# Chapter 6

# From audio to content

*Giovanni De Poli - Luca Mion - Nicola Orio - Antonio Rodà*



## 6.1 Sound Analysis

Models for the *representation* the information from sound are necessary for the description of the information, from the perceptive and operative point of view. Beyond the models, analysis methods are needed to discover the parameters which allow sound description, possibly lossless from the physical and perceptual properties description.

**Audio features extraction.** When aiming to the *extraction* of information for sound, we need to discard every feature which is non relevant. This process of feature extraction consists of various steps, starting from pre-processing the sound, then windowing, extraction, and post-processing procedures. An audio signal classification system can be generally represented as represented in Figure 6.1.

**Pre-processing:** The pre-processing consists of noise reduction, equalization, low-pass filtering. In speech processing (voice has a low-pass behavior) a pre-emphasis is applied by high-pass filtering the signal to smooth the spectrum, to achieve an uniform energy distribution spectrum.

**Windowing:** Second step is to create frames from a continuous waveform signal. Frames are partially overlapping, as discuss in section 6.1.1

**Extraction:** Third step consists in obtaining a vector of features for each frame.

**Post-processing:** In the post-processing phase, the most relevant cues of the features vector are selected. For instance, earlier mel-cepstral coefficients can be lightly weighted when having low frequency noise.

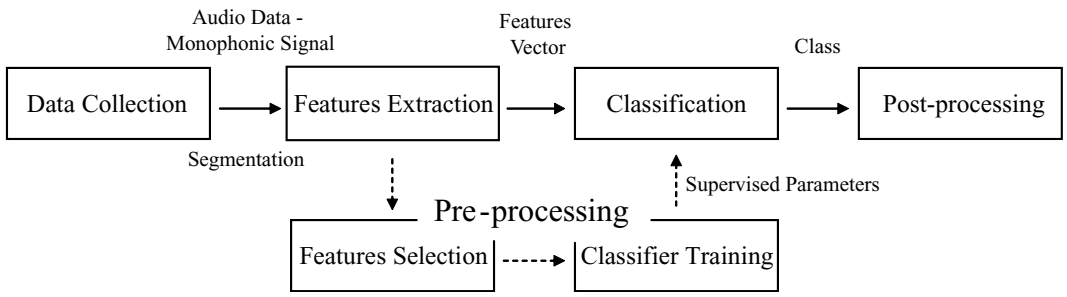



**Figure 6.1:** *Scheme for supervised classification.*

The origins of audio features extraction derive from the research related to speech processing and representation. Initially, they proposed their representation for compressing data (i.e. for telephone applications), and later for speech recognition and synthesis. Features-based representation for non-speech audio was introduced at the end of the 90s, exploring different directions of research from the typical Short Time Fourier Transform; for example, representations based on Cochlear Models, Wavelets, and the MPEG audio compression filterbank. Audio features differ not only in how effective they are but also in their performance requirements, for instance FFT-based features might be more appropriate for a real time applications than a computationally intensive but perceptually more accurate features based on a cochlear filterbank.

Many features have been proposed by different communities in previous studies, e.g. from the musical instrument classification or psycho-acoustical studies. Features can be grouped according to various aspects; *steadiness*, because the features can represent either a value derived from signal, or a parameter from a model of signal behavior (e.g. mean, standard deviation); *time extent* of the description provided by the feature (global or instantaneous descriptors); *abstractness* of the feature, that is how the feature represents (e.g. spectral envelope can be represented via cepstrum or LPC on two different levels of abstraction); *extraction process*, which groups features according if features extracted from waveform, features extracted from transformed signal, features related to models, features based on auditory models.

**Audio framework in MPEG-7** Mpeg7 became ISO/IEC 15398 standard in Fall 2001. It is the standard for describing multimedia content that provides the richest multimedia content description tools for applications ranging from content management, organization, navigation, and automated processing. The MPEG-7 standard defines a large library of core description tools, and a set of system tools provides the means for deploying the description in specific storage and transport environments. MPEG-7 addresses many different applications in many different environments, which means it needs to provide a flexible framework for describing multimedia data, including extensibility (using the Description Definition Language) and restrictibility (via the MPEG-7 Profiles under specification). The Mpeg7 descriptors for audio documents content consist of low-level and high-level descriptors. The task of supplying suitable descriptors for the extraction of significant characteristic is very interesting and it is also pretty hard. Audio signals can belong to different musical genres, played with various instruments, and can refer to speech-non speech sound, environmental sounds, mechanical sounds etc.

According to the features description proposed for the MPEG-7 standard, the extraction process can be summarized as depicted in Fig. 6.2. In the following, we will present many audio cues that are useful; some of them are also used as descriptors for standard MPEG-7 files, in that cases the Mpeg7

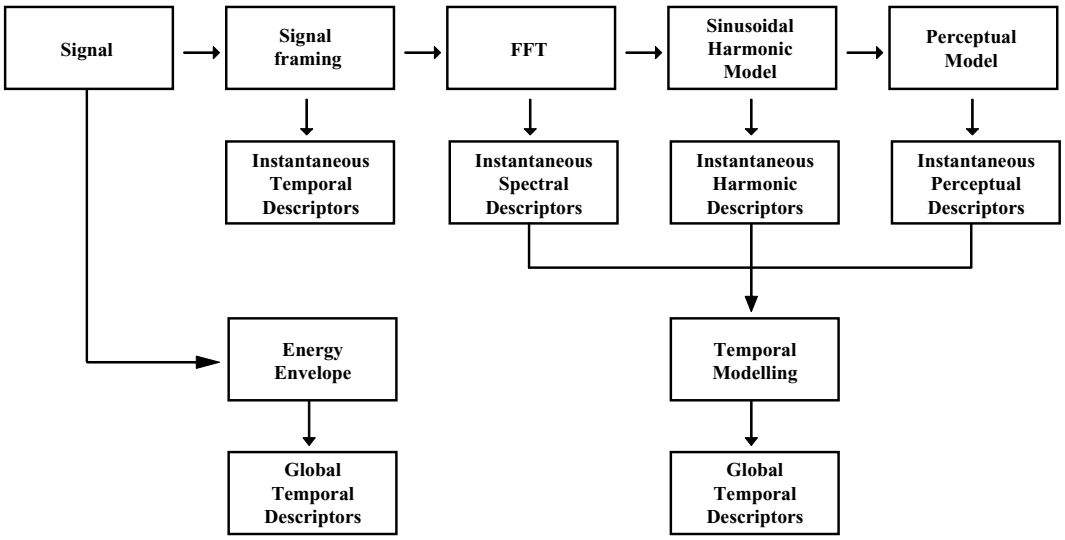



**Figure 6.2:** *Features and extraction processes as proposed by Cuidado Project for MPEG-7 sound descriptors.*

descriptor name will be indicated along with the feature description.

### 6.1.1 Time domain: Short-time analysis

Time domain methods divide according to the time extent of the description provided by the feature. Short and long time analysis techniques apply to different parts of the object (i.e. a sounding note) to describe the attack of the sound (short extent) or the loudness of a note (long extent).

In the case of short-time analysis, an hypothesis of stationarity of sound is now needed, with respect to the sampling rate. That is, we assume the signal to be slowly time-varying over intervals of a few milliseconds to manage the parameters as they were constants within a single frame. Considering the vocal signal for instance, this assumption can be justified by the fact that the words generation is affected by both the vocal chords and the entire phonation apparatus (larynx, mouth, tongue) with modifications not much quick, such to be considered constants within 100-200 ms. Therefore, the signal is divided into frames of e.g. 10 ms. The number of frames computed per second is called frame rate. A tapered window function (e.g. a Gaussian or Hanning window) is applied to each frame to minimize the discontinuities at the beginning and at the end. Consecutive frames are usually considered with some overlap for smoother analysis, and the duration between two fames defines the temporal accuracy, which is chosen according to the target accuracy. This analysis step is called *hop-size* $H$. On short time analysis, the signal is multiplied by a function $w[k]$ on $l = 0, \ldots, N-1$ which is null out the temporal window.

**Windowing** Temporal window defines the frame duration. The choice of the window depends on three factors: *(1)* it has to be short enough to be under the stationarity assumption; *(2)* it has to be long enough to comfortably allow the parameter computation and to reduce noise if affecting the signal; *(3)* windows have to entirely cover the parameter, that is the *frame rate* of the parameter has to be at least the inverse of the window duration.

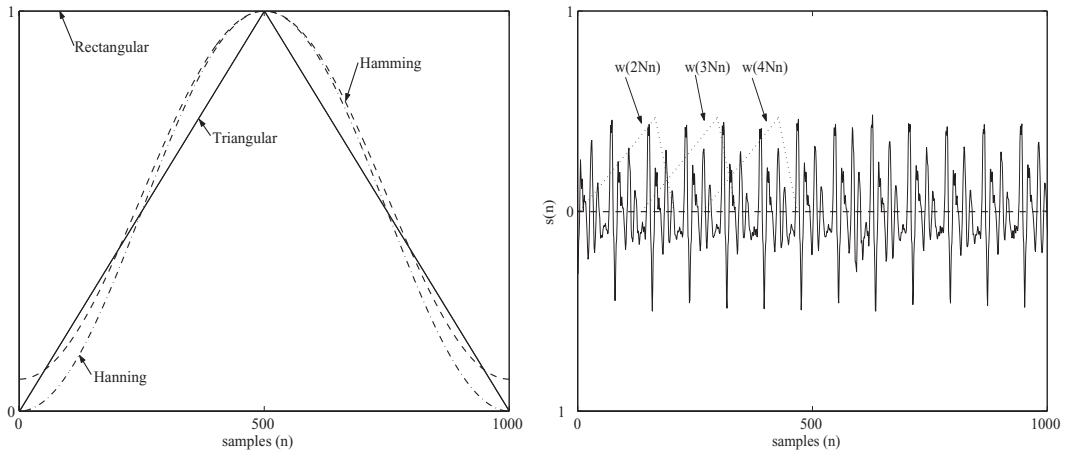
The simplest window is the rectangular window, which gives the poorest quality result of all standard windows and requires a very large number of coefficients to yield a result equivalent to a

higher quality window:

$$r[n] = \begin{cases} 1 & \text{for } 0 \leq n \leq N-1 \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

Many applications use longer windows to satisfy the hypothesis of stationarity, but also changing the shape to emphasize the central samples (see Fig. 6.3). An alternative to the rectangular window 6.1 is the Hamming window, which is formulated from cosines and the like and follow sinusoidal curves:

$$h[n] = \begin{cases} 0.54 - 0.46\cos(\frac{2\pi n}{N-1}) & \text{for } 0 \leq n \leq N-1 \\ 0 & \text{otherwise} \end{cases} \tag{6.2}$$




(a) (b)

**Figure 6.3:** *Various windows on the left; on the right, three windows over the signals*$[n]$*, shifted from origin by* $2N$*,* $3N$ *e* $4N$ *samples*

**Preprocessing** Some parameters in the time domain can also be represented by following formulation:

$$Q[n] = \sum_{m=-\infty}^{\infty} T[s[m]]w[n-m] = T[s] * w[n] \tag{6.3}$$

where $T[\cdot]$ is a (even non-linear) transformation weighted by a window $w[n]$. Before being processed, signal can be filtered to select the correct frequency band. In eq. 6.3, $w[n]$ can be a finite impulse response filter (FIR), which allows us to decrease the frame rate (less computational load), or alternatively an IIR filter; an example of IIR window is:

$$w[n] = \begin{cases} a^n & \text{for } n \geq 0 \\ 0 & \text{for } n < 0 \end{cases} \tag{6.4}$$

where $0 < a < 1$;

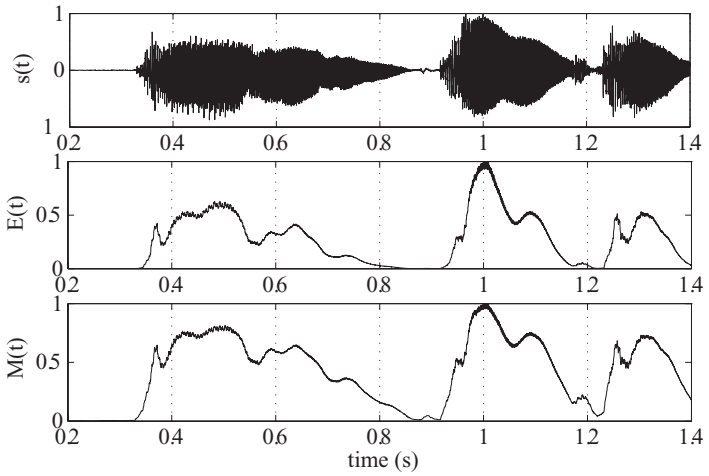



**Figure 6.4:** *On the top: a short excerpt from violin performance of Handel's Flute Sonata in E minor. Other diagrams represent normalized* Short-Time Average Energy *and* Short-Time Average Magnitude*, computed using rectangular windows with N=100 samples and* frame rate *equal to signal sampling rate (8kHz).*

#### 6.1.1.1 Short-Time Average Energy and Magnitude

For a discrete signal, the *Short-Time Average Energy* is defined as follows:

$$E[n] = \frac{1}{N} \sum_{i=n-N+1}^{n} s[i]^2 \tag{6.5}$$

thus is equivalent to $Q[n]$ in equation 6.3 if $T[\cdot] = (\cdot)^2$.

A drawback of *Short-Time Average Energy* is to be affected when large signals occur. To face this problem, one solution is to define *Short-Time Average Magnitude* as follows:

$$M[n] = \frac{1}{N} \sum_{i=n-N+1}^{n} |s[i]| \tag{6.6}$$

which is equivalent to 6.3 when $T[\cdot] = |\cdot|$.

**M-6.1**

Write two MATLAB functions to compute Short-Time Average Energy e Magnitude.

### M-6.1 Solution

```
Nframe=100;       %   numero di campioni per frame
Ns=max(size(s)); %   numero di campioni del segnale

for n=1:Ns;       %   calcola la Short-Time Average Energy
   E(n,1)=sum(s(max(1,n-Nframe+1):n).*...
```

```
      s(max(1,n-Nframe+1):n))/Nframe;
end;

for n=1:Ns;       %   calcola la Short-Time Average Magnitude
    M(n,1)=sum(abs(s(max(1,n-Nframe+1):n)))/Nframe;
end;

      %    disegna E(t) e M(t)
E=E/max(E);       %   normalizza E(t)
tempi = (1/fS)*[1:max(size(E))]; subplot(2,1,1);
plot(tempi,E); xlabel('time (s)'); ylabel('E(t)');

M=M/max(M);       %   normalizza M(t)
tempi = (1/fS)*[1:max(size(M))]; subplot(2,1,2);
plot(tempi,M); xlabel('time (s)'); ylabel('M(t)');
```

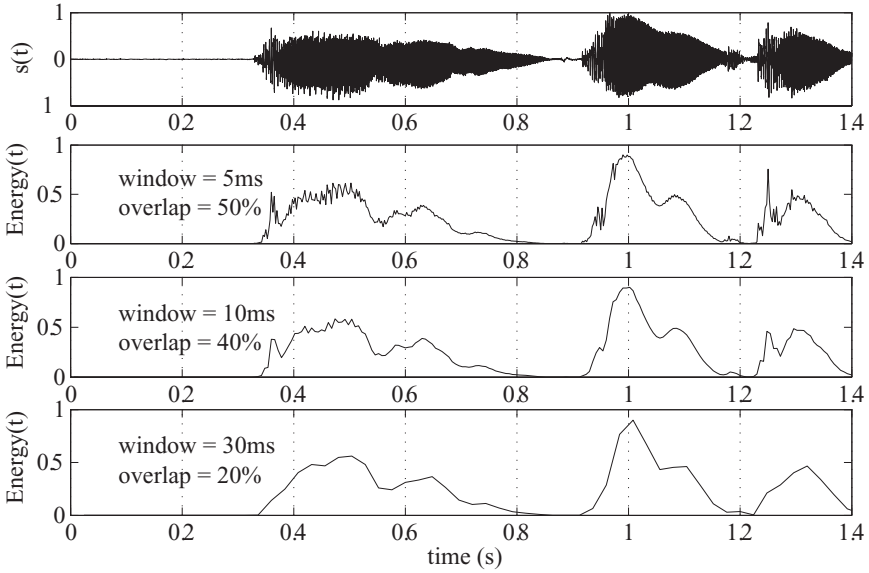



**Figure 6.5:** *On the top: a short excerpt from violin performance of Handel's Flute Sonata in E minor. Other diagrams represent* Short-Time Average Energy *computed with different Hamming windows.*

#### 6.1.1.2 Short-Time Average Zero-Crossing Rate

*Zero-Crossing Rate* (*ZCR*) can yield useful information about the spectrum with low computational costs. *ZCR* represents the number of crossing through the zero signal, and it is mathematically described as the changing of the sign of two following samples. For narrow-band signals (e.g. sinusoids or band-pass filter output), from *ZCR* we obtain the fundamental frequency (F0) of the signal:

$$F0 = \frac{ZCR * F_S}{2} \tag{6.7}$$

where $F_S$ is the signal sampling rate and $ZCR$ is expressed as *zero crossing* for sample. We can have $ZCR = Q[n]$ when in 6.3 we use $T[s[n]] = |\text{sign}(s[n]) - \text{sign}(s[n-1])|/2$, and scaling the window $w[n]$ by a factor $1/N$; thus, we have

$$Z[n] = \frac{1}{N} \sum_{m=n-N+1}^{n} \frac{|\text{sign}(s[m]) - \text{sign}(s[m-1])|}{2} w[n-m] \tag{6.8}$$

where the sign of $s[n]$ is defined as follows:

$$\text{sign}(s[n]) = \begin{cases} 1 & \text{for } s[n] \geq 0 \\ -1 & \text{otherwise} \end{cases} \tag{6.9}$$

In Figure 6.6 the Zero crossing Rate of the word /sono/ is shown. Notice the high ZCR values at the beginning in correspondence of the unvoiced /S/ and low values for the voiced part. This properties can be exploited to distinguish voiced (periodic) from unvoiced sounds.

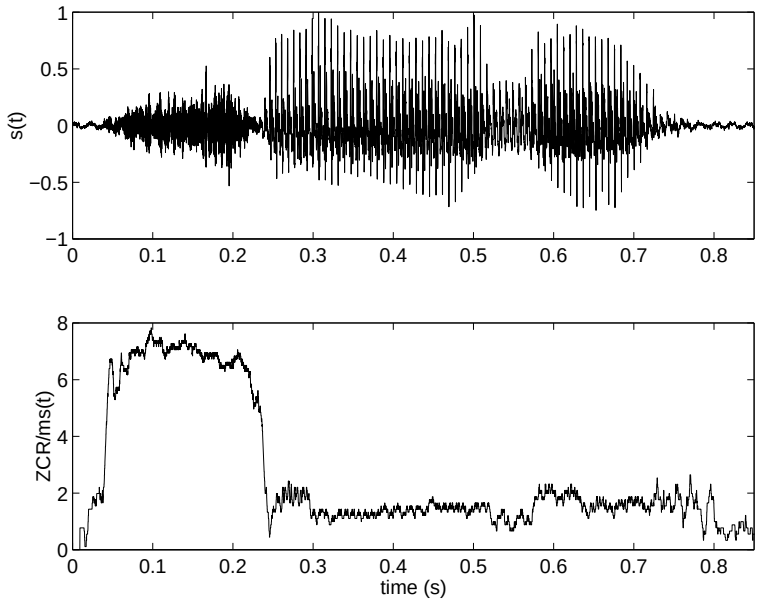



**Figure 6.6:** *Zero-Crossing Rate of the word /SONO/.*

**M-6.2**

Write a MATLAB function for Zero Crossing Rate computation.

### M-6.2 Solution

```
Nframe = 100;          %  numero di campioni per frame
Ns = max(size(s));

for n = 1+Nframe:Ns;  %  calcola la Short-Time Average ZCR
   Z(n,1) = sum(abs(sign(s(n-Nframe+1:n))- ...
      sign(s(n-Nframe:n-1)))/2)/Nframe;
end;

Z=Z*fS/1000;           %  Zero-Crossing per ms

     %  disegna Z(t):
t = (1/fS)*[1:max(size(Z))];
plot(t,Z); xlabel('time (s)'); ylabel('ZCR/ms(t)');
```

#### 6.1.1.3 Short-Time Autocorrelation Function

Signal *autocorrelation* is the inverse Fourier transform of the spectral density of the signal energy $C_s(f)$. It is formulated as follows:

$$\mathcal{F}[\phi[k]] = C_s(f) = |S(f)|^2 \tag{6.10}$$

For a discrete signal, it is defined as

$$\phi[k] = \sum_{m=-\infty}^{\infty} s[m]s[m+k] \tag{6.11}$$

Autocorrelation preserves the information related to harmonics, formants amplitude and their frequencies. Equation 6.11 shows that $\phi[k]$ is somehow representing the signal likeness to its shifted version. So, it will assume higher values when occurring delays $k$ such that $s[m]$ and $s[m+k]$ have similar waveforms.

Some important properties of $\phi[k]$ are the followings:

1. it is an even function: $\phi[k] = \phi[-k]$
2. when $k = 0$ $\phi[k]$ takes its maximum value, $\phi[0] \geq |\phi[k]| \ \ \forall k$
3. $\phi[0]$ corresponds to the signal energy (or to the average power if the signal is periodic or non-deterministic)
4. if the signal is periodic with period $P$, the autocorrelation is periodic with the same period of the analyzed signal: $\phi[k] = \phi[k+P]$ (this is an important property when the signal periodicity has to be estimated). In fact it has maximal values at time lag $P$, $2P$, and so on. If the sound is quasi-periodic, we will have local maxima at lag $P$, $2P$, and so on (see Fig. 6.7).

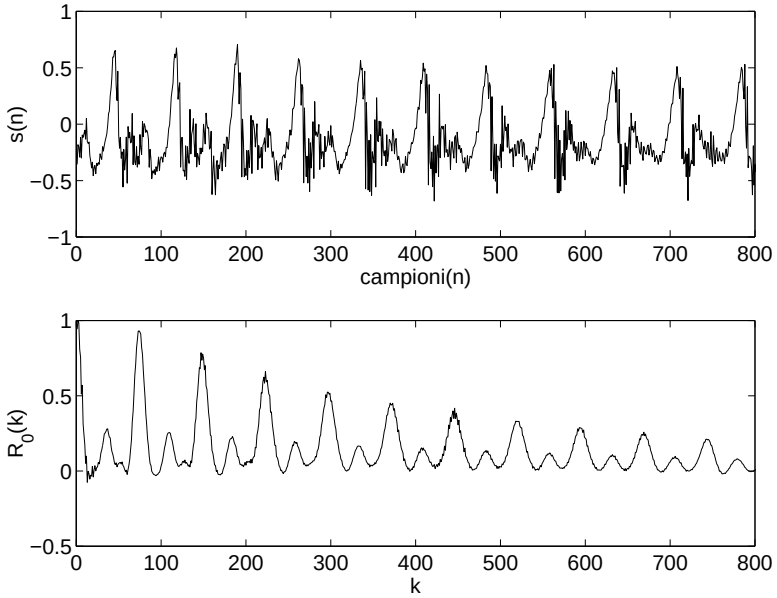



**Figure 6.7:** *Autocorrelation of a voiced sound..*

**M-6.3**

Write a MATLAB function for computing the *Short-Time Autocorrelation Function*.

### M-6.3 Solution

```
Ns = max(size(s));   %no. of samples
window = ones(Ns,1); %rectangular window
s_w = s.*window;

for k = 1:Ns-1;    %compute ST autocorrelation
  R0(k) = sum(s_w(1:Ns-k) .* s_w(k+1:Ns));
end;
%plots
R0=R0/max(abs(R0));   %normalize R0
plot(1:max(size(R0)),R0); xlabel('k'); ylabel('R_0(k)');
```

*Short-Time Autocorrelation Function* (STAF) is applied for pitch extraction applications and speech-non speech signal discriminations. It is yielded by Eq. 6.11 after filtering the signal with windows $w[n]$:

$$R_n[k] = \sum_{m=-\infty}^{\infty} s[m]w[n-m]s[m+k]w[n-k-m] \tag{6.12}$$

This equation can be seen in the form:

$$R_n[k] = \sum_{m=-\infty}^{\infty} [s[n+m]w'[m]] \cdot [s[n+m+k]w'[k+m]] \tag{6.13}$$

where $w'[n] = w(-n)$; if we assume that $w'[n]$ has finite duration $N$ we obtain:

$$R_n[k] = \sum_{m=0}^{N-1-k} [s[n+m]w'[m]] \cdot [s[n+m+k]w'[k+m]] \tag{6.14}$$

#### 6.1.1.4 Short-Time Average Magnitude Difference Function

Beyond the *Short-Time Autocorrelation Function*, the detection of F0 can also be faced by means of the *Short-time Average Magnitude Difference Function* (AMDF). For a periodic signal with period $P$, succession $d[n] = s[n] - s[n-k]$ is zero when $k = 0, \pm P, \pm 2P, \ldots$, so we can consider the absolute value of the difference of $s[m]$ and $s[m-k]$ instead of their product:

$$\gamma_n[k] = \sum_{m=-\infty}^{\infty} |s[n+m]w[m] - s[n+m-k]w[m-k]| \tag{6.15}$$

We can have a simpler formulation when $w[n]$ is rectangular, with duration N:

$$AMDF[k] = \sum_{m=k}^{N-1} |s[m] - s[m-k]| \tag{6.16}$$

The AMDF of the signal of Fig. 6.7 is shown in fig. 6.8.

**M-6.4**

Write a MATLAB function for *Short-time Average Magnitude Difference Function* computing.

### M-6.4 Solution

```
Ns=max(size(s));    %  numero di campioni

window=ones(ceil(Ns/2)+1,1);   %  finestra rettangolare

for k=1:floor(Ns/2)-1;  %  calcola la Short-Time AMDF
   STAMDF(k) = sum(abs(s(floor(Ns/2):Ns).* window - ...
      s(floor(Ns/2)-k:Ns-k).* window));
end;

     %  disegna STAMDF(t):
STAMDF=STAMDF/max(STAMDF);     %  normalizza STAMDF(t)
plot(1:max(size(STAMDF)),STAMDF); xlabel('k'); ylabel('AMDF(k)');
```

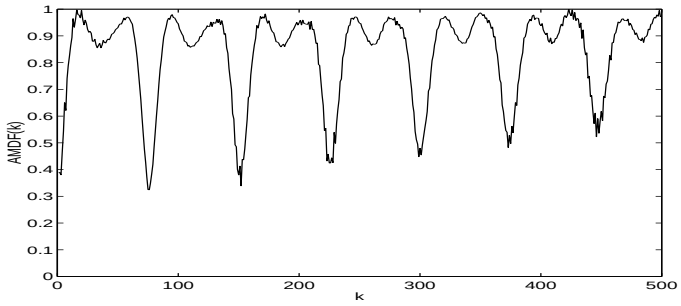



**Figure 6.8:** *Short time AMDF of the voiced sound of fig. 6.7.*

## 6.1.2 Audio segment temporal features

Descriptors computed after long-time localization of sound events (segmentation). These descriptors are extracted from the whole signal.

### 6.1.2.1 Temporal envelope estimation

Amplitude envelope can be defined as the variation of the amplitude of a note while sounding. This is often described through four phases called *attack, decay, sustain release* (ADSR). The basic idea to extract the temporal envelope starts with a low-pass filtering (i.e. 10 - 30 Hz), in order to extract the slow time changing components. One method to detect the envelope is based on *Short-Time Average Energy*, where window acts as a low-pass filter:

$$env[n] = \sqrt{E[n]}$$

### 6.1.2.2 ADSR envelope modelling

Modelling Attack, Decay, Sustain and Release of a note comes from the time representation of energy envelope. This feature is not always easily achievable because of overlapping notes with noise or with other notes. The estimation of attack can be achieved by means of either fixed or adaptive thresholds techniques, often empirically tuned. Fixed threshold methods consists of taking into account the possible presence of noise setting a threshold (e.g. to 20%) on energy envelope. Also, in order to take

into account the possibility that the maximum of the envelope does not occur at the end of the attack, another threshold is set (e.g. to 90%). Adaptive techniques are based on the behavior of the signal during the attack; the best threshold is chosen along multiple notes by repeated tunings, according to the slope of the threshold crossing. I will discuss some onset detection techniques in Sec. 6.4. This set of features consists of:

**Log-Attack Time** [In Mpeg7 is LogAttackTime]"

$$LAT = \log_{10}(attack\_time)$$

where $attack\_time$ is the time duration of note attack. This feature has been proven to be strongly related to perceptual description of timbres.

**Temporal Centroid** [In Mpeg7 is TemporalCentroid]"

$$TC = \frac{\sum_t env(t) \cdot t}{\sum_t env(t)}$$

where $env(t)$ is the temporal envelope. This is a useful parameter to distinguish percussive sounds from sustained sounds.

**Effective Duration** is a measure of the time the signal is perceptually meaningful. It is approximately given by the time the energy is above a given threshold (e.g. 40%).

#### 6.1.2.3 Pitch detection ($F0$) by time domain methods

The general problem of fundamental frequency estimation is to take a portion of signal and to find the dominant frequency of repetition.

Many applications are based on the detection of pitch, i.e. the dominat perceived frequency of a sound. The basic problem is to extract from a sound signal the fundamental frequency $F0$, which is the lowest sinusoidal component, or partial, which relates well to most of the other partials.

Difficulties arise from: (i) Not all signals are periodic; (ii) Those that are periodic may be changing in fundamental frequency over the time of interest; (iii) Signals may be contaminated with noise, even with periodic signals of other fundamental frequencies; (iv) Signals that are periodic with interval $T$ are also periodic with interval $2T$, $3T$ etc, so we need to find the smallest periodic interval or the highest fundamental frequency; (v) Even signals of constant fundamental frequency may be changing in other ways over the interval of interest.

In a pitched signal, most partials are harmonically related, meaning that the frequency of most of the partials are related to the frequency of the lowest partial by a small whole-number ratio. The frequency of this lowest partial is $F0$ of the signal. The simplest approach to periodicity evaluation is based on the investigation of the time domain waveform. We may test each $F0$ hypothesis by testing how well the signal will resemble a delayed version of itself. An evaluation criteria can be either the correlation or the difference between the signal and its delayed version.

From the Short-Time Autocorrelation Function we obtain the information on the signal periodicity (fig. 6.9) by means of $k_M$, that is the first maximum after the one related to $k = 0$:

$$F0 = \frac{F_S}{k_M} \tag{6.17}$$

where $F_S$ is the signal sampling rate. On the other side, if we use the Short-Time AMDF we have to consider the first minimum $k_m$ after the one related to $k = 0$ (fig. 6.10). However sometimes we

get a harmonic or sub-harmonic, depending on the shape of the spectrum. Whitening the signal by center-clipping is effective in minimizing this problem.

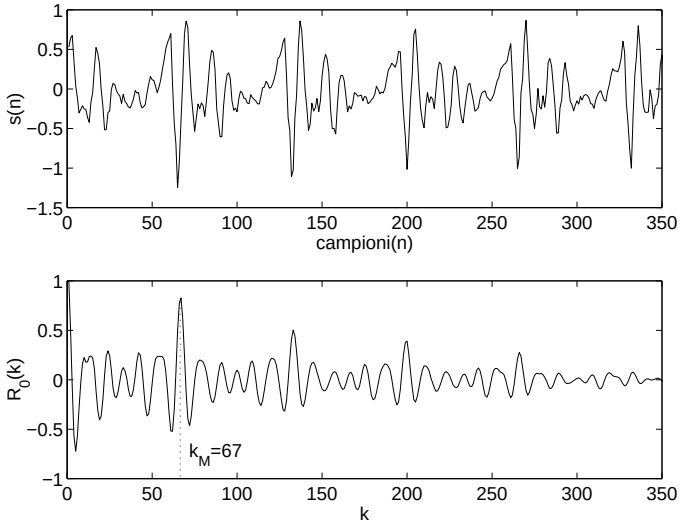



**Figure 6.9:** *Frame of the phoneme /OH/ and its Short-Time Autocorrelation Function. The position of the second maximum at $k_M$ indicates the pitch period.*

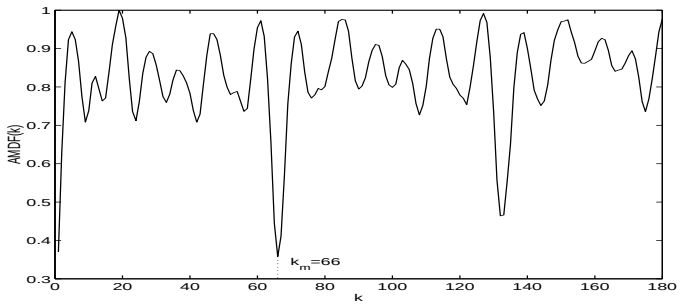



**Figure 6.10:** *AMDF of the frame of the phoneme /OH/ of fig. 6.9. The position of the second minimum at $k_m$ indicates the pitch period.*

A pitch estimator normally proceeds in three steps:

- pre-processing: the signal is filtered to remove high frequency components;
- pitch extraction;
- post-processing to correct possible errors.

**M-6.5**

Compute the pitch with *Short-Time Autocorrelation Function.*

### M-6.5 Solution

```
inizio=floor(fS*0.001);   %  salta il primo massimo
[massimo,kM] = max(R0(inizio:max(size(R0))));
kM=kM + inizio -1;
F0=fS/kM;
```

**M-6.6**

Compute the pitch with *Short-time Average Magnitude Difference Function*.

### M-6.6 Solution

```
inizio=floor(fS*0.001);   %  salta il primo minimo
[minimo,km] = min(STAMDF(inizio:max(size(STAMDF))));
km=km + inizio -1;
F0=fS/km;
```

### 6.1.3 Frequency domain analysis

We have already seen in the signal based sound modeling chapter how a sound can be represented in the frequency domain. Moreover we presented analysis methods which allow the estimation of the spectral representation. In the following sections other methods for sound analysis in the frequency domain will be presented.

#### 6.1.3.1 Energy features

**Sinusoid plus noise representation** Let's suppose $x[n]$ to be a sound, which is constituted by two components $x_S[n]$ (sinusoidal), and $x_R[n]$ (residual): $x[n] = x_S[n] + x_N[n]$, where $x_S[n] = \sum_{i=1}^{I} a_i cos(n2\pi f_i[n]/F_S + \phi_i[n])$.

In this form, $a_i$ represents the amplitude (in linear scale) and $f_i$ the frequency of $i$-th partial. These are time-changing parameters, which often are considered constant during a frame. The most used features, derived from this representation, are:

**Total amplitude of the sinusoidal component**, resulting by sum of the partials (dB expressed) within the frame:

$$AS_{tot} = 20\log_{10}\left(\sum_{i=1}^{I} a_i\right)$$

where $a_i$ is the amplitude of $i$-th partial;
**Total amplitude of the residual component**, resulting by sum of absolute values of the residual within the frame:

$$AR_{tot} = 20\log_{10}\left(\sum_{n=0}^{M-1} |x_R[n]|\right) = 20\log_{10}\left(\sum_{k=0}^{N-1} |X_R[k]|\right)$$

**Total amplitude of the sound**

$$\begin{aligned} A_{tot} &= 20\log_{10}\left(\sum_{n=0}^{M-1} |x[n]|\right) = 20\log_{10}\left(\sum_{k=0}^{N-1} |X[k]|\right) \\ &= 20\log_{10}\left(\sum_{i=1}^{I} a_i + \sum_{k=0}^{N-1} |X_R[k]|\right) \end{aligned}$$

**Total Energy** [In Mpeg7 is AudioPower]" estimates the signal power at a given time. It is estimated directly from the signal frame.

**Noise to Harmonic Ratio - NHR** is defined as the ratio between the energy of noise and the energy of harmonic part. For sustained sounds, energy modulation and fundamental frequency modulation are taken into account.

#### 6.1.3.2 Spectral shape features

**Spectral centroid** [In Mpeg7 is AudioSpectrumCentroid]" is the barycenter of the spectrum. It is computed considering the spectrum as a distribution which values are the frequencies and the probabilities to observe there are the normalized amplitude.

$$BR = \frac{\sum_{k=0}^{N-1} k|X[k]|}{\sum_{k=0}^{N-1} |X[k]|} \cdot \frac{F_S}{N}$$

In the case of harmonic sounds, brightness is related to the fundamental frequency $F0$:

$$BR_{F0} = \frac{\sum_{i=1}^{I} i\ a_i}{\sum_{i=1}^{I} a_i} = \sum_{i=1}^{I} i\ w_i$$

**Bandwidth** is the difference between the lowest and highest frequency components of a signal:

$$BW = \frac{\sum_{k=0}^{N-1} |X[k]| \cdot |f_k - BR|}{\sum_{k=0}^{N-1} |X[k]|}$$

**Spectral spread** [In Mpeg7 is AudioSpectrumSpread]" is the spread of the spectrum around its mean value, i.e. the variance of the spectral distribution.

**Spectral skewness** gives a measure of the asymmetry of a distribution around its mean value.

**Spectral slope** represents the amount of decreasing of the spectral amplitude. It is computed by linear regression of the spectral amplitude.

$$Stilt = \frac{1}{\sum_{i=1}^{I} t_i^2} \cdot \sum_{i=1}^{I} \frac{t_i a_i}{w_i}$$

where

$$t_i = \frac{1}{w_i}\left(f_i - \frac{\sum_{i=1}^{I} f_i/w_i^2}{\sum_{i=1}^{I} 1/w_i^2}\right)$$

**Spectral decrease** still represents the decreasing of spectral amplitude, and it is supposed to be more correlated to human perception.

$$SD = \frac{1}{\sum_{i=2}^{I} a[i]} \sum_{i=2}^{I} \frac{a[i] - a[1]}{i-1}$$

**Spectral roll-off** is the frequency $R_s$ so that 85% of the amplitude distribution is below this frequency. It is correlated to harmonic/noise cutting frequency.

$$\sum_{k=1}^{R_s} |X[k]| = 0.85 \cdot \sum_{k=1}^{N-1} |X[k]|$$

**Spectral Flux** is defined as the Euclidean distance between two amplitude spectrums of two close frames.

$$SF = \sum_{k=1}^{N-1} [N_t[k] - N_{t-1}[k]]^2$$

where $N_t[k]$ and $N_{t-1}[k]$ are respectively the spectral amplitude of frame FFT at instants $t$ and $t-1$.

#### 6.1.3.3 Harmonic features

**Fundamental frequency** [In Mpeg7 is AudioFundamentalFrequency]" is the frequency that a periodic waveform repeats itself. There are many methods in the literature to detect the fundamental frequency $F0$. For the Mpeg7 descriptor it is computed using the maximum likelihood algorithm. The method used in our experiments will be described in section **??**.

**Noisiness** [In Mpeg7 is AudioHarmonicity]" is the ratio between the energy of the noise an the total energy. It is close to zero for purely harmonic sounds.

$$Noisiness = \frac{\sum_{n=0}^{M-1} |x_R[n]|}{\sum_{n=0}^{M-1} |x[n]|}$$

**Inharmonicity** represents the divergence of the signal spectral components from a purely harmonic signal. It is computed as an energy weighted divergence of the spectral components. The range is [0,1].

$$HD = \sum_{i=1}^{I} |f_i - iF_o| \cdot w_i$$

**Harmonic Spectral Deviation** [In Mpeg7 is HarmonicSpectralDeviation]" is the deviation of the amplitude harmonic peaks from the global envelope.

$$HDEV = \frac{1}{I}\sum_{i=1}^{I}[\ a_i - spec\_env(f_i)\ ]$$

where $spec\_env(f_i)$ is the smoothed spectral envelope computed at frequency $f_i$ of $i$-th harmonic.

**Even-odds energy** is useful to distinguish sounds like clarinet sound, which has low energy on even harmonics, differing from the trumpet sound which has similar behavior for both kinds of harmonics.

$$OER = \frac{\sum_{i=even} a_i^2}{\sum_{i=odd} a_i^2}$$

#### 6.1.3.4 Pitch detection from the spectrum

For an harmonic signal F0 is the frequency so that its integer multiple explain the content of the digital spectrum. In fact for a periodic sound, all the partials are multiple of the fundamental frequency, that is $f_i = iF0$. In real sounds this is not completely true, and F0 can be calculated as the weighted sum of frequencies, normalized over all the harmonics.

$$F0 = \sum_{i=1}^{I} \frac{f_i}{i} \cdot w_i \tag{6.18}$$

where

$$w_i = \frac{a_i}{\sum_{i=1}^{l} a_i} \tag{6.19}$$

is the weight of i-th harmonic, with reference to the total sinusoidal component, and $a_i$ is its amplitude. For signals with a more distributed spectrum, cepstrum analysis (sect. 6.2.2) is the form more conventionally used to make the analysis of pitch.

### 6.1.4 Perceptual features

**Specific Loudness** is associated to each Bark band $z$, see Moore et al. [1997] for precise definitions. It can be simply defined as

$$N'(z) = E(z)^{0.23}$$

where $E(z)$ is the energy in the $z$-th bark-band.

**Total Loudness** is the sum of individual loudness.

$$N = \sum_{z=1}^{band} N'(z)$$

**Sharpness** is the perceptual equivalent to the spectral centroid, computed through the specific loudness of he Bark bands.

$$Sh = 0.11 \cdot \frac{\sum_{z=1}^{band} z \cdot g(z) \cdot N'(z)}{N}$$

where z is the index of the band and g(z) is a function defined as follows:

$$g(z) = \begin{cases} 1 & \text{for } z < 15 \\ 0.066 \exp(0.171 z) & \text{for } z \geq 15 \end{cases} \tag{6.20}$$

## 6.2 Spectral Envelope estimation

Families of musical instruments can often be described by typical spectral envelope. When processing sound, operations that preserves the spectral envelope are roughly expressed as *pith shifting* operations with timbre preservations. Various techniques are used to represent the shape of a stationary spectrum. In sect. 2.4.7.2 we already presented the Linear Predictive (LPC) analysis.

**M-6.7**

In LPC analysis, the position of formants (resonances) is related to the poles of the estimated transfer function. Factorize the denominator of the transfer function and estimate the frequency of the formants. Note that if $\theta_k$ is the argument of $z_k$ complex conjugate zero of the denominator, then its corresponding resonant frequency $f_k$ derives from $\theta_k = 2\pi f_k / F_s$; the formant bandwidth $B_k$ is related to the zero modulus by $|z_k| = \exp(-\pi B / F_s)$.

### 6.2.1 Filterbank

Filter-bank is a classical spectral analysis technique which consists in representing the signal spectrum by the log-energies at the output of a filter-bank, where the filters are overlapping band-pass filters spread along the frequency axis. This representation gives a rough approximation of the signal spectral shape while smoothing out the harmonic structure if any. When using variable resolution analysis, the central frequencies of the filters are determined so as to be evenly spread on the warped axis and all filters share the same bandwidth on the warped axis.

**M-6.8**

Write a MATLAB function for the spectral envelope computing, with the filterbank approach. Try a filterbank of frequency linearly spaced filters and logarithmic spaced filters (e.g. third octave filters).

**M-6.9**

Write a MATLAB function for the spectral envelope computing, with the gamma tone filterbank approach. Look in the literature or on the web for gammatone filter definition. gamma tone filters simulate the behaviour of the cochlea.

### 6.2.2 Spectral Envelope and Pitch estimation via Cepstrum

Families of musical instruments can often be described by typical spectral envelope. When processing sound, operations that preserves the spectral envelope are roughly expressed as *pith shifting* operations with timbre preservations. Various techniques are used to represent the shape of a stationary spectrum. Cepstrum method allows the separation of a signal $y[n] = x[n] * h[n]$, (source-filter model), where

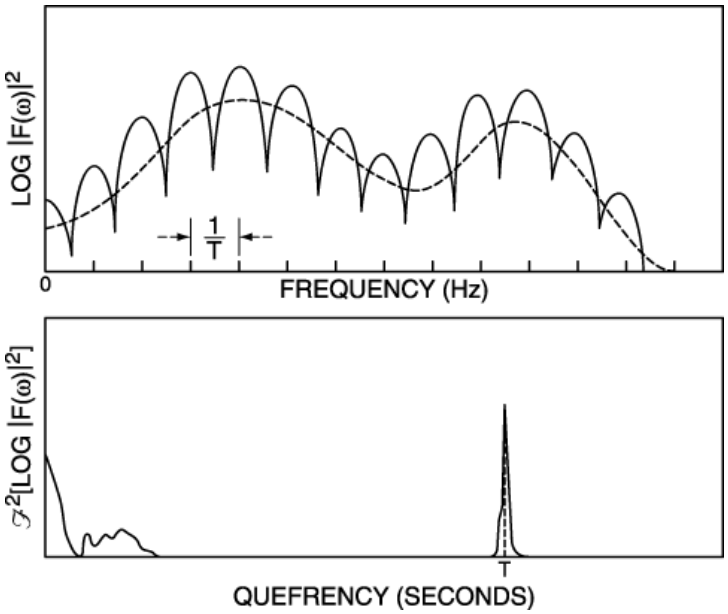



**Figure 6.11:** *Example of cepstrum: on the top* $\log |Y[k]|^2$*; below, the related cepstrum* $c[n] = DFT^{-1}(\log |Y[k]|)$

the source $x[n]$ passes through a filtered described by impulse response $h[n]$. Signal spectrum $y[n]$ results $Y[k] = X[k] \cdot H[k]$, which is the product of two spectrums ($k$ is the discrete-frequencies index). The former is related to the source spectrum, and the latter to the filter spectrum. It's pretty difficult to separate these two spectrums, thus what is usually done is to extract the envelope (real) of the filter, and making the phase related to the source only. Cesptrum idea is based on the properties of logarithms : $\log(a \cdot b) = \log(a) + \log(b)$. Taking into account the logarithm of the absolute value of spectrum $Y[k]$, we get:

$$\log |Y[k]| = \log(|X[k] \cdot H[k]|) = \log |X[k]| + \log |H[k]| \tag{6.21}$$

If we consider the diagram for $\log |Y[k]|$ as a time-domain signal, we can distinguish two components: a quick oscillation, due to harmonic structure (rows), and a slower behavior related to the filter resonances (spectral envelope). We can separate the components by low/high-pass filtering signal $\log |Y[k]|$ (see Fig. 6.11, top). For components separation, one method is based on IFFT:

$$\text{DFT}^{-1}(\log |Y[k]|) = \text{DFT}^{-1}(\log |X[k]|) + \text{DFT}^{-1}(\log |H[k]|) \tag{6.22}$$

The part of $\text{DFT}^{-1}(\log |Y[k]|)$ towards the origin describes the spectral envelope, far from excitation. There is a sort of line in correspondence with $\log |Y[k]|$ periodicity, and thus to sound periodicity (see Fig. 6.11, below). At this point the cepstrum name origin comes out. Cepstrum word corresponds to spectrum when backward reading the former (ceps) part. The pitch can be estimated by the following procedure:

compute cepstrum every 10-20 msec
search for periodicity peak in expected range of $n_p$
**if** found and above threshold
    sound is periodic

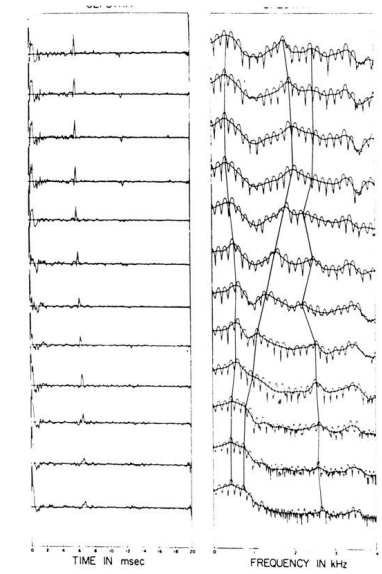



Fig. 7.15 Automatic formant estimation from cepstrally smoothed log spectra. (After Schafer and Rabiner [11].)

**Figure 6.12:** *Automatically formant estimation from cepstrally smoothed log Spectra [from Schaefer Rabiner].*

pitch=location of cepstral peak
**else** sound is not periodic

A drawback of the cepstral coefficients is the linear frequency scale. Perceptually, the frequency ranges 100-200Hz and 10kHz -20kHz should be approximately equally important, and standard cepstral coefficients do not take this into account.

We can notice that the maxima of spectral envelope corresponds to resonances (formats) which are very important to differentiate the vowels. In Fig. 6.12 it is shown how to individuate the formants from the spectral envelope.

**M-6.10**

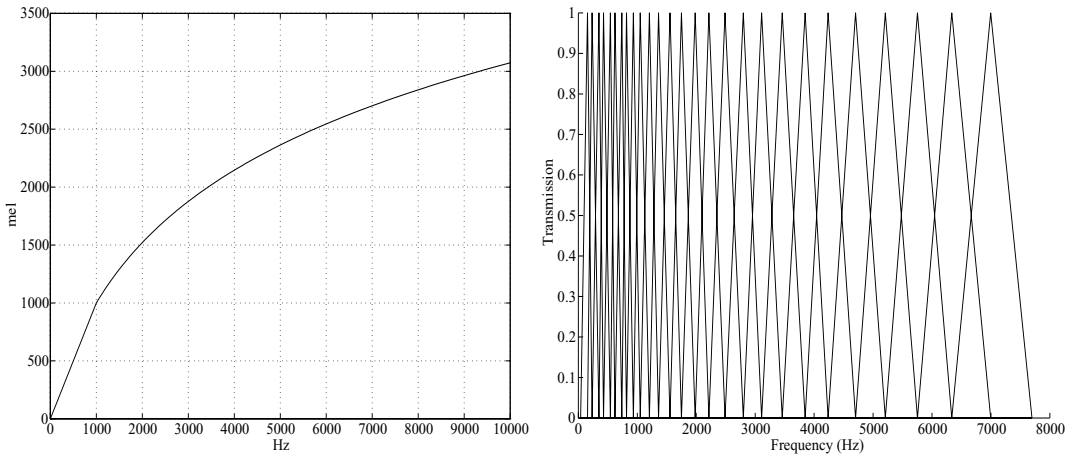
Estimate the formants of a voice in a song and plot their position on the spectrogram.

### 6.2.3 Analysis via mel-cepstrum

Psychoacoustic studies have shown that human perception of frequencies goes with a logarithmic-like scale. Each tone of f (Hz), corresponds to a subjective value of pitch measured on the *m*el scale. As reference on the mel scale, 1000 Hz match 1000 mel. To obtain a value in the mel scale, a non-linear transformation on frequency scale is applied (see Fig. 6.13(a)), computed as follows:

$$\text{mel}(f) = \begin{cases} f & \text{if } f \leq 1 \text{ kHz} \\ 2595 \log_{10}\left(1 + \frac{f}{700}\right) & \text{if } f > 1 \text{ kHz} \end{cases} \tag{6.23}$$

To apply the mel scale to cepstrum, triangular band-pass filterbanks are used, with central frequency in $K$ mel values (see Fig. 6.13(b)). Each filter has bandwidth equal to the distance to previous filter central frequency, multiplied by two. First filter starts from 0. Thus, the bandwidth of filters




(a) (b)

**Figure 6.13:** *(a) Transformation from Hz to mel. (b) Mel-scale filterbank.*

below 1000 Hz is 200 Hz; then it will raise exponentially. Mel-cesptrum aim to estimate the spectral envelope of this filterbank output.

When $Y_n$ is the logarithm of energy exiting from channel $n$, we can use the discrete time cosine transform DCT to obtain the mel-cepstral coefficients MFCC (mel frequency cepstral coefficients) by means of following equation:

$$c_k = \sum_{n=1}^{N} Y_n \cos\left[k\left(n-\frac{1}{2}\right)\frac{\pi}{N}\right] \qquad k = 0,\ldots,K \tag{6.24}$$

We can use the first $K_m$ (with $K_m < K$) coefficients to draw a simplified spectral envelope $\tilde{C}(mel)$, similar to what we have seen for cepstrum:

$$\tilde{C}(mel) = \sum_{k=1}^{K_m} c_k \cos(2\pi k \frac{mel}{B_m}) \tag{6.25}$$

where $B_m =$ is the bandwidth, expressed in mel. A typical value for $K_m$ in music classification is $K_m = 20$. We can notice that the coefficient $c_0$ is the mean value (in dB) of the energy in the channel of the filterbank. Thus it is related to the signal energy and often is not considered when we want to compare two spectral envelopes.

**M-6.11**

Write a MATLAB function for the spectral envelope computing, with the mel-cepstral approach and experiment it for different kinds of sounds. Compare the results obtained with the different spectral envelope algorithms.

In fig. 6.14 an example of mel-cesptrum analysis of a clarinet tone is shown. Spectra in dB, represented on a logarithmic frequency scale are compared: tone spectrum (high left); spectral envelope reconstructed with first 6 mel cepstral coefficients (low right), spectral envelope rebuilt from LPC analysis (low left); spectral envelope estimated with all mel cepstrum coefficients (low right).

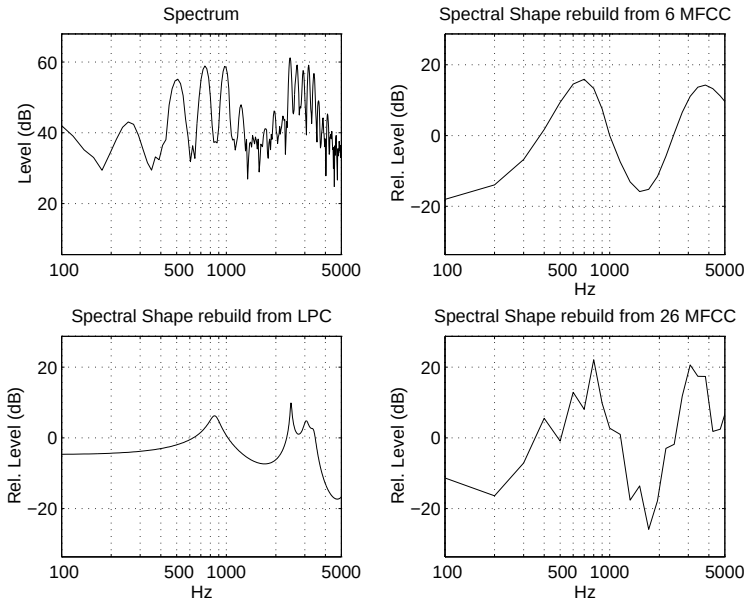



**Figure 6.14:** *Example of mel-cesptrum analysis of a clarinet tone: tone spectrum (high left); spectral envelope reconstructed with first 6 mel cepstral coefficients (low right), spectral envelope rebuilt from LPC analysis (low left); spectral envelope estimated with all mel cepstrum coefficients (low right).*

## 6.3 Mid-level features

### 6.3.1 Chromagram

The chromagram, also called Harmonic Pitch Class Profile, shows the distribution of energy along the pitches or pitch classes. It is computed in two steps:

- First, the spectrum is computed in the logarithmic scale, with selection of the 20 highest dB, and restriction to a certain frequency range that covers an integer number of octaves.
- Then, the spectrum energy is redistributed along the different pitches (i.e., chromas).

The chromagram can be represented along the full pitch scale (Figure 6.15) or can be wrapped along the 12 pitch classes (Figure 6.16).

Calculating the chromagram on a framed audio signal, it is possible to represent the evolution in time of the spectral energy along the pitch classes (Figure 6.17).

### 6.3.2 Keystrength

The Keystrength is a score between -1 and +1, associated with each possible key candidate. In other words, this feature allows an estimation of the prevalent key and mode of a musical excerpt. The Keystrength can be computed through a cross-correlation of the chromagram, wrapped and normalized, with similar profiles representing all the possible tonality candidates (Figure 6.18). These profiles have been estimated in previous studies, such as Krumhansl, 1990[1] and Gomez, 2006[2].

The resulting graph (Figure 6.19) indicate the cross-correlation score for each different tonality candidate.

[1] Krumhansl, C. L., Cognitive foundations of musical pitch. Oxford UP, 1990.

[2] Gomez, E., Tonal description of music audio signal. Phd thesis, Universitat Pompeu Fabra, Barcelona, 2006.

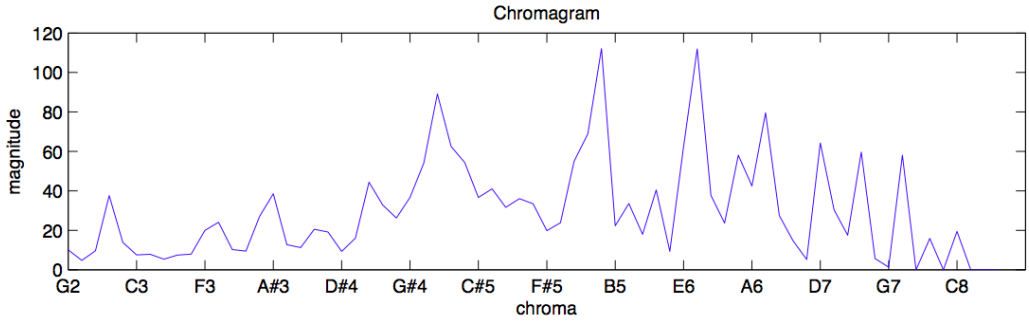



**Figure 6.15:** *Chromagram in the pitch range G2 - C8.*

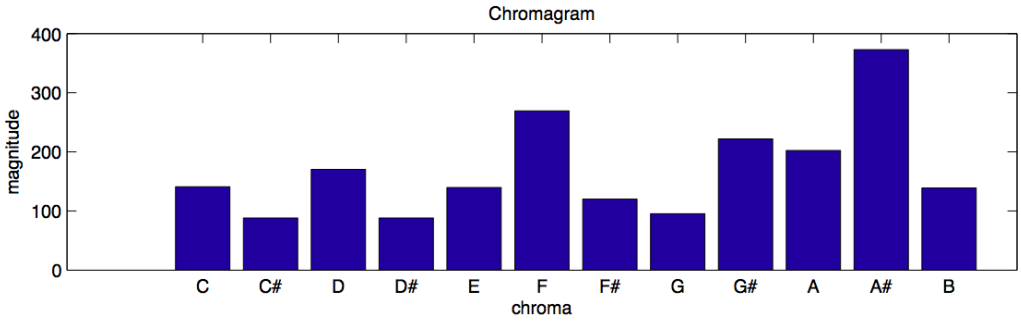



**Figure 6.16:** *Wrapped chromagram.*

### 6.3.3 Tempo

The tempo can be estimated by detecting periodicities from the onset detection curve. Various approaches can be followed:

- computing an autocorrelation function of the onset detection curve (Figure 6.20);
- computing a spectral decomposition of the onset detection curve;
- combining both strategies: the autocorrelation function is translated into the frequency domain in order to be compared to the spectrum curve, and the two curves are subsequently multiplied.

Then a peak picking is applied to the autocorrelation function or to the spectrum representation (Figure 6.21).

## 6.4 Onset Detection

This section is dedicated to the problem of the segmentation of the audio signal, through the onset detection. The concept of onset can be defined as the instant in which a new event starts. Event, in this context, can be defined as an auditory phenomenon that shows continuity inside the normal limits of perception. These auditory phenomenons can be expressive features (legato, vibrato, etc), timbre or notes. Note onset detection and localization is useful in a number of analysis and indexing techniques for musical signals. In most cases, onset will coincide with the start of the transient of the note, or the earliest time at which the transient can be reliably detected.

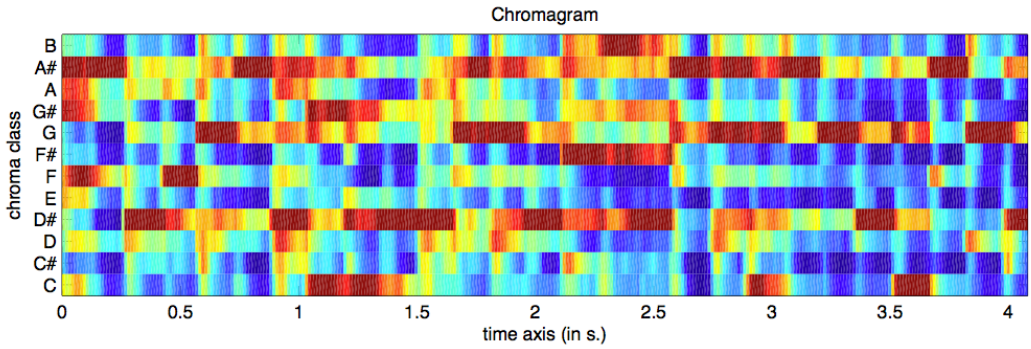



**Figure 6.17:** *Chromagram of a framed signal.*

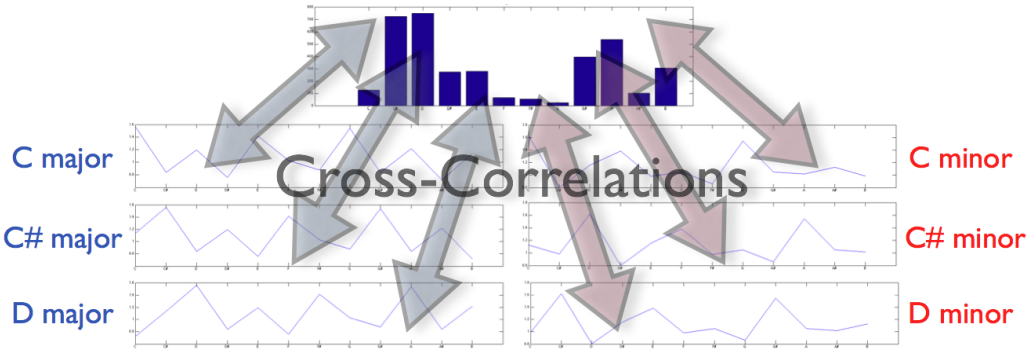



**Figure 6.18:** *The Chromagram is compared with the profiles related to the different tonality candidate.*

First two methods widely used for the onset detection, based in frequency domain and local energy detection, will be presented. Then in section 6.4.3 a more complete method will be presented.

### 6.4.1 Onset detection in frequency domain

In the spectral domain, energy increases linked to transients tend to appear as a broadband event. Since the energy of the signal is usually concentrated at low frequencies, changes due to transients are more noticeable at high frequencies This attack transient noise is particularly noticeable at high frequency locations, since at low frequencies, high concentrations of energy (in the bins corresponding to the first few harmonics of the played note) mask this effect. The High Frequency Content (HFC) function, is defined, for the $j$th frame, by:

$$D_H[j] = \sum_k k|X_j[k]|$$

where $|X_j(.)|$ is the spectral magnitude of the $j$th frame. Aim of this function is to emphasize the high frequency content of the sound and it works well for identifying percussive sounds. If compared with energy, this HFC function has greater amplitude during the transient/attack time. The HFC function produces sharp peaks during attack transients and is notably successful when faced with percussive onsets, where transients are well modeled as bursts of white noise.

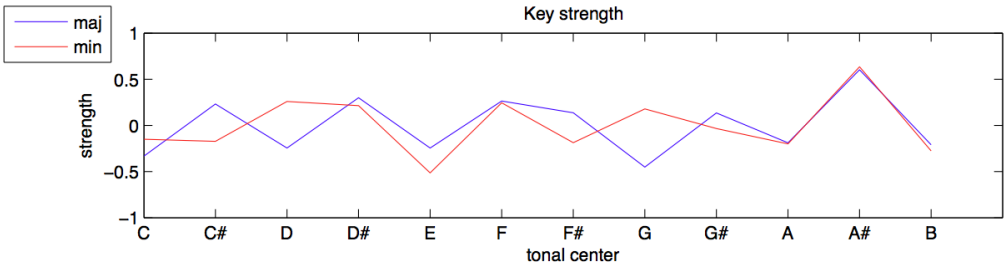



**Figure 6.19:** *Keystrength camputed on a musical excerpt.*

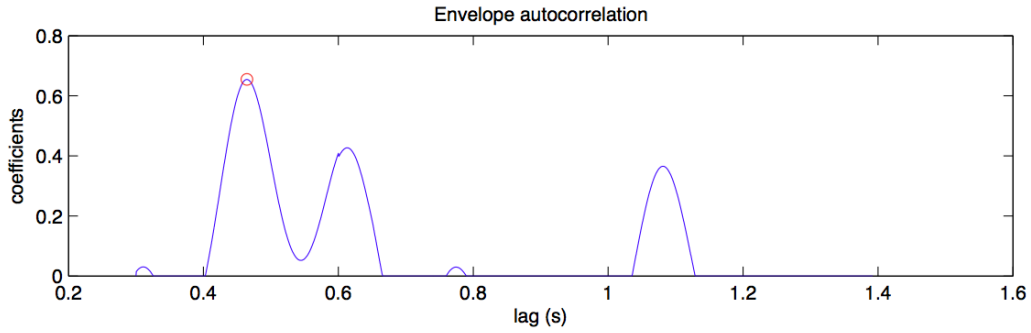



**Figure 6.20:** *Autocorrelation of the onset detection curve.*

### 6.4.2 Onset detection from Local Energy

In order to find onsets, a detection function is cimpuyted, i.e. an intermediate signal that reflects, in a simplified form, the local structure of the suond and manifests the occurrence of transients in the original signal.

Despite the number of variants, practically all time domain methods are based on the calculation of a first order difference function of the signal amplitude envelopes and taking the maximum rising slope as an onset or an onset component. The envelope of the signal is computed as explained in sect. 6.1.2.1.

A common approach is to use as detection function $D(t)$ the time derivative of the energy

$$D(t) = \frac{dE(t)}{dt}$$

(or rather the first difference for discrete-time signals) so that sudden rises in energy are transformed into narrow peaks in the derivative. An example is the algorithm based on the surfboard method of Schloss [1985], which involves smoothing the signal to produce an amplitude envelope and finding peaks in its slope using linear regression.

In Fig. 6.22 the effect of a simple onset detector based on Local energy is shown. In Fig. 6.22(a) the time-domain audio signal; in Fig. 6.22(b) its smoothed amplitude envelope drawn in bold over it, computed by a 40ms windowed RMS smoothing with 75% overlap and in Fig. 6.22(c) peaks in slope shown by dotted lines tangential to the envelope. This method is lossy, in that it fails to detect the

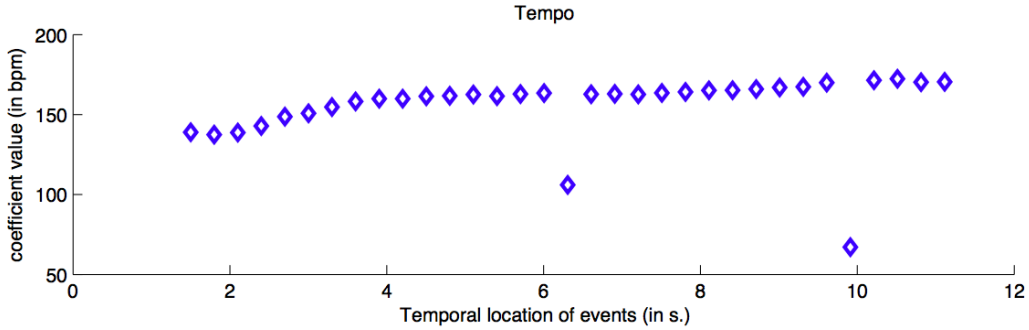
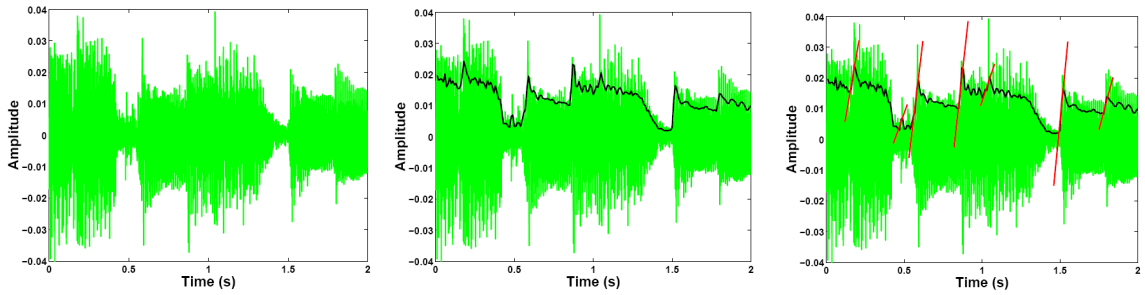



**Figure 6.21:** *Tempo estimation of a framed audio signal.*




(a) (b) (c)

**Figure 6.22:** *Example of onset detector based on local energy: time-domain audio signal (a), 40ms windowed RMS smoothing with 75% overlap (b), peaks in slope of envelope (c).*

onsets of many notes which are masked by simultaneously sounding notes. Occasional false onsets are detected, such as those caused by amplitude modulation in the signal.

When we take into account perceptual aspects, we may notice that psychoacoustics indicates that loudness is perceived logarithmically. This means that changes in loudness are judged relative to the overall loudness, since, for a continuous time signal,

$$D_r(t) = \frac{d(\log E(t)}{dt} = \frac{1}{E(t)} \frac{dE(t)}{dt}$$

Hence, computing the first-difference of $\log(E[n]$ roughly simulates the ears perception of loudness. The relative difference function $D_r(t)$ as detection function effectively solves the problems of low sound, where the amplitude grows slowly, by detecting the onset times ealier and, more importantly, by handling complicated onset tracks, since oscillations in the onset track of a sound do not matter in relative terms after its amplitude has started rising.

### 6.4.3 Combining pitch and local energy information

In this section a case study on how to combine pitch and energy information for onset detection is presented. The analysis of pitch variation can help in finding tone onsets. Figure 6.23 shows the progression of pitch along five seconds of a signal (Handel's Flute Sonata in E minor (Adagio) -

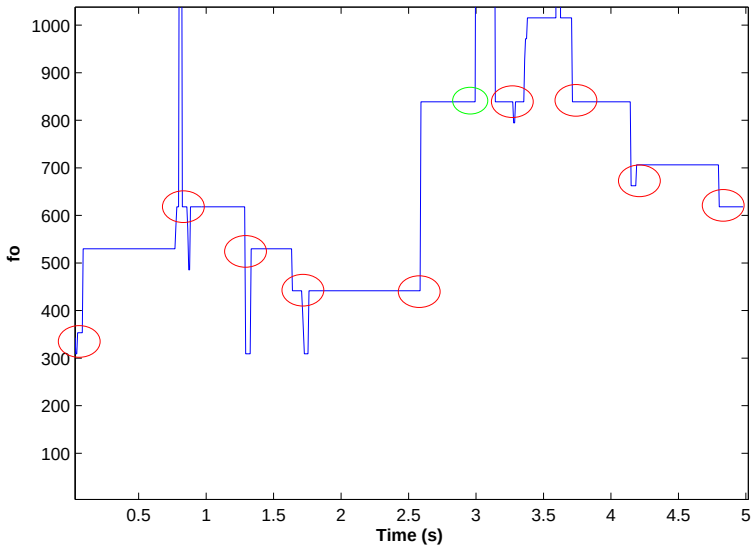



**Figure 6.23:** *$F0$ temporal variation (with a window and one step with the values above considered) calculated by standard spectrum for the 5 first seconds of a considered signal. In this type of study it is verified the variation of the spectral shape in respect to the first maximum of the $FFT$.*

"hard" performed): The red and the green circles show the considered spectral variation zones for the onsets detection. The red circles indicate the zones of effective occurrence of onsets, while that the circles the green indicate the zones where onsets had not occurred.

As we will see in the following section, this type of analysis can be complemented with an analysis of the signal envelope (that it equivalent to a study of the variation of the energy). It can be said that main objectives for that are:

a) The elimination of false detections (circumscribed zones from the green circles), when the spectral disturbances are not follow for a minimum variation of energy (given by a minimum considered limit the note attack)

b) Add great variations of energy, still that are not verified significant spectral disturbances. This election is also made with base in a threshold of variation of energy in the note attack.

c) If there is a set of possible onsets in an interval of maximum distance, it is only considered the onset that corresponds to the lowest energy. All the others possibilities are discard.

**Analysis of the temporal envelope** In fig. 6.24 we see the temporal variation of the RMS envelope along the first 5 seconds of the considered signal. Instead of using a constant threshold that detected the brusque variations of the derivative of the $RMS$ signal, we arrive at the conclusion that one adaptative threshold that follow the signal according to a more global dynamic, through the calculation of the average in a certain neighborhood of each point of the $RMS$, results much more efficient. The average was calculated between the 200 samples that surround each instant (1 second, for a step=0.005 s).The process consists in searching the minimum of $RMS$ between two consecutive values detected by the threshold, since these values define a valley between two peaks.

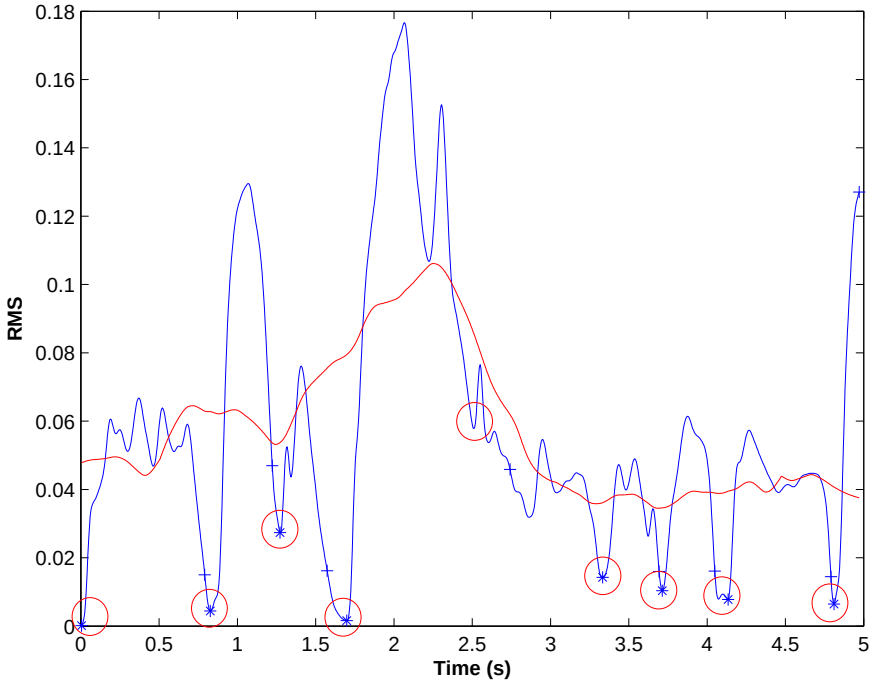



**Figure 6.24:** *envelope of a 5 second window of signal.The blue line represents the $RMS$ temporal variation, while that the red line represents the dynamic threshold that follows the $RMS$ of the signal. The crosses on signal $RMS$ represent the detected onsets. The circles represent the zones of effective onsets.*

**Hybrid approach RMS-Frequency** The analysis of the behavior of pitch and $RMS$ envelope can be combined in the following steps.

1) We calculate all the onsets that result of the envelope analysis

2) In the case of the occurrence of onsets detected too much closed in the time, inside a given limit, is considered only in this interval the onset that has a lesser value of $RMS$. This limit consists of 0.25 of the mean of the measured distances between all detected onsets

3) We have considered that would be false detections (false positives) onsets that don't have a note attack in the energy domain greater than a given limit. We have eliminated to the list of onsets, that we have until this moment, the set of onsets that are below of this threshold. This threshold is calculated in relation to the average of the attacks of onsets considered until 2). Is considered the attack in the positive temporal direction and in the negative temporal direction. That is, the attack is defined as the energy jump that occurs between an onset and the next peak, and is defined also as the energy interval between an onset and the immediately previous peak. This threshold consists of 0.3 of the average of the considered attacks.

4) We calculate all onsets that result of spectral disturbances, that in our case can be seen as disturbances in the temporal progression of pitch.

5) As in 2), in the case of the occurrence of onsets too much closed in time, inside of a given threshold, is considered only the onset that has a lesser value of $RMS$. This threshold is the same that was considered in 2).

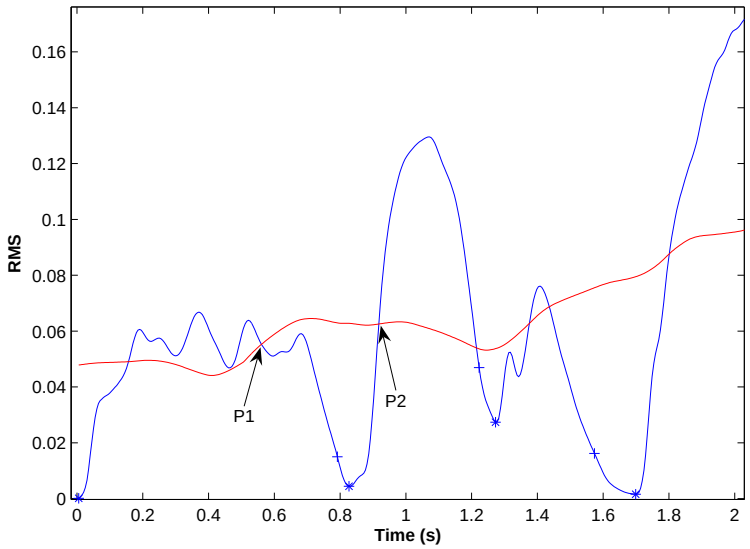



**Figure 6.25:** *The points P1 and P2 are two values of the $RMS$ signal detected by threshold. These values define a valley between two peaks. The value for the onset is the minimum that can be find between these two points.*

6) Are considered here as valid onsets the onsets considered until 3) (valid onsets extracted by $RMS$) that are inside a given threshold of proximity in relation to the more closed onset in the set of onsets considered in 5) (onsets valid extracted by disturbances of pitch). This threshold of proximity corresponds to a maximum absolute distance of 0.1 seconds.

7) Of the onsets calculated by $RMS$ analysis considered valid - before the onsets elimination according to the previous criterion (i.e. the set considered in 3)) we add to the list the onsets that had a note attack superior than a given threshold in the energy domain. The attack is here defined as the difference between the $RMS$ value for the instant of the onset and the value of the next peak (that can be found between this onsets and the consecutive one). This limit is calculated relatively to the mean of all the considered attacks of onsets considered until 6). Numerically corresponds to 0,5 of the mean of these attacks.

## 6.5 Feature Selection

In many applications, reducing the dimensionality of the data by selecting a subset of the original variables may be advantageous for reasons including the expense of making, storing and processing measurements. The art of machine learning starts with the design of appropriate data representations. Better performance is often achieved using features derived from the original input. The reasons for reducing the number of features to a sufficient minimum are that: (i) less features implies simpler models and then less training data needed; (ii) the higher the ratio of the number of training patterns N to the number of free classifier parameters, the better the generalisation properties of the resulting classifier; (iii) computational complexity, correlating (redundant) features.

Good features result in large between-class distance and small within-class variance; to perform this, the approaches divide according to: (i) examine features individually and discard those with

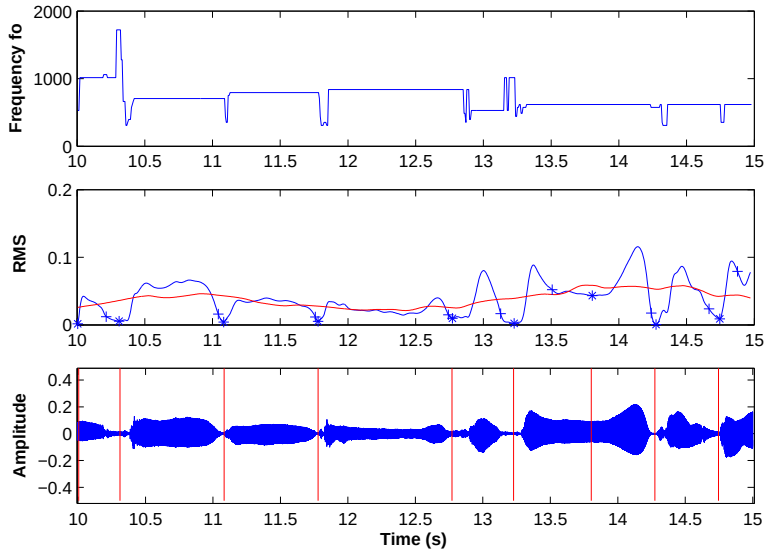



**Figure 6.26:** *Detection of onsets for a five seconds window: the last figure represents the onset detection, where red lines indicate the instants for each detected onset.*

little discrimination ability; (ii) to examine the features in combinations; (iii) linear (or nonlinear) transformation of the feature vector. Conventional statistical approaches such as comparing mean correlations within and between subgroups, principal components analysis (PCA), analysis of variance (ANOVA), and visualization techniques can be appropriately applied to determine which features to select. These techniques allow us to discover which data is redundant, which features have a relatively high variation between the subjects, and how the original feature set can be reduced without a big loss in the variance explained and recognition rate.

Variable subset selection procedure consists on measuring the relevance of a subset of input variables, and an optimization algorithm for searching for the optimal or a near-optimal subset with respect to the subset of variables. Procedures for variable subset selection can be classified into two groups: filter procedures and wrapper procedures. In case of *filter* procedures, the relevance measure is defined independently from the learning algorithm. The subset selection procedure in this case can be seen as a preprocessing step. In case of *wrapper* procedures, the relevance measure is directly defined from the learning algorithm, for example, in terms of the cost of the learning and the precision achieved by classification algorithm. On the other side, wrapper procedures need the number of possible parameters to be as low as possible, so that the algorithm should be highly computationally efficient.

### 6.5.1 One-way ANOVA

The analysis of variance (ANOVA) helps to identify the features which highlight differences between groups (subjects).

Let a database contain records of $g$ subjects (number of groups) and $n_l$ sessions for each subject. Let $X_{l_j}$, where $l$ = 1, 2, ..., $g$ and $j$ = 1, 2, ..., $n_l$, be a random sample of size $n_l$ from a population with mean $\mu_l$, $l$ = 1, 2, ..., $g$. Anova is used to investigate whether the population mean vectors are the same, i.e. the null hypothesis of equality of means could be formulated as $H0: \mu_1 = \mu_2 = ... = \mu_g$,

and if not, which implies components differ significantly. The *F*-test rejects *H*0 at level $\alpha$ if:

$$F = \frac{SSA/(g-1)}{SSW/(\sum_{l=1}^{g} n_l - g)} > F_{g-1,\sum n_l - g}(\alpha) \tag{6.26}$$

where $F_{g-1,\sum n_l - g}(\alpha)$ is the upper $(100\alpha)$th percentile of the $F$-distribution with $g-1$ and $\sum n_l - g$ degrees of freedom; $SSA = \sum_{l=1}^{g} n_l(\overline{x_l} - \overline{x})^2$ is the sum of squares among groups, where $\overline{x_l}$ and $\overline{x}$ are estimates of group and overall sample means respectively; $SSW = \sum_{l=1}^{g}\sum_{j=1}^{n_l}(x_{l_j} - \overline{x_l})^2$ is the total within-group sum of squares, where $x_{l_j}$ is an observation of the feature from subject *l*, session *j*.

### 6.5.2 Principal Component Analysis (PCA)

Principal component analysis (PCA) is concerned with explaining the variance-covariance structure of a set of variables through a few linear combinations of these variables. This means that the original feature space is transformed by applying e.g. a linear transformation via a PCA. PCA is a linear transformation that transforms the data to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on. PCA can be used for dimensionality reduction in a dataset while retaining those characteristics of the dataset that contribute most to its variance, by keeping lower-order principal components and ignoring higher-order ones. Such low-order components often contain the "most important" aspects of the data.

Given a set of *n* observations on *p* observed variables, the purpose of PCA is to determine *r* new variables, where *r* is small relative to *p*. The *r* new variables, called principal components, must together account for most of the variation in the *p* original variables. Principal component analysis operates by replacing the original data matrix $\mathbf{X}$ by an estimate composed of the product of two matrices. In matrix notation the approximation $\widehat{\mathbf{X}}$ of the matrix $\mathbf{X}$ is given by $\widehat{\mathbf{X}} = \mathbf{Z}\mathbf{V}^{'}$, where $\mathbf{Z}$ is the $(n \times r)$ matrix of observations on the first *r* principal components, and $\mathbf{V}(p \times r)$ is the matrix whose columns are the first *r* eigenvectors of $\mathbf{X}^{'}\mathbf{X}$. The matrix $\mathbf{Z}^{'}\mathbf{Z} = \Lambda$, where $\Lambda$ is the diagonal matrix of *r* eigenvalues $\lambda_k, k = 1, 2, ..., r$. The sum of squares and cross products matrix for the principal components is therefore a diagonal matrix with diagonal elements $\lambda_k$, that decline in magnitude. The eigenvalues $\lambda_k$ are given by:

$$\sum_{i=1}^{n} z_{ik}^2 = \lambda_k, k = 1, 2, ..., r$$

The sum of squares for each principal component is therefore given by the corresponding eigenvalue. The quantity to be minimised is given by:

$$tr(\mathbf{X} - \widehat{\mathbf{X}})^{'}(\mathbf{X} - \widehat{\mathbf{X}}) = tr\mathbf{X}^{'}\mathbf{X} - \sum_{k=1}^{r} \lambda_k$$

and hence if $r = p$, then this expression has the value zero. Each of the eigenvectors generates a portion of the total variation (or sum of squares) in $\mathbf{X}$ as measured by $tr(\mathbf{X}^{'}\mathbf{X})$. The contribution to $\sum_{k=1}^{p} \lambda_k$ provided by $\mathbf{z_1} = \mathbf{Z}\mathbf{v_1^{'}}$ is $\mathbf{z_1^{'}}\mathbf{z_1} = \lambda_l$. The proportion of the total variance measured by $tr(\mathbf{X}^{'}\mathbf{X})$, accounted for by the component $\mathbf{z_1}$ is given by $\lambda_l / \sum_{k=1}^{p} \lambda_k$. The number of components actually used for the approximation of $\mathbf{X}$ can be guided by the measure $\sum_{k=1}^{l} \lambda_k / \sum_{k=1}^{p} \lambda_k$, where *l* $\leq p$.

In other words, the first principal component is the axis passing through the centroid of the feature vectors that has the maximum variance therefore explains a large part of the underlying feature structure. The next principal component tries to maximize the variance not explained by the first. In this manner, consecutive orthogonal components are extracted. The principal components depend solely on the covariance or correlation matrix of the data.

### 6.5.3 Further feature subset selection

To find the optimal subset of features, we should form all possible combinations of $M$ features out of the $D$ originally available; the best combination is then selected according to any desired class separability measure $J$. In practice, it is not possible to evaluate all the possible feature combinations. Thus the search is for a satisfactory set of features instead of an optimal set and greedy approaches are used.

Sequential backward selection (SBS) consists of choosing a class separability criterion $J$, and calculate its value for the feature vector which consists of all available features (length = $D$). Eliminate one feature, and for each possible resulting combinations (of length $D$-$1$) compute $J$. Select the best, and continue this for the remaining features, and stop when you have obtained the desired dimension $M$. This is a suboptimal search procedure, since nobody can guarantee that the optimal $r$-$1$ dimensional vector has to originate from the optimal $r$-dimensional one. This method is good for discarding a few worst features.

Sequential forward selection (SFS) consists of computing the criterion $J$ for all individual features and select the best. Form all possible two-dimensional vectors that contain the winner from the previous step, calculate the criterion for each vector and select the best; continue adding features one at time, taking always the one that results in the largest value of the criterion $J$, and stop when the desired vector dimension $M$ is reached. This method is particularly suitable for finding a few good features.

Both SBS and SFS suffer from the nesting effect: once a feature is discarded in SBS (selected in SFS), it cannot be reconsidered again (discarded in SFS).

## 6.6 Music Information Retrieval: Issues, Problems, and Methodologies

### 6.6.1 Introduction

The core problem of Information Retrieval (IR) is to effectively retrieve documents which convey content being relevant to the user's information needs. Effective and efficient techniques have been developed to index, search, and retrieve documents from collections of hundreds of thousands, or millions of textual items.

The most consolidated results have been obtained for collection of documents and user's queries written in textual form and in English language. Statistical and probabilistic techniques have lead to the most effective results for basic system functions and are currently employed to provide advanced information access functions as well. The content description of media being different from text, and the development of different search functions are necessary steps for content-based access to Digital Libraries (DL). This statement mainly applies to cultural heritage domain, where different media and search functions live together.

In order to provide a content-based multimedia access, the development of new techniques for indexing, searching, and retrieving multimedia documents have recently been the focus of many researchers in IR. The research projects in DLs, and specifically those carried out in cultural heritage

domain, have shown that the integrated management of diverse media - text, audio, image, video - is necessary.

The problem with content-based access to multimedia data is twofold.

- On the one hand, each media requires specific techniques that cannot be directly employed for other media.
- On the other hand, these specific techniques should be integrated whenever different media are present in a individual item.

The core IR techniques based on statistics and probability theory may be more generally employed outside the textual case and within specific non-textual application domains. This is because the underlying models, such as the vector-space and the probabilistic models, are likely to describe fundamental characteristics being shared by different media, languages, and application domains.

#### 6.6.1.1 Digital Music and Digital Libraries

There is an increasing interest towards music stored in digital format, which is witnessed by the widespread diffusion on the Web of standards for audio like MP3. There are a number of reasons to explain such a diffusion of digital music.

- First of all, music is an art form that can be shared by people with different culture because it crosses the barriers of national languages and cultural backgrounds. For example, tonal Western music has passionate followers also in Japan and many persons in Europe are keen on classical Indian music: all of them can enjoy music without the need of a translation, which is normally required for accessing foreign textual works.
- Another reason is that technology for music recording, digitalization, and playback, allows for an access that is almost comparable to the listening of a live performance, at least at the level of audio quality, and the signal to noise ratio is better for digital formats than for many analog formats. This is not the case of other art forms, like painting, sculpture or even photography, for which the digital format is only an approximate representation of the artwork. The access to digitized paintings can be useful for studying the works of a given artist, but cannot substitute the direct interaction with the real world works.
- Moreover, music is an art form that can be both cultivated and popular, and sometimes it is impossible to draw a line between the two, as for jazz or for most of ethnic music.

These reasons, among others, may explain the increasing number of projects involving the creation of music DLs. A music DL allows for, and benefits from, the access by users from all over the world, it helps the preservation of cultural heritage, and it is not tailored only to scholars' or researchers' needs. More in general, as music is one of the most important means of expression, the organization, the integration with other media, and the access to the digitized version of music documents becomes an important multimedia DL component. Yet, music has some peculiarities that have to be taken into account when developing a music DL. In figure 6.27 the architecture of a music information retrieval system is shown.

#### 6.6.1.2 Music Information Retrieval

Specific and effective techniques being capable of indexing and retrieving such multimedia documents as the music ones need to be designed and implemented.

Current approaches to Music Information Retrieval (MIR) are based either on string matching algorithms or textual bibliographic catalogue.

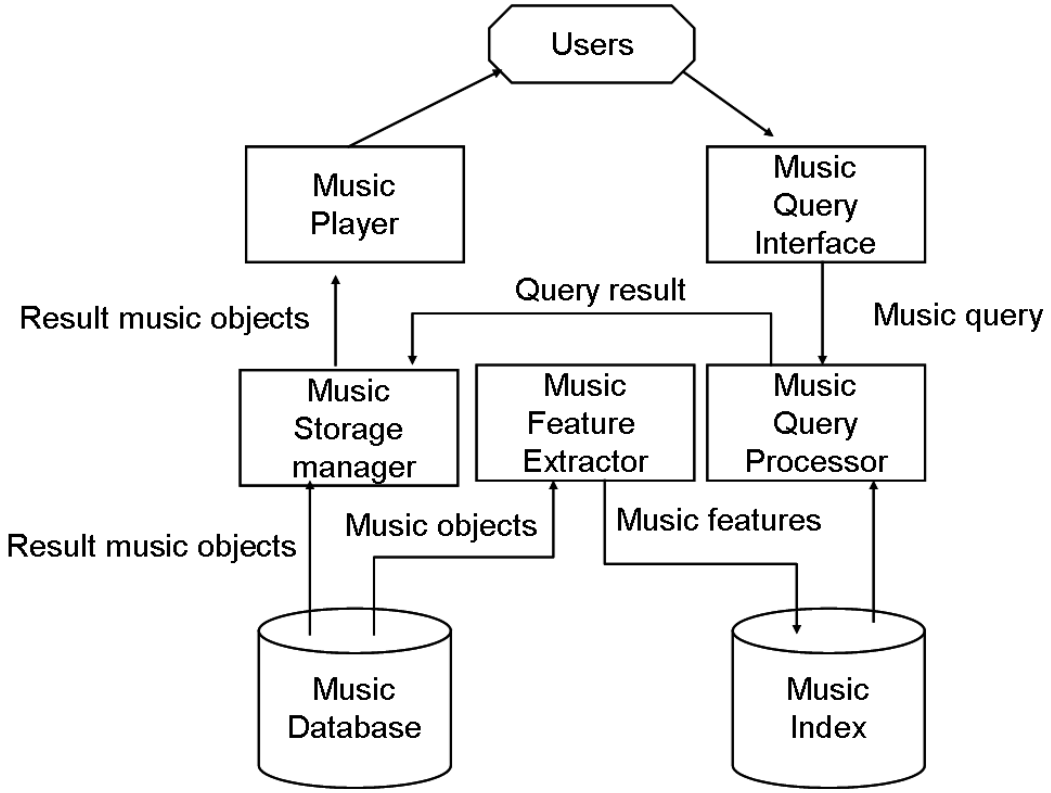



**Figure 6.27:** *Architecture of a music information retrieval system*

- Sting matching approach makes content-based retrieval very difficult - indeed, retrieving textual files using Unix grep-like commands gives poor results.
- Textual bibliographic catalogue approach makes content-based retrieval impossible since the music content cannot be described by bibliographic catalogue.

The requirement for a content-based MIR has been stressed within the research area of music information systems as well. The developments in the representation of music suggest a need for an information retrieval philosophy directed toward non-text searching and eventual expansion to a system that encompasses the full range of information found in multimedia documents. As IR has dealt with the representation and the disclosure of content from its early days, it is natural to think that IR techniques should be investigated to evaluate their application to music retrieval. According to McLane "what has been left out of this discussion, and will no doubt be a topic for future study, is the potential for applying some of the standard principles of text information retrieval to music representations".

- If we follow the hypothesis that the use of standard principles of text information retrieval to index and retrieve music documents is possible, then the design of ad-hoc segmentation algorithms to produce musical 'lexical units' like words in textual documents is required.

  The concept of lexical unit may vary depending on the approach. A lexical unit can be: a fixed-length string, the incipit, a complete theme, a melodic phrase, and so on. Music is a continuous flow of events (e.g., notes, chords, and unpitched percussive sounds) without explicit separators, if not those perceived by listeners. Also music representation lacks of separators of lexical units,

because it conveys information only about macro-events, like changes in tonality or the presence of repetitions. It is therefore necessary to automatically detect the perceived lexical units of a music document to be used like words in textual documents.

- Moreover, content-based MIR requires the design of normalization algorithms. Once detected, musical lexical units occur in documents with many variants like textual words do within textual documents. For example, a melodic pattern may occur in many music works, perhaps composed by different authors, with small deviations of note intervals or timing. Despite these deviations, different patterns may be perceptually similar, hence conveying the same music perception. It is therefore necessary to detect these variants and conflate all the similar musical lexical units into a common stem expressing the same music perception. This conflation process is analogous to the one performed in the textual case for detecting word stems through, for example, the Porter's stemming algorithm.

To allow the integration of automatic music processing techniques with automatic IR techniques, segmentation and normalization algorithms are applied also on music queries.

In a content-based music IR system, users may be able to interact with the system by using the same language, that is the music language. This because content-based MIR requires users to be able of expressing the music document content. The most natural way of express music content is singing and playing music. This approach is often referred to as the query by example paradigm. Therefore, users should be provided with interfaces and search functions so that they can play music and send a music query to the system.

To make content-based music retrieval possible, query content and document content have to be matched: Describing query content is then necessary. If we regard music queries as music documents, segmentation and normalization can be performed also on music queries using the same algorithms used for disclosing document content.

### 6.6.2 Issues of Content-based Music Information Retrieval

Music, in its different representations, can be considered as another medium together with text, image, video, and speech. Nevertheless, there are some issues that make music different from other multimedia IR application domains. The issues we address are form, instantiation, dimension, content, perception, user profile, and formats. The most relevant issues are describes in the following Sections.

#### 6.6.2.1 Peculiarities of the Music Language

The same entity, i.e. a music work, can be represented in two different main forms: the notated and the acoustic form, respectively corresponding to score and performance. Hence the communication in music is performed at two levels:

- the composer translates his intentions in a music structure (music as a composing art),
- the musician translates the written score into sounds (music as a performing art).

Also users may have different needs, in particular the music scholar may look for a given composition, while the melomane may look for a particular performance.

Each music work may have different instantiations. As musicians can interpret scores, the resulting performances may differ and therefore more performances correspond to an individual score. Furthermore, the same music work may be transcribed into different scores, depending on the revisers' choices. As a consequence, different performances and scores may rely to the same music work.

Different dimensions characterize the information conveyed by music. Melody, harmony, rhythm, and structure are dimensions, carried by the written score, that may be all or in part of interest for the final user. In the case of a performance other dimensions should be added, for instance timbre, articulation, and timing. It is likely that the dimensions of interest vary with the level of user's expertise and the specific user's search task. As described in sect. 6.6.2.3, different formats are able to capture only a reduced number of dimensions. Therefore, the choice of a representation format has a direct impact on the degree to which a music retrieval system can describe each dimension.

While text, image, video, or speech-based documents in general convey some information that form their content, it is still unclear what type of content, if any, music works do convey. Let us consider an example: the concept of tempest can be described with a textual document, such as the first chapter of Shakespeare's 'The Tempest', a painting, such as the landscape of Giorgione's 'The Tempest', a video or speech, such as broadcasting news about, for instance, a tornado. All these media are able to convey, among all the other information, the concept of tempest. There are up to forty music works of tonal Western music whose title is related to tempests, among those the most famous probably are Beethoven's Sixth Symphony IV Movement, Rossini's Overture of 'William Tell', and Vivaldi's Concerto 'La Tempesta di Mare'. These works differ in music style, form, key and time signature, and above all the user may be not able to recognize that the work is about a tempest and not just pure music.

In principle, music language does not convey information as, for instance, text or video do. Many composers wrote music to stir up emotions, and in general they aimed to communicate no specific information to the listener. The final user feels emotions on listening to the music, and he interprets some information independently from the composer's and performer's thought and differently from the other users. There is a particular kind of music works, called musica a programma, in which the title (like Vivaldi's 'The Spring') or a lyric (like Debussy's 'Prlude l'aprŁs-midi d'un faune') suggests a meaning to the listener; this sort of textual data would be better managed using a database system rather than a IR system. Moreover in sung music, such as Cantatas, the accompanied text gives the work some meaning, yet that sort of text would require ad-hoc IR techniques to be effectively managed. In general the availability of textual material together with music documents is insufficient.

It is then important to consider how music is perceived and processed by listeners, to highlight which kind of content is carried by this medium. A number of different theories was proposed by musicologists, among which the most popular ones are the Generative Theory of Tonal Music (see Sect. **??**) and the Implication-Realization Model (see Sect. **??**). In both cases it is stated that listeners perceive music as structured and consisting of different basic elements. Therefore, even if music notation and performance lack of explicit separators (like blanks or commas in text) musicians and listeners perceive the presence of small elements which constitute the music work: we can consider these elements as the lexical units for a content-based approach to MIR. It is likely that all the dimensions of music language can be segmented in their lexical units and be used to extract a content from a music document.

#### 6.6.2.2 The Role of the User

As always happens in IR, the effectiveness of techniques does strongly depend on the final user. DL systems does indeed interact with final users of very diverse types and with different levels of expertise in the use of the system itself. This is particularly true for music DLs, because there is a great difference in users' expertise depending on the practice of a musical instrument, the ability of reading a score, the knowledge of harmony rules, the familiarity with composition styles, and so on. Users may have different needs, for instance a music scholar may look on how a given cadenza is

used by different authors, while a melomane may look for a particular performance of a well-known musician. This is a key aspect in the design of a methodology for content-based MIR, because it affects the choice of the dimension to be used for describing a music work, that is which kind of content has to be extracted from it.

Considering that access to DL is widely spread to users of any type, final users of a music DL may not have a deep knowledge of music language. Therefore, melody seems to be the most suitable dimension. In fact, almost everybody can recognize simple melodies and perform them at least by singing or humming. In this case, lexical units can be considered the musical phrases, which may be defined as short excerpts of the melody which constitute a single musical gesture. Moreover, melody carries also explicit information about rhythm and implicit information about harmony.

Melody can be the most suitable evidence for content-based music retrieval, it may however be the case that only a part of the melody can effectively be exploited as useful evidence for music document and query description. This implies that, if phrases can be detected by means of some segmentation algorithms, then it is likely that some of these phrases are 'good' descriptors of the music content from users' point of view, while others can be dropped since they give little contribution to the music content description and may negatively affect efficiency. This latter consideration leads us to contemplating the possibility of building lists of stop phrases, that may be dropped from the index of phrases similarly to the textual case. However, it is still unclear if stop phrases exist how users perceive them. While one can identify a word as stop word because it has no, little, or less meaning than keywords, one cannot identify a phrase as stop phrase because it is very difficult to say what 'phrase meaning' does mean, and frequency-based stop phrase list construction may be a difficult task because, for instance, users may recall melody excerpts just because they are very frequent in a musical genre.

#### 6.6.2.3 Formats of Music Documents

As previously mentioned, the communication in music is achieved at two levels, corresponding to two forms: the composer translates his intentions into a musical structure, that is represented by a music score, and the musician translates the written score into a performance, that is represented by a flow of acoustic events. A number of different digital formats correspond to each form. It can be noted that, as musicians can interpret scores, the resulting performances differ and therefore more than one performance correspond to a single score. Even if the two forms can be considered as instantiations of the same object, they substantially differ in the information that can be manually or automatically extracted from their respective formats.

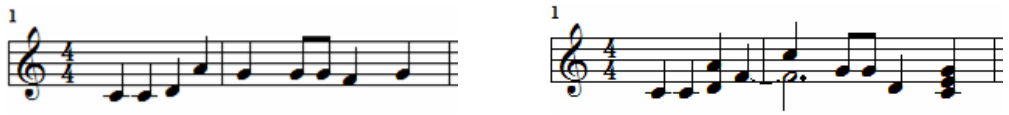
The first problem which arises in the automatic processing of music is then that a music work may be digitally stored in different formats. The same music piece can be represented, for example,

- by a reproduction of the manuscript,
- by a symbolic notation of the score,
- by a sequence of time-stamped events corresponding to pitched and unpitched sounds,
- or by a digital recording of an acoustic performance.

Each format carries different information on the content of the document. For instance, at the state-of-the-art it is impossible to recover informations about the written score from the digital sampling, e.g. stored in a compact disk, of a polyphonic audio signal, and the score carries no information about the timbre, expressive timing and other performing parameters. Hence, the documents format has to be chosen depending on the aims of the DL, which may encompass preservation, displaying, listening, indexing, and retrieval, and so on. As an example, preservation requires high quality audio coding and dissemination over the Internet requires lossy compression.

Formats for digital music documents can be divided in two classes.

- The score is a structured organization of symbols, which correspond to acoustic events; the score is a direct representation of all the dimensions of music (i.e., melody, harmony, and rhythm) and it usually contains all the information that is relevant for classifying and cataloguing: type of movement, time and key signatures, composer's notes, and so on. The symbolic nature of the score allows for an easy representation of its content, and many proposed formats represents score in the form of a textual markup language, for instance ABC and GUIDO.
- The performance is made of a sequence of gestures performed by musicians on their musical instruments; the result is a continuous flow of acoustic waves, which correspond to the vibration induced on musical instruments. Even if all the dimensions of music are embedded in a performance, it requires high-level information processing to recognize them. In particular, only experienced musicians can recognize all the dimensions of music from listening to a performance and, at the state of the art, there is no automatic system that can recognize them from an acoustic recording, apart from trivial cases. The nature of a performance does not allow for an easy representation of its content. The formats adopted to digitally represent performances, such as AIFF (Audio Interchange File Format, proposed by Apple Computers) or MP3 (MPEG1, Layer3), are a plain digital coding of the acoustic sound waves, with a possible data compression.



(a) (b)

**Figure 6.28:** *Example of a melody*

We present now an example of different representations of a melody with reference to fig. 6.28(a). we can represent as absolute or relative values.

- Absolute measure:
  - Absolute pitch: `C5 C5 D5 A5 G5 G5 G5 F5 G5`
  - Absolute duration: `1 1 1 1 1 0.5 0.5 1 1`
  - Absolute pitch and duration:
    `(C5,1)(C5,1)(D5,1)(A5,1)(G5,1)(G5,0.5)(G5,0.5)(F5,1)(G5,1)`
- Relative measure:
  - Contour (in semitones): `0 +2 +7 -2 0 0 -2 +2`
  - IOI (Inter onset interval) ratio: `1 1 1 1 0.5 1 2 1`
  - Contour and IOI ratio:
    `(0,1)(+2,1)(+7,1)(-2,1)(0,0.5)(0,1)(-2,2)(+2,1)`

In a polyphonic case (see fig. 6.28(b)) we can represent in different ways.

- Keep all information of absolute pitch and duration (start_time, pitch, duration)
  `(1,C5,1)(2,C5,1)(3,D5,1)(3,A5,1)(4,F5,4)(5,C6,1)(6,G5,0.5)(6.5,G5,0.5)...`
- Relative note representation: Record difference of start times and contour (ignore duration)
  `(1,0)(1,+2)(0,+7)(1,-4) ...`

- Monophonic reduction, e.g. select one note at every time step (main melody selection)
  `(C5,1)(C5,1)(A5,1)(F5,1)(C6,1)...`
- Homophonic reduction (chord reduction), e.g. select every note at every time step
  `(C5)(C5)(D5,A5)(F5)(C6)(G5)(G5) ...`

With the aim of taking into account all the variety in which music information can be represented, it has been proposed the Standard Music Description Language (SMDL), as an application of the Standard ISO/IEC Hyper-media/Time-based Structuring Language. In SMDL, a music work is divided into different domains, each one dealing with different aspects, from visual to gestural, and analytical. SMDL provides a linking mechanism to external, pre-existing formats for visual representation or storage of performances. Hence SMDL may be a useful way for music representation standardization, but the solution is just to collect different formats rather that proposing a new one able to deal with all the aspects of the communication in music.

**A Note on MIDI** A format that can be considered as a compromise between the score and the performance forms is MIDI (Musical Instrument Digital Interface), which was proposed in 1982 for data exchange among digital instruments. MIDI carries both information about musical events, from which it is possible to reconstruct an approximate representation of the score, and information for driving a synthesizer, from which it is possible to listen to a simplified automatic performance. It seems then that MIDI draws a link between the two different forms for music representation. This characteristics, together with the fortune of MIDI as an exchange format in the early times of the Internet, can explain why many music DLs and most projects regarding music indexing and retrieval refer to it. Some of the research work on music information retrieval take advantage of the availability of MIDI files of about all the different music genres and styles. MIDI files are parsed in order to extract a representation of the music score, and then indexed after different preprocessing.

Nevertheless, MIDI is becoming obsolete and users on the Internet increasingly prefer to exchange digital music stored in other formats such as MP3 or RealAudio, because they allow for a good audio-quality with a considerably small dimension of the documents size. Moreover, if the goal of a music DL is to preserve the cultural heritage, more complete formats for storing both scores and performances are required. Being a compromise between two different needs – i.e., to represent symbols and to be playable – MIDI turns out to fit neither the needs of users who want to access to a complete digital representation of the score, nor to users who want to listen to high-quality audio performances.

#### 6.6.2.4 Dissemination of Music Documents

The effectiveness of a retrieval session depends also on the ability of users to judge whether retrieved documents are relevant to their information needs. The evaluation step, in a classical presentation-evaluation cycle, for an information retrieval session of textual documents usually benefits from tools for browsing the document (e.g., the 'find' function), in particular when the size of documents is large. Moreover, a general overview of the textual content may help users to judge the relevance of most of the retrieved documents.

Users of a music DL cannot take advantage of these shortcuts for the evaluation of documents relevance, when they are retrieving music performances. This is due to the central role played by time in the listening to music. A music performance is characterized by the organization of music events along the time axis, which concatenates the single sounds that form the whole performance. Changing playback speed of more than a small amount may result in a unrecognizable performance. In other

words, it requires about 20 minutes to listen to a performance that lasts 20 minutes. It may be argued that many music works are characterized by their incipit, that is by their first notes, and hence a user could be required to listen only to the first seconds of a performance before judging its relevance to his information needs. Anyway, the relevant passage of a music document – e.g., a theme, the refrain – may be at any position in the time axis of the performance.

A tool that is often offered by playback devices is the 'skip' function, that allows for a fast access to a sequence of random excerpts of the audio files, to help listeners looking for given passages. Everyone who tried to find a particular passage in a long music performance, knows that the aid that the skip function gives when accessing to music documents is not even comparable with the find function for textual documents. This is partially due to the fact that auditory information does not allow a snapshot view of the documents as visual information does. The evaluation of relevance of retrieved music documents may then be highly time-consuming, if tools for a faster access to document content are not provided.

### 6.6.3 Approaches to Music Information Retrieval

There is a variety of approaches to MIR and there are many related disciplines involved. Because of such wide varieties, it is difficult to cite all the relevant work. Current approaches to MIR can broadly be classified into data-based and content-based approaches. For the aims of scientific research on multimedia IR, content-based approaches are more interesting, nevertheless the use of auxiliary textual data structures, or metadata, can frequently be observed in approaches to non-textual, e.g. image or video document indexing. Indeed, textual index terms are often manually assigned to multimedia documents to allow users retrieving documents through textual descriptions.

#### 6.6.3.1 Data-based Music Information Retrieval

Data-based MIR systems allow users for searching databases by specifying exact values for predefined fields, such as composer name, title, date of publication, type of work, etc., in which cases we actually speak about exact match retrieval. Data-based approaches to MIR makes content-based retrieval almost impossible since the music content cannot easily be conveyed simply by bibliographic catalogue only.

Indeed, music works are usually described with generic terms like 'Sonata' or 'Concerto' which are related only to the music form and not the actual content. From an IR point of view, data-based approaches are quite effective if the user can exhaustively and precisely use the available search fields. However, bibliographic values are not always able to describe exhaustively and precisely the content of music works. For example, the term 'Sonata' as value of the type of work cannot sufficiently discriminate all the existing sonatas.

Moreover, many known work titles, such as the Tchaikovskij's 'Pathetic', are insufficient to express a final user's query whenever he would find the title not being a good description of the music work. The use of cataloging number, like K525 for Mozart's 'Eine Kleine Nachtmusic', will be effective only if the user has a complete information on the music work, and in this case a database system will suffice.

Searching by composer name can be very effective. However, some less known composers and their works may not be retrieved if only because the authors are little known. Content-based MIR may allow for the retrieval of these pieces since querying by a known melodic pattern, such as a Mozart's one, may retrieve previously not considered or unknown composers. On the other hand, for

a prolific composer, just like Mozart, a simple query by composer's name will retrieve an extremely high number of documents, unbearable for the final user.

#### 6.6.3.2 Content-based Music Information Retrieval

Content-based approaches take into account the music document content, such as notation or performance, and automatically extract some features, such as incipites or other melody fragments, timing or rhythm, instrumentation, to be used as content descriptors. Typical content-based approaches are based on the extraction of note strings from the full-score music document. If arbitrarily extracted, note strings may be meaningless from a musical point of view because no music information is exploited to detect those strings, yet allows for a good coverage of all the possible features to be extracted.

Content-based approaches to MIR can sometimes be oriented to disclosing music document semantic content using some music information, under the hypothesis that music documents can convey some meaning and then some fragments can effectively convey such meaning. In the latter case, some music information is exploited to detect those strings so that the detected strings can musically make sense if, for instance, they were played.

The research work on this area of MIR can be roughly divided in two categories:

- on-line searching techniques, which compute a match between a representation of the query and a representation of the documents each time a new query is submitted to the system;
- indexing techniques, which extract off-line from music documents all the relevant information that is needed at retrieval time and perform the match between query and documents indexes.

Both approaches have positive and negative aspects.

- From the one hand, on-line search allows for a direct modelling of query errors by using, for instance, approximate pattern matching techniques that deal with possible sources of mismatch, e.g. insertion and/or deletion of notes. This high flexibility is balanced by high computational costs, because the complexity is at least proportional to the size of the document collection (and, depending on the technique, to the documents length).
- From the other hand, indexing techniques are more scalable to the document collection, because the index file can be efficiently accessed through hashing and the computational complexity depends only on query length. The high scalability is balanced by a more difficult extraction of document content, with non trivial problems arising in case of query errors that may cause a complete mismatch between query and document indexes.

Both approaches had given interesting and promising results. Yet, indexing approaches need to be investigated in more detail because of the intrinsic higher computational efficiency.

Previous work on on-line search has been carried out following different strategies. A first approach is based on the use of pattern discovery techniques, taken from computational biology, to compute occurrences of a simplified description of the pitch contour of the query inside the collection of documents. Another approach applies pattern matching techniques to documents and queries in GUIDO format, exploiting the advantages of this notation in structuring information. Approximate string matching has been used. Markov chains have been proposed to model a set of themes that has been extracted from music documents, while an extension to hidden Markov models has been presented as a tool to model possible errors in sung queries.

An example of research work on off-line document indexing has been presented in[8]. In that work melodies were indexed through the use of N-grams, each N-gram being a sequence of N pitch intervals. Experimental results on a collection of folk songs were presented, testing the effects of system

parameters such as N-gram length, showing good results in terms of retrieval effectiveness, though the approach seemed not be robust to decreases in query length. Another approach to document indexing has been presented in[24], where indexing has been carried out by automatically highlighting music lexical units, or musical phrases. Differently than the previous approach, the length of indexes was not fixed but depended on the musical context. That is musical phrases were computed exploiting knowledge on music perception, in order to highlight only phrases that had a musical meaning. Phrases could undergo a number of different normalization, from the complete information of pitch intervals and duration to the simple melodic profile.

Most of the approaches are based on melody, while other music dimensions, such as harmony, timbre, or structure, are not taken into account. This choice may become a limitation depending on the way the user is allowed to interact with the system and on his personal knowledge on music language. For instance, if the query-by-example paradigm is used, the effectiveness of a system depends on the way a query is matched with documents: If the user may express his information need through a query-by-humming interface, the melody is the most likely dimension that he will use. Moreover, for non expert users, melody and rhythm (and lyrics) are the more simple dimensions for describing their information needs.

Query processing can significantly differ within content-based approaches. After a query has been played, the system can represent it either as a single note string, or as a sequence of smaller note fragments. The latter can be either arbitrary note strings, such as n-grams, or fragments extracted using melody information. Regarding the query as a single note string makes content-based retrieval very difficult since it would be similar to retrieving textual files using Unix grep-like commands which provides very poor results. On the contrary, extracting fragments using melody information can result in a more effective query description. We then speak about partial match retrieval.

#### 6.6.3.3 Music Digital Libraries

Digital library projects have been carried out for designing, implementing, and testing real MIR systems. Some of them implement data-based, content-based, or both approaches to MIR. We cite some of the projects being most relevant to our research aims. The reader can access to the cited papers to have a complete description of methods and systems. The VARIATIONS digital library has been reported in [9], while the MELDEX project is reported in [4]. A project involved the University of Milan and the Teatro alla Scala, Milan [10] to implement a multimedia object-relational database storing the music contents of the archive, as well as catalogue data about the nights at the Teatro alla Scala. The access to the archive is basically based on fragment extraction and approximate string matching. A feasibility study was conducted for the ADMV (Digital Archive for the Venetian Music of the Eighteenth century) digital library project [3]. The feasibility study allowed for defining architecture, technology, and search functions for a data and content-based MIR and database management system. The system complexity is due to the number of inter-relationships of all the aspects being typical of a real effective DL: distributed databases, preservation, wide area networking, protection, data management, content-based access.

### 6.6.4 Techniques for Music Information Retrieval

Content-based MIR is a quite new research area, at least compared to classical textual IR. For this reason, most of the techniques applied to retrieve music documents derive from IR techniques. In this section, after introducing some terminology typical of content-based description of music documents, techniques for MIR and their relationship with IR techniques are described. A final example is given

on how evaluation can be carried out.

#### 6.6.4.1 Terminology

There is a number of terms that have a special meaning for the research community on MIR.

A **feature** is one of the characteristics that describe subsequent notes in a score. A note feature can be: the pitch, the pitch interval (PIT) with the previous note, a quantized PIT, the duration, the interonset interval (IOI) with the subsequent note, the ratio of IOI with the previous note, and so on. All the features can be normalized or quantized. For example, many songs can be guessed only by tapping the rhythm of the melody while other ones can be easily recognized even if played with no tempo or rubato. Due to the large availability of audio files, it is increasingly common to compute features directly from audio. For instance, a common feature in MIR are *chroma vectors* that subsum the harmonic content of a short window of the signal in a single octave.

A **string** is a sequence of features. Any sequence of notes in a melody can be considered a string. It can be noted that strings can be used as representative of a melody, which is the idea underlying many approaches to MIR, but the effectiveness by which each string represents a document may differ. For instance, it is normally accepted that the first notes of a melody play an important role in recognition, or that strings that are part of the main theme or motif are good descriptors as well. String length is an important issue: Long strings are likely to be effective descriptors, yet they may lead to problems when the user is request to remember long parts of a melody for querying a MIR system. Often, strings shorter than three notes can be discarded, because they can be considered not significant descriptors. Although the concept of string is strictly related to the sequential nature of the note organization in a score, a string can represent any kind of feature. For instance it is common practice to represent the audio content with strings of chroma vectors.

A **pattern** is a string that is repeated at least twice in the score. The repetition can be due to the presence of different choruses in the score or by the use of the same music material (e.g., motifs, rhythmical cells) along the composition. Each pattern is defined by the string of features, by its length $n$ and by the number of times $r$ it is repeated inside the score. Patterns of audio features are often exploited to segment pop and rock songs in their structural elements, such as intro, chorus, refrain, bridge, and so on. The computation of patterns can be done automatically using well known algorithms for pattern discovery. Given a particular feature, patterns can be considered as effective content descriptors of a music document. Depending on the selected feature, patterns carry different information about document content. For example patterns of MFCCs may be used in a genre identification task, because genre is usually related to sound quality, while chroma vectors may be used to compute the tonality of a song.

It can be noted that a music documents may be directly indexed by its strings. In particular, it can be chosen to describe a document with all its strings of a given length, usually from 3 to 5 notes, that are called *n-grams*. The n-gram approach is a simple, but often effective, alternative to more complex approaches that are based on melodic information. In the following sections, patterns are considered as possible content descriptors, yet the discussion may be generalized to n-grams, musical phrases, and so on. Moreover, in the following discussion, three kinds of features are considered for the pattern selection step – the interonset interval (IOI) normalized to the quarter note, the pitch interval (PIT) in semitones, and both (BTH).

### 6.6.5 Document Indexing

Document indexing is a mandatory step for textual information retrieval. Through indexing, the relevant information about a collection of documents is computed and stored in a format that allows easy and fast access at retrieval time. Document indexing is carried out only when the collection is created or updated, when users are not yet accessing the documents, and then the problems of computational time and efficiency are usually less restrictive. Indexing speeds up retrieval time because it is faster to search for a match inside the indexes than inside the complete documents.

Following the terminology introduced in the previous section, each document may be indexed by a number of patterns of different length and with different multiplicity. If it is assumed that patterns are effective descriptors for document indexing, the first step of document indexing consists in the automatic computation of the patterns of each document. As previously mentioned, relevant features which are usually taken into account are IOI, PIT, and BTH. Pattern computation can be carried out with a ad-hoc algorithms that compute exhaustively all the possible patterns, and store them in a hash table.

An exhaustive pattern discovery approach highlights a high number of patterns that have little or no musical meaning; for instance, a pattern that is repeated only two or three times in a document is likely to be computed by chance just because the combination of features is repeated in some notes combinations. Moreover, some patterns related to scales, repeated notes, or similar musical gestures, are likely to appear in almost all documents and hence to be poor discriminants among documents. In general, the degree by which a pattern is a good index may vary depending on the pattern and on the document. This is a typical situation of textual information retrieval, where words may describe a document to a different extent. For this reason it is proposed to apply the classical $tf \cdot idf$ weighting scheme.

The extent by which a pattern describes a document is the result of the multiplication of two terms. The **term frequency** is the number of occurrences of a given pattern inside a document. Hence, the term frequency of pattern $p$ for document $d$ can be computed as

$$tf_p^d = \text{\# occurrences of } p \in d$$

The **inverse document frequency** takes into account the number of different documents in which a patters appears. The inverse document frequency of pattern $p$ can be computed as

$$idf_p = -log \frac{\text{\# documents containing } p}{\text{\# documents}}$$

Relevant patterns of a document may have a high $tf$ – they are frequent inside the document – and/or a high $idf$ – they are infrequent across the collection.

For the aims of indexing, a document is described by a sparse array, where each element is associated to a different pattern in the collection. The value of each element is given by the $tf \cdot idf$ value. The index is built as an inverted file, where each term of the vocabulary is a different pattern in a given notation (i.e., a text string). Each entry in the inverted file corresponds to a different pattern, and can efficiently be computed in an expected time $O(1)$ with an hashing function. Given the different sets of features, three inverted files are built, respectively for features IOI, PIT, and BTH. Inverted files can be efficiently stored in memory, eventually using compression, and fast accessed at retrieval time. The size of the inverted file and the implementation of the hashing function depend on the number of different patterns of the complete collection.

It may be useful to fix the maximum allowable pattern length to improve indexing. In fact, it is likely that very long patterns are due to repetitions of complete themes in the score and taking

into account also them will give a quite sparse inverted file. Moreover, it is unlikely that a user will query the system singing a complete theme. These considerations suggest that long patterns could be truncated when they are over a given threshold.

#### 6.6.5.1 Query Processing

For the query processing step, it can be assumed that users interact with the system according to a query-by-example paradigm. In particular, users should be able to describe their information needs by singing (humming or whistling), playing, or editing with a simple interface a short excerpt of the melody that they have in mind. Pitch tracking can be applied to the user's query in order to obtain a transcription in a notation format, such as a string of notes. The string representing the translated query needs to undergo further processing, in order to extract a number of descriptors that can be used to match the query with potentially relevant documents. It is normally assumed that a query is likely to contain strings that characterize the searched document, either because they appear very often inside its theme or because they are peculiar of that particular melody. In other words, a query is likely to contain relevant patterns of the searched document, which may have a high $tf$ and/or $idf$.

The automatic detection of relevant strings cannot be carried out through pattern analysis, because normally queries are too short to have repetitions and hence to contain patterns. A simple approach to extract relevant strings, or potential patterns, from a query consists in computing all its possible substrings. That is, from a query of length $q$ notes are automatically extracted $q-2$ strings of three notes, plus $q-3$ strings of four notes, and so on until the maximum allowable length for a pattern is reached. This approach can be considered similar to query expansion in textual information retrieval, which is known to increase recall at the risk of lowering precision. On the other hand, it is expected that most of the arbitrary strings of a query will never form a relevant pattern inside the collection, and then the negative effects on precision could be bounded.

#### 6.6.5.2 Ranking Relevant Documents

At retrieval time, the strings are automatically extracted from the query and matched with the patterns of each document. The computation of potentially relevant documents can be carried out computing the distance between the vector of strings representing the query and the vector of patterns representing each document. Hence, for each document a Retrieval Status Value (RSV) is calculated, the higher the RSV, the closer the document with the query. A rank list of potentially relevant documents is computed from RSVs, obtaining a different rank lists for each of features used.

In general the orderings of documents in the rank lists differ. Differences may be due to many factors, as the diverse importance of rhythm and melodic profile for a the document collection, the effect of errors in the query, the kind of melodic excerpt chosen by the user as a representative of his information needs. It is expected that BTH ranking will give high scoring to the relevant documents when the query is sufficiently long and correctly played, because BTH patterns are a closer representation of the original melody. On the other hand, IOI and PIT are robust to query errors in melodic profile and rhythm, respectively. Moreover, simple representations as IOI and PIT are expected to be less sensitive to query length because of the possible presence of subpatterns of relevant motifs.

It is possible to take advantage from the existence of different rank lists by fusing together the results, in order to give the user a single rank list which takes into account the results of the three parallel approaches. This is a typical problem of **data fusion**, an approach that is usually carried out in the research area of Meta Search Engines, where the results obtained by different indexing and retrieval methodologies are combined – or fused – together according to a predefined weighting

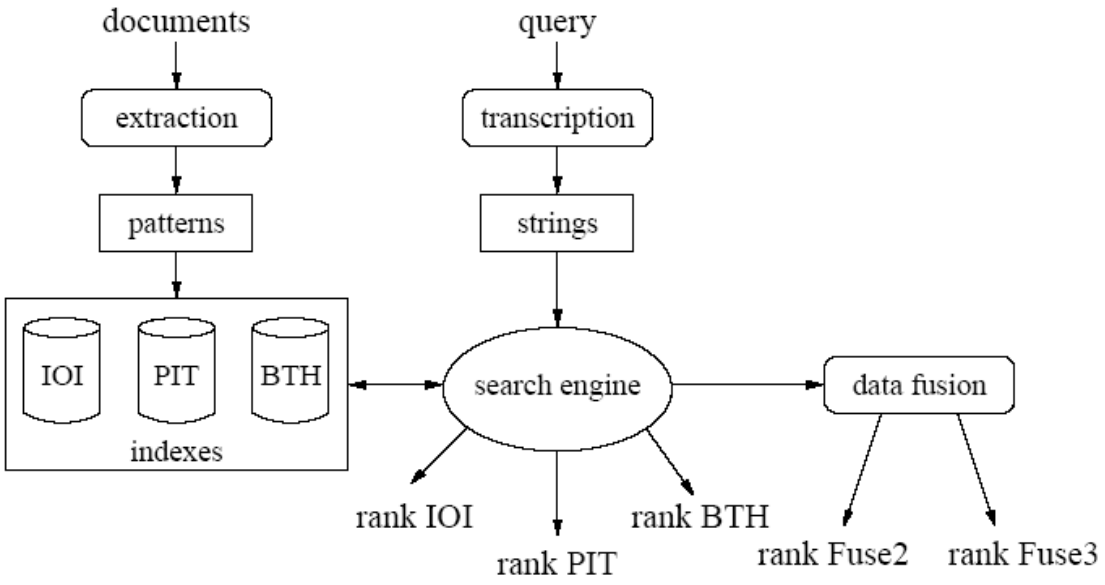



**Figure 6.29:** *The phases of a methodology for MIR: Indexing, retrieval, and data fusion*

scheme. Since the RSVs of individual search engines are not known, or not comparable with others, the classical approach to data fusion is based on the information of rank only. In the case of MIR based on parallel features, the fusion can be carried out directly using the RSVs, because they are all based on the same $tf \cdot idf$ scheme. A new RSV can be computed as a weighted sum of RSVs of single features obtaining a new rank list.

A complete methodology for MIR shown in Figure 6.29, where steps undertaken at indexing time are shown on the left, while the operations that are performed at retrieval time are shown on the right. From Figure 6.29 and the above discussion, it is clear that the computational complexity depends on the query length – i.e., the number of strings that are computed from the query – while it is scalable on the number of documents. This is an important characteristic given by indexing techniques, because the time needed to reply to a query can be reasonably low also for large collections of documents.

#### 6.6.5.3 Measures for Performances of MIR Systems

The output of almost any information retrieval system, and this applies also to MIR, is a ranked list of potentially relevant documents. It is clear that only the final user can judge if the retrieved documents are really relevant to his information needs. That is, the user should evaluate system performances in terms of retrieval effectiveness. There are two main reasons why the user may not be satisfied by the result of an information retrieval system.

- the system does not retrieve documents that are relevant for the user information needs – which is usually called **silence effect**;
- the system retrieves documents that are not relevant for the user information needs – which is usually called **noise effect**

All real systems for MIR try to balance these two negative effects. From the one hand, a high silence effect may result in not retrieving all the music documents that are similar to a given query sung by the user. From the other hand, a high noise effect may cause the user to spend great part of a retrieval

session in listening to irrelevant documents.

Even if user satisfaction plays a central role in the evaluation of performances of a MIR system, and in general of any IR system, user studies are very expensive and time consuming. For this reason, the IR research community usually carries out automatic evaluation of the proposed systems using commonly accepted measures. In particular, there are two measures that are connected to the concepts of silence and noise effects. The first measure is **recall**, which is related to the ability of a system to retrieve the highest percentage of relevant documents (thus minimizing the silence effect). Recall is defined as

$$\text{recall} = \frac{\text{\# relevant retrieved}}{\text{\# total relevant}}$$

that is the number of relevant documents retrieved by the system divided by the total number of relevant documents in the complete database of documents. The second measure is **precision**, which is related to the ability of the system of retrieving the lowest percentage of irrelevant documents (thus minimizing the noise effect). Precision is defined as

$$\text{precision} = \frac{\text{\# relevant retrieved}}{\text{\# total retrieved}}$$

that is the number of relevant documents retrieved by the system divided by the total number of retrieved documents. An ideal system retrieved only relevant documents, and hence has $100\%$ recall and precision. For real systems, high precision is usually achieved at the cost of low recall and viceversa.

Both precision and recall do not take into account that a MIR system may output a ranked list of documents. For this reason it is a common practice to compute these measures also for the first $N$ documents (for $N \in \{5, 10, 20, \dots\}$) and, in particular, to compute the precision at given levels of recall. Another approach is to summarize these measures, and the effect of the documents rank, in a single measure. For instance, the **average precision** is computed as the mean of the different precisions computed each time a new relevant document is observed in the rank list. Given a set of test queries, a measure that is often presented when evaluating the performance of a search engine is the **Mean Average Precision** or **MAP**, which is simply the mean over all the queries of the respective average precisions.

For some task, it is more important to measure the position of the first correct result in a ranked list of retrieved documents. For instance, if the user wants to find the audio file of a given song, it is more important to assess how many songs he has to browse before finding the one of interest, rather than how many alternative copies of the song have been retrieved. To this end, another important measure is the **Mear Reciprocal Rank** or **MRR**, which is the reciprocal of the position of the first correct document in the ranked list of results, averaged over all the test queries.

The evaluation of MIR systems is usually carried out on a test collection according to the Cranfield model for information retrieval, which is used at the Text REtrieval Conference (TREC). A test collection consists in a set of documents, a set of queries, and a set of relevance judgments that match documents to queries. The creation of a common background for evaluation is still an open issue in the MIR community, hence each research group created its own test collection from scratch. A "good" test collection should be representative of real documents and, in particular, of real user's queries. The size of the document set, as well as the way queries are collected, may deeply influence the evaluation results. Relevance judgments should be normally given by a pool of experts in the music domain, which is an expensive task, but they can also be automatically constructed when queries are in the form of excerpts of a known tune. In this latter case, only the document from which the query derives is considered as relevant.

### 6.6.6 A Common MIR Task: Music Identification

At the end of this section, we present a methodology for music identification as an example of a typical task. The main application that motivates the presented approach is the automatic identification of recordings digitized by sound archives. The experimentation has been carried out on a collection of vinyl discs, although the methodology is not linked to a particular carrier and can be readily extended to audio tapes, shellack discs, and so on. Automatic identification allows a music digital library system to retrieve relevant metadata about music works even if this information was incomplete or missing at the time of the digital acquisition. We believe that the availability of systems for the identification of music works will promote the dissemination of music cultural heritage, allowing final users to retrieve and to access individual recordings also when descriptive metadata were not available at the time of acquisition.

A typical approach to automatic music identification is based on the extraction of an *audio fingerprint* from digital recordings. A fingerprint is a compact set of music features that allows for the identification of digital copies even in the presence of noise, distortion, and compression; it can be seen as a content-based signature that summarizes an audio recording. The well known SmartPhone application *Shazam!* is based on audio fingerprint.

The more general task of music identification differs from audio fingerprint because of the variability of timbre, instrumentation and tempo. Identification of *different versions of the same work* is often based on the extraction of a sequence of audio features that are related to high level characteristics of music works – melody, harmony, rhythm – and their alignment using well-known techniques such as Hidden Markov Models (HMMs) and Dynamic Time Warping (DTW).

Identification methodologies based on alignment are usually computationally intensive, because they require the alignment of a query with every item in the reference collection. In parallel of what has been described for score features, quantization of audio features can be exploited to obtain an index-based matching procedure, aimed at efficient music identification. Results showed that these approaches are robust to typical variations due to different interpretation of classical music pieces.

#### 6.6.6.1 Identification Methodology

The described approach to cover identification aims primarily at efficiency. The methodology is inspired by the concept of Locality Sensitive Hashing (LSH), which is a general approach to handle high dimensional spaces by using ad-hoc hashing functions to create collisions between vectors that are close in the high dimensional space. LSH has been applied to efficient search in video and image collections.

In the proposed approach, the problem of music identification is addressed making use of a particularly compact representation of the audio content. This representation allows the similarity among recordings to be computed by means of a *bag of features* representation – a simpler representation than the string of features introduced earlier in this chapter – which allows us to exploit indexing techniques to speed up retrieval.

### 6.6.7 Audio Content Representation

A recording is stored in a digital system as a discrete signal, representing an acoustic pressure measured at regular time intervals. In the proposed approach, this *audio waveform* is divided into short overlapping excerpts of fixed length (dozens of milliseconds), and content-based descriptors are then extracted from each resulting segment. The adopted descriptors are *chroma features* (or chroma vectors), 12-dimensional vectors representing the energy associated to each pitch class in a short time

frame. A pitch class is a set of pitches corresponding to a given note in the chromatic scale. For instance, the pitch class of the note C corresponds to frequencies: 32.7 Hz, 64.4 Hz, 130.8 Hz, and so on.

Chroma extraction is preceded by *tuning frequency adjustment* in order to be robust to adoption of different reference frequencies, and is followed by a *key finding* procedure, which allows us to deal with different versions of the same music work performed in different keys. The choice of key invariant chroma features as content descriptors is based on the consideration that listeners use mostly harmonic and melodic cues to decide whether two recordings are different performances of the same music work. Differences in tonality, tempo and duration do not substantially affect the similarity judgment and thus our representation aims at being robust to changes in these music dimensions.

A general formula to compute chroma vectors from a windowed signal $s(t)$ is the following

$$c_i = \sum_{f=32Hz}^{4000Hz} B_i(f) \cdot S(f) \tag{6.27}$$

where $S(f)$ is a representation of $s(t)$ in the frequency domain and $B_i(f)$ is a bank of bandpass filters, each centered on the semitones belonging to pitch class $i$ and with a bandwidth of a semitone. Usually chroma features are computed only on a limited range of the audible spectrum.

Once audio recordings have been represented by chroma vectors, a *hashing* step is introduced to allow for a compact representation of chroma vectors. The quantized version $q$ of a chroma vector $\mathbf{c}$ is obtained by taking into account the ranks of the chroma pitch classes, sorted by their values. Let $r_i$ be the position in which the $i$-th component $c_i$ would be ranked after a descending sort (starting from 0); a $k$-level rank-representation of $\mathbf{c}$ is constructed by considering a base 12 number computed as:

$$r_i = |\{c_j : c_j > c_i, j = 1, \ldots, 12\}| \qquad i = 1, \ldots, 12 \tag{6.28}$$

$$q = \sum_{i:r_i<k} i \cdot 12^{r_i} \tag{6.29}$$

#### 6.6.7.1 Efficient Identification

The identification algorithm is based on a two-level bag-of-features representation of the audio content: the hash sequence computed from the audio waveform according to the procedure described above is divided into overlapping segments of fixed length (typically corresponding to dozens of seconds). Identification of an unknown recording is carried out computing its similarity with the recordings in the collection; adopting the textual IR terminology, we will refer to them as *query* and *documents*, respectively.

Let $Q$ ($D$) denote a recording associated to a query (audio document in the database), composed of a sequence $\{q_1 \ldots q_{|Q|}\}$ ($\{d_1 \ldots d_{|D|}\}$) of hash sequences, each of length $|q|$ ($|d|$). It is possible to conceptually distinguish two phases in the similarity computation procedure. The first step implies the computation of local similarity between segments $d$ and $q$ as the (normalized) number of descriptors they have in common

$$S_L(d, q) = \sum_{t \in q \cap d} \min\Big(\frac{\mathrm{tf}(t, d)}{|d|}, \frac{\mathrm{tf}(t, q)}{|q|}\Big) \tag{6.30}$$

where $\mathrm{tf}(t, d)$ denotes the count (*term frequency*) of hashes with value $t$ in segment $d$. The second step aggregates the contributions of all the query segments, by computing the geometric mean of the

best local similarity values for each query segment:

$$S(Q,D) = \sqrt[|sQ|]{\prod_{q \in Q} \max_{d \in D} S_L(d,q)} \tag{6.31}$$

An efficient implementation of the similarity computation procedure is possible because any information regarding the *ordering* of segments, and of the hashes contained therein, is discarded. This is reflected in the notation of Equation 6.31 by the exclusive use of *set operations*.

**Data Representation** The computation of the similarity score for a query-document pair requires information on the hash frequency in each query and document segment. Information on the frequency of occurrence of an hash in a specific query segment can be extracted at query time and efficiently accessed by means of data structure maintained in memory. The current implementation of the architecture exploits a list of maps, where each list entry (each map) corresponds to a segment and retains *(hash,frequency)* pairs.

**Indexing** Information on the frequency of a descriptor in a document can be efficiently accessed by means of an inverted index. In such data structure an *inverted list* is associated to each distinct descriptor appearing in at least one of the documents in the collection; the entries of the inverted list are the (identifiers of the) documents where the descriptor occurs. Moreover, additional information necessary by the ranking function can be stored in the inverted list entries, e.g. the frequency of occurrence of the descriptors or the positions where they occur.

In the adopted methodology, the descriptors are chroma hashes, and each segment of a recording is interpreted as a textual document. Figure 6.30 provides an example of the indexing process when applied to a recording, represented as a sequence of hashes.

Such sequence undergoes a segmentation process where possibly overlapping subsequences are extracted from the recording sequence. After segmentation, a recording is represented by a *set* of hash sequences. In accordance with the *bag of features* paradigm, it is hypothesized that information on hash occurrences at a segment level is sufficient for effectively identify a recording; positional information is therefore ignored, thus each segment is effectively treated as a *set* of hashes. An inverted index can efficiently store hash occurrence information: the list of index descriptors is the set of distinct hashes in all the recordings in the collection, while the entries in the inverted list for a hash are the *(segment,frequency)* pairs for that hash.

**Document At A Time Processing** The adoption of an inverted index allows us to compute efficiently the segment similarity score in Equation 6.30. When a query $Q$ is submitted to the system it undergoes the same steps as the documents in the collection. The key finding algorithm mentioned in Section 6.6.7 is applied to the query, thus obtaining diverse transpositions that the system treats as distinct queries, processed in parallel. After the computation of its hash values, the query is split into a number of *segment-queries*. Each segment-query is processed by a Document At A Time (DAAT) strategy. DAAT strategies evaluate the contributions of every query term with respect to a single document before considering the next document. One advantage of the DAAT strategy is that it does not require intermediate document scores to be maintained during the entire ranking process, thus limiting the run-time memory usage.

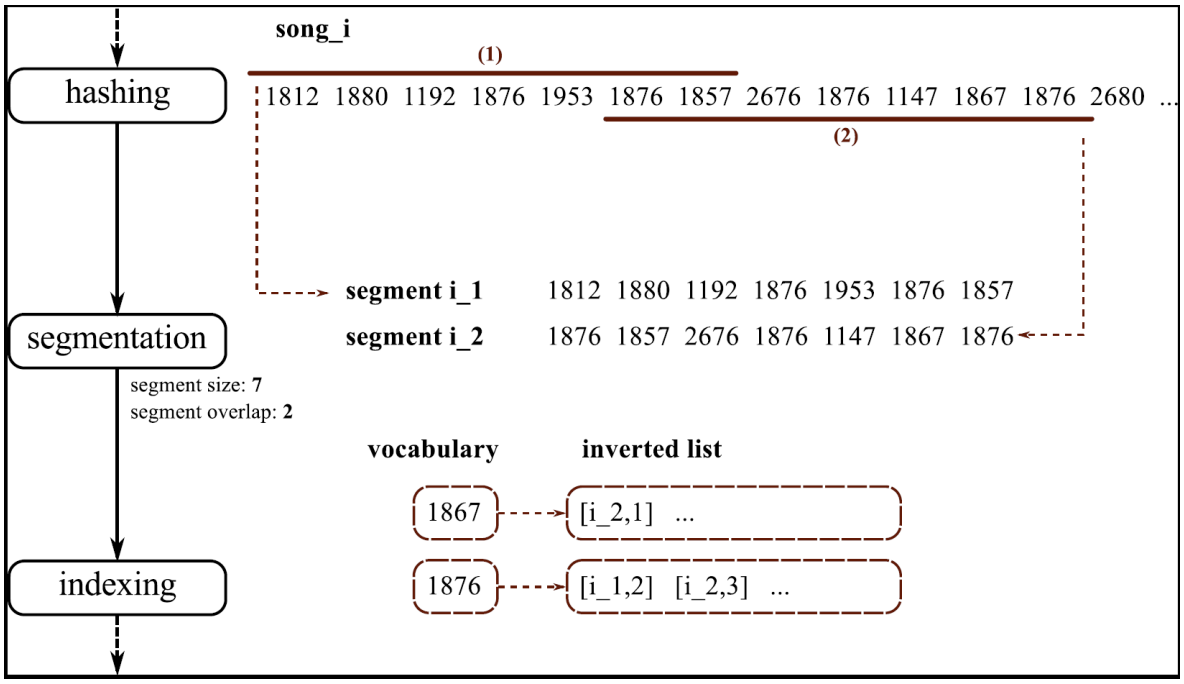



**Figure 6.30:** *Segmentation and indexing of hashed chroma descriptors.*

**Score Fusion** For each segment-query the system returns a ranked list whose entries are the document segments of all the documents in the collection ranked in decreasing order of similarity value computed at the segment level. The maximum among all the document segments is then computed by going through the returned ranked list. The next step consists in the aggregation of the results returned by the segment-queries, according to Equation 6.31. Finally, the results obtained from the diverse considered transpositions are merged to obtained a final results list. An advantage of this retrieval strategy is that the diverse query (transpositions) and the diverse segment-queries can be processed in parallel.

#### 6.6.7.2 Identification of Multiple Works Within a Single Query

The algorithm detailed above assumes that a query can be matched with a recording of the same music material. This is the typical scenario for music identification systems, but in order to handle the case of a query containing multiple works – such as digitizations of tapes or vinyl discs containing several tracks – the methodology must be extended.

To this end, an additional time-resolution level was added to the segmentation hierarchy. Basically, the recording of an LP side is divided into shorter elements, each one considered as an individual query. Figure 6.31 shows the procedure, which provides us with a number of resulting rank lists that are merged into a structure called *rank matrix*. We will refer to these short elements as "chunks", to disambiguate them from the segments in which a single track is divided.

The rationale behind this choice is related to the way the similarity between a query and the documents is computed. As can be seen from Equation 6.31, it involves a maximization over indexed segments of the local similarities for each query segment. As a consequence, the system is designed to support short queries containing a portion of a corresponding recording while it becomes ineffective with long queries containing segments of different recordings, because the geometric mean will be

computed also from local scores with very low similarity values. It should be noticed that, while the maximization and averaging indexes of Equation 6.31 – $d$ and $q$ respectively – could be in principle reversed (there is no theoretical reason for the asymmetry of the global similarity function), several implementation choices aimed at efficiency depend on this configuration.

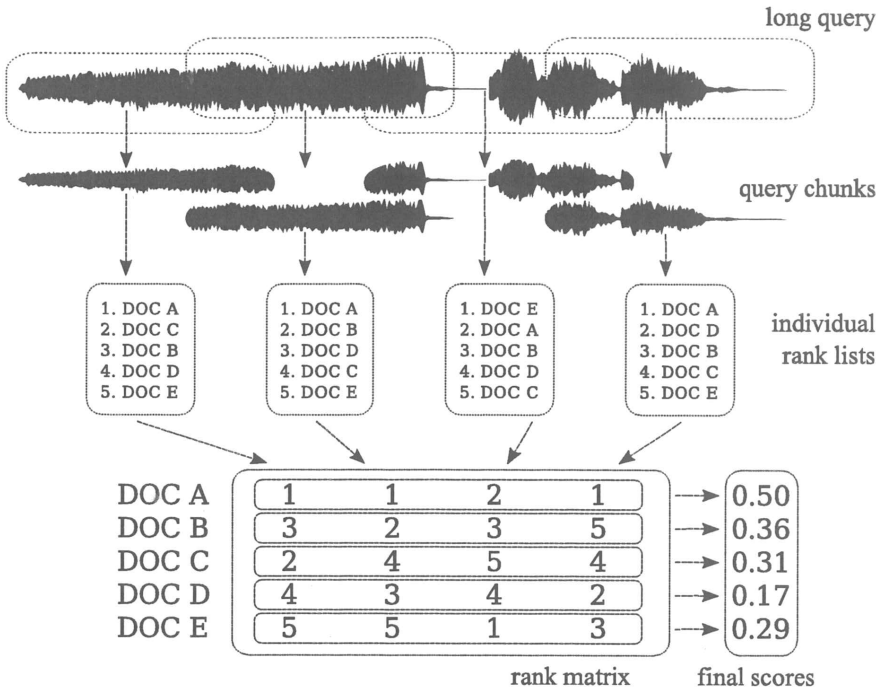



**Figure 6.31:** *Computation of the rank matrix for chunks of a long query.*

The similarity $S_i$, between the query and the recording indexed in the $i$-th row of the rank matrix, is computed with a simple data fusion approach:

$$S_i = \max_{w=1...L-W} \sum_{j=w}^{w+W-1} \frac{1}{(\max(r_{i,j}, C))^2} \tag{6.32}$$

where $L$ denotes the number of query chunks (columns of the rank matrix), and $r_{i,j}$ represents the position of recording $i$ in the rank list produced from chunk $j$. The underlying idea is to consider each row of the rank matrix, and to analyze small windows of length $W$: an indexed recording spanning multiple chunks is supposed to consistently rank in high positions, and the windowing (along with the saturation constant $C$) acts as a filter for documents whose behavior in the ranking sequence is noisy.

The choice of using rank values instead of similarity scores is motivated by the consideration that averaging similarity scores works only if all segments belong to the same recording, whereas different recordings induce radically different similarity values. As for many data fusion approaches, the choice of rank values reduces the effect of large differences in the similarity scores. Moreover, the choice of ranks enables independence of the particular parametrization adopted for the local similarity computation.

**Table 6.1:** *Experimental results, on a database of 6680 recordings, using a 3.4 Ghz machine.*

| Task | MRR | MAP | mean query time | mean query length |
|---|---|---|---|---|
| Track identification | .832 | .766 | 2s | 5'25" |
| LP identification | .900 | .742 | 46s | 47'13" |

#### 6.6.7.3 Experimental Evaluation

The objective of the experiments reported in this section is to evaluate the capability of the proposed methodology to identify multiple works within a music recording. In particular, it should be verified whether a fixed-length segmentation strategy is able to support identification of recordings characterized by radically different durations.

**Test Collection** The adopted experimental collection is composed of two corpora. The first is part of the test collection that was recently introduced and adopted in the Music Identification Task of the MusiCLEF Lab in CLEF2011. It is made up of 6680 tracks, grabbed from commercial CDs; among those, 2671 music works are represented at least twice in the collection, forming 945 cover sets. The second corpus is a collection provided by the Fonoteca of the University of Alicante and constituted of 100 LPs that were digitized using common LP record player equipment. The LPs from the Fonoteca were used to evaluate the procedure detailed in Section 6.6.7.2, matching their contents with recordings in the MusiCLEF2011 collection.

**Results** The accuracy of the identification methodology is greatly affected by the choice of a particular combination of parameters, among which are the chroma extraction and the tuning frequency adjustment algorithms, and the values of the quantization and segmentation parameters. In order to select an effective combination of parameters, we used the reference recording dataset of the MusiCLEF2011 collection. Considering the accuracy/efficiency trade off, we selected the chroma feature extraction algorithm available in the MIRToolbox, using segments of 50s without overlap and 3-level chroma quantization. These experiments have been carried out to highlight the best configuration for a music identification task.

The chosen parametrization was thus used to perform the LP identification task, where multiple works are present within a single query. The effect of different choices for the segmentation and length of the analysis window on accuracy and efficiency of the algorithm was also investigated.

Table 6.1 reports the accuracy of the algorithm in terms of Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP), along with the average query time (in the LP identification case, the average query time refers to the time needed to identify a complete LP). The experiments were performed on a machine featuring a dual-core 3.4 Ghz CPU and a 7200 RPM hard disk.

The proposed approach is both effective and efficient. In particular, identification can be carried out in a small fraction of the time required for ripping a track from a commercial CD and, most of all, digitizing a complete LP.

## Conclusions

This section presents a short overview on some aspects of music IR. In particular, the issues typical of the music language have been discussed, taking into account the problems of formats and the role of the user. A number of approaches that have been proposed in the literature are presented, in particular the ones related to music Digital Libraries.

There are a number of aspects that are beyond the scope of this overview. In particular, all the research work related to audio processing that, even if not central to music IR, plays an important role in creating tools for classification of audio files and automatic extraction of low level features, that may be useful for expert users.

## 6.7 Commented bibliography

The reference book for Auditory scene analysis is Bregman [1990]. The Implication realization model is described in Narmour [1990]. The Local Boundary Detection algorithm is presented in Cambouropoulos [2001]. The Generative Theory of Tonal Music is described in Lerdahl and Jackendoff [1983].

Research on automatic metadata extraction for MIR can be classified in two main fields, depending on the two different classes of formats in which a music document can be represented: the automatic extraction of relevant information from a music score, which is typically achieved through melody segmentation and indexing; the automatic categorization of a music recording, which is typically achieved through audio classification. In this chapter we deal with the first field.

In the case of melody segmentation and indexing, the main assumption is that it is not possible to use textual descriptors for music documents, in particular for compositions and for melodies. Since it is not clear what kind of meaning is conveyed by a music document, the common approach is to describe a document using perceptually relevant elements, that may be in the same form of the document itself (that is the only way to describe music is through music). Clearly, the alternative description of a music document should be more compact and summarize the most relevant information, at least from a perceptual point of view. The music language may be characterized by different dimensions, which may regard the score representation ( e.g., melody, harmony, rhythm) the recording of performances ( e.g., timbre, instrumentation ) and high level information ( e.g., structure, musical form). Among the different dimensions, melody seems to be the most suitable for describing music documents. First of all, users are likely to remember and use, in a query-by-example paradigm, parts of the melody of the song they are looking for. Moreover, most of the dimensions require a good knowledge of music theory to be effectively used, reducing the number of potential users to scholars, composers, and musicians. Finally, melody can benefit from tools for string analysis and processing to extract relevant metadata. For these reasons, most of the research work on metadata extraction focused on melody segmentation and processing. The need for automatic melody processing for extracting relevant information to be used as alternative descriptors, arises from the fact that the melody is a continuous flow of events. Even though listeners perceive the presence of elements in the melodic flow, which may be called lexical units, there is no explicit separator to highlight boundaries between them. Moreover, it is well known that there are parts of the melody ( e.g., the incipit, the theme, the leit-motiv, and so on ) that are more relevant descriptors of a music document than others. Yet, the automatic labelling of these relevant parts needs ad-hoc techniques.

One of the first works, probably the most cited in the early literature on MIR, is Ghias et al. [1995]. In this paper it is proposed the use of a query-by-example paradigm, with the aim of retrieving the documents that are more similar to the melody excerpts sung by the user: both documents and queries are transformed in a different notation that is related to the melodic profile. An alternative approach to MIR is proposed in Blackburn and DeRoure [1998], where metadata is automatic computed and stored in a parallel database. Metadata is in the form of hyperlinks between documents that are judged similar by the system.

Music language is quite different from other media, because it is not clear if music conveys a

meaning and how a music document can be effectively described; this mostly because perception plays a crucial role in the way users can describe music. The important issue of perception is faced in Uitdenbogerd and Zobel [1998], where a user study is presented on users melody representation. The knowledge of music structure is exploited in Melucci and Orio [1999] for extracting relevant information, where music documents and queries are described by surrogates made of a textual description of musical lexical units. Experiments on normalization are also reported, in order to cope with variants in musical lexical units that may describe similar documents. In Bainbridge et al. [1999] is proposed a multimodal description of music documents, which encompasses the audio, a visual representation of the score, the eventual lyrics, and other metadata that are automatically extracted from files in MIDI format.

An alternative approach to automatically compute melodic descriptors of music documents is presented in Bainbridge et al. [1999], which is based on the use of N-grams as musical lexical units. Alternatively, musically relevant phrases are proposed in Melucci and Orio [2000], where an hypertextual structure is automatically created among documents and musical phrases. In this case a document is described by a set of links to similar documents and to its most relevant phrases. Musical structure is exploited in Hoos et al. [2001] for computing a set of relevant features from a music document in a complex notation format.

Alternatively to previous works, in Birmingham et al. [2001] it is proposed that a good descriptor of a music document is its set of main themes, which are units longer than N-grams or musical phrases. Themes are modelled through the use of Markov chains. An extension to hidden Markov models is presented in Shifrin et al. [2002], where possible mismatches between the representation of the query and of the documents are explicitly modelled by emission probabilities of Hidden Markov Models states. An evaluation of different approaches is presented in Hu and Dannenberg [2002], where the problem of efficiency is raised and discussed.

## References

D. Bainbridge, C.G. Nevill-Manning, I.H. Witten, L.A. Smith, and McNab R.J. Musical information retrieval using melodic surface. pages 161–169, 1999.

W.P. Birmingham, R.B. Dannenberg, G.H. Wakefield, M. Bartsch, D. Bykowski, D. Mazzoni, C. Meek, M. Mellody, and W. Rand. Musart: Music retrieval via aural queries. pages 73–82, 2001.

S. Blackburn and D. DeRoure. A tool for content based navigation of music. In *Proc. ACM Multimedia Conference*, pages 361–368, 1998.

A. S. Bregman. *Auditory Scene Analysis*. MIT Press, 1990.

E. Cambouropoulos. The local boundary detection model (lbdm) and its application in the study of expressive timing. 2001.

A. Ghias, J. Logan, D. Chamberlin, and B.C. Smith. Query by humming: Musical information retrieval in an audio database. In *Proceedings of ACM Digital Libraries (DL) Conference*, pages 231–236, 1995.

H.H. Hoos, K. Renz, and M. Gorg. GUIDO/MIR - an experimental musical information retrieval system based on guido music notation. pages 41–50, 2001.

N. Hu and R.B. Dannenberg. A comparison of melodic database retrieval techniques using sung queries. In *Proc. ACM/IEEE Joint Conference on Digital Libraries*, pages 301–307, 2002.

F. Lerdahl and R. Jackendoff. *A Generative Theory of Tonal Music*. The MIT Press, 1983.

M. Melucci and N. Orio. Musical information retrieval using melodic surface. In *Proc. 4th ACM Conference on Digital Libraries*, pages 152–160, 1999.

M. Melucci and N. Orio. Smile: a system for content-based musical s information retrieval environments. In *Proc. Intelligent Multimedia Information Retrieval Systems and Management (RIAO) Conference*, pages 1246–1260, 2000.

B. C. J. Moore, B. R. Glasberg, and T. Baer. A model for the prediction of thresholds, loudness, and partial loudness. *Journal of the Audio Engineering Society*, 45(5):224–240, April 1997.

Eugene Narmour. *The Analysis and cognition of basic melodic structures : the implication-realization model*. University of Chicago Press, 1990.

W. Schloss. *On the Automatic Transcription of Percussive Music: From Acoustic Signal to High Level Analysis*. PhD thesis, Stanford University, 1985.

J. Shifrin, B. Pardo, C. Meek, and W. Birmingham. Hmm-based musical query retrieval. In *Proc. ACM/IEEE Joint Conference on Digital Libraries*, pages 295–300, 2002.

A. Uitdenbogerd and J. Zobel. Manipulation of music for melody matching. In *Proc. ACM Multimedia Conference*, pages 235–240, 1998.

# Contents