Lab 4

Michael Eller

March 28, 2016

Contents

1 Introduction

During this lab, we will be investigating the implementation of Layer III of MPEGG 1, also known as mp3. We will fist develop several subband filters to decompose and reconstruct the original audio signal. We will be using a polyphase pseudo Quadrature Mirror Filter to deconstruct and eventually reconstruct the original audio signal.

2 Cosine Modulated Pseudo Quadrature Mirror Filter: Analysis

In this section, we will manipulate the equations that mathematically describe the analysis filter.

Consider the filter h_K, then the output of the combined filtering by h_k and decimation is given by

$$s_k[n] = \sum_{m=0}^{511} h_k[m]x[32n - m] \tag{1}$$

where

$$h_k[n] = p_n[n] \cos\left(\frac{(2k+1)(r-16)\pi}{64}\right) \quad k = 0, \dots, 31,\ n = 0, \dots, 511 \tag{2}$$

and p_0 is a prototype lowpass filter. The role of h_k is clear: the modulation of p_0 by $\cos\left(\frac{(2k+1)(r-16)\pi}{64}\right)$ shifts the lowpass filter around frequency $(2k+1)\pi/64$. Equation 1 requires $32 \times 512 = 16,384$ combined multiplications and additions to compute the 32 outputs $s_1, \dots, s_{32}$ for each block of 32 samples of the incoming signal. This is simply too slow to be properly effective.

We define:

$$c[n] = \begin{cases} -p_0[n] & \text{if } [n/64] \text{ is odd} \\ +p_0[n] & \text{otherwise} \end{cases} \tag{3}$$

then

$$h_k[64q + r] = c[64q + r] \cos\left(\frac{(2k+1)(r-16)\pi}{64}\right) \tag{4}$$

Using the notations of the standard, we further define

$$\boxed{M_{k,r} = \cos\left(\frac{(2k+1)(r-16)\pi}{64}\right), \quad k = 0, \dots, 31,\ r = 0, \dots, 63} \tag{5}$$

then

$$h_k[64q + r] = c[64q + r]M_{k,r} \tag{6}$$

and

$$s_k[n] = \sum_{r=0}^{63} \sum_{q=0}^{7} c[64q + r]M_{k,r}x[32n - 64q - r] \tag{7}$$

$$= \sum_{r=0}^{63} M_{k,r} \sum_{q=0}^{7} c[64q + r]x[32n - 64q - r] \tag{8}$$

In summary, for every integer $m = 32n$, multiple of 32, the convolution from equation 1 can be quickly computed using the following three steps,

$$z[64q + r] = c[64q + r]x[m - 64q - r], \qquad r = 0, \dots, 63,\ q = 0, \dots, 7 \tag{9}$$

$$y[r] = \sum_{q=0}^{7} z[64q + r], \qquad r = 0, \dots, 63 \tag{10}$$

$$s[k] = \sum_{r=0}^{63} M_{k,r}\, y[r], \qquad k = 0, \dots, 31 \tag{11}$$

Even further speedup can be obtained by using a fast DCT algorithm to compute the matrix-vector multiplication in equation 11.

Assignment

1. Write the MATLAB `pqmf` that implements the analysis filter bank described in equations 5-9. The function will have the following template:

   `[coefficients] = pqmf (input)`

   where `input` is a buffer that contains an integer number of frames of audio data. The output array `coefficients` has the same size as the buffer `input`, and contains the subband coefficients.

   The array `coefficients` should be organized in the following manner:

   $$\texttt{coefficients} = [S_0[0]\ \dots\ S_0[N_S - 1]\ \dots S_{31}[0]\ \dots\ S_{31}[N_S - 1]] \tag{12}$$

   where $S_i[k]$ is the coefficient from subband $i = 0, \dots, 31$ computed for the packet k of 32 audio samples. Also N_S is the total number of packets of 32 samples:

   $$N_S = \frac{\texttt{Samples}}{32} = 18 * \texttt{nFrames} \tag{13}$$

   The organization of `coefficients` is such that the low frequencies come first, and then the next higher frequencies, and so on and so forth.

2. Analyse the first 5 seconds of the following tracks, and display the array `coefficients`,

   - sample1.wav, sample2.wav
   - sine1.wav, sine2.wav
   - handel.wav
   - cast.wav
   - gilberto.wav

   Comment on the visual content of the arrays `coefficients`.

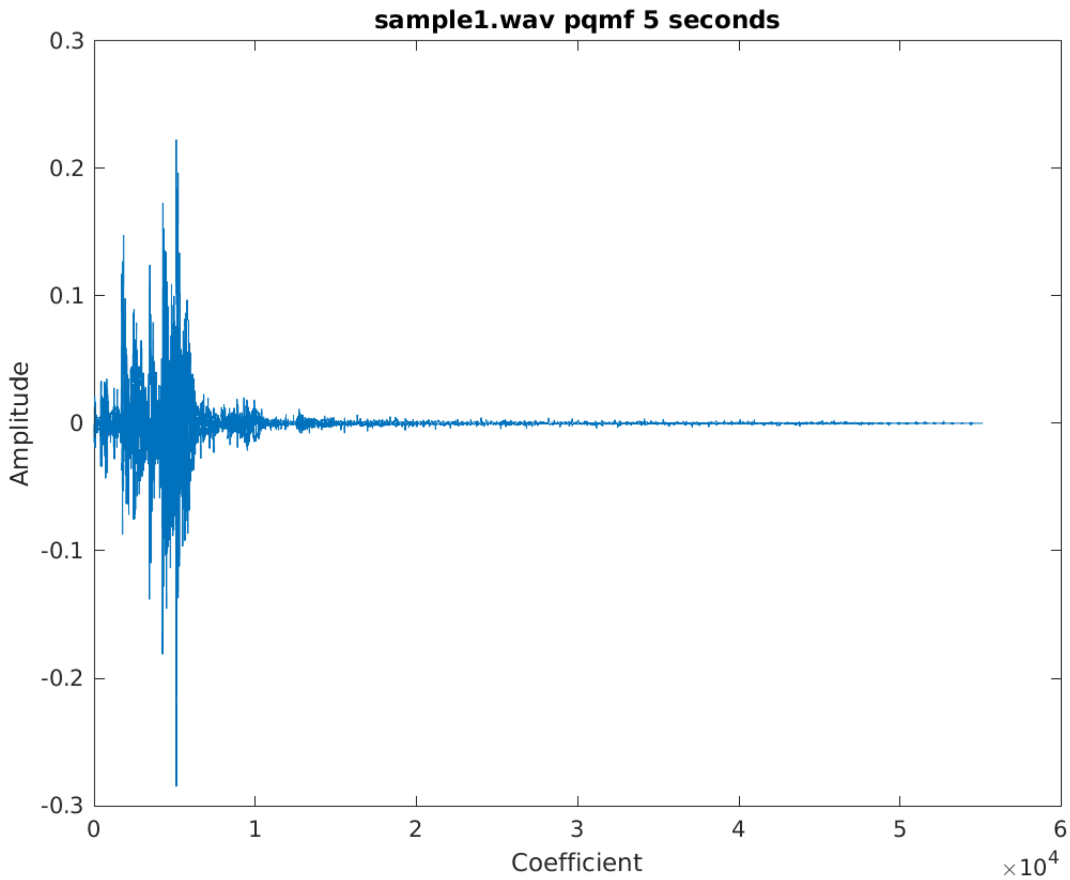


Figure 1: PQFM: Sample1 5 seconds

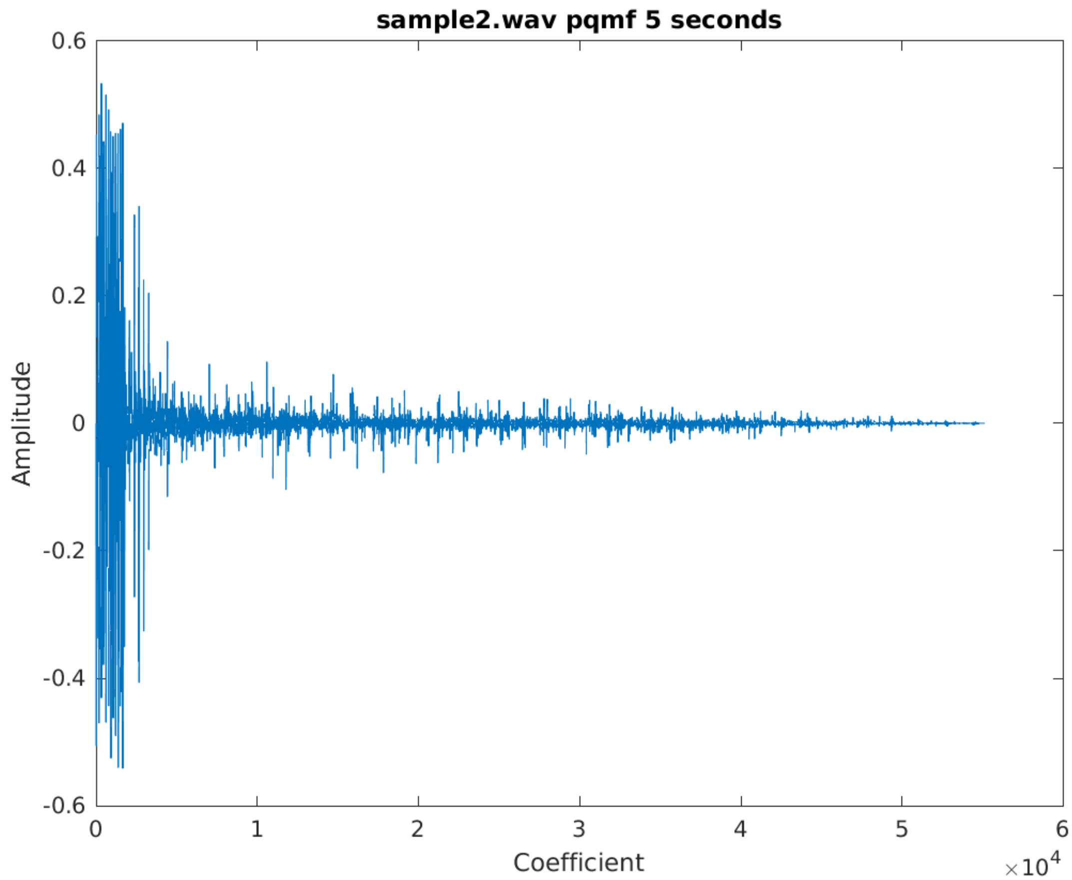
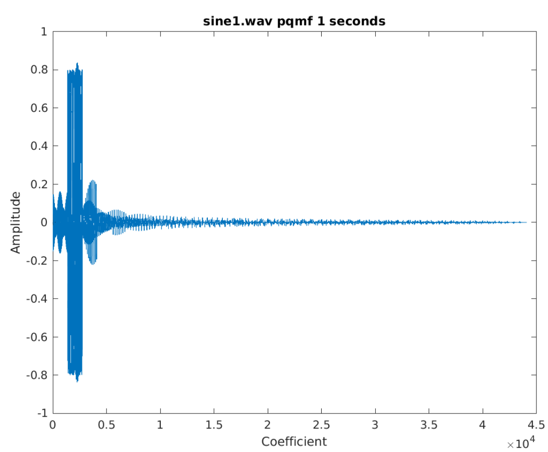
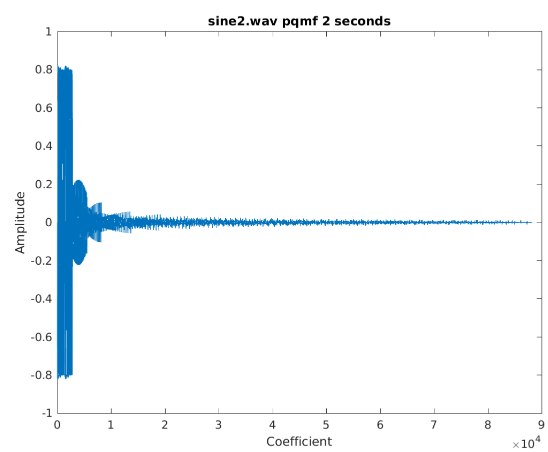


Figure 2: PQFM: Sample2 5 seconds



(a) PQFM: Sine1



(b) PQFM: Sine2

Figure 3: Sine Waves

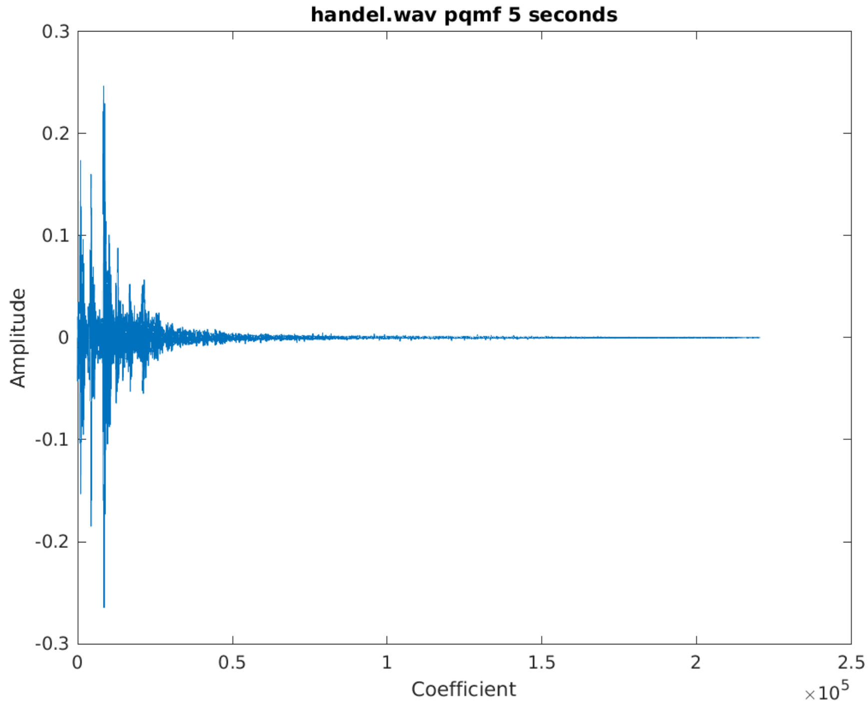


Figure 4: PQFM: Handel 5 seconds

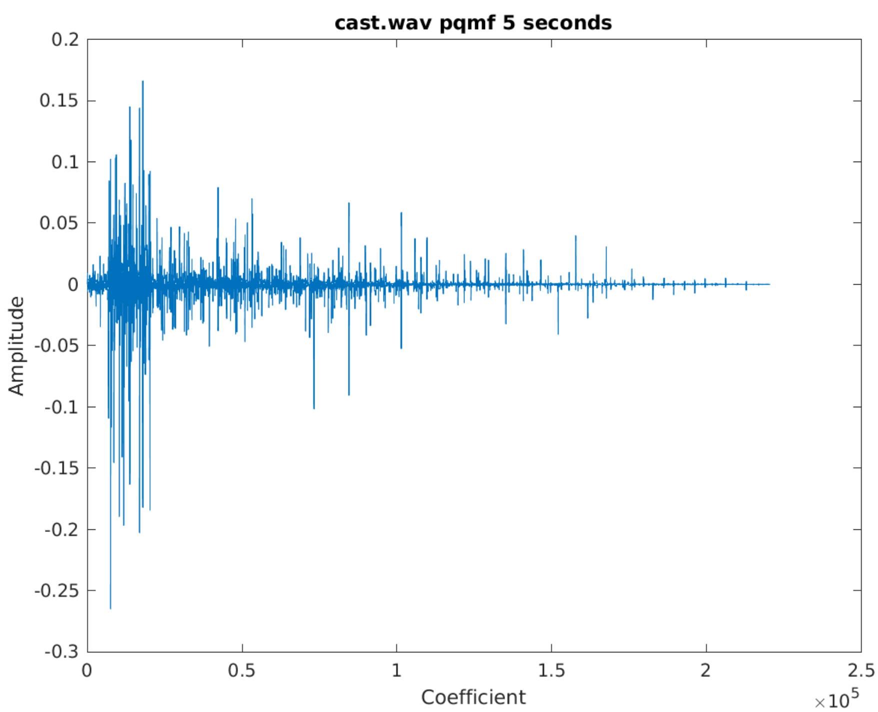


Figure 5: PQFM: Castanets 5 seconds

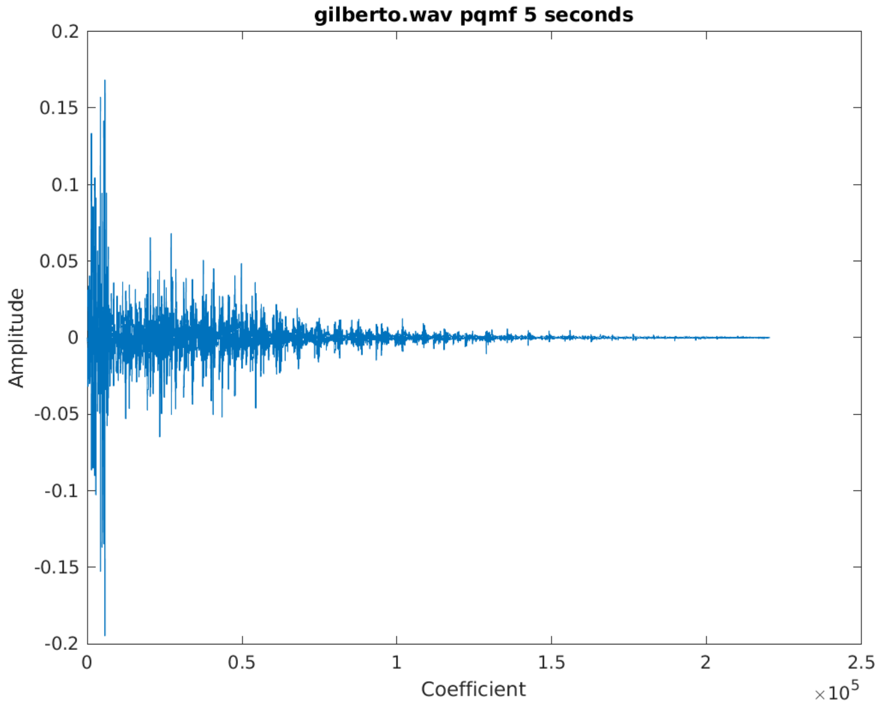


Figure 6: PQFM: Gilberto 5 seconds

The PQFM coefficients essentially show an outline of the frequency spectrums of the various songs.

As seen in Figure 1, the extreme lower end of the PQFM coefficients are low, while they get larger as you approach coefficient number 5000. If one listens to the song, it can be easily heard how the piano lacks a substantial bass, but the higher notes can be heard more easily.

Looking at Figure 2, the lower end of the spectrum is much more substantial, as can be heard by the thumping bass in the song. Since the music is mostly electronic, it lacks many of the stronger overtones that are commonly found on natural classical instruments. As you go even higher in the spectrum, the coefficients do not approach zero as quickly as was the case in Figure 1. Perhaps this is due to the inevitable higher-order harmonics that result from electronic music.

Figures 3a and 3b show the most discernible distinctions. The first sine wave is obviously lower than the second because of the larger concentration around the lower end of the spectrum.

Figure 4 is about the same as Figure 1. There is more activity on the lower end of the spectrum though. This is probably due to the fact that "handel.wav" includes vocals as well as piano and violin.

Figure 5 is quite interesting. While the previous figures had a fairly consistent decline as the coefficients increased, this trend seems fairly haphazard. Something of considerable note though is that this is the first plot to have neighbouring coefficients of unequal magnitude. While the other plots were extremely symmetrical about the 0 amplitude point, Figure 5 is not. Perhaps this is due to the highly percussive nature of the castanets and its atonal sounds.

Gilberto

A Figures

A.1 Distance Matrix

A.1.1 Chroma