BILKENT UNIVERSITY
DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

EEE 321
Signals and Systems
Lab Assignment 3

due: 22.03.2024, Friday at 23:55 on Moodle

An integrator is a fundamental system component used in modeling many physical systems. In this lab assignment, you will work with ideal and non-ideal (imperfect) integrators and implement them on MATLAB to determine their system properties and analyze their integration performance.

Please work on this assignment before coming to the laboratory. At the end of the laboratory session, you must show your completed work for all parts. After the lab session, you will have more time to format your completed work as a report and submit it on Moodle. Some parts will be performed by hand, and others will be done using MATLAB. Please address all the questions asked in the assignment and include all your codes as text and derivations for all parts. Before the submission deadline, make sure to upload your work as a readable, well-formatted single PDF file. Note that the MATLAB codes will be tested on MATLAB R2023a. Ensure your code does not raise an error or a warning in earlier versions. Do NOT forget to add proper captions, axis labels, and titles for any plot you provide.

Part 1

1.1 Ideal (Perfect) Integrator

The input-output relationship of the ideal integrator is defined by the following running integral:

$$y(t) = \int_{-\infty}^{t} x(\tau)\, d\tau \tag{1}$$

where t denotes the present (current) time. The ideal integrator takes the perfect integral of the input signal $x(t)$. Derive the impulse response $h(t)$ and the unit-step response $s(t)$ of the ideal integrator and plot them with respect to time.

Test the linearity, time-invariance, causality, memory, and bounded-input bounded-output (BIBO) stability of the ideal integrator. Whenever possible and if applicable, use the im-

pulse response of the system to test for these system properties. Which one of these system properties appears to be problematic?

You may do all of Part 1.1 by hand.

1.2 Another System

Now, let us introduce a second system, which is a continuous-time linear time-invariant system whose impulse response is a right-sided exponential $h(t) = e^{-at}u(t)$. Here, a is a real constant greater than zero. Find the unit-step response $s(t)$ of this system. Repeat the tests for the system properties given in Part 1.1. Whenever possible and if applicable, use the given impulse response of the system to test for these system properties. Does this system have any advantageous or desirable property (or properties) compared to the ideal integrator introduced back in Part 1.1?

You may do all of Part 1.2 by hand.

1.3 Discretization of the Two Systems

Discretize the ideal integrator introduced in Part 1.1 using a sampling period of $T_s = 0.01$ s. Write the input-output equation of the discrete-time system that you obtained. What is this system called? Implement this discrete-time system in MATLAB and generate its impulse response $h[n]$ and the unit-step response $s[n]$.

Repeat these steps for the second system introduced in Part 1.2.

Part 2

In this part, you will be investigating the BIBO stability of the two discrete-time systems that you have obtained in Part 1.3 numerically, using MATLAB.

The necessary and sufficient condition for a discrete-time LTI system to be bounded-input bounded-output (BIBO) stable is that the impulse response must be absolute summable:

$$\sum_{k=-\infty}^{\infty} |h[k]| < \infty \tag{2}$$

Comment on your expectation of the BIBO stability of the two systems.

Write a MATLAB function named **"sumElements"** where you implement Eq.(2). Although the expression in Eq.(2) corresponds to an infinite sum, we can only sum a finite number of elements in MATLAB. Therefore, we need to truncate the summation by changing its lower and upper limits to $-N$ and N, respectively. The summation operation should be performed for several different values of N. Therefore, the function should get an input that indicates the set of N values that will be used for the summation operation. Each result of the summation operation for different values of N will be placed in an array named **sum_array**. This array will be the output of your function. Your function should look like this: **function [sum_array] = sumElements(h,N_range)** where

- **h** represents the impulse response to be processed
- **N_range** represents the array of N values for which the summation is conducted
- **sum_array** represents an array whose ith element is the summation of the elements of the given impulse response between the indices $-i$ and i

In your code, set **N_range**= [100 : 300 : 10000] and call the function **sumElements** for five different a parameter values of $0, 0.05, 0.10, 0.25$, and 0.5 and save the output arrays. Note that $a = 0$ corresponds to the discrete version of the ideal integrator. Then, using MATLAB's **subplot** function, plot these five arrays with appropriate labels showing the corresponding a values. The x axes of your plots should display the N values and the y axes should show the summation values corresponding to each of these N values.

Part 3

In this part, you will investigate the difference between the outputs of the two systems that we have introduced earlier, in terms of the a parameter.

Let two input sequences $x_1[n]$ and $x_2[n]$ be given as

$$\begin{aligned} x_1[n] &= 8\,(u[n] - u[n-4]) - 4\,(u[n-4] - u[n-13]) \\ x_2[n] &= (0.3)^n\, u[n] \end{aligned}$$

Let us consider the two discrete-time systems that we have investigated in the previous part. Recall that the second system was implemented for four different values of its a parameter. For each of these five cases, that is, $a = 0, 0.05, 0.10, 0.25$, and 0.5, plot (by using the **stem** command) the output sequences $y_1[n]$ and $y_2[n]$ when the sequences $x_1[n]$ and $x_2[n]$ are given as input to the system, respectively. Let the outputs of the system corresponding to $a = 0$ be denoted as $y_1^{\text{ideal}}[n]$ and $y_2^{\text{ideal}}[n]$ since this value of a corrsponds to the discrete version of ideal integrator.

The difference between the outputs for a=0 (denoted as $y_1^{\text{ideal}}[n]$ and $y_2^{\text{ideal}}[n]$) and the outputs for the remaining four cases of a (denoted as $y_1[n]$ and $y_2[n]$) can be found by

$$\mathscr{E}_1[n] = \left| y_1^{\text{ideal}}[n] - y_1[n] \right| \tag{3}$$

$$\mathscr{E}_2[n] = \left| y_2^{\text{ideal}}[n] - y_2[n] \right| \tag{4}$$

Calculate the differences between the output for the system where a=0 ($y^{\text{ideal}}[n]$s) and all other four cases of a ($y[n]$s) for each of the two output sequences and plot the results.

Part 4

4.1 First- and Second-Order Differentiation

The first-order derivative of the signal $x(t)$ can be obtained by taking the following limits:

$$\begin{aligned} y(t) = \frac{dx(t)}{dt} &= \lim_{\Delta t \to 0} \frac{x(t+\Delta t) - x(t)}{\Delta t} \qquad \text{(forward)} \\ y(t) = \frac{dx(t)}{dt} &= \lim_{\Delta t \to 0} \frac{x(t) - x(t-\Delta t)}{\Delta t} \qquad \text{(backward)} \end{aligned}$$

It is possible to approximate the first-order derivative over a short time interval Δt using forward and backward approximation as follows:

$$\begin{aligned} y(t) = \frac{dx(t)}{dt} &\approx \frac{x(t+\Delta t) - x(t)}{\Delta t} \qquad \text{(forward)} \\ y(t) = \frac{dx(t)}{dt} &\approx \frac{x(t) - x(t-\Delta t)}{\Delta t} \qquad \text{(backward)} \end{aligned}$$

While forward approximation is not causal, the backward approximation is causal. We can discretize these expressions by setting Δt equal to one unit of time and replacing t with n. As a result, we obtain the following respective first difference equations:

$$\begin{aligned} y_{\text{fwd}}[n] &= x[n+1] - x[n] \\ y_{\text{bwd}}[n] &= x[n] - x[n-1] \end{aligned}$$

With this background, derive the input-output relationship of a discrete-time system that approximates second-order differentiation. Use backward approximation to obtain the difference equation of the system. Determine if this system is causal, has memory, and BIBO stable. Is it a finite impulse response (FIR) or infinite impulse response (IIR) system?

Implement the second-order difference equation that you have obtained using MATLAB. Derive the impulse response of the system manually and plot it. Then, obtain the impulse response on MATLAB by giving the system a discrete impulse sequence as input. Plot the impulse response and compare it with the function you have derived by hand. Investigate the BIBO stability of the system using the **sumElements** function that you have written. Take $N = 0:5$ and plot the output array using **stem** command. The x axis of your plot should display the N values and the y axis should show the summation values corresponding to each of these N values.

For this system, plot (by using the **stem** command) the output sequences $y_1[n]$ and $y_2[n]$ for the input sequences $x_1[n]$ and $x_2[n]$, respectively.

4.2 Invertibility of Second-Order Difference

Is the system you found in Part 4.1 invertible? If so, explain why and derive the inverse system. If not, discuss why. If the inverse system exists, do you recognize it? What is it called? Also, plot the output sequences $y_1[n]$ and $y_2[n]$ for the input sequences $x_1[n]$ and

$x_2[n]$, respectively. Then, determine whether this system is causal or not and whether it has memory or not. Convolve the systems obtained in Parts 4.1 and 4.2 and observe which function you get. Comment on this. (**Hint:** To obtain the inverse system, you may use the fact that $h[n] * h^{-1}[n] = \delta[n]$.)

5 Final Remarks

Throughout this assignment, you are **NOT** allowed to use symbolic operations in MATLAB. Submit the results of your own work in the form of a well-documented lab report on Moodle. Borrowing full or partial code from your peers or elsewhere is **NOT** allowed and will be punished. The axes of all plots should be scaled and labeled. To modify the styles of the plots, add labels, and scale the plots, use only MATLAB commands; do **NOT** use the GUI of the figure windows. When your program is executed, the figures must appear exactly the same as you provide in your solution. You need to write your MATLAB codes not only correctly but efficiently as well. Please include all evidence (plots, screen dumps, MATLAB codes, MATLAB command window print-outs, etc.) as needed in your report. Append your MATLAB code at the end of your assignment as text, not as an image, and do **NOT** upload it separately. You can use the "Publish" menu of MATLAB to generate a PDF file from your codes and their outputs and append it to the end of your report. If you do this, please also indicate the part that the code corresponds to with a label. Typing your report instead of handwriting some parts will be better. If you decide to write some parts by hand, please use plain white paper. Please do not upload any photos/images of your report. Your complete report should be uploaded on Moodle as a single good-quality PDF file by the given deadline. Please try to upload several hours before the deadline to avoid last-minute problems that may cause you to miss the deadline. Please DO **NOT** submit files by e-mail or as hard copies.

Please name the PDF file you submit on Moodle as follows using only lower-case English characters for your first name, middle name (if any), and last name. Please use your full name as it appears on the Bilkent system.

LAB3_firstname_middle name_lastname.pdf
filename example for Ayşenur Çiğdem Sürücü:
LAB2_aysenur_cigdem_surucu.pdf