Middle East Technical University
Electrical-Electronics Engineering Department

EE301 Term Project (2016-2017 Fall Semester)

Title: Analysis/synthesis and recognition of the sound signals of various musical instruments and human voice using Fourier Series Representation

Background:

Sound signals have a vitally important position in our lives. As a human being with auditory sensors, we are capable of hearing and listening to various kind of sounds. We not only hear or listen but also arrange different sound notes in a systematical way to compose songs, which we play with musical instruments. Furthermore, we also use our voice to sing a song. In this project, we will analyze the sound signals of various musical instruments and human voice in frequency domain. We will see that Fourier series representation has a core position in the analysis of such sound signals.

The Fourier series representation is defined for periodic signals which have a certain frequency. This frequency is also referred to as “fundamental frequency” as you have seen in the EE-301 course. As you have also learnt in EE-301, such periodic signals can be decomposed into components whose frequencies are integer multiples of the fundamental frequency. These frequencies are also referred to as “harmonics”.

Musical instruments that have a string (which are also named as “string instruments”) typically generate a periodic signal that consists of those “harmonics”. For instance, when a certain note is played by picking a string of a guitar, in addition to the frequency of the note that is played (for example the “la” [‘A’ in English] note has a frequency of 440 Hz), the generated sound signal have components with frequencies at 2x440 Hz, 3x440 Hz, …, etc. This reason is related to the harmonic oscillations of the string of the guitar. Such oscillations are presented in Fig.1.

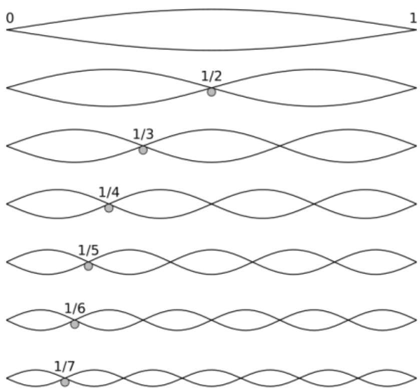

Fig.1: Fundamental and harmonic oscillations of a string

In Fig.1, the uppermost string oscillation has the frequency of the note to be played (for instance, it is 440 Hz for the "la" note). In addition to the oscillation in the fundamental frequency, there are other oscillations with a frequency of integer multiple of the fundamental frequency as can be observed in Fig.1. Such phenomena also apply to the string instrument violin. Even human voice have harmonics, since our vocal cords are also a sort of string. Therefore, the sound signals with such structure are periodic and they have a Fourier series representation. What makes the sound (or "timbre") of the various musical instruments or the human voice different (even if they have the same fundamental frequency, or the note that is played by various instruments or sung by a human are the same) is the weights of the harmonic contents of the sounds they generate. The harmonic content weights correspond to the Fourier series coefficients. Remember that a periodic signal $s(t)$ with a certain fundamental frequency, w_0, can be decomposed as

$$s(t) = \sum_{k=-\infty}^{\infty} a_k e^{jkw_0 t}. \tag{1}$$

Even if the played note has a certain fundamental frequency, the values of a_k will differentiate the kind of the musical instrument or the voice of the human singing the note.

To start with, we will try to find the values of a_k for various musical instruments and a human voice.

Important Note: In this document, the frequency variable f ,whose units is Hz, is used instead of the frequency variable w, which is equal to $2\pi f$ and whose units is in radians/sec.

PHASE 1

Analysis of the sound signals

1) To represent a signal $s(t)$ as in (1), $s(t)$ must be an infinite duration signal. We will try to find a_k's from a finite duration version of $s(t)$, namely $s_w(t)$. Find the continuous time Fourier transform (CTFT) of $s_w(t)$, namely $S_w(f)$, in terms of the Fourier transform of the infinite duration signal $s(t)$, which is denoted as $S(f)$. You may assume that $s_w(t)$ is equal to $s(t)$ times a rectangular window of length L. Without loss of generality, you may assume that the rectangular window is centered at $t = 0$.

2) Express $S_w(f)$ in terms of the Fourier Series coefficients of $s(t)$, namely a_k.

3) Explain how you can obtain a_k's by sampling $S_w(f)$ by assuming that the fundamental frequency w_0 is an integer multiple of $2\pi/L$. What must be the frequency interval between the samples of $S_w(f)$?

4) Now, we will try to obtain $S_w(f)$ for a violin, flute and a human voice by using previously recorded .wav files. You can find those files in "Project, Part-1 Support Files" with the file names "violin_A4.wav", "flute_A4.wav" and "voice_A4.wav". In each .wav file, the "la" note is played with a violin, a flute or sung by a human. You can play those files to listen to their content.

   These .wav files contain samples of the continuous time sound signals, namely $s_w(nTs)$, T_s being the sampling period. Therefore, they are discrete time signals rather than being continuous. As you will learn later in the semester, if the sampling rate $F_s = 1/T_s$ is larger than the bandwidth[1] of $s_w(t)$, one can obtain $S_w(f)$ for $-\frac{F_s}{2} < f < +\frac{F_s}{2}$ based on $s_w(nTs)$. For the sound signals provided in the support files, the sampling rate is larger than the signal bandwidth. Therefore, we can find the CTFT, $S_w(f)$ through the samples $s_w(nT_s)$. Then we can obtain the Fourier series coefficients by inspecting $S_w(f)$ at some values of f. The concepts mentioned in this paragraph may not be clear to you. However, it will suffice that we can somehow find the samples of the CTFT of a signal in MATLAB and the Fourier series coefficients.

You are supposed to use the m-file "analyze_sound.m" in the support files. This m-file is provided as an incomplete m-file. You are supposed to write your own code segments in the m-file following the instructions specified in the m-file as comments. When you complete the m-file, it is supposed to be able to plot the samples of the CTFT for $-\frac{F_s}{2} < f < +\frac{F_s}{2}$, since F_s is larger than the signal bandwidth for an audio signal. The code segments related to CTFT computation are provided to you in the "analyze_sound.m" file.

Plot the time domain signals, $s_w(t)$, for the specified time intervals (1.8<t<1.84 for violin signal, 0.1<t<0.12 for flute and 0.17<t<0.21 seconds for voice signal). Are they periodic? Comment.

Plot also the magnitude and the phase of the samples of CTFT for $-\frac{F_s}{2} < f < +\frac{F_s}{2}$ of $s_w(t)$ for the three different audio signals (violin, flute and voice signals). Read the values of the peaks in the magnitude plot of CTFT at the harmonics of the fundamental frequency (which is 440 Hz for our case). Should these values be equal to a constant multiple of the magnitude of the Fourier series coefficients for the audio signals? Why? Explain the reason by considering your answer to Q3. Also note that the harmonics are not exactly at the multiples of 440 Hz. This is owing to the fact that there are some small tuning errors, thus the musical instrument or human voice does not exactly play or sing the note "la". For example, the first fundamental frequency of the voice signal is at about 463 Hz.

5) List the Fourier series coefficients for the 3 audio signals (violin, flute, voice) in a table. Normalize the values of the coefficients such that the one with the maximum magnitude has a magnitude of 1. You may list the first 10 coefficients (the one corresponding to the fundamental frequency and the higher harmonics) for the flute, 19 coefficients for the violin and 9 coefficients for the singer. Note that you are not required to find the Fourier series coefficient a_0 since the DC level for sound signals is equal to zero in general. You should provide the values of the Fourier coefficients in polar form, that is, you should provide the magnitude and the phase (in radians) of the Fourier series coefficients a_k.

6) Save the m-file that you have created with the file name "analyze_sound_Group_XX.m", where XX is your group number. For example, if your HW group is Group 5, you should save the m-file with the file name "analyze_sound_Group_05.m". Use "publish" command in MATLAB to publish your code in Microsoft Word format. Submit your m-file and the .doc file created by using "publish" command along with the other documents that you submit for Project Part-1 Submission.

[1] A windowed signal cannot be strictly band-limited. Yet, we can define an effective bandwidth which contains almost all the energy and this would be sufficient for the argument here.

PHASE 2

In this phase of the project, in case you have made a mistake in finding the Fourier Series (FS) coefficients in phase 1 Q-5, please check “Fs_coefficients.m” in which the correct values of the FS coefficients are provided. Please correct the values that you have found in Q5 if necessary.

Synthesis of the sound signals

7) Now using the Fourier series coefficients that you have found in Q5, synthesize s(t) using (1) for 3 different sound signals in MATLAB. In case you may not have been able to find the coefficients or found them wrongly, their correct values are provided to you in the m-file “Fs_coefficients.m”. Adjust the duration of the signals you synthesize to be 5 seconds. Listen to the signals you generated, and comment on their similarity to the original sound signals of violin, flute or singer. Plot the time domain waveforms for $1.36<t<1.415$ for the three different sound signals.

   Hint: Since there is even symmetry for the Fourier series coefficients (since the sound signal is real), you can express the sum in (1) as a sum of cosine signals weighted by Fourier series coefficients. The discrete version of the continuous cosine wave $\cos(2\pi f t)$ can be assigned to a vector y as

   $$y = \cos(2\pi f n T_s), \tag{2}$$

   where $T_s = 1/F_s$ is the sampling rate. You can select the sampling rate to be 44100 Hz. You may use “sound” command to play the sound that you synthesize. After you synthesize your sound signal as described and before you play it, you should divide the signal by its maximum value and multiply by 0.5 so that the sound signal that you synthesize will not be clipped by the amplifiers of your computer before being played.

Recognition of sound signal type

8) In this part, we will examine sound recordings from some other violin, flute or singer, yet the played/sung note being the “la” note. For example, the singer whose sound will be examined in this part will be a different person than the singer whose sound is analyzed in the previous parts. The function that you are supposed to write in this part should be able to distinguish whether the input sound signal is a violin, flute or a voice signal. The sound signals that have been obtained from different violin, flute and singers from the ones in the previous parts can be found in the support files with the file names “another_violin_A4.wav”, “another_flute_A4.wav” and “another_signer_A4.wav”.

   The MATLAB function that you will use to complete this question is also provided to you with the file name “recognize_sound.m” in the support files. Again, this m-file is an incomplete one. You should follow the instructions that have been provided to you as comments in the function script to complete the function script so that it will determine the type of the input sound signal to the function.

   To determine the type of the input sound signal, you should compare the Fourier series coefficients that you have found in Q5 with the Fourier series coefficients that you obtain for the new sound signals in this part. If we denote the Fourier series coefficients found for the new sound signals in this part as b_k, and the Fourier series coefficients that you have found in Q5 for violin, flute and voice signals as a_k^{violin}, a_k^{flute} and a_k^{voice}, the error (difference) signals for violin, flute and voice signals can be defined as

$$e_k^{violin} = a_k^{violin} - b_k, \qquad k = -19, \dots, 19 \tag{3}$$

$$e_k^{flute} = a_k^{flute} - b_k, \qquad k = -10, \dots, 10 \tag{4}$$

$$e_k^{voice} = a_k^{voice} - b_k, \qquad k = -9, \dots, 9 \tag{5}$$

For simplicity, just consider the magnitude of the Fourier series coefficients and calculate the error signals as follows:

$$e_k^{violin} = |a_k^{violin}| - |b_k|, \qquad k = -19, \dots, 19 \tag{6}$$

$$e_k^{flute} = |a_k^{flute}| - |b_k|, \qquad k = -10, \dots, 10 \tag{7}$$

$$e_k^{voice} = |a_k^{voice}| - |b_k|, \qquad k = -9, \dots, 9 \tag{8}$$

Then the mean of the squared error signals (which can also be referred to as mean squared error, MSE) can be calculated as

$$MSE^{violin} = \frac{1}{2K^{violin}+1} \sum_{k=-K^{violin}}^{K^{violin}} |e_k^{violin}|^2, MSE^{flute} = \frac{1}{2K^{flute}+1} \sum_{k=-K^{flute}}^{K^{flute}} |e_k^{flute}|^2,$$

$$MSE^{voice} = \frac{1}{2K^{voice}+1} \sum_{k=-K^{voice}}^{K^{voice}} |e_k^{voice}|^2 \tag{9}$$

where $K^{violin} = 19$, $K^{flute} = 10$, $K^{voice} = 9$, from Q5. You may take the DC coefficients for any signal you examine in this part to be zero since the sound signals do not have any DC component. The sound signal type that gives the minimum MSE is the one that you should declare the type of the input sound signal to be. For instance, if you have found that MSE^{violin} is less than MSE^{flute} and MSE^{voice}, you should declare the input signal type to be a violin signal. In this way, we make a sort of minimum MSE (MMSE) detection for the input signal type. MMSE detection and estimation is a commonly used simple technique in telecommunications and signal processing applications.

Find the MSE values (MSE^{violin}, MSE^{flute} and MSE^{voice}) for each of the sound signals “another_violin_A4.wav”, “another_flute_A4.wav” and “another_signer_A4.wav” (This means that you have to find a total of 9 MSE values). Which MSE value is minimum for each sound signal? Why? Comment.

9) Save the MATLAB function script that you have completed in Q8 by following the instructions in the function script with the file name “recognize_sound_Group_XX.m”, where Group_XX is your group number. For instance, if your HW group is Group 05, the function name that you should use should be “recognize_sound_Group_05.m”. You should also change the first line of “recognize_sound_Group_XX.m” script as “function out=recognize_sound_Group_XX(audio_signal_file_name)”. If you do not obey the format (also to the case sensitivity) of the file name, you may not get any point from this part. Submit this m-file as a separate file in addition to the file(s) that you submit for Project-Phase-2 submission. In this part, we will give various sound signal file names randomly as an input to the MATLAB function script that you submit and check whether it generates correct results. For example, when we give ‘violin_A4.wav’ or ‘another_flute_A4.wav’ as an input file name string to your function, it should give an output value of 1 and 2, respectively. Which input signal should give which output value is specified in the instructions in “recognize_sound.m”.

Sampling rate change

10) Read the audio signals “singer_A4.wav”, “violin_A4.wav”, and “flute_A4.wav” using audioread command. Assign the audio signal and sampling rate output arguments of the audioread command to “audio_signal” and “Fs”. Use the following code segment to decrease the sampling rate of the audio signals by 10 and listen to the downsampled audio signal:

    ```
    audio_signal_downsampled= audio_signal(1:10:end); % Take 1 of every 10 samples
    sound(audio_signal_downsampled,Fs/10);
    ```

    Listen to the resulting audio signal. Comment on the changes compared to the original audio signals. For your comments, you may assume that the singer voice is bandlimited to 4 kHz and the sound of the flute and violin are bandlimited to 10 kHz.

11) Repeat part 10 only for “singer_A4.vaw” and decrease the sampling rate of the audio signal by 2 (instead of 10).