Lab 1
Lab report due by 1:00 PM on Monday, February 6, 2017

---

1 Introduction

The goal of this lab is to explore some of the recent techniques developed in the audio industry to organize and search large music collection by content. The explosion of digital music has created a need to invent new tools to search and organize music.

2 Audio signal: from the sound to the wav file

Sounds are produced by the vibration of air; sound waves produce variations of air pressure. The sound waves can be measured using a microphone that converts the mechanical energy into electrical energy. Precisely, the microphone converts air pressure into voltage levels. The electrical signal is then sampled in time at a sampling frequency f_s, and quantized. The current standard for high quality audio and DVD is a sampling frequency of $f_s = 96$kHz, and a 24-bit depth for the quantization. The CD quality is 16 bit sampled at $f_s = 44.1$ kHz.

Assignment

1. Dolphins can only hear sounds over the frequency range $[7 - 120]$ kHz. At what sampling frequency f_s should we sample digital audio signals for dolphins?

2.1 Segments, Frames, and Samples

In order to automatically analyze the music, the audio files are segmented into non overlapping segments of a few seconds. This provides the coarsest resolution for the analysis. The audio features are computed at a much finer resolution. The audio file is divided into overlapping intervals of a few milliseconds over which the analysis is conducted.

In this lab the audio files are sampled at $f_s = 11,025$ Hz, and we consider intervals of size $N = 512$ samples, or 46 ms. This interval of 46 millisecond is called a **frame**. We will implement several algorithms that will return a set of features for each frame. While there is 512 samples per frame, we will usually return a feature vector of much smaller size.

3 The music

In the file `audio.zip` on D2L you will find 12 tracks of various length. There are two examples of six different musical genres:

1. Classical
2. Electronic
3. Jazz
4. Metal and punk
5. Rock and pop
6. World

The name of the file indicates its genre.

The musical tracks are chosen because they are diverse but also have interesting characteristics, which should be revealed by your analysis.

- track201-classical is a part of a violin concerto by J.S. Bach (13-BWV 1001 : IV. Presto).
- track204-classical is a classical piano piece composed by Robert Schumann (Davidsbundlertanze, Op.6. XVI).
- track370-electronic is an example of synthetic music generated a software that makes it possible to create notes that do not have the regular structure of Western music. The pitches may be unfamiliar but the rhythm is quite predictable.
- track396-electronic is an example of electronic dance floor piece. The monotony and the simplicity of the synthesizer and the bass drum are broken by vocals.
- track437-jazz is another track by the same trio as track439-jazz
- track439-jazz is an example of (funk) jazz with a Hammond B3 organ, a bass, and a percussion. The song is characterized by a well defined rhythm and a simple melody.
- track463-metal is an example of rock and metal with vocals, bass, electric guitars and percussion.
- track492-metal is an example of heavy metal with heavily distorted guitars, drums with double bass drum.
- track547-rock is an example of new wave rock of the 80's. The music varies from edgy to hard. It has guitars, keyboards, and percussion.
- track550-rock is an example of pop with keyboard, guitar, bass, and vocals. There is a rich melody and the sound of steel-string guitar.
- track707-world: this is a Japanese flute, monophonic, with background sounds between the notes. The notes are held for a long time.
- track729-world: this is a piece with a mix of Eastern and Western influence using electric and acoustic sarod and on classical steel-string guitars.

Assignment

2. Write a MATLAB function that extract T seconds of music from a given track. You will use the MATLAB function `waveread` to read a track and the function `play` to listen to the track.

   In the lab you will use T =**24 seconds** from the middle of each track to compare the different algorithms. Download the files, and test your function.

4 Low level features: time domain analysis

The most interesting features will involve some sophisticated spectral analysis. There are however a few simple features that can be directly computed in the time domain, We describe some of the most popular features in the following.

4.1 Loudness

The standard deviation of the original audio signal $x[n]$ computed over a frame of size N provides a sense of the loudness,

$$\sigma(n) = \sqrt{\frac{1}{N-1}\sum_{m=-N/2}^{N/2-1}\left[x(n+m) - \mathbb{E}[x_n]\right]^2} \quad \text{with} \quad \mathbb{E}[x_n] = \frac{1}{N}\sum_{m=-N/2}^{N/2-1} x(n+m) \tag{1}$$

4.2 Zero-crossing

The Zero Crossing Rate is the average number of times the audio signal crosses the zero amplitude line per time unit. The ZCR is related to the pitch height, and is also correlated to the noisiness of the signal. We use the following definition of ZCR,

$$\text{ZCR}(n) = \frac{1}{2N}\sum_{m=-N/2}^{N/2-1} |\text{sgn}(x(n+m)) - \text{sgn}(x(n+m-1))|. \tag{2}$$

Assignment

3. Implement the loudness and ZCR and evaluate these features on the different music tracks. Your MATLAB function should display each feature as a time series in a separate figure (see e.g. Fig. 1).

4. Comment on the specificity of the feature, and its ability to separate different musical genre.

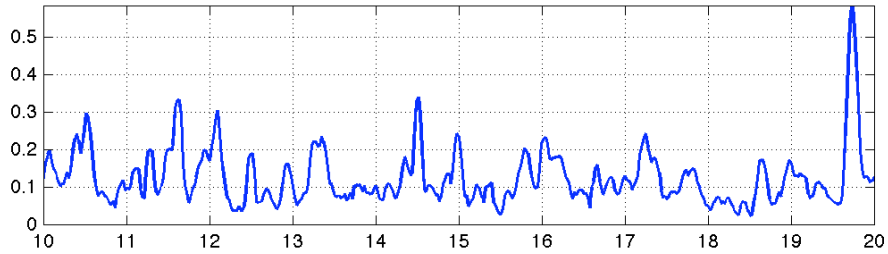

Figure 1: Zero crossing rate as a function of the frame number.

5 Low level features: spectral analysis

5.1 Windowing and spectral leaking

Music is made up of notes of different pitch. It is only natural that most of the automated analysis of music should be performed in the spectral (frequency) domain.

Our goal is the reconstruction of a musical score. Our analysis requires $N = 512$ samples to compute notes over a frame. The spectral analysis proceeds as follows. Each frame is smoothly extracted by multiplying the original audio signal by a tapper window w. The Fourier transform of the windowed signal is then computed. If x_n denotes a frame of size $N = 512$ extracted at frame n, and w is a window of size N, then the Fourier transform, X_n (of size N) for the frame n is given by

$$\begin{aligned} Y &= \text{FFT}(w. * x_n); \\ K &= N/2 + 1; \\ X_n &= Y(1 : K); \end{aligned} \tag{3}$$

There are several simple descriptors that can be used to characterize the spectral information provided by the Fourier transform over a frame.

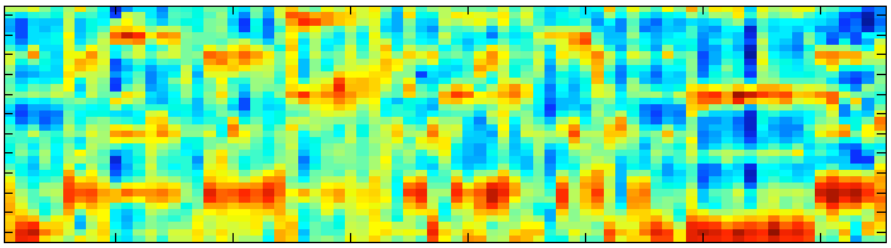

Figure 2: Spectrogram: square of the magnitude (color-coded) of the Fourier transform, $|X_n(k)|^2$, as a function of the frame index n (x-axis), and the frequency index k (y-axis).

Assignment

5. Let
$$x[n] = \cos(\omega_0 n), n \in \mathbb{Z} \tag{4}$$
Derive the theoretical expression of the discrete time Fourier transform of x, given by
$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}. \tag{5}$$

6. In practice, we work with a finite signal, and we multiply the signal $x[n]$ by a window $w[n]$. We assume that the window w is non zero at times $n = -N/2, \ldots, N/2$, and we define
$$y[n] = x[n]w[n - N/2]. \tag{6}$$
Derive the expression of the Fourier transform of $y[n]$ in terms of the Fourier transform of x and the Fourier transform of the window w.

Assignment

7. Implement the computation of the windowed Fourier transform of y, given by (3). Evaluate its performance with pure sinusoidal signals and different windows:
   - Bartlett
   - Hann
   - Kaiser,

8. Compute the spectrogram of an audio track as follows:
   (a) Decompose a track into a sequence of N_f overlapping *frames* of size N. The overlap between two frames should be $N/2$.
   (b) Compute the magnitude squared of the Fourier transform, $|X(k)|^2, k = 1, \ldots, K$ over each frame n.
   (c) Display the Fourier transform of all the frames in a matrix of size $K \times N_f$. The spectrogram should look like Fig. 2.

   You will experiment with different audio tracks, as well as pure sinusoidal tones. Do the spectrograms look like what you hear?

In the rest of the lab, we will be using a Kaiser window to compute the Fourier transform, as explained in (3).

5.2 Spectral centroid and spread

The first and second order moments, given by the mean $\mathbb{E}[|X_n|]$ and standard deviation $\sigma(|X_n|)$ of the magnitude of the Fourier transform.
In fact, rather than working directly with the Fourier transform, we prefer to define a concept of *frequency distribution* for frame n according to,

$$\widehat{X}_n(k) = \frac{|X_n(k)|}{\sum_{l=1}^{K} |X_n(l)|}. \tag{7}$$

Then the first two moments of $\widehat{X}_n(k)$ are given by

$$\sigma(\widehat{X_n}) = \sqrt{\frac{1}{K-1}\sum_{k=1}^{K}\left[\widehat{X}_n(k) - \mathbb{E}[\widehat{X}_n]\right]^2} \quad \text{with} \quad \mathbb{E}[\widehat{X}_n] = \frac{1}{K}\sum_{k=1}^{K} k\widehat{X}_n(k) \tag{8}$$

The spectral centroid $\mathbb{E}[\widehat{X_n}]$ can be used to quantify sound sharpness or brightness. The spread, $\sigma(\widehat{X_n})$, quantifies the spread of the spectrum around the centroid, and thus helps differentiate between tone-like and noise-like sounds.

5.3 Spectral flatness

Spectral flatness is the ratio between the geometric and arithmetic means of the magnitude of the Fourier transform,

$$\text{SF}(n) = \frac{\left(\prod_{k=1}^{K} |X_n(k)|\right)^{1/K}}{\frac{1}{K}\sum_{k=1}^{K} |X_n(k)|} \tag{9}$$

The flatness is always smaller than one since the geometric mean is always smaller than the arithmetic mean. The flatness is one, if all $|X(k)|$ are equal. This happens for a very noisy signal. A very small flatness corresponds to the presence of tonal components. In summary, this is a measure of the noisyness of the spectrum.

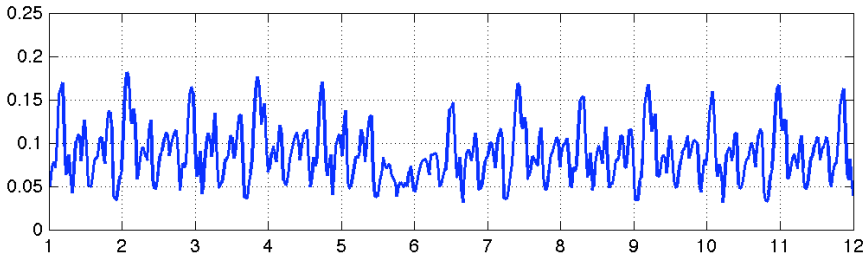

Figure 3: Spectral flatness as a function of the frame number.

5.4 Spectral flux

The spectral flux is a global measure of the spectral changes between two adjacent frames, $n-1$ and n,

$$F_n = \sum_{k=1}^{K}\left(\widehat{X}_n(k) - \widehat{X}_{n-1}(k)\right)^2 \tag{10}$$

where $\widehat{X}_n(k)$ is the normalized frequency distribution for frame n, given by (7).

Assignment

9. Implement all the low level spectral features and evaluate them on the different tracks. Your MATLAB function should display each feature as a time series in separate figure (e.g. see Fig. 3).
10. Comment on the specificity of the feature, and its ability to separate different musical genres.

5.5 Application: Mpeg7 Low Level Audio Descriptors

MPEG 7, also known as "Multimedia Content Description Interface," provides a standardized set of technologies for describing multimedia content. Part 4 of the standard specifies description tools that pertain to multimedia in the audio domain. The standard defines Low-level Audio Descriptors (LLDs) that consist of a collection of simple, low complexity descriptors to characterize the audio content. Some of the features are similar to the spectral features defined above.

6 Basic Psychoacoustic Quantities

In order to develop more sophisticated algorithms to analyze music based on its content, we need to define several subjective features such as timbre, melody, harmony, rhythm, tempo, mood, lyrics, etc. Some of these concepts can be defined formally, while others are more subjective and can be formalized using a wide variety of different algorithms. We will focus on the features that can be defined mathematically.

6.1 Psychoacoustic

Psychoacoustics involves the study of the human auditory system, and the formal quantification of the relationships between the physics of sounds, and our perception of audio. We will describe some key aspects of the human auditory system:

1. the perception of frequencies and pitch for pure and complex tones;
2. the frequency selectivity of the auditory system: our ability to perceive two similar frequencies as distinct;
3. the modeling of the auditory system as a bank of auditory filters;
4. the perception of loudness;
5. the perception of rhythm.

6.2 Perception of frequencies

The auditory system, like the visual system, is able to detect frequencies over a wide range of scales. In order to measure frequencies over a very large range, it operates using a logarithmic scale. Let us a consider a pure tone, modeled by a sinusoidal signal oscillating at a frequency ω. If $\omega < 500$ Hz, then the perceived tone – or pitch – varies as a linear function of ω. When $\omega > 1,000$ Hz, then the perceived pitch increases logarithmically with ω.

Several frequency scales have been proposed to capture the logarithmic scaling of frequency perception.

6.3 The mel/Bark scale

The Bark (named after the German physicist Barkhausen) is defined as

$$z = 7\text{arcsinh}\left(\omega/650\right) = 7\log\left(w/650 + \sqrt{1 + (w/650)^2}\right),$$

where ω is measured in Hz. The mel-scale is defined by the fact that 1 bark = 100 mel. In this lab we will use a slightly modified version of the mel scale defined by

$$m = 1127.01048 * \log\left(1 + \omega/700\right). \tag{11}$$

6.4 The cochlear filterbank

Finally, we need to account for the fact that the auditory system behaves as a set of filterbanks, with overlapping frequency responses. For each filter, the range of frequencies over which the filter response is significant is called the critical band. Our perception of pitch can be quantified using the total energy at the output of each filter bank. All spectral energy that falls into one critical band is summed up, leading to a single number for that frequency band.

We describe in the following a simple model of the cochlear filterbank. The filter bank is constructed using $N_B = 40$ logarithmically spaced triangle filters centered at the frequencies Ω_p, defined by

$$\text{mel}_n = 1127.01048 * \log\left(1 + \Omega_p/700\right), \tag{12}$$

where the sequence of mel frequencies is equally spaced in the mel scale,

$$\text{mel}_n = n\frac{\text{mel}_{\max} - \text{mel}_{\min}}{N_B}, \tag{13}$$

with

$$\text{mel}_{\max} = 1127.01048 * \log\left(1 + 0.5 * \omega_s/700\right), \tag{14}$$

$$\text{mel}_{\min} = 1127.01048 * \log\left(1 + 20/700\right), \tag{15}$$

and $N_B = 40$. Each filter H_p is centered around the frequency Ω_p, and defined by

$$H_p(\omega) = \begin{cases} \dfrac{2}{\Omega_{p+1} - \Omega_{p-1}} \dfrac{\omega - \Omega_{p-1}}{\Omega_p - \Omega_{p-1}} & \text{if} \quad \omega \in [\Omega_{p-1}, \Omega_p), \\ \dfrac{2}{\Omega_{p+1} - \Omega_{p-1}} \dfrac{\Omega_{p+1} - \omega}{\Omega_{p+1} - \Omega_p} & \text{if} \quad \omega \in [\Omega_p, \Omega_{p+1}). \end{cases} \tag{16}$$

Each triangular filter is normalized such that the integral of each filter is 1. In addition, the filters overlap so that the frequency at which the filter H_p is maximum is starting frequency for the next filter h_{n+1}, and the edge frequency of h_{n-1}.

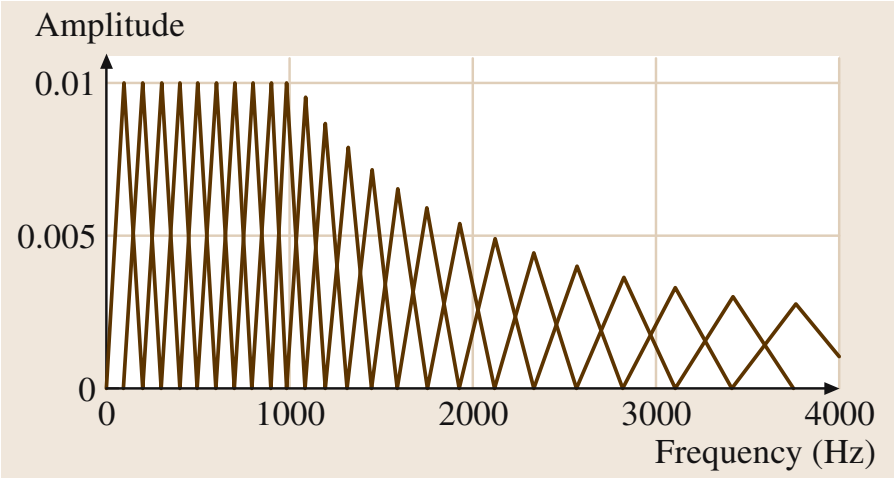

Figure 4: The filterbanks used to compute the mfcc.

Finally, the mel-spectrum (MFCC) coefficient of the n-th frame is defined for $p = 1, \ldots, N_B$ as

$$\texttt{mfcc}[p] = \sum_{k=1}^{K} |H_p(k) X_n(k)|^2 \tag{17}$$

where the filter H_p is a discrete implementation of the continuous filter defined by (16). The discrete filter is normalized such that

$$\sum_j H_p(j) = 1, p = 1, \ldots, N_B. \tag{18}$$

The MATLAB code in Fig. 5 computes the sequence of frequencies Ω_p.

```
nbanks = 40; %% Number of Mel frequency bands

% linear frequencies
linFrq  = 20:fs/2;

% mel frequencies
melFrq = log ( 1 + linFrq/700) *1127.01048;

% equispaced mel indices
melIdx = linspace(1,max(melFrq),nbanks+2);

% From mel index to linear frequency
melIdx2Frq = zeros (1,nbanks+2);

% melIdx2Frq (p) = \Omega_p

for i=1:nbanks+2
   [val indx] = min(abs(melFrq - melIdx(i)));
   melIdx2Frq(i) = linFrq(indx);
end
```

Figure 5: Computation of the sequence of frequencies Ω_p associated with the normalized hat filters defined in (16).

Assignment

11. Implement the computation of the triangular filterbanks $H_p, p = 1, \ldots, N_B$. Your function will return an array `fbank` of size $N_B \times K$ such that `fbank(p,:)` contains the filter bank H_p.

12. Implement the computation of the mfcc coefficients, as defined in (17).