The goal of this lab is to study the sinusoidal response of some simple FIR filters in MATLAB. This leads to a study of the frequency response function and design of an FIR filter.

1. In the experiments of this lab, you will use the MATLAB GUI called `dltidemo` to the frequency response function for FIR filters. If you have installed the *SP-First* Toolbox, you will already have this demo on the `matlabpath`.

2. You will also study the ***Sinusoid-In Gives Sinusoid-Out*** property of LTI systems.

3. You will use the `filterdesign` GUI to design a FIR filter given a set of filter specifications.

The goal of this lab is to study the frequency response and design a FIR filter to meet a set of specifications. The frequency response for FIR filters is the response to inputs such as complex exponentials and sinusoids. Although you can use `firfilt()`, or `conv()`, to implement filters in the *time domain*, the function `freqz()` is used to characterize the filter in the *frequency domain* via its frequency response.[1] As a result, you will learn how to obtain the output when the input is a sum of sinusoids because the frequency response characterizes a filter by telling how the filter responds to different frequency components in the input.

Frequency Response of FIR Filters

The output or *response* of a filter for a complex sinusoid input, $e^{j\hat{\omega}n}$, is a complex exponential at the same frequency. The magnitude and phase of the output will be different, and that change depends on the frequency, $\hat{\omega}$. The dependence of these magnitude and phase changes versus frequency is called the *frequency*

[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*.

response. In effect, the filter is described by how it affects different input frequencies. For example, the frequency response of the two-point averaging filter

$$y[n] = \tfrac{1}{2}x[n] + \tfrac{1}{2}x[n-1]$$

can be found by using a general complex exponential as an input $x[n]$ and observing the output or *response.*

$$\begin{aligned} x[n] &= Ae^{j(\hat{\omega}n+\varphi)} && (1)\\ y[n] &= \tfrac{1}{2}Ae^{j(\hat{\omega}n+\varphi)} + \tfrac{1}{2}Ae^{j(\hat{\omega}(n-1)+\varphi)} && (2)\\ &= Ae^{j(\hat{\omega}n+\varphi)}\left\{\tfrac{1}{2}+\tfrac{1}{2}e^{-j\hat{\omega}}\right\} && (3)\end{aligned}$$

In (3) there are two terms, the original input, and a term that is a function of $\hat{\omega}$. This second term is the frequency response and it is commonly denoted[2] by $H(e^{j\hat{\omega}})$.

$$H(e^{j\hat{\omega}}) = H(\hat{\omega}) = \tfrac{1}{2}\left\{1+e^{-j\hat{\omega}}\right\} \quad (4)$$

The general form of the frequency response for an M-th order FIR linear time-invariant system with filter coefficients $\{b_k\}$ is

$$H(e^{j\hat{\omega}}) = H(\hat{\omega}) = \sum_{k=0}^{M} b_k e^{-j\hat{\omega}k} \quad (5)$$

Once the frequency response, $H(e^{j\hat{\omega}})$, has been determined, the effect of the filter on any complex exponential input may be determined by evaluating $H(e^{j\hat{\omega}})$ at the corresponding input frequency. The output signal, $y[n]$, will be a complex exponential whose complex amplitude has a constant magnitude and phase. The phase of $H(e^{j\hat{\omega}})$ describes the phase change of the complex sinusoid and the magnitude of $H(e^{j\hat{\omega}})$ describes the "gain" applied to the complex sinusoid.

Pre-Lab

The goal of this lab is to learn about frequency response of FIR filters in MATLAB, and then study the response of FIR filters to various signals. You will also design a FIR filter with certain constraints applied to the frequency response of the filter.

MATLAB Function for Frequency Response

MATLAB has a built-in function for computing the frequency response of a discrete-time LTI system. The following MATLAB statements show how to use `freqz` to compute and plot both the magnitude (absolute value) and the phase of the frequency response of a two-point averaging system as a function of $\hat{\omega}$ in the range $-\pi \le \hat{\omega} \le \pi$:

```
1      bb = [0.5, 0.5];          %-- Filter Coefficients
2      ww = -pi:(pi/100):pi;     %-- omega hat frequency vector
3      H = freqz(bb, 1, ww);     %<--freekz(bb,1,ww) is an alternative
4      subplot(2,1,1);
5      plot(ww, abs(H)), grid on
6      subplot(2,1,2);
7      plot(ww, angle(H)), grid on
8      xlabel('Normalized Radian Frequency')
```

[2]The notation $H(e^{j\hat{\omega}})$ is used in place of $H(\hat{\omega})$ for the frequency response because we will eventually connect this notation with the z-Transform, $H(z)$, in Chapter 9.

For FIR filters, the second argument of `freqz( _, 1, _ )` must always be equal to `1`. The frequency vector `ww` should cover an interval of length 2π for $\hat{\omega}$, and its spacing must be fine enough to give a smooth curve for $H(e^{j\hat{\omega}})$. Note: we will always use capital `H` for the frequency response.[3]

Periodicity of the Frequency Response

The frequency responses of discrete-time filters are *always* periodic with period equal to 2π. Explain why this is the case by using the definition of the frequency response (5) and then considering two input sinusoids whose frequencies are $\hat{\omega}$ and $\hat{\omega} + 2\pi$.

$$x_1[n] = e^{j\hat{\omega}n} \qquad \text{versus} \qquad x_2[n] = e^{j(\hat{\omega}+2\pi)n}$$

It should be easy to prove that $x_2[n] = x_1[n]$. Consult Chapter 6 for a mathematical proof that the outputs from each of these signals will be identical. **The implication of periodicity is that a plot of $H(e^{j\hat{\omega}})$ only has to be made over the interval $-\pi \leq \hat{\omega} \leq \pi$.**

Frequency Response of the Four-Point Averager

In Chapter 6 we examined filters that compute the average of input samples over an interval. These filters are called "running average" filters or "averagers" and they have the following form for the L-point averager:

$$y[n] = \frac{1}{L}\sum_{k=0}^{L-1} x[n-k] \tag{6}$$

(a) Use Euler's formula and complex number manipulations to show that the frequency response for the 4-point running average operator can be written as:

$$H(e^{j\hat{\omega}}) = H(\hat{\omega}) = \left(\frac{2\cos(0.5\hat{\omega}) + 2\cos(1.5\hat{\omega})}{4}\right) e^{-j1.5\hat{\omega}} = C(\hat{\omega})e^{j\psi(\hat{\omega})} \tag{7}$$

(b) Implement (7) directly in MATLAB. Use a vector that includes 400 samples covering the interval $[-\pi, \pi)$ for $\hat{\omega}$. Make plots of $C(\hat{\omega})$ and $\psi(\hat{\omega})$ versus $\hat{\omega}$. It is tempting to think that $C(\hat{\omega})$ is the magnitude of the frequency response, but $C(\hat{\omega})$ can go negative, so these are not plots of the magnitude and phase. You would have to use `abs()` and `angle()` to extract the magnitude $|H(e^{j\hat{\omega}})|$ and phase $\angle H(e^{j\hat{\omega}})$ of the frequency response for plotting.

(c) In this part, use `freqz.m` or `freekz.m` in MATLAB to compute $H(e^{j\hat{\omega}})$ numerically (from the filter coefficients) and plot its magnitude and phase versus $\hat{\omega}$. Write the appropriate MATLAB code to plot both the magnitude and phase of $H(e^{j\hat{\omega}})$. Follow the example in Section 2.1. The filter coefficient vector for the 4-point averager is defined via:

```
bb = 1/4*ones(1,4);
```

Recall that the function `freqz(bb,1,ww)` evaluates the frequency response for all frequencies in the vector `ww`. It uses the summation in (5), not the formula in (7). The filter coefficients are defined in the assignment to vector `bb`. How do your results compare with part (b)?
Note: the plots should not be identical, but you should be able to explain why they are equivalent by converting the minus sign in the negative values of $C(\hat{\omega})$ to a phase of π radians, which then modifies the phase plot with jumps of $\pm\pi$ radians.

[3] If the output of the `freqz` function is not assigned, then plots are generated automatically; however, the magnitude is given in decibels which is a logarithmic scale. For linear magnitude plots a separate call to `plot` is necessary.

FIR Nulling Filters

A simple second-order FIR filter can be used to remove a sinusoid from an input signal. The general form for the filter coefficients of an FIR nulling filter is

$$b_0 = 1 \qquad b_1 = -2\cos(\hat{\omega}_{\text{NULL}}) \qquad b_2 = 1 \tag{8}$$

Nulling means that the frequency response will be zero at $\hat{\omega} = \hat{\omega}_{\text{NULL}}$. With $\hat{\omega}_{\text{NULL}} = 0.75\pi$, enter the filter coefficients in the `dltidemo` GUI to see the frequency response; verify that it is zero at $\hat{\omega} = 0.75\pi$. In addition, define the input to be $x[n] = 0.4 + 1.5\cos(0.75\pi n)$ and observe that the sinusoidal component is not present in the output.

Ideal Filters and Practical Filters

Ideal Filters are given by a frequency response, consisting of *perfect* passbands and stopbands. These are not actually FIR filters because there is no finite set of filter coefficients that will produce the ideal frequency response. In the `dltidemo` GUI, you can choose ideal lowpass filters (LPF), highpass filters (HPF) and bandpass filters (BPF).
The ideal LPFs and HPFs have one parameter for the cutoff frequency which determines the boundary between the ideal passband and the ideal stopband. The ideal BPF has a parameter for center frequency which determines where the band is located; its bandwidth (in this GUI) is always 0.4π. All the ideal filters have an additional parameter for the slope of the phase of $H(e^{j\hat{\omega}})$.

(a) Use the `dltidemo` to view the *sinusoid-in gives sinusoid-out* behavior of an ideal bandpass filter (BPF). Choose the center frequency to be 0.4π. Then the two bandedge frequencies for the BPF will be $\hat{\omega}_\ell = 0.2\pi$ and $\hat{\omega}_u = 0.6\pi$. These bandedges define the extent of the ideal passband of the BPF.

(b) Set the input to be $x[n] = 1.5 + 0.9\cos(0.55\pi n)$. Determine which frequency components are present in the output signal.

(c) You can also include an ideal linear phase in the frequency response, so choose the phase slope as -2. Then find the output for the same input as in the previous part. Describe how the output has changed; relate the delay in the output to the phase slope of the frequency response.

(d) When the input is $x[n] = 1.5 + 0.9\cos(0.65\pi n)$, determine the output. Explain why it is zero.

Practical Filters which are approximations to the ideal filters by length-L FIR filters. The ones shown in `dltidemo` were designed using MATLAB's `fir1` function for digital filter design. The GUI has length-15 LPFs and HPFs, and length-21 BPFs. The LPF and HPF designs have one parameter for the cutoff frequency which determines the boundary between the non-ideal passband and stopband. The BPF has a parameter for center frequency which determines where the passband is located. The default bandedges are $\pm 0.2\pi$ from the center frequency, but since this filter is not ideal the frequency response at the bandedges has a magnitude of 0.5.
Notation: the cutoff frequency for HPFs and LPFs will be called $\hat{\omega}_c$.

Note: The running average FIR filter is an *approximate* lowpass filter, but it is not a very good one because it is not very close to an ideal LPF. Better filters can be designed with computer-aided optimization algorithms, or with MATLAB's `fir1` function.

(a) Use the `dltidemo` to view the *sinusoid-in gives sinusoid-out* behavior of an ideal bandpass filter (BPF). Choose the center frequency for the BPF to be $\hat{\omega} = 0.4\pi$. Then the desired bandedges of the passband will be $\hat{\omega}_\ell = 0.2\pi$ and $\hat{\omega}_u = 0.6\pi$. Since the filter is not ideal, these two bandedges define an approximate extent of the BPF's passband.

(b) Set the input to be $x[n] = 1.5 + 0.9\cos(0.65\pi n)$. Determine the output signal. Compare to the output obtained in part (d) for Ideal Filters. Explain why is is not zero.

(c) The output signal might contain both frequency components. Relate the amplitudes of the output signal's two frequency components to the non-ideal nature of the frequency response. Right-click to get values from the frequency response plot.

Windowing and Truncate the Ideal Impulse Response

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

$$w_r[n] = \begin{cases} 1 & 0 \le n \le L-1 \\ 0 & \text{elsewhere} \end{cases} \tag{9}$$

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

$$w_r[n]x[n+n_0] = \begin{cases} 0 & n < 0 \\ 1 \times x[n+n_0] & 0 \le n \le L-1 \\ 0 & n \ge L \end{cases} \tag{10}$$

The name window comes from the idea that we can only "see" L values of the signal x[n + n0] within the window interval when we "look" through the window. Multiplying by w[n] is looking through the window. When we change n0, 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

$$w_m[n] = \begin{cases} 0.54 - 0.46\cos(2\pi n/(L-1)) & 0 \le n \le L-1 \\ 0 & \text{elsewhere} \end{cases} \tag{11}$$

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:

$$h_1[n] = w_r[n]h_{IDEAL}[n-11] = \begin{cases} \frac{\sin(\hat{\omega}_c(n-11))}{\pi(n-11)} & 0 \le n \le 22 \\ 0 & \text{elsewhere} \end{cases} \tag{12}$$

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 (12) to be causal. It is common that the ideal impulse response is time shifted by an amount equal to half the window length.

GUI for Filter Design

The SP-First GUI called `filterdesign` illustrates several filter design methods for LPF, BPF and HPF filters. 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.

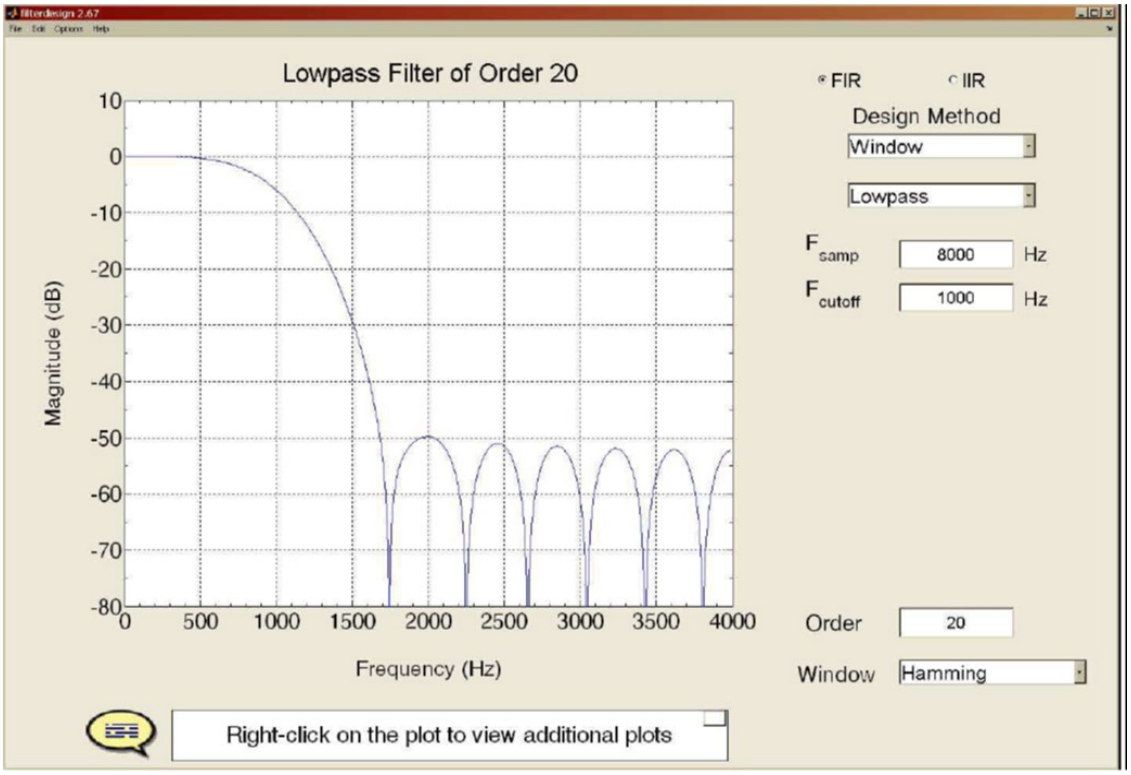

Figure 1: Interface for the `filterdesign` GUI. When the Filter Choice is set to `FIR`, many different window types can be selected, including the Hamming window and the Rectangular window (i.e., no window). The specification of one or more cutoff frequencies (f_{co}) must be entered using continuous-time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

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`. 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`.

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.

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

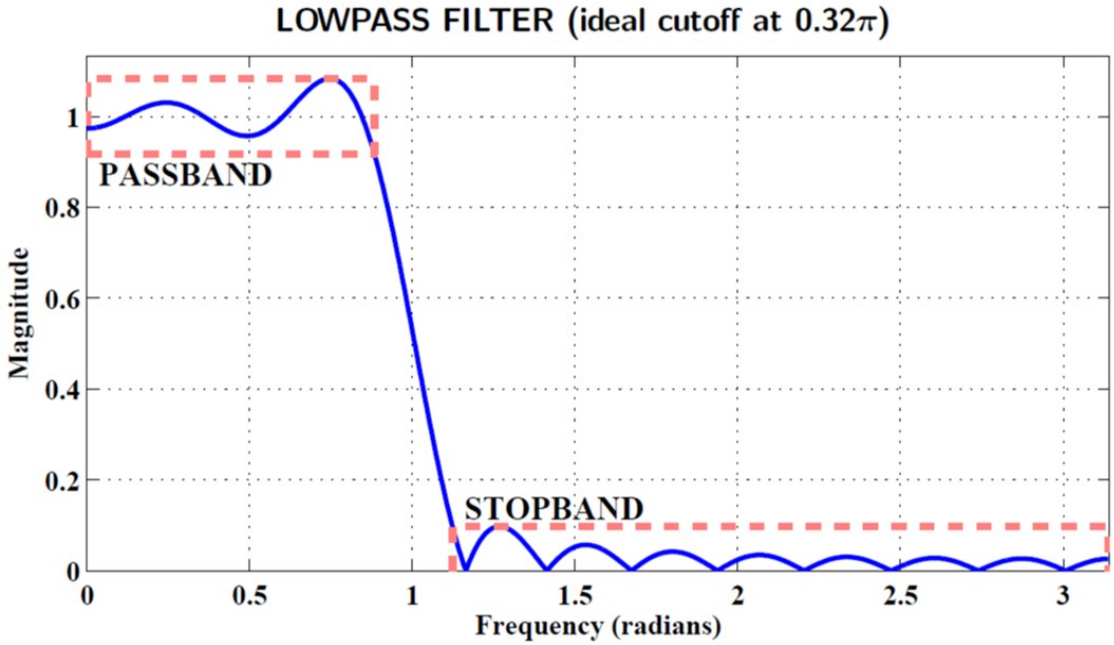

Figure 2: Passband and stopband defined for a typical lowpass filter. This one is of length 23 designed with a rectangular window and a sinc function with a cutoff frequency of $\hat{\omega}_c = 0.32\pi$. The passband and stopband ripples are defined to be 0.1, from which the passband and stopband edges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

lowpass filter whose magnitude is close to one when $0 \leq \hat{\omega} \leq 0.883$. 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.

(a) 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 $|H(e^{j\hat{\omega}})| - 1|$ 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 $\hat{\omega}_1 \leq \hat{\omega} \leq \hat{\omega}_2$.

(b) 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 2.8 determine an *accurate estimate* of ω_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.[4] 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.

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:

[4]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.

(a) For the lowpass filters from Section 2.8, 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.

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

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.

(a) For the two lowpass filters from Section 2.8, 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."

(b) 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.

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

Summary of Filter Specification

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. meets the specs, but if you designed a length-19 filter it would have a transition width that is greater than $\Delta\hat{\omega} = 0.08\pi$.

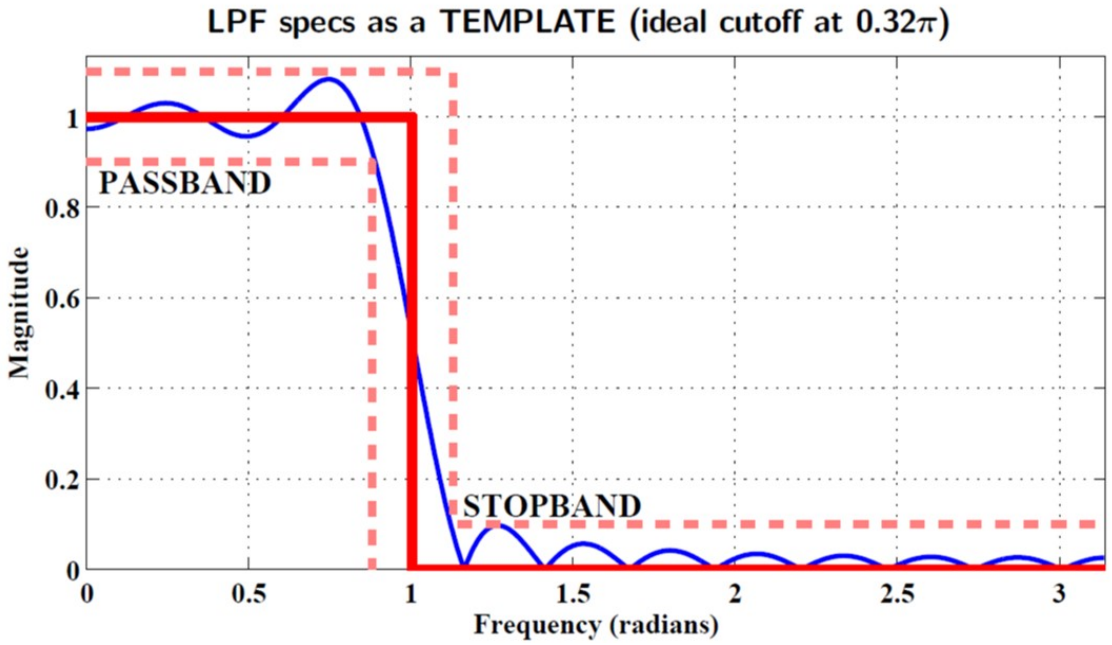

Figure 3: Tolerance scheme drawn around an ideal LPF with a cutoff frequency of $\hat{\omega}_c = 0.32\pi$. Dashed lines indicate the maximum allowable deviation from the ideal LPF. The template uses $\hat{\omega}_p = 0.28\pi$, $\hat{\omega}_s = 0.36pi$, and $\delta_p = \delta_s = 0.1$. The actual FIR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

In-Lab Exercises

The main objective of this lab is to use a MATLAB GUI to demonstrate the frequency response, as well as the *sinusoid-in gives sinusoid-out* property of LTI systems. The frequency response demo, `dltidemo`, is part of the *SP-First Toolbox*.

LTI Frequency Response Demo

The `dltidemo` GUI illustrates the "sinusoid-IN gives sinusoid-OUT" property of LTI systems. In this demo, you can change the amplitude, phase and frequency of an input sinusoid, $x[n]$, and you can change the digital filter that processes the signal. Then the GUI will show the output signal, $y[n]$, which is also a sinusoid (at the same frequency). Figure 4 shows the interface for the `dltidemo` GUI. It is possible to see the formula for the output signal; just click on the `Theoretical Answer` button located at the bottom-middle part of the window. The digital filter can be changed by choosing different `Filter Choice` options in the `Filter Specifications` box in the lower right-hand corner.
Perform the following steps with the `dltidemo` GUI:

(a) Set the input to $x[n] = 1.8\cos(0.2\pi(n-3))$; note that this sinusoid has a positive peak at $n = 3$.

(b) Set the digital filter to be a 7-point averager. Using the middle panels that show the frequency response, measure the magnitude and phase of the frequency response at the input frequency. Right-click to get values from the frequency response plot.

(c) Give an equation that explains how the time delay is related to the phase of $H(e^{j\hat{\omega}})$ at $\hat{\omega} = 0.2\pi$. Remember that phases differing by integer multiples of 2π are the same.

(d) Convert the phase to time-index delay. Then you can determine a formula for the output signal that can be written in the form: $y[n] = A\cos(\hat{\omega}_0(n - n_7))$, where n_7 is an integer. Using n_5 from $y[n]$

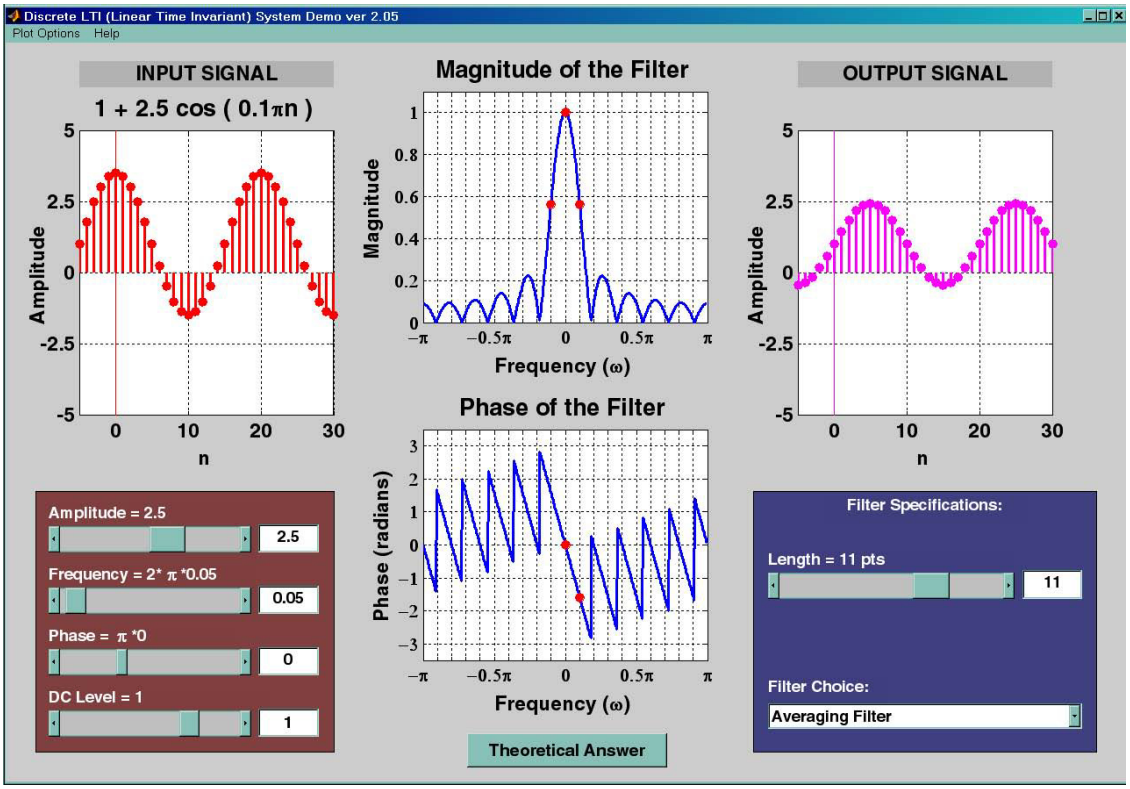

Figure 4: DLTI demo interface. The frequency label should be $\hat{\omega}$, but MATLAB won't display the hat in $\hat{\omega}$.

and the fact that the input signal had a peak at $n = 3$, it should be easy verify how much the peak of the cosine wave has been shifted. This is called the *time delay* through the filter.

Instructor Verification (separate page)

(e) Now, change the frequency of the input signal so that the output will be exactly zero. Sinusoidal components at other frequencies $\hat{\omega}$ will also be nulled—make a list of these nulled frequencies in the range $0 \leq \hat{\omega} \leq \pi$. There are many choices for this frequency; list them all.
Hint: Recall the Dirichlet form for the frequency response of the averaging filter, and where it has regularly spaced zeros versus $\hat{\omega}$.

(f) When the output of an FIR filter is zero, the FIR filter acts as a *Nulling Filter* for a certain input frequency. Design a second-order nulling filter that will remove the cosine component from the following signal: $x[n] = 1 + \cos(0.4\pi n)$. List the three filter coefficients of the nulling filter. Also, the DC component will not be removed, so determine the DC level of the output signal.

Instructor Verification (separate page)

Ideal Filters and Practical Filters

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

Ideal Filters

(a) Define the input signal to be $x[n] = 1.8\cos(0.46\pi n)$.

(b) 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.

(c) 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.

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?

(a) 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.

(b) 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.

(c) 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.

Designing a Lowpass Filter

Design a lowpass FIR filter with M = 30 and $\hat{\omega_c}$ = 0.5π = 2π(f_c/f_s) = 2π(2500/10000), using a Hamming 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`.

(a) 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 δ_p = 0.01, i.e., 1 $\pm$ 0.01.

(b) 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 δ_s = 0.01.

(c) *Question:* is the cutoff frequency half way between ($\hat{\omega_p}$) and ($\hat{\omega_s}$) for the filter?

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.

(a) For the lowpass filter from Section 3.3, determine the transition width.

(b) Design a new Hamming-window LPF that has the same cutoff frequency, $\hat{\omega_c}$ = 0.5π, 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.

(c) Compare the values of $\Delta\hat{\omega}$ from parts (a) and (b); 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

$$\Delta\hat{\omega} = \frac{C}{L} \tag{13}$$

and find the value of the constant C.

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.

(a) Suppose that you are given $\hat{\omega_p}$ = 0.60π, ω_s = 0.64π, δ_p = 0.01, and δ_s = 0.01. Make a sketch of an ideal filter and a template that looks like Fig. 3. Label everything carefully and completely.

(b) Use your Hamming window design formula (from the previous section) to predict the predict the filter length (L) that will be needed to meet the specs. Recall that L = M + 1.

(c) Design the Hamming-windowed FIR filter with the predicted order. Determine the correct value to use for the cutoff frequency. Show the frequency response to your lab instructor for verification. Explain why the resulting frequency response does or does not meet the given specs.

Lab #7
ECE-2026 Summer-2022
VERIFICATION LIST

The lab instructors will ask all or some of the following questions during lab sessions. Demo them using MATLAB as required.

Name: _____ gtLoginUserName: _____ Date: _____

Part	Observations
3.1(b)	Magnitude and phase of the frequency response of the length-7 averager, at the input frequency.
3.1(c)	Give an equation that explains how the filter's *delay* is related to the phase of $H(e^{j\hat{\omega}})$ at $\hat{\omega} = 0.2\pi$.
3.1(d)	Formula for output from a length-7 averager written in the form $y[n] = A\cos(\hat{\omega}_0(n - n_5))$

3.1(e)	List of all frequencies nulled by the running-average FIR filter in part (b), or give a formula.
3.1(e)	Length-3 nulling filter, list three filter coefficients. Give the DC level of the output signal.

3.2.1(b)	Phase slope for Ideal HPF. Explain why phase of $y[n]$ formula is not equal to -1.38π?
3.2.1(c)	Cutoff frequency for Ideal HPF to get $y[n] = 0$.

3.2.2(a)	Output of length-15 LPF with $\hat{\omega}_c = 2\pi(0.29)$ which can be expressed as $A_{\text{out}}\cos(\hat{\omega}_0 n + \varphi)$
3.2.2(b)	LPF output: Determine the delay n_d so that the output can be expressed with the formula, $A_{\text{out}}\cos(\hat{\omega}_0(n - n_d))$
3.2.2(c)	Output of length-15 HPF with $\hat{\omega}_c = 2\pi(0.29)$. Give formula and explain why $y[n]$ is not exactly zero.

3.3(a,b)	Passband and stopband edges of FIR filter designed with Hamming window. Measured values:
3.3(c)	Is the cutoff frequency half way between ($\hat{\omega_p}$) and ($\hat{\omega_s}$) for the filter?

3.4(a)	Transition width of FIR filter designed with Hamming window.
3.4(b)	Measure values of passband and stopband edges, and transition width.
3.4(c)	Find C in $\Delta\hat{\omega}$ = C/L for dependence of transition width on filter order.

3.5(a)	Make a sketch of ideal filter including a tolerance template like Fig. 3. Draw the sketch on the axes below.
3.5(b)	Predict length of the FIR filter to be designed with the Hamming window, using $\Delta\hat{\omega}$ = C/L. L = ?
3.5(c)	Give value of $\hat{\omega_{co}}$, and show frequency response of filter. Zoom in on passband stopband regions to verify.