Music 421A
Spring 2019-2020
Homework #4
FIR Filter Design
Due in one week

Theory Problems

1. (5 pts) Derive the ideal impulse response corresponding to the desired amplitude response

$$H(e^{j\omega T}) = \begin{cases} 1, & 0 \le \omega_1 \le |\omega| \le \omega_2 \le \pi/T \\ 0, & \text{otherwise} \end{cases}$$

   [Hint: Use a Fourier theorem to make use of the answer for the ideal lowpass filter.]

2. (5 pts) When designing an FIR bandpass filter, find the benefit of choosing the lower cut-off frequency f_1 to be equal to the difference between the Nyquist limit $f_s/2$ and the upper cut-off frequency f_2. In other words, what is the benefit of the constraint $f_1 = f_s/2 - f_2$? (Show that the claimed benefit is obtained in general for any $f_1 \in (0, f_s/4)$.) [Hint: Note that this condition enforces symmetry about $f = f_s/4$, which is advantageous computationally.]

3. (5 pts) What length Blackman window is required to resolve a sinusoid at 100 Hz and another one at 101 Hz? State your definition of resolution in this context, and draw a sketch showing the two sinusoids and the window transform in the frequency domain, with the window transform being centered on one of the sinusoidal frequencies.

Lab Assignments

1. (17 pts) Design a length $M = 51$ Chebyshev window with 40 dB of sidelobe attenuation using

   - `chebwin`
   - `firpm` (formerly called `remez`)
   - `linprog` [Hints: $M = 2L+1$, where $L = 25$. Use a frequency grid length K which is much larger than L, such as in the hundreds.]

   in matlab. Normalize each window such that the main lobe of the window transform has peak magnitude 0 dB. For `firpm` and `linprog`, set the normalized transition bandwidth to 0.068 (where 0 is dc and 1 is half the sampling rate, is as commonly used in matlab). Using `cputime` or `tic` and `toc`, etc.,[1] measure the average compute-time for ten iterations of each of the three methods above.

(a) (5 pts) Report the three average compute times. Divide the two longer compute-times by the smallest compute time and report those two speed ratios.

(b) (5 pts) Plot an overlay of the three windows in the time domain. Repeat with the chebwin case subtracted out.

(c) (5 pts) Plot an overlay of the window magnitude spectra in the frequency domain. Use at least a factor of five zero-padding.

(d) (2 pts) Which method is the most accurate and why do you think that is? Describe any numerical considerations you can see.

(e) **Optional:** For each method, find the order (to within 10%) above which numerical problems become significant. [It is probably easiest to inspect the magnitude spectra to detect numerical troubles.] [To obtain an accuracy of 10%, one can increase the order by 10% each trial, or double it each trial, followed by 10% increments over the last interval, etc.]

2. (13 pts) Download the sound file `noisypeaches.wav`[2] containing speech embedded in white noise.

   (a) (5 pts) Plot the spectrogram of `noisypeaches.wav` to help you understand its spectral content but there is no need to submit it. (Matlab's `spectrogram` or Octave's `specgram` function can be used for this.) Design a low pass filter using the window method with a Kaiser window of length 100 and β=10. The cut-off frequency of the filter should be 4 kHz. Plot its impulse response and magnitude of frequency response.

   (b) (3 pts) Apply this filter to the noisy speech signal either by the FFT method or simple filtering. Listen and describe the result compared to the original.

   (c) (5 pts) Now downsample the original noisy speech signal by a factor of two by simply throwing away every other sample. Listen to the result and compare it to the original higher sampling rate. Repeat the same downsampling scheme on the lowpass filtered speech signal and again, compare with its higher sampling rate version. Why does the latter pair (lowpass filtered) sound more similar than the first pair (unfiltered)?

3. (10 pts) Design a real, linear-phase, FIR bandpass filter using `firpm()` in Matlab with the following specifications: Sampling rate f_s = 100 Hz, pass-band from 20 Hz to 30 Hz, stop-band from 0 to 10 Hz and 40 to 50 Hz, δ_s = 0.01 (-40 dB) ripple in the

[1] `https://www.mathworks.com/help/matlab/matlab_prog/measure-performance-of-your-program.html`
[2] `https://ccrma.stanford.edu/~jos/sasp/hw/noisypeaches.wav`

stop-band, and $\delta_p = 0.02$ ripple in the pass-band, which is unity gain. The filter thus has transition bands from 10 to 20 Hz, and from 30 to 40 Hz. Turn in a listing of your Matlab code, and the result of its execution (e.g., using the `diary` command), which should include a print-out of the filter length, a listing of the filter coefficients, and a plot of the filter amplitude response on a dB vertical scale. [Hint: Start with '`help firpmord`' in Matlab. Check to make sure the filter specifications are met using the output of `help firpmord`, and if not, increase the filter order until they are.]