University of Kentucky

EE 422G - Signals and Systems Laboratory

Lab 2 FIR Filters

Objectives:

- Use filter design and analysis tools to create FIR filters based on general filter specification.
- Create a simulation with Simulink.

1. Background

Digital filters are used in a wide variety of signal processing applications, such as spectrum analysis, digital image processing, and pattern recognition. Digital filters eliminate a number of problems associated with their classical analog counterparts and thus are often used in place of analog filters. Digital filters belong to the class of discrete-time LTI (linear time invariant) systems, which are characterized by the properties of causality, recursibility, and stability. They can be characterized in the time domain by their unit-impulse response and in the transform domain by their transfer function. A unit-impulse response sequence of a causal LTI system could be of either finite or infinite duration and this property determines their classification into either finite impulse response (FIR) or infinite impulse response (IIR) system. To illustrate this, we consider the most general case of a discrete time LTI system with the input sequence denoted by $x(kT)$ and the resulting output sequence $y(kT)$ given by:

$$y(kT) = \sum_{\mu=0}^{m} b_\mu x(kT-\mu T) - \sum_{\nu=1}^{n} a_\nu y(kT-\nu T) \;\Leftrightarrow\; H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{\mu=0}^{m} b_\mu z^{-\mu}}{1+\sum_{\nu=1}^{n} a_\nu z^{-\nu}} \qquad (1)$$

If at least one denominator coefficient a_ν is not zero, then system is recursive; its impulse response is of infinite duration (IIR system). If all denominator coefficients are zero, the corresponding system is non-recursive (FIR system); its impulse response is of finite duration and the transfer function $H(z)$ is a simple polynomial. Commonly, b_μ is called the μ^{th} forward filter coefficient and a_ν the ν^{th} feedback or reverse filter coefficient.

The design of digital filters involves the following basic steps:

- Determine the desired response. The desired response can be specified in the frequency domain in terms of the desired magnitude response and/or the desired phase response. In can also be specified in term of a desired impulse response.
- Select a class of filters (for example, linear-phase FIR filters or IIR filters) to approximate the desired response.

- Select the best member in the filter class (i.e. determination of order and coefficient values).
- Implement the best filter using a general purpose computer, a DSP, or a custom hardware chip.
- Analyze the filter performance to determine whether the filter satisfies all the given criteria.

In this lab you will design FIR filter using 2 methods – windowing and the Parks-McClellan algorithm.

2. Pre-Lab

1. For the window functions listed below, write a script to plot each window function on the same graph (use different line styles for each window function). Create another graph and plot its DFT magnitude on a linear scale, and finally create another graph and plot its DFT magnitude on a dB scale.
   a. boxcar

   b. triangular

   c. hamming

   Comment on how the general window shape (steepness of taper) affects the spectral magnitude (impact on width of main lode and height of sidelobes)

2. Repeat number 1 for a Kaiser window of length 128 and with values of 4, 6, and 9.
3. For a sampling rate of 48kHz design an order-40 low-pass filter having cut-off frequency 10kHz by windowing method. In your design, use Hamming window as the windowing function (see help on *fir1*). Use the *freqz* command to plot the filter's magnitude response (in dB) and use the *filter* command to plot the impulse response.
4. For a sampling rate of 48kHz, design an order-40 low-pass filter having transition band between 9kHz and 11kHz using the Parks-McClellan FIR filter design algorithm (see help on *firpm*). Use the *freqz* command to plot the filter's magnitude response (in dB) and use the *filter* command to plot the impulse response.

3. In-Lab exercise

(Assume sampling frequency of 44.1kHz, unless otherwise specified)

1. Design a 127th order linear-phase FIR low-pass filter with a cut-off at 6kHz using the windowing method (*fir1*). Design a filter using each of the following windows:

    a. rectangular (`boxcar`)
    b. triangular (`triang`)
    c. Hamming (`hamming`)

    On a single graph plot the all the impulse responses, on another graph plot all the magnitude responses, and then in individual graphs plot the zero-pole locations of the 3 filters. In the discussion section compare impact of the window on the filter characteristics. Refer to the window plots of the prelab and make a general statement about window characteristics and their impact on the filter characteristics.

2. Repeat Exercise 1 for a 15th order filter. Comment on the impact of filter order.

3. Repeat Exercise 1 for a length-32 linear-phase FIR low-pass filter using a Kaiser window with β = 4, 6, and 9 and pass-band cut off of 6kHz. Plot the impulse response, magnitude response, and zero-pole locations. Compare the characteristics of the magnitude response with each other and with the filters from the first exercise. Discuss how the trade-off between transition bandwidth and ripple varies with β ?

4. Design an optimal 23-point low-pass FIR filter using the Parks-McClellan algorithm (*firpm*). Use a pass-band cutoff of 5.5kHz and a stop-band cutoff of 6.5kHz. Plot the impulse response, magnitude response, and zero-pole locations. Compare the characteristics of the magnitude response to the other designs and comment on differences.

5. Generate a Square wave signal of 400 Hz and determine an approximate bandwidth for this signal (frequency range containing significant energy) by plotting its spectrum by using the FFT command. Use the Parks-McClellan algorithm (*firpm*) to design a 51-order FIR low-pass filter, such that the frequencies starting with the 3rd (i.e. at 1200Hz) harmonic are suppressed by 40 dB. Plot the magnitude response of the filter and also the filtered output. Did it achieve your specification? Use *soundsc* to play the sound before and after filtering. Describe the differences that you hear.

6. Repeat Exercise 5 with an order of 201.

7. Generate the signal x using the following Matlab code:

```
Fs=8e3;  %Specify Sampling Frequency
Ts=1/Fs; %Sampling period.
Ns=512;  %Nr of time samples to be plotted.
```