Department of Electrical Engineering
Indian Institute of Technology Kharagpur

Digital Signal Processing Laboratory (EE39203)

Autumn, 2022-23

Experiment 1 **Discrete and Continuous Time Signals**

Slot: Date:

Student Name: Roll No.:

Grading Rubric

	Tick the best applicable per row			Points
	Below Expectation	Lacking in Some	Meets all Expectation	
Completeness of the report				
Organization of the report (5 pts) *With cover sheet, answers are in the same order as questions in the lab, copies of the questions are included in report, prepared in LaTeX*				
Quality of figures (5 pts) *Correctly labelled with title, x-axis, y-axis, and name(s)*				
Understanding of continuous and discrete-time signals (15 pts) *Matlab figures, questions*				
Ability to compute integral manually and in Matlab (30 pts) *Manual computation, Matlab figures, Matlab codes, questions*				
Ability to define and display functions (1D and 2D) (30 pts) *Matlab figures, Matlab codes, questions*				
Understanding of sampling (15 pts) *Matlab figures, questions*				
			TOTAL (100 pts)	

Total Points (100): TA Name: TA Initials:

1. Learning Objective

The purpose of this lab is to illustrate the properties of continuous and discrete-time signals using digital computers and the Matlab software environment. Please submit well documented and commented codes which are written as a Matlab Function. A continuous-time signal takes on a value at every point in time, whereas a discrete-time signal is only defined at integer values of the "time" variable. However, while discrete-time signals can be easily stored and processed on a computer, it is impossible to store the values of a continuous-time signal for all points along a segment of the real line. In later labs, we will see that digital computers are actually restricted to the storage of quantized discrete-time signals. Such signals are appropriately known as digital signals. How then do we process continuous-time signals? In this lab, we will show that continuous-time signals may be processed by first approximating them by discrete-time signals using a process known as sampling. We will see that proper selection of the spacing between samples is crucial for an efficient and accurate approximation of a continuous-time signal.

Excessively close spacing will lead to too much data, whereas excessively distant spacing will lead to a poor approximation of the continuous-time signal. Sampling will be an important topic in future labs, but for now we will use sampling to approximately compute some simple attributes of both real and synthetic signals.

2. Code of Conduct

Students are expected to behave ethically both in and out of the lab. Unethical behaviour includes, but is not limited to, the following:

- Possession of another person's laboratory solutions from the current or previous years.
- Reference to, or use of another person's laboratory solutions from the current or previous years.
- Submission of work that is not done by your laboratory group.
- Allowing another person to copy your laboratory solutions or work.
- Cheating on quizzes.

The rules of laboratory ethics are designed to facilitate these goals. We emphasize that laboratory TAs are available throughout the week to help the student both understand the basic concepts and answer the questions being asked in the laboratory exercises. By performing the laboratories independently, students will likely learn more and improve their performance in the course as a whole. Please note that it is the responsibility of the student to make sure that the content of their graded laboratories is not distributed to other students. If there is any question as to whether a given action might be considered unethical, please see the professor or the TA before you engage in such actions.

3. Continuous-Time Vs. Discrete-Time

In the following sections, we will illustrate the process of sampling, and demonstrate the importance of the sampling interval to the precision of numerical computations.

3.1. Displaying Continuous-Time Vs. Discrete-Time

It is common to plot a discrete-time signal as dots with stem in a Cartesian coordinate system. This can be done in the Matlab environment by using the stem command. We will

also use the subplot command to put multiple plots on a single figure. A digital computer cannot store all points of a continuous-time signal since this would require an infinite amount of memory. It is, however, possible to plot a signal which looks like a continuous-time signal, by computing the value of the signal at closely spaced points in time, and then connecting the plotted points with lines. The Matlab plot function may be used to generate such plots. As you can see, it is important to have many points to make the signal appear smooth. But how many points are enough for numerical calculations? In the following sections we will examine the effect of the sampling interval on the accuracy of computations.

Lab Report:
Submit a report with copy of the well documented/ commented codes and the plots of two continuous-time "looking" functions and their sampled discrete time representation, by varying the sampling frequency. Label them with the title command, and include your name and roll number in the title of the plot. Comment on the accuracy of each of the continuous time plots. Briefly state any interesting phenomena you observe.

3.2. Vector Index vs. Time

In Matlab, the possible indices of a vector are the natural integers, starting from one (i.e., 1, 2, 3, 4, 5, . . .). Vector indices can neither be negative nor zero. As an example, there is no vector entry corresponding to a[0] or a[−1]: referring to such entries would yield an error message. We saw above that the samples of a continuous-time signal, say x(t), can be stored in a vector in Matlab. It is common practice to use the same variable for the vector and the signal. So one often denotes the samples of x(t) by x[n], even though this is an abuse of notation and lacks rigor. It is important not to confuse the index of a vector x[n] with the value of the independent variable of a function x(t). As an example, Matlab can be used to represent the function x(t) = sin(t) by sampling t at small intervals. The resulting samples may be stored in a vector called "x" in your program. However, it is important to realize that the function "x" and the vector "x" in the program are not the same thing.

Lab Report:
Print the three subplots and explain the difference between the three signals represented. Write Matlab command(s) that would print the graph of sin(t) for the values of t on the interval [3.5, 4.5]. (Pick a suitable increment for t.)

3.3. Analytical Calculation

Compute these two integrals. Do the computations manually.

$$x_1(t) = \int_0^{2\pi} \sin^2(7t)dt$$

$$x_2(t) = \int_0^1 e^t dt$$

Lab Report:
Detail out your calculations of these two integrals. Show all work.

3.4. Numerical Computation of Continuous-Time Signals

One common calculation on continuous-time signals is integration. The Riemann integral approximates the area under a curve by breaking the region into many rectangles and summing their areas. Each rectangle is chosen to have the same width Δt, and the height of each rectangle is the value of the function at the start of the rectangle's interval.

To see the effects of using a different number of points to represent a continuous-time signal, write a Matlab function for numerically computing the integral of the function $\sin^2(7t)$ over the interval $[0, 2\pi]$. The syntax of the function should be I=integ1(N) where I is the result and N is the number of rectangles used to approximate the integral. This function should use the sum command and it should not contain any for loops!

Next write an m-file script that evaluates I(N) for $1 \leq N \leq 100$, stores the result in a vector and plots the resulting vector as a function of N. This m-file script may contain for loops. Repeat this procedure for a second function J=integ2(N) which numerically computes the integral of exp(t) on the interval [0, 1].

Lab Report:
Submit plots of I(N) and J(N) versus N. Use the subplot command to put both plots on a single sheet of paper. Also submit your Matlab code for each function. Compare your results to the analytical solutions obtained in previous section. Explain why I(7) = I(14) = 0.

4. Special Functions

Plot the following two continuous-time functions over the specified intervals. Write separate script files if you prefer. Use the subplot command to put both plots in a single figure, and be sure to label the time axes.

$$x_3(t) = \begin{cases} \dfrac{\sin \pi t}{\pi t}, t \neq 0 \\ 1, t = 0 \end{cases} \; for\; t \in [-10,10]$$

$$x_4(t) = rect(t)\; for\; t \in [-1,2]$$

Hint: The function rect(t) may be computed in Matlab by using a Boolean expression. For example, if `t=-10:0.1:10`, then `y = rect(t)` may be computed using the Matlab command `y = (abs(t)<=0.5).`

Write an .m-script file to stem the following discrete-time function for a = 0.8, a = 1.0 and a = 1.5. Use the subplot command to put all three plots in a single figure. Issue the command `orient('tall')` just prior to printing to prevent crowding of the subplots.

$$x[n] = a^n(u[n] - u[n-10])\; for\; n \in [-20,20]$$

Repeat this procedure for the function

$$x[n] = \cos(\omega n)a^n u[n]\; for\; \omega = \frac{\pi}{4}, and\; n \in [-1,10]$$

Hint: The unit step function y = u[n] may be computed in Matlab using the command `y = (n>=0)`, where n is a vector of values of time indices.

Lab Report:
Submit all three figures, for a total of 8 plots. Also submit the printouts of your Matlab .m-files.

5. Sampling

The word sampling refers to the conversion of a continuous-time signal into a discrete-time signal. The signal is converted by taking its value, or sample, at uniformly spaced points in time. The time between two consecutive samples is called the sampling period. For example, a sampling period of 0.1 seconds implies that the value of the signal is stored every 0.1 seconds.

Consider the signal f(t) = sin(2πt). We may form a discrete-time signal, x[n], by sampling this signal with a period of T_s. In this case,

$$x(n) = f(T_s n) = \sin(2\pi T_s n)$$

Use the stem command to plot the function f(T_sn) defined above for the following values of T_s and n. Use the subplot command to put all the plots in a single figure, and scale the plots properly with the axis command.

1. T_s = 1/10, 0 ≤ n ≤ 100; axis([0,100,-1,1])

2. T_s = 1/3, 0 ≤ n ≤ 30; axis([0,30,-1,1])

3. T_s = 1/2, 0 ≤ n ≤ 20; axis([0,20,-1,1])

4. T_s = 10/9, 0 ≤ n ≤ 9; axis([0,9,-1,1])

Lab Report:
Submit the figure containing all four subplots. Discuss your results. How does the sampled version of the signal with T_s = 1/10 compare to those with T_s = 1/3, T_s = 1/2 and T_s = 10/9?

Acknowledgement: This manual is based on "Purdue University: ECE438 - Digital Signal Processing with Applications (2016)" by Prof. Charles Bouman and Prof. Mireille Boutin.