

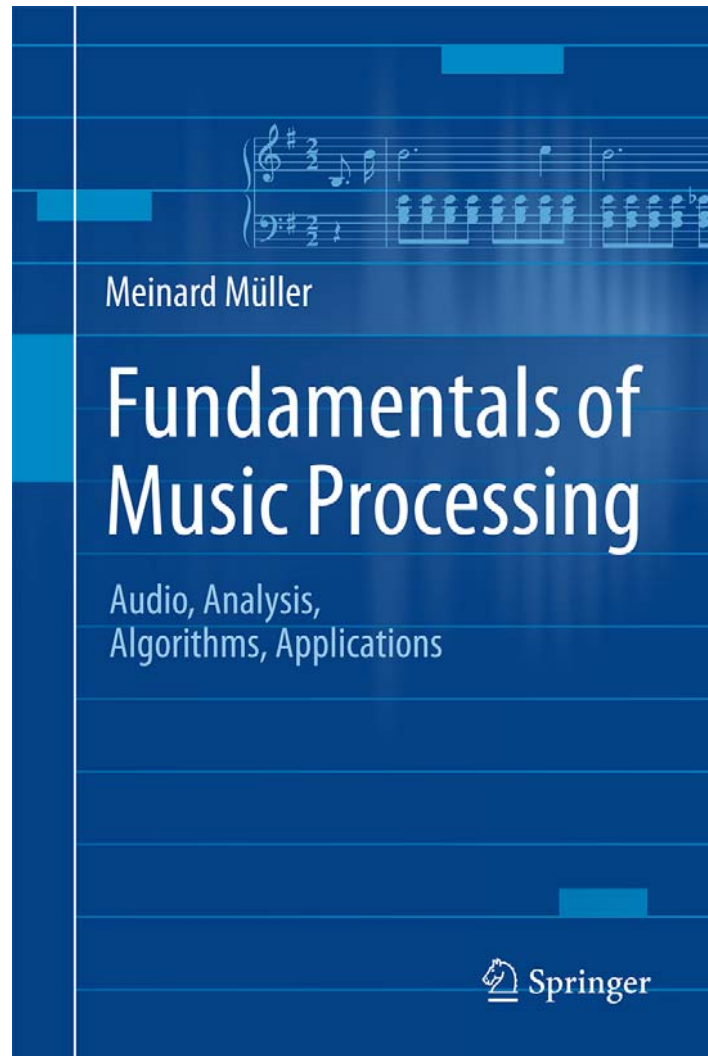
Lecture  
**Music Processing**

# **Music Synchronization**

**Meinard Müller**

International Audio Laboratories Erlangen  
[meinard.mueller@audiolabs-erlangen.de](mailto:meinard.mueller@audiolabs-erlangen.de)

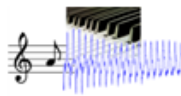

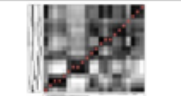


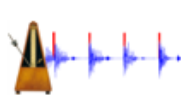
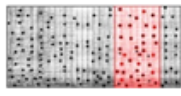
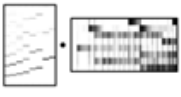
# Book: Fundamentals of Music Processing



Meinard Müller  
Fundamentals of Music Processing  
Audio, Analysis, Algorithms, Applications  
483 p., 249 illus., hardcover  
ISBN: 978-3-319-21944-8  
Springer, 2015

Accompanying website:  
[www.music-processing.de](http://www.music-processing.de)

# Book: Fundamentals of Music Processing

Chapter		Music Processing Scenario
1		Music Representations
2		Fourier Analysis of Signals
3		Music Synchronization
4		Music Structure Analysis
5		Chord Recognition
6		Tempo and Beat Tracking
7		Content-Based Audio Retrieval
8		Musically Informed Audio Decomposition

Meinard Müller

Fundamentals of Music Processing

Audio, Analysis, Algorithms, Applications

483 p., 249 illus., hardcover

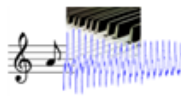




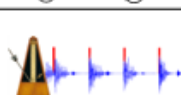
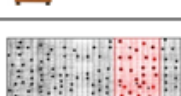

ISBN: 978-3-319-21944-8

Springer, 2015

Accompanying website:

[www.music-processing.de](http://www.music-processing.de)

# Book: Fundamentals of Music Processing

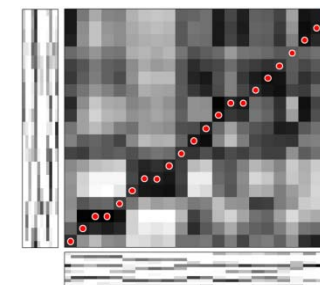
Chapter		Music Processing Scenario
1		Music Representations
2		Fourier Analysis of Signals
3		Music Synchronization
4		Music Structure Analysis
5		Chord Recognition
6		Tempo and Beat Tracking
7		Content-Based Audio Retrieval
8		Musically Informed Audio Decomposition

Meinard Müller  
Fundamentals of Music Processing  
Audio, Analysis, Algorithms, Applications  
483 p., 249 illus., hardcover  
ISBN: 978-3-319-21944-8  
Springer, 2015

Accompanying website:  
[www.music-processing.de](http://www.music-processing.de)

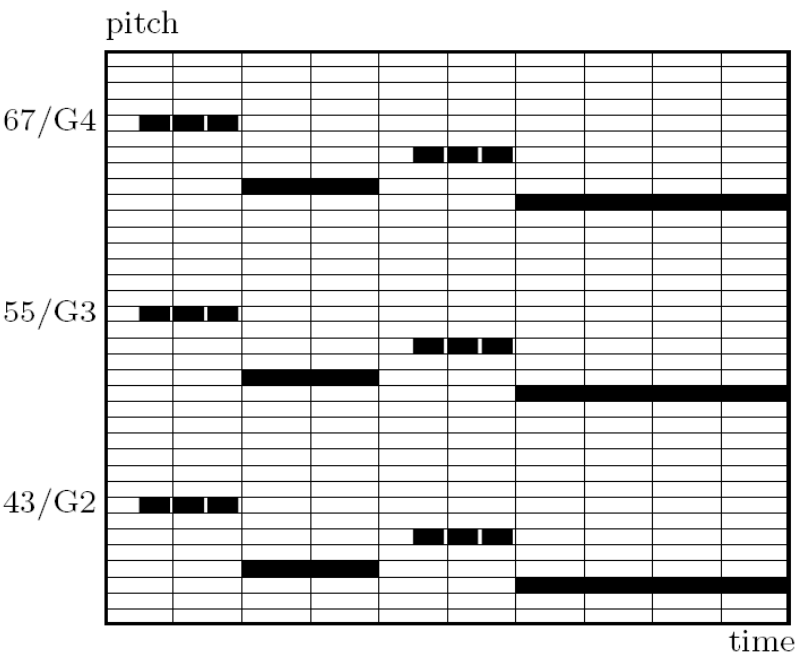
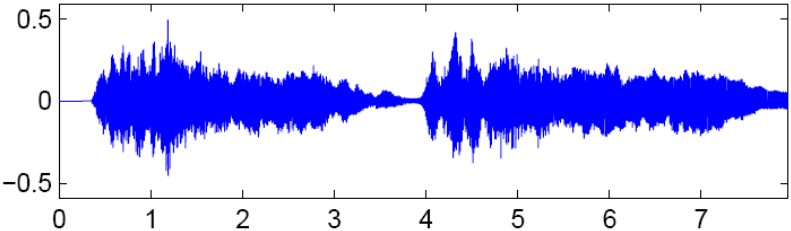
# Chapter 3: Music Synchronization

- 3.1 Audio Features
- 3.2 **Dynamic Time Warping**
- 3.3 **Applications**
- 3.4 Further Notes



As a first music processing task, we study in Chapter 3 the problem of music synchronization. The objective is to temporally align compatible representations of the same piece of music. Considering this scenario, we explain the need for musically informed audio features. In particular, we introduce the concept of chroma-based music features, which capture properties that are related to harmony and melody. Furthermore, we study an alignment technique known as dynamic time warping (DTW), a concept that is applicable for the analysis of general time series. For its efficient computation, we discuss an algorithm based on dynamic programming—a widely used method for solving a complex problem by breaking it down into a collection of simpler subproblems.

# Music Data



# Music Data

Various interpretations – Beethoven's Fifth

---

Bernstein



---

Karajan



---

Scherbakov (piano)



---

MIDI (piano)



---

# Music Synchronization: Audio-Audio

**Given:** Two different audio recordings of the same underlying piece of music.

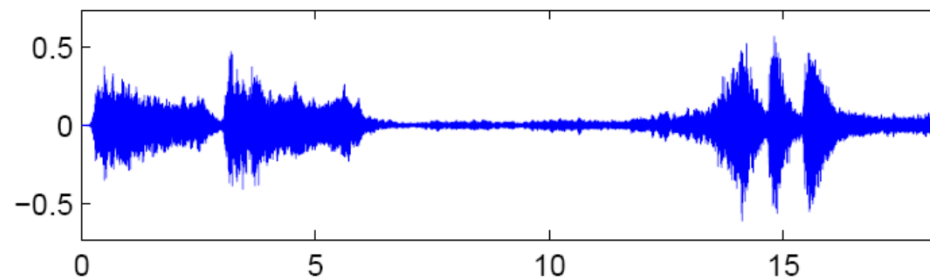
**Goal:** Find for each position in one audio recording the **musically** corresponding position in the other audio recording.



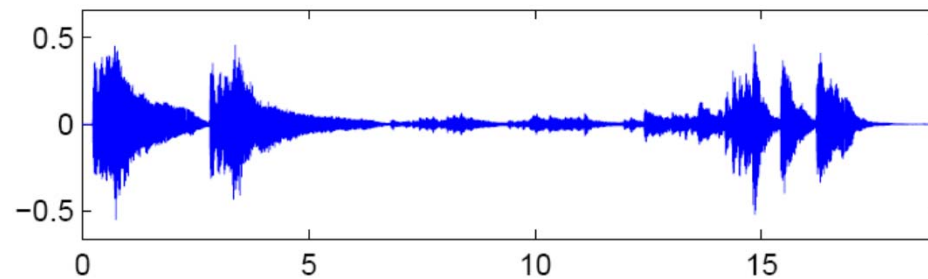
# Music Synchronization: Audio-Audio

Beethoven's Fifth

Karajan



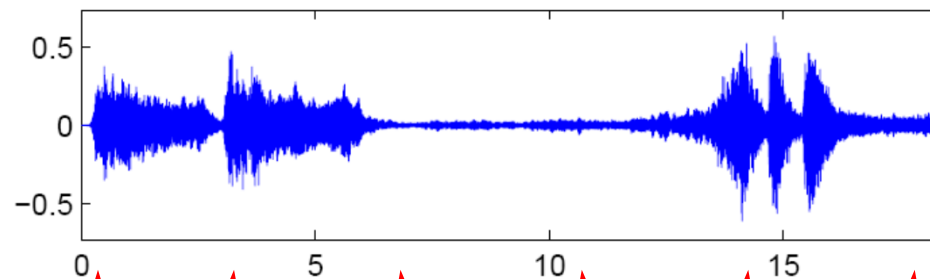
Scherbakov



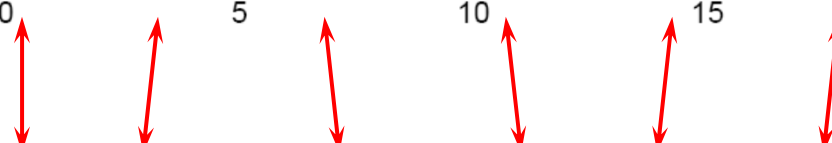
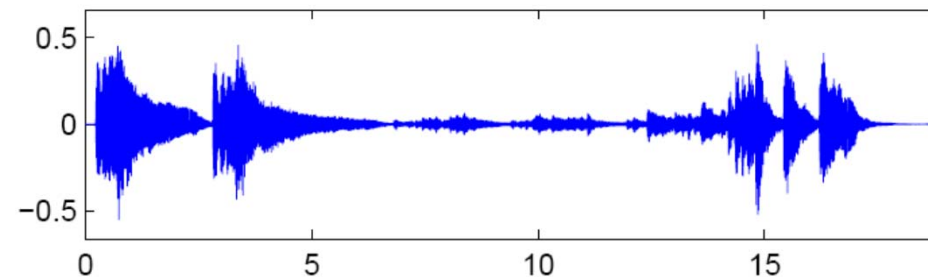
# Music Synchronization: Audio-Audio

Beethoven's Fifth

Karajan



Scherbakov

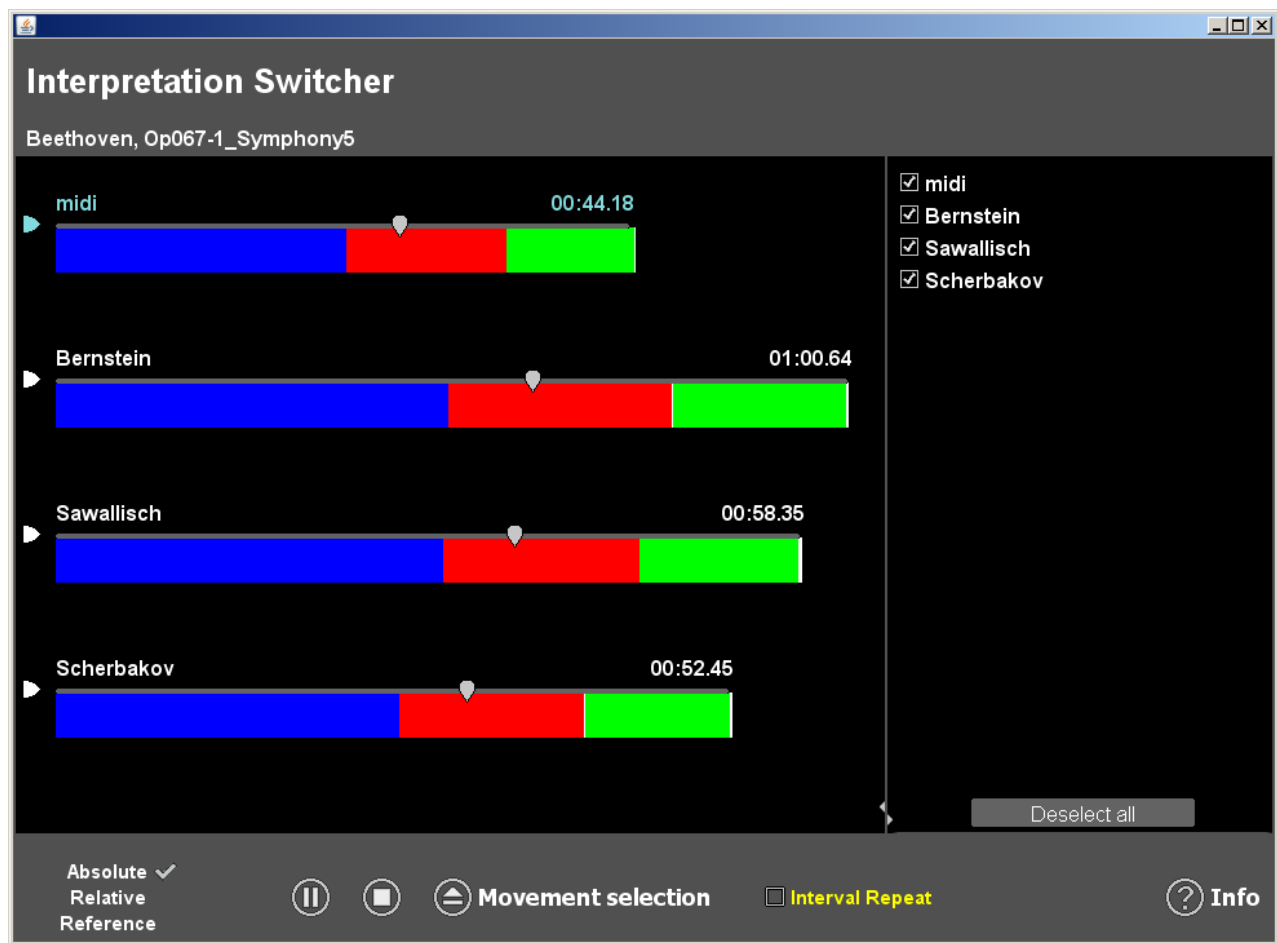


Synchronization: Karajan → Scherbakov



# Music Synchronization: Audio-Audio

Application: Interpretation Switcher



# Music Synchronization: Audio-Audio

Two main steps:

## 1.) Audio features

- Robust but discriminative
- Chroma features
- Robust to variations in instrumentation, timbre, dynamics
- Correlate to harmonic progression

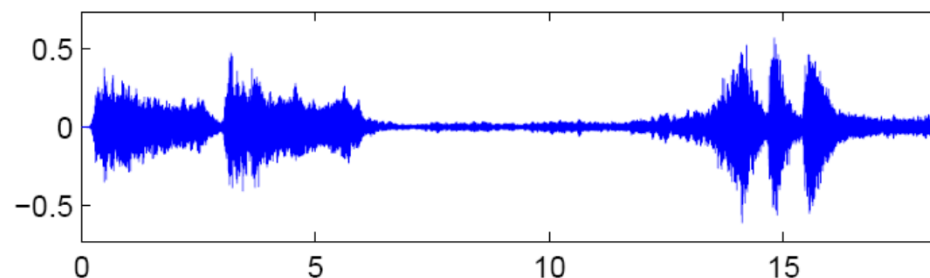
## 2.) Alignment procedure

- Deals with local and global tempo variations
- Needs to be efficient

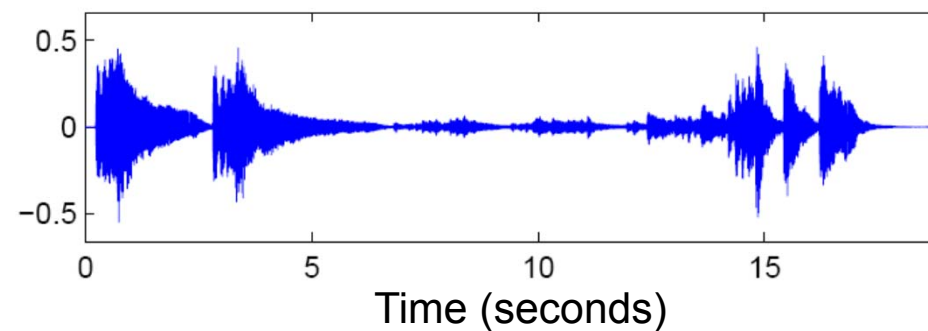
# Music Synchronization: Audio-Audio

Beethoven's Fifth

Karajan



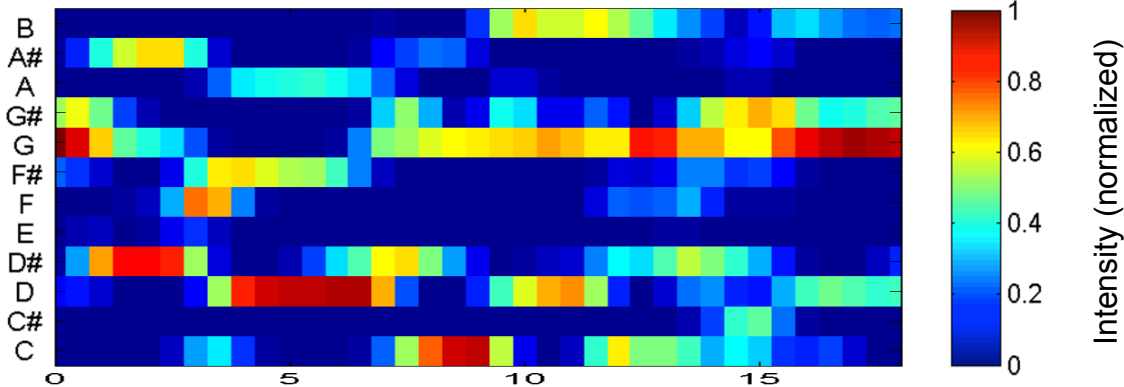
Scherbakov



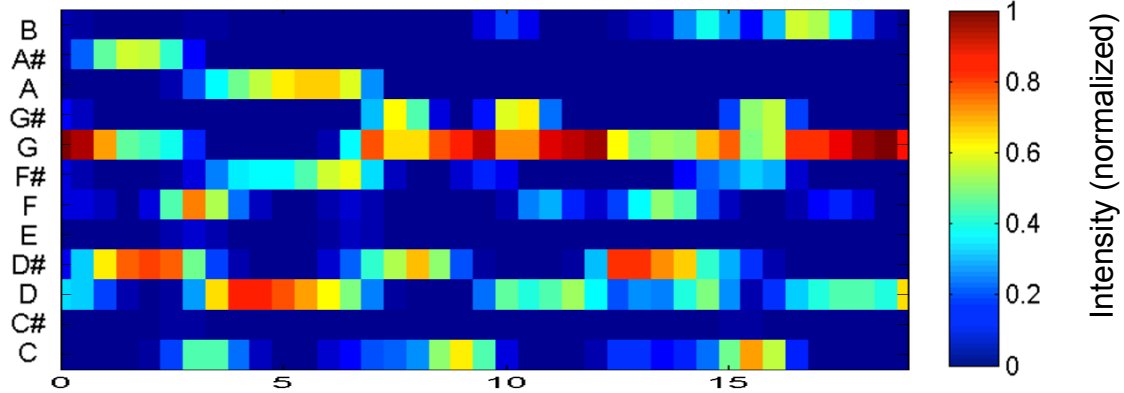
# Music Synchronization: Audio-Audio

## Beethoven's Fifth

Karajan



Scherbakov



Time (seconds)

# Music Synchronization: Audio-Audio

## Beethoven's Fifth

Karajan

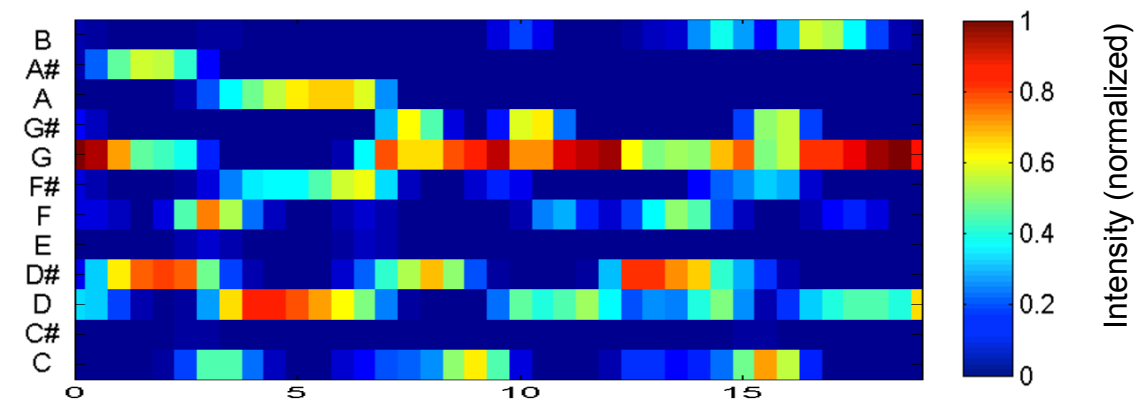
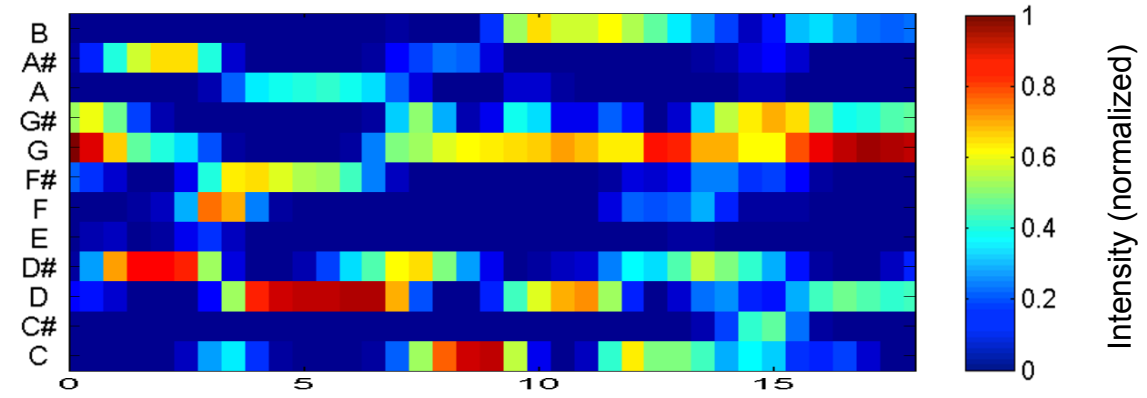


Allegro con brio ( $\text{♩} = 108$ )

*ff*

*Red. \**

Scherbakov



Time (seconds)

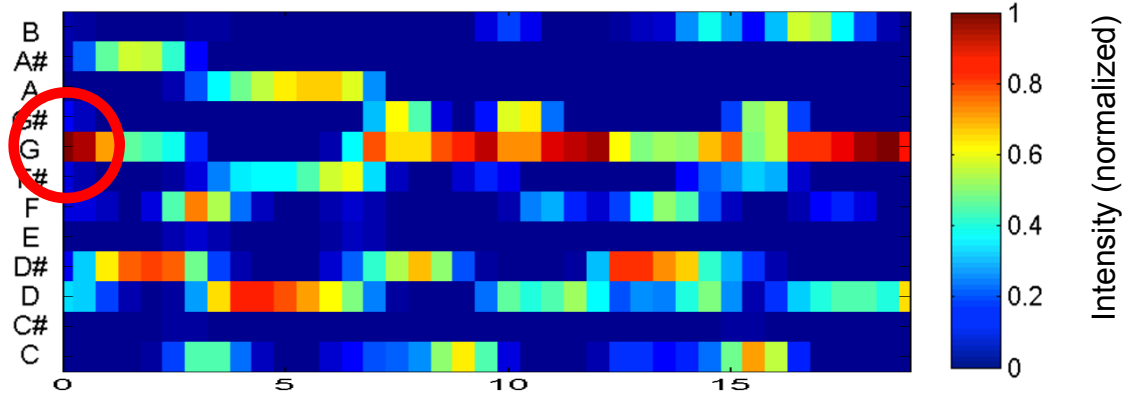
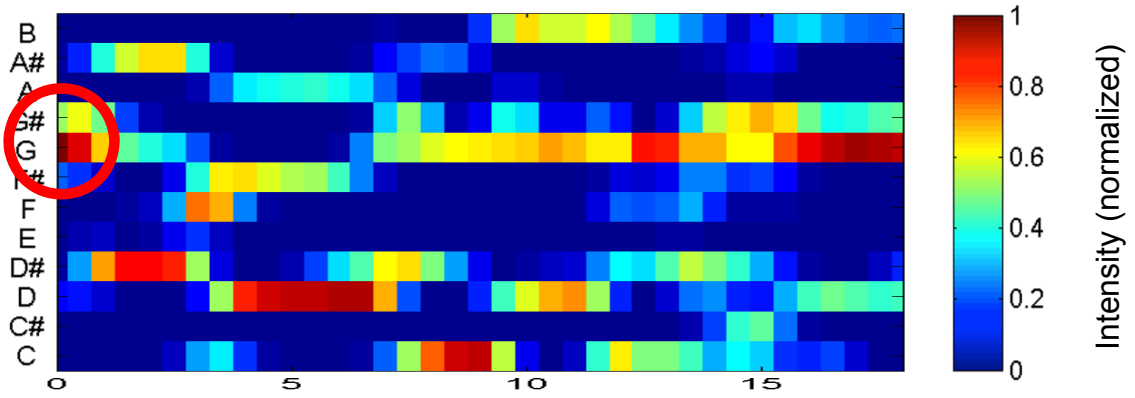
# Music Synchronization: Audio-Audio

## Beethoven's Fifth

Karajan



Scherbakov



Time (seconds)



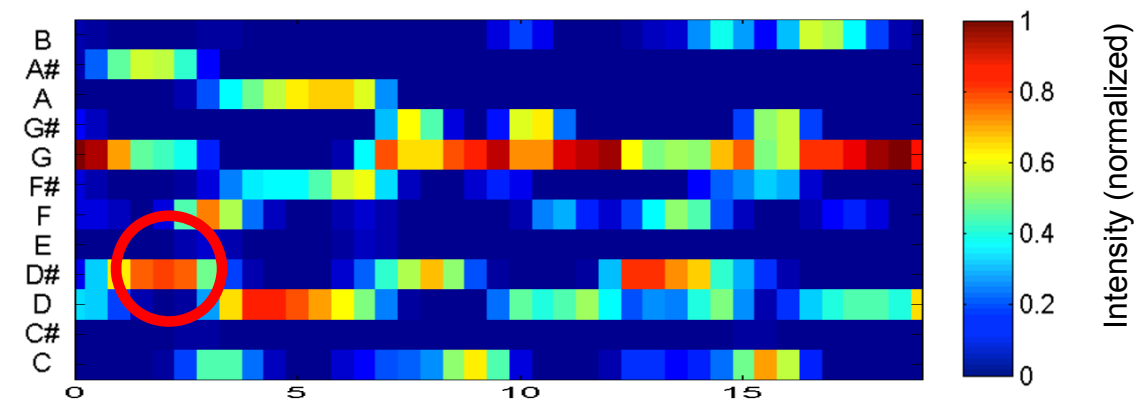
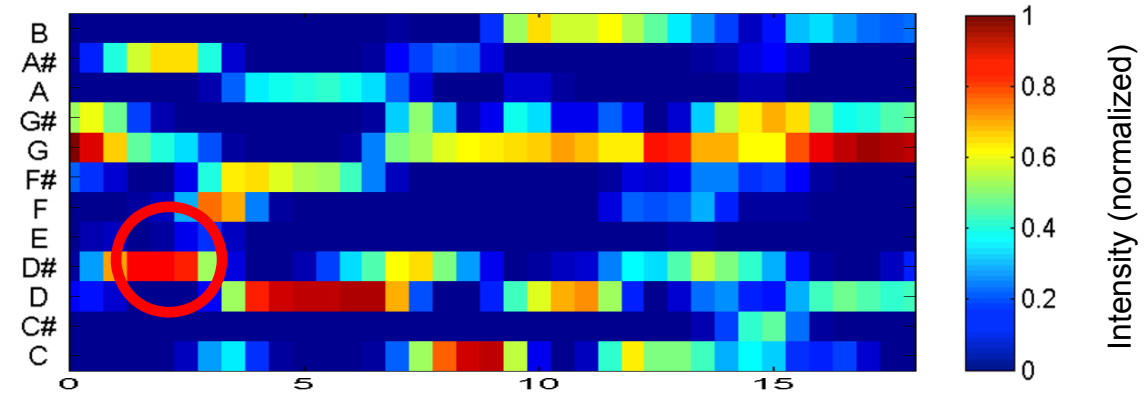
# Music Synchronization: Audio-Audio

## Beethoven's Fifth

Karajan



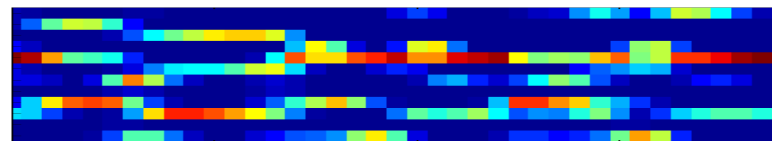
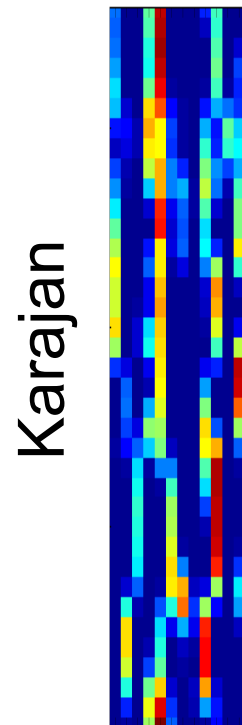
Scherbakov



Time (seconds)

---

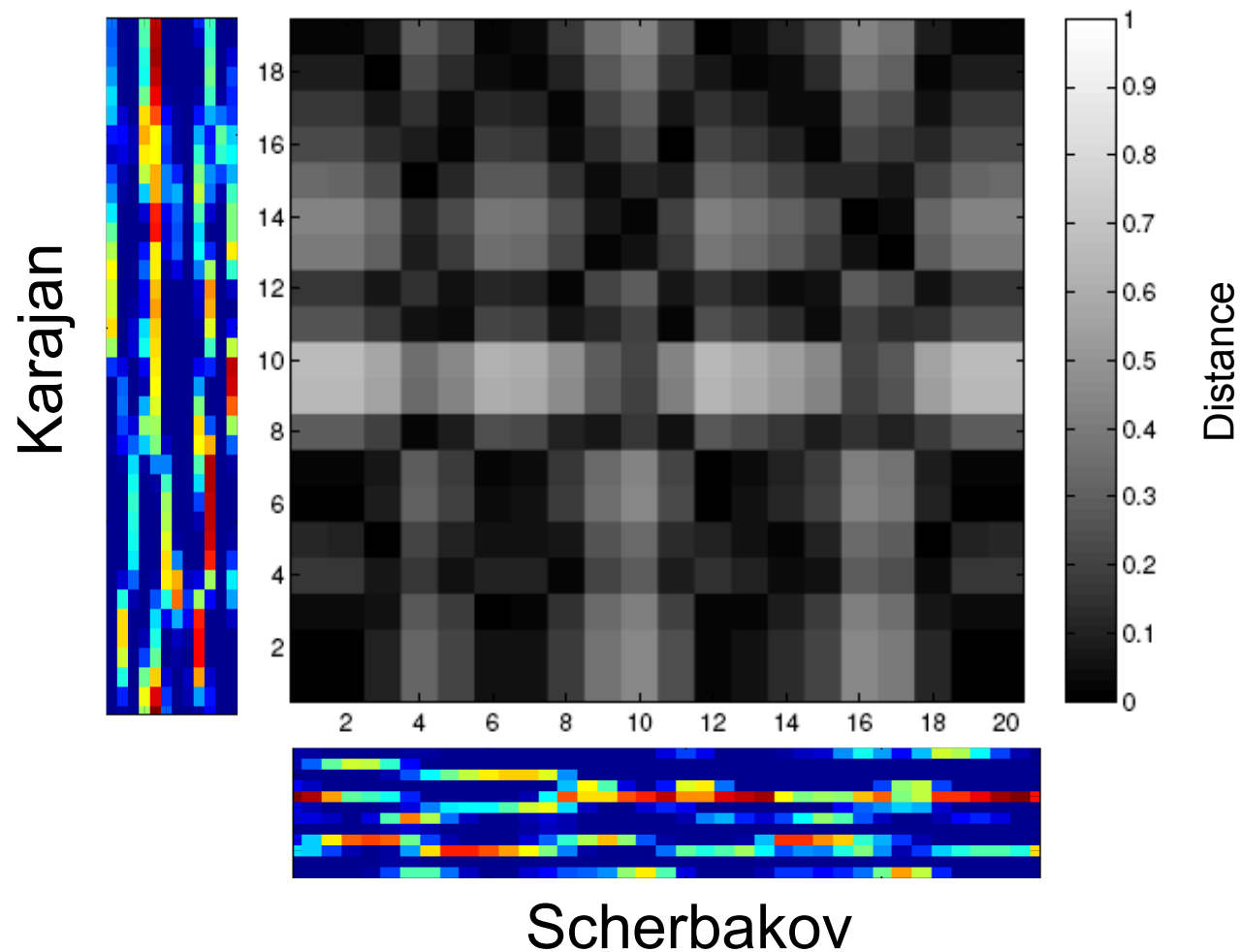
# Music Synchronization: Audio-Audio



Scherbakov

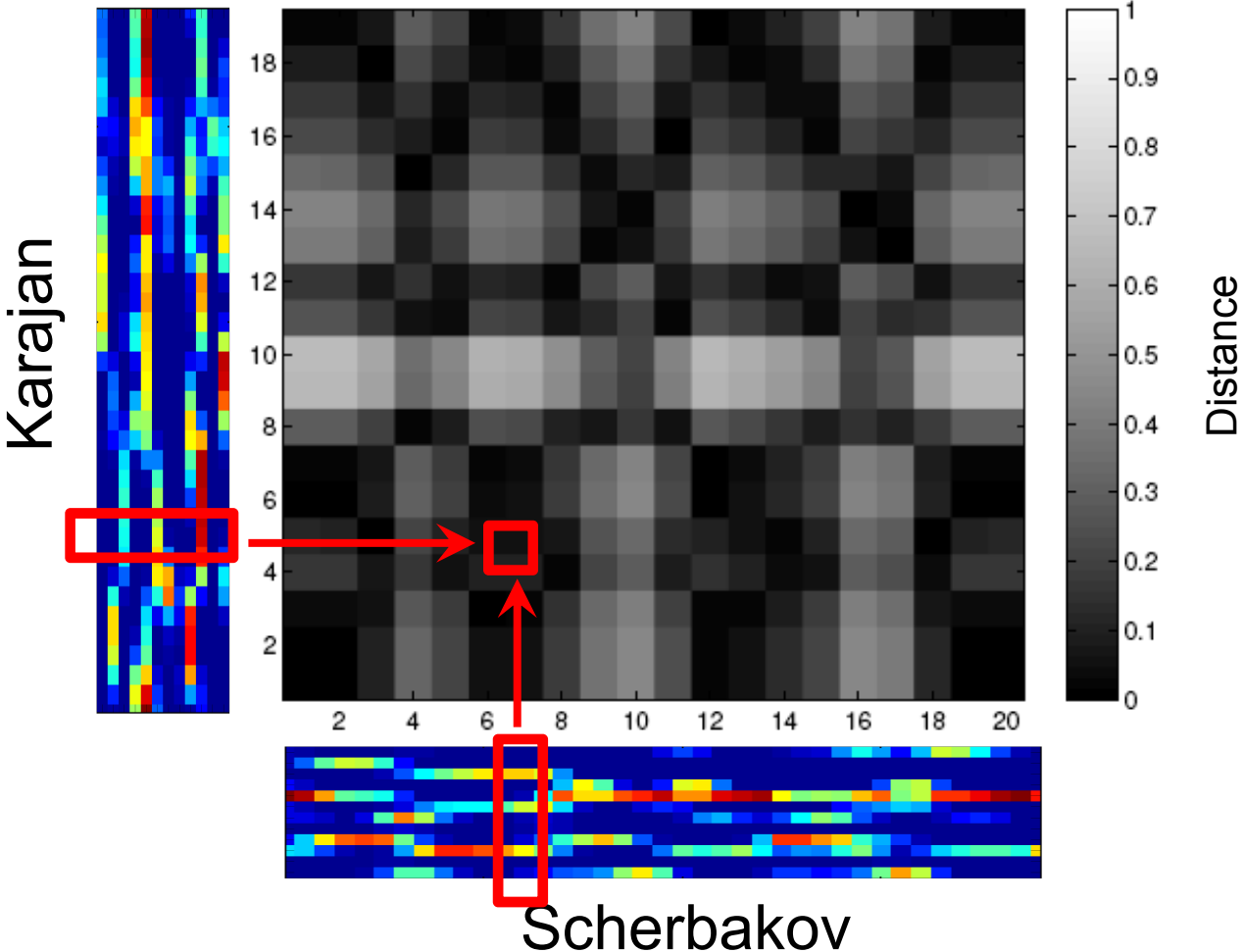
# Music Synchronization: Audio-Audio

Cost matrix



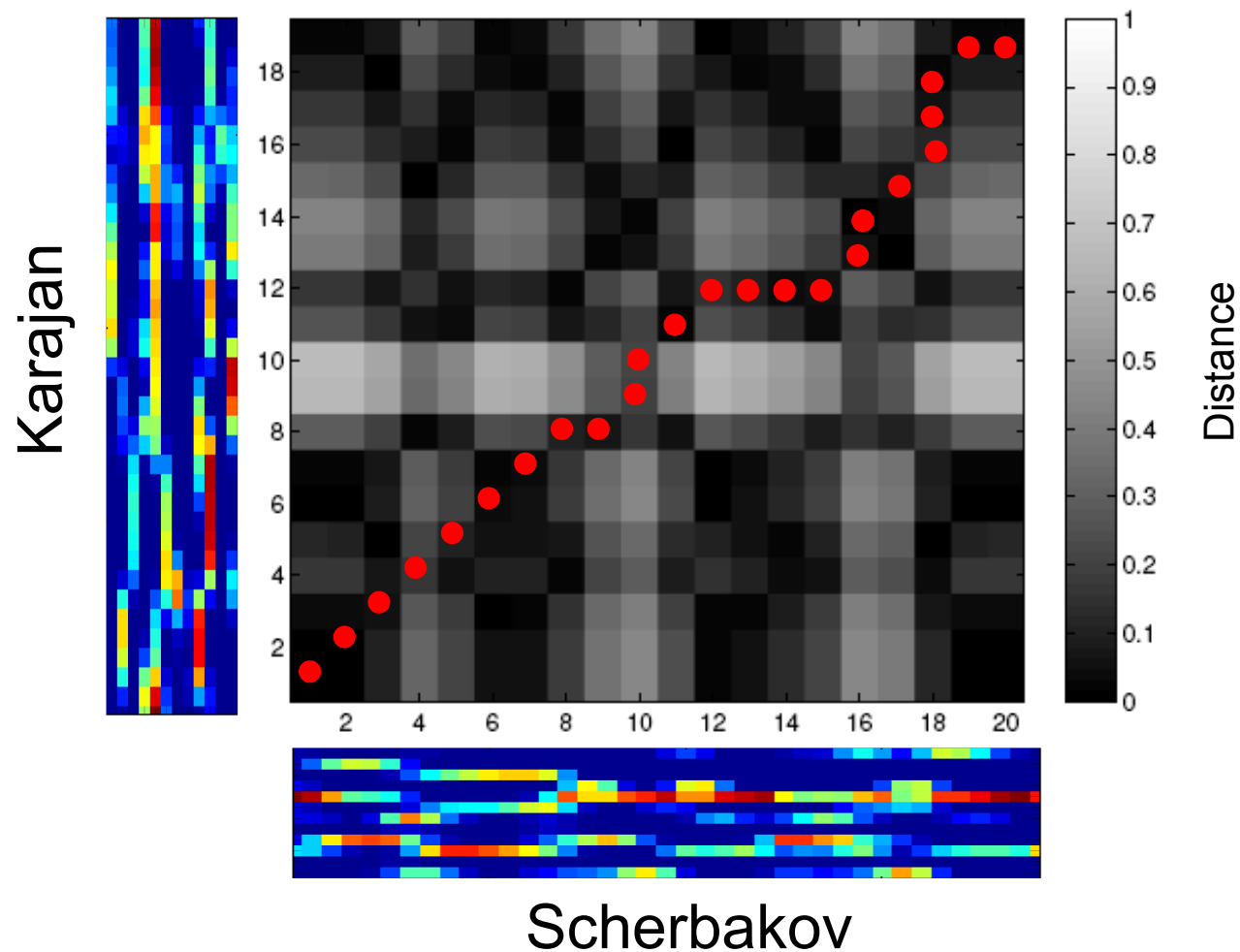
# Music Synchronization: Audio-Audio

Cost matrix



# Music Synchronization: Audio-Audio

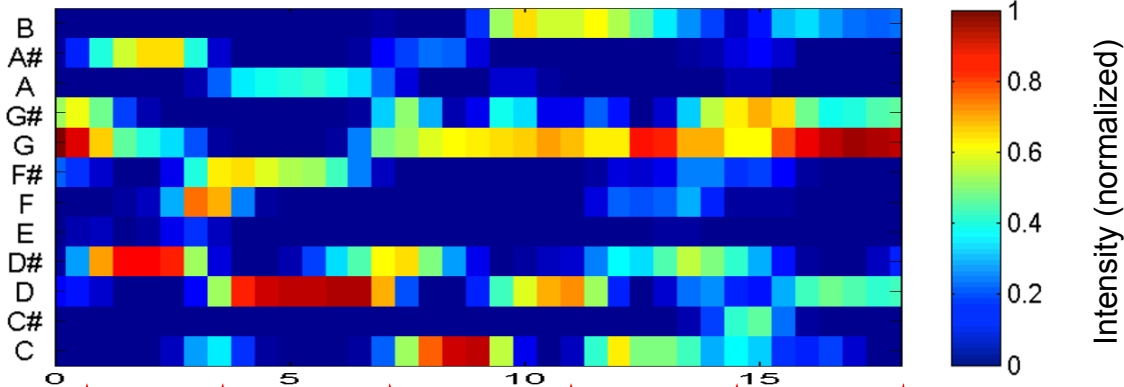
Cost-minimizing alignment path



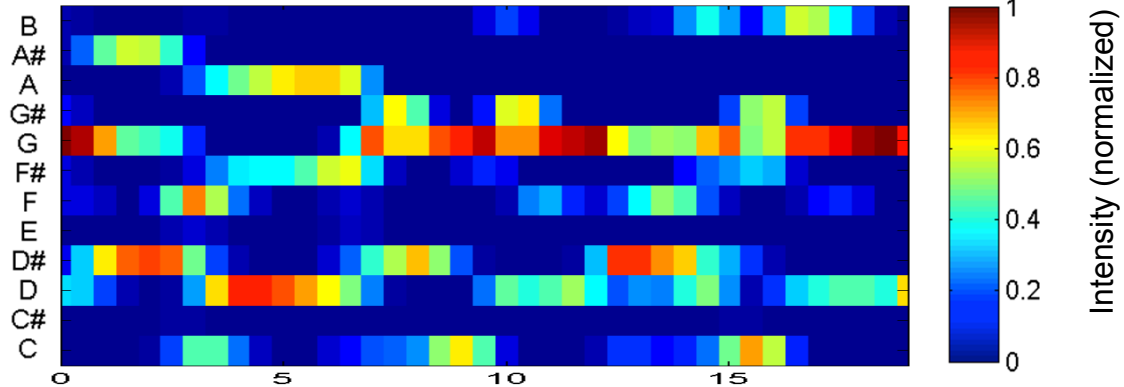
# Music Synchronization: Audio-Audio

## Beethoven's Fifth

Karajan



Scherbakov



Time (seconds)



# Music Synchronization: Audio-Audio

How to compute the alignment?

- ⇒ Cost matrices
- ⇒ Dynamic programming
- ⇒ Dynamic Time Warping (DTW)

# Dynamic Time Warping

- Well-known technique to find an optimal alignment between two given (time-dependent) sequences under certain restrictions.
- Intuitively, sequences are warped in a non-linear fashion to match each other.
- Originally used to compare different speech patterns in automatic speech recognition



# Dynamic Time Warping

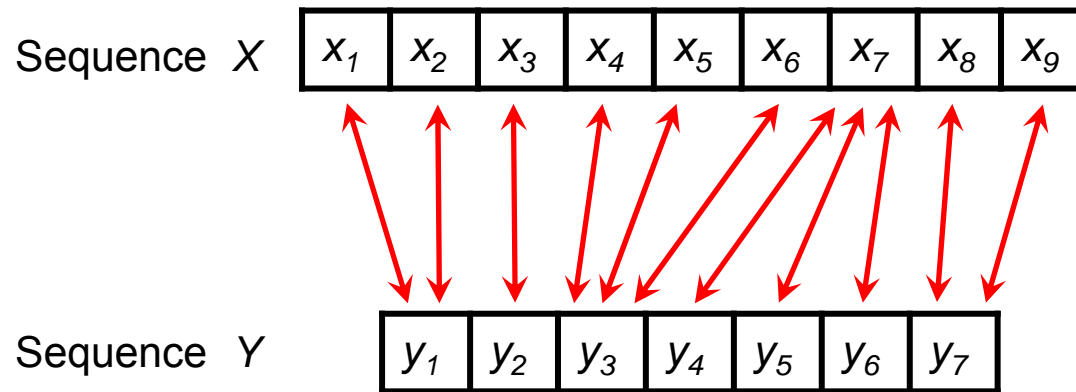
Sequence  $X$ 

$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$x_6$	$x_7$	$x_8$	$x_9$
-------	-------	-------	-------	-------	-------	-------	-------	-------

Sequence  $Y$ 

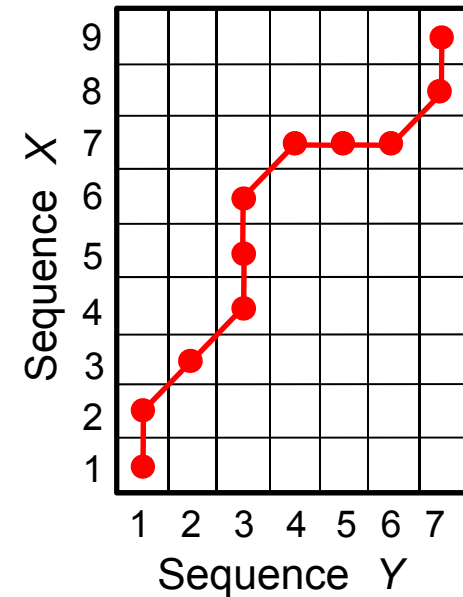
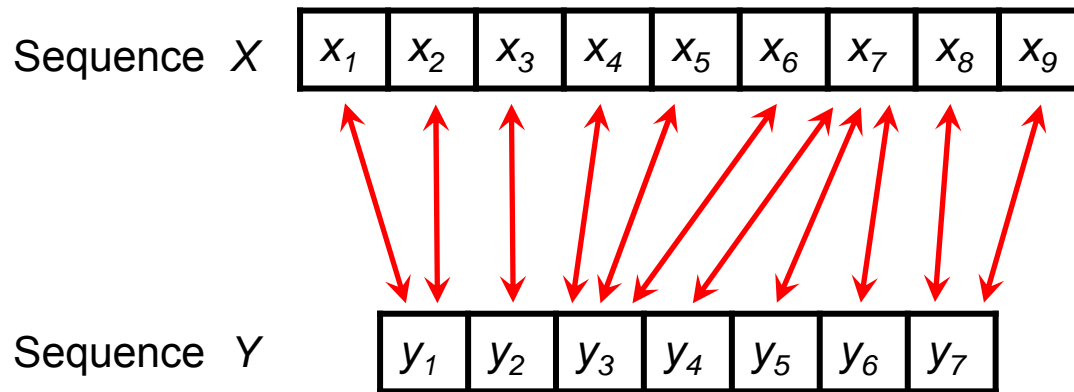
$y_1$	$y_2$	$y_3$	$y_4$	$y_5$	$y_6$	$y_7$
-------	-------	-------	-------	-------	-------	-------

# Dynamic Time Warping



Time alignment of two time-dependent sequences, where the **aligned** points are indicated by the **arrows**.

# Dynamic Time Warping



Time alignment of two time-dependent sequences, where the **aligned** points are indicated by the **arrows**.

# Dynamic Time Warping

The objective of DTW is to compare two (time-dependent) sequences

$$X := (x_1, x_2, \dots, x_N)$$

of length  $N \in \mathbb{N}$  and

$$Y := (y_1, y_2, \dots, y_M)$$

of length  $M \in \mathbb{N}$ . Here,

$$x_n, y_m \in \mathcal{F}, n \in [1 : N], m \in [1 : M],$$

are suitable features that are elements from a given feature space denoted by  $\mathcal{F}$ .

# Dynamic Time Warping

To compare two different features  $x, y \in \mathcal{F}$  one needs a local cost measure which is defined to be a function

$$c : \mathcal{F} \times \mathcal{F} \rightarrow \mathbb{R}_{\geq 0}$$

Typically,  $c(x, y)$  is small (low cost) if  $x$  and  $y$  are similar to each other, and otherwise  $c(x, y)$  is large (high cost).

# Dynamic Time Warping

Evaluating the local cost measure for each pair of elements of the sequences  $X$  and  $Y$ , one obtains the cost matrix

$$C \in \mathbb{R}^{N \times M}$$

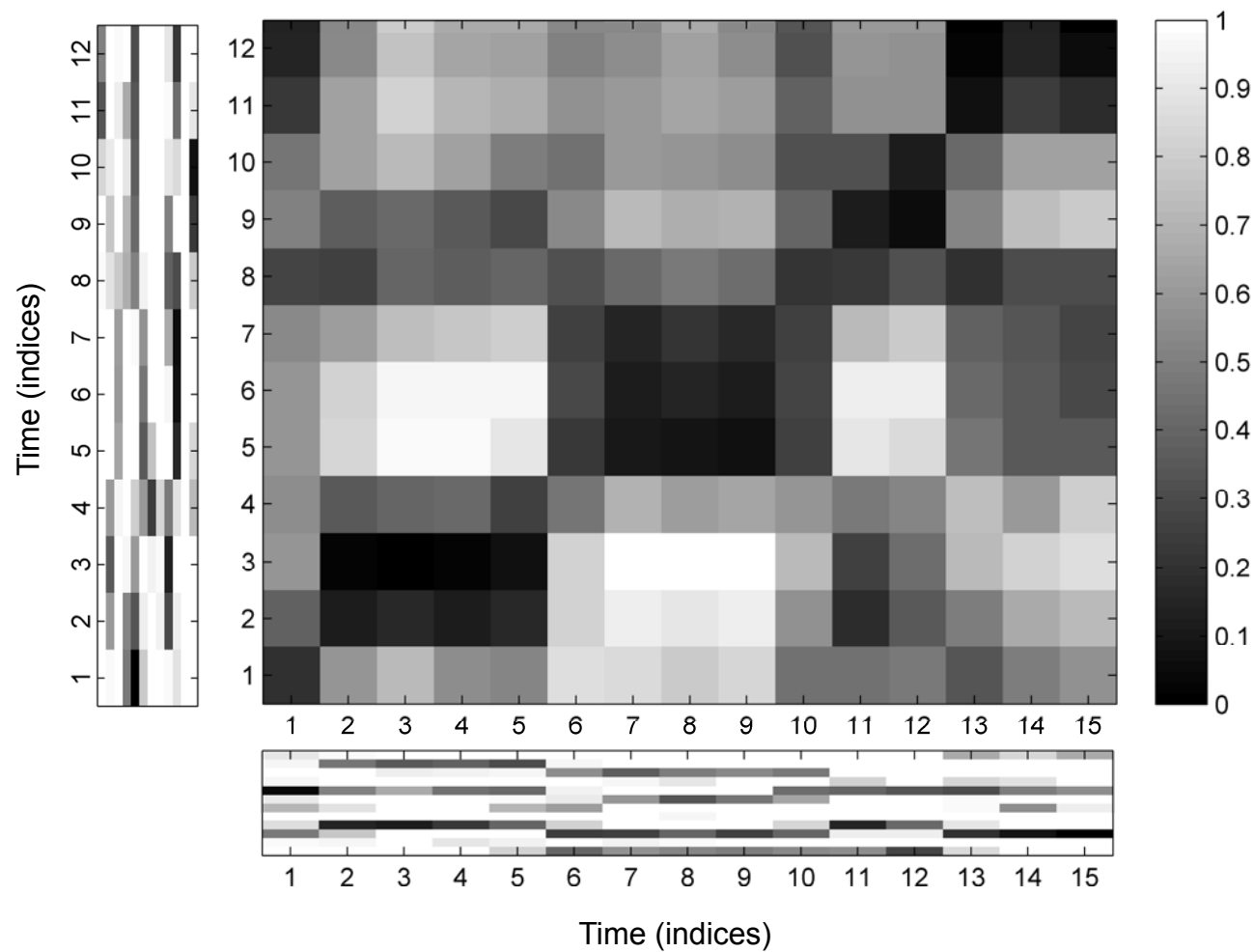
denfined by

$$C(n, m) := c(x_n, y_m).$$

Then the goal is to find an alignment between  $X$  and  $Y$  having minimal overall cost. Intuitively, such an optimal alignment runs along a “valley” of low cost within the cost matrix  $C$ .

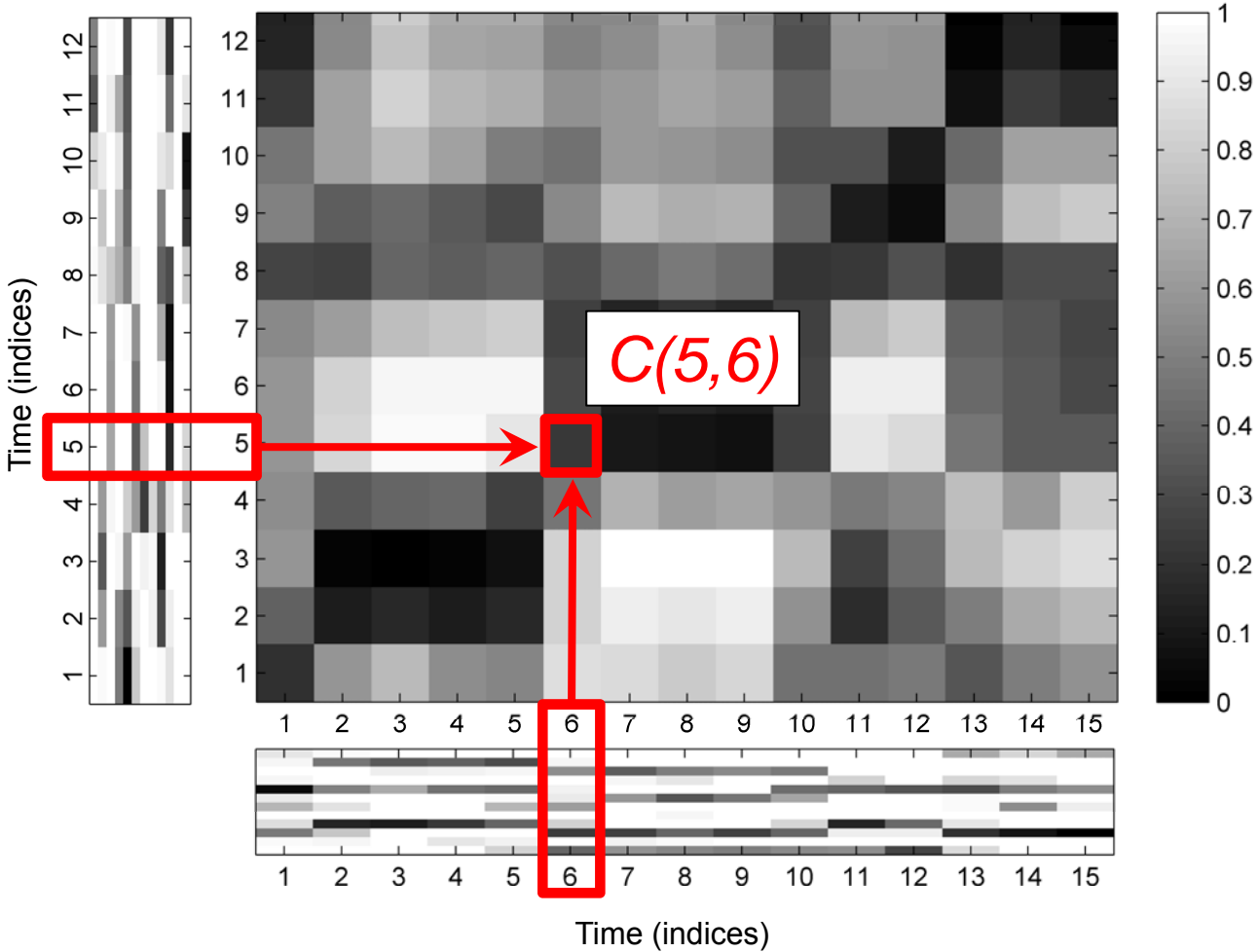
# Dynamic Time Warping

Cost matrix  $C$



# Dynamic Time Warping

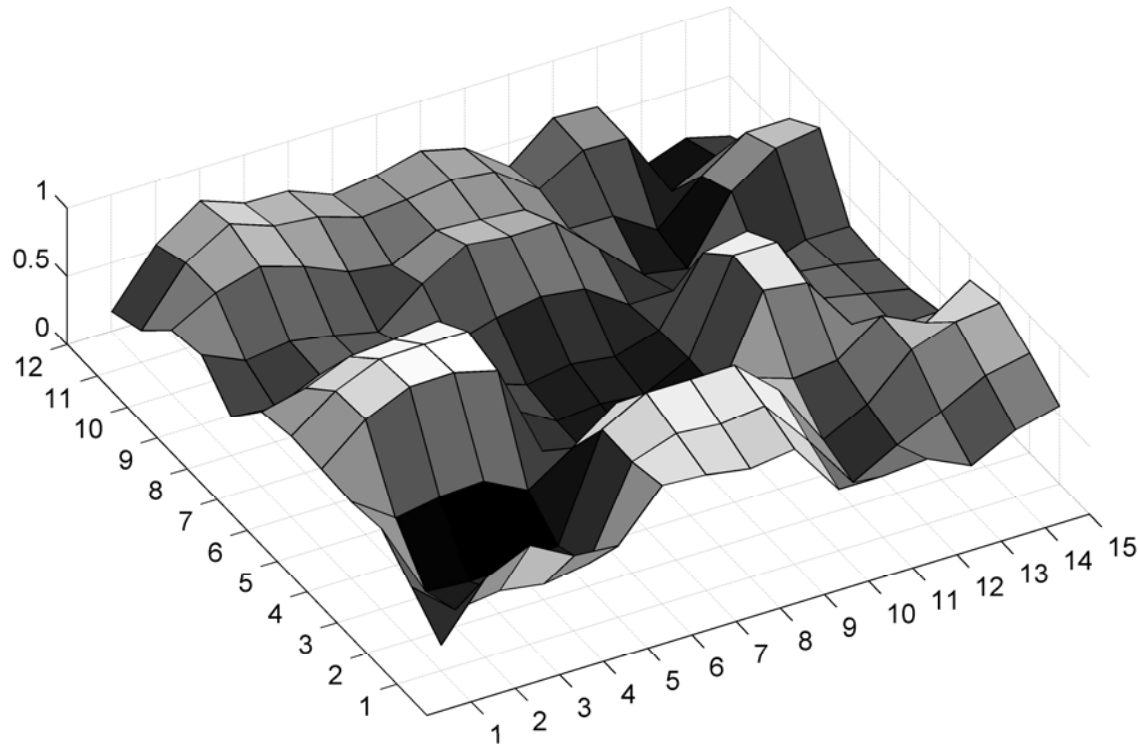
## Cost matrix $C$





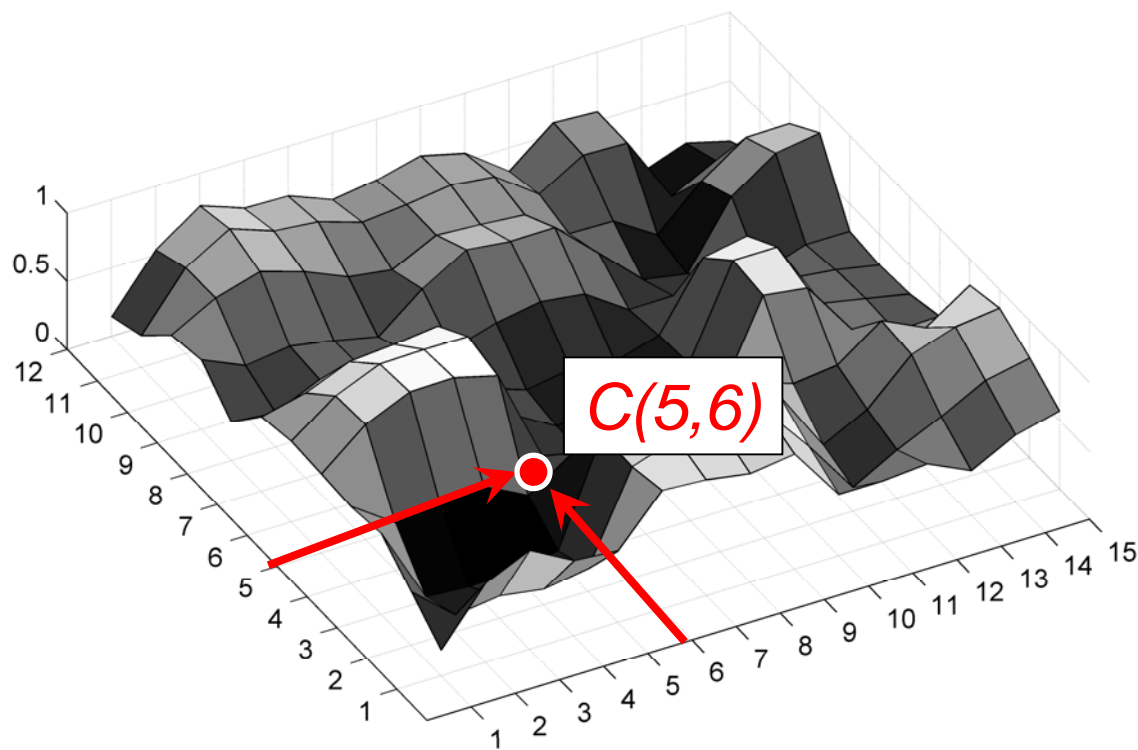
# Dynamic Time Warping

Cost matrix  $C$



# Dynamic Time Warping

Cost matrix  $C$



# Dynamic Time Warping

The next definition formalizes the notion of an alignment.

A **warping path** is a sequence  $p = (p_1, \dots, p_L)$  with

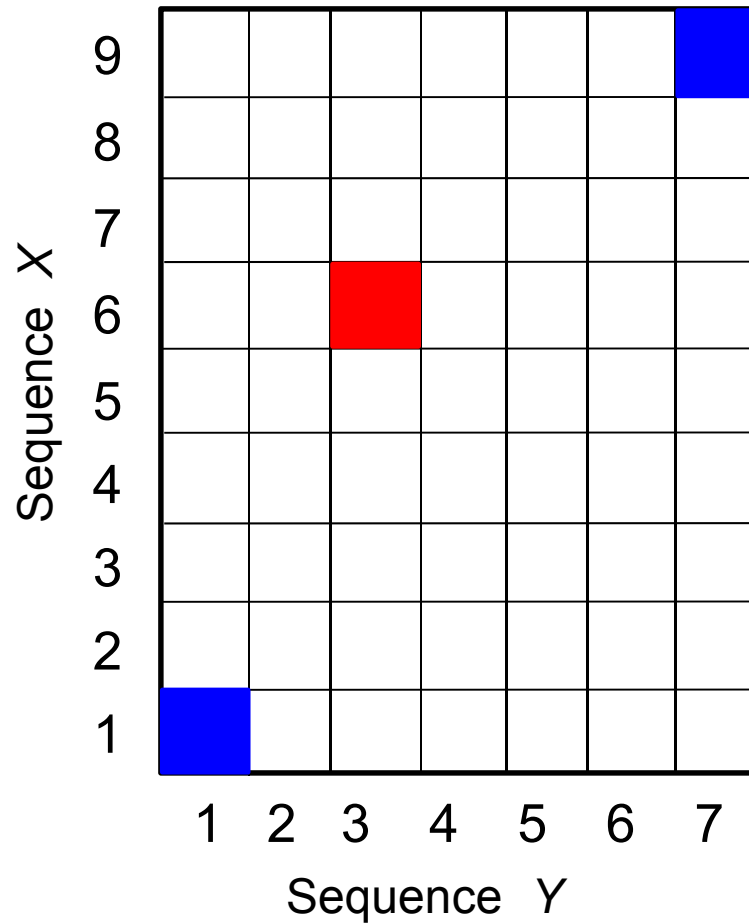
$$p_\ell = (n_\ell, m_\ell) \in [1 : N] \times [1 : M]$$

for  $\ell \in [1 : L]$  satisfying the following three conditions:

- 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  $\ell \in [1 : L - 1]$

# Dynamic Time Warping

## Warping path



Each matrix entry (cell) corresponds to a pair of indices.

Cell = (6,3)

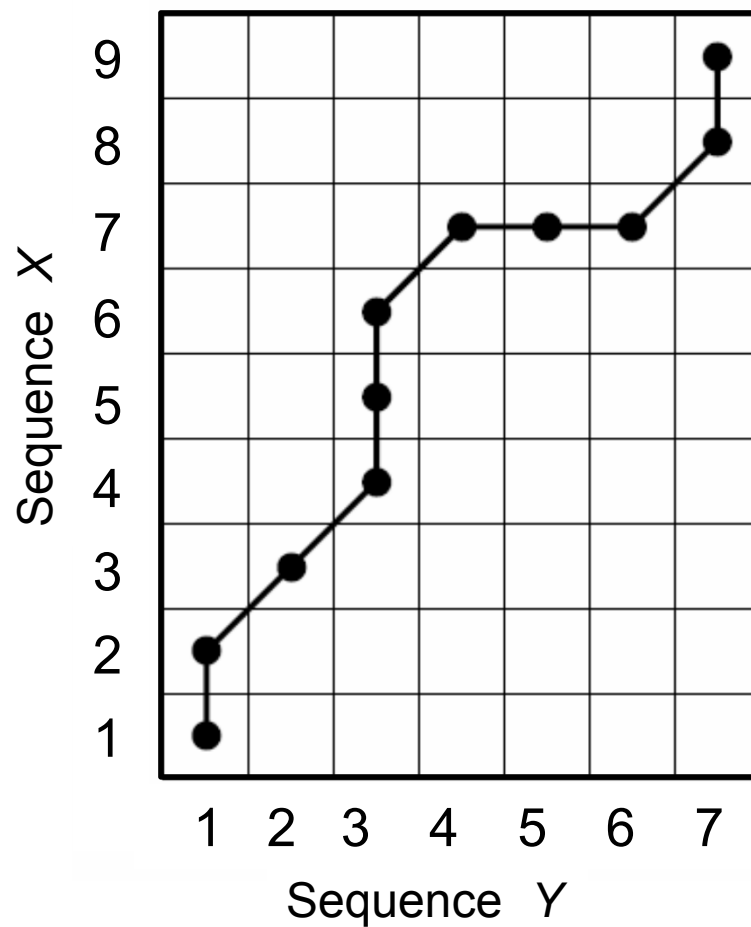
Boundary cells:

$p_1 = (1,1)$

$p_L = (N,M) = (9,7)$

# Dynamic Time Warping

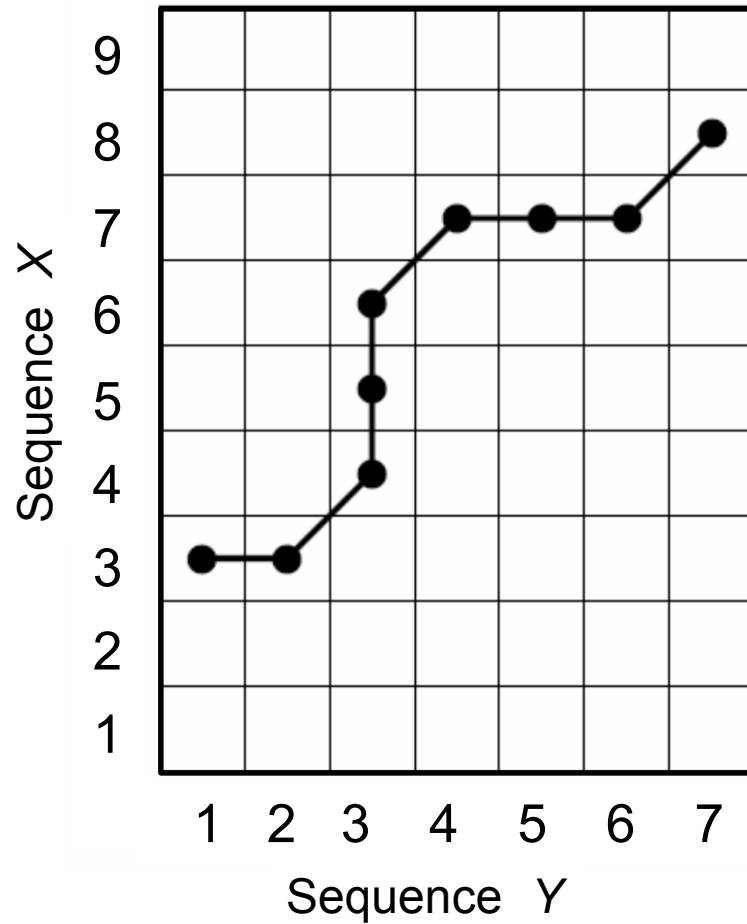
Warping path



Correct  
warping path

# Dynamic Time Warping

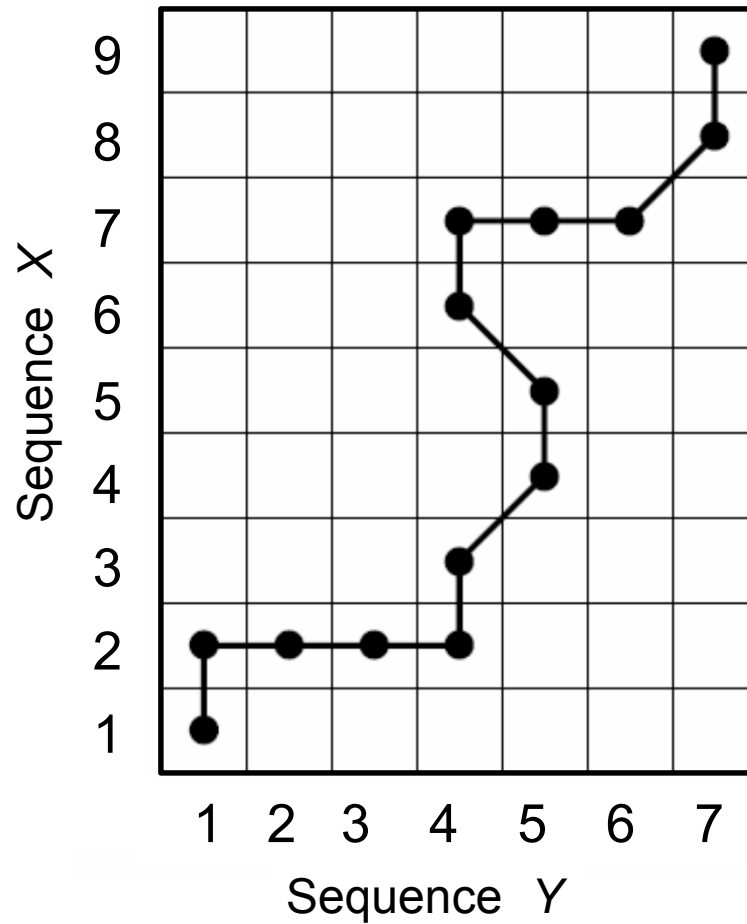
Warping path



Violation of boundary condition

# Dynamic Time Warping

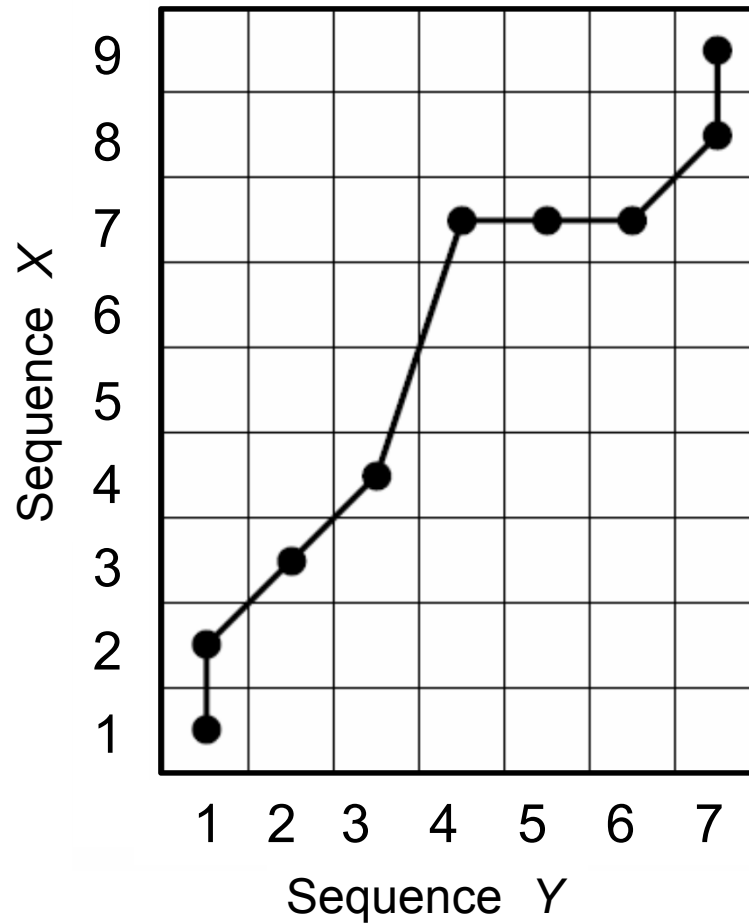
Warping path



Violation of  
monotonicity condition

# Dynamic Time Warping

Warping path



Violation of  
step size condition



# Dynamic Time Warping

The **total cost**  $c_p(X, Y)$  of a warping path  $p$  between  $X$  and  $Y$  with respect to the local cost measure  $c$  is defined as

$$c_p(X, Y) := \sum_{\ell=1}^L c(x_{n_\ell}, y_{m_\ell})$$

Furthermore, an **optimal warping path** between  $X$  and  $Y$  is a warping path  $p^*$  having minimal total cost among all possible warping paths. The **DTW distance**  $\text{DTW}(X, Y)$  between  $X$  and  $Y$  is then defined as the total cost of  $p^*$

$$\begin{aligned} \text{DTW}(X, Y) &:= c_{p^*}(X, Y) \\ &= \min\{c_p(X, Y) \mid p \text{ is a warping path}\} \end{aligned}$$

# Dynamic Time Warping

- The warping path  $p^*$  is not unique (in general).
- DTW does (in general) not define a metric since it may not satisfy the triangle inequality.
- There exist exponentially many warping paths.
- How can  $p^*$  be computed efficiently?

# Dynamic Time Warping

**Notation:**

$$\begin{aligned} X(1 : n) &:= (x_1, \dots, x_n), & 1 \leq n \leq N \\ Y(1 : m) &:= (y_1, \dots, y_m), & 1 \leq m \leq M \\ D(n, m) &:= \text{DTW}(X(1 : n), Y(1 : m)) \end{aligned}$$

The matrix  $D$  is called the **accumulated cost matrix**.

The entry  $D(n, m)$  specifies the cost of an optimal warping path that aligns  $X(1 : n)$  with  $Y(1 : m)$ .

# Dynamic Time Warping

**Lemma:**

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

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

$$(iii) \quad D(n, 1) = \sum_{k=1}^n C(k, 1)$$

$$D(1, m) = \sum_{k=1}^m C(1, k)$$

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

for  $n > 1, m > 1$

**Proof:** (i) – (iii) are clear by definition

# Dynamic Time Warping

**Proof** of (iv): Induction via  $n, m$  :

Let  $n > 1, m > 1$  and  $q = (q_1, \dots, p_{L-1}, p_L)$  be an optimal warping path for  $X(1 : n)$  and  $Y(1 : m)$ . Then  $q_L = (n, m)$  (boundary condition).

Let  $q_{L-1} = (a, b)$ . The step size condition implies

$$(a, b) \in \{(n-1, m-1), (n-1, m), (n, m-1)\}$$

The warping path  $(q_1, \dots, q_{L-1})$  must be optimal for  $X(1 : a), Y(1 : b)$ . Thus,

$$D(n, m) = c_{(q_1, \dots, q_{L-1})}(X(1 : a), Y(1 : b)) + C(n, m)$$



# Dynamic Time Warping

## Accumulated cost matrix

Given the two feature sequences  $X$  and  $Y$ , the matrix  $D$  is computed recursively.

- Initialize  $D$  using (ii) and (iii) of the lemma.
- Compute  $D(n, m)$  for  $n > 1, m > 1$  using (iv).
- $\text{DTW}(X, Y) = D(N, M)$  using (i).

### Note:

- Complexity  $O(NM)$ .
- Dynamic programming: “overlapping-subproblem property”

# Dynamic Time Warping

## Optimal warping path

Given to the algorithm is the accumulated cost matrix  $D$ . The optimal path  $p^* = (p_1, \dots, p_L)$  is computed in reverse order of the indices starting with  $p_L = (N, M)$ .

Suppose  $p_\ell = (n, m)$  has been computed. In case  $(n, m) = (1, 1)$ , one must have  $\ell = 1$  and we are done.

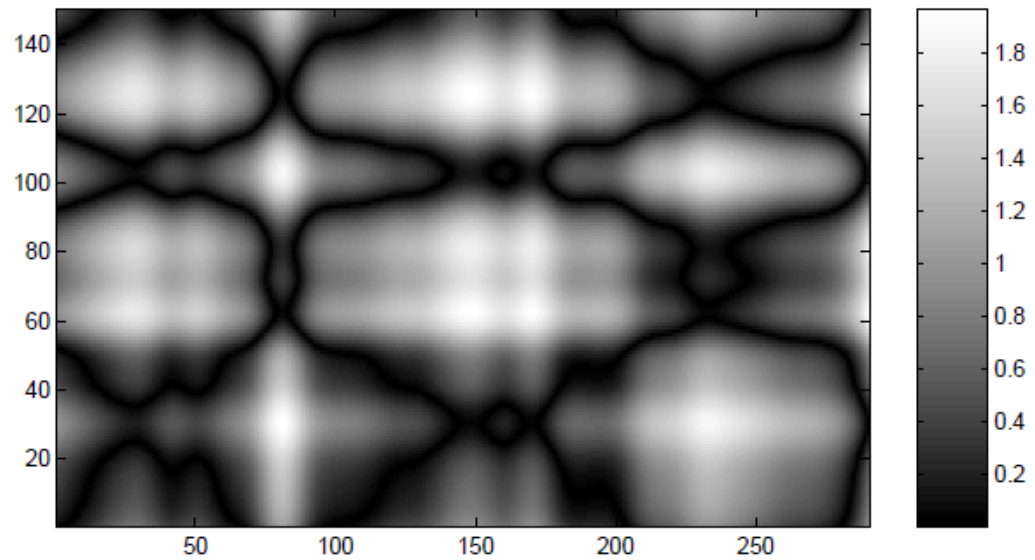
Otherwise,

$$p_{\ell-1} := \begin{cases} (1, m-1), & \text{if } n = 1 \\ (n-1, 1), & \text{if } m = 1 \\ \operatorname{argmin}\{D(n-1, m-1), \\ \quad D(n-1, m), D(n, m-1)\}, & \text{otherwise,} \end{cases}$$

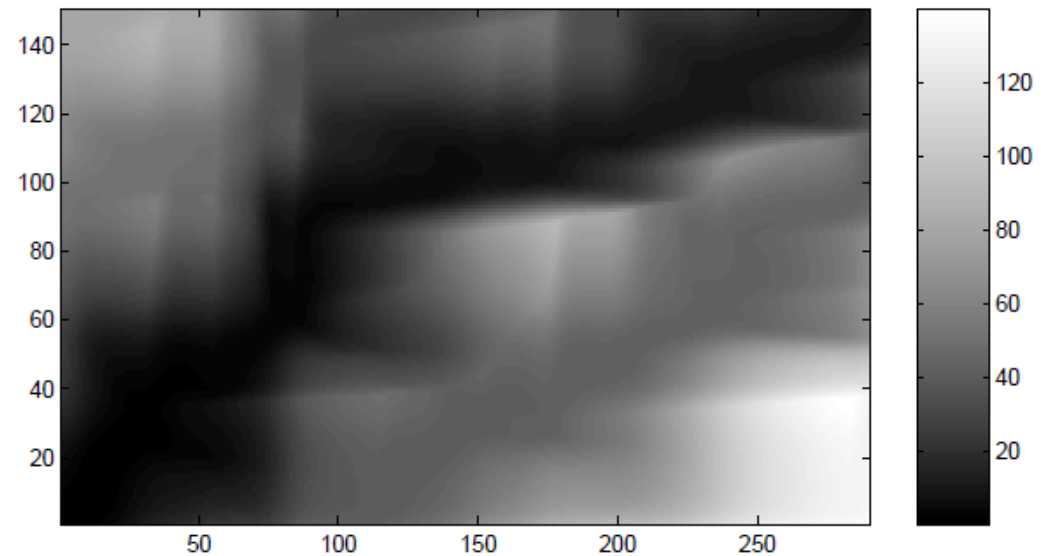
where we take the lexicographically smallest pair in case “argmin” is not unique.

# Dynamic Time Warping

Cost matrix  $C$



Accumulated cost matrix  $D$

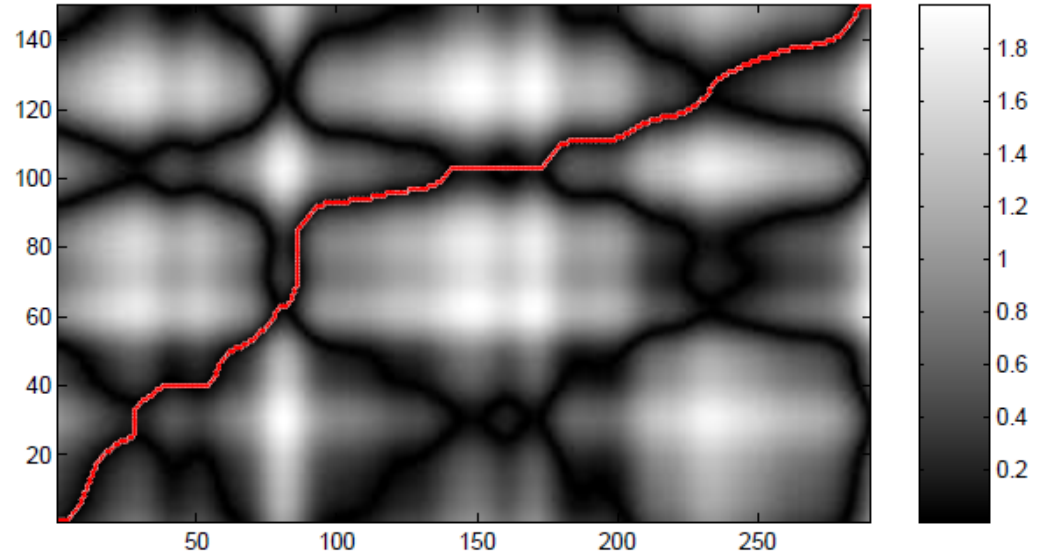




# Dynamic Time Warping

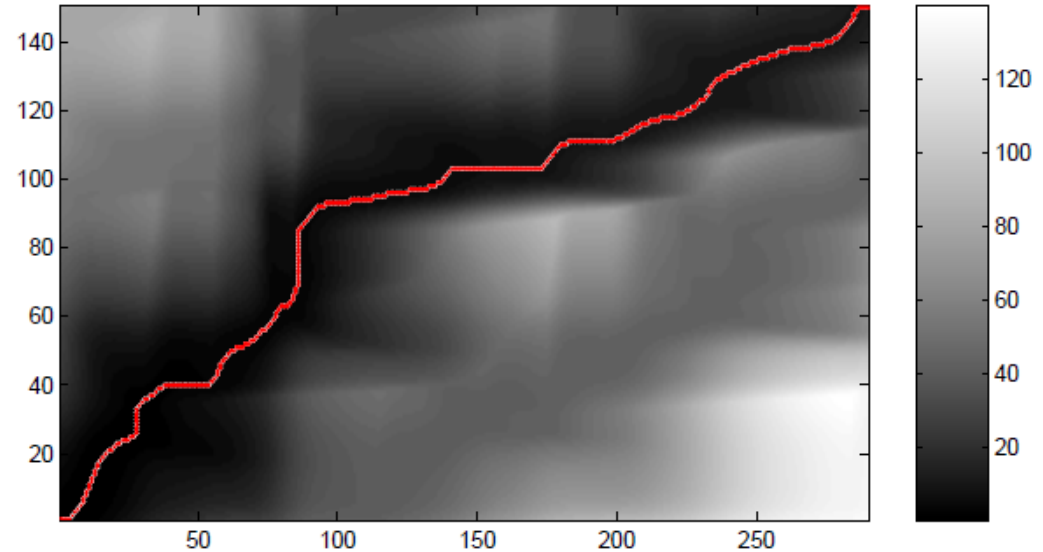
Cost matrix  $C$

Optimal warping path



Accumulated  
cost matrix  $D$

Optimal warping path



# Dynamic Time Warping

1	1	1	1	7	6	1
8	6	8	8	0	1	6
3	1	3	3	5	4	1
3	1	3	3	5	4	1
1	1	1	1	7	6	1

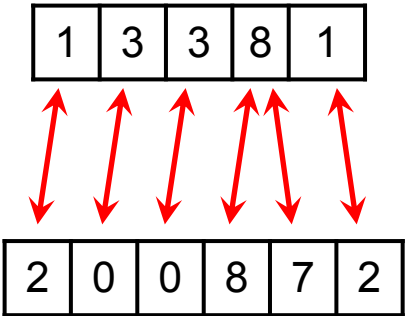
  

2	0	0	8	7	2
---	---	---	---	---	---

1	10	10	11	14	13	9
8	9	11	13	7	8	14
3	3	5	7	10	12	13
3	2	4	5	8	12	13
1	1	2	3	10	16	17

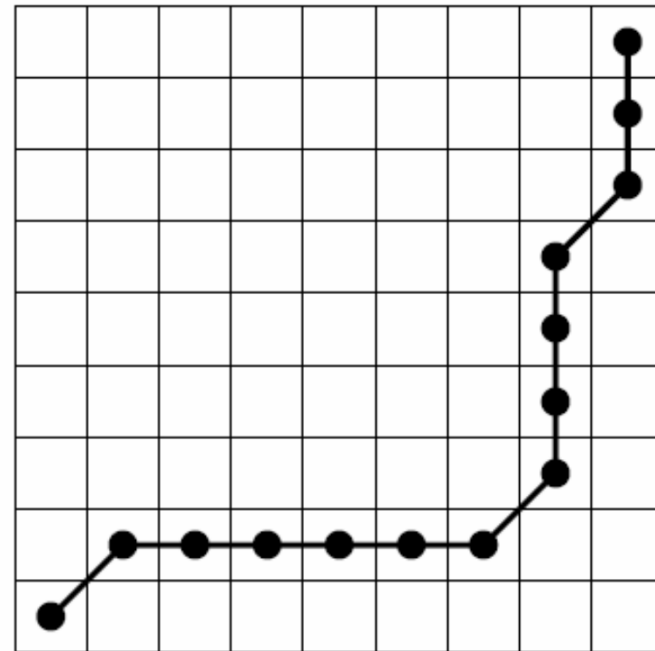
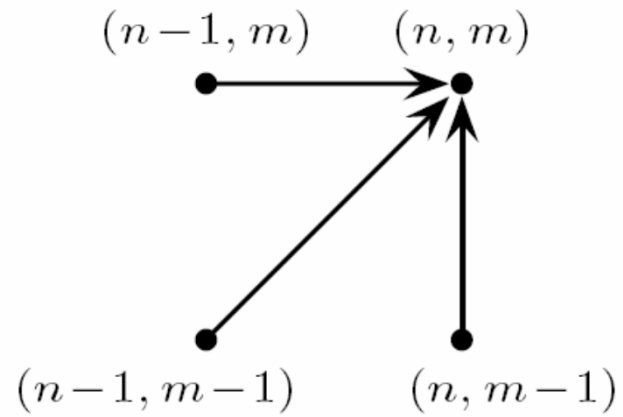
  

2	0	0	8	7	2
---	---	---	---	---	---



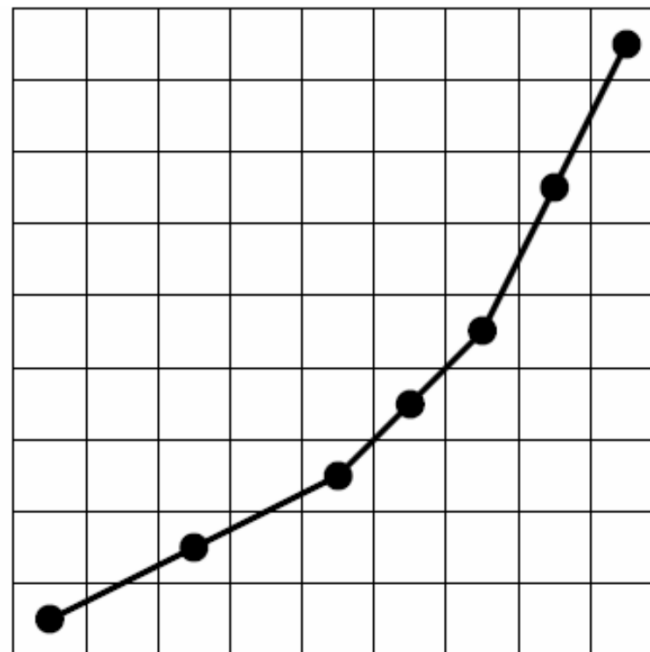
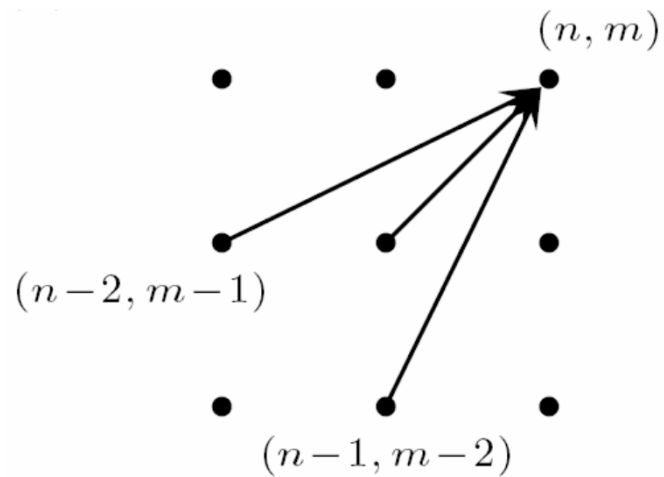
# Dynamic Time Warping

Variation of step size condition



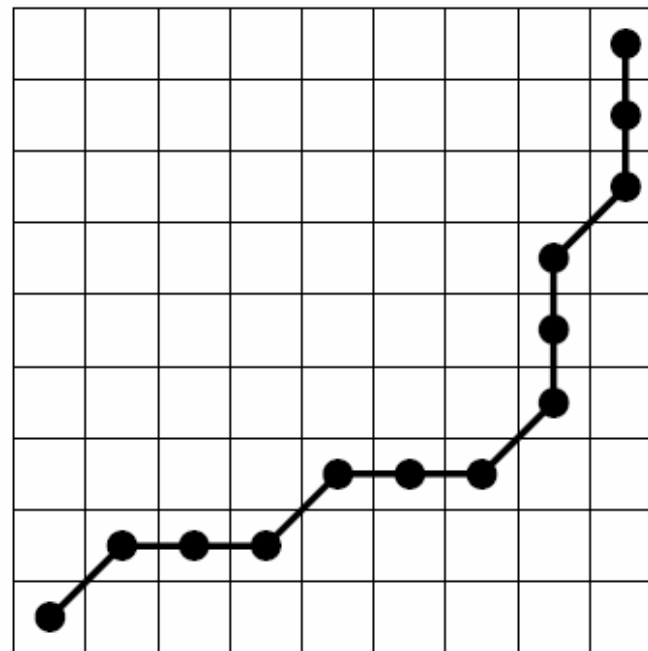
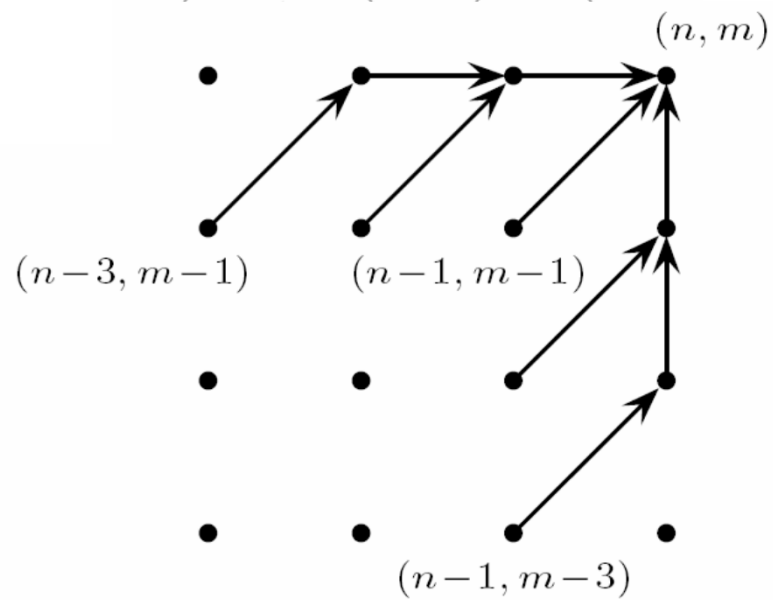
# Dynamic Time Warping

Variation of step size condition



# Dynamic Time Warping

## Variation of step size condition



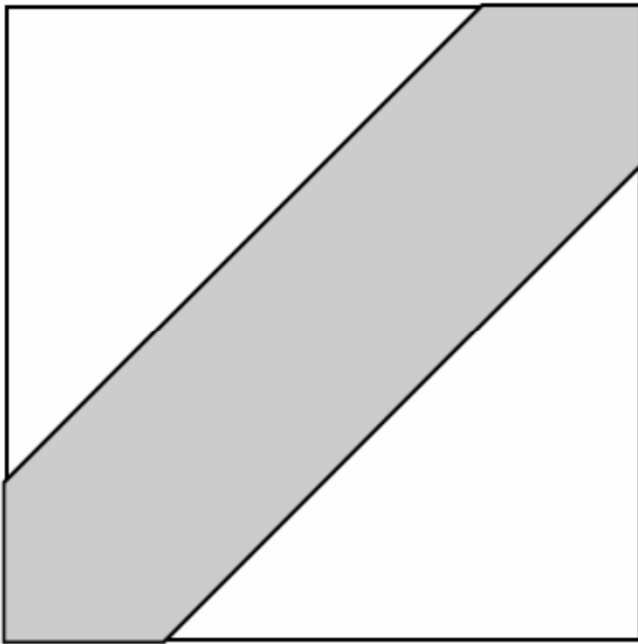
# Dynamic Time Warping

- Computation via dynamic programming
- Memory requirements and running time:  $O(NM)$
- **Problem: Infeasible for large  $N$  and  $M$**
- Example: Feature resolution 10 Hz, pieces 15 min
  - $\Rightarrow N, M \sim 10,000$
  - $\Rightarrow N \cdot M \sim 100,000,000$

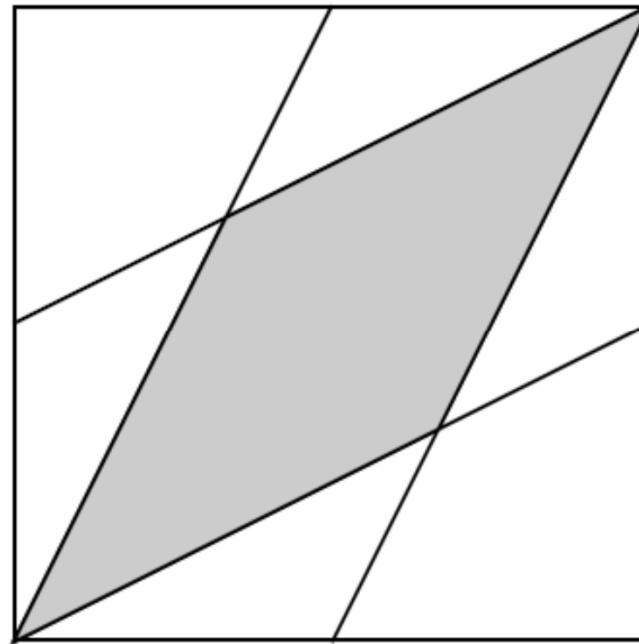
# Dynamic Time Warping

Strategy: Global constraints

Sakoe-Chiba band



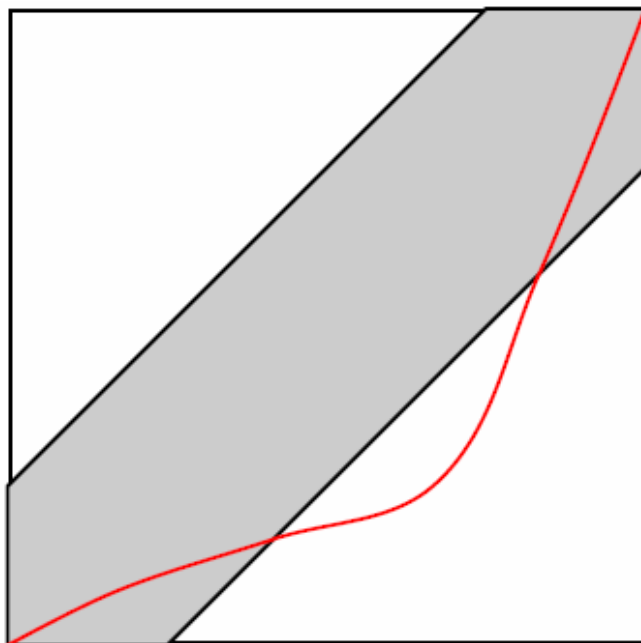
Itakura parallelogram



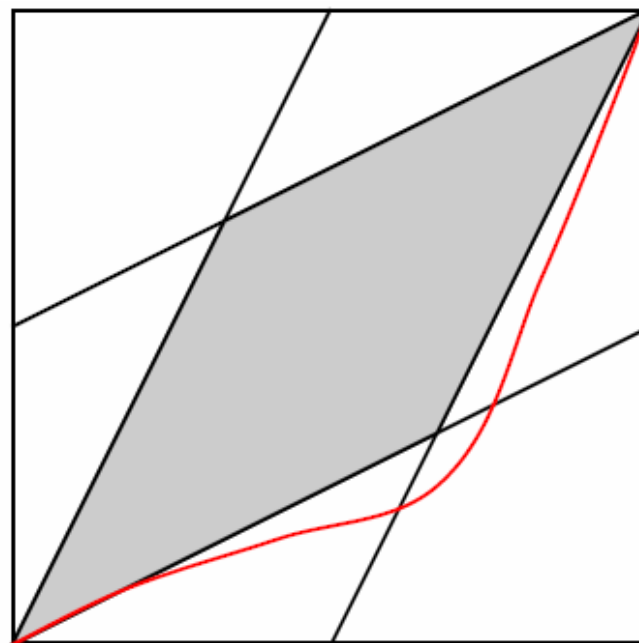
# Dynamic Time Warping

Strategy: Global constraints

Sakoe-Chiba band



Itakura parallelogram

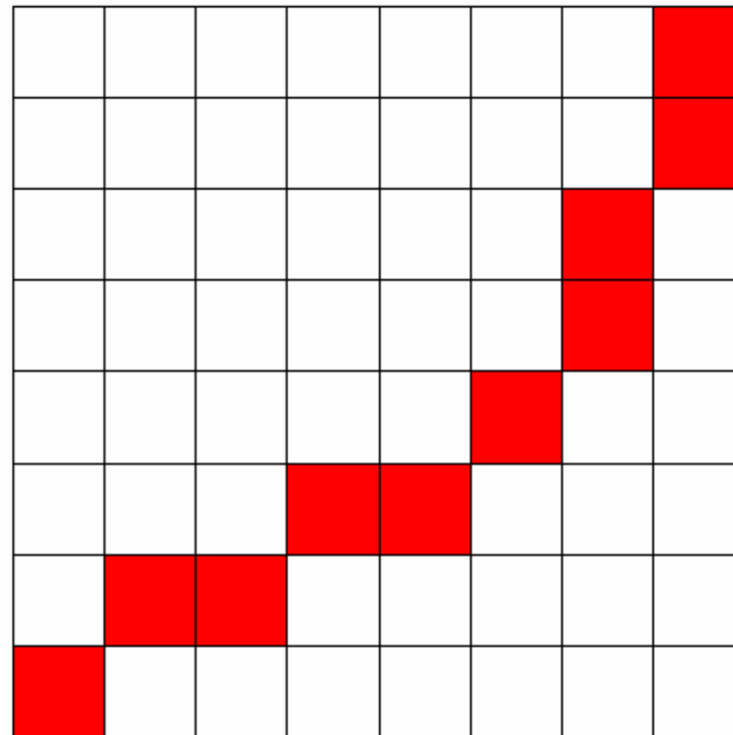


Problem: Optimal warping path not in constraint region



# Dynamic Time Warping

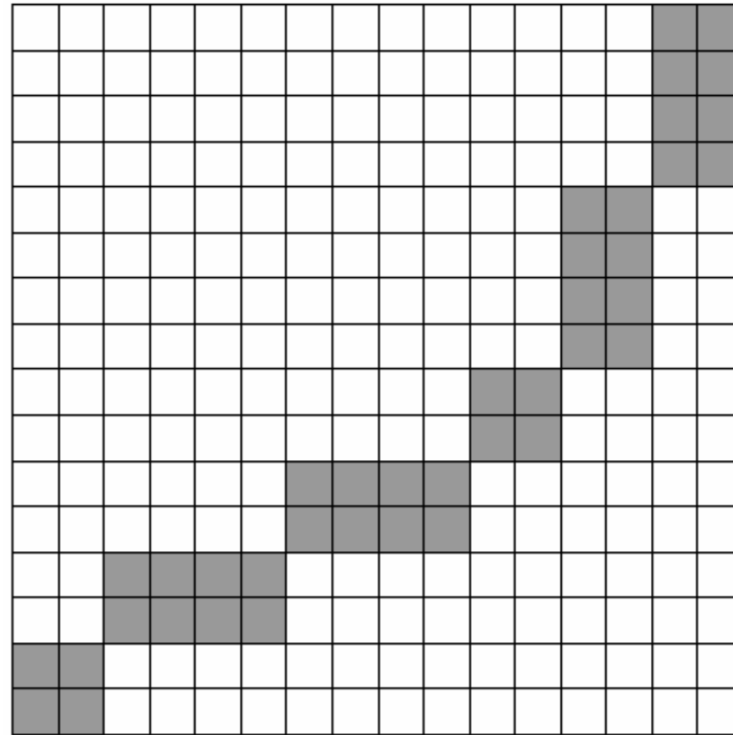
Strategy: Multiscale approach



Compute optimal warping path on coarse level

# Dynamic Time Warping

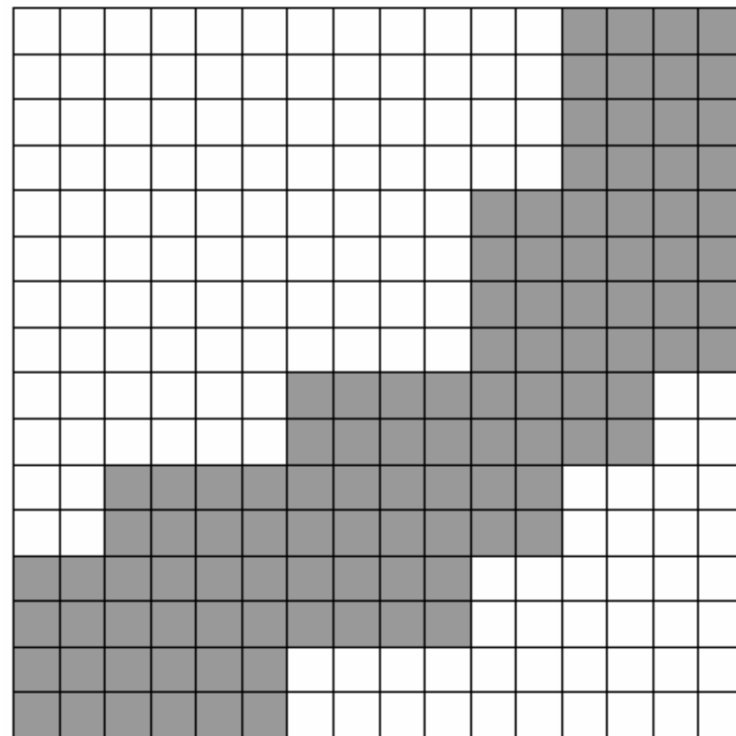
Strategy: Multiscale approach



Project on fine level

# Dynamic Time Warping

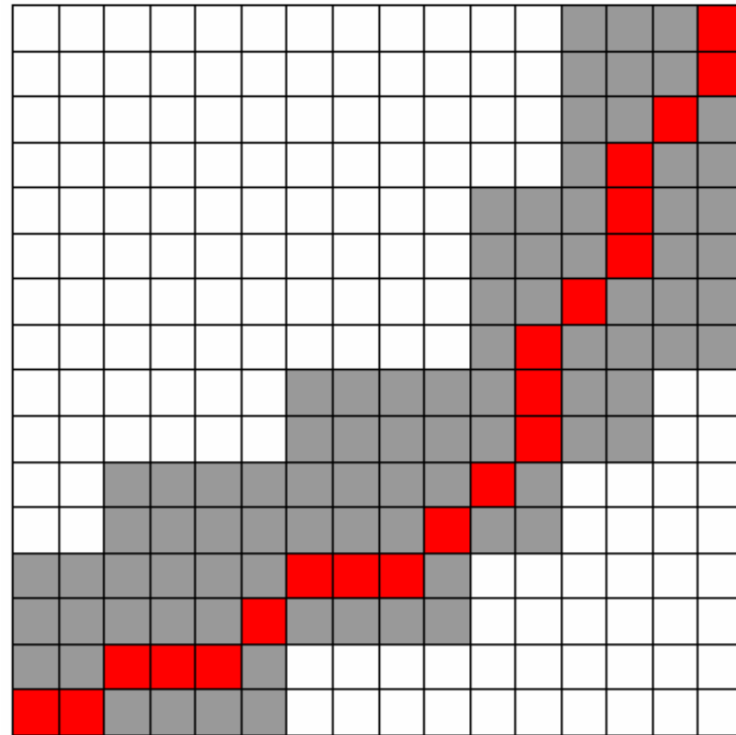
Strategy: Multiscale approach



Specify constraint region

# Dynamic Time Warping

Strategy: Multiscale approach



Compute *constrained* optimal warping path

# Dynamic Time Warping

Strategy: Multiscale approach

- Suitable features?
- Suitable resolution levels?
- Size of constraint regions?

Good trade-off between efficiency and robustness?

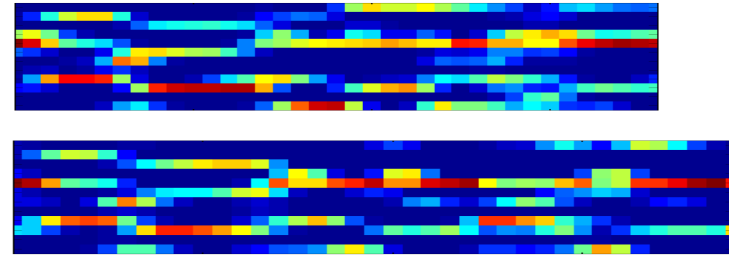
Suitable parameters depend very much on application!

# Music Synchronization: Audio-Audio

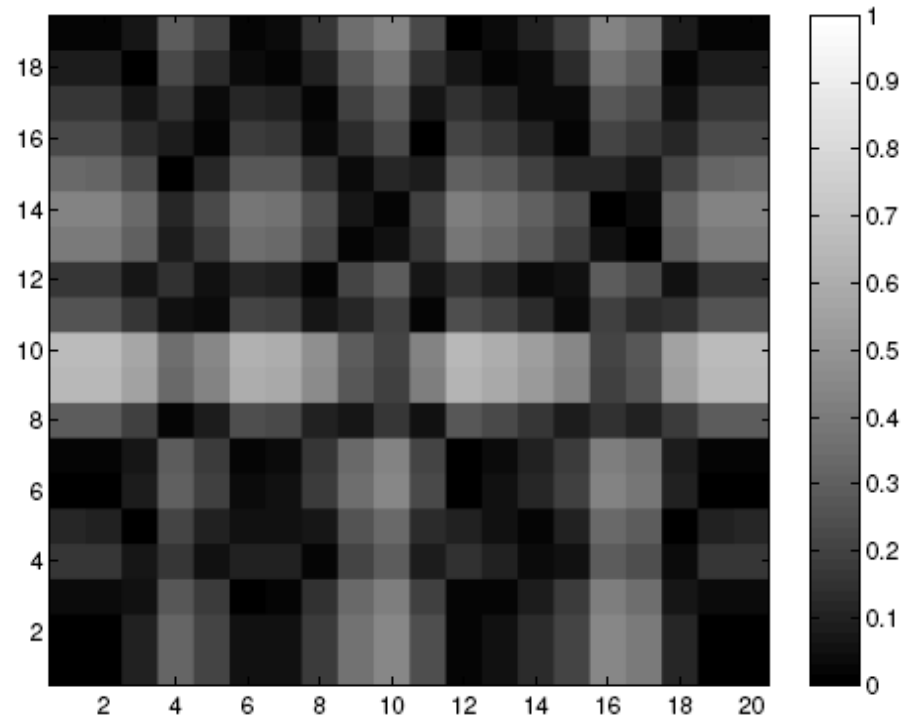
- Transform audio recordings into chroma vector sequences

$$\rightsquigarrow X := (x_1, x_2, \dots, x_N)$$

$$\rightsquigarrow Y := (y_1, y_2, \dots, y_M)$$



- Compute cost matrix  $C(n, m) := c(x_n, y_m)$  with respect to local cost measure  $c$

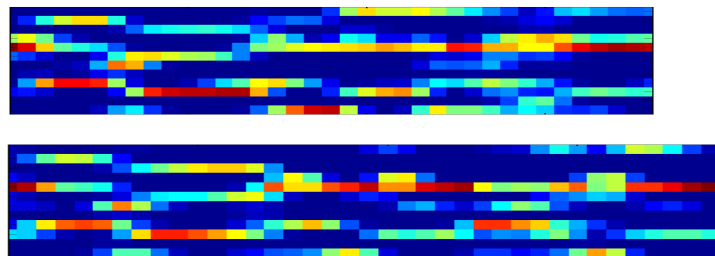


# Music Synchronization: Audio-Audio

- Transform audio recordings into chroma vector sequences

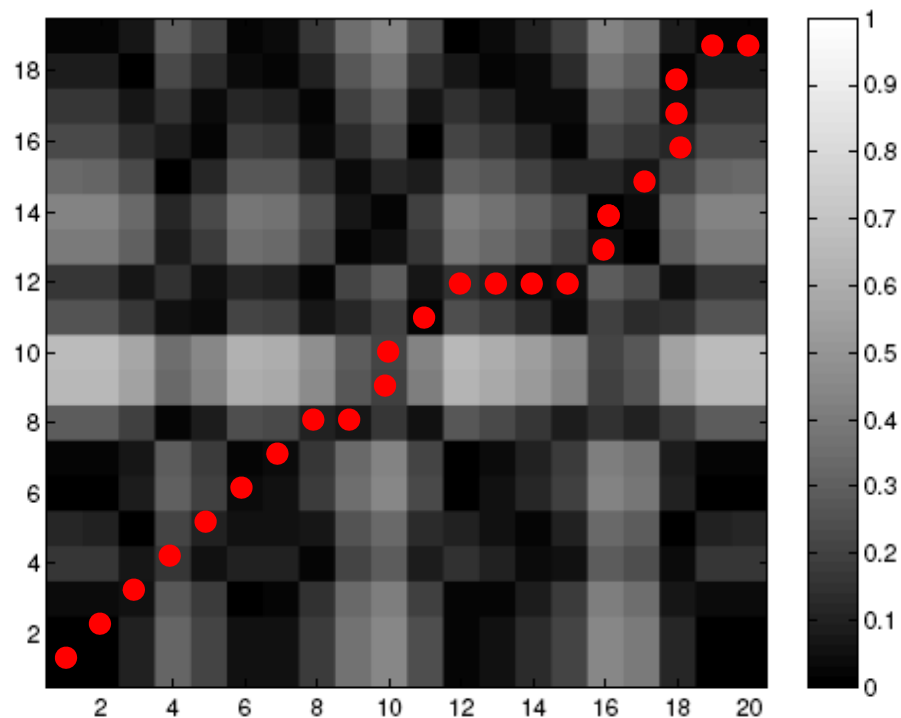
$$\rightsquigarrow X := (x_1, x_2, \dots, x_N)$$

$$\rightsquigarrow Y := (y_1, y_2, \dots, y_M)$$

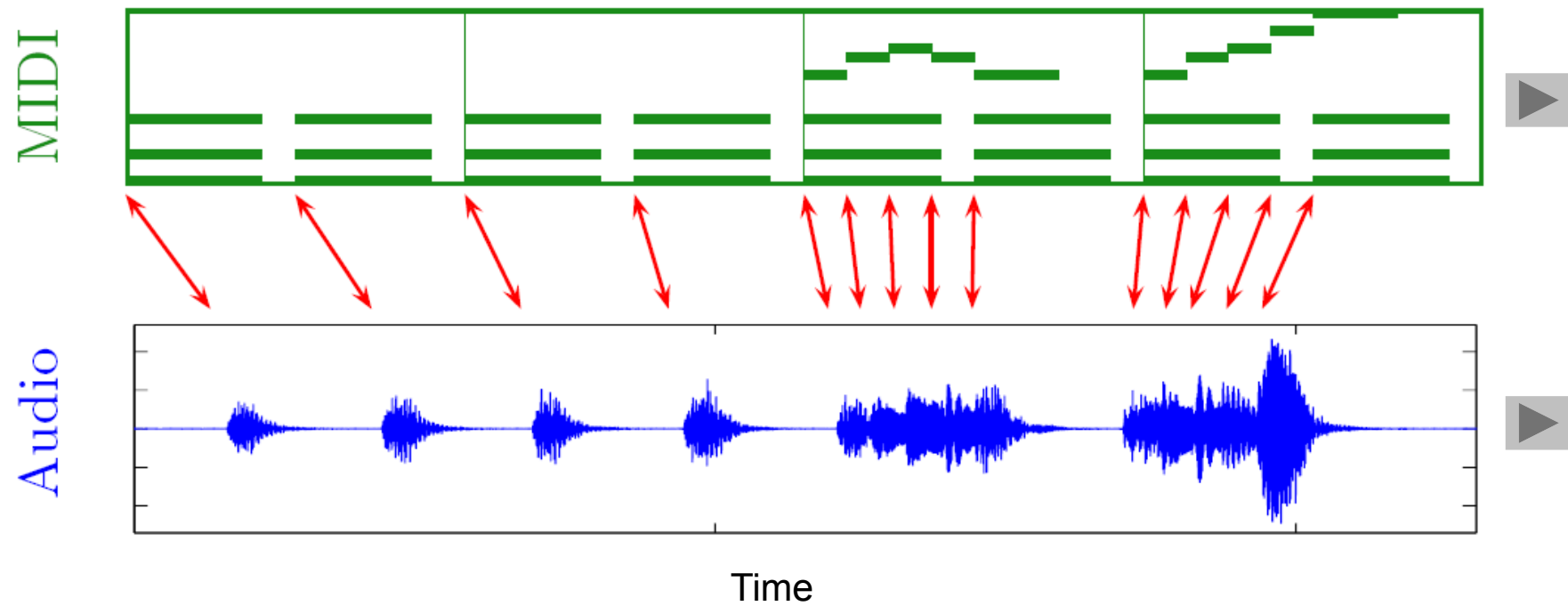


- Compute cost matrix  $C(n, m) := c(x_n, y_m)$  with respect to local cost measure  $c$

- Compute **cost-minimizing warping path** from  $C$



# Music Synchronization: MIDI-Audio





---

# Music Synchronization: MIDI-Audio

MIDI = meta data

Automated annotation

Audio recording

Sonification of annotations



---

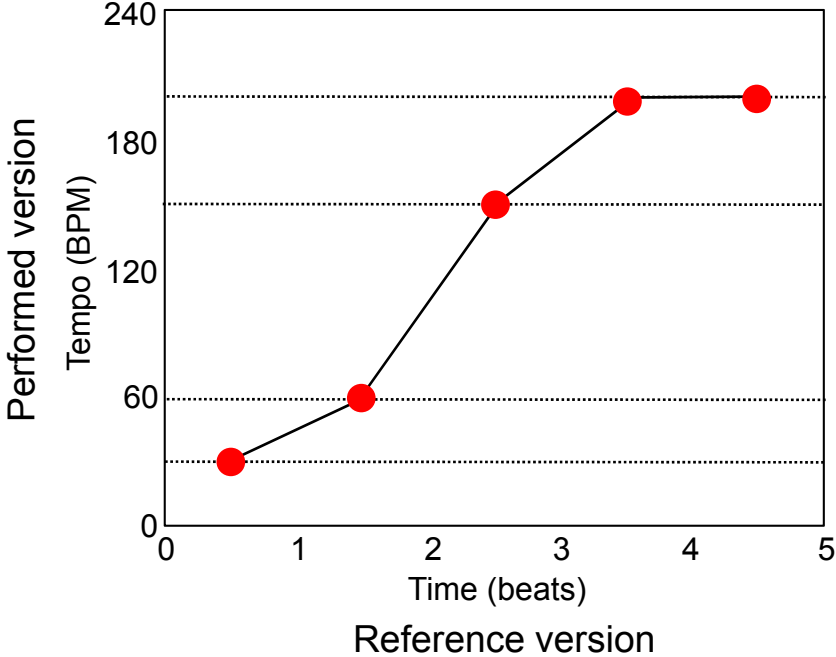
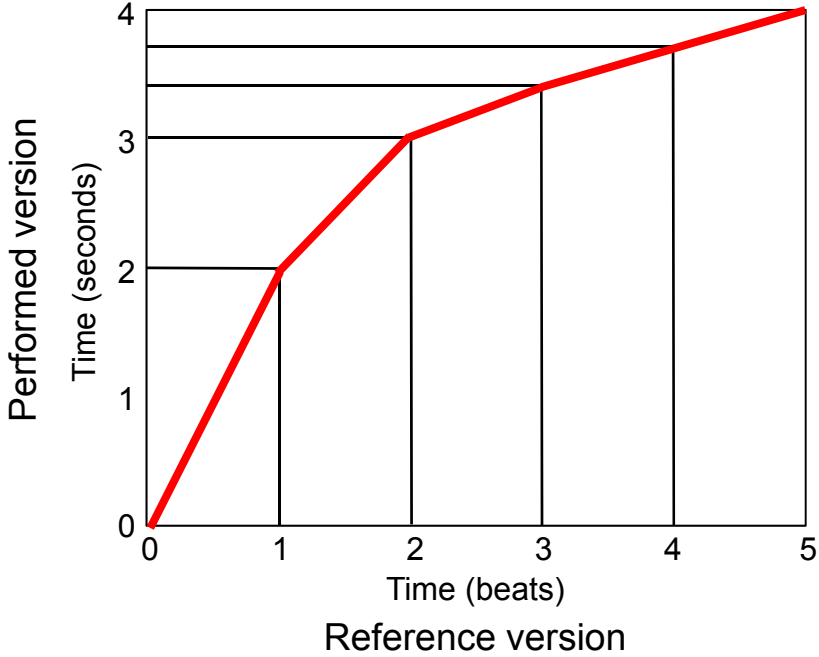
# Music Synchronization: MIDI-Audio

MIDI = reference (score)

Tempo information

Audio recording

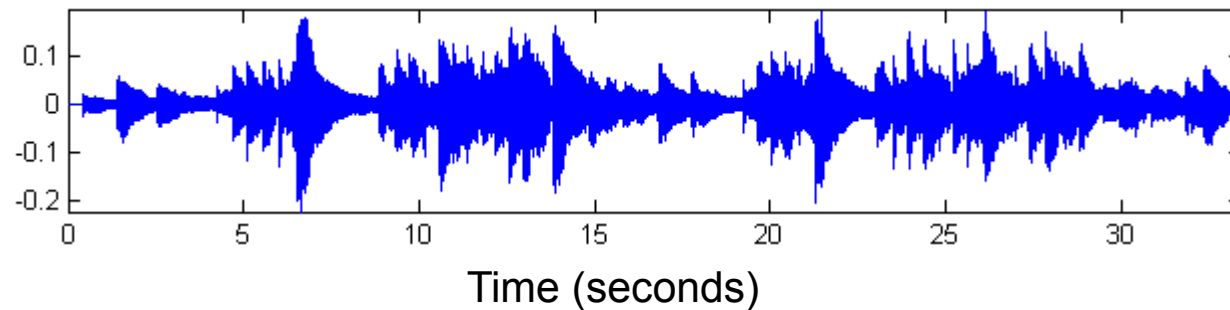
# Performance Analysis: Tempo Curves



# Performance Analysis: Tempo Curves

Schumann: Träumerei

Performance:



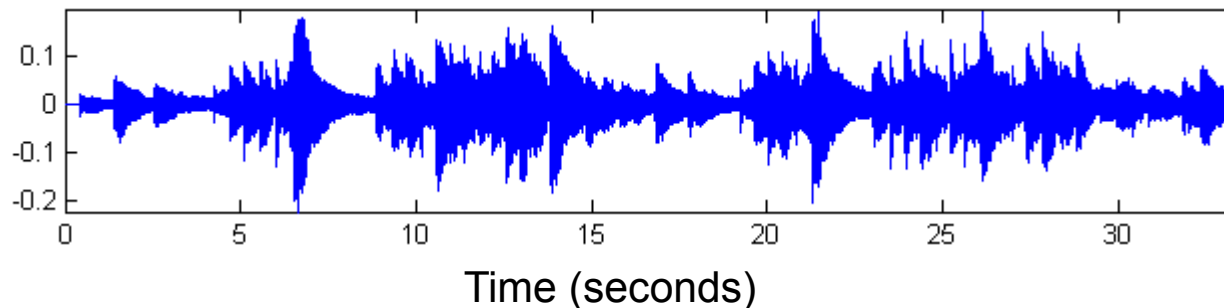
# Performance Analysis: Tempo Curves

Schumann: Träumerei

Score (reference):



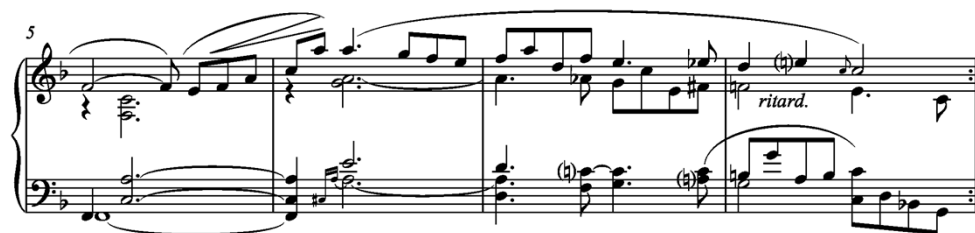
Performance:



# Performance Analysis: Tempo Curves

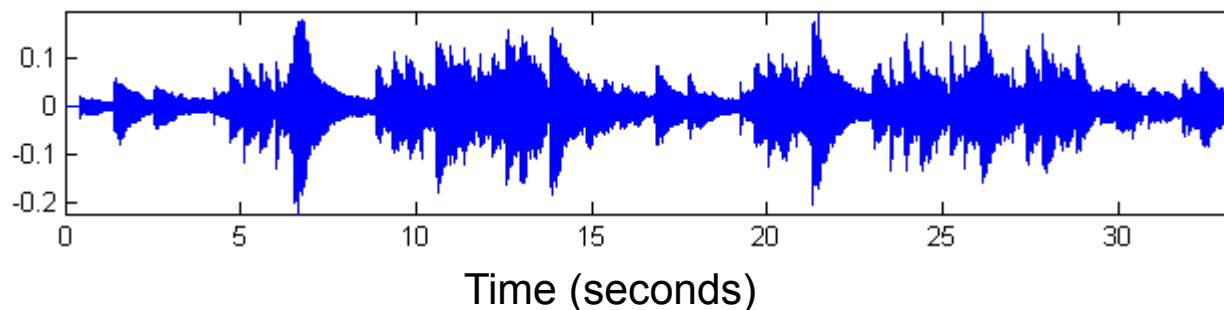
Schumann: Träumerei

Score (reference):



**Strategy: Compute score-audio synchronization and derive tempo curve**

Performance:



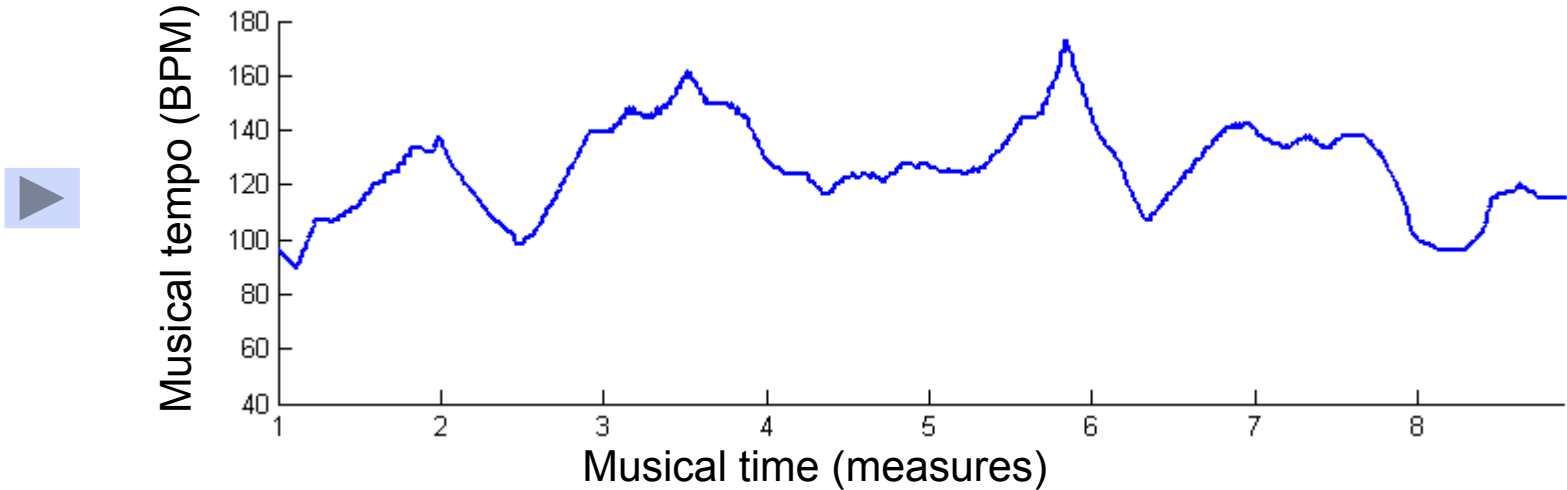
# Performance Analysis: Tempo Curves

Schumann: Träumerei

Score (reference):



Tempo curve:



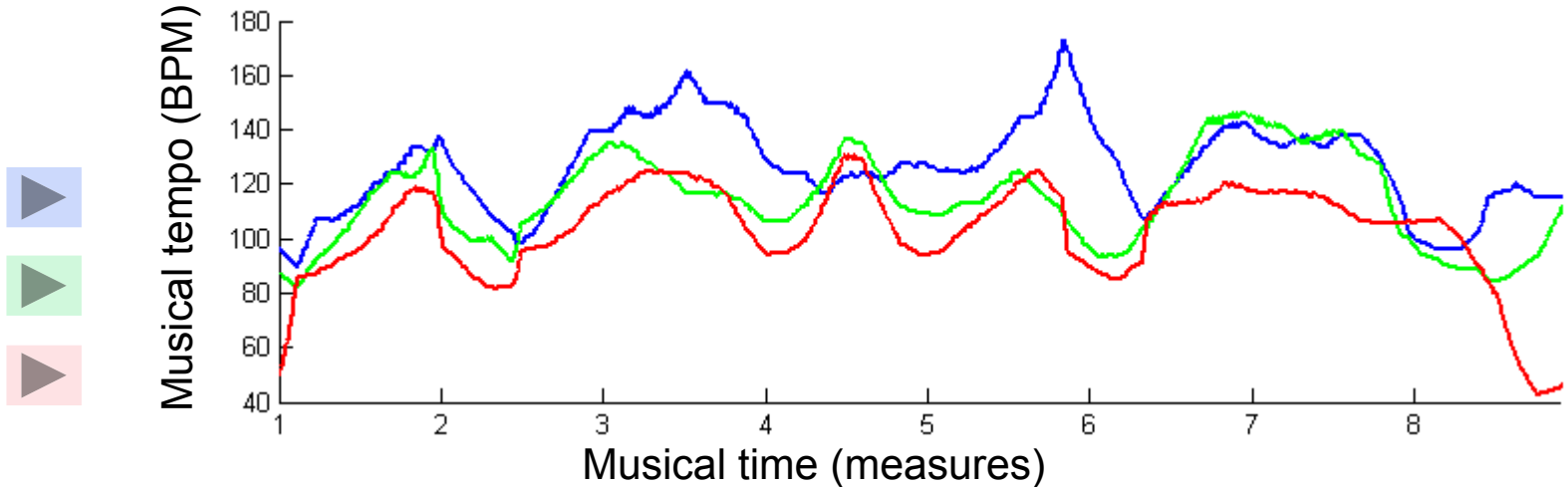
# Performance Analysis: Tempo Curves

Schumann: Träumerei

Score (reference):



Tempo curves:





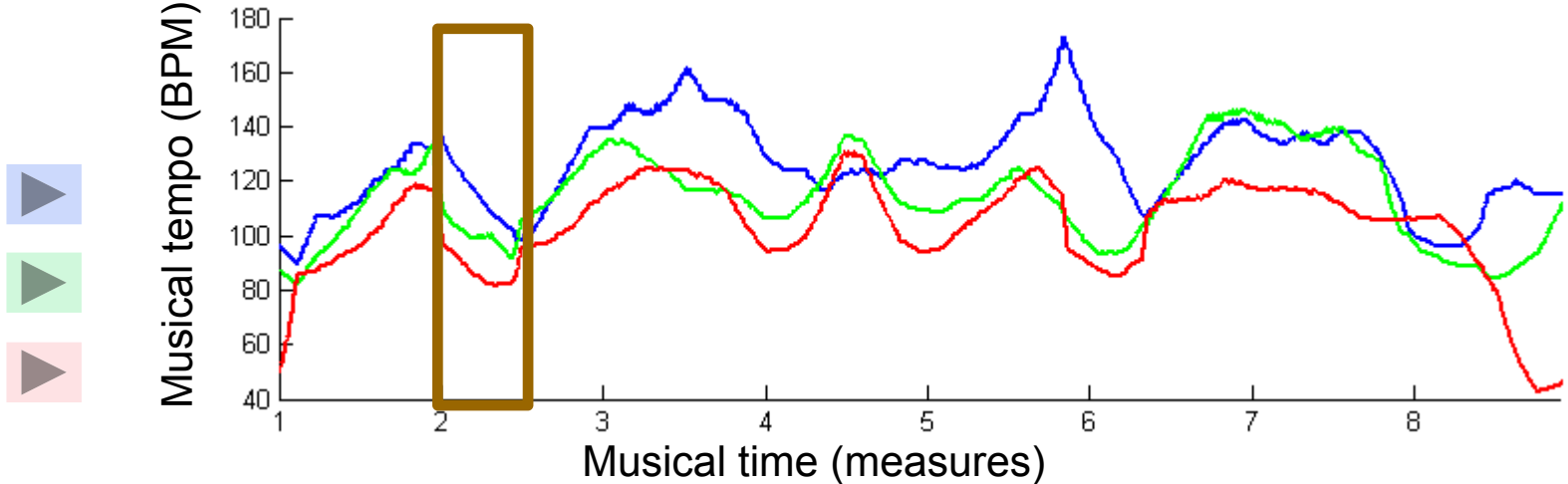
# Performance Analysis: Tempo Curves

Schumann: Träumerei

Score (reference):



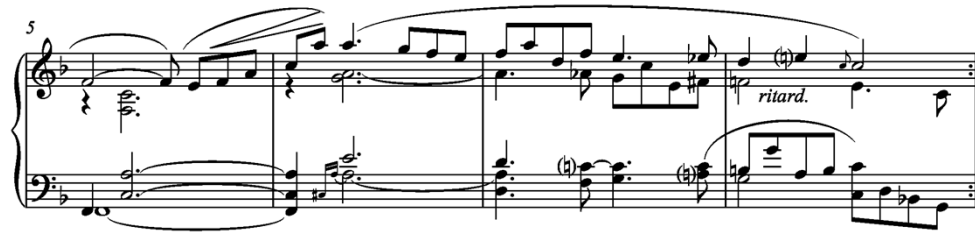
Tempo curves:



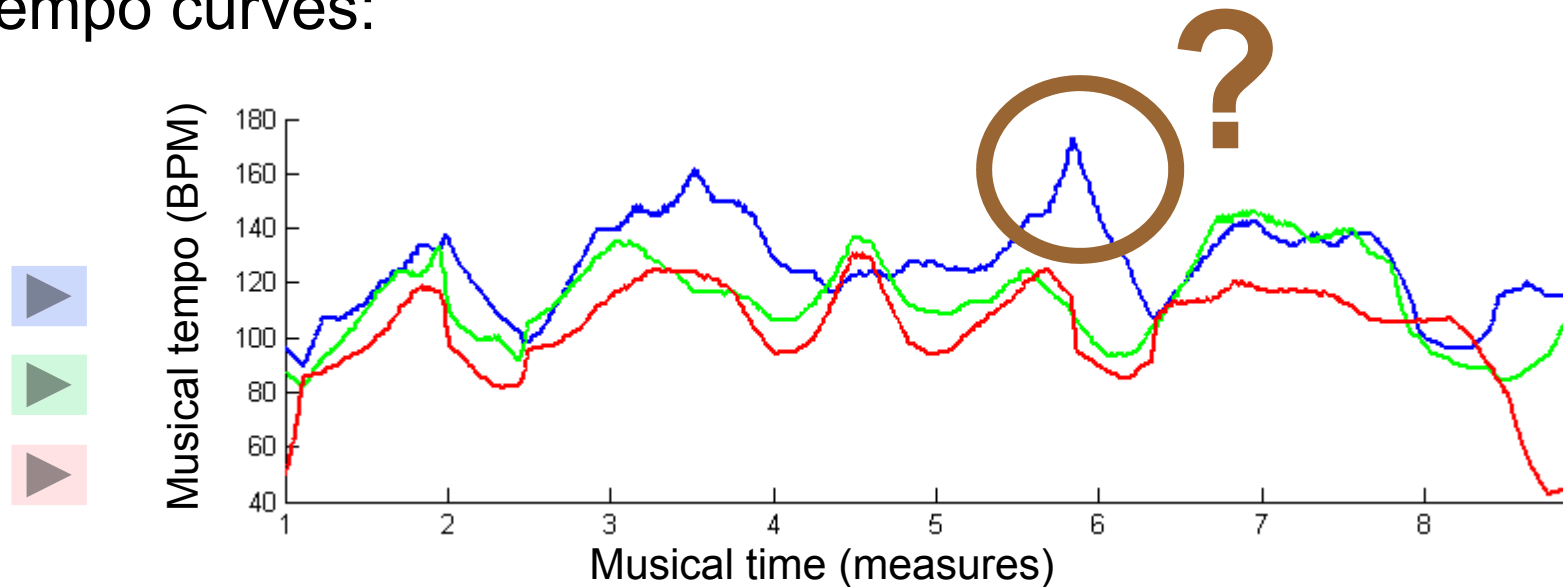
# Performance Analysis: Tempo Curves

Schumann: Träumerei

Score (reference):



Tempo curves:

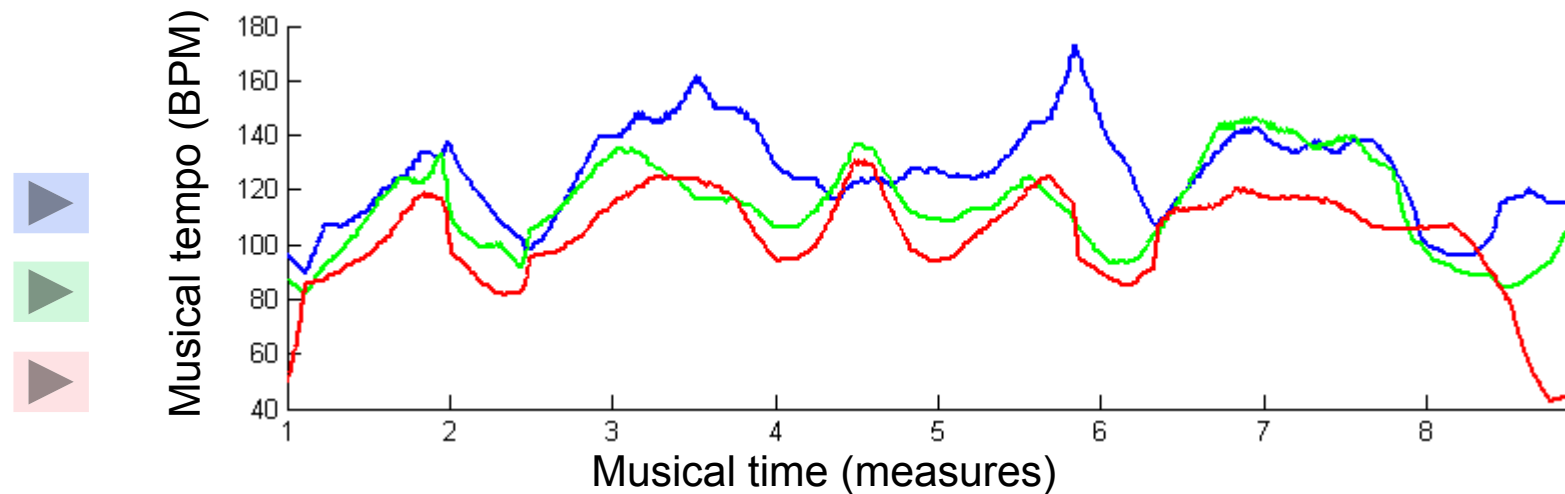


# Performance Analysis: Tempo Curves

Schumann: Träumerei

**What can be done if no reference is available?**

Tempo curves:



# Music Synchronization: MIDI-Audio

## Applications

- Automated audio annotation
- Accurate audio access after MIDI-based retrieval
- Automated tracking of MIDI note parameters during audio playback
- Performance Analysis

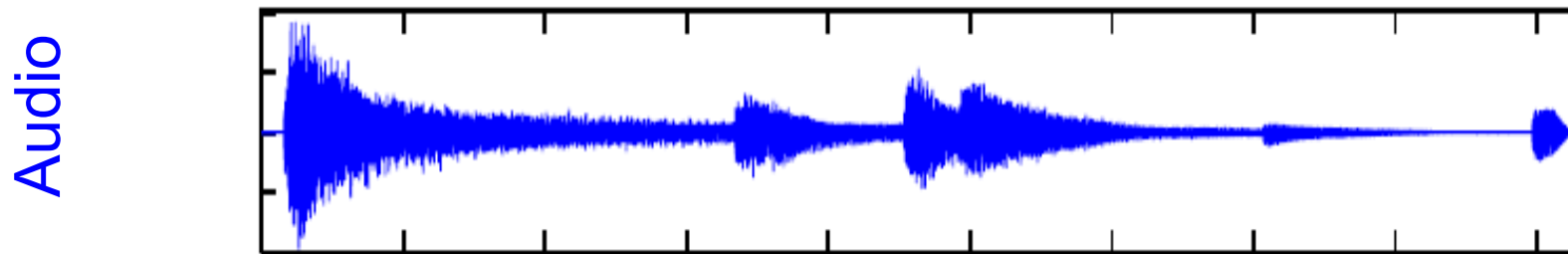
# Music Synchronization: Image-Audio

Grave.

Image



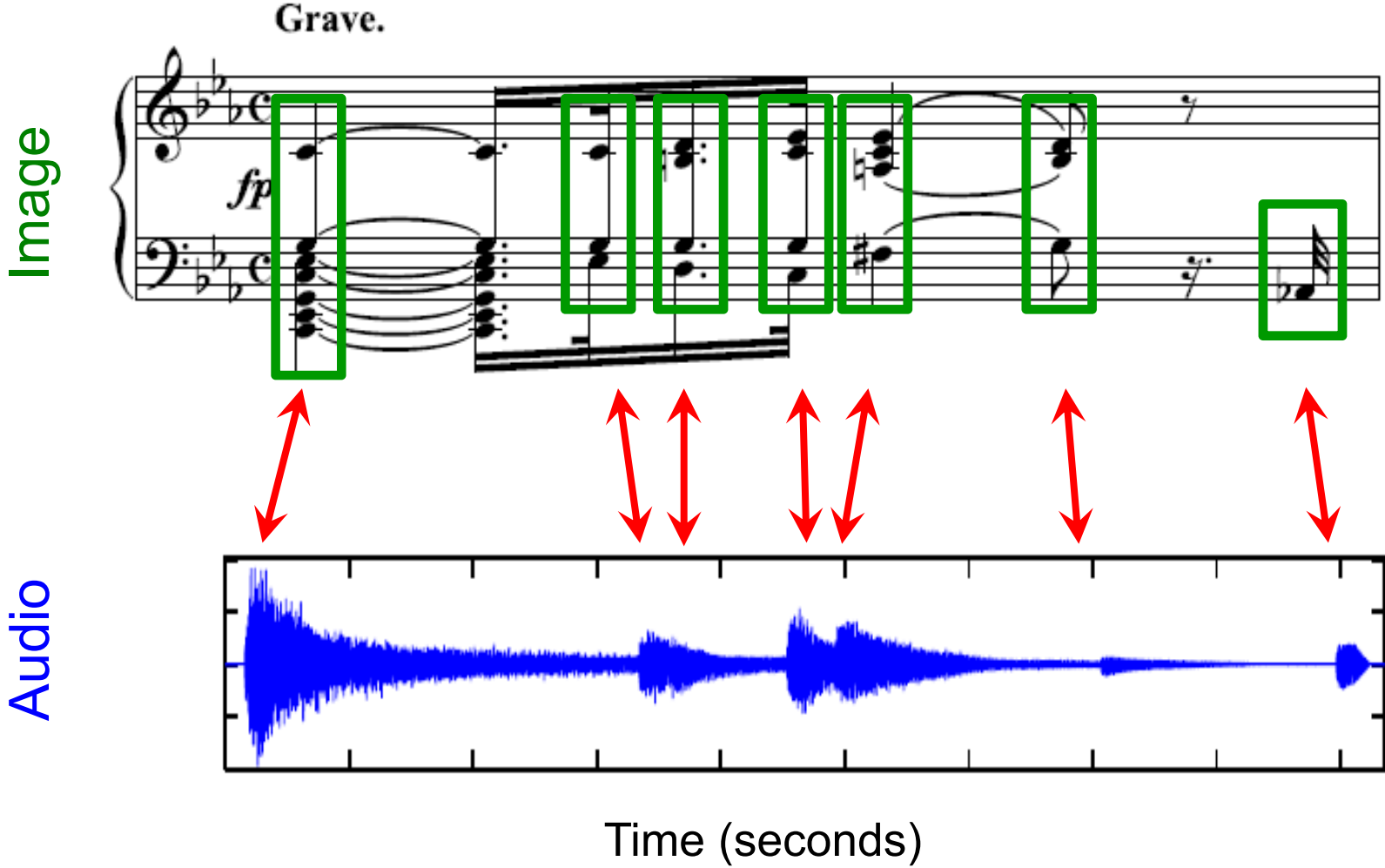
The image shows a musical score for piano, marked "Grave." and "fp". The score is written in a grand staff with a treble and bass clef. The key signature has two flats (B-flat and E-flat), and the time signature is common time (C). The music features a slow, somber melody with a prominent bass line. The first measure is marked "fp" (fortissimo piano). The score includes various musical notations such as slurs, ties, and dynamic markings.



Time (seconds)



# Music Synchronization: Image-Audio



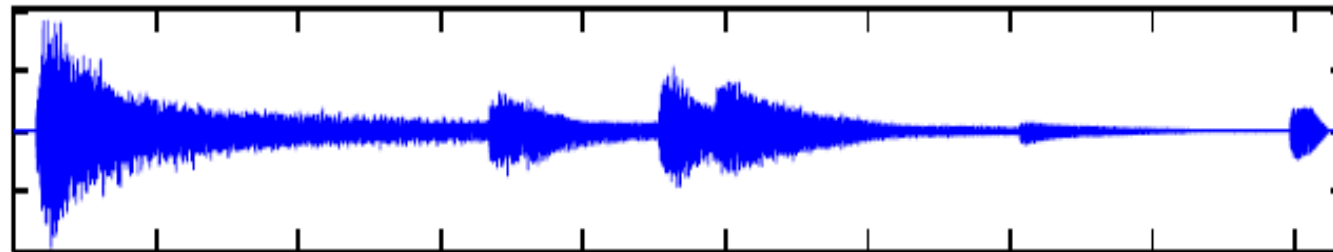
# Music Synchronization: Image-Audio

Convert data into common mid-level feature representation

Image



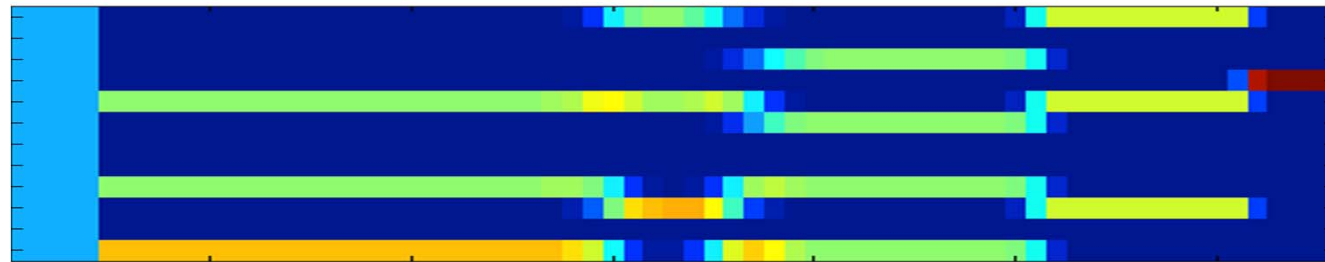
Audio



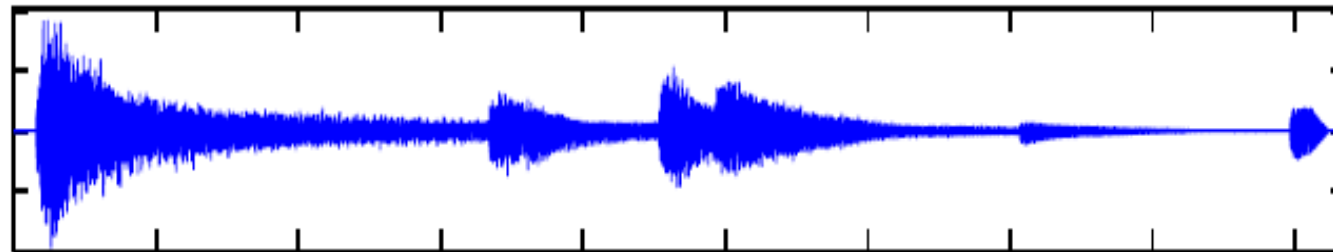
# Music Synchronization: Image-Audio

## Image Processing: Optical Music Recognition

Image



Audio

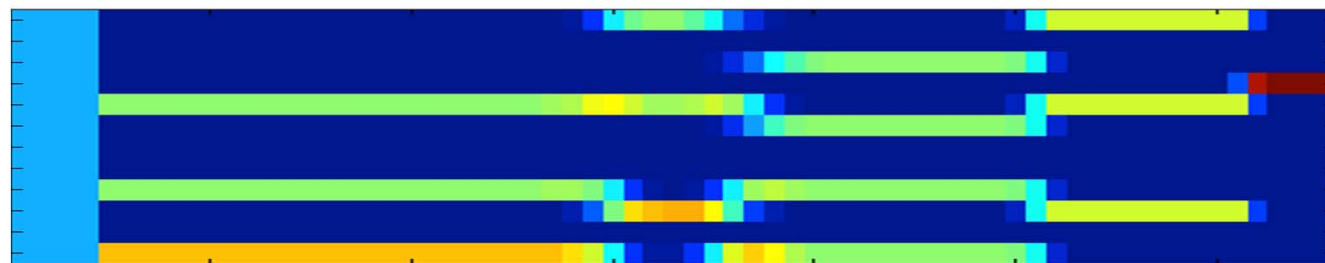




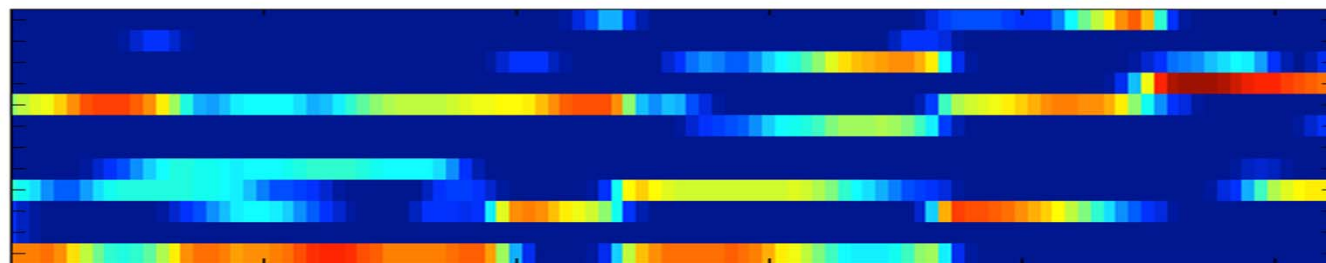
# Music Synchronization: Image-Audio

Image Processing: Optical Music Recognition

Image



Audio

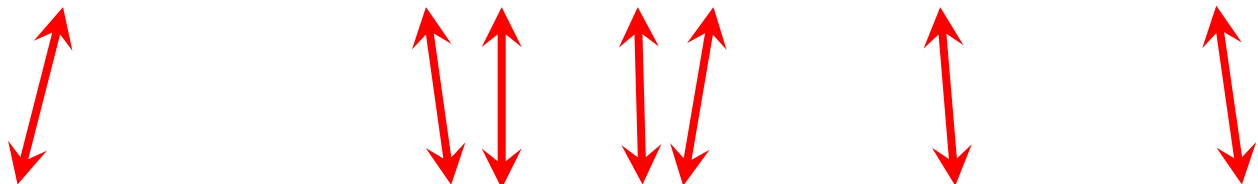
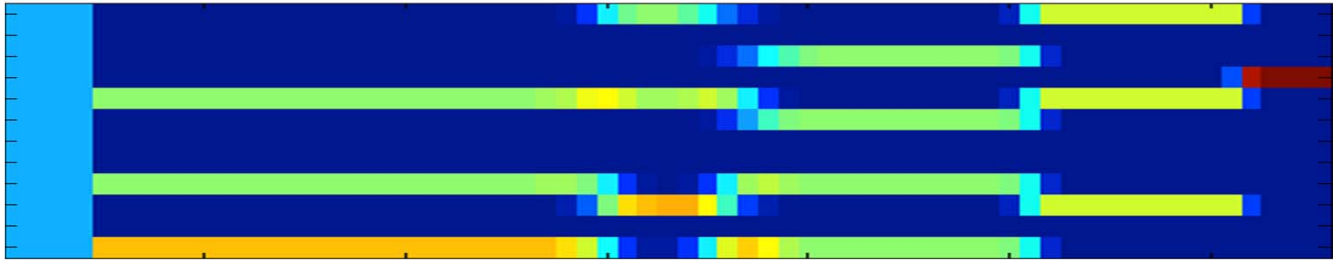


Audio Processing: Fourier Analyse

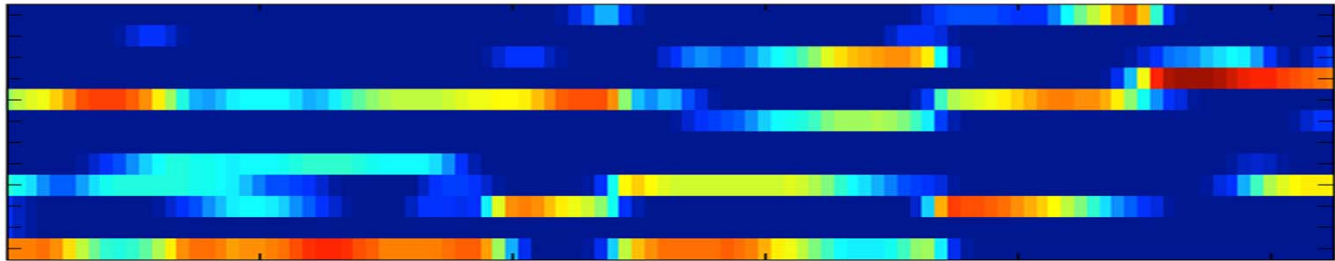
# Music Synchronization: Image-Audio

Image Processing: Optical Music Recognition

Image



Audio



Audio Processing: Fourier Analyse

# Music Synchronization: Image-Audio

Application: Score Viewer

The screenshot displays two overlapping windows from a music synchronization application. The top window, titled "AudioViewer", shows a CD cover for "Beethoven - Complete Piano Sonatas - Daniel Barenboim" and a tracklist. The bottom window, titled "ScoreViewer", displays a page of musical notation for "Beethoven - Klaviersonaten Band 1 - Henle".

**AudioViewer Window:**

- Title: Beethoven - Complete Piano Sonatas - Daniel Barenboim
- Disc: 3
- Tracklist:

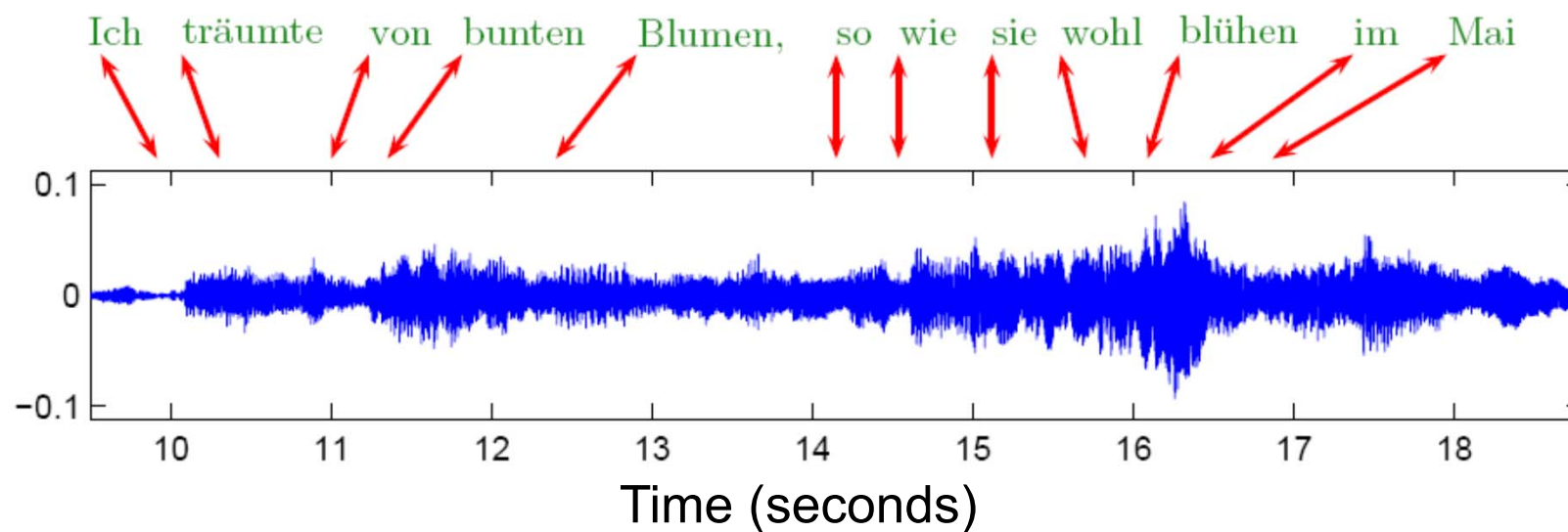
01	Sonata no.7 in D major, op.10 no.3: Presto	7:08
02	Sonata no.7 in D major, op.10 no.3: Largo e mesto	10:02
03	Sonata no.7 in D major, op.10 no.3: Menuetto (Allegro)	2:53
04	Sonata no.7 in D major, op.10 no.3: Rondo (Allegro)	4:05
05	Sonata no.8 in C minor, op.13, "Pathetique" / Allegro di molto e con brio	9:32
06	Sonata no.8 in C minor, op.13, "Pathetique" / Adagio cantabile	5:19
07	Sonata no.8 in C minor, op.13, "Pathetique" / Rondo (Allegro)	4:53
08	Sonata no.9 in E major, op.14 no.1: Allegro	6:48
09	Sonata no.9 in E major, op.14 no.1: Adagio	
10	Sonata no.9 in E major, op.14 no.1: Rondo	
- Disc: 3 / 10
- Track: 7 / 10

**ScoreViewer Window:**

- Title: Beethoven - Klaviersonaten Band 1 - Henle
- Subtitle: Sonata no.8 in C minor, op.13, "Pathetique" / Rondo (Allegro)
- Composer: Barenboim
- Score: Musical notation for the Rondo section of Sonata no. 8 in C minor, op. 13, "Pathetique".
- Track: 29 / 54
- Bar: 9 / 211
- Page: 159 / 285
- Score Following Off (indicated by a red X over the icon)
- Play button
- Stop button



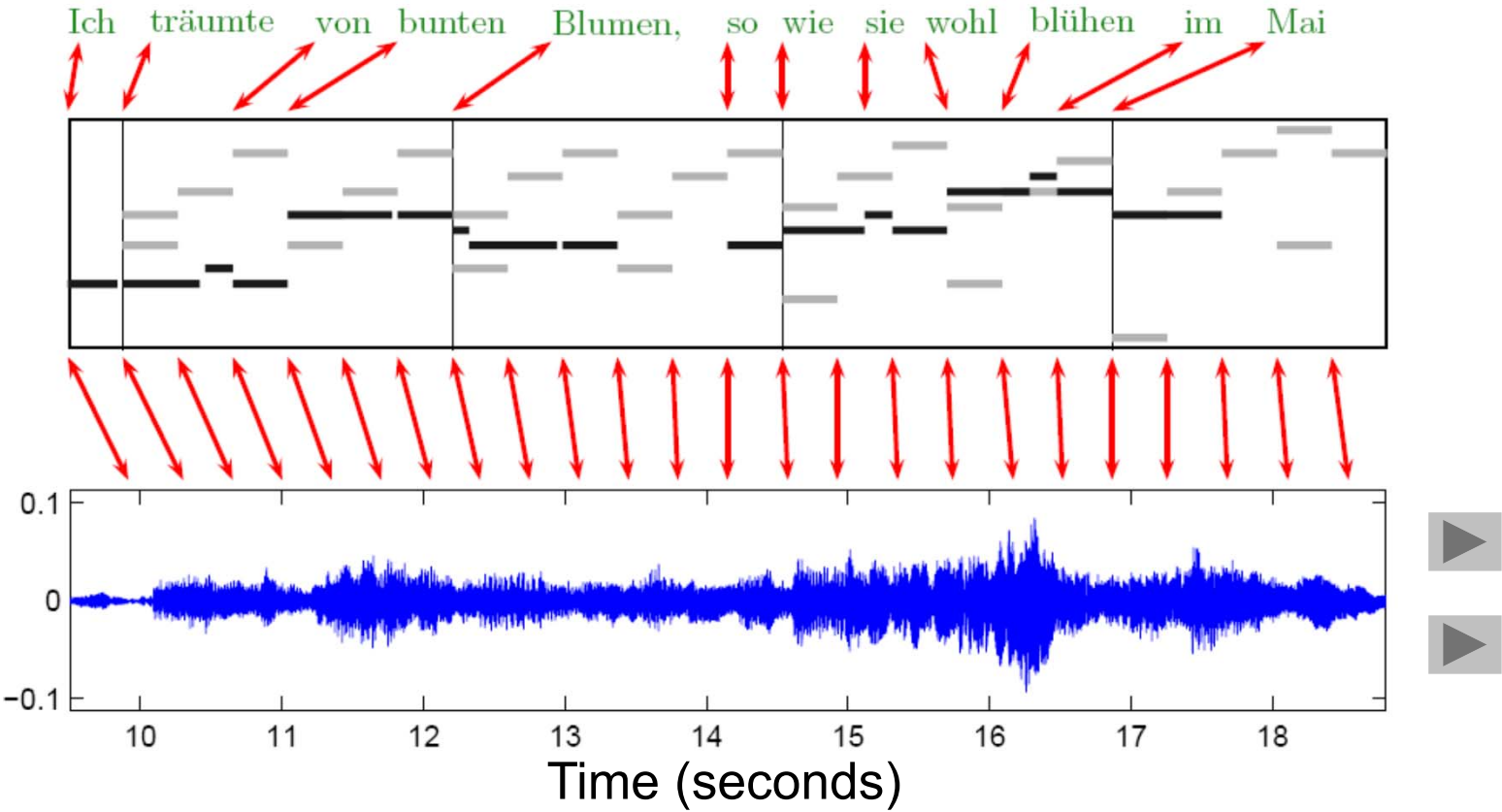
# Music Synchronization: Lyrics-Audio



Difficult task!

# Music Synchronization: Lyrics-Audio

Lyrics-Audio → Lyrics-MIDI + MIDI-Audio



# Music Synchronization: Lyrics-Audio

Application: SyncPlayer/LyricsSeeker



# Source Separation

- Decomposition of audio stream into different sound sources
- Central task in digital signal processing
- “Cocktail party effect”
- Sources are often assumed to be statistically independent
- This is often not the case in music

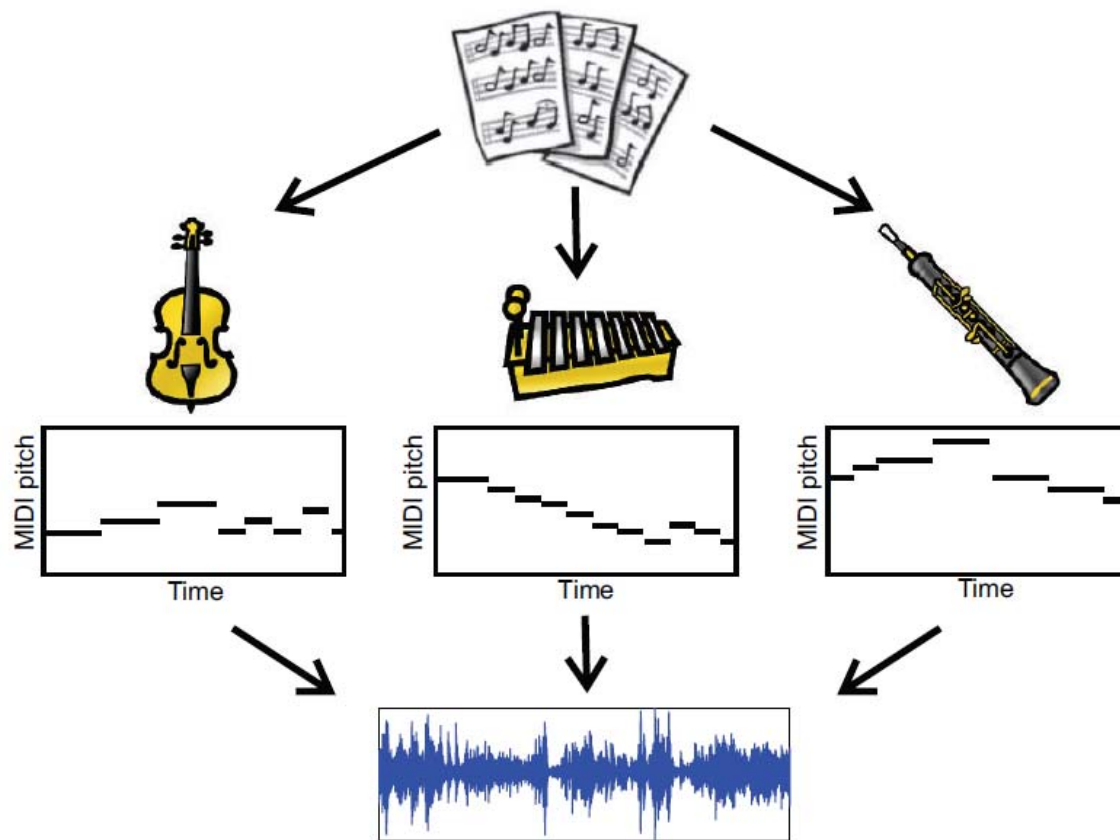
Strategy: Exploit additional information (e.g. musical score)  
to support the separation process

# Score-Informed Source Separation

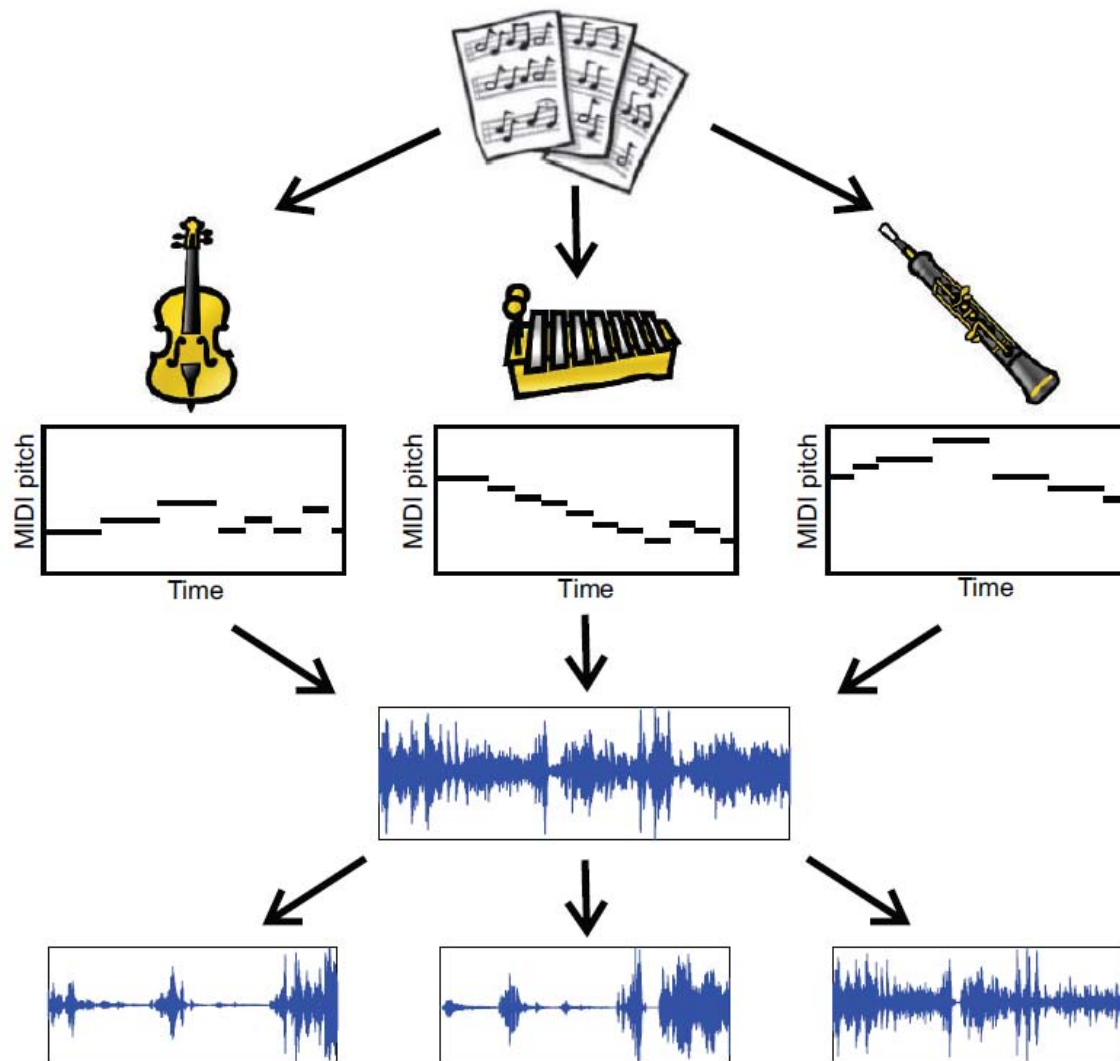




# Score-Informed Source Separation

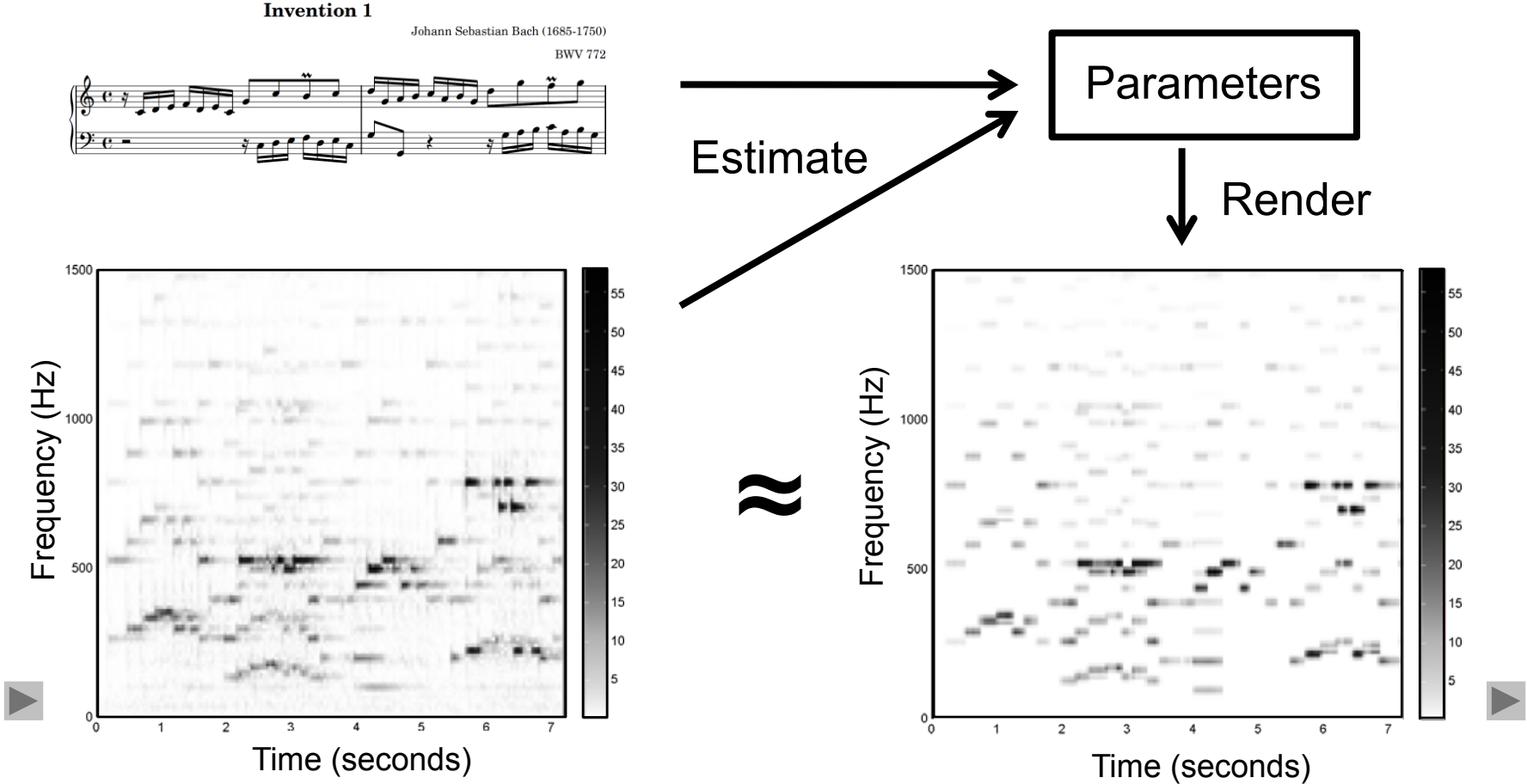


# Score-Informed Source Separation

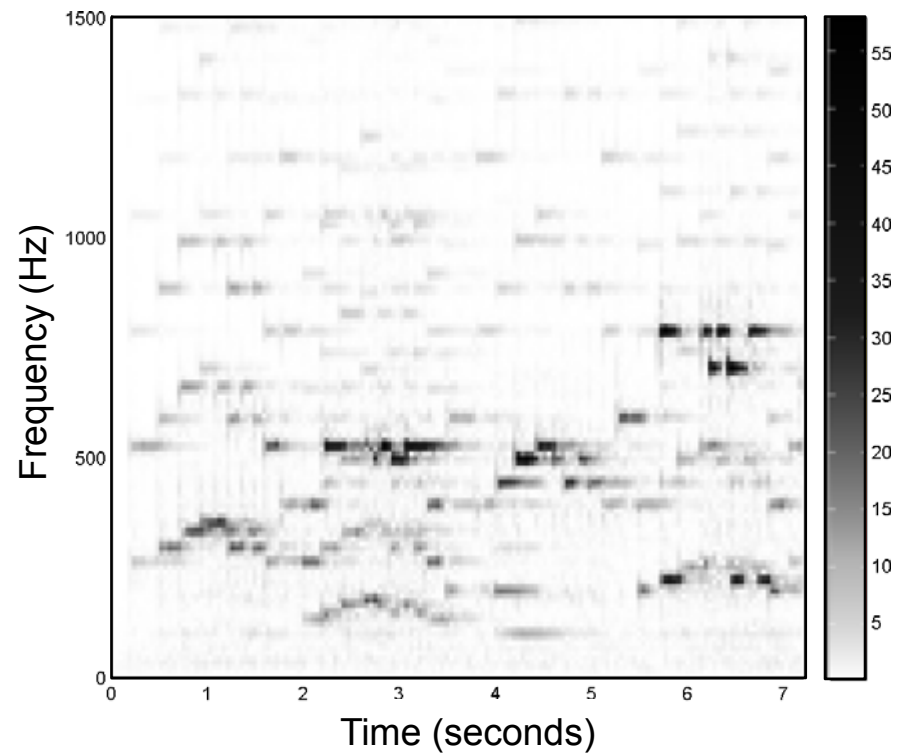


# Score-Informed Source Separation

**Goal:** Approximate spectrogram using a parametric model exploiting availability of score information

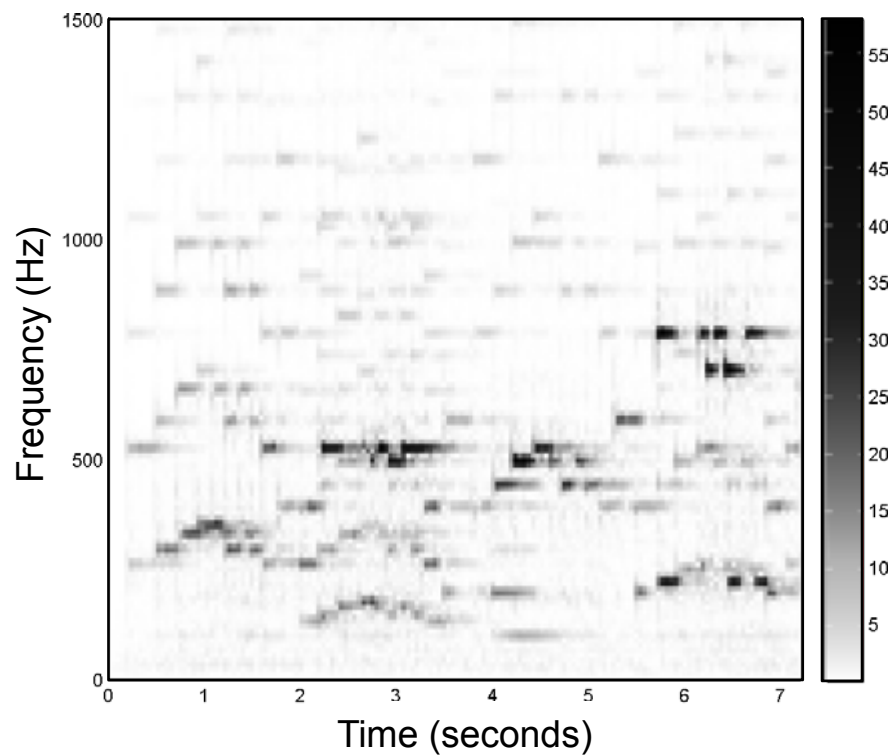


# Score-Informed Source Separation

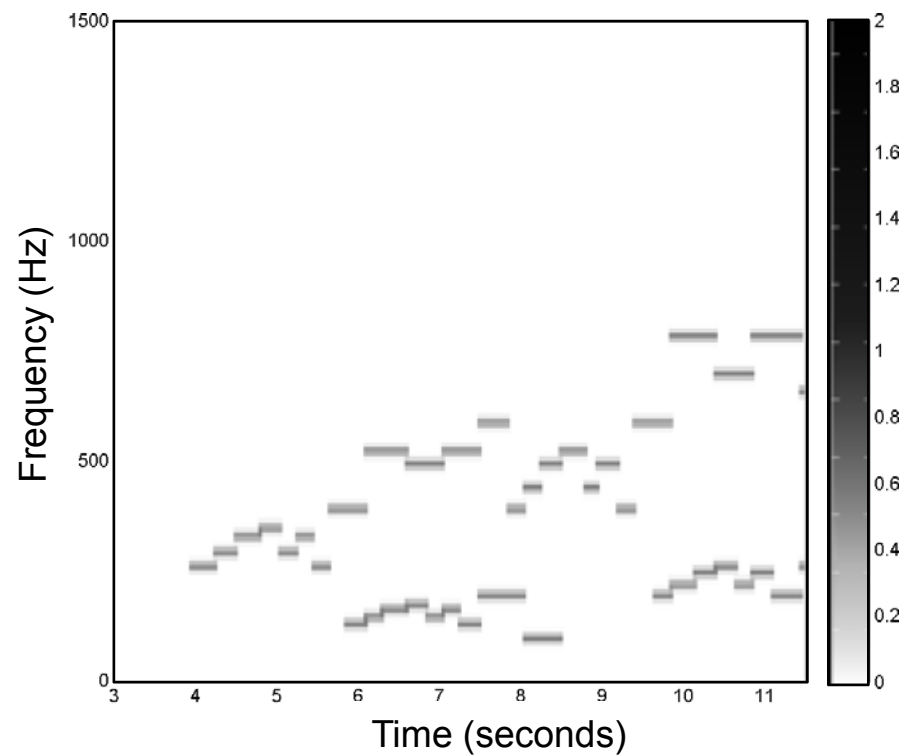


Original audio recording

# Score-Informed Source Separation

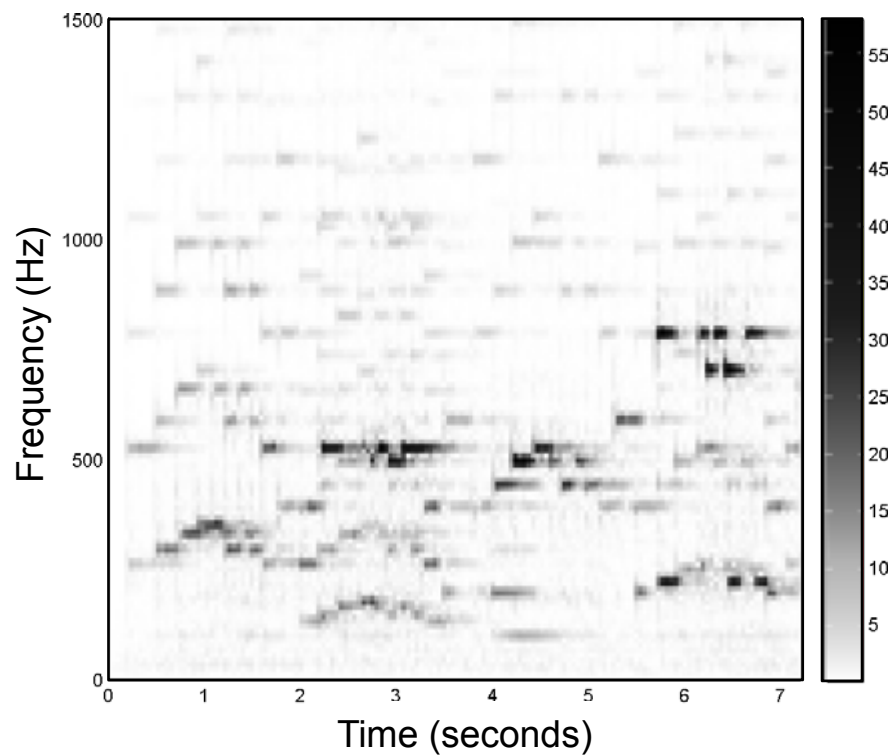


Original audio recording

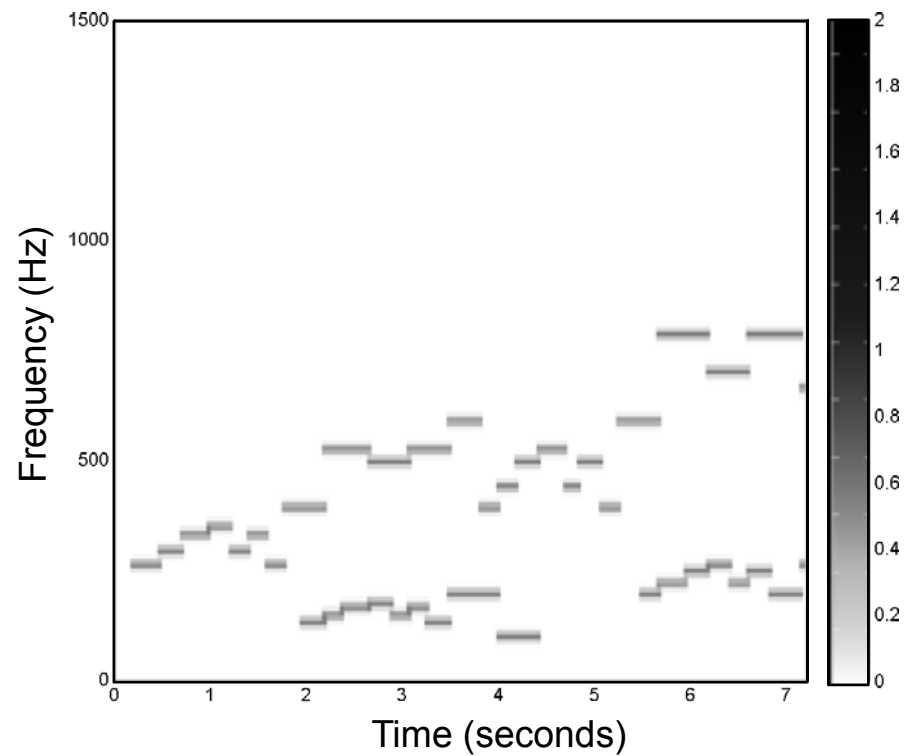


Model initialized with  
MIDI information

# Score-Informed Source Separation

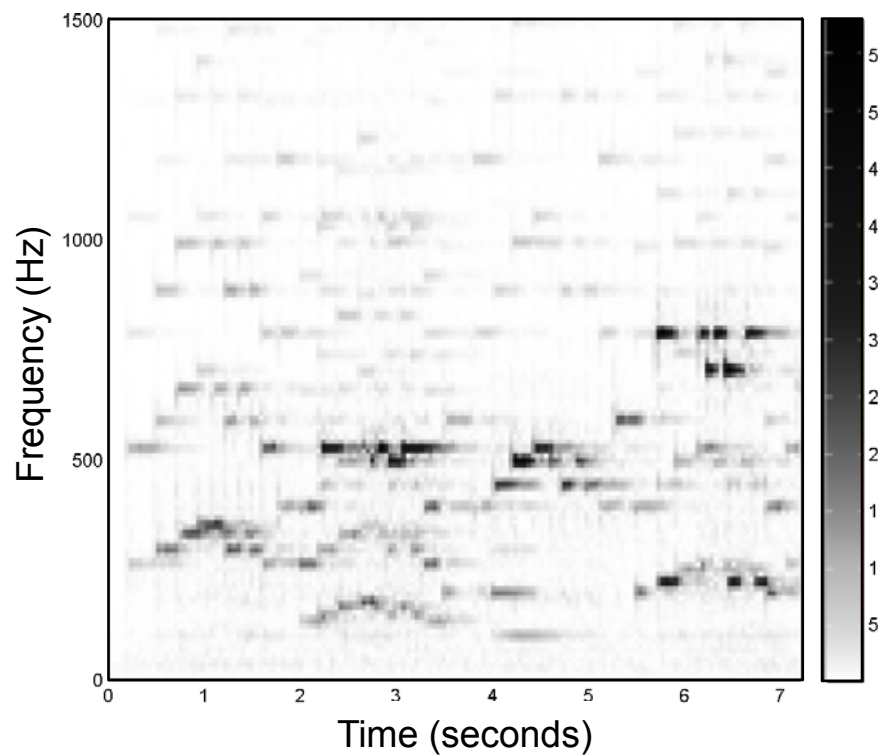


Original audio recording

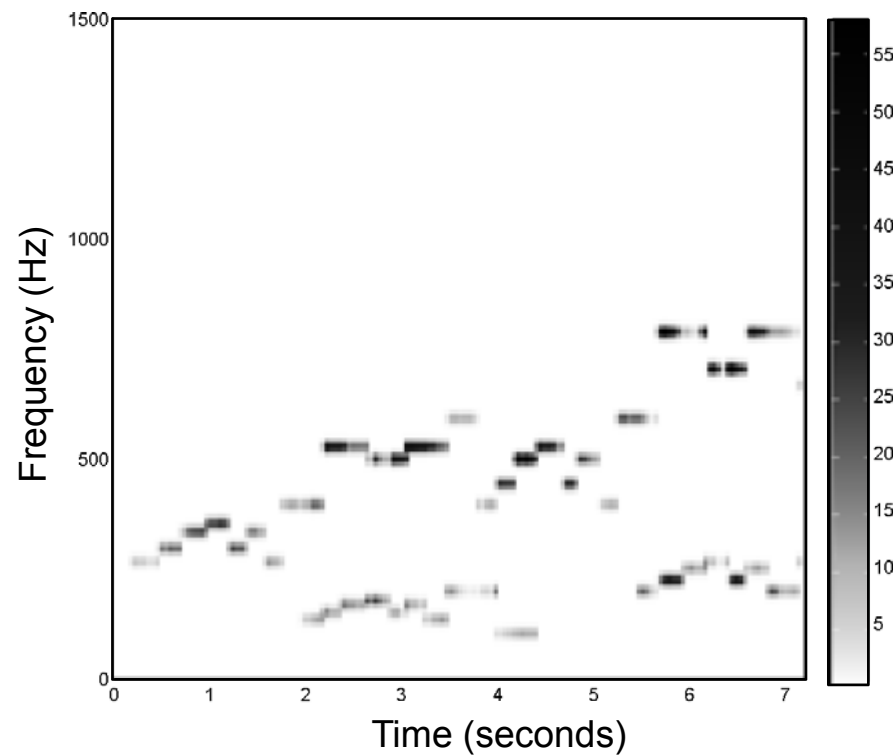


- Temporal synchronization
- Tuning estimation

# Score-Informed Source Separation

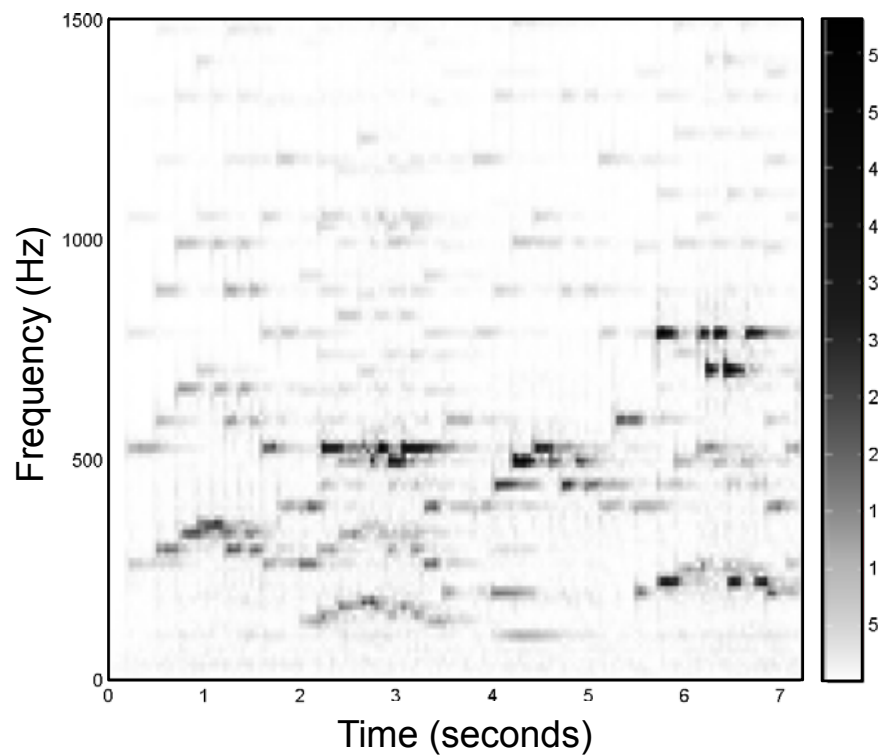


Original audio recording

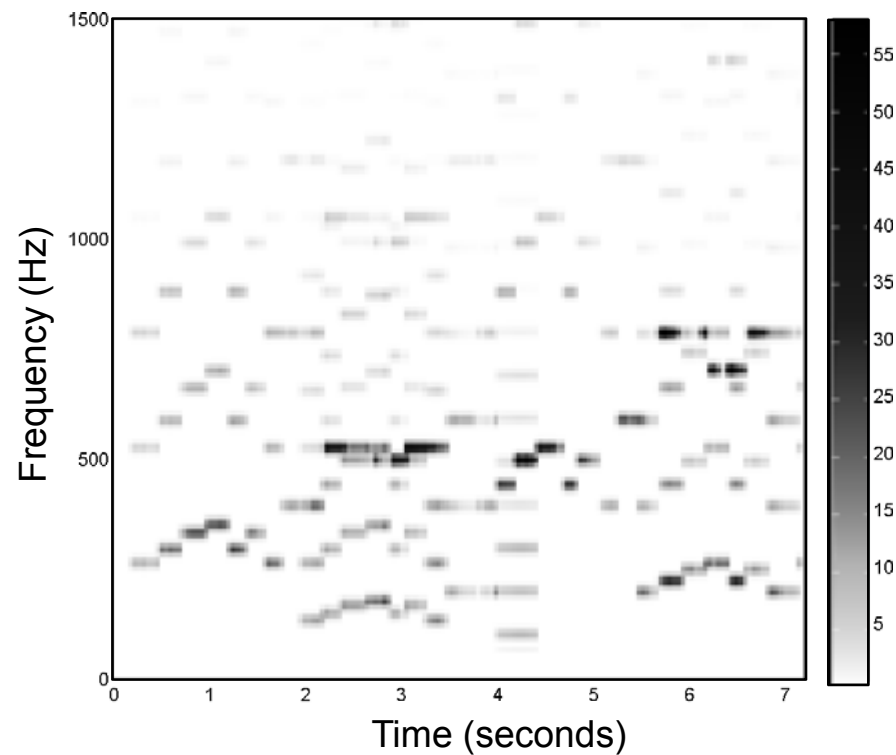


Activity parameters updated

# Score-Informed Source Separation



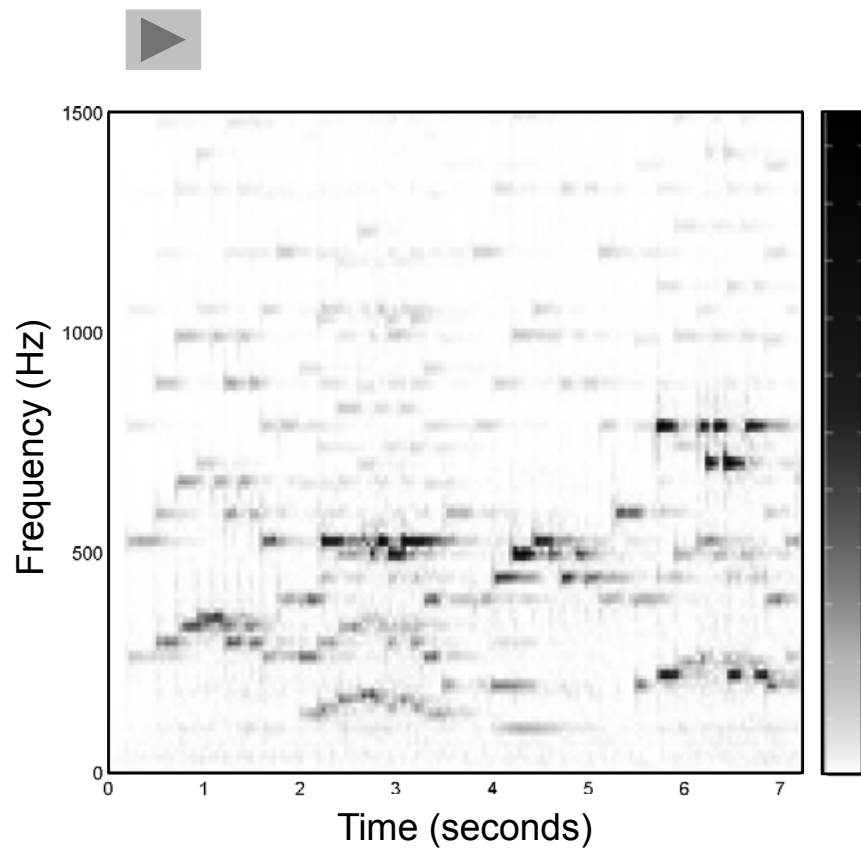
Original audio recording



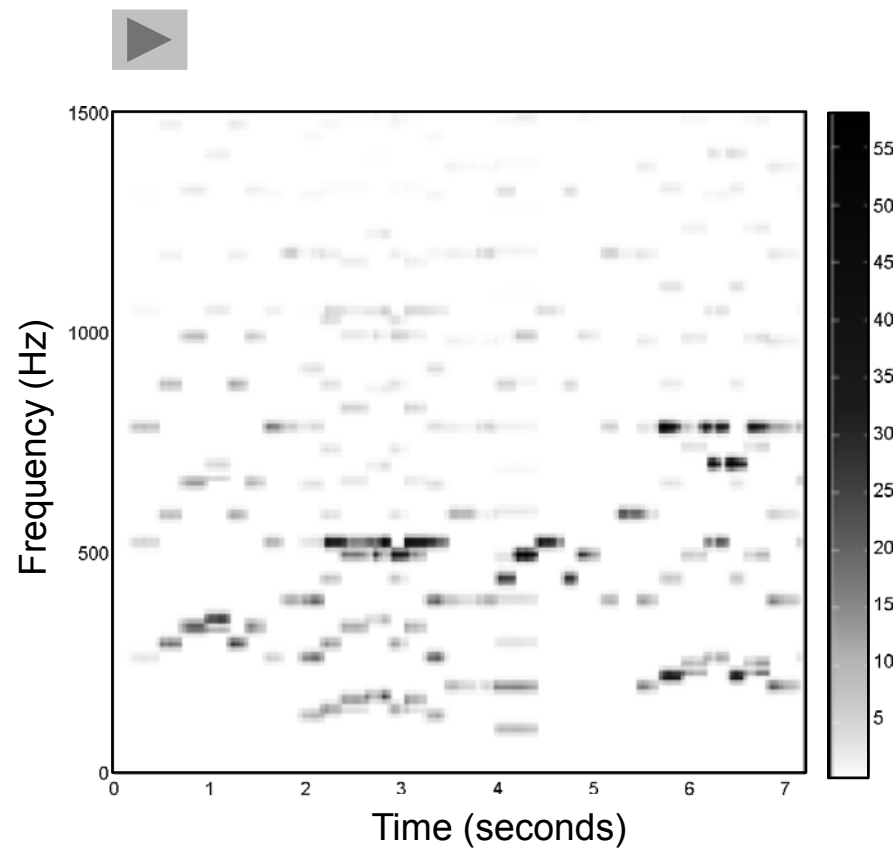
- Partial energy distribution
- Resonance body properties



# Score-Informed Source Separation



Original audio recording



Model after three iterations

**Note: Each note specified by the score parameterizes a portion of the spectrogram**

# Score-Informed Source Separation

Experimental results for separating left and right hands for piano recordings:

FREDERIC CHOPIN (1810-1849)  
OP. 64, No. 1

Molto Vivace

Composer	Piece	Database	Results			
			L	R	Eq	Org
Bach	BWV 875, Prelude	SMD				
Chopin	Op. 28, No. 15	SMD				
Chopin	Op. 64, No. 1	European Archive				

# Score-Informed Source Separation

## Audio editing

