Lab 2

Project Goals

This lab aims to demonstrate an application of Fourier and spectral analysis in audio processing.

Team-working and timeline

All labs should be completed in groups of two. A single lab report is satisfactory for each group. You are encouraged to discuss the assignments with the TAs or your classmates, but the code and report must be your own solution to the lab. Duplication of code and/or data will result in a grade of zero for all parties involved. The deadline to hand in the report can be found in the syllabus. Late Lab reports are accepted, although each week %10 of the maximum points will be deducted. (For instance, if the lab is 150 points, 15 points will be deducted).

License

Dream On by Aerosmith[1] is acquired by the instructor through Amazon Music. The selected part of the song is extracted using QuickTime Player. The file DreamOn_trimmed.m4a is restricted to individuals enrolled in the course ECE 2312_C25 and *must not be shared, distributed, or disclosed to any unauthorized persons outside of this course.*

Report

Final reports (PDF format only), including all the source code, are to be submitted via Canvas.

- A narrative of the process taken during this experiment and the experiences encountered by the student. Photos, diagrams, and other visuals are highly encouraged.
- Responses to all questions indicated in the experiment handout. Please make sure that the responses are of sufficient detail.
- A summary that contains all lessons learned from this hands-on experiment.
- Source code (as an appendix). The codes should have line-by-line comments to explain the computation.
- You are expected to sign off on each lab and answer questions that the TAs will ask about your code.

[1] Aerosmith is a rock band from the United States, founded in Boston in 1970.

Part 1

Speech-music discrimination is a fundamental task in audio analytics, essential for various applications like automatic speech recognition and radio broadcast monitoring. It involves segmenting audio streams and categorizing each segment as either speech or music. In the context of automatic speech recognition (ASR) for broadcast news, this distinction plays a crucial role. One of the early examples was a real-time speech-music discriminator designed for radio receivers to enable automatic monitoring of FM radio content.

A human listener can effortlessly distinguish between speech and music by hearing just a few seconds of an audio signal. Various systems have been developed for the automatic classification of speech and music signals. These systems include ones relying on digital signal processing and AI-enabled discriminators. Here, we start with the first category.

The first approach that we try in the attached code is to simply apply bandpass filters. The core idea is that if we know the voice band, we try to discriminate the vocalist's voice from the background music. Voiceband is the standard frequency range transmitted by traditional telephones. In this case, the typical human voice frequency range is between 300 Hz and 3400 Hz, meaning the most important frequencies for understanding speech fall within this band.

Another option is to consider the typical frequency range of the human voice, which spans from 80 to 250 Hz.

The third option is determining the vocal range of the vocalist singing a specific song.

<u>Questions and steps to take</u>

The code "Lab2.m" is written to discriminate the vocals and music in the song "Dream On" by Aerosmith. Steven Tyler's vocal range in this song is F3-G#5 (174.61-830.61 Hz), and he hits an E6 note (1318.51 Hz) during the song's climax. This demonstrates his wide vocal range and has earned him recognition as one of rock's most distinctive voices. Run the code for these three options: Human-being voiceband, voice frequency range, and Steven Tyler's octave range. Listen to sound.m4a, sound_music.m4a, and sound_voice.m4a.

"sound.m4a" is a recreation of the original part of the song by applying fft and ifft. Why is that noisy? Performing a Fast Fourier Transform (FFT) followed by an Inverse Fast Fourier Transform (IFFT) on a sound signal is designed to perfectly reconstruct the original signal by converting it to the frequency domain and back to the time domain. In theory, this process does not introduce noise. However, in practice, minor rounding errors may occur due to the limitations of computer arithmetic, which can result in a slightly noisy output.

- Which one sounds better? (None of them is perfect! Try to qualitatively assess how well the voice and music are differentiated from one another.)

- Explain what the code does in your own language. What do the arrays "music" and "vocal" represent?

- Why doesn't it make much sense to use a bandpass filter to extract voice from the song? (Hint: is the vocalist the only source creating a signal in the range that the bandpass filter passes?)

Part 2

The version of "Dream On" by Aerosmith that we consider in this lab was recorded in stereo.

<u>Questions and steps to take</u>

- Research to find out how left and right channels are allocated to vocals and music. Does it make a significant difference if we change the channels in lines 45 and 46 (i.e., considering channel 2 for instruments and channel 1 for vocals or vice versa)?

- "Stereo center-channel subtraction method" refers to a technique in audio processing where you can isolate the center channel information from a stereo audio track by subtracting the left and right channels from each other, effectively removing the stereo elements and leaving only the content that is identical in both channels (the center). Run the code and listen to the sound_music_diff.m4a. What differences can you hear when qualitatively comparing this track with sound_music.m4a?

- What does the array "music_diff" represent?

Part 3

The spectral centroid (SC) represents the center of mass of a signal's spectrum and is commonly used in digital signal processing and music analysis. It is derived from the Fourier transform of a signal's frequency and amplitude, serving as an amplitude-weighted average of multiple frequency components. Calculated by averaging the centroid frequency across several temporal windows, SC provides insight into the balance between high and low frequencies in a signal. SC's implementation is embedded in Matlab (https://www.mathworks.com/help/audio/ref/spectralcentroid.html).

Questions and steps to take

- Run the code and check SC for music and vocals. In Figure 2, what differences can you see when comparing music_centroid_diff and music_centroid?

- In Figure 3, a histogram is drawn to compare music_centroid_diff and music_centroid as well as their relative characteristics with respect to the song containing the vocal. Explain what a histogram is (generally, not specific to this code). What is your conclusion after checking the histograms in this code?

Part 4

Matlab's Signal Toolbox™ and Audio Toolbox™ offer a comprehensive set of functions for analyzing the shape, or timbre, of audio signals. SC is one of those functions. Another one is the spectral rolloff point, which has been utilized in various applications, including speech-music classification, music genre classification, acoustic scene recognition, and music mood analysis.
The spectral rolloff point is the frequency at which a specified percentage of an audio signal's total energy is concentrated, serving as a measure of the signal's bandwidth and frequency content. It is typically calculated through the following steps:
1. **Specify the energy percentage** – Common values are 85% or 95%.
2. **Determine the band edges or bins** – Identify the range over which the spectral rolloff point will be computed.
3. **Locate the frequency bin** – Find the frequency at which the specified percentage of energy is contained.

The Matlab's command is available here:
https://www.mathworks.com/help/audio/ref/spectralrolloffpoint.html

Questions and steps to take

- Check the link to the Matlab implementation of the spectral rolloff point. What is the default value for the threshold of the rolloff point?

- How different are the rolloff points for the arrays "music" and "music_diff"?

Before going forward, save all figures generated by the code.

Part 5

The goal of this part is to compare your version of the song with Aerosmith's! As you can see in the previous steps, digital signal processing could help to extract the music from

the song. However, due to quantization errors and minor rounding errors, the quality of the extracted music might be affected.
AI offers powerful tools to extract music and vocals from a song. Listen to the music and vocals extracted using the tool Vocal Remover (https://vocalremover.org). Hear the difference?

Questions and steps to take

- Play the music extracted by AI and sing along as you are the vocalist (karaoke). Record your song using Matlab's audiorecorder (https://www.mathworks.com/help/matlab/ref/audiorecorder.html), recordblocking (https://www.mathworks.com/help/matlab/ref/audiorecorder.recordblocking.html), and getaudiodata (https://www.mathworks.com/help/matlab/ref/audiorecorder.getaudiodata.html)

  To record a stereo song, use this setting:
  nBits should be set to 16 (the standard bit depth for CDs).
  NumChannels should be set to 2 (since we are recording stereo).
  SampleRate should be set to 44100.

  To use recordblocking, set the length to the length of "music_AI.mp3"

- The name of your recorded song should be "mysong.m4a". Now, repeat all parts (parts 1-4, including 4) as described above using your recorded song. This means that you give "mysong.m4a" to the code instead of "DreamOn_trimmed.m4a". Generate all figures. Compare the figures with ones that you have saved when processing "DreamOn_trimmed.m4a" (parts 1-4, including 4). What is your conclusion? (qualitative in the sense that you compare Steven Tyler's vocal range with yours)

  Important: Since we do not know your octave range, replace lines#41-42 with the following lines.

```
lower_threshold = 80; %Lowesr freq. generated by human-being
upper_threshold = 1318.51; %We hope that you can hit E6 (1318.51 Hz), too.
```