Fourier Methods in Musical Analysis

Applied Math 105a Project 2

Due Monday, Jan. 21

You may not have recognized it before, but your ear is a master at computing Fourier coefficients! When two or more notes are played simultaneously on an intrument we call the resulting sound a *chord*. Though the emitted sound is actually a complicated sum of waves, your ear can usually pick out which notes are in the chord with relative ease. This is because different parts of your cochlea are stimulated by different frequencies of vibration of the air. Your ear thus naturally performs a Fourier decomposition, and sends only the transformed waveform to your brain, which allows you to isolate the frequency components in a given sound signal. When many notes are played at once (say 6 or more), your ear may be somewhat overwhelmed and even the most trained musician may have difficulty separating the chord into its various notes. However, the Fourier methods you have learned in class will still apply as they should, and with sufficient effort it is still possible to tell the notes apart. In this project we will play around with these ideas.

1 A little history: the standard chromatic scale

Consonance and dissonance. When two arbitrary frequencies are played together, the result can sound good or bad. In elementary music theory, the source of *dissonance* in such a situation can be traced to "beats", or audible volume fluctuations, that occur in the resulting sound signal. Good sounding, or *consonant* chords, have the feature that the beats are unnoticeable. (Physiologically, beats are noticeable only when the volume fluctuations that compose the sound signal occur at a rate lower than 20 Hz. That is to say, if one were to tap a tabletop at 10 Hz the human ear would perceive the individual taps, but at 30 Hz the ear cannot distinguish the taps and instead interprets the sound as a pitch.) Pythagoras is credited as being the first to recognize the principle of consonance. He observed that two frequencies played together have the least audible beating, and hence a pleasing sound,

when one frequency is a simple rational multiple of the other. That is, the frequency ν and some higher frequency $\alpha\nu$ sound good together when $\alpha = m/n$ for integers m, n with n as small as possible. The simplest nontrivial example, and hence the "best" sounding chord, occurs when $\alpha = \left\{\frac{2}{1}\right\}$. In fact, this chord is *so* pleasing that the two pitches sound almost identical. We call such chords *octaves*, and we say that any frequency of the form $\left\{\ldots, \frac{\nu}{4}, \frac{\nu}{2}, \nu, 2\nu, 4\nu, \ldots\right\}$ has the same *pitch class* as ν.

Scales. We now turn to the concept of a *scale*. In its most general definition, a scale is simply a way of choosing a finite set of frequencies to fill in the gap between a reference frequency ν and its octave 2ν. An example of a scale with 4 pitches might be

$$\nu, \quad 1.3\nu, \quad 1.8\nu, \quad 2\nu$$

Once a scale has been defined between ν and 2ν as above, the scale quickly reapplies to the frequency band between 2ν and 4ν by simply replacing the reference frequency with 2ν. This can be done as well in the other direction; replacing ν with $\nu/2$ would produce the equivalent scale starting the octave below ν. The scale can be continually ascended and descended in this fashion, thereby discretizing the entire range of audible frequencies into a select set of chosen pitches. Any two pitches that differ by a multiple of 2^n occupy the same position in the scale but in different octaves.

There is a rich history behind the forging of the modern chromatic scale in Western music. In short, it was selected so as to offer the most two and three note pairings that are consonant to the ear. The reference frequency is chosen to be "concert A"=440 Hz and the "justly tuned" scale from concert A to its octave contains the following notes, with pitch class determined by a letter (A-G) and possibly a sharp (♯):

440 Hz ×	1	16/15	9/8	6/5	5/4	4/3	7/5	3/2	8/5	5/3	16/9	15/8	2
Pitch class:	A	A♯	B	C	C♯	D	D♯	E	F	F♯	G	G♯	A

By adding or subtracting octaves from the above, we can see that all notes used in modern Western music have frequencies of the form $\alpha \times 2^n \times 440$ Hz, where α is a rational number from the above table and n is any integer, positive or negative, so that 2^n shifts the note to any other octave.

2 Problems

Preliminary Problem: Suppose I have a sound generator that can create signals of the form $A(t) = \sin(n\pi t/T)$ where n can be any integer and T is some fixed constant. What is the smallest possible value for the constant T so that every pitch in the chromatic scale with a frequency of 110 Hz (double low A) or higher can be played by the sound generator?

Series Problem 1: The sound file `majortriad` contains a 4 second sample of one of the most common chords from the modern Western scale. Each note in the chord is played by the sinusoidal sound generator and none of the notes are below double low A or above 14,080 Hz (the A six octaves above concert A; almost the upper extreme of human hearing). Listen to the chord, then use the MATLAB function `wavread` to plot the chord directly as a waveform (amplitude vs time). Use your result from problem 1 and your knowledge of Fourier series to determine the individual frequencies that compose this chord. Identify the pitch class corresponding to each frequency by its letter name. Please include whatever MATLAB code(s) you write to perform the analysis. Using the MATLAB function `wavwrite`, convert each individual frequency into a sound file. As a check to your work, play each of your predicted frequencies and then play the original chord again. You should be able to hear each individual note occuring in the chord.

Series Problem 2: Listen to the 4 second sound file `allexceptone`. This eerie sounding chord contains many pitches spanning several octaves. All pitches are from the chromatic scale and are within the frequency range of 110 Hz - 14,080 Hz. Each pitch is a pure sinusoidal frequency. In this case, your ear won't be of much help to you! Using Fourier series methods again, identify each frequency in this chord, and determine by letter name the only note of the chromatic scale that is not included in this chord.

Transform Problem 1: Now we will employ the ideas developed above to analyze a real chord. However, "real" musical instruments differ from idealized sound generator of the previous problems in two important ways. First, real instruments produce a continous spectrum of frequencies rather than a discrete set, so the proper tool to use is a Fourier transform. Second, real instruments (including the human throat), even when playing a single base pitch ω_B, produce a series of extra pitches ω_H known as *harmonics*, which are integer multiples of ω_B. These arise because they naturally "fit" into imperfect resonators with base frequency ω_B. However, you may not have conciously heard them because they tend to be much quieter than the base pitch.

Listen to the 2 second sound file `beatleschord`, which contains a chord from the Beatles' song "Because." Now load the file into Matlab using the `wavread()` function. The resulting structure will be an 88200×2 matrix contining one soundwave for each channel of a stereo recording (CD's are recorded, by default, at 44100 samples per second). Split each channel into two one-second segments, obtaining four total waveforms to analyze. Compute the fourier transform of each sample (you may use Matlab's built-in `fft()` function), and then find its magnitude, obtaining what is called the *power spectrum* $A(\omega)$. Now average the four results, and plot data points 20-880 on a log-log scale (corresponding to 20-880 Hz). You will see a series of spikes in the amplitude A occuring at various frequencies ω. Some will be "base" pitches that were actually sung by the Beatles, while others will be harmonics of those pitches. Assuming that harmonics are always of smaller magnitude than their generating base pitch, devise and write down an algorithm to isolate the pitches actually sung by the beatles. Using your algorithm, identify the name of each pitch using the table above, and label the appropriate spikes on a printout of the power spectrum.

Transform Problem 2 (Extra Credit): For extra credit, build a spectrum analyzer. Create a 30-second .wav file of your favorite song. Break it into segments that are each, say, 1/10th of a second long (i.e., 4410 samples). You will then perform a fourier transform on each segment, collecing the power spectrum as a function of both frequency and time: $A = A(t, \omega)$. By generating an image from this data set, construct a visual illustration of the distribution of pitches over time. At this point, one might naturally wonder whether this strategy could be used to automatically transcribe any waveform into musical notation! Based on your knowledge of of Fourier series and transforms, and your experiences so far on this project, discuss any difficulties that you can see to implementing such a system.