COMP.SGN.120 Introduction to Audio Processing

Exercise 6
Week 50

In this exercise, you will familiarize yourself with the common features used in music processing like constant-Q spectrogram and chromagram. Additionally, you will also get to know the common music processing techniques such as onset detection. There are two questions with 1 point for each, summing up to 2 points. Bonus question is optional, no grading, to test your skills further. **The submission should consist of a Jupyter notebook with your observations and the python code**.

Problem 1 : Constant-Q spectrogram and chromagram (1 point)

Load the given brahms_hungarian_dance_5_short.wav to get the audio signal and its sampling rate.
Hint: *librosa.load ('*brahms_hungarian_dance_5_short*.wav')*

- Calculate the spectrogram of it. Hint: *librosa.core.stft (audio, sr = sr)*
- Calculate the constant-Q spectrogram of it. Hint: *librosa.core.cqt (audio, sr = sr)*
- Calculate the chromagram of it. Hint: *librosa.feature.chroma_stft (audio, sr)*
- Plot and observe and report differences between the spectrogram, constant-Q spectrogram and chromagram. Hint: *librosa.display.specshow*

Now test some other music samples; repeat the process above and report what you have noticed.

Problem 2 : Onset detection (1 point)

Part a) Onset detection

Onset refers to the beginning of a musical note, in which the amplitude rises from zero to an initial peak. Let us start with the given classic_rock_beat.wav music first and try to find the onsets.

1. Load the audio file. (*librosa.load)*
2. Compute spectral novelty function using librosa library. (*librosa.onset.onset_strength*)
3. Pick peaks to detect the frame indexes of the onsets. (*librosa.util.peak_pick*)
4. Convert the frame indexes into time indexes in unit of seconds. (*librosa.frames_to_time*)
5. Plot the onsets on top of the time domain signal and report your observations. Hint:(*librosa.display.waveplot*), (*plt.vlines(onset_times)*)
6. Plot the onsets on top of the spectrogram and report your observations. Hint:(*librosa.display.specshow*), (*plt.vlines(onset_times)*)
7. Adjust the *librosa.util.peak_pick* parameters and observe how they affect the detected onsets.

Part b) Add clicks to the onsets

At this point you have detected the onsets. Now let us add clicks at the detected onset frames.

1. Create a signal with the same length as the analyzed music example, consisting of clicks at specified locations. Pass the detected onset frames, sampling rate and the length of the music to the function *librosa.clicks(frames=onset_frames, sr=sr, length=len(x))*
2. Add the two signals together sample-wise (original music example and clicks signal) and play. Hint:*music + clicks, sounddevice.play*
3. Instead of adding two signals sample-wise, you can also stack two signals to create stereo signal and listen to it. Hint: *np.vstack([music, clicks]), sounddevice.play*

Now test some other music samples; repeat the same process above and report what you have noticed.

Bonus : Implement your own spectral novelty function

In Problem 2, part a), step 2, you have computed spectral novelty function using librosa library. (*librosa.onset.onset_strength*). Now let us implement it by our own!

1. Calculate the STFT of the signal to get the spectrogram $X(n,k)$, where n is the time frame index and k is the frequency index . Hint: *librosa.core.stft*
2. Convert the amplitude to dB scale. Hint: librosa.amplitude_to_db(stft,ref=np.max)
3. Apply log-compression to magnitude spectrogram: $Y(n,k) = \log(1 + \gamma|X(n,k)|)$, where $X(n,k)$ is the spectrogram obtained from the step 2, and when $\gamma = 1$, weak components become visible.
4. Calculate the spectral novelty function given below.
   $\Delta(n) = \sum_{k=0}^{K-1}|Y(n+1,k) - Y(n,k)|$
   Hint: np.sum(np.diff(log_compression), axis=0)

Once you get your own spectral novelty function, repeat the same following steps (step 3, 4, 5, 6) as in question 2, part a). Observe the differences and report it.