ECE438 - Laboratory 2:
Discrete-Time Systems
By Prof. Charles Bouman, Prof. Mireille Boutin and Wenwei Zhou
Fall 2019

1   Introduction

A discrete-time system is anything that takes a discrete-time signal as input and generates a discrete-time signal as output.[1] The concept of a system is very general. It may be used to model the response of an audio equalizer or the performance of the US economy.

In electrical engineering, *continuous-time* signals are usually processed by electrical circuits described by differential equations. For example, any circuit of resistors, capacitors and inductors can be analyzed using mesh analysis to yield a system of differential equations. The voltages and currents in the circuit may then be computed by solving the equations.

The processing of *discrete-time* signals is performed by discrete-time systems. Similar to the continuous-time case, we may represent a discrete-time system either by a set of difference equations or by a block diagram of its implementation. For example, consider the following difference equation.

$$y[n] = y[n-1] - 2x[n] + 3x[n-1] \tag{1}$$

This equation represents a discrete-time *system*. It operates on the input signal $x[n]$ to produce the output signal $y[n]$. This system may also be defined by a system diagram as in Figure 1.

Mathematically, we use the notation $y = S[x]$ to denote a discrete-time system S with input signal $x[n]$ and output signal $y[n]$. Notice that the input and output to the system are the complete signals for **all** time n. This is important since the output at a particular time can be a function of past, present and future values of $x[n]$.

It is usually quite straightforward to write a computer program to implement a discrete-time system from its difference equation. In fact, programmable computers are one of the easiest and most cost-effective ways of implementing discrete-time systems.

While Equation (1) is an example of a linear time-invariant system, other discrete-time systems may be nonlinear and/or time varying. In order to understand discrete-time systems,

---

Questions or comments concerning this laboratory should be directed to Prof. Mireille Boutin, School of Electrical and Computer Engineering, Purdue University, West Lafayette IN 47907

[1]A more general *behavioral* view of systems is anything that imposes constraints on a set of signals.
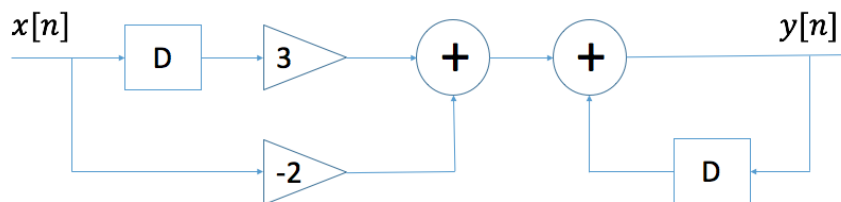
Figure 1: Diagram of a discrete-time system. (D = unit delay)

it is important to first understand their classification into categories of linear/nonlinear, time-invariant/time-varying, causal/noncausal, memoryless/with-memory, and stable/unstable. Then it is possible to study the properties of restricted classes of systems, such as discrete-time systems which are linear, time-invariant and stable.

2 Example of Discrete-time Systems

Discrete-time digital systems are often used in place of analog processing systems. Common examples are the replacement of photographs with digital images, and conventional NTSC TV with direct broadcast digital TV. These digital systems can provide higher quality and/or lower cost through the use of standardized, high-volume digital processors.

The following two continuous-time systems are commonly used in electrical engineering:

$$\text{differentiator:} \quad y(t) = \frac{d}{dt}x(t) \tag{2}$$

$$\text{integrator:} \quad y(t) = \int_{-\infty}^{t} x(\tau)d\tau \tag{3}$$

To illustrate how a discrete-time system can be derived from the corresponding continuous-time system, we will show how the above two continuous-time systems can be formulated into corresponding discrete-time systems.

Formulation of a discrete-time system that approximates the continuous-time differentiator:
Since $y(t) = \frac{d}{dt}x(t)$, by the definition of differentiation, $y(t) = lim_{\Delta t \to 0}\frac{x(t+\Delta t)-x(t)}{\Delta t}$, so sampling $y(t)$ at time nT with sufficiently small T gives

$$y(nT) \approx \frac{x(nT) - x((n-1)T)}{T} \tag{4}$$

Clearly, this approximation is more accurate when T is sufficiently small. If we define $y[n] \triangleq y(nT)$ and $x[n] \triangleq x(nT)$, then we arrive at a discrete-time system that approximates the continuous-time differentiator: $y[n] = (x[n] - x[n-1])/T$.

Formulation of a discrete-time system that approximates the continuous-time integrator:
Since $y(t) = \int_{-\infty}^{t} x(\tau)d\tau$, so sampling $y(t)$ at time nT gives

$$
\begin{aligned}
y(nT) \quad &= \int_{-\infty}^{nT} x(\tau)d\tau \\
&= \int_{-\infty}^{(n-1)T} x(\tau)d\tau + \int_{(n-1)T}^{nT} x(\tau)d\tau \\
&\approx y\left((n-1)T\right) + x(nT) \cdot T
\end{aligned}
$$

Clearly, this approximation is more accurate when T is sufficiently small. If we define $y[n] \triangleq y(nT)$ and $x[n] \triangleq x(nT)$, then we arrive at a discrete-time system that approximates the continuous-time integrator: $y[n] = y[n-1] + x[n] \cdot T$.

INLAB REPORT:

For each of these two systems, do the following:

i. Draw block diagrams of both the discrete-time differentiator and integrator as in Fig. 1.

ii. Apply both the discrete-time differentiator and integrator to the signal $u[n] - u[n - (N+1)]$, with $N = 10$, for $-10 \leq n \leq 20$. (This assumes a time-step of $T = 1$)

iii. Use the discrete-time differentiator to numerically evaluate $\frac{d}{dt}x(t)$ of $x(t) = sin(2\pi t)$ for $t \in [0, 10]$, try $T = 0.1$ and $T = 0.001$ and compare the results. You can use SIMULINK if you know how.

---

3 Difference Equations

In this section, we will study the effect of two discrete-time filters. The first filter, $y = S_1[x]$, obeys the difference equation
$$y[n] = x[n] - x[n-1]$$
and the second filter, $y = S_2[x]$, obeys the difference equation

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

Write Matlab functions to implement each of these filters. Then use these functions to calculate the impulse response of each of the following 5 systems: S_1, S_2, $S_1(S_2)$ (i.e., the series connection with S_1 following S_2), $S_2(S_1)$ (i.e., the series connection with S_2 following S_1), and $S_1 + S_2$.

Note: In Matlab, when implementing a difference equation using a loop structure, it is very good practice to pre-define your output vector before entering into the loop. Otherwise,

Matlab has to resize the output vector at each iteration. For example, say you are using a FOR loop to filter the signal $x[n]$, yielding an output $y[n]$. You can pre-define the output vector by issuing the command `y = zeros(1,N)` before entering the loop, where N is the final length of y. For long signals, this speeds up the computation dramatically.

INLAB REPORT:

For each of the five systems, draw and submit a system diagram (use only delays, multiplications and additions as in Fig. 1). Also submit plots of each impulse response. Discuss your observations.

Download music.au
https://engineering.purdue.edu/VISE/ee438L/lab2/data/music.zip
How to play audio signals
https://engineering.purdue.edu/VISE/ee438L/matlab/help/pdf/audio.pdf

Download the audio file music.au. Use the command *audioread* to load the file into Matlab. Then use the Matlab function *sound* to listen to the signal.

Next filter the audio signal with each of the two systems S_1 and S_2. Listen to the two filtered signals.

INLAB REPORT:
How do the filters change the sound of the audio signals? Explain your observations.

---

4 Inverse Systems

Consider the system $y = S_2[x]$ from Section 3. Find a difference equation for a new system $y = S_3[x]$ such that $\delta = S_3[S_2[\delta]]$ where δ denotes the discrete-time impulse function $\delta(n)$. Since both systems S_2 and S_3 are LTI, the time-invariance and superposition properties can be used to obtain $x = S_3[S_2[x]]$ for *any* discrete-time signal x. We say that the systems S_3 and S_2 are inverse filters because they cancel out the effects of each other.

Hint: The system $y = S_3[x]$ can be described by the difference equation

$$y[n] = ax[n] \ + \ bx[n-1]$$

where a and b are constants.

Write a Matlab function `y = S3(x)` which implements the system S_3. Then obtain the impulse response of both S_3 and $S_3[S_2[\delta]]$.

INLAB REPORT:
Draw a system diagram for the system S_3, and submit plots of the impulse responses for S_3 and $S_3(S_2)$.

---

5   System Tests

<div style="border:1px solid black;">

Download bbox.zip
`https://www.projectrhea.org/rhea/images/e/e9/Bbox.zip`

</div>

Often it is necessary to determine if a system is linear and/or time-invariant. If the inner workings of a system are not known, this task is impossible because the linearity and time-invariance properties must hold true for all possible inputs signals. However, it is possible to show that a system is non-linear or time-varying because only a single instance must be found where the properties are violated.

The zip file bbox.zip contains three "black-box" systems in the files bbox1.p, bbox2.p, and bbox3.p. These files work as Matlab functions, with the syntax `y=bboxN(x)`, where x and y are the input and the output signals, and *N = 1, 2 or 3*. Exactly one of these systems is non-linear, and exactly one of them is time-varying. Your task is to find the non-linear system and the time-varying system.

Hints:

1. You should try a variety of input signals until you find a counter-example.

2. When testing for time-invariance, you need to look at the responses to a signal and to its delayed version. Since all your signals in MATLAB have finite duration, you should be very careful about shifting signals. In particular, if you want to shift a signal x by M samples to the left, x should start with at least M zeros. If you want to shift x by M samples to the right, x should end with at least M zeros.

3. When testing for linearity, you may find that simple inputs such as the unit impulse do not accomplish the task. In this case, you should try something more complicated like a sinusoid or a random signal generated with the `rand` command.

<div style="border:1px solid black;">

INLAB REPORT:

- State which system is non-linear, and which system is time-varying.

- Submit plots of input/output signal pairs that support your conclusions.

- Indicate on the plots why they support your conclusions.

</div>

6   Stock Market Example

6.1   Moving Averages

One reason that digital signal processing (DSP) techniques are so powerful is that they can be used for very different kinds of signals. While most continuous-time systems only

process voltage and current signals, a computer can process discrete-time signals which are essentially just sequences of numbers. Therefore DSP may be used in a very wide range of applications. Let's look at an example.

A stockbroker wants to see whether the average value of a certain stock is increasing or decreasing. To do this, the daily fluctuations of the stock values must be eliminated. A popular business magazine recommends three possible methods for computing this average.

$$\text{averagevalue(today)} = \frac{1}{3}\left(\text{value(today)+value(yesterday)+value(2 days ago)}\right) \tag{5}$$

$$\text{averagevalue(today)} = 0.6 * \text{averagevalue(yesterday)} + 0.4 * (\text{value(today)}) \tag{6}$$

$$\text{averagevalue(today)} = \text{averagevalue(yesterday)} + \frac{1}{3}\left(\text{value(today) - value(3 days ago)}\right) \tag{7}$$

INLAB REPORT:

- For each of these three methods, 1) write a difference equation, 2) draw a system diagram, and 3) calculate the impulse response.

- Explain why methods (5) and (7) are known as moving averages.

6.2   Application

Download stockrates.mat
`https://engineering.purdue.edu/VISE/ee438L/lab2/data/stockrates.zip`
Help on loading Matlab files
`https://engineering.purdue.edu/VISE/ee438L/matlab/help/pdf/load.pdf`

Load the file stockrates.mat into Matlab. This file contains a vector, called *rate*, of daily stock market exchange rates for a publicly traded stock.

Apply Filters (6) and (7) from Section 6 of the background exercises to smooth the stock values. When you apply Filter (6) you will need to initialize the value of *averagevalue(yesterday)*. Use an initial value of 0. Similarly, in Filter (7), set the initial values of the *value* vector to 0 (for the days prior to the start of data collection). Use the *subplot* command to plot the original stock values, the result of filtering with (6), and the result of filtering with (7).

INLAB REPORT:
Submit your plots of the original and filtered exchange-rates. Discuss the advantages and disadvantages of the two filters. Can you suggest a better method for initializing the filter outputs?