EEE - 321: Signals and Systems
Lab Assignment 5

Please carefully study this assignment before coming to the laboratory. Within one week, complete the assignment in the form of a report and turn it into the assistant. 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

$$g(t) = \begin{cases} -2 & \text{if } -1 \leq t < 0 \\ 3 & \text{if } 0 < t \leq 1 \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where the time unit is in seconds. Sketch $g(t)$ with your hand. Now, let $f(t) = 4g(2t-1)$. Sketch $f(t)$ with your hand, and clearly indicate all signal amplitudes and time points on the sketch. Repeat for $h(t) = 3g(-3(t-1))$. 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 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 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 the discrete domain computations. That is

$$\bar{x}[n] = x(nT_s) \tag{3}$$

for $n \in \mathcal{Z}$, $-\infty < n < \infty$.

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 $\bar{x}(t)$ by an interpolating pulse $p(t)$. In your report, show that the reconstructed signal $x_R(t)$ can be written as follows:

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

Ideally, we would want to have $x_R(t) = x(t)$ for all t, but only in certain cases this is 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, Equation 4 does not guarantee that $x_R(nT_s) = x(nT_s)$, either, where $n \in \mathcal{Z}$, $-\infty < n < \infty$. However, in many applications, it is desired that $x_R(nT_s) = x(nT_s)$ even if the perfect reconstruction is not achieved. By doing so, for all n we can obtain the original samples $\bar{x}[n]$ when we sample $x_R(t)$ with T_s once 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

- $p_Z(t) = \text{rect}\left(\frac{t}{T_s}\right)$ where
$$\text{rect}(t) = \begin{cases} 1 & \text{if } -\frac{1}{2} \le t < \frac{1}{2}, \\ 0 & \text{otherwise.} \end{cases}$$
in which case the reconstruction process is called **zero order hold interpolation**,
- $p_L(t) = \text{tri}\left(\frac{t}{T_s}\right)$ where
$$\text{tri}(t) = \begin{cases} 1 - \frac{|t|}{0.5} & \text{if } -0.5 \le t \le 0.5, \\ 0 & \text{otherwise.} \end{cases}$$
in which case the reconstruction process is called **linear interpolation**
- and $p_I(t) = \text{sinc}\left(\frac{t}{T_s}\right)$ where
$$\text{sinc}(t) = \begin{cases} 1 \text{ if } t = 0, \\ \frac{\sin(\pi t)}{\pi t} \end{cases} \quad \text{otherwise.}$$
in which case the reconstruction process is called **ideal bandlimited interpolation**.

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 to your report.

Part 3

In this part, you will write a Matlab function that generates one of the three interpolating functions given in the previous part. Your function will look like **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 function, 1 indicates linear interpolation function and 2 indicates ideal interpolation function
- **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, we defined the interpolating functions as continuous functions. However, here you have to generate a discrete version of them with the aim of doing a simulation of the interpolation process. Therefore, the discrete interpolating function that you generate here should be sampled much denser than the signal to be interpolated so that it behaves like a continuous function. For example, **Ts**/500 can be chosen for the sampling period of the interpolating function.

Note: You are NOT allowed to use built-in commands of Matlab to generate the pulses.

Include your Matlab code in your report.

Now let **dur** equal to your ID number in modulo 7 (take it 3 if it is 0) and **Ts** equal to **dur**/5. Using **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 your 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 exactly implement 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:

$$x_R(t) = \sum_{n'=0}^{N-1} \bar{x}[n']p(t-n'T_s) \tag{5}$$

Your programs should look like:
function xR=DtoA(type,Ts,dur,Xn)
where

- **type** denotes the type of the interpolation. It can be either 0, 1 or 2. Here 0 indicates zero order interpolation, 1 indicates linear interpolation and 2 indicates ideal interpolation is going to be performed.
- **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 $0, T_s, 2T_s, ..., (N-1)T_s$, so that **Xn(1)**=$x(0)$, **Xn(2)**=$x(T_s)$, ..., **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 need to generate the time variable **t** as well. You can generate **t** according to the explanation made in the previous part. Also, in your code, do not use any for loop over the **t** array 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 a sampled version of $g(t)$ for $-3 \leq t \leq 3$ with T_s equals to $1/(20a)$ seconds, where a is a random integer that you will generate using Matlab and it should be between 2 and 6. Include both your code and stem plot of $g(nTs)$ in your report.

Now, using **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.

Now, increase Ts gradually, generate $g(nTs)$ and their reconstructed versions for each Ts. Examine the reconstructed signals. Does the reconstruction become more successful while increasing Ts. Discuss your observations. (You do not need to put any code or plot for this question)

Part 6

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

$$D \equiv D_7 \mod 7$$

with $0 \leq D_7 \leq 6$. You can compute D_5 using the **rem** command of Matlab. To learn how **rem** works, type **help rem** in the Matlab command window.

Let

$$x(t) = \begin{cases} 0.25\cos\left(2\pi 3t + \frac{\pi}{8}\right) + 0.4\cos\left(2\pi 5t - 1.2\right) + 0.9\cos\left(2\pi t + \frac{\pi}{4}\right) & \text{if } -2 \le t \le 2 \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

- a) Let $T_s = 0.005(D_7 + 1)$. Compute 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 the 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 a for $T_s = 0.25 + 0.01D_7$.
- c) Repeat a for $T_s = 0.18 + 0.005(D_7 + 1)$.
- d) Repeat 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. 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 $(0.1 \le T_s \le 0.2)$? Why does this happen? Include your comments in your report.