EEE - 321: Signals and Systems

Lab Assignment 5

Part 1

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

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

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

Suppose $g(t)$ is regularly sampled with the sampling period T_s.

Shannon–Nyquist criteria:

A continuous-time signal that is bandlimited to $|\omega| \leq \omega_{\max}$ can be perfectly reconstructed from its samples if the sampling frequency satisfies

$$\omega_s \geq 2\omega_{\max}.$$

a) According to the Shannon–Nyquist criterion, is there any T_s that allows perfect reconstruction of $g(t)$ from its samples? Justify your answer both in the time domain, using $g(t)$, and in the frequency domain, by computing $G(j\omega)$ analytically.

b) Plot $|G(j\omega)|$ in Matlab for $|\omega| \leq 40$ rad/s.

c) Since perfect reconstruction is not possible, define the *energy bandwidth* ω_B as the smallest frequency for which the band $|\omega| \leq \omega_B$ contains a given fraction of the energy of $g(t)$. First compute the total energy of $g(t)$ in the time domain,

$$E = \int_{-\infty}^{\infty} |g(t)|^2 \, dt.$$

Then, using Parseval's relation,

$$E(\omega_B) = \frac{1}{2\pi} \int_{-\omega_B}^{\omega_B} |G(j\omega)|^2 \, d\omega,$$

find ω_B numerically in Matlab for 90% and 99% of the energy. For each case, assume that $g(t)$ is bandlimited to $|\omega| \leq \omega_B$ (i.e., neglect the energy outside this band) and

find the largest T_s that satisfies the Shannon–Nyquist criterion for this band. Note that, since $g(t)$ is not actually bandlimited, sampling with this T_s does not remove aliasing completely.

d) After completing Part 5, take the T_s you found for 99% of the energy, rounded to three decimal places, and sample $g(t)$ at $t = nT_s$ (n integer) for $-3 \leq t \leq 3$. Reconstruct $g(t)$ from these samples with the ideal interpolator using `dur = 12`, and determine the RMS error between $g_R(t)$ and $g(t)$ over $-3 \leq t \leq 3$. Is capturing 99% of the energy enough for a visually good reconstruction? Explain.

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 $p(t)$. 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.

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

- **Zero-order hold interpolation:**

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

- **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?

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

`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.

Now let `dur` equal your ID number modulo 6 (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.

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.

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

$$T_s = \frac{1}{25a} \quad \text{seconds},$$

where a is a random integer that you will generate using Matlab and it should be between 2 and 6. Display $g(nT_s)$ using the `stem` command.

Note: Your `DtoA` function assumes that the first sample, `Xn(1)`, is taken at $t = 0$. Here, the first sample is taken at a negative time. Therefore, shift the time axis of $g_R(t)$ by the time of the first sample before plotting it or comparing it with $g(t)$; otherwise the reconstructed signal will appear shifted in time.

Now, using the `DtoA` function, generate $g_R(t)$ for each interpolating method separately and plot 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.

Part 6

Let D denote your ID number, and let D_6 denote your ID number in modulo 6. That is,

$$D \equiv D_6 \pmod 6$$

with $0 \leq D_6 \leq 5$. You can compute D_6 using the `rem` command of Matlab. To learn how `rem` works, type `help rem` in the Matlab command window.

Let

$$x(t) = \begin{cases} 0.4\cos\left(2\pi \cdot 2t + \dfrac{\pi}{6}\right) + 0.3\cos\left(2\pi \cdot 4t - \dfrac{1}{3}\right) + 0.7\cos\left(2\pi \cdot t - \dfrac{\pi}{5}\right), & -2 \leq t \leq 2, \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

where the time unit is seconds.

(a) Let $T_s = 0.004(D_6 + 1)$. 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.

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. 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?

(b) Repeat part (a) for $T_s = 0.30 + 0.01D_6$.

(c) Repeat part (a) for $T_s = 0.15 + 0.005(D_6 + 1)$.

(d) Repeat part (a) for $T_s = 0.124$.

Run your codes with several other T_s values between $0.01 < T_s < 0.25$ 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 $0.01 < T_s < 0.125$? What happens afterwards $(0.125 \leq T_s \leq 0.25)$? Why does this happen?