Assignment 2 & 3
Music, Mind & Technology

Question A

1. Harmonics

A. Create the Baby Shark melody using sine tones with the first 10 harmonics (link: timestamps 0:24 to 0:33). The notes needed for the rhyme are: D (293.5 Hz), E (329.5 Hz), and G (392 Hz)
B. Create two versions of the melody, one with the odd harmonics (3, 5, and 7) and another with the even harmonics (2, 4, and 6)
C. What are the perceptual differences between all 3 versions

Answer to above parts :

When the Part A, B, C all three versions are generated we can make a melody comparison in part A and part B where part A has a deeper melody, where as in part C is also lighter than part A

2. Virtual Pitch

A. Use the melody created in part 1a.
B. Remove the fundamentals to create the same melody with virtual pitch
C. Remove the first harmonics to form another melody
D. What is the perceptual difference between all 3 melodies

Answer to above parts :

When fundamentals in part B are removed the melody strength of part B will be higher than usual , whereas in part C it is becoming inaudible and fundamentals are not heard so it has very low audible frequency implies it is virtual pitch.

Question B

You are allowed to use either Librosa or MIRToolbox for the following questions:

1. Rhythm & Meter

Aims: To get familiar with tempo estimation from audio using the MIRtoolbox. To assess the performance of the tempo estimation method.

Part 1.

- Read the sample music ‘michael_jackson.mp3’, ‘dream_theater.mp3’, ‘mozart.mp3’,‘queen.mp3’ and ‘taylor_swift.mp3’ using the miraudio command. Estimate the tempos of the files in the folder. For this, go to the URL **http://www.metronomeonline.com/** where you can find an online metronome.
- Play each excerpt using the mirplay command Eg: mirplay(‘queen.mp3’);
- Adjust the speed of the metronome to match with the tempo of the music. Write down each tempo. If the tempo varies within the excerpt, try to estimate the range of tempi within the excerpt.

Part 2.

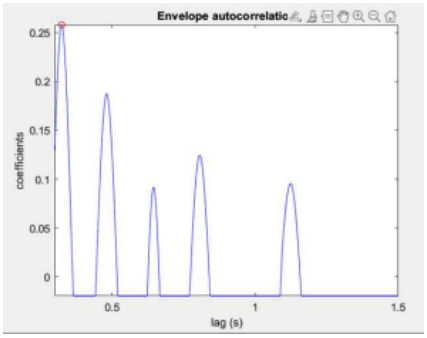
- Use the function mirtempo to computationally estimate the tempi of the excerpts.
- Compare the computational estimates with your perceptual estimates.
- To what extent do they agree? What are the typical discrepancies? Can you explain the reason for them?

Answers for both Part 1 & Part 2 :

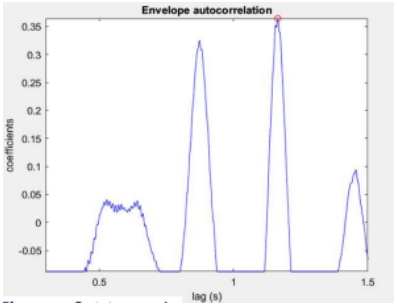
Sample music	Percieved bpm
michael_jackson.mp3	82 bpm
dream_theatre.mp3	130-170 bpm
mozart.mp3	112-140 bpm
queen.mp3	66 bpm
taylor_swift.mp3	40 bpm

Sample music	Calculated bpm
michael_jackson.mp3	97.5 bpm
dream_theatre.mp3	185.3 bpm
mozart.mp3	140.8 bpm
queen.mp3	150.3 bpm
taylor_swift.mp3	40 bpm

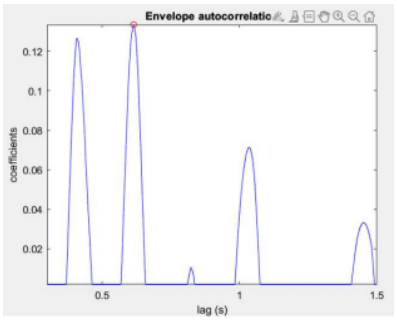
Plots for the wav files :



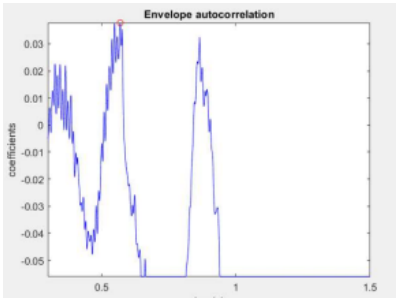
- michael_jackson.mp3



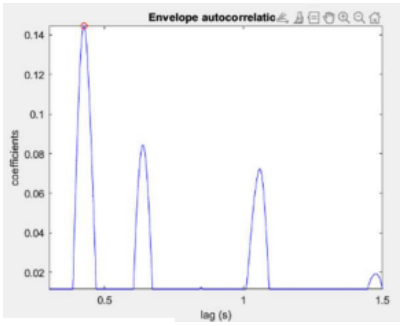
- mozart.mp3



- dream_theatre.mp3



- taylor_swift.mp3



- queen.mp3

Analysis :

All the other tracks have a pretty steady tempo, except for 'queen.wav',. I haven't noticed any clear reason for the differences, but it seems like the tempo I feel is a bit slower than what's

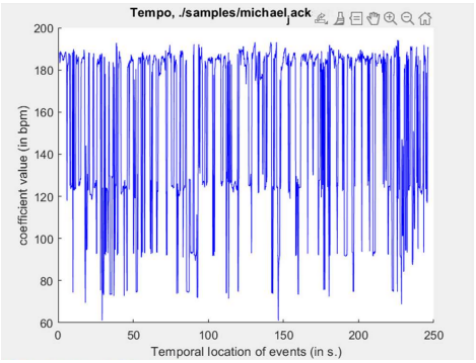
calculated. I think this small difference might be because everyone taps along to music differently. As for 'queen.wav', I think its sound waves have some tiny ups and downs that we might not notice, but they still affect the tempo, which could explain the gap.

Part 3.

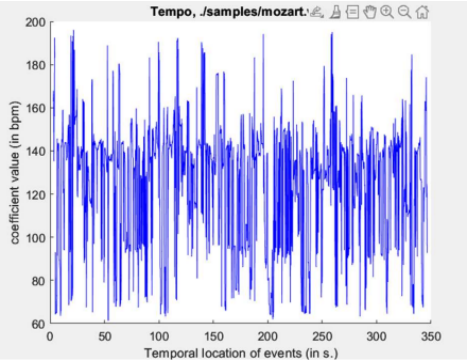
- Find out which samples have variable tempi. Use frame-based tempo analysis to estimate the time course of development tempo.
- To this end, decompose the samples with variable tempi using the 'Frame' option of mirtempo (see help mirtempo; use a frame length of at least 2 s).
- What are the ranges of variation of tempi? Do they correspond to your estimates?

Answer for Part-3 :

Plots :



michael_jackson.mp3



mozart.mp3

Analysis :

For 'michael.wav', the tempo ranges from 80 to 185, but most of it falls between 120 and 180, which matches what I hear. As for 'Mozart.wav', the tempo varies between 60 and 200, with the main part being between 90 and 140, which is pretty close to what I thought I heard.

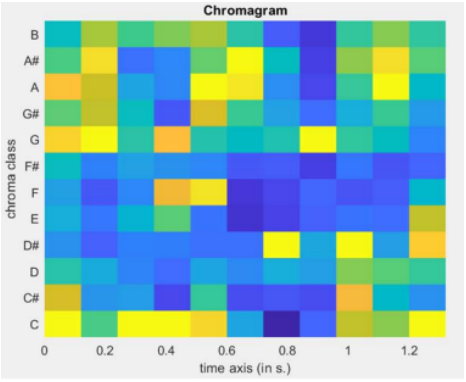
2. Repetition in Music

Assume x = Rollnumber % 7

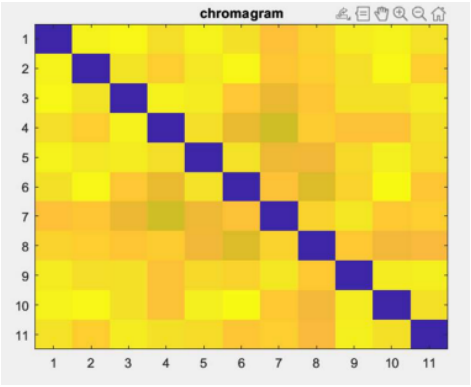
- First, analyze the song ‘x.wav’ from the data folder.
- Extract the chromagram vectors of this excerpt frame by frame, using the mirchromagram operator and the ‘Frame’ keyword.
- Compute the similarity matrix of the sequence of chromagram, using the mirsimatrix operator.
- Look at the results in the figure representing the similarity matrix. Try to understand the link between the lines in the similarity matrix and the checkered rectangles.
- Try to see the impact of any change of the different parameters of the model in the final results. You can change for instance:
- the audio feature (chromagram, mfcc, and spectrum)
- the frame length and hop factor,

- Which of the features best represents your notion of perceptual segmentation and repetition?
- Try the same kind of analysis on all files provided in the samples folders.

File to be analyzed is 5.wav as per x,
Generating chromogram and similarity matrix using the mixmatrix



- Chromogram

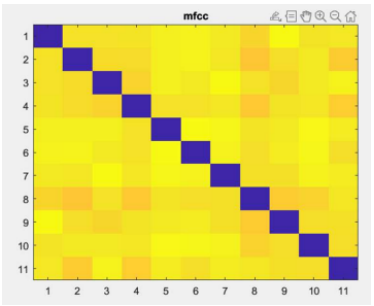


- Similarity

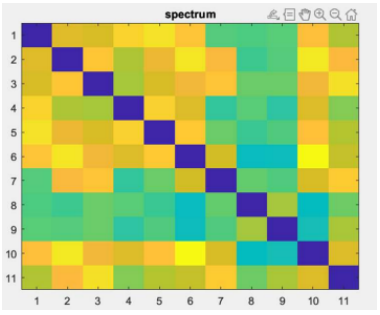
The similarity matrix for a music file helps us see how alike different parts of the music are. The checkered rectangles show where nearby sections sound similar, probably because they're close together in time. And the lines in the matrix mark the spots where the music shifts to a new section or phrase. Basically, this matrix lets us spot patterns in the music and figure out how its different parts fit together.

Parameter Change :

a) Audio feature

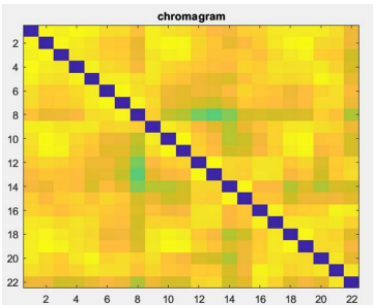
i) 

- mfcc

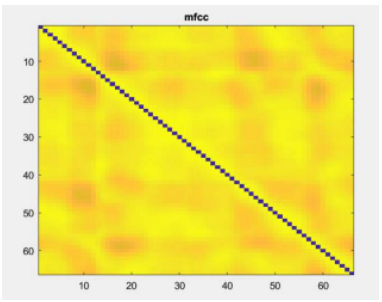
ii) 

- spectrum

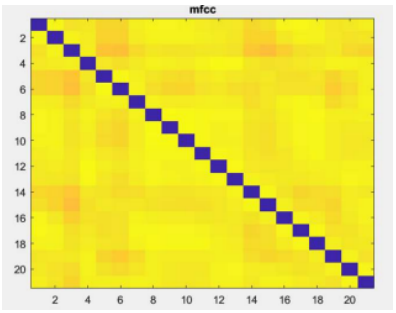
b) Frame length and hop factor

i) 

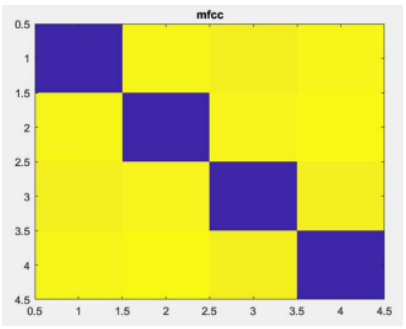
- l = 0.2, h = 0.3

ii) 

- l = 0.2, h= 0.01

iii) 

- l =0.3, h=0.2

iv) 

- l = 0.5, h = 0.6

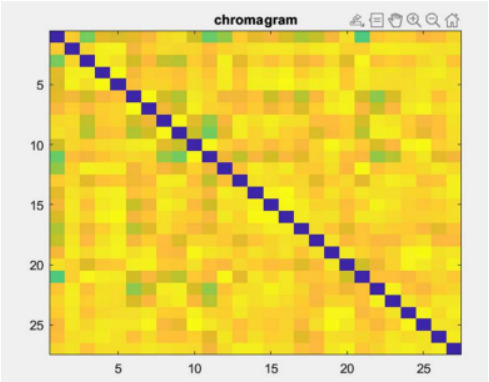
Part 4 :

- Changing settings in the analysis tools can really change what we find. For example, using different audio features like chromagram, MFCC, or spectrum will give us different views of the music. This might affect how accurate our similarity matrix is. Also, tweaking how long each frame is and how much they overlap can change things. Longer frames show the big

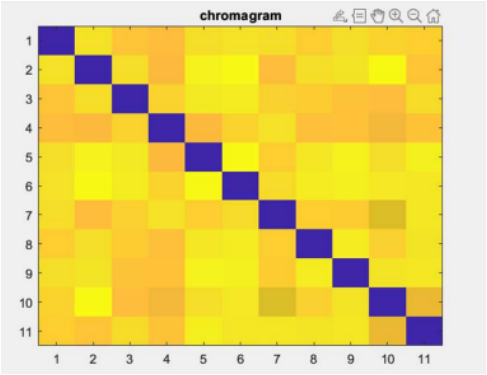
picture of the music, while shorter ones focus more on the little details. The amount they overlap also makes a difference, with more overlap making things smoother. To find the best settings for what we're trying to do, we need to try out different combinations and see what works best. We can do this by testing things out or using methods like grid search or random search to explore all the possibilities.

- Music is full of patterns and repeats, but figuring them out can be tricky. One way we can do it is by using something called a chromagram. It's like a map of the pitches in the music, showing which notes are being played. This helps us spot melodies and chords, which are important for finding where the music repeats or changes. The chromagram is really good at capturing these big musical structures, like chords and keys, which help us understand how the music is organized. While other tools like MFCC and spectrum can also help, the chromagram is often the go-to choice because it's so good at spotting these patterns.

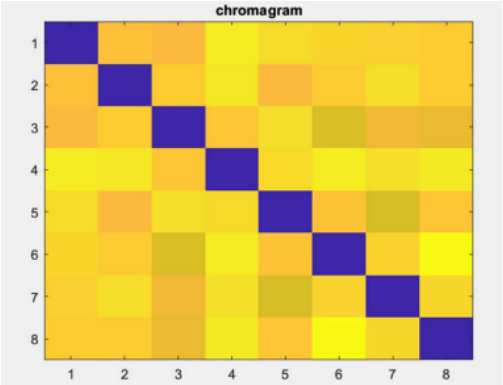
Analyzing 3.wav



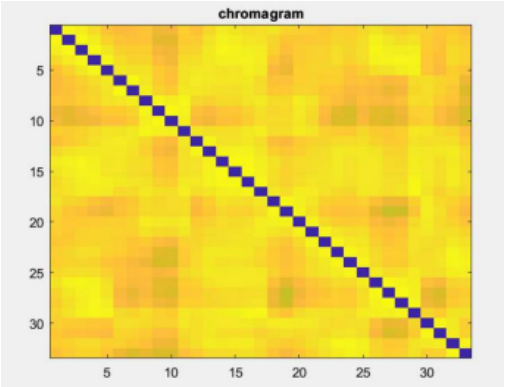
- l=0.09 and h=0.6



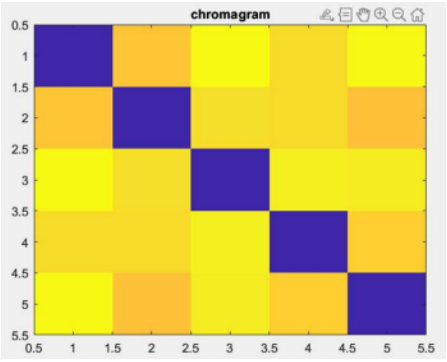
- l = 0.2 and h = 0.6



- l = 0.2 and h=0.9



- l = 0.2 and h = 0.2



- l = 0.4 and h = 0.6

Analysis :
When we shorten the frame length in audio processing, especially for features like chromagram and MFCC (Mel-frequency cepstral coefficients), it really makes a difference. It's like zooming in on a picture – you get to see more details of the sound. Now, when we compare chromagram and MFCC, the chromagram stands out, particularly the box in the top left corner of the image. It seems to capture more of what we're interested in. So, adjusting the frame length helps us focus on the smaller, more local aspects of the sound. And finding the best settings often means trying out different options until we find the ones that work best for what we're doing.