PIC 16, Winter 2018 – Assignment 7F

Assigned 2/23/2018. Code (a single `.py` file) due by the end of class 2/28/2018 on CCLE. Hand in a printout of this document with the self-assessment portion completed by the end of class on 2/28/2018.

In this assignment, you will eliminate "Woo Guy" from Super Bowl 50 using SciPy's `fftpack`.

Task

If you don't know who I mean by "Woo Guy", consider yourself lucky. You can read about him to your heart's content by searching Google for "Woo Guy Super Bowl 50".

The whole time the game was on I wondered when CBS would catch on and filter him out of the broadcast, but they never did. It would have been so simple. The pitch of Woo Guy's incessant calls were so consistent that simply reducing the amplitude of a band of frequencies would have improved the viewing/listening experience for millions of fans.

Your job is to show CBS how it's done and filter "Woo Guy" out of a snippet of the game's audio. Apparently CBS has the footage on lockdown so the only clip I could find was this. I've extracted the audio here. If you can't hear "Woo Guy", I've amplified him here, and if that's not enough, I've added a nasty pure tone around his pitch here.

Take the FFT of the entire signal and plot the *magnitude* (of the resulting complex numbers) against the frequencies (returned by `fftfreq`). If you're unfamiliar with FFT, this plot shows you the relative amplitude (power) of each pitch (frequency) in the audio file. (The plot is symmetric for technical reasons; see this for much more information.) Woo Guy stounds out as one of the peaks (and its mirror); a range of frequencies that stands out above the rest. He's not the highest peak, but he's up there. You can find out which frequencies he is by:

- comparing the pitch you hear to a known frequency (try this tone generator), or
- eliminating peaks (by setting small ranges around their frequencies to zero) and their mirrors, converting back to audio, and listening to whether the undesirable sound goes away.

Eliminate Woo Guy from the audio by zeroing the FFT ranges corresponding with his frequencies, take the IFFT, save the resulting to a `.wav` file, and enjoy the clip free of that annoying noise.

To save you some time, some hints:

- This is stereo audio, that is, there is separate audio for each ear, so when you read the `.wav`, you'll get a 2D array. It's OK with me if you reduce the audio to mono (discard one audio channel), or you can specify the axis along which you want to take the FFT
- Be sure to eliminate *both* peaks that correspond with Woo Guy – the one on the left, and its twin on the right (due to symmetry) – before taking the IFFT.
- Pay attention to the data type of the array you're saving to the `.wav` file. As in the preparation, turn the volume down before listening with headphones for the first time (in case it's too loud).

Self-Assessment

Does your code plot the magnitude of the FFT of the original signal? (30 points)

Plot the magnitude of the FFT of the signal *after* zeroing Woo Guy's frequencies for 30 more points.

Does your code eliminate "Woo Guy" and save the edited sound file? (40 points)