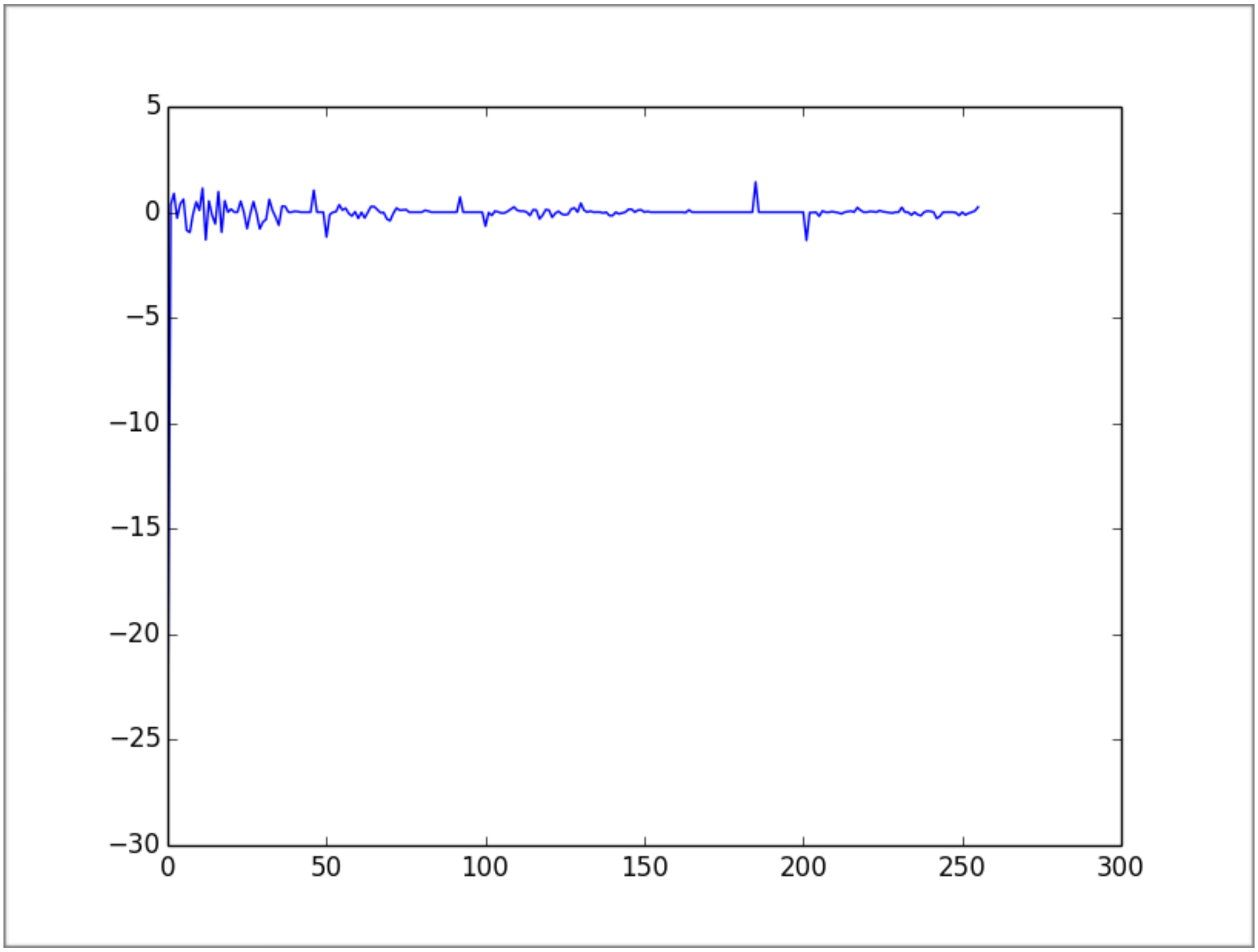
Neil Lakin
10/8/14

Homework 1, Signal Processing

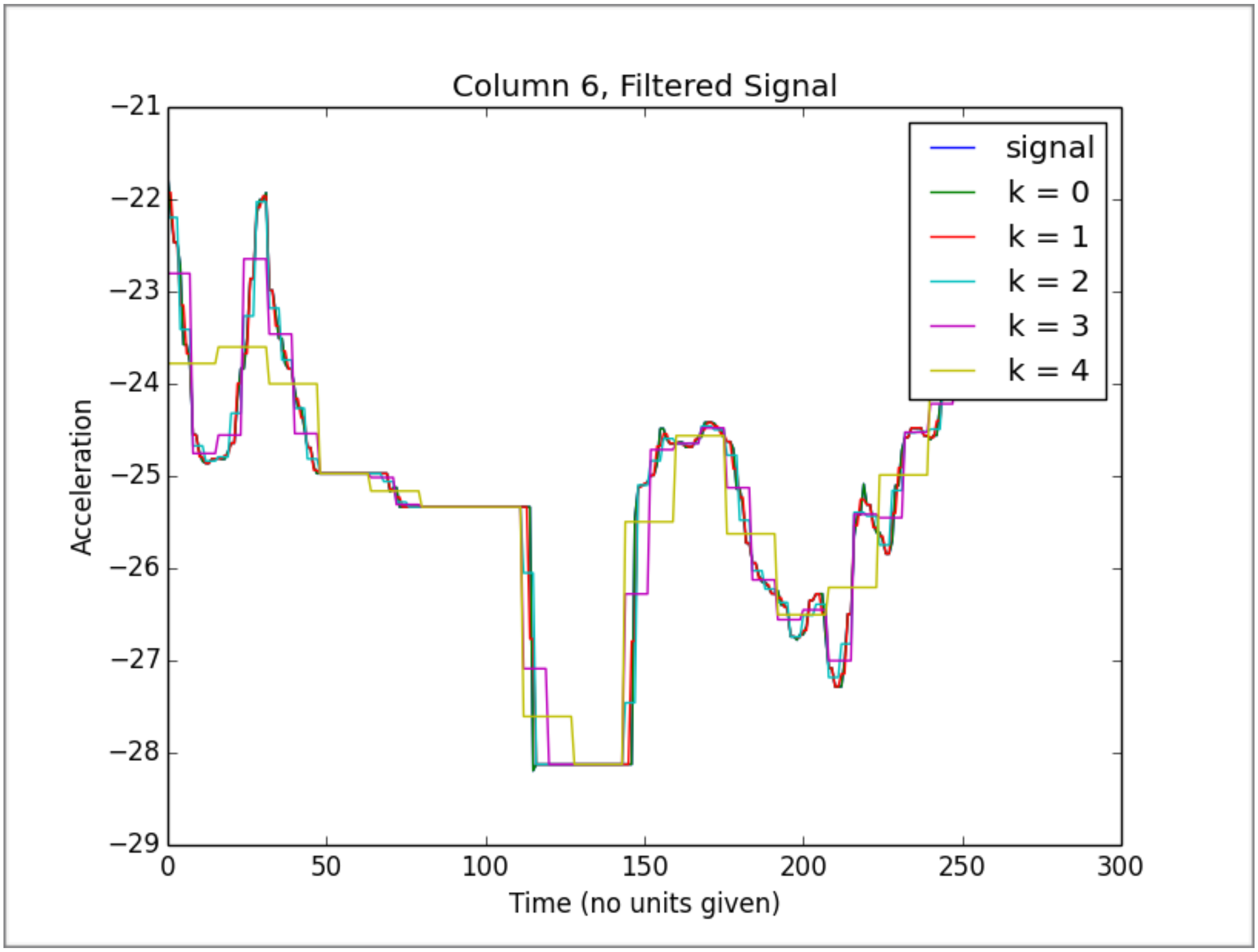
Task 2

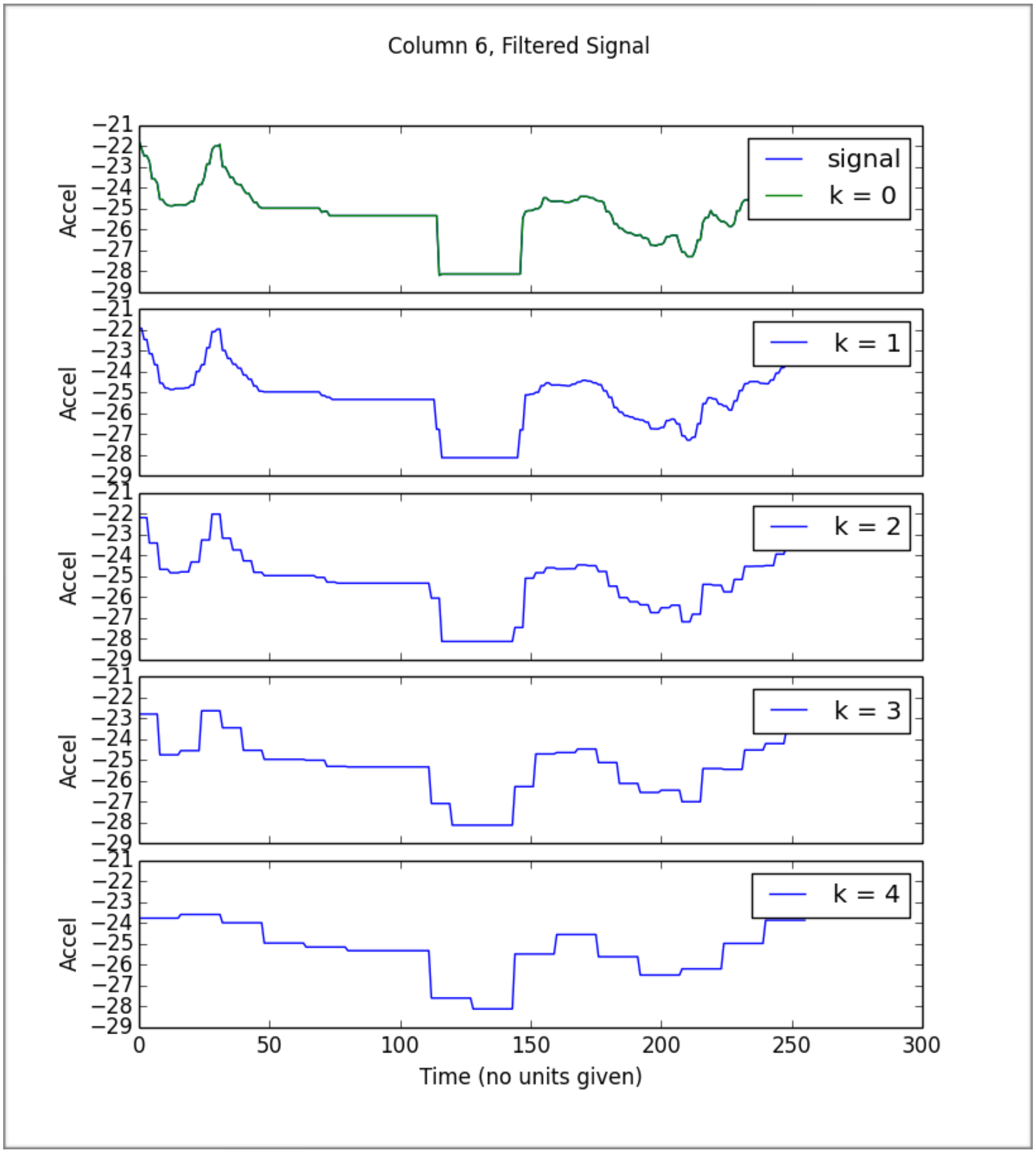
A) Here's the output for plotting the Haar transform—as in the example script, I took the data from column 6 in the original (unlabeled) CSV.



B) The first order differences are the distances from the pairwise mean for each pair of points in the dataset, so the most positive values represent the largest negative changes from one point to another (since the difference is calculated by subtracting the average from the leftmost point in the pair, a positive value means the average is lower than the leftmost point and therefore the rightmost point is the lower of the two) while the most negative values represent the largest point-to-point increase. Finding the maximum values in the right half of the transform is equivalent to finding the points in the signal where the absolute value of the derivative is highest.

C) The plots for parts 1-3 are shown below. The original signal is labeled ‘signal’, the signal that has been Haar transformed, then inverse Haar transformed, is labeled ‘k = 0’, and the filtered signals are labeled by their value of k. The first figure shows all of the filtered signals in one plot; the second has them stacked in subplots individually. Writeup follows.





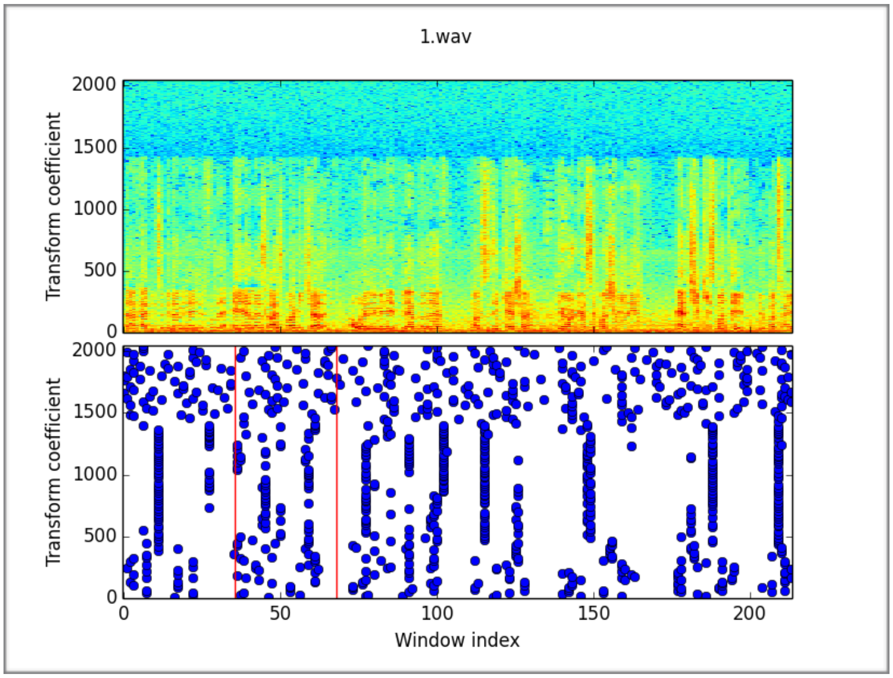
C, continued)

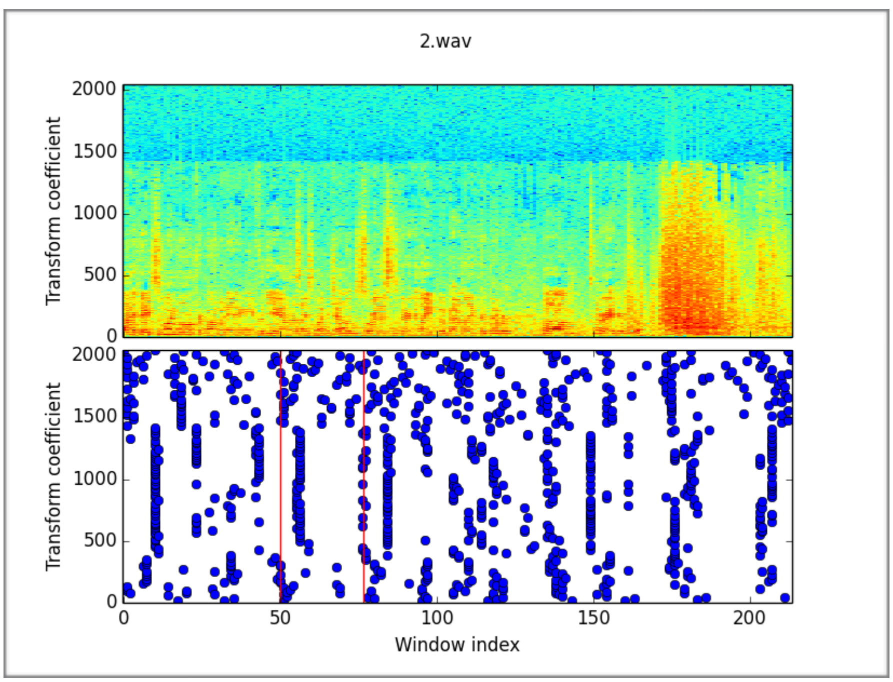
1. As can be seen from the above graphs, the original signal x and $Haar^{-1}(Haar(x))$ are indeed identical.

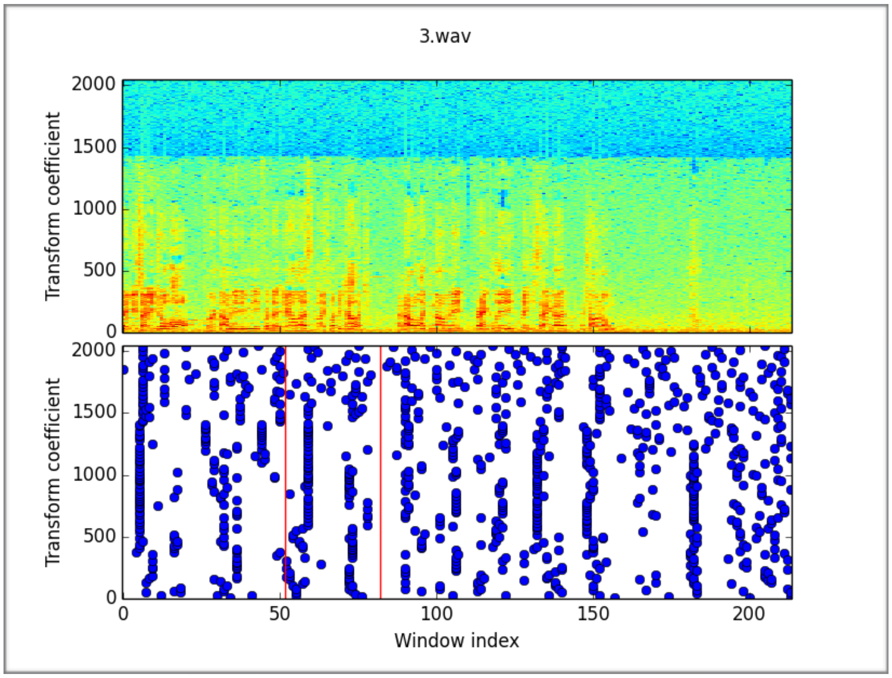
2 - 3) As k increases, rapid changes in the signal are smoothed away. As k approaches 4, fairly significant features are smoothed away, such as the local minimum around 15. Note that 'truncating' the signal — reducing the size of the array by half, and running the inverse Haar on the result — results in output that looks identical to the output above, but compressed by half in the time domain, while replacing the second half with zeros maintains the original scale.

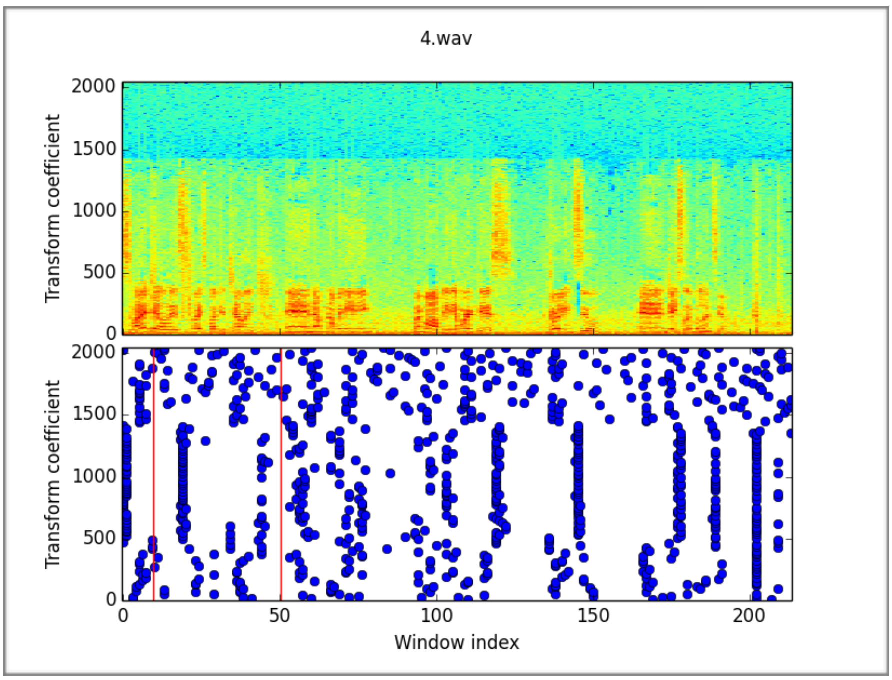
Task 3A

1-3) Below are plots showing the STFT for four audio samples, their peaks, and the start and endpoints from marks.xlsx.









4) The data is pretty structured, especially for frequencies below Transform Coefficient = 1400. The strong vertical bars look promising. However, the start times indicated by the red lines don't necessarily correlate well to the locations of the 'bars' of peaks; this is particularly true in 4.wav. For higher frequencies (> 1400), the data looks like noise; at lower frequencies (< 300) the structure looks much more complex, but still orderly. Note that 2.wav appears to catch the splash—the large smear around window index 170. Even for the splash, which should be pretty much white noise, there is a hard frequency cutoff around 1400, which means that cutoff is likely an artifact of the recording format in all samples.

5) I built my 1D array by taking the peaks array of peaks, ignoring frequencies > 1375, and taking the largest value for each window index.

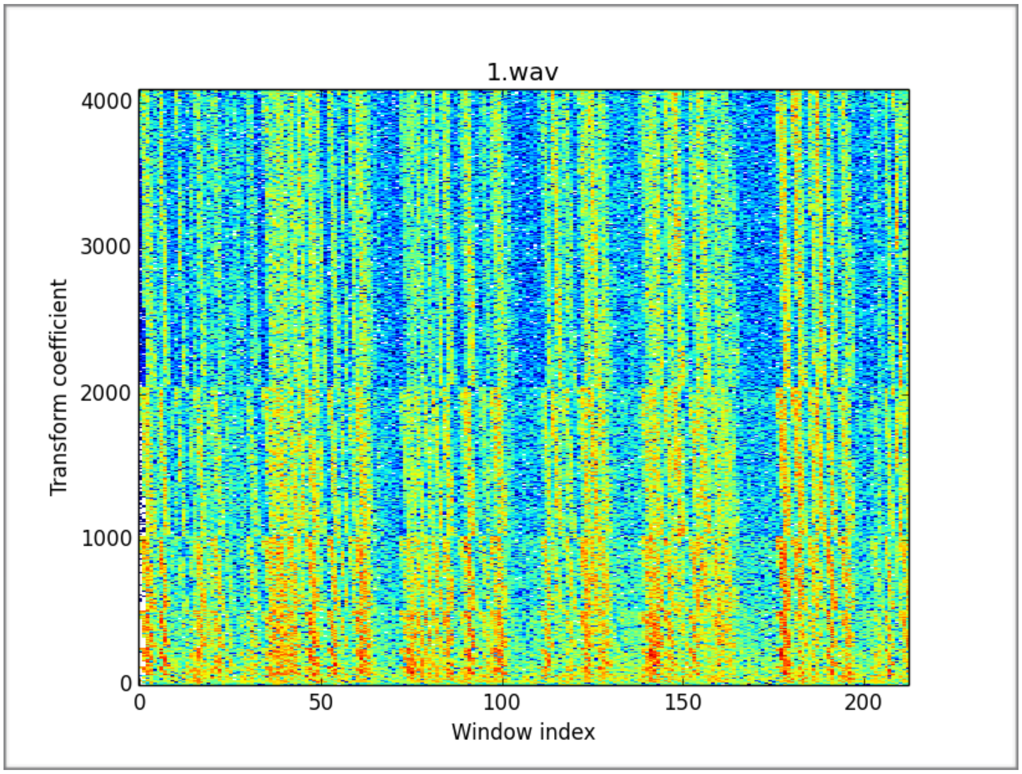
6) 2.wav seemed to be nicely structured, with the first vertical bar immediately preceding a distinctive feature in the peak graph, so I tried using it as a template to cross-correlate 1.wav, 3.wav, and 4.wav. I got the following shifts:

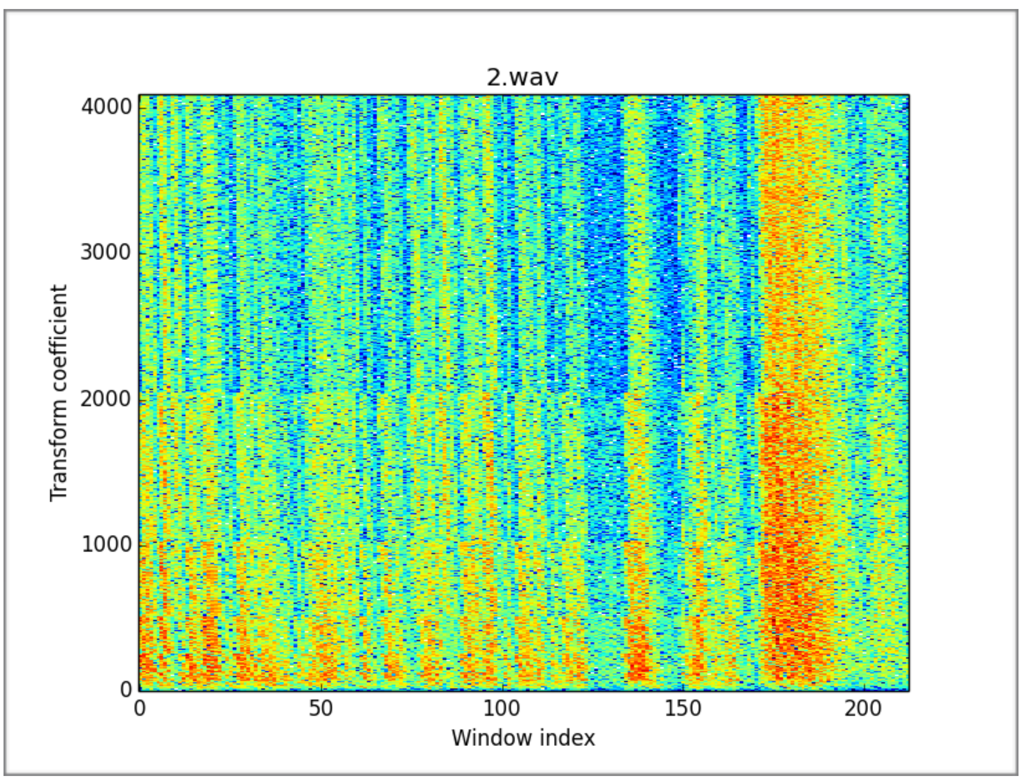
Offsets from 2.wav in seconds:
1.wav: 0.14, real value = - 0.67s
3.wav: 0.046, real value = 0.090s
4.wav: -0.046, real value = -1.87s

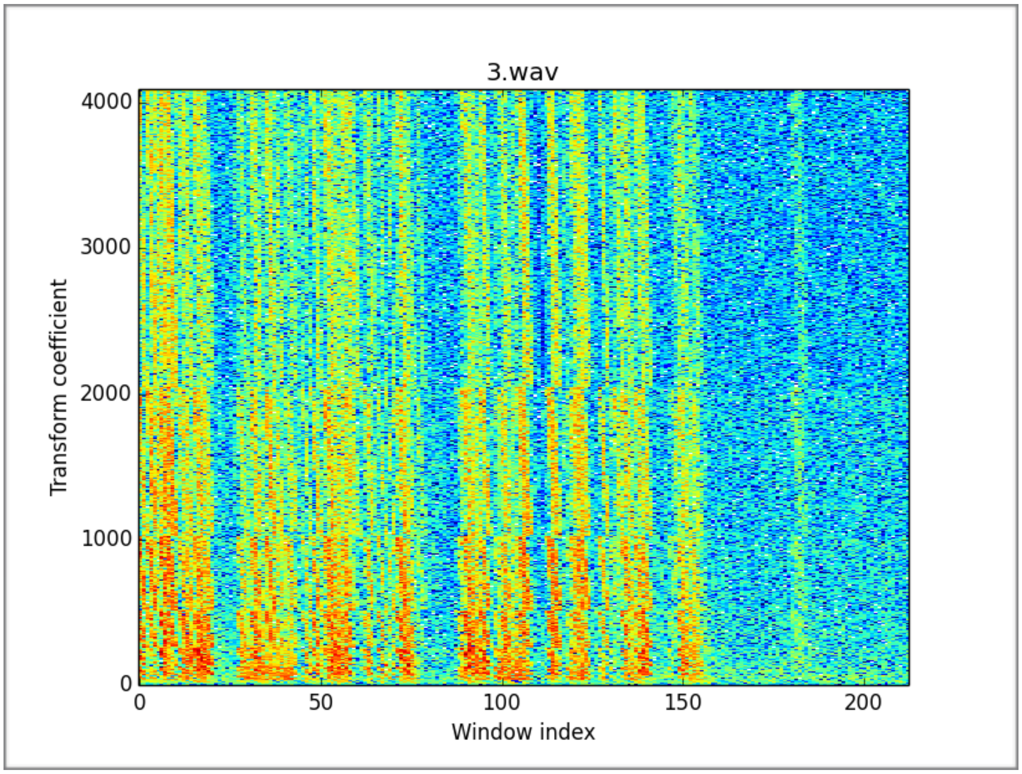
Obviously, this isn't great. It did the best job matching 3.wav, but was still off by ~100%. 4.wav did the worst, which isn't surprising—as noted earlier, the 'start' line doesn't correlate to any distinctive feature in the peak plot, unlike the other 3 files. This could possibly be much improved if I used a more sophisticated mechanism for 'flattening' the peaks into a 1D array, or if I tried correlating to the 2D peaks graph.

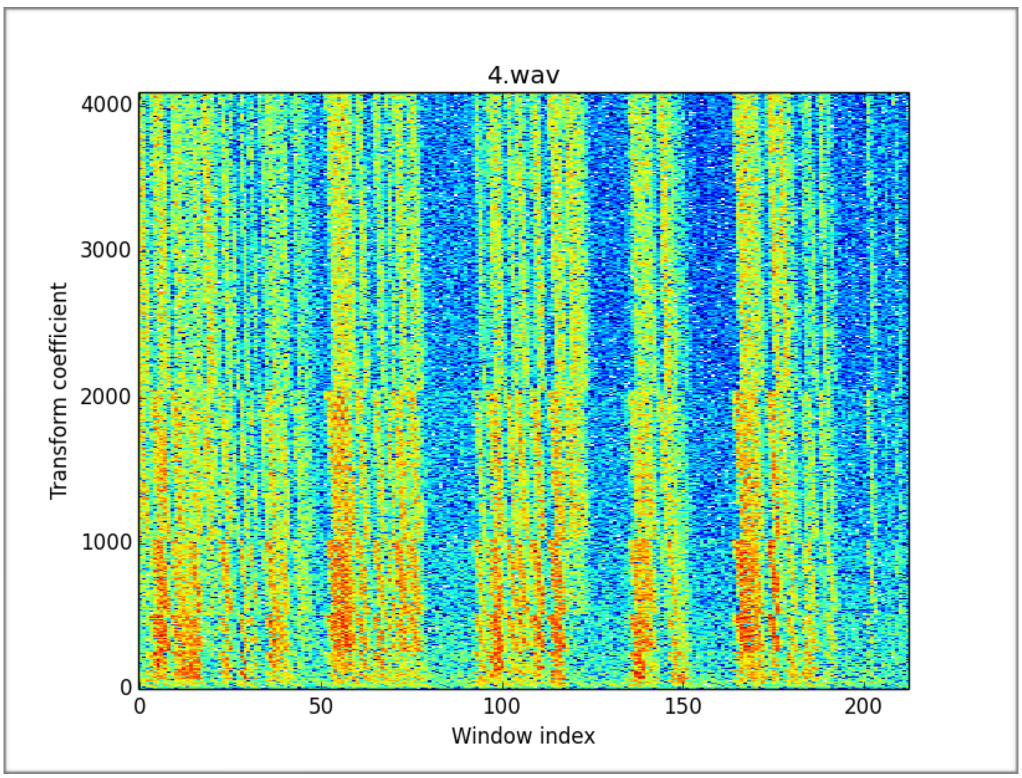
Task 3B

1-2) Here are the short time Haar transforms for the clips discussed in part A:









3)

The biggest advantage of the STFT over the Haar transform is that it takes advantage of the structure of an audio signal in a way that the Haar transform does not. Audio signals actually are composed of a bunch of sinusoids of different frequencies, so taking the Fourier transform of the signal is really breaking it up into its components. Because the signal is fundamentally sinusoidal, taking the STFT results in distinctive features. Clearly, there is structure in the Haar transform, as shown in these diagrams—the vertical bars in the Haar transforms, which correlate to time windows where the signal is changing rapidly, line up fairly well with the peaks in the STFT—but a fair amount of information is lost. This loss is seen in the way these diagrams are basically just vertical lines: while the STFT reveals a distinct structure for each frequency (especially in the difference between the low frequencies and the mid frequencies), the Haar transform just gives a vertical bar wherever something interesting is happening. The low-frequency component is almost totally lost, as well—none of the patterns you can see for coefficients < 300 in the STFT are visible here.

Task 3C

It may be worth trying a wavelet transform with a different wavelet—maybe a sinc function? A sinusoidal wavelet might be better for extracting some of the frequency information from the signal.