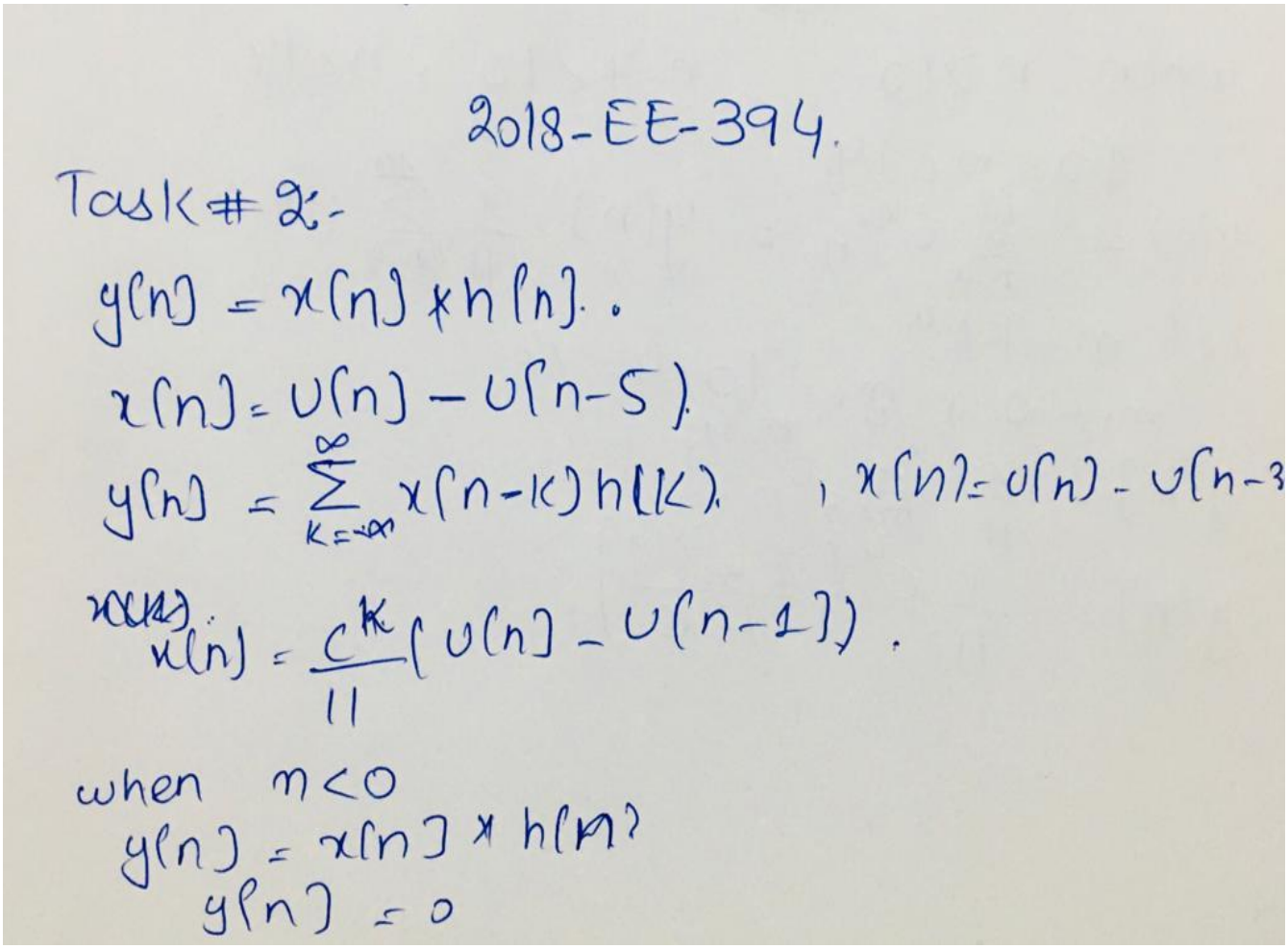
Questions

1. What are the parameters for the MATLAB function that you used?

2. Plot input and output of the system overlaid on each other in the same graph when c = 1. What is the name of this system with c = 1.

3. Repeat part 1 with c = 0.9, c = 0.8, c = 0.5. What is the difference between different outputs? What can you say about the effect of the constant c on the output.

4. Find out the answers of the questions above on paper.



when $n > 0$

$n - 4 < 0$, $\quad 0 < n < 4.$

$$\sum_{0}^{\infty} 1 \cdot \frac{(1)^{n-k}}{11} - \frac{1}{11} \sum_{0}^{4} (1)^{n-k}.$$

$$y[n] = \frac{(1)^k}{11} \left(\frac{1-(1)^{n+1}}{1-1} \right) \quad 0 < n < 4.$$

$$= \frac{1}{11} \sum_{k=0}^{4} c^{k} = \frac{1}{11} \left(\frac{1 - c^{k+1}}{1-c} \right)$$

for $n - 4 > 0$

$n < 10$, $n > 4$, $n < 10$.

$4 < n < 10$

$$y[n] = \sum_{k=n-4}^{4} \frac{c^k}{11} \Rightarrow \text{let } l = k+4 \Rightarrow l = n-4+4 = 0$$

$$= \frac{1}{11} \sum_{l=0}^{4+4} c^{l-n} = \frac{1}{11} \sum_{l=0}^{n+4} c^{m+4}.$$

$$= \frac{1}{11} c^{-4} \sum_{l=0}^{n+14} c^{l}.$$

when $n > 10$ $\quad n - 4 < 10$, $n < 14$.

$10 < n < 14.$

$$y(n) = \sum_{n-4}^{12} c^k / 11 = y[n] = \frac{1}{11} \sum_{k=n-4}^{10} c^{k}.$$

let $m = k + 4.$

$m_1 = 0$, $m_2 = 10 + 4 = 14.$

$$y[n] = \frac{1}{11} \sum_{m=0}^{14} c^{n-4}.$$

$$y[n] = \frac{1}{11} c^{4} \left(\frac{1 - c^{15}}{1 - c} \right).$$

Task 3

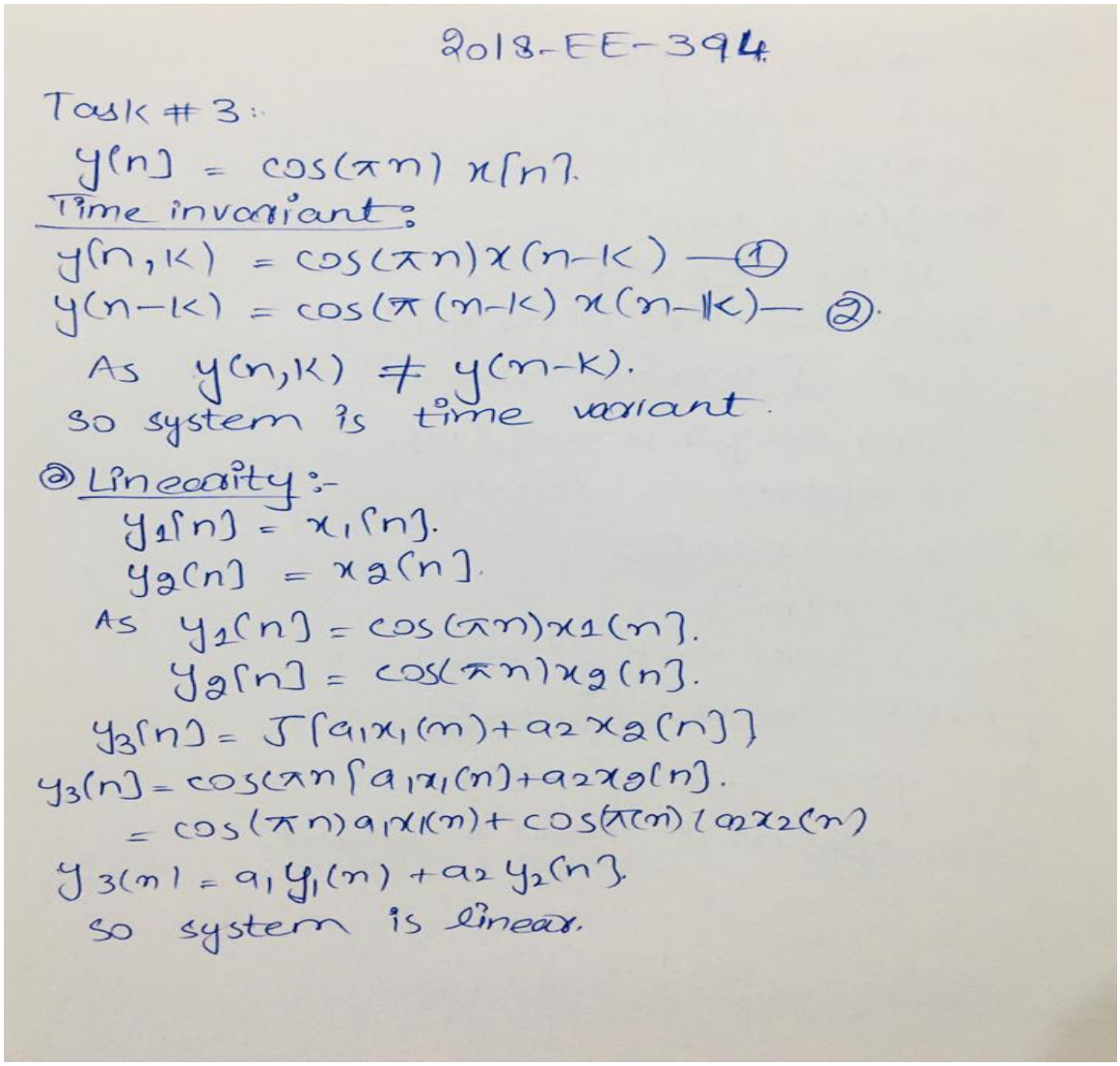
Consider the following systems:

1. $\boldsymbol{y[n] = cos(\pi n)x[n]}$.

2. $\boldsymbol{y[n] = x[n^2]}$.

Questions

1. Prove on paper whether these systems are linear and time invariant?



2018-EE-394

Task # 3:

$y[n] = \cos(\pi n)\, x[n]$.

Time invariant:

$y(n,k) = \cos(\pi n)\, x(n-k)$ — (1)

$y(n-k) = \cos(\pi(n-k))\, x(n-k)$ — (2).

As $y(n,k) \neq y(n-k)$.

So system is time variant.

(2) Linearity:-

$y_1[n] = x_1[n]$.

$y_2[n] = x_2[n]$.

As $y_1[n] = \cos(\pi n)\, x_1[n]$.

$y_2[n] = \cos(\pi n)\, x_2[n]$.

$y_3[n] = T[a_1 x_1(n) + a_2 x_2[n]]$

$y_3[n] = \cos(\pi n)[a_1 x_1(n) + a_2 x_2[n]]$.

$= \cos(\pi n)\, a_1 x_1(n) + \cos(\pi(n))\, a_2 x_2(n)$

$y_3(n) = a_1 y_1(n) + a_2 y_2[n]$.

So system is linear.

(b) :- $x_1[n^2] = y[n]$

$y[n] = x[n^2]$

Linearity :-

$y_1(n] = x_1[n^2]$.

$y_2[n] = x_2[n^2]$

As $y_3[n] = a_1 x_1[n^2] + a_2 x_2[n^2]$

$\therefore y_3[n] = \mathcal{T}[a_1 x_1(n) + a_2 x_2(n]$.

So $y_3[n] = a_1 y_1[n] + a_2 y_2[n]$.

So system is linear.

(c) Time invariant :-

$y(n] = x[n^2]$.

$y(n,k) = x[n-k]^2$ —(1)

$y(n-k) = x[n-k]^2$ —(2).

As $y(n,k) = y(n-k)$.

So system is time invariant.

2. Verify your answers using MATLAB.

MATLAB Code: 1. $y[n] = cos(\pi n)x[n]$.

a)linearity:

```
%2018-EE-394
%LAB#4 task#3
%linearity
n=5;
y=@(x) cos(pi*n)*x(n);
x1=1:6;
x2=11:16;
y(x1)+y(x2)
y(x1+x2)
```

Output:

```
Command Window
>> DSPtask3a_linearity

ans =

   -20


ans =

   -20
```

MATLAB Code:

a)Time invariant:

```
%2018-EE-394
%lab#4 Task 3
%time invariant
n=1;
k=4;
y=@(x) cos(pi*n)*x(n);
x1=2;
s1=y(x1)-k
s2=y(x1-k)
```

Output:

```
Command Window
>> DSPLAB4task3a_timeInvarience

s1 =

    -6


s2 =

     2
```

MATLAB Code: 2. $y[n] = x[n^2]$.

a)linearity:

```
%2018-EE-394
%lab#4 Task 3
%linearity
n=2;
a=1;
y=@(x) a.*x(n.^2);
x1=1:4;
x2=11:14;
y(x1)+y(x2)
y(x1+x2)
```

Output:

```
Command Window
>> DSPLAB4task3b_linearity

ans =

    18


ans =

    18
```

MATLAB Code: 2. $y[n] = x[n^2]$.

b)Time Invariant:

```
%2018-EE-394
%lab#4 Task 3
%Time Invariant
n=1;
k=4;
a=1;
y=@(x) a.*x(n.^2);
x1=1;
s1=y(x1)-k
s2=y(x1-k)
```

Output:

```
Command Window
>> DSPLAB4task3b_timeInvarience

s1 =

    -3


s2 =

    -3
```

Task 4

Let

$$y[n] - 0.25y[n-1] - \left(\frac{1}{8}\right)y[n-2] = 3x[n]$$

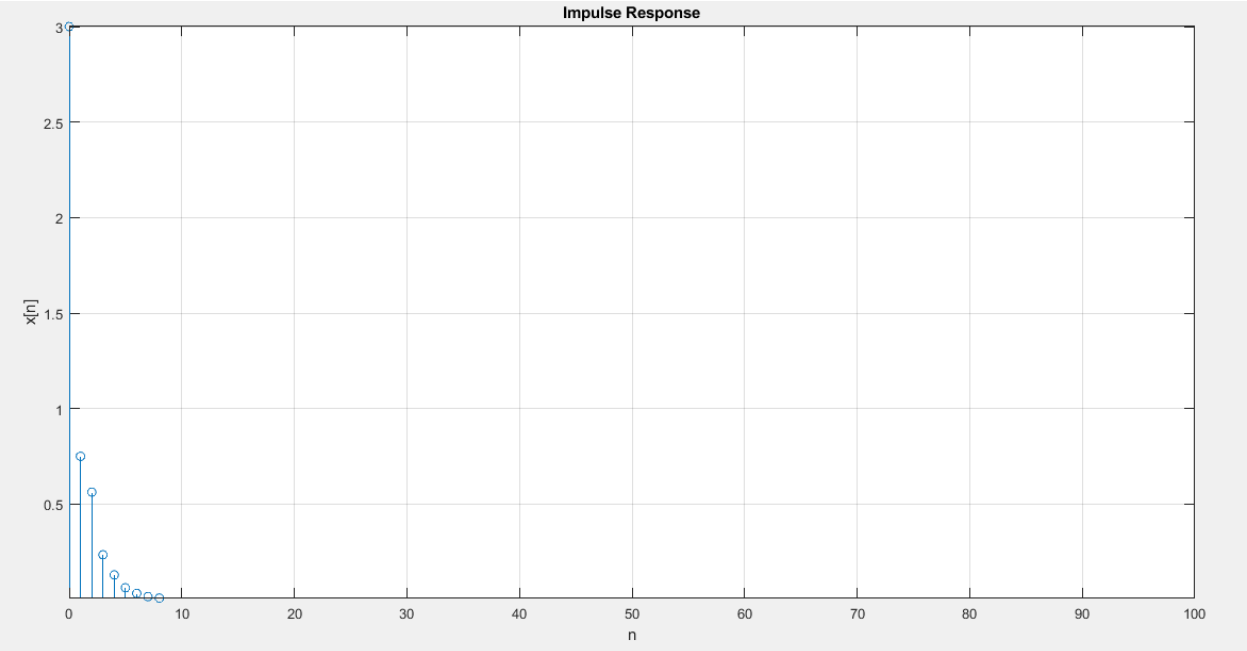
Calculate the impulse response at n= [0:100] (by using filter command)

Is this **a stable** system? Why or why not? (plot impulse response)

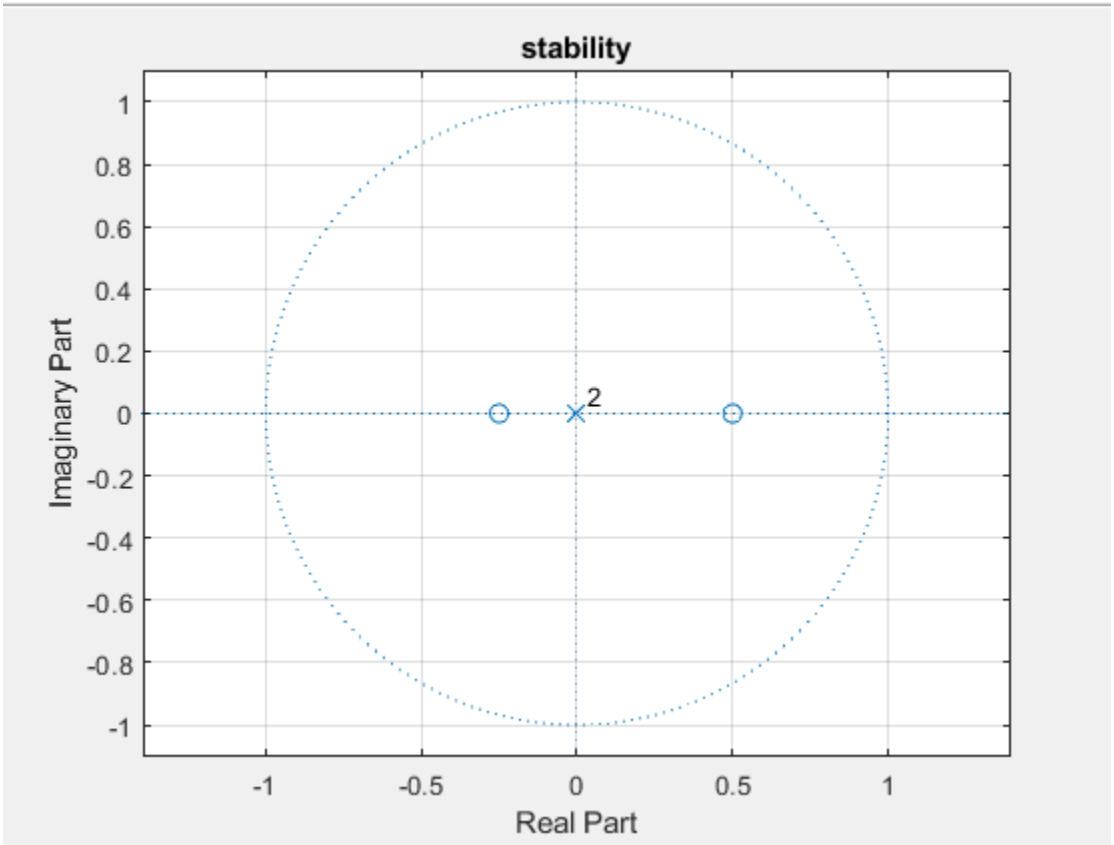
MATLAB Code:

```
%2018-EE-394
%LAB#4 task#4
a=[1 -0.25 -0.125]; %left side of equation (y)
b=[3]; %Right side of equation (x)
n=0:100;
x=(n==0);
y=filter(b,a,x);
stem(n,y)
xlabel('n');
ylabel('x[n]');
title('Impulse Response');
grid on
```

OUTPUT:



Stability:


stability
Imaginary Part
Real Part
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1
2

Name	Hamza Mobeen
Reg.NO	2018-EE-394

EXPERIMENT # 5

Z Transform

Task#01:

i) $x(n) = (1/4)^n\, u(-n)$.

MATLAB Code:

DSPLab5task1a.m

```
%2018-EE-394
%LAB#5 Task#1(a)
syms z n;
ztrans(1/4^-n,z^-1)
```

Output:

Command Window

```
>> DSPLab5task1a

ans =

1/(z*(1/z - 4))
```

Handwritten Solution:

2018-EE-394

Task#1:

① $x(n) = (1/4)^n U(-n)$.

$$X(z) = \sum_{-\infty}^{0} (1/4)^n z^{-n}$$

$$= \sum_{0}^{\infty} (1/4)^{-n} z^{n}$$

$$= \frac{1}{1-(\frac{1}{4})^{-1}z} \Rightarrow \frac{1}{1-4z}.$$

$$= \frac{1}{z(\frac{1}{z}-4)}.$$

$$\boldsymbol{ii})\ \boldsymbol{x}(n) = (0.8)^n\, u(-n-1).$$

MATLAB Code:

```
%2018-EE-394
%LAB#5 Task#1(b)
syms z n;
ztrans(0.8^-n,z^-1)*z^-1*4^1
```

Output:

Command Window

```
>> DSPLab5task1b

ans =

4/(z^2*(1/z - 5/4))
```

Handwritten Solution:

2018-EE-394

(ii) $x(n) = (0.8)^n u(-n-1)$.

$$X(z) = \sum_{-\infty}^{-1} (0.8)^n z^{-n}.$$

$$X(z) = \sum_{n=1}^{\infty} (0.8)^{-n} z^n$$

$$= \sum_{n=0}^{\infty} (0.8)^{-(n+1)} z^{n+1}.$$

$$= \sum_{0}^{\infty} (0.8)^{-n} \cdot (0.8)^{-1} \cdot z^n \cdot z^1.$$

$$= (0.8)^{-1} \cdot z \sum_{0}^{\infty} (0.8)^{-n} z^n.$$

$$= (0.8)^{-1} \cdot z \left[\frac{1}{1 - (0.8)^{-1} z} \right].$$

$$= (0.8)^{-1} z \times \sum_{0}^{\infty} (0.8)^{-n} z^n$$

$$= \left(\frac{4}{5}\right)^{-1} z \times \left[\frac{1}{1 - \left(\frac{4}{5}\right)^{-1} z^{-1}} \right].$$

$$= \frac{5}{4} z \left[\frac{1}{1 - \frac{5}{4} z} \right].$$

$$= \frac{5}{4} z \left[\frac{4}{4 - 5z} \right] = \frac{5z}{4 - 5z}.$$

$iii)\ x(n) = (4^n)\ u(1-n).$

MATLAB Code:

```
%2018-EE-394
%LAB#5 Task#1(c)
syms z n;
ztrans(4^-n,z^-1)*z^-1*4^1
```

Output:

```
Command Window
>> DSPLab5task1c

ans =

4/(z^2*(1/z - 1/4))
```

Handwritten Solution:

2018-EE-394

(iii): $x(n) = (4)^n u(1-n)$

$$X(z) = \sum_{-\infty}^{+1} (4)^n z^{-n}$$

$$X(z) = \sum_{-1}^{\infty} (4)^{-n} z^{-n} = X(z) = \sum_{0}^{\infty} (4)^{-(n-1)} z^{n-1}$$

$$X(z) = \sum_{n=0}^{\infty} (4)^{-n} \cdot 4^1 \cdot z^n \cdot z^{-1}$$

$$= \frac{4}{z} \sum_{0}^{\infty} (4)^{-n} z^n = \frac{4}{z} \left[\frac{1}{1 - \left(\frac{1}{4}\right) z} \right]$$

$$= \frac{4}{z} \left[\frac{4}{4-z} \right] = \frac{16}{z(4-z)}$$

$$= \frac{16}{4z - z^2}$$

$$iv)\ x(n) = (n+1)(3^{n-2})\,u(n).$$

MATLAB Code:

```
%2018-EE-394
%LAB#5 Task#1(e)
syms z n;
ztrans((n+1)*(3^(n-2)))
```

Output:

Command Window

```
>> DSPLAB5taskle

ans =

z/(9*(z - 3)) + z/(3*(z - 3)^2)
```

Handwritten Solution:

2018-EE-394

(iv) $x(n) = (n+1)(3^{n-2})\,u(n)$

$X(z) = [n(3^{n-2}) + (3^{n-2})]\,u(n)$.

$X(z) = [n(3^{n-2})u(n) + (3^{n-2})u(n)$.

As z-transform of $(3^{n-2})u(n)$ is

$$X(z) = \sum_{0}^{\infty} (3^{n-2})u(n) = 3^{-2}\sum_{0}^{\infty}(3^n)z^{-n}$$

$$= \frac{1}{9(1-3z^{-1})}$$

$$x(n) = \left(n(3^{n-2}) + 3^{n-2}\right)u(n)$$

As $-z\frac{dX(z)}{dz} = -z\frac{d}{dz}\left[\frac{1}{9(1-3z^{-1})}\right]$.

$$= \frac{z^{-1}}{3(1-3z^{-1})^2}$$

$$X(z) = \frac{z^{-1}}{3(1-3z^{-1})^2} + \frac{1}{9(1-3z^{-1})}$$

$$X(z) = \frac{z}{3(1-3z^{-1})^2} + \frac{z}{9(z-3)^2} = \frac{z}{3(z-3)^2} + \frac{z}{9(z-3)}$$

$$v)\ x(n) = nSin(\pi n/3)\, u(n) + (0.9)^n\, u(n-1)$$

MATLAB Code:

```
1    %2018-EE-394
2    %LAB#5 Task#1(f)
3 -  syms z n;
4 -  ztrans(n*sin(pi*n/3))+ztrans(0.9^n)*z^-1
5
6
```

Output:

```
Command Window
>> DSPLAB5task1f

ans =

1/(z - 9/10) + (3^(1/2)*z*(z^2 - 1))/(2*(z^2 - z + 1)^2)
```

HandwrittenSolution:

2018-EE-394

(V): $n\sin(n\pi/3)u(n) + (0.9)^n u(n-1)$.

$x(n) = \sin(\pi/3\, n)u(n)$.

$= \frac{1}{2j}\left[\left(e^{j\pi/3}\right)^n - \left(e^{-j\pi/3}\right)^n\right]u(n)$.

$X(z) = \frac{1}{2j}\sum_{0}^{\infty}\left[e^{j\pi/3\, n}z^{-1} + -e^{-j\pi/3\, n}z^{-1}\right]$.

$= \frac{1}{2j}\left[\frac{1}{1-e^{j\pi/3}z^{-1}} - \frac{1}{1-e^{-j\pi/3}z^{-1}}\right]$.

$= \frac{1}{2j}\left[\frac{1-e^{-j\pi/3}z^{-1} - (1-e^{j\pi/3}z^{-1})}{(1-e^{j\pi/3}z^{-1})(1-e^{-j\pi/3}z^{-1})}\right]$

$= \frac{1}{2j}\left[\frac{1-e^{-j\pi/3}z^{-1} - 1 + e^{j\pi/3}z^{-1}}{1-e^{-j\pi/3}z^{-1} - e^{+j\pi/3}z^{-1} + z^{-2}}\right]$.

$= \frac{1}{2j}\left[\frac{z^{-1}\left[e^{j\pi/3} - e^{-j\pi/3}\right]}{1-\left(e^{+j\pi/3} + e^{-j\pi/3}\right)z^{-1} + z^{-2}}\right]$.

$= \frac{1}{2j}\left[\frac{2j\, z^{-1}\left[\frac{e^{j\pi/3} - e^{-j\pi/3}}{2j}\right]}{1-2z^{-1}\left(\frac{e^{j\pi/3} + e^{-j\pi/3}}{2}\right)+z^{-2}}\right]$.

$= \frac{z^{-1}\sin(\pi/3)}{1-2z^{-1}\cos(\pi/3)+z^{-2}} = \frac{z^{-1}\left(\sqrt{3}/2\right)}{1-2z^{-1}(1/2)+z^{-2}}$

$= \sin(n\pi/3)u(n) = \frac{\sqrt{3}}{2}\left[\frac{z^{-1}}{\frac{z^2-z+1}{z^2}}\right]$.

$$= \frac{\sqrt{3}}{2}\left[\frac{z(2z-1) - (z^2-z+1)}{(z^2-z+1)^2}\right].$$

$$= \frac{\sqrt{3}}{2}\left[\frac{2z^2 - z - z^2 + z - 1}{(z^2-z+1)^2}\right].$$

$$= \frac{\sqrt{3}}{2}\left[\frac{z^2-1}{(z^2-z+1)^2}\right] \Rightarrow \left[\frac{\sqrt{3} \times z \times (z^2-1)}{2 \times (z^2-z+1)^2}\right].$$

$$= \frac{\sqrt{3}}{2}\left[\frac{(z(z^2-1)}{(z^2-z+1)^2}\right].$$

$$x_2(n) = (0.9)^n u(n-1).$$

$$X_2(z) = \sum_{1}^{\infty} (0.9)^n \cdot z^{-1}.$$

$$X_2(z) = \frac{z^{-1}}{1 - 0.9z^{-1}} = \frac{1}{z - 0.9}.$$

$$X(z) = X_1(z) + X_2(z).$$

$$= \frac{\sqrt{3}(z)(z^2-1)}{2 \times (z^2-z+1)^2} + \frac{1}{z-0.9}.$$

Task#02:

Let $X_1(z) = 2 + 3z^{-1} + 4z^{-2}$ and $X_2(z) = 3 + 4z^{-1} + 5z^{-2} + 6z^{-3}$. Determine $X_3(z) = X_1(z)X_2(z)$.

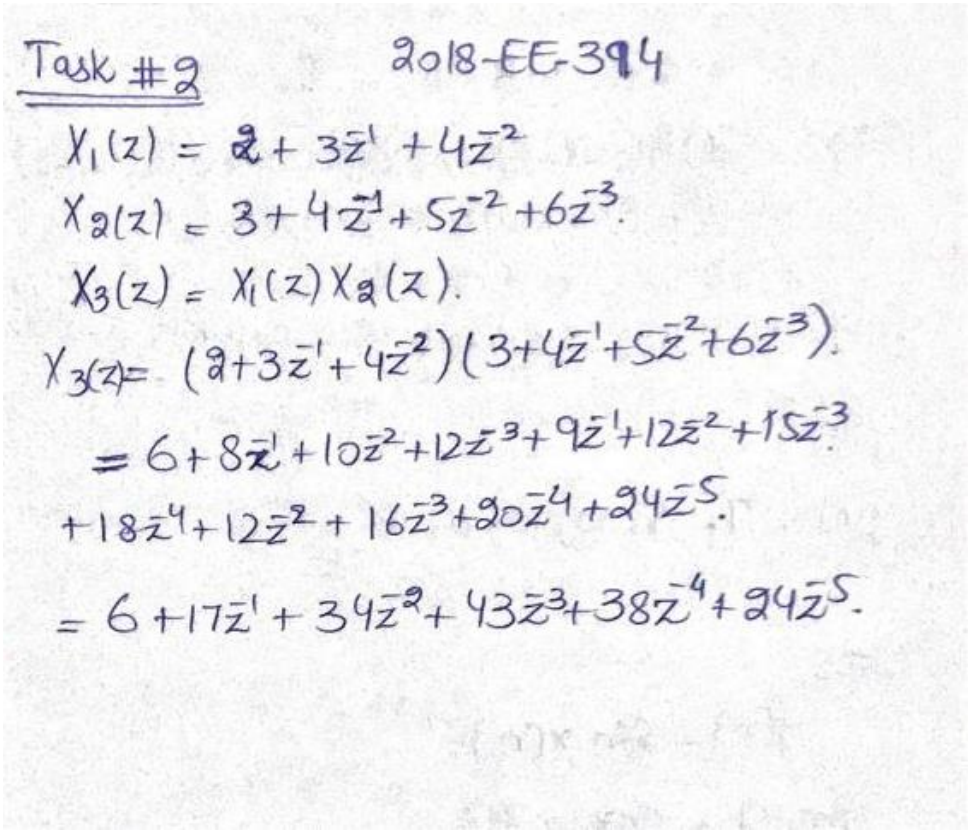
MATLAB Code:

```
%2018-EE-394
%LAB#5 Task#2
X1=[2 3 4];
X2=[3 4 5 6];
X3=conv(X1,X2)

```

Output:

```
Command Window

>> DSPLab5task2

X1 =

     2     3     4


X2 =

     3     4     5     6


X3 =

     6    17    34    43    38    24
```

Handwritten Solution:



Task #2 2018-EE-394

$X_1(z) = 2 + 3z^{-1} + 4z^{-2}$

$X_2(z) = 3 + 4z^{-1} + 5z^{-2} + 6z^{-3}$.

$X_3(z) = X_1(z) X_2(z)$.

$X_3(z) = (2 + 3z^{-1} + 4z^{-2})(3 + 4z^{-1} + 5z^{-2} + 6z^{-3})$.

$= 6 + 8z^{-1} + 10z^{-2} + 12z^{-3} + 9z^{-1} + 12z^{-2} + 15z^{-3}$

$+ 18z^{-4} + 12z^{-2} + 16z^{-3} + 20z^{-4} + 24z^{-5}$.

$= 6 + 17z^{-1} + 34z^{-2} + 43z^{-3} + 38z^{-4} + 24z^{-5}$.

Date: 23-5-2021 **Registration#_2018-EE-394**

University of Engineering and Technology, Lahore

Experiment # 6

Title: Frequency Response of Linear Time Invariant Systems

Equipment Required: Personal computer (PC) with windows operating system and MATLAB software

Task 1:

Let

$$h[n] = a^n\{u[n] - u[n-30]\}, \qquad a \in \{1, 0.9, 0.5, 1\}$$

(i) a=1

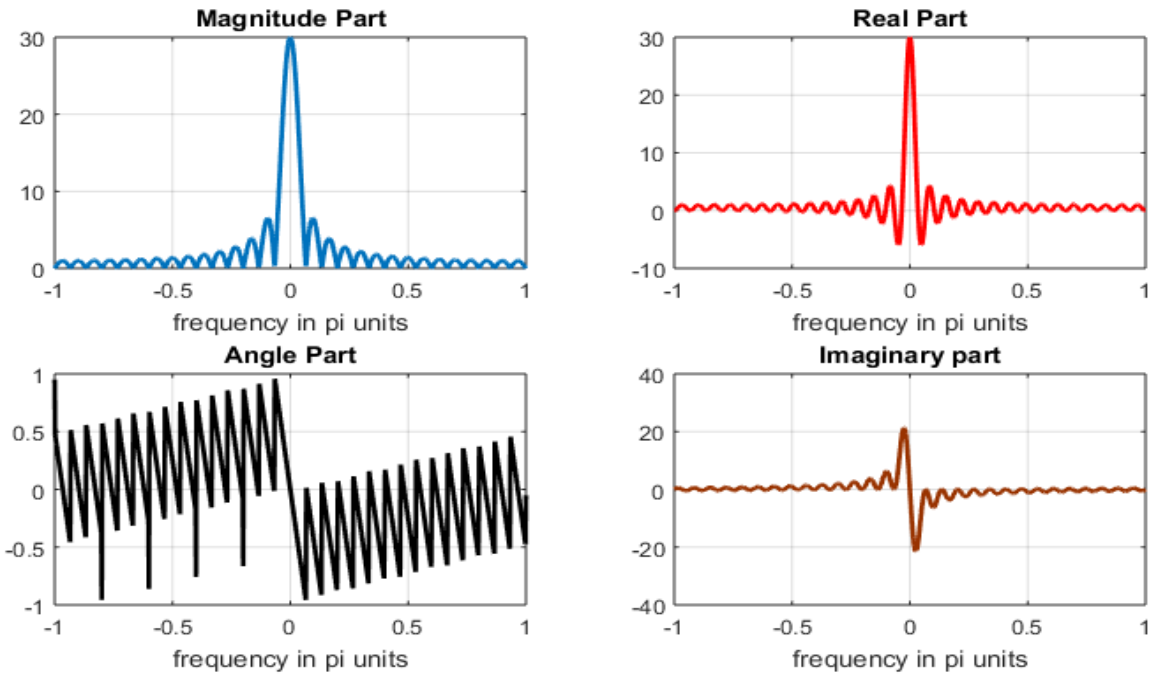
Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 1
%for a=1
n=[0:1:100];
a=1;
x = ((a).^n).*([n>=0]-[n>=30]);
k = -500:500; w = (pi/500)*k;
w=(pi/-500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

Date: 23-5-2021 **Registration#_2018-EE-394**

Plot:



(ii) a=0.9

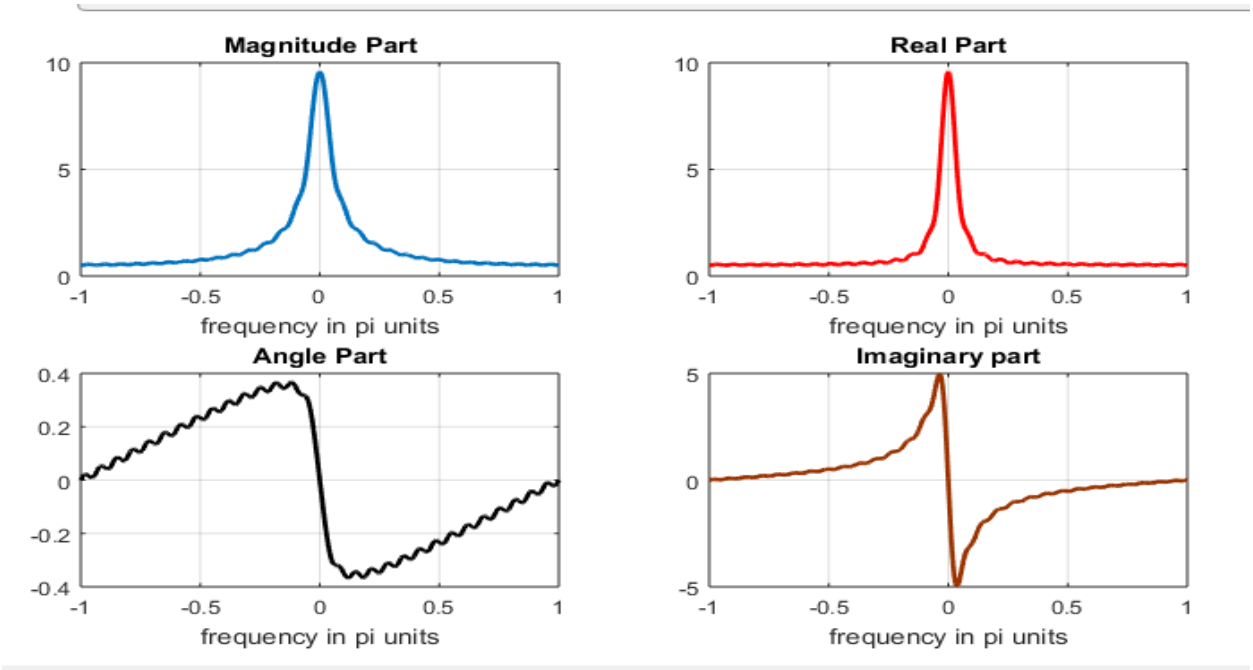
Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 1
%for a=0.9
n=[0:1:100];
a=0.9;
x = ((a).^n).*([n>=0]-[n>=30]);
k = -500:500; w = (pi/500)*k;
w=(pi/-500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

Date: 23-5-2021 **Registration#_2018-EE-394**

Plot:



(iii) a=0.5

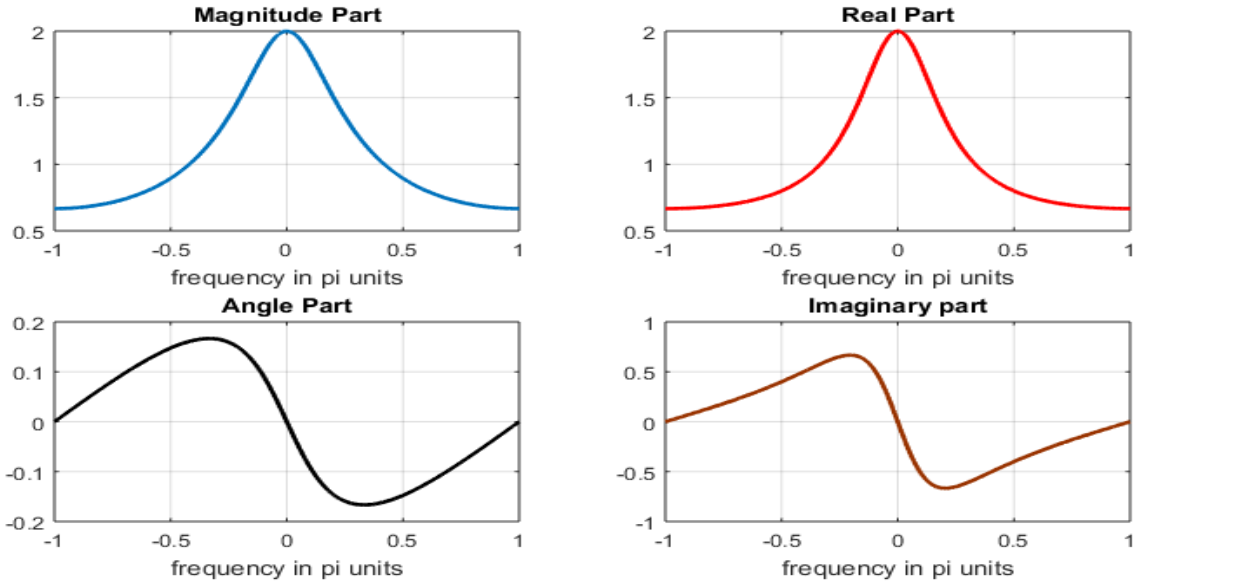
Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 1
%for a=0.5
n=[0:1:100];
a=0.5;
x = ((a).^n).*([n>=0]-[n>=30]);
k = -500:500; w = (pi/500)*k;
w=(pi/-500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

Date: 23-5-2021 **Registration#_2018-EE-394**

Plot:



(iv) a=0.1:

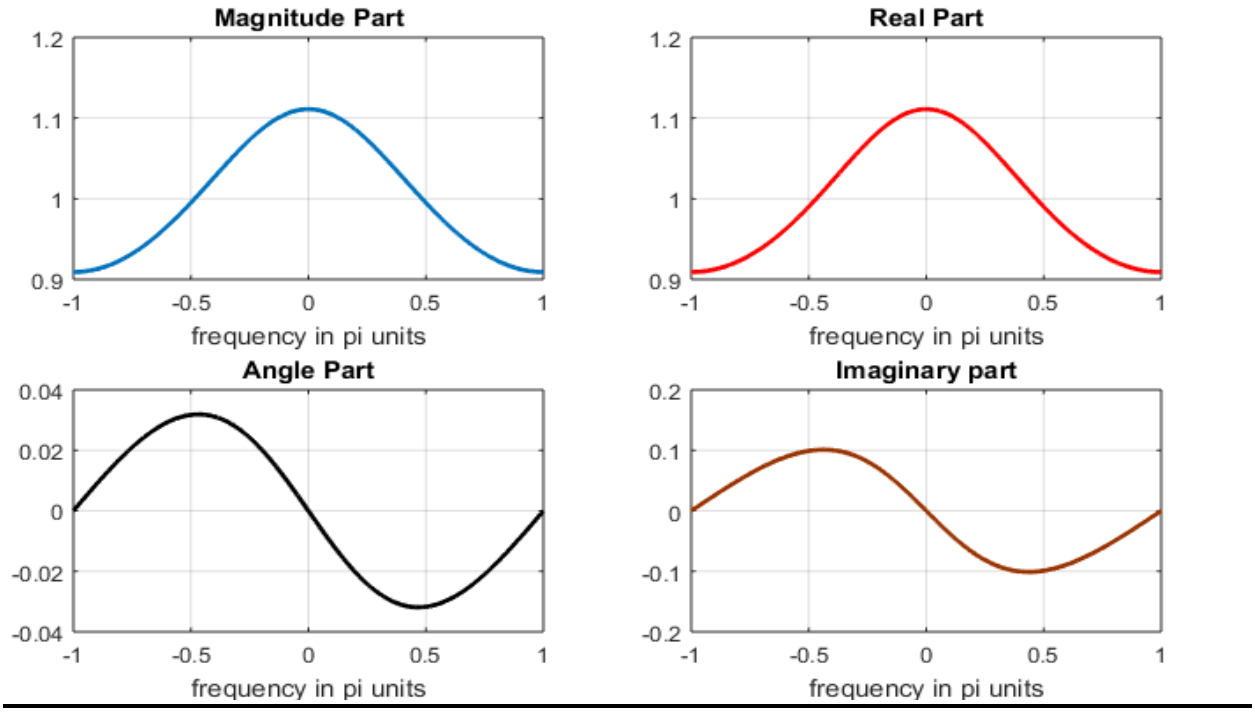
Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 1
%for a=0.1
n=[0:1:100];
a=0.1;
x = ((a).^n).*([n>=0]-[n>=30]);
k = -500:500; w = (pi/500)*k;
w=(pi/-500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

Date: 23-5-2021 **Registration#_2018-EE-394**

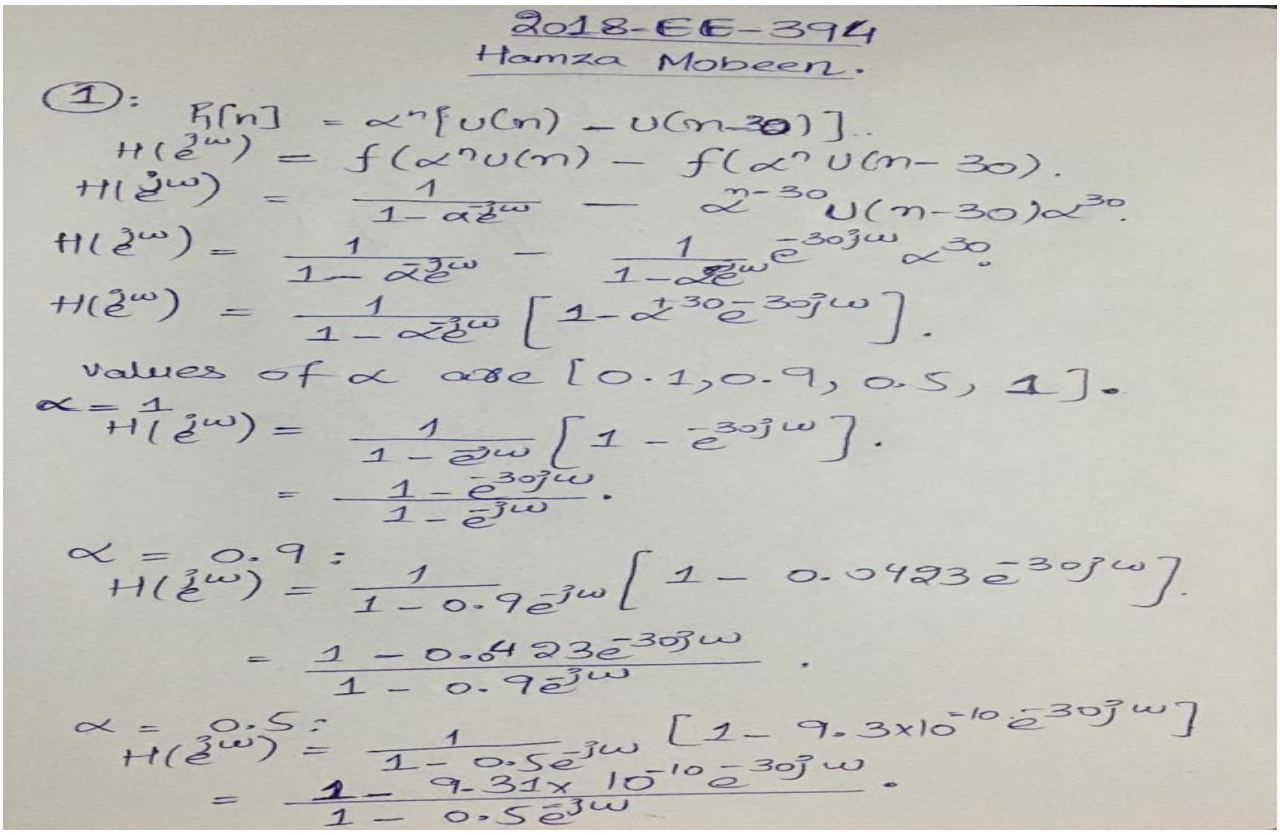
Plot:



Date: 23-5-2021 **Registration#_2018-EE-394**

Questions

1. Calculate Fourier transform H ($e^{j\omega}$) of h[n] on paper for all values of a. What is the system called when a=1. For which value of a, h[n] has the fastest decay?



2018-EE-394
Hamza Mobeen.

(1): $h[n] = \alpha^n [u(n) - u(n-30)]$.

$H(e^{j\omega}) = f(\alpha^n u(n)) - f(\alpha^n u(n-30))$.

$H(e^{j\omega}) = \frac{1}{1-\alpha e^{-j\omega}} - \alpha^{n-30} u(n-30)\alpha^{30}$.

$H(e^{j\omega}) = \frac{1}{1-\alpha e^{-j\omega}} - \frac{1}{1-\alpha e^{-j\omega}} e^{-30j\omega} \alpha^{30}$.

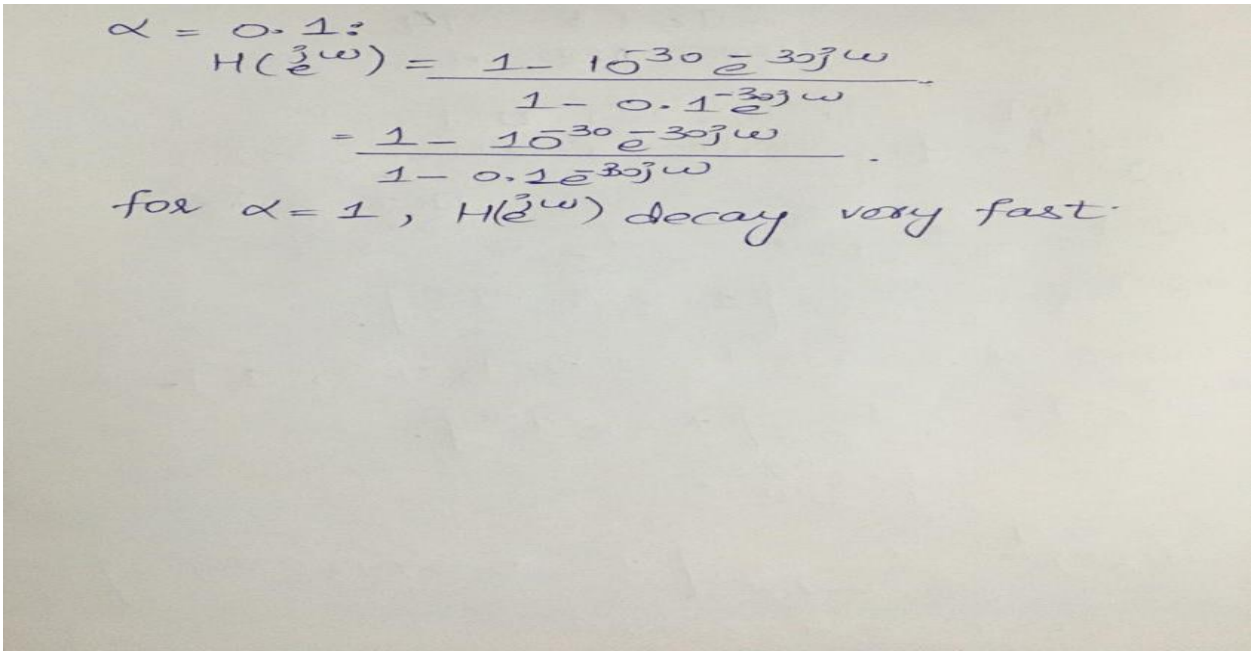
$H(e^{j\omega}) = \frac{1}{1-\alpha e^{-j\omega}} [1 - \alpha^{30} e^{-30j\omega}]$.

values of α are [0.1, 0.9, 0.5, 1].

$\alpha = 1$:
$H(e^{j\omega}) = \frac{1}{1-e^{j\omega}} [1 - e^{-30j\omega}]$.
$= \frac{1 - e^{-30j\omega}}{1 - e^{-j\omega}}$.

$\alpha = 0.9$:
$H(e^{j\omega}) = \frac{1}{1-0.9e^{-j\omega}} [1 - 0.0423 e^{-30j\omega}]$.
$= \frac{1 - 0.0423 e^{-30j\omega}}{1 - 0.9 e^{-j\omega}}$.

$\alpha = 0.5$:
$H(e^{j\omega}) = \frac{1}{1-0.5e^{-j\omega}} [1 - 9.3\times10^{-10} e^{-30j\omega}]$
$= \frac{1 - 9.31 \times 10^{-10} e^{-30j\omega}}{1 - 0.5 e^{-j\omega}}$.



$\alpha = 0.1$:
$H(e^{j\omega}) = \frac{1 - 10^{-30} e^{-30j\omega}}{1 - 0.1 e^{-30j\omega}}$.
$= \frac{1 - 10^{-30} e^{-30j\omega}}{1 - 0.1 e^{-30j\omega}}$.

for $\alpha = 1$, $H(e^{j\omega})$ decay very fast.

Date: 23-5-2021 **Registration#_2018-EE-394**

2. Plot Fourier transform H ($e^{j\omega}$) of h[n] using MAT LAB. Since ω is continuous, MAT LAB cannot simulate it as it is. You can discretize ω axis, calculate H ($e^{j\omega}$) on these discrete values and then use plot command to join them to give you a " continous" look of the Fourier transform. Plot Fourier transform from -2π to -2π. Do you see any repetition of Fourier transform?

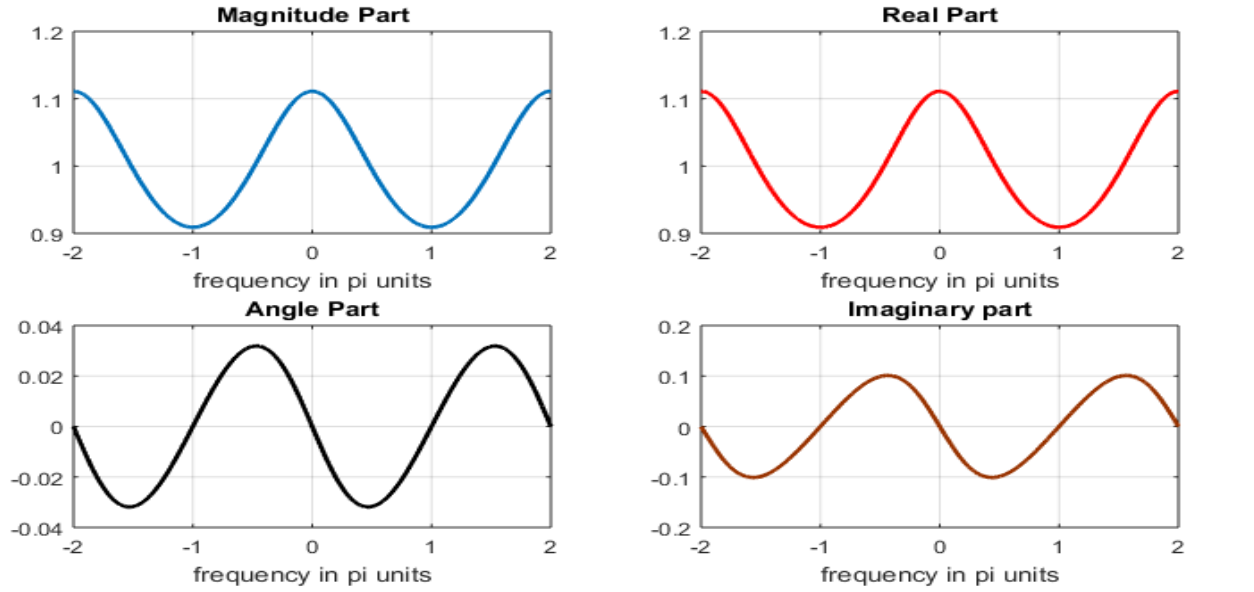
(i) a=0.1:

Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 1
%for a=0.1
n=[0:1:100];
a=0.1;
x = ((a).^n).*([n>=0]-[n>=30]);
k = -1000:1000; w = (pi/500)*k;
w=(pi/-500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')
```

Plot:



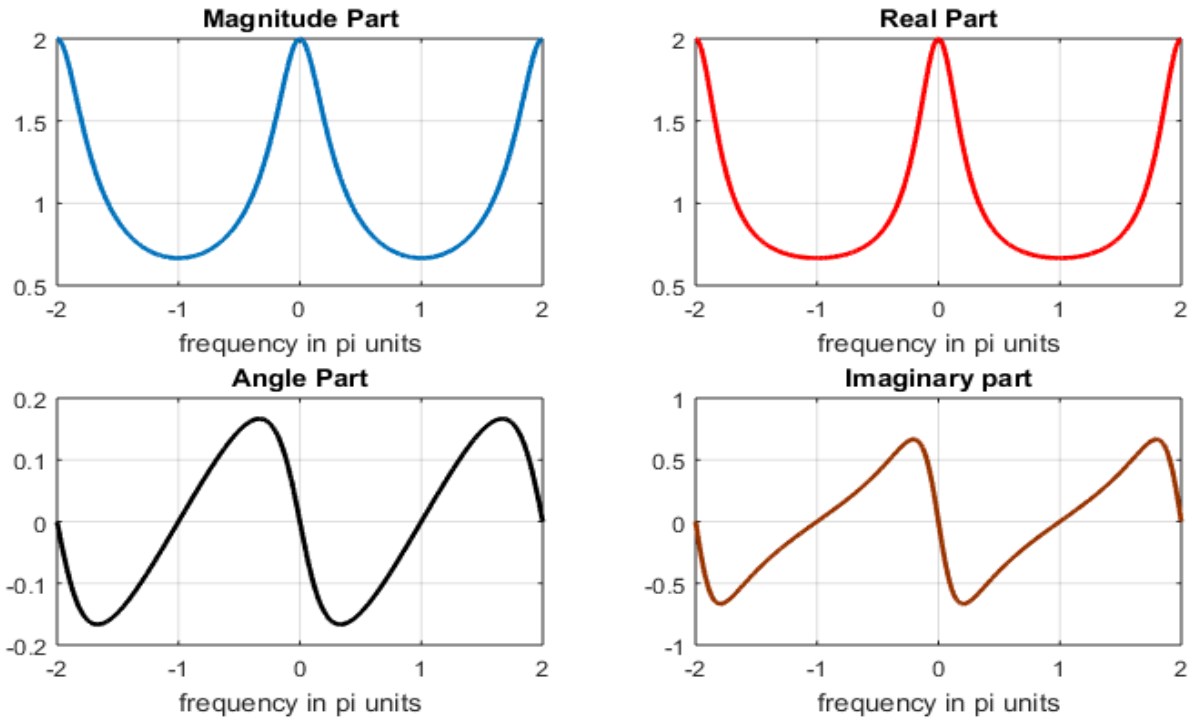
(ii) 0.5

Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 1
%for a=0.5
n=[0:1:100];
a=0.5;
x = ((a).^n).*([n>=0]-[n>=30]);
k = -1000:1000; w = (pi/500)*k;
w=(pi/-500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

Plot:



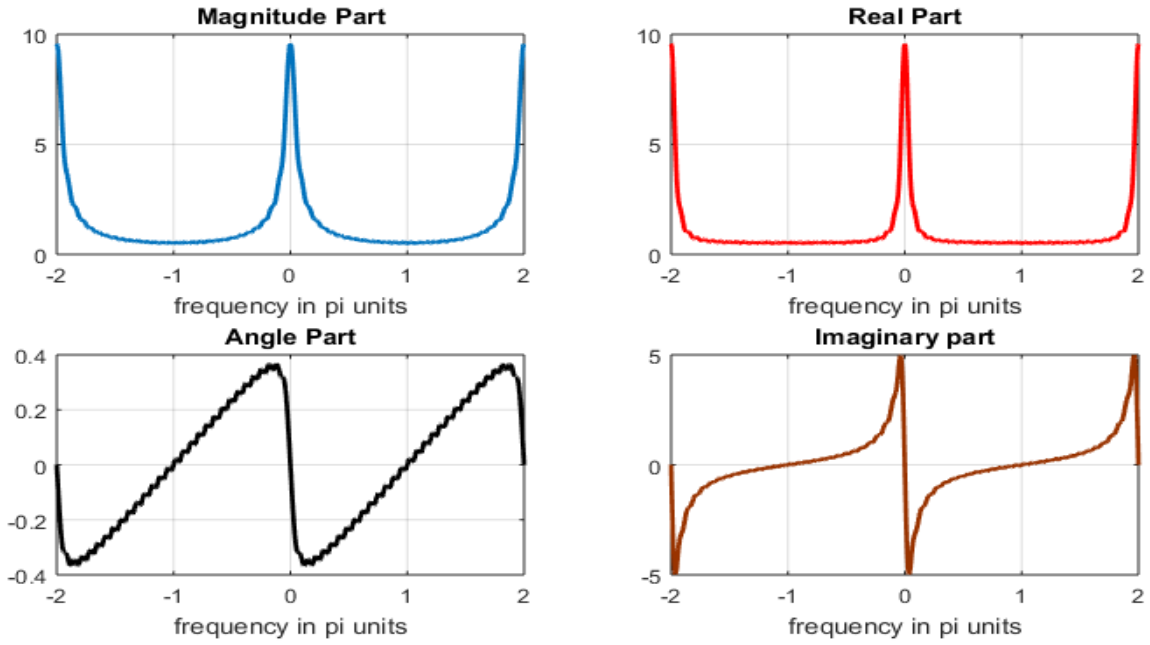
(iii) 0.9

Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 1
%for a=0.9
n=[0:1:100];
a=0.9;
x = ((a).^n).*([n>=0]-[n>=30]);
k = -1000:1000; w = (pi/500)*k;
w=(pi/-500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

Plot:



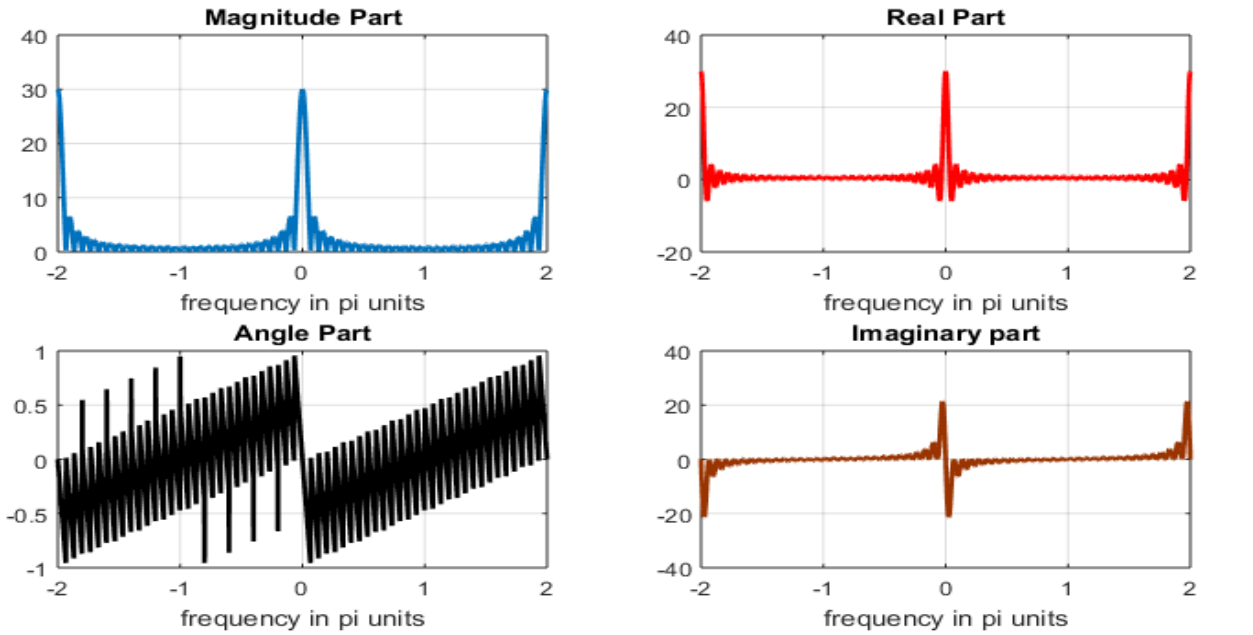
(iv) a=1:

Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 1
%for a=1
n=[0:1:100];
a=1;
x = ((a).^n).*([n>=0]-[n>=30]);
k = -1000:1000; w = (pi/500)*k;
w=(pi/-500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')
```

Plot:



Date: 23-5-2021 **Registration#_2018-EE-394**

3. For what values of a, Fourier transform has higher frequency components and why?

For **a = 1** Fourier transform has highest frequency components because signal is **conjugate symmetric**

Task 2:

Let $h[n] = (a^n + j(\frac{a}{10})^n)\{u[n] - u[n-30]\}$, $a \in \{1, 0.9, 0.5, 1\}$

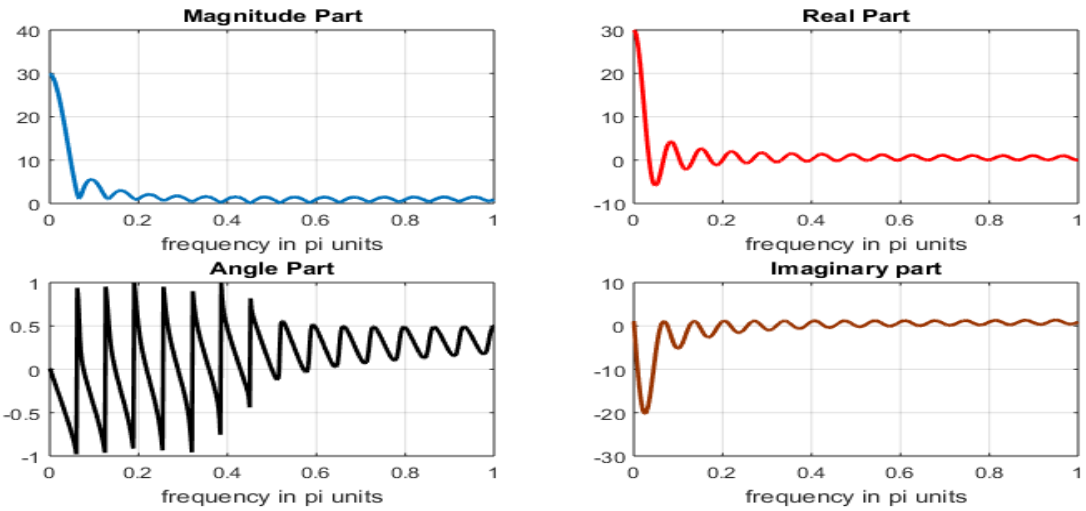
(i) a=1:

Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 2
%for a1
a=1;
n=[0:1:100];
x = ((a.^n)+j*((a/10).^n)).*([n>=0]-[n>=30]);
k = 0:500; w = (pi/500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

Plot:



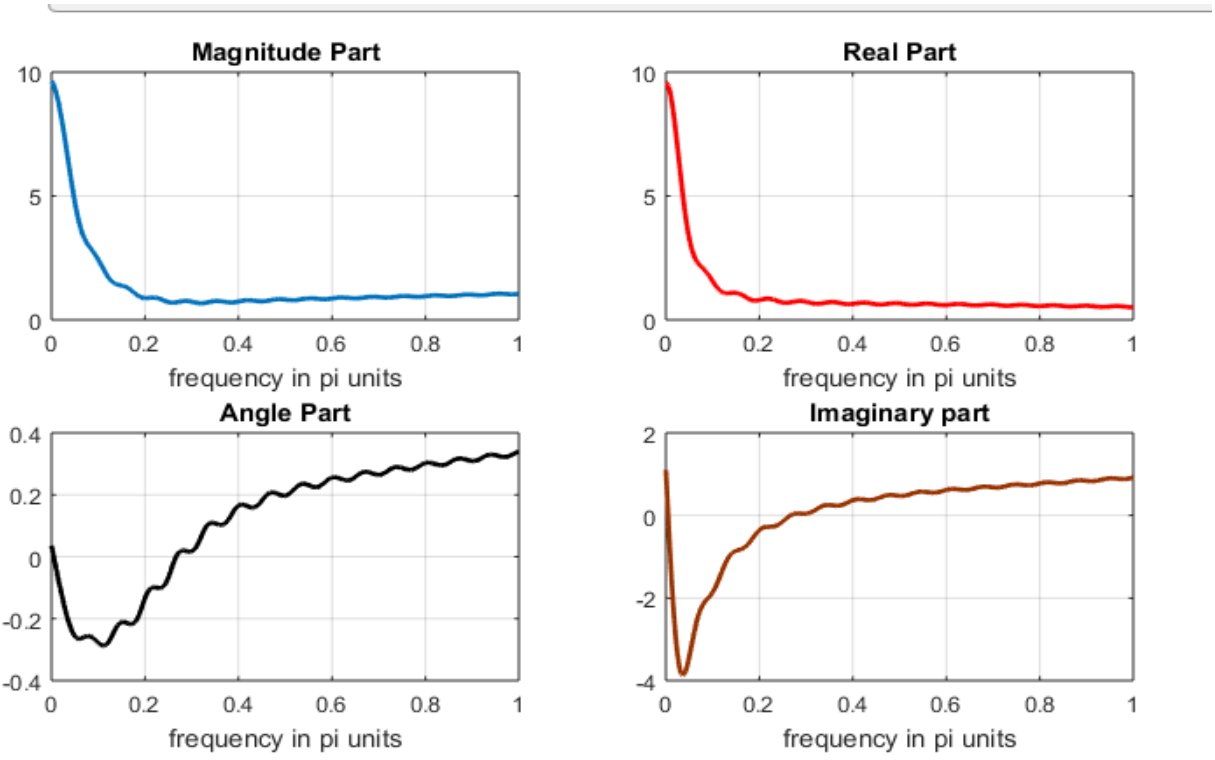
(ii) a=0.9

Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 2
%for a=0.9
a=0.9;
n=[0:1:100];
x = ((a.^n)+j*((a/10).^n)).*([n>=0]-[n>=30]);
k = 0:500; w = (pi/500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

Plot:



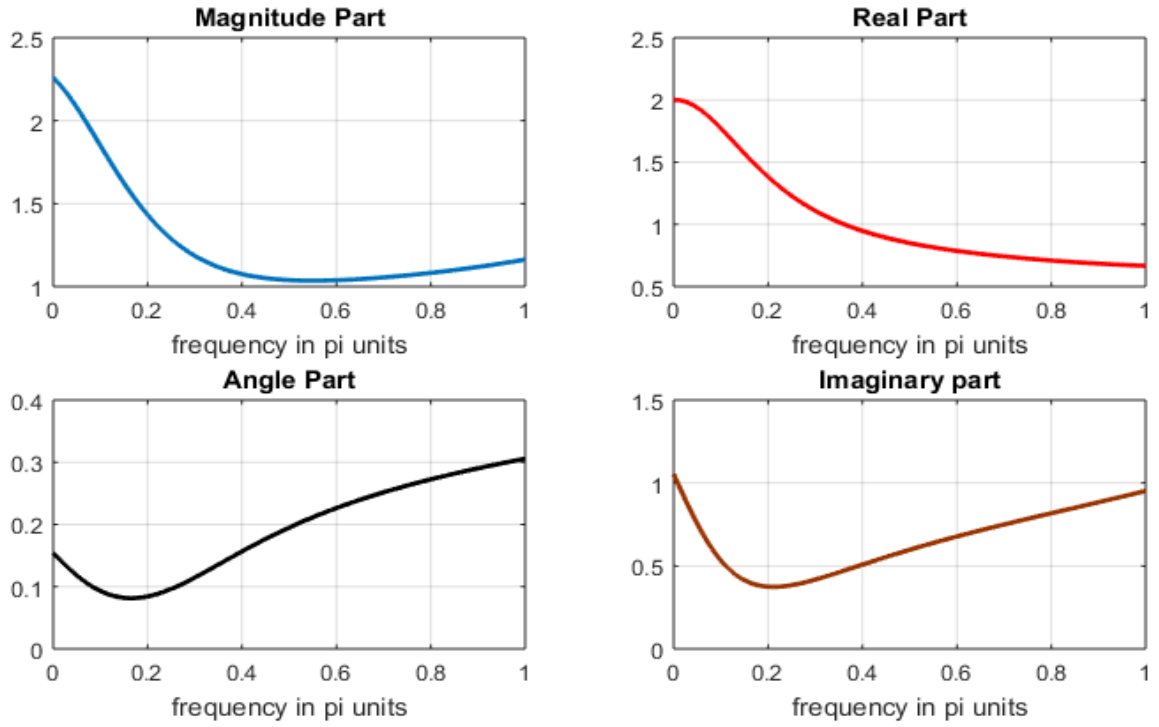
(iii) a=0.5

Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 2
%for a=0.5
a=0.5;
n=[0:1:100];
x = ((a.^n)+j*((a/10).^n)).*([n>=0]-[n>=30]);
k = 0:500; w = (pi/500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')
```

Plot:



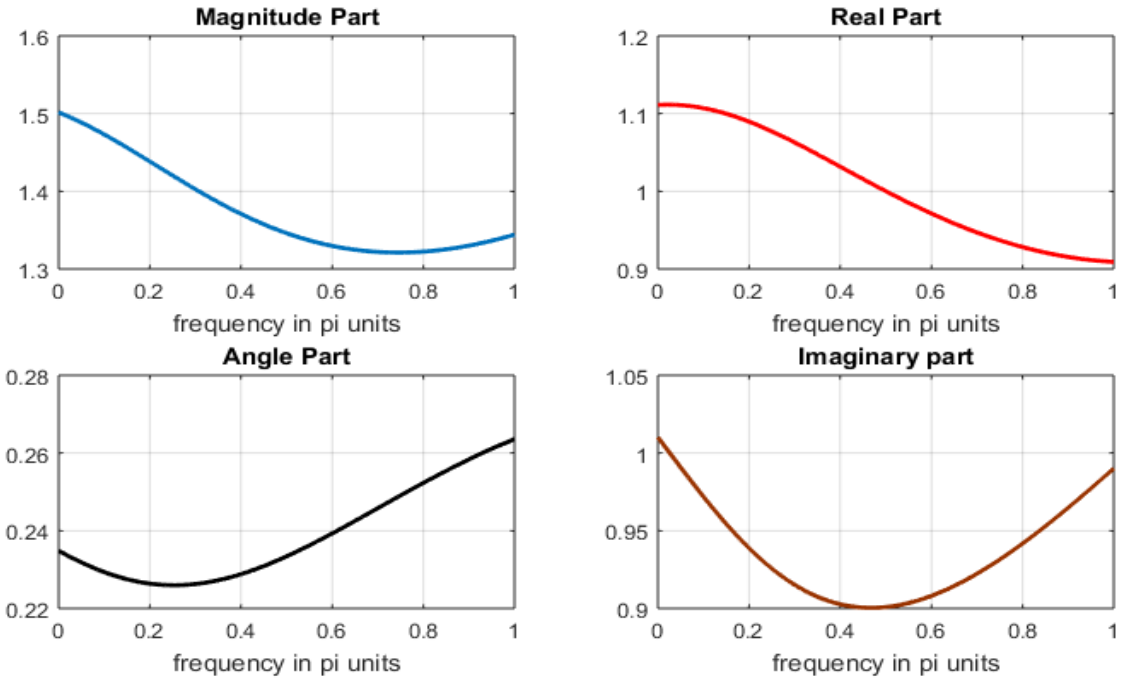
(iv) a=0.1:

Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 2
%for a=0.1
a=0.1;
n=[0:1:100];
x = ((a.^n)+j*((a/10).^n)).*([n>=0]-[n>=30]);
k = 0:500; w = (pi/500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

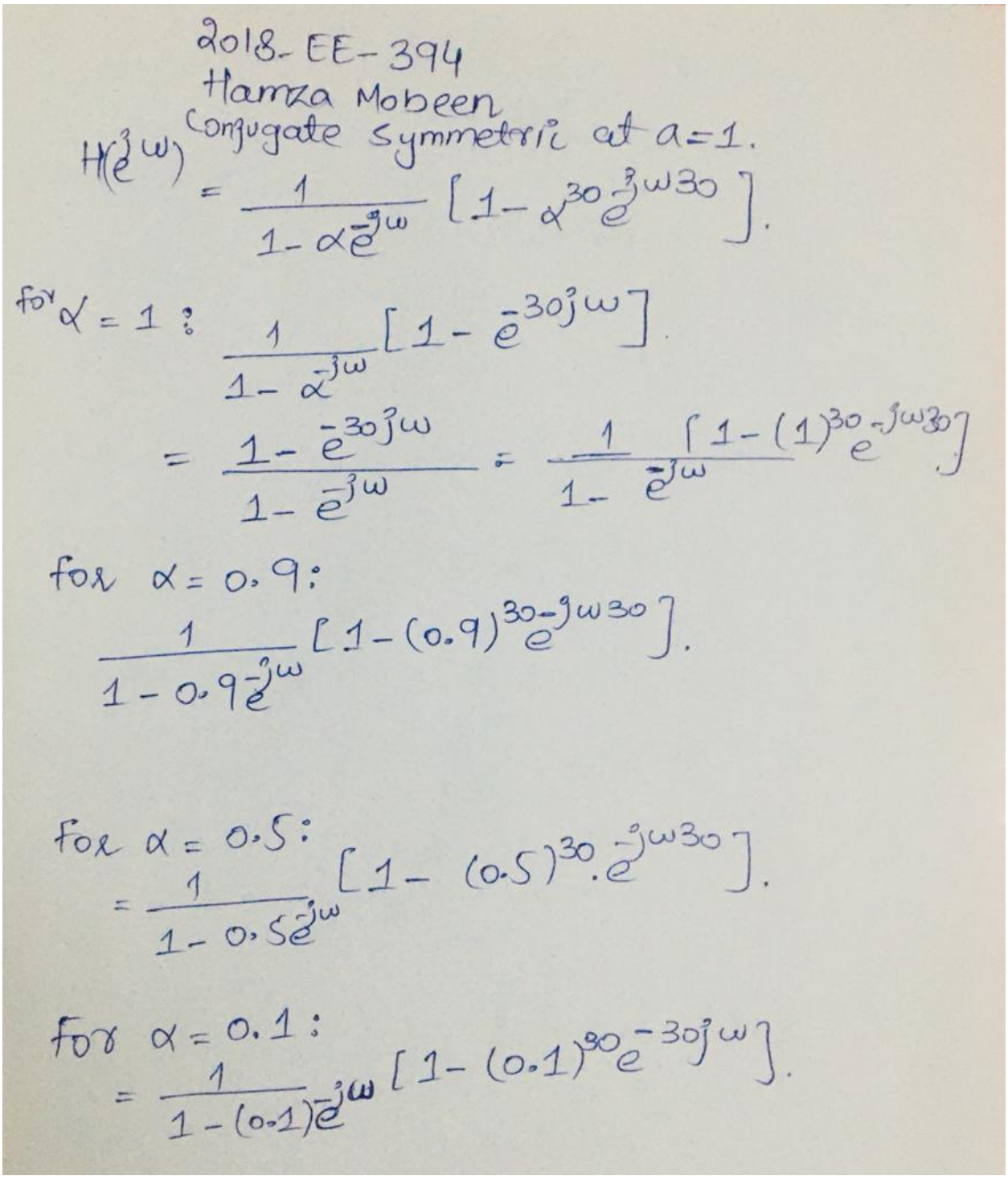
Plot:



Date: 23-5-2021 **Registration#_2018-EE-394**

Question

1. Plot Fourier transform H ($e^{j\omega}$) of h[n] using MAT LAB. Is Fourier transform conjugate symmetric?



Date: 23-5-2021 **Registration#_2018-EE-394**

Task 3:

Plot Fourier transform of following functions.

$$h[n] = (a^n \cos\left(\frac{\pi}{4}n\right) u[n], \qquad a \in \{0.5\}$$

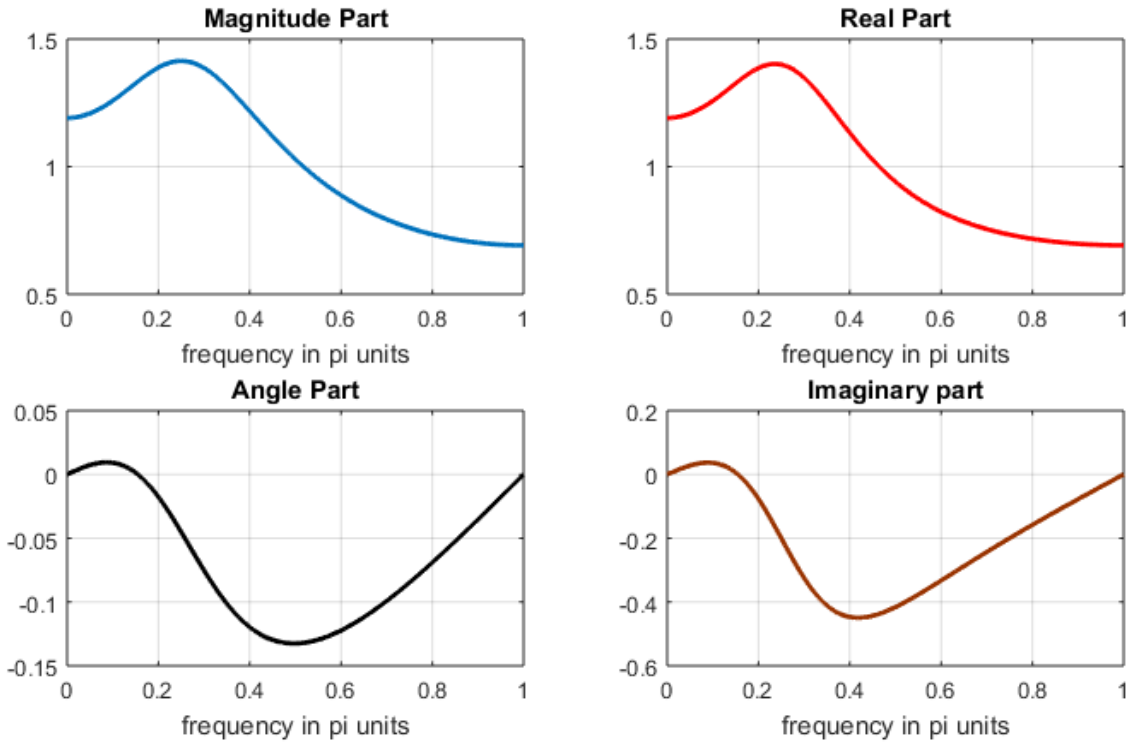
Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 3 part 1
n>=0;
x=(((0.5).^n).*cos((pi/4).*n));
k=0:500;
w=(pi/500)*k;
X=x*(exp(-j*pi/500)).^(n'*k);
magX=abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')

```

Plot:

Date: 23-5-2021 **Registration#_2018-EE-394**



$h[n] = \{7,0,0,0,1,6,1,4\}$ $-4 \leq n \geq 3$

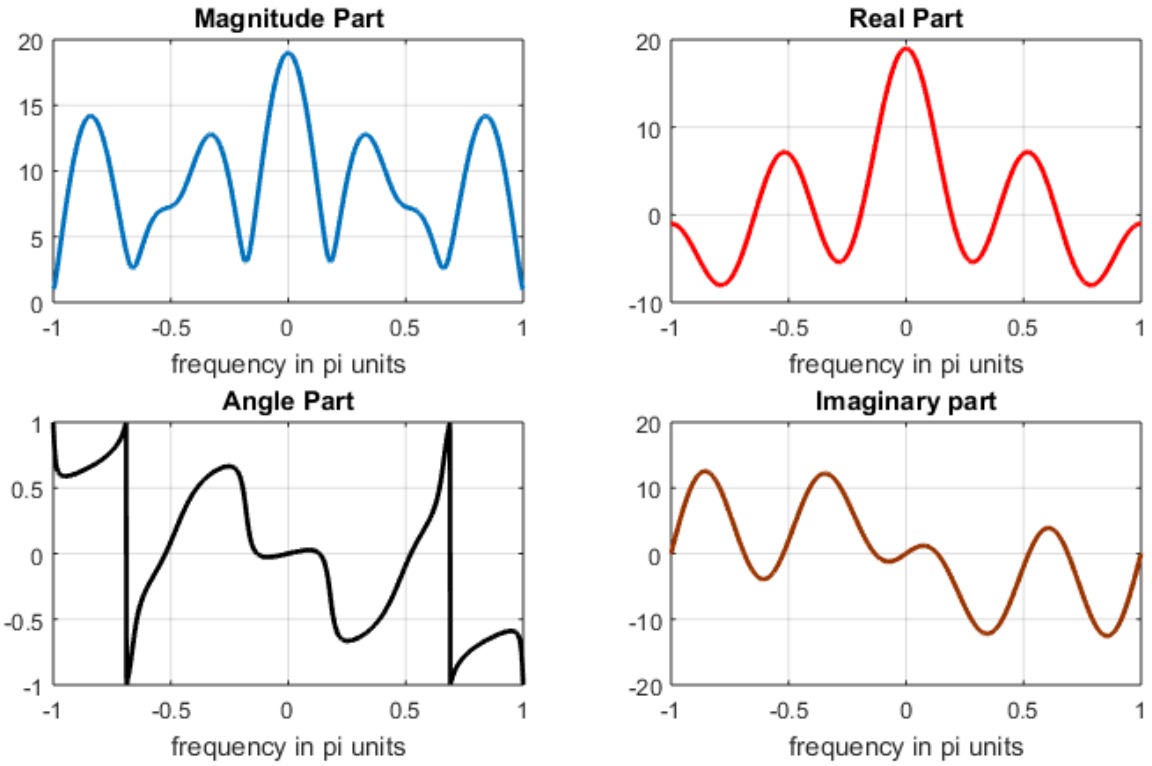
Code:

lab6_1.m | lab6_2.m | lab6_3_1.m | lab6_3_2.m

```
%2018-EE-394
%Hamza Mobeen
%lab 6 task 3 part 2
n=-4:3;
x=[7,0,0,0,1,6,1,4];
k=-500:500; w=(pi/500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX = abs(X); angX = angle(X);realX = real(X); imagX = imag(X);
subplot(2,2,1); plot(k/500,magX);grid
xlabel('frequency in pi units'); title('Magnitude Part')
subplot(2,2,3); plot(k/500,angX/pi);grid
xlabel('frequency in pi units'); title('Angle Part')
subplot(2,2,2); plot(k/500,realX);grid
xlabel ( 'frequency in pi units'); title ('Real Part')
subplot(2,2,4); plot(k/500,imagX);grid
xlabel('frequency in pi units'); title('Imaginary part')
```

Plot:

Date: 23-5-2021 **Registration#_2018-EE-394**



Date: 30-05-2021 **Registration#_2018-EE-394**

University of Engineering & Technology Lahore, FSD Campus

Experiment # 7

Title: **Inverse z-transform**

Equipment Required: Personal computer (PC) with windows operating system and MATLAB software

Task 1

$$H(z) = \frac{1-2\cos(0.18\pi)z^{-1}+2z^{-2}}{1+0.5z^{-1}}$$

Questions

1. Find inverse Z-transform of H(z) on paper. Find all poles and zeros and draw them in z-plane

2018-EE-394
Hamza Mobeen

Task #1 : $H(z) = \frac{1 - 2\cos(0.18\pi)z^{-1} + 2z^{-1}}{1 + 0.5z^{-1}}$

$H_1(z) = \frac{1}{z^{-1}(z+0.5)}$, $H_2(z) = \frac{-2\cos(0.18\pi)z^{-1}}{z^{-1}(z+0.5)}$.

$H_3(z) = \frac{2z^{-1}}{z^{-1}(z+0.5)}$.

$\alpha^n u[n] \xleftrightarrow{z} \frac{1}{a - z^{-1}} = \frac{z}{z-a}$.

$H_1[n] = (-0.5)^n u[n]$

$H_2[n] = \frac{2\cos(0.18\pi)}{z+0.5} = 2\cos(0.18\pi)(-0.5)^{n-1}u[n-1]$

$H_3[n] = \frac{2}{z^2+0.5z} = \frac{2}{z(z+0.5)} = \frac{A}{z} + \frac{B}{z+0.5}$

By Partial fraction Method.

$A = 4$, $B = -4$.

$H_3[n] = 4\delta(n-1) - 4(-0.5)^{n-1}u[n-1]$.

$H[n] = (-0.5)^n u[n] - 2\cos(0.18\pi)(-0.5)^{n-1}u(n-1)$
$+ 4\delta[n-1] - 4(0.5)^{n-1}u(n-1)$.

$$= (-0.5)^n u(n) - 2\cos(0.18\pi)(-0.5)^n (-0.5)^{-1} u(n-1)$$
$$- 4(-0.5)^n (-0.5)^{-1} u(n-1) + 4\delta(n-1).$$

$$= (-0.5)^n u(n) - 2\cos(0.18\pi)(-0.5)^n (-0.5)^{-1} u(n-1)$$
$$-4(-0.5)^n (-0.5)^{-1} u(n-1) + 4\delta(n-1).$$

$$= (-0.5)^n u(n) + (3.377(-0.5)^n 4\delta(-0.5)^n] +$$
$$(n-1) + 4\delta(n-1).$$

$$= (-0.5)^n u(n) + 11.377(-0.5)^n u(n-1) + 4\delta(n-1)$$

$$\delta(n) = u[n] - u[n-1]$$
$$u[n-1] = u[n] - \delta[n]$$

$$h[n] = 12.377(0.5)^n u[n] - 11.3778[n] + 4\delta(n-1)$$

2. Find inverse Z-transform of H(z) using MATLAB. Plot all poles and zeros in z-plane using MATLAB. Read the help topics on the following functions:

   a) roots **b)** residuez **c)** zplane

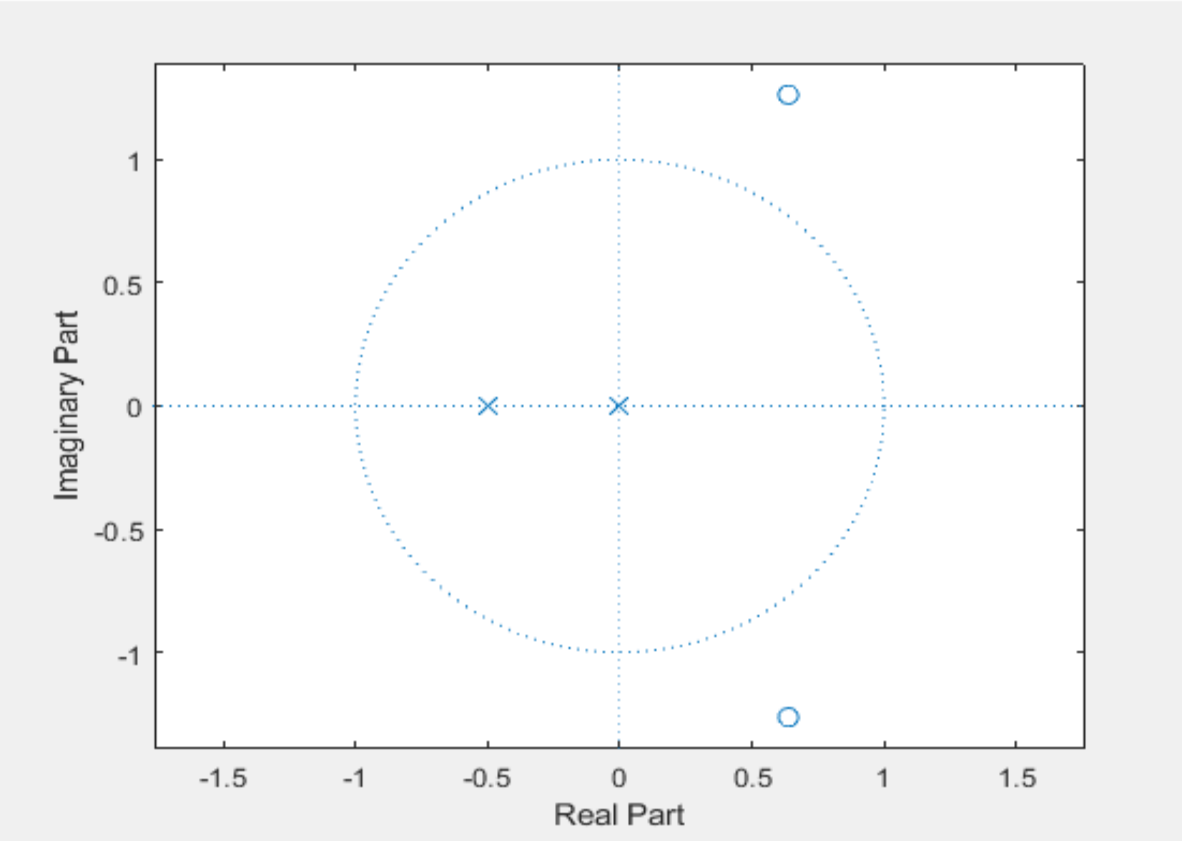
MATLAB Code:

```
1       %2018-EE-394
2       %DSP  lab 7 %task 1_2
3 -     a=[1 0.5];
4 -     b=[1 -2*(cos(0.28*pi)) 2];
5 -     zplane(b,a);
6 -     [R P C]=residuez(b,a)
```

Command Window

```
R =

   11.5497


P =

   -0.5000


C =

  -10.5497    4.0000
```

Output:

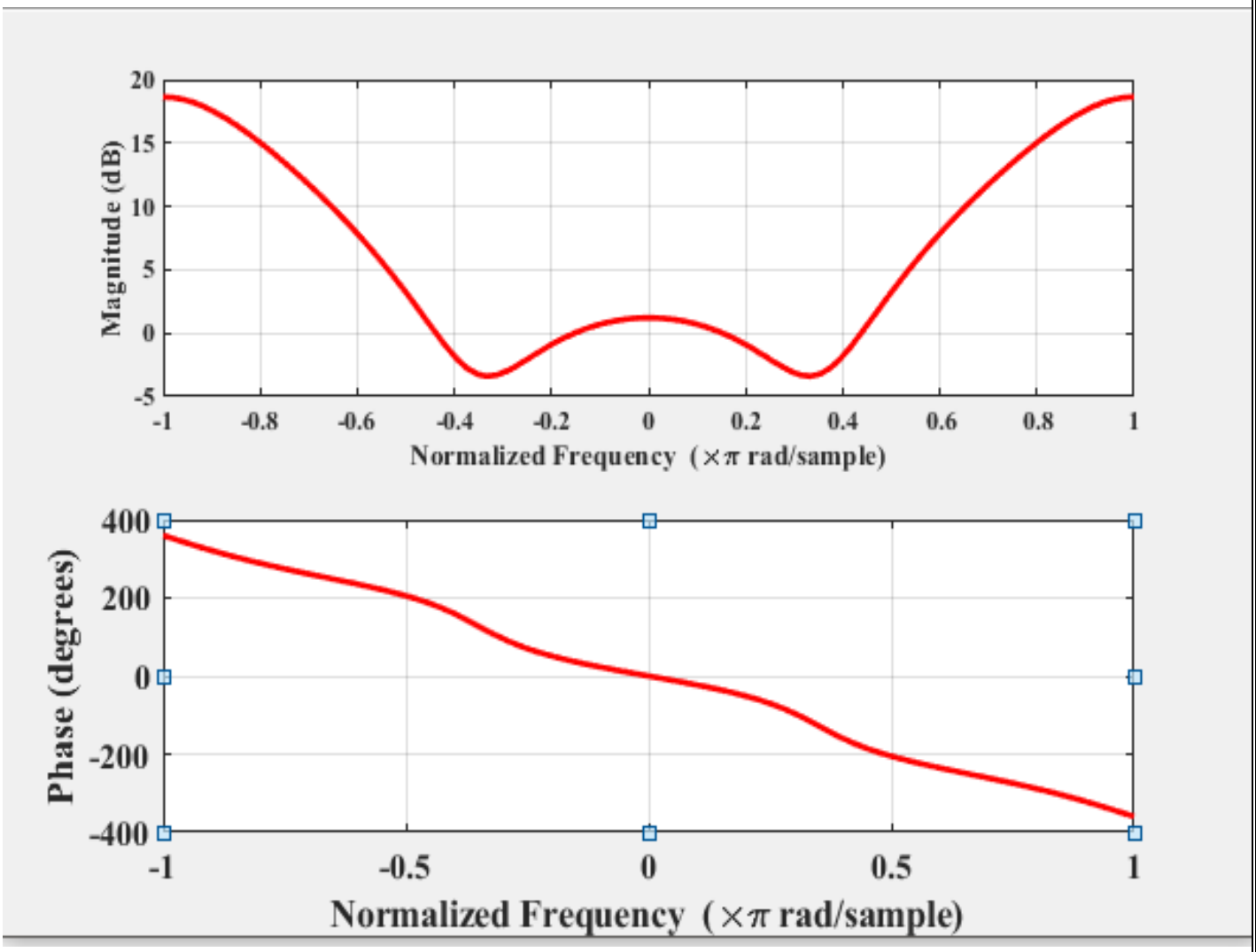


3. Substitute $z = e^{j\omega}$ in the expression above and calculate its Fourier transform H($e^{j\omega}$) on paper. Plot the response using MATLAB. Based on the shape of it's response, what kind of a system do you think it is (stable or not)?

MATLAB Code:

```
%2018-EE-394
%DSP LAB 7 task 1_3
a=[1 0.5];
b=[1 -2*(cos(0.28*pi)) 2];
zplane(b,a);
c=linspace(-pi,pi);
freqz(b,a,c)
```

Output:



Theoretically:

2018-EE-394

$(z) \rightarrow e^{j\omega}$

$$H(e^{j\omega}) = \frac{1 - 2\cos(0.18\pi)e^{-j\omega} + 2e^{-2j\omega}}{(1 + 0.5e^{-j\omega})}$$

$$= \frac{1}{(1 + 0.5e^{-j\omega})} - \frac{2\cos(0.18\pi)e^{-j\omega}}{(1 + 0.5e^{-j\omega})} + \frac{2e^{-2j\omega}}{(1 + 0.5e^{-j\omega})}.$$

$$H_1(e^{j\omega}) = \frac{1}{1 + 0.5e^{-j\omega}} \Rightarrow h_1[n] = (-0.5)^n u[n].$$

$$H_2(e^{j\omega}) = \frac{2\cos(0.18\pi)e^{-j\omega}}{1 + 0.5e^{-j\omega}} = 2\cos(0.18\pi)(0.5)^{n-1} u(n-1).$$

$$H_3(e^{j\omega}) = \frac{2e^{-2j\omega}}{(1 + 0.5e^{-j\omega})} = 2(-0.5)^{n-2} u(n-2].$$

$$h[n] = [(-0.5)^n u(n] - 2\cos(0.18\pi)(-0.5)^{n-1} u(n-1) + 2(-0.5)^{n-2} u(n-2).$$

Task 2

$$X(z) = \frac{1}{(1 + 0.9z^{-1})(1 - 0.9z^{-1})^2}$$

1. Compute the inverse z transform using “residuez” function and verify result using MATLAB.

MATLAB Code:

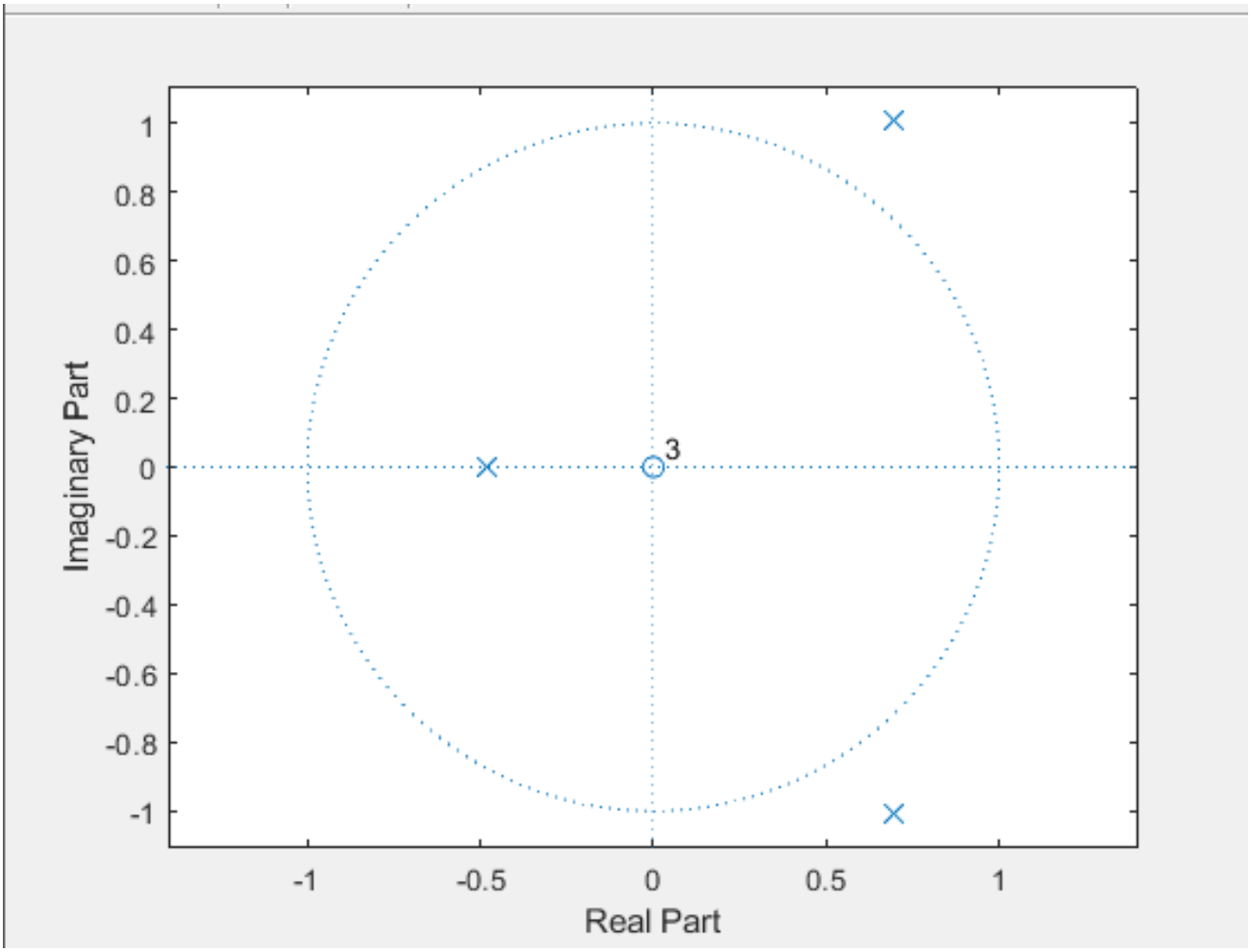
Lab08Task2.m

```
%2018-EE-394
%DSP LAB 07 %Task 2
b=[1];
a=[1 -0.9 0.82 0.72];
[r,p,k]=residuez(b,a)
zplane(b,a)

```

Command Window

```
r =

   0.4511 - 0.1607i
   0.4511 + 0.1607i
   0.0978 + 0.0000i


p =

   0.6918 + 1.0052i
   0.6918 - 1.0052i
  -0.4835 + 0.0000i


k =

     []
```

Output:



Task 3

Let $y[n] = 0.9y[n - 1] + x[n]$

Questions

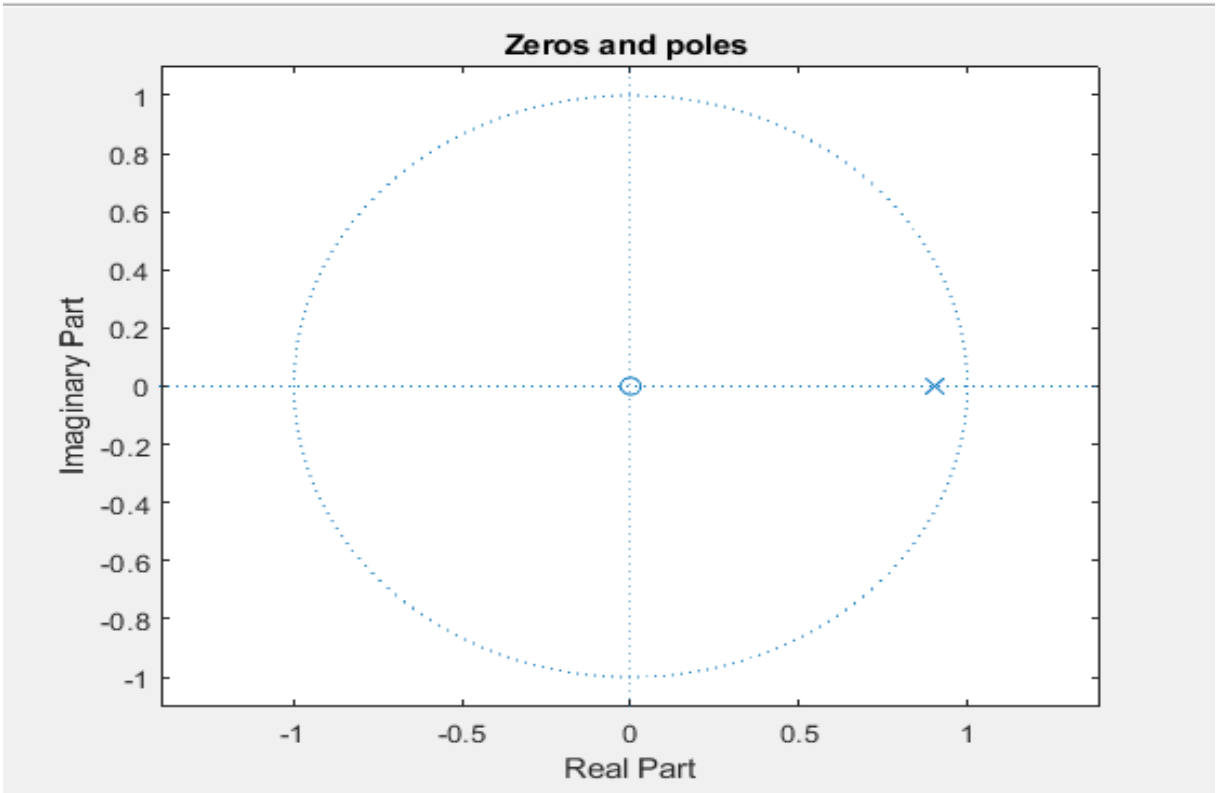
1. Determine H(z) using MATLAB and sketch all poles and zeros in z-plane

MATLAB Code:

Lab8Task3_1.m

```
%2018-EE-394
%DSP LAB 07 %Task 3_1
b=[1];
a=[1 -0.9];
zplane(b,a)
title('Zeros and poles ');

```

Output:



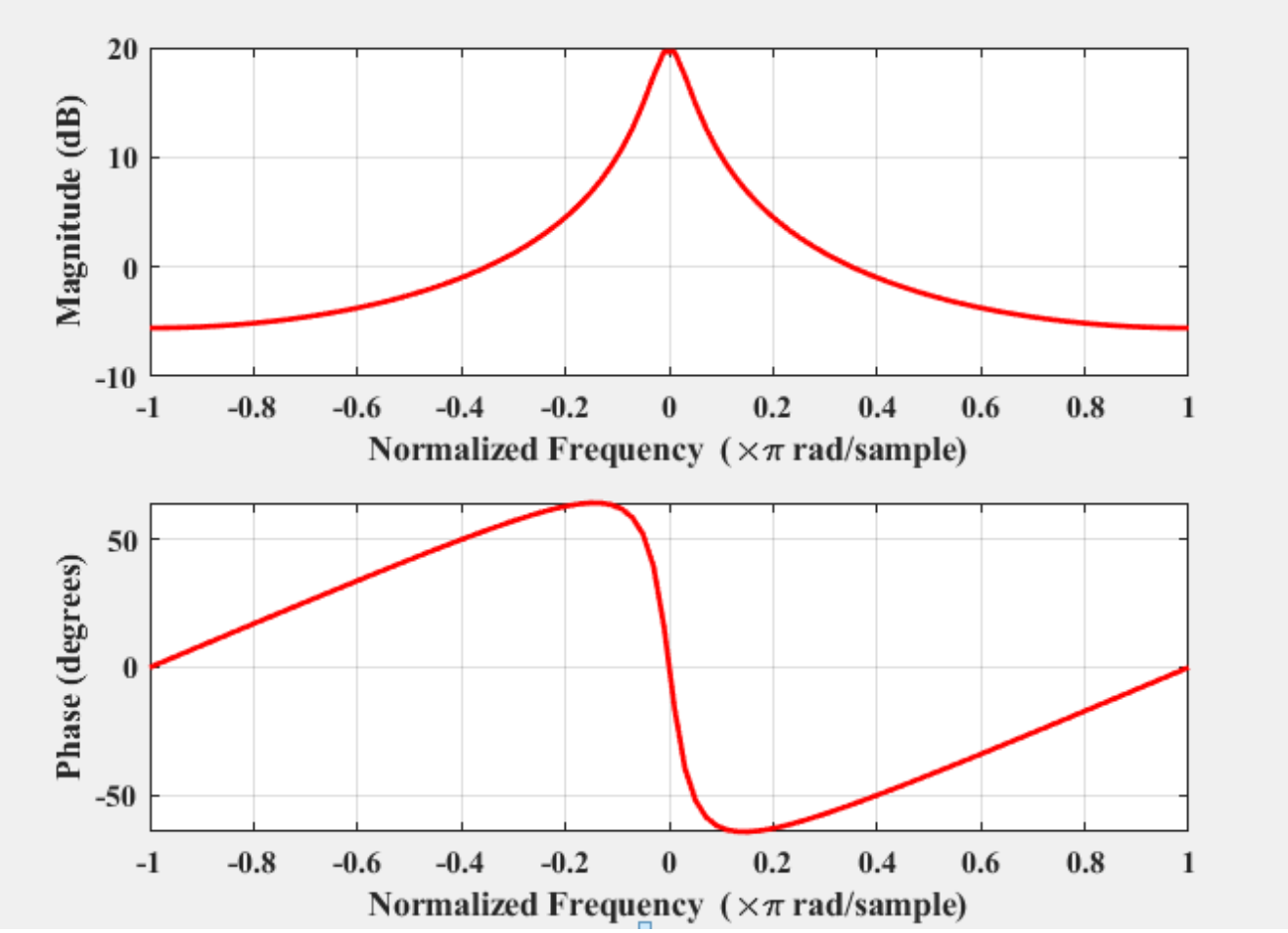
2. Determine and magnitude and angle of H($e^{j\omega}$) using MATLAB. Read the help topic (freqz)

MATLAB Code:

LAB8Task3_2.m

```
%2018-EE-394
% LAB 07 %TasDSPk 3_2
b=[1];
a=[1 -0.9];
zplane(b,a)
title('Zeros and poles ');
L=linspace(-pi,pi);
freqz(b,a,L)
```

Output:



Conclusion:

In this lab we have learnt about the Inverse z transform ,we have learnt some new commands like residuez and freqz. Residuez command helps to converts the partial fraction expansion back to the original polynomial coefficients. freqz determines the transfer **function** from the (real or complex) numerator and denominator polynomials you specify, and returns the complex frequency response $H(e^{j})$ of a digital filte.We have also find zeroes and poles of z transform functions

Date: 05-06-2021 **Registration#_2018-EE-394**

University of Engineering & Technology Lahore, FSD Campus

Experiment # 8

Title: **Multirate Signal Processing**

Equipment Required: Personal computer (PC) with windows operating system and MATLAB.

Task 1

Generate the following sequences:

$$x_1(n) = \sin(2\pi f n), \quad f = 0.1\ H$$

$$x_2(n) = \sin(2\pi f n), \quad f = 0.3\ Hz$$

$$x(n) = x_1(n) + x_2(n)$$

$$y(n) = x_1(n) * x_2(n)$$

Read the helping topics

- DOWNSAMPLE
- UPSAMPLE
- INTERP
- DECIMATE
- RESAMPLE

Questions

Perform the factor-of-4 down-sampling. Plot the original and down-sampled sequences.

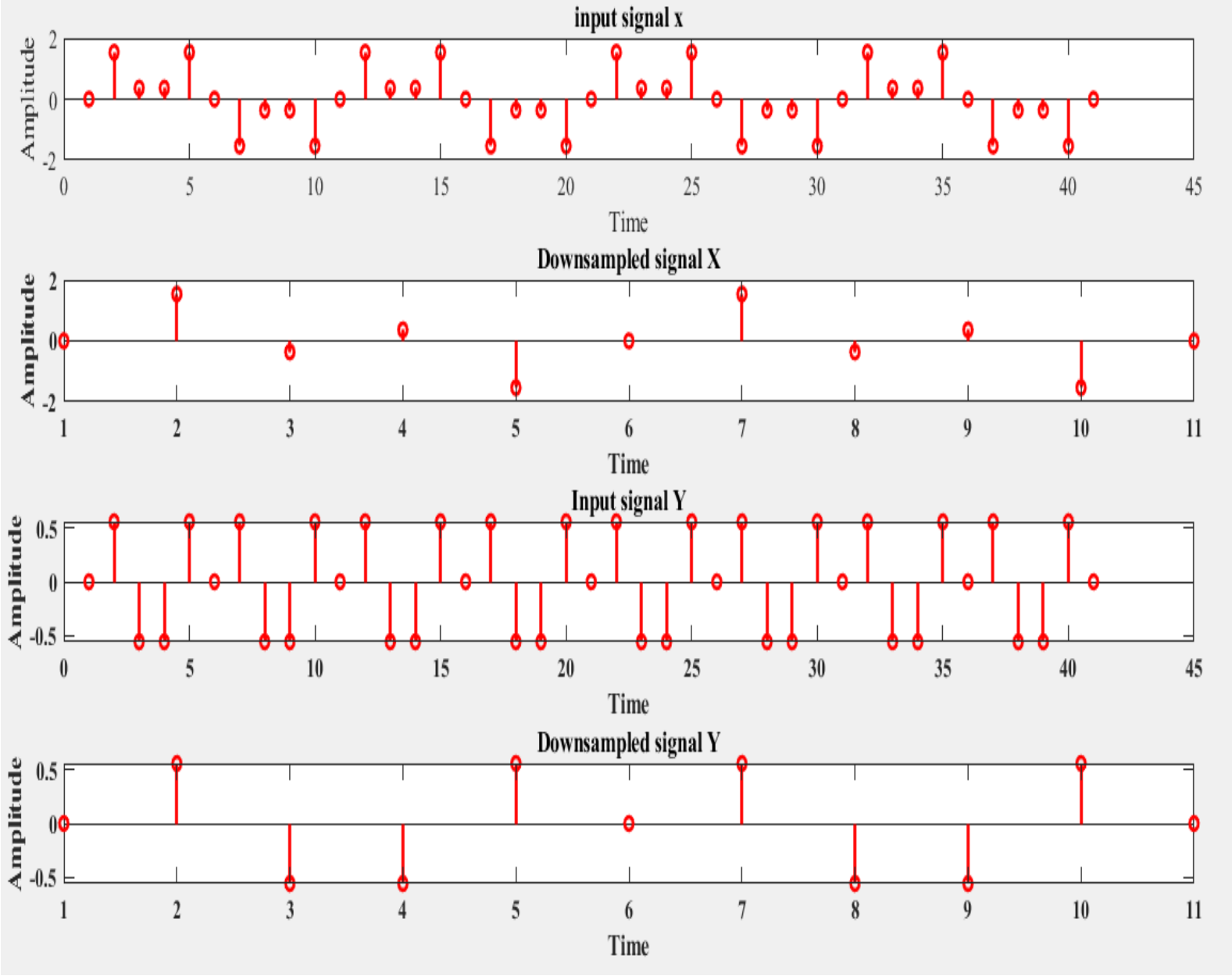
Part (a) for the factor-of-5 up-sampler. Plot the original and up-sampled sequences.

1. Perform the factor-of-4 down-sampling. Plot the original and down-sampled sequences.

MATLAB CODE:

```
%2018-EE-394 LAB08 Task 1 DSP
n = 0:40;
x1 = sin(2*pi*0.1*n);
x2 = sin(2*pi*0.3*n);
x = x1+x2;
y = x1.*x2;
M = 4;
X = downsample(x,M);
Y = downsample(y,M);
subplot(4,1,1);
stem(x)
xlabel('Time');
ylabel('Amplitude');
title('input signal x');
subplot(4,1,2);
stem(X)
xlabel('Time');
ylabel('Amplitude');
title('Downsampled signal X');
xlabel('Time');
subplot(4,1,3);
stem(y)
xlabel('Time');
ylabel('Amplitude');
title('Input signal Y');
xlabel('Time');
subplot(4,1,4);
stem(Y)
xlabel('Time');
ylabel('Amplitude');
title('Downsampled signal Y');
xlabel('Time');
```

OUTPUT:

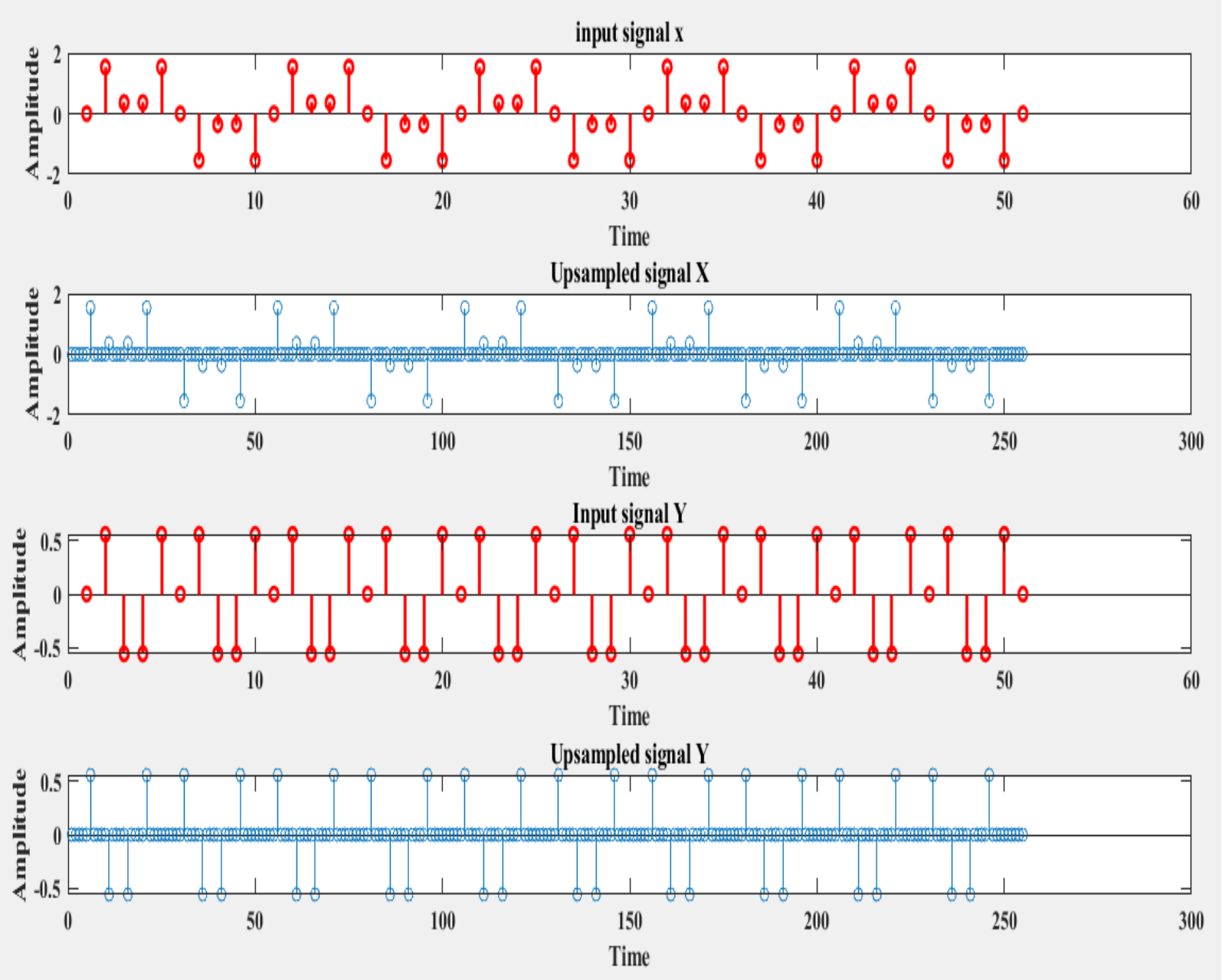

input signal x
Amplitude
Time
Downsampled signal X
Amplitude
Time
Input signal Y
Amplitude
Time
Downsampled signal Y
Amplitude
Time

2.Part (a) for the factor-of-5 up-sampler. Plot the original and up-sampled sequences

MATLAB Code:

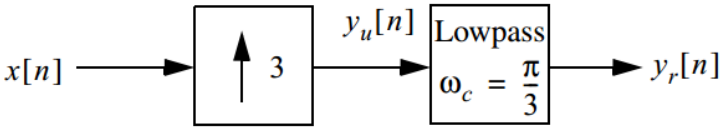
```
%2018-EE-394 LAB08 Task 1b DSP
n = 0:5;
x1 = sin(2*pi*0.1*n);
x2 = sin(2*pi*0.3*n);
x = x1+x2;
y = x1.*x2;
M = 5;
X = upsample(x,M);
Y= upsample(y,M);
subplot(4,1,1);
stem(x)
xlabel('Time');
ylabel('Amplitude');
title('input signal x');
subplot(4,1,2);
stem(X)
xlabel('Time');
ylabel('Amplitude');
title('Upsampled signal X');
subplot(4,1,3);
stem(y)
xlabel('Time');
ylabel('Amplitude');
title('Input signal Y');
xlabel('Time');
subplot(4,1,4);
stem(Y)
xlabel('Time');
ylabel('Amplitude');
title('Upsampled signal Y');
```

Output:


input signal x
Amplitude
Time
Upsampled signal X
Amplitude
Time
Input signal Y
Amplitude
Time
Upsampled signal Y
Amplitude
Time

Task 2

Consider the system below:



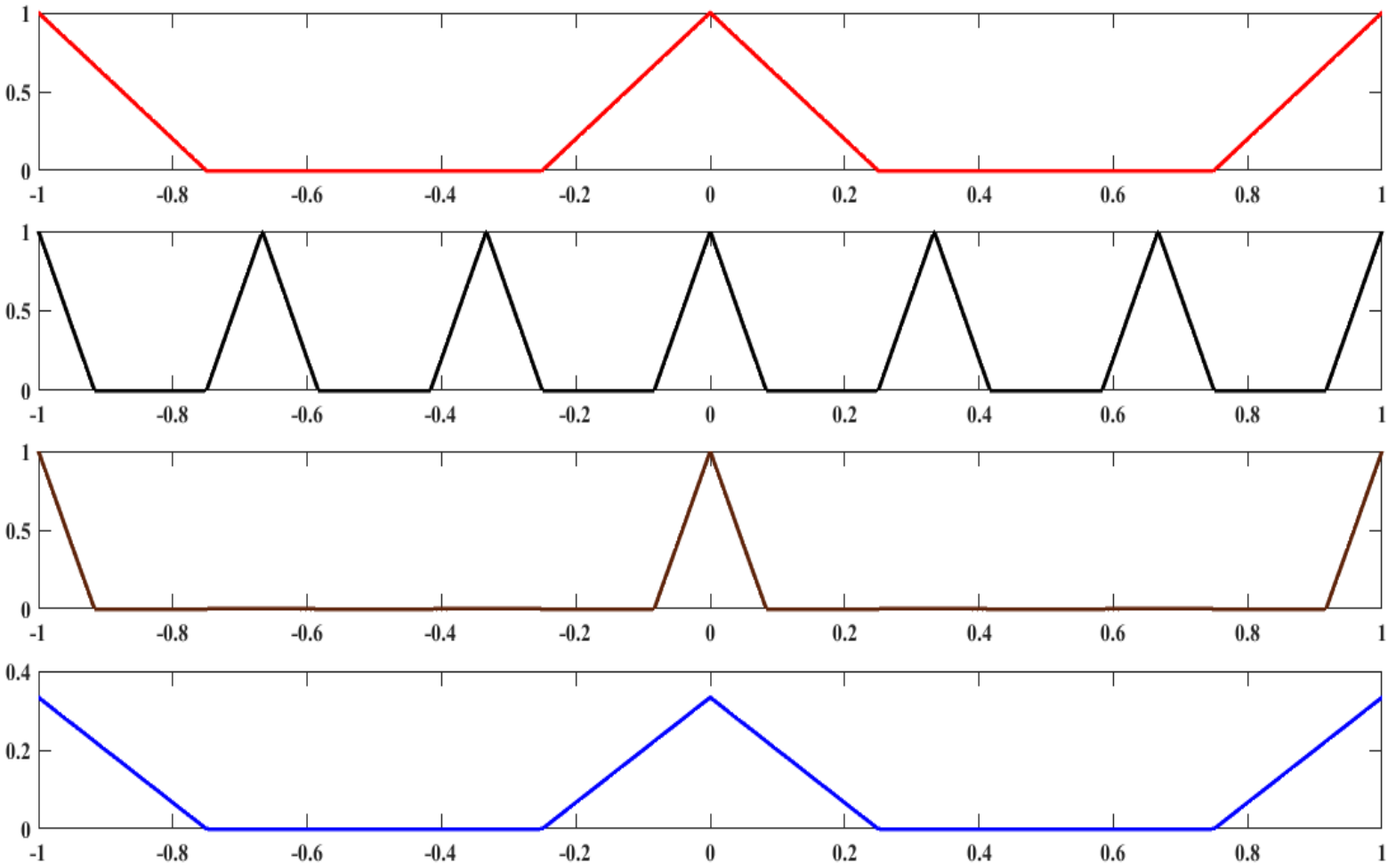
Questions

1. Read the documentation of MAT LAB's command firl. Desing a lowpass FIR filter using the command.
2. Plot the frequency response of the lowpass FIR filter, $y_u[n]$ and output sequence.
3. Modify the above system to get same output as an input.

MATLAB Code:

```
%2018-EE-394 LAB 08 Task 2(1,2)
n = 0:1024;
f = -1:1/512:1;
x = 1/4*sinc(1/4*(n-512)).^2;
n = upsample(x,3);
M = freqz(x,1,2*pi*f);
N= freqz(n,1,2*pi*f);
H = firl(128-1, 1/3);
m = filter(H,1,n);
Mf = freqz(m,1,2*pi*f);
s=downsample(m,3) ;
Ms = freqz(s,1,2*pi*f);
subplot(4,1,1);
plot(f,abs(M))
xlabel('')
subplot(4,1,2);
plot(f,abs(N))
subplot(4,1,3);
plot(f,abs(Mf))
subplot(4,1,4);
plot(f,abs(Ms))
```

Output:



4. **Implement an equivalent polyphase decomposition based decimation system for the system above in MATLAB. How many additions and multiplications are required per unit time?**

Answer: Multiplication=341.3

Addition= 342.3

5. **Do you see any gains in number of computations (i.e. without decimation and decimation)?**

Answer: While we are using the polyphase system, It will allow us to save Significant amount of Multiplications and Additions

Conclusion:

In this lab, we have learnt about up sampling and down sampling of Digital signals. Down sampling is basically way of reducing a sampling rate by an integer factor. Up sampling can describe an entire process of expansion and filtering, after up sampling by a factor of M , the new sampling period becomes T/M.We have also learnt about the phenomenon of multi rate signal processing in which the sampling rate of a signal is changed to increase the efficiency of various signal processing operations. And in the last task we have designed the lowpass FIR filter which removes high frequency signal components from the input

Registration Number: 2018-EE-394

University of Engineering & Technology Lahore

Experiment # 9

Title: **Discrete Fourier Transform (DFT)**

Equipment Required: Personal computer (PC) with windows operating system and MATLAB software

Theory:

The Discrete Fourier Transform (DFT) is one of the most important tools in Digital Signal Processing. Firstly, the DFT can calculate a signal's *frequency spectrum*. This is a direct examination of information encoded in the frequency, phase, and amplitude of the component sinusoids. For example, human speech and hearing use signals with this type of encoding. Secondly, the DFT can find a system's frequency response from the system's impulse response, and vice versa. This allows systems to be analyzed in the *frequency domain*, just as convolution allows systems to be analyzed in the *time domain*. Third, the DFT can be used as an intermediate step in more elaborate signal processing techniques. The classic example of this is *FFT convolution*, an algorithm for convolving signals that is hundreds of times faster than conventional methods.

In Lab Experiment 6, we have discussed in detail the Discrete Time Fourier Transform (DTFT) for the analysis of signals. While DTFT is very useful analytically, it usually cannot be exactly evaluated on a computer because equation requires an infinite sum and evaluation of an integral.

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x(n)e^{-j\omega n}$$

$$x(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(e^{j\omega})e^{j\omega n}$$

The discrete Fourier transform (DFT) is a sampled version of the DTFT, hence it is better suited to numerical evaluation on computers.

Analysis Equation (DFT): $$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}$$

Synthesis Equation (IDFT): $$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2\pi kn/N}$$

Where *X(k)* is an N-point DFT of x[n]. Note that *X(k)* is a function of a discrete integer k, where k ranges from 0 to N-1.

Using the matrix vector multiplication technique used to compute the DTFT, we can calculate the DFT:

$$W_N^{nk} = e^{-j2\pi kn/N}$$

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{nk}$$

DFT Function

```
function [Xk] =dft (xn,N)
%Compute Discrete Fourier
Transform n=[0:1:N-1]; k= [0:1:N-
1];
WN = exp (-j * 2 * pi / N);
nk = n' * k;
WNnk = WN.^ nk;
Xk = xn * WNnk;
```

Inverse DFT Function

```
function [xn] =idft (Xk, N)
%Compute Inverse Discrete Transform
n= [0:1:N-1];
k = [0:1:N-1];
WN = exp (-j * 2 * pi / N);
nk = n' * k;
WNnk = WN .^ (-nk) ;
xn = (Xk * WNnk)/N;
```

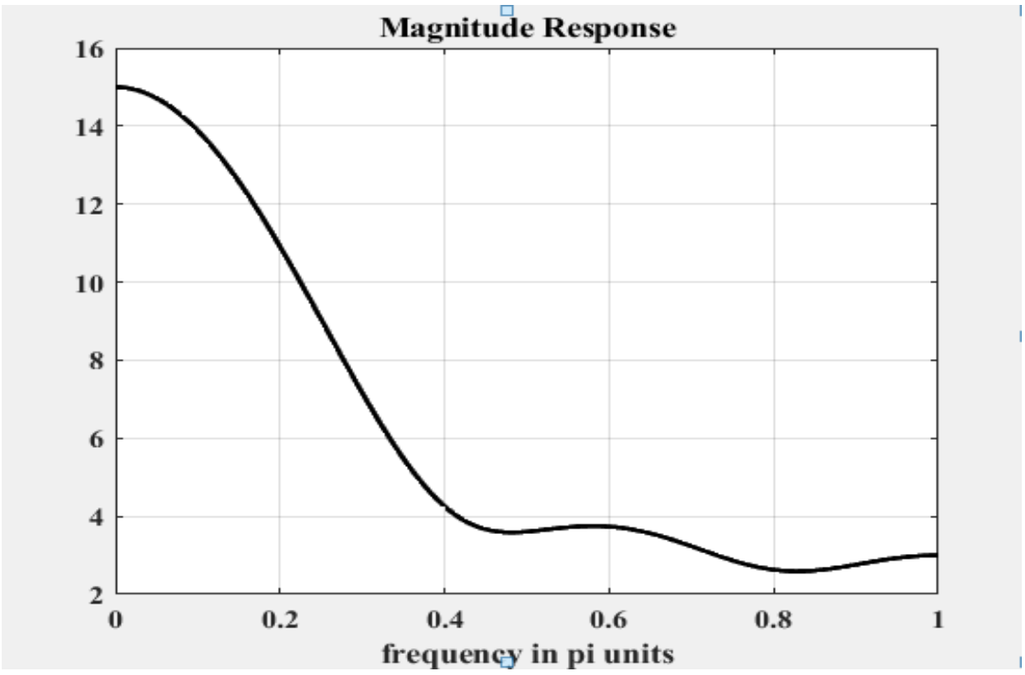
Example 1:

Let x(n)= [1 2 3 4 5]. Evaluate and plot magnitude of DTFT, DFT and IDFT.

MATLAB Code for DTFT

```
%2018-EE-394
%LAB 9 EX#1 DTFT
n = -1:3; x = 1:5;
k = 0:500; w =(pi/500)*k;
X = x * (exp(-j*pi/500)) .^ (n'*k);
magX = abs(X); angX = angle(X);
realX = real(X); imagX = imag(X);
plot(k/500,magX);grid
xlabel('frequency in pi units');
title('Magnitude Response');
```

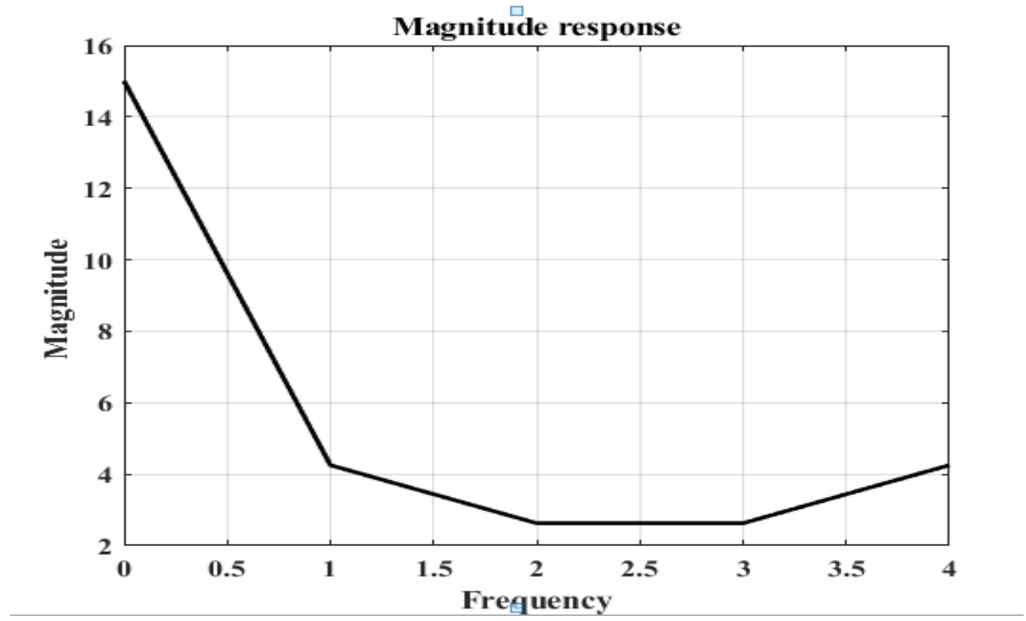
Output :



MATLAB Code for DFT:

```
%2018-EE-394
%LAB#9 EX#1 DFT
x=[1,2,3,4,5];
N= 5;
k= [0:1:N-1];
X=dft(x,N);
mag= abs(X);
magX = abs(X); angX = angle(X);
realX = real(X); imagX = imag(X);
plot(k,magX);grid on
xlabel('k')
title('Magnitude response')
xlabel('Frequency')
ylabel('Magnitude ')
```

Output :



Zero-Padding:

It is an operation in which more zeros are appended to the original sequence. The resulting longer

DFT provides closely spaced samples of the discrete time Fourier transform of the original sequence. In MATLAB zero padding is implemented using the zeros function. **The zero padding gives high-density spectrum and provides a better displayed version for plotting.** But it does not give a high resolution spectrum because no new information is added to the signal: only additional zeros are added in the data.

Example 2:

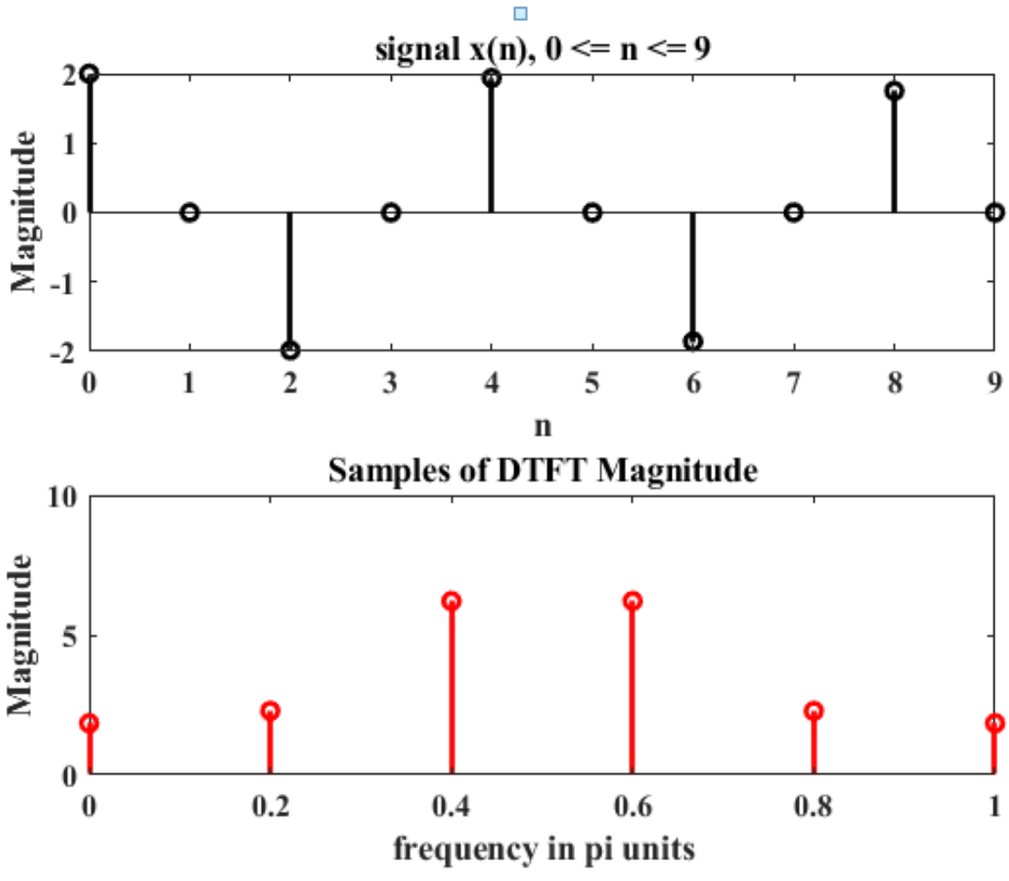
Determine and plot x(n) and its discrete time Fourier transform for appropriate value of n.

$$x(n) = cos(0.48\pi n) + cos(0.52\pi n)$$

a) Spectrum based on the first 10 samples of x(n)

```
%2018-EE-394
%LAB#9 EX#2
n1=[0:1:9];
x=cos(0.48*pi*n1) + cos(0.52*pi*n1)
y1 =x( 1:1:10);
subplot(2,1, 1);
stem(n1,y1);
title('signal x(n), 0 <= n <= 9');
xlabel('n')
ylabel('Magnitude')
Y1=dft(y1, 10);
magY1=abs(Y1( 1:1:6));
k1=0:1:5; w1 =2*pi/10*k1; subplot(2,1 ,2);
stem(w1/pi,magY1 );
title('Samples of DTFT Magnitude');
xlabel('frequency in pi units')
ylabel('Magnitude')
axis([0,1 ,0, 10])
```

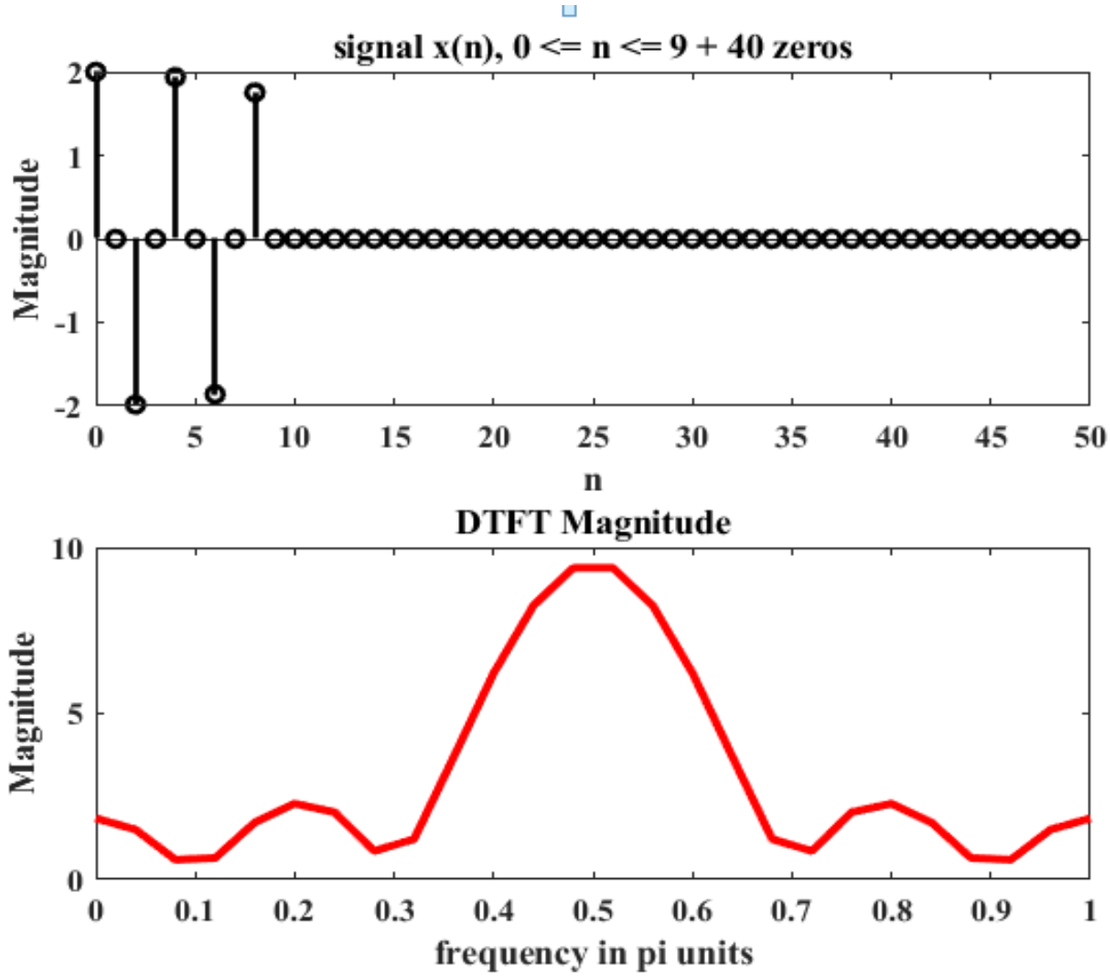
Output :



b) High density Spectrum (50 samples) based on the first 10 samples of x(n)

```
%2018-EE-394
%LAB#9 EX# 2(b)
n1=[0:1:49];
x=cos(0.48*pi*n1) + cos(0.52*pi*n1);
y2=[x(1: 1:10) zeros(1,40)];
subplot(2,1, 1);
stem(n1,y2);
title('signal x(n), 0 <= n <= 9 + 40 zeros');
xlabel('n')
ylabel('Magnitude')
Y2=fft(y2);
magY2=abs(Y2(1:1:26));
k2=0: 1:25;w2=2*pi/50*k2; subplot(2,1 ,2);
plot(w2/pi,magY2);
title('DTFT Magnitude');
xlabel('frequency in pi units')
ylabel('Magnitude')
axis([0,1,0,10])
```

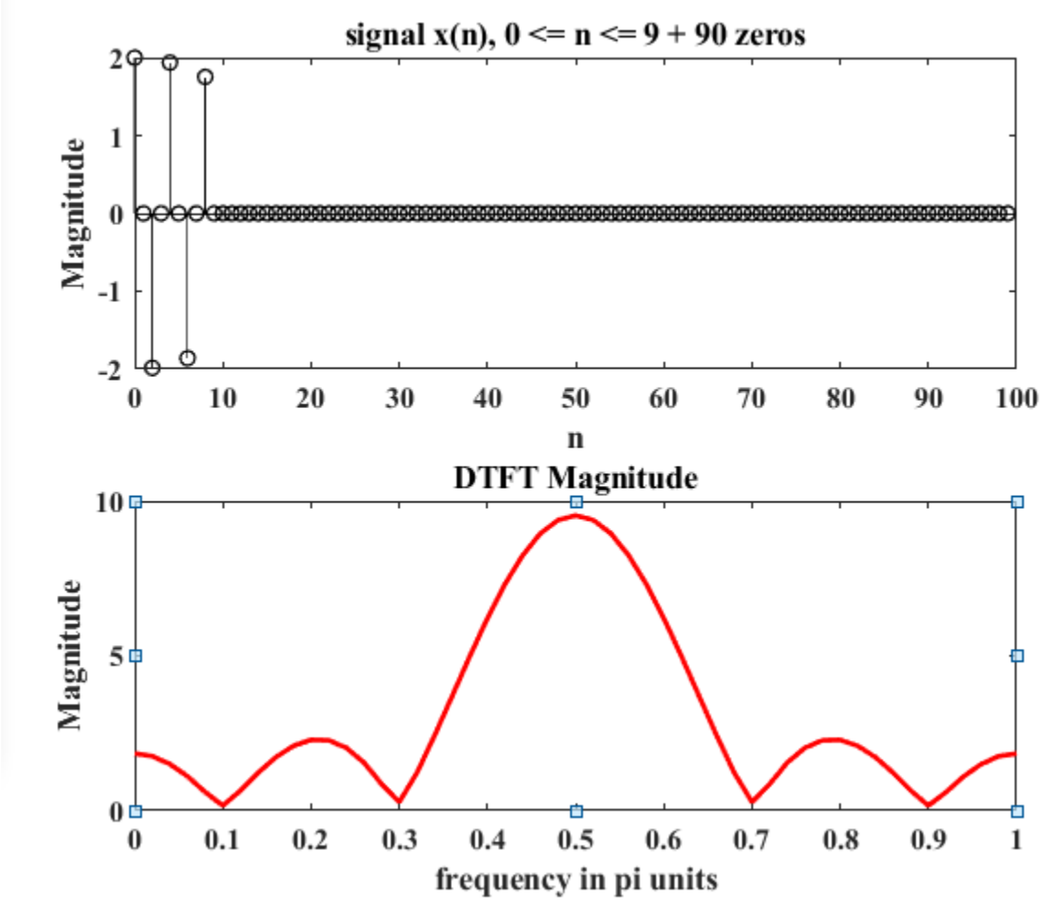
Output ;



c) High density Spectrum (100 samples) based on the first 10 samples of x(n)

```
%2018-EE-394
%LAB#9 EX#2(c)
nl=[0: 1:99];
x=cos(0.48*pi*nl) + cos(0.52*pi*nl);
y3=[ x(1: 1:10) zeros( 1,90)];
subplot(2,1, 1);
stem(nl,y3);
title('signal x(n), 0 <= n <= 9 + 90 zeros');
xlabel('n')
ylabel('Magnitude')
Y3=dft(y3, 100);magY3=abs(Y3(1: 1:51));
k3=0:1:50;w3=2*pi/100*k3;
subplot(2,1 ,2);
plot(w3/pi,magY3);
title('DTFT Magnitude');
xlabel('frequency in pi units')
ylabel('Magnitude')
axis([0, 1,0, 10])
```

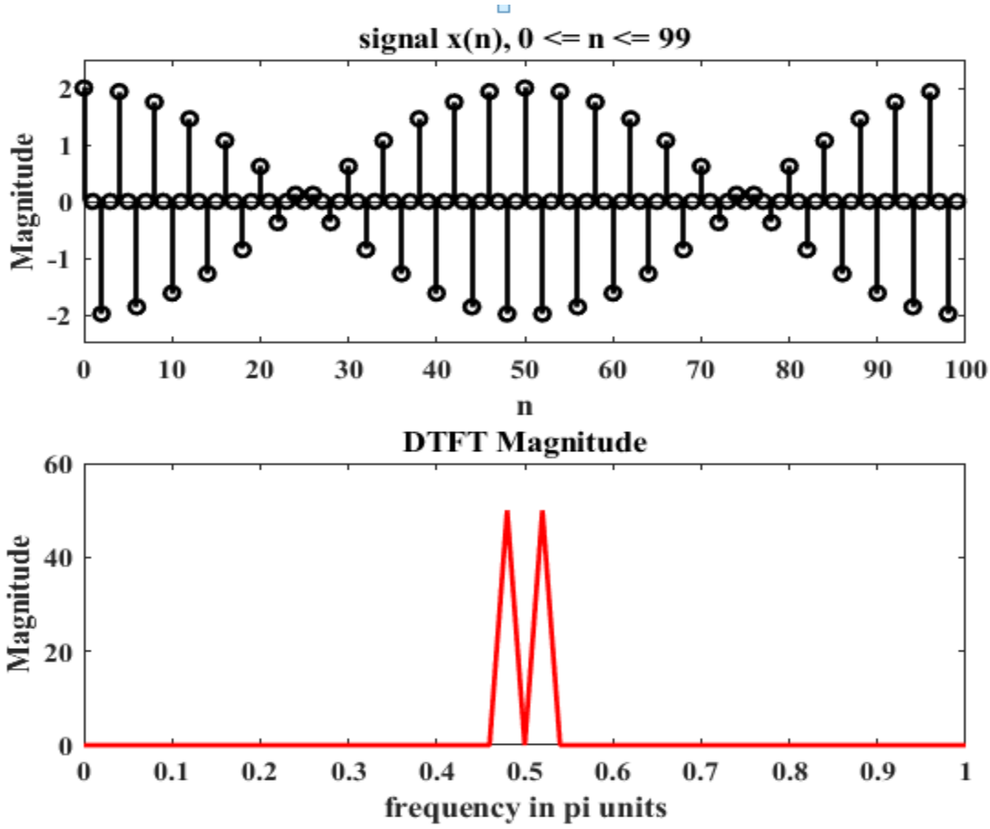
Output :



d) High resolution spectrum based on 100 samples of the signal x(n)

```
%2018-EE-394
%LAB#9 EX#2(d)
n=[0: 1:99];
x=cos(0.48*pi*n) + cos(0.52*pi*n);
subplot(2,1, 1);
stem(n,x);
title('signal x(n), 0 <= n <= 99');
xlabel('n')
ylabel('Magnitude')
axis([0,100,-2.5,2.5])
X=dft(x,100);
magX=abs(X(1:1:51));
k=0:1:50;
W=2*pi/100*k;
subplot(2,1 ,2);
plot(W/pi,magX);
title('DTFT Magnitude');
xlabel('frequency in pi units')
ylabel('Magnitude')
axis([0, 1,0, 60])
```

Output :



Write a note on what you have learnt from this example.

We've discovered that adding extra zeros to the original sequence results in closely spaced samples of the DTFT of the original sequence using DFT. We obtain a higher density spectrum when we increase the number of appended zeros, and when we plot it, we get a better signal. However, we do not obtain a high resolution spectrum since no new information is added to the signal; just more zeros are appended to the original data. Furthermore, when the density increases after adding zeros, it is simple to retrieve the signal.

Fast Fourier Transform:

Discrete Fourier Transform (DFT) is discrete version of FT which transforms a signal (discrete sequence) from Time Domain representation to it's Frequency Domain representation, while Fast

Fourier Transform (FFT) is an efficient algorithm for calculation of DFT. Computing a DFT of N points using just it's definition, takes Θ(N^2) time, while an FFT can compute the same result in only Θ(N log N) steps, which's quite substantial gain for large sequences. In MATLAB "fft" command is used for fast Fourier transform.

Example 3:

Solve example 2 using Fast Fourier transform

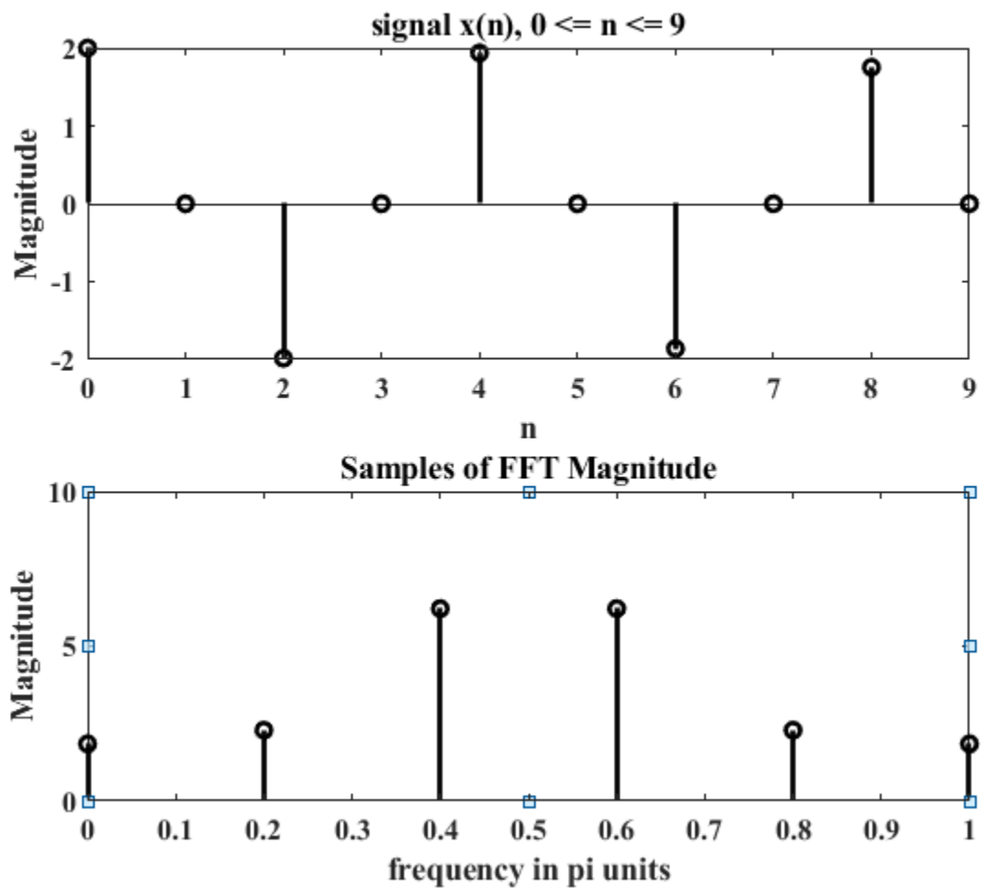
Determine and plot x(n) and its discrete time Fourier transform for appropriate value of

$$x(n) = cos(0.48\pi n) + cos(0.52\pi n)$$

a) Spectrum based on the first 10 samples of x(n)

```
%2018-EE-394
%LAB#9 EX# 3(a)
nl=[0:1:9];
x=cos(0.48*pi*nl)+cos(0.52*pi*nl);
subplot(2,1, 1);
stem(nl,x);
ylabel('Magnitude')
title('signal x(n), 0 <= n <= 9');
xlabel('n')
xl=fft(x);
magYl=abs(xl( 1:1:6));
kl=0:1:5;wl =2*pi/10*kl;
subplot(2,1 ,2);
stem(wl/pi,magYl )
title('Samples of FFT Magnitude');
xlabel('frequency in pi units')
ylabel('Magnitude')
axis([0,1 ,0, 10])
```

Output :

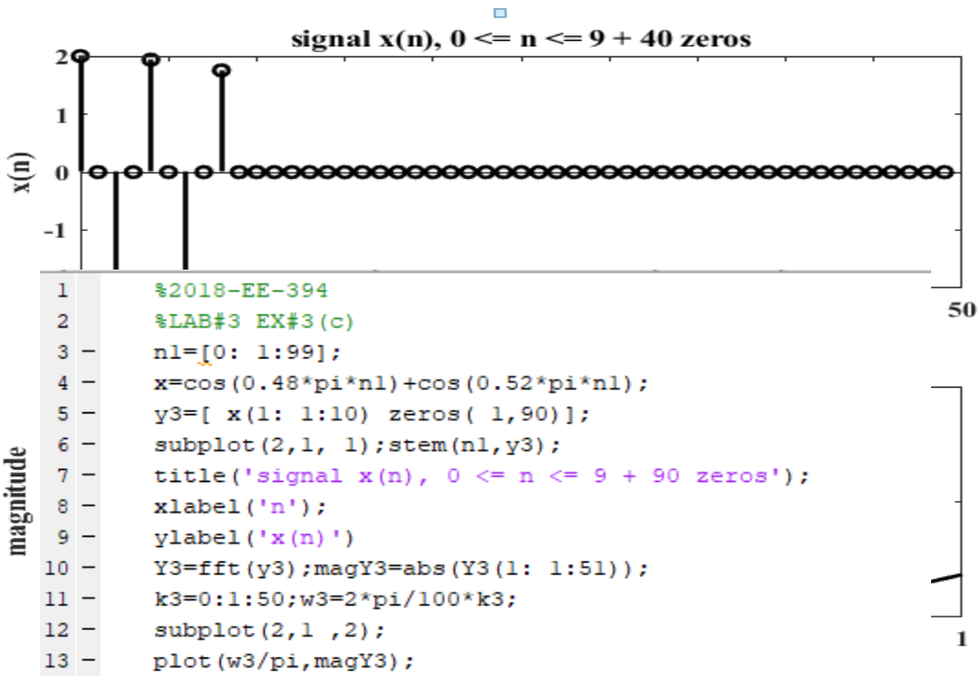


b) High density Spectrum (50 samples) based on the first 10 samples of x(n)

Output

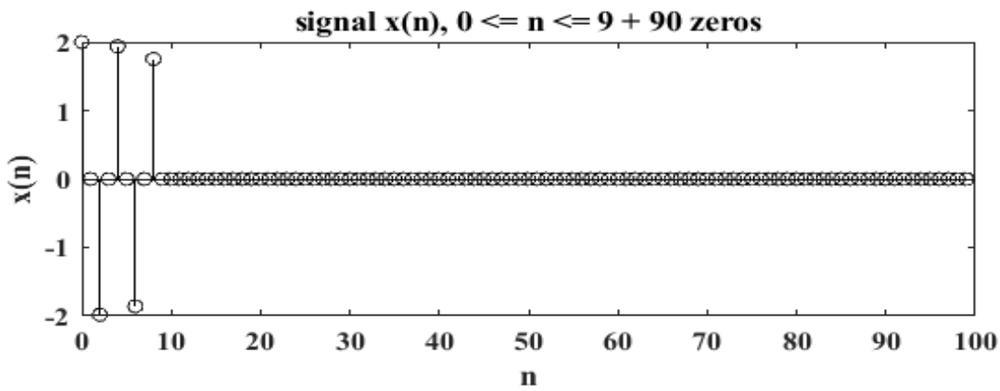
```
%2018-EE-394
% LAB#3 EX#3(b)
nl=[0:1:49];
x=cos(0.48*pi*nl)+cos(0.52*pi*nl);
y=[x(1:1:10) zeros(1,40)];
subplot(2,1, 1);stem(nl,y);
title('signal x(n), 0 <= n <= 9 + 40 zeros');
xlabel('n')
ylabel('x(n)')
Y2=fft(y);magY2=abs(Y2(1:1:26));
k2=0: 1:25;w2=2*pi/50*k2; subplot(2,1 ,2);
plot(w2/pi,magY2);
title('FFT Magnitude');
xlabel('frequency in pi units')
ylabel('magnitude')
axis([0,1,0,10])
```

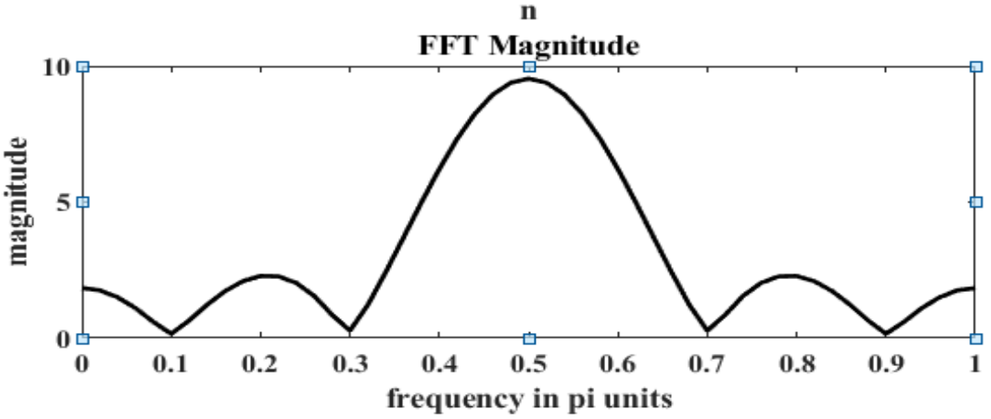
c) High density Spectrum (100 samples) based on the first 10 samples of x(n)




```
%2018-EE-394
%LAB#3 EX#3(c)
nl=[0: 1:99];
x=cos(0.48*pi*nl)+cos(0.52*pi*nl);
y3=[ x(1: 1:10) zeros( 1,90)];
subplot(2,1, 1);stem(nl,y3);
title('signal x(n), 0 <= n <= 9 + 90 zeros');
xlabel('n');
ylabel('x(n)')
Y3=fft(y3);magY3=abs(Y3(1: 1:51));
k3=0:1:50;w3=2*pi/100*k3;
subplot(2,1 ,2);
plot(w3/pi,magY3);
title('FFT Magnitude');
xlabel('frequency in pi units');
ylabel('magnitude');
axis([0, 1,0, 10])
```

Output :

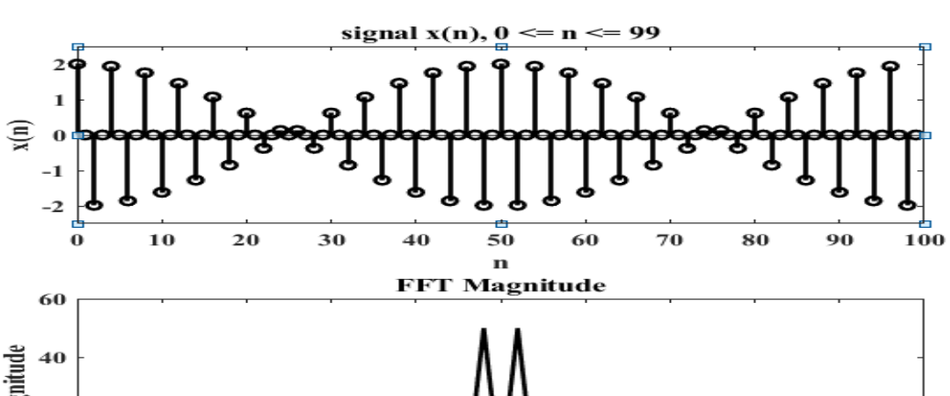


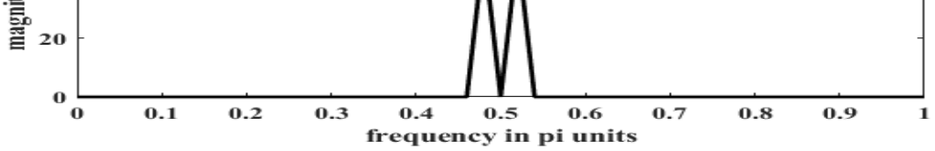


d) High resolution spectrum based on 100 samples of the signal x(n)

```
%2018-EE-394
%LAB#3 EX#3(d)
n=[0:1:99];
x=cos(0.48*pi*n)+cos(0.52*pi*n);
subplot(2,1,1);stem(n,x);
title('signal x(n), 0 <= n <= 99');
xlabel('n')
ylabel('x(n)')
axis([0,100,-2.5,2.5])
X=fft(x);
magX=abs(X(1: 1:51));
k=0:1:50;w=2*pi/100*k;
subplot(2,1,2);
plot(w/pi,magX);
title('FFT Magnitude');
xlabel('frequency in pi units');
ylabel('magnitude')
axis([0,1,0,60])
```

Output :





Properties:

Linearity: The DFT is a linear transform

$$a_1x_1(n) + a_2x_2(n) \overset{\text{DFT}}{\underset{N}{\longleftrightarrow}} a_1X_1(k) + a_2X_2(k)$$

If $x_1(n)$ and $x_2(n)$ have N_1-points and N_2 -points respectively, then $N_3 = \max(N_1, N_2)$

Circular Folding: If an N point sequence is folded, then the result x(-n) would not be an N point sequence, and it would not be possible to compute its DFT. Therefore, we use modulo-N operation on the argument(-n) and define the folding by

$$x((-n))_N = \begin{cases} x(0), & n = 0 \\ x(N-n), & 1 \le n \le N-1 \end{cases}$$

This a called a circular folding. To visualize it, imagine that the sequence x(n) is wrapped around a circle in the counterclockwise direction that n= 0 and n=N overlap. Then $x((-n))_N$ can be viewed as a clockwise wrapping of x(n) around the circle. That's why it is called circular folding.
In MATLAB the circular folding can be achieved by $x = x(mod(-n, N) + 1$.

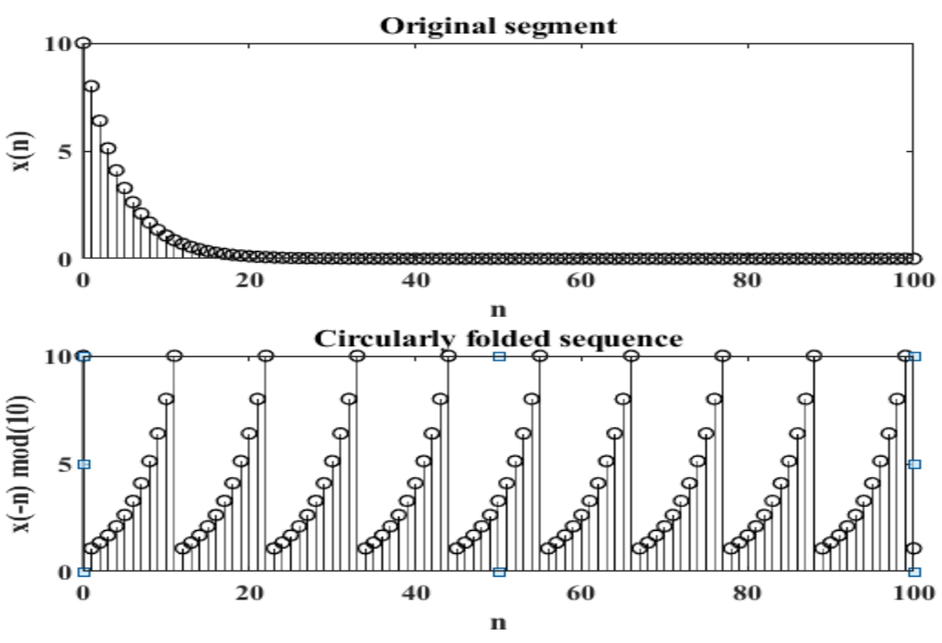
Example 4: Let $x(n) = 10(0.8)^n \ \ 0 \le n \le 10$

a. Determine and plot $x((-n))_{11}$
b. Verify the circular folding property

A,

```
%2018-EE-394
% LAB#3 EX#4
n=0:100;
x= 10*(0.8).^n;
y=x(mod(-n,11)+1);
subplot(2,1,1);
stem(n,x);
title('Original segment');
xlabel('n')
ylabel('x(n)')
subplot(2,1,2);
stem(n, y);
title('Circularly folded sequence');
xlabel('n');
ylabel('x(-n) mod(10)');
```

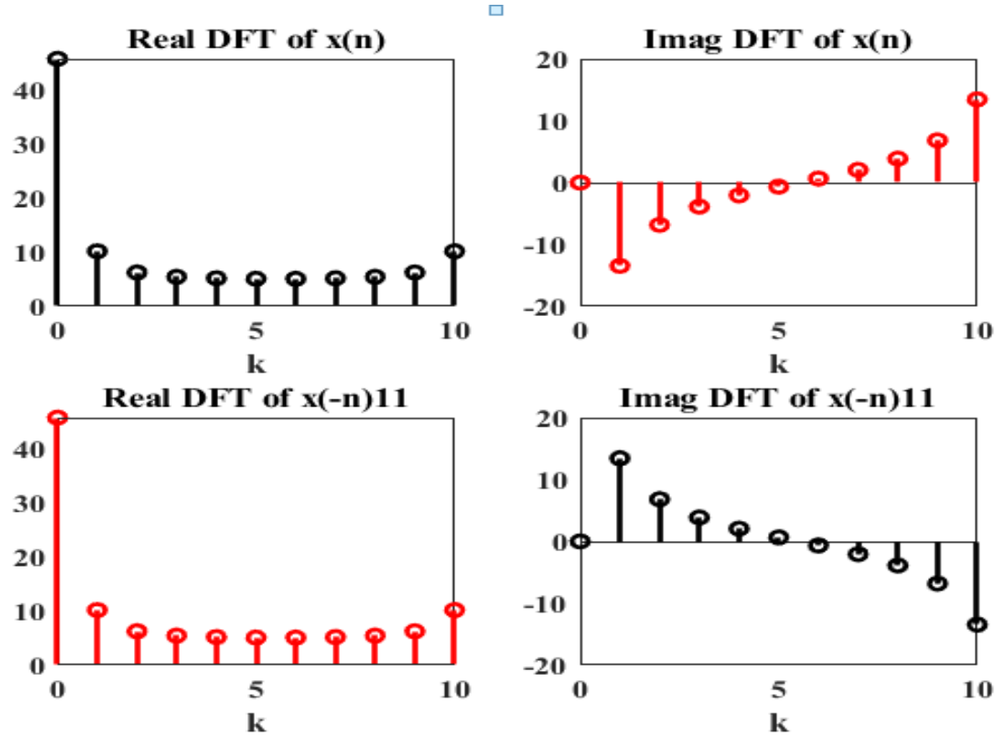
Output :



b.

```
%2018-EE-394
% LAB#3 EX#4(b)
n=0:10;
x= 10*(0.8).^n; X= fft(x,11);
Y=fft(x(mod(-n,11)+1), 11);
subplot(2,2,1); stem(n, real(X));
title('Real {DFT of x(n)}'); xlabel('k')
subplot(2,2,2); stem(n, imag(X));
title('Imag {DFT of x(n)}'); xlabel('k')
subplot(2,2,3); stem(n, real(Y));
title('Real {DFT of x(-n)11}'); xlabel('k')
subplot(2,2,4); stem(n, imag(Y));
title('Imag {DFT of x(-n)11}'); xlabel('k')
```

Output :



Circular Shift property: If an N-point sequence is shifted in either direction, then the result is no longer between 0 ≤n ≤ N-1. Therefore, we first convert *x(n)* into its periodic extension, and then *shift* it by *m* samples to obtain

$$\tilde{x}(n-m) = x((n-m))_N$$

This is called periodic shift. then this periodic shift is converted into N point sequence. Resulting sequence is

$$\tilde{x}(n-m)\mathcal{R}_N(n) = x((n-m))_N\,\mathcal{R}_N(n)$$

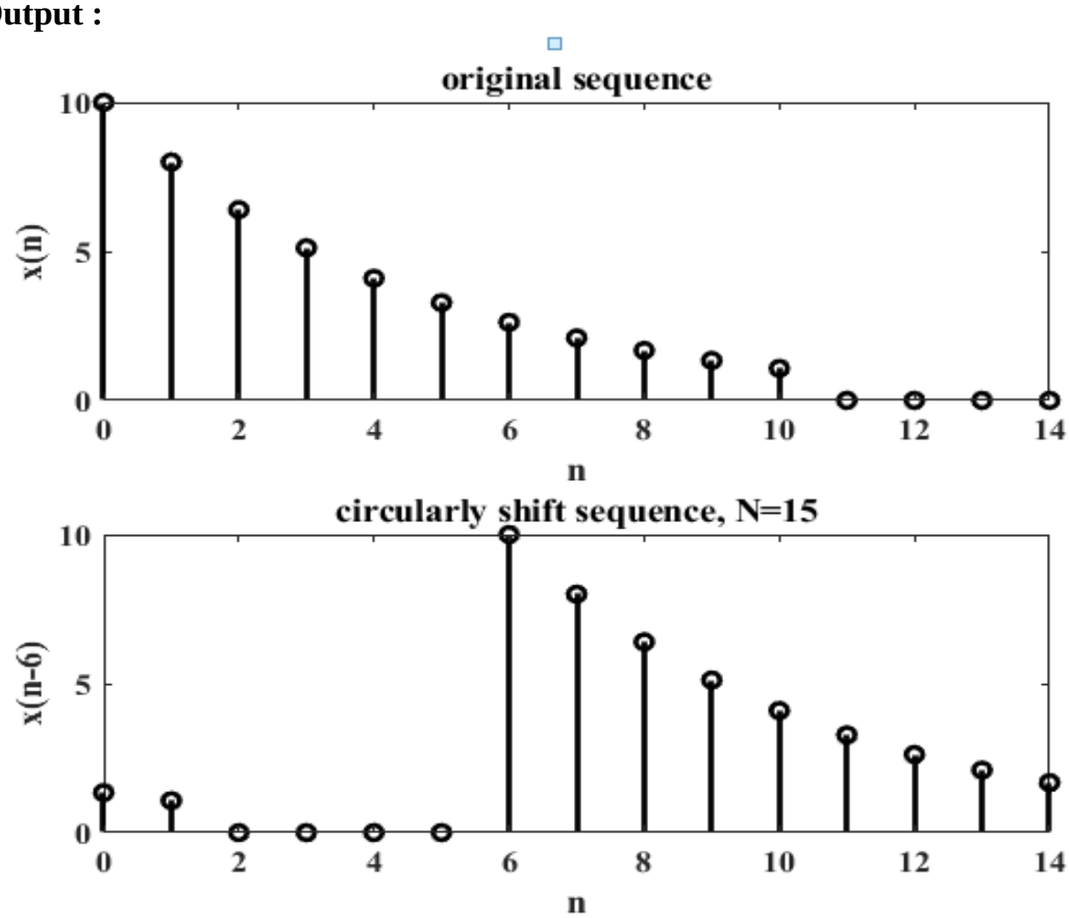
Example 5: Let $x(n) = 10(0.8)n \ \ 0 \le n \le 10$. Determine and plot $x((n-6))_{15}$

MATLAB Code:

```
%2018-EE-394
%LAB#9 EX# 5
function y = cirshftt(x,m,N)
% Check for length of x
if length(x) > N
error('N must be >= the length of X
end
X = [x zeros(1, N-length(x))];
n = [0:1:N-1];
n = mod(n-m,N) ;
y = X(n+1)
```

```
%2018-EE-394
%LAB#9 EX# 5(a)
n=0:10;
x=10*(0.8).^n;
y= cirshftt(x, 6,15);
n= 0:14;
x= [x, zeros(1,4)];
subplot(2,1,1); stem(n,x);title('original sequence')
xlabel('n'); ylabel('x(n)');
subplot(2,1,2); stem(n,y); title('circularly shift sequence, N=15')
xlabel('n'); ylabel('x(n-6)')
```

Output :



Tasks:

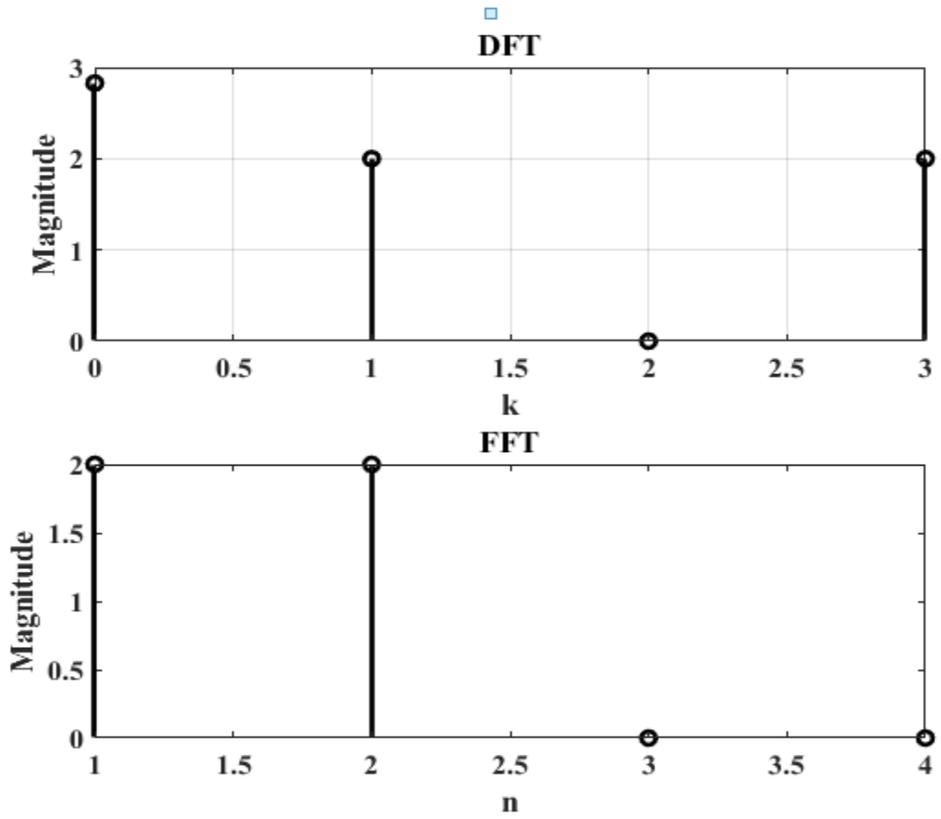
1. Determine the DFT and IDFT of following signals.

   a. $x(n) = \{1, j, j, 1\} \;\; N = 4$

Code:

```
%2018-EE-394
% LAB# 9 task#1
x=[1,j,j,1];
N= 4;
k= [0:1:N-1];
X_1=dft(x,N); mag= abs(X_1);
magX = abs(X_1); angX = angle(X_1);
realX = real(X_1);imagX = imag(X_1);
subplot(2,1,1)
stem(k,magX)
grid
xlabel('k'); ylabel('Magnitude');
title('DFT')
X_2=fft(x); mag= abs(X_2);
magX = abs(X_2); angX = angle(X_2);
realX = real(X_2); imagX = imag(X_2);
subplot(2,1,2); stem(X_2)
xlabel('n'); ylabel('Magnitude');
title('FFT')
```

Output :

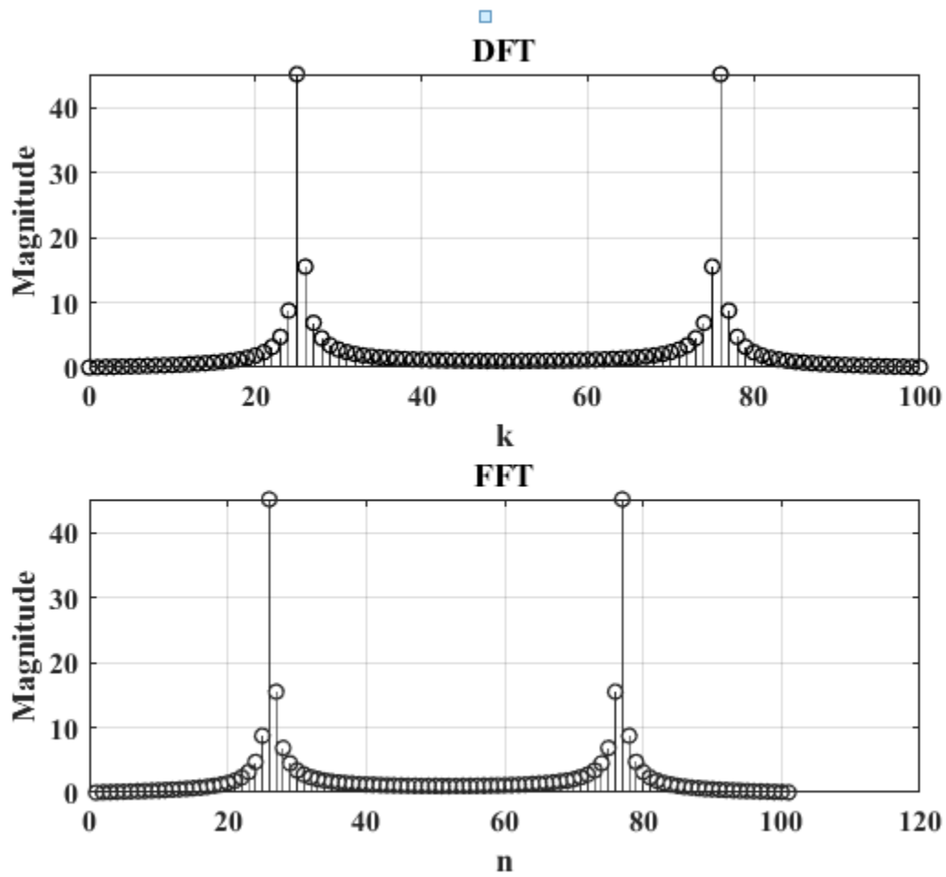


$$b.\ x(n) = \{Sin(0.5\pi n)\}\ \ 0 \le n \le 100$$

Code:

```
%2018-EE-394
%lab 9 % task 1(b)
n=0:1:100;
x=sin(0.5*pi*n);
N= 101;
k= [0:1:N-1];
X_1=dft(x,N);
X_2=abs(fft(x));
mag= abs(X_1);
magX = abs(X_1); angX = angle(X_1);
realX = real(X_1); imagX = imag(X_1);
subplot(2,1,1)
stem(k,magX);grid
xlabel('k'); ylabel('Magnitude');
title('DFT');
subplot(2,1,2)
stem(X_2);grid
xlabel('n'); ylabel('Magnitude');
title('FFT')
```

Output :



Task#2

2. **Let $x(n)$ be a 4 point sequence**

$$x(n) = \begin{cases} 1 & 0 \le n \le 3 \\ 0 & otherwise \end{cases}$$

a. ***Compute samples of DFT for***

$$n = 10,$$

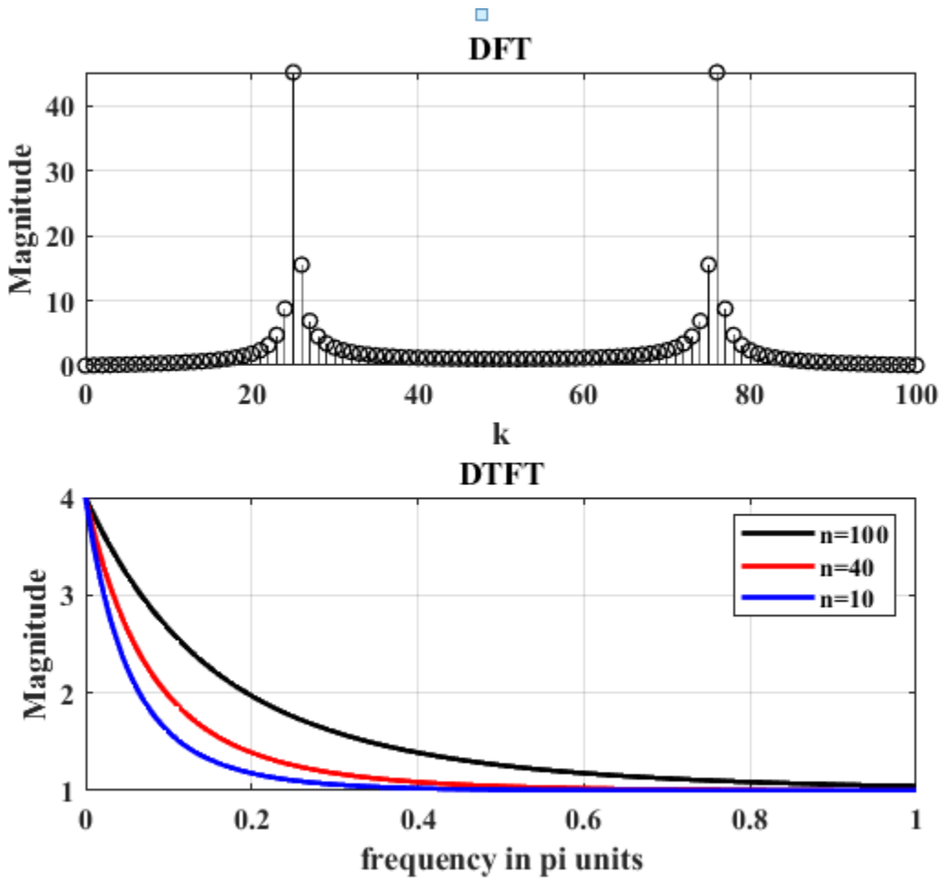
$$n = 40,$$

$$n = 100.$$

Code:

```
%2018-EE-394
%LAB #9 task #2
n =[10 40 100] ;
for j=1:length(n)
if n(1,j)==4
x=[1 1 1 1];
else
x=[1 1 1 1 zeros(1,(n(1,j)-4))];
end
m=0:n(1,j)-1;
k = 0:500;
w =(pi/500)*k;
X = x*(exp(-j*pi/500)) .^ (m'*k);
magX = abs(X); angX = angle(X);
realX = real(X); imagX = imag(X);
plot(k/500,magX);grid; hold on;
xlabel('frequency in pi units');
ylabel('Magnitude');
title('DTFT')
end
legend('n=100','n=40','n=10','n=4')
```

Output :



3. **The first five values of the 8-point DFT of a real-valued sequence x*(n)* are given by**

$$\{0.25, 0.125 - j0.3, 0, 0.125 - j0.06, 0.5\}$$

Determine the DFT of

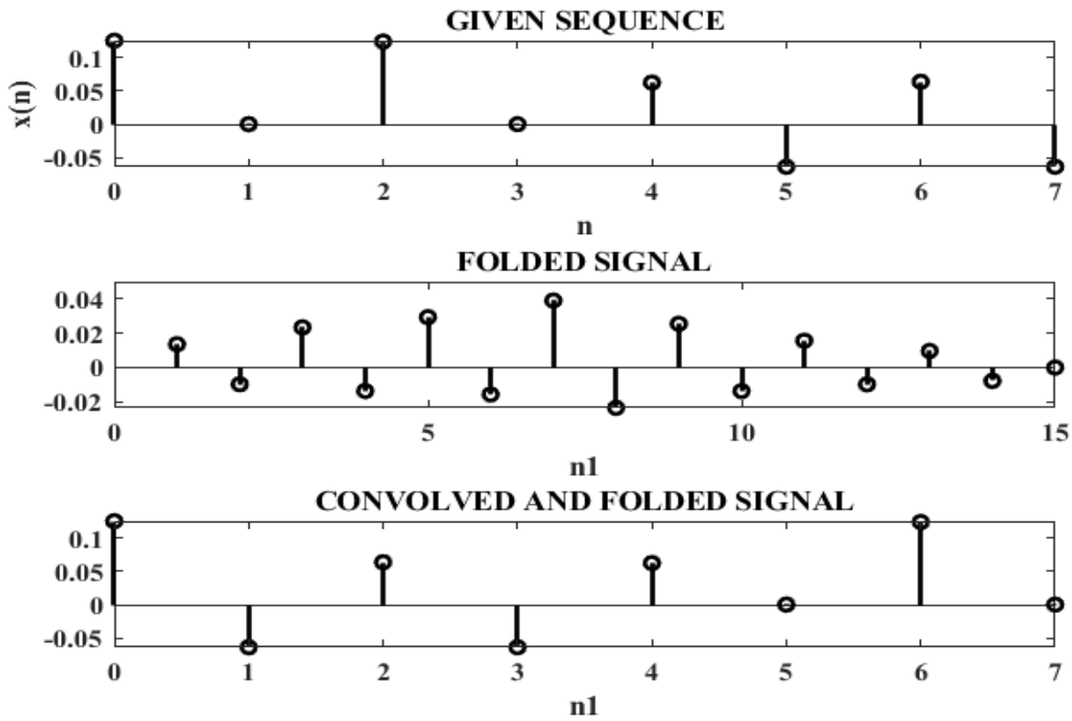
a. $x(n) = x((n+5))10$

b. $y(n) = circularconvolve\ x(n)\ and\ x((-n))_8\ with\ n = 8$,

Code:

```
%2018-EE-394
%LAB #9 task #3
N=8;
k= [0:1:N-1];
n= [0:1:N-1];
Xk=[0.25,0.125-0.3*j,0,0.125-0.06*j,0.5,0,0,0];
xn=idft(Xk,N);
N1=8;
n1 =[0:1:N1-1];
y=xn(mod(-n,8)+1);
Y1=cconv(xn,y);
subplot(3,1,1);
stem(n,xn);
title('GIVEN SEQUENCE')
xlabel('n'); ylabel('x(n)');
subplot(3,1,2);
stem(Y1);
title('FOLDED SIGNAL') ; xlabel('n1');
 subplot(3,1,3);
 stem(n,y);
title('CONVOLVED AND FOLDED SIGNAL')
xlabel('n1');
```

Output :



Conclusion:

In this lab, we have observed three different approaches as DTFT, DFT, and FFT analyses. The Fourier transform is used to sample continuous time domain signals in DTFT. DFT and FFT are methods for sampling DTFT signals in the discrete frequency domain. As the discrete Fourier transform or DFT is the transform that deals with a finite discrete-time signal and a finite or discrete number of frequencies. The FFT approach is used, because it is more faster and more efficient than DFT. However, both DFT and FFT methods sample the signal in frequency domains and provide the same output in the end.

Registration#____2018-EE-394_

University of Engineering & Technology Lahore

Experiment # 10

Title: **Designing of FIR Filters**

Equipment Required: Personal computer (PC) with windows operating system and MATLAB software

Theory:

In signal processing, a finite impulse response (FIR) filter is a filter whose impulse response (or response to any finite length input) is of *finite* duration, because it settles to zero in finite time. FIR filters can be discrete-time or continuous-time, and digital or analog. For a causal discrete time FIR filter of order *N*, each value of the output sequence is a weighted sum of the most recent input values.

$$\begin{aligned} y[n] &= b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N] \\ &= \sum_{i=0}^{N} b_i \cdot x[n-i], \end{aligned}$$

Where,

- x[n] is the input signal,
- y[n] is the output signal,
- N is the filter order; an Nth-order filter has (N+1) terms on the right-hand side
- b_i is the value of the impulse response at the *i*'th instant for $0 < i < N$ of an Nth-order FIR filter. If the filter is a direct form FIR filter then b_i is also a coefficient of the filter.

This computation is also known as discrete convolution.

The frequency response $H_{2\pi}(\omega)$ can also be written as $H(e^{j\omega})$ where function H is the Z transform of the impulse response:

$$H(z) \stackrel{\text{def}}{=} \sum^{\infty} h[n] \cdot z^{-n}.$$

One cycle of the periodic frequency response can be found in the region defined by $z = e^{j\omega}$, $-\pi < \omega < \pi$, which is the unit circle of the z-plane.

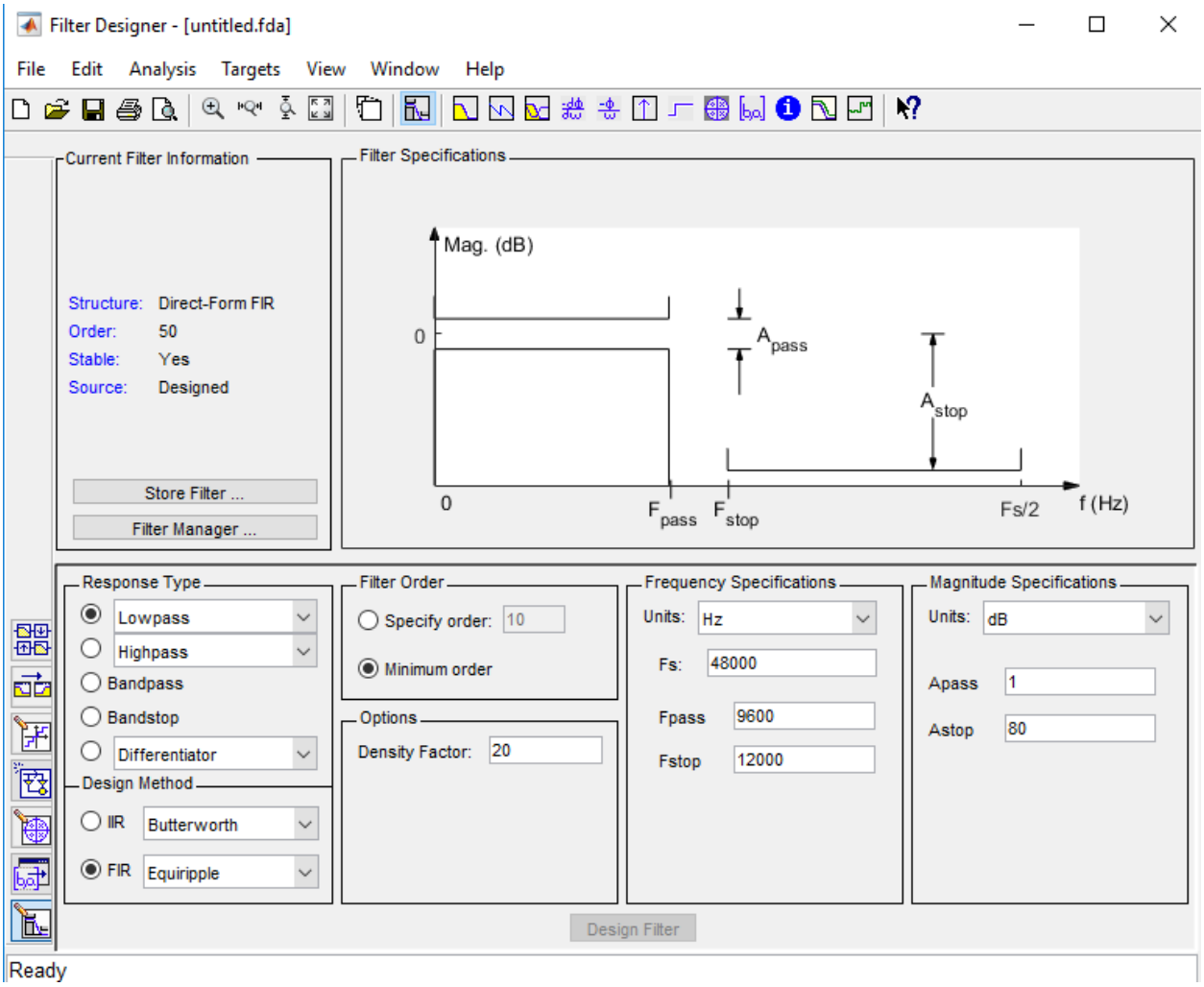
Design of FIR filters using MATLAB Commands:

Filters can be designed by using

- filterDesigner
- filterBuilder
- Filter Visualization Tool

filterDesigner

Write “filterDesigner” on command window

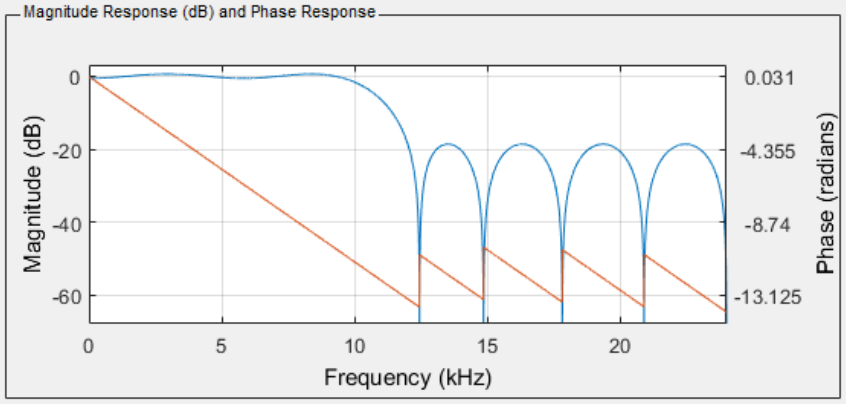


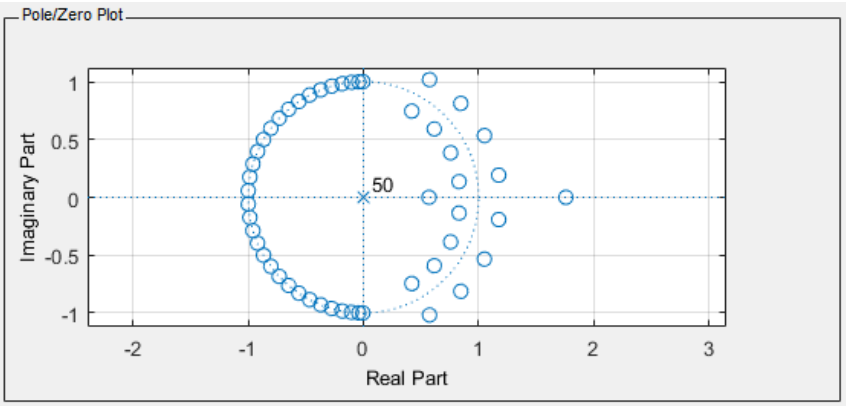
Example:

Design a low pass, high pass, band stop and band pass equiripple FIR filter of minimum order. Plot magnitude response and phase response and pole zero spectrum.

Specifications:

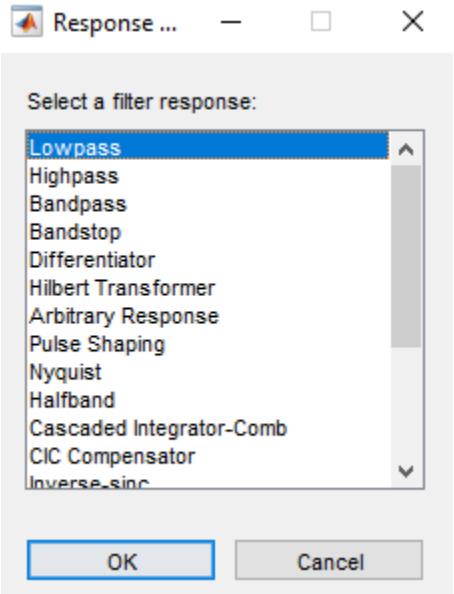
F_s = 48 kHz, F_{pass} = 9600 Hz, F_{stop} = 12 kHz





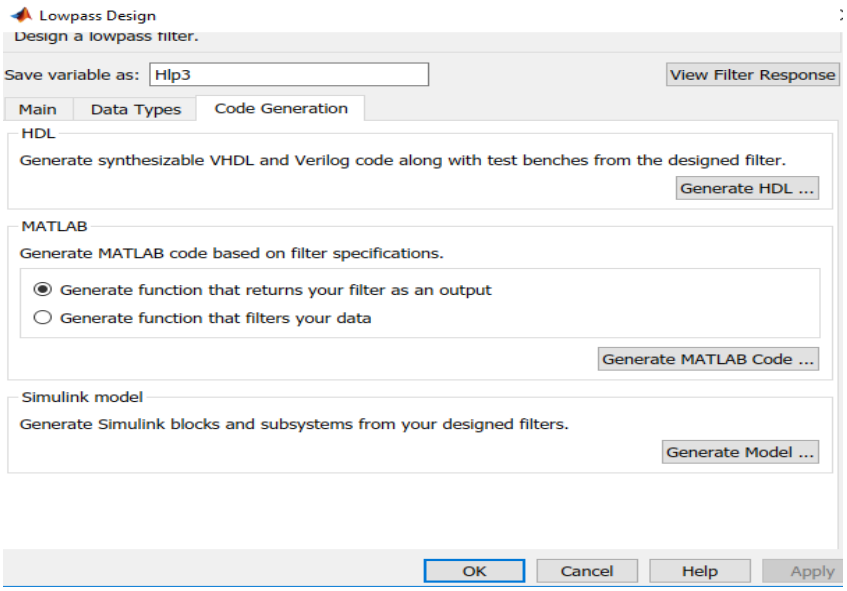
filterBuilder

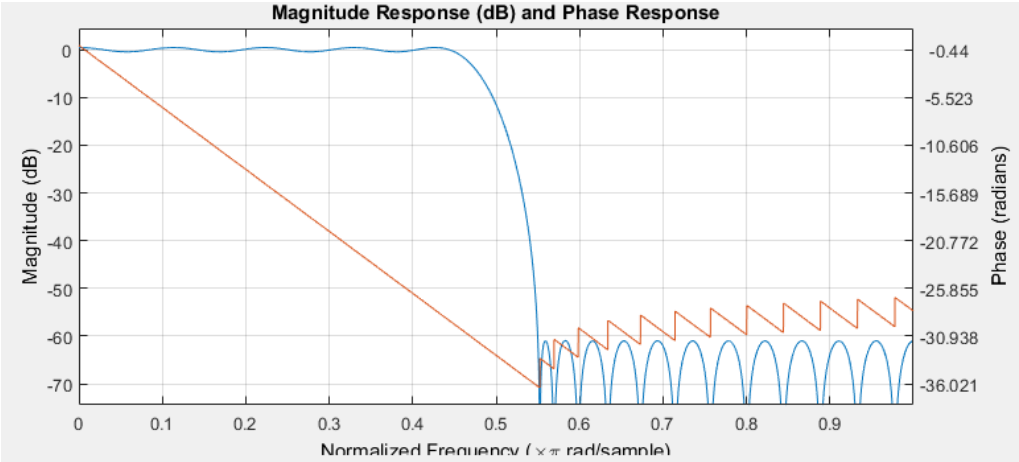
Write “filterbuilder” on command window

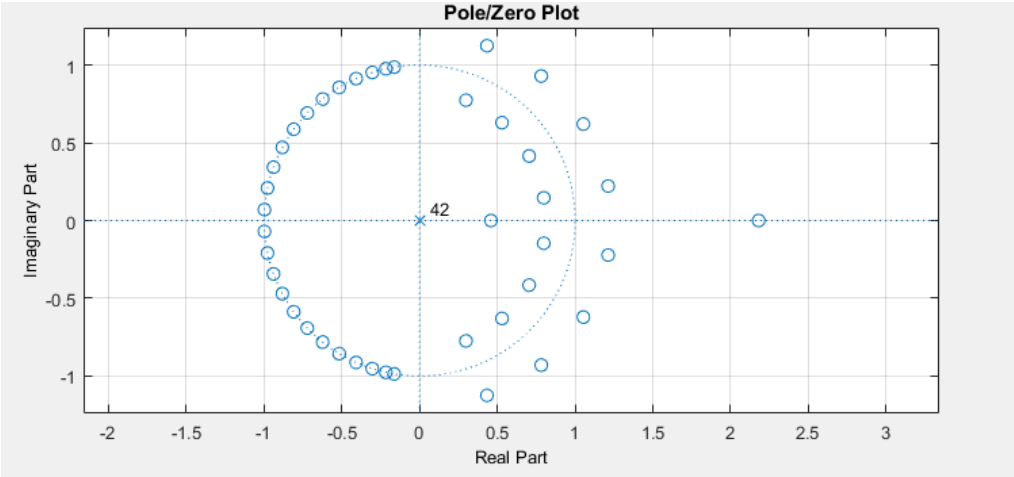


Select filter response i.e. low pass filter in this example. Now select parameters and check filter response.

```
Fpass = 0.45;   % Passband Frequency
Fstop = 0.55;   % Stopband Frequency
Apass = 1;      % Passband Ripple (dB)
Astop = 60;     % Stopband Attenuation (dB)
```







Filter Visualization Tool(fvtool)

fvtool(b,a) opens FVTool and displays the magnitude response of the digital filter defined with numerator, b and denominator, a. Using FVTool you can display the phase response, group delay, impulse response, step response, pole-zero plot, and coefficients of the filter.

Analysis Type	Analysis Option
Magnitude plot	'magnitude'
Phase plot	'phase'
Magnitude and phase plot	`freq'
Group delay plot	'grpdelay'
Phase delay plot	`phasedelay'
Impulse response plot	'impulse'
Step response plot	'step'
Pole-zero plot	'polezero'
Filter coefficients	'coefficients'
Filter information	'info'

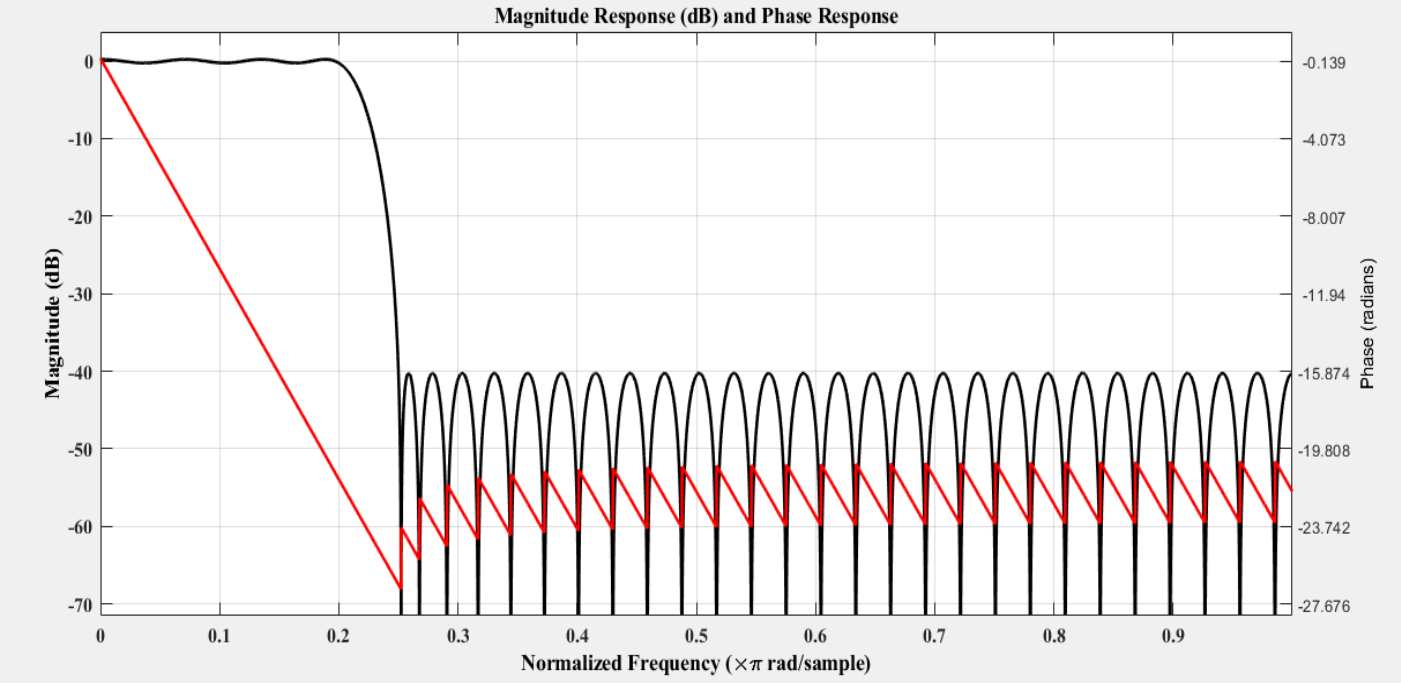
Example:

Design an FIR equiripple lowpass filter. Specify a passband edge frequency of 0.2π rad/sample and a stopband edge frequency of 0.25π rad/sample. Set the passband ripple to 0.5 dB and the stopband attenuation to 40 dB. Use the default equiripple method. Using FVTool display the magnitude, phase response, impulse response, pole-zero plot, and coefficients of the filter.

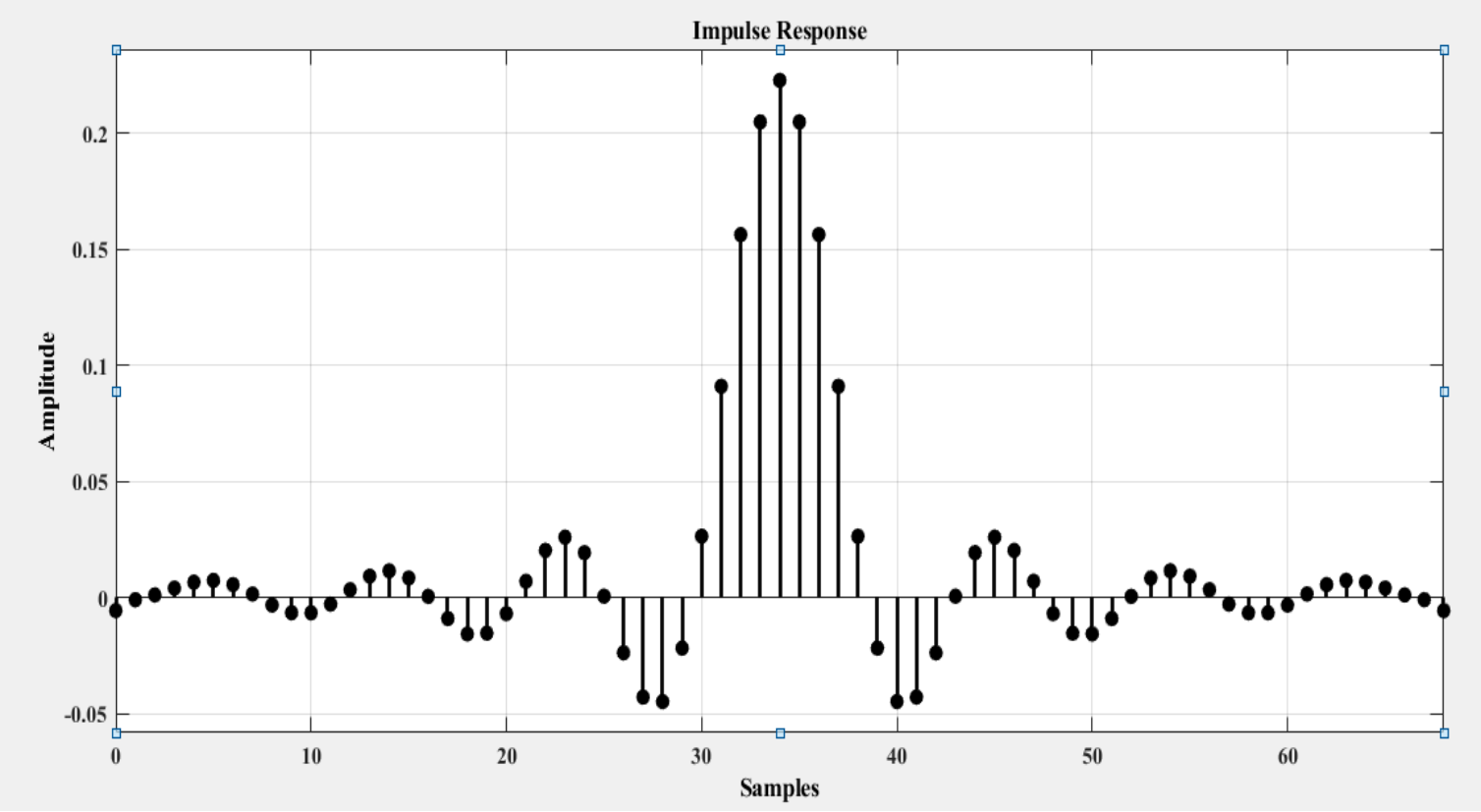
```
D = fdesign.lowpass('Fp,Fst,Ap,Ast',0.2,0.25,0.5,40);
filt = design(D,'SystemObject',true);
fvt = fvtool(filt,'Analysis','_________');
```

Simulation:

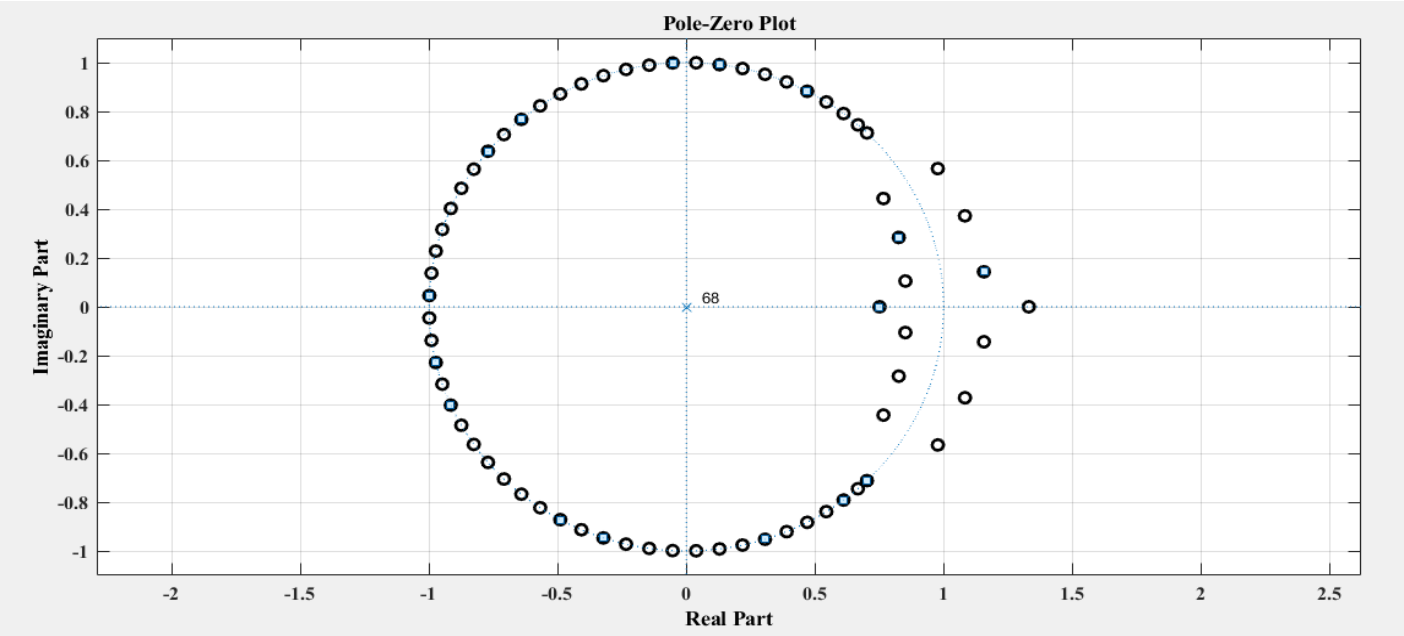
Magnitude response and phase response:



Impulse response:



Pole zero plot:



Coefficients of filter:

```
Numerator:
-0.005713054591710156095152317590191159979
-0.001021447524381082589892133505315996445
 0.001049702122842498122970433804823642276
 0.003988937915096374235701670585285683046
 0.006532258799406769934903227436961969943
 0.007300593696911443671038188796273971093
 0.005497090762148022259625523844306371757
 0.001448166553268463675394062128987116012
-0.003323873222556575791203359315773013805
-0.006642931676935723978028036640353093389
-0.006669247463532027540844548951781689539
-0.002903074918893845841882939140532471356
 0.003304342869975843578062013605745050882
 0.009125541973894759142926780270954623l2
 0.011425109392336258232192314210351469228
 0.008334804663731137236437618298623419832
 0.000428739260994064552093979481028895862
-0.009136568235614251493048598717905406374
-0.015752077069816919352440720558661269024
-0.015441693943204238062549826793201646069
-0.006989679883267950477598873249007738195
 0.006909000614141445374694061598574990057
 0.020226114958739140059496008916539722122
 0.025930222974783763423278770687829819508
 0.019252979023575266664947136519003834ll
 0.000510489867257231020279684141627285499
-0.023891724237570740457803353251620137598
-0.042962445196508186606720869349373970181
-0.044901161118913582304124076927109854296
-0.021859728226650550902965264299382397439
 0.026348474987236629452080904911781544797
 0.090946882317054167366165984276449307799
 0.156333829003829111181644861972017679363
 0.204876993153564668226351841440191492438
```

What is the difference between Analog filter and Digital filter?

1.The main difference between the analog and the digital filter is that a digital filter needs to sample the input signal (analog signal) and then convert it into binary numbers.
2 Analog filters are designed with various components like resistor, inductor and capacitor whereas digital Filters are designed with digital hardware like FF, counters shift registers, ALU and software s like C or assembly language.
3.Analog filters less accurate & because of component tolerance of active components & more sensitive to environmental changes. Whereas digital filters are less sensitive to the environmental changes, noise and disturbances. Thus periodic calibration can be avoided. Also they are extremely stable.

Task1

Design high pass and band pass equiripple FIR filter of minimum order using **filterDesigner** command. Plot magnitude response and pole zero spectrum.

Specifications:

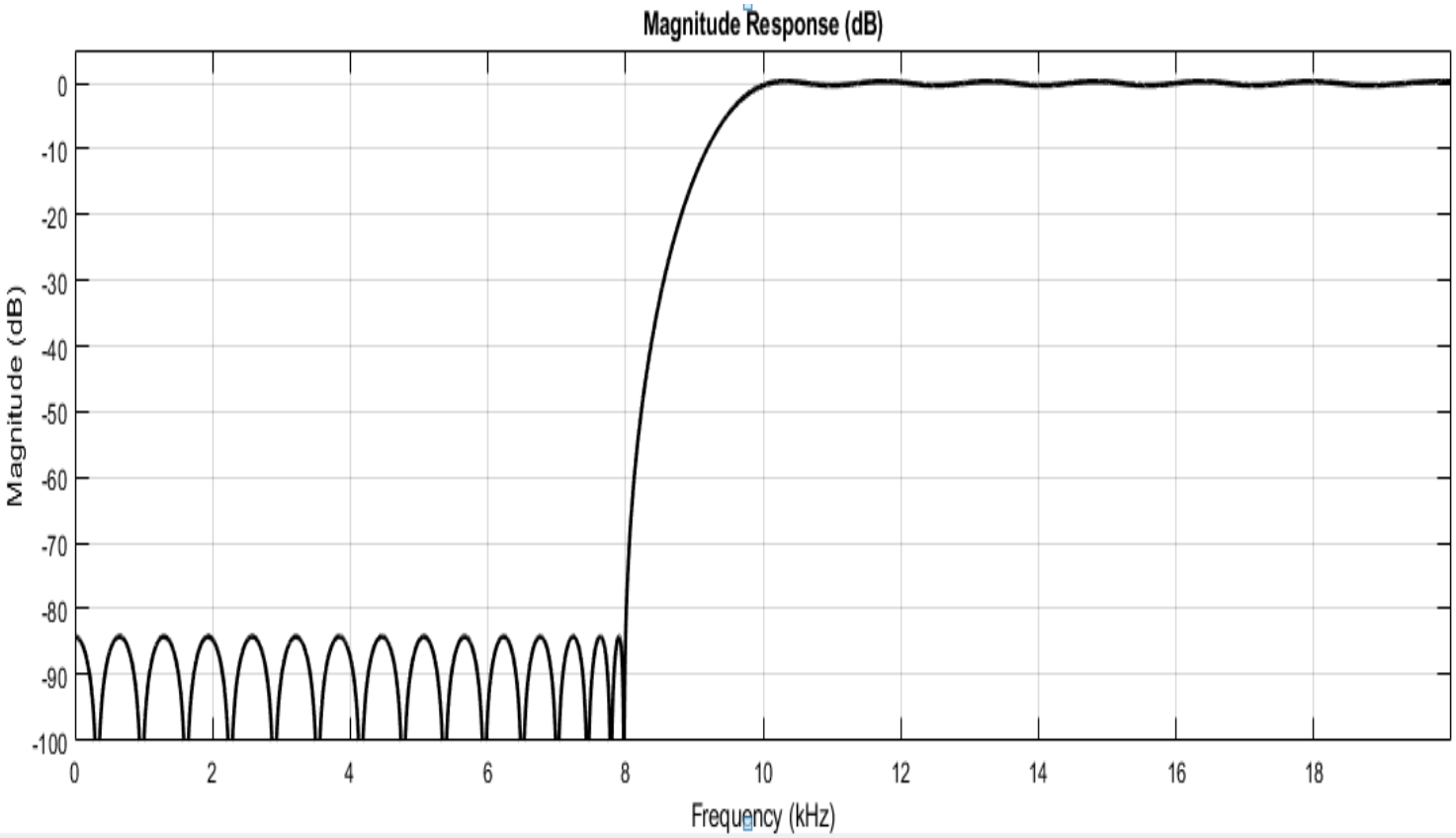
F_s = 40 kHz

F_{pass} = 10 kHz

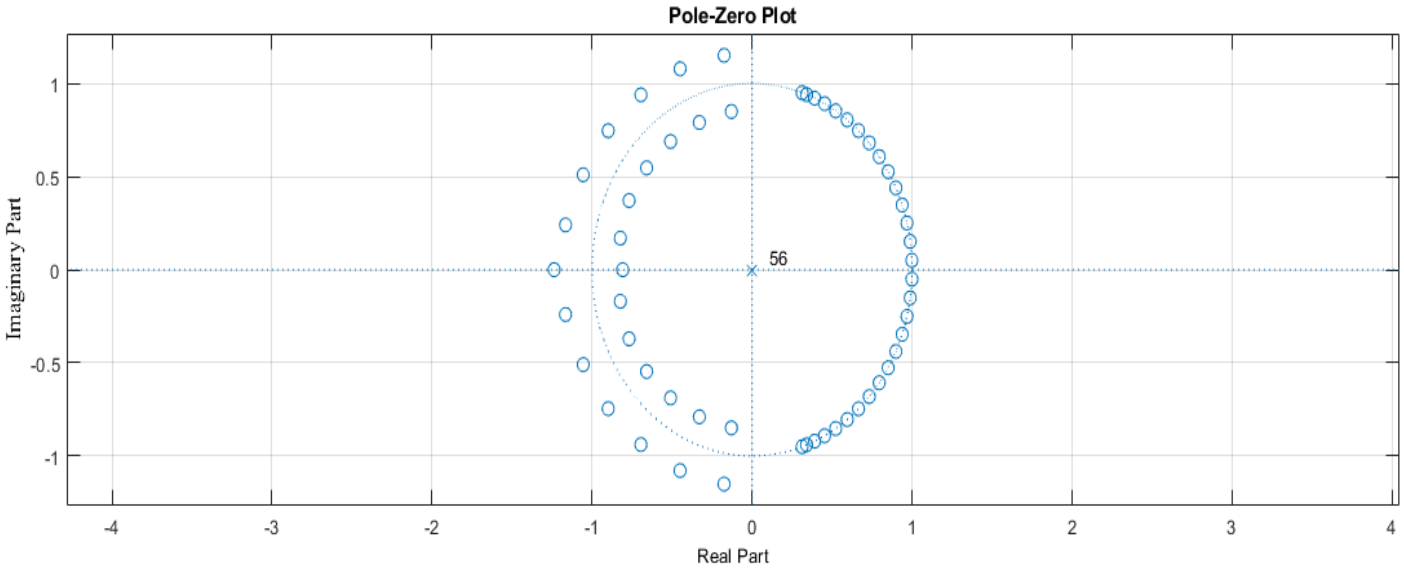
F_{stop} = 8kHz

High pass filter:

Magnitude response of high pass filter.:



Pole zero plot.:



Bandpass:

F_s = 40 kHz

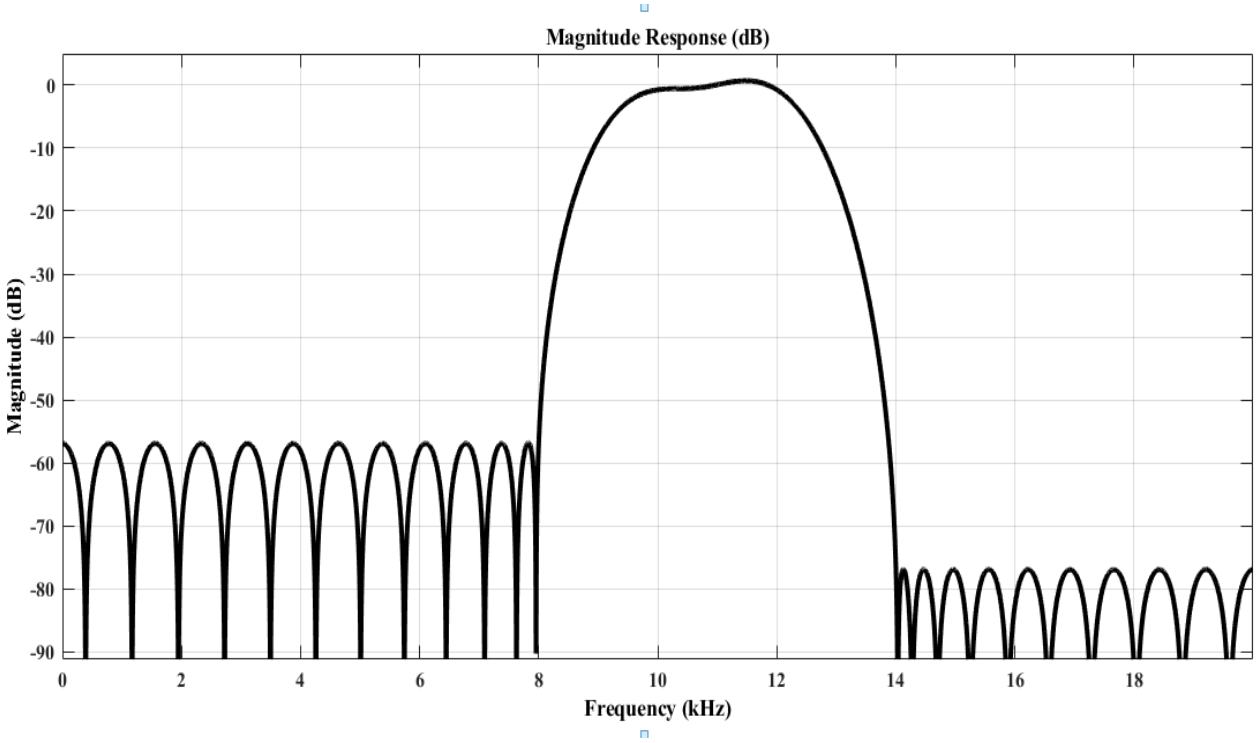
$F_{pass\ 1}$ = 10 kHz

$F_{stop\ 1}$ = 8 kH

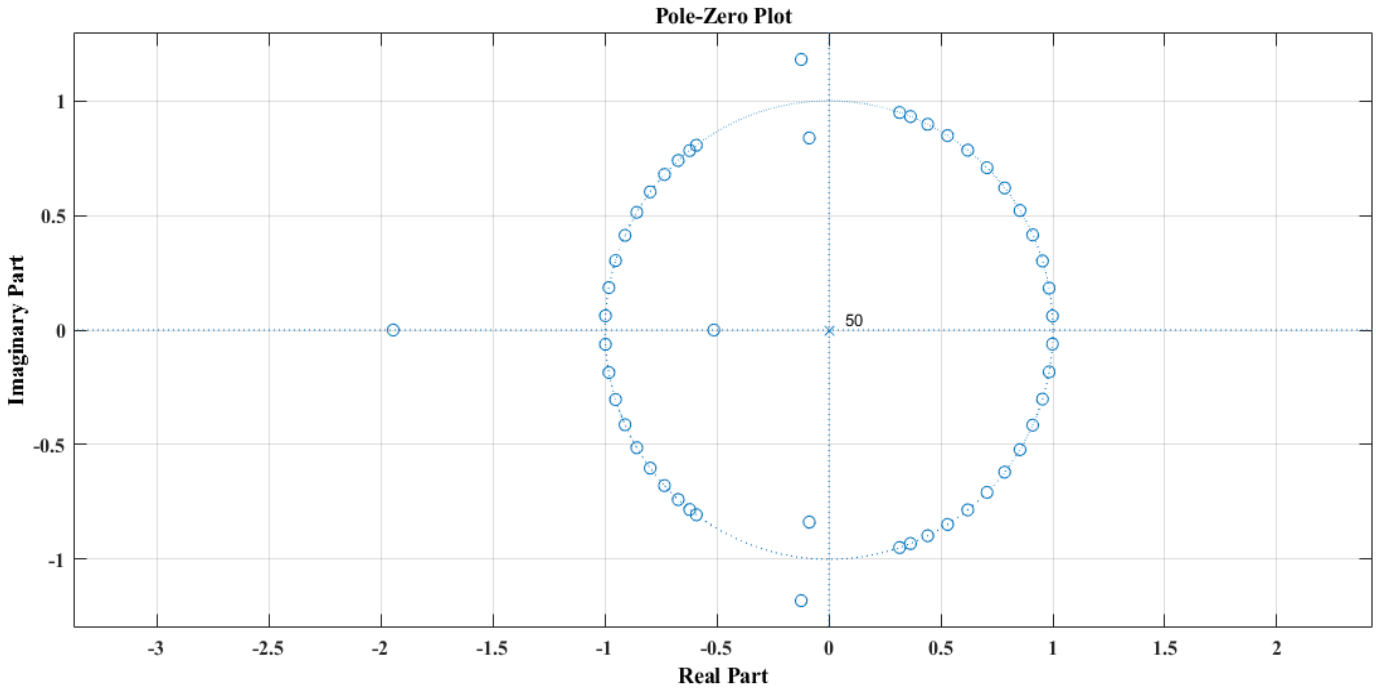
$F_{pass\ 2}$ = 12 kHz

$F_{stop\ 2}$ = 14 kHz

Magnitude response:



Pole zero plot:



Task2

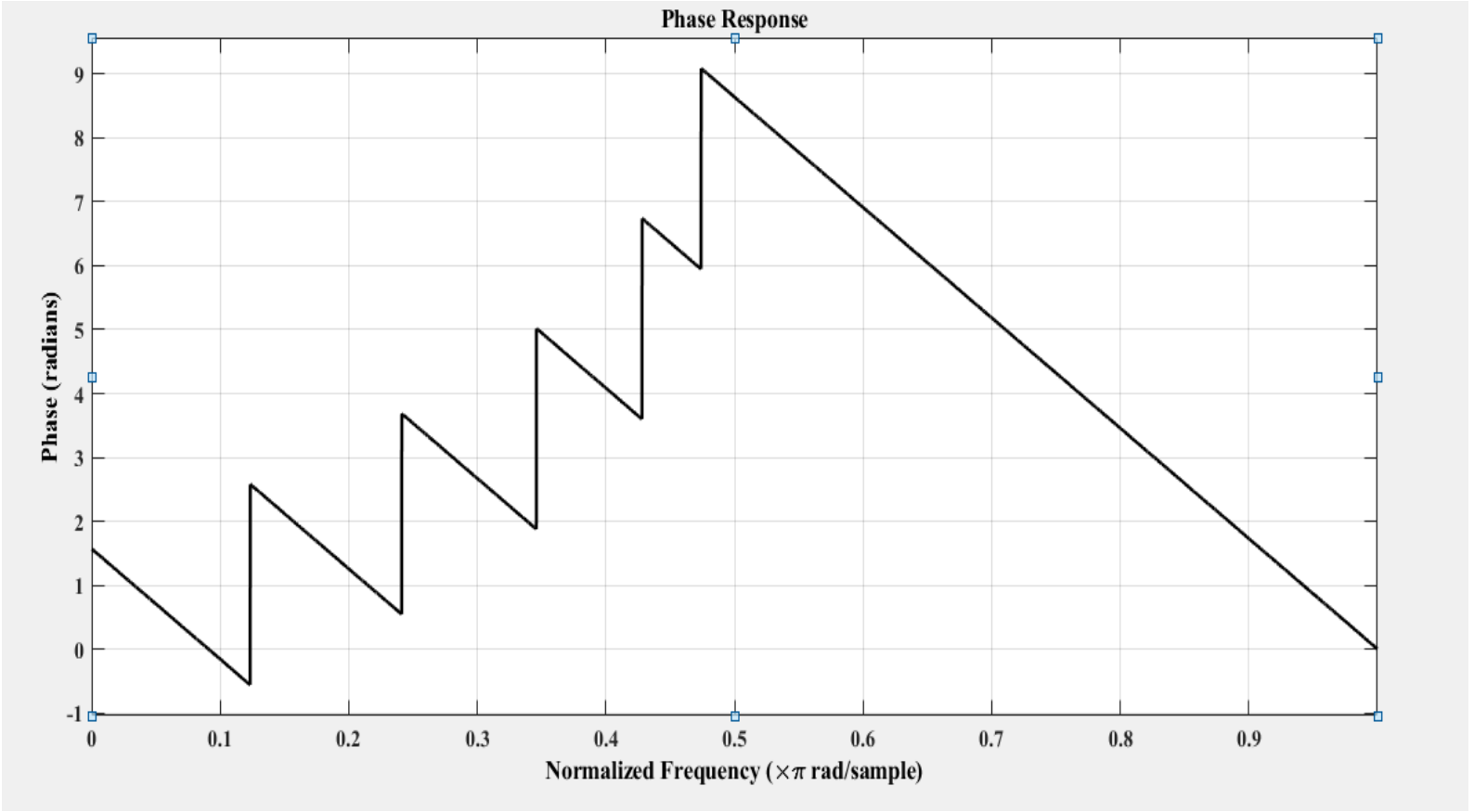
Design high pass, band stop and band pass equiripple FIR filter of minimum order using **filterBuilder** command. Plot phase response and impulse response spectrum.

Specifications:

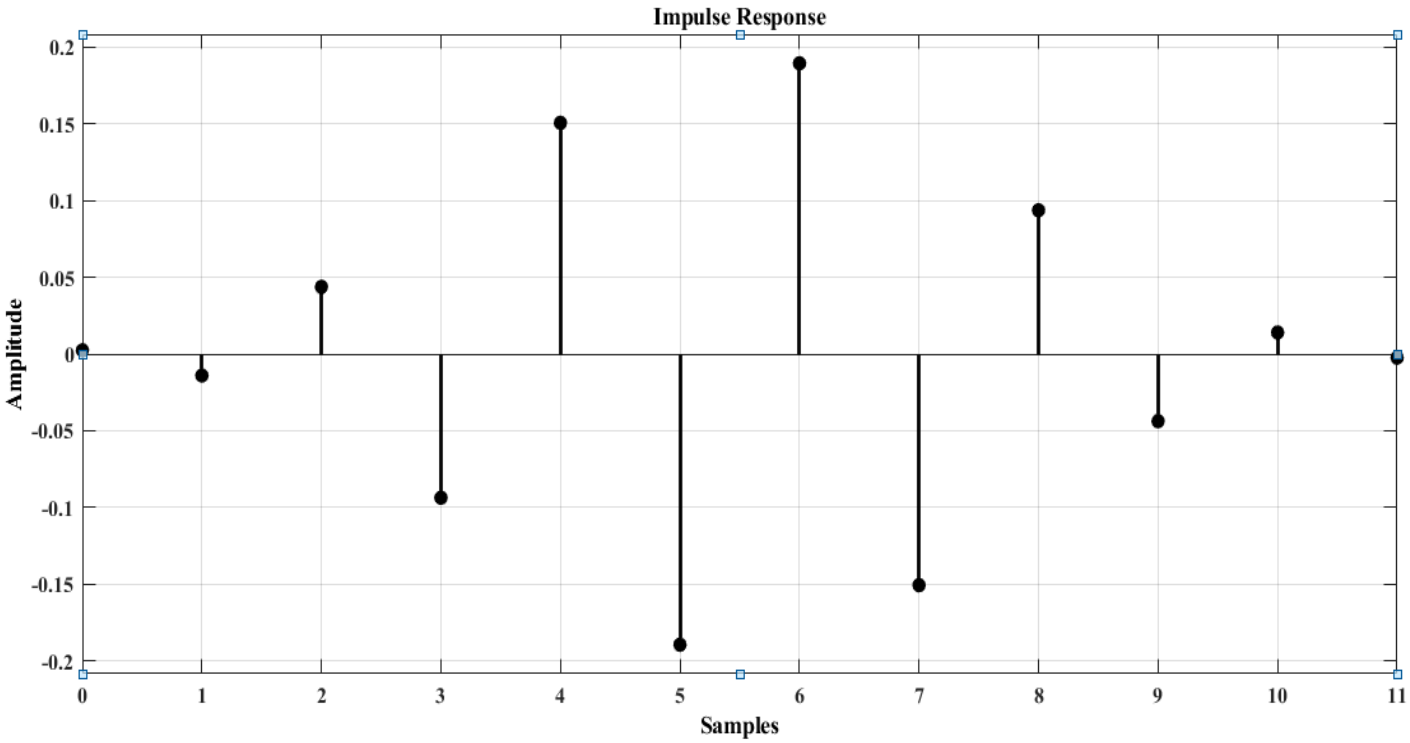
```
Fpass = 0.48;   % Passband Frequency
Fstop = 0.96;   % Stopband Frequency
Apass = 1;      % Passband Ripple (dB)
Astop = 80;     % Stopband Attenuation (dB)
```

High pass:

Phase response.:



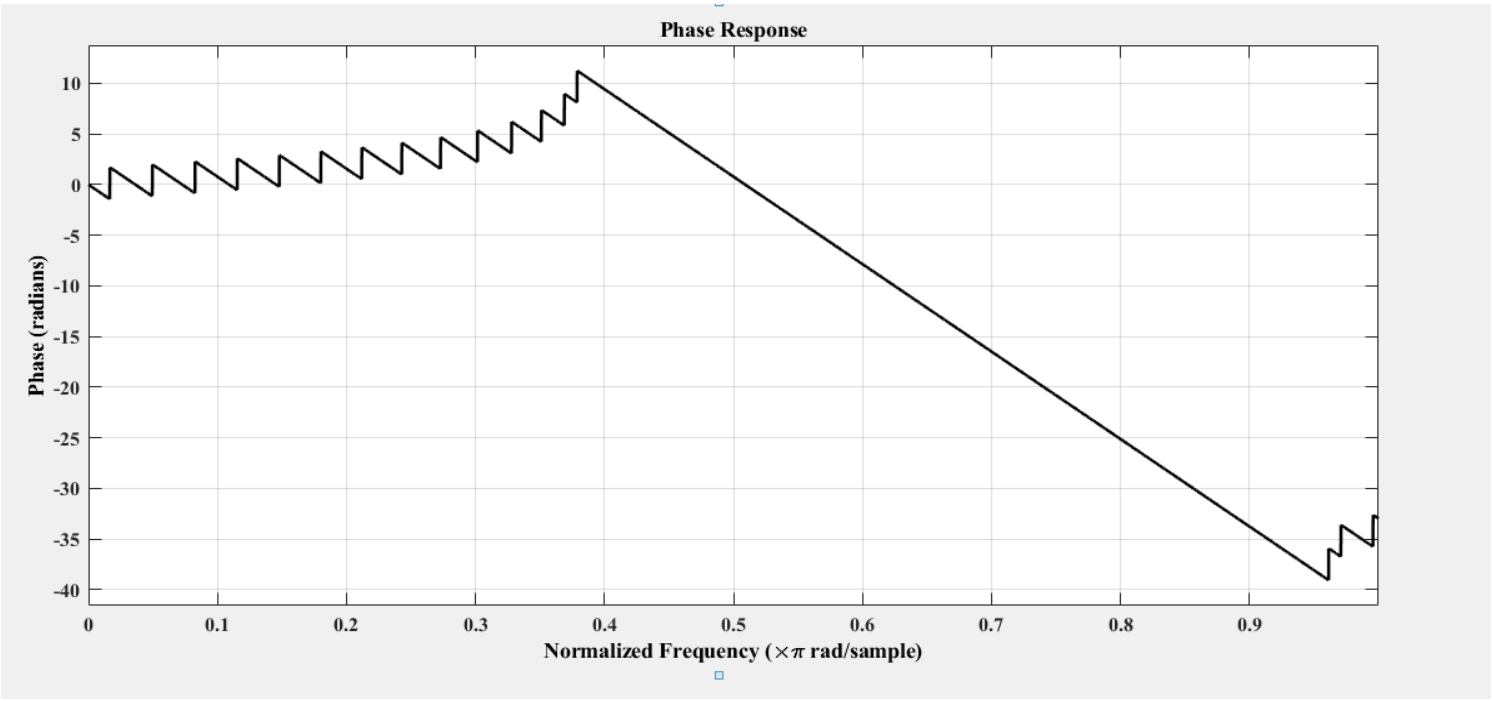
Impulse response:



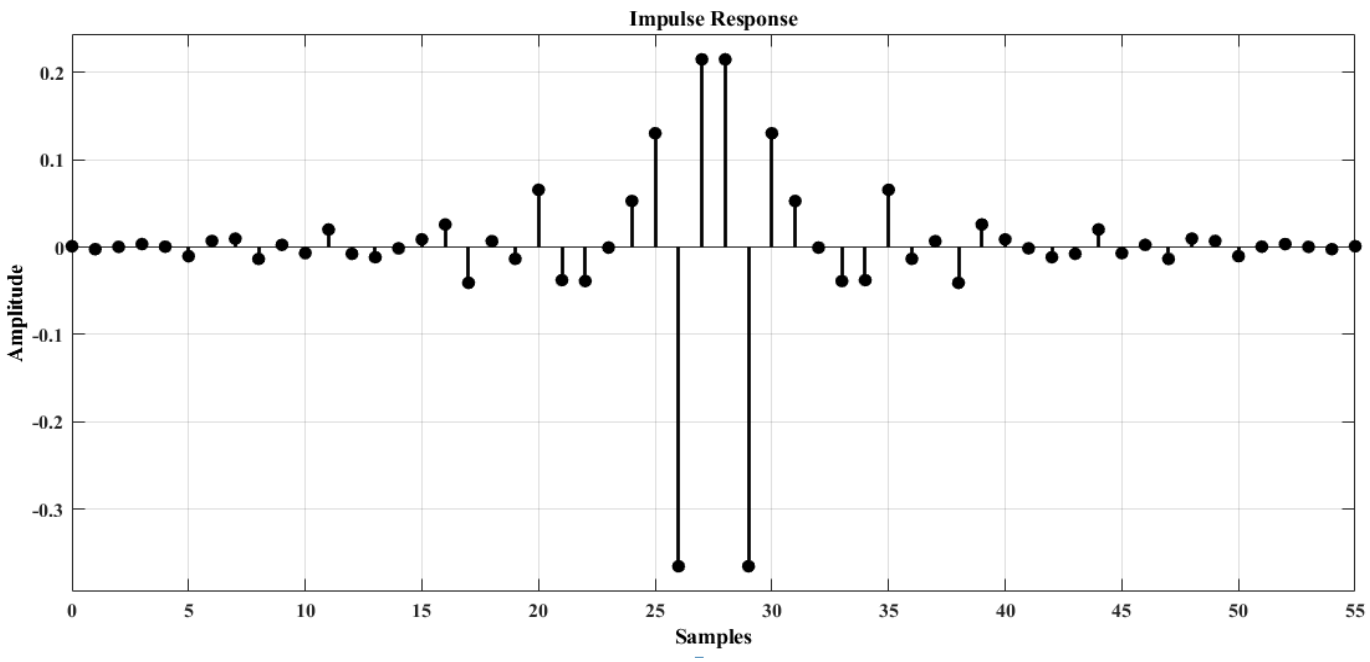
Bandpass:

```
Fpass1 = 0.35;   % Passband Frequency
Fstop = 0.48;   % Stopband Frequency
Apass = 1;      % Passband Ripple (dB)
Astop = 80;     % Stopband Attenuation (dB)
Fpass2 = 0.96;   % Passband Frequency
Fstop2 = 0.85;   % Stopband Frequency
```

Phase response:

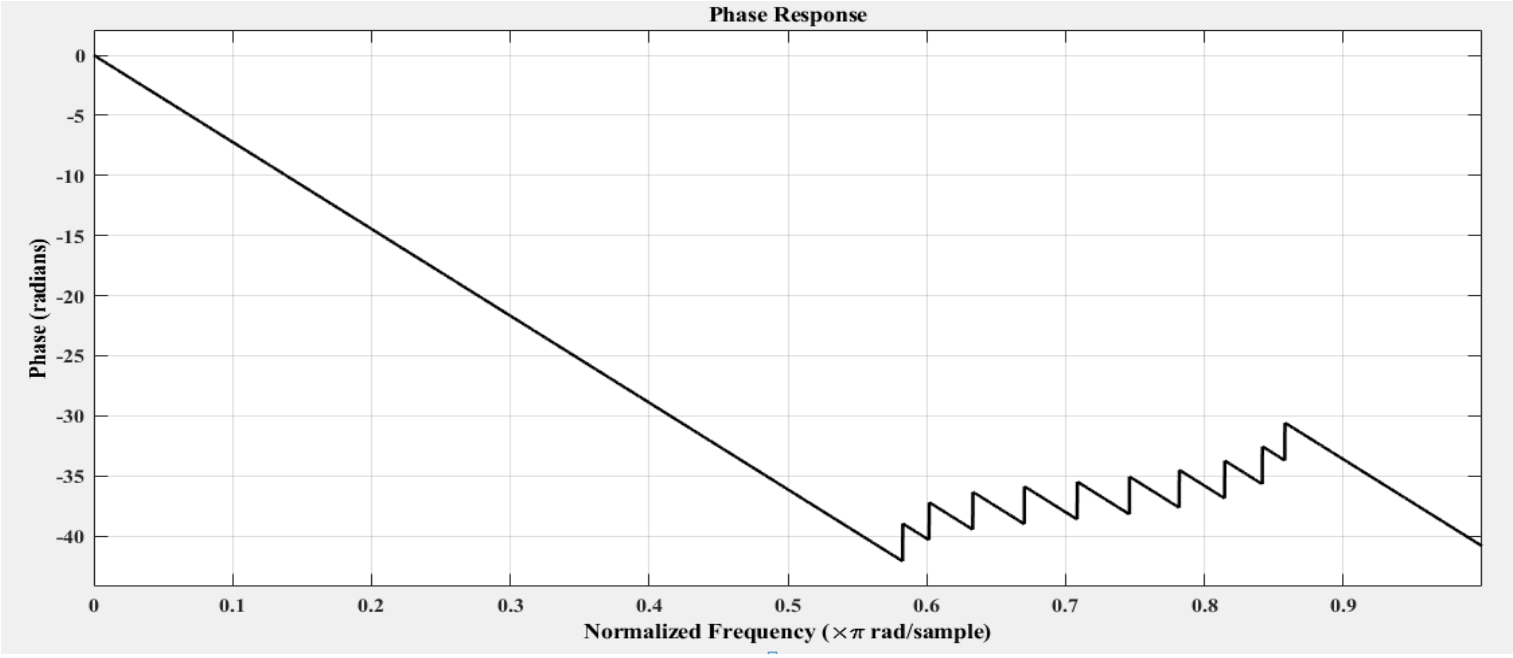


Impulse response:

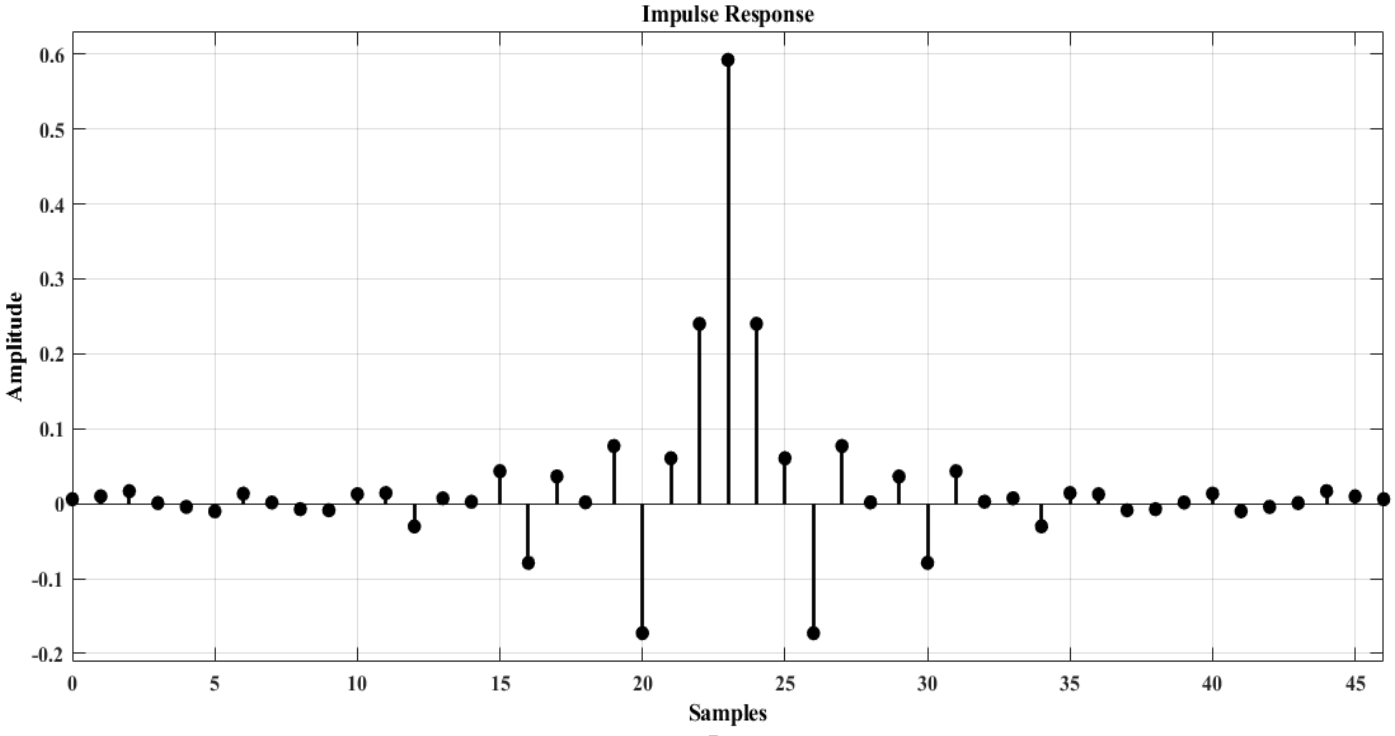


Bandstop:

Phase response:



Impulse response:



Conclusion:

In this lab, we have designed FIR filters using filterbuilder and filterDesigner command in MATLAB.We have designed high pass, bandpass ,bandstop and lowpass filters using MATLAB build in commands . We know that FIR filter are always stable So the impulse response of filter that does not goes indefinitely without becoming exactly zero at some point that it is impulse response of an FIR filter.

Date: 21_06_2021 **Registration#2018-EE-394**

University of Engineering & Technology Lahore

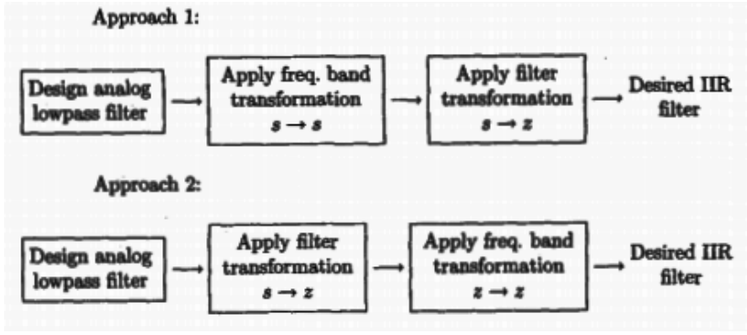
Experiment # 11

Title: Designing of IIR Filters

Equipment Required: Personal computer (PC) with windows operating system and MATLAB software

Theory:

IIR systems or IIR filters, and are distinguished by having an impulse response which does not become exactly zero past a certain point, but continues indefinitely. This is in contrast to a finite impulse response (FIR) in which the impulse response h(t) does become exactly zero at times t > T for some finite T, thus being of finite duration. There are two approaches or basic techniques of IIR filter design:



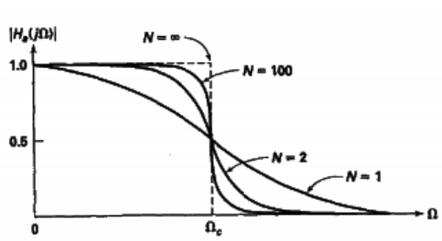
The first approach is used in MATLAB to design IIR filters. Three basic prototypes (Butterworth, Chebyshev, Elliptic) are widely used in practice.

Butterworth Filter:

This filter is characterized by the property that its magnitude response is worth flat in both passband and stopband. The magnitude-squared response of Nth-order lowpass fiter is given by

$$|H_a(j\Omega)|^2 = \frac{1}{1+\left(\frac{\Omega}{\Omega_c}\right)^{2N}}$$

Where, N is the order of the filter and Ω_c is the cutoff frequency in rad/sec. the magnitude response of butterworth lowpass filter is



To determine the system function $H_a(s)$

$$H_a(s)H_a(-s) = |H_a(j\Omega)|^2\Big|_{\Omega=s/j} = \frac{1}{1+\left(\frac{s}{j\Omega_c}\right)^{2N}} = \frac{(j\Omega)^{2N}}{s^{2N}+(j\Omega_c)^{2N}}$$

$$H_a(s) = \frac{\Omega_c^N}{\prod_{\text{LHP poles}}(s-p_k)}$$

Date: 21_06_2021 **Registration#2018-EE-394**

Designing Equations:

$$N = \left\lceil \frac{\log_{10}\left[\left(10^{R_p/10}-1\right)/\left(10^{A_s/10}-1\right)\right]}{2\log_{10}\left(\Omega_p/\Omega_s\right)} \right\rceil$$

N = Order of the filter

$\Omega_c = \ Cutoff\ frequency$

$\Omega_p = Passband\ cutoff\ frequency$

$\Omega_s = Stopband\ cutoff\ frequency$

$R_p = Passband\ ripple$

$A_s = stopband\ ripple$

$$\Omega_c = \frac{\Omega_s}{2N\sqrt{\left(10^{A_s/10}-1\right)}} \qquad \Omega_c = \frac{\Omega_p}{2N\sqrt{\left(10^{R_p/10}-1\right)}}$$

Example 1:

Design an analog lowpass butterworth IIR filter to satisfy

Passband cutoff = 0.2π; Passband ripple: = 7dB
Stopband cutoff = 0.3 π; Stopband ripple: = 16dB

MATLAB Script

```
function [b,a] = u_buttap(N, Omegac)
% Unnormalized Butterworth Analog Lowpass Filter Prototype
% [b,a] = u_buttap(N,Omegac) ;
% b = numerator polynomial coefficients of Ha(s)
% a = denominator polynomial coefficients of Ha(s)
% N = Order of the Butternorth Filter
% Omegac = Cutoff frequency in radians/sec
[z,p,k] = buttap(N);
p = p*Omegac;
k = k*Omegac^N;
B = real(poly(z));
b0 = k;
b = k*B;
a = real(poly(p));
```

```
function [b,a] = afd_butt (Wp,Ws,Rp,As)
% Analog Lowpass Filter Design: Butterworth
% [b,a] = afd-butt(Wp,Ws,Rp,As);
% b = Numerator coefficients of Hab)
% a = Denominator coefficients of Ha(d
% Wp = Passband edge frequency in rad/sec; Wp > 0
% Ws = Stopband edge frequency in rad/sec; Ws > Wp > 0
% Rp = Passband ripple in +dB; (Rp > 0)
% As = Stopband attenuation in +dB; (As > 0)

if Wp <= 0
error('Passband edge must be larger than 0')
end
if Ws <= Wp
error('Stopband edge must be larger than Passband edge')
end
if (Rp <= 0) || (As < 0)
error('PB ripple and/or SB attenuation must be larger than 0')
end
N = ceil((log((10^(Rp/10)-1)/(10^(As/10)-1)))/(2*log(Wp/Ws)));
```

```
fprintf('\n*** Butterworth Filter Order = %2.of \n' , N);
OmegaC = Wp/((10^(Rp/10)-1)^(1/(2*N))) ;
[b,a]= u_buttap(N,OmegaC) ;
```

```
function [db,mag,pha,w] = freqs_m(b,a,wmax);
% Computation of s-domain frequency response: Modified version
% .....................................................
% [db,mag,pha,w] = freqs_m(b,a,wmax);
% db = Relative magnitude in db over [0 to wmax]
% mag = Absolute magnitude over [0 to wmax]
% pha = Phase response in radians over [0 to wmax]
% W = array of 500 frequency samples between [0 to max]
% b = Numerator polynomial coefficents of Ha(s)
% a = Denominator polynomial coefficents of Ha(s)
% wmax = Maximm frequency in rad/sec over which response is desired

w = [0:1:500]*wmax/500;
H = freqs(b,a,w);
mag = abs(H);
db = 20*log10( (mag+eps)/max(mag));
pha = angle(H) ;
```

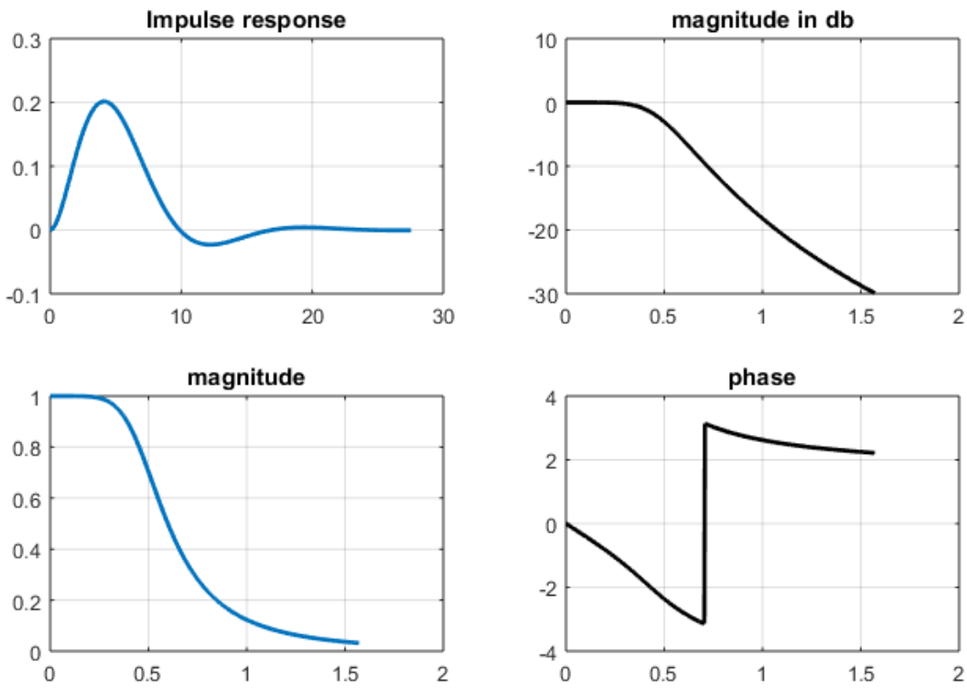
```
Wp = 0.2*pi; Ws = 0.3*pi; Rp = 7; As = 16;
Ripple = 10 ^(-Rp/20); Attn = 10 ^(-As/20);
 % Analog filter design:
 [b,a] = afd_butt( Wp , Ws , Rp , As );
 %Butterworth Filter Order = ??

 % Calculation of Frequency Response:
[db,mag,pha,w] = freqs_m(b,a,0.5*pi);
 % Calculation of Impulse response:
[ha,x,t] = impulse(b,a);
```

```
1     %2018-EE-394
2     %lab dsp 11
3 -   Wp = 0.2*pi; Ws = 0.3*pi; Rp = 7; As = 16;
4 -   Ripple = 10 ^(-Rp/20); Attn = 10 ^(-As/20);
5     % Analog filter design:
6 -   [b,a] = afd_butt( Wp , Ws , Rp , As );
7 -   [db,mag,pha,w] = freqs_m(b,a,0.5*pi);
8     % Calculation of Impulse response:
9 -   [ha,x,t] = impulse(b,a);
10 -  subplot(2,2,1)
11 -  plot(t,ha)
12 -  title('Impulse response')
13 -  subplot(2,2,2)
14 -  plot(w,db)
15 -  title('magnitude in db')
16 -  subplot(2,2,3)
17 -  plot(w,mag)
18 -  title('magnitude')
19 -  subplot(2,2,4)
20 -  plot(w,pha)
21 -  title('phase')
22
```

Command Window

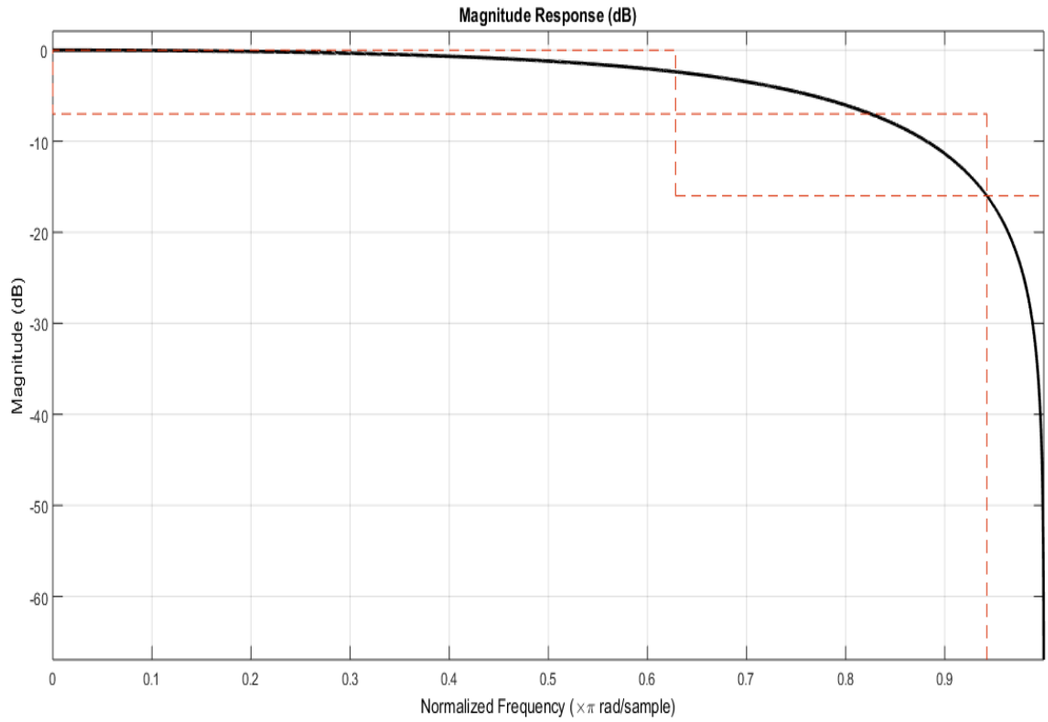
Date: 21_06_2021 **Registration#2018-EE-394**

Simulation:



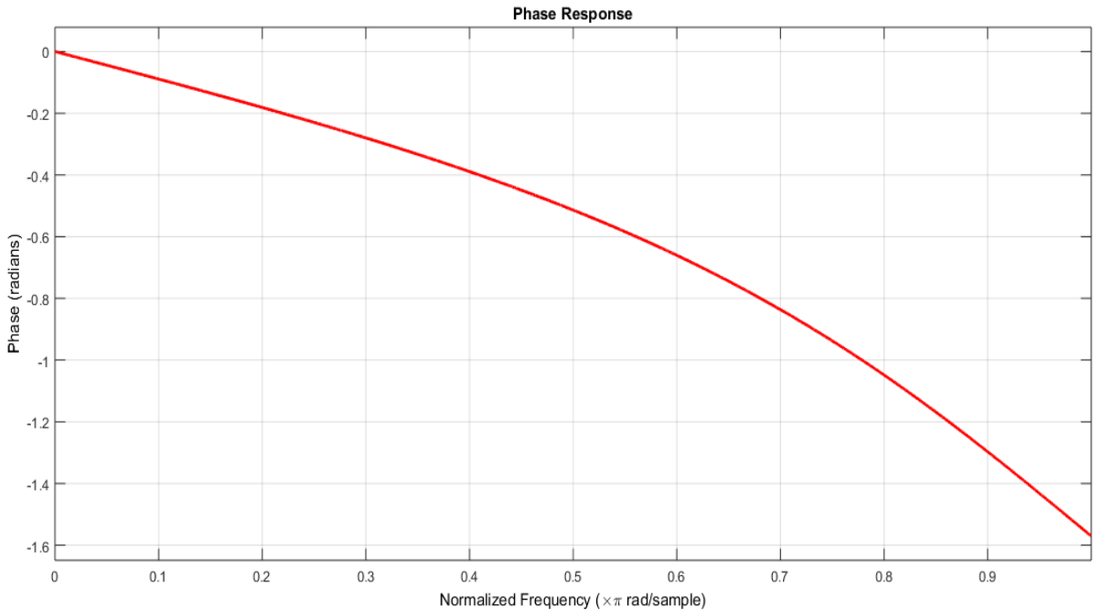
Design same filter using "filterBuilder" Tool.

Magnitude response:

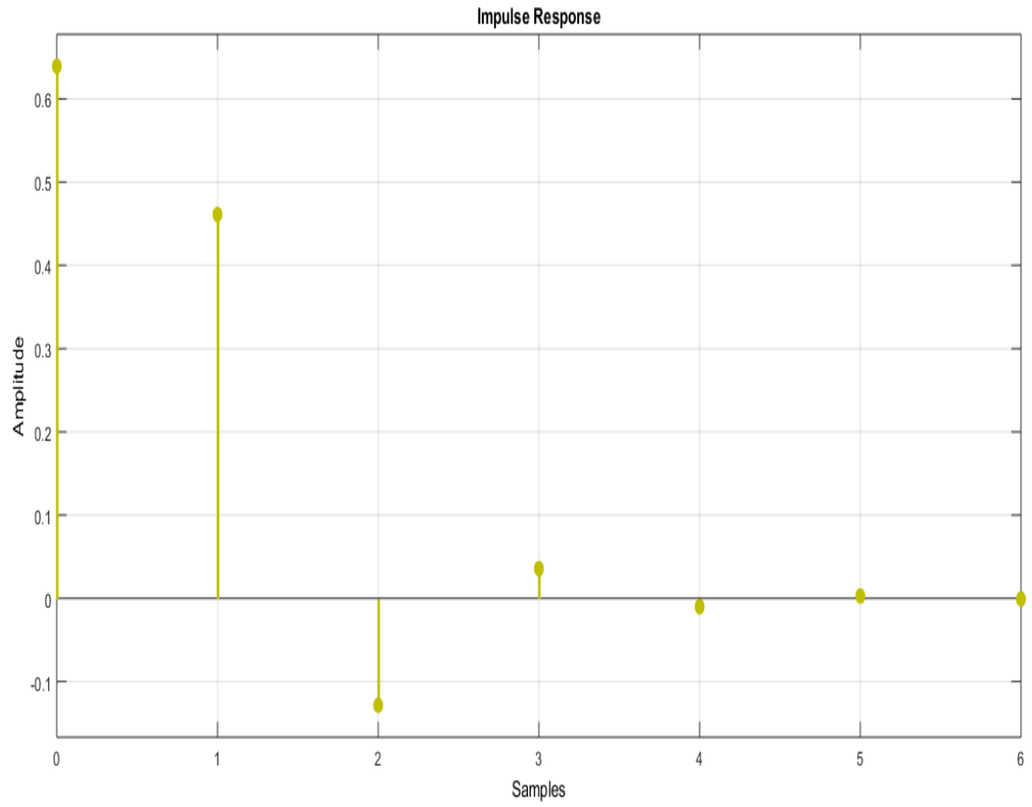


Date: 21_06_2021 **Registration#2018-EE-394**

Phase response:



Impulse response:



Transformation Analog to Digital filter:

These transformations are complex valued mappings. To preserve the shape of the impulse response from analog to digital filter, impulse invariance transformation technique is extensively used.

In this design method, the digital filter impulse response to look similar to that of a frequency selective analog filter. Hence, sample $h_a(t)$ at some sampling interval T to obtain h(n).

$$h(n) = h_a(nT).$$

Parameter T is selected so that the shape of $h_a(t)$ is captured by the samples. The analog to digital frequencies are related

$$e^{j\omega} = e^{j\Omega T} \text{ or } \omega = \Omega T$$

$$z = e^{sT}$$

Procedure:

Given the digital lowpass filter specifications ω_p, $\omega_s, R_p$and A_s. To determine H(z) by first designing an equivalent analog filter and then mapping it into the desired digital filter. The steps required for this procedure are

1. Choose T and determine the analog frequencies

$$\Omega_p = \frac{\omega_p}{T_p} \quad \text{and} \quad \Omega_s = \frac{\omega_s}{T}$$

2. Design an analog filter H_a(s) using the specifications**.**
3. Using partial fraction expansion, expand H_a(s) into

$$H_a(s) = \sum_{k=1}^{N} \frac{R_k}{s - p_k}$$

4. Now transform analog poles ***(p_k)*** into digital poles ***($e^{p_k T}$)*** to obtain the digital filter

$$H(z) = \sum_{k=1}^{N} \frac{R_k}{1 - e^{p_k T} z^{-1}}$$

Example 2:

Design a digital lowpass digital filter using butterworth prototype to satisfy

Passband frequency = 0.2π; Passband ripple: = 1dB
Stopband frequency = 0.3 π; Stopband ripple: = 15dB

MATLAB Script

```
function [C,B,A] = dir2par(b,a);
% Dim-form to Parallel-form conversion
% ------------------------------------
% [C,B,A] = dir2par(b,a)
% C = Polynomial part when length(b) >= length(a)
% B = K by 2 matrix of real coefficients containing bk's
% A = K by 3 matrix of real coefficients containing ak's
% b = numerator polynomial coefficients of DIRECT form
% a = denominator polynomial coefficients of DIRECT form

M = length(b) ; N = length(a);
[r1,p1,C] = residuez(b,a);
p = cplxpair(p1,10000000*eps);
I = cplxcomp(p1 ,p);
r = r1(I);
K = floor(N/2); B = zeros(K,2); A = zeros(K,3);
if K*2 == N; %N even, order of A(z) odd, one factor is first order
for i=1:2:N-2
Brow = r(i:1:i+1,:);
Arow = p(i:1:i+1,:);
[Brow,Arow] = residuez(Brow,Arow,[]);
B(fix((i+1)/2), :) = real(Brow) ;
A(fix((i+1)/2), :) = real(Arow);
end
[Brow,Arow] = residuez(r(N-1) ,p(N-1), []) ;
B(K,:) = [real(Brow) 0]; A(K,:) = [real(Arow) 0];
else
for i=1:2:N-1
```

```
Brow = r(i:1:i+1,:);
Arow = p(i:1:i+1,:);
[Brow,Arow] = residuez(Brow, Arow,[] );
B(fix((i+1)/2), :) = real(Brow) ;
A(fix((i+1)/2), :) = real(Arow);
end
end
```

```
function I = cplxcomp(p1,p2)

%  I = cplxcomp(p1,p2)

% Compares two complex pairs which contain the same scalar elements

%  but (possibly) at different indices.  This routine should be

%  used after CPLXPAIR routine for rearranging pole vector and its

%  corresponding residue vector.

%      p2 = cplxpair(p1)

I=[];

for j=1:1:length(p2)

   for i=1:1:length(p1)

   if (abs(p1(i)-p2(j)) < 0.0001)

      I=[I,i];

       end

   end

end

I=I';
```

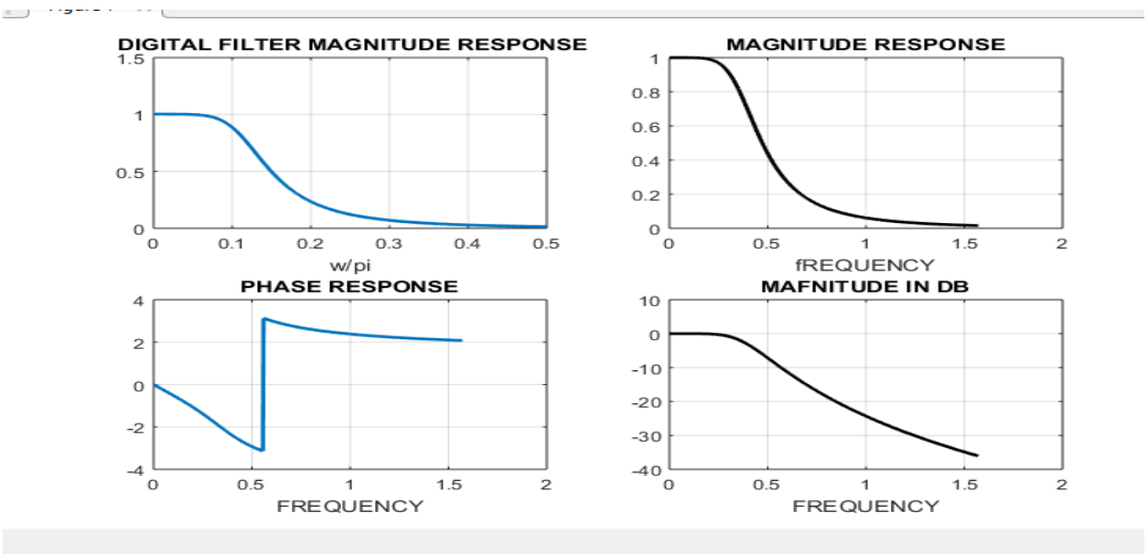
```
function [b,a]= imp_invr(c,d,T)
%impulse invariance transformation from analog to digital filter
%b= numerator polynomial in z^(-1) of digital filter
%a= Denominator polynomial in z^(-1) of digital filter
%c= numerator polynomial in s of analog filter
%d= Denominator polynomial in s of analog filter
%T= Sampling(transformation) parameter
[R,p,k]= residue(c,d)
p= exp(p*T);
[b,a]= residuez(R,p,k);
b=real(b');
a=real(a');
end
```

Date: 21_06_2021 **Registration#2018-EE-394**

```
%2018-EE-394
%Lab dsp
% Digital Filter Specifications:
wp = 0.1*pi; % digital Passband freq in Hz
ws = 0.3*pi; % digital Stopband freq in Hz
Rp = 1; % Passband ripple in dB
As = 15; % Stopband attenuation in dB
Ripple = 10 ^(-Rp/20); Attn = 10 ^(-As/20);
% Analog filter design:
[b,a] = afd_butt(wp,ws,Rp,As)
%Butterworth Filter Order = ??
% Calculation of Frequency Response:
[db,mag,pha,w] = freqs_m(b,a,0.5*pi);
% Analog Prototype Specifications: Inverse mapping for frequencies
T = 1; % Set T=1
OmegaP = wp / T; % Prototype Passband freq
Omegas = ws / T; % Prototype Stopband freq
% Analog Butterworth Prototype Filter Calculation:
[cs ,ds] = afd_butt(OmegaP,Omegas,Rp,As)
% Impulse Invariance transformation:
[b,a] = imp_invr(cs,ds,T);
[C,B,A] = dir2par(b,a)
z=exp(j.*w)
Hz=((-0.3935+.3670.*z.^-1)./(1-1.5483.*z.^-1+0.674.*z.^-2))+((0.395)./(1-0.6747.*z.^-1))
subplot(2,2,1)
plot(w/pi,abs(Hz));
  xlabel('w/pi');title(' DIGITAL FILTER MAGNITUDE RESPONSE')
  subplot(2,2,2)
  plot(w,mag);
  title('MAGNITUDE RESPONSE')
  xlabel('fREQUENCY')
  subplot(2,2,3)
  plot(w,pha);
  title('PHASE RESPONSE')
  xlabel('FREQUENCY')
  subplot(2,2,4)
  plot(w,db);grid on
  title('MAFNITUDE IN DB')
  xlabel('FREQUENCY')
```

Simulation:

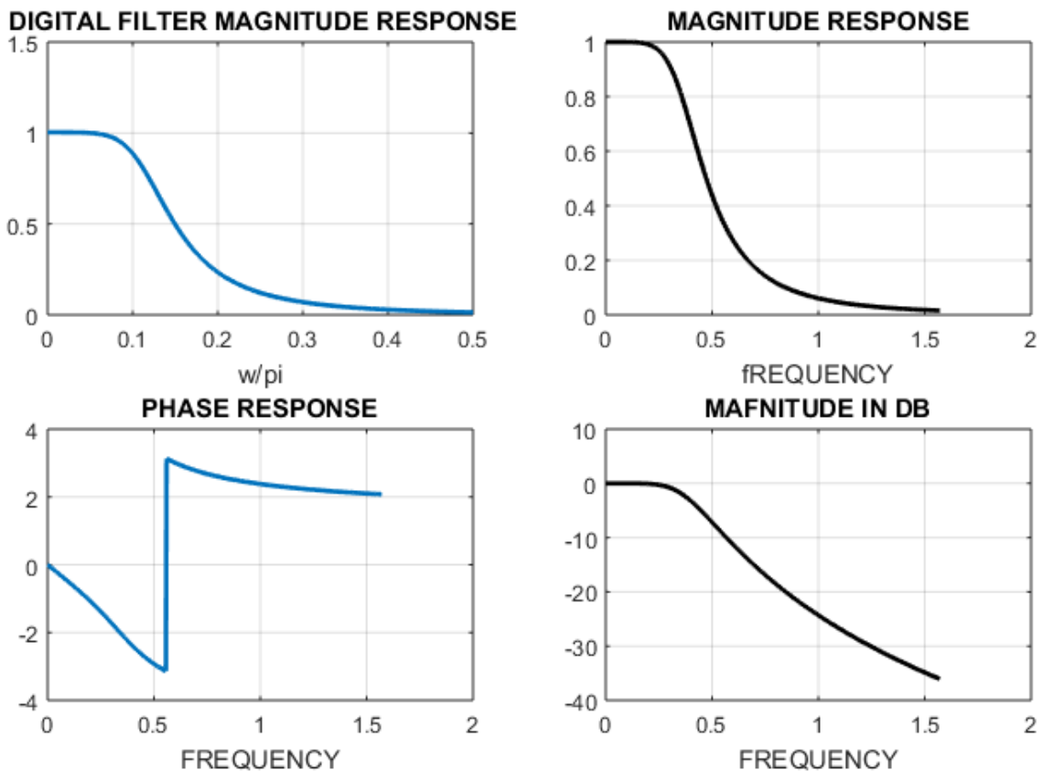
When T=1



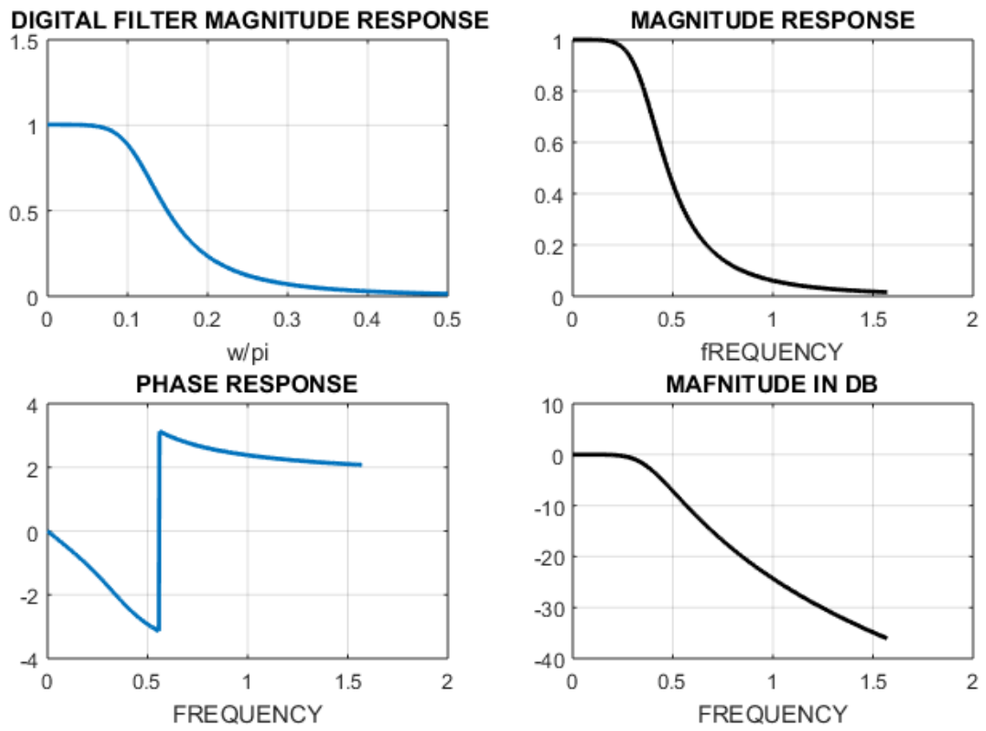
Simulate above example for T= 2 and T=0.5

T=2:

Date: 21_06_2021 **Registration#2018-EE-394**



T=0.5



What is the effect of transformation parameter (T) on magnitude response of filters (analog & digital)?

There is no changing in magnitude responses of filters analog and digital when we change the transformation parameter in 1 2 and 0.5 no prominent change on any response

Task:

Design a digital and analog lowpass butterworth filter to satisfy

Passband cutoff = 0 .2π; Passband ripple: = 1dB
Stopband cutoff = 0.3 π; Stopband ripple: = 15dB

Date: 21_06_2021 **Registration#2018-EE-394**

T = 1

Determine the system function in the rational form and plot its magnitude response (plot magnitude response of analog and digital filter).

```
%2018-EE-394
%lab dsp 11
function [b,a] = u_buttap(N, Omegac)
% Unnormalized Butterworth Analog Lowpass Filter Prototype
% [b,a] = u_buttap(N,Omegac) ;
% b = numerator polynomial coefficients of Ha(s)
% a = denominator polynomial coefficients of Ha(s)
% N = Order of the Butternorth Filter
% Omegac = Cutoff frequency in radians/sec
[z,p,k] = buttap(N);
p = p*Omegac;
k = k*Omegac^N;
B = real(poly(z));
b0 = k;
b = k*B;
a = real(poly(p));

```

```
%2018-EE-394
%lab dsp 11
function [b,a] = afd_butt (Wp,Ws,Rp,As)
% Analog Lowpass Filter Design: Butterworth
if Wp <= 0
error('Passband edge must be larger than 0')
end
if Ws <= Wp
error('Stopband edge must be larger than Passband edge')
end
if (Rp <= 0) || (As < 0)
error('PB ripple and/or SB attenuation must be larger than 0')
end
N = ceil((log((10^(Rp/10)-1)/(10^(As/10)-1)))/(2*log(Wp/Ws)));
fprintf('\n*** Butterworth Filter Order = %2.of \n' , N);
OmegaC = Wp/((10^(Rp/10)-1)^(1/(2*N))) ;
[b,a]= u_buttap(N,OmegaC) ;
```

```
%2018-EE-394
%lab dsp
function [db,mag,pha,w] = freqs_m(b,a,wmax);
w = [0:1:500]*wmax/500;
H = freqs(b,a,w);
mag = abs(H);
db = 20*log10( (mag+eps)/max(mag));
pha = angle(H) ;


```

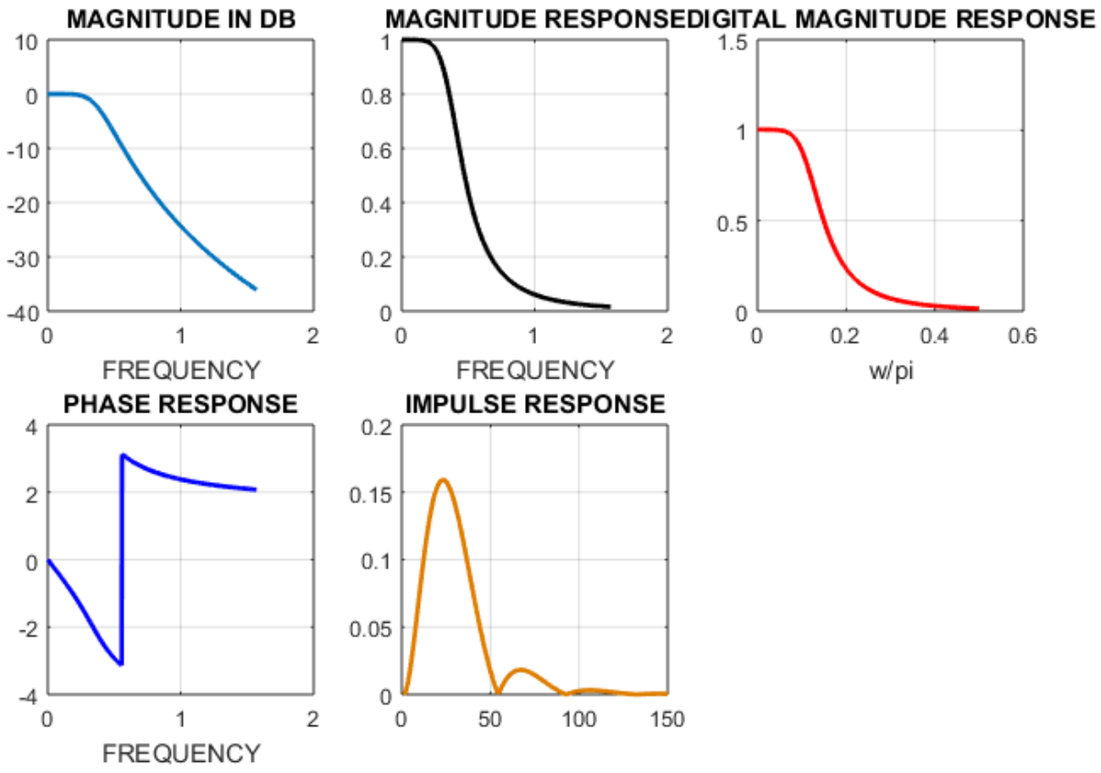
Date: 21_06_2021 **Registration#2018-EE-394**

```
%2018-EE-394
%lab dsp 11
Ripple = 10 ^(-Rp/20); Attn = 10 ^(-As/20);
 [b,a] = afd_butt( Wp , Ws , Rp , As );
  % Calculation of Frequency Response:
 [db,mag,pha,w] = freqs_m(b,a,0.5*pi);
 % Calculation of Impulse response:
 [ha,x,t] = impulse(b,a);
```

```
%2018-EE-394
%lab dsp 11
wp = 0.1*pi;
ws = 0.3*pi;
Rp = 1; % Passband ripple in dB
As = 15;%Stopband attenuation in dB
Ripple = 10 ^(-Rp/20); Attn = 10 ^(-As/20);
% Analog filter design:
[b,a] = afd_butt( wp , ws , Rp,As )
%Butterworth Filter Order = ??
% Calculation of Frequency Response:
[db,mag,pha,w] = freqs_m(b,a,0.5*pi);
% Calculation of Impulse response:
 [ha,x,t] = impulse(b,a);
T = 1; % Set T=1
OmegaP = wp / T;
% Prototype Passband freq
Omegas = ws / T;
% Prototype Stopband freq
% Analog Butterworth Prototype Filter Calculation:
[cs ,ds] = afd_butt(OmegaP,Omegas,Rp,As)
%Impulse Invariance transformation:
[b,a] = imp_invr(cs,ds,T);
[C,B,A] = dir2par(b,a)
z=exp(j.*w)
Hz=((-0.3935+.3670.*z.^-1)./(1-1.5483.*z.^-1+0.674.*z.^-2))+((0.395)./(1-0.6747.*z.^-1))
   subplot(2,3,1);
   plot(w,db);grid on
   title('MAGNITUDE IN DB')
   xlabel('FREQUENCY')
   subplot(2,3,2)
   plot(w,mag);grid on
   title('MAGNITUDE RESPONSE')
   xlabel('FREQUENCY')
   subplot(2,3,3)
   plot(w/pi,abs(Hz));grid
   xlabel('w/pi');title('DIGITAL MAGNITUDE RESPONSE')
   subplot(2,3,4)
   plot(w,pha);grid on
   title('PHASE RESPONSE')
   xlabel('FREQUENCY')
   subplot(2,3,5)
   plot(abs(ha));grid on
   title('IMPULSE RESPONSE')
```

Date: 21_06_2021 **Registration#2018-EE-394**

Simulation:



Conclusion:

In this lab We have created IIR filters using a Butterworth prototype. We created our desired IIR filter by first designing an analogue low pass filter, then performing frequency band transformation and filter transformation. We can observe from the simulation of the butterworth prototype analogue filter that it exhibits a flat magnitude response in the passband and stopband. This may be turned into a digital IIR filter. To preserve the impulse response similar to that of an analogue filter, we employed the impulse invariance approach. We can plainly see from the impulse response of the filter that it does not reach zero at any single point, but rather continues endlessly without reaching zero at some point, indicating that it is the impulse response of an IIR filter.

Date: 25-0602021 **Registration#**2018-EE-394

Name: Hamza Mobeen

University of Engineering & Technology Lahore

Experiment # 12

Title: **Chebyshev Digital IIR Filters**

Equipment Required: Personal computer (PC) with windows operating system and MATLAB software

Theory:

There are three basic prototypes (Butterworth, Chebyshev, Elliptic) of IIR filters are widely used in practice. We have discussed butterworth analog and digital filter in previous lab. In this lab, we will design chebyshev filter and its transformation in digital filters.

There are two types of Chebyshev filters. The Chebyshev-I filters have **equiripple response in the passband,** while the Chebyahev-II filters have **equiripple response in the stopband.** Butterworth filters have monotonic response in both bands. By choosing a filter that has an equiripple rather than a monotonic behavior, we can obtain a lower-order filter. Therefore, Chebyshev filters provide lower order than Butterworth filters for the same specifications

$$|H_a(j\Omega)|^2 = \frac{1}{1+\epsilon^2 T_N^2\left(\frac{\Omega}{\Omega_c}\right)}$$

where N is the order of the filter, ϵ is the passband ripple factor, which is related to Rp, and T_N(X) is the Nth-order Chebyshev polynomial given

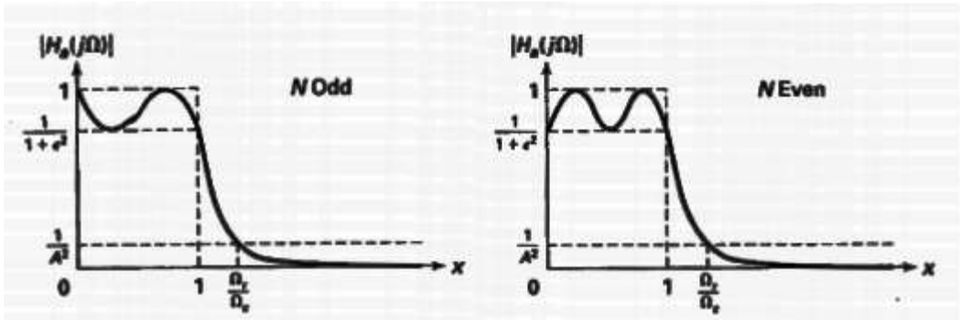
$$T_N(x) = \begin{cases} \cos\left(N\cos^{-1}(x)\right), & 0 \le x \le 1 \\ \cosh\left(\cosh^{-1}(x)\right), & 1 < x < \infty \end{cases} \quad \text{where } x = \frac{\Omega}{\Omega_c}$$

The equiripple response of the Chebyshev filters is due to this polynomial T_N(x). Its key properties are

(a) for 0 < x < 1, T_N(x) oscillates between -1 and 1,

(b) for 1< x < ∞, T_N(x) increases monotonically to ∞.

There are two possible shapes of $|H_a(j\Omega)|^2$,one for N odd and one for N even **as** shown below. Note that X = Ω/Ω_c is the normalized frequency.



Designing Equations:

MATLAB provides a function called [z, p, k] =cheb1ap(N, Rp) to design a *normalized* Chebyshev-I analog prototype filter of order N and pass band ripple Rp and that returns zeros in z array, poles in p array, and the gain value k. Un-normlized Chebyshev-I filter with arbitrary R is achieved by scaling the array p of the normalized filter by R,. Similar to the Butterworth prototype, this filter

has no zeros. In the following function, called u-chb1ap(N, Rp, Omegac), we design an un-normalized Chebyshev-I analog prototype filter that returns Ha*(s)* in the direct form.

$$H_a(s) = \frac{K}{\prod_k (s - p_k)} \qquad \epsilon = \sqrt{10^{0.1R_p} - 1} \quad \text{and} \quad A = 10^{A_s/20}$$

$$\Omega_c = \Omega_p \quad \text{and} \quad \Omega_r = \frac{\Omega_s}{\Omega_p}$$

$$g = \sqrt{(A^2 - 1)/\epsilon^2}$$

$$N = \left\lceil \frac{\log_{10}\left[g + \sqrt{g^2 - 1}\right]}{\log_{10}\left[\Omega_r + \sqrt{\Omega_r^2 - 1}\right]} \right\rceil$$

N = Order of the filter

$\Omega_c = \ Cutoff\ frequency$

$\Omega_p = Passband\ cutoff\ frequency$

$\Omega_s = Stopband\ cutoff\ frequency$

$R_p = Passband\ ripple$

$A_s = stopband\ ripple$

Example 1:

Design a lowpass chebyshev-1 analog and digital filter using to satisfy

Passband cutoff = 0.2π; Passband ripple: = 1dB
Stopband frequency = 0.3π; Stopband ripple: = 16dB

Plot magnitude response.

MATLAB Script

```
function [b,a] = u_chb1ap(N,Rp,Omegac);
% Unnormalized Chebyshev-1 Analog Lowpass Filter Prototype
% ---------------------------------------------------------
% [b,a] = u_chb1ap(N,Rp,Omegac);
% b = numerator polynomial coefficients
% a = denominator polynomial coefficients
% N = Order of the Elliptic Filter
% Rp = Passband Ripple in dB; Rp > 0
% Omegac = Cutoff frequency in radians/sec
%
[z,p,k] = cheb1ap(N,Rp); a = real(poly(p)); aNn = a(N+1);
p = p*Omegac; a = real(poly(p)); aNu = a(N+1);
k = k*aNu/aNn;
b0 = k; B = real(poly(z)); b = k*B;
```

```
function [b,a] = afd_chb1(Wp,Ws,Rp,As);
% Analog Lowpass Filter Design: Chebyshev-1
% -----------------------------------------
% [b,a] = afd_chb1(Wp,Ws,Rp,As);
% b = Numerator coefficients of Ha(s)
% a = Denominator coefficients of Ha(s)
% Wp = Passband edge frequency in rad/sec; Wp > 0
% Ws = Stopband edge frequency in rad/sec; Ws > Wp > 0
% Rp = Passband ripple in +dB; (Rp > 0)
% As = Stopband attenuation in +dB; (As > 0)
%
if Wp <= 0
error('Passband edge must be larger than 0')
end
if Ws <= Wp
error('Stopband edge must be larger than Passband edge')
end
if (Rp <= 0) || (As < 0)
```

```
error('PB ripple and/or SB attenuation ust be larger than 0')
end
ep = sqrt(10^(Rp/10)-1); A = 10^(As/20);
OmegaC = Wp; OmegaR = Ws/Wp; g = sqrt(A*A-1)/ep;
N = ceil(log10(g+sqrt(g*g-1))/log10(OmegaR+sqrt(OmegaR*OmegaR-1)));
fprintf('\n*** Chebyshev-1 Filter Order = %2.0f \n',N)
[b,a]=u_chb1ap(N,Rp,OmegaC);
end
```

```
function [b,a]= imp_invar(c,d,T)
%impulse invariance transformation from analog to digital filter
%b= numerator polynomial in z^(-1) of digital filter
%a= Denominator polynomial in z^(-1) of digital filter
%c= numerator polynomial in s of analog filter
%d= Denominator polynomial in s of analog filter
%T= Sampling(transformation) parameter

[R,p,k]= residue(c,d)
p= exp(p*T);
[b,a]= residuez(R,p,k);
b=real(b');
a=real(a');
end
```

```
function [C,B,A] = dir2par(b,a);
% Dim-form to PWLEL-form conversion
% -----------------------------------
% [C,B,A] = dir2par(b,a)
% C = Polynomial part when length(b) >= length(a)
% B = K by 2 matrix of real coefficients containing bk's
% A = K by 3 matrix of real coefficients containing ak's
% b = numerator polynomial coefficients of DIRECT form
% a = denominator polynomial coefficients of DIRECT form

M = length(b) ; N = length(a);
[r1,p1,C] = residuez(b,a);
p = cplxpair(p1,10000000*eps);
I = cplxcomp(p1 ,p);
r = r1(I);
K = floor(N/2); B = zeros(K,2); A = zeros(K,3);
if K*2 == N; %N even, order of A(z) odd, one factor is first order
for i=1:2:N-2
Brow = r(i:1:i+1,:);
Arow = p(i:1:i+1,:);
[Brow,Arow] = residuez(Brow,Arow,[]);
B(fix((i+1)/2), :) = real(Brow) ;
A(fix((i+1)/2), :) = real(Arow);
end
[Brow,Arow] = residuez(r(N-1) ,p(N-1), []) ;
B(K,:) = [real(Brow) 0]; A(K,:) = [real(Arow) 0];
else
for i=1:2:N-1
Brow = r(i:1:i+1,:);
Arow = p(i:1:i+1,:);
[Brow,Arow] = residuez(Brow, Arow,[] );
B(fix((i+1)/2), :) = real(Brow) ;
A(fix((i+1)/2), :) = real(Arow);
end
end
```

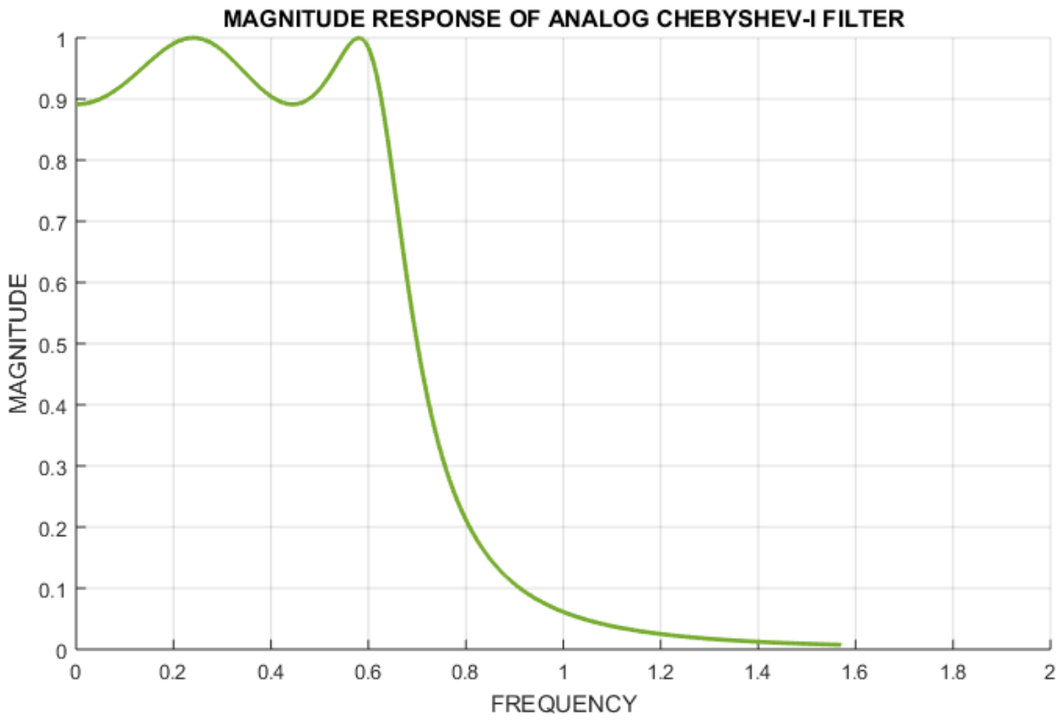
```
function [db,mag,pha,w] = freqs_m(b,a,wmax);
w = [0:1:500]*wmax/500;
H = freqs(b,a,w);
mag = abs(H);
db = 20*log10( (mag+eps)/max(mag));
pha = angle(H) ;
```

Code:

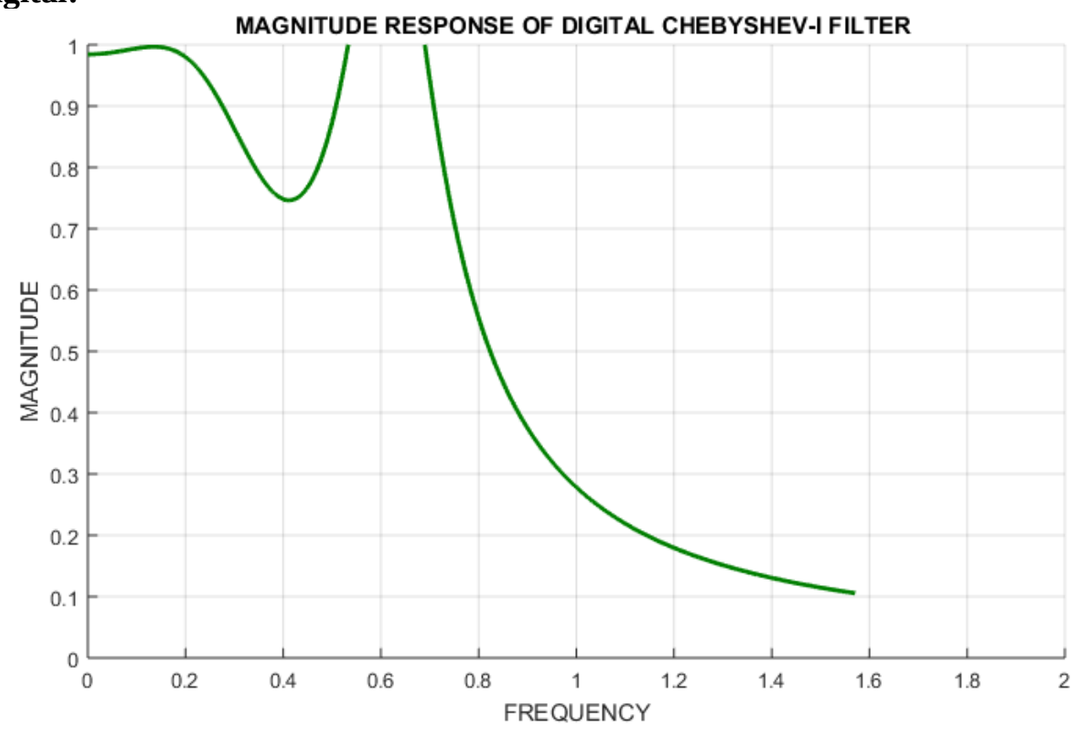
LAB12_EXAMPLE1.m

```
%2018-EE-394
%DSP lab 12
%example 1 Hamza mobeen
wp = 0.2*pi;
ws = 0.3*pi;
Rp = 1;
As = 16;
Ripple = 10 ^ (-Rp/20);
Attn = 10 ^ (-As/20);
% Analog filter design:
[cs,ds] = afd_chb1(wp,ws,Rp,As);
%*** Chebyshev-1 Filter Order = 4
% Calculation of Frequency Response:
[db,mag,pha,w] = freqs_m(cs,ds,0.5*pi);
% Calculation of Impulse response:
[ha,x,t] = impulse(cs,ds);
% Digital filter design:
T = 1; % Set T=1
OmegaP = wp / T; % Prototype Passband freq
Omegas = ws / T; % Prototype Stopband freq
% Analog Filter Calculation:
[cs1 ,ds1] = afd_chb1(OmegaP,Omegas,Rp,As);
% Impulse Invariance transformation:
[b1,a1] = imp_invar(cs1,ds1,T);
[C1,B1,A1] = dir2par1(b1,a1)
figure(1)
hold on
plot(w,mag)
axis([0,2,0 1])
xlabel('FREQUENCY')
ylabel('MAGNITUDE')
title('MAGNITUDE RESPONSE OF ANALOG CHEBYSHEV-I FILTER')
n=exp(j.*w);
f=((-0.0833-0.0246.*(n.^-1))./(1-1.4934.*(n.^-1)+0.8392.*(n.^-2)));
g=((-0.0833+0.0234.*(n.^-1))./(1-1.5658.*(n.^-1)+0.6549.*(n.^-2)));
H=f+g
figure(2)
hold on
plot(w,abs(H))
axis([0,2,0 1])
title('MAGNITUDE RESPONSE OF DIGITAL CHEBYSHEV-I FILTER ')
xlabel('FREQUENCY')
ylabel('MAGNITUDE')
hold off
```

Simulation
Analog:



Digital:



Task:

Design a lowpass chebyshev-*II* digital filter using to satisfy

$\omega_p = 0.2\pi$; $R_p = 1$dB
$\omega_s = 0.3\pi$; $A_s = 16$dB $T = 2$

Plot magnitude response, phase response, magnitude in dB of analog filter and magnitude response of digital filter.

MATLAB Script

```
function [b,a] = u_chb2ap(N,As,Omegac);
% Unnormalized Chebyshev-2 Analog Lowpass Filter Prototype
% [b,a] = u_chb2ap(N,As,Omegac);
% b = numerator polynomial coefficients
% a = denominator polynomial coefficients
% N = Order of the Elliptic Filter
% As = Stopband Ripple in dB; As > 0
% Omegac = Cutoff frequency in radians/sec
[z,p,k] = cheb2ap(N,As);
a = real(poly(p)); aNn = a(N+1);
p = p*Omegac; a = real(poly(p)); aNu = a(N+1);
b = real(poly(z)); M = length(b); bNn = b(M);
z = z*Omegac; b = real(poly(z)); bNu = b(M);
k = k*(aNu*bNn)/(aNn*bNu);
b0 = k; b = k*b;      end
```

```
function [b,a] = afd_chb2(Wp,Ws,Rp,As);
% Analog Lowpass Filter Design: Chebyshev-2
% ----------------------------------------
% [b,a] = afd_chb2(Wp,Ws,Rp,As);
% b = Numerator coefficients of Ha(s)
% a = Denominator coefficients of Ha(s)
% Wp = Passband edge frequency in rad/sec; Wp > 0
% Ws = Stopband edge frequency in rad/sec; Ws > Wp > 0
% Rp = Passband ripple in +dB; (Rp > 0)
% As = Stopband attenuation in +dB; (As > 0)
%
if Wp <= 0
error('Passband edge must be larger than 0')
end
if Ws <= Wp
error('Stopband edge must be larger than Passband edge')
end
if (Rp <= 0) || (As < 0)
error('PB ripple and/or SB attenuation ust be larger than 0')
end
ep = sqrt(10^(Rp/10)-1); A = 10^(As/20);
OmegaC = Wp; OmegaR = Ws/Wp; g = sqrt(A*A-1)/ep;
N = ceil(log10(g+sqrt(g*g-1))/log10(OmegaR+sqrt(OmegaR*OmegaR-1)));
fprintf('\n*** Chebyshev-2 Filter Order = %2.0f \n',N)
[b,a]=u_chb2ap(N,As,Ws);
end
```

```
function [db,mag,pha,w] = freqs_m(b,a,wmax);
w = [0:1:500]*wmax/500;
H = freqs(b,a,w);
mag = abs(H);
db = 20*log10( (mag+eps)/max(mag));
pha = angle(H) ;
```

Code:

lab12task1.m

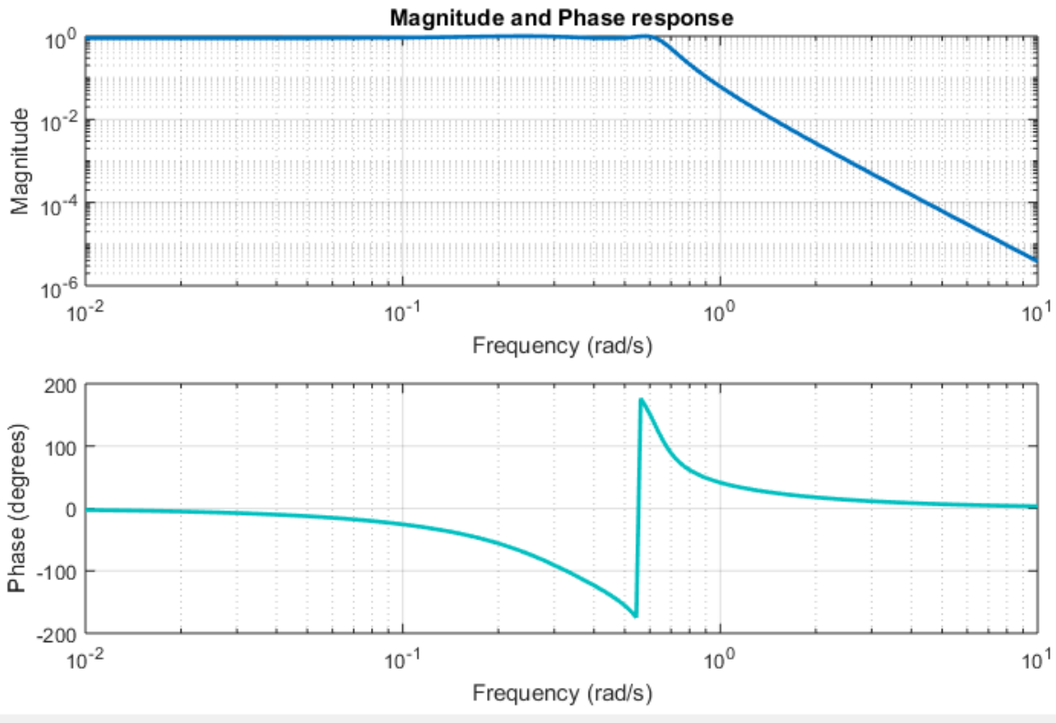
```
%2018-EE-394
%DSPlab 12
%task 1 Hamza mobeen
Wp = 0.2*pi; Ws = 0.3*pi; Rp =1; As= 16;
Ripple=10^(-Rp/20); Attn=10^(-As/20);
%Analog filter design
[b,a] = afd_chb2(Wp,Ws,Rp,As);
%Calculation of second-order sections:
%Calculation of Frequency Response:
[db,mag,pha,w] = freqs_m(b,a,0.5*pi);
%Calculation of Impulse response:
[ha,x,t]=impulse(b,a);
subplot(3,1,1);
plot(w,mag);
xlabel('frequency');ylabel('magnitude')
title('Magnitude response of Analog Chebyshev-II filter')
subplot(3,1,2);
plot(w,pha);
xlabel('frequency');ylabel('phase')
title('Phase response of Analog Chebyshev-II filter')
subplot(3,1,3);
plot(w,db);
xlabel('frequency');ylabel('dB')
title('Magnitude(dB) response of Analog Chebyshev-II filter')
figure()
plot(t,ha)
```

```
xlabel('time');ylabel('response')
title('Impulse response')
figure()
freqs(b,a) %frequency response of filter through Bode plot
title('Magnitude and Phase response');
```

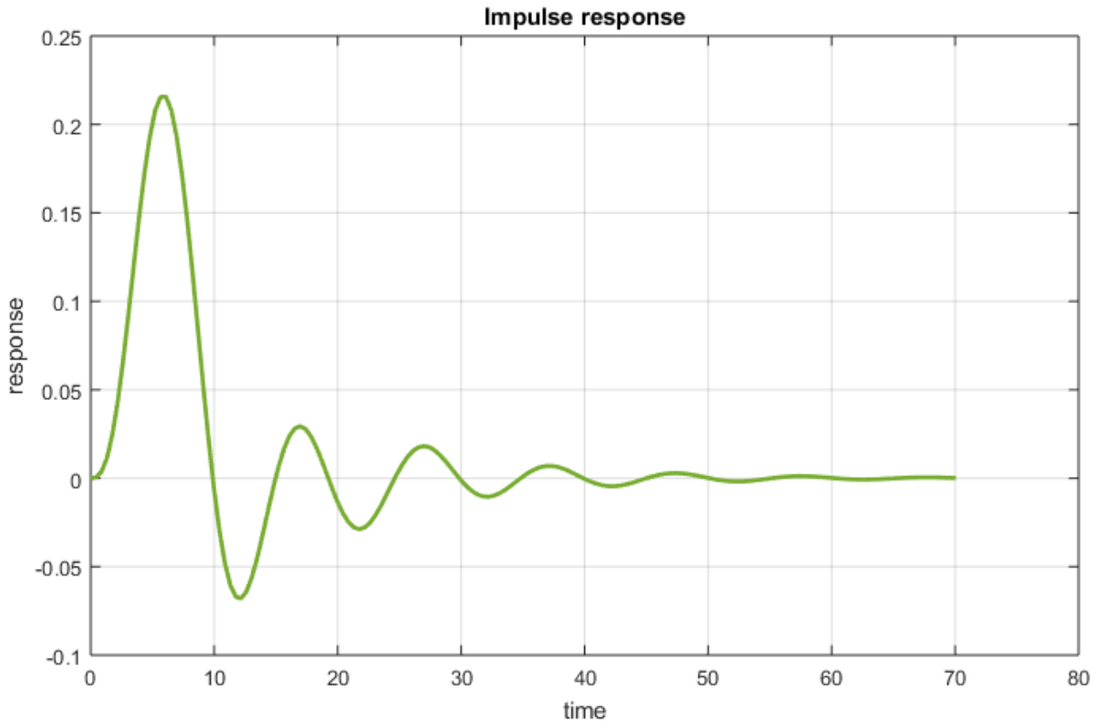
Command Window

Simulation:

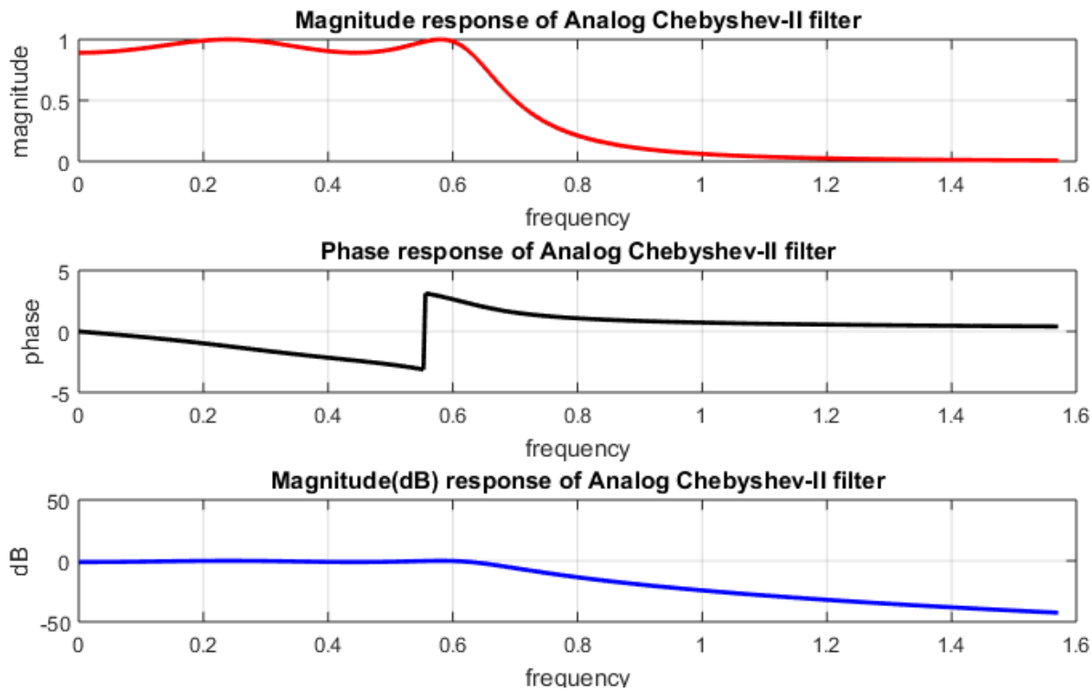
Digital:



Impulse response:



Analog:



- **Compare graphs of analog and digital filters and comment in your own words.**

By examining the analogue and digital filter graphs of tasks, it is revealed that the analogue and digital responses of the Chebyshev-I low pass filter are almost similar when T=1. The analogue and digital answers do not match when the value of T in Chebyshev-II is altered. The response varies independently of impulse invariance. When T=2, the digital response in task01 is half the size of the analogue response.

Conclusion:

In this lab, The analogue Chebyshev filter was observed and learned. The phase, magnitude, and impulse responses of the filter were all shown. The passband frequency of the Chebyshev filter has a 1 dB ripple. The system response had four poles since the filter was built to be of fourth order. The pass band and cut-off frequencies were the same

Name	Hamza Mobeen
Reg-NO	2018-EE-394

Experiment No. 13
Introduction to DSP Starter Kit (DSK) TMS320C6713

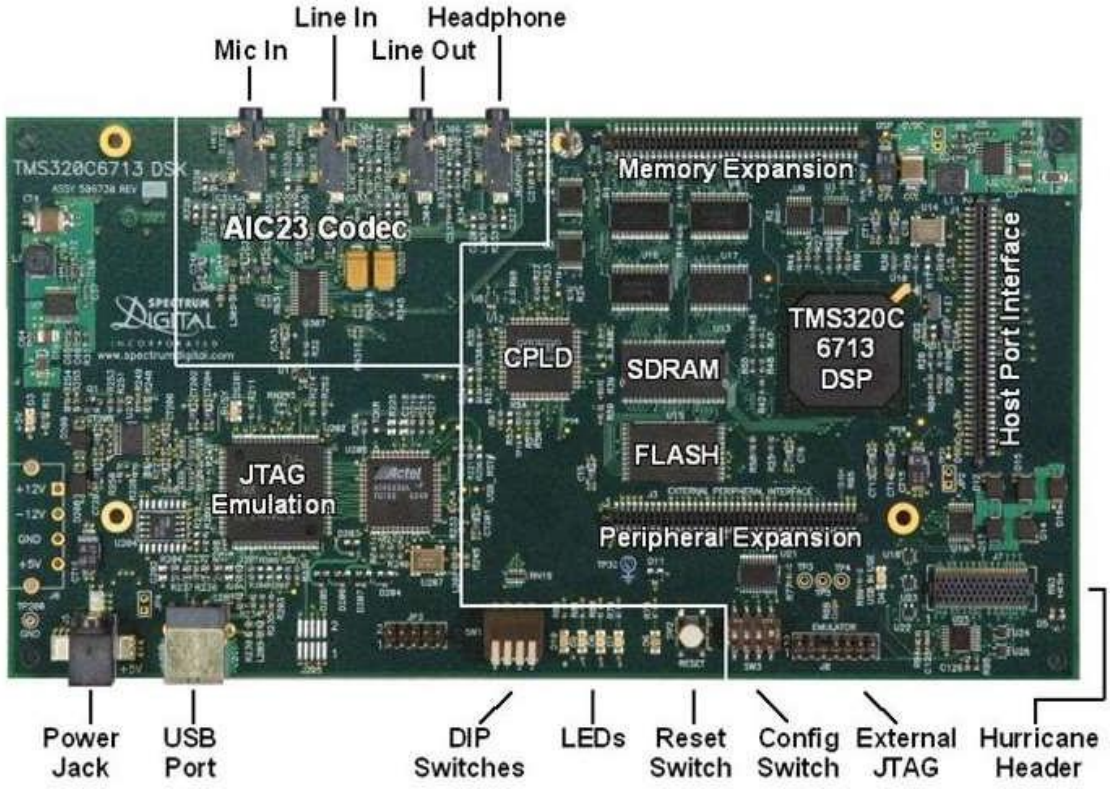
Objective:

The objective of this lab includes learn how to use the DSP Module.

Introduction:

Digital signal processor (DSP) takes real world signals like voice, audio, video, temperature or position that have been digitized and then mathematically manipulate them. A DSP is designed for performing mathematical functions like “add”, “subtract”, “multiply” and “divide” very quickly.

Signals need to be processed so that the information that they contain can be displayed, analyzed, or converted to another type of signal that may be of use. In the real-world, analog products detect signals such as sound, light, temperature or pressure and manipulate them. Convertors such as Analog-to-Digital convertor then take the real-world signal and turn it into the digital format of 1’s and 0’s. from here, the DSP takes over by capturing the digitized information and processing

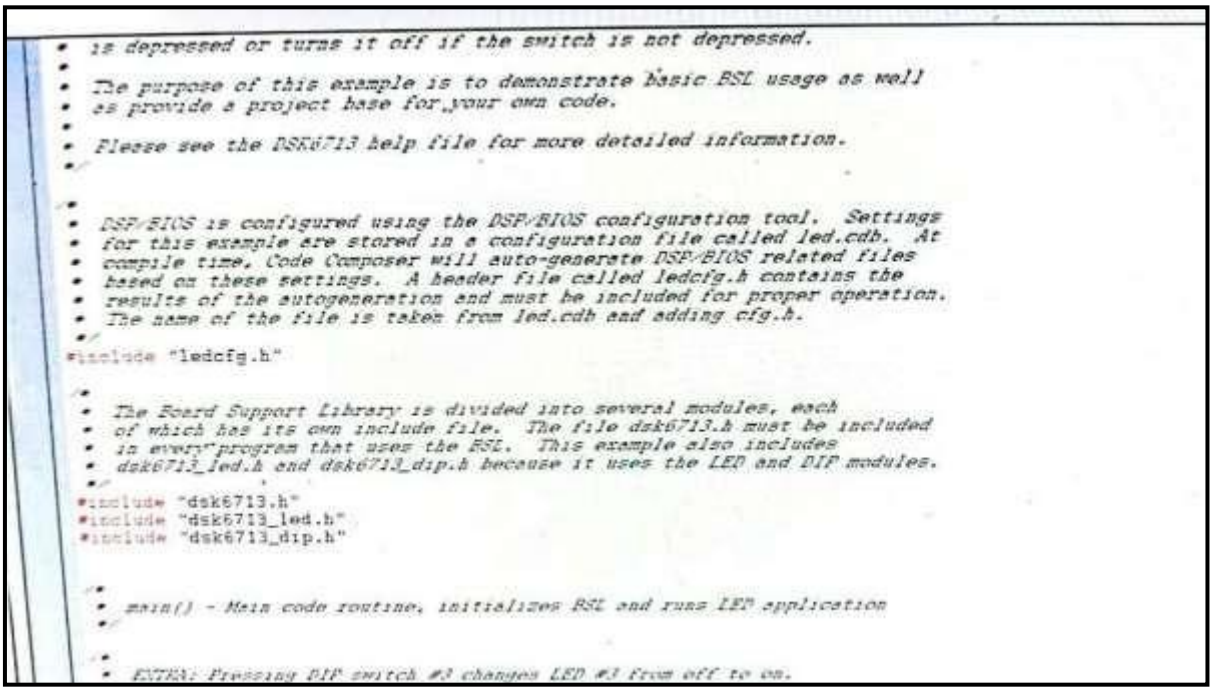


it. It the feeds the digitized information back for use in the real-world. It does this in one of two ways, either digitally or in analog format by going through a Digital-to-Analog convertor. All of this occurs at very high speed.

FEATURES:

1. An AIC23 stereo codec
2. 16MB of synchronous DRAM
3. 512KB of non-volatile flash memory (256KB usable in default configuration)
4. Compatible with wire wrap prototype card
5. Software board configuration through registers implemented in CPLD
6. Configurable boot options
7. Standard expansion connectors for daughter card use
8. JTAG emulation through on board JTAG emulator with USB host interface or external emulator
9. Single voltage power supply (+5V)
10. On board IEEE 1149.1 JTAG connection for optional emulator debug

MATLAB Program: Blinking and Toggling of LED.



```
 *  is depressed or turns it off if the switch is not depressed.
 *
 *  The purpose of this example is to demonstrate basic BSL usage as well
 *  as provide a project base for your own code.
 *
 *  Please see the DSK6713 help file for more detailed information.
 */

/*
 *  DSP/BIOS is configured using the DSP/BIOS configuration tool.  Settings
 *  for this example are stored in a configuration file called led.cdb.  At
 *  compile time, Code Composer will auto-generate DSP/BIOS related files
 *  based on these settings.  A header file called ledcfg.h contains the
 *  results of the autogeneration and must be included for proper operation.
 *  The name of the file is taken from led.cdb and adding cfg.h.
 */
#include "ledcfg.h"

/*
 *  The Board Support Library is divided into several modules, each
 *  of which has its own include file.  The file dsk6713.h must be included
 *  in every program that uses the BSL.  This example also includes
 *  dsk6713_led.h and dsk6713_dip.h because it uses the LED and DIP modules.
 */
#include "dsk6713.h"
#include "dsk6713_led.h"
#include "dsk6713_dip.h"

/*
 *  main() - Main code routine, initializes BSL and runs LED application
 */

/*
 *  EXTRA: Pressing DIP switch #3 changes LED #3 from off to on.
```

Conclusion:

In this lab, we have observed DSK Module and blinked the LED using Kits Different Key elements. DSK Kit comes with a wide range of on-board peripherals and interfaces that allow us to develop of a variety of signal processing. This kit is the special kit for Digital Signal Processing. The applicationsof this kit are

1. DSP Evaluation
2. Algorithm Development
3. Benchmarking