Project 6: Filter Design of FIR Filters

You will write up a formal lab report in IEEE double-column format with figures integrated with the text. You are allowed four pages for this week's writeup. The exercises should written up in this week's lab report. You should **label** the axes of your plots, have a caption, and Figure number for every plot. Every plot should be referenced by Figure number in your text discussion. Include each plot *inlined* within your report.

Forgeries and plagiarism are a violation of the honor code and will be referred to the Dean of Students for disciplinary action. You are allowed to discuss lab exercises with other students, but you cannot give or receive any written material or electronic files. In addition, you are not allowed to use or copy material from old lab reports from previous semesters. Your submitted work must be your own original work.

1 Introduction to FIR Filter Design

The goal of this lab is to learn some methods for designing practical FIR filters in MATLAB. These filters will have a finite number of coefficients, and a frequency response that approximates an ideal frequency response shape. We will go through the following three steps:

1. *Windowing:* The concept of windowing is widely used in DSP when dealing with finite-length signals.
2. *Filter Specs:* The quality of a designed filter is measured by how closely the actual response matches the desired ideal response. Often the desired match is set prior to the actual filter design step by drawing a tolerance region around the ideal filter shape. Then the minimum-order filter that fits inside the tolerance region is designed.
3. *Design Methods:* Two very common approaches to FIR filter design are *windowing* and computer optimization. The `filterdesign` GUI in *SP-First* can do both.

There are many ways to approximate an ideal frequency response with a practical filter. For FIR filters the frequency response $H(e^{j\hat{\omega}})$ is a function of $\hat{\omega}$ that summarizes a LTI system's response to inputs such as complex exponentials and sinusoids. The MATLAB function `freqz.m` is used to compute samples of the frequency response in the *frequency domain*.[1] A filter design method produces filter coefficients $\{b_k\}$ for the time-domain implementation of the FIR filter as a difference equation

$$y[n] = b_0x[n] + b_1x[n-1] + b_2x[n-2] + ... + b_Mx[n-M]$$

so the `freqz` function is an essential step for making plots of the magnitude and phase of the designed frequency response and assessing how closely it matches the desired ideal response.

[1] If you are working at home and do not have the function `freqz.m`, there is a substitute available called `freekz.m`. You can find it in the *SP-First Toolbox*.

2 First part to FIR Filtering

2.1 Windowing

The concept of windowing is widely used in signal processing. The basic idea is ***to extract a finite section*** of a very long signal $x[n]$ ***via multiplication*** $w[n]x[n]$. This approach works if the window function $w[n]$ is zero outside of an interval. For example, consider the simplest window function which is the L-point *rectangular window* defined as

$$w_r[n] = \begin{cases} 1 & 0 \le n \le L-1 \\ 0 & \text{elsewhere} \end{cases} \tag{1}$$

The important idea is that the product $w_r[n]x[n+n_0]$ will extract L values from the signal $x[n]$ starting at $n = n_0$. Thus the following are equivalent

$$w_r[n]x[n+n_0] = \begin{cases} 0 & n < 0 \\ \cancel{w_r}^{1}[n]x[n+n_0] & 0 \le n \le L-1 \\ 0 & n \ge L \end{cases} \tag{2}$$

The name *window* comes from the idea that we can only "see" L values of the signal $x[n+n_0]$ within the window interval when we "look" through the window. Multiplying by $w[n]$ is looking through the window. When we change n_0, the signal shifts, and we see a different length-L section of the signal.

The nonzero values of the window function do not have to be all ones, but they should be positive. For example, the L-point Hamming window is defined as

$$w_m[n] = \begin{cases} 0.54 - 0.46\cos(2\pi n/(L-1)) & 0 \le n \le L-1 \\ 0 & \text{elsewhere} \end{cases} \tag{3}$$

The MATLAB function `hamming(L)` will generate a vector with values given by (3). A stem plot of the Hamming window would show that the values are larger in the middle and taper off near the ends.

- Make a stem plot of the Hamming window for $L = 23$, over the index range $0 \le n \le L-1$.
- Determine the maximum value of the window and the index location of the maximum. Also, determine the values of the window at $n = 0$, $n = 11$, and $n = 22$.
- The Hamming window is said to have *even symmetry.* What does this mean?

2.2 Ideal Filters and Practical Filters

Ideal Filters are given by their frequency response, consisting of *perfect* passbands and stopbands. The impulse responses of the ideal LPF are infinitely long sinc functions. They cannot be FIR filters with a finite set of coefficients because ideal LPFs are realized by the following DTFT pair:

$$h_i[n] = \frac{\sin(\hat{\omega}_c n)}{\pi n} \quad \Longleftrightarrow \quad H_i(e^{j\hat{\omega}}) = \begin{cases} 1 & |\hat{\omega}| \le \hat{\omega}_c \\ 0 & \hat{\omega}_c < |\hat{\omega}| \le \pi \end{cases} \tag{4}$$

where $\hat{\omega}_c$ is the *cutoff frequency* of the ideal LPF, which separates the passband from the stopband.
In the `dltidemo` GUI, you can choose ideal lowpass filters (LPF), highpass filters (HPF) and bandpass filters (BPF) because the GUI does not use the impulse response. The ideal LPFs and HPFs have one parameter for the cutoff frequency. The ideal BPF has a parameter for center frequency which determines where the

band is located; its bandwidth (in the `dltidemo` GUI) is always 0.4π. All the ideal filters have an additional parameter for the slope of the phase of $H(e^{j\hat{\omega}})$.

Practical Filters are causal length-L FIR filters whose filter coefficients are chosen so that the resulting frequency response will closely approximate the desired frequency response of an ideal filter. The process of choosing the filter coefficients is called *filter design.* The practical FIR filters shown in `dltidemo` were designed using MATLAB's `fir1` function for digital filter design. The GUI offers length-15 LPFs and HPFs, and length-21 BPFs. The LPF and HPF designs depend on specifying one parameter for the cutoff frequency which lies midway between the non-ideal passband and stopband. The BPF requires two parameters: one for center frequency which determines where the passband is located, and other for the width of the passband. In the `dltidemo` GUI, the default BPF cutoffs are $\pm 0.2\pi$ from the center frequency, so only the center frequency can be changed. These practical filters do not match ideal filters exactly, and this is readily apparent at the cutoff frequencies where the frequency response has a magnitude of 0.5.

2.2.1 Truncate the Ideal Impulse Response

One simple approach to designing a practical FIR filter is to truncate the impulse response of an ideal filter. This can be accomplished with a window function, so we usually say that the practical FIR filter has an impulse response that is a *windowed* version of the ideal impulse response. For example, we could make a length-23 FIR lowpass filter by taking the center portion of the sinc function:

$$h_1[n] = w_r[n]h_{\text{IDEAL}}[n-11] = \begin{cases} \dfrac{\sin(\hat{\omega}_c(n-11))}{\pi(n-11)} & 0 \le n \le 22 \\ 0 & \text{elsewhere} \end{cases} \tag{5}$$

where $w_r[n]$ is a 23-point rectangular window. The sinc function must be time-shifted to put its peak in the middle of the window at $n = 11$, because we want the system defined by $h_1[n]$ in (5) to be causal. It is common that the ideal impulse response is time shifted by an amount equal to half the window length.

- For the time-windowed sinc function in (5), set $\omega_c = 0.7\pi$. Then make a stem plot of the impulse response $h_2[n]$.
- Use MATLAB to determine (samples of) the DTFT of $h_2[n]$ and make a plot of the DTFT magnitude, $|H_2(e^{j\hat{\omega}})|$.
- We could experiment with different window functions. Change the window to the 23-point Hamming window, and define a new impulse response as

  $$h_3[n] = w_m[n]\left(\frac{\sin(\hat{\omega}_c(n-11))}{\pi(n-11)}\right)$$

  Plot the impulse response.
- Use MATLAB to determine (samples of) the DTFT of $h_3[n]$ and make a plot of the DTFT magnitude, $|H_3(e^{j\hat{\omega}})|$. Explain why this "practical filter" is a better LPF than $H_2(e^{j\hat{\omega}})$ when evaluated as an approximation to the ideal LPF.

2.3 GUI for Filter Design

The *SP-First* GUI called `filterdesign` illustrates several filter design methods for LPF, BPF and HPF filter. The interface is shown in Fig. 1. Both FIR and IIR filters can be designed, but we will only be interested in the FIR case which would be selected with FIR button in the upper right. The default design

method is the *Window Method* using a Hamming window. The window type can be selected from the drop-down list in the lower right. To specify the design it is necessary to set the order of the FIR filter and choose one or more cutoff frequencies; these parameters can be entered in the edit boxes.

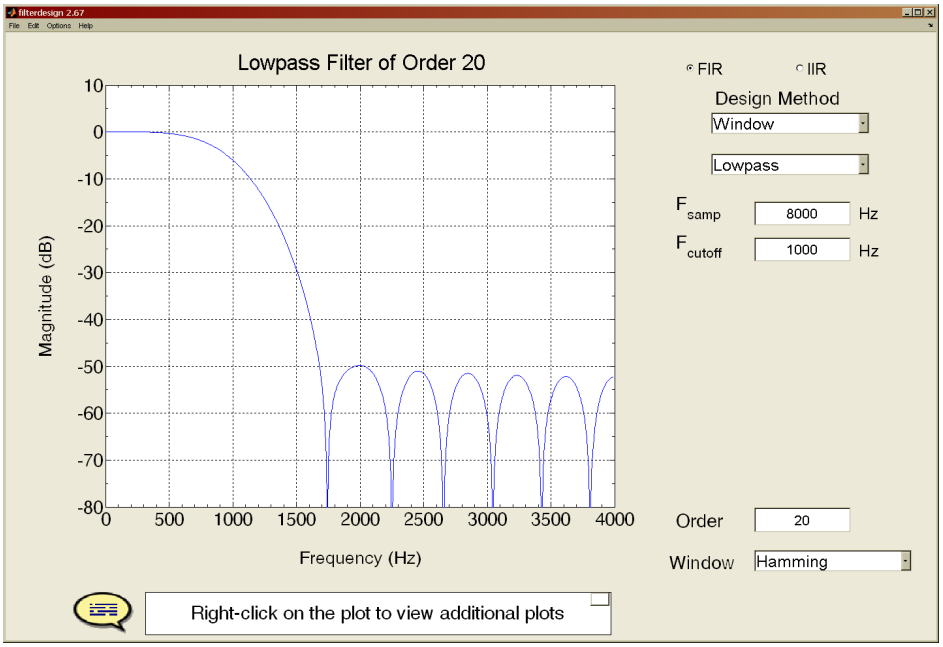

Figure 1: Interface for the `filterdesign` GUI. When the Filter Choice is set to `FIR`, many different window types can be selected, including the Hamming window and the Rectangular window (i.e., no window). The specification of one or more cutoff frequencies (f_{co}) must be entered using continuous-time frequency (in Hz), along with a sampling rate (f_s, also in Hz). In normalized frequency, the cutoff frequency is $\hat{\omega}_{co} = 2\pi(f_{co}/f_s)$.

The plot initially shows the frequency response magnitude on a linear scale, with a frequency axis in Hz. Clicking on the word Magnitude will toggle the magnitude scale to a log scale in dB. Clicking on the word Frequency will toggle the frequency axis to normalized frequency $\hat{\omega}$, and also let you enter the cutoff frequency using $\hat{\omega}$. Recall that $\hat{\omega} = 2\pi(f_{co}/f_s)$. The plotting region can also show the phase response of $H(e^{j\hat{\omega}})$, or the impulse response of the filter $h[n]$. Right click on the plot region to get a menu.

The filter coefficients can be "exported" from the GUI by using the menu `File->Export Coeffs`. To have some filters for comparison, redo the designs from the Section 2.2.1, and export the filter coefficients to the workspace under unique names. Then you can make your own plot of the frequency response in MATLAB using the `freekz` function (or `freqz`) followed by a `plot` command.

The Options menu provides zooming and a grid via `Options->Zoom` and `Options->Grid`.

2.4 Design Filters with the `filterdesign` GUI

For practice, use the `filterdesign` GUI to design two lowpass FIR filters with order $M = 22$ (or length $L = 23$). Use a cutoff frequency of $\hat{\omega}_c = 0.32\pi = 2\pi(1600/10000)$. Create one using a Hamming window, the other with a Rectangular window which should give a result like Fig. 2. Right click on the plot to see options for displaying the impulse response either windowed or unwindowed. The unwindowed version just displays the truncated sinc function, i.e., rectangular windowing.

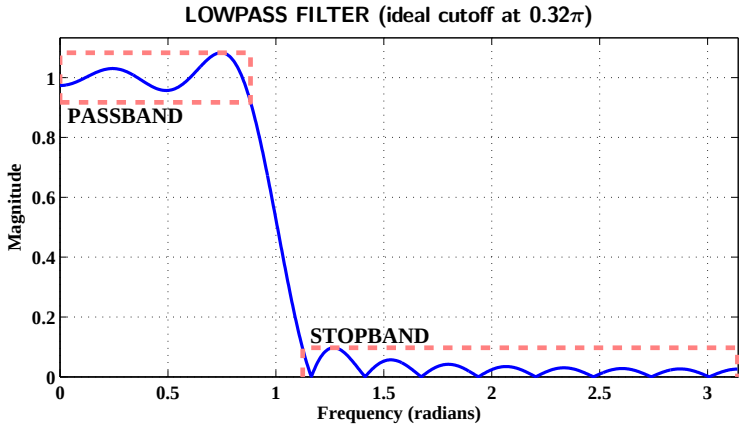

Figure 2: Passband and stopband defined for a typical lowpass filter. This one is of length 23 designed with a rectangular window and a sinc function with a cutoff frequency of $\hat{\omega}_c = 0.32\pi$. The passband and stopband ripples are defined to be 0.1, from which the passband and stopband edges can be measured. The approximate value of the passband edge is $\hat{\omega}_p = 0.281\pi \approx 0.883$; the stopband edge, $\hat{\omega}_s = 0.358\pi \approx 1.125$.

2.4.1 Passband Defined for the Frequency Response

Frequency-selective digital filters, e.g., LPFs, BPFs and HPFs, have a frequency response magnitude that is close to one in some frequency regions, and close to zero in others. For example, the plot in Fig. 2 is a lowpass filter whose magnitude is close to one when $0 \leq \hat{\omega} < 0.883$. This region where the magnitude is close to one is called the ***passband*** of the filter. It will be useful to have a precise definition of the passband edges, so that the passband width can be measured and we can compare different filters.

- From the plot of the magnitude response, e.g, in MATLAB or in the `filterdesign` GUI, it is possible to determine the set of frequencies where the magnitude is very close to one, as defined by $\big||H(e^{j\hat{\omega}})| - 1\big|$ being less than δ_p. This deviation from one is called the *passband ripple*. A common choice for the passband ripple is between 0.01 and 0.1, i.e., 1% to 10%. The set of frequencies in a passband should be a region of the form $\hat{\omega}_1 \leq \hat{\omega} \leq \hat{\omega}_2$.

- For a lowpass filter, the passband region extends from $\hat{\omega} = 0$ to $\hat{\omega}_p$, where the parameter $\hat{\omega}_p$ is called the *passband edge*. For the two LPFs designed in Section 2.4 determine an *accurate estimate* of $\hat{\omega}_p$ assuming a passband ripple (δ_p) of 0.1 for the Rectangular window case, and $\delta_p = 0.01$ for the Hamming window case.[2] Compare these *actual passband edges* to the design parameter $\hat{\omega}_c$ which is called the *cutoff frequency*.
  Note: There is often confusion that $\hat{\omega}_c$ and $\hat{\omega}_p$ are the same, but after doing a few examples it should become clear that is not the case.

2.4.2 Stopband Defined for the Frequency Response

When the frequency response (magnitude) of the digital filter is close to zero, we have the ***stopband*** region of the filter. In the lowpass filter example of Fig. 2, the magnitude is close to zero when the frequency $1.125 \leq \hat{\omega} \leq \pi$, i.e., high frequencies. When the frequency response of a LPF is plotted only for nonnegative frequencies, the stopband will be a region of the form $\hat{\omega}_s \leq \hat{\omega} \leq \pi$. The parameter $\hat{\omega}_s$ is called the *stopband edge*. We can make a precise measurement of the *stopband edge* as follows:

[2]The `filterdesign` GUI has a zoom capability (`Options->zoom`), and the grid can be turned on (`Options->grid`). Also, when the pointer is placed to hover over the frequency response the coordinates are read from the plot.

- For the lowpass filters from Section 2.4, zoom in on the plot of frequency response magnitude in the `filterdesign` GUI to measure the stopband edge, or use a dB magnitude plot. Then determine the set of frequencies where the magnitude is nearly zero, as defined by $|H(e^{j\hat{\omega}})|$ being less than $\delta_s = 0.1$ for the Rectangular window case, and less than $\delta_s = 0.01$ for the Hamming window design.

- Compare the values of $\hat{\omega}_s$ found in the previous part to the design parameter $\hat{\omega}_c$ (the cutoff frequency).

2.4.3 Transition Zone of the LPF

The difference between the stopband edge and the passband edge is called the *transition width* of the filter: $\Delta\hat{\omega} = \hat{\omega}_s - \hat{\omega}_p$. The smaller the transition width, the better the filter because it is closer to the *ideal filter* which has a transition width of zero.

- For the two lowpass filters from Section 2.4, determine the transition width.
  Note: Comment on the statement, "when comparing equal-order FIR filters, the one with smaller transition width will have larger ripples."

- Design two new LPFs that have the same cutoff frequency, $\hat{\omega}_c = 0.32\pi$, but twice the order, i.e., $M = 44$. Repeat the measurement of $\hat{\omega}_p$, $\hat{\omega}_s$ and $\Delta\hat{\omega}$ for these two LPFs.

- Compare to the values of $\Delta\hat{\omega}$ from part (a). When the order doubles, describe what happens to the transition width.

2.4.4 Summary of Filter Specifications

The foregoing discussion of ripples, bandedges, and transition width can be summarized with the tolerance scheme shown in Fig. 3. An acceptable filter design would be an FIR filter whose magnitude response lies entirely within the template shown in red dash lines. The length-23 FIR filter shown in Fig. 3 meets the specs, but if you designed a length-19 filter it would have a transition width that is greater than $\Delta\hat{\omega} = 0.08\pi$.

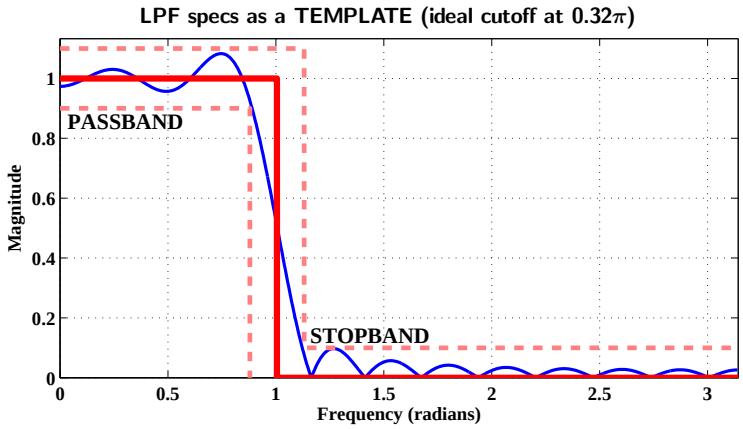

Figure 3: Tolerance scheme drawn around an ideal LPF with a cutoff frequency of $\hat{\omega}_c = 0.32\pi$. Dashed lines indicate the maximum allowable deviation from the ideal LPF. The template uses $\hat{\omega}_p = 0.28\pi$, $\hat{\omega}_s = 0.36\pi$, and $\delta_p = \delta_s = 0.1$. The actual FIR filter shown is the length-23 FIR filter from Fig. 2 which just barely meets these specs.

3 In-Lab Exercise

The objective of the lab exercise is to design FIR filters that can be lowpass, highpass or bandpass filters. The exercises will involve using the *SP-First* GUI `filterdesign` in which FIR filters are designed via the window method, or with a computer optimization technique.

3.1 Ideal Filters and Practical Filters

In `dltidemo`, it is possible to choose from two classes of filters: ideal filters and practical filters.

3.1.1 Ideal Filters

- Define the input signal to be $x[n] = 1.8\cos(0.46\pi n)$.
- LPF: Set the filter to be an ideal LPF with a cutoff frequency of $\hat{\omega}_c = 2\pi(0.29)$. Determine a value for the phase slope so that the output will be delayed by 3, i.e., $y[n] = 1.8\cos(0.46\pi(n-3))$. In addition, get the formula for the output signal from the *Theoretical Answer.* Explain why the phase of the *theoretical* output signal is not equal to -1.38π. Consider multiples of 2π in the phase which must be taken into account when relating the phase to the delay.
- HPF: Change to an ideal HPF; use the same phase slope as in the previous part. Determine the minimum value of the cutoff frequency so that the output signal will be zero.

3.1.2 Practical Filters

Since ideal filters cannot be implemented with numerical computation, it is necessary to study how much degradation there will be when actual *implementable* filters are used. In this section, the filters are order-14 FIR filters with 15 filter coefficients. For both tests, use the same input signal as before: $x[n] = 1.8\cos(0.46\pi n)$. When you observe the output signal, think about the following question: Do you expect the signal to be in the stopband or passband of the filter, i.e., do you expect the output to be zero or to be equal to the input?

- LPF: Set the filter type to a length-15 LPF, with its cutoff frequency at $\hat{\omega}_c = 2\pi(0.29)$. The cutoff frequency determines the boundary between the stopband and the passband. Use the GUI to determine the output signal *passed* by the LPF. Comment on how close this filter is to the ideal.
- In the lowpass case, the output can be written as $y[n] = A_{\text{out}}\cos(0.46\pi(n-n_d))$, where n_d represents a delay. Use the time-domain plots, or the phase slope and phase value to determine n_d which must be an integer. Comment on the degradation of the output amplitude from the ideal.
- HPF: Set the filter type to a length-15 HPF, with its cutoff frequency at $\hat{\omega}_c = 2\pi(0.29)$. The cutoff frequency determines the boundary between the stopband and the passband. Use the GUI to determine how well the output signal *rejected* by the HPF versus an ideal filter, i.e., determine the amplitude of the output signal which should be close to zero. Comment on the degradation of the output amplitude from the ideal.

3.2 Designing Two Lowpass Filters

Design two lowpass FIR filters with $M = 30$ and $\hat{\omega}_c = 0.5\pi = 2\pi(f_c/f_s) = 2\pi(2500/10000)$, one using a Hamming window, the other with a Rectangular window. For the measurement of passband and stopband edges, there are two approaches: use the `filterdesign` GUI and read numbers from the plot,

zooming when necessary, or export the filter coefficients from the GUI and use MATLAB to make plots of the magnitude of the frequency response using `freekz` (or `freqz`) and `plot`. In MATLAB zooming would be more precise and reliable because the frequency sampling can be specified in the call to `freekz`.

- For the filter obtained with the rectangular window, determine an *accurate measurement* of the passband edge ($\hat{\omega}_p$) assuming the passband ripple specification is $\delta_p = 0.1$, i.e., 1 ± 0.1.
- For the filter obtained with the rectangular window, determine an *accurate measurement* of the stopband edge ($\hat{\omega}_s$) assuming the stopband ripple specification is $\delta_s = 0.1$.
- For the filter obtained with the Hamming window, determine an *accurate measurement* of the passband edge ($\hat{\omega}_p$) assuming the passband ripple specification is $\delta_p = 0.01$, i.e., 1 ± 0.01.
- For the filter obtained with the Hamming window, determine an *accurate measurement* of the stopband edge ($\hat{\omega}_s$) assuming the stopband ripple specification is $\delta_s = 0.01$.
- *Question:* is the cutoff frequency half way between ($\hat{\omega}_p$) and ($\hat{\omega}_s$) for both filters?

3.3 Transition Zone of the LPF

The difference between the stopband edge and the passband edge is called the *transition width* of the filter: $\Delta\hat{\omega} = \hat{\omega}_s - \hat{\omega}_p$. The smaller the transition width, the better the filter because it is closer to the *ideal filter* which has a transition width of zero.

- For the two lowpass filters from Section 3.2, determine the transition width.
- *Comment:* "when comparing two M^{th} order filters, the one with a smaller transition width will have larger ripples."
- Design a new Hamming-window LPF that has the same cutoff frequency, $\hat{\omega}_c = 0.5\pi$, but twice the order, i.e., $M = 60$. Repeat the measurement of $\hat{\omega}_p$, $\hat{\omega}_s$ and $\Delta\hat{\omega}$ for this LPF.
- Compare the values of $\Delta\hat{\omega}$ from parts (a) and (c); when the order doubles, describe what happens to the transition width. Use this observation to explain that the following Hamming window design formula should be true
$$\Delta\hat{\omega} = \frac{C}{L}$$
and find the value of the constant C.

3.4 Designing FIR Filter to Meet Given Specifications

Filter design for lowpass filters involves five parameters: two band edges, ripple heights in two bands, and the filter order. There is a sixth factor, which is the type of filter such as a Hamming windowed FIR filter. A typical design problem would be stated as follows: given the band edges and ripple heights, determine the *minimum order* filter that will meet the specs.

- Suppose that you are given $\hat{\omega}_p = 0.60\pi$, $\hat{\omega}_s = 0.64\pi$, $\delta_p = 0.01$, and $\delta_s = 0.01$. Make a sketch of an ideal filter and a template that looks like Fig. 3. Label everything carefully and completely.
- Use your Hamming window design formula (from the previous section) to predict the filter length (L) that will be needed to meet the specs. Recall that $L = M + 1$.
- Design the Hamming-windowed FIR filter with the predicted order. Determine the correct value to use for the cutoff frequency. Explain why the resulting frequency response does or does not meet the given specs.

3.5 Filter Design via Optimization

Many different methods have been developed for filter design via mathematical optimization. One of the widely used methods is `firpm` in MATLAB. For designing a LPF, it uses the following two step process:

- Use the desired specifications for $\hat{\omega}_p$, $\hat{\omega}_s$, δ_p, and δ_s to estimate the filter order (M) that will be needed. This is done with the MATLAB function `firpmord`.
- Use the outputs from `firpmord` as inputs to the function `firpm` to run the optimization and obtain the FIR filter coefficients that should meet the specs on δ_p and δ_s. In effect, the inputs to `firpm` are $\hat{\omega}_p$, $\hat{\omega}_s$, M, and the ratio δ_p/δ_s.

For the calling arguments of these functions, do `help firpmord` and `help firpm`.

- Suppose that you are given $\hat{\omega}_p = 0.60\pi$, $\hat{\omega}_s = 0.64\pi$, $\delta_p = 0.05$, and $\delta_s = 0.01$. Notice that the specs on δ_p and δ_s can be different, unlike the window method that always has $\delta_p = \delta_s$.
  Carry out the two design steps above to get the filter order M and the filter coefficients $\{b_k\}$.
- Make a plot of the impulse response of the filter. Recall that the filter coefficients of the FIR filter are the values of the impulse response. Also, the DTFT of the impulse response is the frequency response of the FIR filter.
- Make a plot of the frequency response magnitude versus $\hat{\omega}$. Then check whether or not the ripple specs (δ_p, δ_s) have been met. The band edges should definitely be correct because $\hat{\omega}_p$ and $\hat{\omega}_s$ are inputs to `firpm`.
- If the ripple specs are not met with the predicted order, then increase the order by one and try again. A higher order such as $M + 1$ or $M + 2$ should meet the specs.
- The phase of this FIR filter will be linear phase. Determine the slope of the linear phase.
- After you have designed your bandpass filter, generate a signal by adding two sinusoids, one in the stopband and one in the passband. Run the signal through the bandpass filter to verify that the filter works properly. Plot spectrograms of the input and output signals. There should be only one sinusoid with the passband frequency present after filtering. Explain why this is the case.

3.5.1 Linear Phase in the Frequency Response

The phase of the frequency response can be related to time delay. In the lowpass filter example of Section **??**, the phase plot appears to be jagged, but it is actually (piecewise) linear.

- Determine the slope of the linear segments of the frequency response for the $M = 50$ filter in Section **??**. Your answer should be an integer.
- Plot the impulse response of the digital filter in Section **??** for the range $n = 0, 1, \ldots, M$. Then determine the symmetry point for $h[n]$, i.e., find the integer n_s such that $h[n_s + n] = h[n_s - n]$.
- Compare the value of the slope (from part (a)) to the "symmetry point" of $h[n]$. Verify that the phase slope is equal to $-n_s$.

3.6 Bandpass Filter Designed from Lowpass Filters

Here's an idea: subtract two LPFs to get a BPF. If you think in terms of the magnitude response only, then it's relatively easy to see that this could be true.

- Use your Hamming-sinc M-file to design two 50-th order FIR filters, $h_1[n]$ and $h_2[n]$, one with a cutoff frequency at $\hat{\omega}_c = 0.6\pi$, the other at $\hat{\omega}_c = 0.3\pi$. Subtract the impulse responses ($h_1[n] - h_2[n]$) and make a stem plot of the resulting impulse response (call it $h_3[n]$).

- Use `freqz` (or `freekz`) to calculate the frequency response from $h_3[n]$. Then plot the magnitude versus $\hat{\omega}$ over the range $0 \leq \hat{\omega} \leq \pi$. Annotate the plot to show where the cutoff frequencies (0.3π and 0.6π) lie.

- Use the `filterdesign` GUI to create a BPF with the same two cutoff frequencies (0.3π and 0.6π). Export the filter coefficients and make a stem plot to verify that you get exactly the same result. Compare the frequency response in the `filterdesign` GUI to what you plotted in part (b).

- Finally, determine the upper and lower stopband edges and passband edges of the BPF, as well as the upper and lower transition widths. There are two transition zones in a BPF. Compare these measurements to those obtained earlier for the LPF case.

- Export the filter coefficients in order to use them with the `firfilt` function and the `freqz` function in part (d). Note that the GUI will use the default names `num` and `den`.

- Use the GUI's plot of the frequency response to verify the correct bandedges and ripples in the passband and stopbands.