University of Pennsylvania
Department of Electrical and System Engineering
Digital Signal Processing

Final Project: Filter Design and Filter Banks   Sunday, March 21st

Due: Monday, April 5th, 11:59PM

Project Teams: You should work individually and are required to complete Part A and Part B for full credit. Part C and D are extra credit.

1. **Part A:** FIR FILTER DESIGN 3-WAYS In this problem you will design FIR filters using three different methods discussed in lecture.

   (a) Design an FIR filter using Truncation
   Ideally, a low-pass filter with cutoff frequency ω_c should have a frequency response of

   $$H_{ideal}(e^{j\omega}) = \begin{cases} 1, & |\omega| \leq \omega_c \\ 0, & \omega_c < \omega \leq \pi \end{cases}$$

   and a corresponding impulse response of

   $$h_{ideal}[n] = \frac{\omega_c}{\pi} sinc\left(\frac{\omega_c n}{\pi}\right) \text{ for } -\infty < n < \infty$$

   However, no real filter can have this frequency response because $h_{ideal}[n]$ is infinite in duration. One method for creating a realizable approximation to an ideal filter is to truncate this impulse response. A truncated impulse response is of finite duration, yet the filter is still noncausal. In order to make the FIR filter causal, it must be shifted to the right by M units.

   To examine the effect of filter size on the frequency characteristics of the filter, write a Matlab function `LPFtrunc(N)` that computes the truncated and shifted impulse response of size N for a low pass filter with a cutoff frequency of $\omega_c = 2.0$. For each of the following filter sizes, plot the magnitude of the filters DTFT in decibels. Hints: The magnitude of the response in decibels is given by $|H_{dB}(e^{j\omega})| = 20log_{10}|H(e^{j\omega})|$. Note that the `log` command in Matlab computes the natural logarithm. Therefore, use the `log10` command to compute decibels. To get an accurate representation of the DTFT make sure that you compute at least 512 sample points using the command `[X,w]=DTFT(filterResponse,512)`. Design two filters with size N=21 and N=101.

   - Submit the plots of the magnitude response for the two filters (not in decibels). On each of the plots, mark the passband, the transition band and the stopband.
   - Submit the plots of the magnitude response in decibels for the two filters.

- Explain how the filter size effects the stopband ripple. Why does it have this effect?

Now download the noisy speech signal `nspeech2.mat`, and load it into the Matlab workspace. Apply the two filters with the above sizes to this signal. Since these are FIR filters, you can simply convolve them with the audio signal. Listen carefully to the unfiltered and filtered signals, and note the result. Can you hear a difference between the two signals? In order to hear the filtered signals better, you may want to multiply each of them by 2 or 3 before using sound.

(b) Design a symmetric low-pass FIR filter using `firpmord` and `firpm` in Matlab to meet the design specifications given below:

$$\begin{aligned}
\omega_p &= 1.8 \\
\omega_s &= 2.2 \\
\delta_p &= 0.05 \\
\delta_s &= 0.005
\end{aligned}$$

Compute the DTFT of the filters response for at least 512 points, and use this result to compute the passband and stopband ripple of the filter that was designed. Adjust the filter length until the minimum order which meets the design constraints is found. Plot the magnitude of the DTFT in dB of the final filter design.

Submit the final measured values of filter length, passband ripple, and stopband ripple. How accurate was the filter order computation using Matlabs `firpmord`? Submit the plot of the filters DTFT.

(c) Now you must design a low-pass filter for the removal of high-pass noise from a speech signal.

For this part of the project, the file `projIA.mat` will need to be first loaded into Matlab. After downloading this file from the course website, copy it into your Matlab working directory and type `load projIA`.

In this part of the project, it may be helpful to use Matlabs order estimation functions (e.g. `buttord`, `cheb1ord`, `cheb2ord`, ...). When using these functions the following definitions may be helpful. A specification of ripple magnitude δ yields a FIR filter design with a maximum passband gain of $(1 + \delta)$ (or $20log10(1 + \delta)$ dB), and a minimum passband gain of $(1 - \delta)$ (or $20log10(1 - \delta)$ dB). A specification of stopband gain ξ yields an FIR filter design with a maximum stopband gain of ξ (or $20log10(\xi)$ dB).

Clearly, to compare filters designed with Matlab, we must find a correspondence between our desired specifications and the input parameters to the filter design functions. Let us thus define G_{pbmax} (dB), G_{pbmin} (dB), and G_{sbmax} (dB) to refer to the desired maximum passband gain, minimum passband gain, and maximum

stopband gain, respectively. In order to have free control over each of these parameters, we also introduce k, a linear scaling term.

Before beginning the design calculate the relevant parameters and include your solutions as part of the project writeup: Find δ, ξ, and k in terms of G_{pbmax}, G_{pbmin}, and G_{sbmax}

Now you should design a low-pass filter for the removal of high-pass noise from a speech signal. The noisy signal is stored in the variable `noisy` and was sampled at $f_s = 44100$ Hz. It consists of a summation of a speech signal which was band-limited at 4 kHz using a low-pass filter with a very narrow transition band, and a noise signal which was filtered at 4 kHz using a high-pass filter with a very narrow transition band. To filter the noise we must design a discrete-time filter with the following parameters:

- Passband edge: 2500 Hz.
- Stopband edge: 4000 Hz.
- Maximum gain in the passband G_{pbmax}: 40 dB.
- Minimum gain in the passband G_{pbmin} : 37 dB.
- Maximum gain in the stopband G_{sbmax}: -55 dB.

Because it may take a few iterations to get each filter right, we suggest you write a .m Matlab script for each filter. Note that in this part of the project, you may end up with very high order filters since the specifications are rather severe.

Design a DT filter of each of the following types based on the specifications given above: Butterworth, Chebyshev Type I, and Chebyshev Type II.

For each of the designs, provide the following in your project write-up:

i. The order of the filter.
ii. The number of multiplication operations per input sample required to implement the filter. Be sure to explain the structure you assume.
iii. Plot the magnitude response (in dB) from $\omega = 0$ to $\omega = \pi$ using `freqz`. Plot a detailed view of the magnitude response, focusing on the passband ripple (linear scale). Plot the group delay (in samples) using `grpdelay`. (Use subplot to fit the three plots on the same page for each filter.)
iv. Plot the pole-zero diagram.
v. Plot the impulse response using filter and stem for 100 samples. (You can use `subplot` to fit the pole-zero diagram and the impulse response on the same page.) Filter noisy using your de-noising filter. Listen to the filtered and original files. How do they compare?

2. **Part B:** MULTI-CHANNEL FIR FILTER-BANK IN ONE DIMENSION
   You are to implement a perfect reconstruction filter-bank (PR-FB) using Matlab, and

apply it to one-dimensional signals (audio). You can use Matlab routines such as `firpr2chfb` and `mfilt.firdecim`, etc., for this purpose.

(a) Design a 4-channel octave band scheme, in which the full-band is decomposed into 2 low frequency bands (1/8 of full-band each), an intermediate 1/4 of full-band, and the upper 1/2 high-frequency band. Use reasonable parameter values (filter length and band edges for the filters). Filter length should not be unreasonably long.

(b) To test the filter-bank, use a simple test signal such as a number of sinusoids at different frequencies, or any other test signal of your choice. Submit plots to show your test signals to demonstrate the filter-bank operation.

Project Submission: Your report submission for this project will consist of two parts.

- Project Report:
  You should submit by the due date a single file project report (preferably pdf) explaining what you did and the results you obtained, including figures, test cases, interpretations and comments, as well as responses to any specific questions asked above. You do not need to include your code in your report unless you refer to it in your write-up. If you must include it do so **in an Appendix**. The report must be uploaded to Canvas by midnight on the due date.

- MATLAB Code and Other Soft Files: Also submit (upload) by the due date through the Assignments Area on the ESE 531 Canvas Site all supporting material (all your Matlab code files, test input/output files, and any other results files). Ideally all placed in a compressed .zip file. Please make sure you follow the proper procedure for submitting files through Canvas.

3. **Part C:** FILTER-BANK EQUALIZER
   Use the filter-bank from Part B on an audible waveform such as a segment of music or speech.

   (a) Use the filter-bank to implement a simple equalizer for the speech or music; by this we mean impart different gains on the 4 subband signals before reconstruction, to enhance certain frequency bands relative to others (e.g. bass boost, high-frequency boost, etc). Submit plots to show your test signals to demonstrate the equalizer performance.

   (b) Find out if introduction of small relative delays in the four subbands prior to reconstruction affects your perception of the audio (i.e. determine to what extent differential phase shifts or delays are important in perception of speech or music). Show any tests conducted, describe the audible results and any conclusions you find.

4. **Part D:** MULTI-CHANNEL FIR FILTER-BANK IN TWO DIMENSIONS
   Extending what you did in Part B, you are to implement a perfect reconstruction filter-bank (PR-FB) using Matlab, and apply it to images. You can use Matlab routines such as `firpr2chfb` and `mfilt.firdecim`, etc., for this purpose. Image Processing and related toolbox functions in Matlab may be useful (e.g. `imread, imshow, mat2gray, hist, imhist,` etc.)

   (a) Design a one-dimensional 2-channel PR-FB. This can be applied for separable (rows and columns) sub-band analysis/synthesis of two-dimensional images. For image processing, strict spatial frequency separation may not be as critical as in other applications. Experiment with a small filter order (between 6 and 12) and a longer one. Use a reasonable choice for filter band-edge. Your design will yield the analysis and synthesis filters H0, H1, G0, and G1. (You might want to first test the filter bank on some simple one-dimensional test signal.)

   (b) You will now use a grayscale 8-bit, 512x512 image to test image sub-band decomposition. A couple of possible test images (`man.gif, owl.gif`) are posted on the course calendar; any other reasonable test image can be used. Apply your 2-channel analysis filter bank of part (a) to your test image(s), to get decomposition into (256x256) sub-band components (Low-Low or LL, Low-High, High-Low, and High-High). The image filtering can be implemented as separable row-column filtering operations. Test for exact reconstruction when synthesizing the image back from these four sub-bands.

   (c) Perform reconstruction or synthesis as above, but this time with all pixels of the LH, HL, and HH sub-band images set to 0. Comment on the nature of the reconstruction you see.

   (d) Now instead of using single zero levels as in (c) for all pixels of the LH, HL, and HH sub-band images, experiment with less severe compression by using one bit (two levels) per pixel quantization for these three sub-band images, and also with three level (and, if you want, four level or 2-bit) per pixel quantization. To

design reasonable quantization schemes (levels and thresholds) you may consider histograms of pixel values for the sub-bands.

(e) Do a further decomposition of the Low-Low (LL) image obtained in part (b) by applying the same 2-channel analysis filter bank to it. (This corresponds for 1-D signals to octave- band filtering into 3 sub-bands). Now apply considerations similar to those of parts (c) and (d) to see if reasonable reconstructions can be obtained with simple quantization. Note: There can be different delays or shifts for different sub-band octave components.