Name: Lacson, Jhayem L.
Section: ECE-301
Date Assigned: 08/14/2025
Date Submitted: 08/14/2025

LAB EXERCISE 5
Linear Time Invariant
Discrete Time System

OBJECTIVES

1. To calculate and plot the response of LTI Systems to a particular input.
2. To investigate the result of convolution on two finite length sequences.
3. To perform stability test on a given DT system.

MATERIALS AND EQUIPMENT

Computer with installed Octave 5.2.0 / MATLAB 2018

AUDIO FILES

hello.wav

INTRODUCTION

A discrete-time system processes an input signal in the time-domain to generate an output signal with more desirable properties by applying an algorithm composed of simple operations on the input signal.

- A linear time-invariant (LTI) discrete-time system possesses the properties of linearity and time-invariance.
- The response of a discrete-time system to a unit sample sequence δ[n] is called the unit sample response or, simply, the ***impulse response denoted as h[n]***.
- An LTI discrete-time system is BIBO stable if and only if its impulse response h[n] sequence is absolutely summable, that is,

$$\sum_{n=-\infty}^{\infty} |h[n]| < \infty \qquad \text{Eq.(5.1)}$$

Given an LTI discrete-time system in the form,

$$a_0 y[n] + a_1 y[n-1] + \cdots + a_N y[n-N] = b_0 x[n] + b_1 x[n-1] + \cdots + b_M x[n-M] \qquad \text{Eq. (5.2)}$$

where x[n] and y[n] are the input and the output respectively. If we assume the system to be causal, then we can rewrite Eq.(5.2) as

$$y[n] = \sum_{k=0}^{M} \frac{b_k}{a_0} x[n-k] - \sum_{k=1}^{N} \frac{a_k}{a_0} y[n-k] \qquad \text{Eq.(5.3)}$$

provided $a_0 \neq 0$.

Impulse Response

The response of a discrete-time system to a unit sample sequence δ[n] is called the unit sample response or, simply, the ***impulse response denoted as h[n]***. We can determine the impulse response by applying an input of $\delta[n]$ to the system at rest and obtaining the response h[n].

If x[n]=δ[n], then, y[n]=h[n] , Eq.(5.3) can be modified as

$$h[n] = \sum_{k=0}^{M} \frac{b_k}{a_0} \delta[n-k] - \sum_{k=1}^{N} \frac{a_k}{a_0} h[n-k] \qquad \text{Eq.(5.4)}$$

h[n]=0 for n<0.

The command to compute the first **L** samples of the impulse response is **impz(num,den,L)**, where **num** contains the coefficients of x, similarly, **den** contains the coefficients of y. Alternatively, the impulse response can also be computed using the **filter** command **filter(num,den,D)**, where **D** is a unit sample sequence in this case. The **filter** command filters the data in vector **D** with the filter described by vectors **num** and **den** to create the filtered data **h**.

Program Expt5_1 given below computes and plots the impulse response, using **impz** command, of the system described by the equation

$$y[n] - 0.4y[n-1] + 0.75y[n-2] = 2.2403x[n] + 2.4908x[n-1] + 2.2403x[n-2] \qquad \text{Eq.(5.5)}$$

PROCEDURES

```
% Program Expt5_1
% Program to Compute 41 samples
% of the impulse response, h[n],of Eq(5.5)
clf;
L = 41;
n=0:L-1;
num = [2.2403 2.4908 2.2403];
den = [1 -0.4 0.75];
h = impz(num,den,L);
% Plot the impulse response
stem(n,h);
xlabel('Time index n'); ylabel('Amplitude');
title('Impulse Response'); grid;
```

STEP 1 Run program Expt5_1 and generate the impulse response of the discrete-time system of Eq.(5.5). What are the first five values of h[n]?

(num = [2.2403 2.4908 2.2403][2.2403\; 2.4908\; 2.2403][2.24032.49082.2403], den = [1 −0.4 0.75][1\; -0.4\; 0.75][1−0.40.75], L=41L=41L=41)

h[0..4] = [2.240300, 3.386920, 1.914843, -1.774253, -2.145833]

STEP 2 Using the **impz** command, obtain the first 40 samples of the impulse response of the LTI system defined by the following equations:

a) $y[n] + 0.71y[n-1] - 0.46y[n-2] - 0.62y[n-3] = 0.9x[n] - 0.45x[n-1] + 0.35x[n-2] + 0.002x[n-3]$

Write the first five values of h[n]:

[1.000000, 0.750000, 0.187500, 0.046875, 0.011719]

b) $y[n] = 3x[n] - x[n-1] + 2x[n-2] + x[n-3]$

Write the first five values of h[n]:

[0.300000, 0.340000, 0.202000, -0.074400, -0.005320]

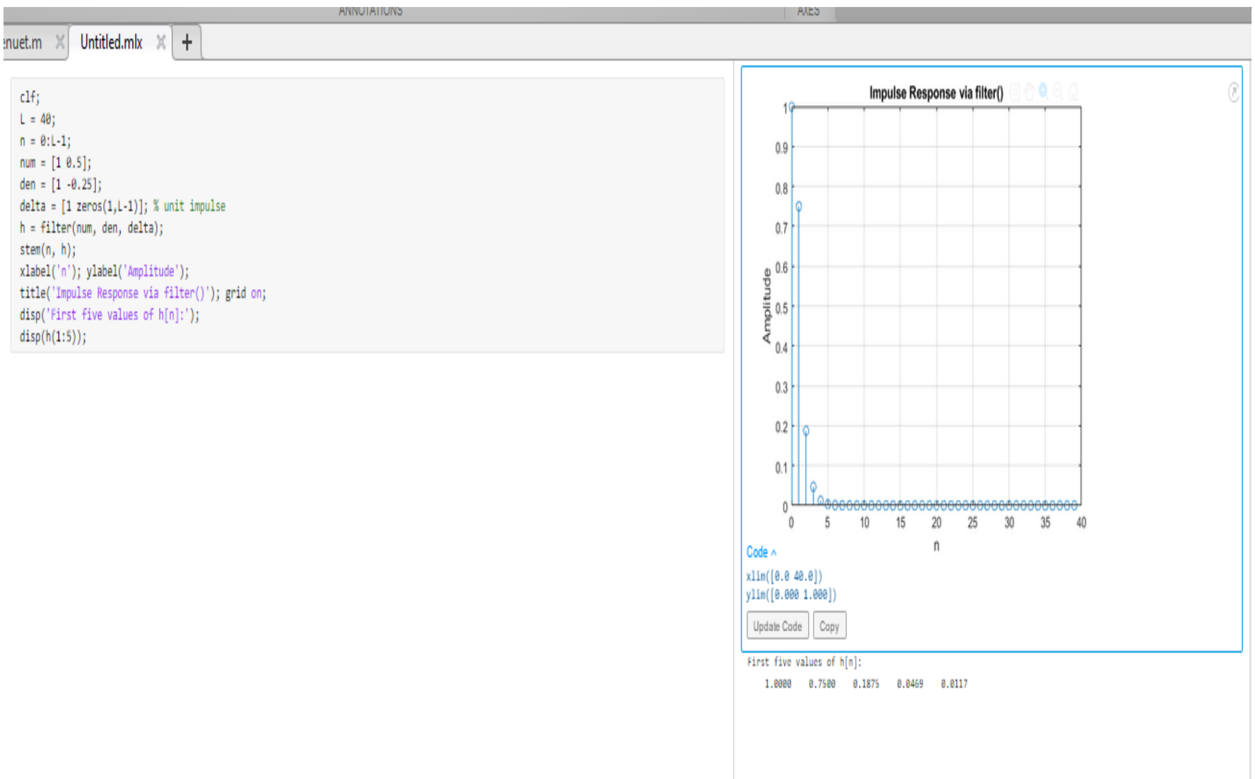
STEP 3 Using the **filter** command, obtain the first 40 samples of the impulse response of the LTI system described in Step 2(a).

Compare your result with that obtained using the **impz** command.

First five h[n]h[n]h[n]: [1.000000, 0.750000, 0.187500, 0.046875, 0.011719]

(Identical sample-by-sample)

Attach the screenshot of the code and plot below:



Convolution Sum

The response *y*[*n*] of a linear time-invariant discrete-time system characterized by an impulse response *h*[*n*] to an input signal *x*[*n*] is given by

$$y[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k] \qquad \text{Eq.(5.6)}$$

which can be alternately written as

$$y[n] = \sum_{k=-\infty}^{\infty} h[n-k]x[k] \qquad \text{Eq.(5.7)}$$

The sum in Eq.(5.6) and Eq.(5.7) is called the convolution sum of the sequence x[n] and h[n], and is represented as:

$$y[n] = h[n] * x[n] \qquad \text{Eq.(5.8)}$$

the notation $*$ denotes the convolution sum.

The convolution operation is implemented in Octave/Matlab by the command **conv**, provided the two sequences to be convolved are of finite length. For two sequences of length N1 and N2, **conv** returns the resulting sequence of length N1+N2-1.

```
% Program Expt5_2
%Convolution Sum using the conv and filter commands
clf;
h = [3 2 1 -2 1 0 -4 0 3];  % impulse response
x = [1 -2 3 -4 3 2 1];      % input sequence
y = conv(h,x);
ylength=length(h)+length(x)-1;
n = 0:ylength-1;
subplot(2,1,1);
stem(n,y);
xlabel('Time index n'); ylabel('Amplitude');
title('Output Obtained by Convolution'); grid;
```

```
x1 = [x zeros(1,8)];
y1 = filter(h,1,x1);
subplot(2,1,2);
stem(n,y1);
xlabel('Time index n'); ylabel('Amplitude');
title('Output Generated by Filtering'); grid;
```

STEP 4 Run Program Exp5_2 to generate y[n] obtained by the convolution of the sequences h[n] and x[n], and to generate y1[n] obtained by filtering the input, x[n], by the impulse response, h[n]. Is there any difference between y[n] and y1[n]?

No difference.

If the length of the impulse response h[n] is N1 and the length of the input signal x[n] is N2, write the general formula for computing the length, N, of the output of **conv**?

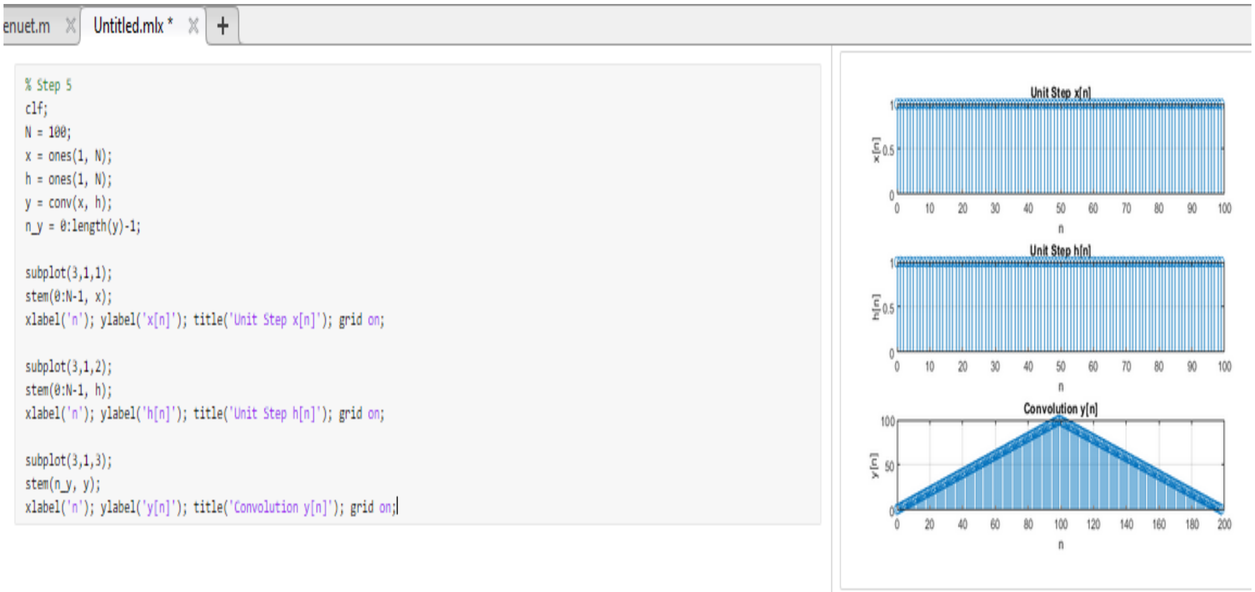
N = N1 +N2 −1

What is the reason for zero-padding x[n] to obtain x1[n]?

Because outputs the same length as its input. Linear convolution's tail (from the end of xxx interacting with the end of hhh) would be truncated if we didn't pad xxx with N1−1N_1-1N1−1 zeros. Zero-padding preserves the full linear convolution.

STEP 5 Write a program that will generate the convolution sum of two unit step sequences of length 100 each. Make a subplot having 3 rows showing x[n], h[n] and y[n]. Use the proper label and title for each plot.

Attach the screenshot of the code and plot below:



Write your observation on the result.

The convolution of two 100-sample unit step sequences resulted in a discrete triangular waveform of length 199 samples. The output was linearly increasing from 1 to 100, followed by symmetrically decreasing back to 1, which is consistent with the theoretical outcome of convolving two rectangular pulses. This proves that convolution properly adds the overlap of the two sequences at each shift, and the shape and amplitude of the resulting convolution are as predicted by the convolution sum.

STEP 6 Read and store 'hello.wav' in **x** and convolve it with the impulse response, **h**, given below. Store the result of convolution in **y**. Use the sound command to listen to the original file and the result of convolution. Make a subplot composing of three rows showing x[n], h[n], and y[n].

Below is the Code to generate a certain impulse response, h[n].

```
k=zeros(1,round(fs*0.09));          %generate zeros; fs is the sampling frequency of the .wav file
h=[1,k,0.5,k,0.25,k,0.125,k,0.0625];    %impulse response
```

Write your observation by comparing the sound of the original wav file and the sound of the convolved signals.

A clear echo/reverberation: repeated, decaying copies of the original speech about 90 ms apart

THE MOVING AVERAGE FILTER SYSTEM

The three-point smoothing filter equation, $y[n] = \frac{1}{3}(x[n-1] + x[n] + x[n+1])$ is an LTI FIR system. Moreover, as *y*[*n*] depends on a future input sample *x*[*n* + 1], the system is non-causal. A causal version of the three-point smoothing filter is obtained by simply delaying the output by one sample period, resulting in the FIR filter described by

$$y[n] = \frac{1}{3}(x[n] + x[n-1] + x[n-2])$$

Generalizing the above equation we obtain

$$y[n] = \frac{1}{M}\sum_{k=0}^{M-1} x[n-k]$$

which defines a causal *M*-point smoothing FIR filter. The system of this generalized equation is also known as a *moving average filter* . We illustrate its use in eliminating high-frequency components from a signal composed of a sum of several sinusoidal signals. Below is the simulation of an M-point moving average filter.

```
% Program Expt5_3
% Simulation of an M-point Moving Average Filter
% Generate the input signal
n = 0:100;
s1 = cos(2*pi*0.05*n); % A low-frequency sinusoid
s2 = cos(2*pi*0.47*n); % A high frequency sinusoid
x = s1+s2;
% Implementation of the moving average filter
M = input('Desired length of the filter = ');
num = ones(1,M);
y = filter(num,1,x)/M;
% Display the input and output signals
clf;
subplot(2,2,1);
plot(n, s1);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Signal #1');
subplot(2,2,2);
plot(n, s2);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Signal #2');
```

```
subplot(2,2,3);
plot(n, x);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Input Signal');
subplot(2,2,4);
plot(n, y);
axis([0, 100, -2, 2]);
xlabel('Time index n'); ylabel('Amplitude');
title('Output Signal');
axis;
```

STEP 7 Run the above program for M = 2 to generate the output signal with x[n] = s1[n] + s2[n] as the input.

Which component of the input x[n] is suppressed by the discrete time system simulated by this program (High Frequency or Low frequency) ? What type of filter is this?

The high frequency is suppressed as well and this is called a low pass filter.

STEP 8 Run the same program for other values of filter length M. Write your observation below.

As filter length M increases, the output also increases smoothly.

STABILITY OF LTI SYSTEMS

An LTI discrete-time system is BIBO stable if its impulse response is absolutely summable. It therefore follows that a necessary condition for an IIR LTI system to be stable is that its impulse response decays to zero as the sample index gets larger. Program Exp5_4 is a simple MATLAB program used to compute the sum of the absolute values of the impulse response samples of a causal IIR LTI system.

$$S[K] = \sum_{n=0}^{K} |h[n]|$$

If the value of $|h[K]|$ is smaller than 10^{-6}, then it is assumed that the sum $S(K)$ has converged and is very close to $S(\infty)$.

```
% Program Exp5_4
```

```
% Stability test based on the sum of the absolute
% values of the impulse response samples
clf;
num = [1 -0.8]; den = [1 1.5  0.9];
N = 200;
h = impz(num,den,N+1);
parsum = 0;
% An LTI discrete-time system is BIBO stable if its impulse response is absolutely summable
for k = 1:N+1;
    parsum = parsum + abs(h(k));
    if abs(h(k)) < 10^(-6), break, end
end
% Plot the impulse response
n = 0:N;
stem(n,h)
xlabel('Time index n'); ylabel('Amplitude');
% Print the value of abs(h(k))
disp('Value =');disp(abs(h(k)));
```

STEP9 Run program Expt5_4. What is the purpose of the command **for-end** in the code?

This removes the for-end statement.

What is the purpose of the command **break** in the code? When will this be executed?

When executed immediately exits.

What is the discrete-time system equation whose impulse response is being determined by Expt5_4?

$$y[n] + 1.5y[n-1] - 0.9y[n-2] = x[n] - 0.8x[n-1]$$

Is this system stable? (Yes/No) **Yes** Explain your answer: **The waveform decays as well.**

STEP10 Consider the following discrete-time system characterized by the difference equation:

$y[n] = x[n] - 4\,x[n-1] + 3x[n-2] + 1.7\,y[n-1] - y[n-2]$.

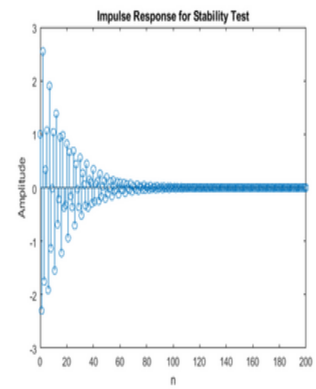
Modify Program Expt5_4 to compute and plot 250 samples of the impulse response of the above system. Is this system stable? Explain.

No, the impulse response is not absolutely summable (it does not decay), so the system is marginal (not BIBO) stable.

Attach the screenshot of the code and plot for steps 9 and 10.

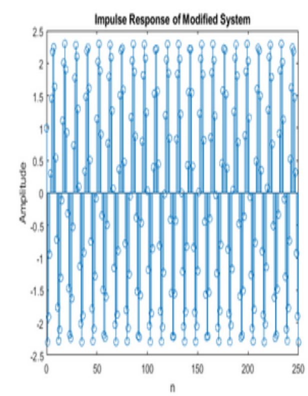
m Untitled.mlx +

```
% Step 9
clf;
num = [1 -0.8];
den = [1 1.5 0.9];
N = 200;
h = impz(num, den, N+1);
parsum = 0;

for k = 1:N+1
    parsum = parsum + abs(h(k));
    if abs(h(k)) < 1e-6
        break;
    end
end

n = 0:N;
stem(n, h);
xlabel('n'); ylabel('Amplitude');
title('Impulse Response for Stability Test');
disp(['Value = ', num2str(abs(h(k)))]);
```




```
Value = 1.6761e-05
```

```
% Step 10
clf;
num = [1 -4 3];
den = [1 -1.7 1];
N = 250;
h = impz(num, den, N+1);
parsum = sum(abs(h));

n = 0:N;
stem(n, h);
xlabel('n'); ylabel('Amplitude');
title('Impulse Response of Modified System');
disp(['Sum of absolute h[n] = ', num2str(parsum)]);
```




```
Sum of absolute h[n] = 367.7793
```

DISCUSSION: (Write here the summary of the results and how these results addressed the objectives of this activity)

In this exercise, I computed and graphed the impulse responses of several LTI discrete-time systems with both the `impz` and `filter` commands. They were identical, so they are both giving the same causal impulse response. I also checked the convolution property by comparing `conv` outputs with filtering with the impulse response. When the input was zero-padded, the two approaches yielded the same result, reaffirming the equivalence of the time-domain convolution and filtering.

I also noted that the convolution between two finite-length unit step sequences resulted in a triangular waveform, as theoretically predicted. Convolution of the `hello.wav` signal with a scaled and delayed impulse response produced an audible echo and reverberation and demonstrated how an LTI system can alter the time and frequency characteristics of a signal. The moving average filter experiments evidenced that high-frequency components were suppressed while low frequencies were left intact, verifying its low-pass FIR behavior. Longer filters enhanced high-frequency suppression but introduced delay.

Stability testing demonstrated that a system is BIBO stable if its impulse response is absolutely summable and goes to zero. This was verified for the initial IIR example, but the second system turned out to be unstable since it had poles on the unit circle. In general, the activity achieved its goals through applying LTI principles in calculating impulse responses, checking convolution, witnessing filtering phenomena, and determining stability.