ECEn 380 — Laboratory Assignment 5
Discrete Time Filtering

Due Date:

You will have a bit more than two weeks to complete this laboratory assignment. The lab exercises are organized into 3 main tasks: 1) learn about discrete-time filter design, 2) design a discrete-time FIR lowpass anti-aliasing filter, and 3) design a bank of 10 discrete-time IIR bandpass filters. The tasks of this lab will prepare you for ECEn 390's *laser tag* project. Work through the exercises and neatly track your progress in your lab book. Save all of your MATLAB code, and answer all questions in your lab book. A complete lab report is due on December 10th (for Section 001) or December 11th (for Section 002) **by 10:00 pm**. This means you can come to lab that week to get sign offs, but your lab report must be turned in later that same night.

The report must consist of the following to be deemed complete: 1) a detailed log of all laboratory exercises including derivations, code listings (which are quite important for this lab, so please include them), code output with well-labeled figures, answers to all questions posed in the lab, and thoughtful responses and/or descriptions of lab book content; 2) TA signatures verifying completion of each of the main tasks of the lab (1 signature for each of the last two tasks; Task 1 does not require a sign-off); 3) a summary paragraph of the exercises you completed in the lab; and 4) a conclusions paragraph identifying important principles from the lab and discussing what you have learned. This lab will be important to you again next semester when you are taking ECEn 390. Thus, sometime next semester you will either thank yourself or curse yourself depending on how well you commented your code and documented your experience in this lab assignment.

Prelab:

This lab brings together a number of different concepts, and as such, is a nice way to end the class. A major portion of this lab assignment is to review pertinent material from the class, and learn additional skills in filter design. Before showing up to your designated lab section, complete the following tasks:

1. read this entire lab assignment and **highlight important information that may be hard to find later**;
2. be familiar with MATLAB functions: `load`, `:`, `length`, `log10`, `exp`, `conv`, `fft`, `freqz`, `fftshift`, `zplane`, `filter`, `pwelch`, `window`, `rectwin`, `hamming`, `bartlett`, `chebwin`, `butter`, `ellip`, `cheby2`, `cheb2ord`, `plot`, `subplot`, `grid`, `xlabel`, `ylabel`, `axis`, `bar`;
3. be familiar with the following concepts from the class (review sections of your text where needed): analog Butterworth filters, sampling, discrete-time convolution, the z-transform, causality and stability, the DTFT, system frequency response, windowing, and basic discrete-time filter concepts;

Objectives:

The purpose of this assignment is to introduce you to the design and testing of discrete-time filters in MATLAB, and to apply these skills to designing the discrete-time filters that will be needed for the ECEn 390 laser tag project next semester.

The laser tag system supports ten different players, each assigned a different laser modulation frequency ranging from approximately 1.4 kHz to 4.2 kHz. In order to analyze the data in discrete time, we must use a sampling frequency that is greater than twice the highest player frequency. However, we don't want to use too high of a sampling frequency since we have limited processing power in the laser tag system. A sampling frequency of 10 kHz is a good choice, since it is above the Nyquist rate (2*4.2 kHz = 8.4 kHz).

However, the boards that you will be using in ECEn 390 capture data at a sampling frequency of 100 kHz (100,000 samples/sec). We will need to decimate, or down-sample, this data to 10 kHz (10,000 samples/sec). In Task 2, you will design a discrete time anti-aliasing (low pass) filter with a cut-off frequency of 4.6 kHz. You will apply this filter to data captured at a 100 kHz sampling frequency, so that you mostly eliminate frequencies over about 5 kHz (remember the cut-off will not be perfect at 4.6 kHz). You are then ready to take the filtered 100 kHz data and throw away 9 out of every 10 samples, yielding data sampled at 10 kHz. As previously mentioned, this process is called "decimation."

Once we have the laser tag signal data decimated to 10 kHz, we will pass the decimated signal through 10 different discrete-time bandpass filters, each with a passband centered around one of the player frequencies. In Task 3, you will design and test these filters. By comparing the signal energy of the 10 outputs from these filters, you should be able to determine whether there was a player "hit" in the signal, and which player it was.

Task 1—Introduction to Discrete-Time Filter Design: FIR and IIR Filters

Read the handout entitled "An Introduction to Discrete-Time Filter Design," by Dr. Michael Rice. Note that this document was written a few years ago while referencing a different textbook than L&G, but the meanings should still be clear as you read through material. Please pay special attention to the design of finite impulse response (FIR) filters (ending on page 18 of the handout). You should also read the remainder of the handout (pages 19 and beyond), which talks about the recursive, or infinite impulse response (IIR) filters. However, we are not going to emphasize much of what is covered for IIR filters, and are going to allow you to use the `butter` command in MATLAB to design your bank of 10 IIR bandpass filters. There will also be a short lecture that covers some of the important material of this handout at the beginning of lab.

Task 2—Design of a Discrete-Time Low-Pass Anti-Aliasing FIR Filter

In this task you will design your discrete-time anti-aliasing filter that will allow you to filter the 100 kHz sampled data as part of the decimation process to take it down to 10 kHz sampled data. The filter should have a length of 81 samples and a corner frequency of 4.6 kHz.

Procedure

As mentioned, all of this lab will be conducted using MATLAB.

1. Design several versions of an FIR filter with total length of 81 and corner frequency of 4.6 kHz.
    (a) Review the FIR filter design covered on the first 18 pages of the handout that you read in Task 1. Note that in the reading (starting on page 9), they actually show you how to design a band-pass filter (not a low-pass filter). All we need to do to use similar code to produce a low-pass instead of a band-pass filter is specify a center frequency of 0 (which makes the cosine in the derived filter impulse response always equal to 1, so it goes away and we are left with a simple sinc function as our impulse response, which we would expect). Note that in the handout the passband width

is given by B cycles-per-sample, but with a center frequency of zero, the resulting lowpass filter has a corner frequency (f_corner) of $B/2$ cycles-per-sample, so we let $B = 2 *$ f_corner for our desired corner frequency.

(b) First, design an "ideal" filter with no additional windowing, using the following code:

```
N = 81; % total number of samples in the filter
L = (N-1)/2; % the filter will go from -L to L
n = (-L:L); % this is our sample index

f_corner = 4600; % corner frequency of our low-pass filter in Hz
f_s = 100000; % our sampling frequency in Hz

h_ideal_FIR = 2*f_corner/f_s*sinc(n*2*f_corner/f_s); % ideal sinc
% IMPORTANT: in the code above we convert f_corner from
% Hz to cycles/sample by dividing by f_s

stem(n, h_ideal_FIR);
```

Now look at the frequency response of this "ideal" filter over a frequency range from 0 to 10 kHz:

```
f_axis = linspace(0, 10000, 2000); % frequency axis with 2000 points
H_ideal_FIR = freqz(h_ideal_FIR, 1, f_axis, f_s);
% IMPORTANT: in the code above, the first two parameters
% to freqz are the 'b' and 'a' coefficients for the filter.
% Remember that the 'b' coefficients for an FIR filter
% are just the impulse response values, and a_0 = 1 while
% all of the other 'a' coefficients are zero.
plot(f_axis, 20*log10(abs(H_ideal_FIR)));
xlabel('frequency (Hz)'); ylabel('magnitude (dB)');
```

(c) Show both the frequency response and a stem plot of the impulse response of this filter in your lab notebook. Comment in your lab notebook on how "ideal" the frequency response of this low-pass FIR filter is. Does your corner frequency match the -3dB point? Is there much "ripple" in the passband of the filter? What about the side lobes?

(d) Now add to your code above to design 3 additional filters using any 3 of the different windows shown on page 15 of the filter design handout (one of them should be the Hamming window). Note: You will need to transpose the output of the window functions to get the result to match the size of h_ideal_FIR, which is 1x81. By default, the window functions will produce something that is 81x1.

(e) Plot both the stem plots and the frequency response of each of these filters and include them in your lab notebook. You can make use of the MATLAB hold on command if you wish to reduce the total number of graphs. Comment on how the frequency response is improved when you use a window. Which of your filters do you think will perform the best?

(f) **IMPORTANT: Save your MATLAB code! You will want it for ECEn 390 next semester.**

2. Test all four of your filters using input sampled data with sinusoids at a variety of frequencies.

(a) Create a signal sampled at 100 kHz that is 500 msec in duration and contains sinusoids at frequencies 4 kHz, 4.6 kHz, 6 kHz, 25 kHz, and 40 kHz using the following code:

```
t = 0:1/f_s:(0.5-1/f_s); % time axis in seconds
sin1 = sin(2*pi*4000*t); % sinusoid at 4 kHz
```

```
sin2 = sin(2*pi*4600*t); % sinusoid at 4.6 kHz
sin3 = sin(2*pi*6000*t); % sinusoid at 6 kHz
sin4 = sin(2*pi*25000*t); % sinusoid at 25 kHz
sin5 = sin(2*pi*40000*t); % sinusoid at 40 kHz
s1 = sin1+sin2+sin3+sin4+sin5; % create the overall signal
```

(b) Plot this signal in the frequency domain using the following code:

```
S1 = fft(s1); % take the FFT of the signal s1
f1 = linspace(0, 50000, length(S1)/2); % freq axis from 0 to 50 kHz
plot(f1, abs(S1(1:length(S1)/2)));
```

Show this plot in your lab notebook. Does it look like you would expect?

(c) Now filter this signal through each of your four filters using the MATLAB `filter` command, and then plot each of the filtered signals in the frequency domain (using code similar to that in (b) above. Include each of these plots in your lab notebook, and comment on how well each filter performed.

Task #2 Pass-Off: Show the TAs your code, and the stem plots and the frequency response of all four of your filters. Also show the TAs the frequency domain plots of the signal `s1`, and the frequency domain plots of the four filtered versions of `s1`.

Task 3—Design and Test Ten Band-Pass IIR Filters

In this task, you will design 10 band-pass IIR filters, each with a center frequency around one of the 10 player frequencies. The filters should each have a full bandwidth of 50 Hz (plus or minus 25 Hz from the center frequency), and the filters will have a total length of 11. Use the following 10 center frequencies for players 1–10 respectively:

1471, 1724, 2000, 2273, 2632, 2941, 3333, 3571, 3846, 4167 (all in Hz).

You will then test your bank of 10 filters on some sample data and see if you can identify a player "hit."

Procedure

Create a new .m file for this task, and clear your MATLAB workspace.

1. Design your 10 IIR filters with appropriate center frequencies. When we say "design your filters," we mean determine the `a` and `b` coefficients for each filter. There will be 11 of each type of coefficient for each filter. Below we show you how to create the first filter at the Player 1 frequency. (You should be clever in coding this into a loop to create all ten of your filter. You might think about producing arrays of `a` and `b` coefficients that are 11×10 to store the values for all of your filters. You will need to modify some of the variables in the code below to do this.)

```
 f_players = [1471, 1724, 2000, 2273, 2632, 2941, 3333, 3571, 3846, 4167];
% Player center frequencies in Hz
f_s = 10000; % sampling frequency (remember we have decimated)
BW = 50; % Full bandwidth of filters in Hz
f_lower = f_players(1) - BW/2; % Lower corner frequency of BPF
f_upper = f_players(1) + BW/2; % Upper corner frequency of BPF
```

```
[b1, a1] = butter(5, [f_lower*2/f_s, f_upper*2/f_s]);
% IMPORTANT: 'butter' takes frequency specifications in half cycles/sample,
% not Hz, hence the division by f_s and multiplication by 2

f_axis = linspace(1000, 4500, 2000); % create a freq axis for plotting from 0 to 5 KHz
H1 = freqz(b1, a1, f_axis, f_s); % calculates the freq response H
% at the frequencies in f_axis
plot(f_axis, abs(H1)); % frequency response
```

2. Plot the frequency response of all 10 of your filters on a single graph (using `hold on`) and include this in your lab write-up. Keep the vertical axis in absolute units (not dB), as shown in the code above. Do they look correct?

3. **IMPORTANT: Save your MATLAB code so you can quickly generate the `a` and `b` coefficients for all 10 filters for 390 next semester!**

4. Load the provided sample data (`lab5_data_sets.mat`) into MATLAB, which includes 10 sample signals. Play around with filtering each of these 10 sample signals through each of your 10 filters (again using the MATLAB `filter` command). For each sample signal, say $x[n]$, calculate the total energy

$$E_i = \sum_n |y_i[n]|^2, \tag{1}$$

where $y_i[n]$ is the output of the ith filter when $x[n]$ is the input. Create a bar graph showing these 10 energies for each of the 10 sample signals. Include this bar graph for two of the sample signals in your lab notebook, one of the easy ones and one of the hard ones. Can you identify a player hit in the sample data? If so, what player in each case?

Task #3 Pass-Off: Show the TAs your code, the frequency response of all 10 of your IIR bandpass filters, and the bar plots showing the filtered signal energies for each of the 10 filters for an easy signal and a hard signal.

Before turning in your lab report, complete all lab entries thoroughly and include all required work as stated in the beginning of this lab assignment (2nd paragraph).