

2025 edition

Deep Learning for Music Analysis and Generation

Miscellaneous Topics



Yi-Hsuan Yang Ph.D.
yhyangtw@ntu.edu.tw

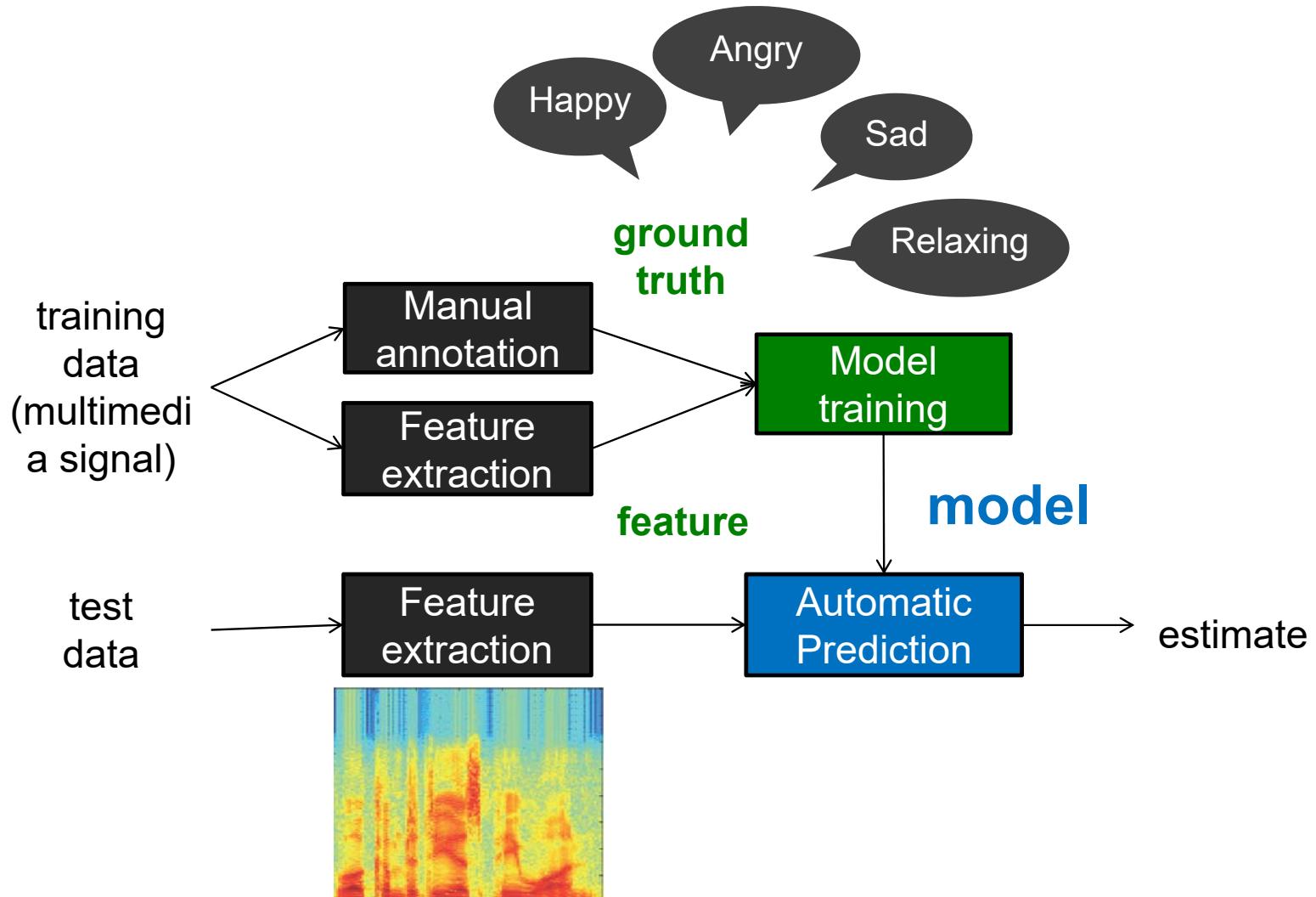
Outline

- **Music emotion**
- Structure/form analysis
- Alignment
- Rhythm & beat tracking

Music & Emotion (Episode 1: Around 2006)

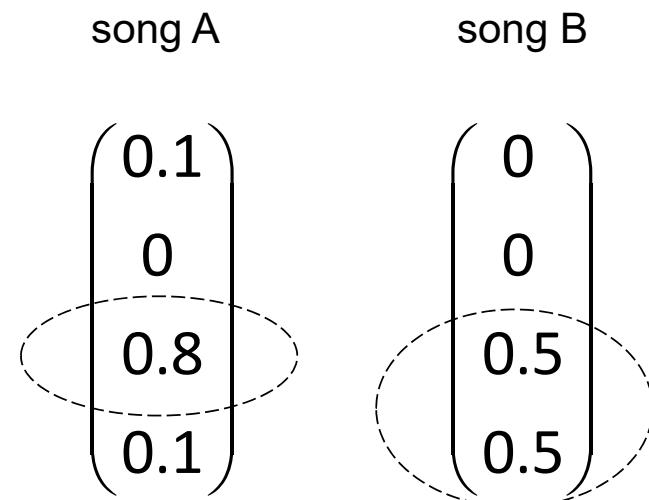
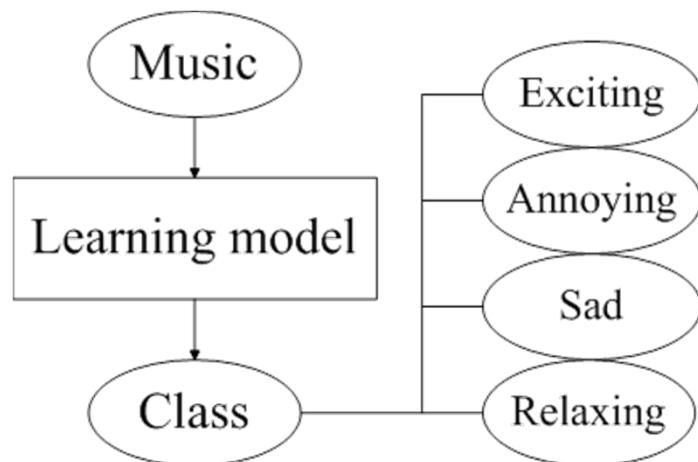
- Yang et al, “Music emotion classification: A fuzzy approach,”
ACM Multimedia, short paper, **2006** [CIEEE Undergraduate Research Award]
- Yang et al, “A regression approach to music emotion recognition,”
IEEE Transactions on Audio, Speech and Language Processing, Feb **2008**
[IEEE Signal Processing Society Young Author Best Paper Award]

Automatic Music Emotion Classification



Subjectivity in Emotion Perception & the Fuzzy Approach

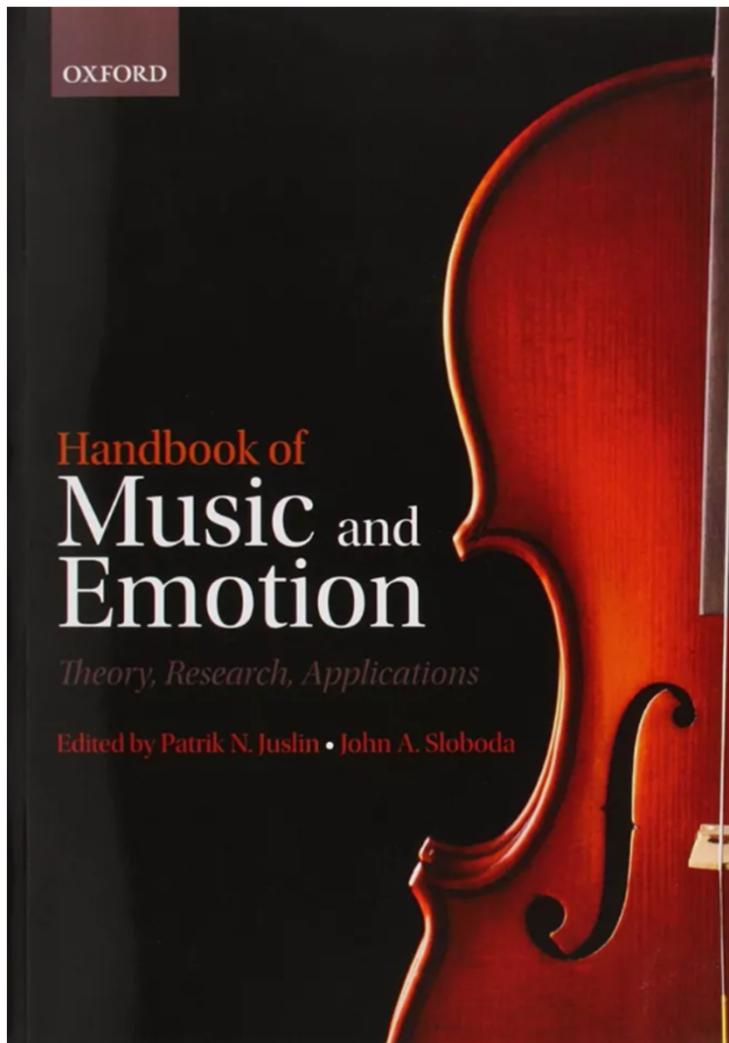
- Traditional classifiers
→ provide an answer (*hard*)
 - NN, k-NN, SVM
- Fuzzy classifiers
→ provide a **fuzzy vector** (*soft*)
 - fuzzy k-NN, Nearest-Mean



Ambiguity & Granularity of Emotion Taxonomy

| Work | # | Emotion description |
|---|----|---|
| Katayose et al [icpr98] | 4 | Gloomy, urbane, pathetic, serious |
| Feng et al [sigir03] | 4 | Happy, angry, fear, sad |
| Li et al [ismir03], Wieczorkowska et al [imtci04] | 13 | Happy, light, graceful, dreamy, longing, dark, sacred, dramatic, agitated, frustrated, mysterious, passionate, bluesy |
| Wang et al [icsp04] | 6 | Joyous, robust, restless, lyrical, sober, gloomy |
| Tolos et al [ccnc05] | 3 | Happy, aggressive, melancholic+calm |
| Lu et al [taslp06] | 4 | Exuberant, anxious/frantic, depressed, content |
| Yang et al [mm06] | 4 | Happy, angry, sad, relaxed |
| Skowronek et al [ismir07] | 12 | Arousing, angry, calming, carefree, cheerful, emo- tional, loving, peaceful, powerful, sad, restless, tender |
| Wu et al [mmm08] | 8 | Happy, light, easy, touching, sad, sublime, grand, exciting |
| Hu et al [ismir08] | 5 | Passionate, cheerful, bittersweet, witty, aggressive |
| Trohidis et al [ismir08] | 6 | Surprised, happy, relaxed, quiet, sad, angry |

Inspirations from Music Psychology Literature



BOOK

Handbook of Music and Emotion: Theory, Research, Applications

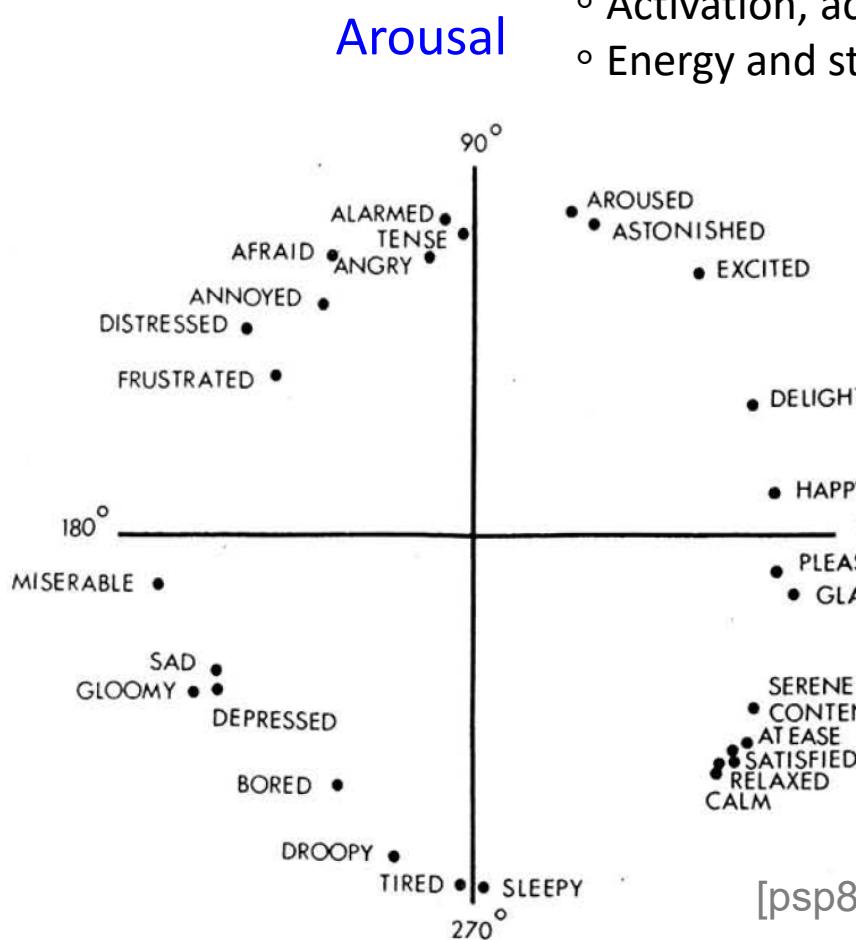
[Get access >](#)

Patrik N. Juslin

Published online: 22 March 2012

Published in print: 1 January 2010

Dimensional Model

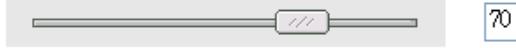


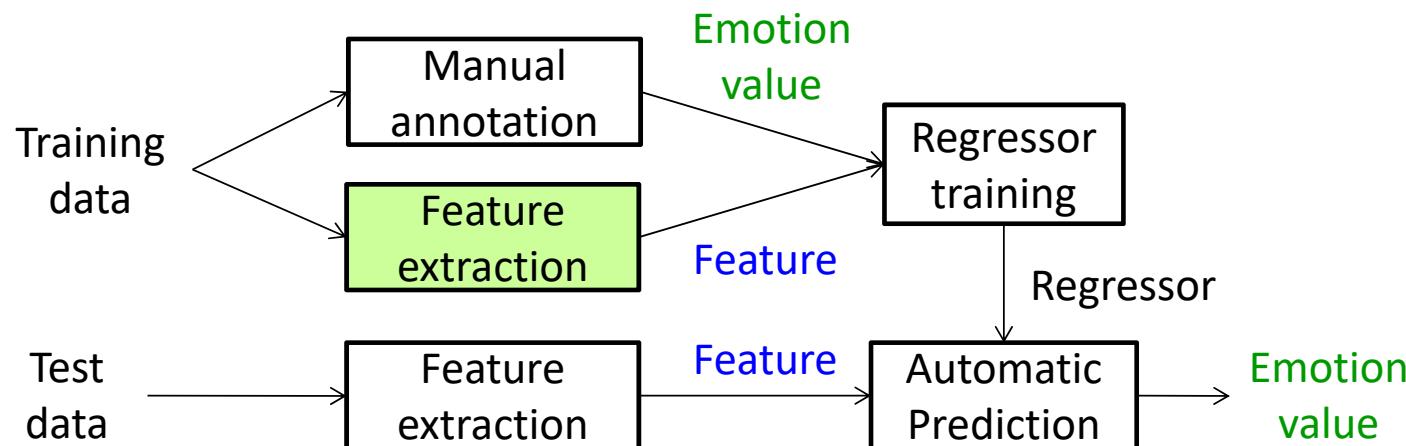
- Activation, activity
- Energy and stimulation level

Valence

- Pleasantness
- Positive and negative affective states

Rating instead of Labeling/Classifying

- Manual annotation
 - Rates the VA values of each song
 - Ordinal rating scale 1 2 3 4 5
 - Scroll bar 

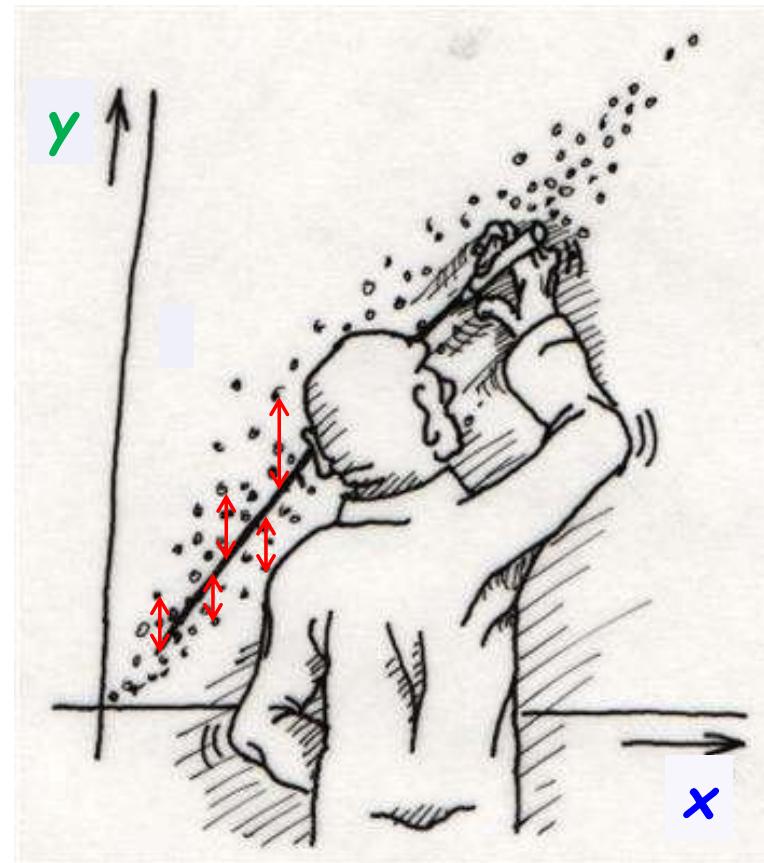


Regression instead of Classification

- Regression
 - Given features, predict a numerical value

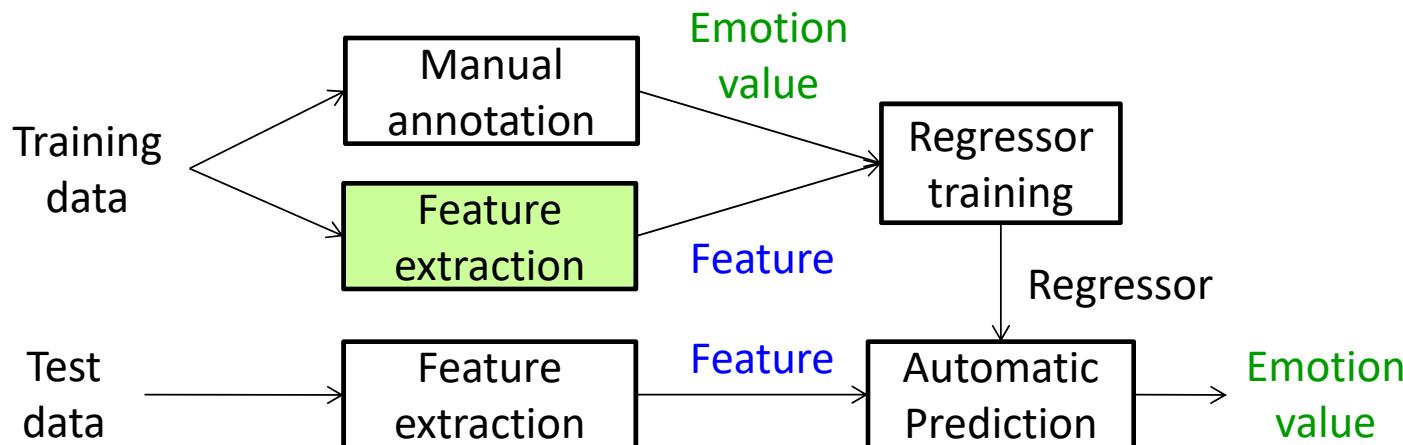
Given N inputs (x_i, y_i) , $1 \leq i \leq N$, where x_i is feature and y_i is the numerical value to be predicted, train a regression model $R(\cdot)$ such that the following mean squared error (MSE) is minimized

$$\min_f \frac{1}{N} \sum_{i=1}^N (y_i - f(x_i))^2$$

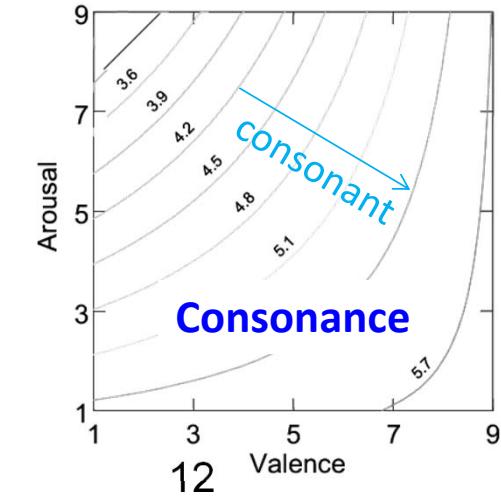
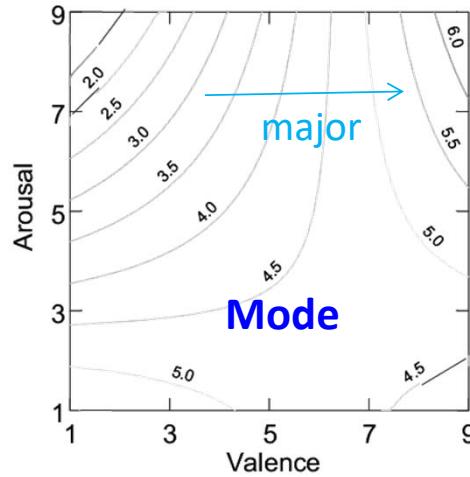
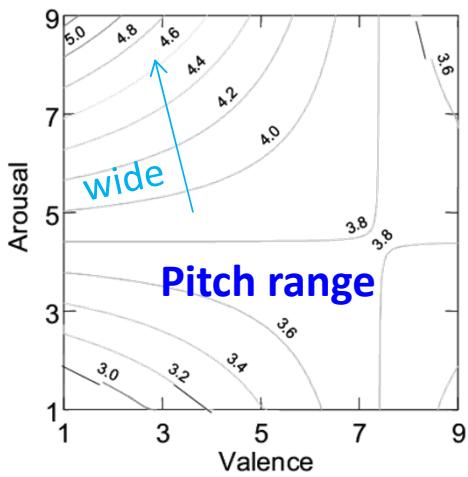
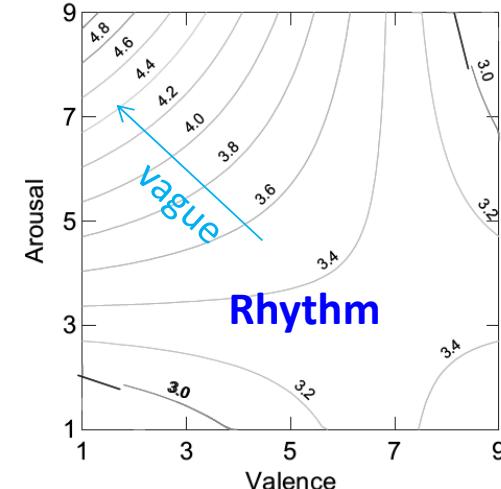
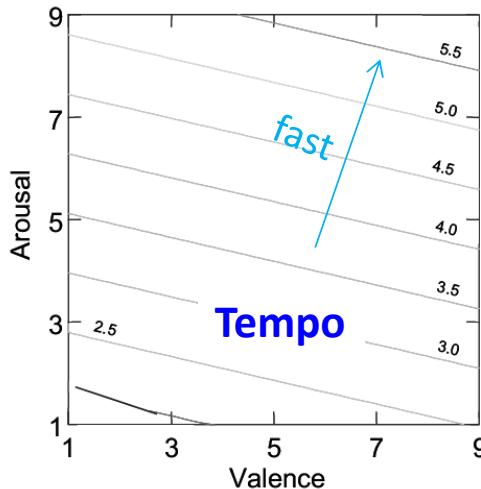
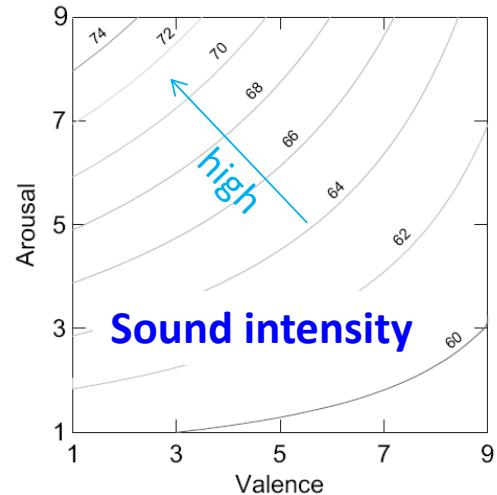


Computational Framework

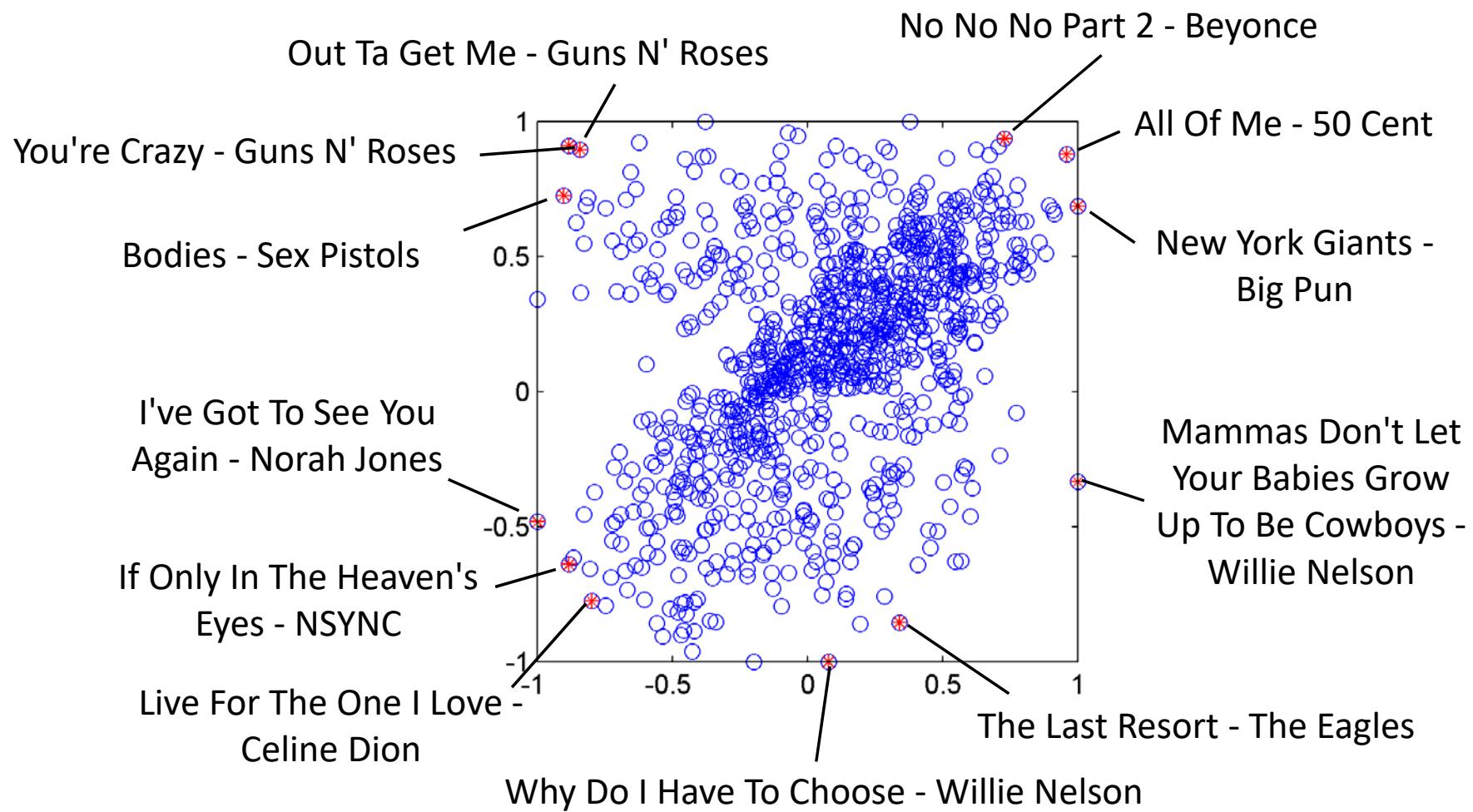
- Feature extraction
 - Arousal
 - **Pitch:** high/low
 - **Tempo:** fast/slow
 - **Timbre:** bright/soft
 - Valence
 - **Harmony:** consonant/dissonant
 - **Mode:** major/minor
 - **Tonality:** tonal/atonal



Extracting Relevant Audio Features

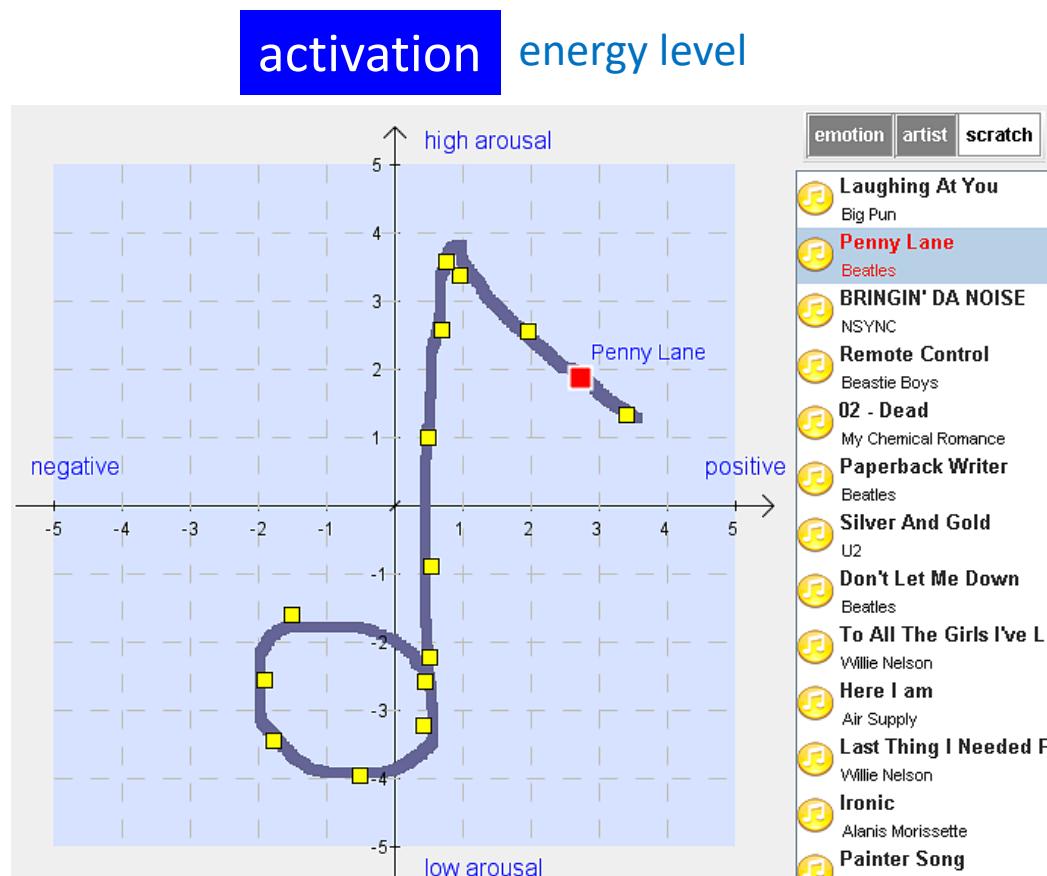


Mapping Music to the Emotion Space



Music Retrieval in the Emotion Space

<https://www.youtube.com/watch?v=ra55xO20UHU>



- Simple 2D user interface
- Useful for mobile devices

Difficulty of Emotion Annotation

- Rating emotion in a continuum is difficult
 - User fatigue
 - Less reliable ground truth
 - Small-scale dataset

1 2 3 4 5

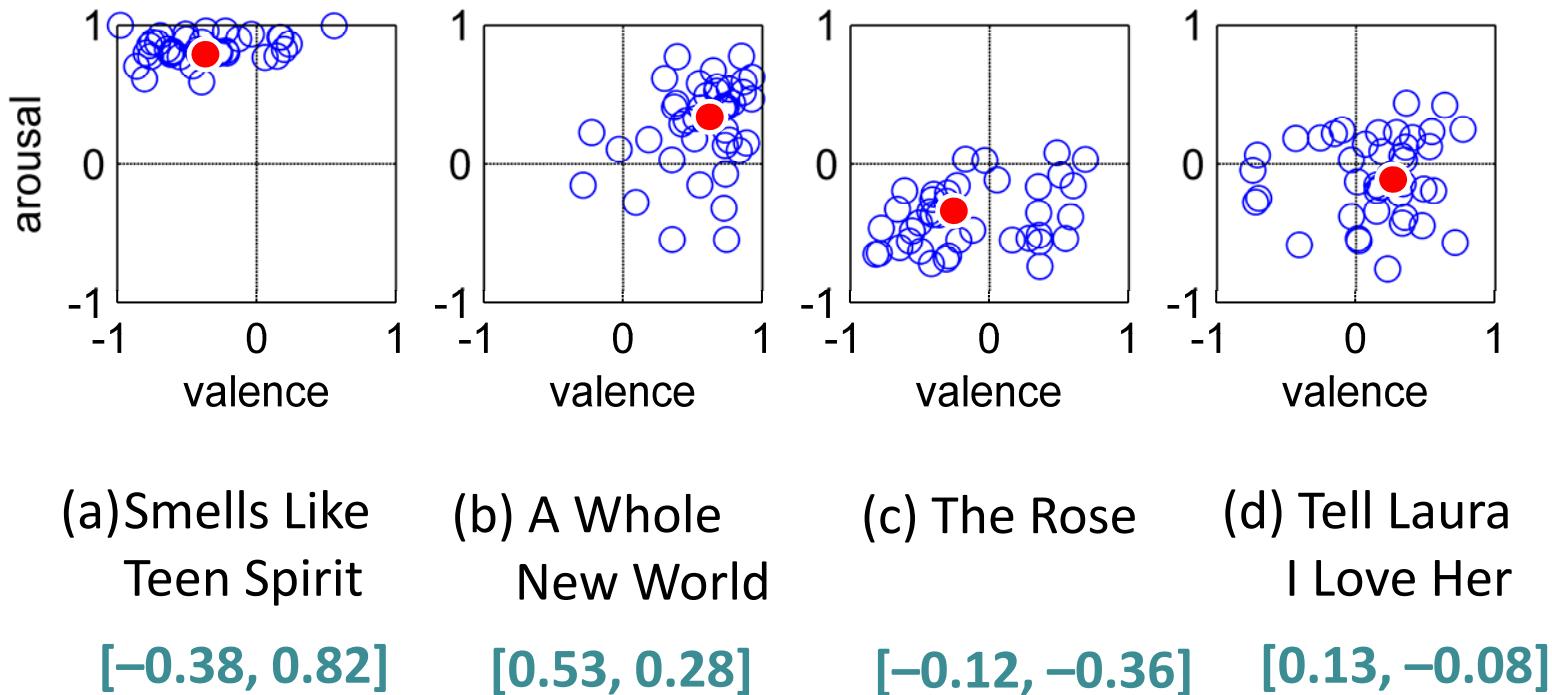
| | user A | user B |
|----|--------|--------|
| 1 | 10 | 10 |
| 2 | 10 | 8 |
| 3 | 10 | 8 |
| 4 | 10 | 10 |
| 5 | 0 | 0 |
| 6 | 0 | 0 |
| 7 | 5 | 10 |
| 8 | 5 | 5 |
| 9 | 0 | 0 |
| 10 | 0 | 2 |
| 11 | 0 | 3 |
| 12 | 0 | 0 |

AnnoEmo: GUI for Emotion Rating

- Encourages differentiation

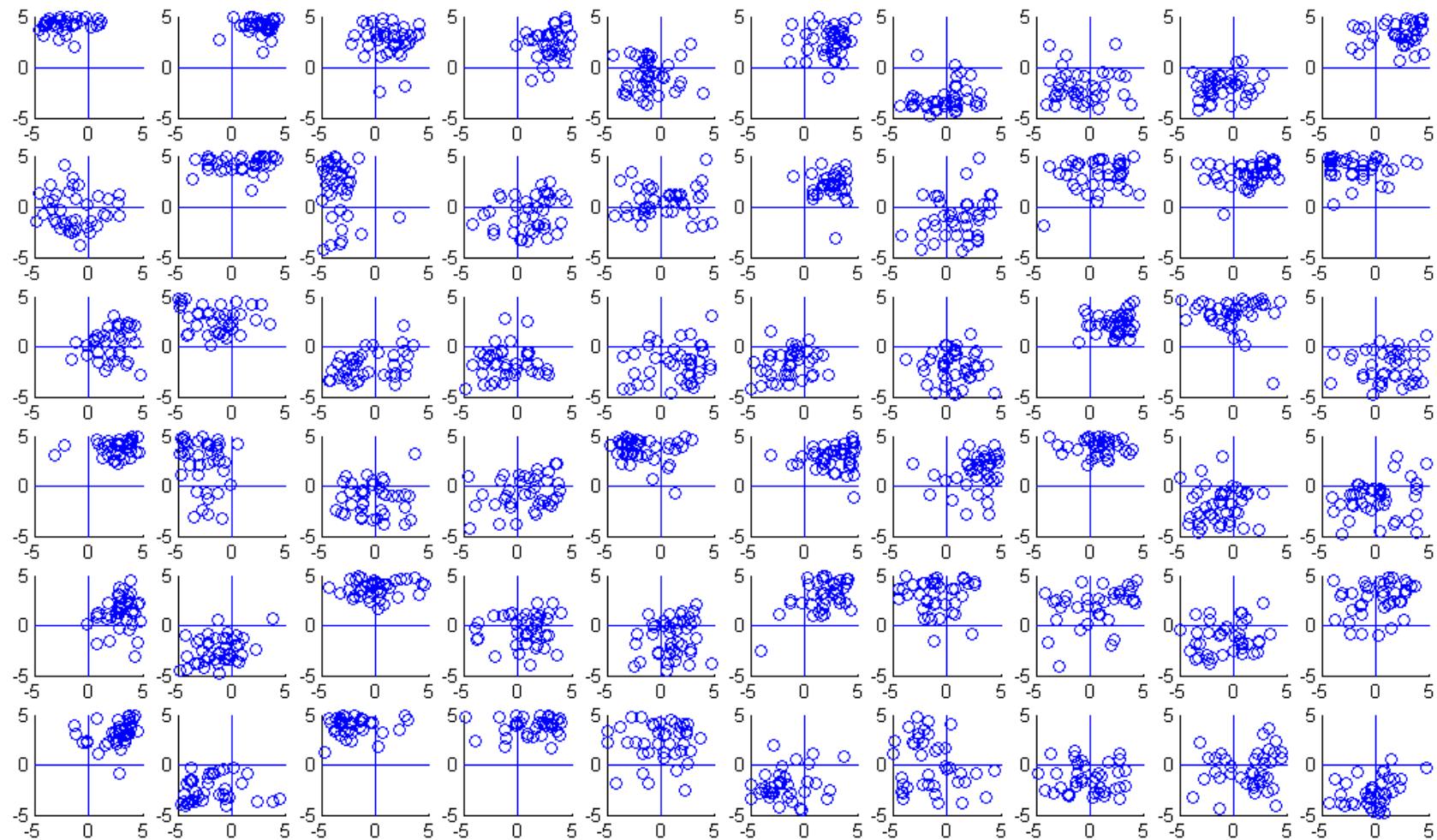


Subjectivity Issue



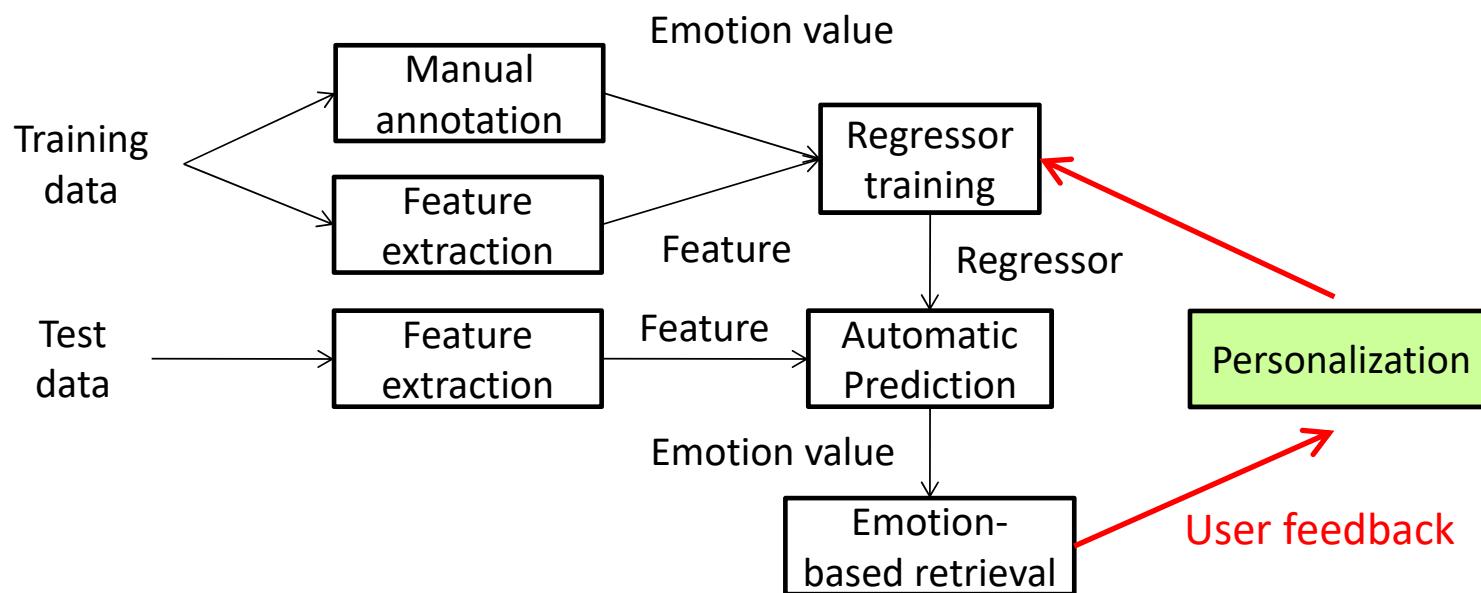
- <https://youtu.be/hTWKbfoikeg?si=f2Wyltz3NIhU9vVB&t=57>
- <https://youtu.be/EXTLJmYsaUQ?si=Xg300FmmdwU7x5x1&t=53>
- https://youtu.be/VTCQBuYhq_s?si=0lUhrxeOLM7ZuHvu&t=52
- <https://youtu.be/pTjQgkHzbTk?si=CciunYjEyLQXtYHH&t=75>

Subjectivity Issue



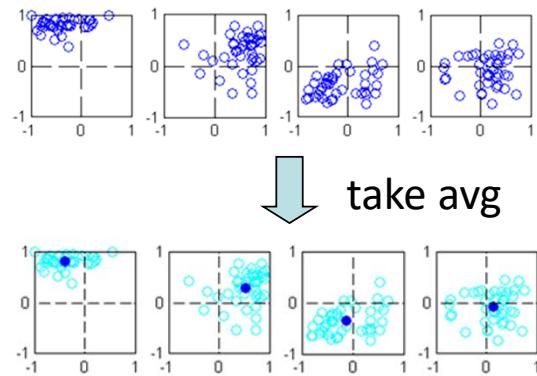
Personalized Music Emotion Recognition

- From $P(e|d)$ to $P(e|d,u)$
 - General regressor → personal regressor
 - Utilize user feedback



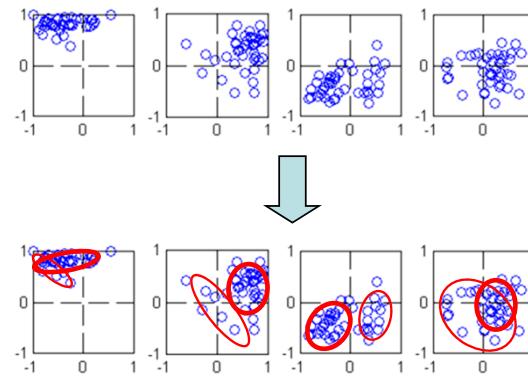
Addressing the Subjectivity Issue

Single point



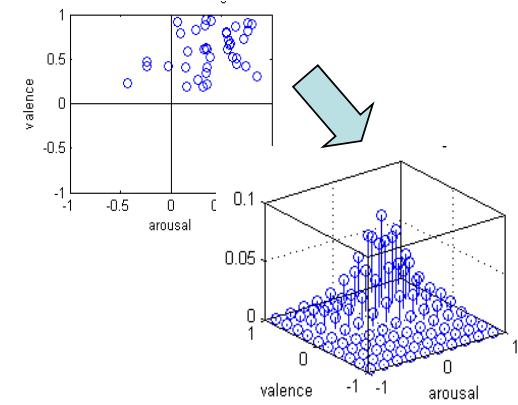
- {valence, arousal}
- Two regressors
- Simply representing a song as a point is not enough

Gaussian mixture



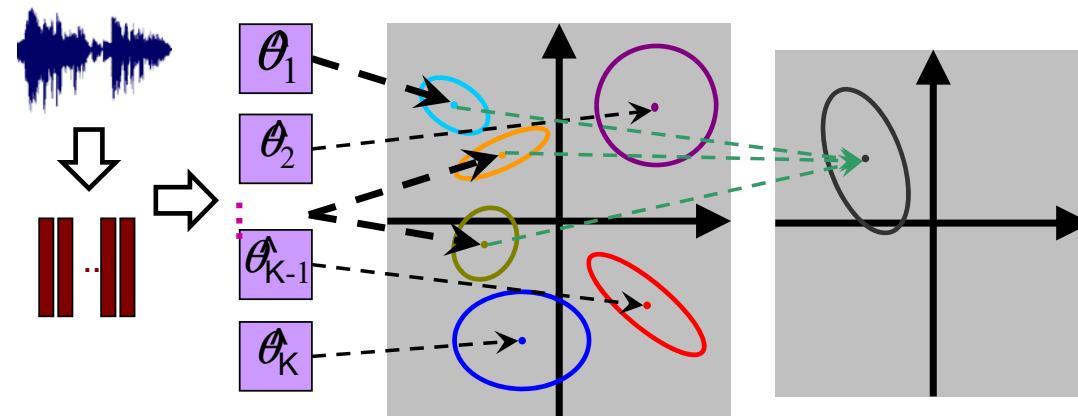
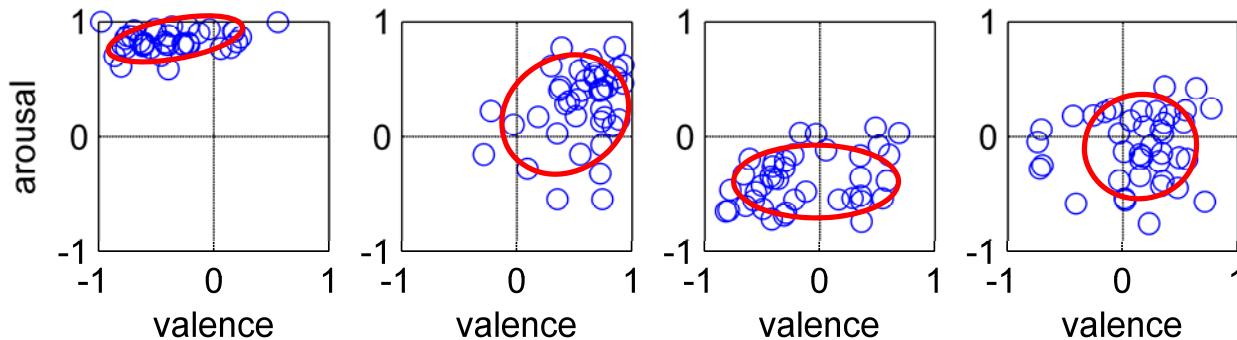
- $\sum w_i N(\mu_i, \sigma_i)$
- Multiple regressors
- Hard to determine the number of components
- May be **ill-posed** (multiple GMMs fit a distribution well)

Probabilistic distribution



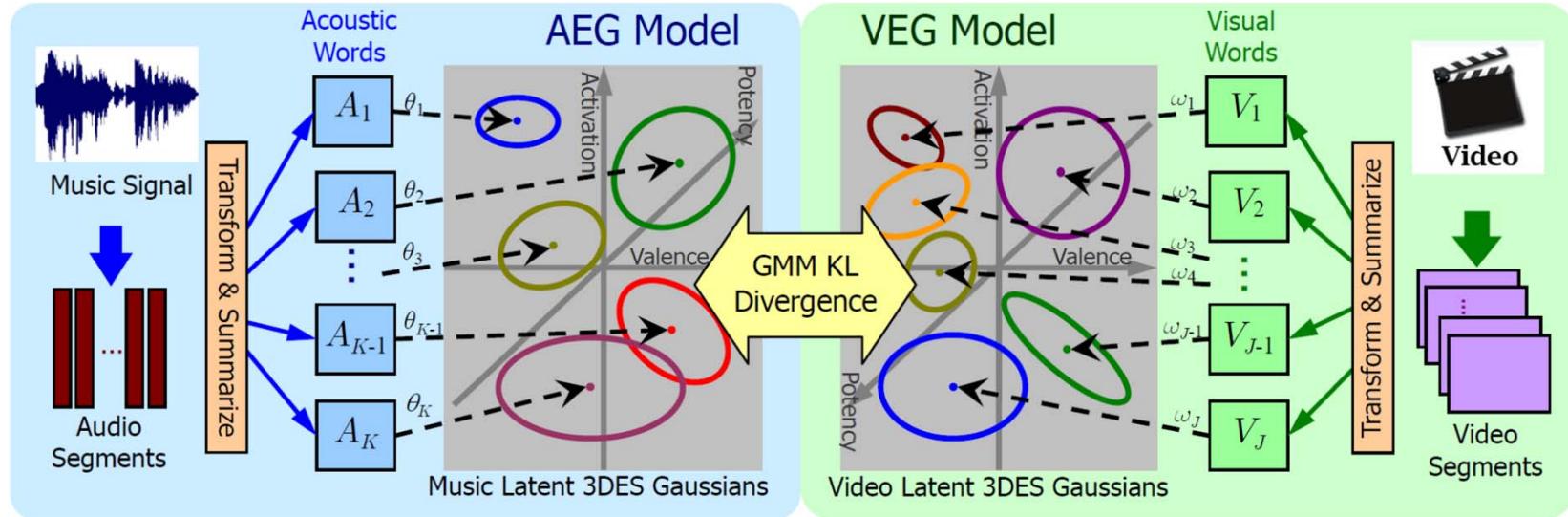
- Predict the *emotion mass* at every point in the emotion plane
- Using a **10x10** grid → 100 regressors
- Predict $P(e^{ij} | d)$

Probabilistic Framework for Music Emotion Recognition



Ref: Wang et al., "The Acoustic Emotion Gaussians model for emotion-based music annotation and retrieval," ACM Multimedia 2012

Emotion-Based MV Composition

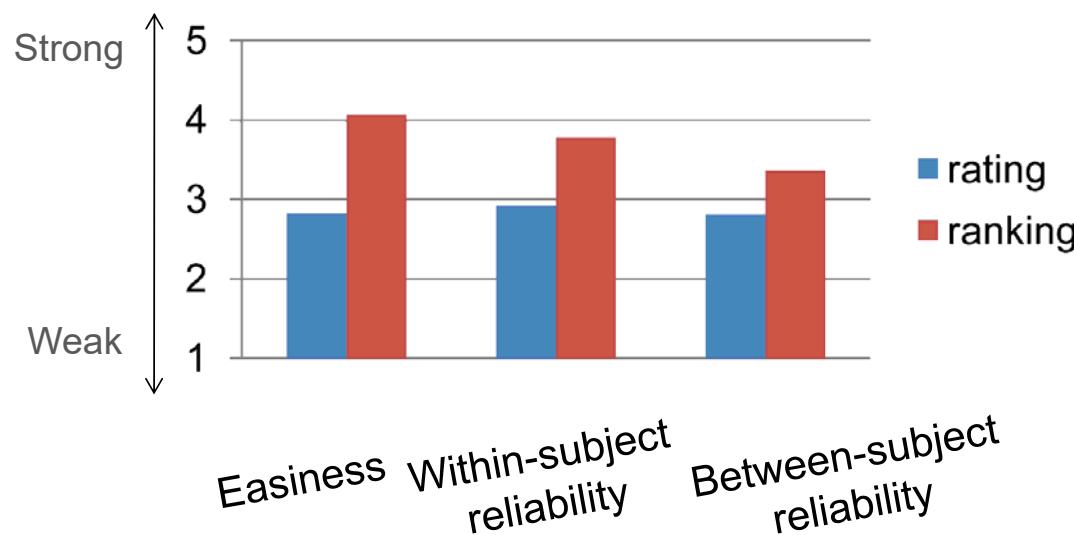
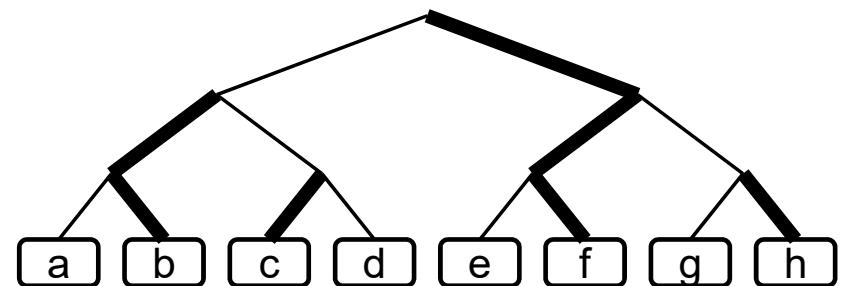


- Audio
 - Sound energy
 - Tempo and beat strength
 - Rhythm regularity
 - Pitch
- Video
 - Lighting key
 - Shot change rate
 - Motion Intensity
 - Color (saturation, color energy)

Ref: Wang et al., “The Acousticvisual Emotion Gaussians model for automatic generation of music video,” ACM MM Grand Challenge 2012 [ACM MM Multimedia Grand Challenge First Prize]

Alternatives: Ranking-Based Emotion Annotation

- Emotion tournament
 - Which song is more positive?
- *Ranking* is easier than *rating*



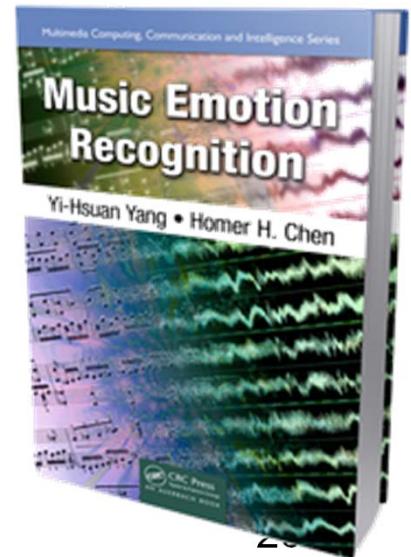
Ranking-Based Emotion Recognition

- Determine the position of a song
 - By the **relative** ranking with respect to other songs (*learning-to-rank*)
 - Rather than by the **exact** emotion values



Music & Emotion (Episode 2: 2006-2010)

- Yang & Chen, “Ranking-based emotion recognition for music organization and retrieval,” *IEEE Transactions on Audio, Speech and Language Processing*, May **2011**
- Yang & Chen, “Prediction of the distribution of perceived music emotions using discrete samples,” *IEEE Transactions on Audio, Speech and Language Processing*, Sept **2011**
- Yang & Chen, *Music Emotion Recognition*, CRC Taylor & Francis Books, Feb **2011**
- Yang & Chen, “Machine recognition of music emotion: A review,” *ACM Transactions on Intelligent Systems and Technology*, May **2012**



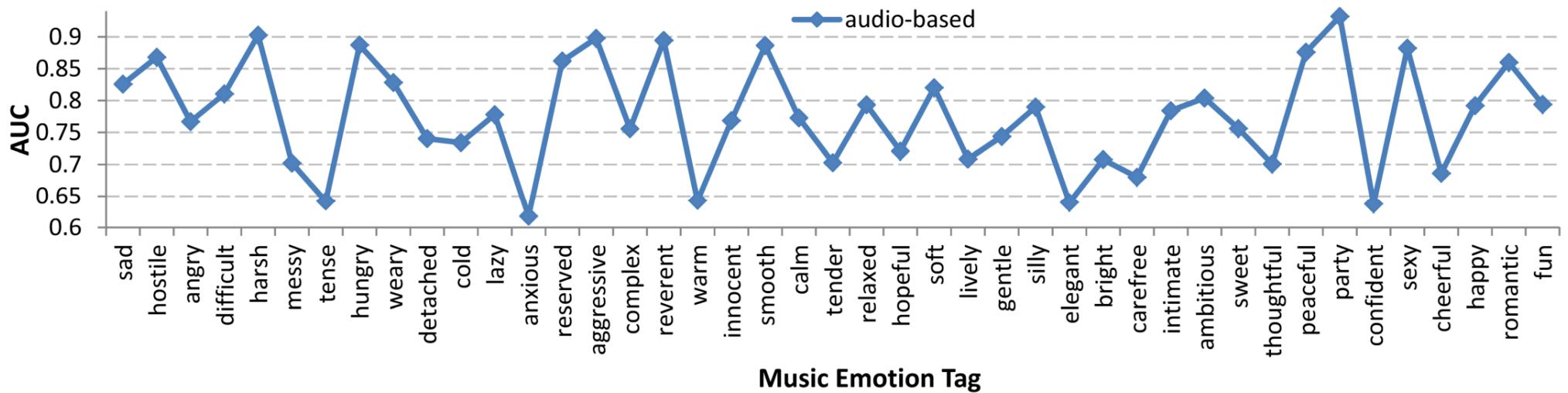
Music & Emotion (Episode 3: 2011—)

- Yang & Liu, “Quantitative study of music listening behavior in a social and affective context,” *IEEE Transactions on Multimedia*, 2013
- Yang & Teng, “Quantitative study of music listening behavior in a smartphone context,” *ACM Transactions on Interactive Intelligent Systems*, 2015

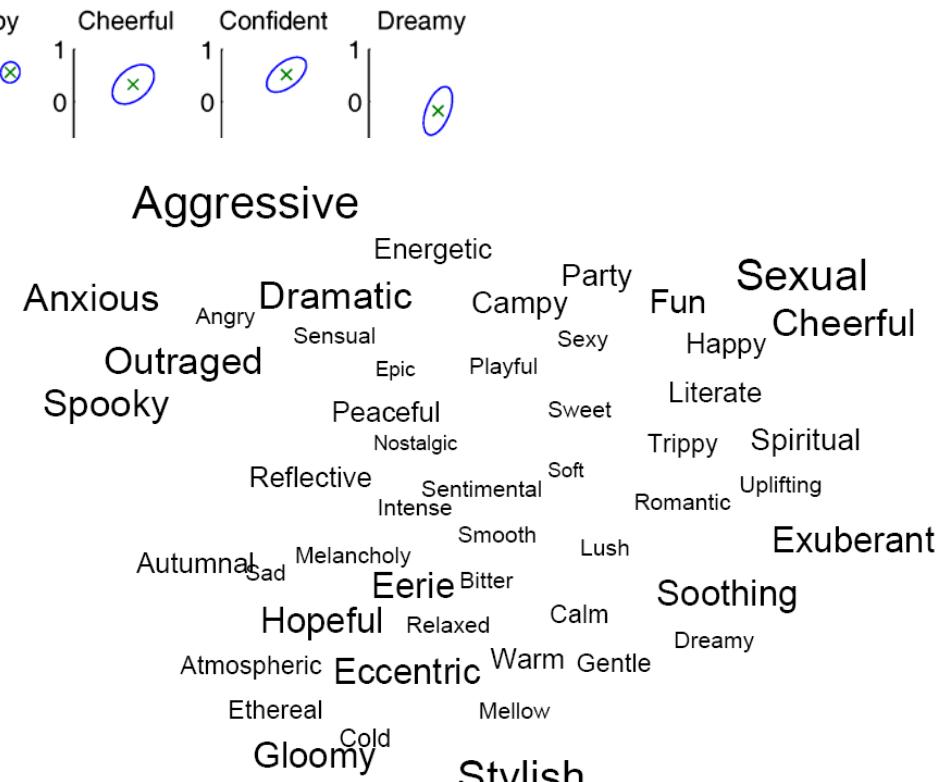
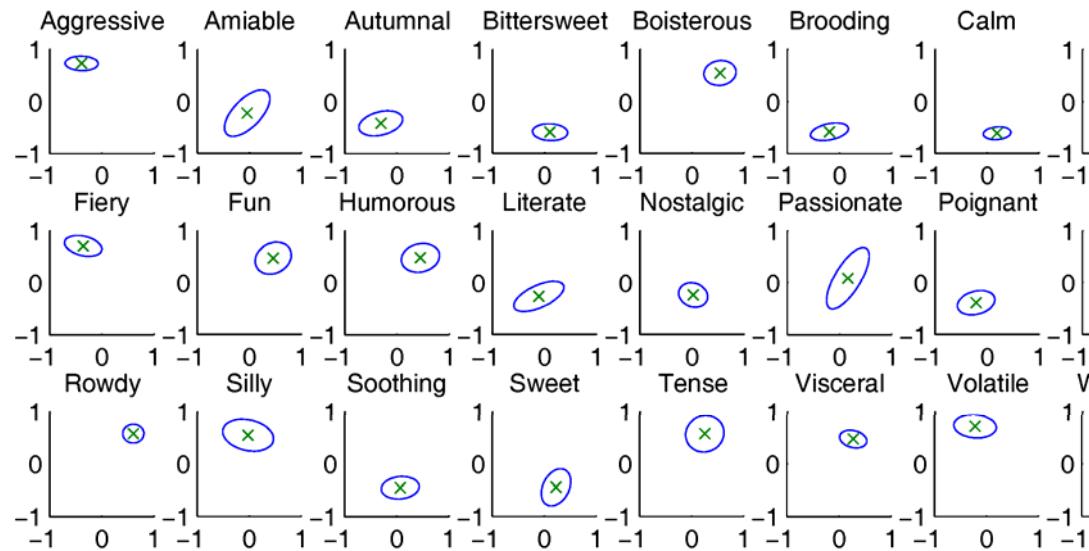
Extension 1: Modeling More Emotion Tags

- Use Allmusic tags (selected by music editors) to consider 190 emotion classes
- Crawl 31k songs from last.fm for training/test

| Modality | Feature | Dim | AUC | NDCG |
|----------|---------|-------------|--------|--------|
| Random | — | — | 0.4999 | 0.2806 |
| Music | Audio | Energy | 24 | 0.6969 |
| | | Rhythm | 5 | 0.6229 |
| | | Timbre | 176 | 0.7432 |
| | | Tonal | 38 | 0.7060 |
| | | Late fusion | — | 0.7614 |



Mapping Mood Tags to the 2D Emotion Space

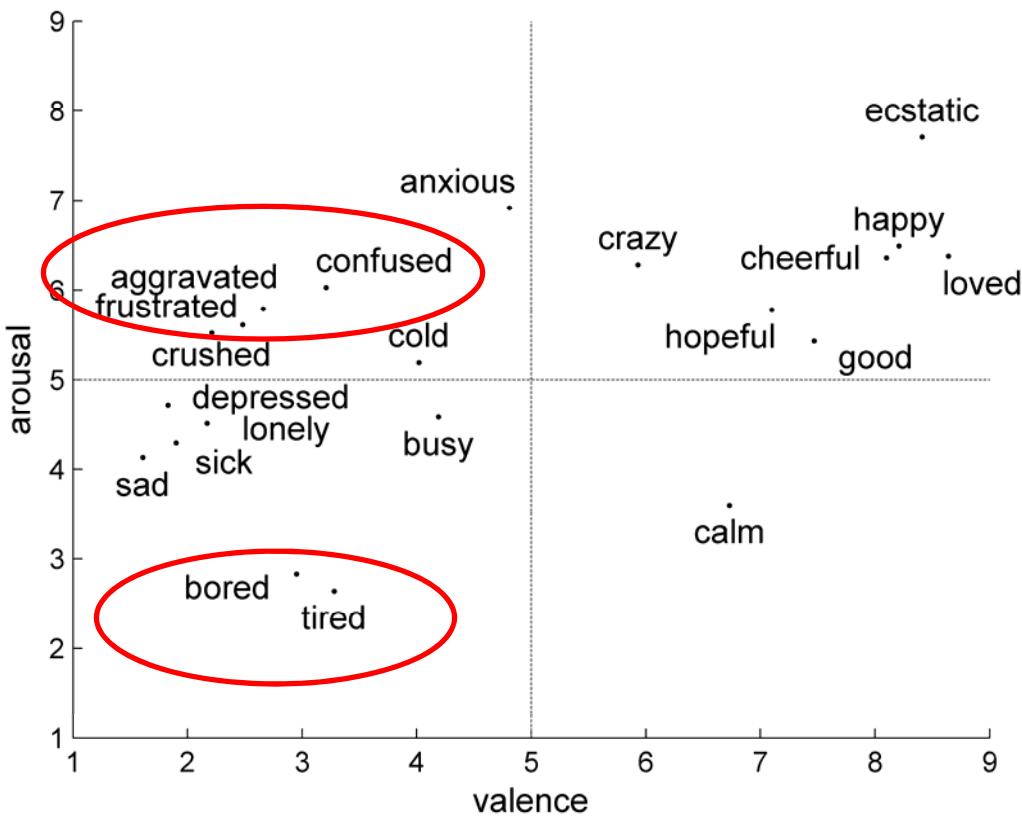


- Data driven: the mapping is directly learned from data with audio feature

Ref: Wang et al., "Exploring the relationship between categorical and dimensional emotion semantics of music," ACM MM Workshop 2012

Extension 2: User Mood vs. Music Emotion

- Everyday emotion vs. aesthetic emotion



LiveJournal: A Large-Scale, In-Situ Dataset

hey friends, its been awhile since I last updated but my weekend was really good at the retreat and then monday was a blaze..soooo tired and yesterday I went and bought my Halloween Costume with Jelinek yay haha im going to be a fairy its basically pink!!! my favorite color..im happy haha..gotta be ready for our Halloween Dance put on by yearbook..yeah u better be there ALL OF U!!! haha well cyas later

Current Mood:  happy

Current Music: Something Corporate- Down

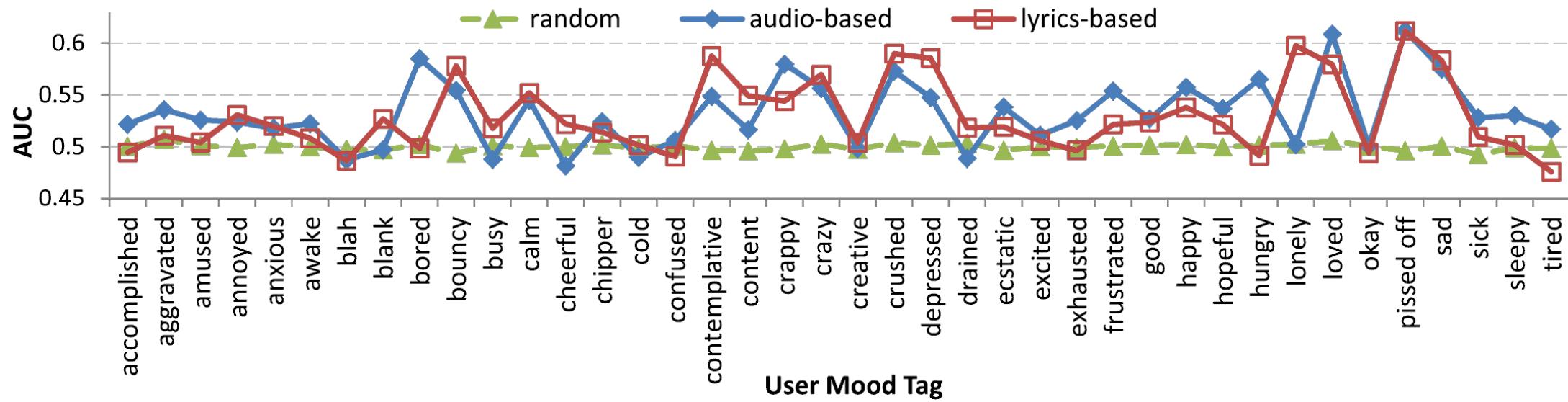
welll my internet is still not working..soo its been awhile since i updated..im sry but nothing hasnt been life while to write about..but now there is...looks like some things dont always go well and some people know how to make ppl feel like crap..hmm..way to go asshole

Current Mood:  angry

Current Music: goo goo dolls- black balloon

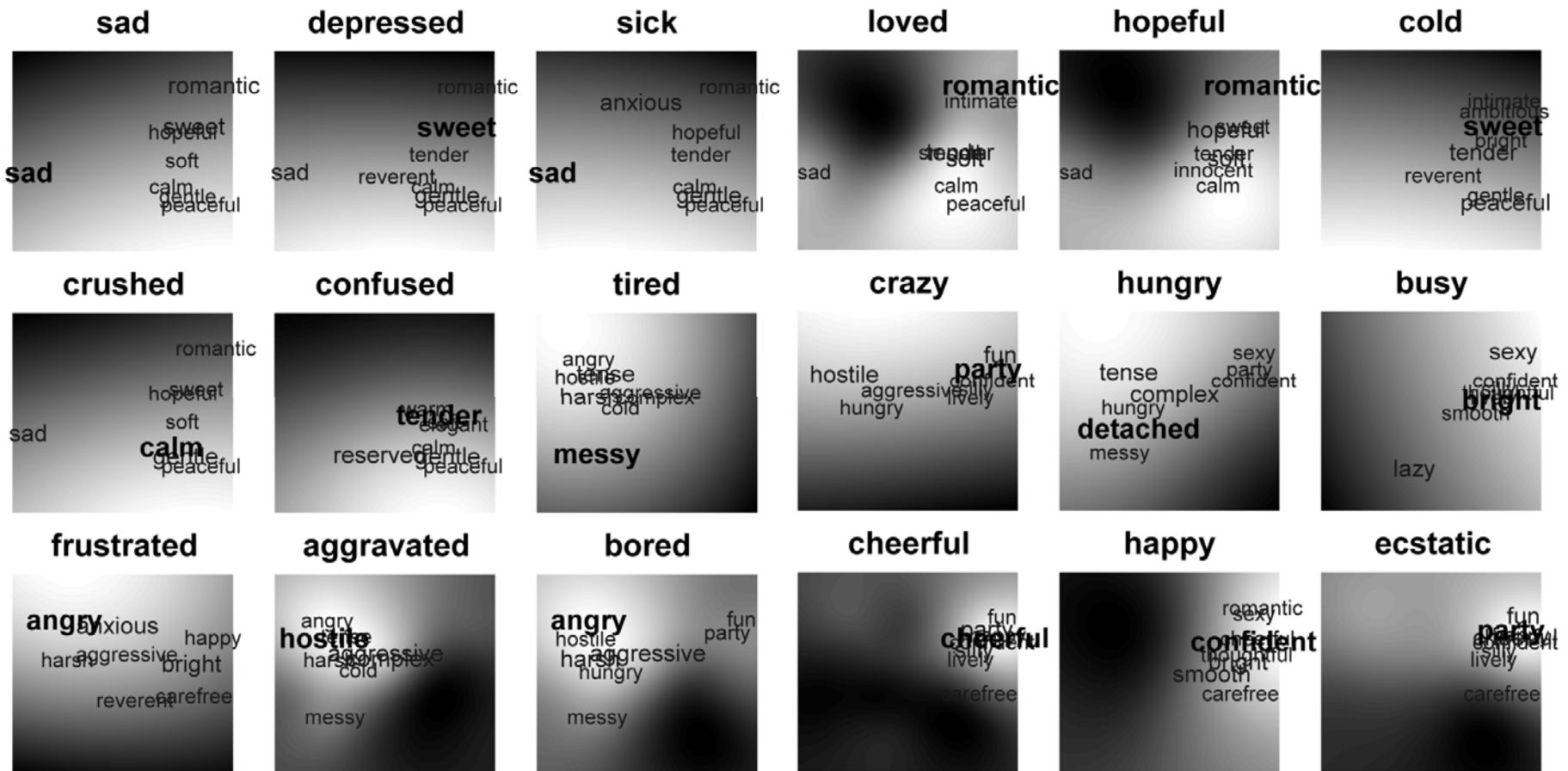
- In addition to blog writing, users
 - enter an emotion tag ([user mood](#))
 - enter a song title & artist name ([music emotion](#))

Predicting User's Emotional State from the Music Listening Behavior



- Audio-based model perform better for “aggravated,” “awake,” “contemplative,” “drained,” and “tired”
- Lyrics-based model is good for “crazy,” “creative,” “crushed,” and “pissed-off”
- Both are poor for “blah,” “blank,” and “okay”

Mood-Congruent or -Incongruent



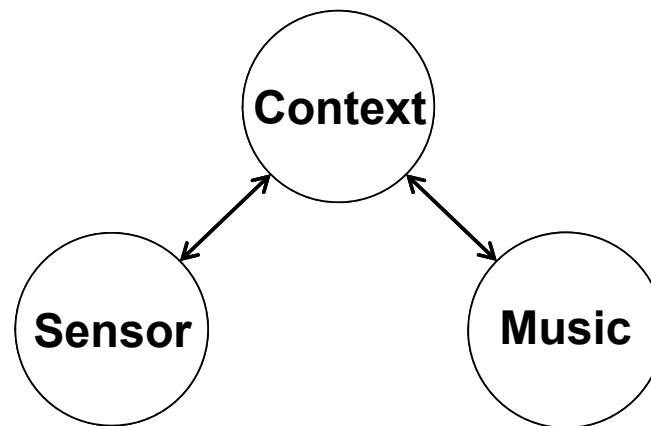
Yang & Liu, "Quantitative study of music listening behavior in a social and affective context," IEEE Transactions on Multimedia, 2013

The Role of Personality

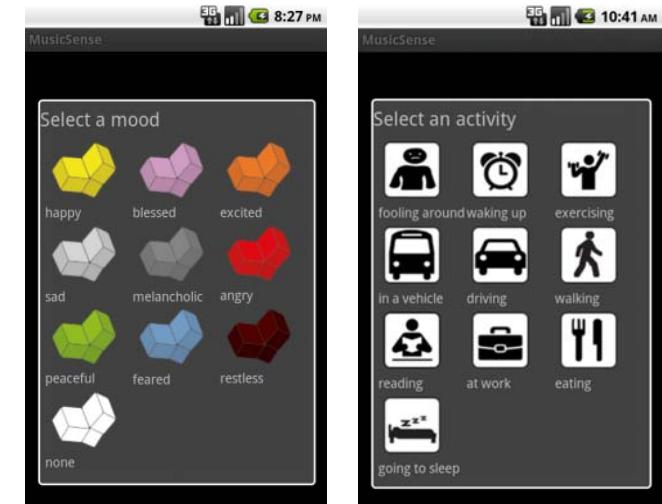
- Estimate from writing styles (linguistic features) by *Personality Recognizer* developed by MIT
- Use two-sample, one-tailed t-test for testing hypothesis

| Personality | User mood | Music emotion |
|----------------------|-----------|---------------------|
| High in Extraversion | loved | more party music |
| High in Extraversion | sick | more party music |
| Low in Extraversion | anxious | less angry music |
| Low in Extraversion | awake | less angry music |
| Low in Extraversion | blank | less party music |
| Low in Extraversion | bored | less angry music |
| Low in Extraversion | bored | more peaceful music |
| Low in Extraversion | cheerful | less angry music |
| Low in Extraversion | drained | less angry music |
| Low in Extraversion | drained | more peaceful music |

Understanding Users: from Smartphones



| | |
|---------------|--------------|
| Accelerometer | Microphone |
| Ambient light | Proximity |
| Compass | Running apps |
| Dual cameras | Time |
| GPS | Wifi |
| Gyroscope | |

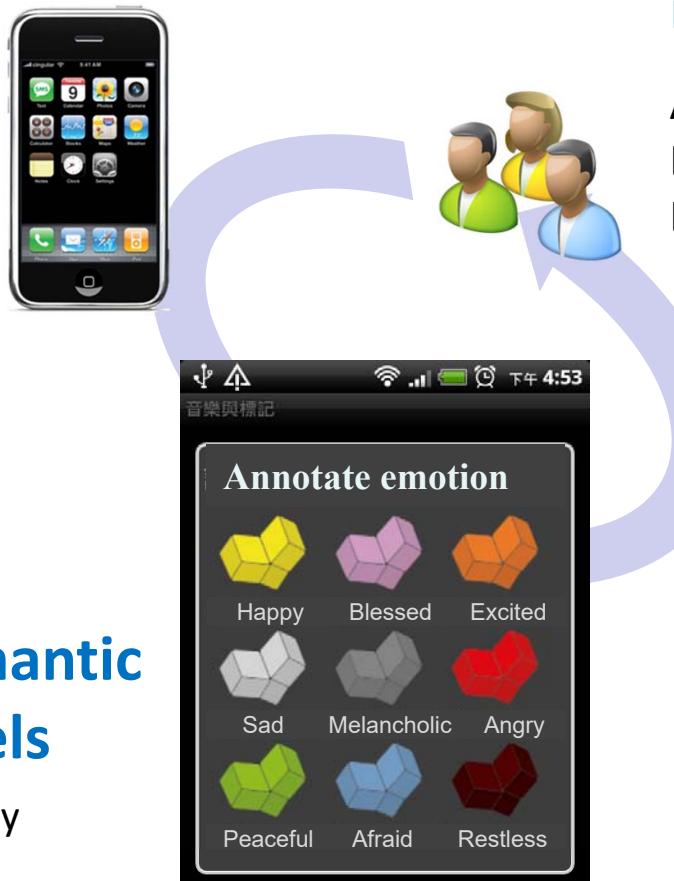


Yang & Teng, "Quantitative study of music listening behavior in a smartphone context,"
ACM Transactions on Interactive Intelligent Systems, 2015

Affect-Based Music Recommendation

Sensor data

Accelerometer
Ambient light
Ambient sound
Compass
Dual cameras
GPS
Gyroscope
Microphone
Proximity
Running apps
Time
Wifi



Semantic labels

Activity
Mood
Location

User data

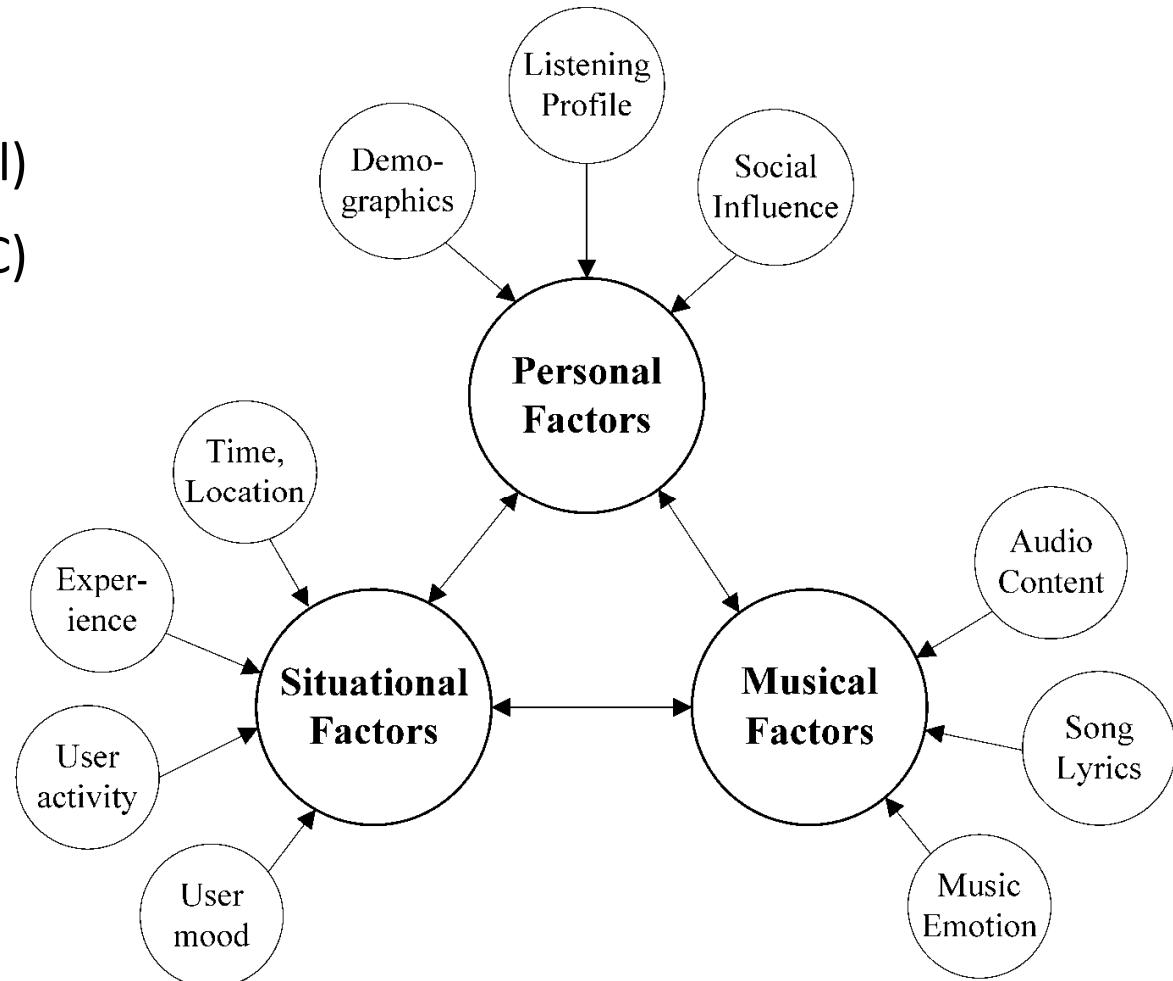
Age 22, Male, Live in Taipei
Easy-going, inventive, friendly
Love Classic songs; can play piano

Music listening profile

2012/7/25 15:02:23 song A
2012/7/25 15:20:08 song B
2012/7/26 08:42:30 song C
2012/7/26 20:11:07 song D
2012/7/27 08:35:51 song C

Personal, Situational, and Musical Factors

Blogs (LiveJournal)
Smartphones (HTC)



Context of Music Listening

Experienced emotion = structural features \times performance features \times listener features \times context features

Where

Structural features = β_1 (segmental features) \times β_2 (suprasegmental features)

Performance features = β_3 (performer skills) \times β_4 (performer state)

Listener features = β_5 (musical expertise) \times β_6 (stable dispositions) \times β_7 (mood state)

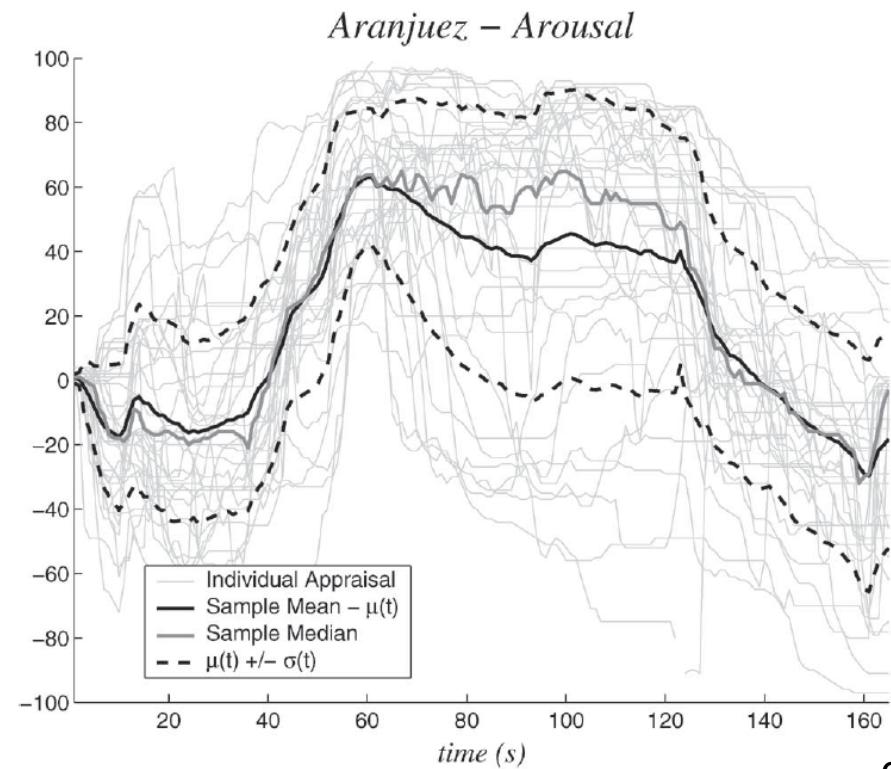
Context features = β_8 (location) \times β_9 (sound acoustics) \times β_{10} (event)

Adapted from Scherer and Zentner (2001). β represents a variable.

- Listening mood/context
- Familiarity/associated memory
- Preference of the singer/performer/song
- Social relationship

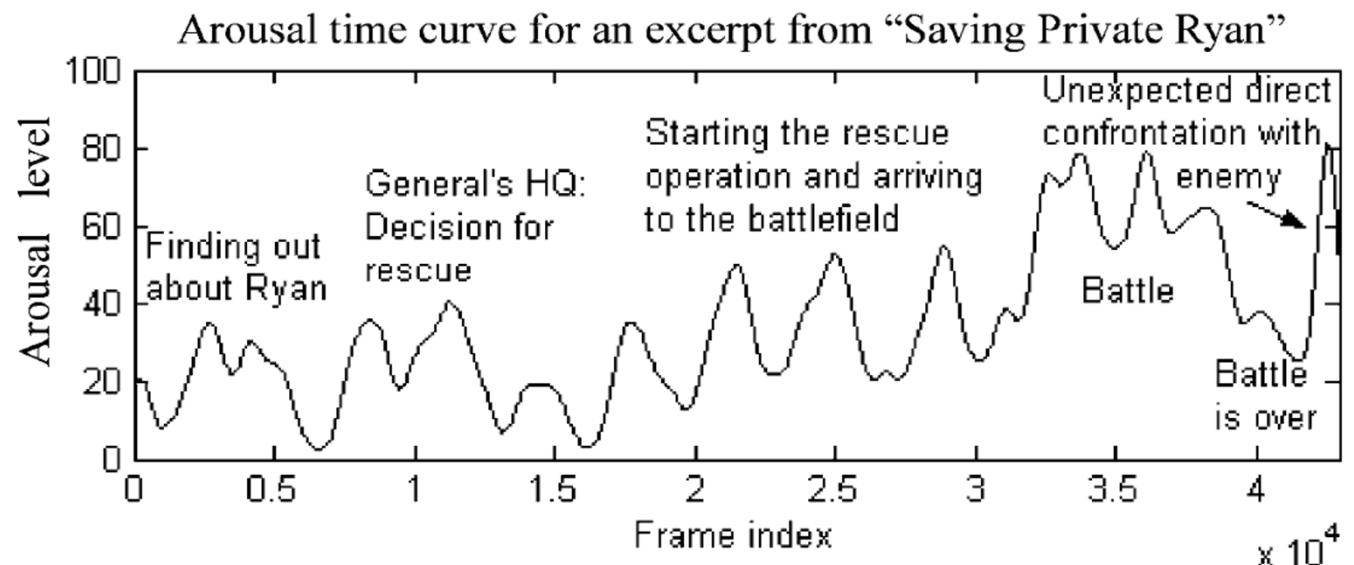
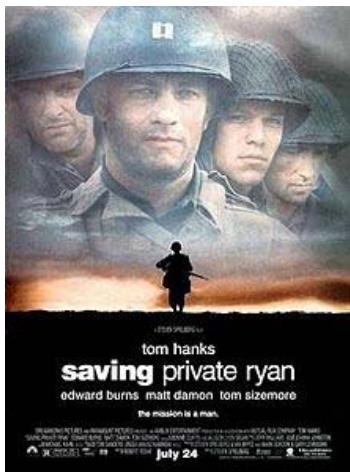
Less Studied: Temporal Context of Music

- “Sweet anticipation”
 - Music’s most expressive qualities probably relate to structural changes across time
- Music emotion can also vary within an excerpt



Less Studied: Temporal Context of Music

- Application to Video content understanding

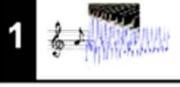
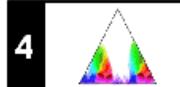
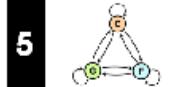
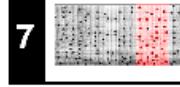
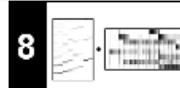


Outline

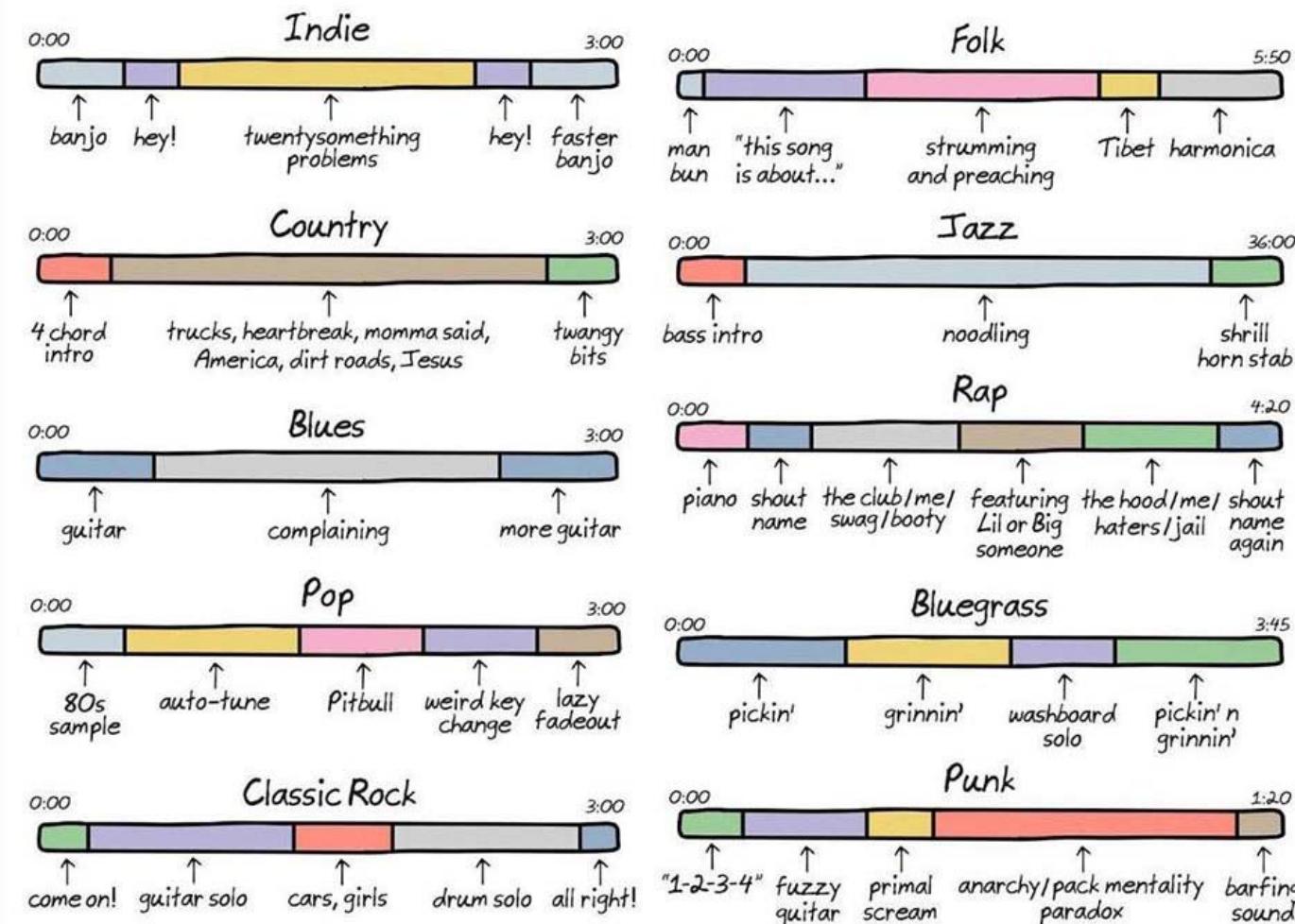
- Music emotion
- **Structure/form analysis**
- Alignment
- Rhythm & beat tracking

Reference: FMP Notebook

<https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4.html>

| Part | Title | Notions, Techniques & Algorithms | HTML | IPYNB |
|------|--|--|------------------------|-------------------------|
| B |  Basics | Basic information on Python, Jupyter notebooks, Anaconda package management system, Python environments, visualizations, and other topics | [html] | [ipynb] |
| 0 |  Overview | Overview of the notebooks (https://www.audiolabs-erlangen.de/FMP) | [html] | [ipynb] |
| 1 |  Music Representations | Music notation, MIDI, audio signal, waveform, pitch, loudness, timbre | [html] | [ipynb] |
| 2 |  Fourier Analysis of Signals | Discrete/analog signal, sinusoid, exponential, Fourier transform, Fourier representation, DFT, FFT, STFT | [html] | [ipynb] |
| 3 |  Music Synchronization | Chroma feature, dynamic programming, dynamic time warping (DTW), alignment, user interface | [html] | [ipynb] |
| 4 |  Music Structure Analysis | Similarity matrix, repetition, thumbnail, homogeneity, novelty, evaluation, precision, recall, F-measure, visualization, scape plot | [html] | [ipynb] |
| 5 |  Chord Recognition | Harmony, music theory, chords, scales, templates, hidden Markov model (HMM), evaluation | [html] | [ipynb] |
| 6 |  Tempo and Beat Tracking | Onset, novelty, tempo, tempogram, beat, periodicity, Fourier analysis, autocorrelation | [html] | [ipynb] |
| 7 |  Content-Based Audio Retrieval | Identification, fingerprint, indexing, inverted list, matching, version, cover song | [html] | [ipynb] |
| 8 |  Musically Informed Audio Decomposition | Harmonic/percussive separation, signal reconstruction, instantaneous frequency, fundamental frequency (F0), trajectory, nonnegative matrix factorization (NMF) | [html] | [ipynb] |

Music Structure / Form



(Image from the Internet)

Hungarian Dance No. 5 by Johannes Brahms (Ormandy)

MIR tasks

1. Alignment
 2. Transcription
 3. Instrument recognition
 4. Tempo
estimation
 5. Key detection
 6. Dynamics
 7. Emotion

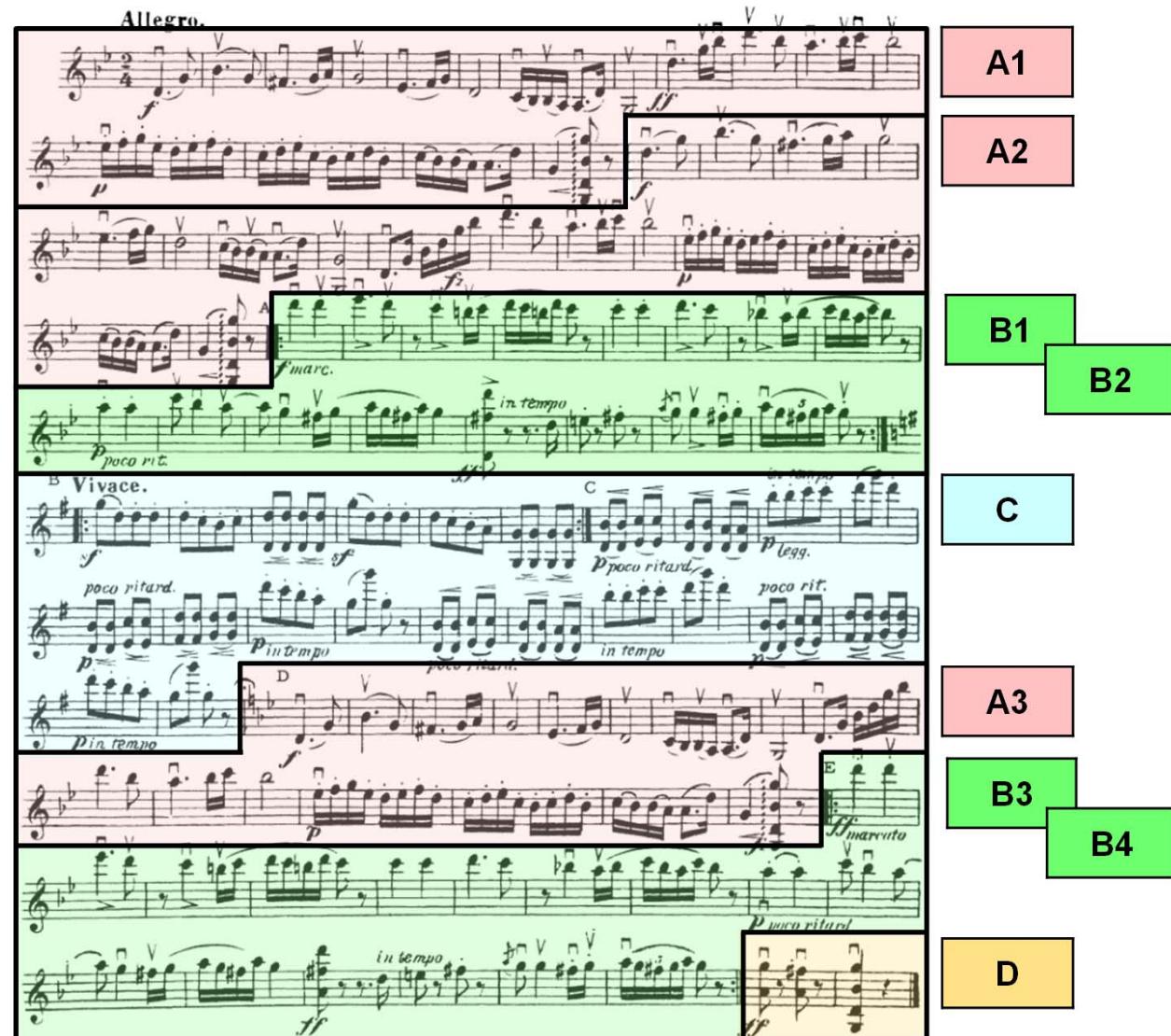


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Hungarian Dance No. 5 by Johannes Brahms (Ormandy)

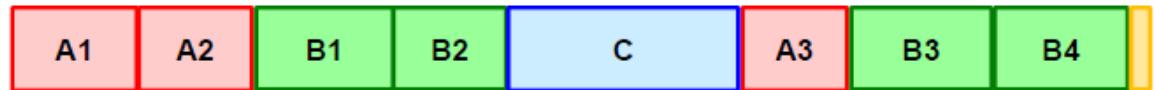
8. Structure

<https://www.youtube.com/watch?v=QAMxkietiik>



Music Structure Analysis

- General goal
 1. **Segmentation:** Divide an audio recording into temporal segments corresponding to musical parts
 2. **Grouping:** Cluster or label these segments into musically meaningful categories
- Examples
 - *Intro, verse, chorus, bridge, outro* of a pop song
 - *Exposition, development, recapitulation, coda* of a sonata



Demonstration: Songle

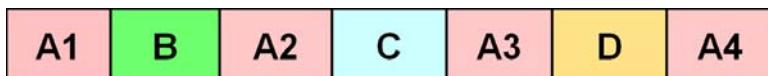
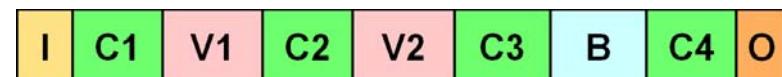
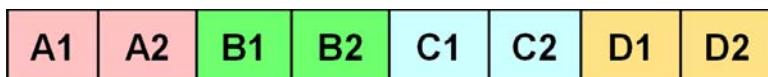
<https://songle.jp/>

The screenshot shows the Songle website interface. At the top, there is a red header bar with the Songle logo, navigation links for "Songs" and "Ranking", and buttons for "How to use" and "Login". A search bar is also present. Below the header, the title of the song is displayed: "【初音ミク】 PROLOGUE 【ばかりす+ぼかうお】 by VocaListener (AIST)".

Below the title, there are statistics: 18909 views, 8385 likes, and 5 comments. There is also a "Tweet" button.

The main content area features a musical visualization. It includes a timeline at the top showing playback progress at 00:40 / 04:56. Below the timeline are three horizontal tracks: a blue track with vertical bars, a red track with vertical bars, and a yellow track with vertical bars. The bottom track has colored boxes indicating chords: #m, F#m, Abm, C#m, DM7, Ab, A, B, Ab/C, C#m, F#/A#, F#m7, G#m, B, F, E, D, B, E. The entire visualization is set against a grid background.

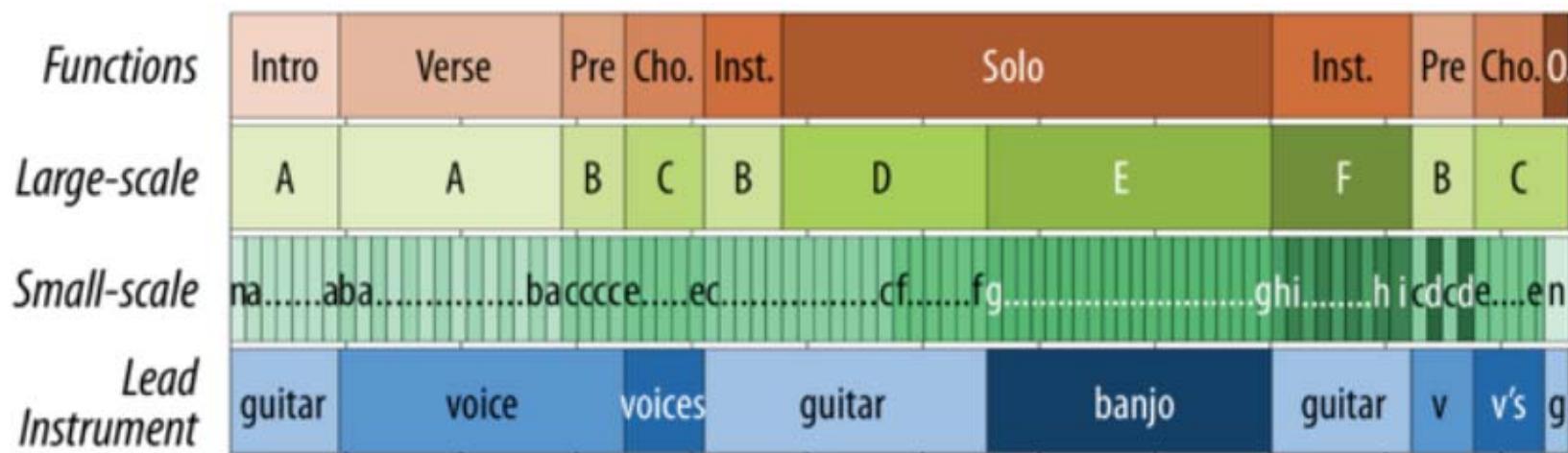
Music Structure / Form



- Strophic form
- Chain form
- Rondo form
- Sonata form
- “Tell Me Why” by Beatles
- “Yesterday” by Beatles

Ambiguity in Structure Analysis

- “Musical structure naturally exhibits a **hierarchical** organization where a variety of cues can trigger boundaries between segments of different length, depending on the time scale at which they are observed”



Ref 1: Buisson et al, “Learning multi-level representations for hierarchical music structure analysis,” ISMIR 2022

Ref 2: Smith et al, “Design and creation of a large-scale database of structural annotations,” ISMIR 2011

General Principles of Music Structure Analysis

1) Novelty, 2) Homogeneity, 3) Repetition

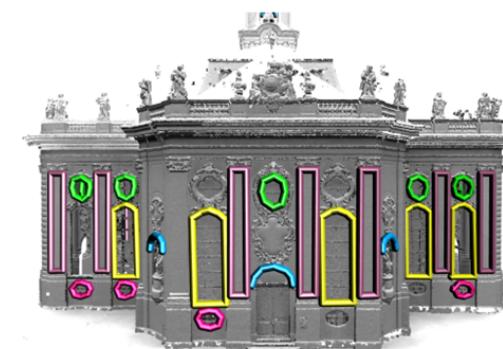
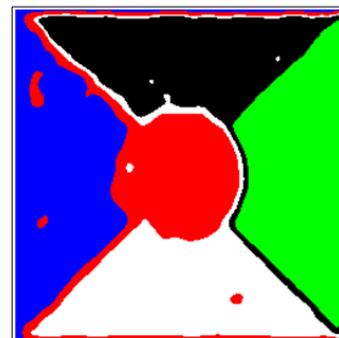
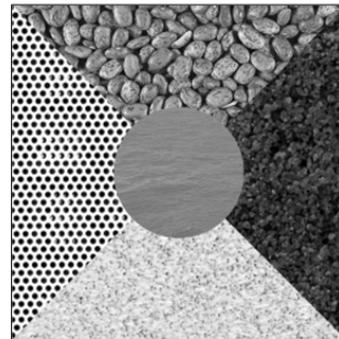
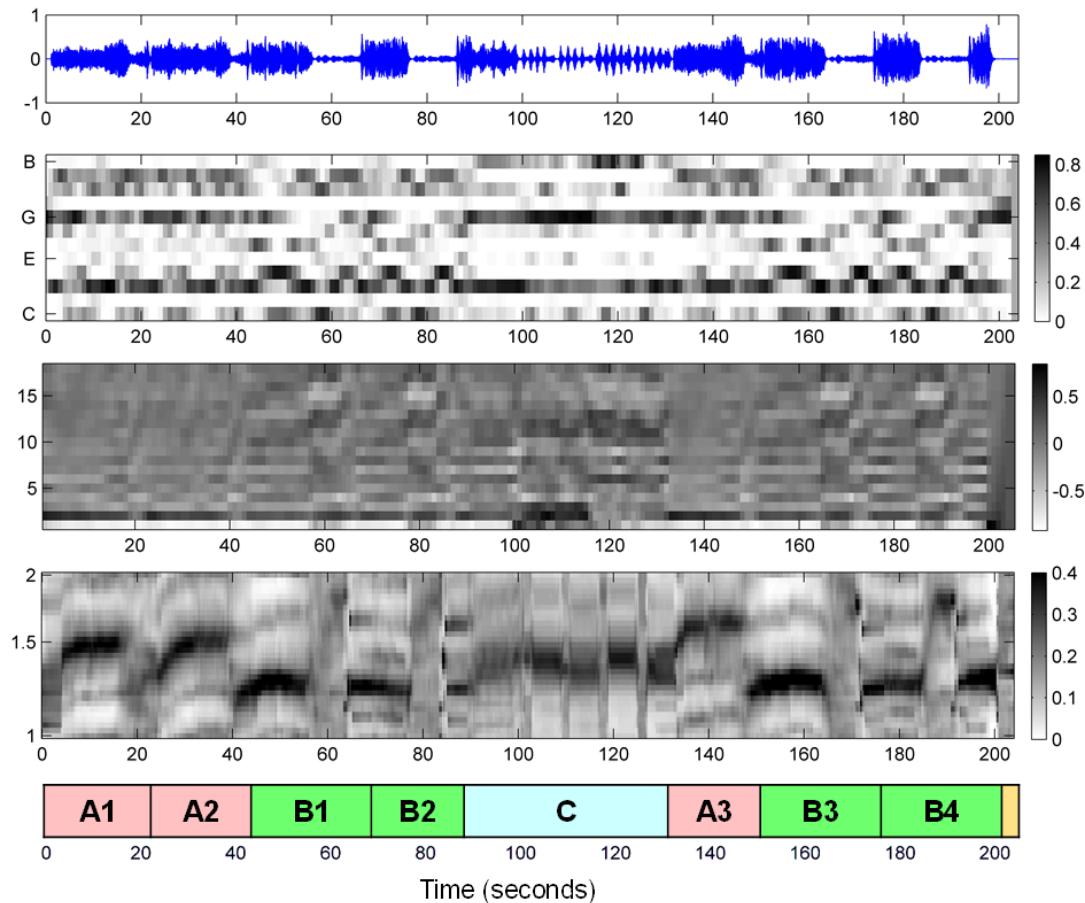


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Acoustic Features Cues

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S1_MusicStructureGeneral.html

- Wav
- Chromagram
- MFCC
- Tempogram
- Groundtruth



Feature Representation

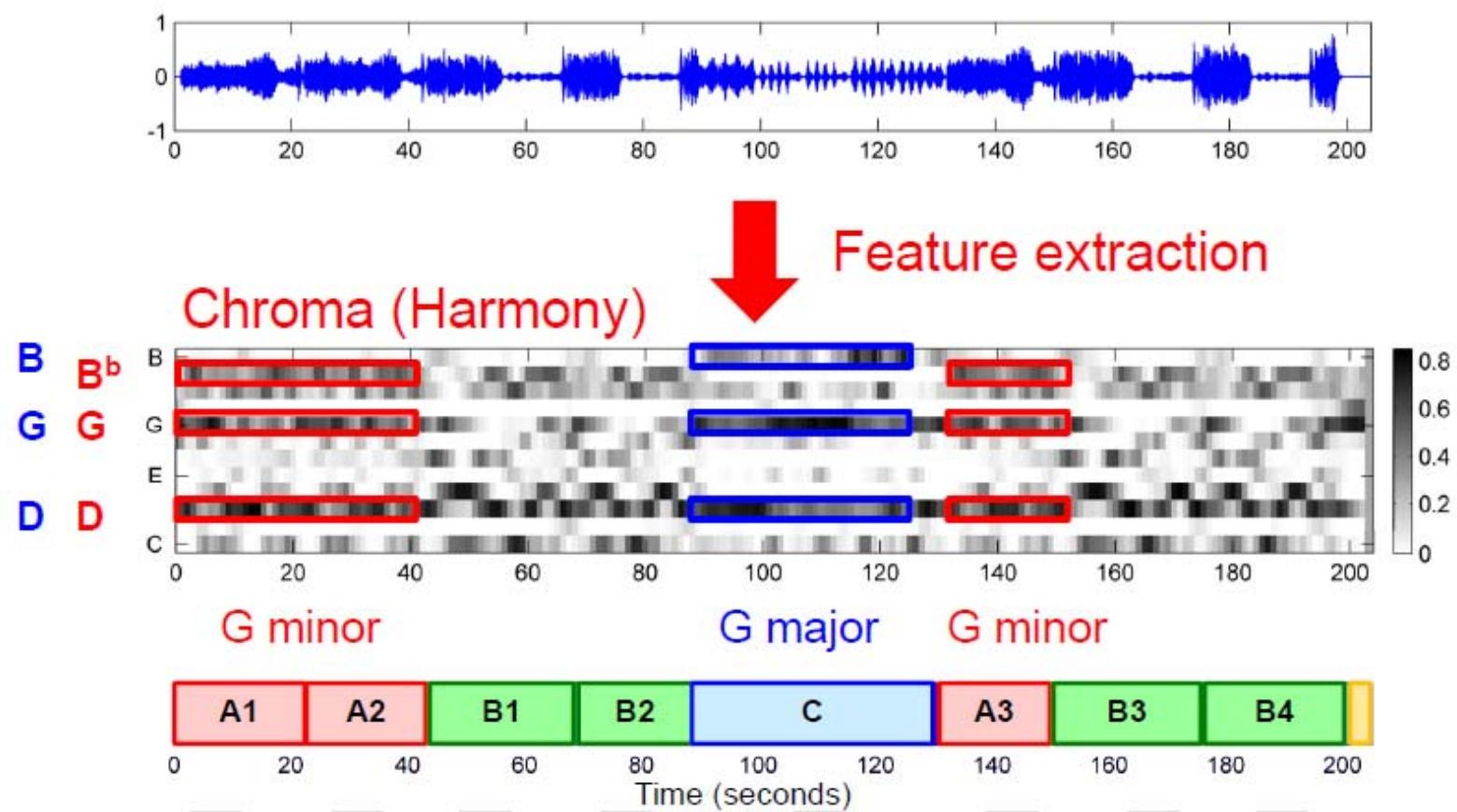


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Self-Similarity Matrices (SSM)

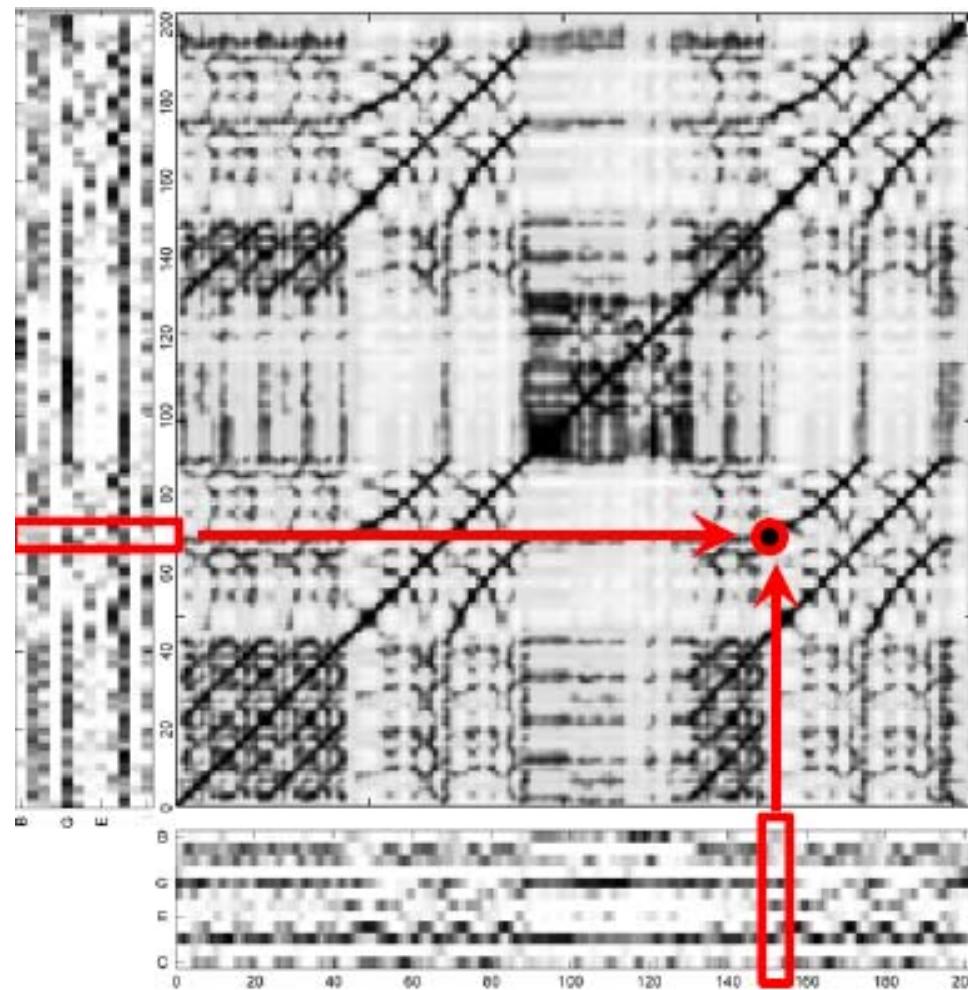


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Blocks → Homogeneity

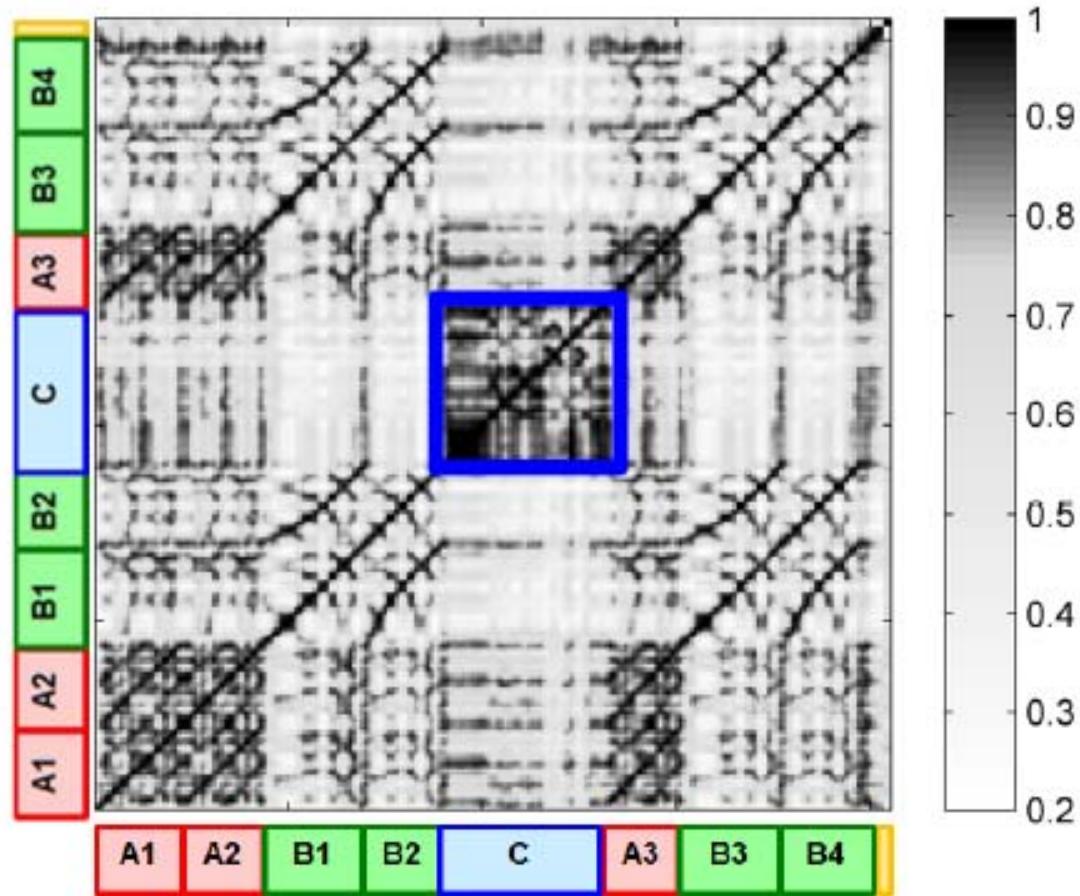


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Paths → Repetition

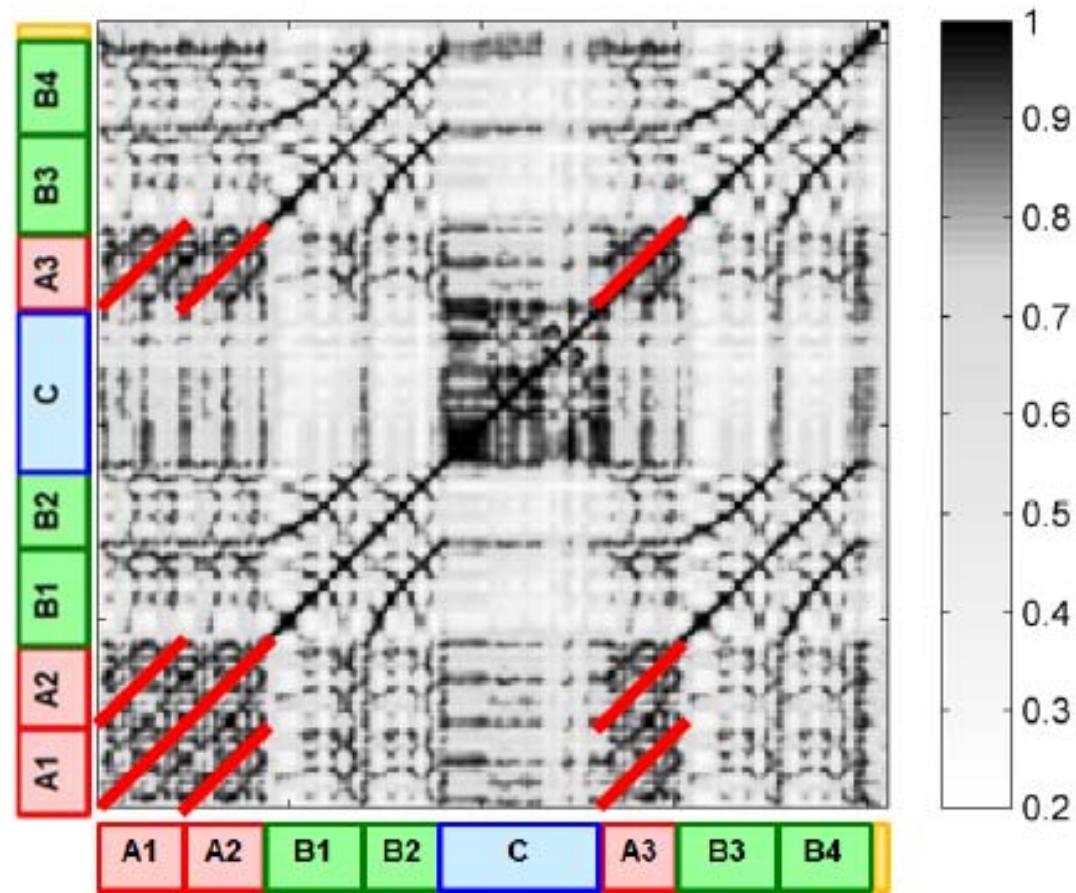


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Paths → Repetition

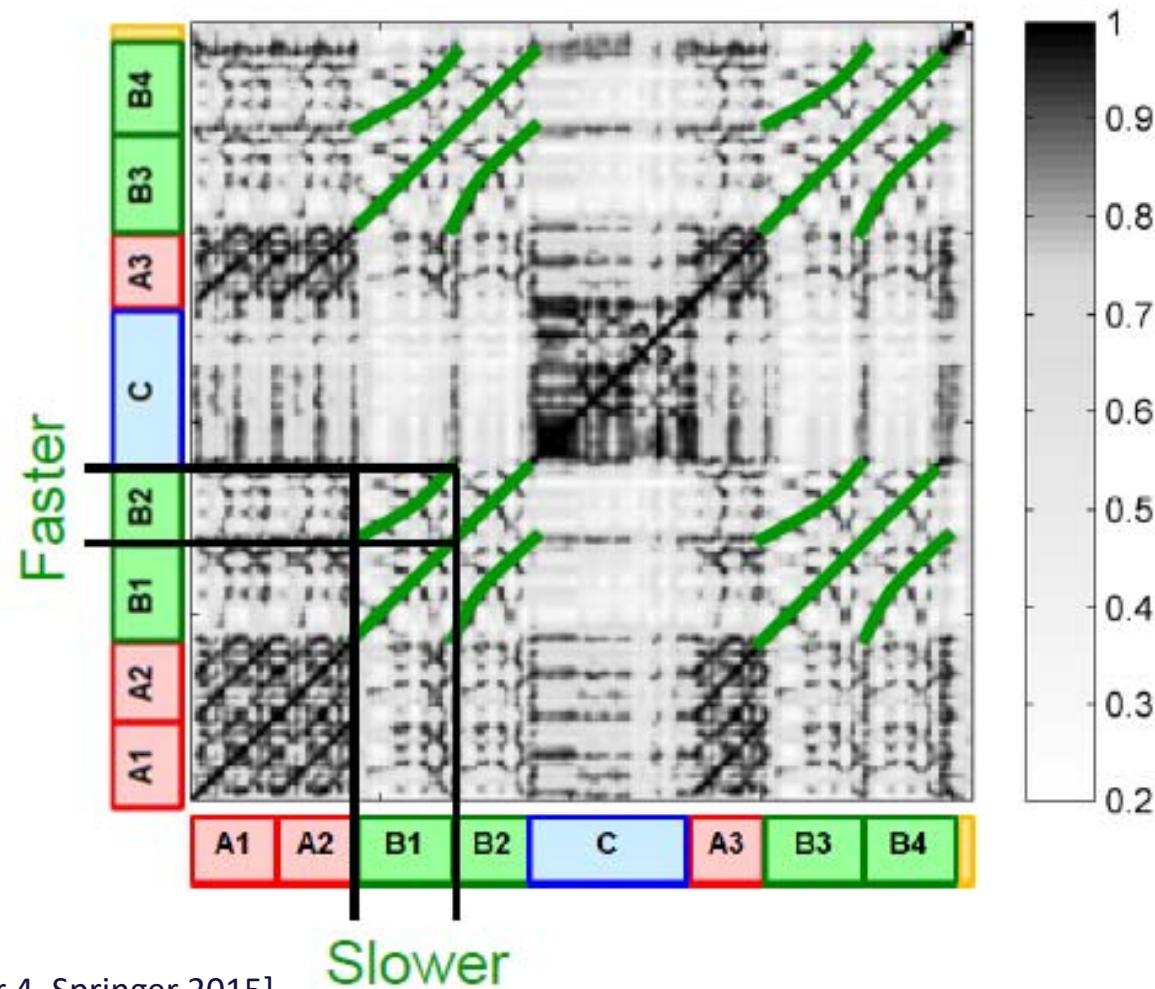


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Self-Similarity Matrices (SSM)

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S2_SSM.html

Blocks: Homogeneity

Paths: Repetition

Corners: Novelty

Idealized SSM

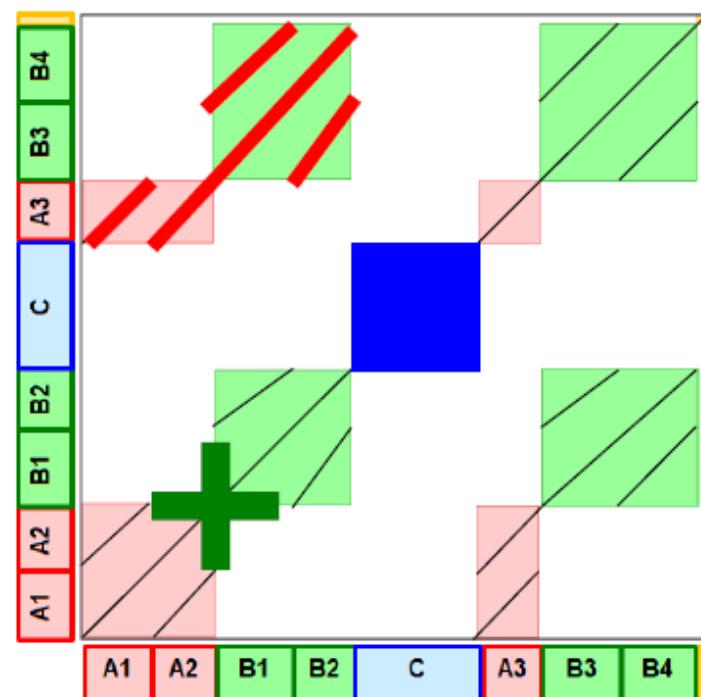
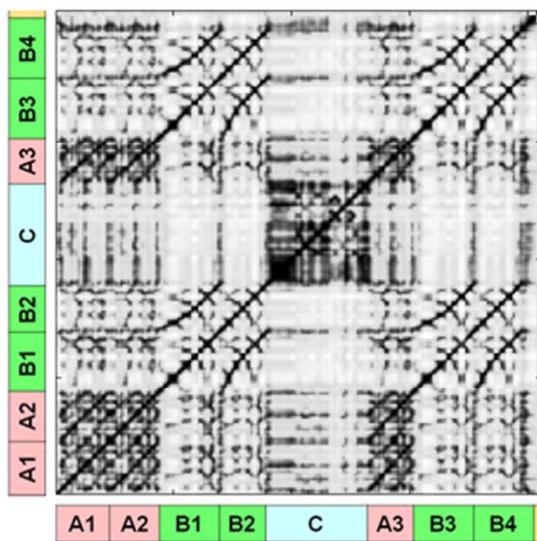


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

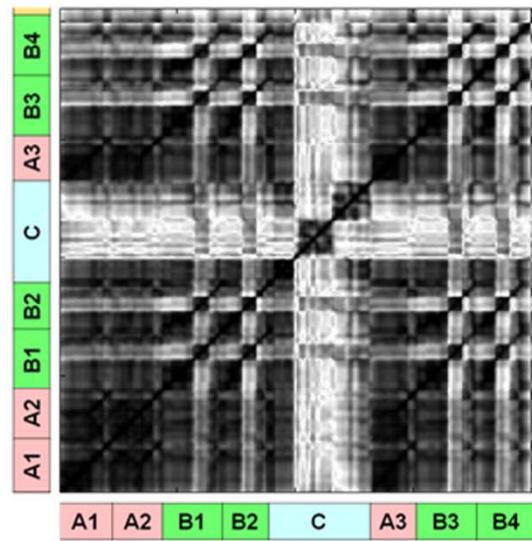
56

Self-Similarity Matrices (SSM)

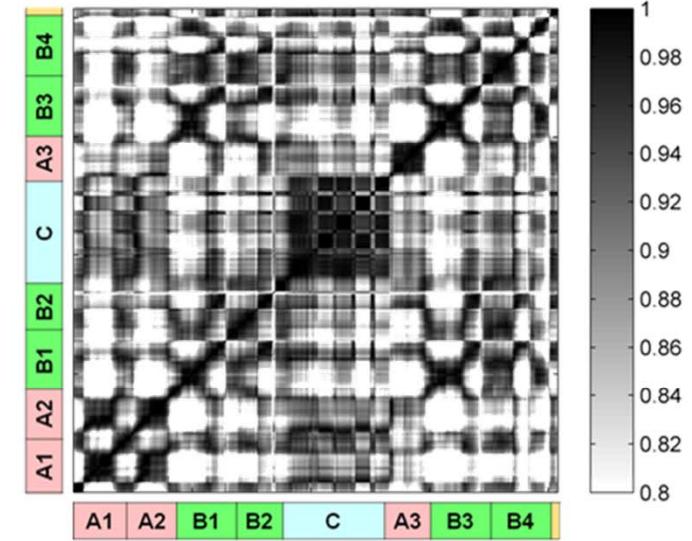
https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S2_SSM.html



SSM from chromagram



SSM from MFCC

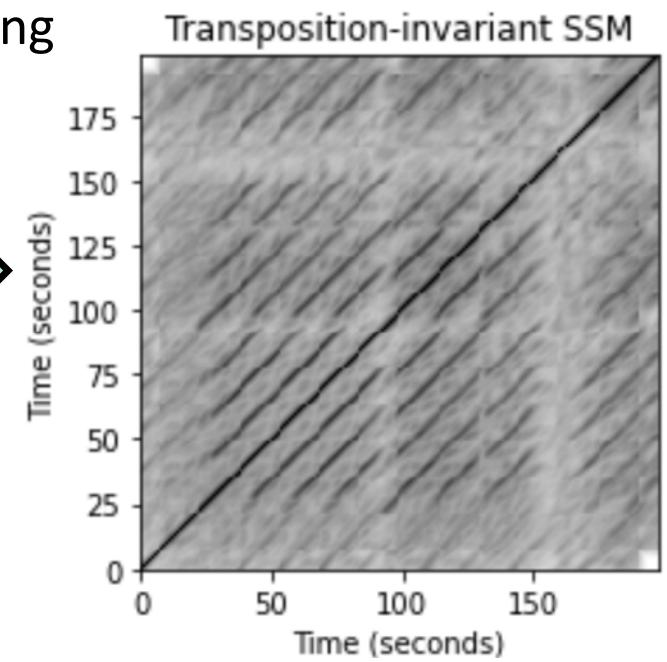
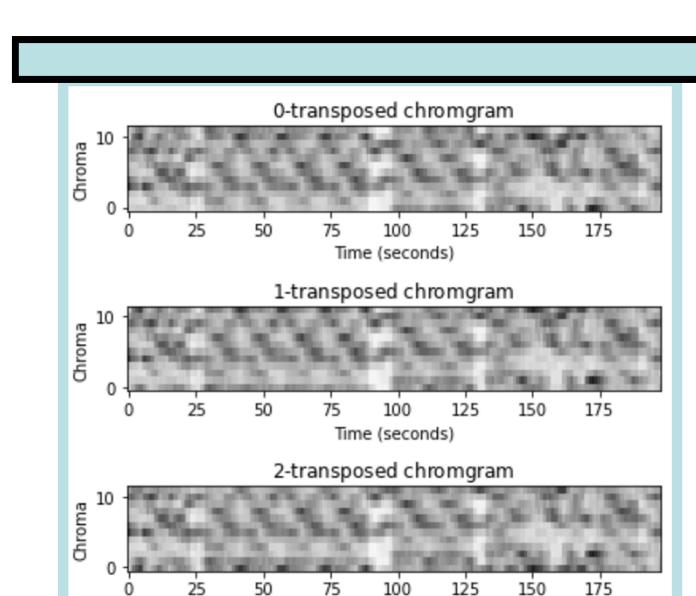
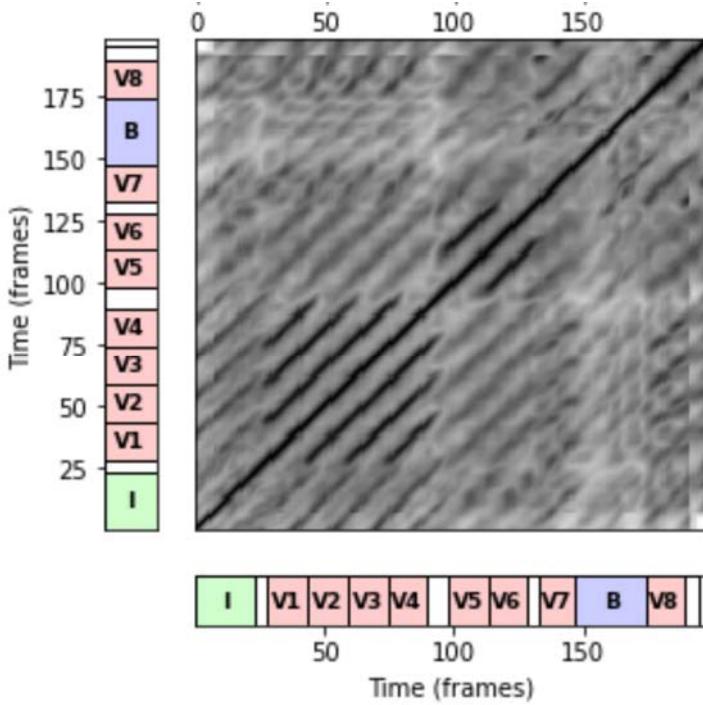


SSM from tempogram

Transposition-Invariant SSM

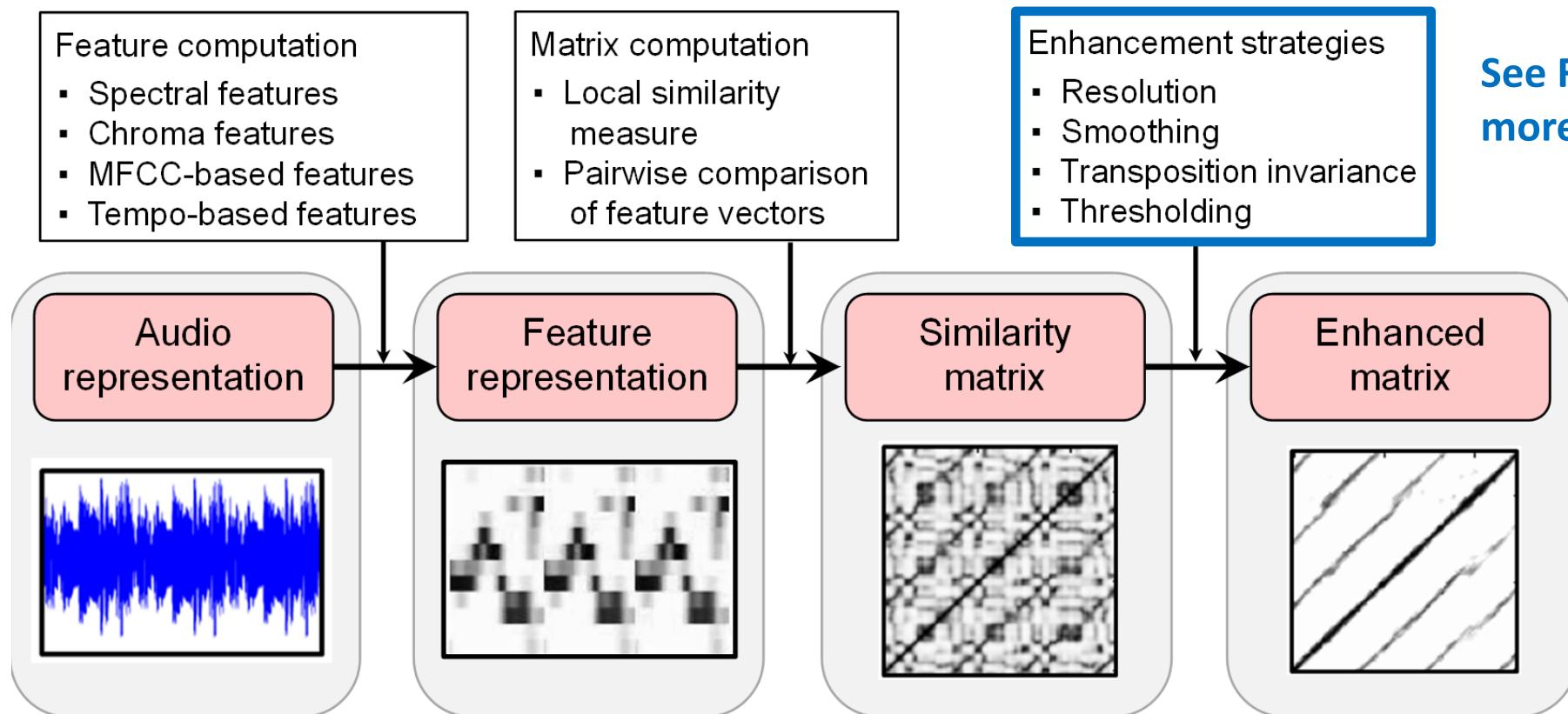
https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S2_SSM-TranspositionInvariance.html

1. Cyclic shift the chromagram
2. Compare the original recording with the transposed ones
3. Cell-wise **max** over 12 shifts



Procedure

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S1_MusicStructureGeneral.html



Novelty-based Segmentation

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S4_NoveltySegmentation.html

General goals:

- Find instances where musical changes occur.
- Find transition between subsequent musical parts.

Idea (Foote):

Use checkerboard-like kernel function to detect corner points on main diagonal of SSM.

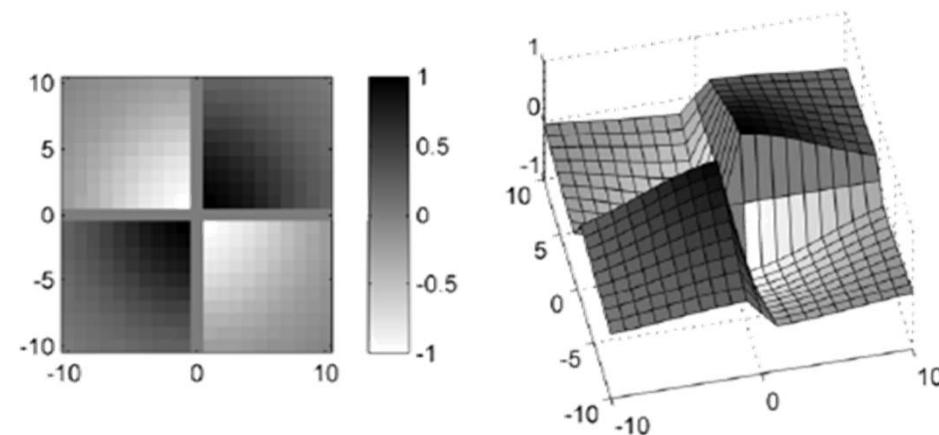
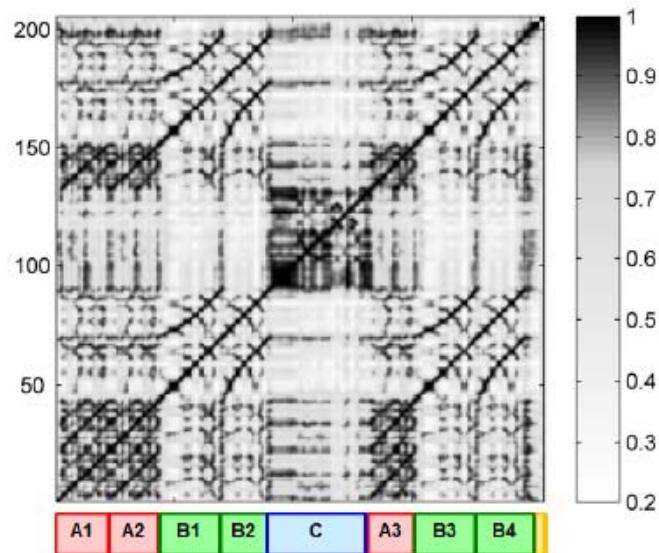


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Gaussian checkerboard kernel

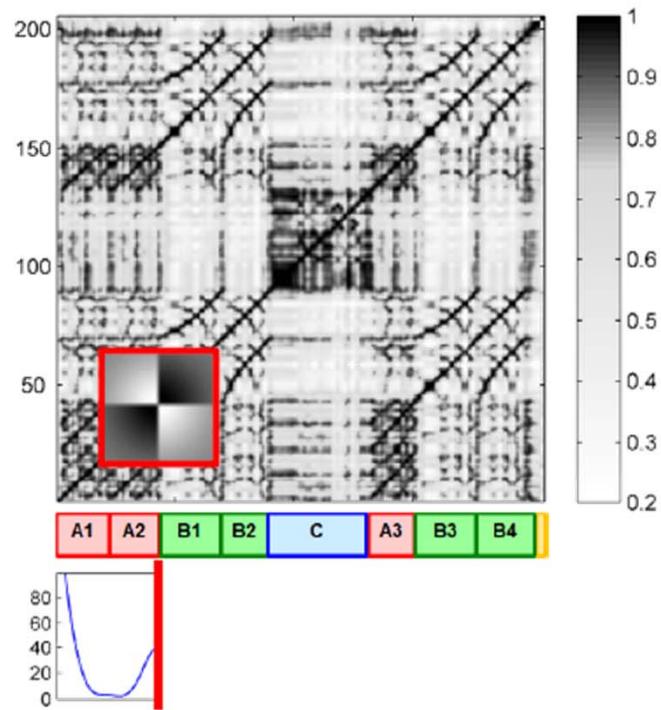
Novelty-based Segmentation



Idea (Foote):

Use checkerboard-like kernel function to detect corner points on main diagonal of SSM.

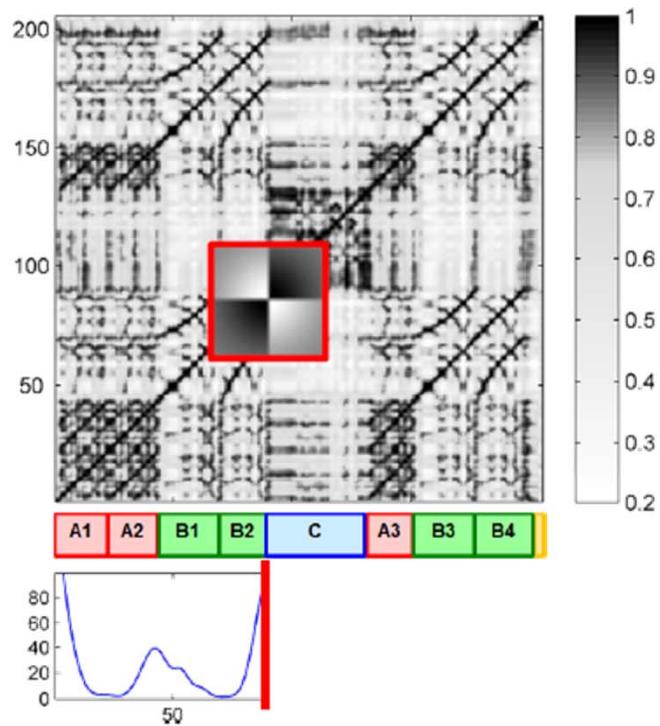
Novelty-based Segmentation



Idea (Foote):

Use checkerboard-like kernel function to detect corner points on main diagonal of SSM.

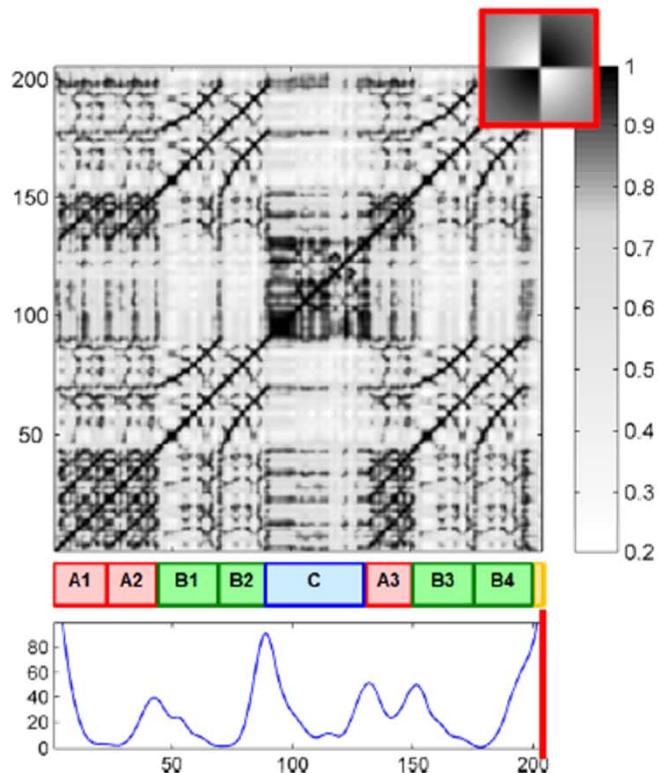
Novelty-based Segmentation



Idea (Foote):

Use checkerboard-like kernel function to detect corner points on main diagonal of SSM.

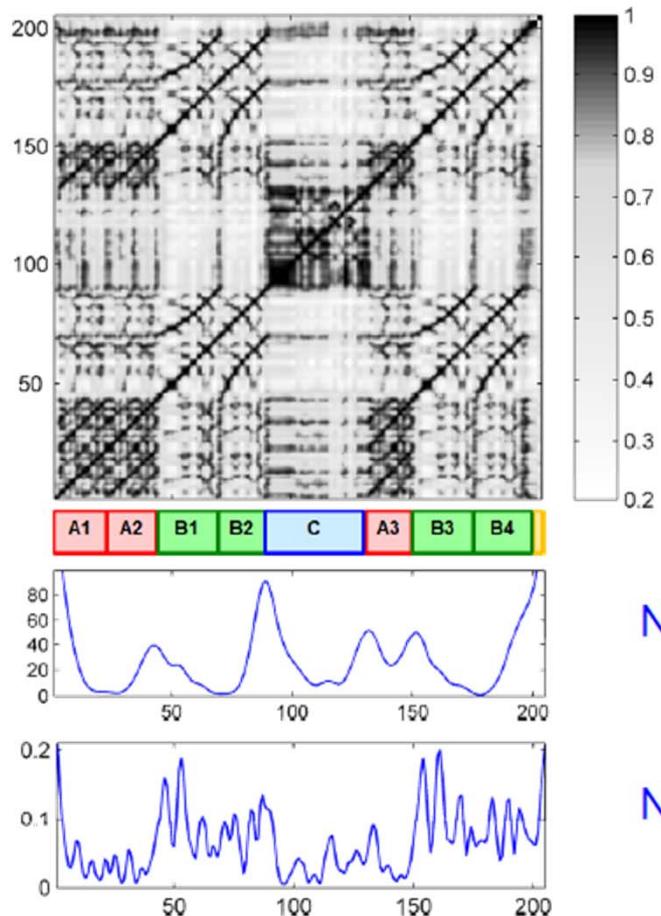
Novelty-based Segmentation



Idea (Foote):

Use checkerboard-like kernel function to detect corner points on main diagonal of SSM.

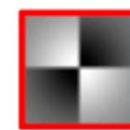
Novelty-based Segmentation



Idea (Foote):

Use checkerboard-like kernel function to detect corner points on main diagonal of SSM.

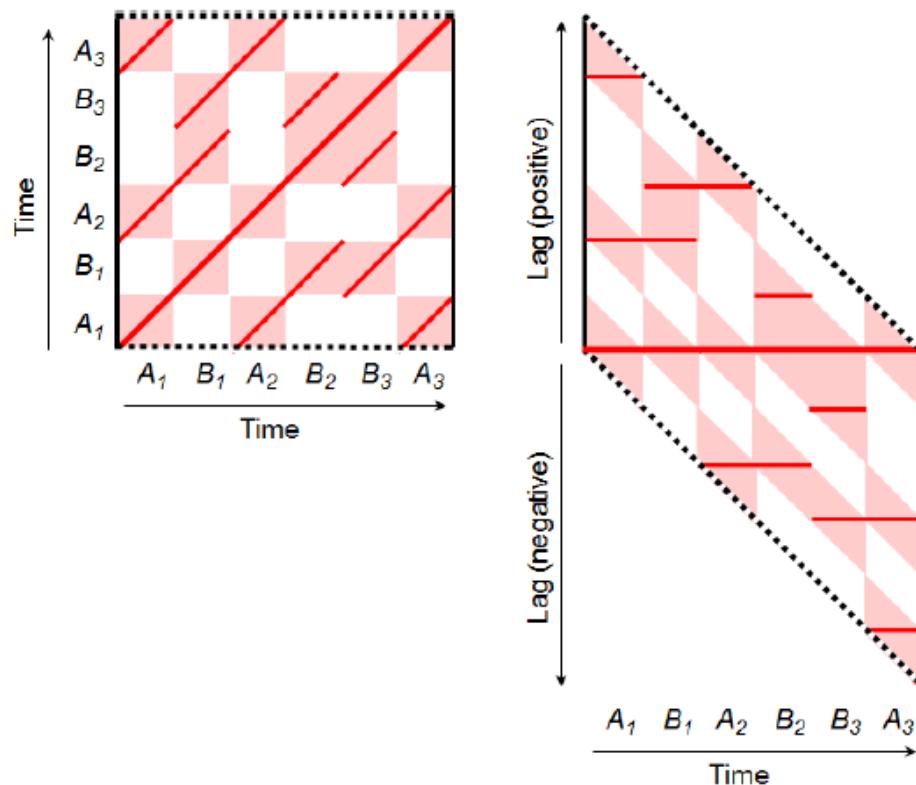
Novelty function using



Novelty function using



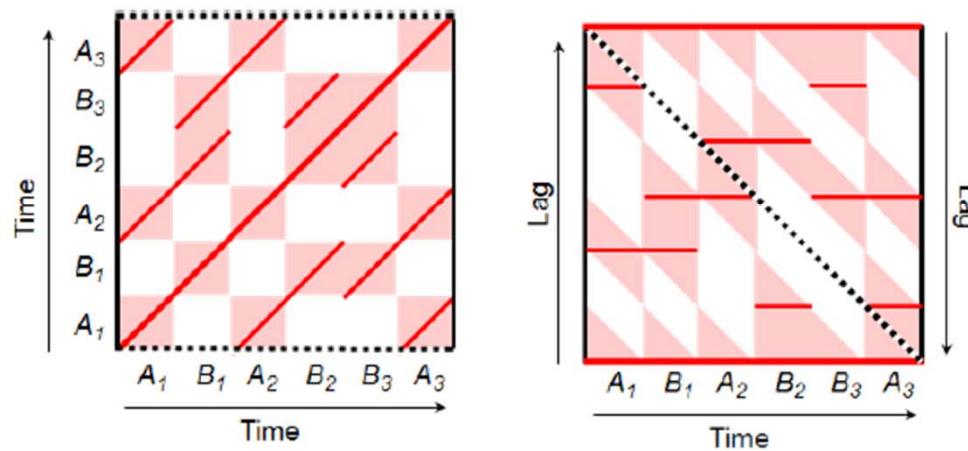
Structure Feature-based Segmentation



Structure features

- Enhanced SSM
- Time-lag SSM

Structure Feature-based Segmentation

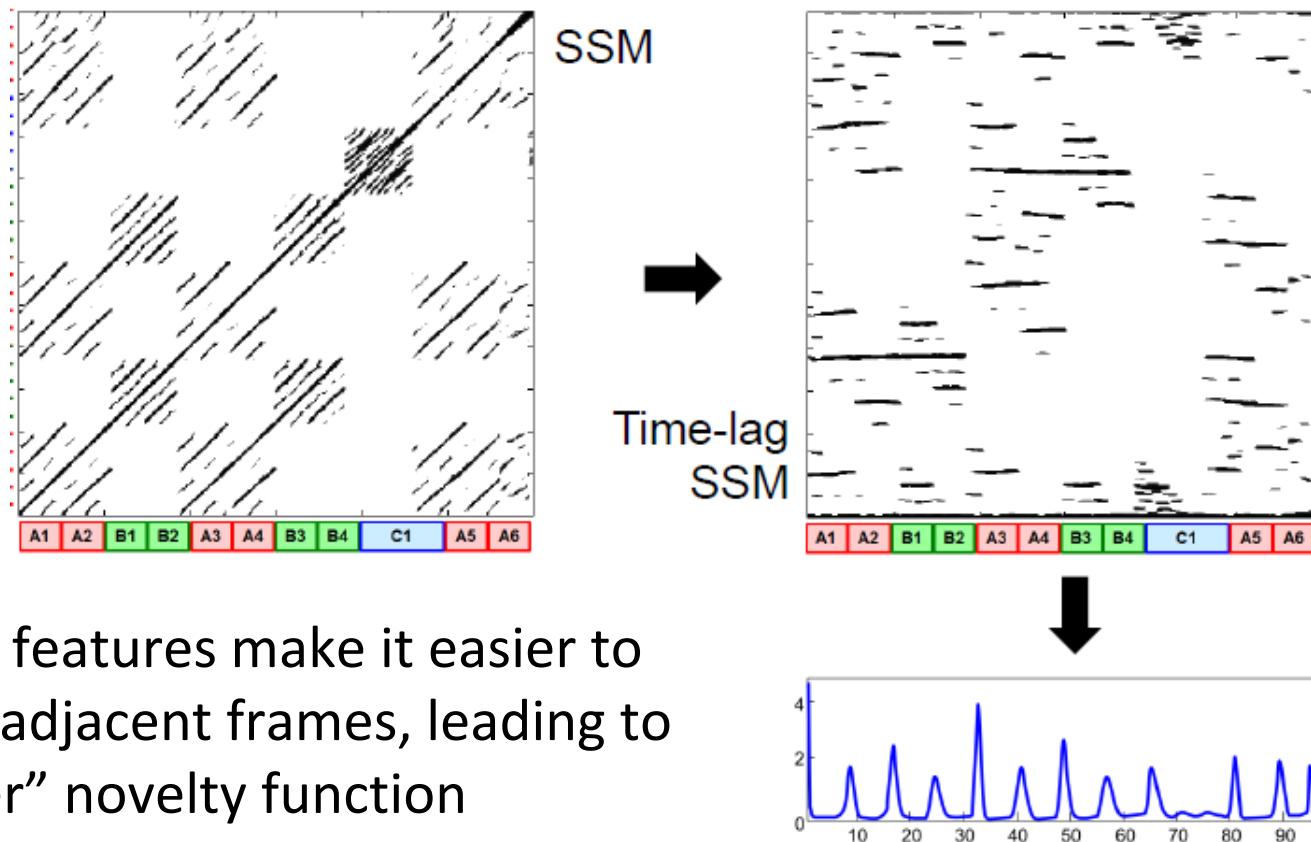


Structure features

- Enhanced SSM
- Time-lag SSM
- Cyclic time-lag SSM

Structure Feature-based Segmentation

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S4_StructureFeature.html



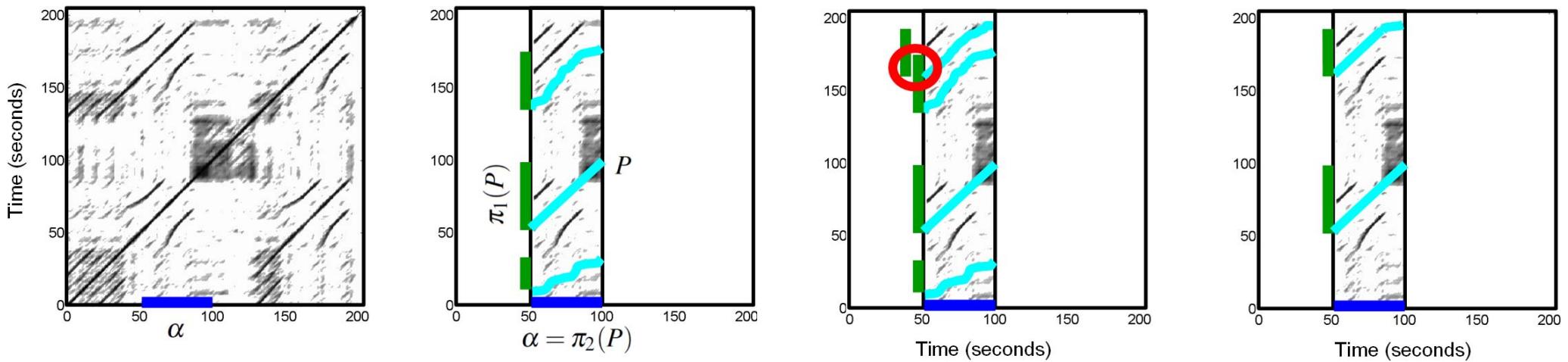
- Structure features make it easier to compare adjacent frames, leading to a “sharper” novelty function

Figure from [Mueller, FPM, Chapter 4, Springer 2015]

68

Application of SSM: Thumbnailing

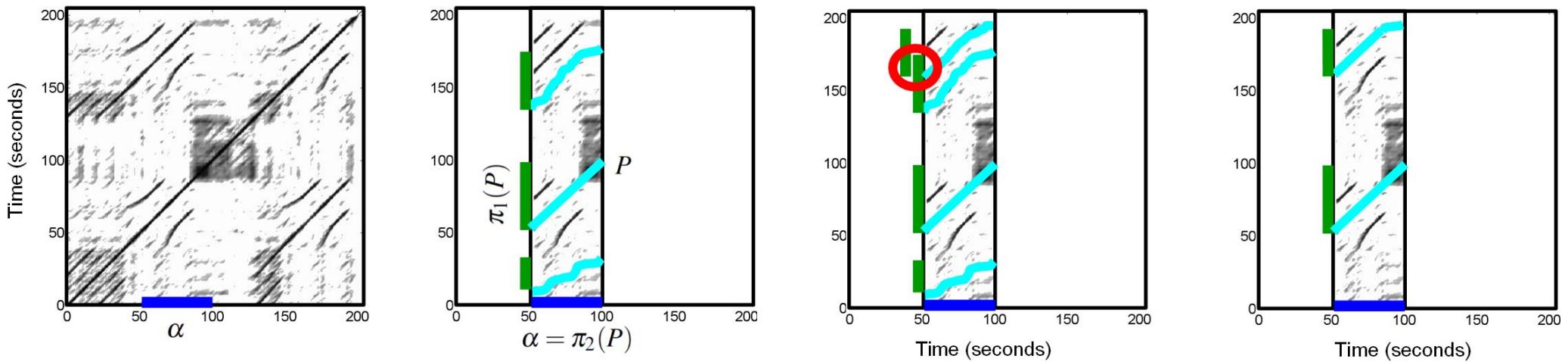
https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S3_AudioThumbnailing.html



- **Segment family:** the set of all the time intervals (**segments**) in the music that are identified as non-overlapping, high-quality repetitions of a given **segment**
- **Normalized score:** average similarity of the segments
- **Normalized coverage:** sum of the segments' length divided by the song length

Application of SSM: Thumbnailing

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S3_AudioThumbnailing.html



- **Fitness:** harmonic mean of normalized score and normalized coverage
- **Audio thumbnail:** the segment of maximal fitness
- **Scape plot representation:** a way to visualize the fitness values

Application: Scape Plot Representation

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S3_ScapePlot.html

- A segment can be defined by its starting and end points
- Or by its center and length
- Let's visualize its fitness score!

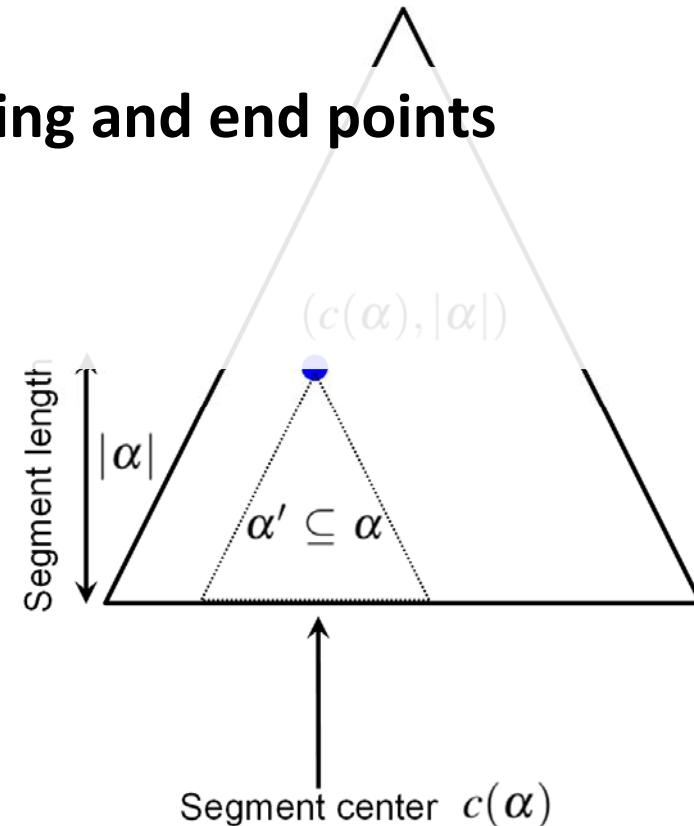
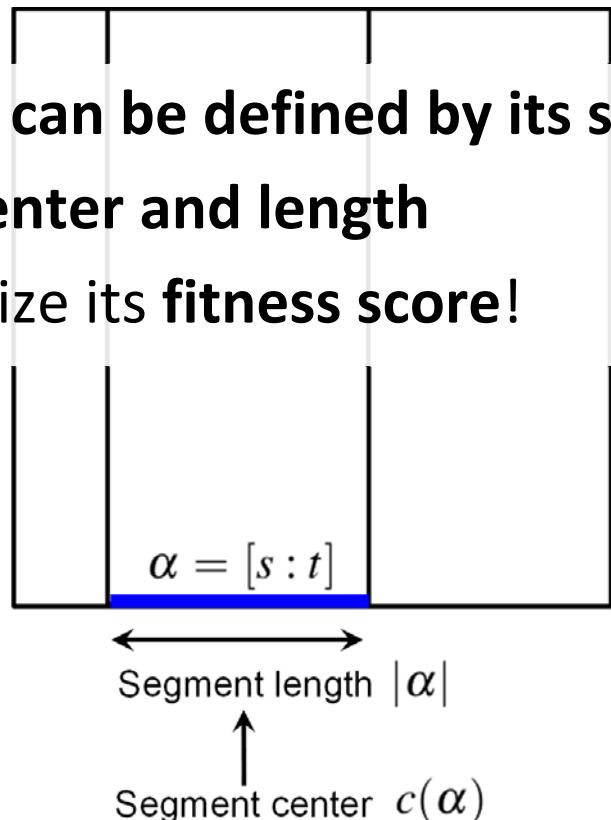


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Application: Scape Plot Representation

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S3_ScapePlot.html

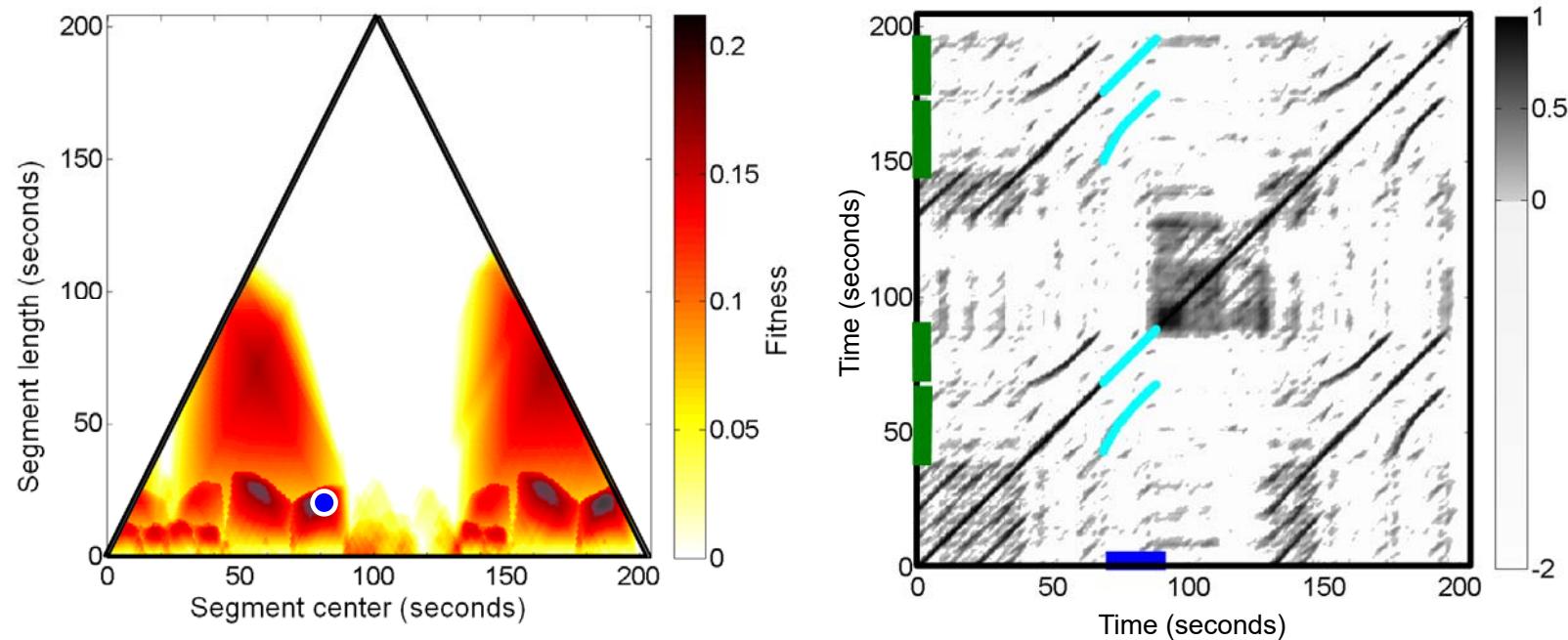


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Application: Scape Plot Representation

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C4/C4S3_ScapePlot.html

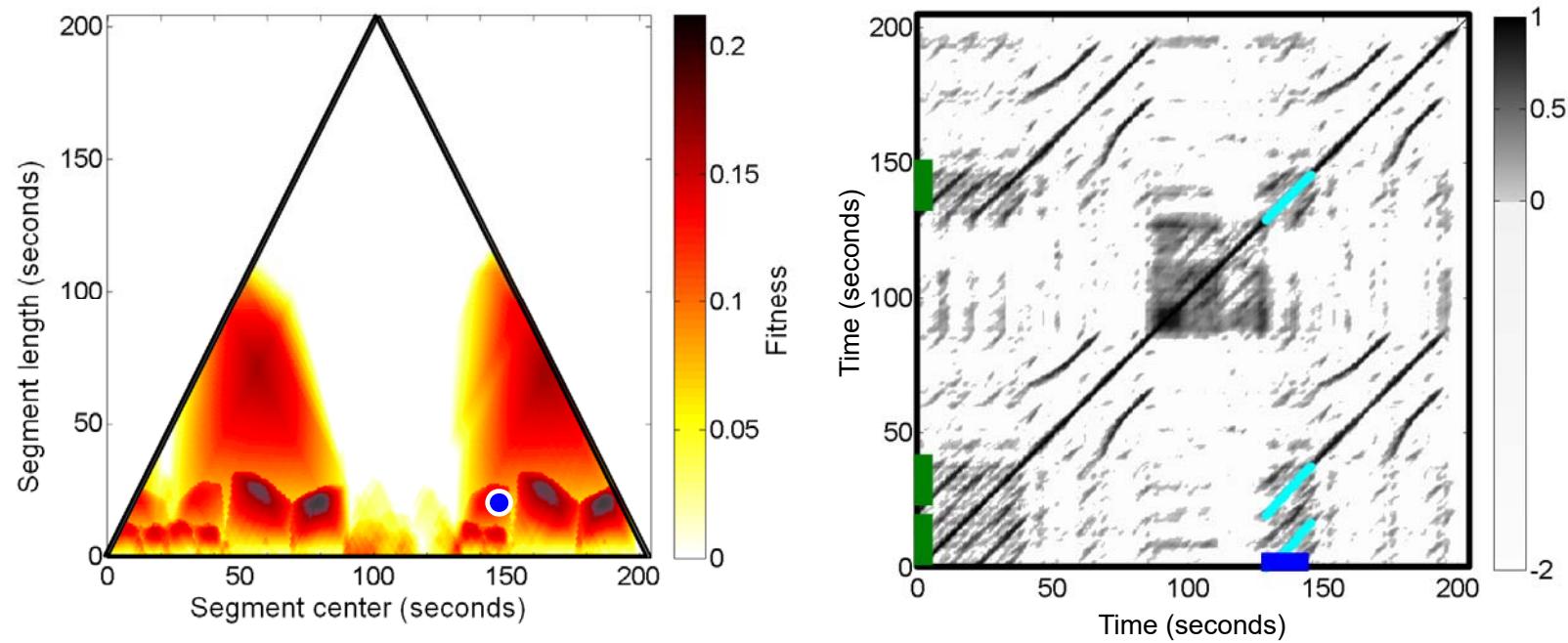


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Application: Scape Plot Representation

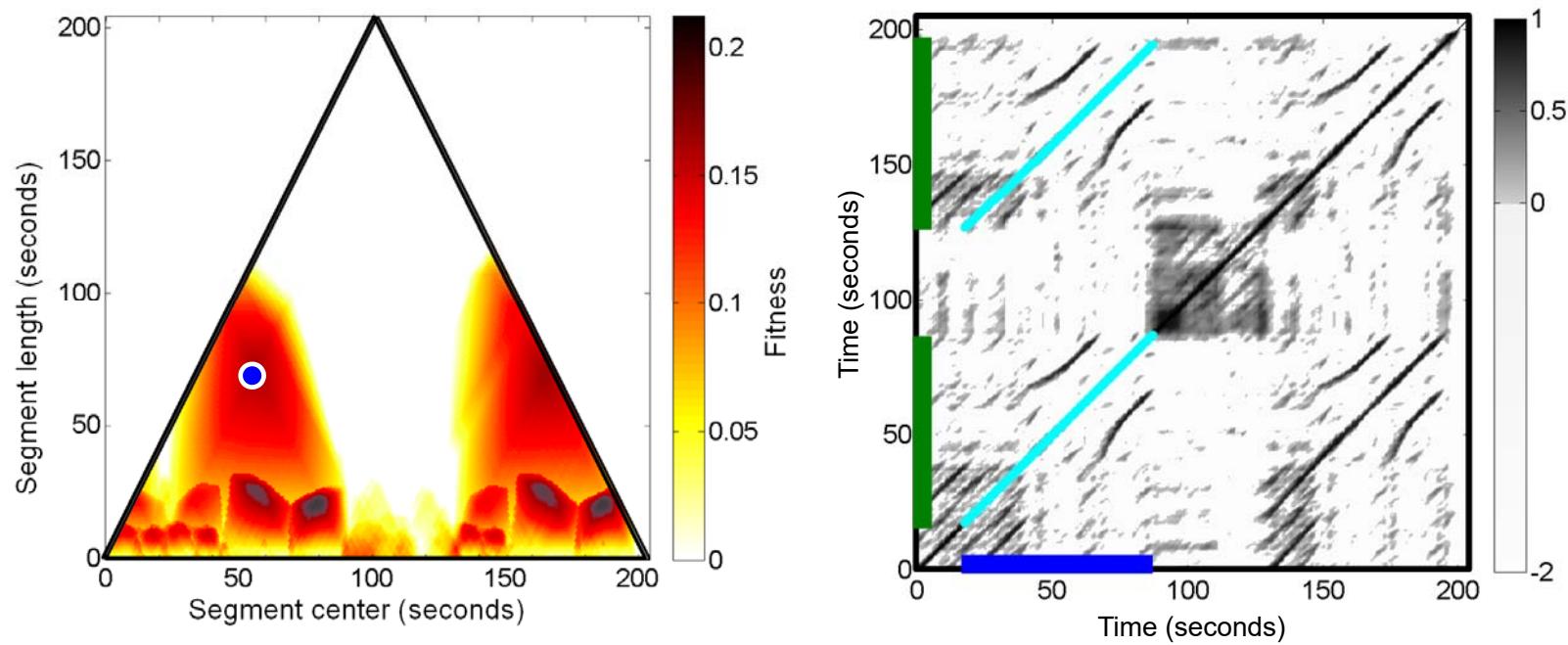


Figure from [Mueller, FPM, Chapter 4, Springer 2015]

Library: Librosa

<https://librosa.org/doc/latest/segment.html>

Temporal segmentation

Recurrence and self-similarity

| | |
|--|--|
| <code>recurrence_matrix</code> (data[, k, width, metric, ...]) | Compute a recurrence matrix from a signal. |
| <code>recurrence_to_lag</code> (rec[, pad, axis]) | Convert a recurrence matrix into a lag matrix. |
| <code>lag_to_recurrence</code> (lag[, axis]) | Convert a lag matrix into a recurrence matrix. |
| <code>timelag_filter</code> (function[, pad, index]) | Filtering in the time-lag domain. |

Temporal clustering

| | |
|--|---------------------------------------|
| <code>agglomerative</code> (data, k[, clusterer, axis]) | Bottom-up temporal segmentation. |
| <code>subsegment</code> (data, frames[, n_segments, axis]) | Sub-divide a segmentation by feature. |

Library: MSAF

<https://github.com/urinieto/msaf>

Boundary Algorithms

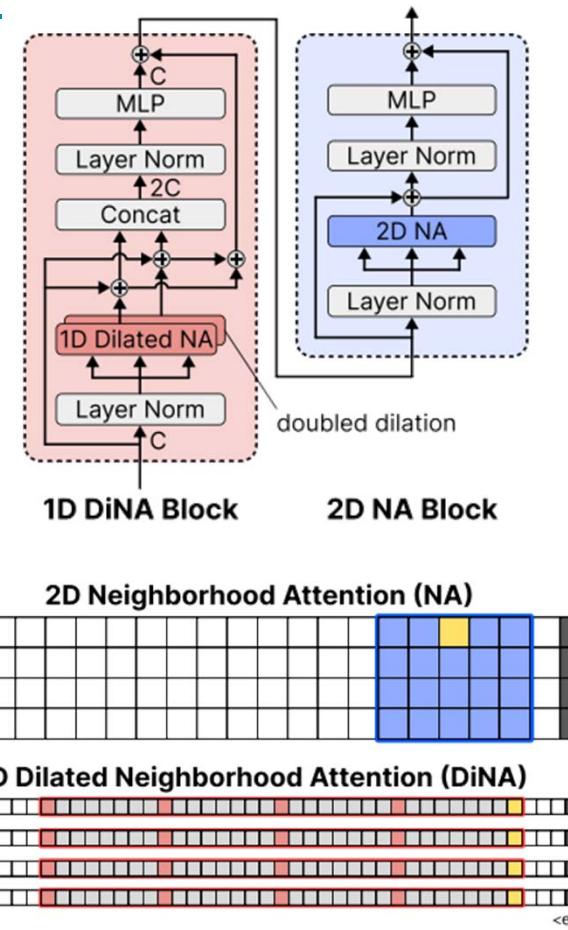
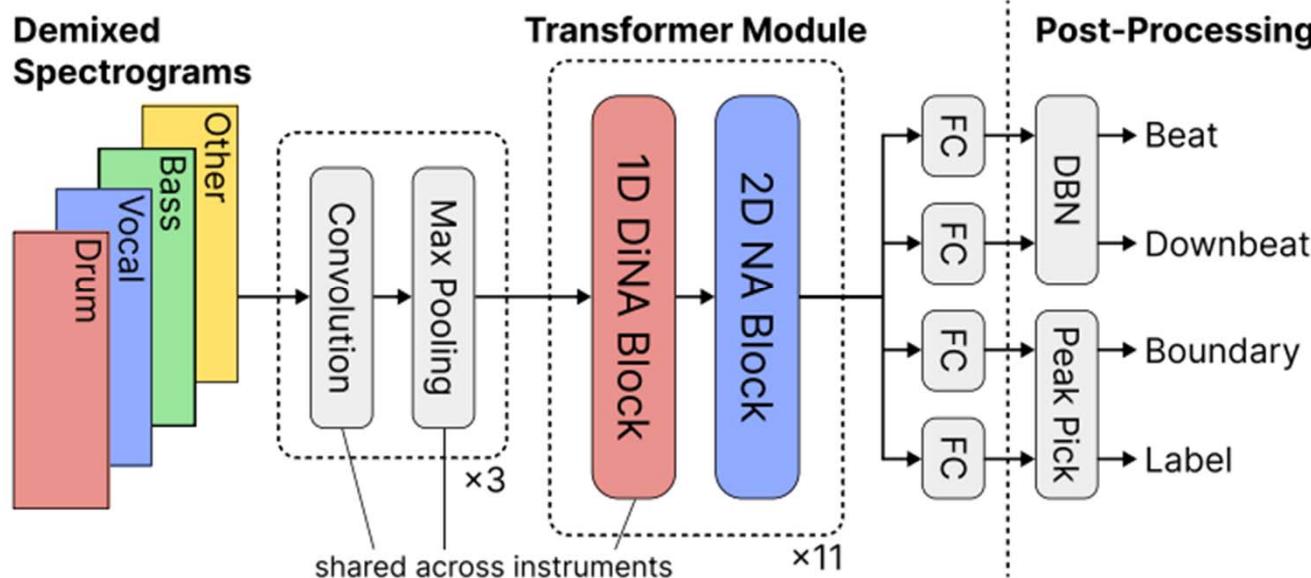
- Improved C-NMF (Nieto & Jehan 2013)
- Checkerboard-like Kernel (Foote 2000)
- OLDA (McFee & Ellis 2014) (original source code from [here](#))
- Spectral Clustering (McFee & Ellis 2014) (original source code from [here](#))
- Structural Features (Serrà et al. 2012)

Labeling Algorithms

- Improved C-NMF (Nieto & Jehan 2013)
- 2D Fourier Magnitude Coefficients (Nieto & Bello 2014)
- Spectral Clustering (McFee & Ellis 2014) (original source code from [here](#))

All-In-One

<https://github.com/mir-aidj/all-in-one>



Ref: Kim & Nam, "All-In-One metrical and functional structure analysis with neighborhood attentions on demixed audio," WASPAA 2013

All-In-One

<https://github.com/mir-aidj/all-in-one>

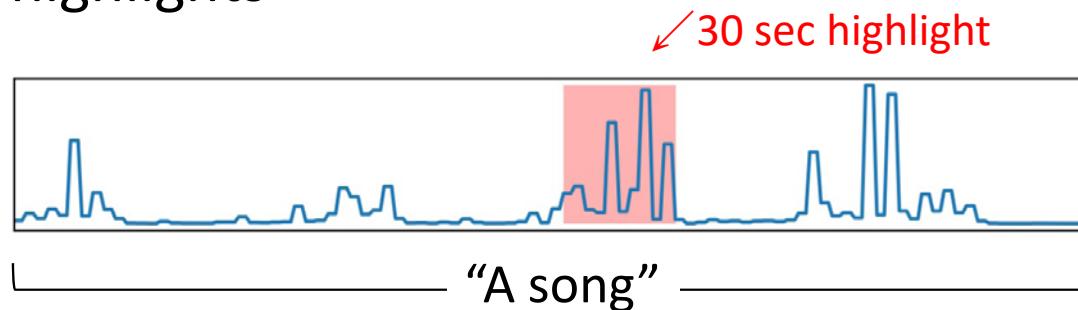
| Model | # of Params | Beat | | | Downbeat | | | Segment | | Label | |
|-------------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|---|
| | | F1 | CMLt | AMLt | F1 | CMLt | AMLt | HR.5F | PWF | Sf | |
| SpecTNT-TCN [11]* | 4.7 M | .953 | .939 | .959 | .908 | .872 | .928 | — | — | — | — |
| Beat Transformer [5]* | 9.3 M | .954 | .905 | .957 | .898 | .863 | .919 | — | — | — | — |
| DSF+Scluster [9] | N/A | — | — | — | — | — | — | .497 | .684 | .743 | |
| SpecTNT [10]* | N/A | — | — | — | — | — | — | .558 | .712 | .724 | |
| TCN w/o demix [3]† | 74 K | .954 | .900 | .961 | .886 | .842 | .920 | .594 | .687 | .694 | |
| TCN† | 93 K | .946 | .898 | .950 | .894 | .850 | .919 | .619 | .715 | .738 | |
| TCN-Large† | 301 K | .953 | .906 | .960 | .901 | .853 | .924 | .626 | .717 | .746 | |
| All-In-One-Small (Ours) | 46 K | .943 | .891 | .952 | .901 | .854 | .929 | .616 | .713 | .745 | |
| All-In-One (Ours) | 300 K | .958 | .913 | .964 | .915 | .873 | .932 | .660 | .738 | .769 | |

Ref: Kim & Nam, “All-In-One metrical and functional structure analysis with neighborhood attentions on demixed audio,” WASPAA 2013

Extension 1: Music Thumbnailing (Highlight Detection)

https://remyhuang.github.io/music_thumbnailing/

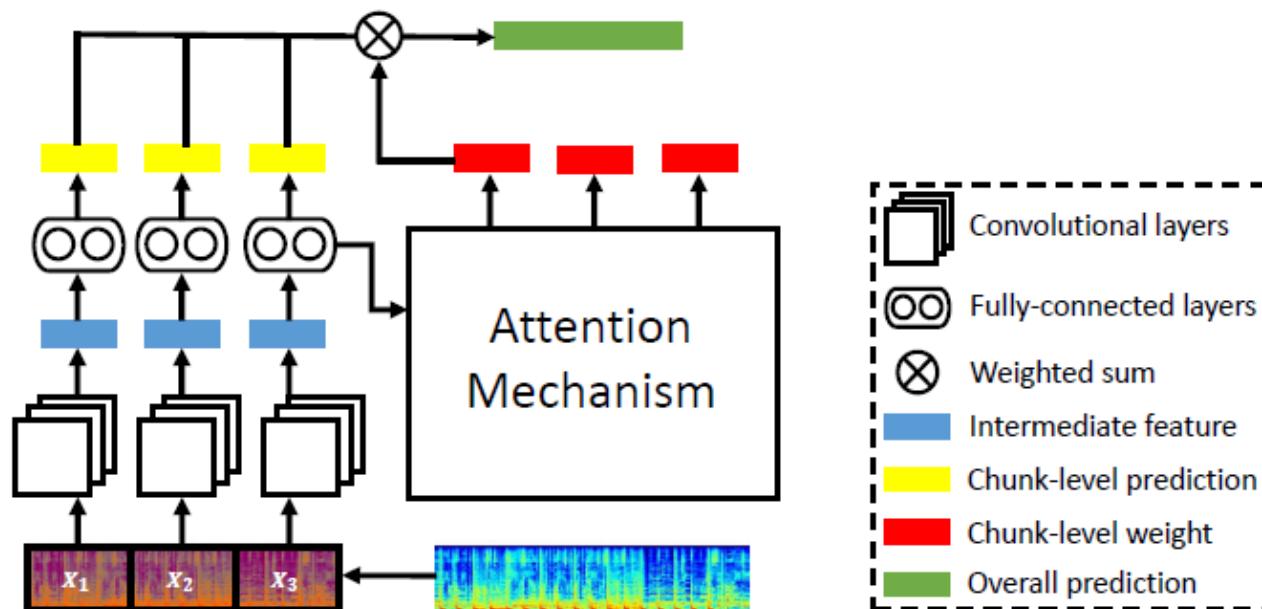
- Extract music highlights



- Application
 - music browsing
 - ringtone generation
 - song representation

Extension 1: Music Thumbnailing (Highlight Detection)

- CNN for **emotion prediction** + **attention** (predicting weights of different parts of a song)
- *Transfer learning*: no need of structural (chorus) labels



Ref: Huang et al., "Pop music highlighter: Marking the emotion keypoints," TISMIR 2018

Extension 1: Music Thumbnailing (Highlight Detection)

- Performance of different music highlight extraction methods on “chorus detection”
 - Training data: MER31K (emotion labels)
 - Test data: RWC-Pop100 (chorus labels)
 - The baseline method is based on spectral energy



Extension 2: Music Sequencing

<https://www.youtube.com/watch?v=yE5DiniY45w>

- Find an ordering of music pieces within a music playlist
- Or “medley” (mashup) generation (ref: “Pop Danthology”)



Daniel Kim •

@canadankim • 1.06M subscribers • 76 videos

Content creator most known for making "Pop Danthology" ...[more](#)

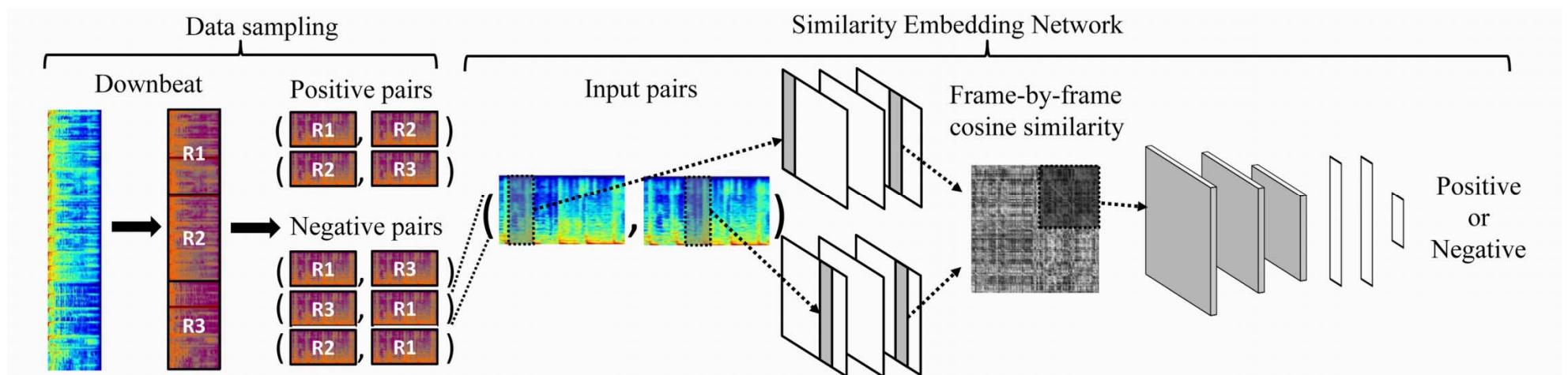
facebook.com/canadankim and 3 more links

[Subscribe](#)

Extension 2: Sequencing via Solving the Music Puzzle Game

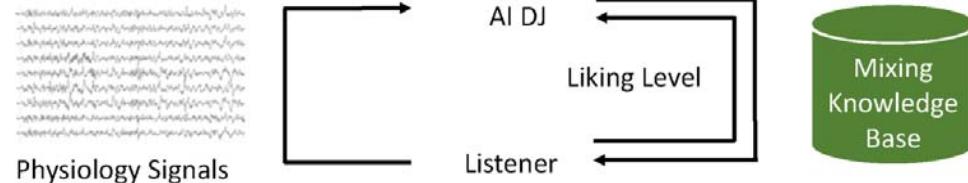
https://remyhuang.github.io/music_puzzle_game/

- Divide a song into non-overlapping chunks
- Learn to order them by contrastive learning
 - Positive pair: R1R2, R2R3
 - Negative pair: R2R1, R3R2, R1R3, R3R1



AI DJ

Affective Measurement and Computation

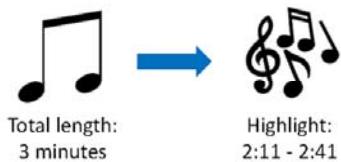


DJ Side

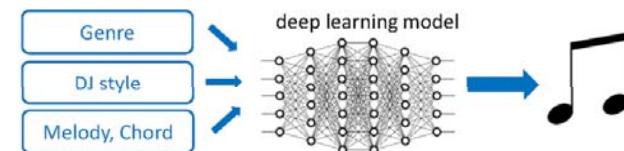


Music Side

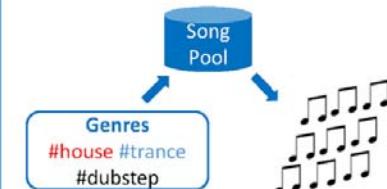
Highlight Detection



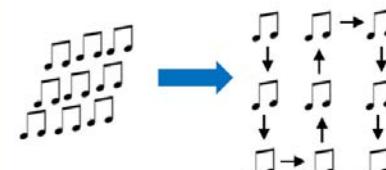
Electronic Dance Music Generation



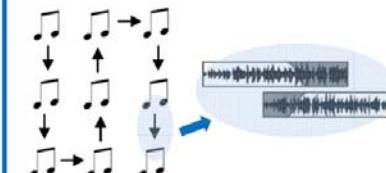
Selection



Sequencing



Mixing

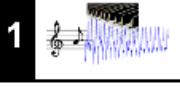
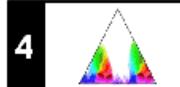
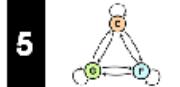
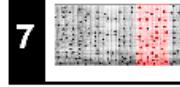
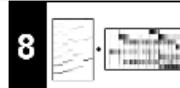


Outline

- Music emotion
- Structure/form analysis
- **Alignment**
- Rhythm & beat tracking

FMP Notebook

<https://www.audiolabs-erlangen.de/resources/MIR/FMP/C3/C3.html>

| Part | Title | Notions, Techniques & Algorithms | HTML | IPYNB |
|------|--|--|------------------------|-------------------------|
| B |  Basics | Basic information on Python, Jupyter notebooks, Anaconda package management system, Python environments, visualizations, and other topics | [html] | [ipynb] |
| 0 |  Overview | Overview of the notebooks (https://www.audiolabs-erlangen.de/FMP) | [html] | [ipynb] |
| 1 |  Music Representations | Music notation, MIDI, audio signal, waveform, pitch, loudness, timbre | [html] | [ipynb] |
| 2 |  Fourier Analysis of Signals | Discrete/analog signal, sinusoid, exponential, Fourier transform, Fourier representation, DFT, FFT, STFT | [html] | [ipynb] |
| 3 |  Music Synchronization | Chroma feature, dynamic programming, dynamic time warping (DTW), alignment, user interface | [html] | [ipynb] |
| 4 |  Music Structure Analysis | Similarity matrix, repetition, thumbnail, homogeneity, novelty, evaluation, precision, recall, F-measure, visualization, scape plot | [html] | [ipynb] |
| 5 |  Chord Recognition | Harmony, music theory, chords, scales, templates, hidden Markov model (HMM), evaluation | [html] | [ipynb] |
| 6 |  Tempo and Beat Tracking | Onset, novelty, tempo, tempogram, beat, periodicity, Fourier analysis, autocorrelation | [html] | [ipynb] |
| 7 |  Content-Based Audio Retrieval | Identification, fingerprint, indexing, inverted list, matching, version, cover song | [html] | [ipynb] |
| 8 |  Musically Informed Audio Decomposition | Harmonic/percussive separation, signal reconstruction, instantaneous frequency, fundamental frequency (F0), trajectory, nonnegative matrix factorization (NMF) | [html] | [ipynb] |

Audio-to-Audio Synchronization: Intro

- Two interpretations – Tchaikovsky Violin Concerto

Jascha Heifetz

<https://youtu.be/kFaq9kTlcaY?t=39s>



David Oistrakh

<https://youtu.be/fNCeYKfAOZI?si=0dGo2LBEPRHgfaDI&t=108>

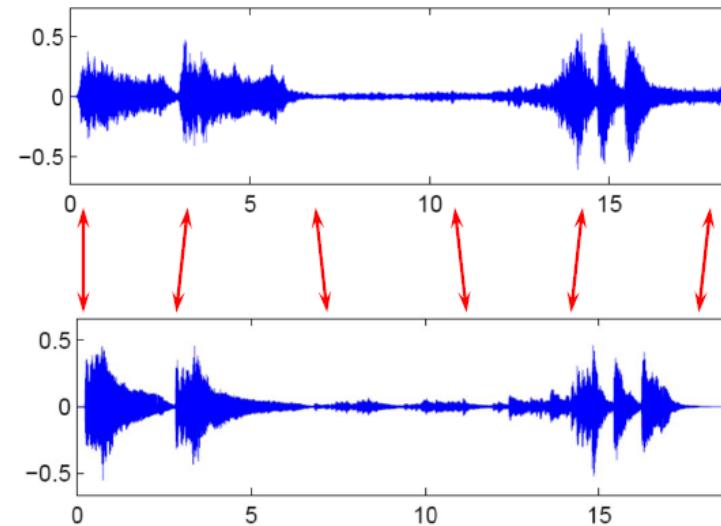


Audio-to-Audio Synchronization: Intro

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C3/C3_MusicSynchronization.html

Beethoven's Fifth

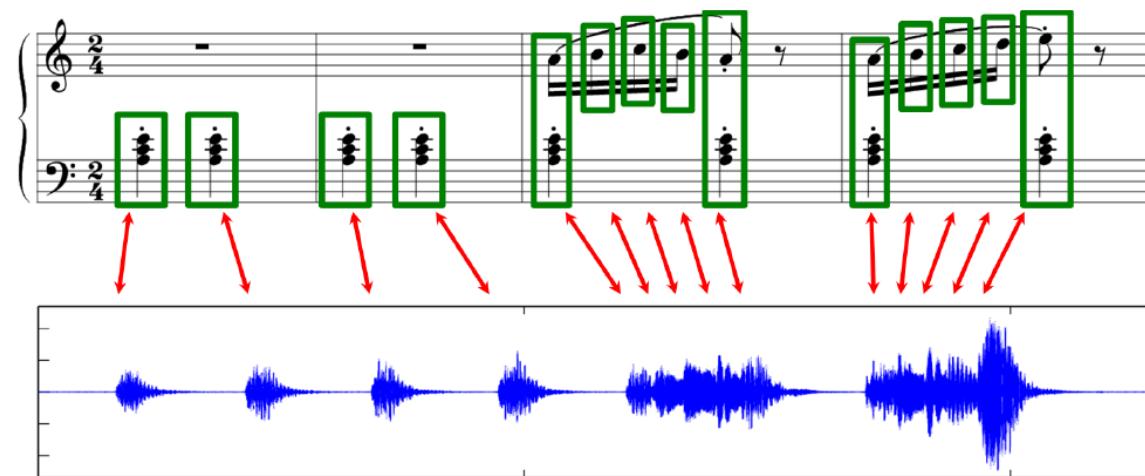
Karajan



Scherbakov



Audio-to-Score Synchronization: Intro



- Need to take into account tempo variations

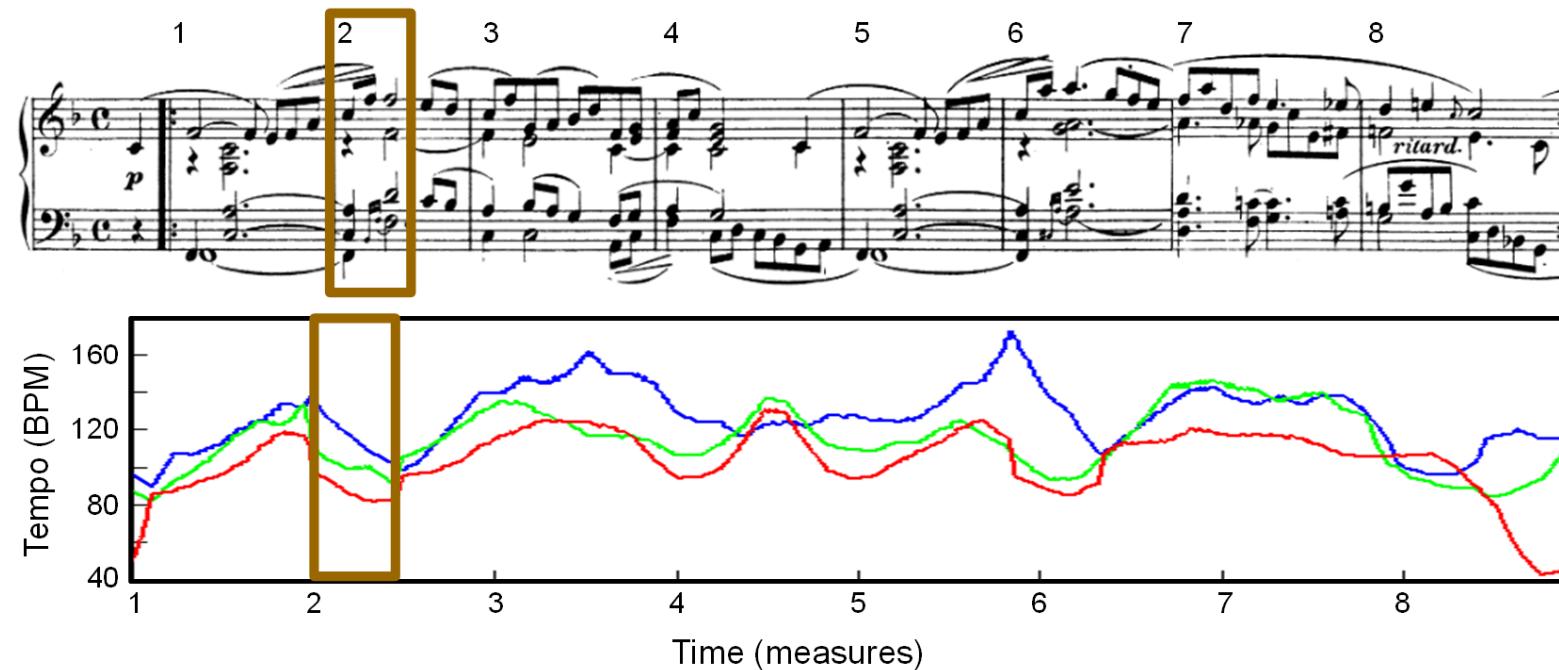
Figure from [Mueller, FPM, Chapter 3, Springer 2015]

Audio-to-Score Synchronization: Intro

- **Goal:** given the score of a music piece and an audio recording of it, align them in time
- **Offline**
 - the whole audio recording is given
 - application: *performance analysis, feature extraction*
- **Online**
 - the alignment is done on-the-fly progressively while the recording is being played
 - a.k.a. **score following**
 - constraint: **cannot “look into the future”** (i.e. has to be *causal*)
 - application: *automatic accompaniment, page turner*
 - **harder** due to the causal and low-latency constraint

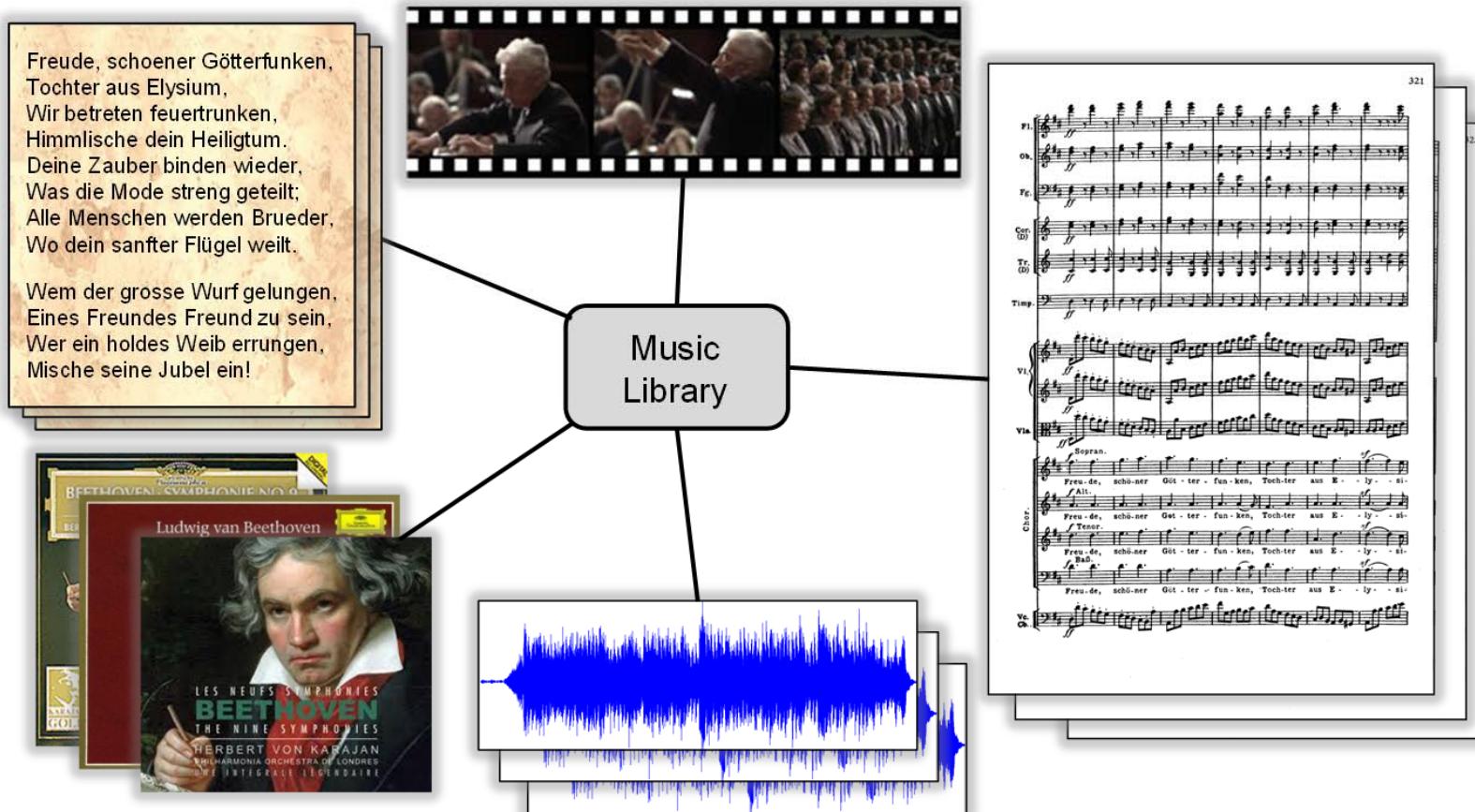
Offline Application: Performance Analysis

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C3/C3S3_MusicAppTempoCurve.html



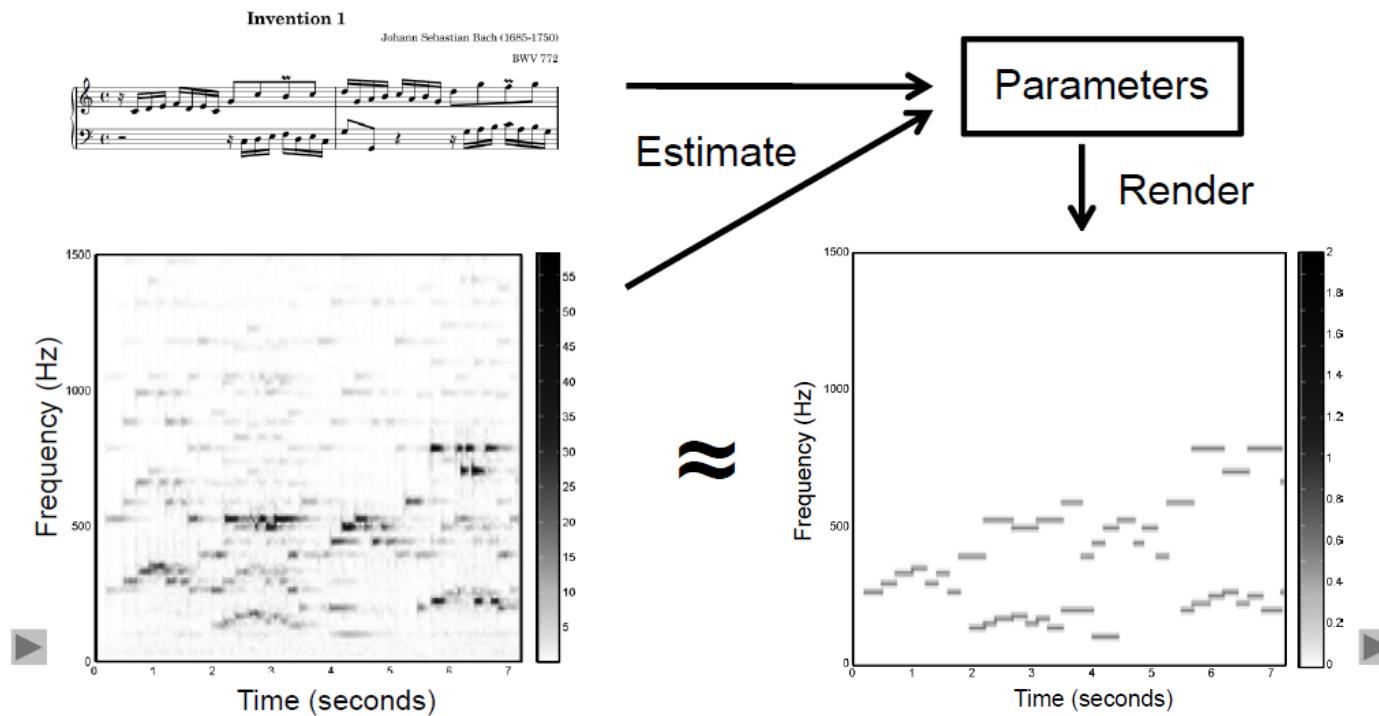
Offline Application: Feature Extraction

https://www.audiolabs-erlangen.de/resources/MIR/FMP/C3/C3S3_MusicAppNav.html



Offline Application: Score-Informed Source Separation

Goal: approximate spectrogram using a parametric model exploiting the score (requires *offline* synchronization)



Online Application: Page Turner

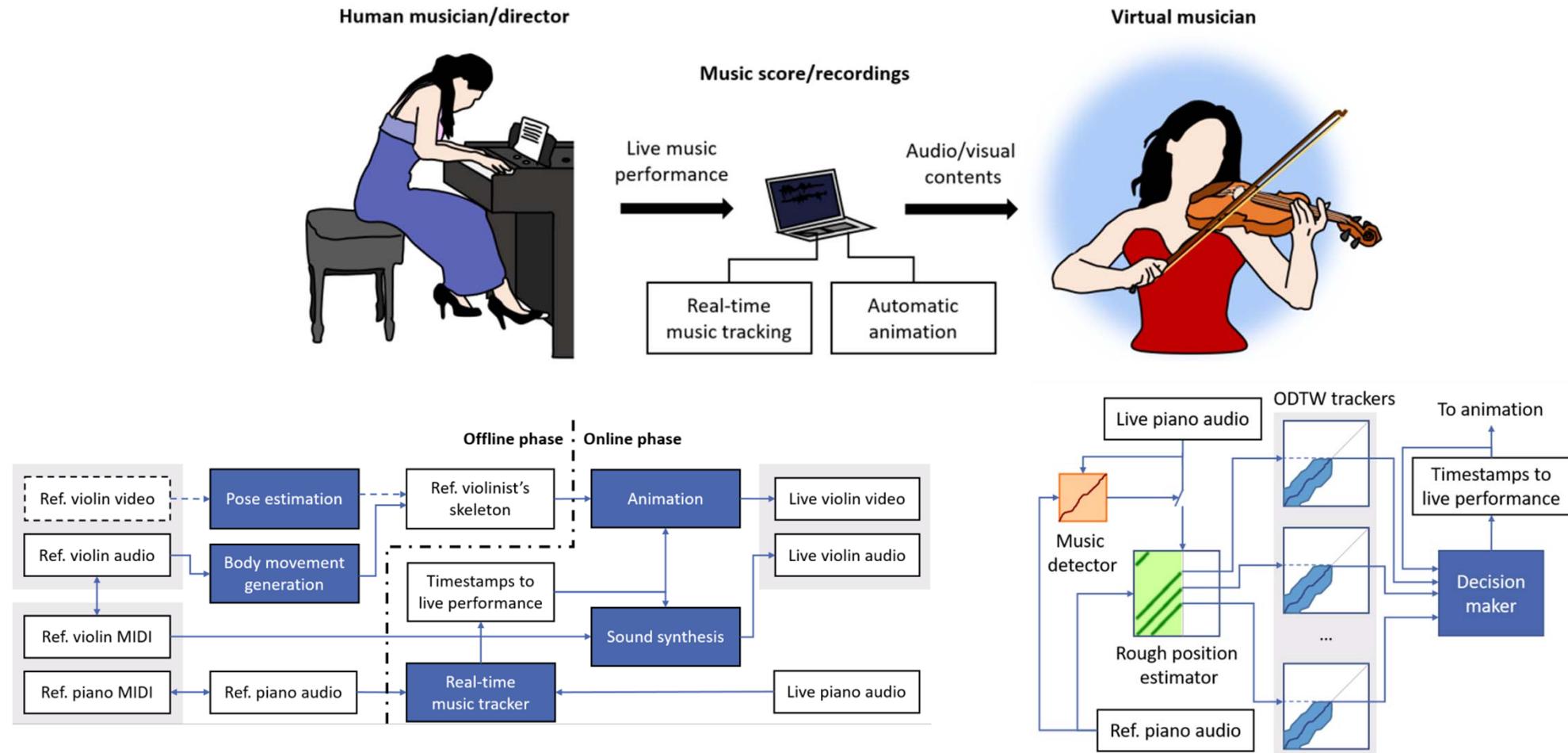
<https://www.youtube.com/watch?v=HBXJZKTOcpw&t=25s>

- **Goal:** a computer listens to the music and knows “when” to turn the pages (requires *online* synchronization)



This block contains four separate panels of sheet music, each showing a different section of the musical score. The music is for two staves: piano (top) and violin (bottom). The panels are arranged vertically. The first panel shows measures 1-4, the second panel shows measures 5-8, the third panel shows measures 9-12, and the fourth panel shows measures 13-16. The piano part includes dynamics like 'Lento' and 'Pianissimo'. The violin part includes various bowing and string markings. A blue vertical bar highlights a specific measure in the second panel, likely indicating where a page turn would occur.

Online Application: Automatic Accompaniment



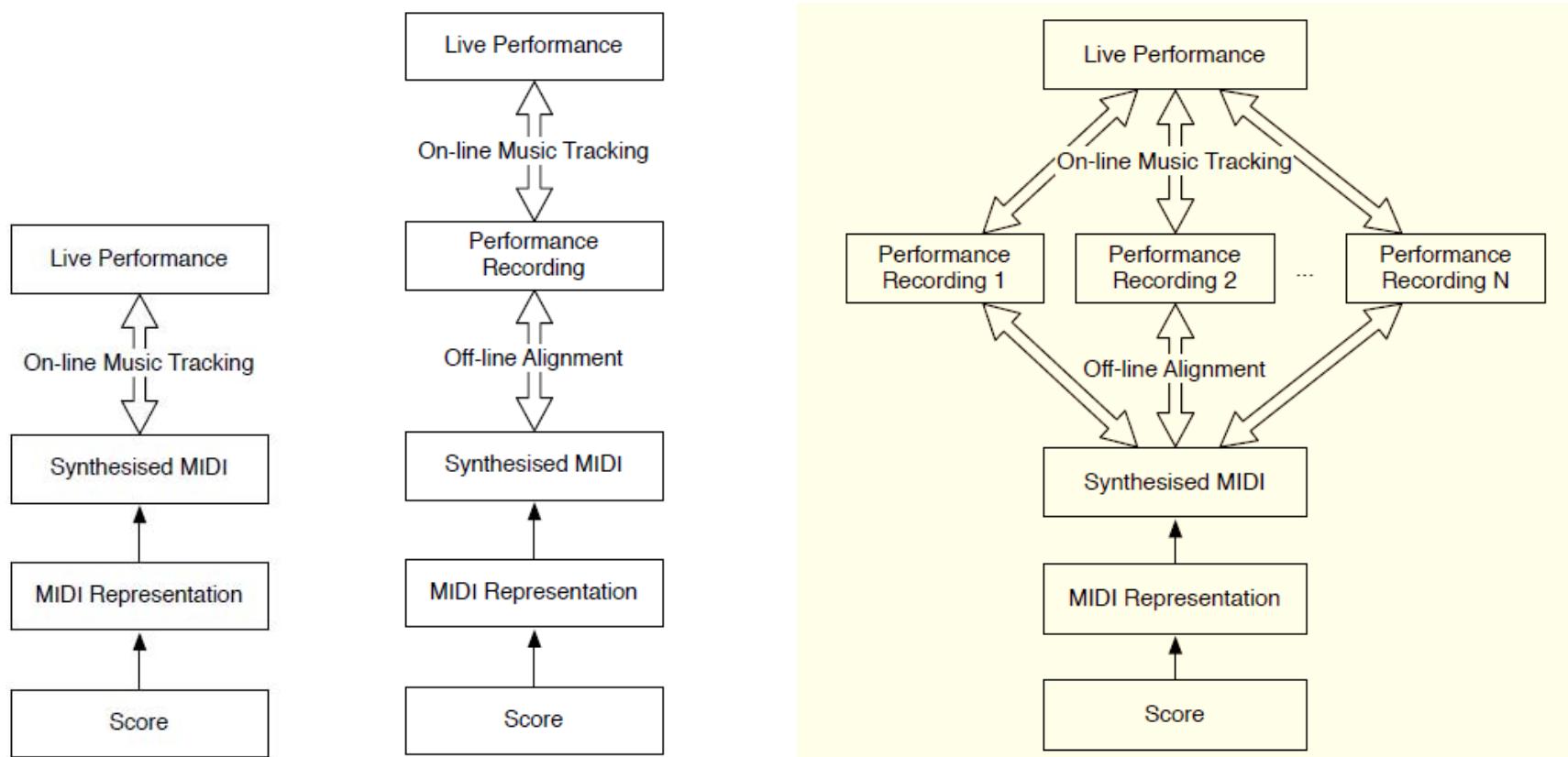
Ref: Lin et al, “A human-computer duet system for music performance,” ACM Multimedia 2020

Audio-to-Score Synchronization: Approach

- Approach 1
 - use **MIDI synthesizers** (e.g. fluidsynth) to convert a score representation into a sound file
 - then treat it as an **audio-to-audio synchronization** problem
 - advantage
 - easy to implement
 - applicable for both offline and online scenario
 - drawback
 - low quality/deadpan/mechanical
 - a performance implicitly encodes a lot of information that is missing in the symbolic score

- **Approach 2**

- use different performances of the piece as reference



Ref: Arzt et al, “Real-time music tracking using multiple performances as a reference,” ISMIR 2015

Recap

- Many music synchronization problems **can be viewed as an audio-to-audio synchronization problem**
- **Question:** how to compare two audio sequences and find musically corresponding time positions
 1. Feature representation
 2. Alignment technique, offline and online
- First research dates back to 1984

Ref 1: "The synthetic performer in the context of live performance," ICMC 1984

Ref 2: "An on-line algorithm for real-time accompaniment," ICMC 1984

Two Main Components of Synchronization

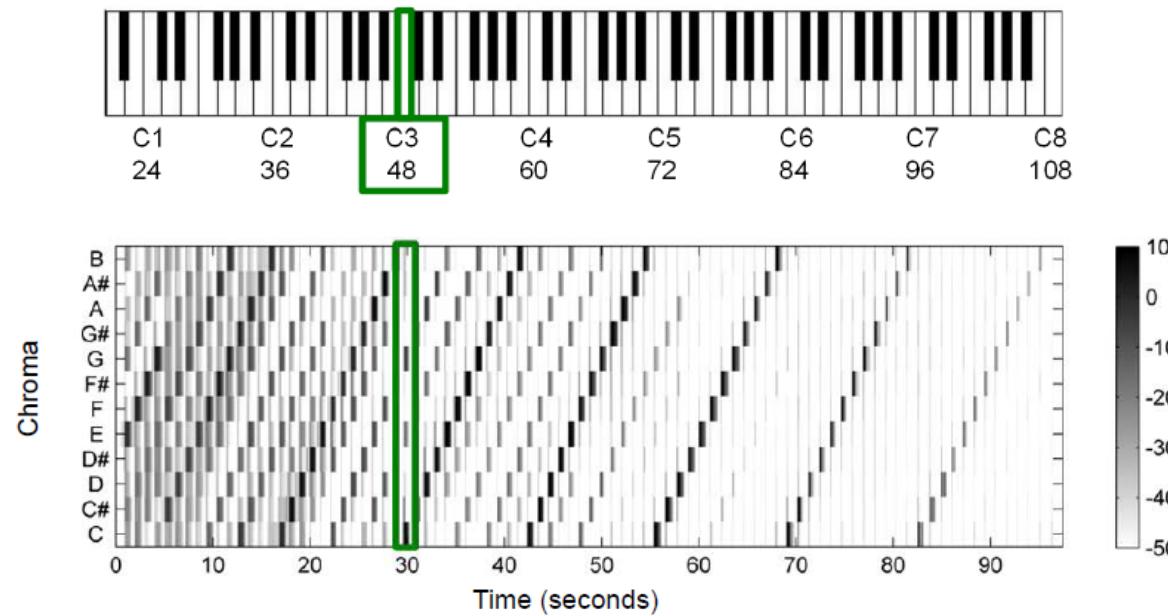
1. Feature representation

- robust but discriminative
- robust to variations in instrumentation, timbre, and dynamics
- correlate to harmonic progression

2. Alignment technique

- deals with local and global tempo variations
- needs to be efficient

Chroma Features (Chromagrams)



- **Timbre-invariant:** 1st, 2nd, 4th, 8th harmonics are in the same chroma band (similarly for 3rd, 6th, 12th harmonics)
- Correlate to harmonic progression

Figure from [Mueller, FPM, Chapter 3, Springer 2015]

Chroma Features

- **Timbre-invariant:** frame-wise normalization reduces differences in energy decay

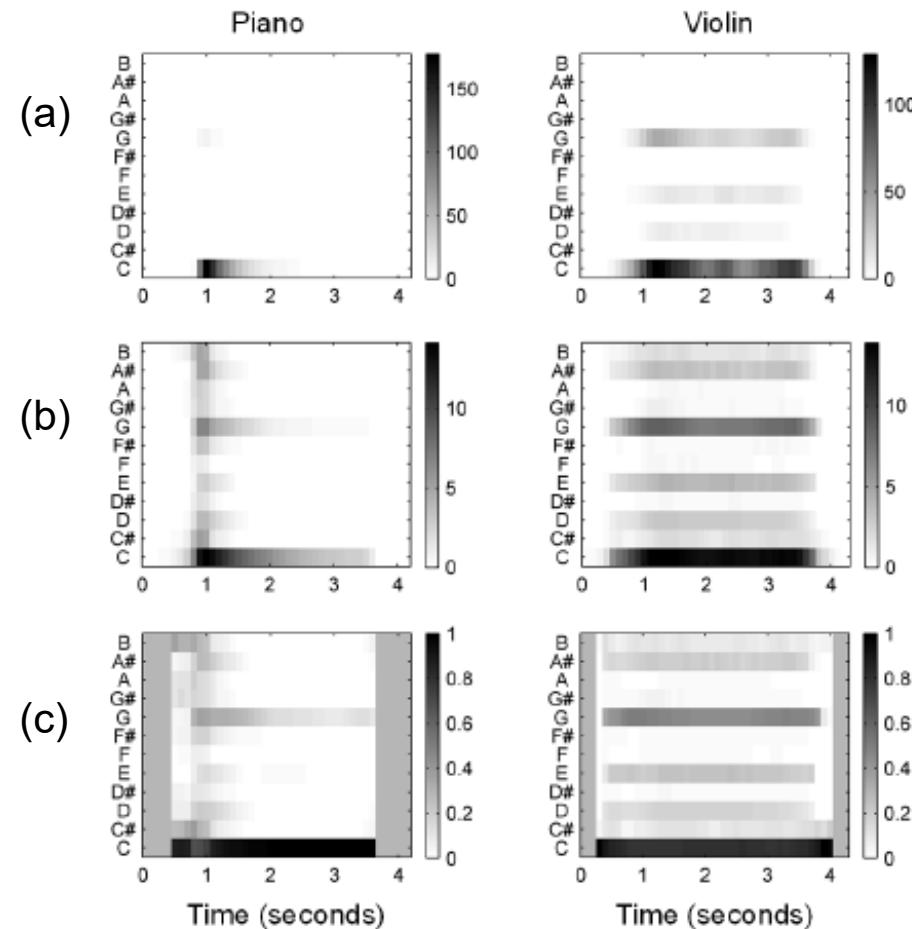


Figure from [Mueller, FPM, Chapter 3, Springer 2015]

Chroma Features

- **Tuning:** global frequency deviation

- a. original chroma
- b. log compression + frame-wise normalization
- c. chroma of a piano tuned 40 cents upwards
- d. chroma after correcting for tuning

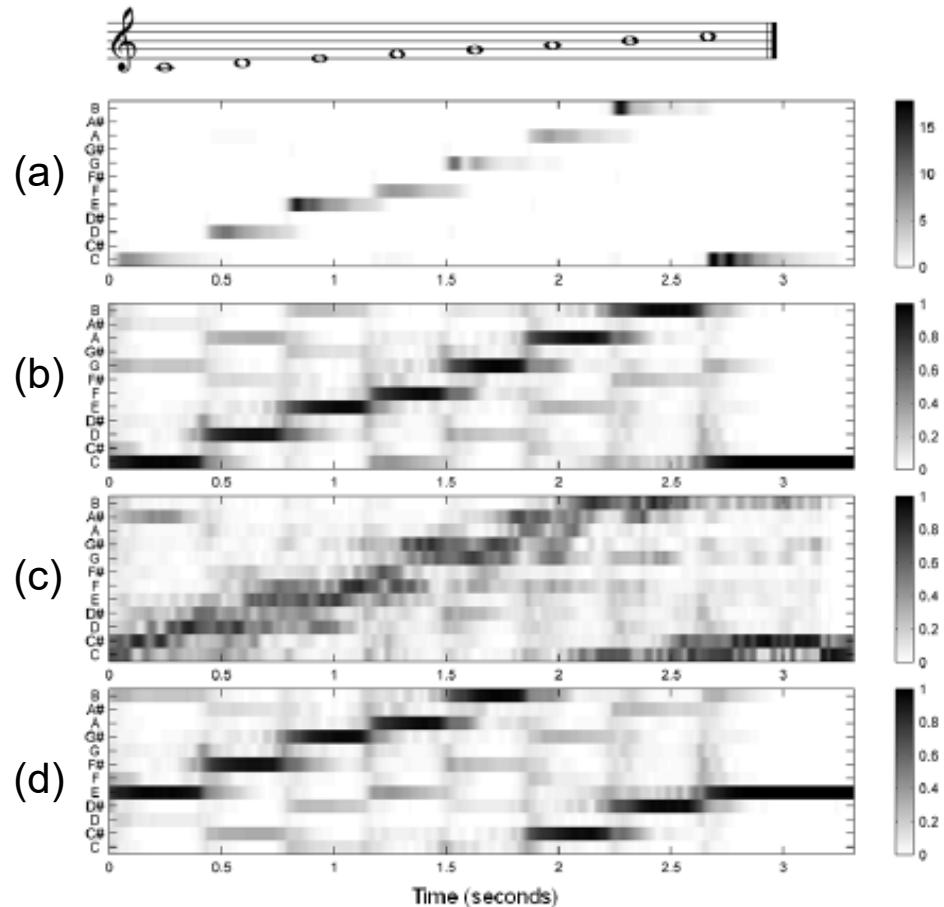


Figure from [Mueller, FPM, Chapter 3, Springer 2015]

Chromagrams for Two Performances

- Compressed and normalized chromagram

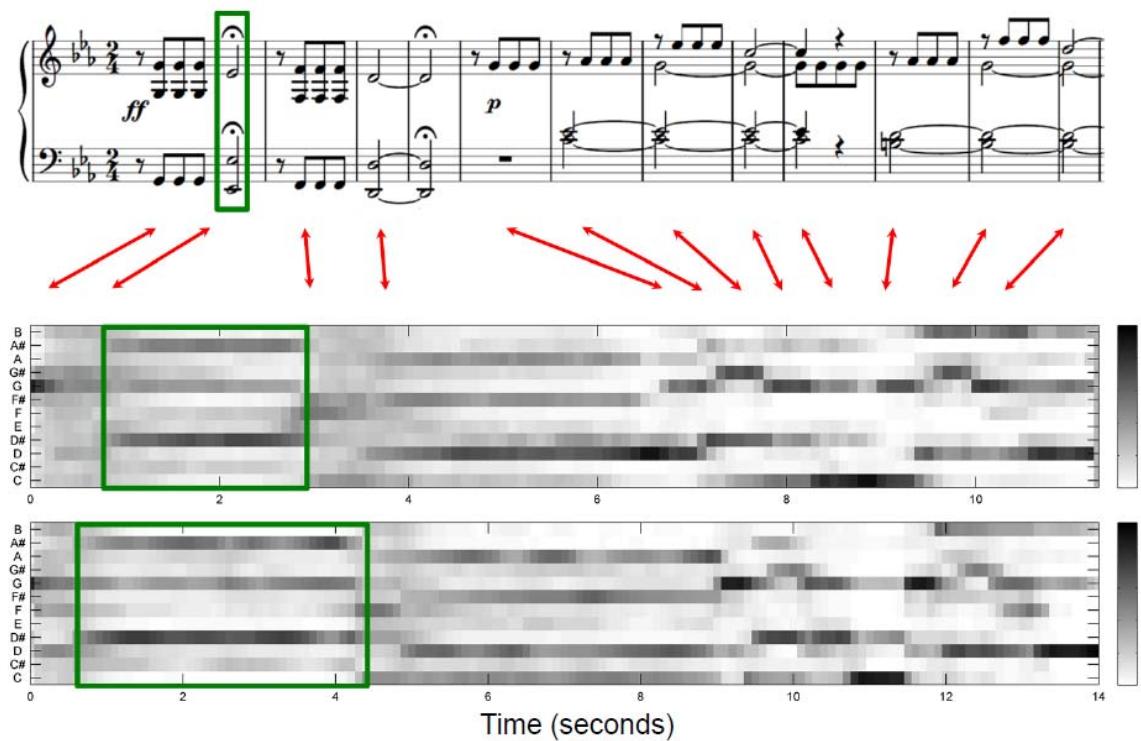


Figure from [Mueller, FPM, Chapter 3, Springer 2015]

Chromagrams for Two Performances

- Further enhanced chromagram (by quantization and smoothing)

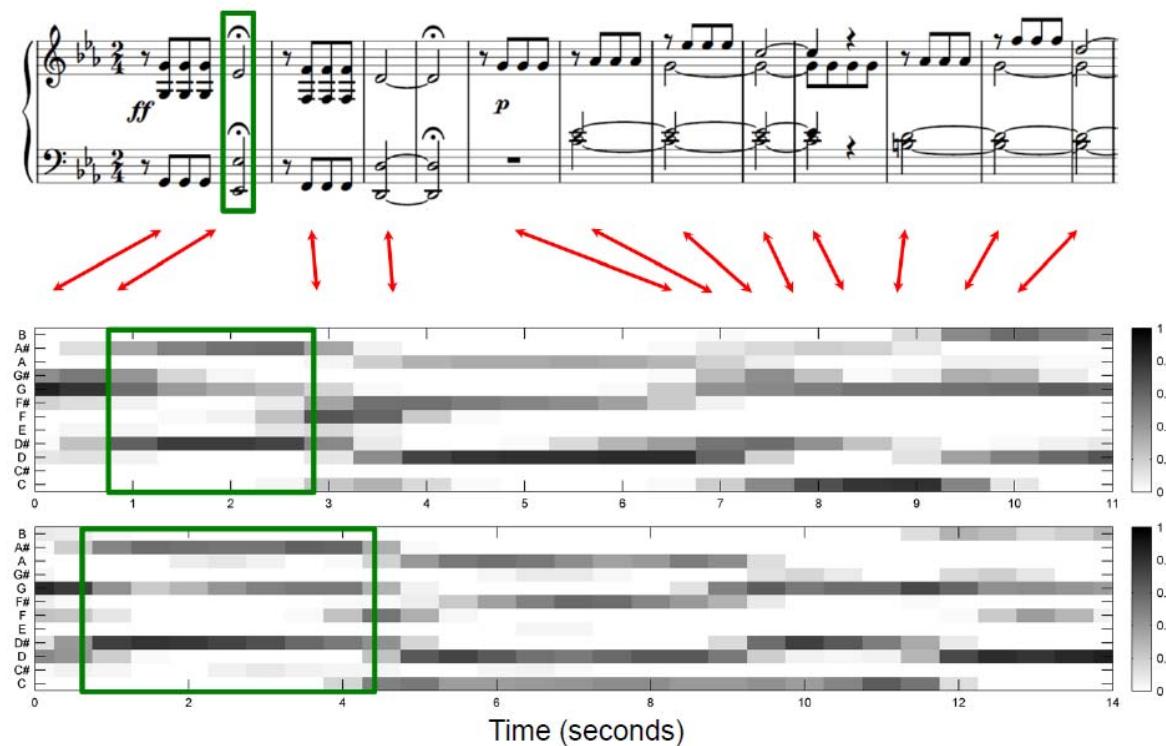


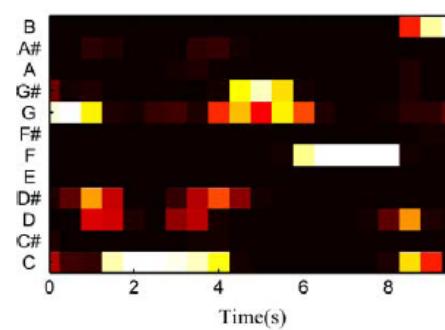
Figure from [Mueller, FPM, Chapter 3, Springer 2015]

Chroma Toolbox (in Matlab)

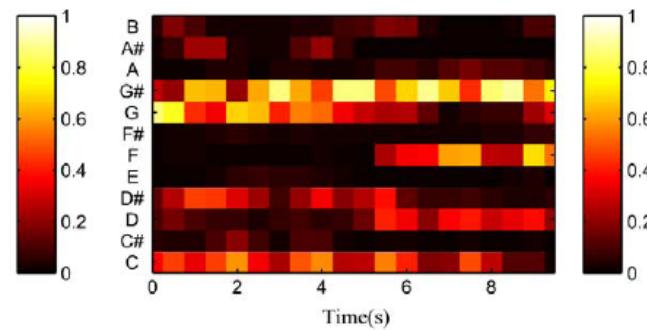
<https://www.audiolabs-erlangen.de/resources/MIR/chromatoolbox/>



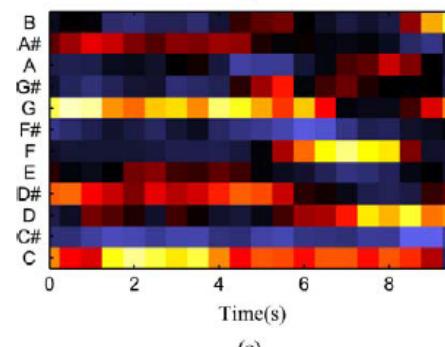
clarinet
(woodwind)



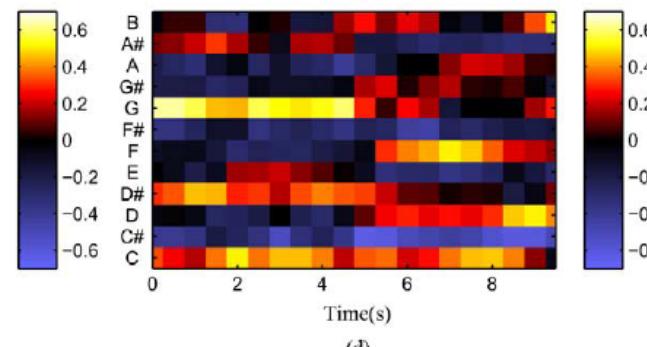
(a)



(b)



(c)



(d)



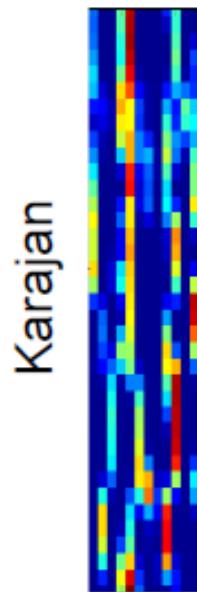
trombone
(brass)

Figure from the
chromatoolbox
paper

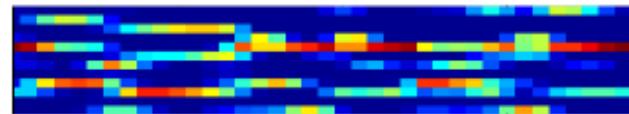
Fig. 4. Various chromograms of the passages E_1 (clarinet) and E_3 (trombone) in the Yablonsky recording of the Shostakovich Waltz. (a)/(b) Conventional chromagram of E_1/E_3 . (c)/(d) CRP(55) chromagram of E_1/E_3 . All chroma vectors are normalized w.r.t. the Euclidean norm.

Music Synchronization: Audio-to-Audio

- Two chromagrams



Karajan



Scherbakov

Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Music Synchronization: Audio-to-Audio

- Compute the frame-by-frame **cost matrix**

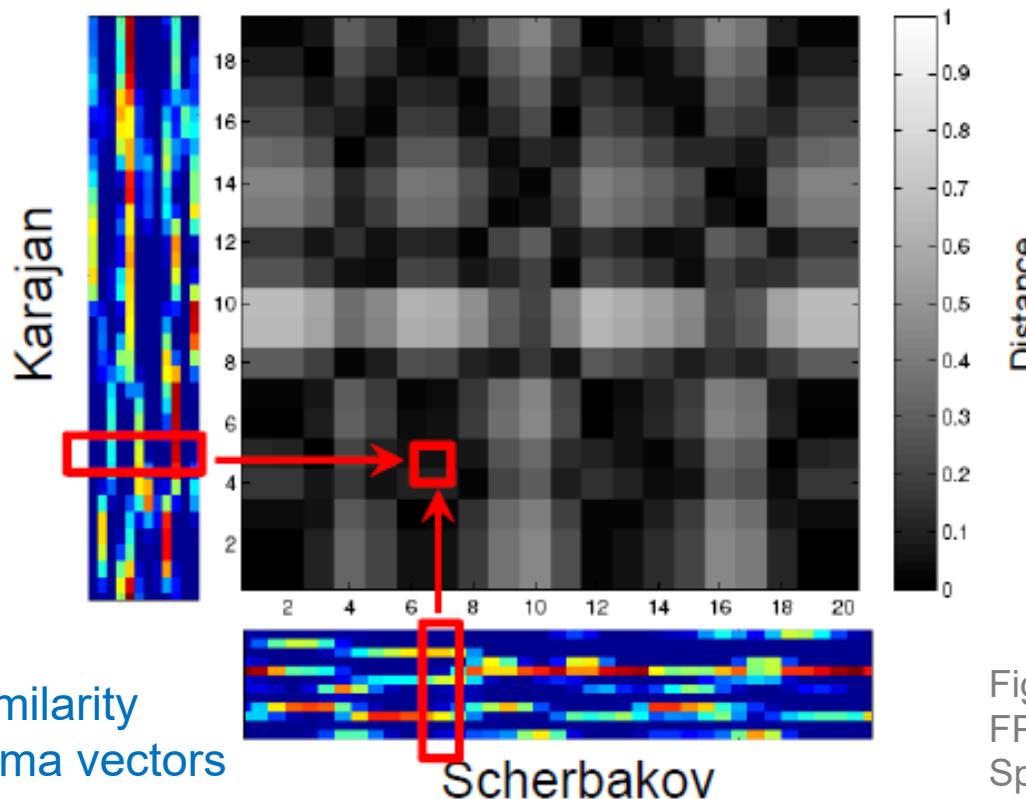


Figure from [Mueller, FPM, Chapter 3, Springer 2015]

Music Synchronization: Audio-to-Audio

- Cost minimizing alignment path

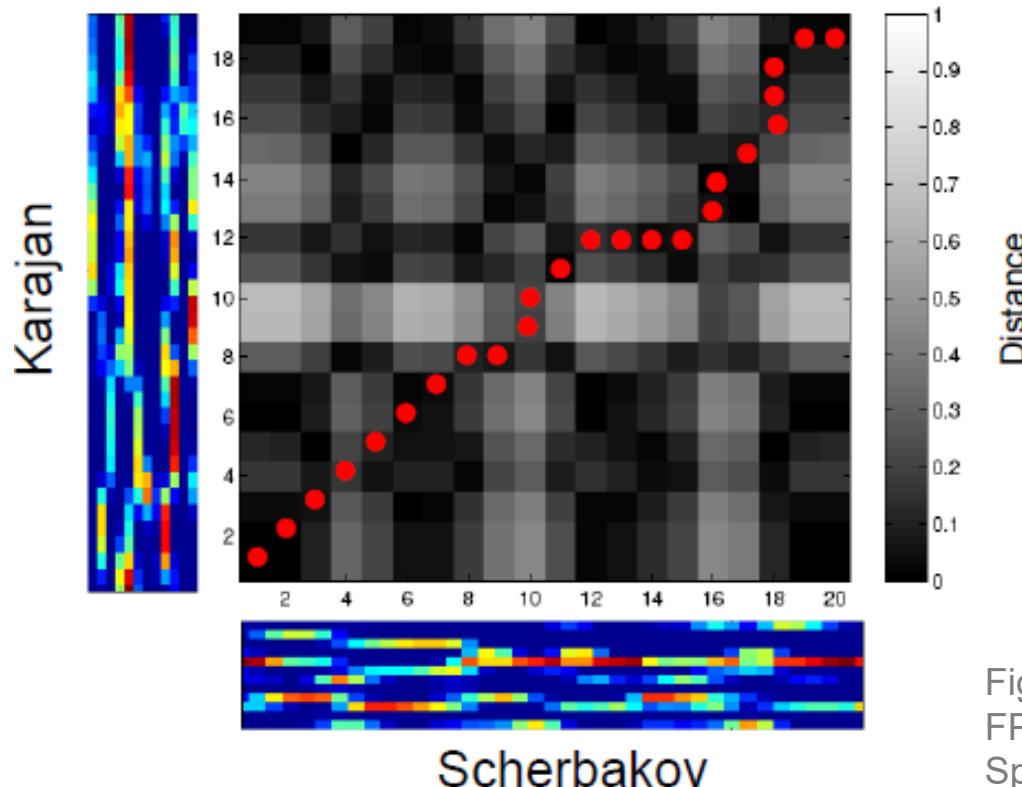


Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Music Synchronization: Audio-to-Audio

- Cost minimizing alignment path

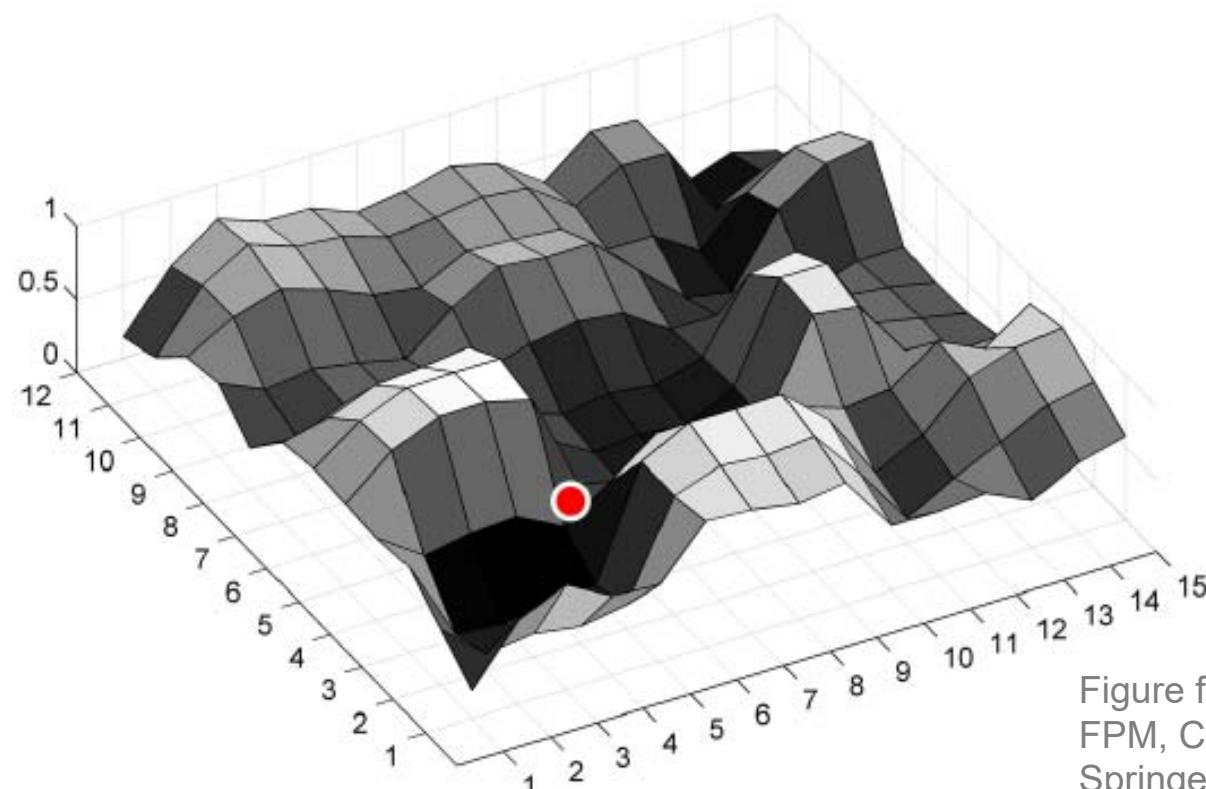


Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Two Main Components of Synchronization

1. Feature representation

- robust but discriminative
- robust to variations in instrumentation, timbre, and dynamics
- correlate to harmonic progression

2. Alignment technique

- deals with local and global tempo variations
- needs to be efficient

How to Compute the Alignment?

- Cost matrix
- Constraints
 - boundary condition
 - monotonically increase
 - step size condition
- Problem formulation
 - given two sequences $X = (x_1, x_2, \dots, x_N)$ and $Y = (y_1, y_2, \dots, y_M)$
 - find a **warping path** $p = (p_1, p_2, \dots, p_L)$ with $p_l = (n_l, m_l) \in [1:N] \times [1:M]$

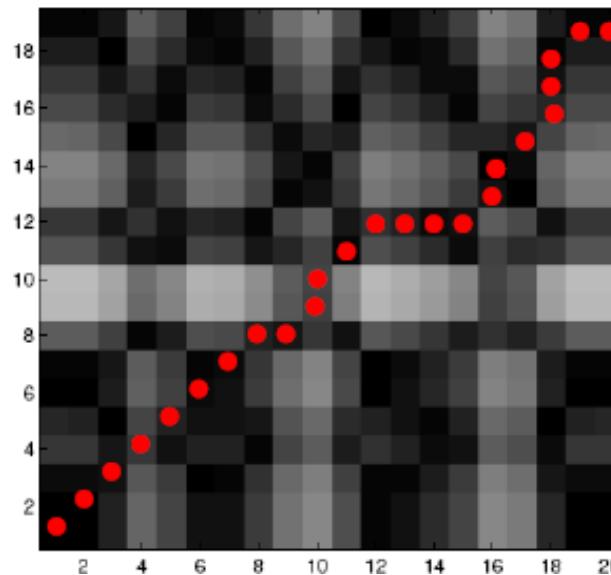
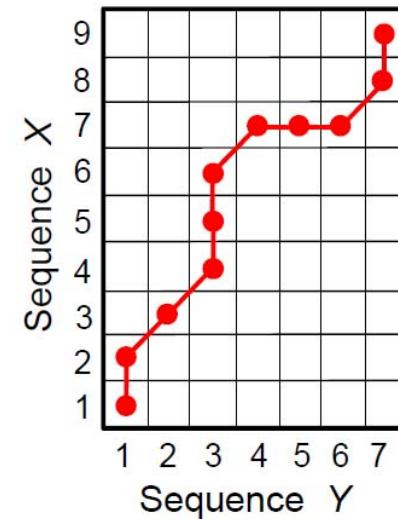
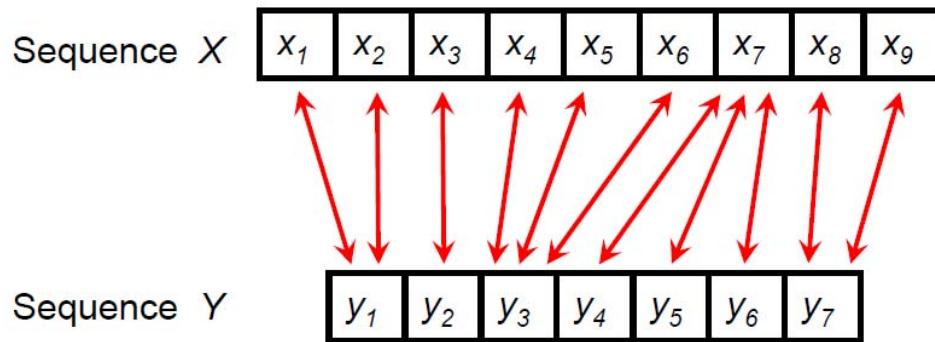


Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Alignment

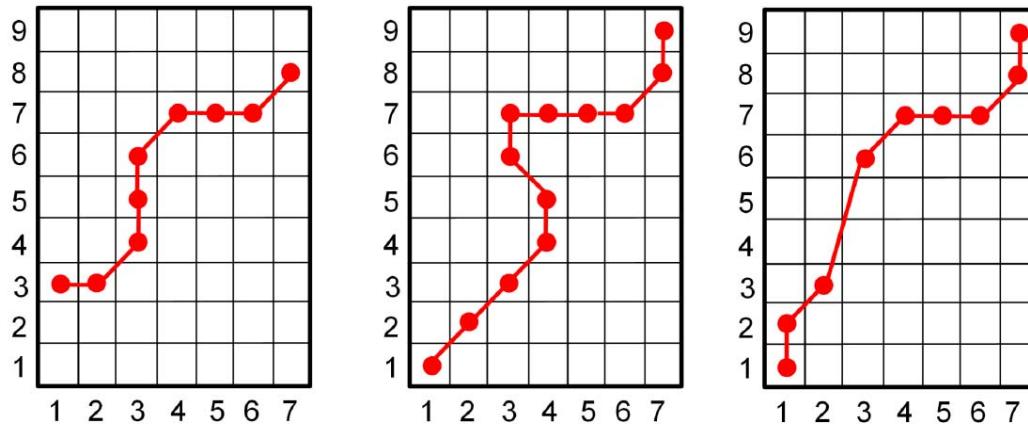


- Boundary condition: $p_1 = (1, 1)$ and $p_L = (N, M)$
- Monotonicity condition: $n_1 \leq n_2 \leq \dots \leq n_L$ and $m_1 \leq m_2 \leq \dots \leq m_L$
- Step size condition: $p_{\ell+1} - p_\ell \in \{(1, 0), (0, 1), (1, 1)\}$ for $\ell \in [1 : L - 1]$

Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Alignment

- Invalid alignments



- Boundary condition: $p_1 = (1, 1)$ and $p_L = (N, M)$
- Monotonicity condition: $n_1 \leq n_2 \leq \dots \leq n_L$ and $m_1 \leq m_2 \leq \dots \leq m_L$
- Step size condition: $p_{\ell+1} - p_\ell \in \{(1, 0), (0, 1), (1, 1)\}$ for $\ell \in [1 : L - 1]$

Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Dynamic Time Warping

- Dynamic programming
 - optimal sub-structure
 - overlapping sub-problems
- Complexity
 $O(NM)$
- Trace back

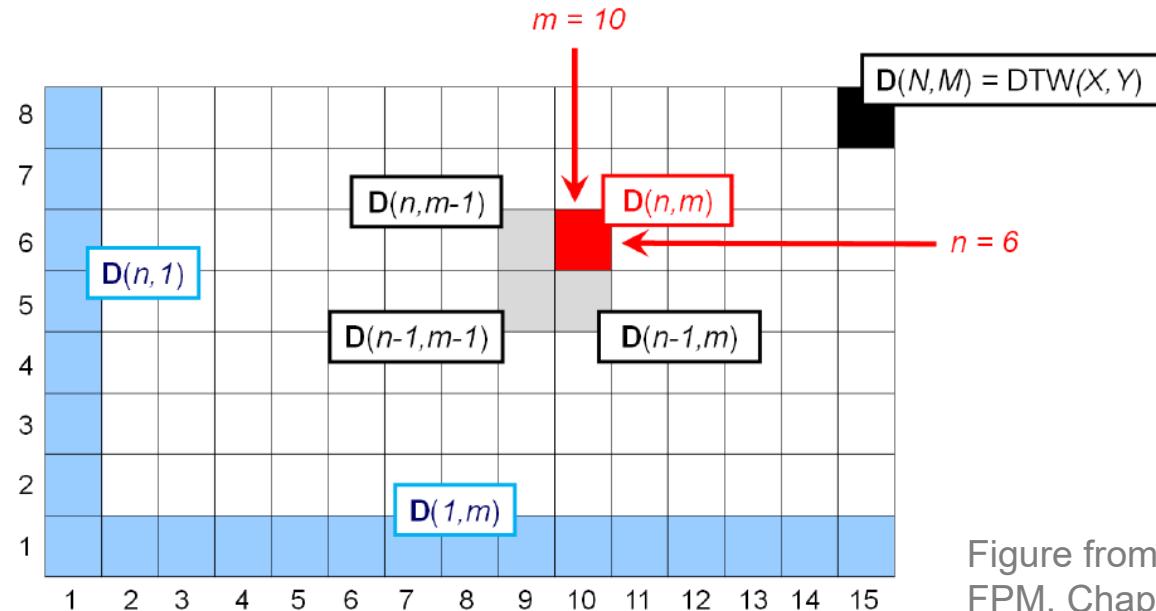


Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Dynamic Time Warping

$$(i) \quad D(N, M) = \text{DTW}(X, Y)$$

$$(ii) \quad D(1, 1) = C(1, 1)$$

$$(iii) \quad \begin{aligned} D(n, 1) &= \sum_{k=1}^n C(k, 1) \\ D(1, m) &= \sum_{k=1}^m C(1, k) \end{aligned}$$

$$(iv) \quad D(n, m) = \min \begin{pmatrix} D(n-1, m-1) \\ D(n-1, m) \\ D(n, m-1) \end{pmatrix} + C(n, m)$$

for $n > 1, m > 1$

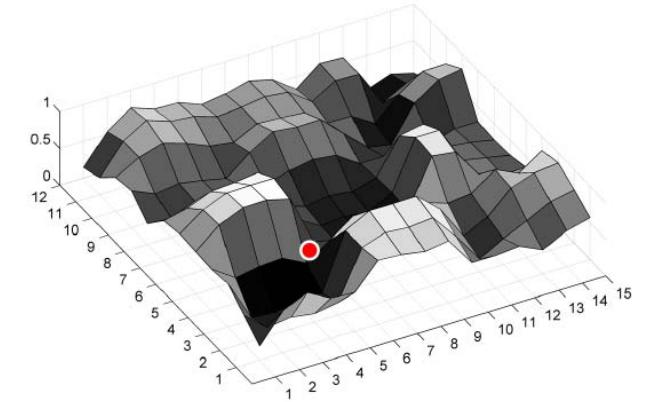


Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Dynamic Time Warping

Algorithm: DTW

Input: Cost matrix \mathbf{C} of size $N \times M$

Output: Accumulated cost matrix \mathbf{D}
Optimal warping path P^*

Procedure: Initialize $(N \times M)$ matrix \mathbf{D} by $\mathbf{D}(n, 1) = \sum_{k=1}^n \mathbf{C}(k, 1)$ for $n \in [1 : N]$ and $\mathbf{D}(1, m) = \sum_{k=1}^m \mathbf{C}(1, k)$ for $m \in [1 : M]$. Then compute in a nested loop for $n = 2, \dots, N$ and $m = 2, \dots, M$:

$$\mathbf{D}(n, m) = \mathbf{C}(n, m) + \min \{\mathbf{D}(n - 1, m - 1), \mathbf{D}(n - 1, m), \mathbf{D}(n, m - 1)\}.$$

Set $\ell = 1$ and $q_\ell = (N, M)$. Then repeat the following steps until $q_\ell = (1, 1)$:

Increase ℓ by one and let $(n, m) = q_{\ell-1}$.

If $n = 1$, then $q_\ell = (1, m - 1)$,

else if $m = 1$, then $q_\ell = (n - 1, 1)$,

else $q_\ell = \operatorname{argmin} \{\mathbf{D}(n - 1, m - 1), \mathbf{D}(n - 1, m), \mathbf{D}(n, m - 1)\}$.

(If ‘argmin’ is not unique, take lexicographically smallest cell.)

Set $L = \ell$ and return $P^* = (q_L, q_{L-1}, \dots, q_1)$ as well as \mathbf{D} .

From [Mueller, FPM,
Chapter 3, Springer 2015]

Dynamic Time Warping: Example

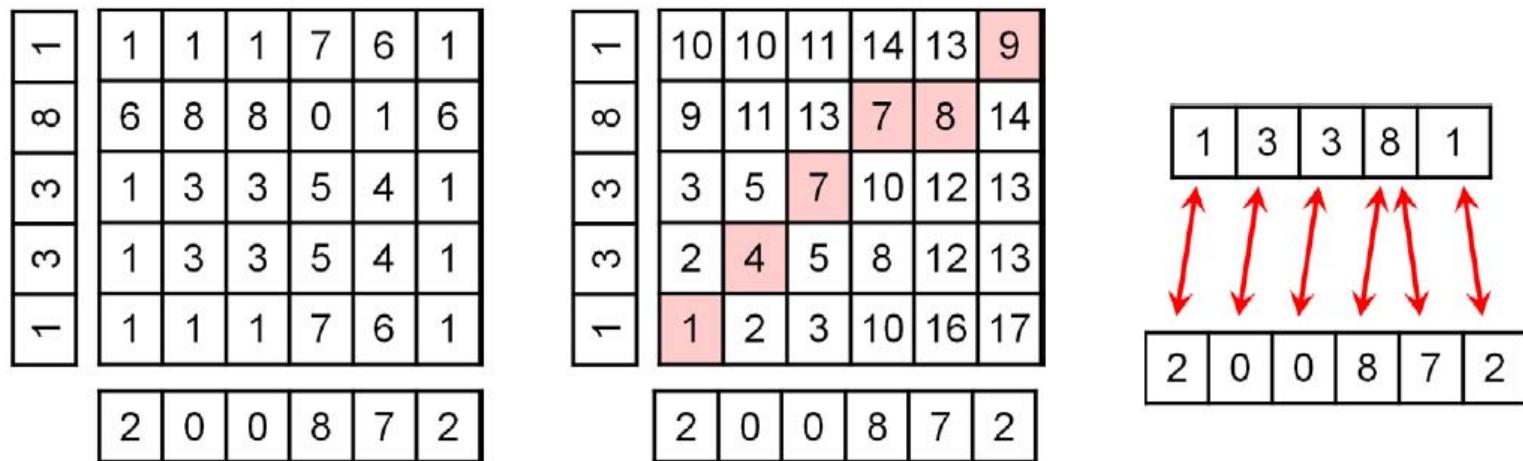


Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Dynamic Time Warping: Step Size

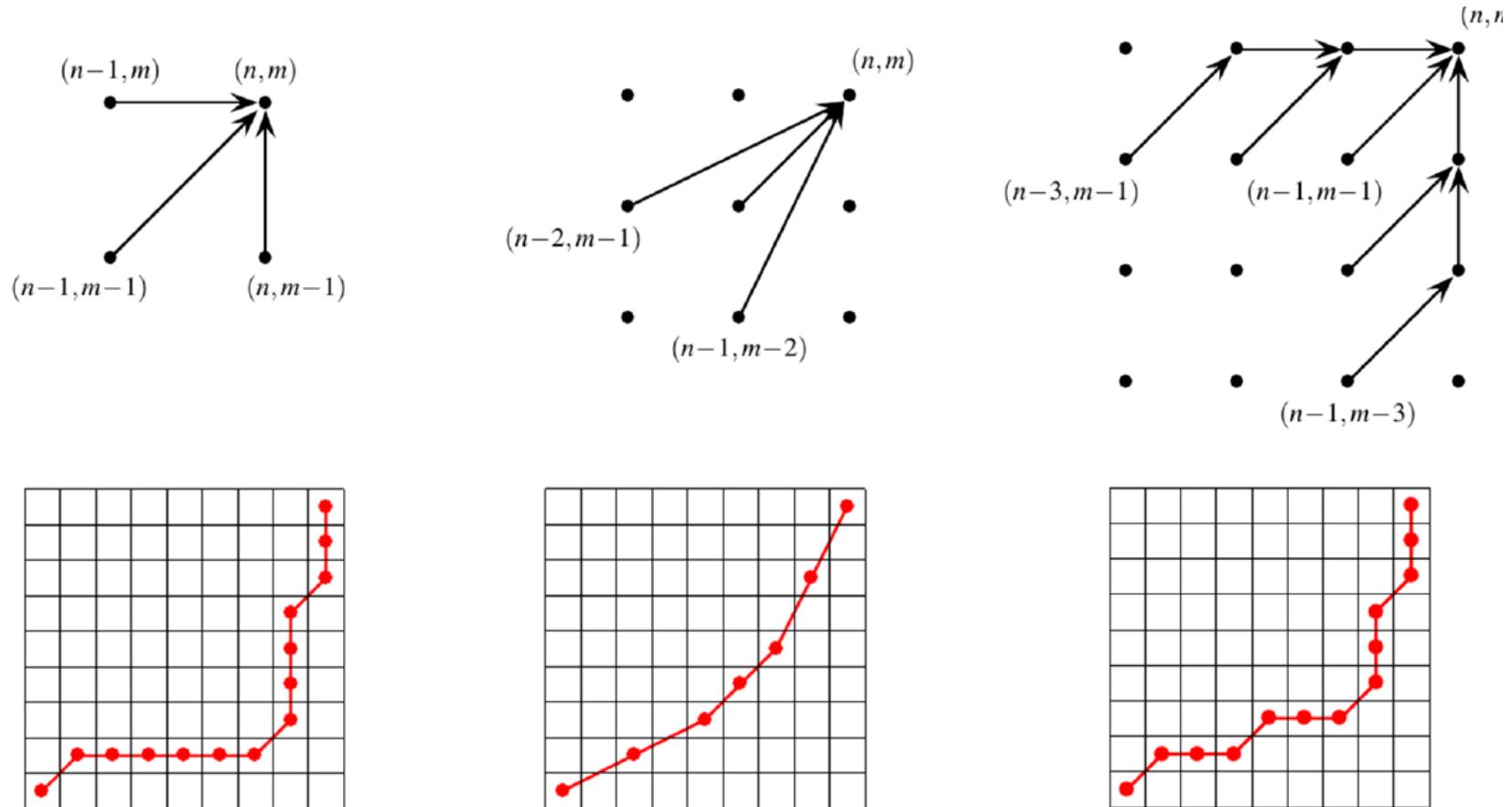
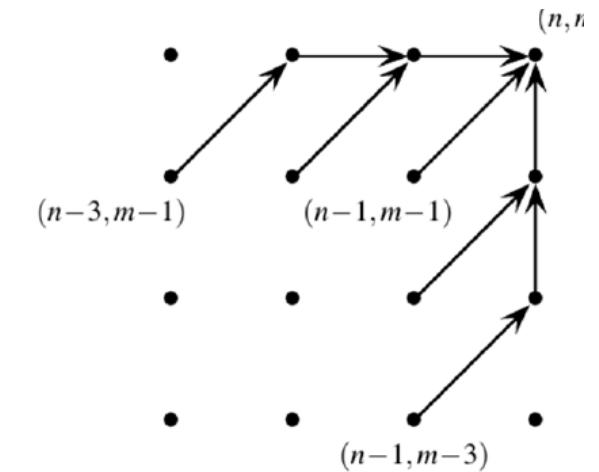


Figure from [Mueller, FPM, Chapter 3, Springer 2015]

Dynamic Time Warping: Step Size

$$D(n, m) = C(n, m) +$$

$$\min \left(\begin{array}{l} D(n-1, m-1) \\ D(n-2, m-1) + C(n-1, m) \\ D(n-1, m-2) + C(n, m-1) \\ D(n-3, m-1) + C(n-2, m) + C(n-1, m) \\ D(n-1, m-3) + C(n, m-2) + C(n, m-1) \end{array} \right)$$



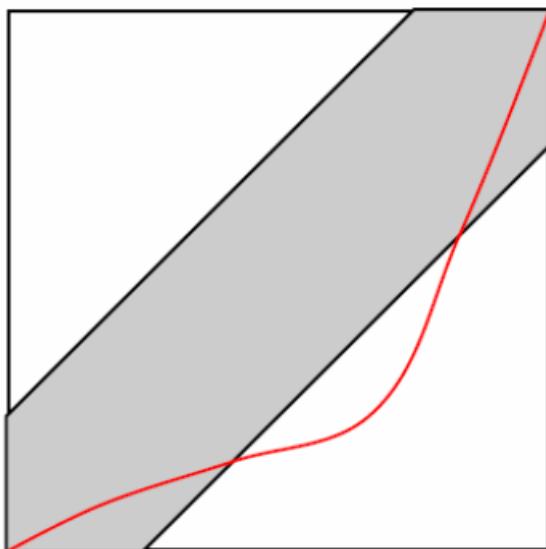
- Another variant (e.g. $w_d = 2$, $w_v = w_h = 1$):

$$D(n, m) = \min \left(\begin{array}{l} D(n-1, m-1) + \textcolor{blue}{w_d} C(n, m) \\ D(n-1, m) + \textcolor{blue}{w_v} C(n, m) \\ D(n, m-1) + \textcolor{blue}{w_h} C(n, m) \end{array} \right)$$

Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Dynamic Time Warping: Global Constraint

Sakoe-Chiba band



Itakura parallelogram

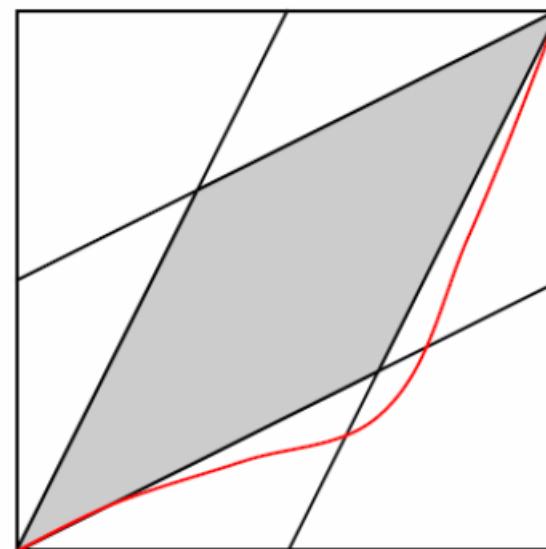


Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Dynamic Time Warping: Multi-resolution

- Coarse → fine

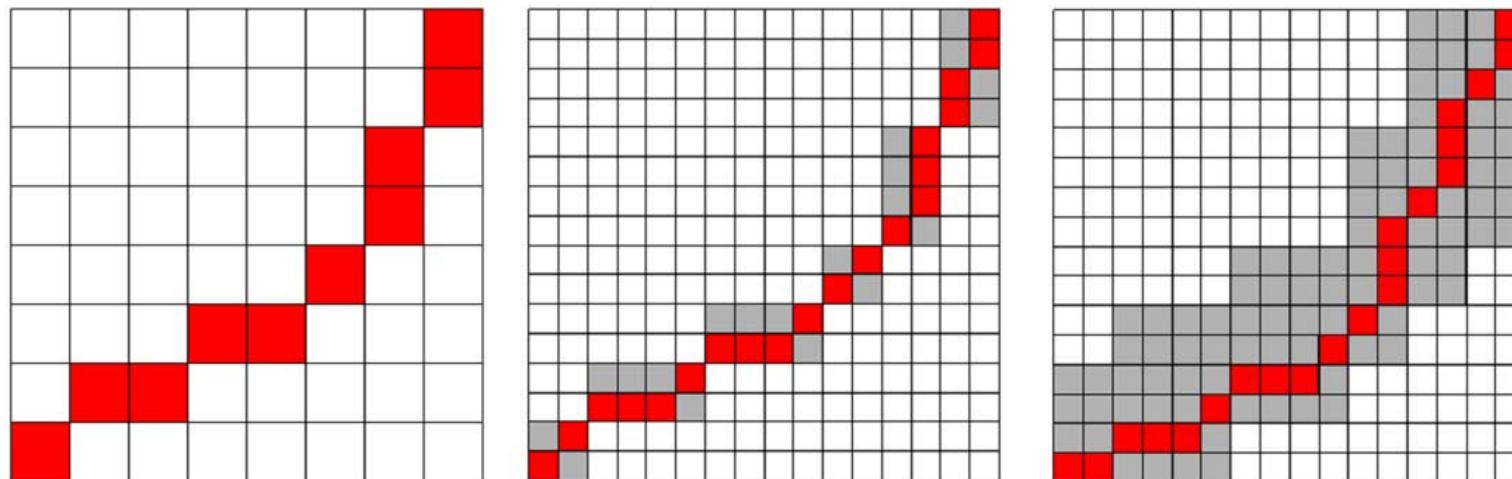


Figure from [Mueller,
FPM, Chapter 3,
Springer 2015]

Score Following: Model the Tempo Variation

- Dynamical system

- Position:

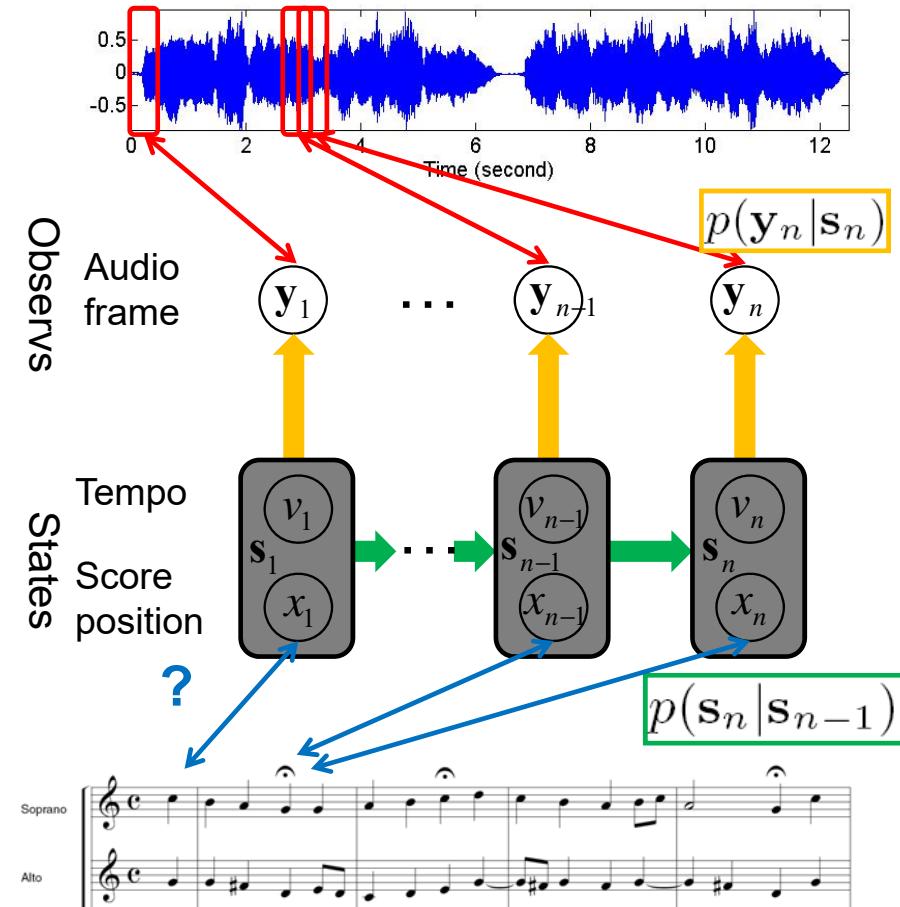
$$x_n = x_{n-1} + l \cdot v_{n-1}$$

- Tempo:

$$v_n = \begin{cases} v_{n-1} + n_v \\ v_{n-1} \end{cases}$$

where

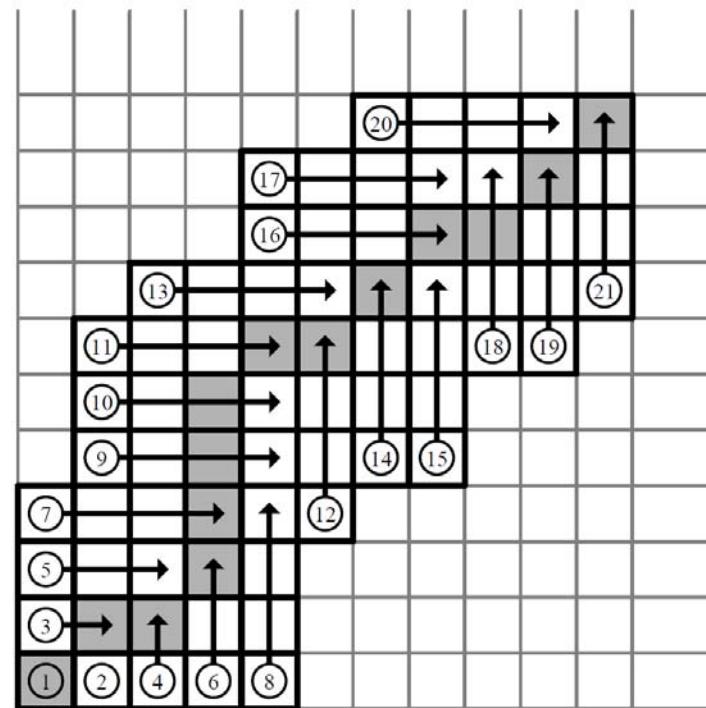
$$n_v \sim \mathcal{N}(0, \sigma_v^2)$$



Ref: Duan et al, "Soundprism: An online system for score-informed source separation of music audio," JSTSP 2011

Score Following: Online DTW

- Key changes
 - compute the path incrementally
 - reduce the complexity to being linear w.r.t. length of the input
 - reconsider past decisions to increase robustness

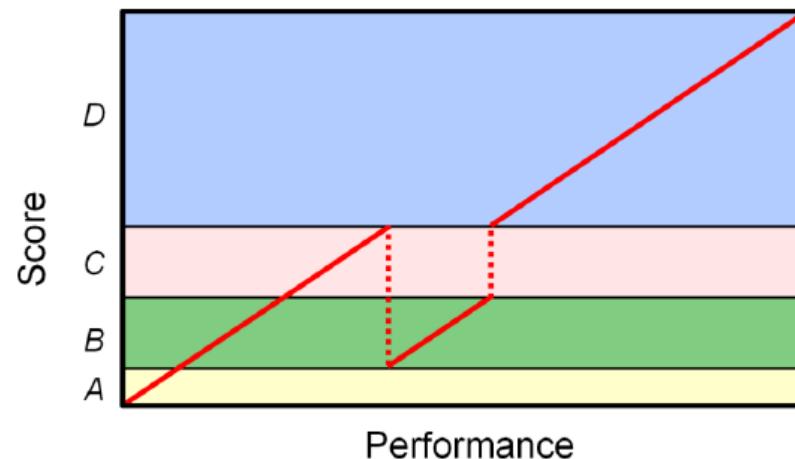


Ref 1: Dixon, "Live tracking of musical performances using on-line time warping," DAFX 2005

Ref 2: Arzt et al, "Automatic page turning for musicians via real-time machine listening," ECAI 2008

Score Following: Advanced Issues

- Errors, repeats, skips
- Structure difference between the score and the performance (i.e. repeated sections)
- Multiple instruments

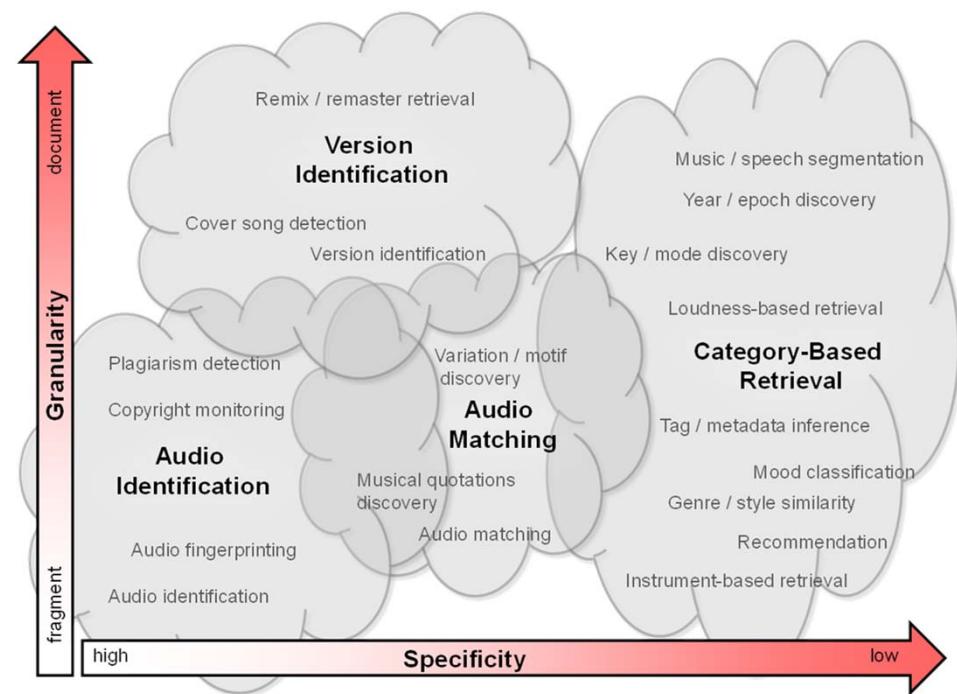
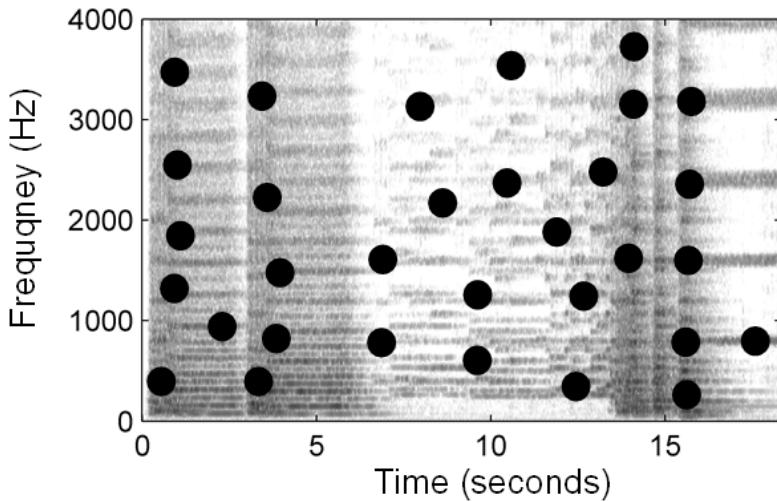


Ref 1: Fremerey et al, "Handling repeats and jumps in score performance synchronization," ISMIR 2010

Ref 2: Grachten et al, "Automatic alignment of music performances with structural differences," ISMIR 2013

Extension: Query-by-humming and Friends

- **Query-by-humming:** monophonic
- **Audio fingerprinting:** polyphonic, possibly noisy
- **Cover song identification**



Tool: Sync Toolbox

<https://github.com/meinardmueller/synctoolbox>

Sync Toolbox

This repository contains a Python package called Sync Toolbox, which provides open-source reference implementations for full-fledged music synchronization pipelines and yields state-of-the-art alignment results for a wide range of Western music.

Using suitable feature representations and cost measures, the toolbox's core technology is based on dynamic time warping (DTW), which brings the feature sequences into temporal correspondence. To account for efficiency, robustness and, accuracy, our toolbox integrates and combines techniques such as [multiscale DTW \(MsDTW\)](#), [memory-restricted MsDTW \(MrMsDTW\)](#), and [high-resolution music synchronization](#).

Ref: Müller et al, “Sync Toolbox: A Python package for efficient, robust, and accurate music synchronization,” JOSS 2011

Tool: Matchmaker

<https://github.com/pymatchmaker/matchmaker>

Matchmaker

Matchmaker is a Python library for real-time music alignment.

Music alignment is a fundamental MIR task, and real-time music alignment is a necessary component of many interactive applications (e.g., automatic accompaniment systems, automatic page turning).

Unlike offline alignment methods, for which state-of-the-art implementations are publicly available, real-time (online) methods have no standard implementation, forcing researchers and developers to build them from scratch for their projects.

We aim to provide efficient reference implementations of score followers for use in real-time applications which can be easily integrated into existing projects.

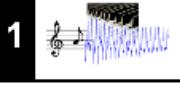
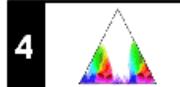
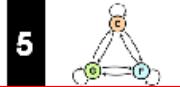
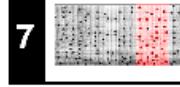
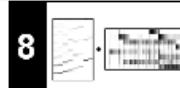
Ref: Park et al, “Matchmaker: A Python library for real-time music alignment,” ISMIR-LBD 20125

Outline

- Music emotion
- Structure/form analysis
- Alignment
- **Rhythm & beat tracking**

Reference 1: FMP Notebook

<https://www.audiolabs-erlangen.de/resources/MIR/FMP/C6/C6.html>

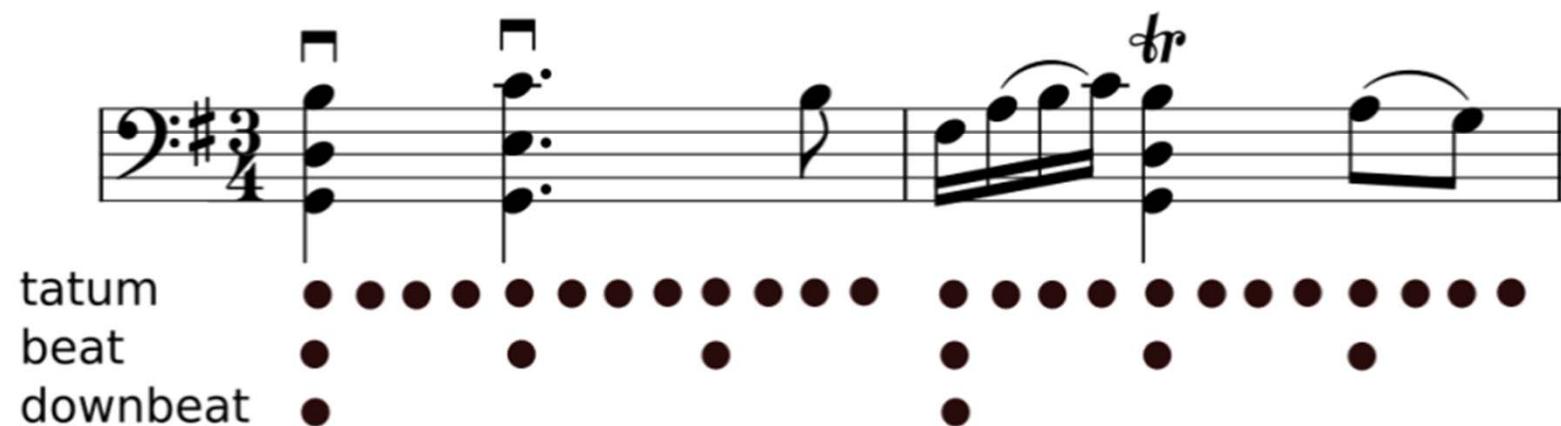
| Part | Title | Notions, Techniques & Algorithms | HTML | IPYNB |
|------|--|--|------------------------|-------------------------|
| B |  Basics | Basic information on Python, Jupyter notebooks, Anaconda package management system, Python environments, visualizations, and other topics | [html] | [ipynb] |
| 0 |  Overview | Overview of the notebooks (https://www.audiolabs-erlangen.de/FMP) | [html] | [ipynb] |
| 1 |  Music Representations | Music notation, MIDI, audio signal, waveform, pitch, loudness, timbre | [html] | [ipynb] |
| 2 |  Fourier Analysis of Signals | Discrete/analog signal, sinusoid, exponential, Fourier transform, Fourier representation, DFT, FFT, STFT | [html] | [ipynb] |
| 3 |  Music Synchronization | Chroma feature, dynamic programming, dynamic time warping (DTW), alignment, user interface | [html] | [ipynb] |
| 4 |  Music Structure Analysis | Similarity matrix, repetition, thumbnail, homogeneity, novelty, evaluation, precision, recall, F-measure, visualization, scape plot | [html] | [ipynb] |
| 5 |  Chord Recognition | Harmony, music theory, chords, scales, templates, hidden Markov model (HMM), evaluation | [html] | [ipynb] |
| 6 |  Tempo and Beat Tracking | Onset, novelty, tempo, tempogram, beat, periodicity, Fourier analysis, autocorrelation | [html] | [ipynb] |
| 7 |  Content-Based Audio Retrieval | Identification, fingerprint, indexing, inverted list, matching, version, cover song | [html] | [ipynb] |
| 8 |  Musically Informed Audio Decomposition | Harmonic/percussive separation, signal reconstruction, instantaneous frequency, fundamental frequency (F0), trajectory, nonnegative matrix factorization (NMF) | [html] | [ipynb] |

Reference 2: ISMIR 2021 Tutorial

<https://tempobeatdownbeat.github.io/tutorial/intro.html>

Tempo, Beat, and Downbeat Estimation

By Matthew E. P. Davies, Sebastian Böck, Magdalena Fuentes



What is Beat/Downbeat Tracking?

https://tempobeatdownbeat.github.io/tutorial/ch2_basics/definition.html

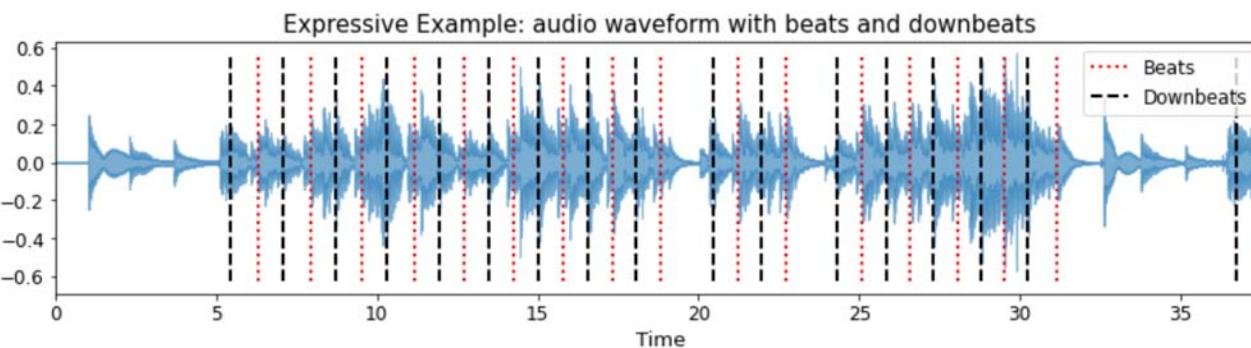
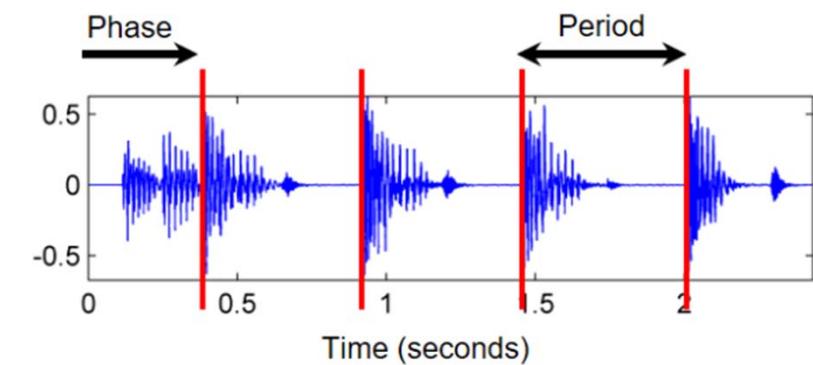
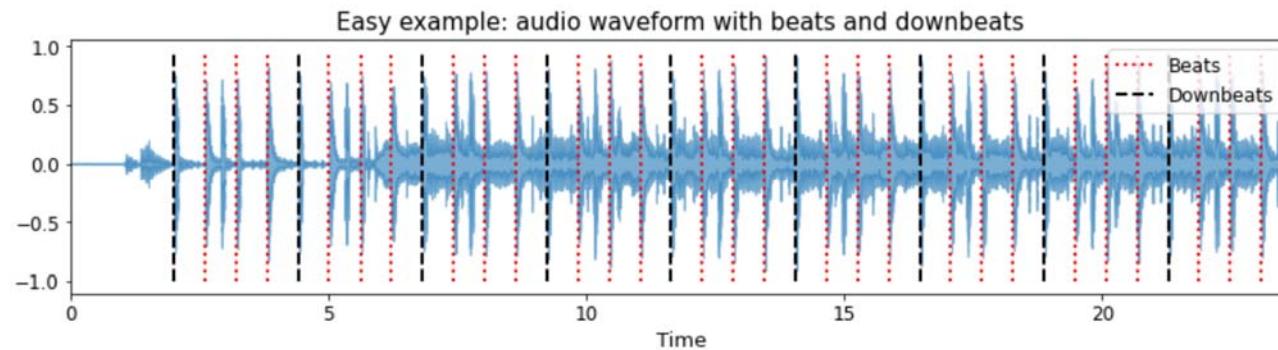
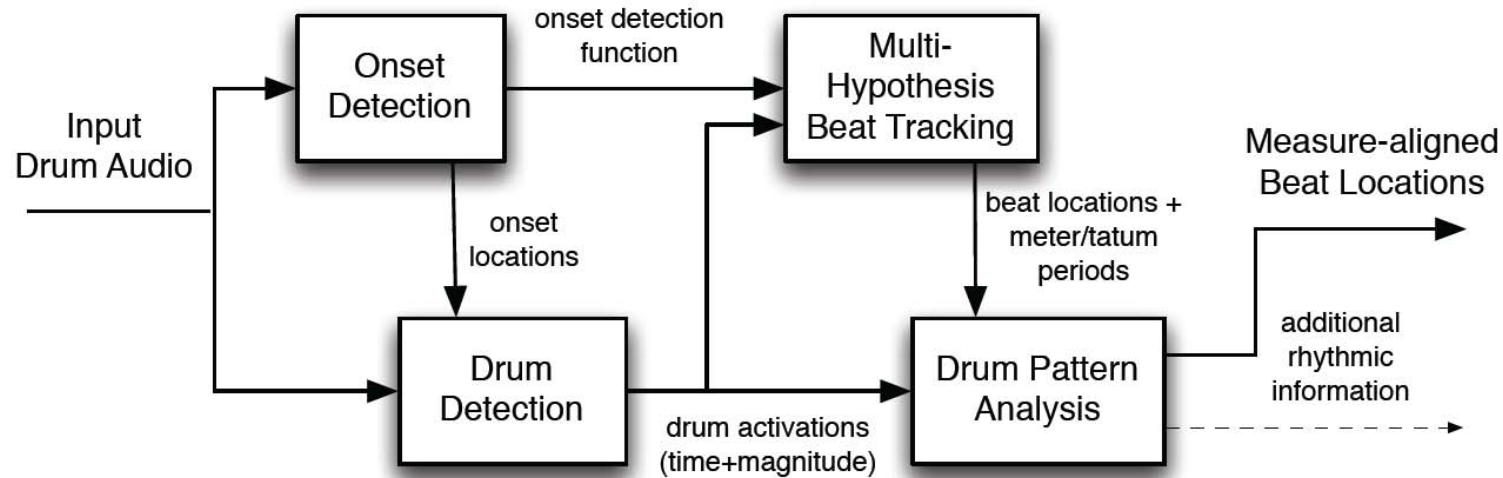


Figure 6.1 from [Müller, FMP, Springer 2015]

Motivation: Beat, Tempo, Rhythmic Pattern

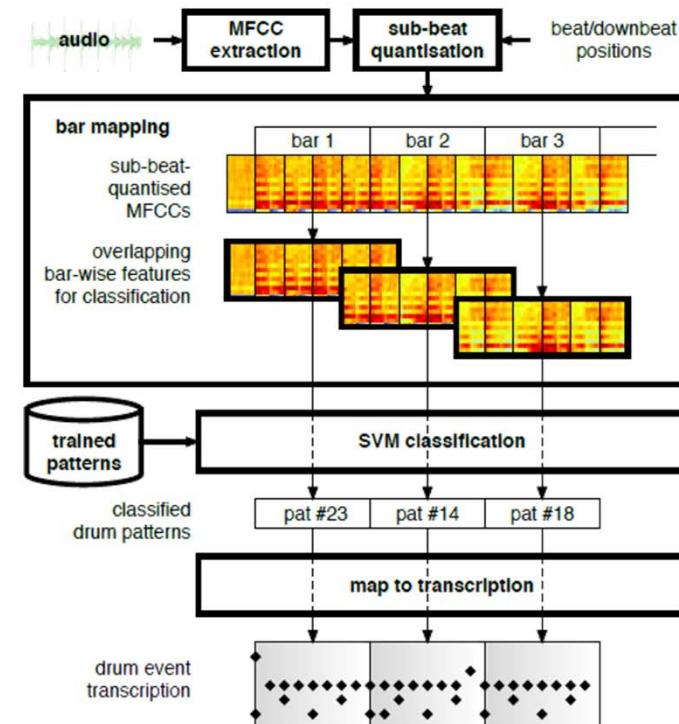
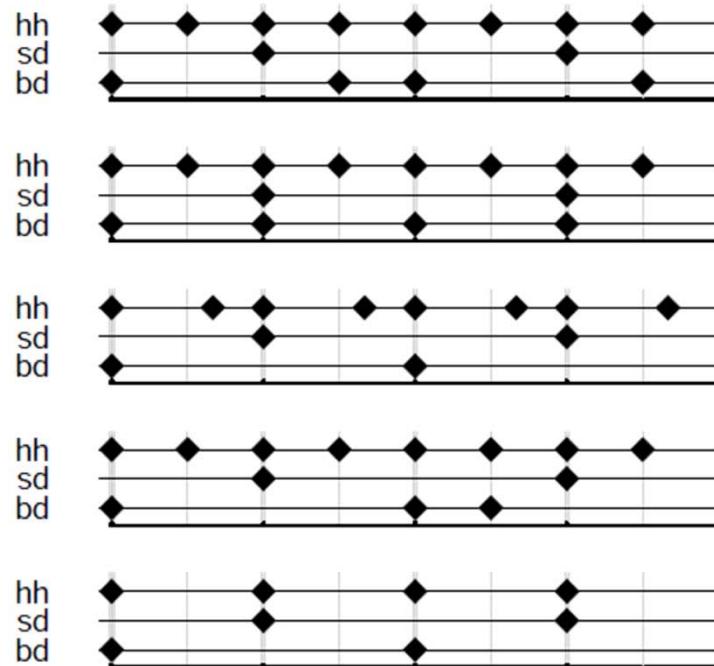
- The drum track in popular music conveys information about tempo, rhythm, style, and possibly the structure of a song



Ref: Battenberg, "Techniques for machine understanding of live drum performances," PhD Thesis, UC Berkeley, 2012

Motivation: Drum Pattern Analysis

https://youtu.be/lm_oz7P8TWE?t=12m28s

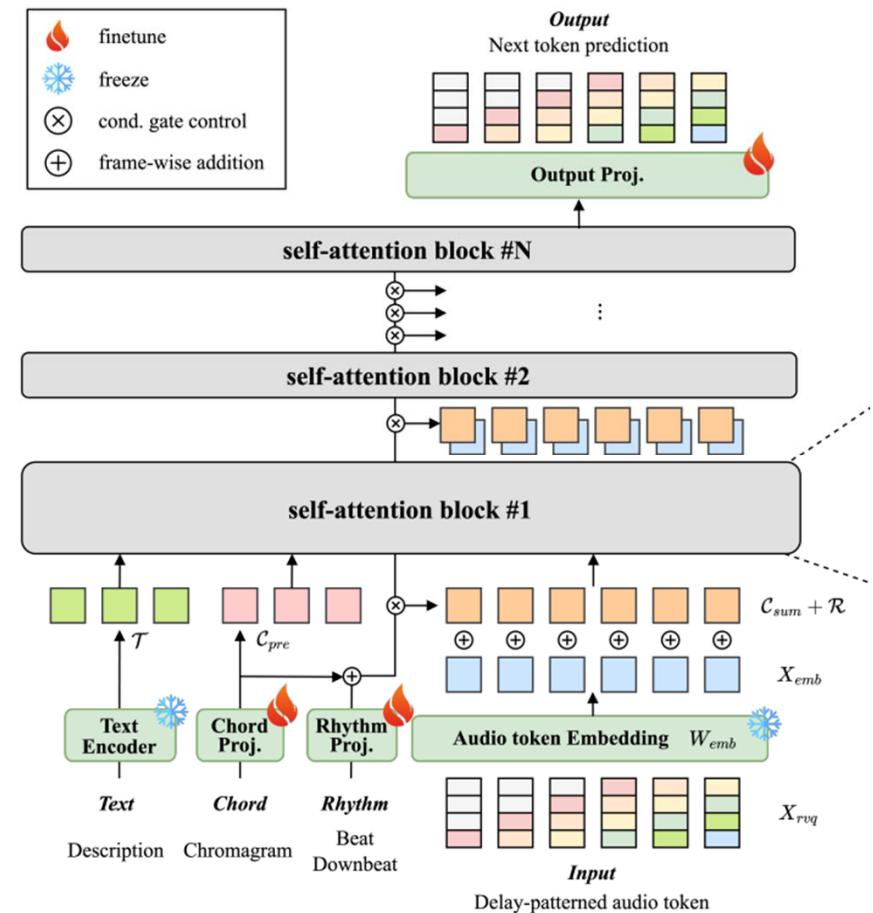
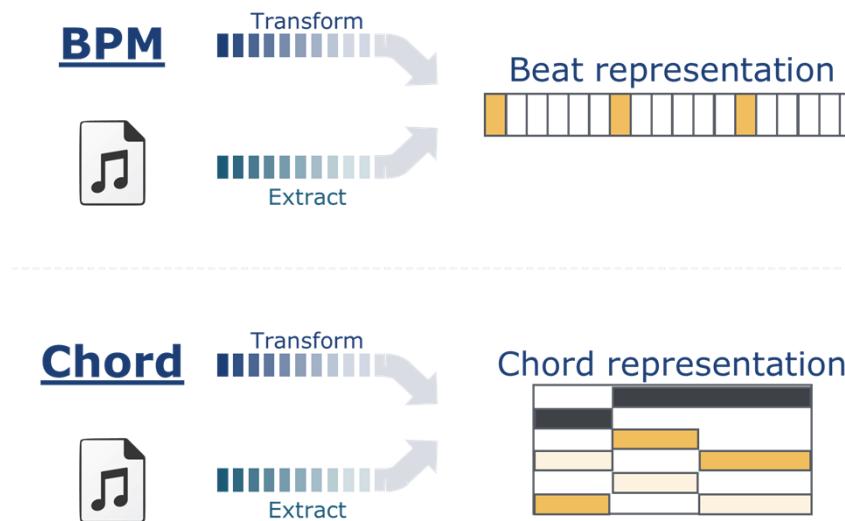


Ref 1: Mauch & Dixon, "A corpus-based study of rhythm patterns," ISMIR 2012

Ref 2: Thompson et al, "Drum transcription via classification of bar-level rhythmic patterns," ISMIR 2014

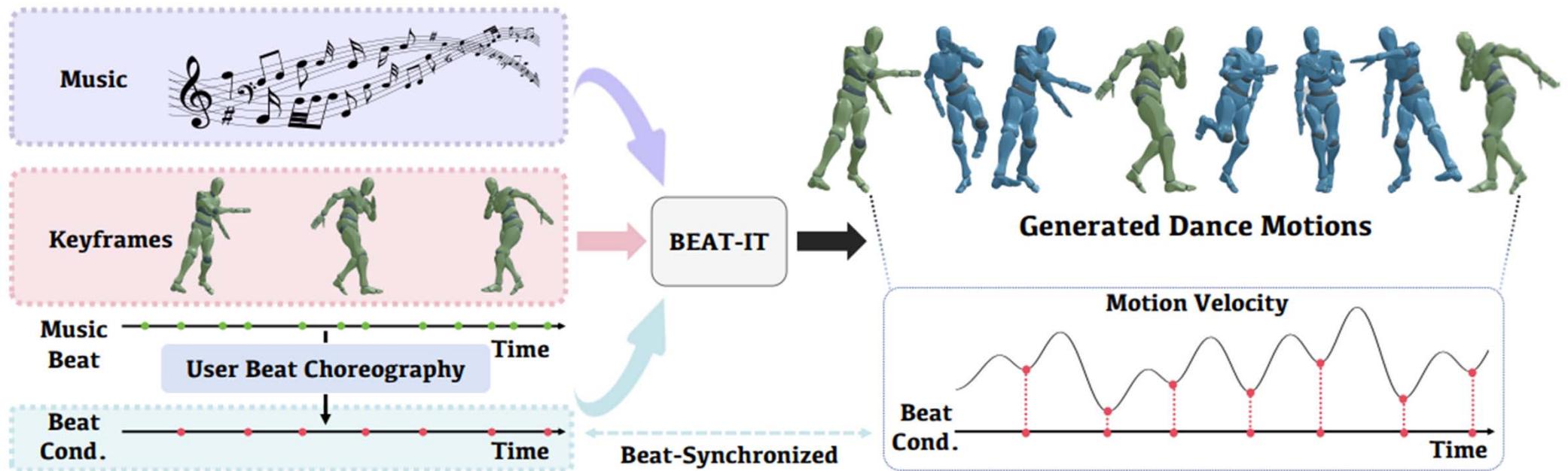
Motivation: Backing Track Generation

[https://musicongen.github.io/
musicongen_demo/](https://musicongen.github.io/musicongen_demo/)



Ref: Lan et al, "MusiConGen: Rhythm and chord control for transformer-based text-to-music generation," ISMIR 2024

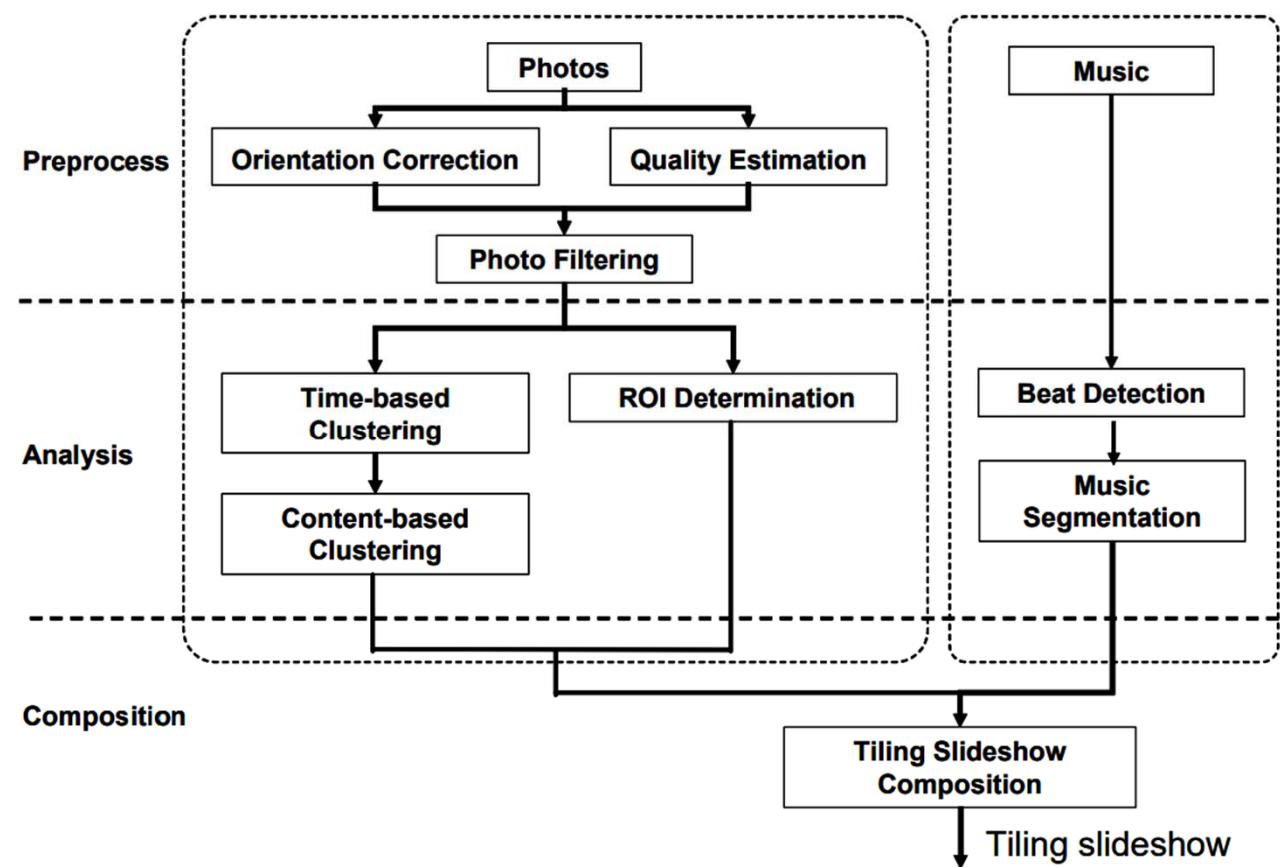
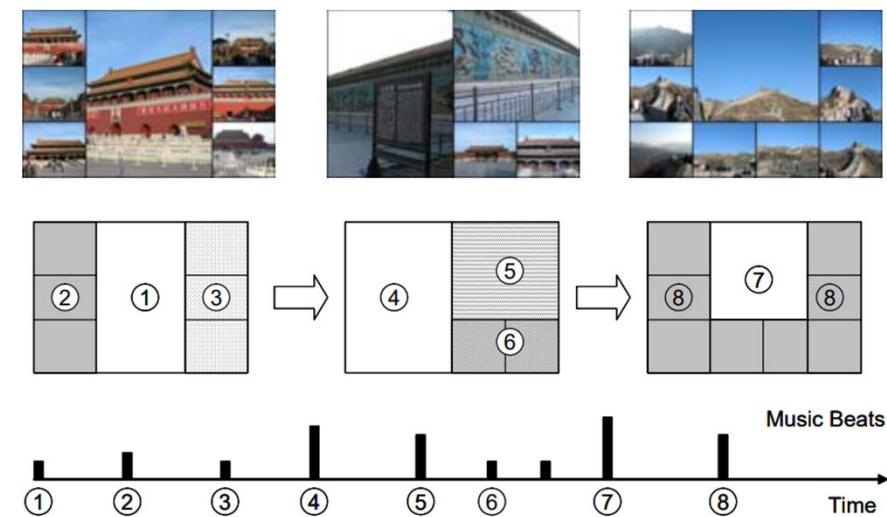
Motivation: Dance Generation



Ref: Huang et al, “Beat-It: Beat-synchronized multi-condition 3d dance generation,” ECCV 2024

135

Motivation: Music Video/Slideshow Generation



Ref: Chen et al, "Tiling slideshow," ACM Multimedia 2006

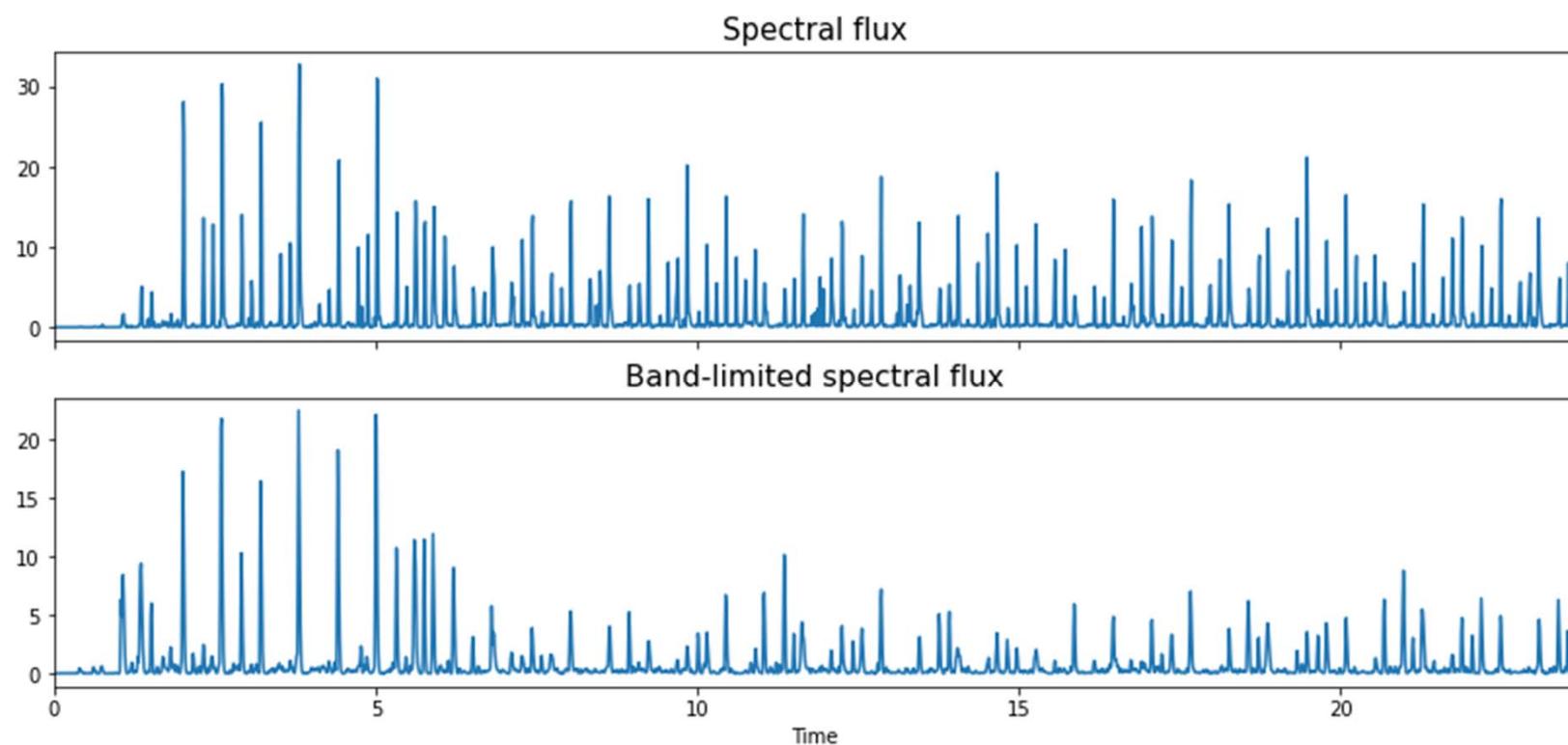
Beat/Downbeat Annotation

https://tempobeatdownbeat.github.io/tutorial/ch2_basics/annotation.html

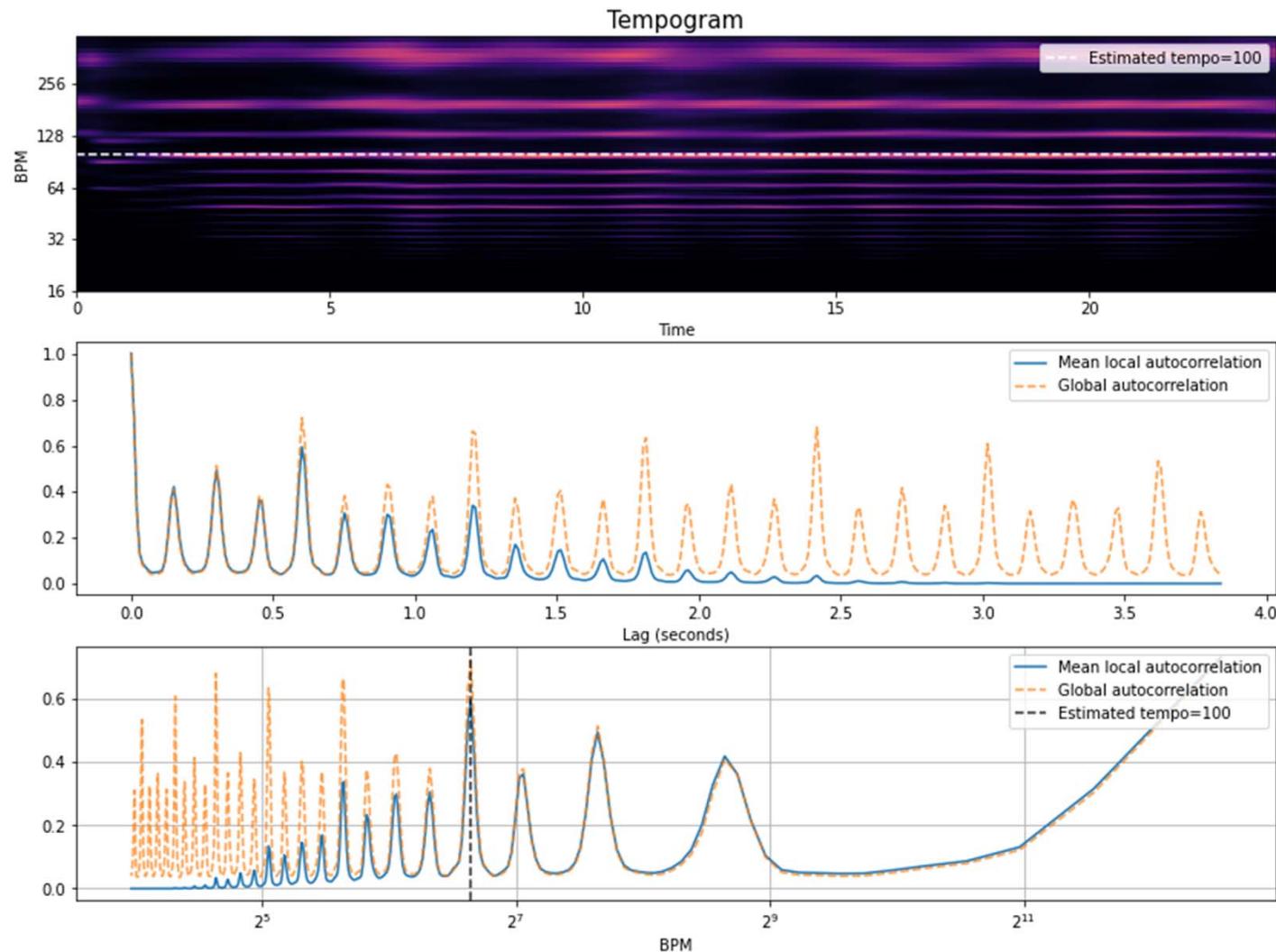
- Annotation is hard!
- It takes a long time, and the more challenging the material to annotate the greater the likelihood of this being helpful for learning.
- On the plus side, annotation is a fantastic way to learn about the task of beat and downbeat estimation so it's a really great exercise.
- We always need more data, so do consider doing some annotating!
- As hard as we try, annotation "mistakes" are made, so they made need correcting.
- This makes comparative evaluation more challenging, so it's always worthwhile to ensure you are using the most up to date version of any annotations.

Baseline Approach

https://tempobeatdownbeat.github.io/tutorial/ch2_basics/baseline.html



Baseline Approach

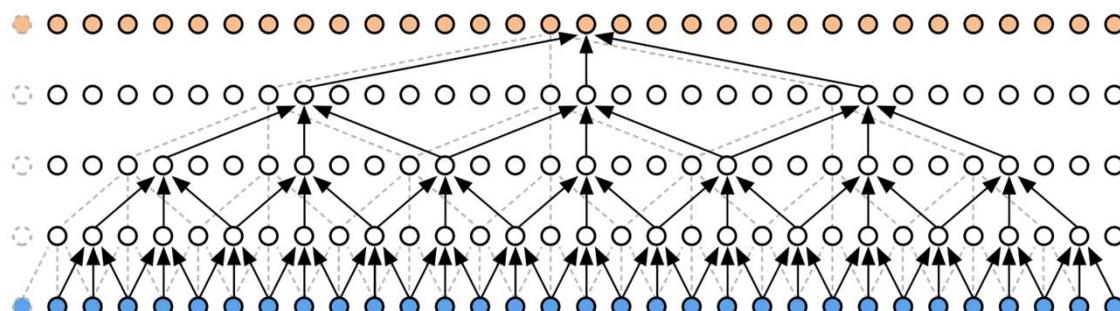


Deep Learning Approaches

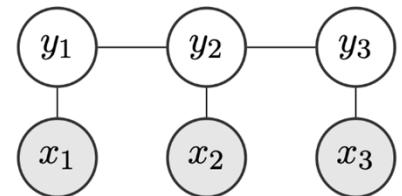
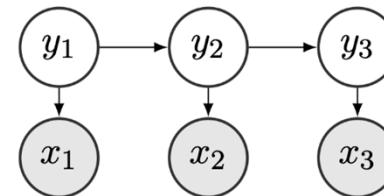
https://tempobeatdownbeat.github.io/tutorial/ch3_going_deep/overview.html



https://tempobeatdownbeat.github.io/tutorial/ch3_going_deep/dnns.html



https://tempobeatdownbeat.github.io/tutorial/ch3_going_deep/postprocessing.html



Hands on!

https://colab.research.google.com/drive/1tuOqNyO9gdMmYjsj33fP_QOfpRsm2tmt?usp=sharing

| | <i>F-measure</i> | <i>CMLt</i> | <i>AMLt</i> | | <i>F-measure</i> | <i>CMLt</i> | <i>AMLt</i> |
|-----------------------|------------------|-------------|-------------|----------------------------|------------------|-------------|-------------|
| <i>Ballroom</i> | | | | | | | |
| Böck et al. [28] | 0.938 | 0.892 | 0.953 | Böck et al. [28] | 0.863 | 0.834 | 0.931 |
| Elowsson [51] ‡ | 0.925 | 0.903 | 0.932 | Durand et al. [8] | 0.797 | 0.616 | 0.916 |
| Davies and Böck [22] | 0.933 | 0.881 | 0.929 | Fuentes et al. [9] | 0.83 | - | - |
| Ours (beat tracking) | 0.956 | 0.935 | 0.958 | Ours (sequential tracking) | 0.900 | 0.894 | 0.953 |
| Ours (joint tracking) | 0.962 | 0.947 | 0.961 | Ours (joint tracking) | 0.916 | 0.913 | 0.960 |
| <i>Hainsworth</i> | | | | | | | |
| Böck et al. [5] | 0.884 | 0.808 | 0.916 | Böck et al. [28] | 0.684 | 0.628 | 0.832 |
| Elowsson [51] ‡ | 0.742 | 0.676 | 0.792 | Durand et al. [8] | 0.664 | 0.500 | 0.804 |
| Davies and Böck [22] | 0.874 | 0.795 | 0.930 | Fuentes et al. [9] | 0.67 | - | - |
| Ours (beat tracking) | 0.904 | 0.851 | 0.937 | Ours (sequential tracking) | 0.713 | 0.686 | 0.855 |
| Ours (joint tracking) | 0.902 | 0.848 | 0.930 | Ours (joint tracking) | 0.722 | 0.696 | 0.872 |
| <i>SMC</i> | | | | | | | |
| Böck et al. [5] | 0.529 | 0.428 | 0.567 | <i>Beatles</i> | | | |
| Elowsson [51] ‡ | 0.375 | 0.225 | 0.332 | Böck et al. [28] | 0.831 | 0.730 | 0.858 |
| Davies and Böck [22] | 0.543 | 0.432 | 0.632 | Durand et al. [8] | 0.847 | 0.722 | 0.875 |
| Ours (beat tracking) | 0.552 | 0.465 | 0.643 | Fuentes et al. [9] | 0.86 | - | - |
| Ours (joint tracking) | 0.544 | 0.443 | 0.635 | Ours (sequential tracking) | 0.829 | 0.748 | 0.860 |
| <i>GTZAN</i> | | | | | | | |
| Böck et al. [5] | 0.864 | 0.768 | 0.927 | Ours (joint tracking) | 0.837 | 0.742 | 0.862 |
| Davies and Böck [22] | 0.843 | 0.715 | 0.914 | <i>GTZAN</i> | | | |
| Ours (beat tracking) | 0.883 | 0.808 | 0.930 | Böck et al. [28] | 0.640 | 0.577 | 0.824 |
| Ours (joint tracking) | 0.885 | 0.813 | 0.931 | Durand et al. [8] | 0.607 | 0.480 | 0.774 |
| | | | | | | | |
| | | | | | | | |

Table 2: Beat tracking results on datasets used for training with 8-fold cross validation (top), and on unseen test data (bottom). ‡ was trained on *Ballroom* data only.

Ref: Böck & Davies, “Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation,” ISMIR 2020

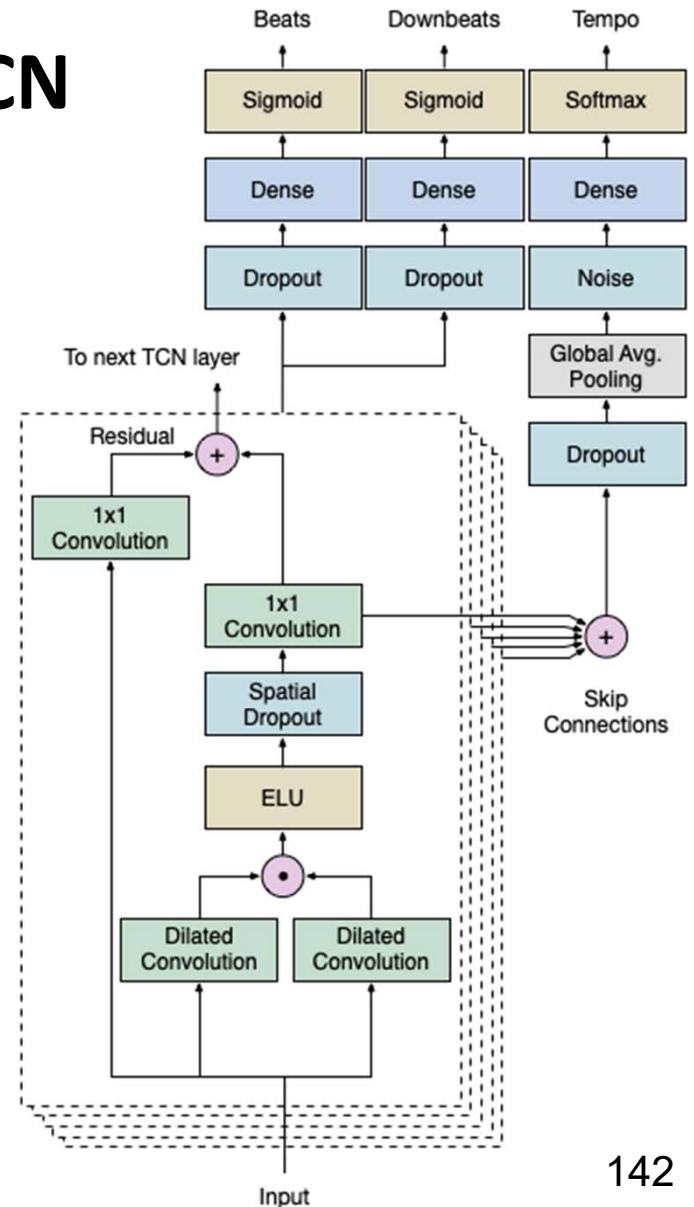
Table 3: Downbeat tracking results on datasets used for training with 8-fold cross validation (top), and on unseen test data (bottom).

Main Architecture - TCN

```

def residual_block(x, i, activation, num_filters, kernel_size, padding, dropout_rate=0, name=''):
    # name of the layer
    name = name + '_dilation_%d' % i
    # 1x1 conv. of input (so it can be added as residual)
    res_x = Conv1D(num_filters, 1, padding='same', name=name + '_1x1_conv_residual')(x)
    # two dilated convolutions, with dilation rates of i and 2i
    conv_1 = Conv1D(
        filters=num_filters,
        kernel_size=kernel_size,
        dilation_rate=i,
        padding=padding,
        name=name + '_dilated_conv_1',
    )(x)
    conv_2 = Conv1D(
        filters=num_filters,
        kernel_size=kernel_size,
        dilation_rate=i * 2,
        padding=padding,
        name=name + '_dilated_conv_2',
    )(x)
    # concatenate the output of the two dilations
    concat = keras.layers.concatenate([conv_1, conv_2], name=name + '_concat')
    # apply activation function
    x = Activation(activation, name=name + '_activation')(concat)
    # apply spatial dropout
    x = SpatialDropout1D(dropout_rate, name=name + '_spatial_dropout_%f' % dropout_rate)(x)
    # 1x1 conv. to obtain a representation with the same size as the residual
    x = Conv1D(num_filters, 1, padding='same', name=name + '_1x1_conv')(x)
    ...

```



PreNet

```

def create_model(input_shape, num_filters=20, num_dilations=11, kernel_size=5, activation='elu', dropout_rate=0.15):
    # input layer
    input_layer = Input(shape=input_shape)

    # stack of 3 conv layers, each conv, activation, max. pooling & dropout
    conv_1 = Conv2D(num_filters, (3, 3), padding='valid', name='conv_1_conv')(input_layer)
    conv_1 = Activation(activation, name='conv_1_activation')(conv_1)
    conv_1 = MaxPooling2D((1, 3), name='conv_1_max_pooling')(conv_1)
    conv_1 = Dropout(dropout_rate, name='conv_1_dropout')(conv_1)

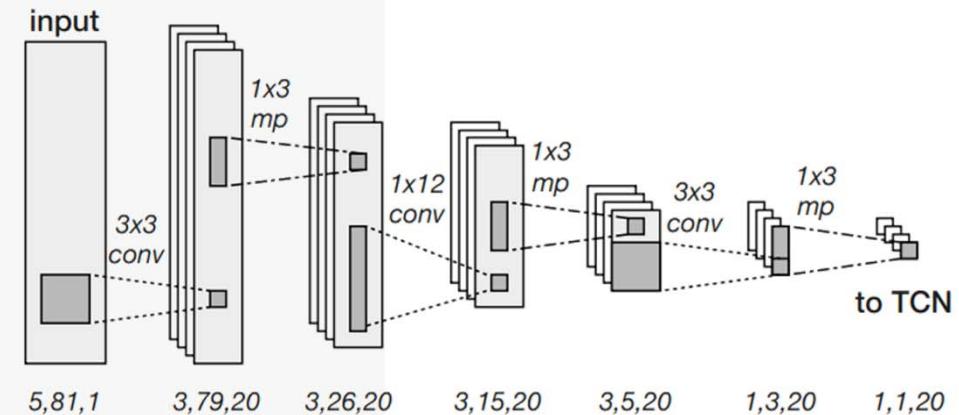
    conv_2 = Conv2D(num_filters, (1, 10), padding='valid', name='conv_2_conv')(conv_1)
    conv_2 = Activation(activation, name='conv_2_activation')(conv_2)
    conv_2 = MaxPooling2D((1, 3), name='conv_2_max_pooling')(conv_2)
    conv_2 = Dropout(dropout_rate, name='conv_2_dropout')(conv_2)

    conv_3 = Conv2D(num_filters, (3, 3), padding='valid', name='conv_3_conv')(conv_2)
    conv_3 = Activation(activation, name='conv_3_activation')(conv_3)
    conv_3 = MaxPooling2D((1, 3), name='conv_3_max_pooling')(conv_3)
    conv_3 = Dropout(dropout_rate, name='conv_3_dropout')(conv_3)

    # reshape layer to reduce dimensions
    x = Reshape((-1, num_filters), name='tcn_input_reshape')(conv_3)

    # TCN layers
    dilations = [2 ** i for i in range(num_dilations)]
    tcn, skip = TCN(
        num_filters=[num_filters] * len(dilations),
        kernel_size=kernel_size,
        dilations=dilations.

```



Data Sequence Handling & Target Widening

```
# infer (global) tempo from beats
def infer_tempo(beats, hist_smooth=15, fps=FPS, no_tempo=MASK_VALUE):
    ibis = np.diff(beats) * fps
    bins = np.bincount(np.round(ibis).astype(int))
    # if no beats are present, there is no tempo
    if not bins.any():
        return NO_TEMPO
    intervals = np.arange(1+len(bins))
    # smooth histogram bins
    if hist_smooth > 0:
        bins = madmom.audio.signal.smooth(bins, hist_smooth)
    # create interpolation function
    interpolation_fn = interp1d(intervals, bins, 'quadratic')
    # generate new intervals with 1000x the resolution
    intervals = np.arange(intervals[0], intervals[-1], 0.001)
    tempi = 60.0 * fps / intervals
    # apply quadratic interpolation
    bins = interpolation_fn(intervals)
    peaks = argrelextrema(bins, mode='wrap')[0]
    if len(peaks) == 0:
        # no peaks, no tempo
        return no_tempo
    else:
        # report only the strongest tempo
        sorted_peaks = peaks[np.argsort(bins[peaks])[:-1]]
    return tempi[sorted_peaks][0]
```

```
# pad features
def cnn_pad(data, pad_frames):
    """Pad the data by repeating the first and last frame N times."""
    pad_start = np.repeat(data[:1], pad_frames, axis=0)
    pad_stop = np.repeat(data[-1:], pad_frames, axis=0)
    return np.concatenate((pad_start, data, pad_stop))

def widen_beat_targets(self, size=3, value=0.5):
    for y in self.beats.values():
        # skip masked beat targets
        if np.allclose(y, MASK_VALUE):
            continue
        np.maximum(y, maximum_filter1d(y, size=size) * value, out=y)

def widen_downbeat_targets(self, size=3, value=0.5):
    for y in self.downbeats.values():
        # skip masked downbeat targets
        if np.allclose(y, MASK_VALUE):
            continue
        np.maximum(y, maximum_filter1d(y, size=size) * value, out=y)

def widen_tempo_targets(self, size=3, value=0.5):
    for y in self.tempo.values():
        # skip masked tempo targets
        if np.allclose(y, MASK_VALUE):
            continue
        np.maximum(y, maximum_filter1d(y, size=size) * value, out=y)
```

Data Augmentation

- “We use a simple approach of adding the same training examples with **changed hop-size when computing the STFT**. This results in the same song being represented with a different numbers of frames. This way the **beat positions are “stretched” or “squeezed”** and the tempo changes accordingly.”

```
[ ] for fps in [95, 97.5, 102.5, 105]:  
    ds = DataSequence(  
        tracks={f'{k}_{fps}': v for k, v in tracks.items() if k in train_files},  
        pre_processor=PreProcessor(fps=fps),  
        pad_frames=pad_frames,  
    )  
    ds.widen_beat_targets()  
    ds.widen_downbeat_targets()  
    ds.widen_tempo_targets(3, 0.5)  
    ds.widen_tempo_targets(3, 0.5)  
    train.append(ds)
```

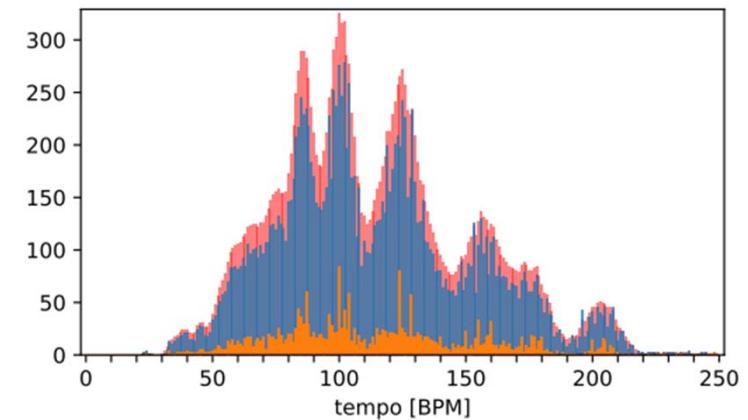


Figure 2: Tempo distribution of original tempo annotations (orange, foreground), after data augmentation (blue) and target widening (red, background).

Post-processing

```
[ ] # track beats with a DBN
beat_tracker = madmom.features.beats.DBNBeatTrackingProcessor(
    min_bpm=55.0, max_bpm=215.0, fps=FPS, transition_lambda=100, threshold=0.05
)

# track downbeats with a DBN
# as input, use a combined beat & downbeat activation function
downbeat_tracker = madmom.features.downbeats.DBNDownBeatTrackingProcessor(
    beats_per_bar=[3, 4], min_bpm=55.0, max_bpm=215.0, fps=FPS, transition_lambda=100
)

# track bars, i.e. first track the beats and then infer the downbeat positions
bar_tracker = madmom.features.downbeats.DBNBarTrackingProcessor(
    beats_per_bar=(3, 4), meter_change_prob=1e-3, observation_weight=4
```

```
def detect_tempo(bins, hist_smooth=11, min_bpm=10):
    min_bpm = int(np.floor(min_bpm))
    tempi = np.arange(min_bpm, len(bins))
    bins = bins[min_bpm:]
    # smooth histogram bins
    if hist_smooth > 0:
        bins = madmom.audio.signal.smooth(bins, hist_smooth)
    # create interpolation function
    interpolation_fn = interp1d(tempi, bins, 'quadratic')
    # generate new intervals with 1000x the resolution
    tempi = np.arange(tempi[0], tempi[-1], 0.001)
    # apply quadratic interpolation
    bins = interpolation_fn(tempi)
    peaks = argrelmax(bins, mode='wrap')[0]
    if len(peaks) == 0:
        # no peaks, no tempo
        tempi = np.array([], ndmin=2)
    elif len(peaks) == 1:
        # report only the strongest tempo
        ret = np.array([tempi[peaks[0]], 1.0])
        tempi = np.array([tempi[peaks[0]], 1.0])
    else:
        # sort the peaks in descending order of bin heights
        sorted_peaks = peaks[np.argsort(bins[peaks])[:-1]]
        # normalize their strengths
        strengths = bins[sorted_peaks]
        strengths /= np.sum(strengths)
```

Libraries

- Madmom

<https://github.com/CPJKU/madmom>

madmom.features

- `madmom.features.beats`
- `madmom.features.beats_crf`
- `madmom.features.beats_hmm`
- `madmom.features.chords`
- `madmom.features.downbeats`
- `madmom.features.key`
- `madmom.features.notes`
- `madmom.features.notes_hmm`
- `madmom.features.onsets`
- `madmom.features.tempo`

- BeatNet

<https://github.com/mjhydri/BeatNet>

BeatNet is state-of-the-art (Real-Time) and Offline joint music beat, downbeat, tempo, and meter tracking system using CRNN and particle filtering. (ISMIR 2021's paper implementation).

Madmom

<https://github.com/CPJKU/madmom>

| | <i>F-measure</i> | <i>CMLt</i> | <i>AMLt</i> | | <i>F-measure</i> | <i>CMLt</i> | <i>AMLt</i> |
|-----------------------|------------------|-------------|-------------|----------------------------|------------------|-------------|-------------|
| <i>Ballroom</i> | | | | <i>Ballroom</i> | | | |
| Böck et al. [28] | 0.938 | 0.892 | 0.953 | Böck et al. [28] | 0.863 | 0.834 | 0.931 |
| Elowsson [51] ‡ | 0.925 | 0.903 | 0.932 | Durand et al. [8] | 0.797 | 0.616 | 0.916 |
| Davies and Böck [22] | 0.933 | 0.881 | 0.929 | Fuentes et al. [9] | 0.83 | - | - |
| Ours (beat tracking) | 0.956 | 0.935 | 0.958 | Ours (sequential tracking) | 0.900 | 0.894 | 0.953 |
| Ours (joint tracking) | 0.962 | 0.947 | 0.961 | Ours (joint tracking) | 0.916 | 0.913 | 0.960 |
| <i>Hainsworth</i> | | | | <i>Hainsworth</i> | | | |
| Böck et al. [5] | 0.884 | 0.808 | 0.916 | Böck et al. [28] | 0.684 | 0.628 | 0.832 |
| Elowsson [51] ‡ | 0.742 | 0.676 | 0.792 | Durand et al. [8] | 0.664 | 0.500 | 0.804 |
| Davies and Böck [22] | 0.874 | 0.795 | 0.930 | Fuentes et al. [9] | 0.67 | - | - |
| Ours (beat tracking) | 0.904 | 0.851 | 0.937 | Ours (sequential tracking) | 0.713 | 0.686 | 0.855 |
| Ours (joint tracking) | 0.902 | 0.848 | 0.930 | Ours (joint tracking) | 0.722 | 0.696 | 0.872 |
| <i>SMC</i> | | | | <i>Beatles</i> | | | |
| Böck et al. [5] | 0.529 | 0.428 | 0.567 | Böck et al. [28] | 0.831 | 0.730 | 0.858 |
| Elowsson [51] ‡ | 0.375 | 0.225 | 0.332 | Durand et al. [8] | 0.847 | 0.722 | 0.875 |
| Davies and Böck [22] | 0.543 | 0.432 | 0.632 | Fuentes et al. [9] | 0.86 | - | - |
| Ours (beat tracking) | 0.552 | 0.465 | 0.643 | Ours (sequential tracking) | 0.829 | 0.748 | 0.860 |
| Ours (joint tracking) | 0.544 | 0.443 | 0.635 | Ours (joint tracking) | 0.837 | 0.742 | 0.862 |
| <i>GTZAN</i> | | | | <i>GTZAN</i> | | | |
| Böck et al. [5] | 0.864 | 0.768 | 0.927 | Böck et al. [28] | 0.640 | 0.577 | 0.824 |
| Davies and Böck [22] | 0.843 | 0.715 | 0.914 | Durand et al. [8] | 0.607 | 0.480 | 0.774 |
| Ours (beat tracking) | 0.883 | 0.808 | 0.930 | Ours (sequential tracking) | 0.654 | 0.619 | 0.817 |
| Ours (joint tracking) | 0.885 | 0.813 | 0.931 | Ours (joint tracking) | 0.672 | 0.640 | 0.832 |

Table 2: Beat tracking results on datasets used for training with 8-fold cross validation (top), and on unseen test data (bottom). ‡ was trained on *Ballroom* data only.

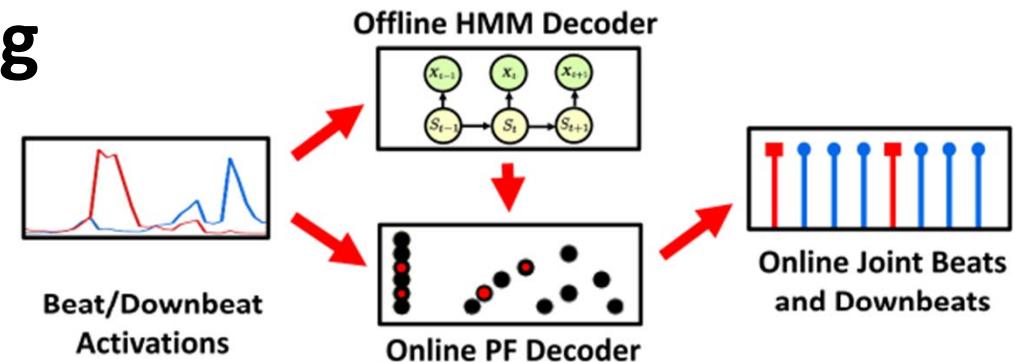
Table 3: Downbeat tracking results on datasets used for training with 8-fold cross validation (top), and on unseen test data (bottom).

Ref: Böck & Davies, “Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation,” ISMIR 2020

Ref: Böck et al, “Joint beat and downbeat tracking with recurrent neural networks,” ISMIR 2016

Challenge: Vocal Beat Tracking

- Harder in general (low F1)
- Online vs offline



| Metric | No Skip | | | | Skip First 5 Seconds | | | |
|-----------------------------|----------------|--------------------|-----------------|---------------------|----------------------|--------------------|-----------------|---------------------|
| | Beat (70ms) | Downbeat (70ms) | Beat (200ms) | Downbeat (200ms) | Beat (70ms) | Downbeat (70ms) | Beat (200ms) | Downbeat (200ms) |
| <i>In-House Dataset</i> | | | | | | | | |
| BeatNet (baseline) | 53.90 | 30.06 | 71.40 | 42.33 | 54.90 | 31.58 | 71.97 | 43.90 |
| SingNet (default inference) | 56.48 | 34.36 | 72.84 | 46.25 | 57.57 | 36.55 | 73.64 | 48.43 |
| SingNet (salience-informed) | 58.09 | 35.74 | 74.54 | 47.23 | 59.51 | 37.99 | 75.53 | 49.22 |
| SingNet (past-informed) | 57.38 | 35.06 | 73.78 | 46.80 | 58.86 | 37.59 | 74.74 | 48.90 |
| SingNet (combined) | 58.1 | 35.95 | 74.62 | 47.38 | 59.53 | 38.04 | 75.65 | 49.24 |
| Online DBN | 38.94 | 20.16 | 55.32 | 27.14 | 40.83 | 20.73 | 55.20 | 27.90 |
| Offline DBN (oracle) | 60.67 | 45.17 | 79.70 | 55.11 | 60.81 | 45.22 | 79.91 | 55.20 |

Ref: Heydari et al, "SingNet: A real-time singing voice beat and downbeat tracking system," ICASSP 2023