ECE 320 SIGNALS & SYSTEMS II

Matlab Project 2

Spring 2017

Assigned: Thursday, April 13, 2017 **Report Due:** Thursday, May 4, 2017

The purpose of this project is to learn one type of FIR filter design. You will develop a Matlab function to design an FIR filter using the window method. This method generates the impulse response of an FIR filter by truncating the impulse response of the ideal discrete-time lowpass filter. You will explore how the filter length and the type of window used for truncation affect the filter's frequency characteristics. Using the modulation theorem, you will also use the prototype lowpass filters you design to create bandpass and highpass filters. Finally, you will apply the filters you design to process a noisy audio signal.

Please read the lab report guidelines on the ECE 320 website before beginning the project. Also, take a look at the Appendix for a description of how to use the `filter` and `freqz` commands.

You may use either the `freqz` or `fft` commands to compute the frequency responses for this project. Also you may use either the `filter` or `conv` command to implement the filtering for this project. Please indicate which methods you decide to use in your report.

The preliminary research part of the project (Section 1) is to be done **individually**. The remainder of the project (Sections 2-6) may be done in pairs. If you choose to work with a partner, please notify Prof. Wage via email by Thursday, April 20.

1 Preliminary Research **20 points**

The purpose of this preliminary research project is to learn about finite impulse response (FIR) filters, which you will use in the remainder of the project. For this part you will prepare and submit a short report. The report should answer the following questions:

- How are FIR filters designed? This project will use one approach (window design), but there are other approaches. Provide a brief description of the other approaches you find. State the advantages and disadvantages of each approach.
- What types of applications use FIR filters?
- Search the IEEE Xplore database for references on FIR filter design. Provide brief summaries of **three** papers that discuss these topics.
- In addition to the technical information, your report should include a discussion of how you located the references you used. Which sources were most helpful to you? Why? Are there any other types of sources you could potentially consult?

Your report should be typed. It must be 2-4 pages. See the IEEE style guide for information on citations: `http://www.ieee.org/documents/ieeecitationref.pdf` Important reminders:

- The 4-page limit includes **everything**: references, title and author information, etc.
- Your references must be cited in the technical discussion, not just at the end of the paper.
- Your preliminary research project should be submitted in a file of its own, i.e., separate from the rest of the project.

Reminder: this part must be done indvidually. Each student must turn in their own preliminary research project.

2 Analytical Calculations **5 points**

Consider the ideal discrete-time lowpass filter with the frequency response $H_{\text{lpf}}(e^{j\Omega})$ shown below.

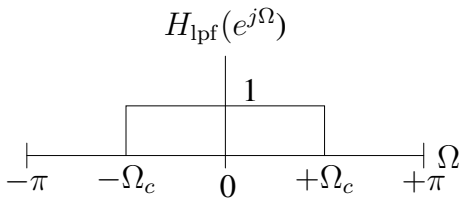

Figure 1: Ideal discrete-time lowpass filter frequency response.

(a) Using the inverse Fourier transform, determine the impulse response $h_{\text{lpf}}[n]$ for the ideal lowpass filter. Make a fully-labeled sketch (by hand) of this impulse response.

3 FIR Filter Design via the Window Method **30 points**

Note that the impulse response of the ideal lowpass filter is infinitely long in both directions, *i.e.*, in general it is non-zero from $n = -\infty$ to $+\infty$. In this section you will design a lowpass filter by truncating the impulse response of the ideal filter. While the resulting filter no longer has the "ideal" frequency response, it can work sufficiently well to be useful in a number of applications.

In window design the filter impulse response is created by multiplying the ideal impulse response by a finite-length "window", *i.e.*,

$$h[n] = w[n]h_{\text{lpf}}[n], \tag{1}$$

where the window $w[n]$ is non-zero for $-M \leq n \leq +M$. This results in an impulse response that has $2M + 1$ non-zero points. The simplest choice for a $w[n]$ is the rectangular window, which is defined as

$$w_{\text{rect}}[n] = \begin{cases} 1 & -M \leq n \leq +M \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Given that $h[n]$ is the multiplication of $w[n]$ and $h_{\text{lpf}}[n]$, the resulting filter frequency response is the convolution of the transform of the window with the ideal lowpass filter characteristic, *i.e.*,

$$H(e^{j\Omega}) = \frac{1}{2\pi} W(e^{j\Omega}) * H_{\text{lpf}}(e^{j\Omega}). \tag{3}$$

(a) Analytically determine the transform of the rectangular window $w_{\text{rect}}[n]$. Make a sketch (by hand) of this transform. Comment on how the transform changes as the length of the window increases.

(b) In Matlab $w_{\text{rect}}[n]$ is simply a vector containing $2M + 1$ ones. It can be generated using the `ones` command or the `rectwin` command. Generate rectangular windows of length 11, 51, and 101 points and store them in separate vectors in Matlab. Use either the `fft` method (from Project 2) or the `freqz` command to compute 1024 points of the frequency response for each of these windows. Plot the frequency response magnitudes for each of windows on a single plot. How do they compare? Were your predictions of how the transform changes with the length of the window correct?

Plotting hint: use different linestyles for each of the frequency responses, and label your plot using the `legend` *command. You can use different linestyles as follows:*

```
plot(omega/pi,abs(W11),'b-',omega/pi,abs(W51),'r--',omega/pi,abs(W101),'g-.')
legend('11pt rectwin','51pt rectwin','101pt rectwin')
```

(c) In this part you will create 3 lowpass filters using the 3 windows you defined in the previous section. The cutoff frequency of these lowpass filters should be $\frac{\pi}{4}$. To design the filters you will need to generate samples of $h_{\text{lpf}}[n]$ for $-M \leq n \leq +M$ and then multiply those samples by the window $w_{\text{rect}}[n]$. Be careful defining $h_{\text{lpf}}[n]$! When $n = 0$ you have to use L'Hopital's rule to determine the value. The Matlab code you write should account for this. For this part, you are using rectangular windows (which consist of a vector of 1's), so multiplying your `hlpf` vector by the window isn't really necessary. We'll generalize this later, so it's probably a good idea to go ahead and include the multiplication by the window vector.

(i) Use the `stem` and `subplot` commands to plot the impulse response of the 3 filters you defined.

(ii) Use the `fft` or `freqz` command to find the frequency response of the 3 filters. Plot these on 3 subplots.

(iii) Compare the 3 filters and describe their frequency responses. Given the relationship between $H(e^{j\Omega})$ and $H_{\text{lpf}}(e^{j\Omega})$ (defined in Eq. 3), are the results what you were expecting? How does the length of the window affect the frequency response?

(d) The rectangular window simply truncates the ideal lowpass filter impulse response. Other types of windows used in filter design scale the values of $h_{\text{lpf}}[n]$ in addition to truncating the response. One such window is the Blackman window. This window is implemented by the `blackman` function in Matlab.

(i) Generate 3 Blackman windows of lengths 11, 51, and 101. Plot these windows using the `stem` and `subplot` commands. Comment on how these windows look compared to the rectangular window.

(ii) Plot the frequency response magnitudes of these 3 Blackman windows and compare them to the frequency response magnitudes of the corresponding rectangular windows you used in the previous part. To make the comparisons a little easier, you may want to do the plots on a log scale, *i.e.*, `plot(omega/pi,10log10(abs(Wblack)))`. Include a discussion of your comparisons in the report. Based on the Fourier transform of the Blackman windows, how do you expect the resulting filter frequency responses to look?

(iii) Use the 3 Blackman windows to design lowpass filters with a cutoff frequency of $\frac{\pi}{4}$ and lengths of 11, 51, and 101. Plot the frequency response magnitudes of these filters and compare them to the results you obtained using the rectangular windows. Were your predictions correct? (Again: using a log scale for these plots may be helpful.)

(iv) Discuss the relative merits of the rectangular window design versus the Blackman window design. Does one type of filter perform better than the other? If so, how? Comment on how you would decide which window to use.

(e) In this part you will write a function to implement window design of lowpass filters. The first line of your function will be

```
function [h,n]=firwinlpf(wc,M,winfxn)
```

The input to the function includes the cutoff frequency `wc` of the discrete time filter (in rad/sec), an integer `M` that defines the length of the filter (length=2M+1), and a string `winfxn` containing the name of the window function to use. The output of the function is the vector `h` containing the impulse response of the filter and the vector `n` containing the time indexes associated with `h`, *i.e.*, `n=-M:1:M`.

Note that your function will be designed to use any of the standard windows defined in Matlab, since you just provide the name of the window as a string variable. In order to generate the window, you will need to use the `feval` command, which allows you to call a function by name. Type `help feval` for additional information on how to use this command.

(f) Test your function. Describe your testing procedure. Include a description of how you verified that your function was working and a few plots to show the results. You do not need to include plots for all the tests you do (though you do need to describe the tests). Include just a couple plots to demonstrate that the function is working correctly.

4 Design of Bandpass and Highpass FIR Filters **15 points**

In the previous section you developed a function to design lowpass filters via the window method. In this section you will explore how a lowpass filter can be turned into bandpass and highpass filters. Consider designing a new filter by modulating a prototype filter with a cosine as as follows:

$$h_{\text{new}}[n] = Ah[n]\cos(\Omega_0 n) \tag{4}$$

where $h[n]$ is an existing prototype filter, Ω_0 is a known frequency, and A is a scale factor chosen to set the filter amplitude.

(a) Determine an analytical expression for $H_{\text{new}}(e^{j\Omega})$ in terms of $H(e^{j\Omega})$.

(b) For this part assume that $h[n]$ is the ideal lowpass filter with a cutoff frequency equal to $\frac{\pi}{3}$, and that $\Omega_0 = \pi$. First sketch the signal $\cos(\Omega_0 n)$, and then sketch the frequency response $H(e^{j\Omega})$ of the resulting filter.

(c) Use your `firwinlpf` function to design a 101-point lowpass filter with cutoff frequency $\frac{\pi}{3}$. Then create the impulse response of a new filter by multiplying it by $\cos(\pi n)$. (Remember that the Matlab expression `.*` (period followed by asterisk) allows you to do point-by-point multiplication of two vectors.) Use the `freqz` command to compute the frequency response of this new filter. Make a plot to include in your report. Does your plot correspond to what you expected based on your analysis in parts a and b?

(d) Suppose that you wish to use the technique described above to design a bandpass filter with the following characteristics

$$H_{\text{bp}}(e^{j\Omega}) = \begin{cases} 0 & |\Omega| < \frac{\pi}{5} \\ 1 & \frac{\pi}{5} \le |\Omega| \le \frac{3\pi}{5} \\ 0 & \frac{3\pi}{5} < |\Omega| \le \pi \end{cases}$$

(i) Sketch $H_{\text{bp}}(e^{j\Omega})$ for $-\pi \le \Omega \le +\pi$.

(ii) What cutoff frequency would you choose for the prototype ideal lowpass filter to use in this design? What value would you choose for Ω_0? Provide sketches to support your answers.

(iii) Using the values you determined in part ii and your `firwinlpf` function, generate the impulse response of a bandpass filter that approximately meets the specifications stated above. You can decide on the length of the filter and the type of window to use. Make plots of the impulse response and the frequency response magnitude of this filter to include in your report.

5 Audio Filtering Example **20 points**

Lowpass filtering problem
A Hollywood production company is preparing to re-release one of Prof. Wage's favorite old movies. Unfortunately, they're having problems cleaning up the audio track, which has been corrupted by highpass noise. They have asked signals and systems classes from around the country for help designing discrete-time filters to clean up the audio. Your job is to design lowpass filters to meet the production company's specifications and to apply these filters to a short segment of the audio track. You have the following information:

- The filter must pass all frequencies up to 3,500 Hz. The allowable amplitude distortion (ripple) in the passband is $\pm 2\%$., *i.e.*, $\delta_1 = 0.02$.
- Above 5,000 Hz, the filter must have an attenuation of at least 40 dB, *i.e.*, $20\log_{10}(\delta_2) = -40$.
- The corrupted audio signal is stored in the file `noisytrack.mat` on the course website. The file contains both the noisy audio signal stored in `xaudio` and the associated sampling rate `fs`.

In this project you will use your function `firwinlpf` to design a lowpass filter to meet these specifications.

(a) Determine and sketch the specifications of the discrete-time lowpass filter. In other words sketch the tolerances for $H(e^{j\Omega})$ for $-\pi \leq \Omega \leq +\pi$.

(b) Window Design
Try to design the filter using each of the five standard windows (rectangular, Bartlett, Hanning, Hamming, and Blackman).

For each window, you should design the **minimum** length filter that meets the specifications. It may not be possible to achieve the specifications with all of the windows. If certain windows do not work, explain why.

The writeups for each filter that does meet specifications should include the values of the design parameters sent to `firwinlpf` and plots of the magnitude and phase characteristics of the filter. In particular, there should be a zoomed plot of both passband magnitude and stopband magnitude, clearly showing amount of ripple and confirming that the design meets the specification at the pass/stop band edges. *It is important that the magnitude and phase characteristics are plotted as a function of continuous-time frequency (in Hz) so that proper comparisons can be made to the design specifications (which are also given in Hz). Note that the* `freqz` *command will return a frequency vector in Hz if you specify the sampling frequency.*

(c) Application: Filtering the noisy audio signal

(i) Load the noisy signal by typing `load noisytrack.mat`. Play the noisy signal using the command `soundsc(xaudio,fs)`, where `xaudio` is the signal vector and `fs` is the sampling frequency in Hz.

(ii) Filter the signal with each of the filters you designed in part b. Use the `filter` command to implement the filtering operation. (See the appendix for help on `filter`.) What does the filtered signal sound like? (Be specific and descriptive.)

(iii) How do your filters perform? Do they remove all of the noise? Does one filter peform better than the others or do they all produce identical results? Write up your comparison of the filter performance. If you notice differences, try to explain them based on what you know about the impulse and frequency responses of your filters. (Note: you may need to turn the volume up to hear differences. Alternatively, using a good pair of headphones might help.)

(iv) Please include plots of the noisy signal and the filtered signals in your report. You should plot them versus an appropriate time vector (remember the signals are sampled at a rate of `fs` Hz).

6 Lab Report **10 points**

Prepare a report according to the guidelines posted on the ECE 320 website.

If you are working in pairs, both members should submit a copy of the same report.

In addition to submitting the report, you will submit a separate .mat file containing the filter coefficients for each filter from Part 5 that meets specifications. The filters should be called `filter1`, `filter2`, etc. (count increases to include all filters that meet specifications). Your written report should reference the variable names used for each of your filters when it describes them. Your .mat file should be called `firstlastname1_firstlastname2.mat`, where firstlastname1 is the first and last name of the first team member and firstlastname2 is the first and last name of the second team member.

A Tutorial on using `freqz` and `filter`

Matlab's `freqz` function can be used to compute the frequency response of a discrete-time system. The `filter` function can be used to implement a linear constant-coefficient difference equation. This tutorial provides a guide to how to use `freqz` to plot the frequency response of an FIR system and `filter` compute the output of an FIR system given the input. First consider the FIR system that has the impulse response $h[n]$ given below:

$$h[n] = b_0\delta[n] + b_1\delta[n-1] + b_2\delta[n-2] + \cdots + b_M\delta[n-M]. \tag{5}$$

Taking the z-transform of $h[n]$, we can obtain the system function:

$$H(z) = b_0 + b_1z^{-1} + b_2z^{-2} + \cdots + b_Mz^{-M} = \sum_{m=0}^{M} b_mz^{-m}. \tag{6}$$

Recalling that the frequency response is equal to the z-transform for $z = e^{j\Omega}$, we can find the frequency response of the system:

$$H(e^{j\Omega}) = H(z)|_{z=e^{j\Omega}} = \sum_{m=0}^{M} b_me^{-j\Omega m}. \tag{7}$$

Recalling that $H(z) = \frac{Y(z)}{X(z)}$, we can find a difference equation for this system:

$$y[n] = \sum_{m=0}^{M} b_mx[n-m]. \tag{8}$$

Thus the impulse response, system function, and frequency response of an FIR system is completely defined by the set of b_m coefficients. These coefficients could be stored in a vector:

$$\mathbf{b} = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_M \end{bmatrix}. \tag{9}$$

Note that the general form of a linear constant-coefficient difference equation is:

$$\sum_{k=0}^{N} a_ky[n-k] = \sum_{m=0}^{M} b_mx[n-m]. \tag{10}$$

This difference equation is completely defined by its coefficients (a_k's and b_m's) and the numbers N and M, which indicate how many a_k's and b_k's there are. Equation 8 fits the general form if N=0 and $a_0 = 1$, *i.e.*, there is only 1 term on the left-side of the equation and it is equal to $y[n]$.

The `freqz` function is used as follows:

```
[H,omega]=freqz(b,a,L,'whole');
```

The variables `b` and `a` in the function call are vectors containing the b_m and a_k coefficients in the difference equation. The number `L` tells the function how many points of the frequency response to compute.

The points are equally spaced between 0 and 2π if the `'whole'` string included in the function call. The returned vector `H` contains the frequency response (in general a set of complex numbers) and the returned vector `w` contains the frequencies corresponding to the numbers contained in `H`. Thus, to plot the magnitude of the frequency response you would type `plot(w,abs(H))`. The phase can be plotted in a similar manner using the `arg` function in place of `abs`.

The `filter` function can be used in a similar fashion. For `filter` the function call is

```
y=filter(b,a,x);
```

where the `b` and `a` vectors are as defined above, and the vector `x` contains the input $x[n]$ to the system. The output of the system is returned in `y`.

Note that for an FIR system, the `a` vector contains only the number `1`, thus the function calls can be written as follows:

```
[H,omega]=freqz(b,1,L,'whole');
y=filter(b,1,x);
```

Use Matlab's built-in help function (*e.g.*, `help freqz`) for additional information about these functions.