Homework #1, EE 351
Spring 2004

Due: Friday, January 23 by 4 pm in the homework slot outside 121 EE East

The purpose of this first homework is to examine the differences between a continuous-time, discrete-time, and discrete-time/amplitude (digital) signals. This HW also illustrates some of the MATLAB commands/structures that you will use throughout this semester.

Consider the following discrete-time system which is used to remove DC offset from a signal using a highpass digital filter.

1. First, we'll just look at the first 2 blocks.
   a) Assume that the input signal is given by \( x_c(t) = 3\cos(2\pi \cdot 500t) + 1 \). Calculate the first 8 values (n=0, 1, ..., 7) of \( x_q[n] \) if the sampling period of this system is \( T = 125 \mu \text{sec} \) (0.000125 seconds). This is the same as saying the sampling rate is 8000 (1/0.000125) Hz.

   b) Assume that the quantizer truncates each sample value to the closest smaller integer, rounding to \(-\infty\) (e.g. -9.25 would truncate to -10). Draw the corresponding quantizer map and use your results in (a) to calculate the first 8 values of \( x_q[n] \).

   c) Without changing the step size of the quantizer in (b), how else could you change the quantizer to minimize the maximum possible quantization error? Draw the corresponding quantizer map and determine the maximum quantization error magnitude for this case.

2. Use the help command in MATLAB to learn what the commands fix, floor, ceil, and round do.
   a) For each of these MATLAB commands, sketch the appropriate quantizer map (using \( \Delta = 1 \)).
   b) Which of these commands is equivalent to the quantizer described in 1b)?

   Hint: if you can't figure out exactly what each command does from the description alone, try each command out in MATLAB with some various values.
3. Now we’re going to simulate the system above using MATLAB. We’ll run the simulation for 5 ms. The basic code is given below. You’ll have to add/change parts of it in the questions below.

a) Use a text editor to type in the following m-file. Call the file hw1.m.

```matlab
% EE 351 - Spring 2004
% MATLAB script m-file
% Any text preceded by a percent sign (%) is a comment and is ignored by MATLAB

clear   % It is a good idea to clear your MATLAB memory when starting a new program

% Set up a simulation time base t from 0 to 5 ms
% The increment value (dt) should be a very small number. 0.00001 should be ok in this course.
dt = 0.00001;   % The semi-colon at the end of every command suppresses screen printing
tstart = 0;    % tstart and tend should both be multiples of td
tend = 0.005;
t=[tstart : dt : tend];  % The notation for vectors is [start value: increment value: stop value].

% Generate a “continuous-time” message signal xc(t) = 3*cos(2*pi*500*t)+1.
% Note: Because MATLAB is a computer program, this generated signal is not technically a
% continuous-time signal. Nevertheless, we will call it a “continuous-time” signal.
xc=3*cos(2*pi*500*t)+1;

% To plot this “continuous-time” signal we use the MATLAB command plot.
figure(1)   % each new figure should be preceded by the figure command
plot(t,xc)  % plot( ) connects data points with a smooth curve
xlabel('time')
ylabel('xc(t)')
title('input signal')

% Now we sample xc(t) by defining a sample period T = 125 us (which is a sampling frequency of
% 8000 samples/second), setting up the corresponding discrete-time base n, and finally
% replacing t by n*T. The sample period T should always be a multiple of the increment value
% defined earlier. Likewise, the start and end times MUST be multiples of T.
T = 0.000125;
n = [tstart/T : tend/T];   % This vector is composed entirely of integers
xd=3*cos(2*pi*500*n*T)+1;  % xd[n] is generated by replacing t with n*T

% We plot discrete-time signals like xd[n] above using the stem command in MATLAB.
figure(2)
stem(n,xd)  %stem( ) does not connect data points together
xlabel('sample number')
ylabel('xd[n]')
title('sampled signal')
```

b) Start MATLAB and change the working directory (using the CD command) to the directory
which contains the file you just edited. Execute the m-file by typing its name. The 2 plots should
appear on the screen. Obtain a **hard copy** of the 2 generated plots. If you prefer, you can edit the file to put the 2 plots on the same page (using the `subplot` command in MATLAB).

c) Edit the program above to implement each of the 4 types of integer quantizers (see problem 2). Call the quantizer outputs \( xq1[n] \), \( xq2[n] \), \( xq3[n] \), and \( xq4[n] \) and obtain a **hard copy** of the stem plot of the output for each of the quantizers. You can put all 4 plots on the same page. Clearly label what type of quantizer was used in each case.

d) For each of the quantizers in (c), generate and plot the error signal \( xe[n]=xq[n]−xd[n] \). Obtain a hard copy of each error signal and comment on the results.

e) For the case where we do rounding quantization, repeat (c) but use a quantizer with step size \( \Delta = 0.25 \) instead of \( \Delta = 1 \). Call this signal \( xq5[n] \). Generate the error signal as well. Obtain a **hard copy** of both the output and the error (plotted on the same page) and comment on your results.

4. Now we’ll look at the last part of the system. To remove the DC component of this signal, we will use a particular digital filter called a **highpass Butterworth filter**, which is studied in EE 453 (a follow-up course). Digital filters are implemented using **difference equations** (a discrete-time counterpart to a differential equation and the basic building block of all discrete-time systems; difference equations will be studied in detail later in this course). The pre-defined MATLAB function `filter` is useful for generating the output of a difference equation. The format of this function (assuming no initial conditions) is as follows:

\[
y = \text{filter}(b, a, x) \quad \text{-- passes the data in vector } x \text{ through the difference equation described by vectors } a \text{ and } b \text{ (which must be pre-defined in your program) to generate the filtered data } y. \quad \text{The number of output points calculated will be equal to the number of input points. Vectors } b \text{ and } a \text{ contain the coefficients of the difference equation. For example, if } a=[1 \ 2 \ 3] \text{ and } b=[4 \ 5 \ 6], \text{ then the following difference equation would be calculated: } y[n] + 2y[n-1] + 3y[n-2] = 4x[n] + 5x[n-1] + 6x[n-2]. \quad \text{NOTE: The coefficient associated with } y[n] \text{ is always assumed to be 1.}
\]

Using the `filter` command as described above, edit your program to generate appropriate vectors \( a \) and \( b \) (see the discussion above) and pass the quantized discrete-time signal generated in 3e (\( xq5[n] \)) through a digital filter described by the difference equation below to generate the output \( y[n] \). Note: You will learn how to analyze these digital filters later in this course and you’ll learn how to design them in a follow-up course (EE 453).

\[
y[n]−1.7786y[n−1]+0.8008y[n−2]=0.8949x[n]−1.7897x[n−1]+0.8949x[n−2]
\]

Generate a stem plot of \( y[n] \). Execute this program and obtain a **hard copy** of the stem plot. Compare the signal plot before and after passing it through this filter. Does the filter remove the DC component completely or is there some distortion? Discuss.

**** **Turn in your complete m-file and all requested plots with your homework.****