M2 Data Science

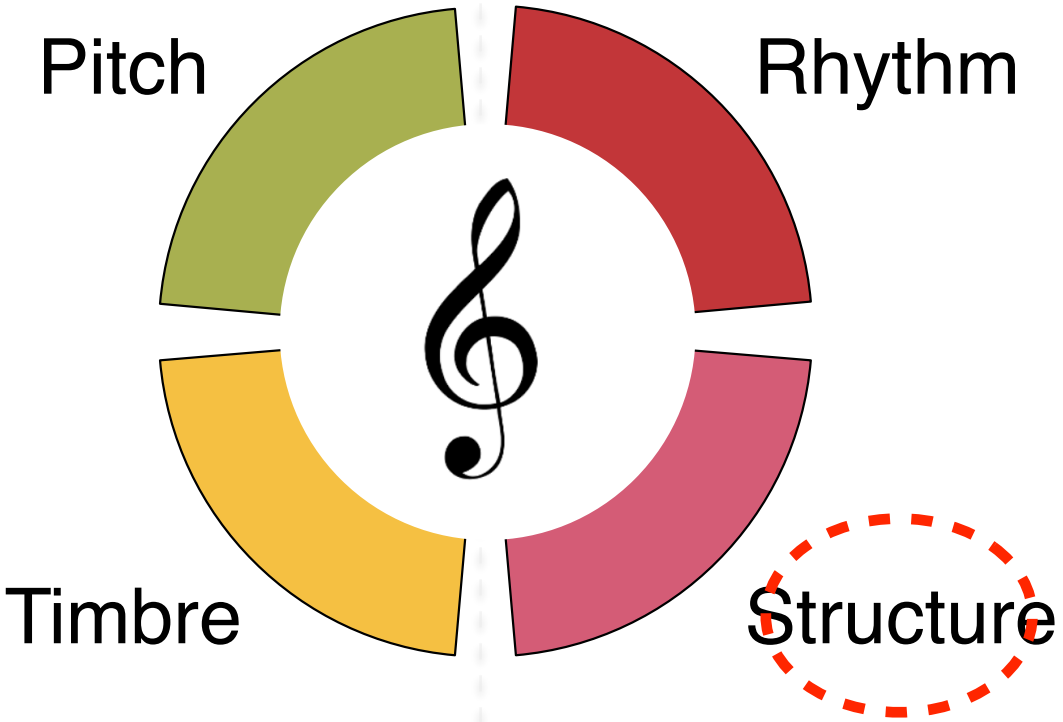
DS-telecom-15 "Audio and Music Information Retrieval"

Geoffroy Peeters

contact: geoffroy.peeters@telecom-paris.fr
Télécom-Paris, IP-Paris, France

Reminder of Musical Concepts

What is structure ?



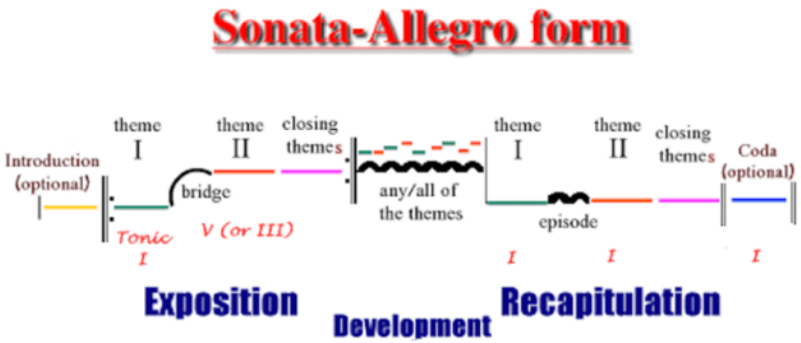
Music structure
Classical music

- **Baroque period**
  - Concerto grosso
    - two groups of musicians alternating
  - Sonata
    - accompaniment= continuo (improvisation, figured bass
  - Suite
    - series of dances
  - Fugue

https://youtu.be/Y_5K8f5CZpg?list=PLZjrBvSPdGwS7Qv1qScnWcQcqiSBlD6cR

https://youtu.be/ZG4SKgCpppE?list=PLZjrBvSPdGwS7Qv1qScnWcQcqiSBlD6cR

- **Classical period**
  - Symphony
    - a sonata for orchestra
    - 4 movements (sequences journey)
      - (1) important, dramatic (sonata form)
      - (2) slow
      - (3) minuet/scherzo (dance form)
      - (4) finale, lively (sonata or rondo form)

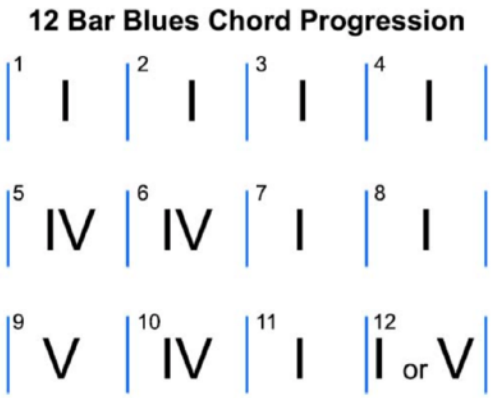


https://www.youtube.com/watch?v=HzHS7QL-B-c&t=548s

  - Concerto
    - a sonata for one or more solo instruments playing along side an orchestra
  - Sonata ≠ sonata form
    - piano solo
    - or solo instruments (violon/flute) + piano
  - Vocal music, opera

Music structure
Popular music

- Blues form



- < 1955: Broadway music AABA (Chorus/Bridge)

Fig. 6: Ira Gershwin – George Gershwin, 'The Man I Love', as recorded by Marion Harris (Victor 21116-B, 1927)

- > 1955: Beatles (Verse/ Chorus)

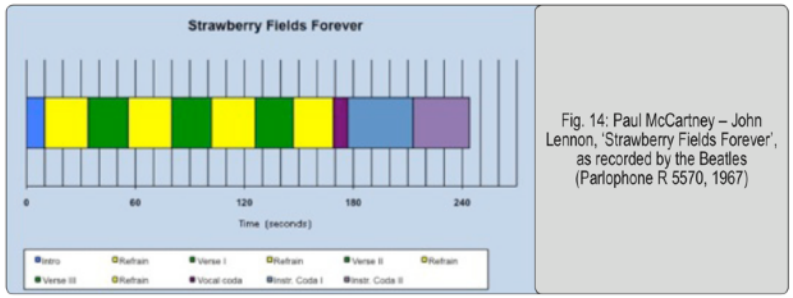
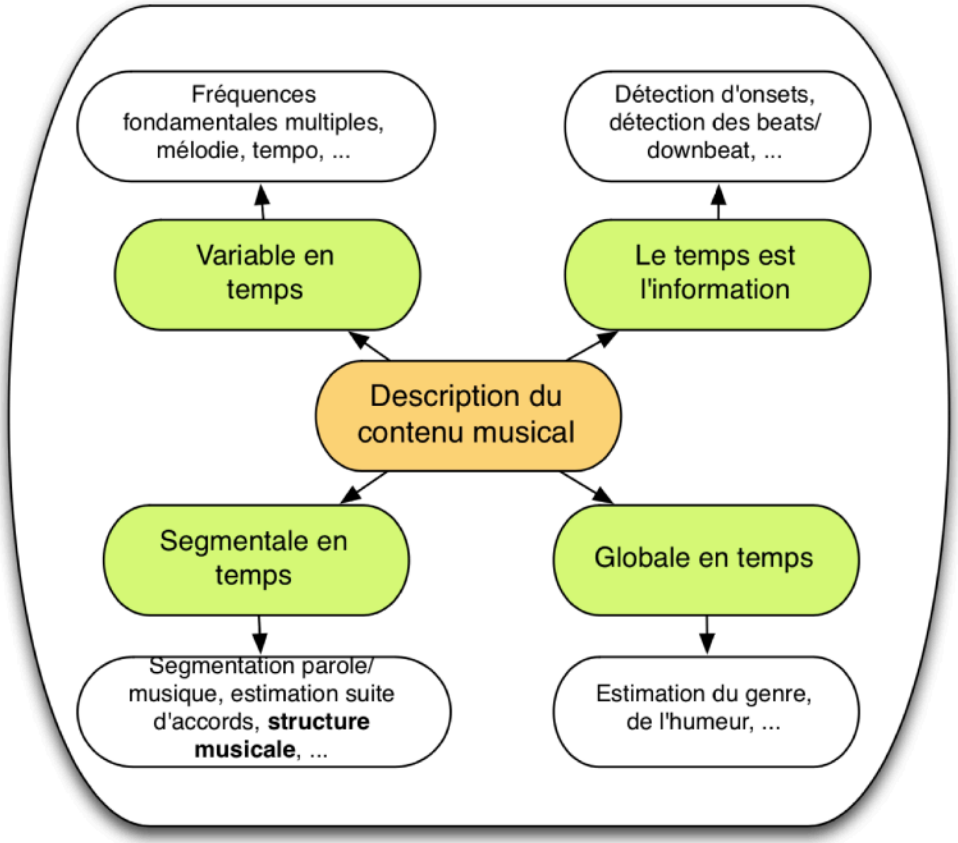


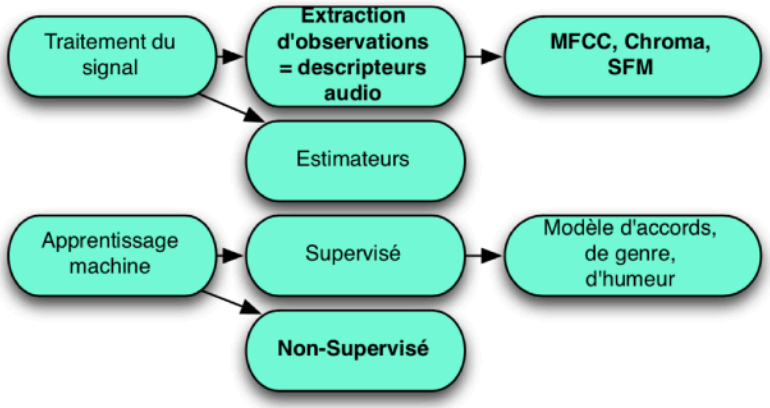
Fig. 14: Paul McCartney – John Lennon, 'Strawberry Fields Forever', as recorded by the Beatles (Parlophone R 5570, 1967)

Music Structure Discovery (MSD) - Audio Summary

Music Structure Discovery (MSD) - Audio Summary

Different type of description of musical content





Music Structure Discovery (MSD) - Audio Summary

Estimation of *the* Musical Structure → of *a* Musical Structure

– **Goal**
  - Estimate a structure of a music track
  - Automatically generate an **audio summary** which is representative of the content of the music track

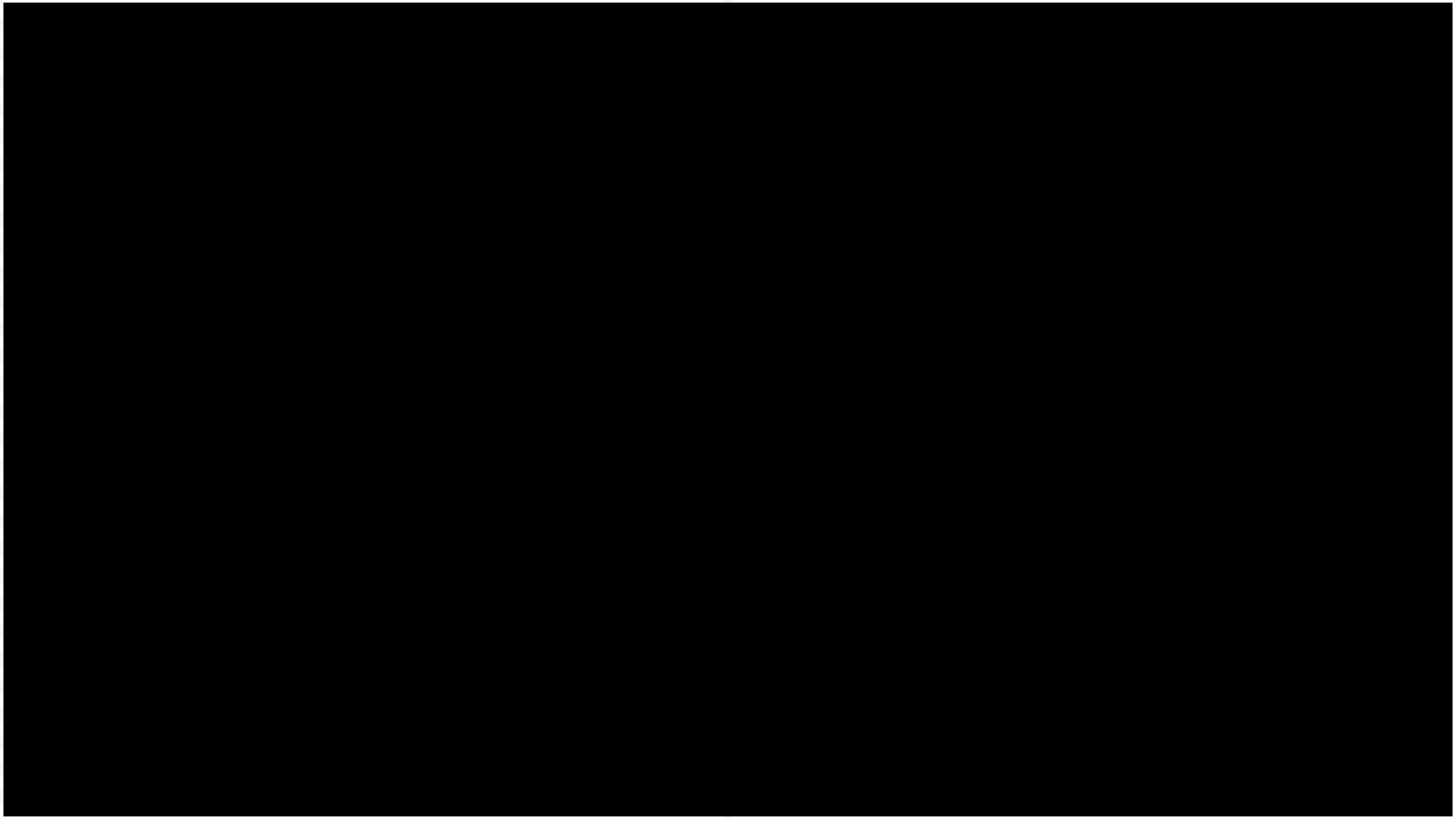
– **Applications**
  - Interactive listing:
    – interactive music player
  - Fast preview of music track
  - **Audio and video examples**

– **Systems**
  - Audio features extraction
  - Structure visualisation
  - Structure estimation
    – Traditional approaches: **unsupervised** learning
    – Recent approaches: **supervised** learning

source : Quaero project, MSSE-Orange interface

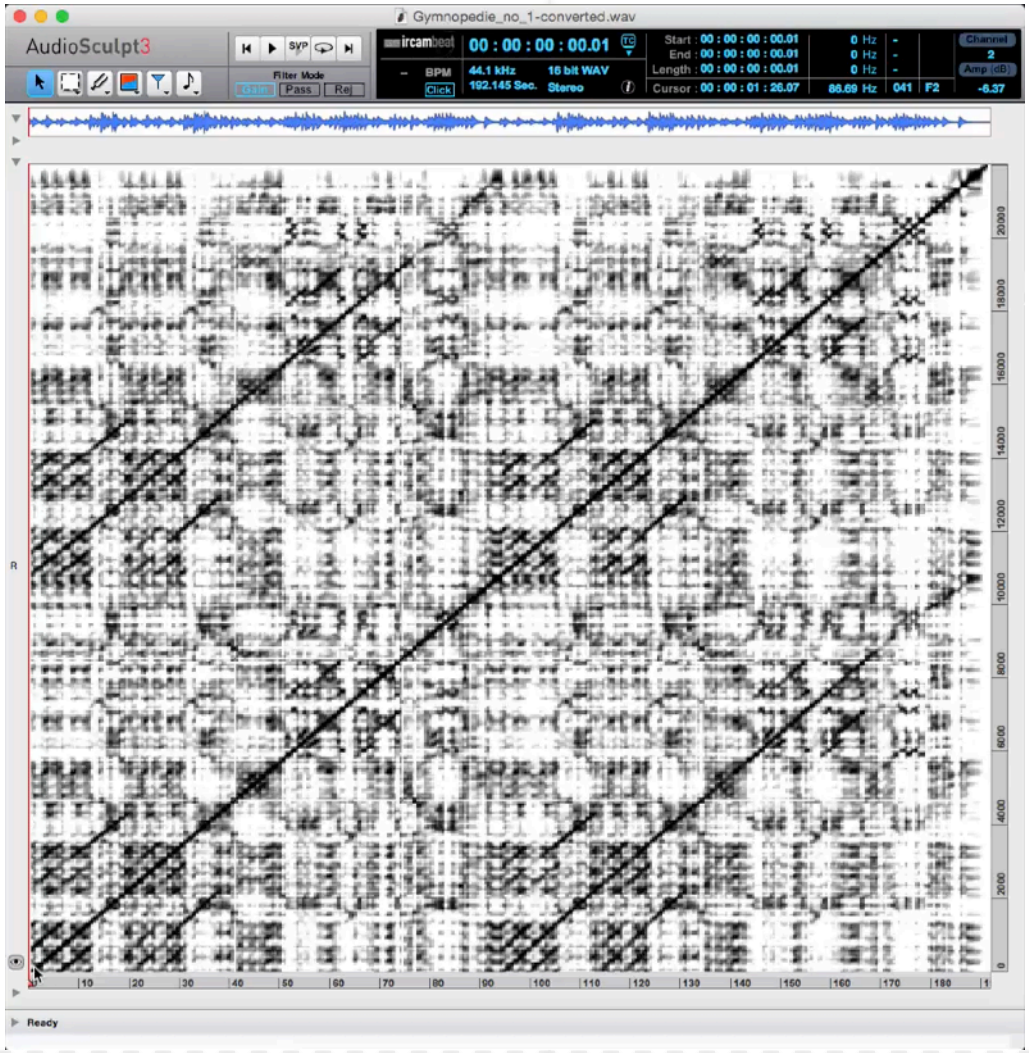
Music Structure Discovery (MSD) - Audio Summary

Example: Self-Similarity-Matrix



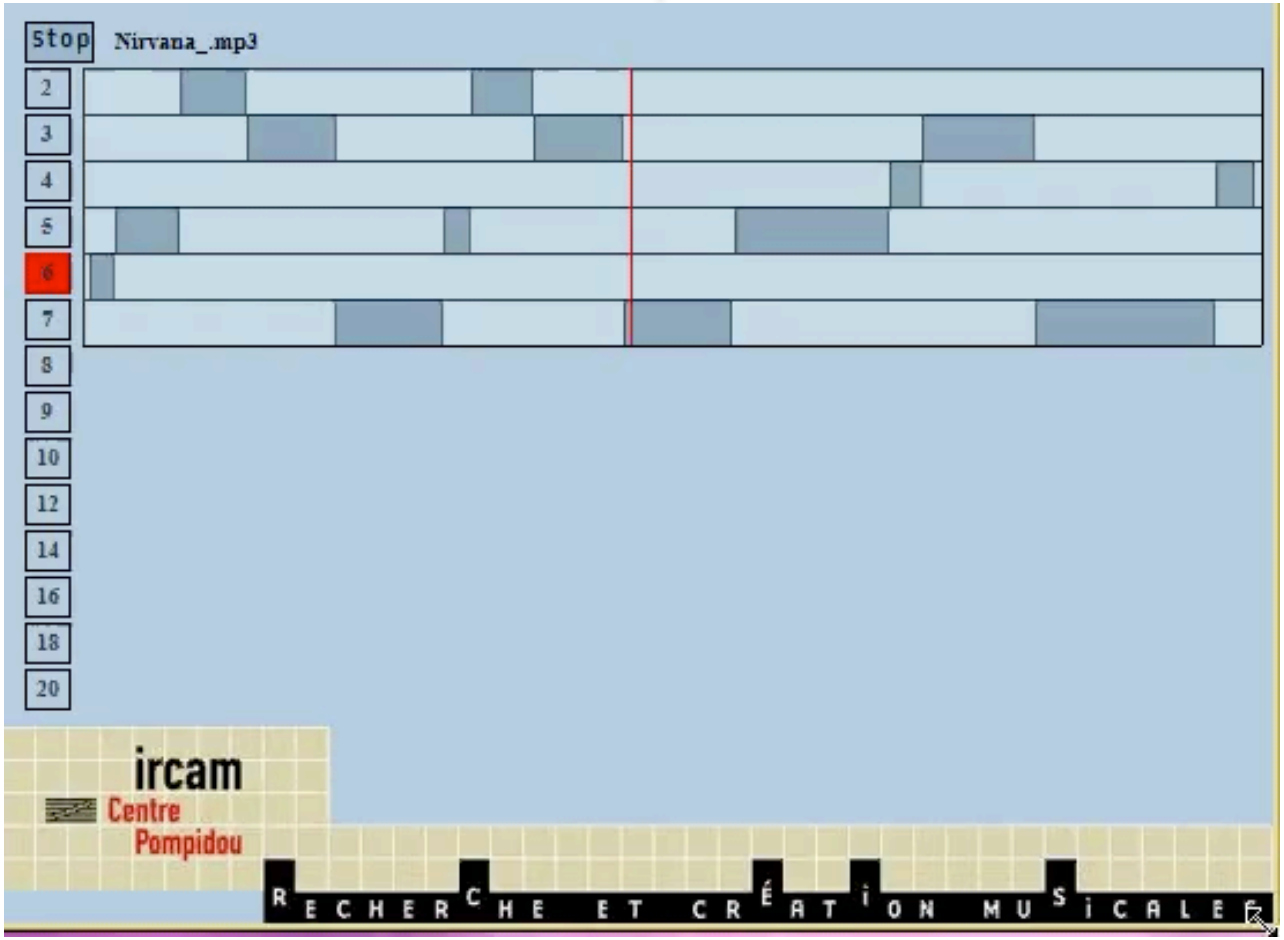
Music Structure Discovery (MSD) - Audio Summary

Example: Self-Similarity-Matrix



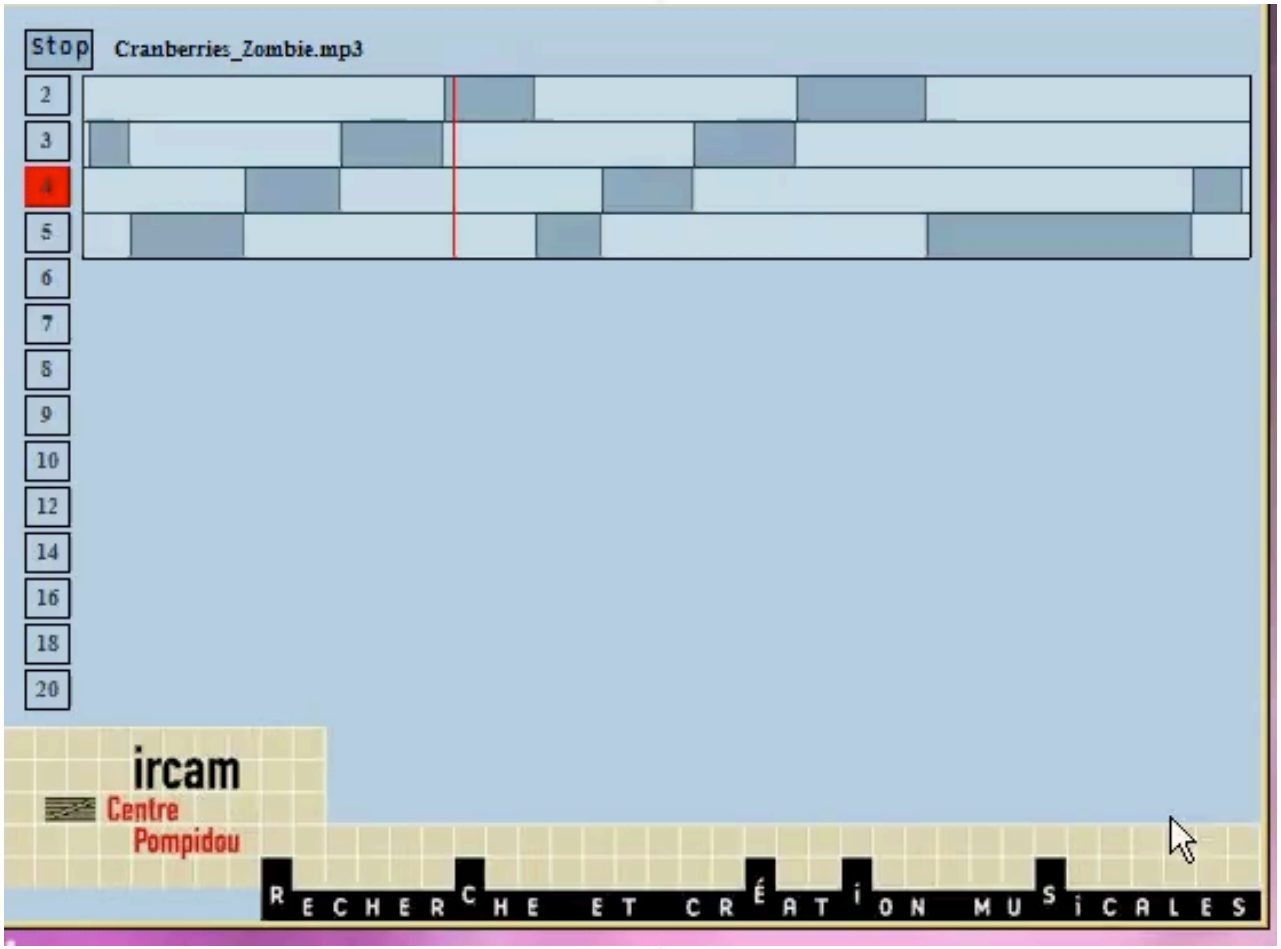
Music Structure Discovery (MSD) - Audio Summary

Example: Navigating into the structure



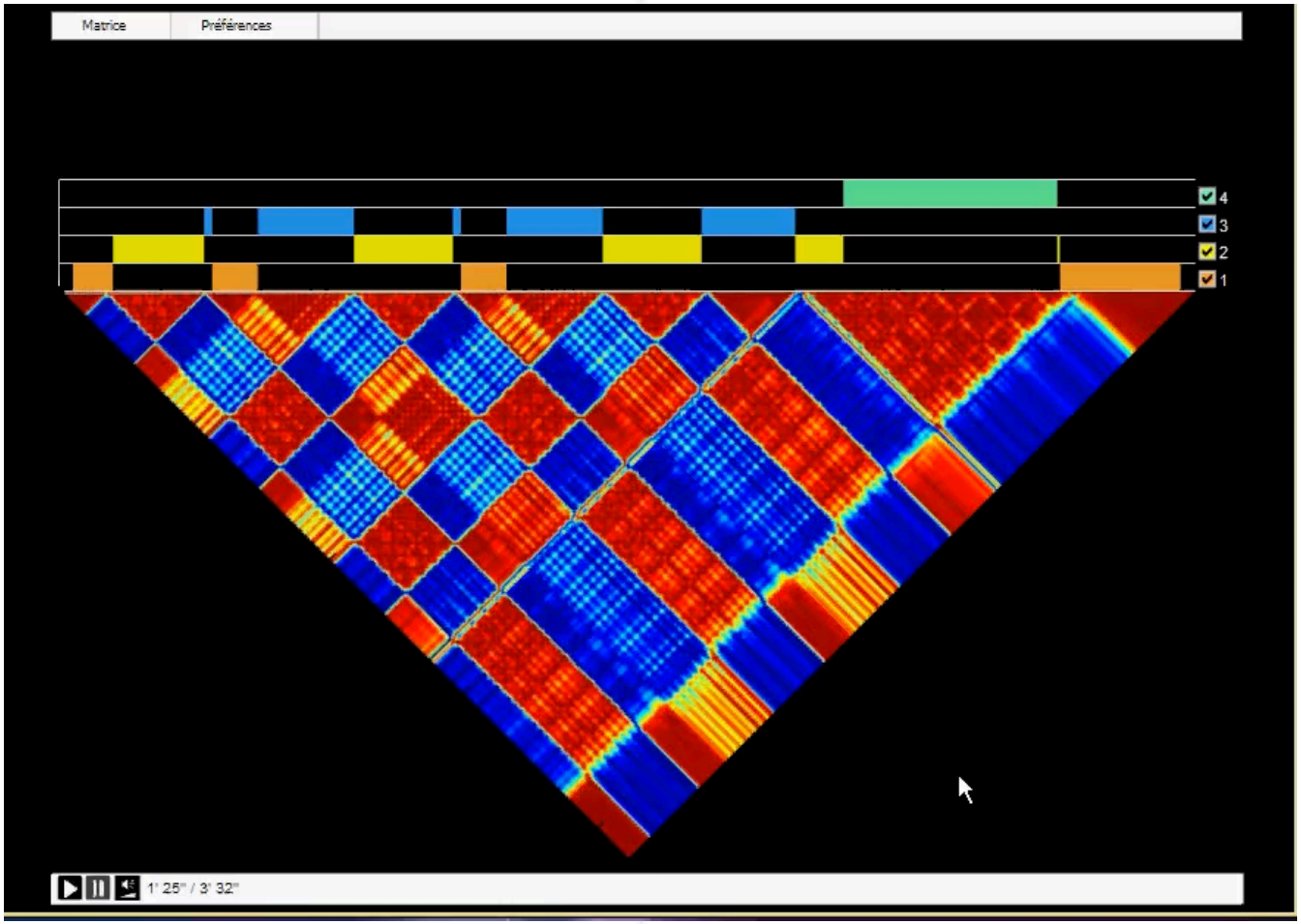
Music Structure Discovery (MSD) - Audio Summary

Example: Navigating into the structure



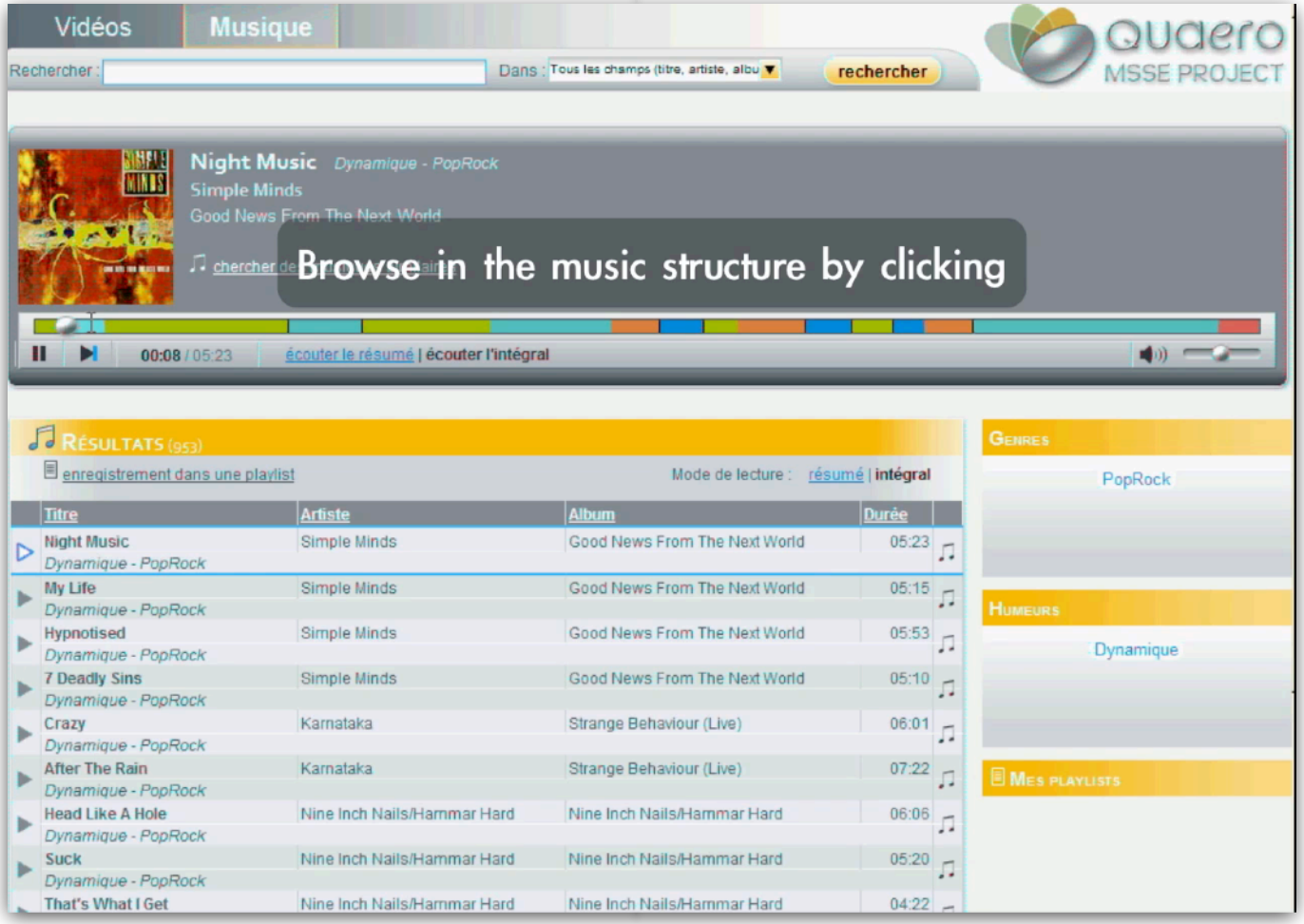
Music Structure Discovery (MSD) - Audio Summary

Example: Navigating into the structure



Music Structure Discovery (MSD) - Audio Summary

Example: Navigating into the structure



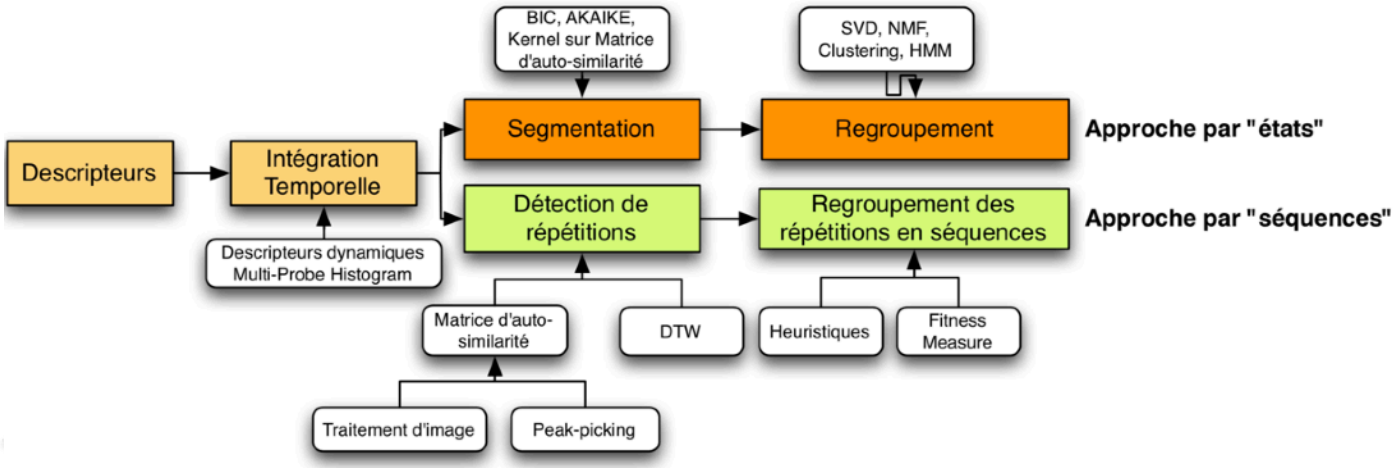
Music Structure Discovery (MSD) - Audio Summary

Example: Audio Summary

Systems for Music Structure Estimation

- 1) Extraction of meaningful observation from the audio signal
  - **Audio features**: allows to highlight different facets of the content (timbre, harmonicity, noise, ...)

- 2) Analyze the observations to estimate a structure
  - **State** approach
    - **segmenting** temporal stream of observation
    - **grouping** repeated homogeneous segments
  - **Sequence** approach
    - **detecting** non-homogeneous **repetitions**
    - **grouping** repeated segments into sequences



Music Structure Discovery (MSD) - Audio Summary Systems

Music Structure Discovery (MSD) - Audio Summary Systems

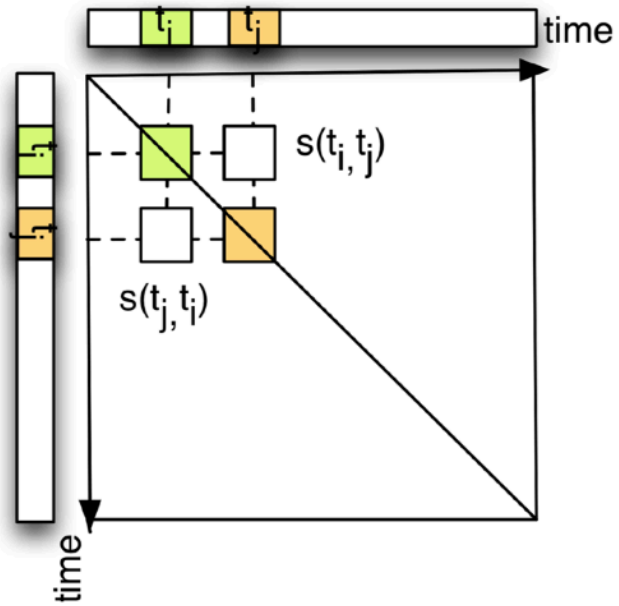
Brief overview of systems evolution

- *as usual, the **first systems** define the task, the performance measures, and provide a first test-set; **later systems** deals with scalability issues and create large test-set; **current systems** use this large dataset to train systems using deep-learning*

- (1) Self-Similarity-Matrix
  - **1999** → J. Foote. Visualizing music and audio using self-similarity. In ACM Multimedia, 1999
- (2) Kernel-based segmentation
  - **2000** → J. Foote. Automatic audio segmentation using a measure of audio novelty. In IEEE ICME, 2000
- (3) SSM-based audio summary generation
  - **2002** → M. Cooper and J. Foote. Automatic music summarization via similarity analysis. In ISMIR, 2002.
- (4) Structure-based audio summary generation
  - **2002** → G. Peeters et al. Toward automatic music audio summary generation. In ISMIR, 2002
- (5) DTW approach
  - **2013** → M. Mueller et al. A robust fitness measure for capturing repetitions, In IEEE TASLP
  - **2014** →V. Bisot and G. Peeters Estimation de la structure musicale par DTW locale, In M2 ATIAM
- (6) Deep learning approaches
  - **2014** → K. Ullrich et al. Boundary detection ... using convolutional neural networks. In ISMIR, 2014
  - **2017** → A. Cohen-Hadria and G. Peeters. Music structure boundaries In AES Semantic Audio, 2017
- (7) Metric Learning/ Self-Supervised Learning
  - **2019** →M. McCallum. Unsupervised learning of deep features for music segmentation. In ICASSP, 2019

Self-Similarity-Matrix (time, time)

- Visual representation of the temporal structure of a music track
- Indicates the similarity between two times t_i et t_j
- Similarity is computed by using the observations extracted from the signal around time frames i and j: $\mathbf{d}^{<i>}$ and $\mathbf{d}^{<j>}$
  - $s(t_i, t_j) = s(\mathbf{d}^{<i>}, \mathbf{d}^{<j>})$
- Self-Similarity-Matrix= values $s(t_i, t_j)$ represented as a matrix
  - $\mathbf{S}_{ij} = s(t_i, t_j) \quad \forall i, j$



- **How to read ? Interpretation ?**
  - High value in $\mathbf{S}_{ij}$ = high similarity between times t_i and t_j
  - If $t_i \simeq t_{i+1} \simeq t_{i+2}$, we observe an **homogeneous block** in $\mathbf{S}$
  - If the **sequence of times** $\{t_i, t_{i+1}, t_{i+2}, \ldots\}$ is similar to the sequence of times $\{t_j, t_{j+1}, t_{j+2}, \ldots\}$, we observe a lower/upper diagonal (symmetry) in $\mathbf{S}$

[J. Foote. Visualizing music and audio using self-similarity. In ACM Multimedia, 1999]

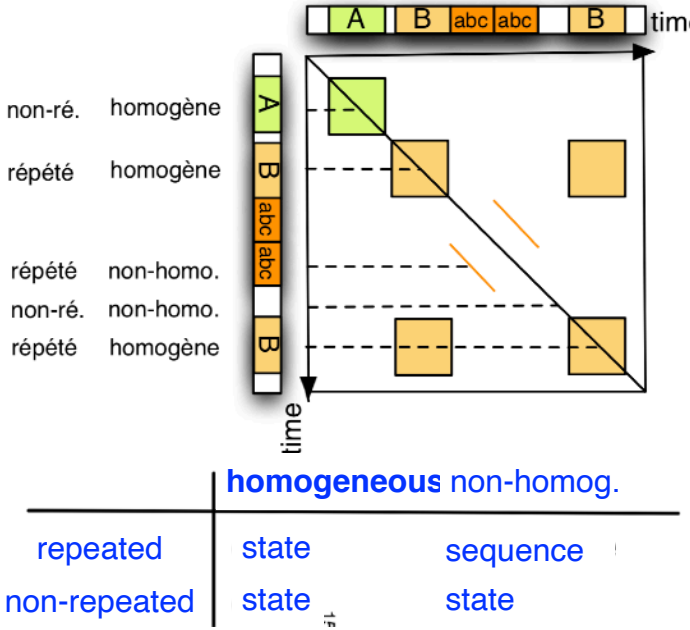
Homogeneity

– **Assumption:**
  - the music track is made of a succession of **homogeneous** $t_i \simeq t_{i+1} \simeq t_{i+2}, \ldots$ and non-homogeneous time segments

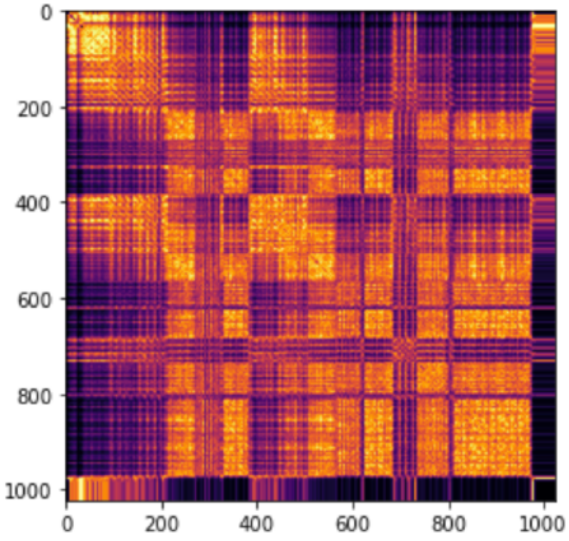
– **Homogeneous ?**
  - which contains a similar information according to the observation criteria
  - "A"' and "B"' in the Figure

– Example:
  - music accompaniment during a verse or a chorus

– Method:
  - **"state" approach**



	homogeneous	non-homog.
repeated	state	sequence
non-repeated	state	state



Representations, interpretations

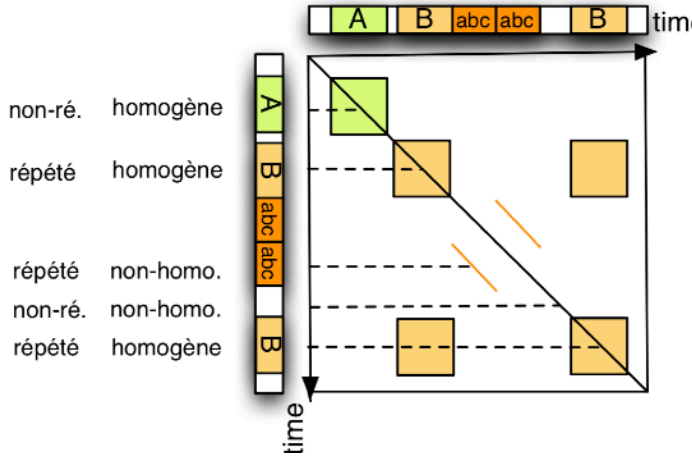
Repetitions

– **Assumption:**

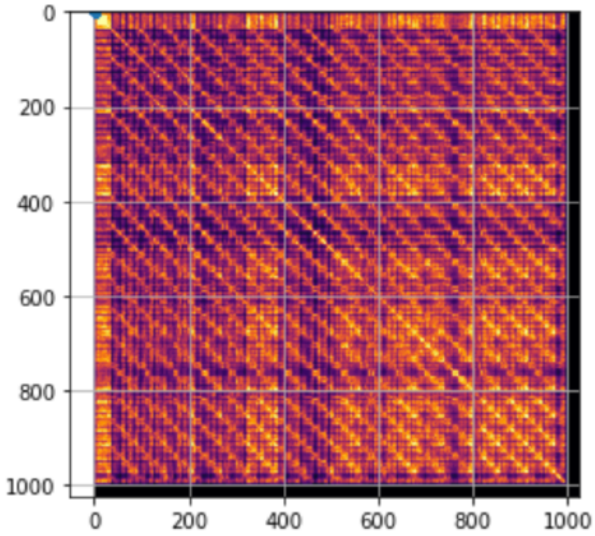
- the music track contains le morceau renferme des **répétitions** temporelles repetitions

– **Repetition ?**

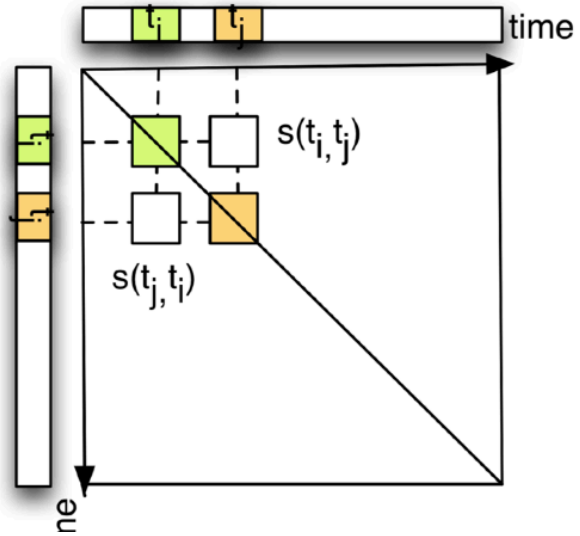
- can correspond to the repetition of **homogeneous** segments
  - $\{t_j, t_{j+1}, t_{j+2}\} \simeq \{t_i, t_{i+1}, t_{i+2}\}$ and $t_i \simeq t_{i+1} \simeq t_{i+2}$
  - "B" in the Figure
  - Method: **"state" approach**
- can correspond to the repetition of **non-homogeneous** segments
  - $\{t_j, t_{j+1}, t_{j+2}\} \simeq \{t_i, t_{i+1}, t_{i+2}\}$ but $t_i \neq t_{i+1} \neq t_{i+2}$
  - sequence "abc" in the Figure
  - Method: **"sequence" approach**



	homogeneous	non-homog.
repeated	state	sequence
non-repeated	state	state



Self-Similarity-Matrix (time, time)



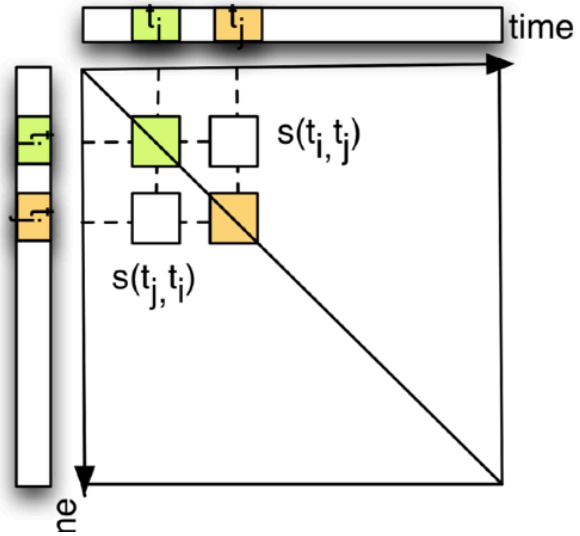
- Similarity between time t_i and t_j
- Similarity between the signal observations at time frame i and j:
  - $\mathbf{S}_{ij} = s(\mathbf{d}^{<i>}, \mathbf{d}^{<j>})$
- Audio features (multi-dimensional)
  - $\mathbf{d} = \{d_k\}\ k \in K$

– Choice of a distance

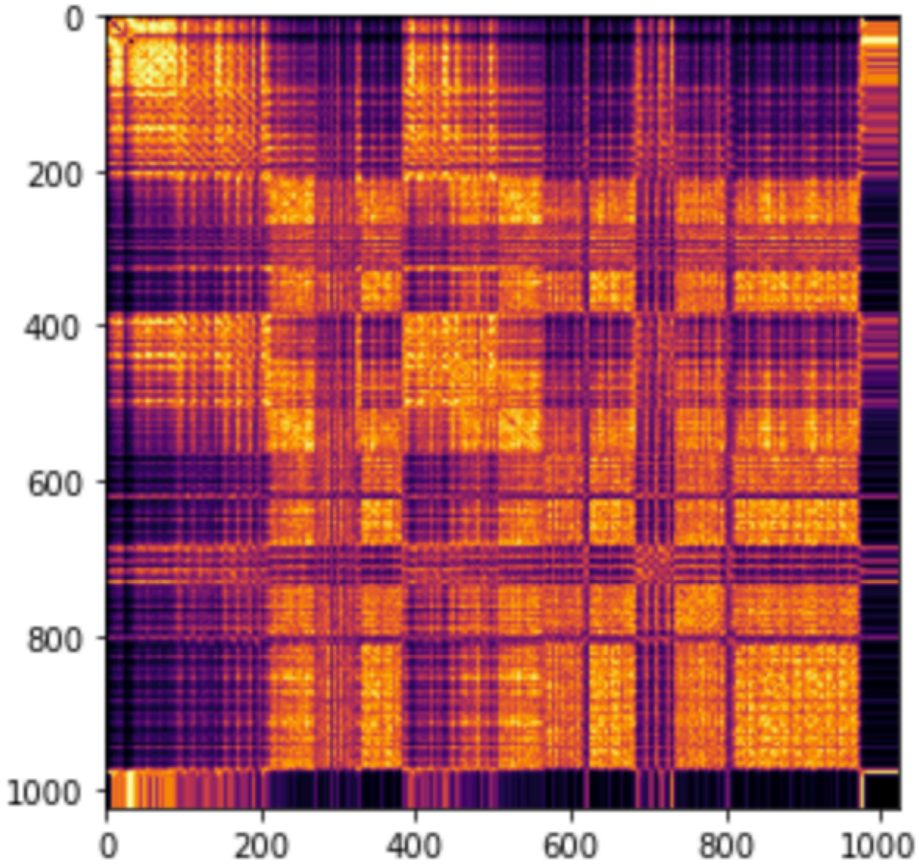
- Euclidean distance: $\sqrt{\sum_k (d_k^{<i>} - d_k^{<j>})^2}$
- Correlation: $\sum_k (d_k^{<i>} \cdot d_k^{<j>})$
- Cosine distance: $\frac{\sum_k (d_k^{<i>} \cdot d_k^{<j>})}{\sqrt{\sum_k (d_k^{<i>})^2}\sqrt{\sum_k (d_k^{<j>})^2}}$
- Pearson correlation: $\frac{\sum_k (d_k^i - \mu^i) \cdot (d_k^j - \mu^j)}{\sqrt{\sum_k (d_k^i - \mu^i)^2}\sqrt{\sum_k (d_k^j - \mu^j)^2}}$

Self-Similarity-Matrix (time, time)

- What is $\mathbf{d}$?
- Audio features, hand-crafted/ hand-designed audio features
  - Chroma/ PCP; can be compute easily and more precisely from the CQT
  - MFCCs
  - and many others



Segmentation based on homogeneity → Blocks



Segmentation based on homogeneity → Blocks

Temporal segmentation: traditional approaches

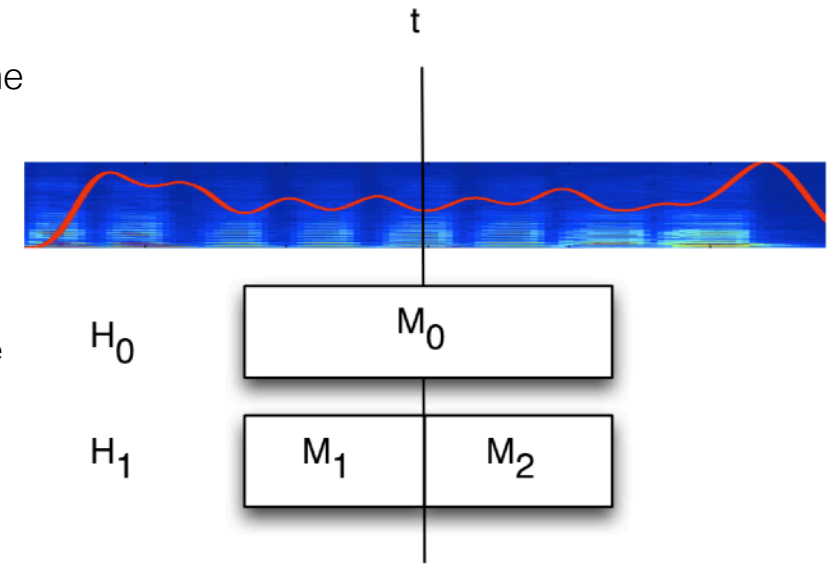
– **Frame-by-frame segmentation**
  - We measure the variation of $\mathbf{d}^{<t>}$ frame-by-frame

– **BIC (Bayes Information Criteria) criteria**
  - At each time t (a candidate for a potential rupture), we compare two hypothesis
    – H0: the signal follows the same distribution before and after t, denoted by $M_0(\mu_0, \Sigma_0)$
    – H1: there is a change at t, and the signal follows different models before and after t, denoted by $M_1(\mu_1, \Sigma_1)$ et $M_2(\mu_2, \Sigma_2)$
  - Delta BIC criteria

$$\Delta BIC = R(t) - \lambda P$$

$$R(t) = \frac{1}{2}(N \log(|\Sigma_0|) - t \log(|\Sigma_1|) - (N-t)\log(|\Sigma_2|)$$

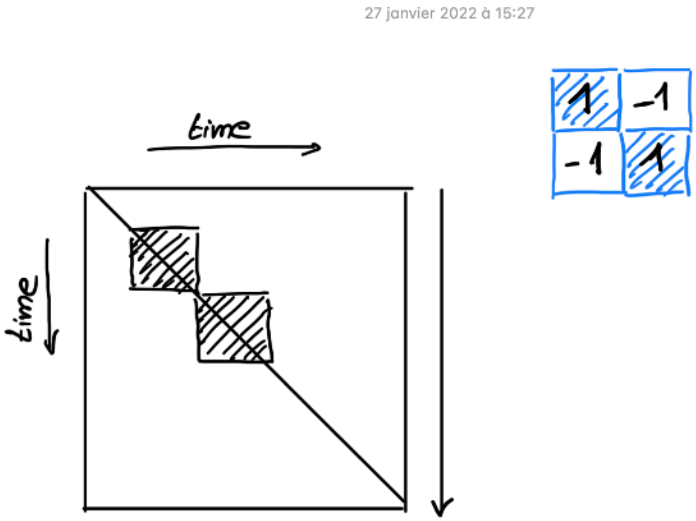
    – if $\Delta BIC > 0$, H1 is checked
  - Parameters:
    – P: proportional to the difference of the number of parameters for each hypothesis
    – λ: penalty factor chosen such that $\Delta BIC > 0$ if H1 is true



Segmentation based on homogeneity → Blocks

Temporal segmentation: Kernel-based approach

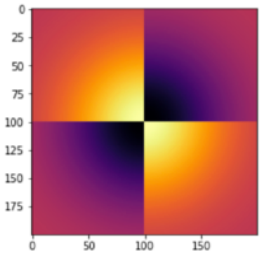
- Take into a count the specific structure of a Self-Similarity-Matrix $\mathbf{S}$
  - intra-segment similarity (homogeneity) of the content
  - dis-similarity between the content of the left and right segments
  - leads to a more robust segmentation
- How ?
  - Convolve the Self-Similarity-Matrix $\mathbf{S}$ with a check-board ("damier") kernel

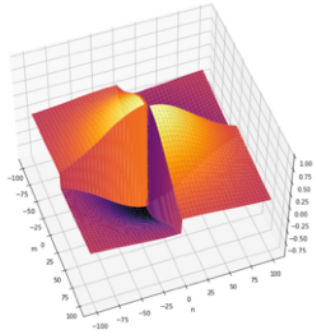


[J. Foote. Automatic audio segmentation using a measure of audio novelty. In Proc. of IEEE ICME, 2000.]

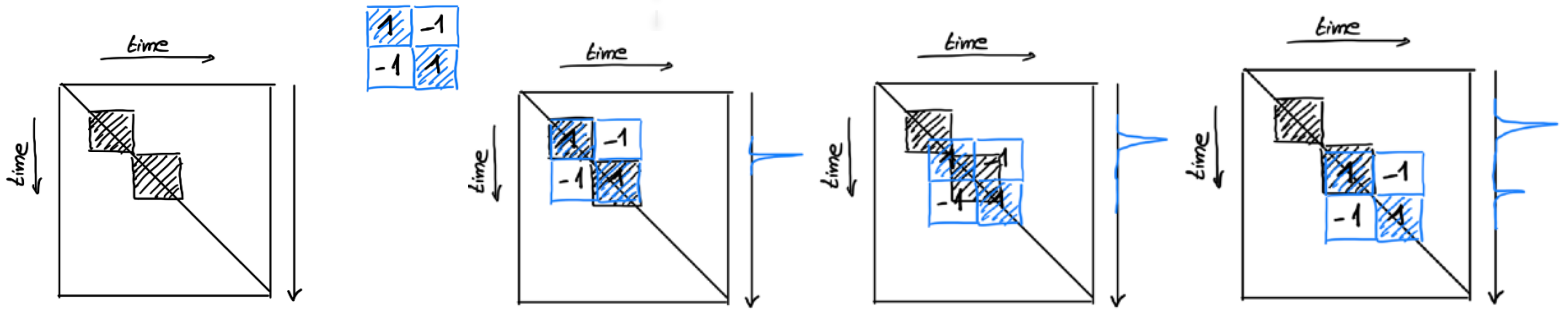
Segmentation based on homogeneity → Blocks

- **Temporal segmentation: Kernel-based approach**





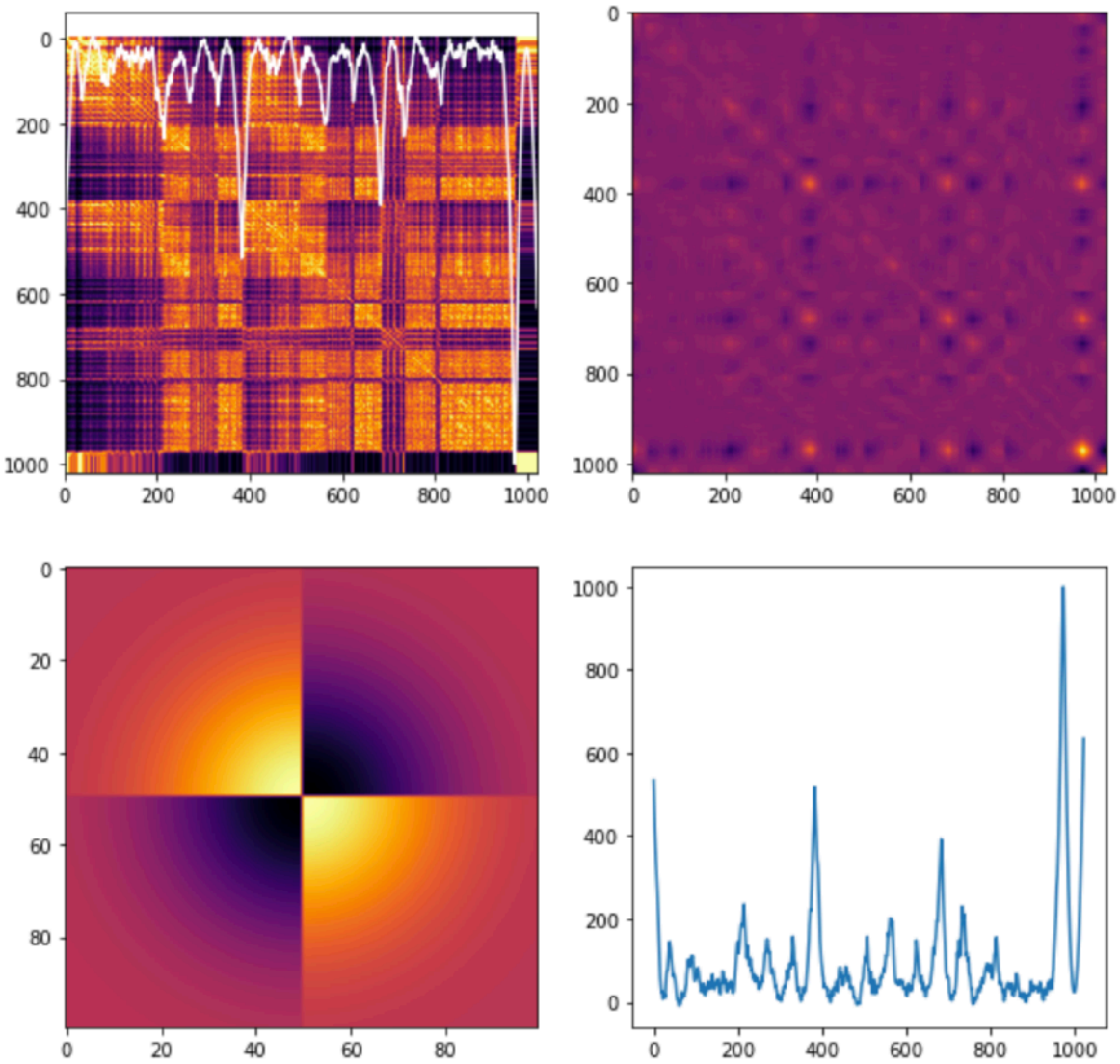
  - **"Checker-board" kernel ?**
    - $\mathbf{C} = \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$
  - Smooth the borders of the kernel by convolving it with a Gaussian kernel
    - $C(m,n) = sign(m) \cdot sign(n) \cdot e^{-\frac{m^2+n^2}{2\sigma^2}}$
  - Value over the main diagonal (i,i) of the 2D convolution
    - $N(i) = \sum_{m=-L/2}^{L/2} \sum_{n=-L/2}^{L/2} C(m,n) S(i+m, i+n)$
  - The value at (t,t) on the main diagonal of the filtered matrix represent the similarity/ dis-similarity between the left segment $[t-\Delta, t]$ and right segment $[t, t+\Delta]$



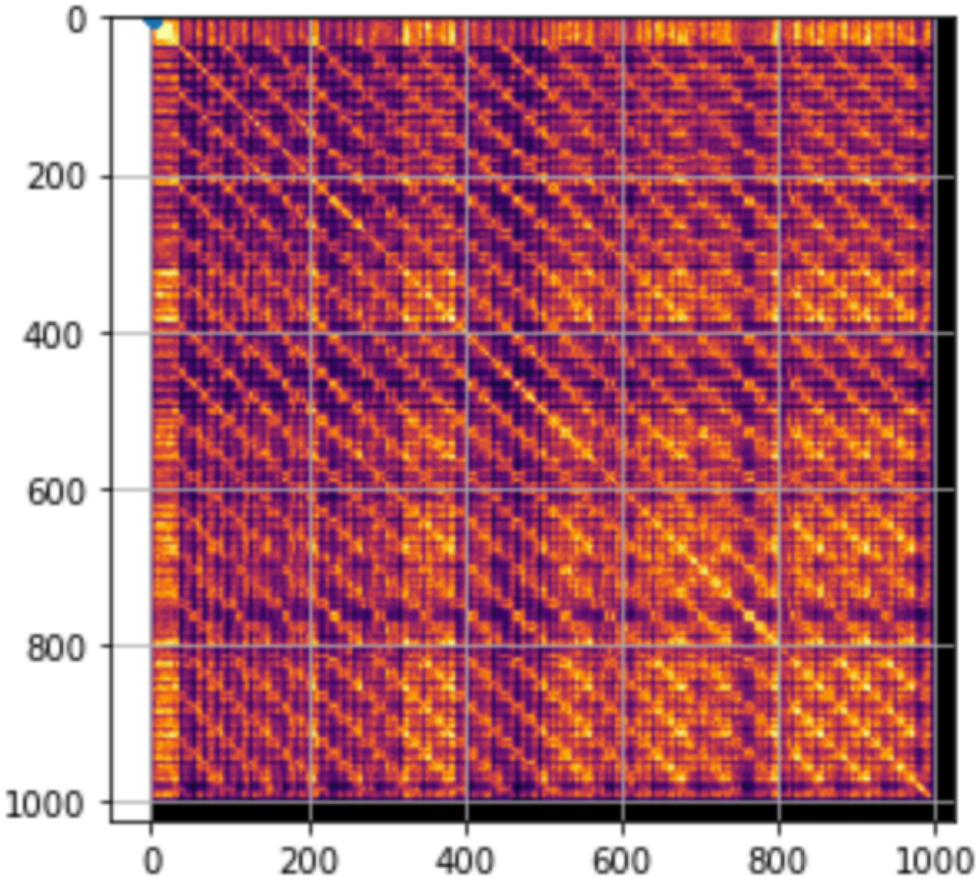
[J. Foote. Automatic audio segmentation using a measure of audio novelty. In Proc. of IEEE ICME, 2000.]

Segmentation based on homogeneity → Blocks

Temporal segmentation: Kernel-based approach



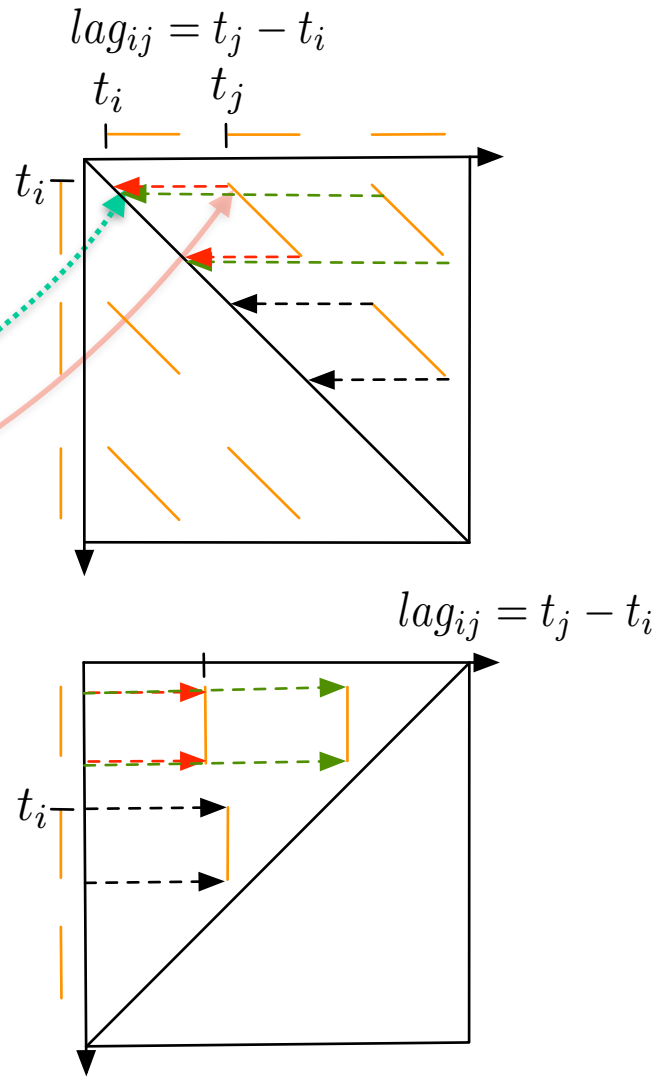
Segmentation based on repetition → sub-diagonals



Segmentation based on repetition → sub-diagonals

Self-Similarity-Matrix (time, lag)

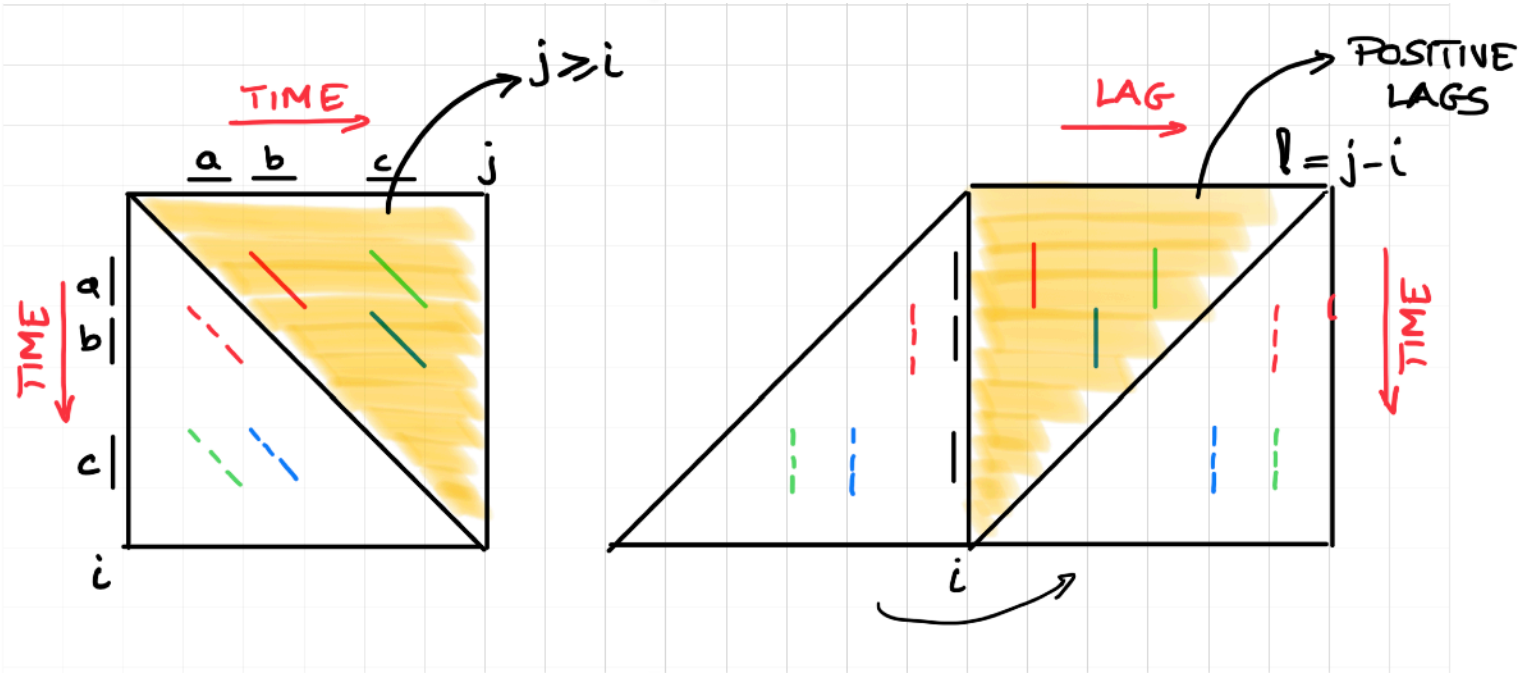
- A high value $\mathbf{S}_{ij}$ represents a large similarity between time t_i and t_j
- If a sequence of time $\{t_i, t_{i+1}, t_{i+2}, \ldots\}$ is similar to a sequence of time $\{t_j, t_{j+1}, t_{j+2}, \ldots\}$, we observe a lower/ upper diagonal (symmetry) in $\mathbf{S}$
- **Lag** =
  - distance between the repetition (which starts at t_j) and the original sequence (which starts at t_i)
  - it is given by the distance between t_j and its projection on the main diagonal which is t_i:
    - $lag_{ij} = t_j - t_i$
  - it has a constant value for a repetition without tempo modification
- **Lag matrix:**
  - $\mathbf{L}_{i,l} = \mathbf{S}(t_i, t_j = t_i + l) \quad l \in [0, T - t_i]$
  - diagonals in a SSM (time, time) →vertical lines in a SSM (time, lag)



Segmentation based on repetition → sub-diagonals

Self-Similarity-Matrix (time, lag)

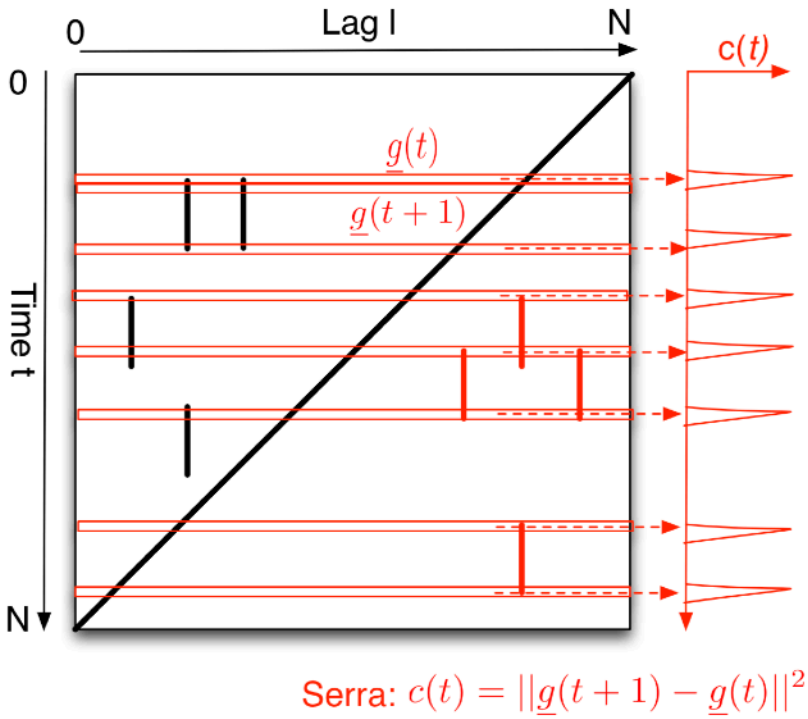
- Lag matrix is only defined for **positive lags** $l \geq 0$
- Generate a FULL Lag-Matrix by **mirroring the negative axis**



Segmentation based on repetition → sub-diagonals

Temporal segmentation: "Structural features" approach

- Compute the Self-Similarity-Matrix (time, lag)
- We consider each row (all the lags l for a given time t as a "structural feature" $\underline{g}^t$
- We compute the frame-to-frame difference of $\underline{g}^t$: $||\underline{g}^{t+1} - \underline{g}^t||^2$

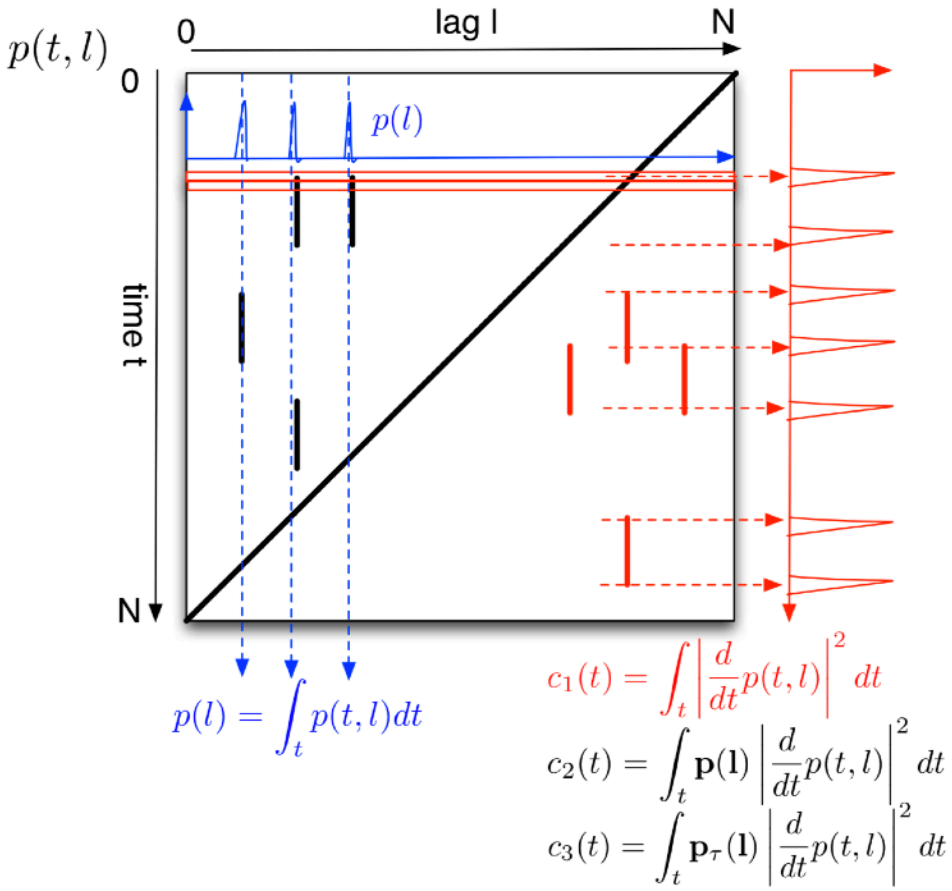


[J. Serra et al. Unsupervised detection of music boundaries by time series structure features. In Proc. of AAAI, 2012.]

Segmentation based on repetition → sub-diagonals

Temporal segmentation: "Structural features" approach with lag-priors

- We weight the "structural feature" with a prior probability $p(l)$ of observing a repetition at this given lag l
- The prior $p(l)$ is obtained using Goto-2003 method
- We finally compute the frame-to-frame difference of $\underline{g}^t$: $||\underline{g}^{t+1} - \underline{g}^t||^2$

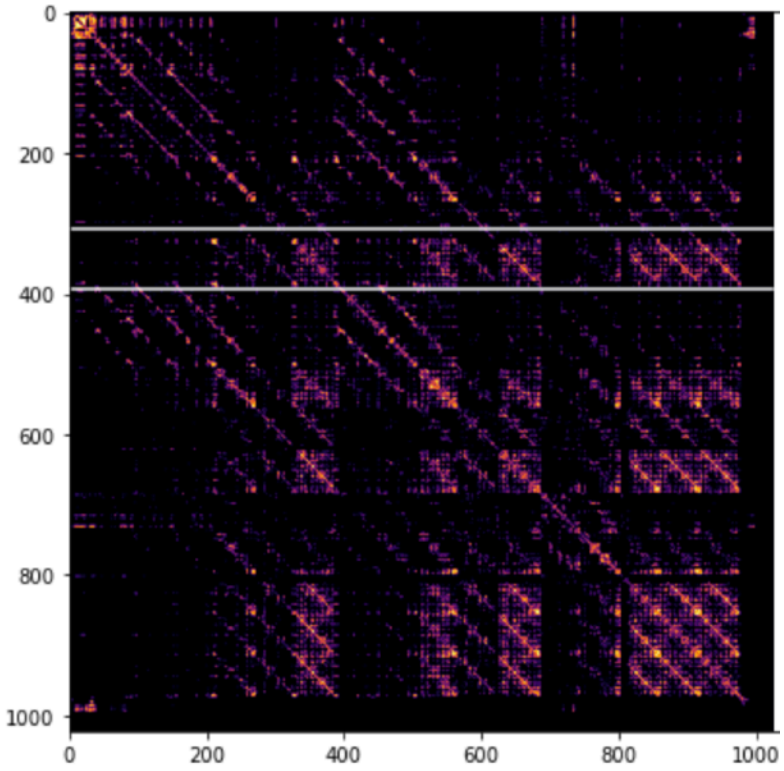


[G. Peeters and V. Bisot. Improving music structure segmentation using lag-priors. In ISMIR, 2014.]

Simple summary generation

SSM-based audio summary generation

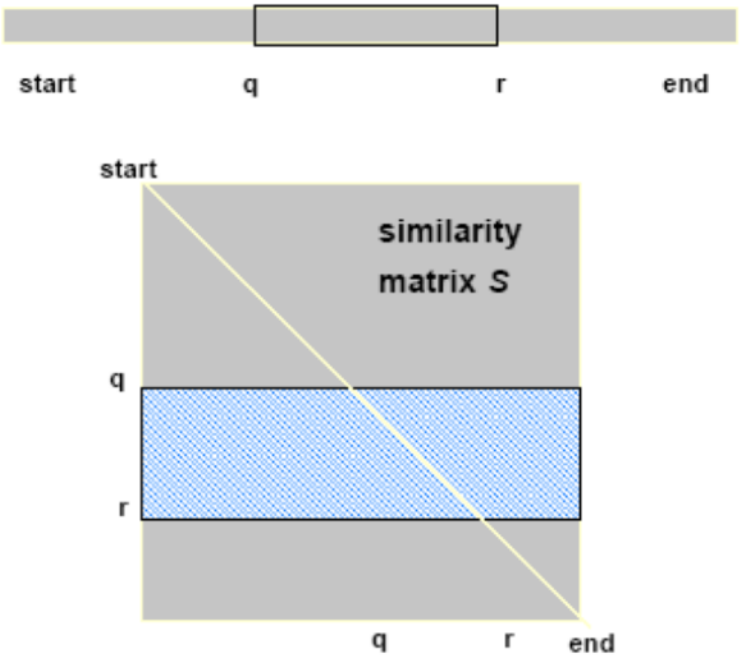
- Method:**"summary score"**
- Search for the continuous time segment which best represents the content of the music track according to a similarity criteria
- → generation of music "preview"



[M. Cooper and J. Foote. Automatic music summarization via similarity analysis. In ISMIR, 2002.]

SSM-based audio summary generation (cont.)

- Search for the segment which starts at q of duration $L = r - q$ which best explains the observed repetitions
- Average similarity between time q with all times of the music track
  - $\frac{1}{N}\sum_{n=1}^{N} \mathbf{S}_{q,n}$
- Average similarity between **segment** $[q, r]$ (of duration $L = r - q$) with all times of the music track
  - $R_{q,L} = \frac{1}{LN}\sum_{m=q}^{r}\sum_{n=1}^{N} \mathbf{S}_{m,n}$
- For a given L, we look for q^* which maximizes $R_{q,L}$
  - $q_L^* = \text{argmax}_{1 \le i \le N-L} R_{q,L}$
- Improvement: to favor the detection of summary at the start of the music track,
  - we add a weighting $w(n)$ function decreasing over time
  - $R_{q,L} = \frac{1}{LN}\sum_{m=q}^{r}\sum_{n=1}^{N} w(n) \cdot S_{m,n}$



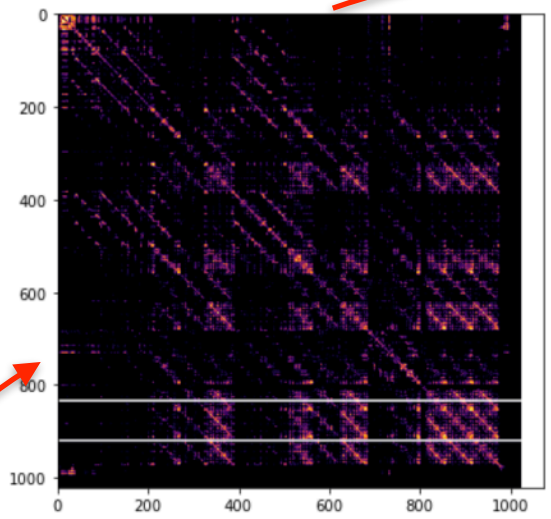
source : [Cooper and Foote, 2002, ISMIR]

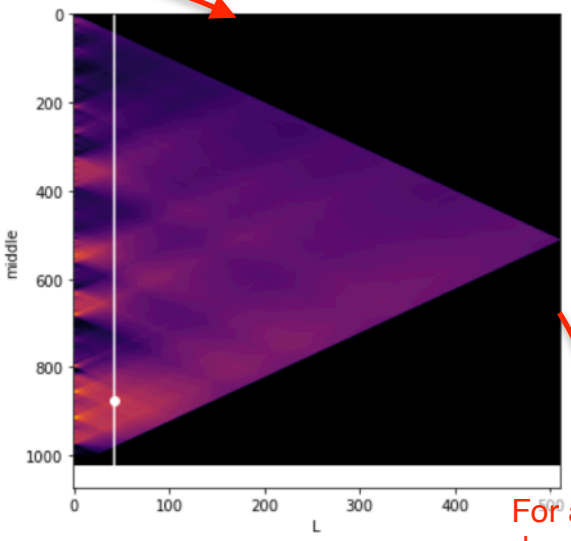
[M. Cooper and J. Foote. Automatic music summarization via similarity analysis. In ISMIR, 2002.]

Simple summary generation

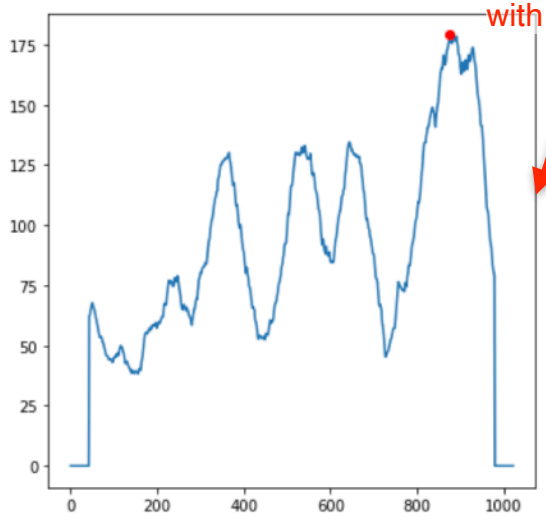
SSM-based audio summary generation (cont.)

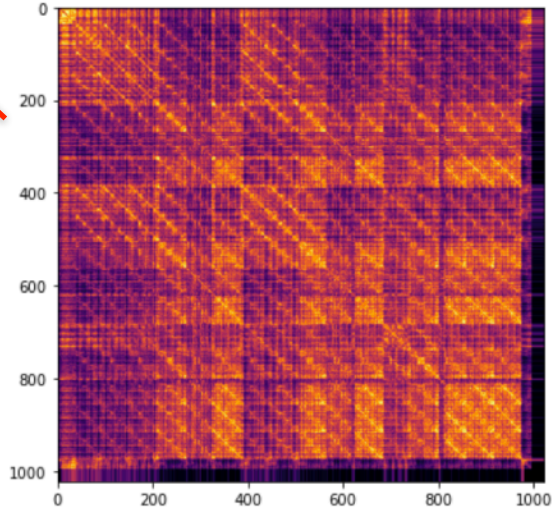
For each starting time and duration, compute the "summary score"





For a given duration (summary duration), choose the starting time with the maximum "summary score"



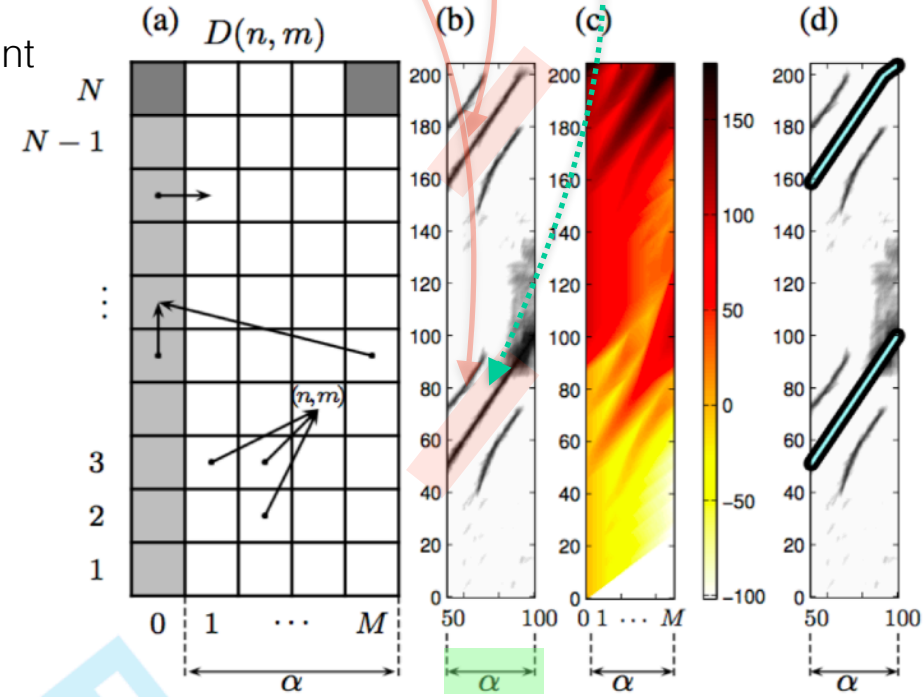


Threshold the SSM to better contrast strong repetitions

Enhanced summary score based on DTW

Detecting repetitions based on DTW

- For a candidate segment,
  find its repetitions
  by performing a DTW over (n,m) on the given corridor α
- Subset of possible paths ($\rightarrow$, $\leftarrow$, $\uparrow$, …)
- Repeat the process for each possible corridor starting time and duration
- Select the one which best explain the content



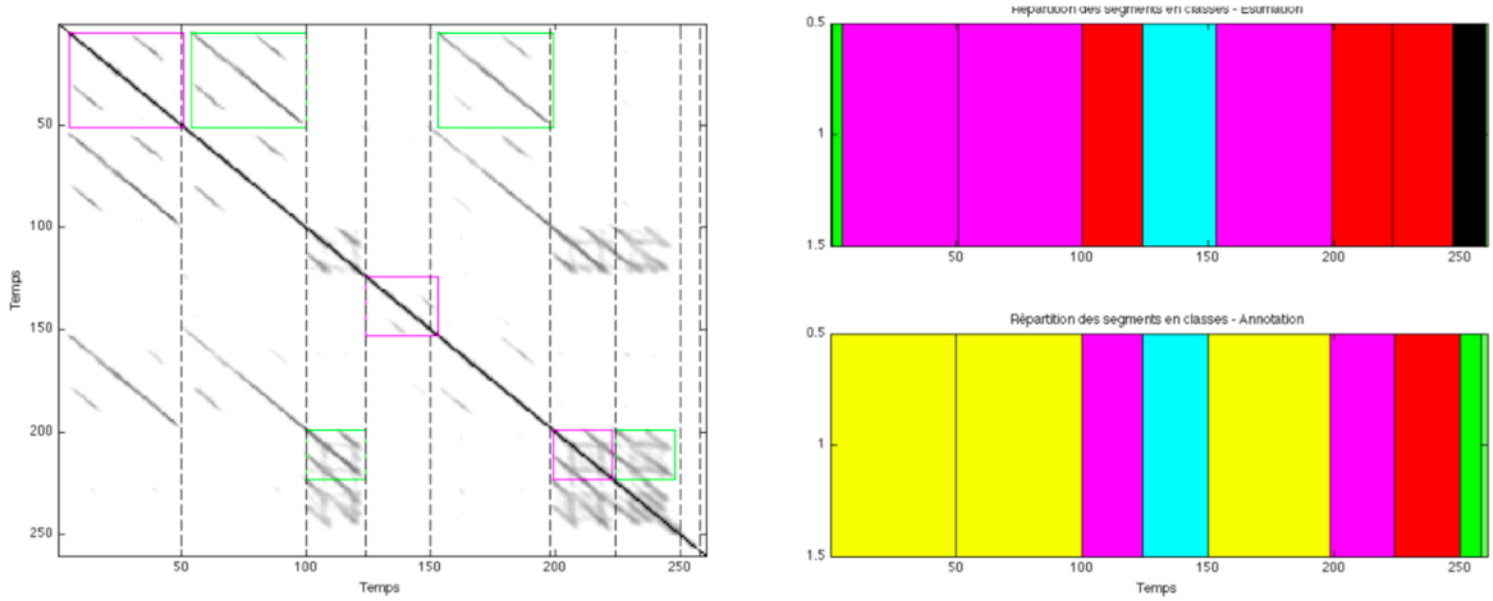
[M. Mueller et al. A robust fitness measure for capturing repetitions, In IEEE TASLP, 2013]

source : Mueller

Use this DTW to estimate the structure

Detecting repetitions based on DTW

- Iterative algorithm
  - Estimate the segment α which best explain the content
  - Subtract from the SSM the "explained" segments
  - Repeat the process



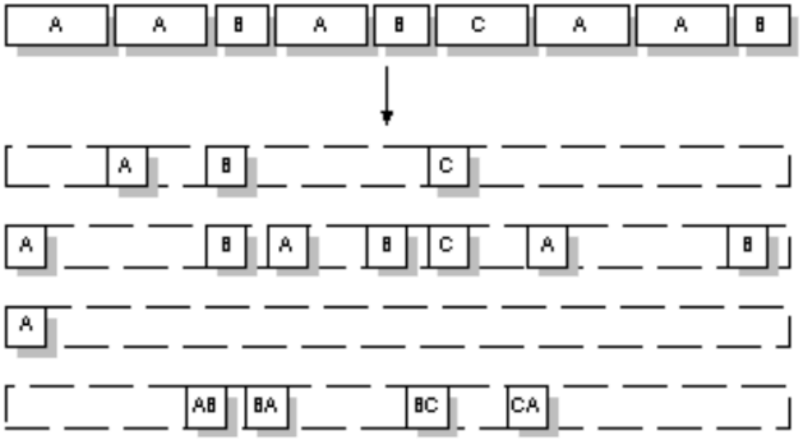
source : Bisot

[V. Bisot and G. Peeters Estimation de la structure musicale par DTW locale, In M2 ATIAM, 2014]

Multi-part summary generation

Structure-based audio summary generation

- Proposed strategy
  - select specific audio extracts according to the content derived from the sequence/state approach
- Summary construction
  - The signal is represented as a succession of sequences/ states A A B A B C A A B
  - Which sequences/ states for the summary?
    - a single example of each sequence/state
    - reproduce the temporal succession of sequences/ states
    - the most important sequence/ state (in terms of number of repetitions, in terms of temporal extension)
    - audio example of transitions between states

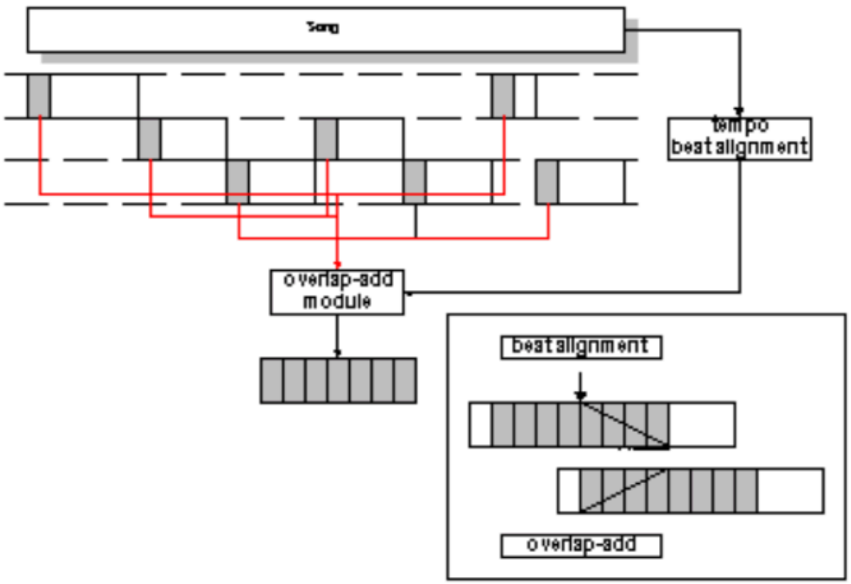


[G. Peeters et al. Toward automatic music audio summary generation from signal analysis. In ISMIR, 2002]

Multi-part summary generation

Structure-based audio summary generation

- Construction of the audio signal:
  - Short extracts of audio signal corresponding to the chosen sequences/ states
  - Must provide a "coherent" and "intelligent" construction
  - Continuity of information:
    - Overlap and Add,
    - respect of tempo/beat,
    - segment size = N x 4 or N x 3 bars,
    - synchronisation to beat positions

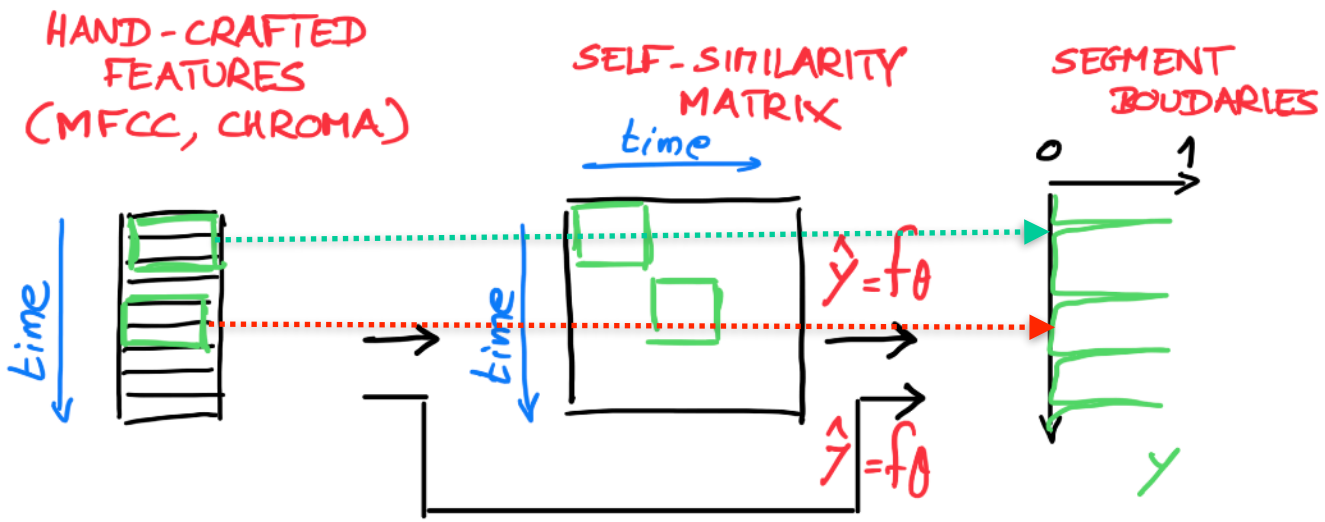


[G. Peeters et al. Toward automatic music audio summary generation from signal analysis. In ISMIR, 2002]

Deep Learning approches
(1) estimation of segment boundaries

Main idea:

- process x = a patch (several time frames centered on τ) to estimate $y \in \{0,1\}$
  - $y = 0 \rightarrow$ no boundary at τ
  - $y = 1 \rightarrow$ boundary at τ
- repeat the process for all τ

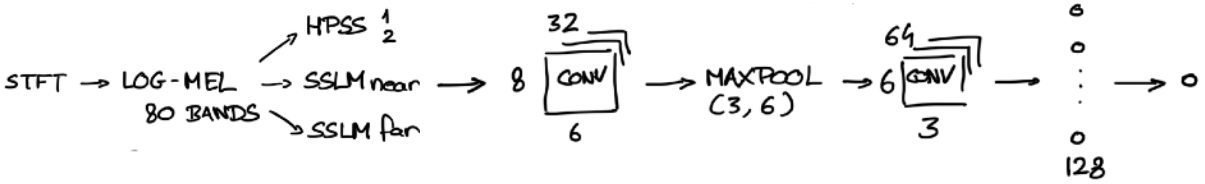


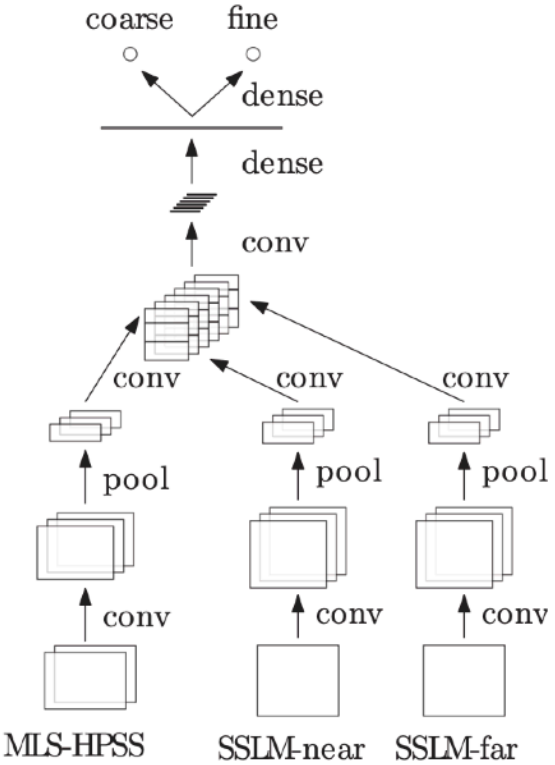
- f_θ is a ConvNet that operates on an input T/F representation (Log-Mel-Gram or SSM-time-**lag**)

[K. Ullrich et al. Boundary detection in music structure analysis using convolutional neural networks. In ISMIR, 2014.]

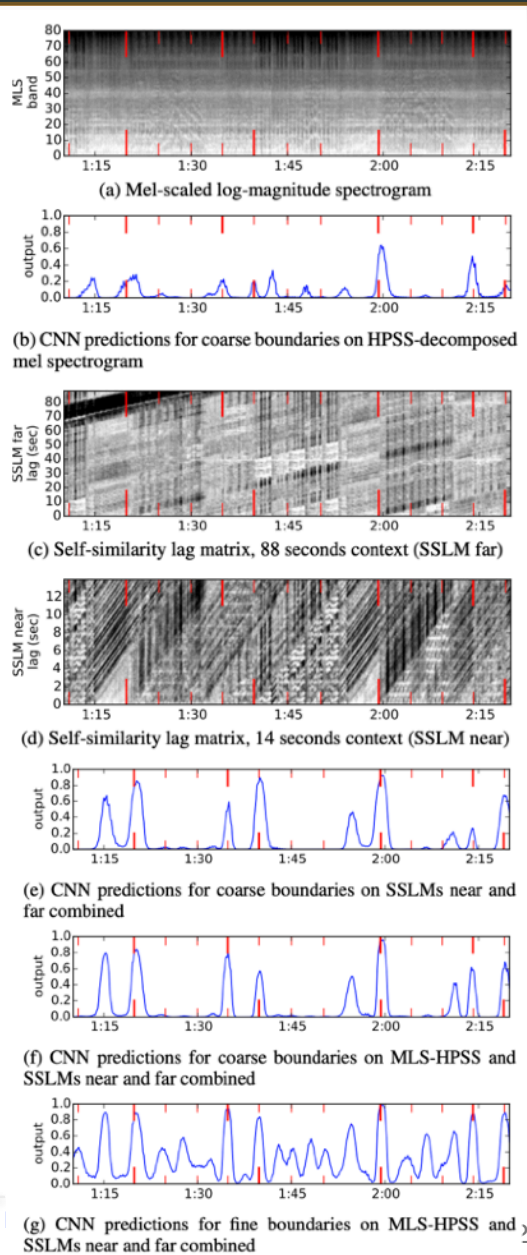
Deep Learning approches
(1) estimation of segment boundaries

Architecture details





[K. Ullrich et al. Boundary detection in music structure analysis using convolutional neural



(a) Mel-scaled log-magnitude spectrogram

(b) CNN predictions for coarse boundaries on HPSS-decomposed mel spectrogram

(c) Self-similarity lag matrix, 88 seconds context (SSLM far)

(d) Self-similarity lag matrix, 14 seconds context (SSLM near)

(e) CNN predictions for coarse boundaries on SSLMs near and far combined

(f) CNN predictions for coarse boundaries on MLS-HPSS and SSLMs near and far combined

(g) CNN predictions for fine boundaries on MLS-HPSS and SSLMs near and far combined

Deep Learning approches
(1) estimation of segment boundaries

Results

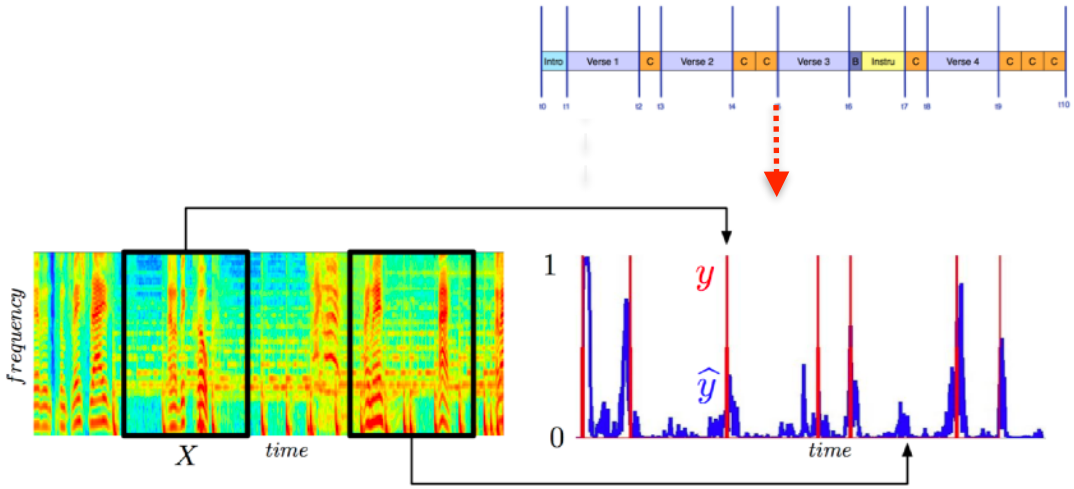
Algorithm	F_1	$F_{0.58}$	Recall	Prec.
Upper bound (est.)	0.74	0.74		
All features, multi+fine ann.	**0.508**	0.529	0.502	0.572
MLS+SSLM-near, multi+fine	0.496	0.506	0.509	0.536
MLS+SSLM-near, single ann.	0.469	0.466	0.504	0.475
SUG1 (2014)	0.422	0.442	0.422	0.490
MP2 (2013)	0.294	0.280	0.362	0.271
MP1 (2013)	0.276	0.270	0.311	0.269
NB1 (2014)	0.270	0.246	0.374	0.229
KSP2 (2012)	0.263	0.231	0.422	0.209
Baseline (est.)	0.15	0.21		

[K. Ullrich et al. Boundary detection in music structure analysis using convolutional neural networks. In ISMIR, 2014.]

Deep Learning approches
(2) improved estimation of segment boundaries

Main ideas

– [Grill and Schluter, 2015], [Ullrich et al., 2014] ideas is good
- X= 2D representation of audio, $y \in \{0,1\}$ the boundaries associated



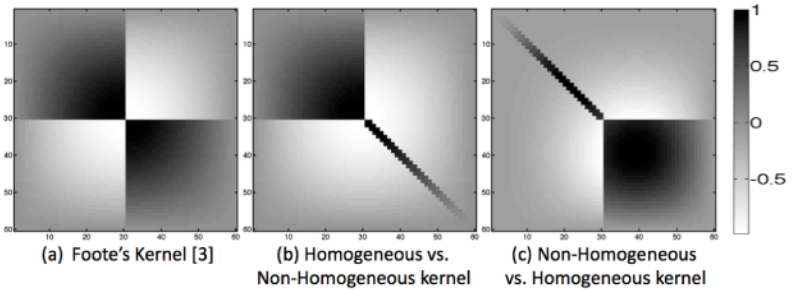
– **BUT**
- (1) we need to perform convolution (ConvNet) on the **SSM-time-time** (as the checkerboard kernels
- (2) we need **several point-of-views** on the content (harmonic, timbre) → Chroma AND MFCCs

[A. Cohen-Hadria and G. Peeters. Music structure boundaries estimation using multiple SSM. In AES Semantic Audio, 2017.]

Deep Learning approches
(2) improved estimation of segment boundaries

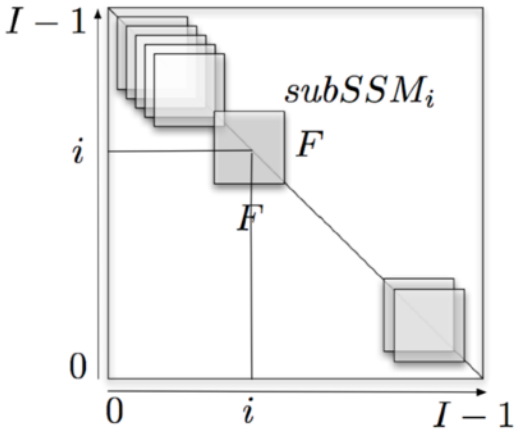
(1) we need to perform convolution (ConvNet) on the SSM-time-time

- Instead of $SSM_{time,lag}$ use $SSM_{time,time}$
  - We will use ConvNet to improve over Foote checkerboard kernels
  - Already used by [Foote, 2000] or by [Kaiser and Peeters, 2013]
  - Provides sharper edges at the beginning and ending of segments than $SSM_{time,lag}$



Kaiser and Peeters multi-kernels for SSM segmentation

- Use **square-sub-matrices** centered on the main diagonal of a Self-Similarity-Matrix time-time as input

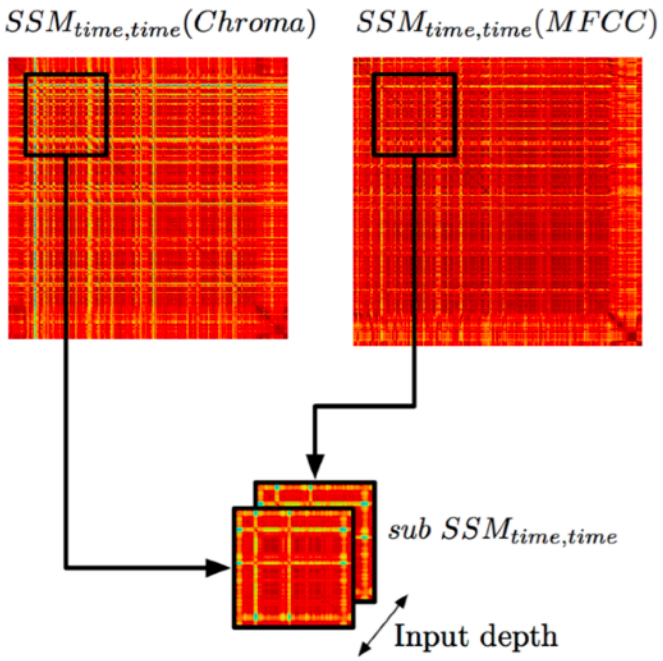


Deep Learning approches
(2) improved estimation of segment boundaries

(2) we need several point-of-views on the content (harmonic, timbre)

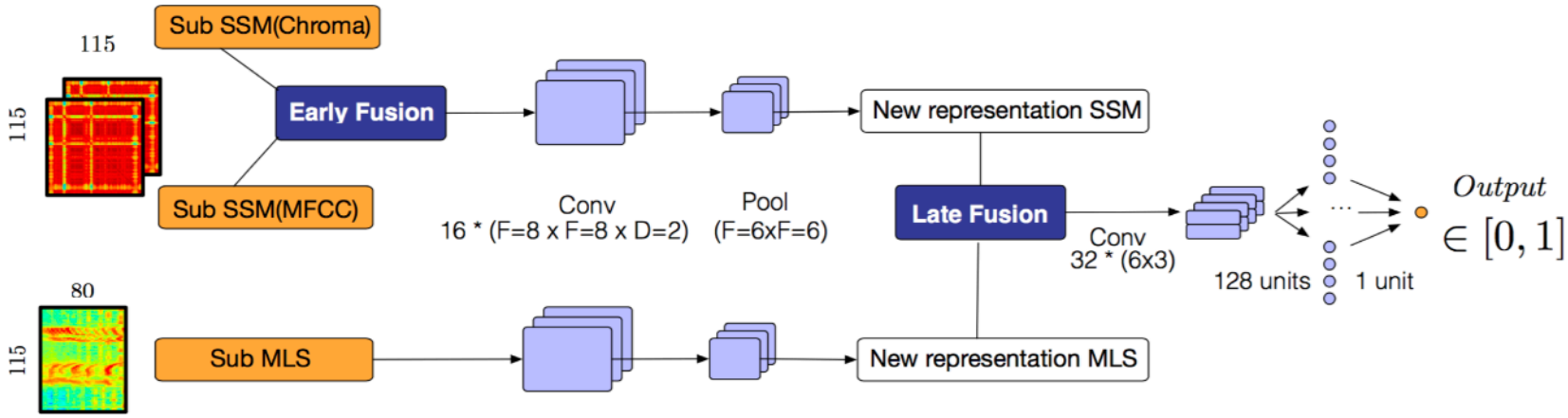
- Use **several SSMs** that highlight the content according to various viewpoints
  - harmony: Chroma
  - timbre: MFCCs

- **How to combine the SMMs ?**
  - Use the depth of the input layer
    - Originally used to represent Red, Blue and Green (RGB) components of the input image.



Deep Learning approches
(2) improved estimation of segment boundaries

Architecture details



– **Training**

- loss: binary-cross entropy
- gradient update: AdaMax
- mini-batch of 128 inputs
- bagging over 5 networks
- number of epochs: when the error on the validation set stop decreasing
- dealing with class unbalancing
  - duplicate frames with $y = 1$ during training to deal with unbalancing
  - temporal smoothing of frames with $y = 1$

Deep Learning approches
(2) improved estimation of segment boundaries

Results

- **Systems compared;**
  - (1) MLS + $SSM_{time,time}$ (MFCC)
  - (2) MLS + $SSM_{time,time}$ (Chroma)
  - MLS + Depth-$SSM_{time,time}$
    - (3) With peak picking
    - (3') With a threshold on the output curve
  - MLS + $SSM_{time,lag}$ (MFCC) [Grill and Schluter, 2015]
    - (4) reimplemented
    - (5) published

- **Results:**
  - Didn't reach state-of-the-art results
    - Maybe because of the size of the dataset
  - Using the self-similarity matrix expressed in time (1) rather than in lag (4) provides an improvement at ±0.5 s and ±3 s.
  - Using the depth of the input layer to combine the two $SSM_{time,time}$ (3) allows us to increase the F-measure at ±0.5 s. and ±3 s.
  - Replacing the peak-picking algorithm (3) by a direct threshold on the network output (3') decreases the results

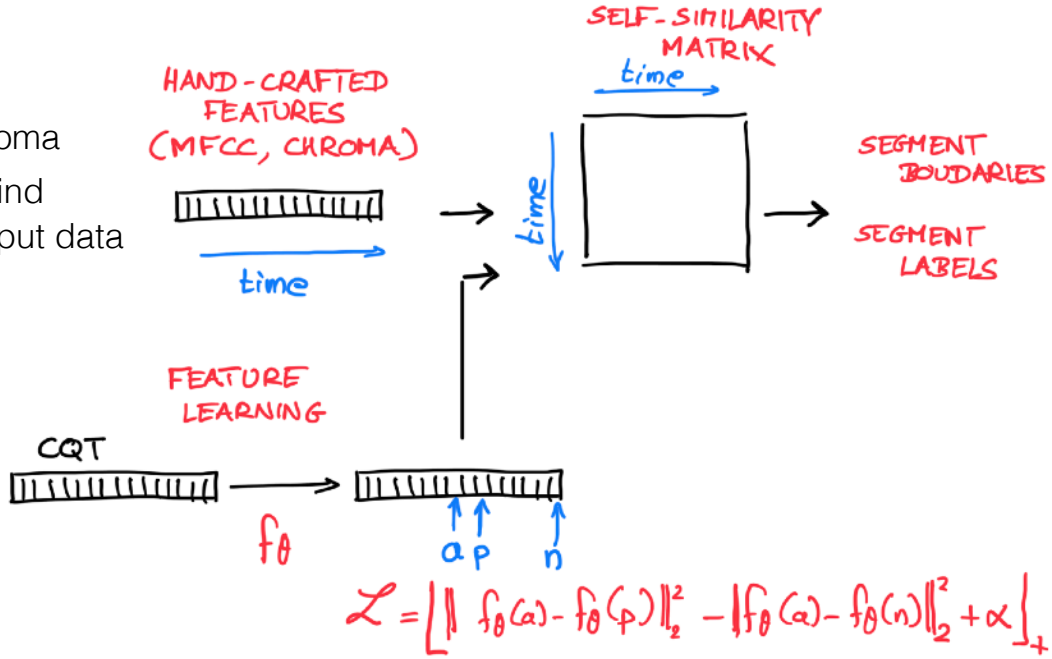
Model	±0.5 s. tolerance				±3 s. tolerance			
	F-m. (std)	Prec.	Rec.	AUC	F-m. (std)	Prec.	Rec.	AUC
① MLS + $subSSM^{mfcc}$	0.273 (0.132)	0.279	0.30	**0.810**	0.551 (0.158)	0.563	0.602	**0.946**
② MLS + $subSSM^{chroma}$	0.270 (0.135)	0.43	0.215	0.800	0.540 (0.153)	0.604	0.555	0.922
③ MLS + Depth($subSSM^{mfcc}$,$subSSM^{chroma}$)	**0.291** (0.120)	0.470	0.225	0.792	**0.629** (0.164)	0.755	0.624	0.930
③' MLS + Depth($subSSM^{mfcc}$,$subSSM^{chroma}$)	0.211 (0.08)	0.128	0.699	0.792	0.618 (0.156)	0.502	0.878	0.930
④[19] re-implemented: MLS+SSLM(MFCC)	0.246 (0.112)	0.291	0.239	0.774	0.580 (0.150)	0.666	0.568	0.927
⑤[19] published: MLS+SSLM(MFCC)	0.523	0.646	0.484					

Deep Learning approches
(3) feature learning by Self-Supervised-Learning

Feature Learning ?

– **Objective:**

- find better features than MFCC or Chroma
- train a Deep Neural Network $f_\theta(\,.\,)$ to find a more meaningful projection of the input data (beat-synchronized CQT)



– **Problem:**

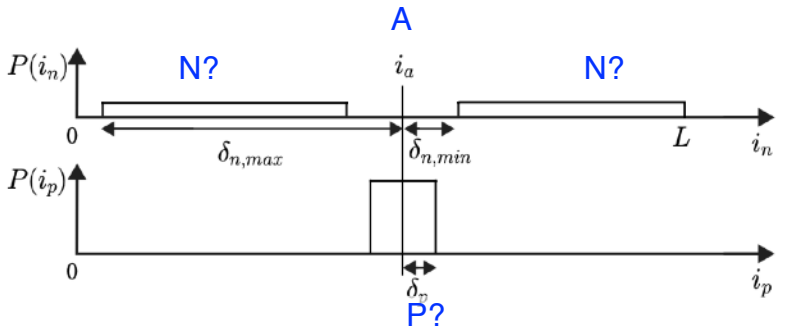
- no large scale training sets for the task of MSD
  - the largest trining set, SALAMI, is only 1360 audio files
- use
  - unsupervised learning (more exactly Self-Supervised Learning)
  - metric learning (more exactly triplet loss): Train $f_\theta(\,.\,)$ such that P is closer to A than N is to A
- use un-labelled data (28,345 non-annotated songs)

[M. McCallum. Unsupervised learning of deep features for music segmentation. In ICASSP, 2019]

Deep Learning approches
(3) feature learning by Self-Supervised-Learning

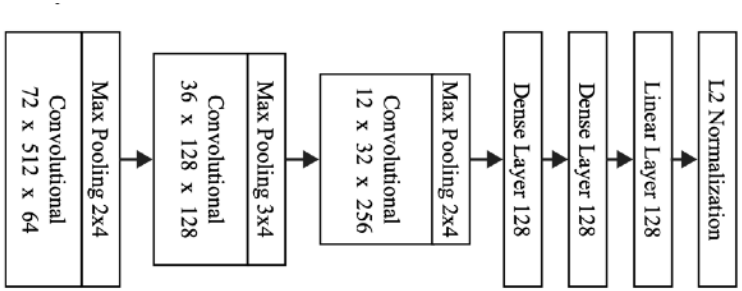
Metric Learning by Triplet Loss:

- **For a given A, how to define P and N**
  - P is sampled in $\{\max(i_a - \delta_p, 0) \dots \min(i_a + \delta_p, L-1)\}$
  - N is sampled in $\{\max(i_a - \delta_{n,\max}, 0) \dots \max(i_a - \delta_{n,\min}, 0)\}$ and $\{\min(i_a + \delta_{n,\min}, 0) \dots \min(i_a + \delta_{n,\max}, L-1)\}$



- **Parameters:**
  - Musical phrases often last 16 beats,
    therefore $\delta_p \geq 16$ discourage distinct clustering of features within a single phrase
  - $\delta_{n,\min} > 28$ and $\delta_{n,\max} > 116$

- **Architecture details**
  - **Input:**
    - dim 12*6=72
    - time 512 frames



[M. McCallum. Unsupervised learning of deep features for music segmentation. In ICASSP, 2019]

Deep Learning approches
(3) feature learning by Self-Supervised-Learning

Results ?

– **Post-processing from learned features**
- Compute SMM using Euclidean distance
- 2D median filtering
- Novelty function $\eta(\nu)$: checkerboard kernel
- Peak-picking:
  - peak-to-mean ratio exceeds a given threshold τ
  - $$\frac{\eta(\nu)}{\frac{1}{2T+1}\sum_{t=-T}^{T}\eta(\nu+t)} > \tau$$

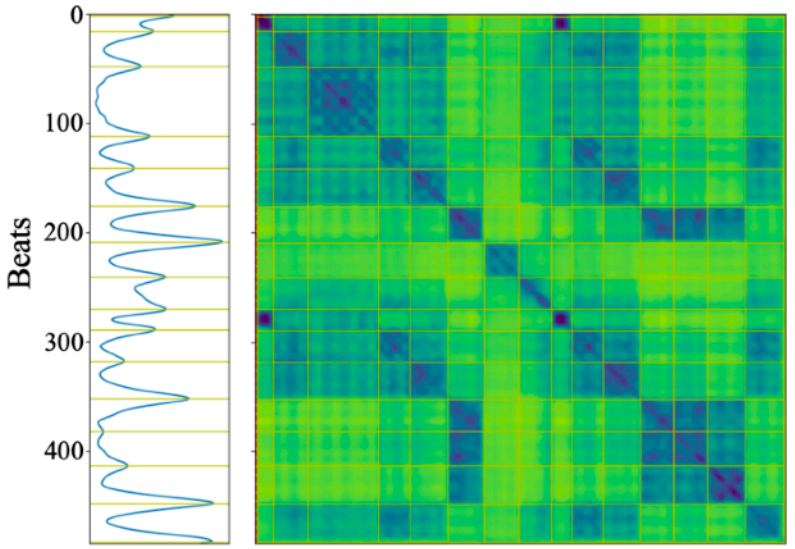


Table 1. Performance metrics for the BeatlesTUT dataset

Algorithm	F-Measure	Precision	Recall
[4]	0.503 ± 0.18	0.579 ± 0.21	0.461 ± 0.17
[6]	0.651 ± 0.17	0.622 ± 0.19	**0.708** ± 0.19
Unsynchronized	0.597 ± 0.17	0.589 ± 0.19	0.625 ± 0.17
Beat-Synchronized	0.648 ± 0.17	0.647 ± 0.20	0.677 ± 0.18
Biased Sampling	**0.662** ± 0.17	**0.663** ± 0.20	0.691 ± 0.19

Table 2. Performance metrics for the SALAMI-A dataset

Algorithm	F-Measure	Precision	Recall
[4]	0.446 ± 0.17	0.457 ± 0.21	0.483 ± 0.19
[6]	0.493 ± 0.17	0.454 ± 0.20	0.595 ± 0.19
Unsynchronized	0.497 ± 0.16	0.429 ± 0.18	0.653 ± 0.15
Beat-Synchronized	**0.535** ± 0.15	**0.491** ± 0.20	**0.660** ± 0.16
Biased Sampling	0.533 ± 0.16	**0.491** ± 0.21	0.656 ± 0.16

[M. McCallum. Unsupervised learning of deep features for music segmentation. In ICASSP, 2019]

Music Structure Discovery (MSD) - Audio Summary Evaluation

Music Structure Discovery (MSD) - Audio Summary Evaluation

Task definition

- https://www.music-ir.org/mirex/wiki/2012:Structural_Segmentation

- Given an audio track
  → estimate the set of temporal (start:end) segments and associated structure label

```
0.0     Silence
0.464399092     Intro
14.379863945    no_function
23.986213151    no_function
33.622494331    Verse
42.956916099    no_function
49.681020408    Transition
67.005941043    Pre-Chorus
76.881292517    Chorus
86.425396825    no_function
98.689433106    Verse
108.166303854   no_function
115.474489795   Transition
129.466938775   Chorus
137.682789115   no_function
160.601927437   no_function
167.620181405   Pre-Chorus
177.151723356   Chorus
194.691836734   no_function
242.415328798   Outro
250.54893424    Fade-out
263.205419501   Silence
264.885215419   End
```

Music Structure Discovery (MSD) - Audio Summary Evaluation

Datasets

- https://www.audiocontentanalysis.org/data-sets/
- QMUL Isophonics (Beatles, Carole King, Queen, Michael Jackson)
  - http://isophonics.net/datasets
- AIST RWC (Real World Computing)
  - INRIA annotations: http://musicdata.gforge.inria.fr/structureAnnotation.html
- INRIA
  - Eurovision, Quaero: http://musicdata.gforge.inria.fr/structureAnnotation.html
- SALAMI (Structural Analysis of Large Amounts of Music Information)
  - https://ddmal.music.mcgill.ca/research/SALAMI/
- Harmonix-Set
  - https://github.com/urinieto/harmonixset
- ...

Music Structure Discovery (MSD) - Audio Summary Evaluation

Performance measures

– **Two criteria to evaluate**

- **(1)** get the correct segment boundaries (independently of the labels)
- **(2)** get the correct label at each time → **labels are relatives !!!**

H. Lukashevich. Toward quantitative measures of evaluating song segmentation. In Proc. of ISMIR (International Society for Music Information Retrieval), Philadelphia, PA, USA, 2008.

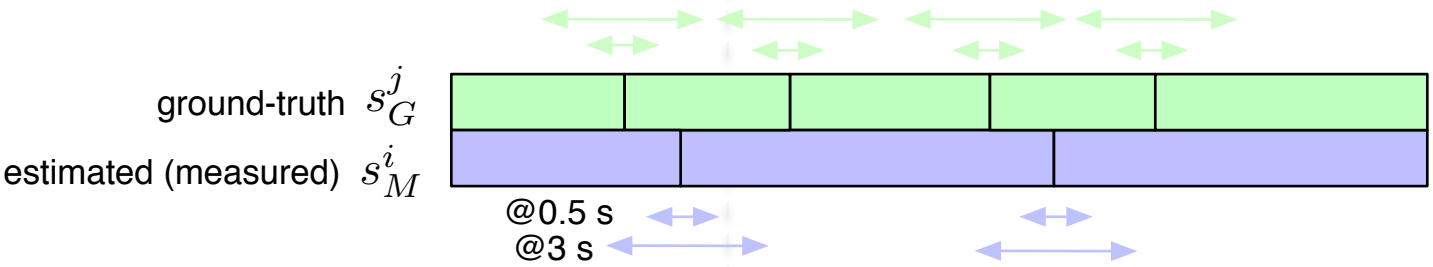
Music Structure Discovery (MSD) - Audio Summary Evaluation

(1) get the correct segment boundaries (independently of the labels)

- S_G^j: ground-truth segments
- S_M^i: estimated (measured) segments

- Segment boundary recovery
  - Recall, Precision, F-measure @0.5 s, @3 s

- Median distance from
  - an annotated segment boundary S_G^j to the closest found boundary S_M^i
  - a found segment boundary S_M^i to the closest annotated boundary S_G^j



(2) get the correct label at each time → labels are relatives !!!

– **Pairwise recall, precision and F-measure**

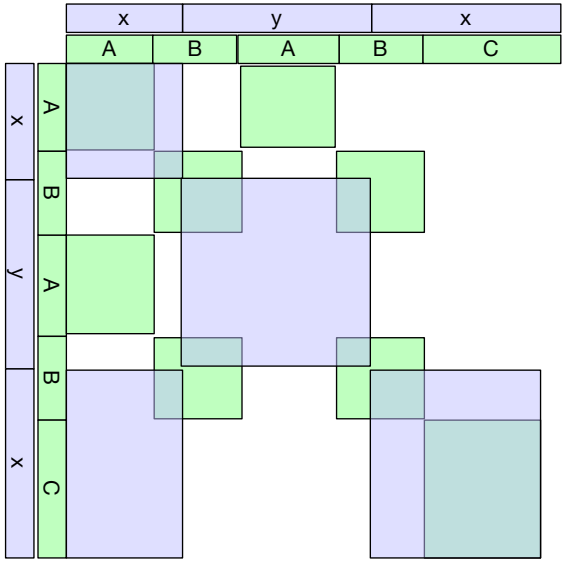
- M_G: set of identically labeled pairs of frames in the **ground-truth** segmentation, i.e. pairs of frames that belong to the same state.
- M_M: set of identically labeled frames in the **estimated** segmentation.

- Pairwise pairwise recall (R_p), precision (P_p),, and pairwise F-measure (F_p) are defined as

$$R_p = \frac{|M_M \cap M_G|}{|M_G|}$$

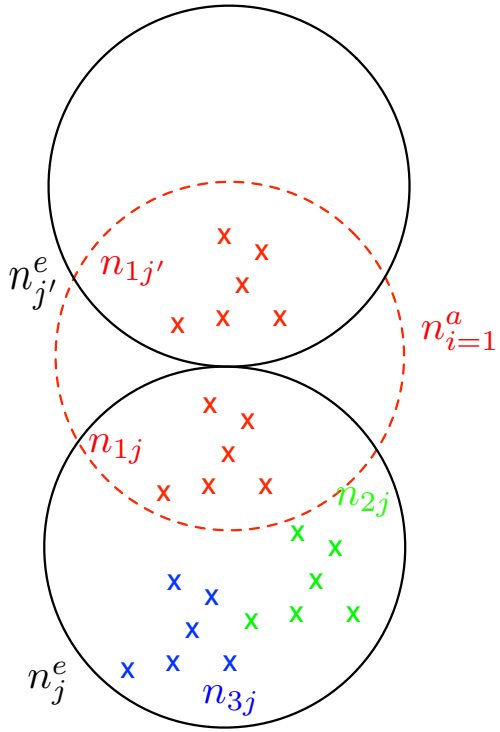
$$P_p = \frac{|M_M \cap M_G|}{|M_M|}$$

$$F_p = \frac{2P_p \cdot R_p}{P_p + R_p}$$



Music Structure Discovery (MSD) - Audio Summary Evaluation

- **(2) get the correct label at each time → labels are relatives !!!**
  - **Cluster purity**
    - Speaker clustering:"to what extent all the utterances from the cluster came from the same speaker"
      - "speaker" → states of the ground-truth segmentation
      - "cluster" → assigned to the states of the estimated one
    - n_{ij}: # frames that simultaneously belong to the annotated-state i estimated-state j
    - n_i^a: # frames that belong to the annotated-state i
      - N_a: # states in the annotated segmentation
    - n_j^e: # of frames that belong to the estimated-state j
      - N_e: # states in the estimated segmentation
    - **Cluster purity / average cluster purity (acp)**
      - $r_j^e = \sum_{i=1}^{N_a} \frac{n_{ij}^2}{(n_j^e)^2}$ $\quad acp = \frac{1}{N} \sum_{j=1}^{N_e} r_j^e \cdot n_j^e$
      - highlights possible under-segmentation errors
    - **Speaker purity / average speaker purity (asp)**
      - $r_i^a = \sum_{j=1}^{N_e} \frac{n_{ij}^2}{(n_i^a)^2}$ $\quad asp = \frac{1}{N} \sum_{i=1}^{N_a} r_i^a \cdot n_i^a$
      - highlights possible over-segmentation errors
    - **Final score:**
      - $K = \sqrt{acp \cdot asp}$



Music Structure Discovery (MSD) - Audio Summary Evaluation

Example of results

Summary

Legend

Submission code	Submission name	Abstract PDF	Contributors
KSP1	ircamstructure_va2	PDF	Florian Kaiser, Thomas Sikora, Geoffroy Peeters
KSP2	ircamstructure_vt1	PDF	Florian Kaiser, Thomas Sikora, Geoffroy Peeters
KSP3	ircamstructure_mph1	PDF	Florian Kaiser, Thomas Sikora, Geoffroy Peeters
MHRAF1	Simbals_Structure	PDF	Benjamin Martin, Pierre Hanna, Matthias Robine, Julien Allali, Pascal Ferraro
OYZS1	OYZS	PDF	Nobutaka Ono, Shinya Yaku, Yuko Zou, Shigeki Sagayama
SBV1	Music Structure Inference System	PDF	Gabriel Sargent, Frédéric Bimbot, Emmanuel Vincent
SMGA1	SMGA1	PDF	Joan Serrà, Meinard Müller, Peter Grosche, Josep Lluis Arcos
SMGA2	SMGA2	PDF	Joan Serrà, Meinard Müller, Peter Grosche, Josep Lluis Arcos
SP1	ircamstructure_va1	PDF	Florian Kaiser, Thomas Sikora, Geoffroy Peeters

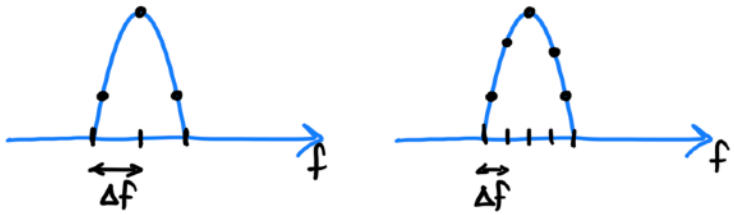
Summary Results [top]

Algorithm	Normalised conditional entropy based over-segmentation score	Normalised conditional entropy based under-segmentation score	Frame pair clustering F-measure	Frame pair clustering precision rate	Frame pair clustering recall rate	Random clustering index	Segment boundary recovery evaluation measure @ 0.5sec	Segment boundary recovery precision rate @ 0.5sec	Segment boundary recovery recall rate @ 0.5sec	Segment boundary recovery evaluation measure @ 3sec	Segment boundary recovery precision rate @ 3sec	Segment boundary recovery recall rate @ 3sec	Median distance from an annotated segment boundary to the closest found boundary	Median distance from a found segment boundary to the closest annotated one
MHRAF1	0.6319	0.5227	0.5722	0.5640	0.6723	0.6432	0.1879	0.1944	0.1992	0.4229	0.4446	0.4402	6.2389	5.4963
SMGA1	0.6231	0.6759	0.5809	0.6762	0.5826	0.6999	0.1924	0.1563	0.2816	0.4920	0.4040	0.7028	1.7681	6.7133
OYZS1	0.6215	0.5456	0.5006	0.5817	0.5954	0.5954	0.2874	0.4580	0.2527	0.4368	0.6409	0.3970	20.4822	3.7538
SP1	0.5899	0.5062	0.5543	0.5490	0.6395	0.6385	0.2789	0.2237	0.4371	0.4906	0.3924	0.7676	1.2193	7.3354
SMGA2	0.5542	0.7400	0.5282	0.7285	0.4712	0.6985	0.1782	0.1460	0.2572	0.4789	0.3959	0.6779	1.9191	6.5524
KSP3	0.5526	0.6052	0.5309	0.6120	0.5261	0.6663	0.2789	0.2237	0.4371	0.4906	0.3924	0.7676	1.2193	7.3354
KSP1	0.5251	0.6789	0.5019	0.6653	0.4464	0.6755	0.2787	0.2234	0.4368	0.4902	0.3921	0.7671	1.2171	7.3263
KSP2	0.5229	0.5026	0.5283	0.5503	0.5792	0.6353	0.2860	0.2291	0.4486	0.4899	0.3915	0.7676	1.1987	7.3180
SBV1	0.4773	0.6481	0.4596	0.6271	0.4250	0.6469	0.1566	0.1359	0.2095	0.4344	0.3781	0.5744	2.6786	8.1783

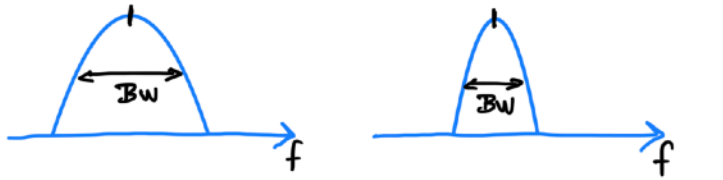
Constant-Q-Transform (CQT)

Constant-Q-Transform (CQT)

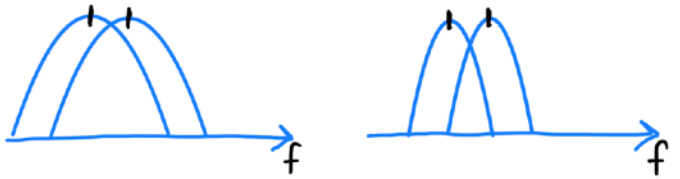
- Discrete Fourier Transform (DFT)
  - _ Definition : Spectral **precision** : $\Delta f = \frac{sr}{N}$
    - it is the step-size at which the Fourier spectrum is sampled
    - it depends on the size of the DFT: N
    - we can improve the precision by increasing N



  - _ Definition : Spectral **resolution** : $B_w = \frac{C_w}{L}$
    - it describes the ability to discriminate (separate in the spectrum) two adjacent simultaneous frequencies



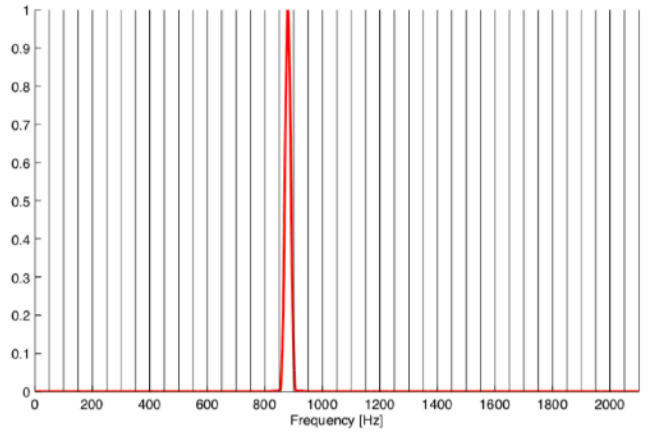
  - – Warning :
    - even if we increase N (zero-padding) while keeping L constant will not improve the resolution !



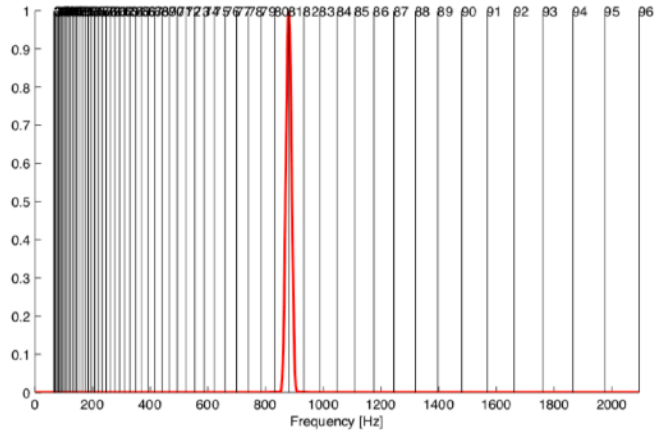
- In a DFT:
  - – Spectral precision and resolution are constant over frequencies

Constant-Q-Transform (CQT)

- In musical audio
  - the frequencies of the pitches are logarithmically spaced $f_k = f_0 \cdot 2^{\frac{k}{12}}$
    - if we choose A-4 = la-3 = 440 Hz as the reference
    - to go from midi-pitches m_k to frequencies f_k:
      - $f_k = 440 \cdot 2^{\frac{m_k - 69}{12}}$
    - to go from frequencies f_k to midi-pitches m_k:
      - $m_k = 12 \log_2 \frac{f_k}{440} + 69$
  - pitch frequencies are
    - close together in low frequencies,
    - distant in high frequencies
  - The **spectral resolution** of the DFT
    - is not sufficient (to separate adjacent notes) in low frequencies
    - is too large for high frequencies



Espacement linéaire de la DFT



Espacement logarithmique des hauteurs de notes

Constant-Q-Transform (CQT)

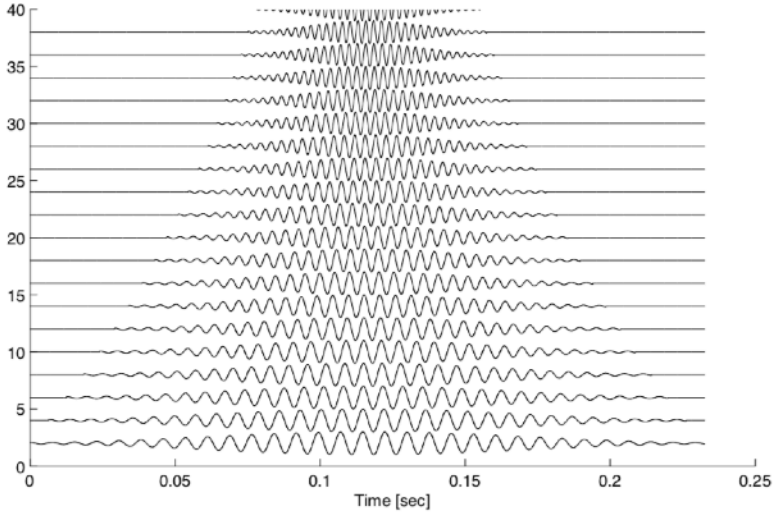
- Solution ?
  - Change the spectral resolution B_w depending on the frequency f_k being considered


An efficient algorithm for the calculation of a constant Q transform

Judith C. Brown

Miller S. Puckette

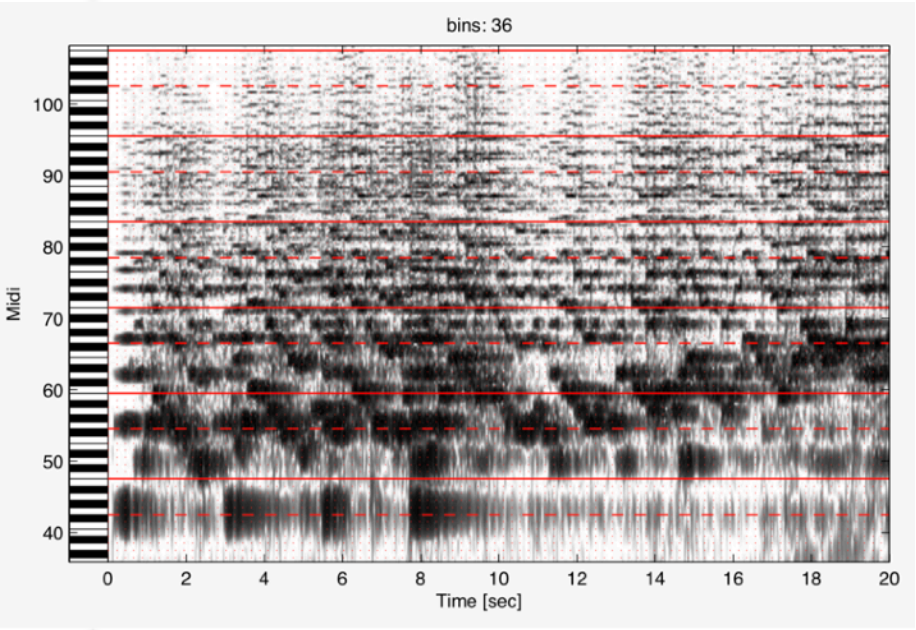
- How ?
  - By changing the window length L for eadh frequency f_k
  - The factor $Q = \frac{f_k}{f_{k+1} - f_k}$ should remains constant in frequency
    - $Q = \frac{f_k}{Bw} = \frac{f_k}{Cw/L} = \frac{f_k \cdot L}{Cw}$
  - We choose a different L for each frequency f_k
    - $L_k = \frac{Q \cdot Cw}{f_k}$



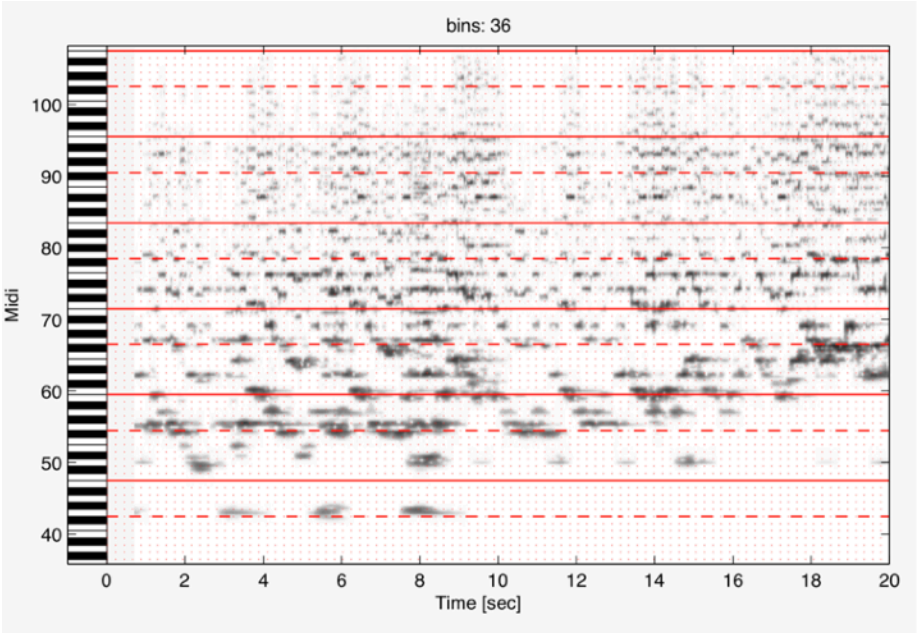
[J. Brown and M. Puckette. An efficient algorithm for the calculation of a constant q transform. JASA, 1992.]

Constant-Q-Transform (CQT)

Example (using DFT)

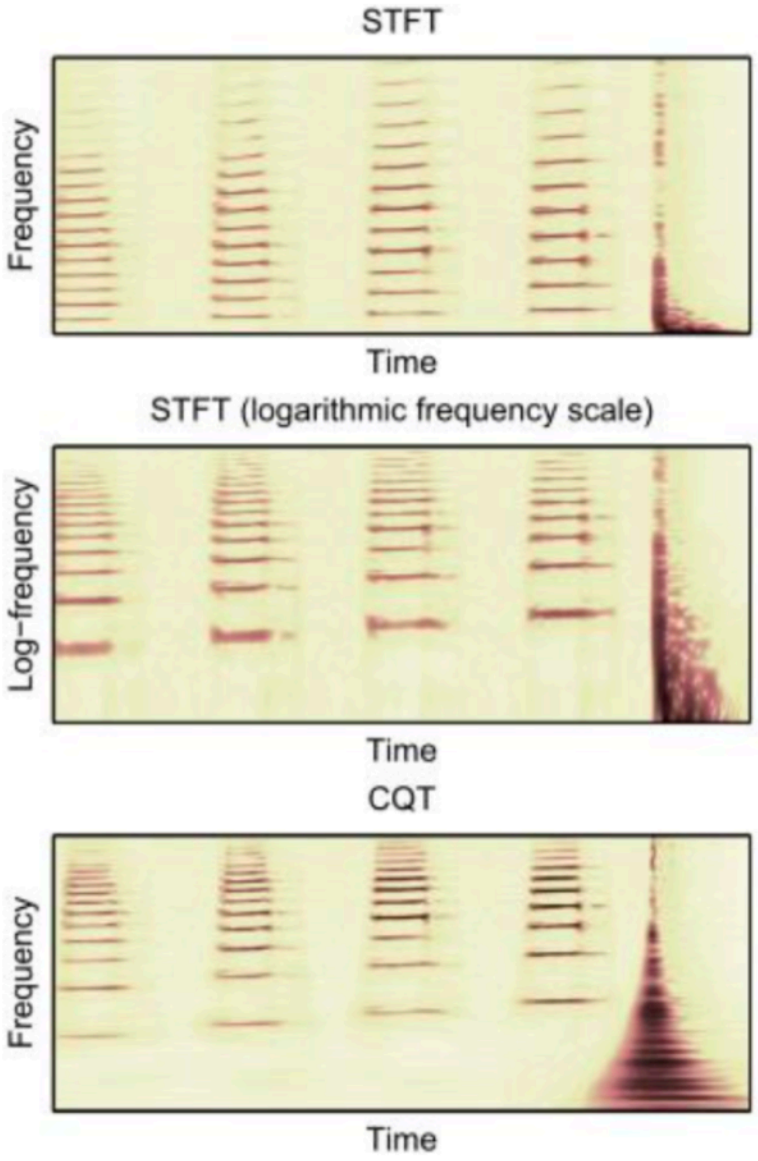


Example (using the CQT)



Constant-Q-Transform (CQT)

- In the Constant-Q-Transform (CQT):
  - A pitch difference corresponds to a translation over the (log) frequency axis



Traditional machine-learning approach
(1) Audio features

Audio features

- Audio features ?
  - A numerical value extracted/estimated from the audio signal which the aim of highlighting a specific content of the audio signal
  - Why not using directly the waveform ? the STFT ?
    - much too high dimensional, much too complex to interpret

- Constraint
  - interpretability
  - low-dimensional
  - same number of dimension for all the data

- Computation ?
  - Mathematical formula
  - Estimation

[G. Peeters. *A large set of audio features for sound description (similarity and classification) in the cuidado project*. Cuidado project report, Ircam, 2004.]

Audio features

- Various **forms**:
  - **scalar**: spectral centroid, spectral spread, fundamental frequency, spectral roll-off, spectral flux, zero-crossing rate, RMS, ...
  - **vector**: Mel Frequency Cepstral Coefficients, coefficients LPC, coefficients PLP, ...

- Various **time validity**:
  - represent one **frame** of the audio signal → "instantaneous" feature
  - represent the content of a **set of local frames** →texture windows
  - represent **globally** the audio signal

- Highlight different facets of the audio **content**:
  - **timbre** content: Mel Frequency Cepstral Coefficients, LPC coefficients, PLP coefficients, ...
  - **harmonic** content: Pitch Class Profiles/ Chroma, ...
  - **noise** content: Spectral Flatness Measure, ...
  - **rhythmic** content: ...

[G. Peeters. *A large set of audio features for sound description (similarity and classification) in the cuidado project*. Cuidado project report, Ircam, 2004.]

Audio features examples

Zero-crossing rate (zcr)

- Measures the number of times the audio waveform cross the zero-axis
  - $$zcr = \frac{1}{N}\sum_{n=1}^{N} |sign(x_n) - sign(x_{n-1})|$$
- Usage: allows to distinguish
  - harmonic sounds → low zcr
  - noise sounds→ high zcr

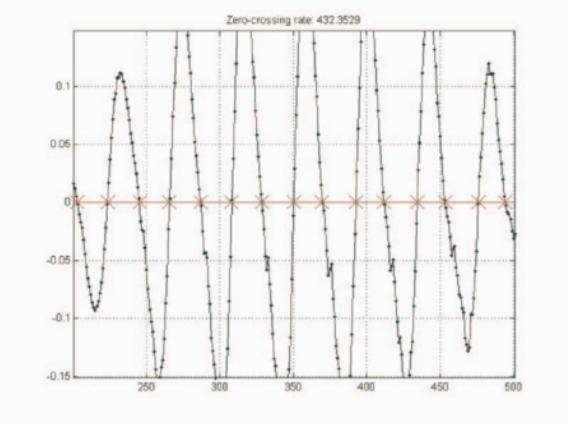


Figure 12 Zero-crossing rate (=432) during voiced speech region

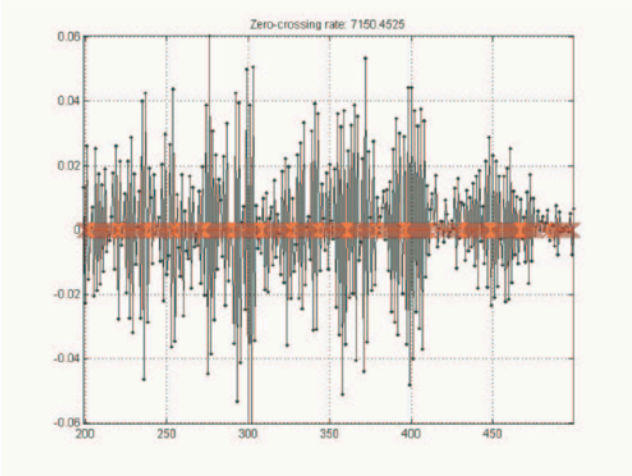


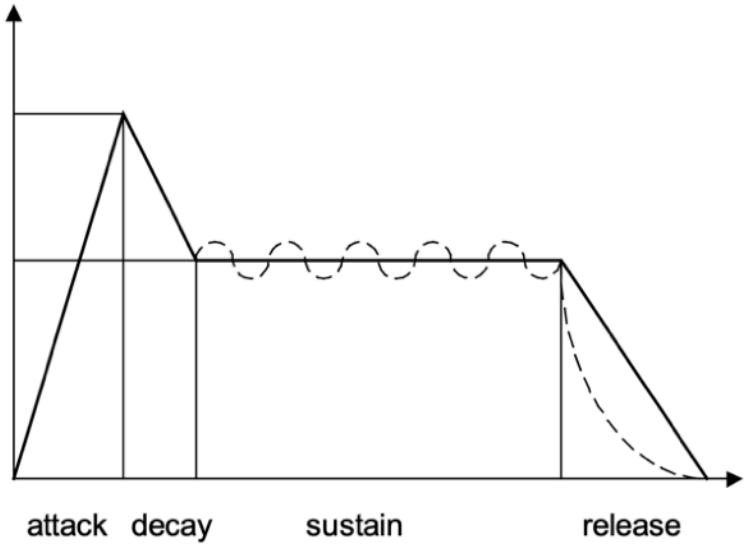
Figure 13 Zero-crossing rate (=7150) during unvoiced speech region

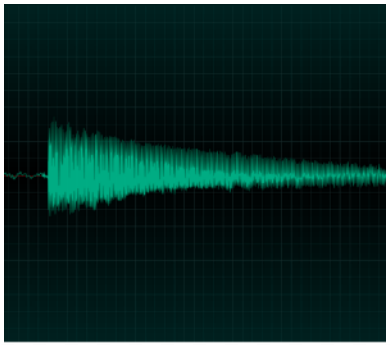
Audio features examples

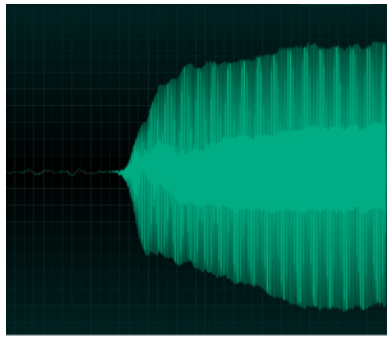
ADSR (Attack, Decay, Sustain, Release) temporal enveloppe

- Model of the temporal evolution (enveloppe) of the energy of a musical note

- Usage: allows to distinguish
  - fast attacks (percussive sounds) /slow attacks
  - fast decrease(non-sustained sounds) / slow decrease (sustained sounds)







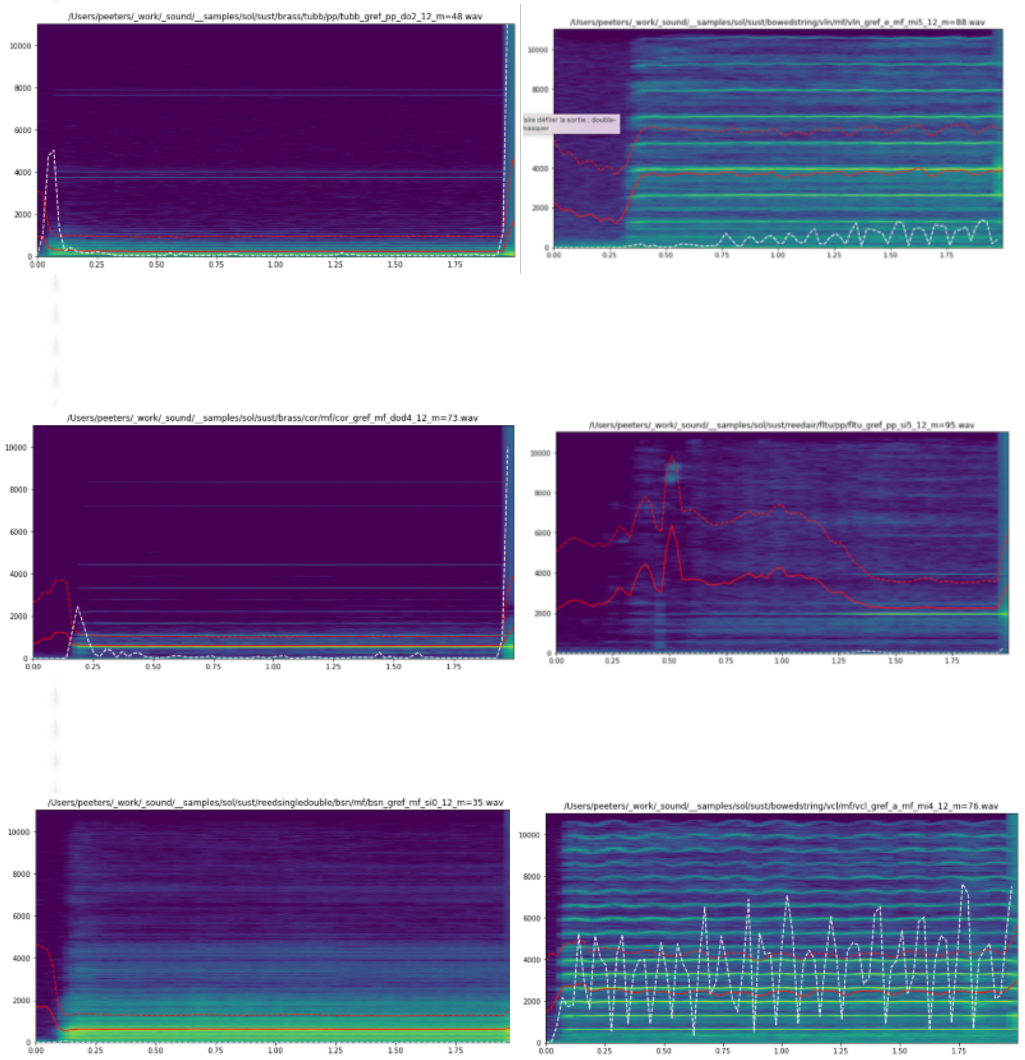
Audio features examples

Spectral shape description

- **Spectral centroid**
  - $$cs = \frac{\sum_k f_k A_k}{\sum_k A_k}$$
  - allows to distinguish between "dull" and "bright" sounds
- **Spectral spread**
  - $$es = \sqrt{\frac{\sum_k (f_k - cs)^2 A_k}{\sum_k A_k}}$$
  - allows to distinguish between "poor" and "rich" sounds
- **Spectral flux**
  - Measure the temporal variation of the spectrum

$$fs = \sum_k (A_k(t) - A_k(t-1))^2$$

  - allows to distinguish between "poor" and "rich" sounds



Audio features examples
Chroma / Pitch-Class-Profile

- **See Lecture 1**

Complex cepstrum

– **Goal**

- describe the shape of the spectrum (the timbre) of a signal using a reduced set of coefficients

– **Complex cepstrum** $c(\tau)$

$$c(\tau) = TF^{-1}\left[\log(X(\omega))\right]$$
$$= \frac{1}{2\pi}\int_{\omega} \log[X(\omega)] \cdot e^{j\omega\tau} d\omega$$

- τ is named "**quefrency**" (=frequency in reverse order)
- $x(t) \xrightarrow{TF} X(\omega) \xrightarrow{\log} \log(X(\omega)) \xrightarrow{TF^{-1}} c(\tau)$

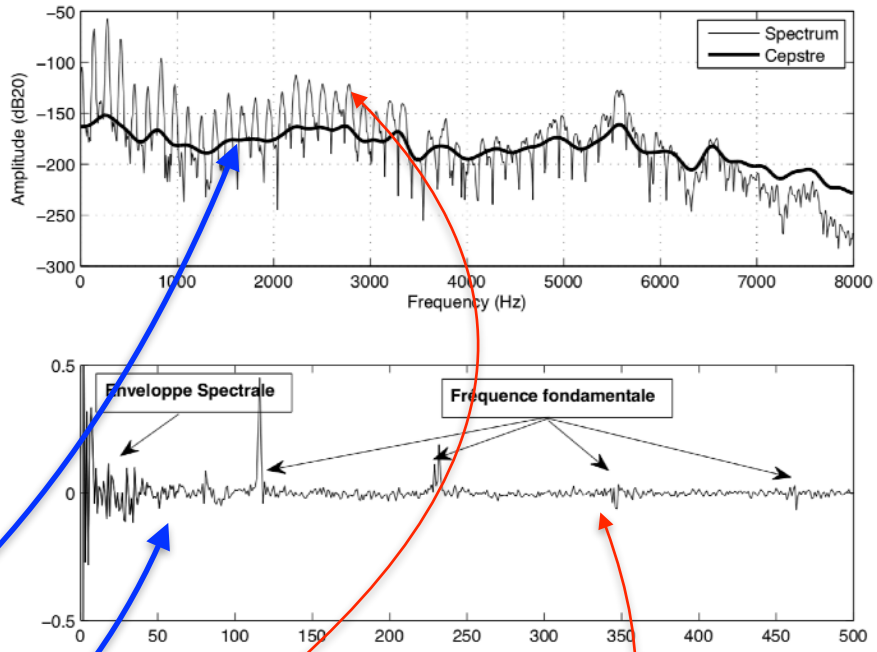
Audio features examples
Mel Frequency Cepstral Coefficients (2)

Source/filter model

- Source $e(t)$: periodic signal
- Filter $g(t)$: resonant/ anti-resonant filter

$$x(t) = e(t) \circledast g(t)$$

$$\xrightarrow{TF} X(\omega) = E(\omega) \cdot G(\omega)$$



$$\xrightarrow{\log} \log(X(\omega)) = \underbrace{\log[G(\omega)]}_{\text{slow variations over } \omega} + \underbrace{\log[E(\omega)]}_{\text{fast variations over } \omega}$$

$$\xrightarrow{TF^{-1}} TF^{-1}\left[\log(X(\omega))\right] = \underbrace{TF^{-1}\left[\log[G(\omega)]\right]}_{\text{energy at quefrency } \tau<<} + \underbrace{TF^{-1}\left[\log[E(\omega)]\right]}_{\text{energy at quefrency } \tau>>}$$

Real cepstrum

– **Real ?** = cepstrum computed on the real part of the log-spectrum

$$X(\omega) = A(\omega) \cdot e^{j\phi(\omega)}$$
$$\log[X(\omega)] = \log[A(\omega)] + j\phi(\omega)$$
$$\Re\{\log[X(\omega)]\} = \log[A(\omega)]$$

$$\text{real cepstrum} = TF^{-1}\left[\Re\{\log[X(\omega)]\}\right]$$
$$= TF^{-1}\left[\log[A(\omega)]\right]$$
$$c(\tau) = \frac{1}{2\pi}\int_{\omega} \log[A(\omega)] \cdot e^{j\omega\tau} d\omega$$

– The amplitude spectrum $A(\omega)$ is real and symmetric
  - its Fourier Transform reduces to the real part
    – reduces to the projection of $\log[A(\omega)]$ on a set of cosinus → Discrete Cosine Transform (DCT)

Audio features examples
Mel Frequency Cepstral Coefficients (4)

Mel Frequency Cepstral Coefficients (MFCCs)

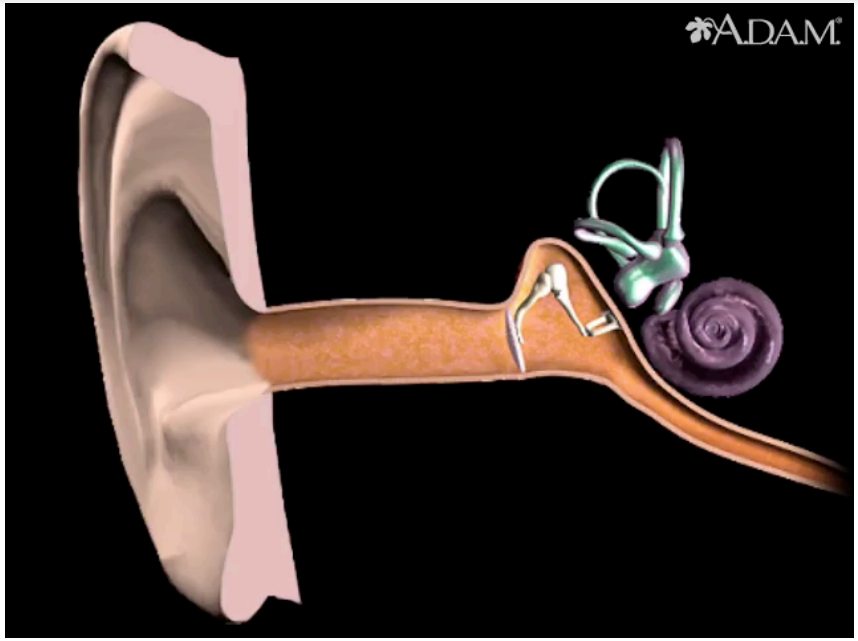
- MFCC ? = real cepstrum computed on the power spectrum $|X(\omega)|^2$ converted to the Mel scale (a perceptual scale)

- **Why perceptual scales ?**
  - Fourier Transform
    - decomposition on a set of sinusoidal components which frequencies are linearly spaced (f_k = 10Hz, 20Hz, 30Hz, ... Hz)
  - Human hearing:
    - decomposition on a set of filters which frequencies are logarithmically spaced (10, 20, 40, 80, ... Hz).
    - highest resolution in low-frequencies, lowest resolution in high frequencies
    - in speech, formants/resonances are closer together in low frequencies
  - MFCCs allows a more compact representation than the real cepstrum
- **How ?**
  - Use of perceptual scales: Mel-scale, Bark-scale, ERB-filters, Gamma-tone filters
- **Usage ?**
  - MFCCs are the most used features in audio: speech, music, environmental sounds recognition, ...

Audio features examples
Mel Frequency Cepstral Coefficients (5)

Human hearing

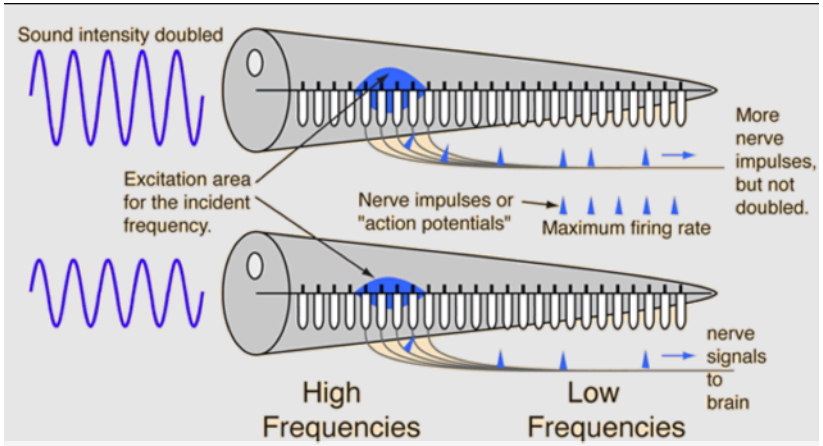
- Cochlea



https://medlineplus.gov/ency/anatomyvideos/000063.htm

- Critical bands
  - perception of two tones at f_1 and f_2
  - perception of a beating-tone at $\frac{f_1+f_2}{2}$

$$\cos f_1 + \cos f_2 = 2\cos\frac{f_1-f_2}{2}\cos\frac{f_1+f_2}{2}$$



source: http://hyperphysics.phy-astr.gsu.edu/hbase/Sound/loud.html

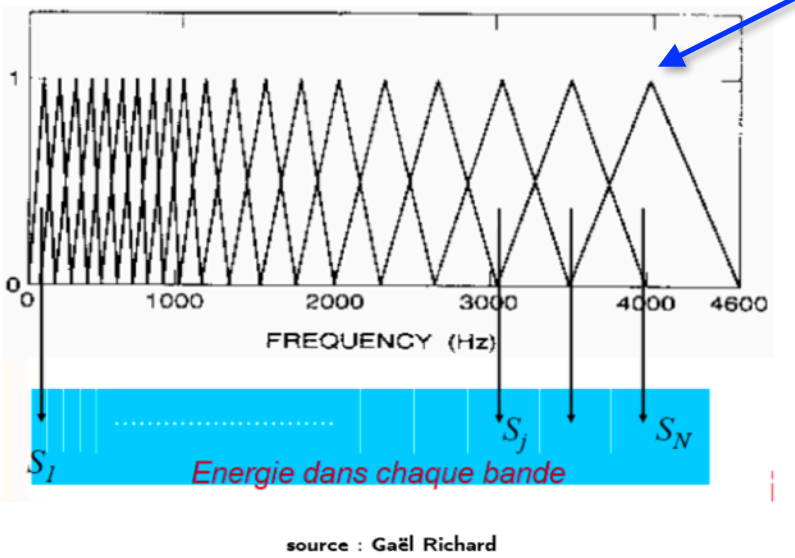
Audio features examples
Mel Frequency Cepstral Coefficients (6)

Mel scale ?

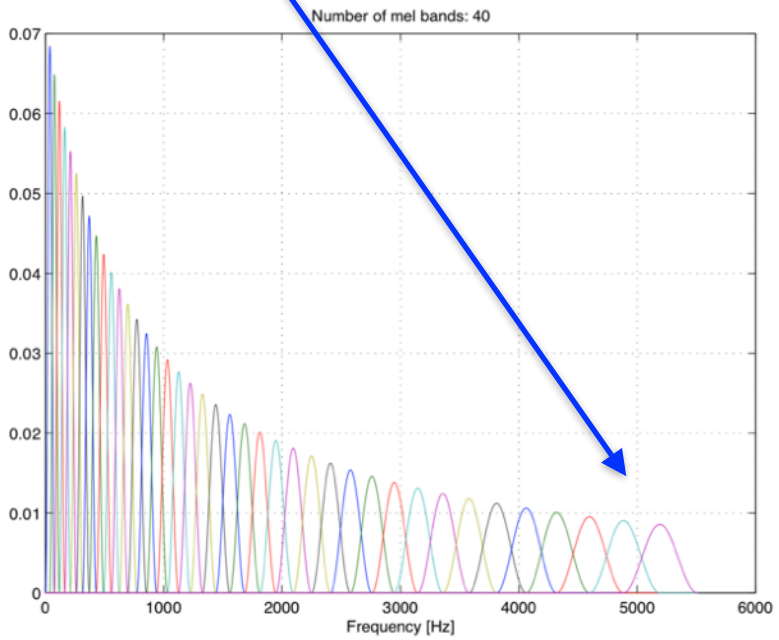
$$mel(f) = \frac{1000}{\ln 2} \ln\left(1 + \frac{f}{1000}\right)$$

- Remark: variations of the constant exist

different shapes for the filter: triangular, hanning, tanh



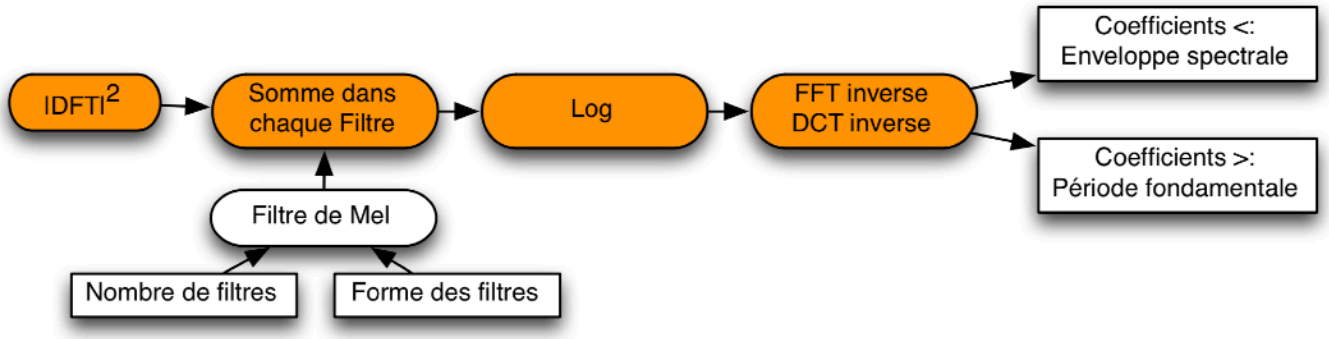
source : Gaël Richard



Fant, Gunnar. (1968). *Analysis and synthesis of speech processes.* In B. Malmberg (Ed.), *Manual of phonetics* (pp. 173-177). Amsterdam: North-Holland.

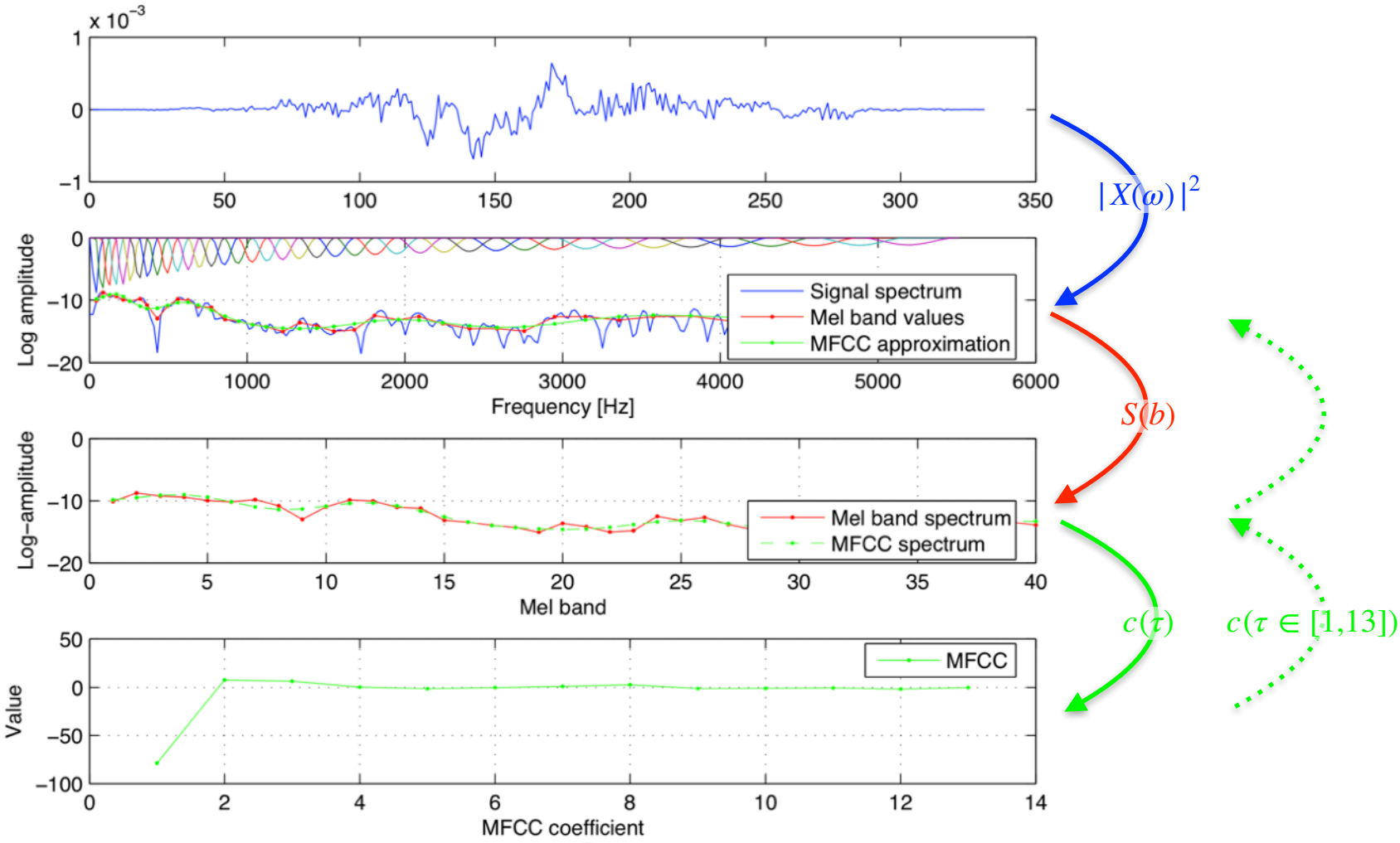
Computation steps for MFCCs

- Compute the power spectrum: $|X(\omega)|^2$
- Compute the Mel filters: $H_b(\omega)$ with $b \in [1,B]$
  - choice of the number of filters B: 40
  - choice of the shape of each filter: triangular, hanning, tanh, ...
- Convert the power spectrum to Mel bands: $S(b) = \sum_{\omega} |X(\omega)|^2 \cdot H_b(\omega)$
- Convert to logarithmic scale: $\log(S(b))$
- Compute the IFFT (or the IDCT): $c(\tau)$
- Select the first coefficients, close to 0 (usually the first 13 coefficients)
  - coefficients close to zero represent the decomposition of the Mel bands content on a set of cosinus with slow variations



Audio features examples
Mel Frequency Cepstral Coefficients (8)

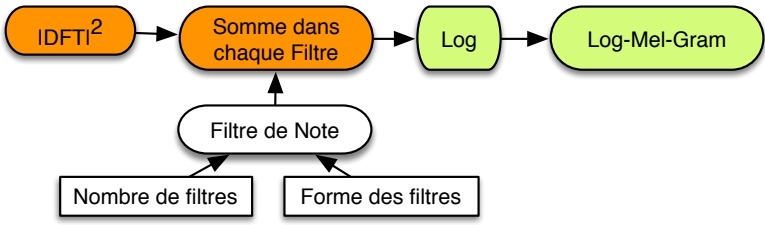
Example of the computation of MFCCs

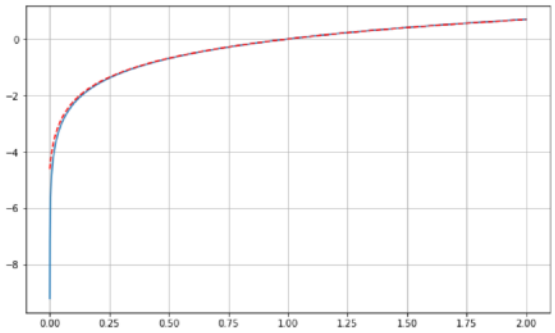


Audio features examples
Mel Frequency Cepstral Coefficients (9)

Variation for the case of DNN inputs: the Log-Mel-gram

- In the **Cepstrum**
  - the DCT is used to separate the contribution of the source and the filter
- In the **Mel Frequency Cepstral Coefficients**
  - the Mel-bands already mostly represent the filter contribution
  - the DCT is mostly used to de-correlated the dimensions
    - (latter used in GMM:diagonal covariance matrix Σ)
- **Log-Mel-Gram**
  - When using DNN, we don't need such a de-correlation of the inputs
  - we then bypass the DCT of the MFCC → Log-Mel-Gram
  - Tricks: to avoid singularity of $\log(x)$ → replace log by $\log(1+\gamma x)$ with $\gamma = 100$





Audio features examples
Spectral Flatness (1)

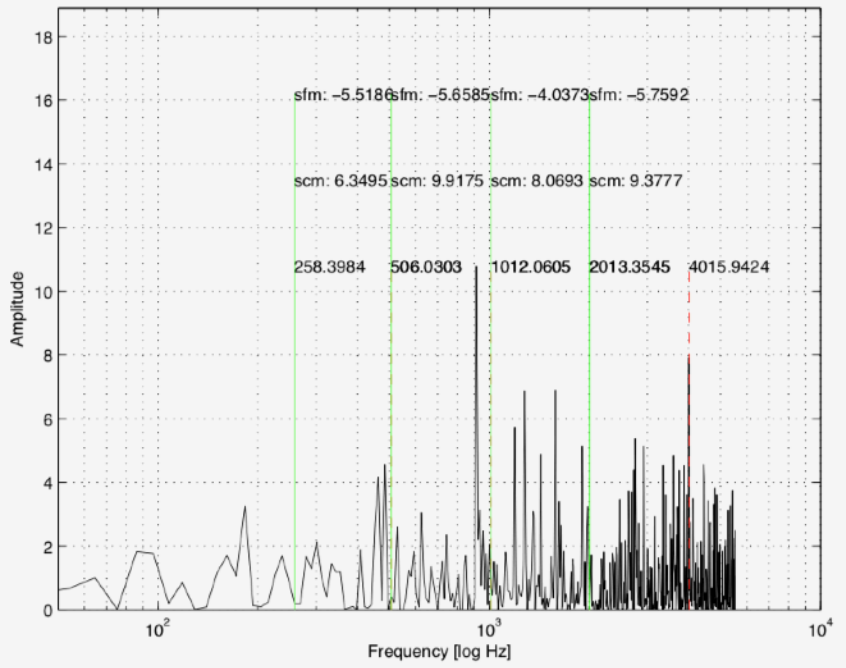
Spectral Flatness Measure (SFM)

- **Motivation:**
  - MFCCs will take the same value wether the content within a Mel-Band is harmonic (peaky) or noisy (flat)
- **Spectral Flatness Measure:**
  - Highlight the harmonic or noise content within each frequency band
  - Measure the flatness of of the spectrum within a frequency band
    - If the band content noise→ spectrum is flat
    - If the band content sinusoidal components → spectrum is peaky
  - Computation: ratio of geometric mean over arithmetic mean
    - $SFM = \frac{\left(\prod_{k \in K} a(k)\right)^{1/K}}{\frac{1}{K}\sum_{k \in K} a(k)}$
    - $SFM \simeq 0$ for tonal signal, $SFM \simeq 1$ for noise signal
  - Computation is done in several frequency bands
    - $[250-500], [500-1000], [1000-2000], [2000-4000]$ Hz (MPEG-7)
- **Tonality measure**
  - $SFM_{dB} = 10\log_{10}(SFM) \qquad Tonality = \min\left(\frac{SFM_{dB}}{-60}, 1\right)$
  - $Tonality \simeq 0$ for noisy signal, $Tonality \simeq 1$ for tonal signal

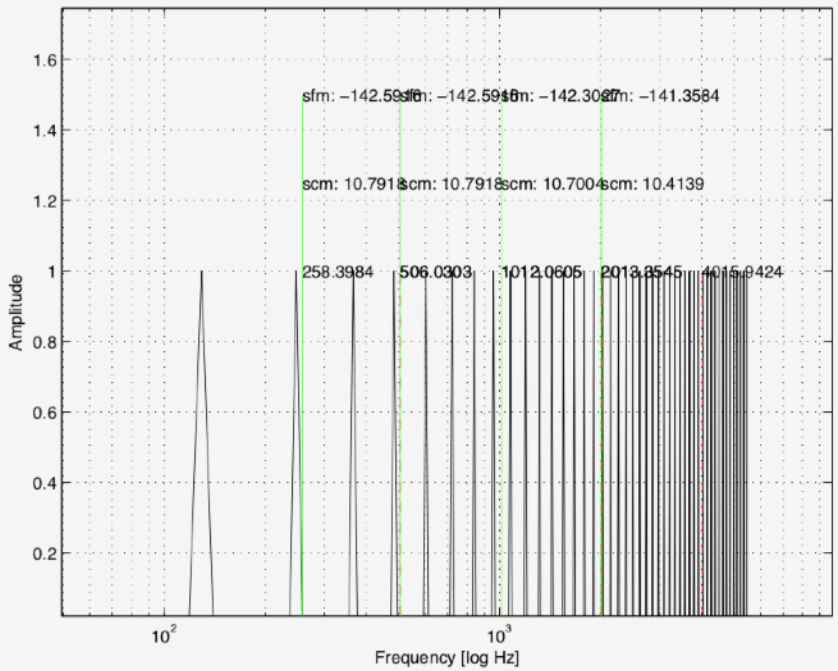
O. Izmirli. Using a spectral flatness based feature for audio segmentation and retrieval. In Proc. of ISMIR, 2000.

Spectral Flatness Measure (SFM) (cont.)

Exemple cas bruité



Exemple cas non-bruité



Séparation de source par TFCT

What you need to know

What you need to know

- Audio features
  - what is the Constant-Q-Transform
  - what are the Cepstrum, real-Cepstrum, MFCCs
- Representations
  - What are a Self-Similarity-Matrix, Lag-Matrix
  - What are the homogeneity, repetition assumptions
- Methods
  - What is the summary score ?
  - What is the checker-board/ novelty curve segmentation method
- Deep learning method
  - How to apply a ConvNet for the structure estimation problem
  - How to apply Self-Supervised-Learning for structure estimation
- Evaluation
  - How to evaluate the performance of a MSD system