ECES 631 HW 3:
Z-Transform, Sampling, and Intro. to Multirate DSP

1. **Linear Tracking Loop Analysis**: In this problem, we will consider various simple discrete time systems designed to estimate and track a sequence \( x[n] \) with a sequence \( \hat{x}[n] \). At each time instant \( n \), the systems are only able to observe the error at the previous time instant \( e[n-1] = x[n-1] - \hat{x}[n-1] \) (as well as remember the error at all time instants before that i.e. all \( e[k] \) for \( k < n \)). These tracking systems will be turned on at time \( n = 0 \), and we will model this by having \( x[n] = \hat{x}[n] = e[n] = 0 \) for all \( n < 0 \).

   (a) First consider the system which refines its estimates according to:

   \[
   \hat{x}[n] = \hat{x}[n-1] + \mu e[n-1]
   \]  

   i. Implement this system in MATLAB, and observe its response to \( x[n] = \alpha u[n] \) (select some \( \alpha \) and \( u[n] \) is the normal unit step). Can you find a \( \mu \) so that the error \( e[n] \) goes to zero as \( n \to \infty \)?

   ii. Next, observe your MATLAB implementation’s response to \( x[n] = \beta nu[n] + \alpha u[n] \), with \( \beta \neq 0 \). Can you select \( \mu \) so that the error becomes small over time? Does the \( e[n] \) converge to 0?

   (b) Use the z-transform and the partial fraction expansion inverse z-transform to analytically justify the behavior your saw in the previous problem. In particular:

   i. Using equation (1) write \( e[n] \) in terms of \( e[n-1], x[n], x[n-1] \) exclusively.

   ii. Is this effective system (brought about by this tracking system) mapping the input signal \( x[n] \) to the error signal \( e[n] \) a LTI system?

   iii. What is the z-transform of this effective system’s impulse response? (Find by taking the z-transform of the equation from the previous problem by using the time shift property and writing

   \[
   H(z) = \frac{E(z)}{X(z)}
   \]  

   iv. Find the z-transform \( X(z) \) of the signal \( x[n] = \alpha u[n] \).

   v. Find the resulting tracking error signal \( e[n] \)’s z-transform \( E(z) \) resulting from this signal \( x[n] = \alpha u[n] \) by multiplying your previous answer with \( H(z) \) (found two answers ago).

   vi. Use partial fraction expansion to find \( e[n] \) from \( E(z) \) for this input \( x[n] = \alpha u[n] \). How does this justify what you observed for this input signal in problem 1?

   vii. Repeat the previous 3 parts (d,e,f) for the input signal \( x[n] = \beta nu[n] + \alpha u[n] \).

   (c) Using what you learned from analyzing this problem with the z-transform and partial fraction expansion, suggest a alternate set of system dynamics (i.e. a new alternate equation mapping \( \{e[n-k] \mid k > 0\} \) and \( \{\hat{x}[n-k] \mid k > 0\} \) to a new guess \( \hat{x}[n] \)) for which you can make the error \( e[n] \) resulting from the input \( x[n] = \beta nu[n] + \alpha u[n] \) go to 0 as \( n \to \infty \).

   (d) Show that your suggested alternate system works (i.e. achieves \( e[n] \to 0 \) as \( n \to \infty \)) in MATLAB.
2. **Multirate DSP Introduction with Speech Data**: Even though we hear frequencies well up to around $20 kHz$, most of the information we derive from human speech is contained in a band from about $300 Hz$ to $4 kHz$. In this problem you will take a speech signal (link next to the link for this homework) recorded at a moderate rate (16kHz), and resample it to a low sample rate $8 kHz$ for storage. You will convert your stored samples at $8 kHz$ back into samples at a rate of $16 kHz$. Submit your MATLAB code along with plots and written answers to the questions. *Make sure you answer all of the questions.*

(a) **Lowering the Sample Rate** Investigate lowering the sample rate 2 ways:

i. by simply discarding every other sample (decimation by 2), and

ii. by first low pass filtering the $16 kHz$ signal with an anti-aliasing filter (with a cut off frequency corresponding to $4 kHz$) then decimating by 2. Design your low pass filter with the commands `firls` or `firpm` (we will learn more about these commands later). Plot the frequency response to the filter you created using the command `freqz`. Make sure that the low pass filter you designed looks like a low pass filter with the proper cut off frequency. If it doesn’t be sure to tweak your arguments to the functions. From a computational standpoint is a system like this (filter followed by decimation) efficient? Why not? (We will learn about an efficient alternative next week.)

In both cases, plot the magnitude of the Fourier transform of the signal (using either `fft` or `freqz`) before and after lowering the sampling rate. The MATLAB commands `wavread` and `soundsc` will be useful for loading a wav speech file and listening to it, respectively. What differences can you observe between the two cases?

(b) **Raising the Sample Rate**

i. “Upsample” your $8 kHz$ signal (whichever one you found to be higher quality by listening) by a factor of 2. That is, if x is the matlab vector containing your $8 kHz$ speech data, create a new vector with the two commands `y=zeros(1,length(x)*2);` and `y(1:2:end)=x;`. Plot the magnitude of the Fourier transform of this upsampled signal using `fft` or `freqz`. How is it related to the Fourier transform of your $8 kHz$ sampled signal? Justify your observation using an argument involving sampling.

ii. Low pass filter your upsampled signal with the filter your anti-aliasing filter designed when you lowered the sample rate. Plot the Fourier transform of this signal, and compare it with the Fourier transform of the original $16 kHz$ signal. Listen to it with `soundsc`. How does it compare with the original signal? In what way is such a system (upsample followed by a filter) inefficient? (Again, we will learn about an efficient alternative next week.)

(c) Use the MATLAB `resample` command to lower and raise the sample rate as done in the previous two parts. Plot the associated Fourier transforms and compare with the ones you obtained. Also, roughly compare the run time of the resample command with the run time of your code.