ENE 610/Spring 07/Project II: FFT and Filter Design

Samir Chettri

March 25, 2007

Abstract

In class I have repeatedly mentioned the FFT (Fast Fourier Transform) which is a fast way of computing the DFT (Discrete Fourier Transform). In this project we will use the FFT to design equiripple filter designs. NOTE: Though I am assigning a project involving the FFT prior to your being taught this important algorithm, rest assured, there is something very important to learn in doing this project which will become evident when you perform the calculations.

Students will write a function in MATLAB to enable them to create different filters.

Keywords: FFT, iterative equiripple filter design, Parks McLellan algorithm.

Contents

1 The FFT

The FFT has been named one of the top ten algorithms of the 20^{th} century - amath.colorado.edu/resources/archive/topten.pdf . MATLAB has a function called fft to compute the DFT (to be defined in class). Students should familiarize themselves with this function by typing help fft within MATLAB. Also do the same for ifft.

2 FIR Filters and specifications

We have studied convolution in class. This given by

$$y[n] = \sum_{k \in I} h[k]x[n-k]. \tag{1}$$

In equation 1, $x[n], y[n]$ are the input and output sequences respectively and $h[n]$ is the impulse response of the filter which has support of I. In this project we assume I is finite (hence the filter is FIR) and is symmetric, i.e., $I = -N, -N+1, \ldots, -1, 0, 1, \ldots, N$. As defined, $h[n]$ is not causal, but we can get a casual filter by delaying $h[n]$ by N samples.

In order to design an FIR filter we have to calculate $h[n]$ such that any constraints placed on the filter are satisfied. For example we could say "ideal low–pass filter," with pass–band up to 0.2π and our

constraints are immediately evident. Typically, we place constraints in the frequency domain. The frequency response of an FIR filter with support I is

$$H(e^{j\omega}) = \sum_{n \in I} h[n] e^{-j\omega n}. \tag{2}$$

Because $H(e^{j\omega})$ is 2π periodic we can use a region of $[-\pi \leq \omega \leq \pi]$ though we are not constrained to do so.

Figure 1 shows the specifications of a (non–ideal) LP filter. The passband region is defined as $F_p = [-\omega_p \ \omega_p]$ and the filter magnitude is nominally expected to take on unity value here. The stopband is $F_s = [-\pi \ \ -\omega_s] \cup [\omega_s \ \pi]$ where $\omega_s > \omega_p$. Because brick–wall characteristics are impossible to obtain in practice, there exists a transition region F_t between F_p and F_s. We permit rippling in the pass and stopbands and these are defined by $1 - \delta_p \leq |H(\omega)| \leq 1 + \delta_p$ and $|H(\omega)| \leq \delta_s$ respectively. δ_p and δ_s are small tolerance parameters typically much less than unity. Hence a LP FIR filter is completely defined in the frequency domain by F_p, F_s and δ_p, δ_s, given which we intend to find $h[n]$ where the order (length) of the filter is $2N + 1$. Note that the filter order is not known a priori and is generally obtained through an iterative process.

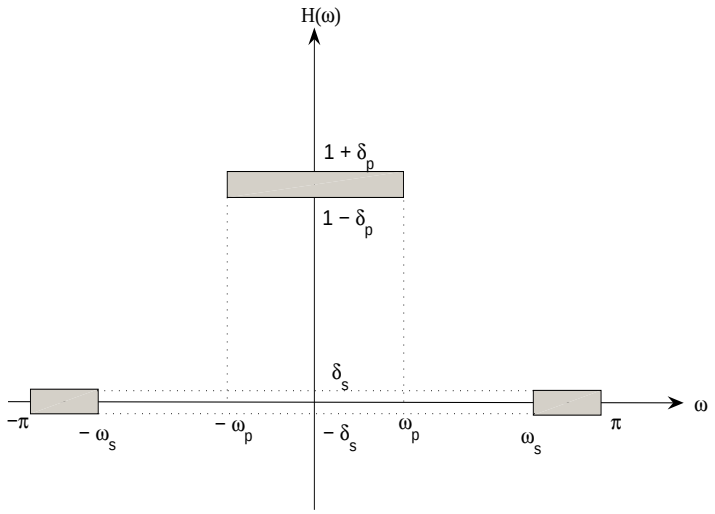

Figure 1: Frequency response specifications for a low pass filter. See text for details.

To complete the design we need to specify the phase of the filter as well. If the FRF is real then

$$H(e^{j\omega}) = H^*(e^{j\omega}).$$

The student should show that in the time domain this corresponds to $h[n] = h[-n]$ if $h[n]$ is real.

3 Iterative procedure

Specify bounds on the FRF as below

$$H_{id}(e^{j\omega}) - E_d(\omega) \leq H(e^{j\omega}) \leq H_{id}(e^{j\omega}) + E_d(\omega), \tag{3}$$

with $H_{id}(e^{j\omega})$ being the FRF of the ideal LP filter. To be specific:

$$H_{id}(e^{j\omega}) = \begin{cases} 1 & \omega \in F_p \\ 0 & \omega \in F_s \end{cases} \tag{4}$$

and

$$E_d(\omega) = \begin{cases} \delta_p & \omega \in F_p \\ \delta_s & \omega \in F_s \end{cases} \tag{5}$$

We have already defined the support $I = -N, -N+1, \ldots, -1, 0, 1, \ldots, N$ and this implies that the filter must have zero values outside this region.

The iterative algorithm consists of the following steps:

1. Begin with the $h[n]$ of the ideal frequency response (students should know this). This is essentially an IIR filter, therefore the student should drop terms outside the support I. $h_0[n]$ represents the 0^{th} iteration.

$$h_0[n] = \begin{cases} h_{id}[n] & n \in I \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

   A reasonable approximation to the support is shown later.

2. Here is what the k^{th} iteration looks like:

   - Compute the FFT of $h_k[n]$ on a dense grid of frequencies. If $2N+1$ is length of the filter then there should at least be $10N$ points in the FFT. If your edge frequencies are not on the grid, then choose frequency values on the grid that enable a tighter passband.
   - Apply the frequency domain constraint given below

$$G_k(e^{j\omega}) = \begin{cases} H_{id}(e^{j\omega}) + E_d(\omega) & \text{if } H_k(e^{j\omega}) > H_{id}(e^{j\omega}) + E_d(\omega) \\ H_{id}(e^{j\omega}) - E_d(\omega) & \text{if } H_k(e^{j\omega}) > H_{id}(e^{j\omega}) - E_d(\omega) \\ H_k(e^{j\omega}) & \text{otherwise} \end{cases} \tag{7}$$

   - Compute the ifft of $G_k(e^{j\omega})$ to get $g_k[n]$.
   - Any $g_k[n]$ that is non–zero outside I should be zeroed as below

$$h_{k+1}[n] = \begin{cases} g_k[n] & n \in I \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

   - If the MSE is less than a predefined threshold, terminate. MSE is defined as

$$\frac{\sum_{n=1}^{N} \left(h_k[n] - h_{k+1}[n]\right)^2}{N}$$

     Another way to check convergence is to see if we meet the filter specifications after some finite, but large number of iterations.

Are we expected to attain convergence? There is a theory in mathematics that gives a positive answer, known as the method of projections onto convex sets. The student should accept this but may research it on–line for their own edification.

How do we determine the filter order? Kaiser [1] showed that the filter order can be written as:

$$N = \lceil \frac{-20 \log_{10} \sqrt{\delta_p \delta_s} - 13}{14.6(\omega_s - \omega_p)/(2\pi)} + 1 \rceil.$$

This filter order should be considered a lower bound on the order, i.e., so as to meet the filter specifications we might need an order greater than that described above.

4 Issues

In reality, despite the theory of the method of projections onto convex sets, convergence can be exceedingly slow. The student should consider the following:

1. The placement of the stopband and the passband need not be exactly ω_s and ω_p but could be conservative.
2. The tolerance parameters (i.e., that permitting rippling) could be changed to slightly smaller values that δ_s and δ_p, again to be conservative, in order to speed convergence.

5 Example

A zero phase LP filter which has the following characteristics has to be designed:

$$H_{id}(e^{j\omega}) = \begin{cases} 1 & 0 \leq \omega \leq 0.4\pi \\ 0 & 0.6\pi \leq \omega \leq \pi \end{cases} \tag{9}$$

After the design process is followed we get a FRF as shown in Figure 2. Note the rippling in the passband and the stopband.

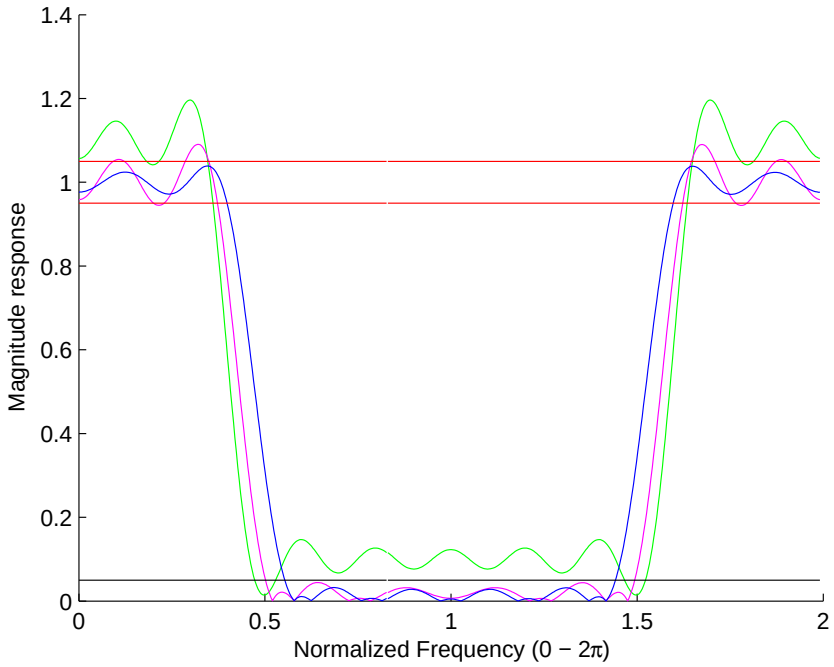

Figure 2: Example 1. This is a lowpass filter design for the example described by equation (9). The iterations 1, 3 and the final step are shown. In the final step, all the conditions on δ_p, δ_s, ω_p and ω_s are met.

6 What to do?

1. Write a MATLAB script (not function) that computes the filter given in Example 1 and plots a result similar to that given in Figure 2. Also hand in your filter coefficients, i.e., your code will print out the coefficients on the screen. Since you will be trying various filter sizes and difference tolerance parameters (δ_p, δ_s, ω_p and ω_s), also produce a table with columns showing number of iterations, the tolerance parameters and a final column, whether it converged or not. This final column will be filled in with a 'y' or a 'n'.
2. Use the Parks-McLellan algorithm (in MATLAB) to design the same filter. Plot your results in a graph similar to Figure 2. Compare the coefficients with the previous design step.

Due date: 11:59pm on 13 April 2007 by email (single script file + table) only to Haleh Safavi.

References

[1] J. F. Kaiser. Nonrecursive digital filter design using i_0–sinh window function. *Proc. IEEE Int. Sympo. Circuits and Systems*, 25:821–837, 1974.