Stanford University
EE 264: Digital Signal Processing
Summer, 2018

Homework #04

Date assigned: July 20, 2018 Date due: July 27, 2018

Reading: This assignment covers primarily lectures 3 to 5, which corresponds to chapter 4 of the text book **DTSP 3e**.
Homework submission: Please submit your solutions on Gradescope. Create a single .pdf file containing all your analytical derivations, sketches, plots, and Matlab code (if any).

Problem 1: (25 points) Same as problem 4.47 in DTSP3e.

Consider the following system for discrete-time processing of the continuous-time input signal $g_c(t)$.

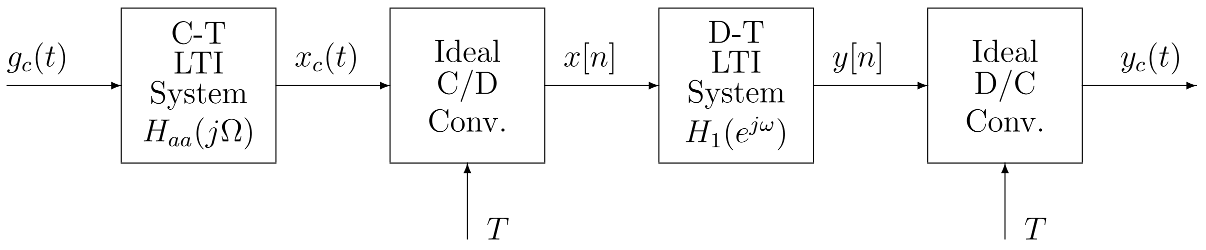


Figure 1: System Block Diagram

The continuous-time input signal to the overall system is of the form $g_c(t) = f_c(t) + e_c(t)$ where $f_c(t)$ is considered to be the "signal" component and $e_c(t)$ is considered to be an "additive noise" component. The Fourier transforms of $f_c(t)$ and $e_c(t)$ are as follows:

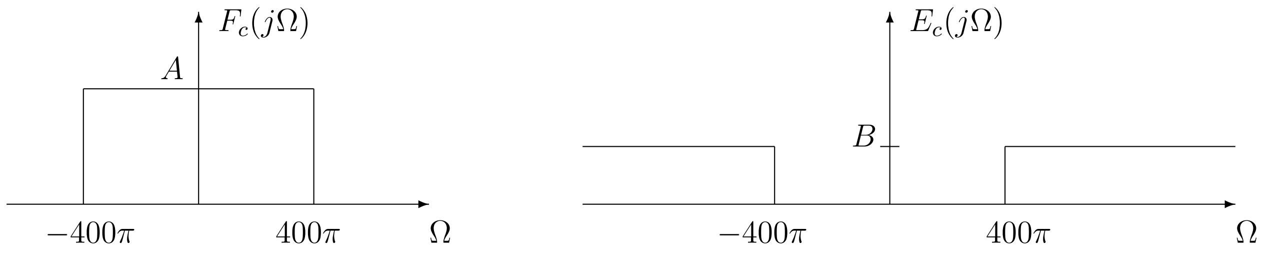


Figure 2: "Typical" Fourier transforms of the input signal components.

Since the total input signal $g_c(t)$ does not have a bandlimited Fourier transform, a zero-phase continuous-time anti-aliasing filter is used to combat aliasing distortion. Its frequency response is:

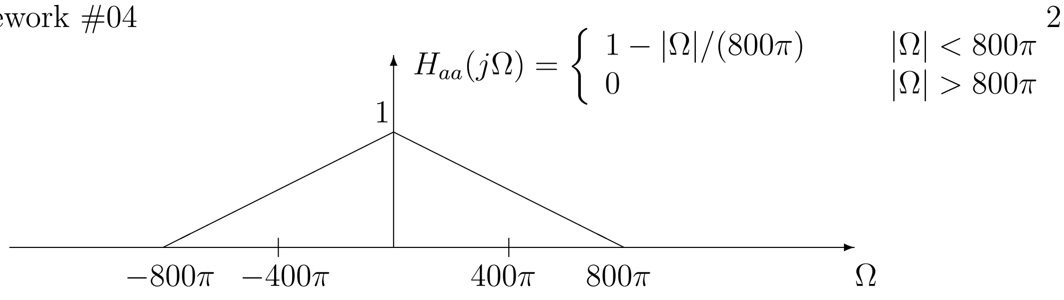


Figure 3: Anti-aliasing filter frequency response.

(a) If in Figure 1, the sampling rate is $2\pi/T = 1600\pi$, and the discrete-time system has frequency response

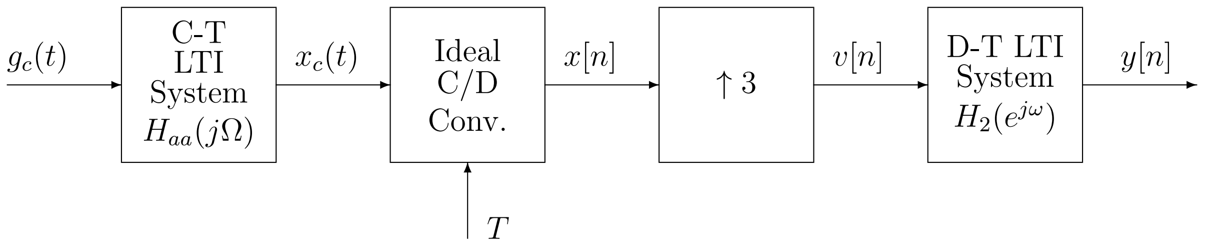
$$H_1(e^{j\omega}) = \begin{cases} 1 & |\omega| < \pi/2 \\ 0 & \pi/2 < |\omega| \leq \pi \end{cases}$$

sketch the Fourier transform of the continuous-time output signal for the input whose Fourier transform is defined in Figure 2.

(b) If the sampling rate is $2\pi/T = 1600\pi$, determine the magnitude and phase of $H_1(e^{j\omega})$ (the frequency response of the discrete-time system) so that the output of the system in Figure 1 is $y_c(t) = f_c(t - 0.1)$. You may use any combination of equations or carefully labeled plots to express your answer.

(c) It turns out that since we are only interested in obtaining $f_c(t)$ at the output, we can use a lower sampling rate than $2\pi/T = 1600\pi$ while still using the anti-aliasing filter in Figure 3. Determine the minimum sampling rate that will avoid aliasing distortion of $F_c(j\Omega)$ and determine the frequency response of the filter $H_1(e^{j\omega})$ that can be used so that $y_c(t) = f_c(t)$ at the output of the system in Figure 1.

(d) Now consider the following system, where $2\pi/T = 1600\pi$, and the input signal is defined in Figure 2 and the anti-aliasing filter is as shown in Figure 3.



$g_c(t)$
C-T LTI System $H_{aa}(j\Omega)$
$x_c(t)$
Ideal C/D Conv.
T
$x[n]$
$\uparrow 3$
$v[n]$
D-T LTI System $H_2(e^{j\omega})$
$y[n]$

Figure 4: Filtering and interpolation system.

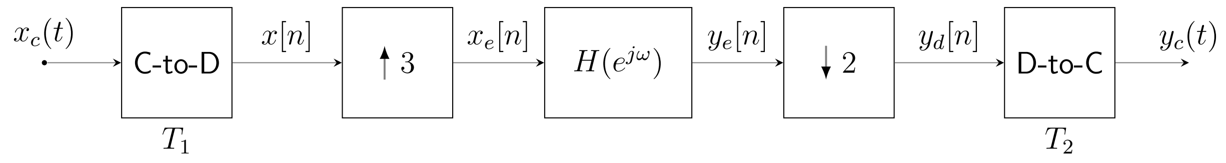
where

$$v[n] = \begin{cases} x[n/3] & n = 0, \pm 3, \pm 6, \ldots \\ 0 & \text{otherwise} \end{cases}$$

What should $H_2(e^{j\omega})$ be if it is desired that $y[n] = f_c(nT/3)$?

Problem 2: (30 points)

Consider the following system for discrete-time processing of the continuous-time signal $x_c(t)$.



Assume that the input signal has a bandlimited continuous-time Fourier transform such that $X_c(j\Omega) = 0$ for $|\Omega| \geq 2\pi(2000)$.

(a) Note that the sampling periods T_1 and T_2 **cannot** be chosen independently if we wish to maintain a consistent time scale between the input and output. Give the relationship that is required between T_2 and T_1.

(b) Determine the **maximum** value of T_2 (equivalently *minimum* sampling rate $f_{s2} = 1/T_2$) so that aliasing cannot occur in sampling at the input or in the processing and reconstruction of the output of the system. Also determine the corresponding value of T_1.

(c) For the values of T_2 and T_1 determined in part (b), Sketch the required frequency response $H(e^{j\omega})$ so that $y_c(t) = x_c(t)$.

(d) For the values of T_2 and T_1 determined in part (b), determine the frequency response $H(e^{j\omega})$ so that $y_c(t) = 2x_c(t - 4T_1)$.

(e) For the values of T_2 and T_1 determined in part (b), determine the frequency response $H(e^{j\omega})$ so that the continuous-time Fourier transform of the output satisfies the condition

$$Y_c(j\Omega) = \begin{cases} 0 & |\Omega| \leq 2\pi(500) \\ X_c(j\Omega) & 2\pi(500) < |\Omega| < 2\pi(1500) \\ 0 & |\Omega| \geq 2\pi(1500). \end{cases}$$

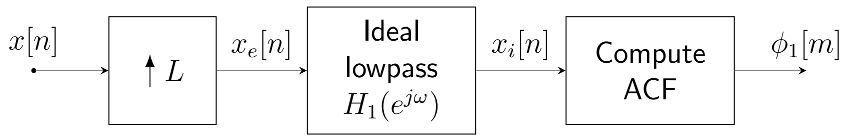
Plot your answer.

(f) For the value of T_2 determined in part (b), determine (give an equation for) the frequency response $H(e^{j\omega})$ so that

$$y_c(t) = \frac{dx_c(t)}{dt}.$$

Problem 3: (15 points) (adapted from the midterm of 2014)

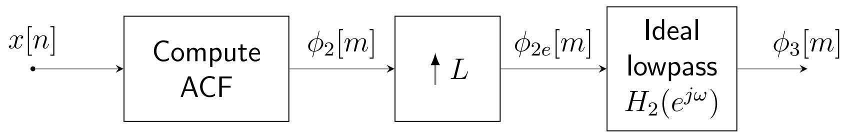
We are given a deterministic signal $x[n]$. We wish to compute the deterministic autocorrelation function (ACF) of an interpolated signal $x_i[n]$ as in the following block diagram



where $H_1(e^{j\omega})$ is the ideal lowpass with gain L and cutoff frequency $\pm\pi/L$. The box labeled "Compute ACF" computes the deterministic autocorrelation function of its input:

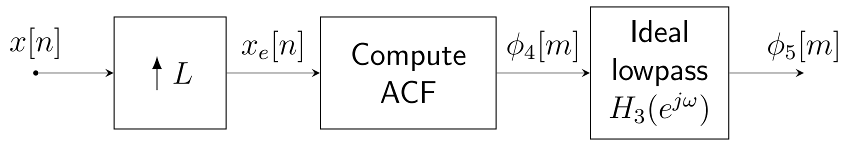
$$\phi_1[m] = \sum_{n=-\infty}^{\infty} x_i[m+n]x_i[n] \qquad \text{(Assume that this inifite sum exists)}$$

(a) An EE 264 student suggests that we can accomplish the same thing with the following system:



Note that in this diagram the ACF $\phi_2[m]$ is treated as just another sequence that can be upsampled and filtered. Specify the gain and cutoff frequency of the ideal lowpass filter $H_2(e^{j\omega})$ that will accomplish the task of ensuring $\phi_3[m] = \phi_1[m]$.

(b) Now another EE 264 student says that the following system also works.



Is the second student right? i.e., is $\phi_5[m] = \phi_1[m]$? If not, why not? If so, how should the gain and cutoff of $H_3(e^{j\omega})$ be set?

Problem 4: Quantization and quantization noise shaping (50 points)

In this problem, you will *hear* how quantization noise affects speech signals and how we can use noise shaping to minimize the effects of coarse quantization.

On Canvas, download the Matlab scritpt `quantization_noise_shaping_template.m` and the audio file `speech_dft.wav`. The script `quantization_noise_shaping_template.m` loads the speech signal from the file `speech_dft.wav`, so you can use it in your simulations. The speech in `speech_dft.wav` is of someone saying the phrase "The discrete Fourier transform of a real-valued signal is conjugate symmetric".

Part 1: Implementing a quantizer

(a) Write a function `quantizer` with the following call:

```
[xq, e] = quantizer(x, B, range_lims),
```

where `x` is the input signal to the quantizer, `xq` is the quantized signal, and `e = x - xq` is the quantization error. `B` is the number of bits of resolution of the quantizer, and `range_lims` is a 1×2 vector that defines the range limits of the quantizer. In this exercise, the speech signal varies from -1 to $+1$. Hence, `range_lims = [-1, 1]`. Note that the dynamic range of the quantizer is simply `range_lims(2) - range_lims(1)`.

Hints:

- You can use the Matlab function `quantiz` to actually perform the quantization. You'll need to use `B` and `range_lims` to calculate the partitions and codebook used by `quantiz`.
- It's up to you to choose whether your quantizer will be mid-tread or mid-rise. Either one will work fine.
- To check your implementation, you can see whether the quantization error `e` is approximately zero mean and whether its histogram resembles an uniform distributed signal.

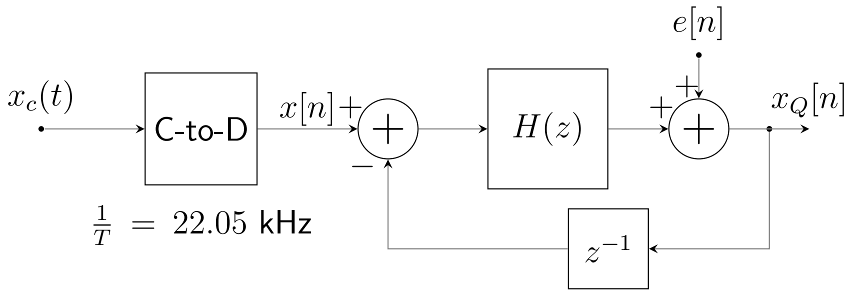
(b) Use your function to quantize the speech signal assuming a resolution of $B = 4$ bits. Use the function `sound(xq, Fs)` to play the quantized speech signal. In class we assumed that the quantization noise is white, plot the empirical autocorrelation function of $e[n]$. What's the quantization noise average power? Repeat your calculations for $B = 10$.

Hint: You may use the Matlab function `xcorr(e, e, maxLag, 'unbiased')` to obtain the empirical autocorrelation function, and the function `stem` to plot it.

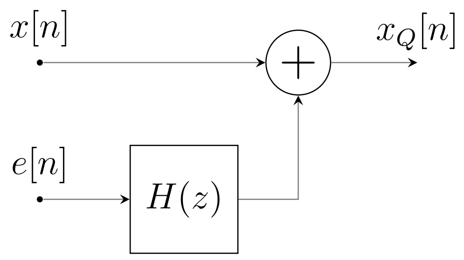
Part 2: Noise shaping without oversampling

To keep things simple, we will not implement the feedback loop (Figure 5a) necessary to achieve quantization noise shaping. Instead, we will use superposition to treat signal and quantization noise separately, as illustrated in Figure 5b. In this equivalent system, the signal $x[n]$ is unaltered, but the quantization noise is filtered by $H(z)$. The quantization noise $e[n]$ will be the quantization noise you obtained by calling your function `quantizer` on the input signal $x[n]$.

As discussed in class, noise shaping can only work well when oversampling is sufficiently high, otherwise the aliased noise will still fall in the signal band. In this first system, we assume that there is no oversampling. The audio recording is done at 22.05 kHz, and we assume that this corresponds to the Nyquist rate of that signal.



(a) Block diagram of noise shaping in analog-to-digital converters, as discussed in lecture 5, slide 24.



(b) Equivalent block obtained by applying superposition in diagram of Figure 5a. Signal is unaffected, while quantization noise is filtered by $H(z)$

Figure 5: Block diagrams for noise shaping without oversampling.

(c) Implement system of Figure 5b to calculate the new quantized signal $x_Q[n]$ after noise shaping. Assume that the quantizer had $B = 4$ bits of resolution, and that the noise was shaped by $H(z) = 1 - z^{-1}$. Use the function `sound(xq, Fs)` to play the new quantized signal.

Hint: To filter the quantization noise you may use the function `filter(b, a, e)`. The vector `e` is the vector you obtained using your function `quantizer`.

Part 3: Noise shaping with oversampling

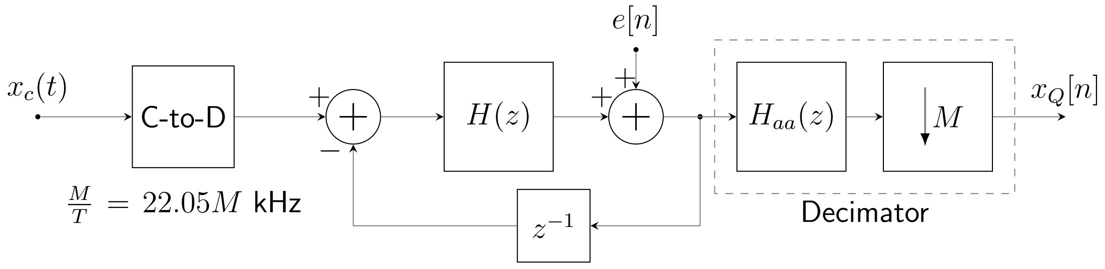
Now we'll assume that the speech signal was recorded using the analog-to-digital converter illustrated in Figure 6a. Note that in this system, the signal is sampled by the C-to-D with sampling frequency $22.05 \times M$ kHz (M times greater than before). All the noise shaping processing is performed at this new rate. Then, at the end, the signal is decimated by M, so that $x_Q[n]$ has the same rate of the previous system.

The equivalent system is shown in Figure 6b. Note that the signal component is unaffected and it's the same as before (i.e., the discrete-time speech signal at rate 22.05 kHz). As for the quantization noise, we'll use the same noise as before (the noise you obtained using your `quantizer` function). However, before doing noise shaping we must upsample the noise by M.

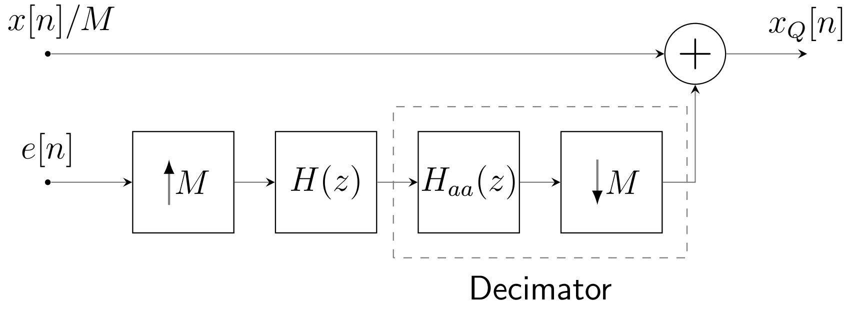
(d) Implement system of Figure 6b to calculate the new quantized signal $x_Q[n]$ after noise shaping. Assume that we had the same conditions as before: $B = 4$ and $H(z) = 1-z^{-1}$. Assume further that the oversampling factor is $M = 3$.

Hints:

- Use the Matlab function `upsample(e, M)` to implement upsampling.



(a) Block diagram of noise shaping in oversampled analog-to-digital converters.



(b) Equivalent block diagram of first-order noise shaping system with oversampling factor M.

Figure 6: Block diagrams for noise shaping with oversampling.

- For decimation, you may use the Matlab function `decimate(e, M)`. This function already includes the anti-aliasing pre-filtering $H_{aa}(z)$. By default, `decimate(e, M)` uses as anti-aliasing filter the Chebyshev Type I IIR filter of order 8.

Note: In order to calculate the newly quantized signal after noise shaping, the input path in Fig. 6b is multiplied by 1/M in order to account for the fact that in Fig. 6a the signal is also decimated. Any solutions that do not correct the input signal by a 1/M factor (dividing it by the oversampling factor) will not be acceptable.

(e) Calculate the quantization noise power after decimation. By how much was the power reduced compared with the original quantization noise? Express this reduction in dB and state how many extra bits of resolution would be needed to achieve the same improvement.