EEE - 321: Signals and Systems

Lab Assignment 5

Please carefully study this assignment before coming to the laboratory. You may begin working on it or even complete it if you wish, but you do not have to. There will be a short quiz in the lab session to test your understanding of the content of the assignment. Within one week, complete the assignment in the form of a report and upload to Moodle. Some of the exercises will be performed by hand and others by using Matlab. What you should include in your report is indicated within the exercises.

Part 1

Suppose $g(t)$ is a piecewise function such that

$$g(t) = \begin{cases} 1, & -1 \leq t < 0, \\ -2, & 0 < t \leq 1, \\ 0, & \text{otherwise}, \end{cases} \tag{1}$$

where the time unit is in seconds. Sketch $g(t)$ by hand.

Now, let

$$f(t) = 5g(4t - 3).$$

Sketch $f(t)$ by hand, and clearly indicate all signal amplitudes and time points on the sketch. Repeat for

$$h(t) = 3g\Big(-3(t-2)\Big).$$

Include the sketches in your report. Also explain the steps that you follow while generating $f(t)$ and $h(t)$ from $g(t)$.

Suppose this signal is regularly sampled with the sampling period T_s. According to the Shannon–Nyquist criteria, is it possible to fully recover this signal from its samples? If so, what values of T_s allow for perfect reconstruction? Justify your answers by using concrete mathematical calculations and graphical illustrations.

Part 2

Let $x(t)$ be a continuous signal. The sampling operation can be formulated as a multiplication by an impulse train. So the sampled signal in the continuous domain with the sampling period

T_s, $\bar{x}(t)$, becomes

$$\bar{x}(t) = x(t) \sum_{n=-\infty}^{\infty} \delta(t - nT_s) = \sum_{n=-\infty}^{\infty} x(nT_s)\, \delta(t - nT_s). \tag{2}$$

Now, the coefficients of the shifted impulses, $x(nT_s)$, can be considered as the sampled version of the signal and can be used in discrete–time computations. That is,

$$\bar{x}[n] = x(nT_s), \qquad n \in \mathbb{Z}. \tag{3}$$

An important problem of signal processing is the reconstruction of the original continuous signal $x(t)$ from its sampled version $\bar{x}[n]$, which is also named discrete-to-continuous conversion, digital-to-analog conversion, or interpolation. Let us denote the reconstructed signal with $x_R(t)$. A common practice is to form $x_R(t)$ by convolving $\bar{x}(t)$ with an interpolating pulse $p(t)$. In your report, show that the reconstructed signal $x_R(t)$ can be written as

$$x_R(t) = \sum_{n'=-\infty}^{\infty} \bar{x}[n']\, p(t - n'T_s). \tag{4}$$

Ideally, we would want to have $x_R(t) = x(t)$ for all t, but only in certain cases is this exactly possible. Usually, with our selections for $p(t)$ and T_s, we try to minimize the difference between $x_R(t)$ and $x(t)$.

Also note that (4) does *not* guarantee that $x_R(nT_s) = x(nT_s)$ for $n \in \mathbb{Z}$. However, in many applications, it is desired that $x_R(nT_s) = x(nT_s)$ even if perfect reconstruction is not achieved. By doing so, for all n we obtain the original samples $\bar{x}[n]$ when we sample $x_R(t)$ with T_s again. If $x_R(nT_s) = \bar{x}[n]$ for all n, then the interpolation is said to be *consistent*. Show that we have $x_R(nT_s) = \bar{x}[n]$ for all n if $p(0) = 1$ and $p(kT_s) = 0$ for all nonzero integers k. Include your work in your report.

Three common choices for $p(t)$ are:

- **Zero-order hold interpolation:**

$$p_Z(t) = \mathrm{rect}\left(\frac{t}{T_s}\right), \qquad \mathrm{rect}(t) = \begin{cases} 1, & -\frac{1}{2} \le t < \frac{1}{2}, \\ 0, & \text{otherwise.} \end{cases}$$

- **Linear interpolation:**

$$p_L(t) = \mathrm{tri}\left(\frac{t}{2T_s}\right), \qquad \mathrm{tri}(t) = \begin{cases} 1 - \dfrac{|t|}{0.5}, & -0.5 \le t \le 0.5, \\ 0, & \text{otherwise.} \end{cases}$$

- **Ideal bandlimited interpolation:**

$$p_I(t) = \mathrm{sinc}\left(\frac{t}{T_s}\right), \qquad \mathrm{sinc}(t) = \begin{cases} 1, & t = 0, \\ \dfrac{\sin(\pi t)}{\pi t}, & t \neq 0. \end{cases}$$

Answer the following questions:

a) What are the values of $p_Z(t)$, $p_L(t)$, and $p_I(t)$ at $t = 0$?

b) What are the values of $p_Z(t)$, $p_L(t)$, and $p_I(t)$ at $t = kT_s$ where k is a nonzero integer?

c) Based on your answers to items (a) and (b), are the interpolations performed using $p_Z(t)$, $p_L(t)$, and $p_I(t)$ consistent?

Include your answers in your report.

Part 3

In this part, you will write a Matlab function which generates one of the three interpolating functions given in the previous part. Your function will look like

$$\texttt{p = generateInterp(type, Ts, dur)},$$

where

- `type` determines the type of the function. It can be either 0, 1, or 2. Here, 0 indicates zero-order interpolation, 1 indicates linear interpolation, and 2 indicates ideal interpolation.
- `Ts` determines the sampling rate.
- `dur` determines the duration of the signal. For example, if `dur = 2`, then the interval of the interpolating pulse will be between -1 and 1.

Important: Note that in the previous part, the interpolating functions were defined as continuous functions. However, here you have to generate a *discrete* version of them in order to simulate the interpolation process. Therefore, the discrete interpolating function that you generate should be sampled much more densely than the signal to be interpolated, so that it behaves like a continuous function. For example, a sampling period of

$$T_s/1000$$

may be chosen for the interpolating pulse.

Note: You are **not** allowed to use built-in Matlab commands to generate the pulses. Include your Matlab code in your report.

Now let `dur` equal your ID number modulo 7 (take it 3 if it is 0), and let

$$T_s = \frac{\texttt{dur}}{4}.$$

Using the `generateInterp` function that you wrote, compute all three pulses $p_Z(t)$, $p_L(t)$, and $p_I(t)$, and plot them versus t. Include these plots in your report.

Part 4

In this part, you are going to write a program in order to simulate the interpolation process. In your program, the continuous-like (now you know why the word "like" is added here) signal

$x(t)$ will be interpolated from its samples

$$\bar{x}[n] = x(nT_s)$$

according to Eq. (4). Note that for a general $x(t)$, the formula in Eq. (4) is impossible to implement exactly because it contains an infinite number of samples. Usually, it is assumed that $x(t)$ has a finite duration so that it produces a finite number of nonzero samples when sampled (say N). Under this assumption, we have

$$x_R(t) = \sum_{n_0=0}^{N-1} \bar{x}[n_0]\, p(t - n_0 T_s). \tag{5}$$

Your program should have the following form:

$$\texttt{function } xR = DtoA(type, Ts, dur, Xn)$$

where

- `type` denotes the type of interpolation. It can be either 0, 1, or 2. Here, 0 indicates zero-order interpolation, 1 indicates linear interpolation, and 2 indicates ideal interpolation.
- `Ts` and `dur` have the same meaning as in the previous part.
- `Xn` (of size $1 \times N$) contains the samples of $x(t)$, which are assumed to be taken at $0, T_s, 2T_s, \ldots, (N-1)T_s$, so that

$$\texttt{Xn(1)} = x(0), \quad \texttt{Xn(2)} = x(T_s), \quad \ldots, \quad \texttt{Xn(N)} = x((N-1)T_s).$$

- `xR` (of size $1 \times M$) denotes the reconstructed signal.

While writing this function, make use of the function `generateInterp`. Within your code, you also need to generate the time variable t. You can generate t according to the explanation given in the previous part. Also, in your code, do not use any `for` loop over the elements of t, and write your function as efficiently as possible. Include your code in your report.

Part 5

In this part, you will compare the efficiency of the three interpolating methods on the reconstruction of the signal $g(t)$, which is given in Part 1.

First, generate the sampled version of $g(t)$ for $-3 \leq t \leq 3$ with

$$T_s = \frac{1}{20a} \quad \text{seconds,}$$

where a is a random integer that you will generate using Matlab and it should be between 3 and 8. Include both your code and the stem plot of $g(nT_s)$ in your report.

Now, using the `DtoA` function, generate $g_R(t)$ for each interpolating method separately. Include the plots of the reconstructed signals. Discuss and compare the success of the interpolating methods.

Next, gradually increase T_s, generate $g(nT_s)$ and their reconstructed versions for each new value of T_s. Examine the reconstructed signals. Does the reconstruction become more successful while increasing T_s? Discuss your observations. (You do not need to include any code or plots for this question.)

Part 6

Let D denote your ID number, and let D_7 denote your ID number in modulo 7. That is,

$$D \equiv D_7 \pmod 7$$

with $0 \leq D_7 \leq 6$. You can compute D_7 using the `rem` command of Matlab. To learn how `rem` works, type `help rem` in the Matlab command window.

Let

$$x(t) = \begin{cases} 0.25\cos\left(2\pi 3t + \frac{\pi}{8}\right) + 0.4\cos\left(2\pi \cdot 5t - \frac{1}{2}\right) + 0.9\cos(2\pi \cdot t + \frac{\pi}{4}), & -2 \leq t \leq 2, \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

where the time unit is seconds.

(a) Let $T_s = 0.005(D_7 + 1)$. Compute the continuous-like function $x(t)$ and display it versus t over the interval $[-2, 2]$ using the `plot` command. On the same figure, using the `stem` command, display the sampled function $x(nT_s)$ with a different color. Include this plot in your report.

Next, using the samples `Xn`, compute $x_R(t)$ for all three interpolators, using the function you developed in Part 4, and then plot the reconstructed signals. Include these plots in your report as well. Examine the results. Which interpolator seems to be the most successful one? In particular, can you distinguish the reconstruction of the ideal interpolator from the original signal? If there is a great difference, why? Include your comments in your report.

(b) Repeat part (a) for $T_s = 0.25 + 0.01D_7$.

(c) Repeat part (a) for $T_s = 0.18 + 0.005(D_7 + 1)$.

(d) Repeat part (a) for $T_s = 0.099$.

Run your codes with several other T_s values between $0.01 < T_s < 0.2$ and examine the results (you do not need to provide any plots). In particular, what do you recognize about the performance of the ideal bandlimited interpolator? Can you notice any difference between the original and the reconstructed signal as long as $0.01 < T_s < 0.1$? What happens afterwards $(0.1 \leq T_s \leq 0.2)$? Why does this happen? Include your comments in your report.