

Music Retrieval and Recommendation

Part II: Content-based Music Information Retrieval

Markus Schedl
Peter Knees

{markus.schedl, peter.knees}@jku.at

Department of Computational Perception
Johannes Kepler University (JKU)
Linz, Austria

How to Describe Audio Content?

Idea: get features that describe music the way humans do and compute similar songs based on this information

Unfortunately we are not able to extract most of these features reliably (or at all...)

- even “simple” human concepts are difficult to model (“**semantic gap**”)
- even tempo estimation is very hard...
- NB: a human annotation approach is done in the Music Genome Project (cf. Pandora’s automatic radio station service)

Furthermore some of these features are quite subjective (e.g., mood)

Need to find computable descriptors that capture these dimensions somehow (...and work acceptably)

Descriptors of Content

Acoustic property to describe:

- **Loudness:** perceived strength of sound; *e.g., energy*
- **Pitch:** frequency, psychoacoustic ordering of tones (on scale; from low to high); *e.g., chroma-features*
- **Timbre:** “tone color”, what distinguishes two sounds with same pitch and loudness; *e.g., MFCCs*
- **Chords and harmony:** simultaneous pitches
- **Rhythm:** pattern in time; *e.g., FPs*
- **Melody:** sequence of tones; combination of pitch and rhythm



cf. (Casey et al.; 2008)

Categorization of Content-Based Features

Domain:



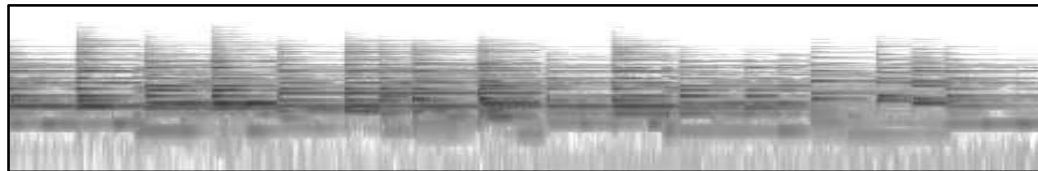
- **Time domain**

consider signal in time/amplitude representation (“waveform”)



- **Frequency domain**

consider signal in frequency/magnitude representation



Transformation from time to frequency domain using, e.g.,
Fast Fourier Transform (FFT)

Categorization of Content-Based Features

Temporal scope:



- **Instantaneous**

feature is valid for a “point in time” (NB: time resolution of ear is several msec!)

- **Segment**

feature is valid for a segment, e.g., phrase, chorus (on a high level), or a chunk of n consecutive seconds in the audio signal

- **Global**

feature is valid for whole audio excerpt or piece of music

Categorization of Content-Based Features

Level of abstraction:



High-level

Examples: instrumentation, key, chords, melody, rhythm, tempo, lyrics, genre, mood



Mid-level

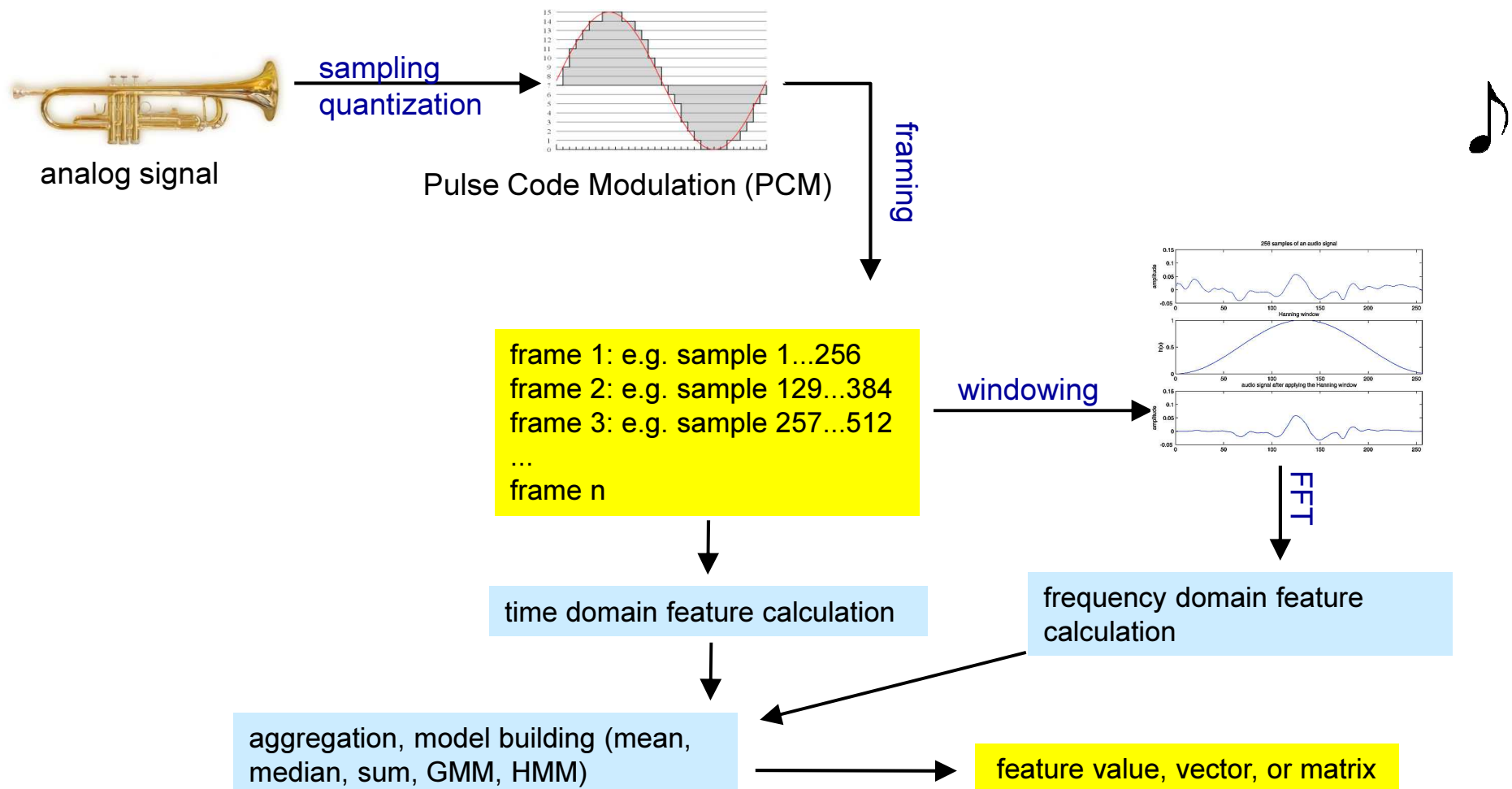
Examples: pitch- and beat-related descriptors, such as note onsets, rhythm patterns, MFCCs



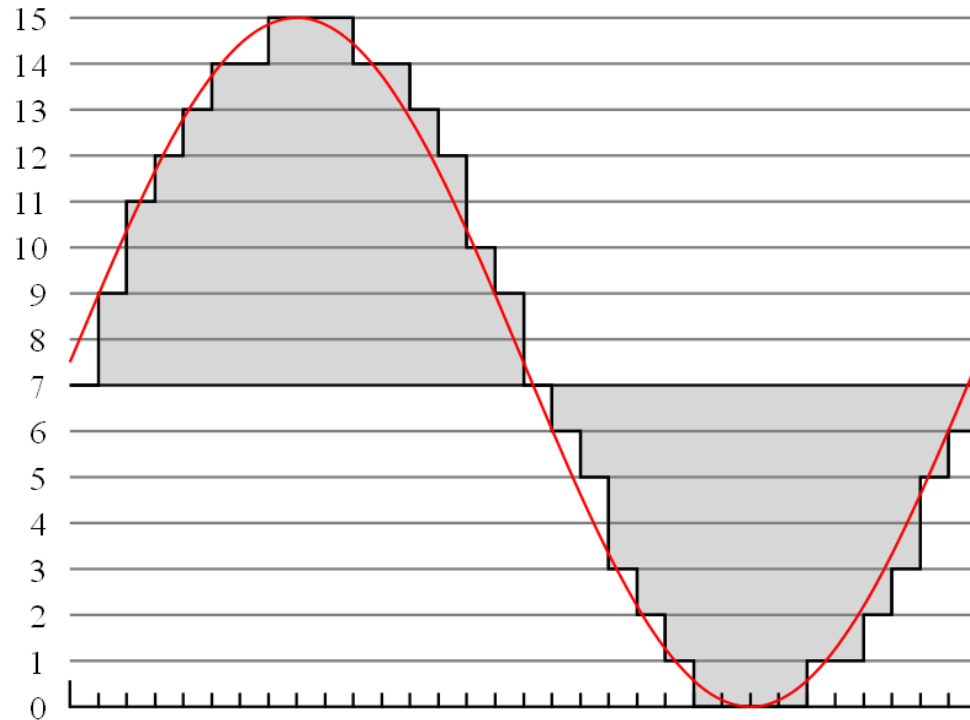
Low-level

Examples: amplitude envelope, energy, spectral centroid, spectral flux, zero-crossing rate

Scheme of Content-Based Feature Extraction

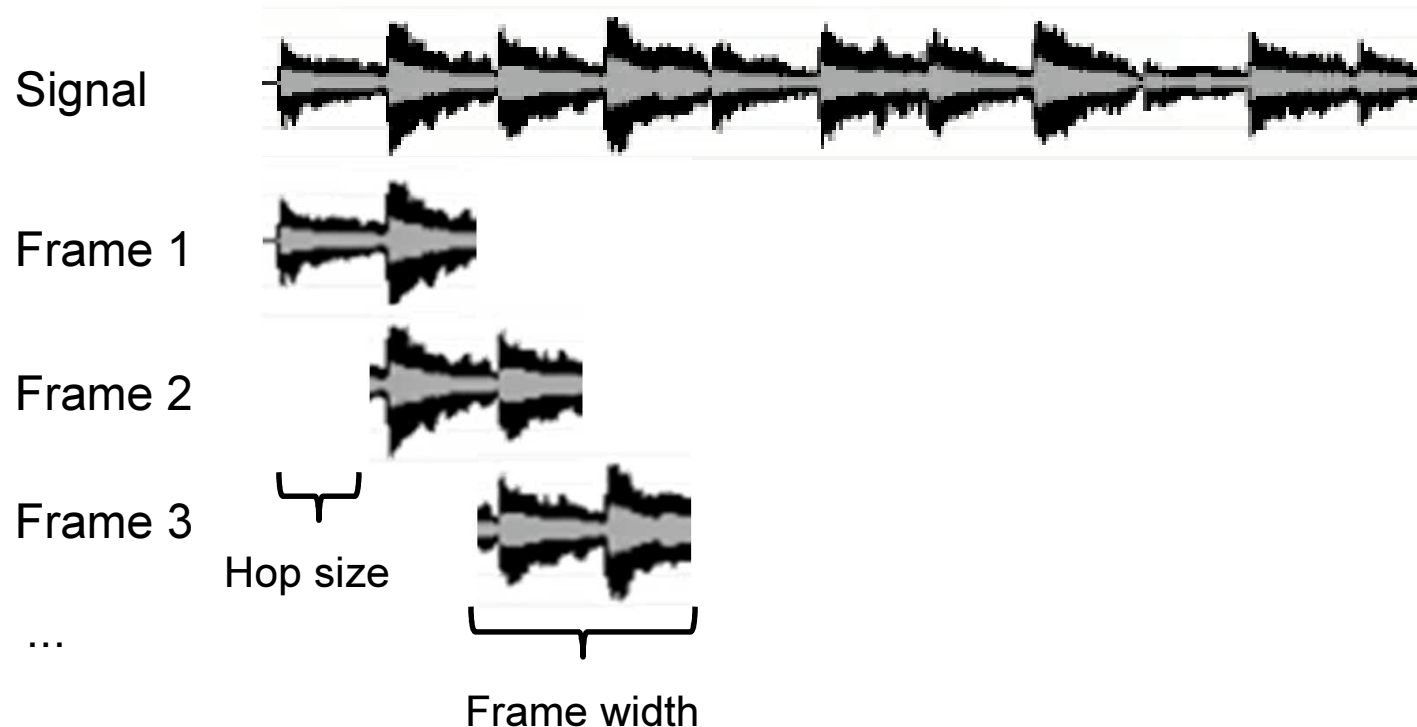


Analog-Digital-Conversion (ADC)



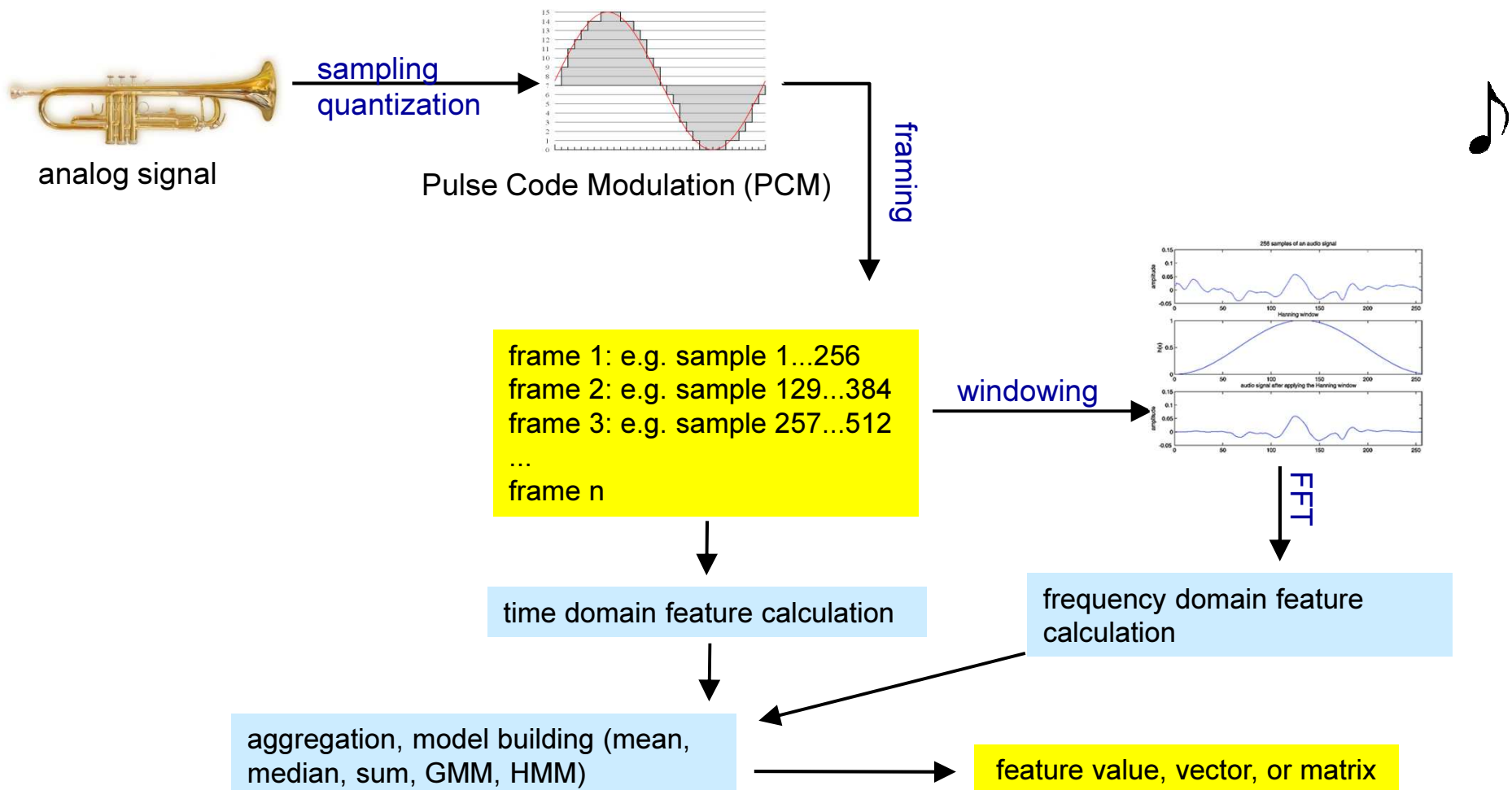
Pulse Code Modulation (PCM): analog signal is **sampled** at equidistant intervals (e.g., at a frequency of 44,100 Hz) and **quantized** in order to store it in digital form (here with 4 bits, typically ~16 bits)

Framing



In short-time signal processing, pieces of music are cut into segments of fixed length, called frames, which are processed one at a time; typically, a frame comprises 256 - 4096 samples.

Scheme of Content-Based Feature Extraction



Low-Level Feature: Zero Crossing Rate

Scope: time domain

$s(k)$...amplitude of k^{th} sample in time domain
 K ...frame size (number of samples in each frame)

Calculation:

$$ZCR_t = \frac{1}{2} \cdot \sum_{k=t \cdot K}^{(t+1) \cdot K - 1} |\text{sgn}(s(k)) - \text{sgn}(s(k+1))|$$

Description:

number of times the amplitude value changes its sign within frame t

Remarks:

commonly used as part of a low-level descriptor set

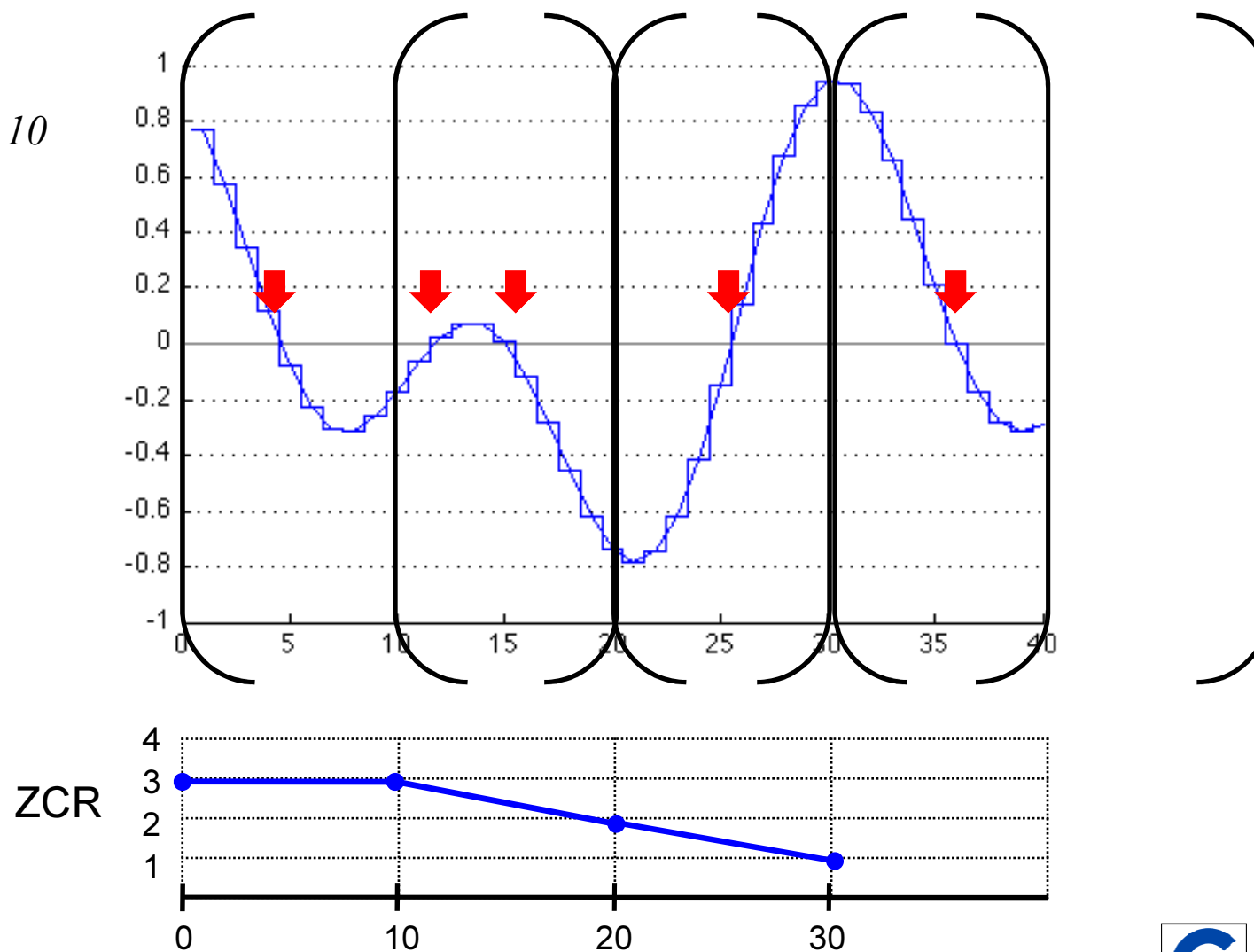
+ might be used as an indicator of pitch

+ sometimes stated to be an approximate measure of the signal's noisiness

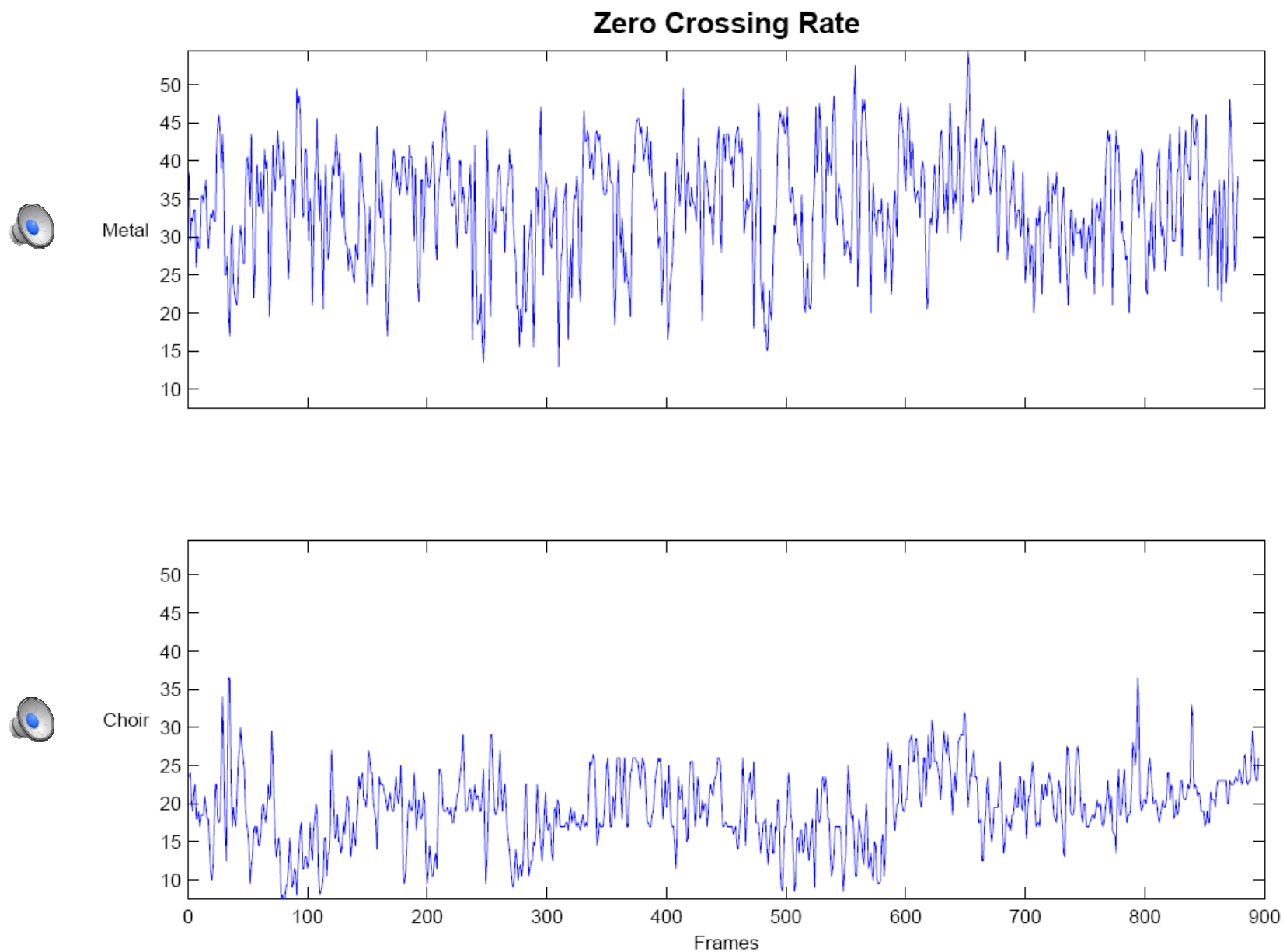
– in general, low discriminative power

Zero Crossing Rate: Illustration

$K=20$
 $\text{hop size} = 10$



Zero Crossing Rate: Examples



Department of
Computational
Cognition

Low-Level Feature: RMS Energy

Root-Mean-Square Energy (aka RMS power, RMS level, RMS amplitude)

Scope: time domain

Calculation:

$$RMS_t = \sqrt{\frac{1}{K} \cdot \sum_{k=t \cdot K}^{(t+1) \cdot K - 1} s(k)^2}$$

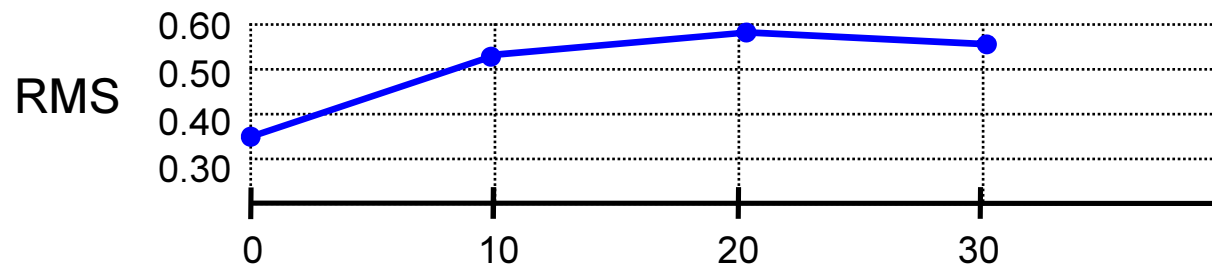
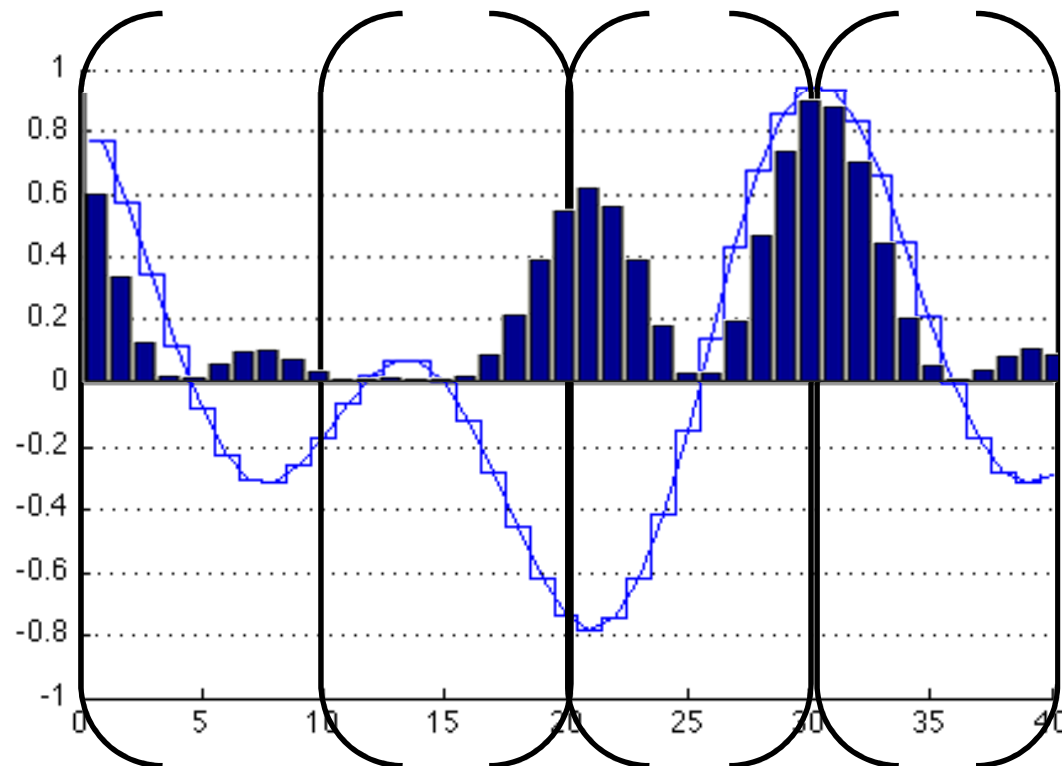
Remarks:

$s(k)$...amplitude of k^{th} sample in time domain
 K ...frame size (number of samples in each frame)

- + beat-related feature, can be used for beat detection
- + related to perceived intensity
- + good loudness estimation
- discriminative power not clear

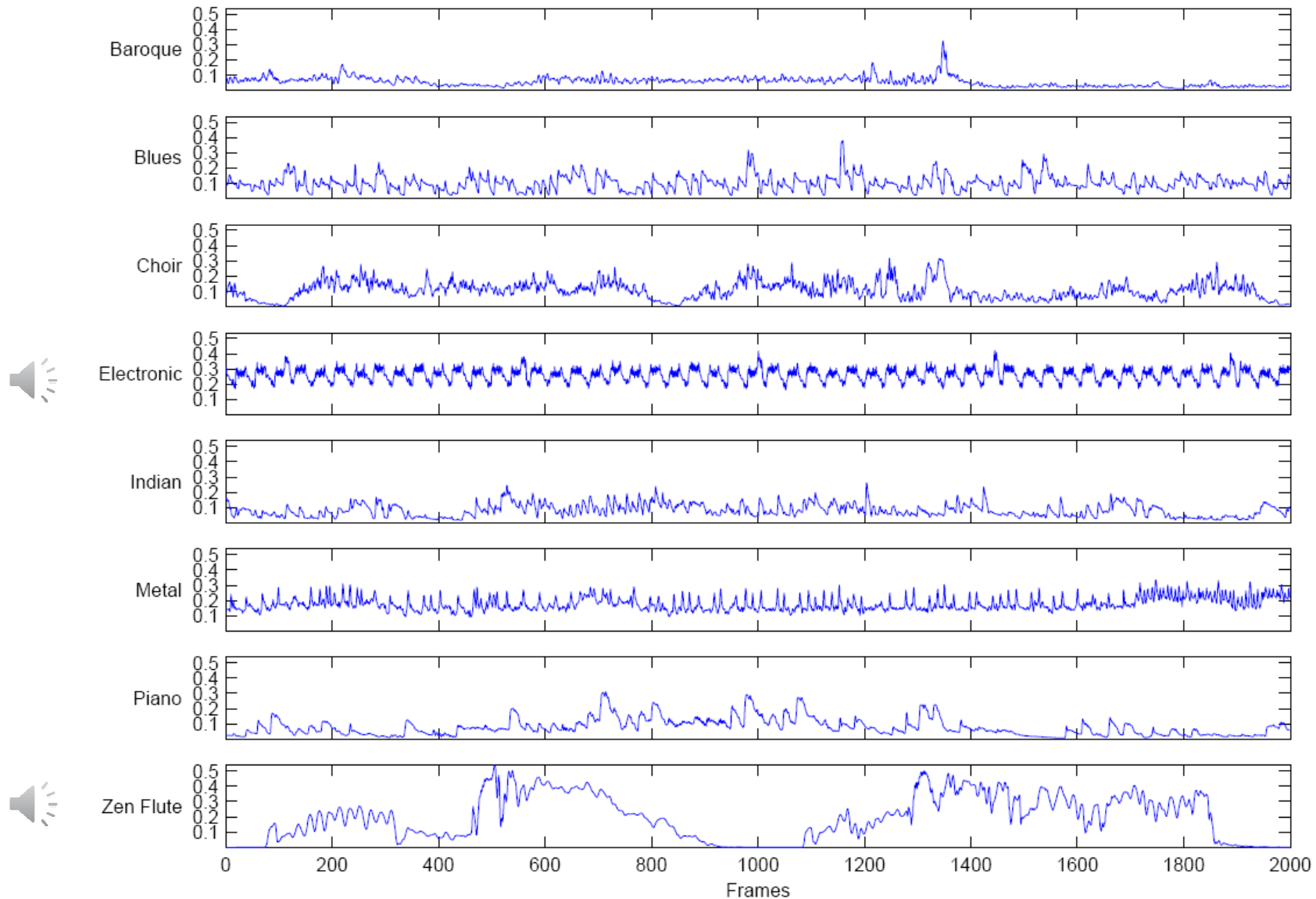
RMS Energy: Illustration

$K=20$
 $\text{hop size} = 10$



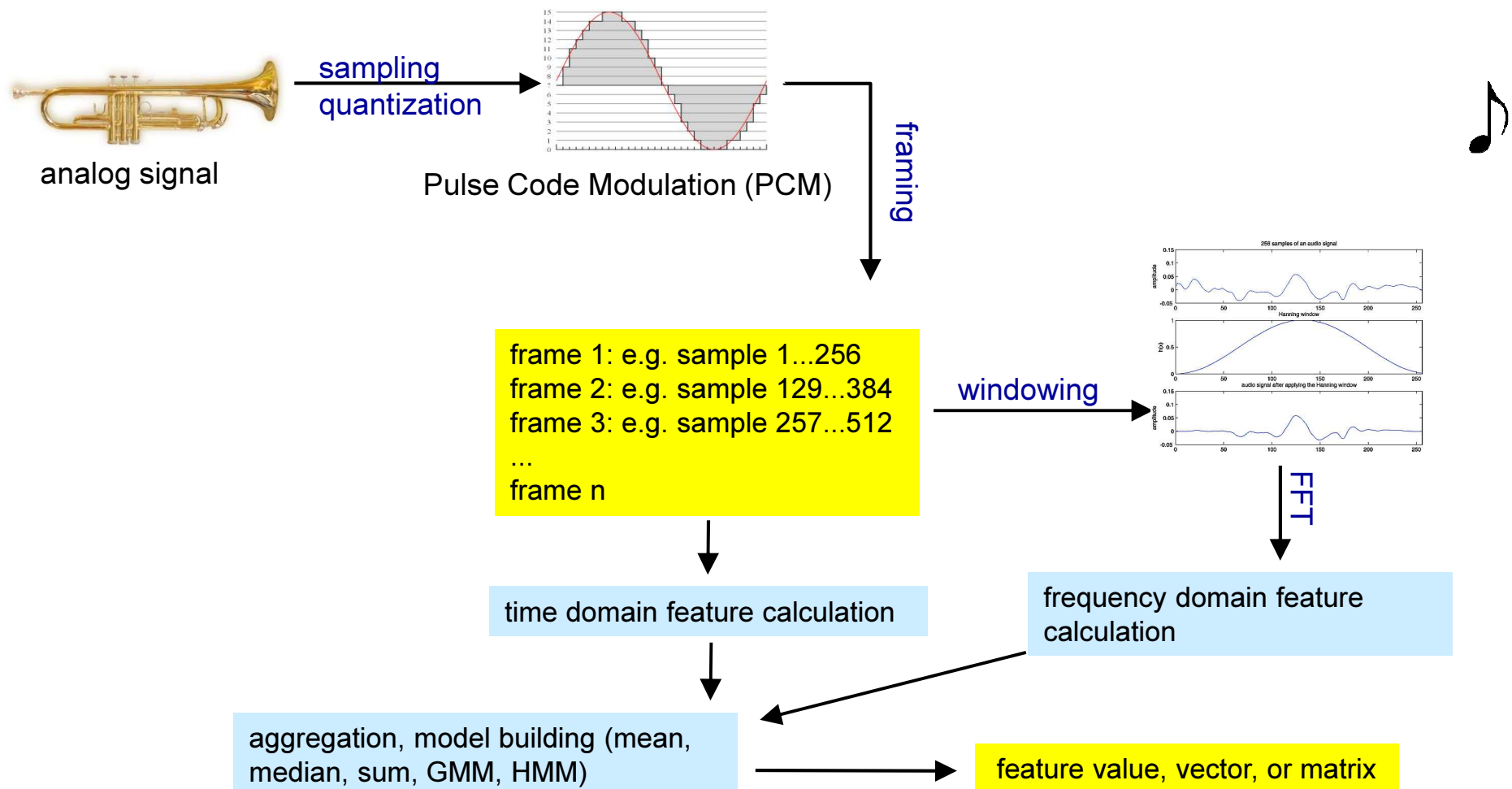
RMS Energy: Examples

Root Mean Square



Department of
Computational
Acoustics

Scheme of Content-Based Feature Extraction



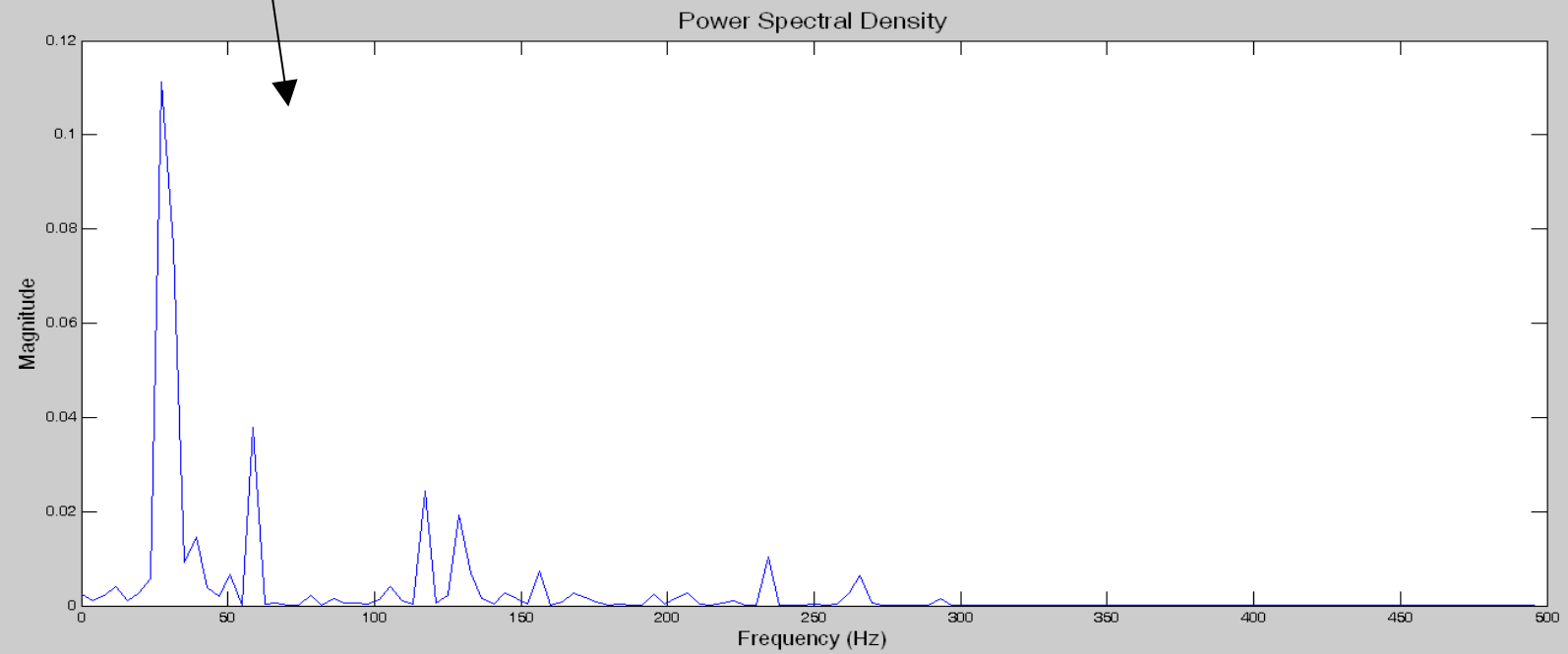
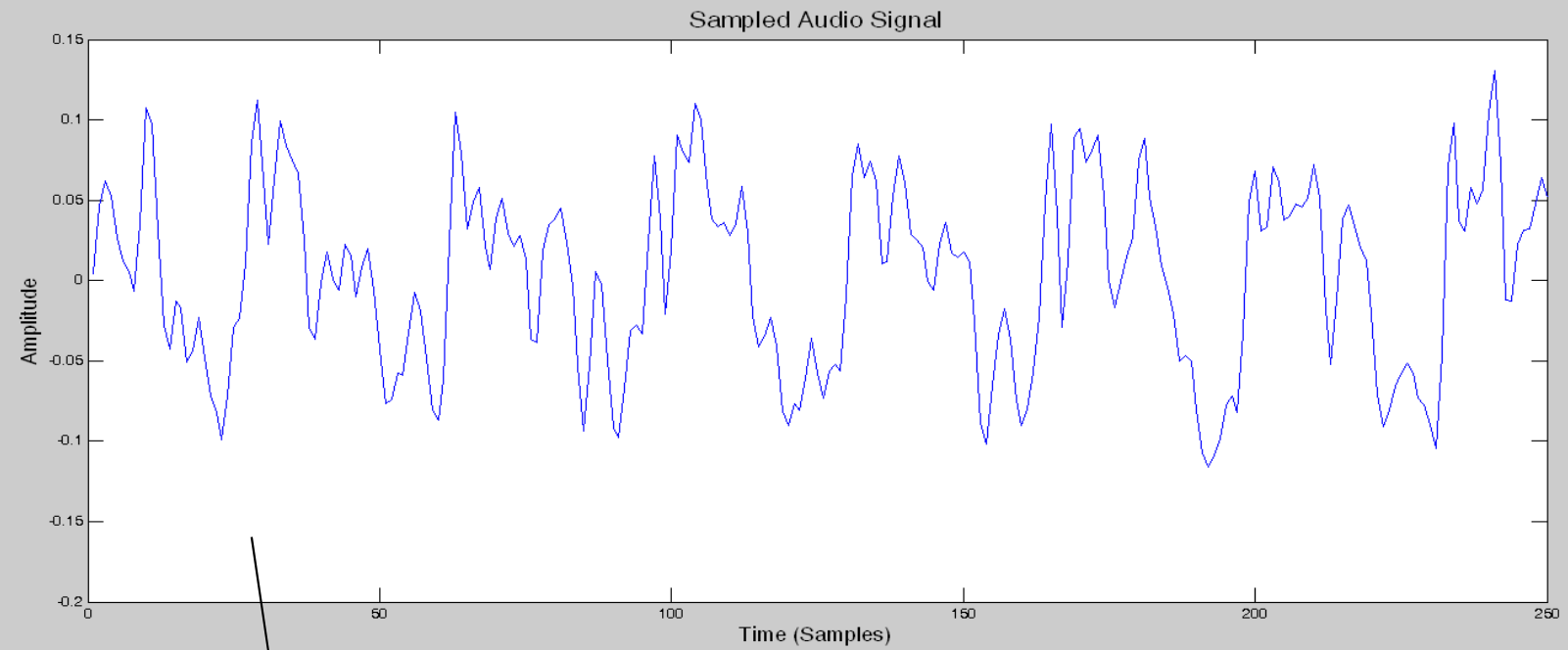
Fourier Transform

Transformation of the signal
from **time domain** (time vs. amplitude)
to **frequency domain** (frequency vs. magnitude)

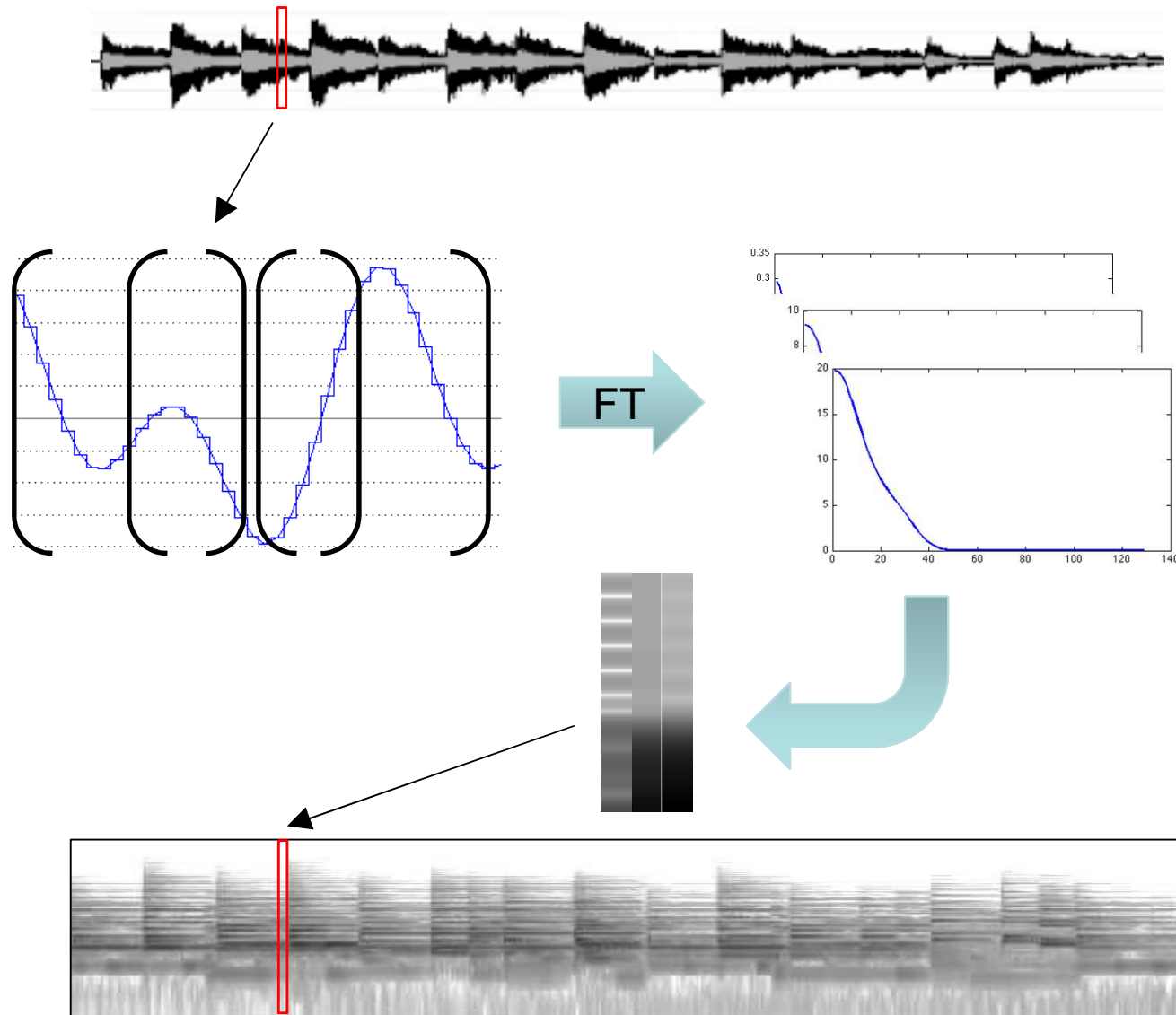


*Jean Baptiste
Joseph Fourier*

- Any (periodic) audio signal can be decomposed into an infinite number of overlapping waves
- Periodicity is achieved by multiplying the PCM magnitude values of each frame with a suited function, e.g., a Hanning window (**windowing**)
- In our case: **Discrete Fourier Transform (DFT)**
- In practice efficiently calculated via **Fast Fourier Transform (FFT)** (Cooley, Tukey; 1965)

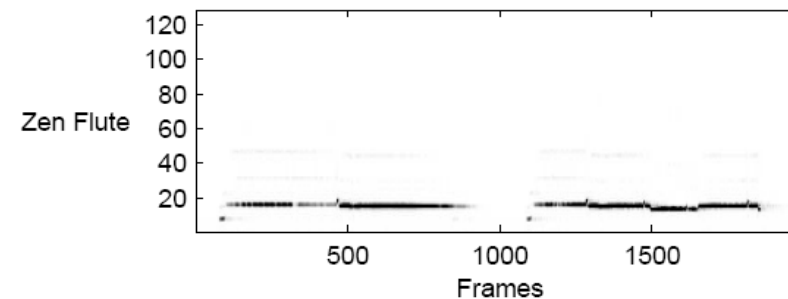
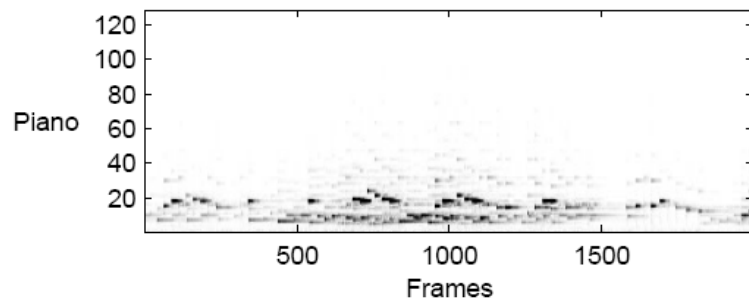
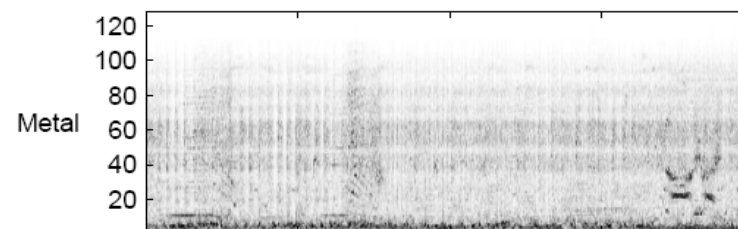
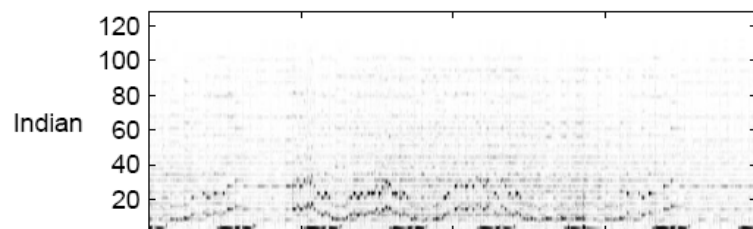
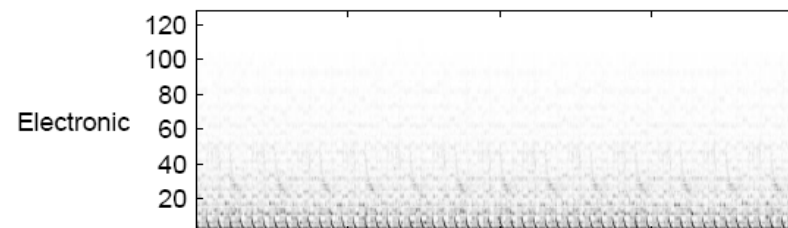
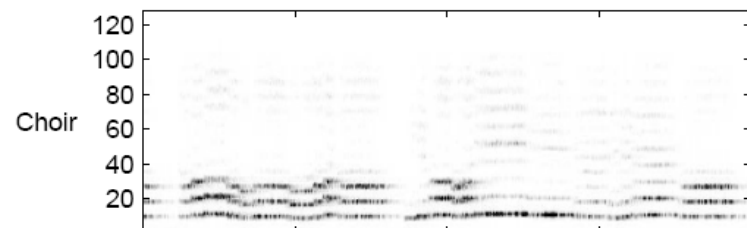
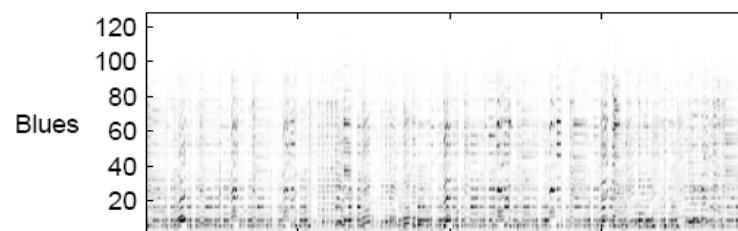
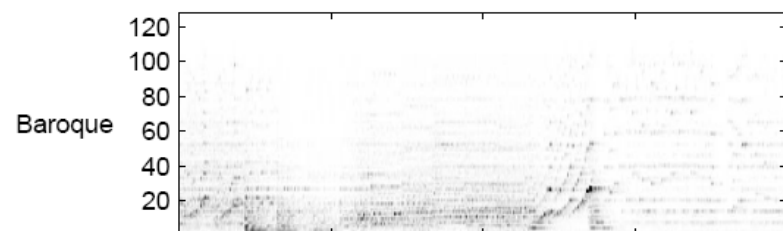


Spectrogram



Representation as STFT Spectrogram

STFT



Low-Level Feature: Spectral Centroid

Scope: frequency domain

Calculation:

$$C_t = \frac{\sum_{n=1}^N M_t(n) \cdot n}{\sum_{n=1}^N M_t(n)}$$

$M_t(n)$...magnitude in frequency domain at frame t and frequency bin n

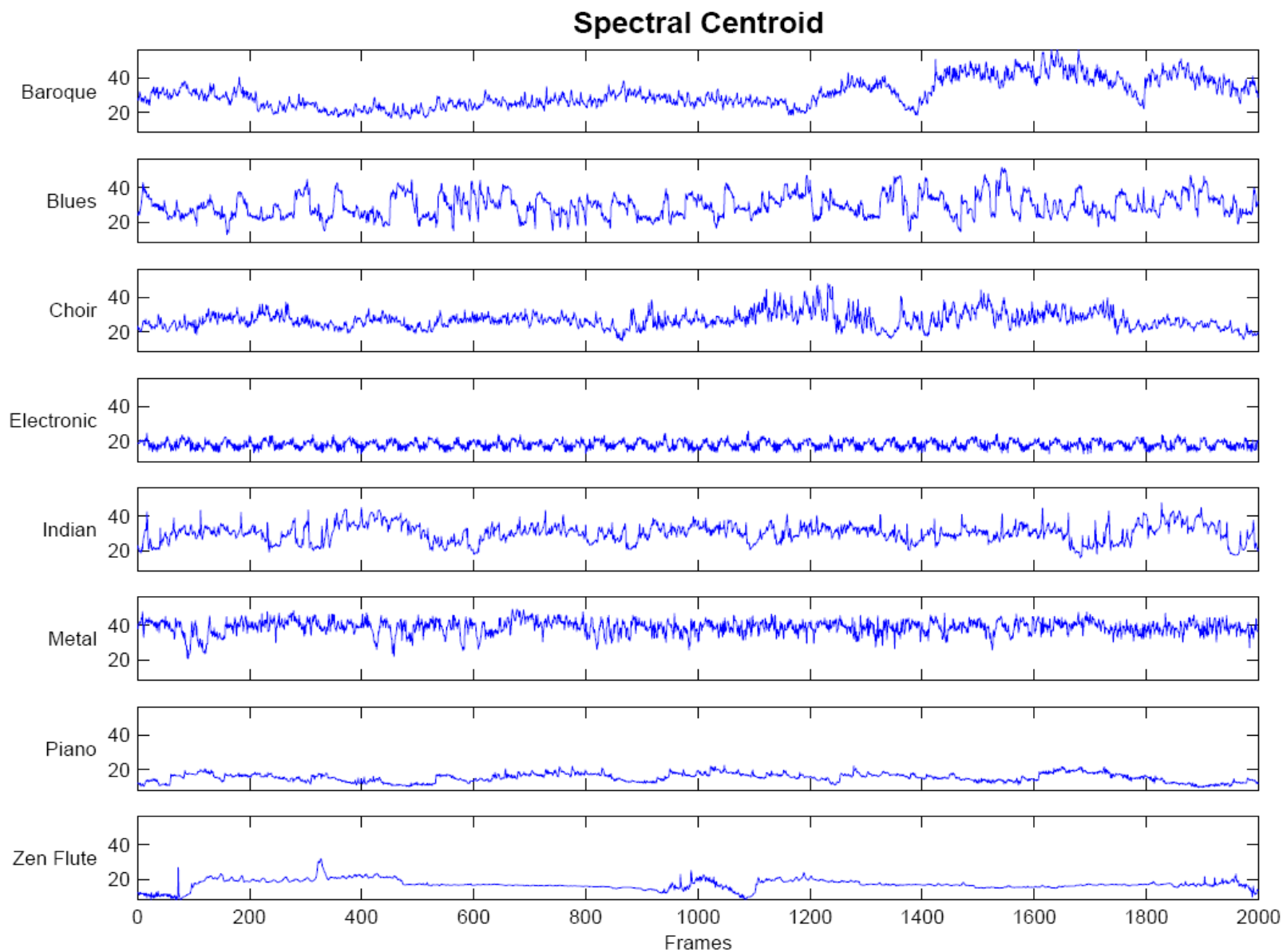
N ...number of highest frequency band

Description: center of gravity of the magnitude spectrum of the DFT, i.e. the frequency (band) region where most of the energy is concentrated

Remarks:

used as measure of sound sharpness (strength of high frequency energy)
– sensitive to low pass filtering (downsampling) as the high frequency bands are given more weight

Spectral Centroid: Illustration



ment of
utational
ption

Mid-level feature extraction and similarity calculation

Pitch Class Profiles: related to Western music tone scale, melodic retrieval

MFCCs: related to timbral properties, frame-level

Block-Level Features

- Fluctuation Patterns: related to rhythmic/periodic properties

Mid-level Feature Processing Overview

Convert signal to *frequency domain*, e.g.,
using an FFT

(Psycho)acoustic transformation

(Mel-scale, Bark-scale, Cent-scale, ...):

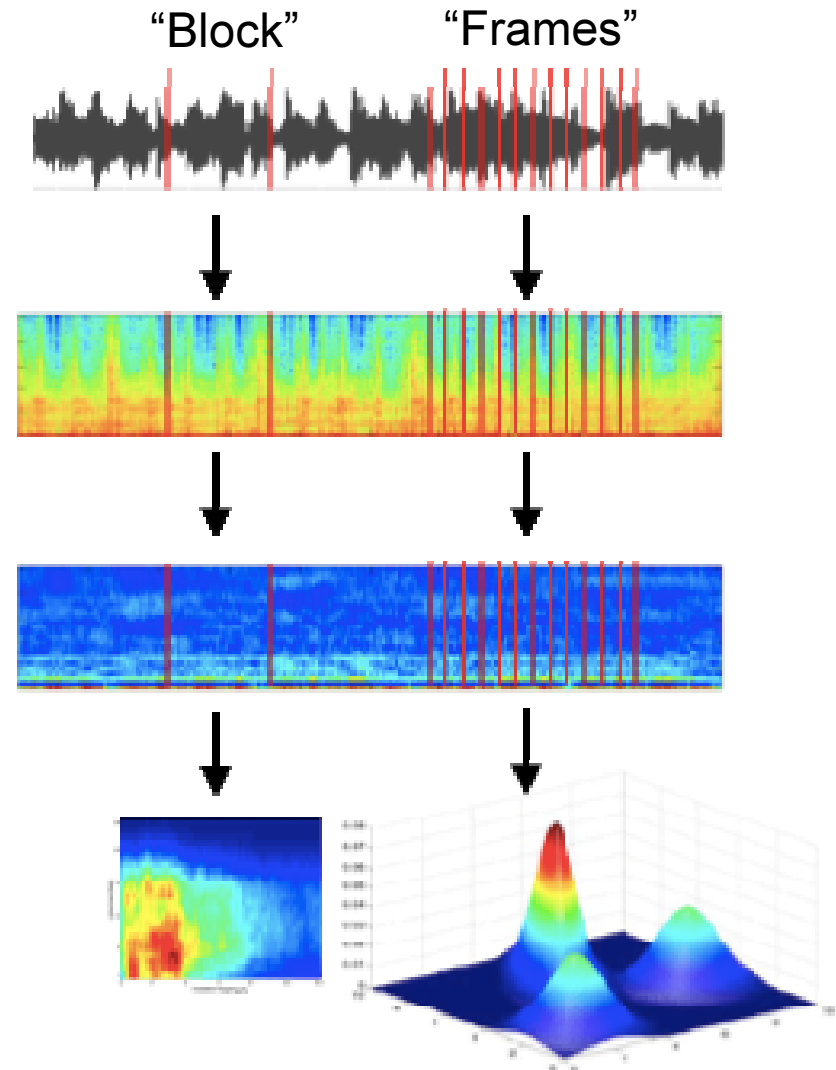
mimics human listening process

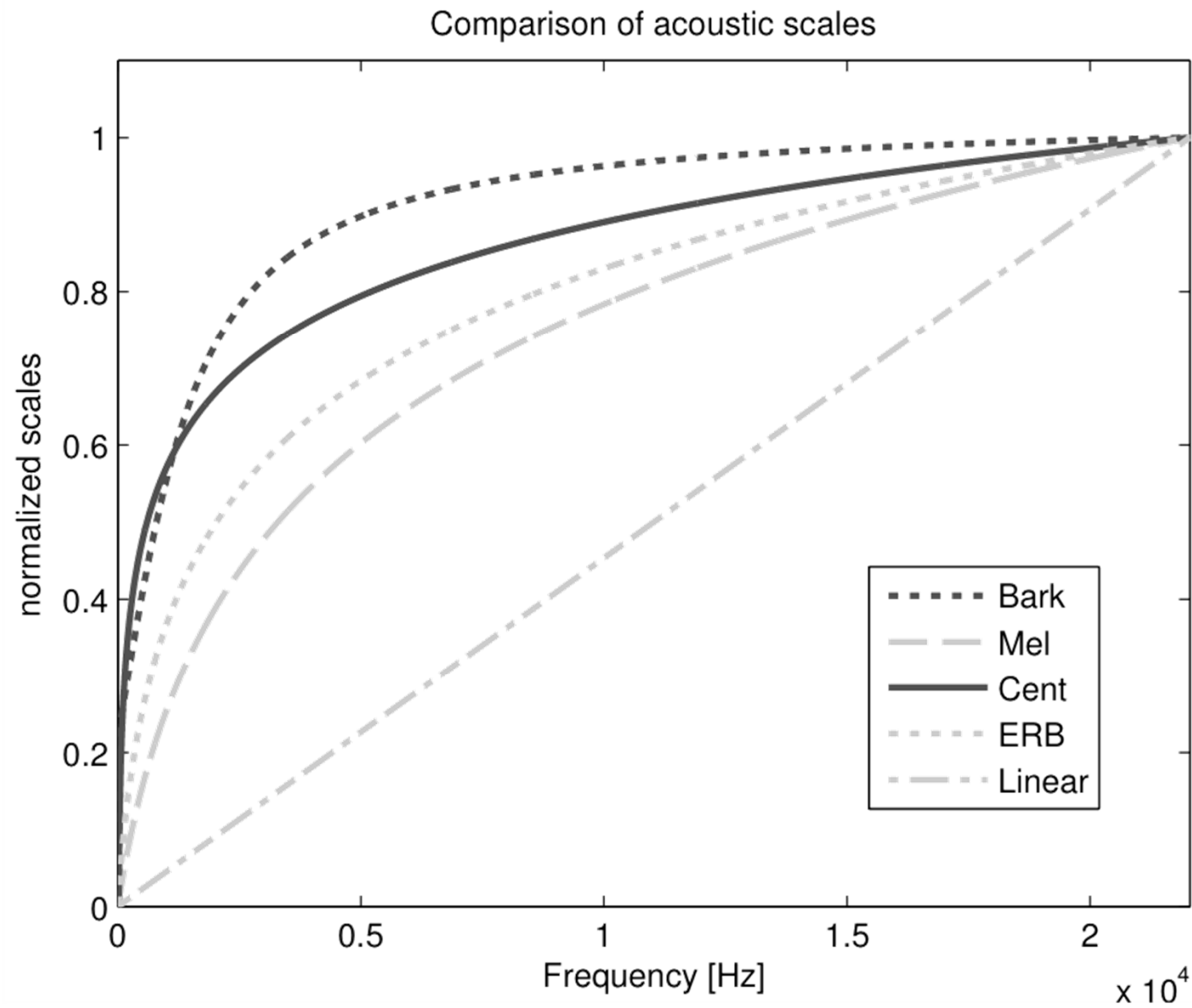
(not linear, but logarithmic!),

removes aspects not perceived by humans,
emphasizes low frequencies

Extract features

- *Frame-level*
(short time windows, e.g., 25 ms)
needs feature distribution model
- *Block-level*
(large time windows, e.g., 6 sec)





Mid-level feature extraction and similarity calculation

Pitch Class Profiles: related to Western music tone scale, melodic retrieval

MFCCs: related to timbral properties, frame-level

Block-Level Features

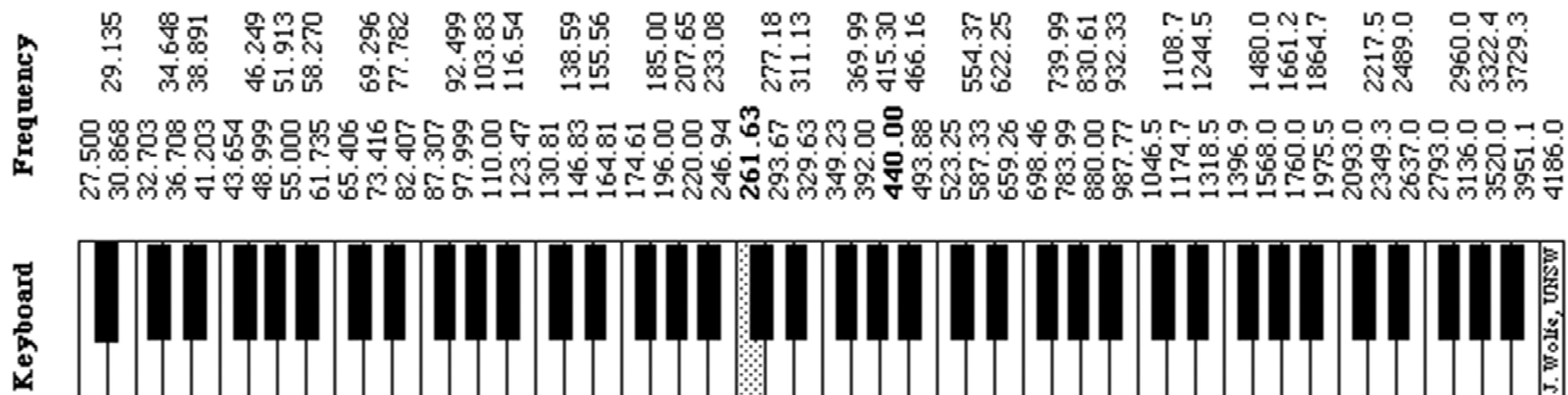
- Fluctuation Patterns: related to rhythmic/periodic properties

Pitch Class Profiles

(Fujishima; 1999)

(aka *chroma vectors*)

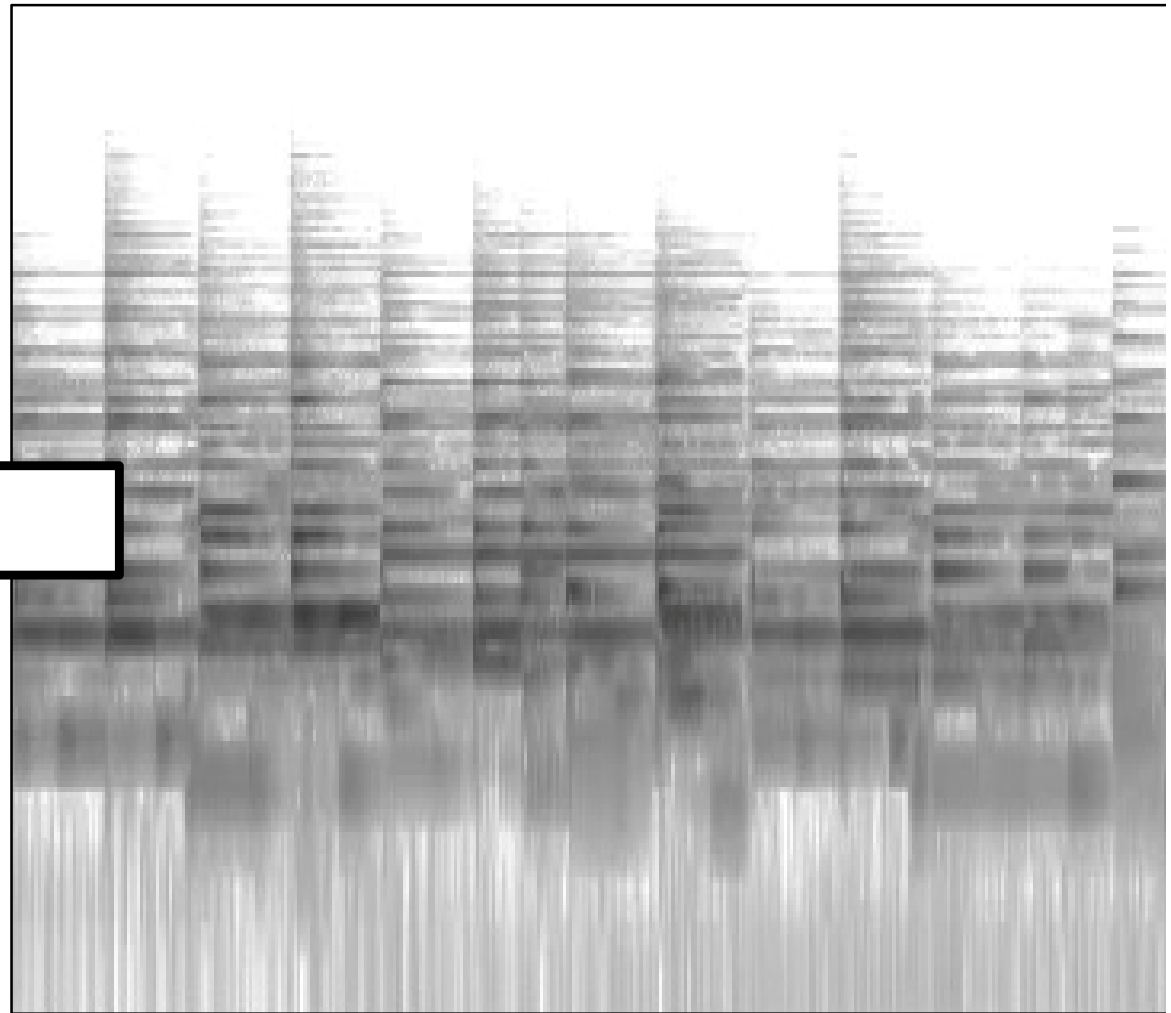
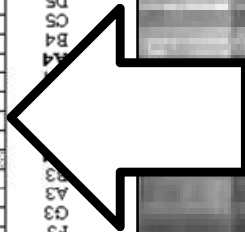
- Transforming the frequency activations into well known musical system/representation/notation
- Mapping to the equal-tempered scale (each semitone equal to one twelfth of an octave)
- For each frame, get intensity of each of the 12 semitone (pitch) classes



Mapping Frequencies to Semitones



Frequency	Keyboard	Note name
27.500		A0
29.135		B0
30.868		C1
32.703		D1
34.648		E1
36.708		F1
38.891		G1
41.203		A1
43.654		B1
46.249		C2
48.999		D2
51.913		E2
55.000		F2
58.270		G2
61.735		A2
65.406		B2
69.296		C3
73.416		D3
77.782		E3
82.407		F3
87.307		G3
92.499		A3
97.999		B3
103.83		C4
110.00		D4
116.54		E4
123.47		F4
130.81		G4
138.59		A4
146.83		B4
155.56		C5
164.81		D5
174.61		E5
185.00		F5
196.00		G5
207.65		A5
220.00		B5
233.08		C6
246.94		D6
261.63		E6
277.18		F6
293.67		G6
311.13		A6
329.63		B6
349.23		C7
369.99		D7
392.00		E7
415.30		F7
440.00		G7
466.16		A7
493.88		B7
523.25		C8
554.37		
587.33		
622.25		
659.26		
698.46		
739.99		
783.99		
830.61		
880.00		
932.33		
987.77		
1046.5		
1108.7		
1174.7		
1244.5		
1318.5		
1396.9		
1480.0		
1568.0		
1661.2		
1760.0		
1864.7		
1975.5		
2093.0		
2217.5		
2349.3		
2489.0		
2637.0		
2793.0		
2960.0		
3136.0		
3322.4		
3520.0		
3729.3		
3951.1		
4186.0		



Semitone Scale

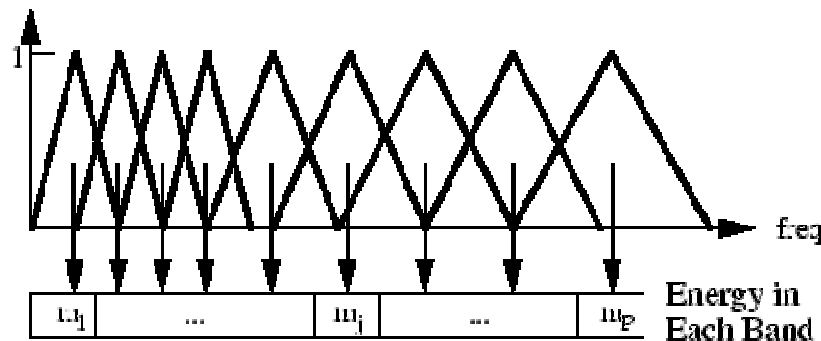
Map data to semitone scale to represent (western) music

Frequency doubles for each octave

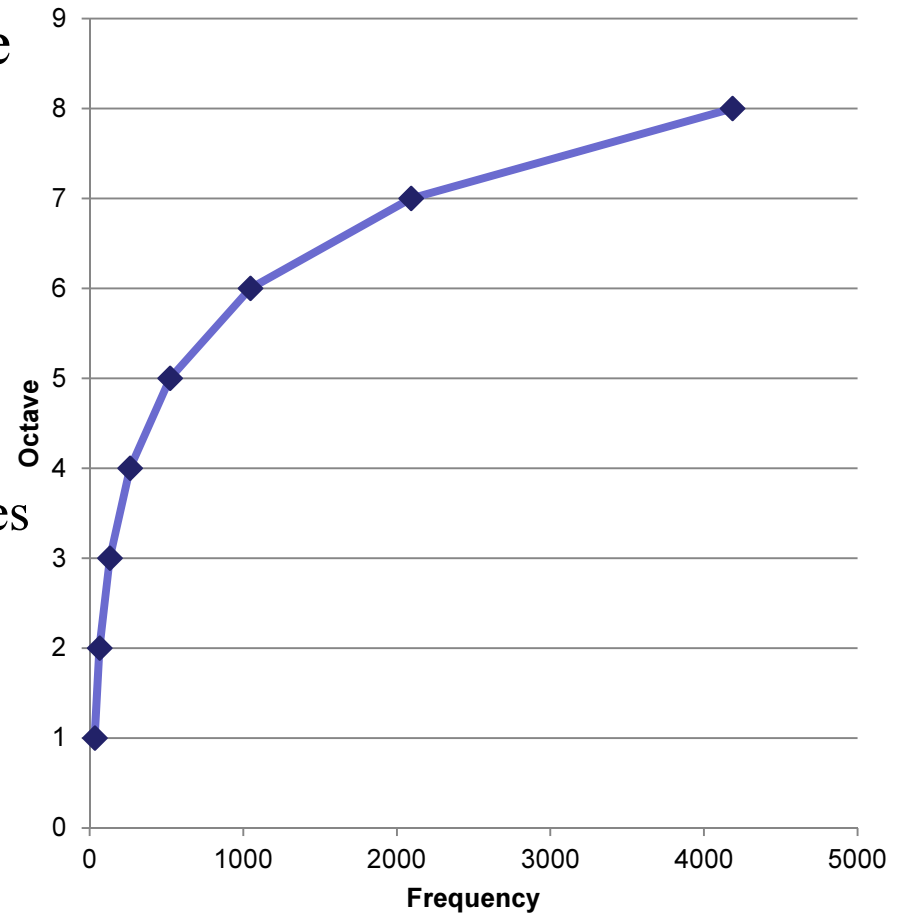
- e.g. pitch of A3 is 220 Hz, compared to 440 Hz of A4

Mapping, e.g., using filter bank with triangular filters

- centered on pitches
- width given by neighboring pitches
- normalized by area under filter

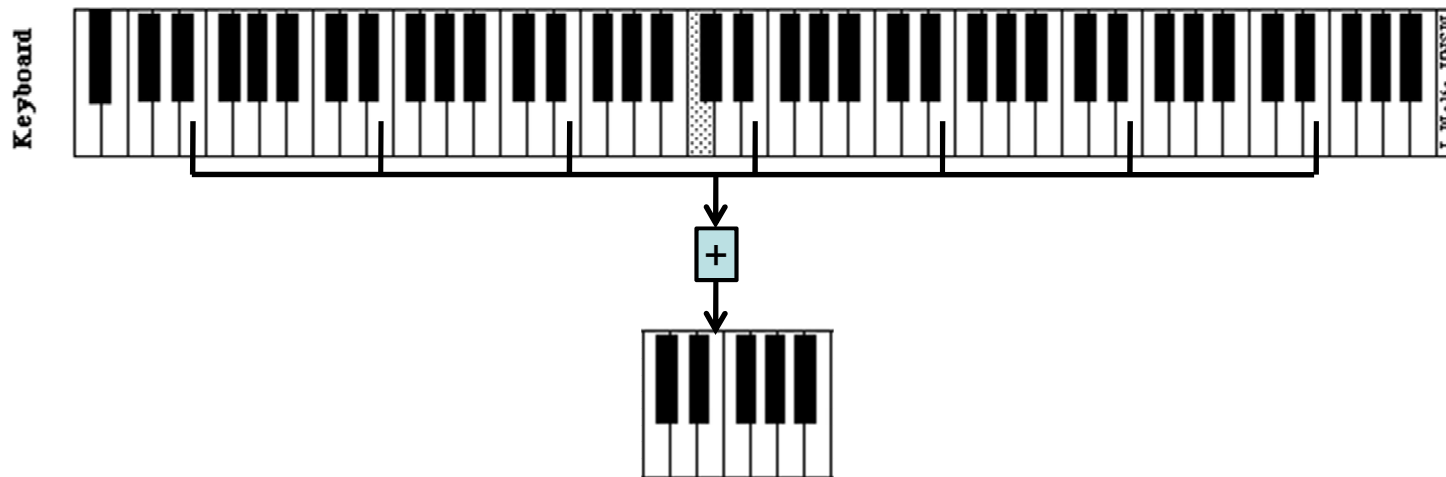


The note C in different octaves vs. frequency



Pitch Class Features

Sum up activations that belong to the **same class of pitch**
(e.g., all A, all C, all F#)

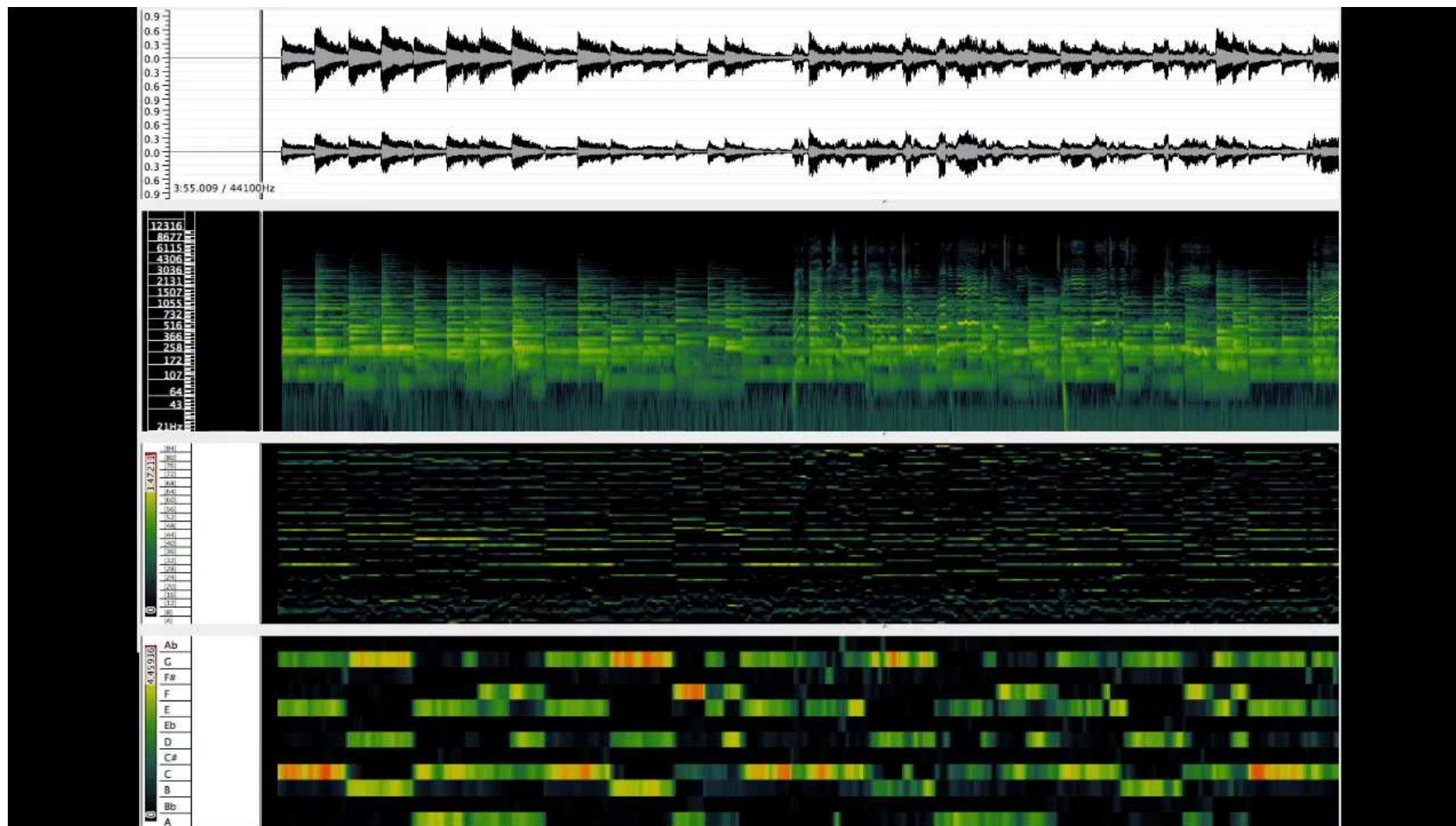


Results in a 12-dimensional feature vector for each frame

PCP feature vectors describe tonality

- Robust to noise (including percussive sounds)
- Independent of timbre (~ played instruments)
- Independent of loudness

Pitch Class Profiles in Action



Sonic Visualizer by QMUL, C4DM; <http://www.sonicvisualiser.org>

Application: Automatic Page Turner



Music Retrieval Scenarios

PCPs used in classification, key/chord estimation, melody extraction and retrieval (e.g., for **cover song retrieval**, i.e., finding songs that are based on the same melody/tune, independent of instrumentation)

Another scenario is to find different songs that nevertheless “**sound similar**” (frequently related to timbre), but MFCCs have shown to be better descriptors for this task

Mid-level feature extraction and similarity calculation

Pitch Class Profiles: related to Western music tone scale, melodic retrieval

MFCCs: related to timbral properties, frame-level

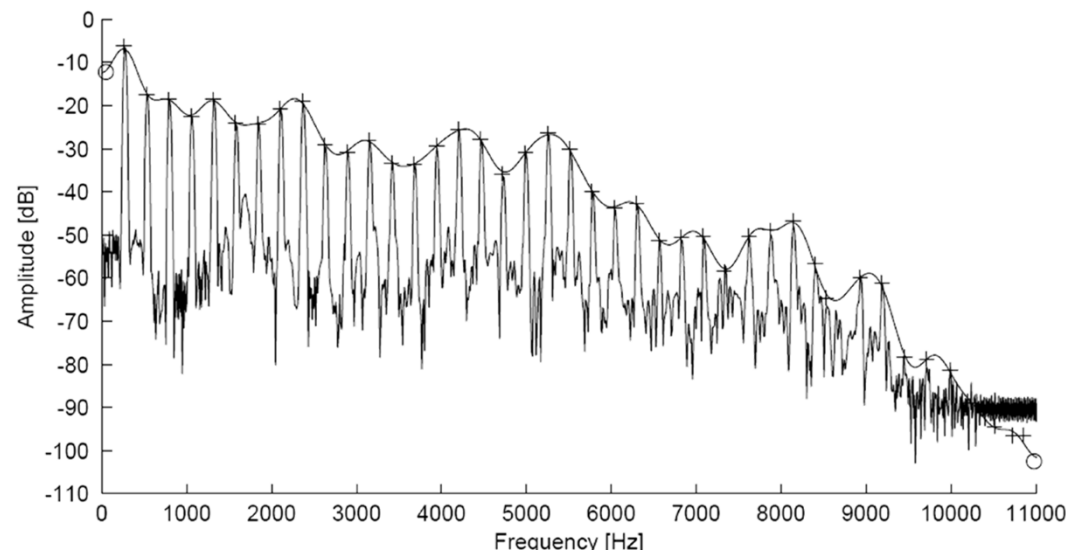
Block-Level Features

- Fluctuation Patterns: related to rhythmic/periodic properties
- Correlation Patterns: temporal relation of frequencies

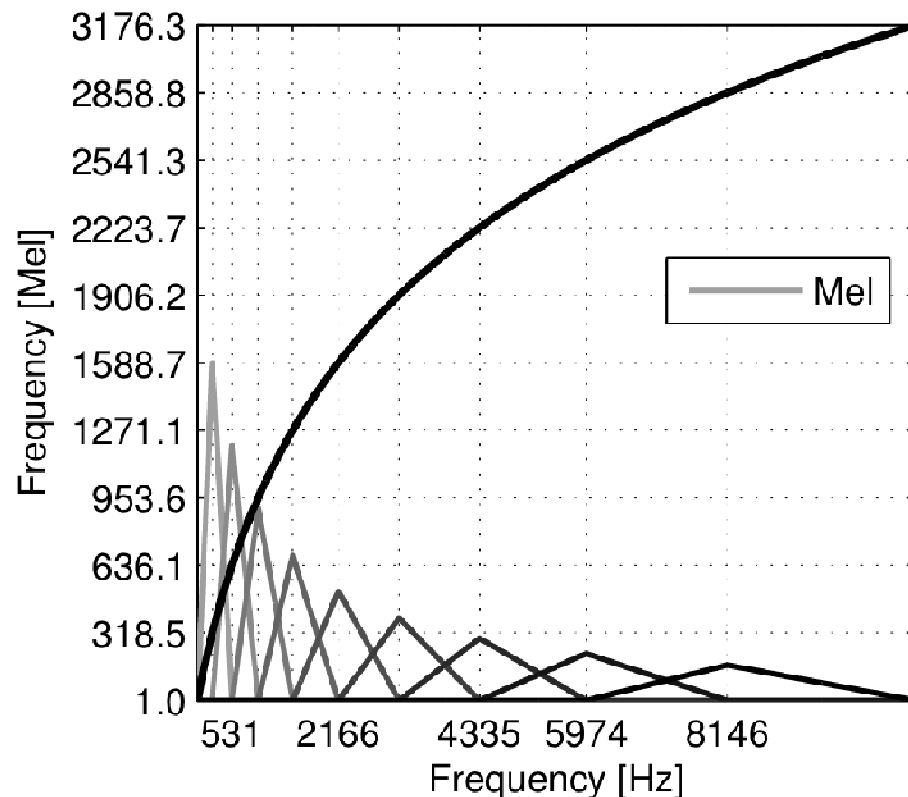
MFCCs

Mel Frequency Cepstral Coefficients (MFCCs) have their roots in speech recognition and are a way to represent the *envelope of the power spectrum* of an audio frame

- the spectral envelope captures perceptually important information about the corresponding sound excerpt (*timbral aspects*)
- most important for music similarity: sounds with similar spectral envelopes are generally perceived as similar.



The Mel Scale

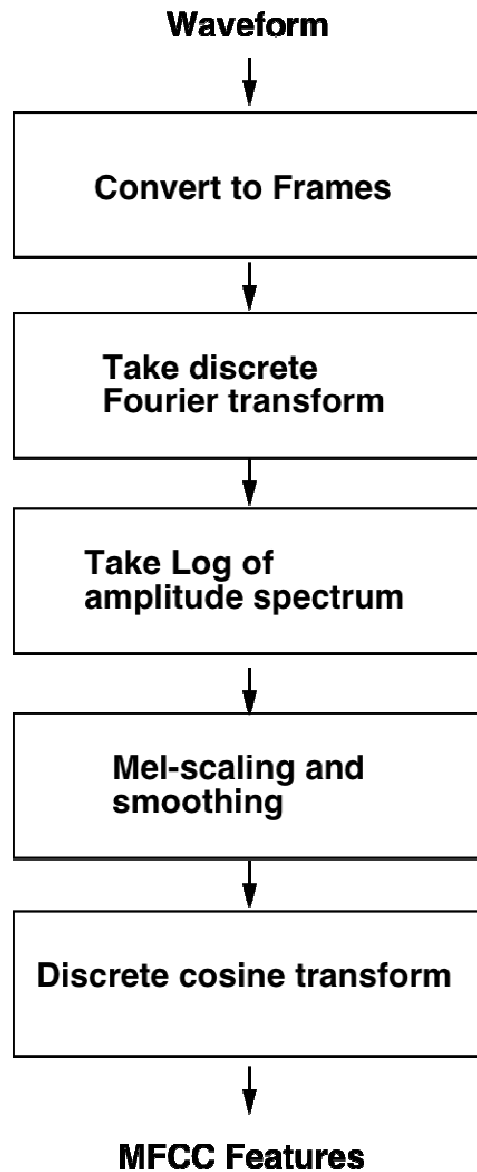


Perceptual scale of pitches judged by listeners to be equal in distance from one another

Given Frequency f in Hertz, the corresponding pitch in Mel can be computed by

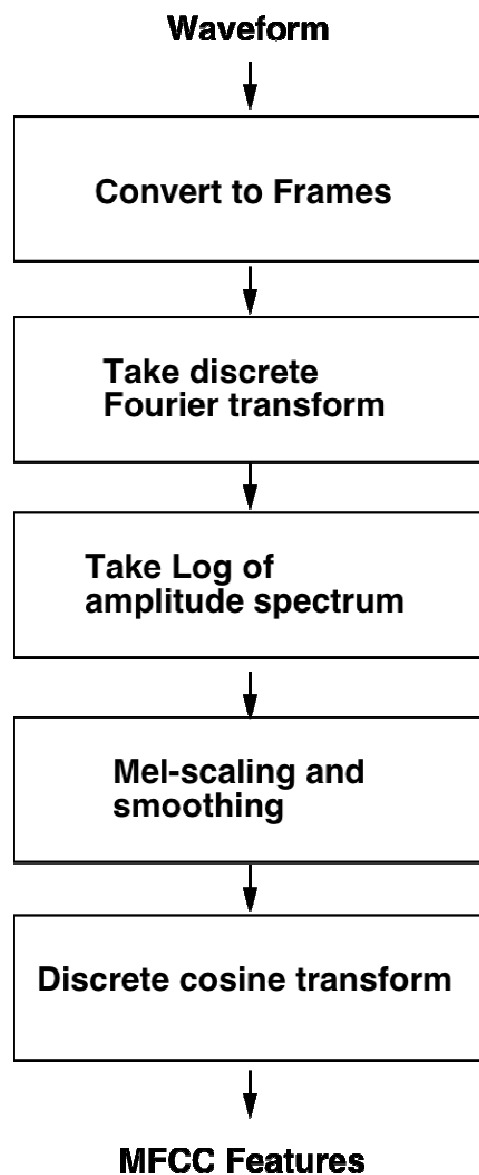
$$m = 2595 \log_{10} \left(1 + \frac{f}{700} \right)$$

MFCCs



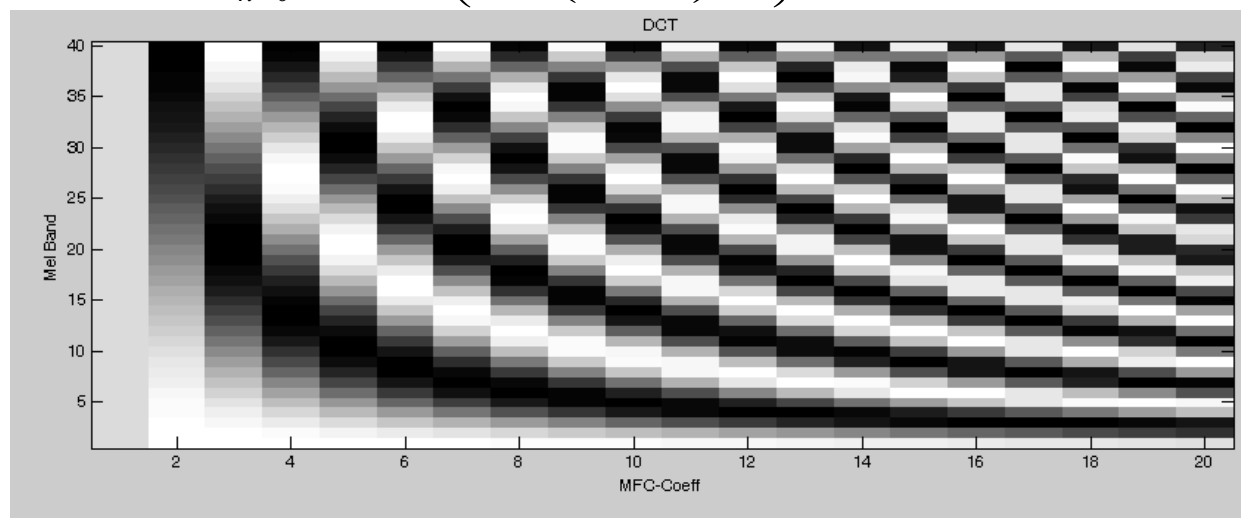
MFCCs are computed per frame

1. STFT: short-time Fourier transform
2. the logarithm of the amplitude spectrum is taken (motivated by the way we humans perceive loudness)
3. mapping of the amplitude spectrum to the Mel scale
4. quantize (e.g., 40 bins) and make linear (DCT doesn't operate on log scale)



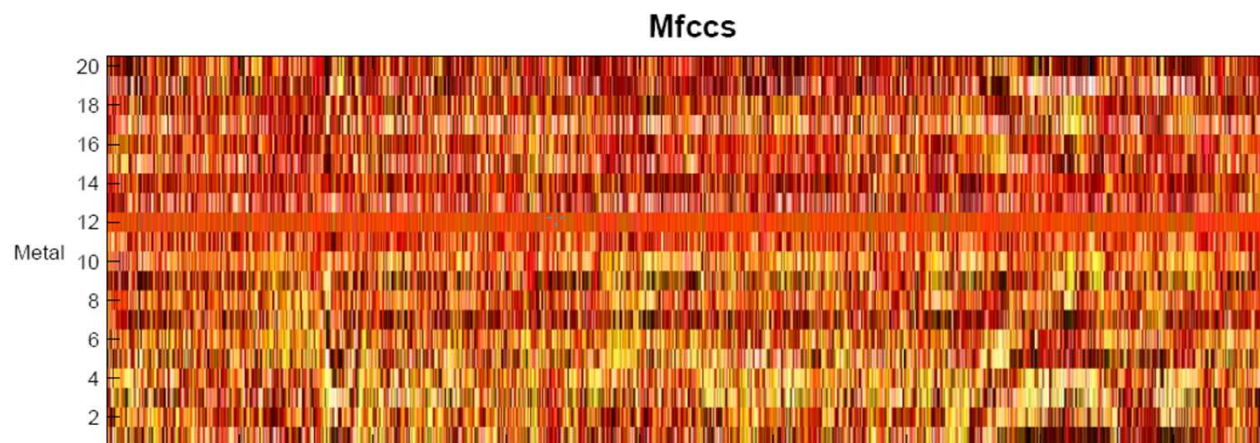
5. perform Discrete Cosine Transform to de-correlate the Mel-spectral vectors
- similar to FFT; only real-valued components
 - describes a sequence of finitely many data points as sum of cosine functions oscillating at different frequencies
 - results in N coefficients (e.g., $N = 20$)

$$X_k = \sum_{n=0}^{N-1} x_n \cdot \cos\left(\frac{\pi}{N} \cdot \left(n + \frac{1}{2}\right) \cdot k\right) \quad k = 0, \dots, N-1$$

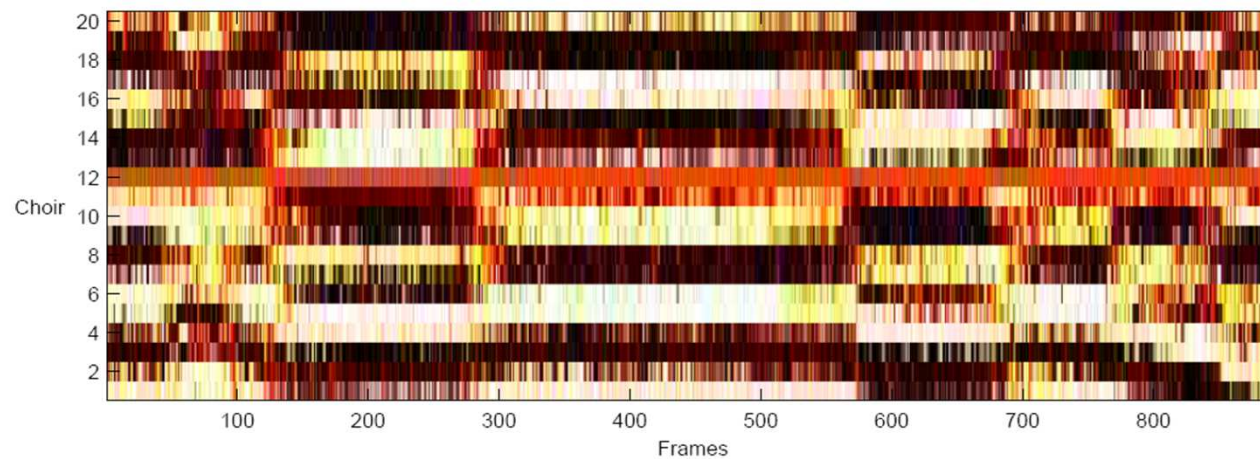


MFCC Examples

Metal



Choir



“Bag-of-frames” Modeling

Full music piece is now a set of MFCC vectors; number of frames depends on length of piece

Need summary/aggregation/modeling of this set

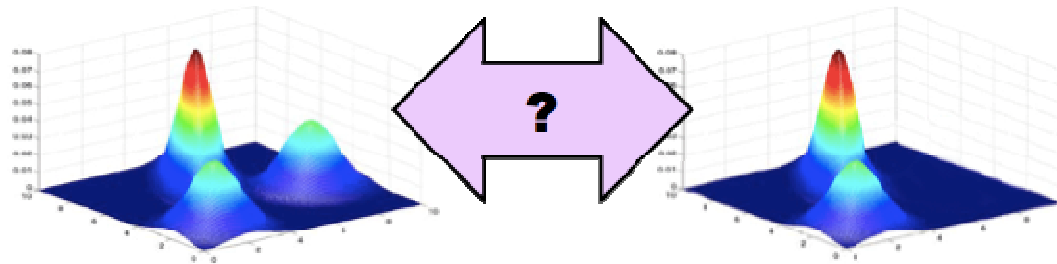
- Arithmetic mean over all frames? Sum?

Statistically model the distribution of all these local features:

- State-of-the-art until 2005: learn a Gaussian Mixture Model (GMM)
- a GMM estimates a probability density as the weighted sum of M simpler Gaussian densities, called components of the mixture
- each song is modeled with a GMM
- the parameters of the GMM are learned with the classic Expectation-Maximization (EM) algorithm
 - this can be considered a shortcoming of this approach as this step is very time consuming

“Bag-of-frames” Modeling

Comparing two GMMs is non-trivial and expensive



- The Kullback-Leibler divergence can be used (approximated)

$$D_{KL}(P||Q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx$$

- Basically, this requires to (Monte-Carlo) sample one GMM and calculate the likelihood of these observations under the other model and vice versa (non-deterministic, slow)

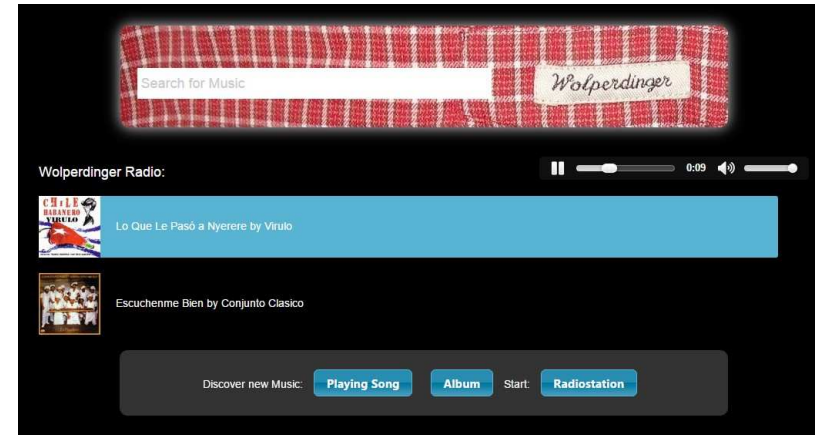
“Bag-of-frames” Modeling

State-of-the-Art since 2005: Single Gaussian Model

- mean and covariance matrix to model a whole piece
- closed-form solution for KL divergence exists
- much more efficient than GMM
(instantaneous retrieval of 10Ks of pieces)
- applicable to real-world tasks (e.g. query-by-example or automated playlist generation)

Large-scale Music Retrieval

- Research prototype: “Wolperdinger”
- Retrieval in real-time
 - full database ~2.3m songs
 - played song model compared to all whenever played
 - no caching necessary
- Single Gaussian MFCC as music similarity measure
- FastMap on feature vectors (vectorized cov. matrix of SG model) for highly efficient retrieval
- Similar approaches in commercial applications (e.g., <http://fm4.orf.at/soundpark>)



Limitations of Bag-of-Frames Approaches

Loss of temporal information:

- temporal ordering of the MFCC vectors is completely lost because of the distribution model (bag-of-frames)
- possible approach: calculate delta-MFCCs to preserve difference between subsequent frames

Hub problem (“always similar problem”)

- depending on the used features and similarity measure, some songs will yield high similarities with many other songs without actually sounding similar (requires post-processing, e.g., rectify the similarity space)
- general problem in high-dimensional feature spaces

Mid-level feature extraction and similarity calculation

Pitch Class Profiles: related to Western music tone scale, melodic retrieval

MFCCs: related to timbral properties, frame-level

Block-Level Features

- Fluctuation Patterns: related to rhythmic/periodic properties

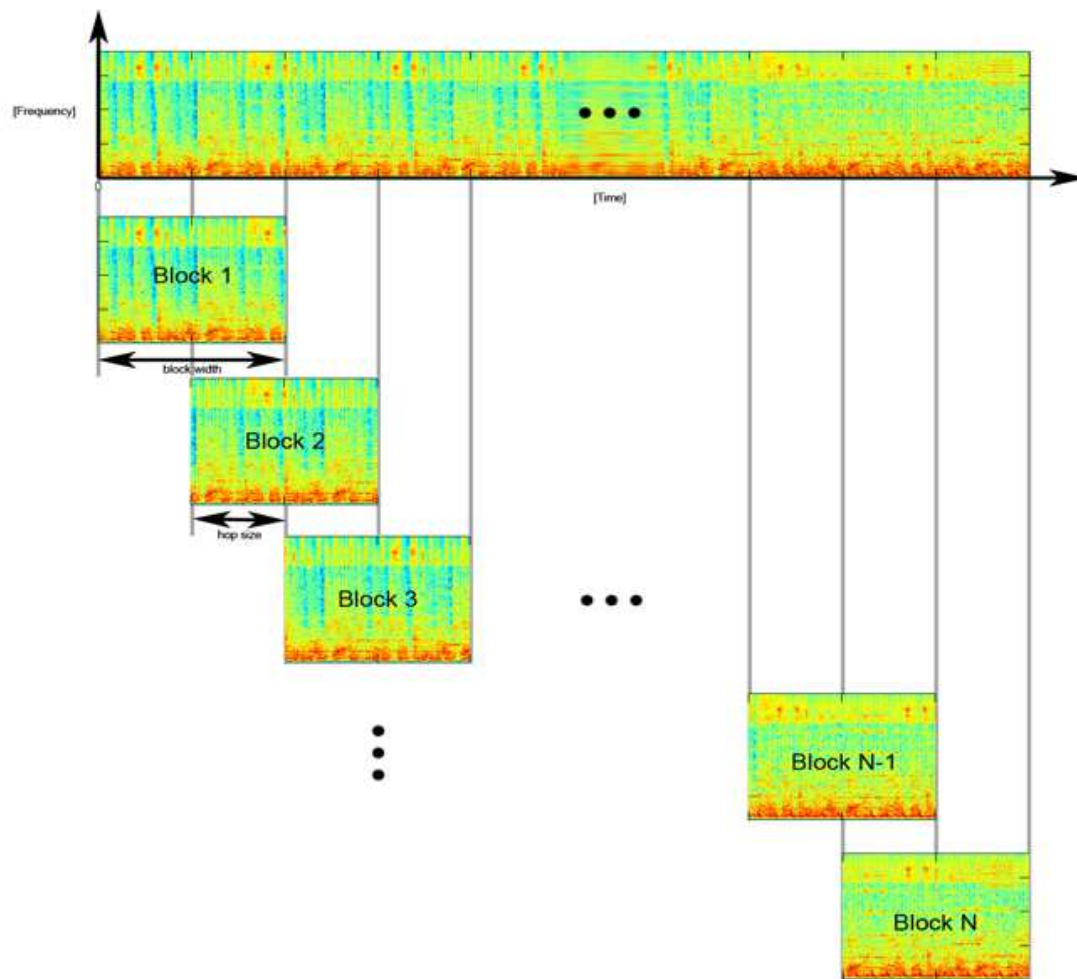
Block-Level Features

Instead of processing single frames, compute features on larger blocks of frames

- blocks are defined as consecutive sequences of audio frames
- thus features are (to some extent) able to capture local temporal information

Afterwards the blocks are summarized to form a generalized description of the piece of music

Several features defined in “Block-Level Framework” (Seyerlehner; 2010); Ex.: Fluctuation Patterns (Pampalk; 2001)



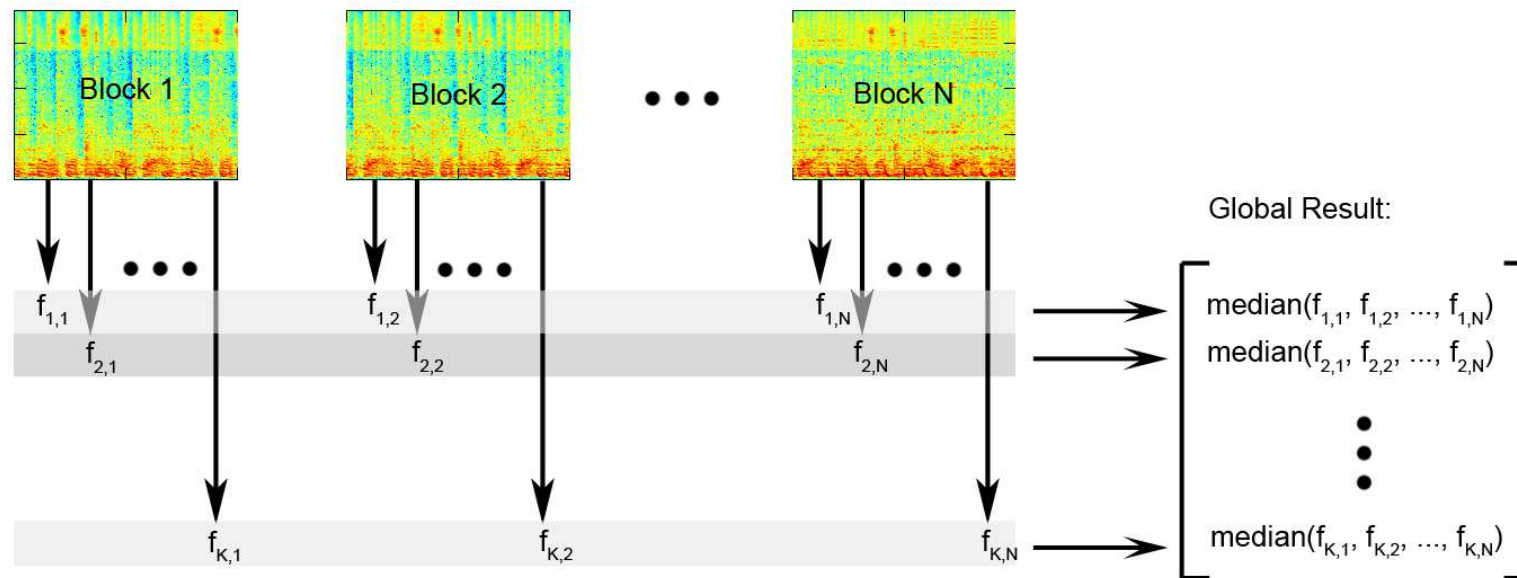
Block Processing

- The whole spectrum is processed in terms of blocks
- Each block consists of a fixed number of frames (block size W)
- Number of rows H is defined by the frequency resolution
- Blocks may overlap (hop size)

$$\text{Block } k = \begin{bmatrix} x_{k,1} & x_{k,2} & \dots & x_{k,W} \\ \vdots & \vdots & \ddots & \vdots \\ x_{k,H} & x_{k,H+1} & \dots & x_{k,H+W} \end{bmatrix}$$

Generalization

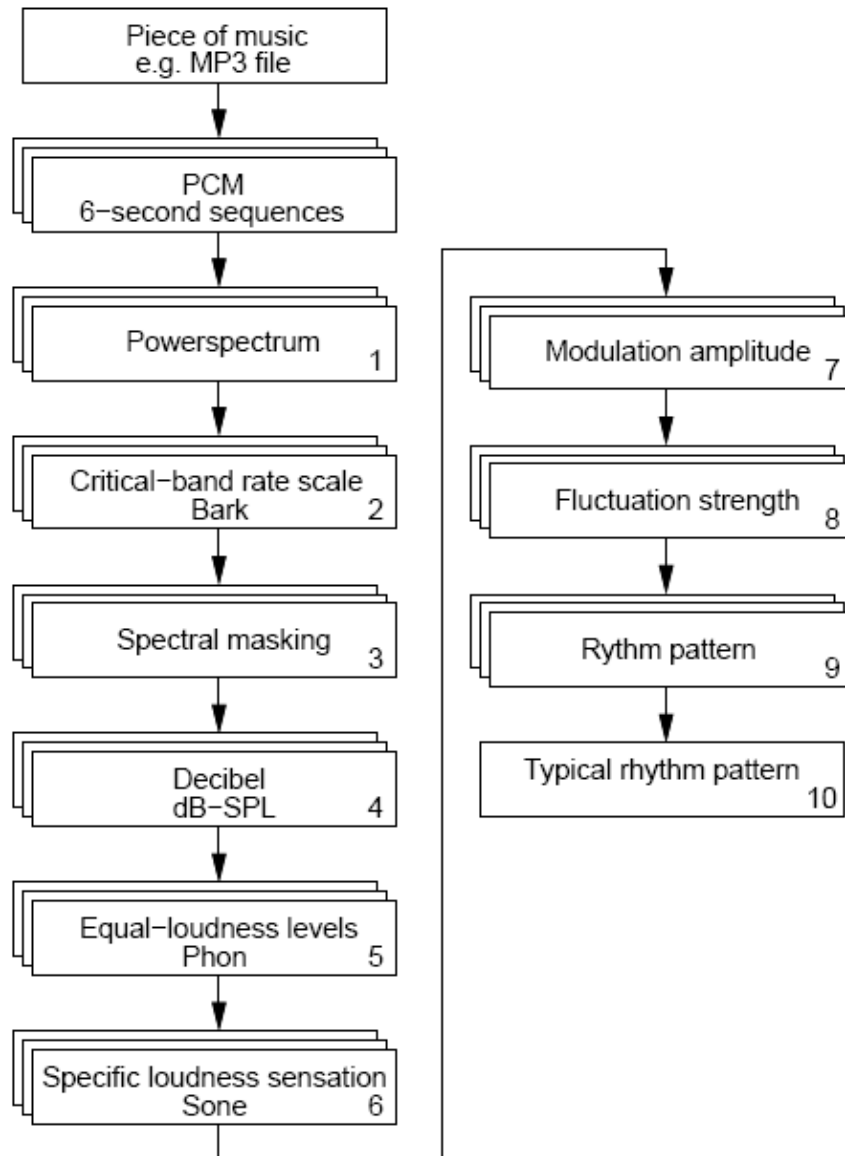
- To come up with a global feature vector per song, the local feature vectors must be combined into a single representation
- This is done by a **summarization function** (e.g., mean, median, certain percentiles, variance, ...)



Fluctuation Patterns (FPs)

- Idea: measure how strong and fast beats are played within certain perceptually adjusted frequency bands
- Aims at capturing periodicities in the signal (“rhythmic properties”)
- Incorporates several psychoacoustic transformations
 - Logarithmic perception of frequencies (Bark scale)
 - Loudness
 - Periodicities
- Results in a vector description for each music piece
 - Vector Space Model
 - Favorable for subsequent processing steps and applications: classification, clustering, etc.

Fluctuation Patterns



Extract 6 sec blocks

- discard beginning and end

In each block:

FFT on Hanning-windowed frames (256 samples)

Convert spectrum to **20 critical bands** according to *Bark scale*

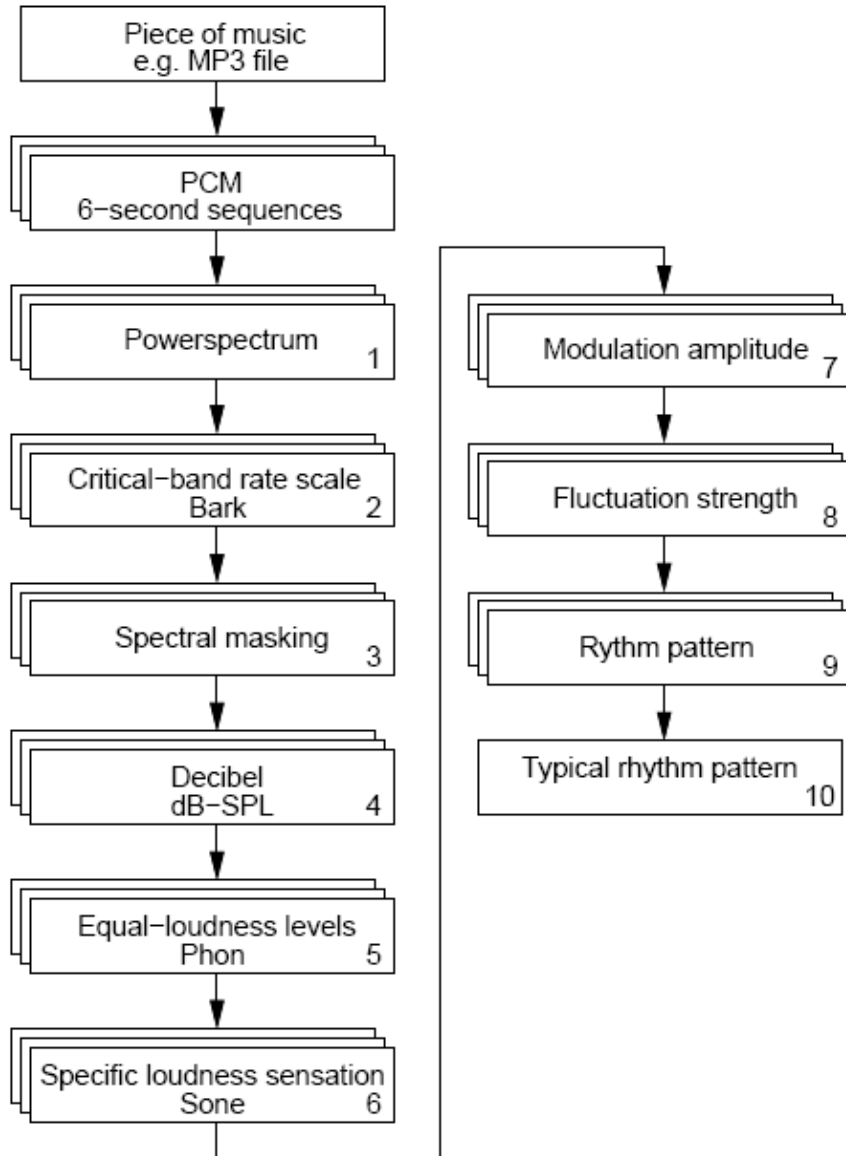
Calculate Spectral Masking effects

- (i.e. occlusion of a quiet sound when a loud sound is played simultaneously)

Several loudness transformations:

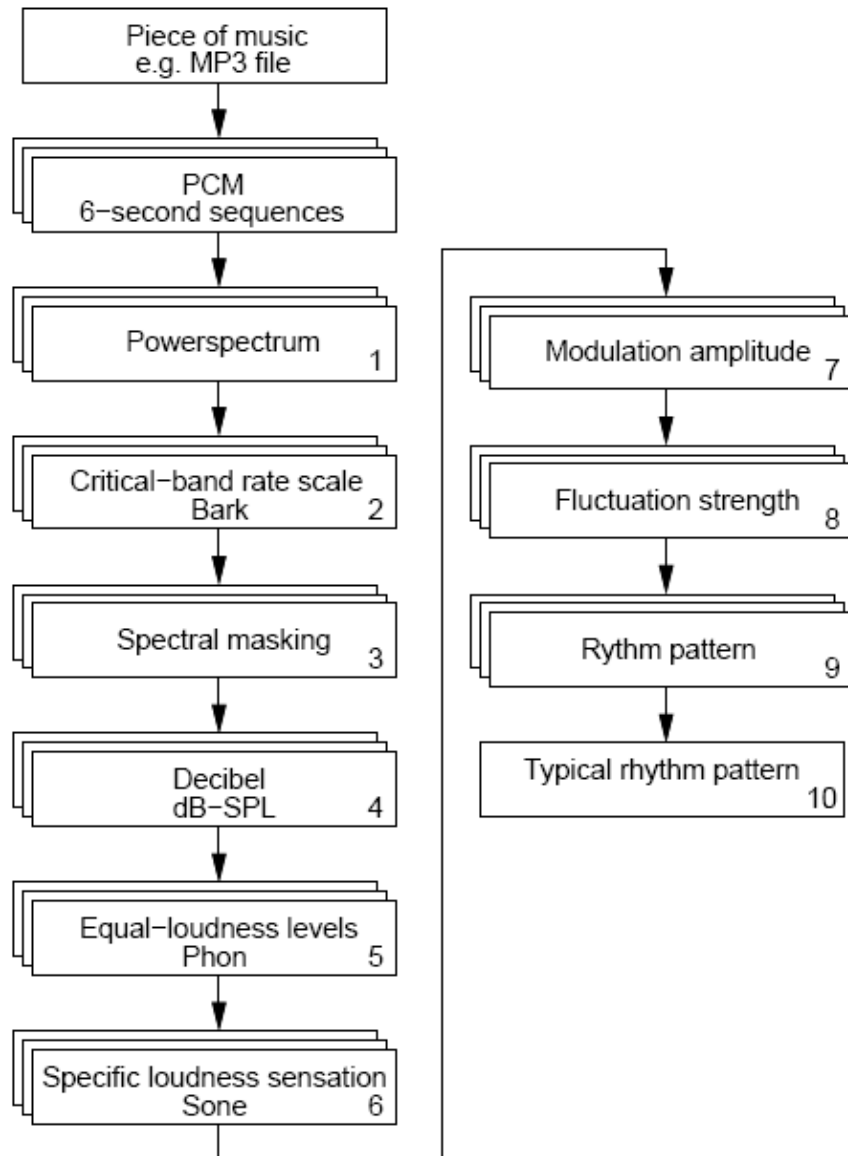
1. to dB (sound intensity)
2. to phon (human sensation: log)
3. to sone (back to linear)

Fluctuation Patterns



- **Second FFT** reveals information about amplitude modulation, called *fluctuations*.
 - Fluctuations show how often frequencies reoccur at certain intervals within the 6-sec-segment
 - “frequencies of the frequencies”
- Psychoacoustic model of fluctuation strength
 - perception of fluctuations depends on their periodicities
 - reoccurring beats at 4Hz perceived most intensely
 - 60 levels of modulation (per band) (ranging from 0 to 600bpm)
- Emphasize distinctive beats

Fluctuation Patterns



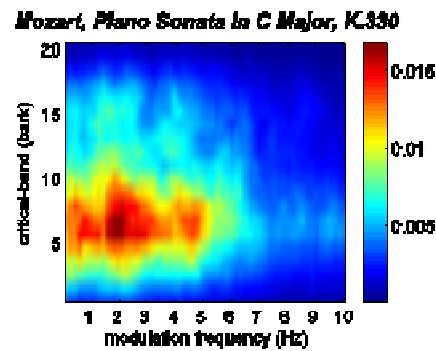
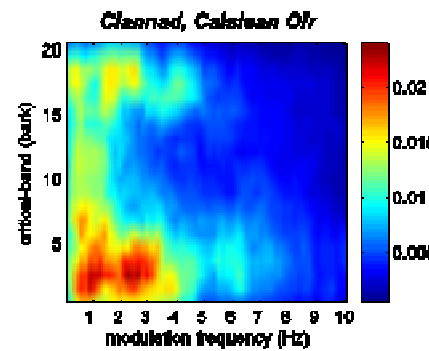
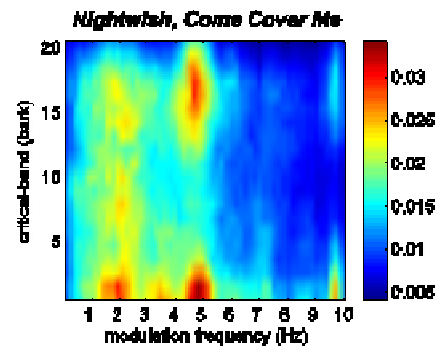
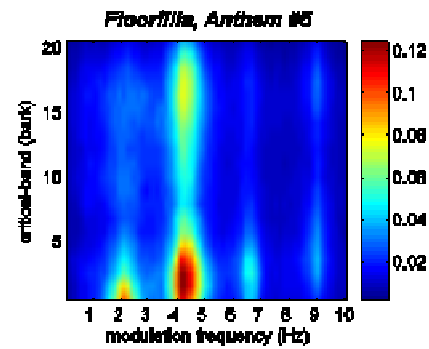
Each block is now respresented as a matrix of fluctuation strengths with 1,200 entries (20 critical bands x 60 levels of modulation)

Aggregation of all blocks by taking *median* of each component

This results in a **1,200 dimensional feature vector** for each music piece

Comparison of two music pieces is done by calculating the *Euclidean distance* between their feature vectors

Examples



Wrapping up FPs and VSM

(Some) temporal dependencies are modeled within segments of 6 seconds length

Properties:

- + Vector Space Model: The whole mathematical toolbox of vector spaces is available.
- + easy to use in classification
- + song models can be visualized
- high dimensional feature space (often a PCA is applied to reduce dim.)

More comprehensive block-level features by (Seyerlehner; 2010)
currently best performing similarity measure according to MIREX:

- Spectral Pattern (SP): frequency content
- Delta-Spectral Pattern (DSP): SP on delta frames
- Variance Delta-Spectral Pattern (VDSP): *variance* used to aggregate DSP
- Logarithmic Fluctuation Pattern (LFP): more tempo invariant
- Correlation Pattern (CP): temporal relation of frequency bands
- Spectral Contrast Pattern (SCP): estimate “tone-ness”
- Block **aggregation** via *percentiles*; **similarity** via *Manhattan distance*