

INTERNATIONAL AUDIO LABORATORIES ERLANGEN
A joint institution of Fraunhofer IIS and Universität Erlangen-Nürnberg



Learning-By-Doing: Using the FMP Python Notebooks for Audio and Music Processing

Meinard Müller

International Audio Laboratories Erlangen
meinard.mueller@audiolabs-erlangen.de

NUS Computer Science Research Week 2022

Singapore, January 6, 2022



Meinard Müller



- Mathematics (Diplom/Master)
Computer Science (PhD)
Information Retrieval (Habilitation)
- Since 2012: Professor
Semantic Audio Processing
- Former President of the International Society for
Music Information Retrieval (MIR)
- Member of the Senior Editorial Board of the
IEEE Signal Processing Magazine
- IEEE Fellow for contributions to Music Signal Processing



Meinard Müller: Research Group

Semantic Audio Processing

- Sebastian Rosenzweig
- Michael Krause
- Yigitcan Özer
- Peter Meier (external)
- Christof Weiß

- Frank Zalkow
- Christian Dittmar
- Stefan Balke
- Jonathan Driedger
- Thomas Prätzlich
- ...



International Audio Laboratories Erlangen



- Fraunhofer Institute for Integrated Circuits IIS
- Largest Fraunhofer institute with ≈ 1000 members
- Applied research for sensor, audio, and media technology



- Friedrich-Alexander Universität Erlangen-Nürnberg (FAU)
- One of Germany's largest universities with $\approx 40,000$ students
- Strong Technical Faculty

International Audio Laboratories Erlangen



Audio

International Audio Laboratories Erlangen

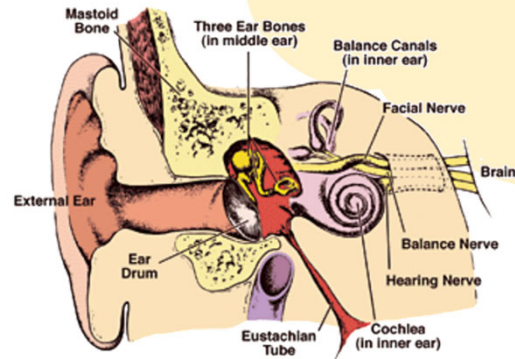
Audio Coding



3D Audio



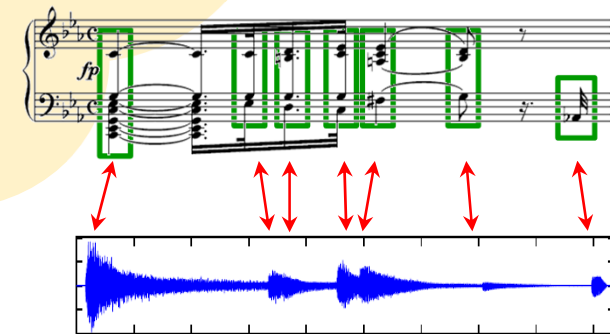
Audio



Psychoacoustics



Internet of Things



Music Processing

AudioLabs – FAU

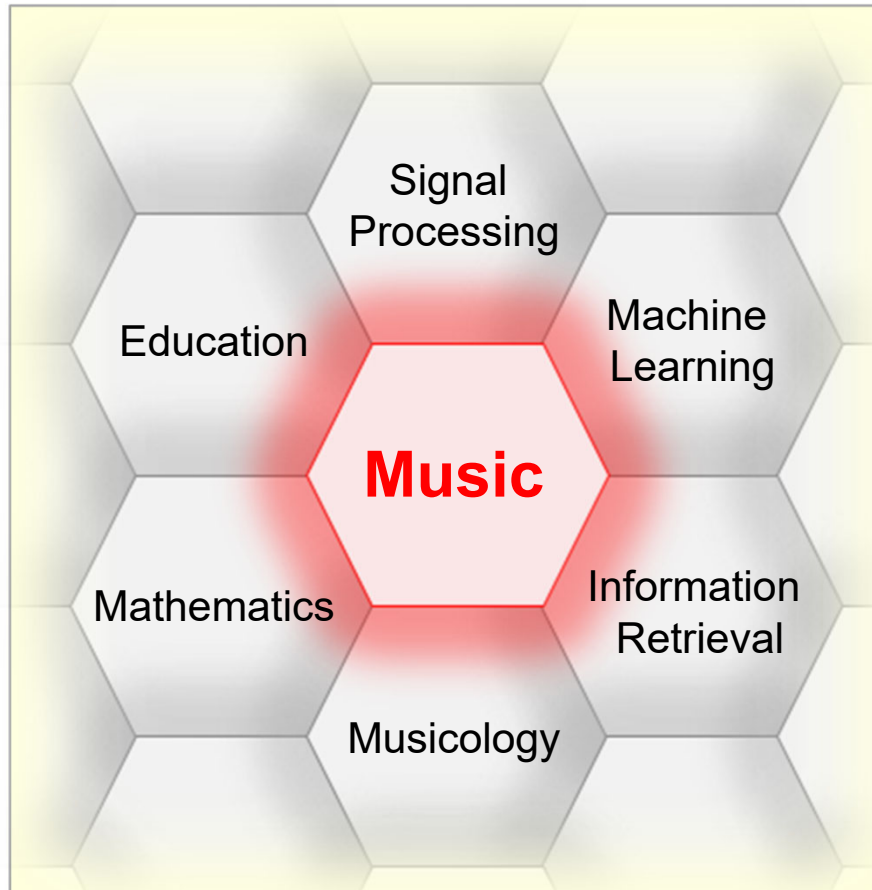
- Prof. Dr. Jürgen Herre
Audio Coding
- Prof. Dr. Bernd Edler
Audio Signal Analysis
- Prof. Dr. Meinard Müller
Semantic Audio Processing
- Prof. Dr. Emanuël Habets
Spatial Audio Signal Processing
- Prof. Dr. Nils Peters
Audio Signal Processing
- Dr. Stefan Turowski
Coordinator AudioLabs-FAU



Music Processing

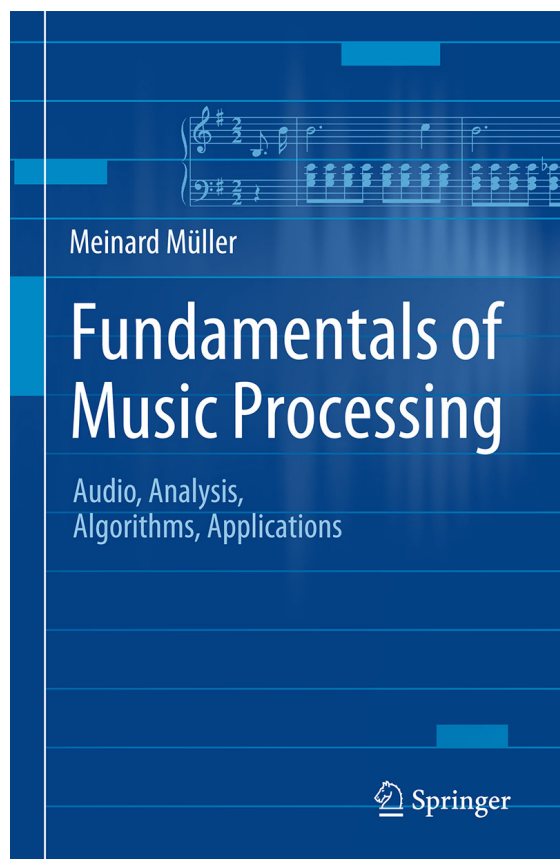


Music Processing: A Multifaceted Research Area



- Music is a ubiquitous and vital part of our lives
- Digital music services: Spotify, Pandora, iTunes, ...
- Music yields intuitive entry point to support and motivate education in technical disciplines
- Music bridges the gap between engineering, computer science, mathematics, and the humanities

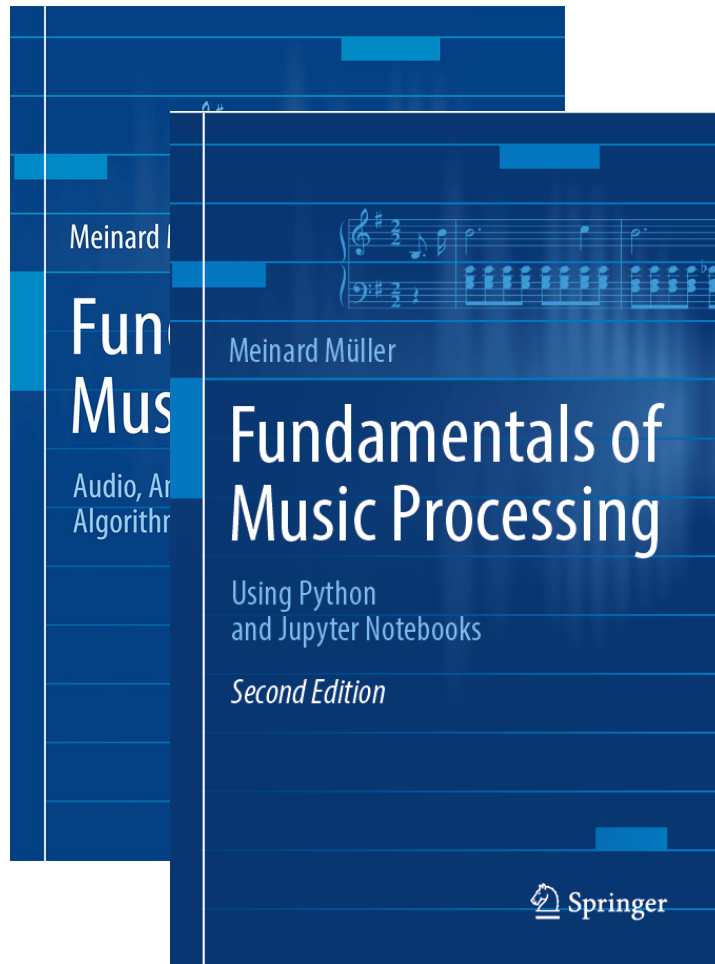
Fundamentals of Music Processing (FMP)



Meinard Müller
Fundamentals of Music Processing
Audio, Analysis, Algorithms, Applications
Springer, 2015

Accompanying website:
www.music-processing.de

Fundamentals of Music Processing (FMP)

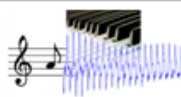

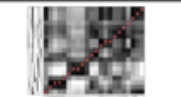
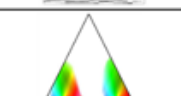

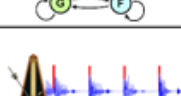




Meinard Müller
Fundamentals of Music Processing
Audio, Analysis, Algorithms, Applications
Springer, 2015

Accompanying website:
www.music-processing.de

2nd edition
Meinard Müller
Fundamentals of Music Processing
Using Python and Jupyter Notebooks
Springer, 2021

Fundamentals of Music Processing (FMP)

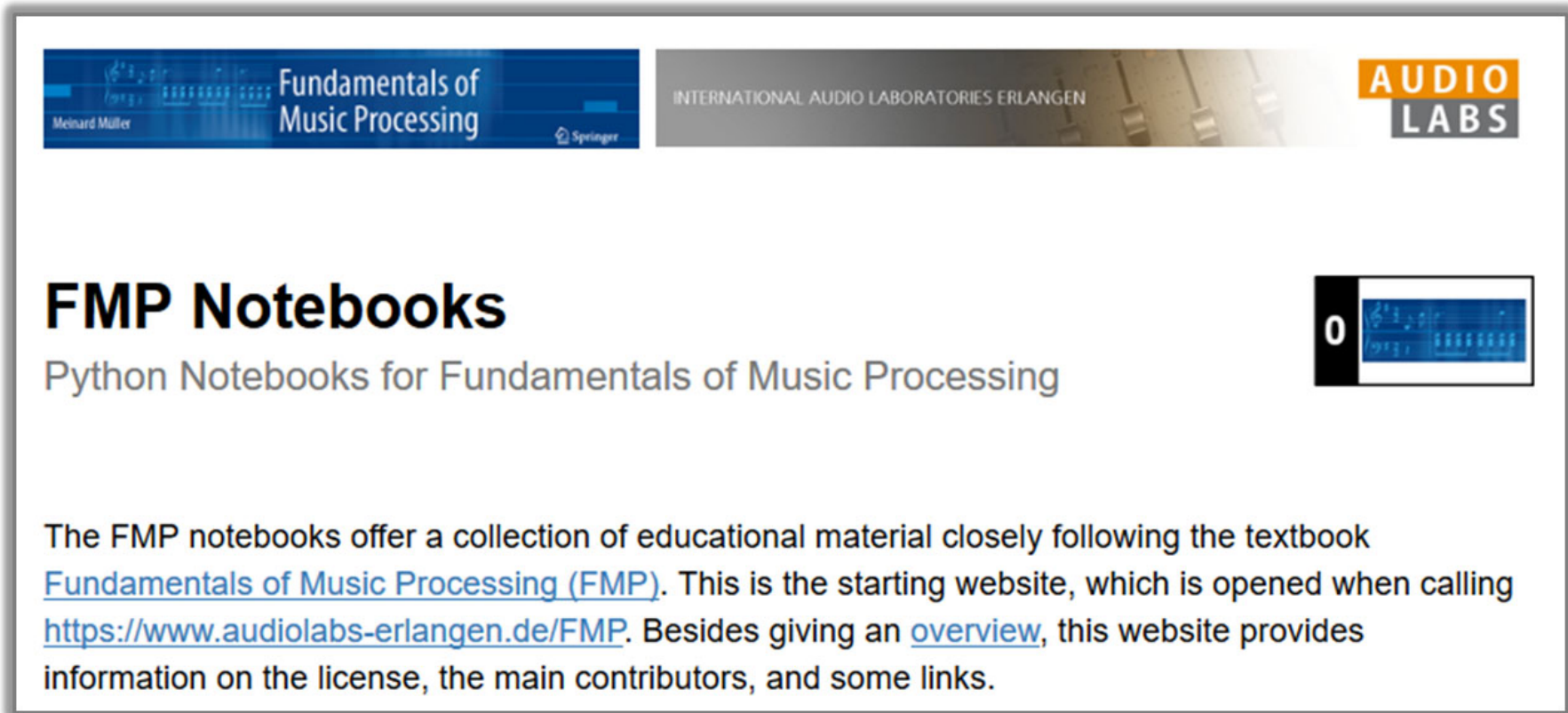
Chapter	Music Processing Scenario
1	 Music Representations
2	 Fourier Analysis of Signals
3	 Music Synchronization
4	 Music Structure Analysis
5	 Chord Recognition
6	 Tempo and Beat Tracking
7	 Content-Based Audio Retrieval
8	 Musically Informed Audio Decomposition

Meinard Müller
Fundamentals of Music Processing
Audio, Analysis, Algorithms, Applications
Springer, 2015

Accompanying website:
www.music-processing.de

2nd edition
Meinard Müller
Fundamentals of Music Processing
Using Python and Jupyter Notebooks
Springer, 2021

FMP Notebooks: Education & Research



The screenshot shows the header of the FMP Notebooks website. On the left, there is a blue banner for the book "Fundamentals of Music Processing" by Meinard Müller, published by Springer. In the center, it says "INTERNATIONAL AUDIO LABORATORIES ERLANGEN". On the right, there is the "AUDIO LABS" logo. Below the header, the main heading is "FMP Notebooks" in a large, bold, black font. Underneath it, the subtitle "Python Notebooks for Fundamentals of Music Processing" is displayed in a smaller, grey font. To the right of the subtitle is a small icon of a notebook with a blue cover and a white page, with a black circle containing the number "0" next to it. Below the subtitle, there is a paragraph of text: "The FMP notebooks offer a collection of educational material closely following the textbook [Fundamentals of Music Processing \(FMP\)](#). This is the starting website, which is opened when calling <https://www.audiolabs-erlangen.de/FMP>. Besides giving an [overview](#), this website provides information on the license, the main contributors, and some links."



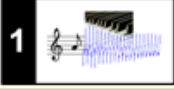

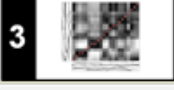
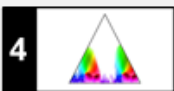

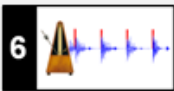


<https://www.audiolabs-erlangen.de/FMP>

FMP Notebooks: Education & Research

- ... provide educational material for teaching and learning fundamentals of music processing.
- ... combine textbook-like explanations, technical concepts, mathematical details, Python code examples, illustrations, and sound examples.
- ... bridge the gap between theory and practice being based on interactive Jupyter notebook framework.
- ... are freely accessible under a Creative Commons license.



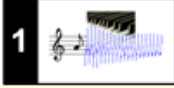
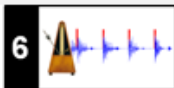


<https://www.audiolabs-erlangen.de/FMP>

FMP Notebooks

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

FMP Notebooks



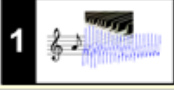

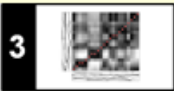
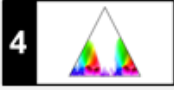

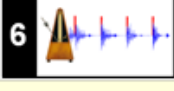

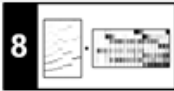
Structured in 10 parts

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

FMP Notebooks

Structured in 10 parts





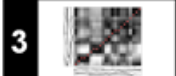
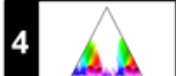

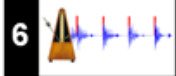

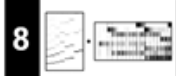
- Part B: Basic introductions to
 - Jupyter notebook framework
 - Python programming
 - Other technical concepts underlying these notebooks

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

FMP Notebooks

Structured in 10 parts

- Part B: Basic introductions to
 - Jupyter notebook framework
 - Python programming
 - Other technical concepts underlying these notebooks
- Part 0: Starting notebook

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



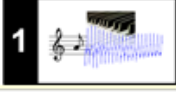

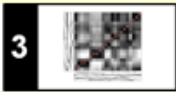
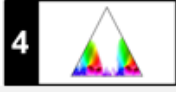



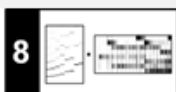
FMP Notebooks

Structured in 10 parts

- Part B: Basic introductions to
 - Jupyter notebook framework
 - Python programming
 - Other technical concepts underlying these notebooks

- Part 0: Starting notebook

- Part 1 to Part 8: Different music processing scenarios

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

FMP Notebooks

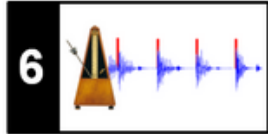
Structured in 10 parts

- Part B: Basic introductions to
 - Jupyter notebook framework
 - Python programming
 - Other technical concepts underlying these notebooks
- Part 0: Starting notebook
- Part 1 to Part 8: Different music processing scenarios

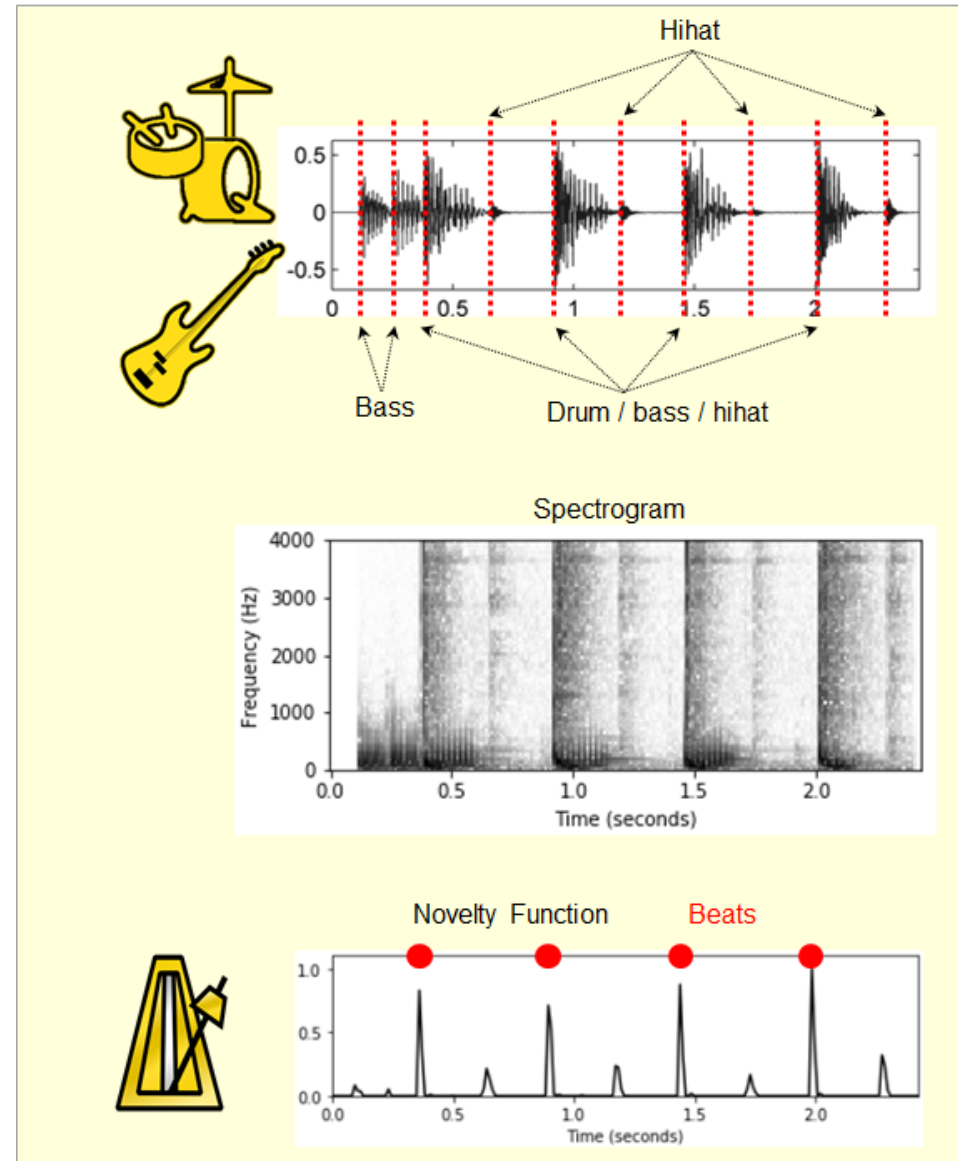
Part	Title	Notions, Techniques & Algorithms	HTML	IPYNB
	Basics	Basic information on Python, Jupyter notebooks, Anaconda package management system, Python environments, visualizations, and other topics	[html]	[ipynb]
	Overview	Overview of the notebooks (https://www.audiolabs-erlangen.de/FMP)	[html]	[ipynb]
	Music Representations	Music notation, MIDI, audio signal, waveform, pitch, loudness, timbre	[html]	[ipynb]
	Fourier Analysis of Signals	Discrete/analog signal, sinusoid, exponential, Fourier transform, Fourier representation, DFT, FFT, STFT	[html]	[ipynb]
	Music Synchronization	Chroma feature, dynamic programming, dynamic time warping (DTW), alignment, user interface	[html]	[ipynb]
		Similarity matrix, repetition,	[html]	[ipynb]
			[html]	[ipynb]
	Tempo and Beat Tracking	Onset, novelty, tempo, tempogram, beat, periodicity, Fourier analysis, autocorrelation	[html]	[ipynb]
	Content-Based Audio Retrieval	Identification, fingerprint, indexing, inverted list, matching, version, cover song	[html]	[ipynb]
	Musically Informed Audio Decomposition	Harmonic/percussive separation, signal reconstruction, instantaneous frequency, fundamental frequency (F0), trajectory, nonnegative matrix factorization (NMF)	[html]	[ipynb]

Part 6: Tempo and Beat Tracking

Part 6: Tempo and Beat Tracking



- When listening to a piece of music, we as humans are often able to tap along with the musical beat
- Automated beat tracking: Simulate this cognitive process by a computer



Tempo and Beat Tracking

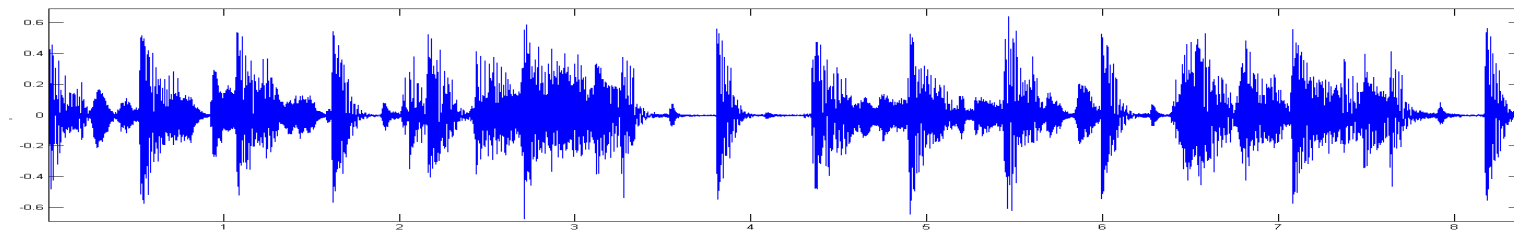
Basic task: “Tapping the foot when listening to music”



Tempo and Beat Tracking

Basic task: “Tapping the foot when listening to music”

Example: Queen – Another One Bites The Dust

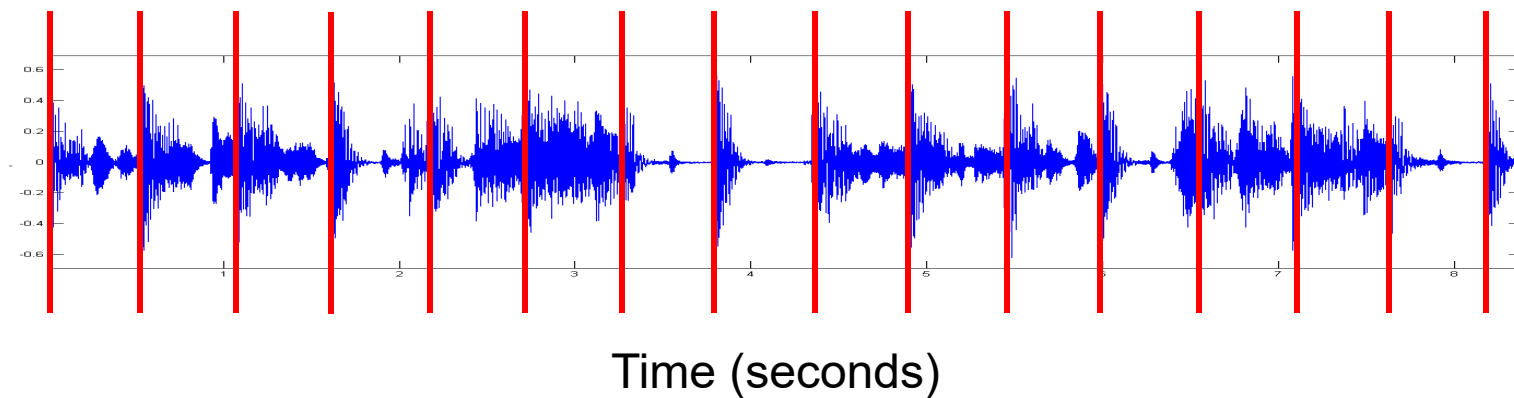


Time (seconds)

Tempo and Beat Tracking

Basic task: “Tapping the foot when listening to music”

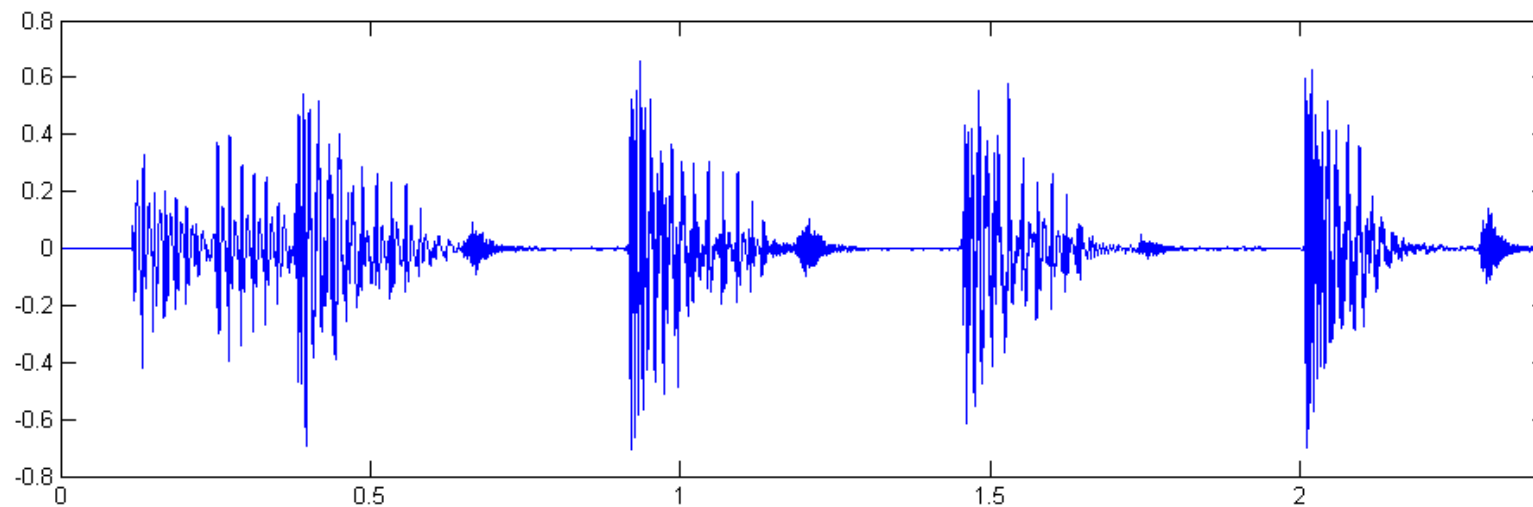
Example: Queen – Another One Bites The Dust



Tempo and Beat Tracking

Tasks

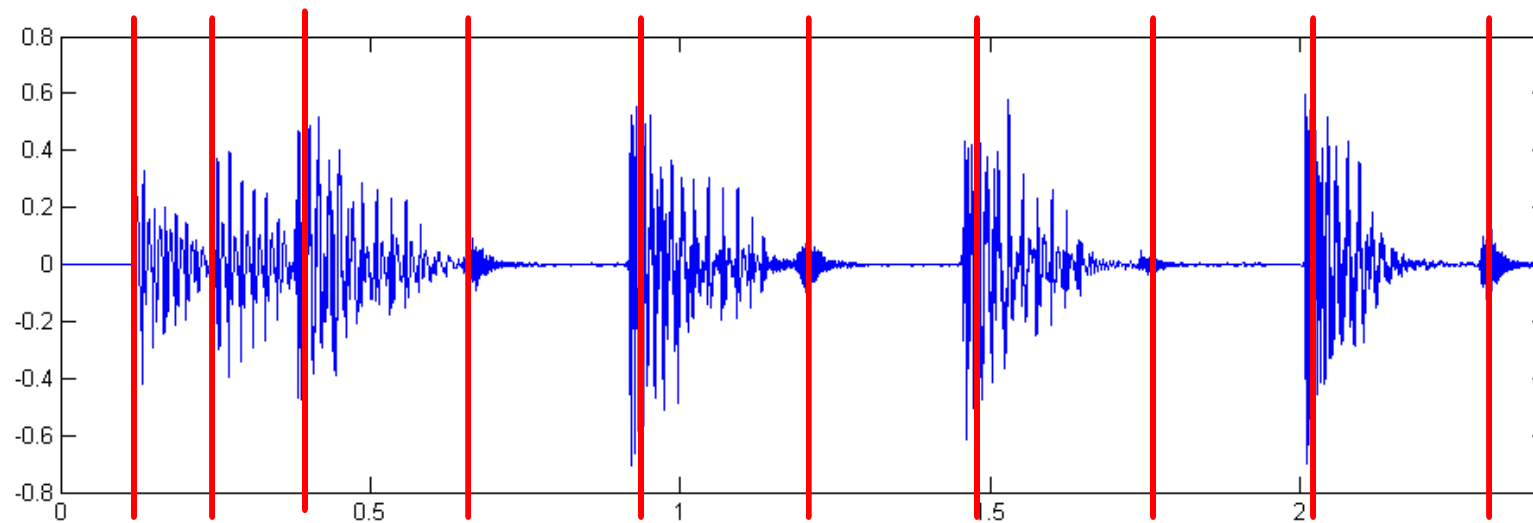
- Onset detection
- Beat tracking
- Tempo estimation



Tempo and Beat Tracking

Tasks

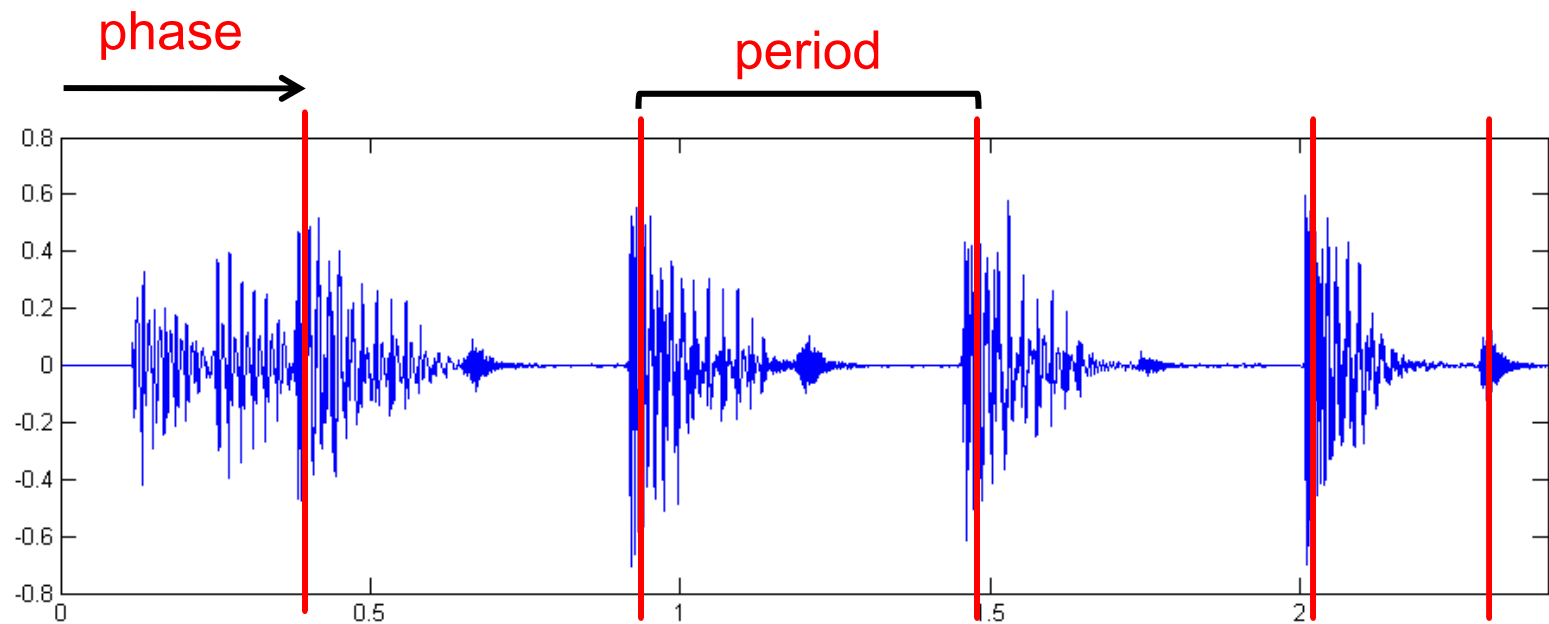
- Onset detection
- Beat tracking
- Tempo estimation



Tempo and Beat Tracking

Tasks

- Onset detection
- **Beat tracking**
- Tempo estimation



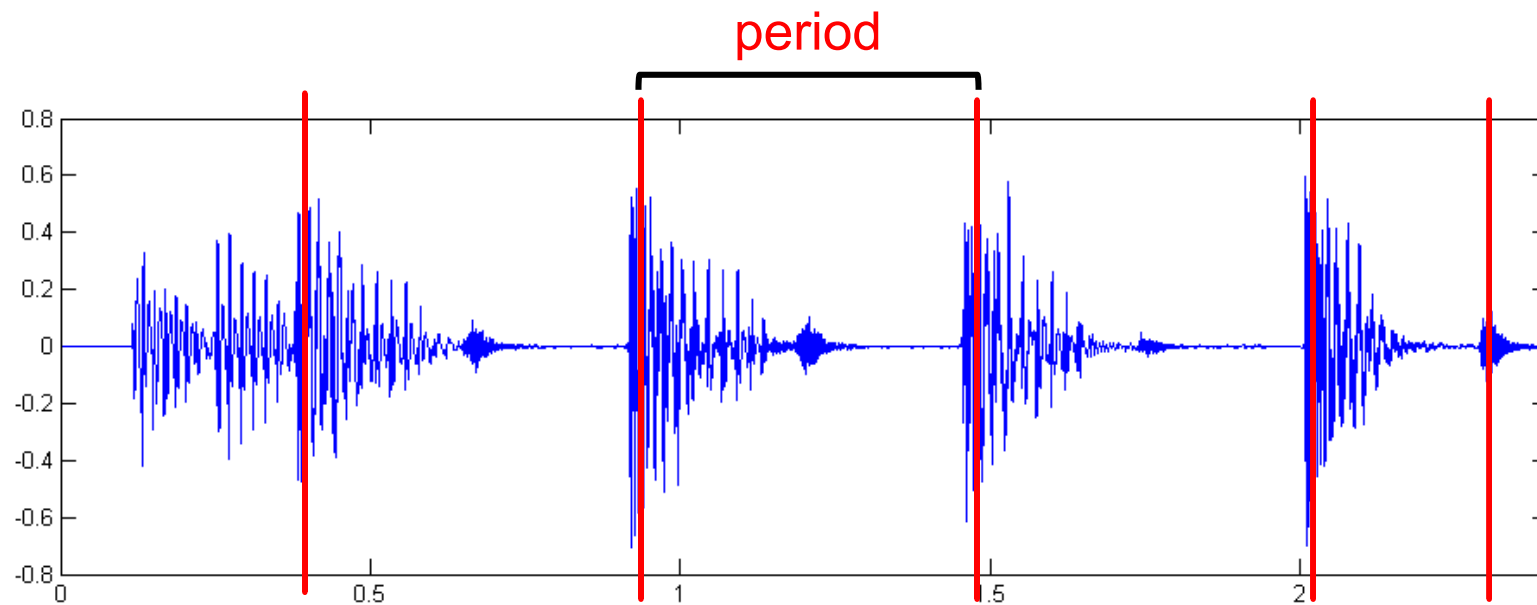
Tempo and Beat Tracking

Tasks

- Onset detection
- Beat tracking
- **Tempo estimation**

Tempo := 60 / period

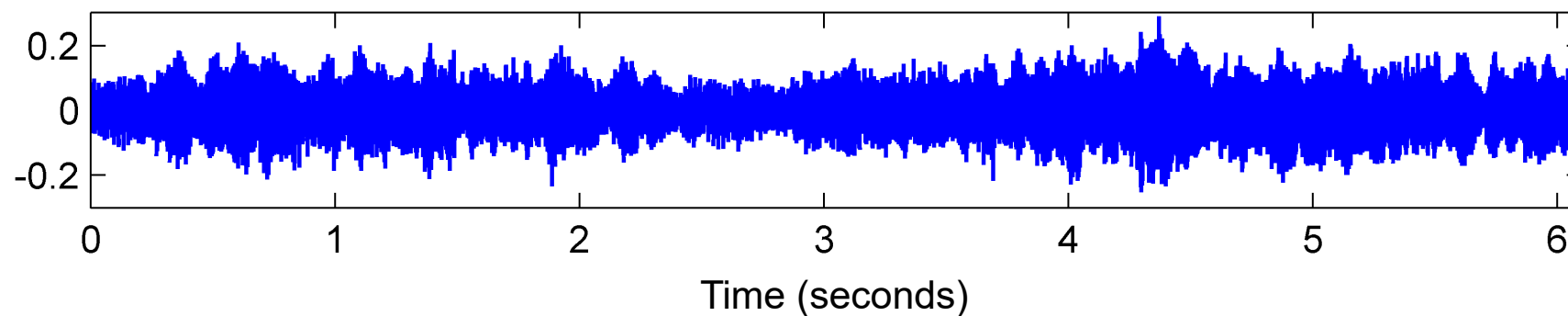
Beats per minute (BPM)



Onset Detection (Spectral Flux)

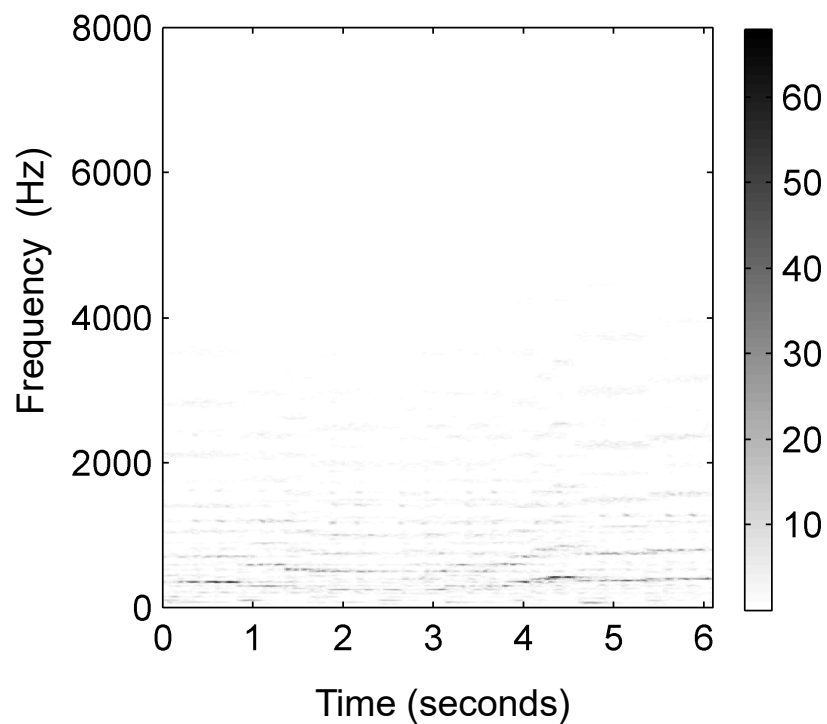


Audio recording



Onset Detection (Spectral Flux)

Magnitude spectrogram $|X|$

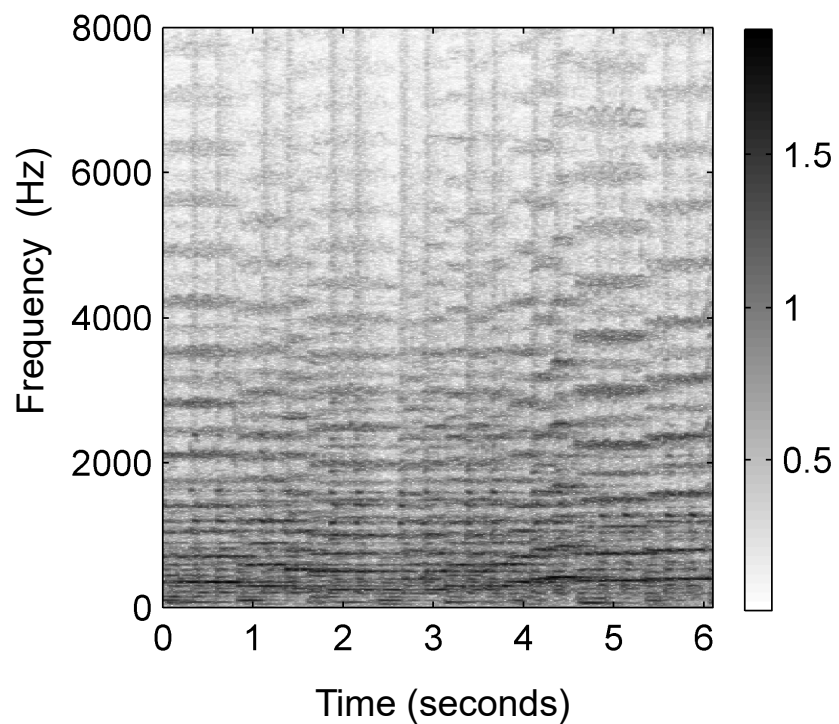


Steps:

1. Spectrogram

Onset Detection (Spectral Flux)

Compressed spectrogram Y

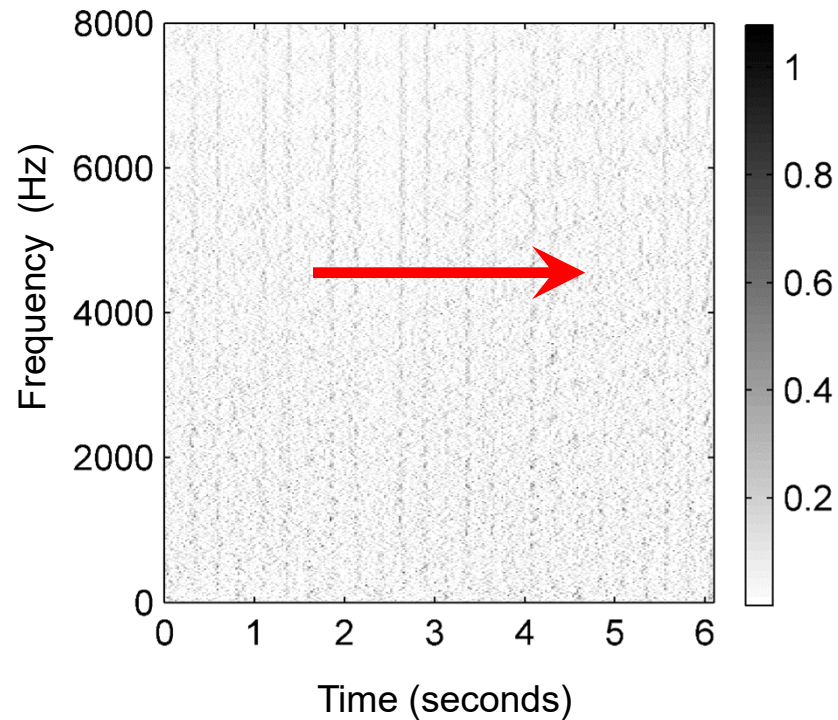


Steps:

1. Spectrogram
2. Logarithmic compression

Onset Detection (Spectral Flux)

Spectral difference

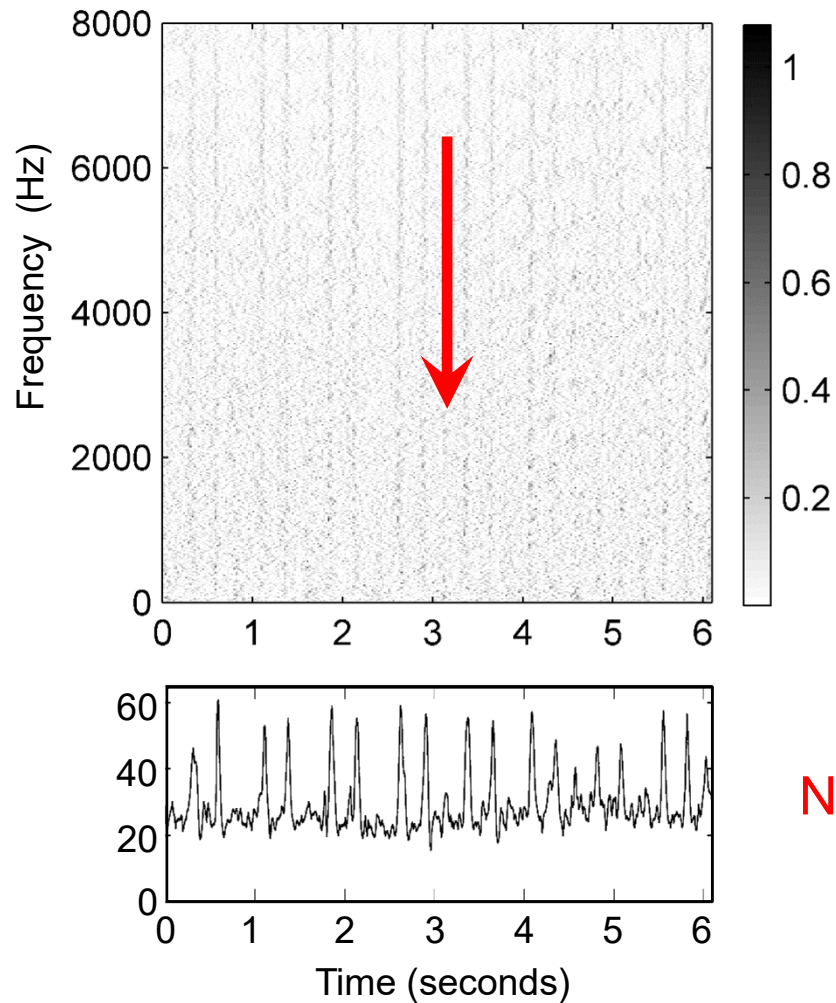


Steps:

1. Spectrogram
2. Logarithmic compression
3. Differentiation & half wave rectification

Onset Detection (Spectral Flux)

Spectral difference



Steps:

1. Spectrogram
2. Logarithmic compression
3. Differentiation & half wave rectification
4. Accumulation

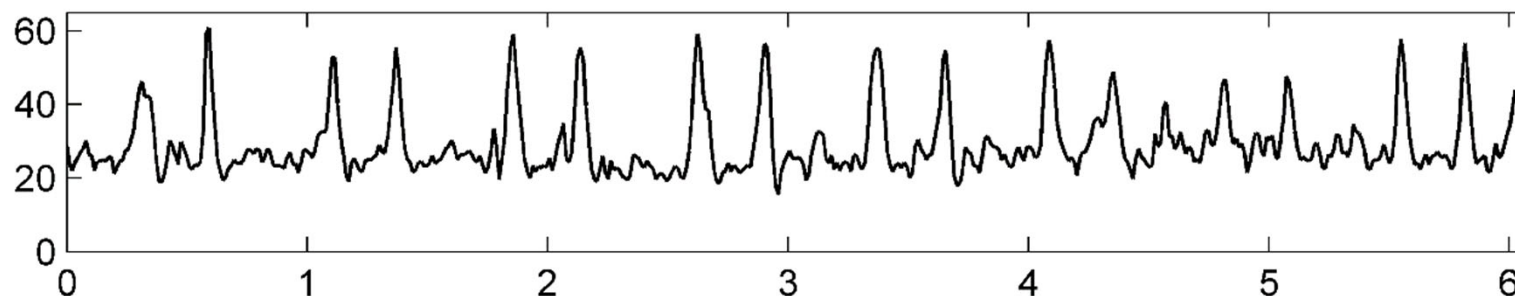
Novelty curve

Onset Detection (Spectral Flux)

Steps:

1. Spectrogram
2. Logarithmic compression
3. Differentiation & half wave rectification
4. Accumulation

Novelty function



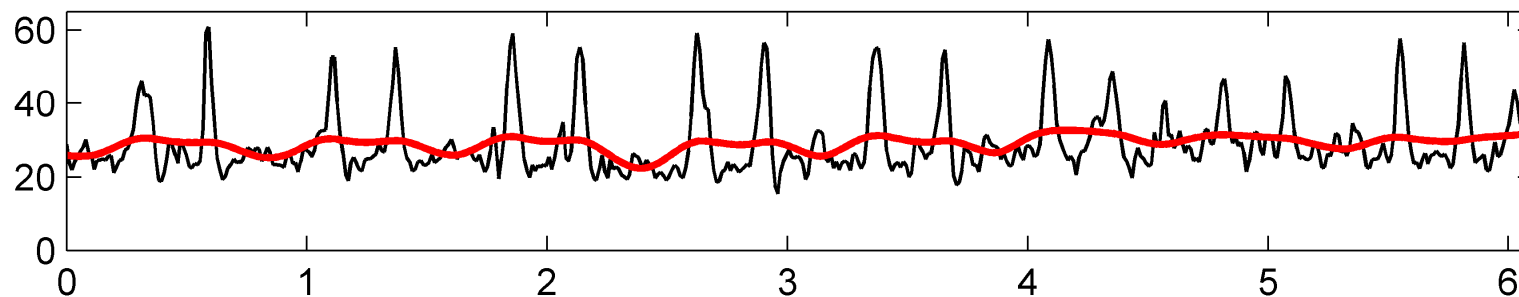
Onset Detection (Spectral Flux)

Steps:

1. Spectrogram
2. Logarithmic compression
3. Differentiation & half wave rectification
4. Accumulation
5. Normalization

Novelty function

Substraction of local average

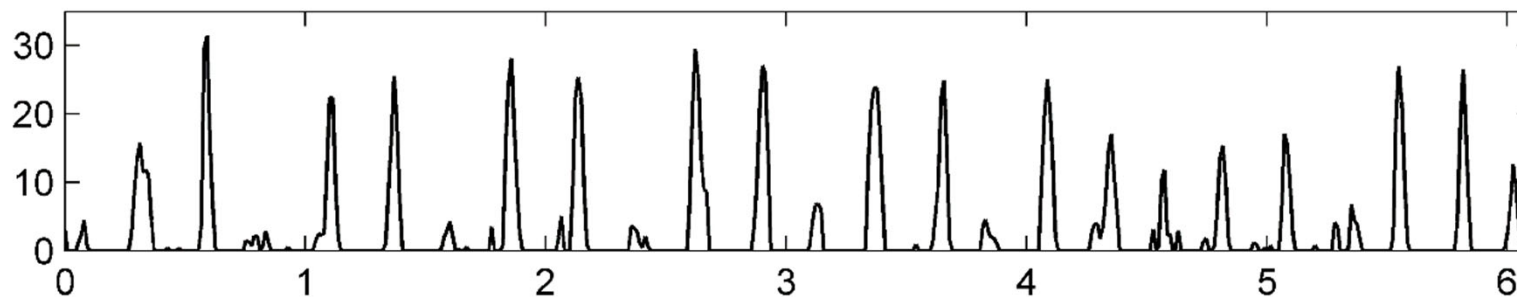


Onset Detection (Spectral Flux)

Steps:

1. Spectrogram
2. Logarithmic compression
3. Differentiation & half wave rectification
4. Accumulation
5. Normalization

Normalized novelty function



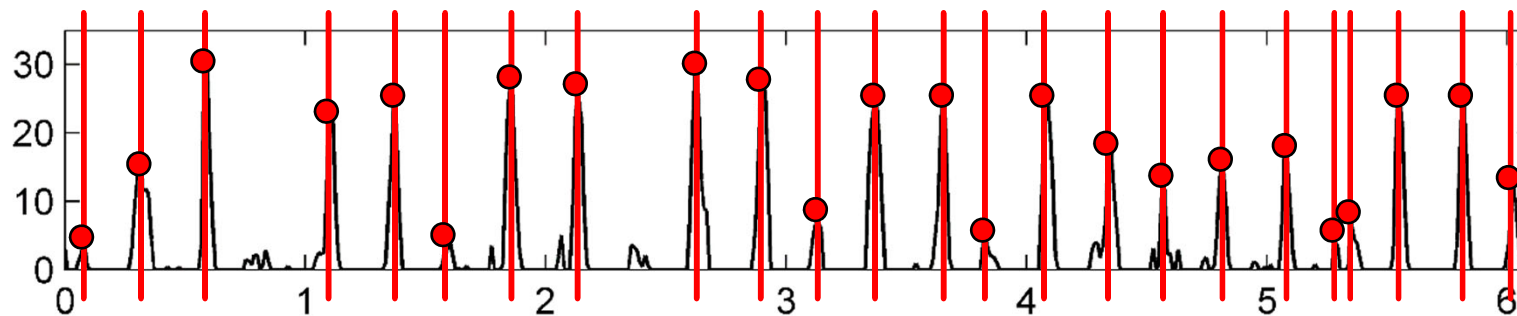
Onset Detection (Spectral Flux)

Steps:

1. Spectrogram
2. Logarithmic compression
3. Differentiation & half wave rectification
4. Accumulation
5. Normalization

Normalized novelty function

Peak positions indicate beat candidates



Onset Detection (Spectral Flux)

Deep Learning

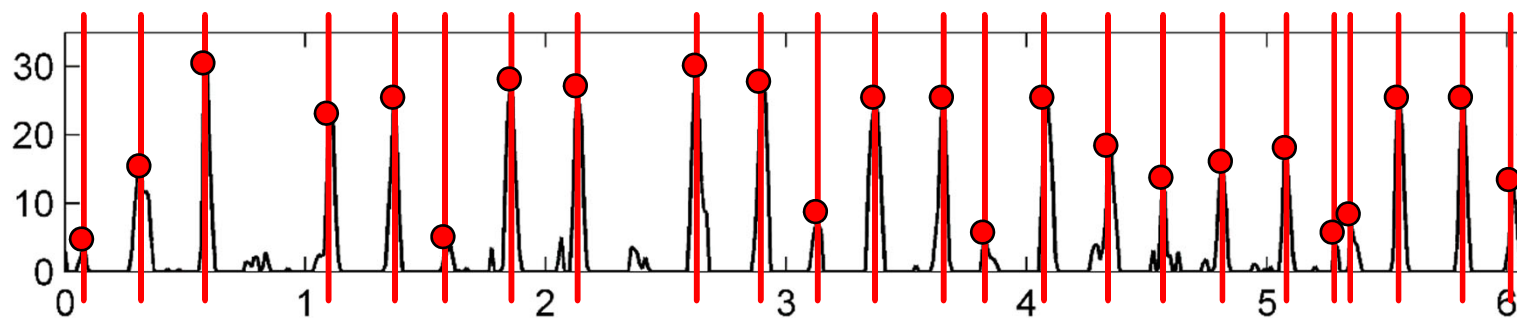
1. Input representation
2. Sigmoid activation
3. Convolution & rectified linear unit (ReLU)
4. Pooling
5. Convolution & ReLU

Steps:

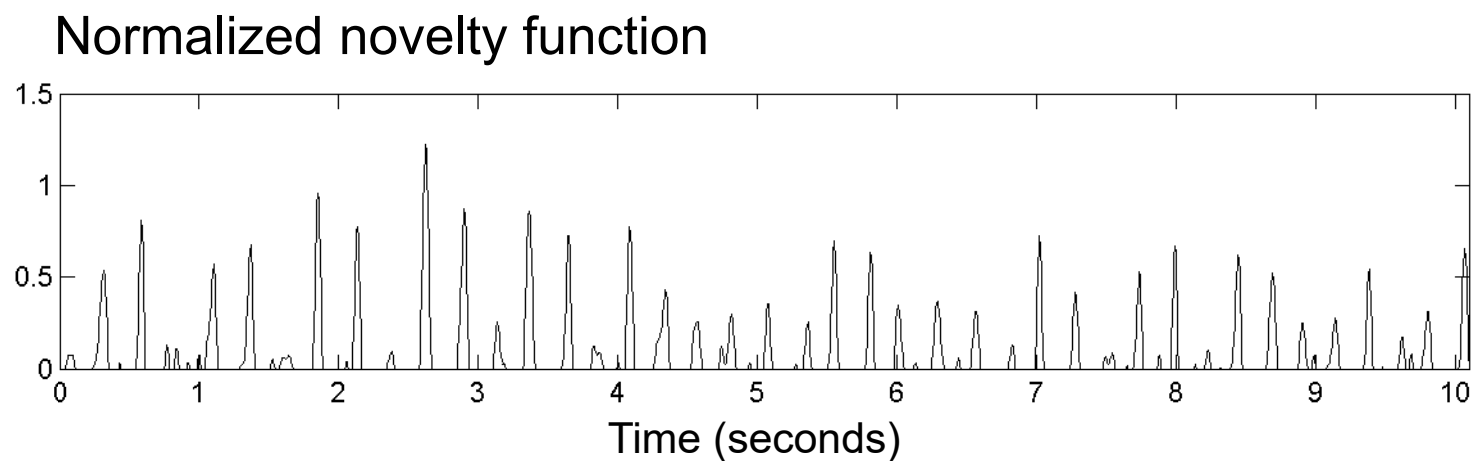
1. Spectrogram
2. Logarithmic compression
3. Differentiation & half wave rectification
4. Accumulation
5. Normalization

Normalized novelty function

Peak positions indicate beat candidates

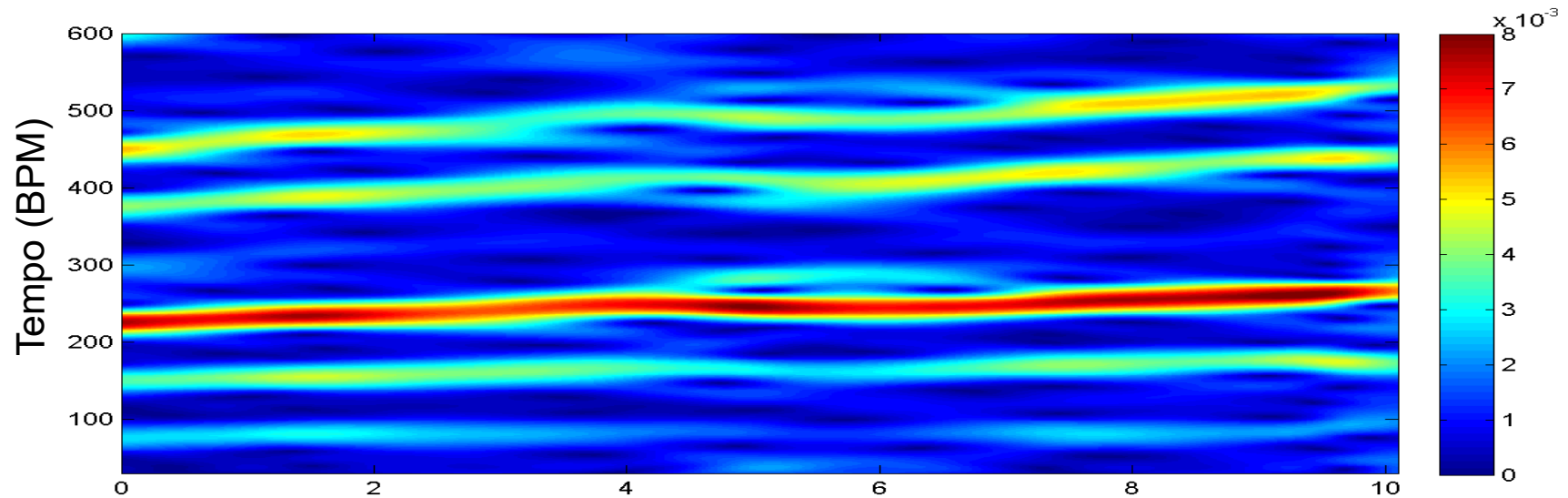


Local Pulse and Tempo Tracking

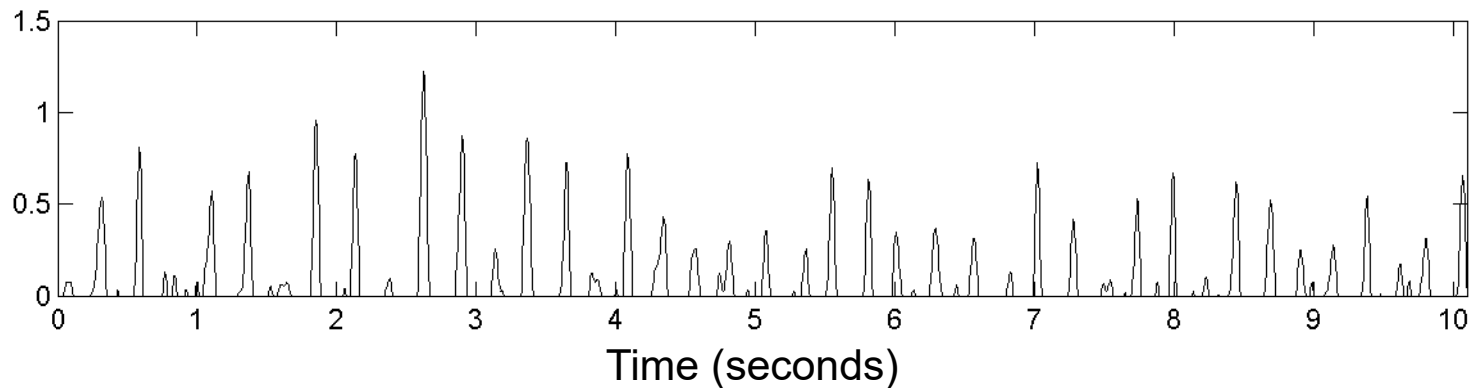


Local Pulse and Tempo Tracking

Fourier temogram (STFT of novelty function)

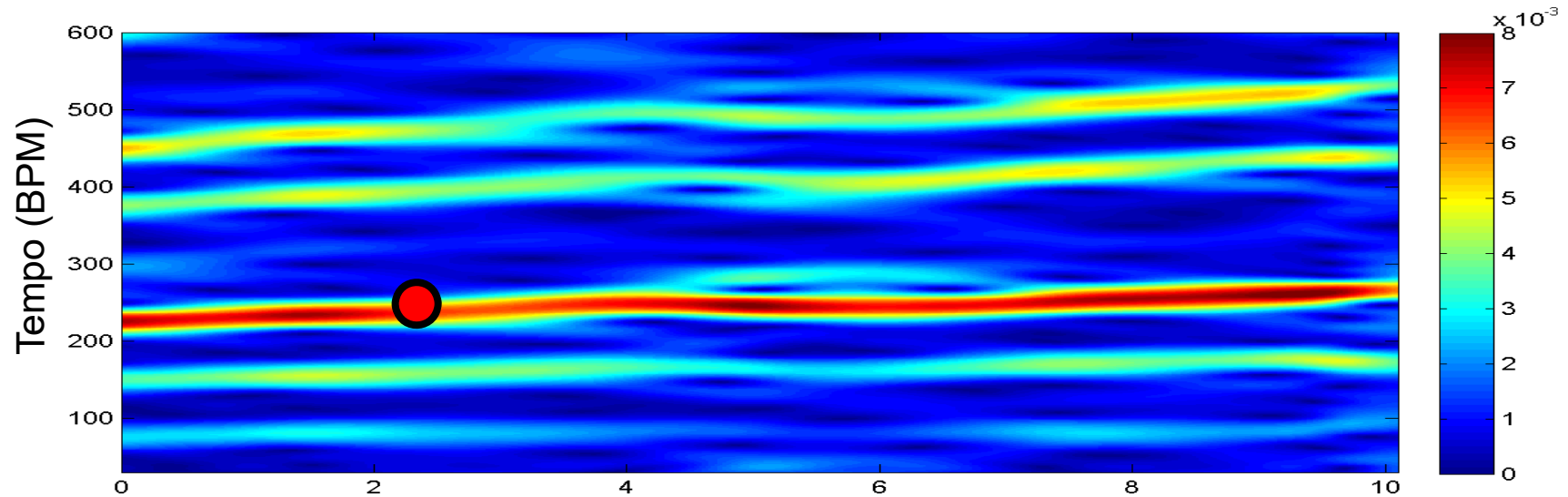


Normalized novelty function

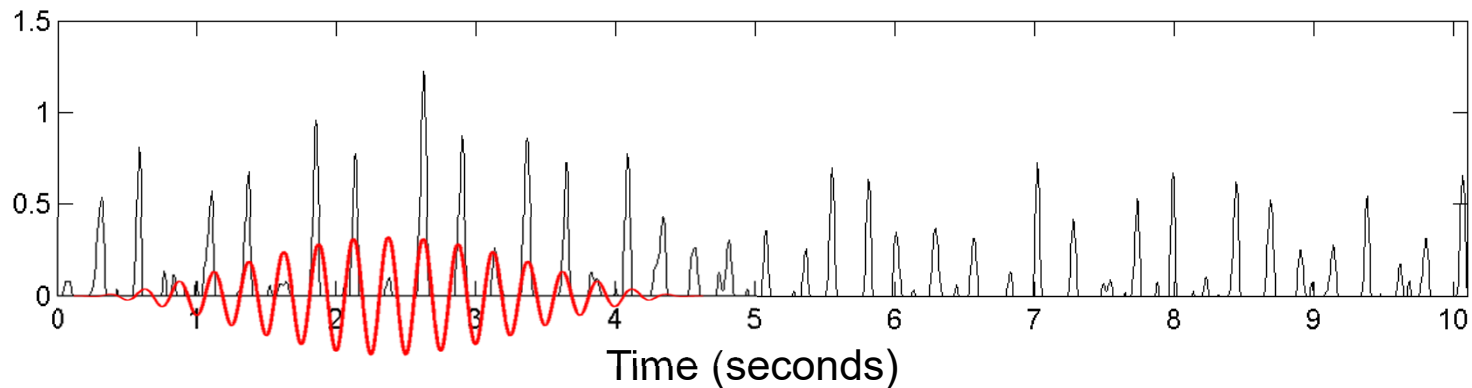


Local Pulse and Tempo Tracking

Fourier temogram (STFT of novelty function)

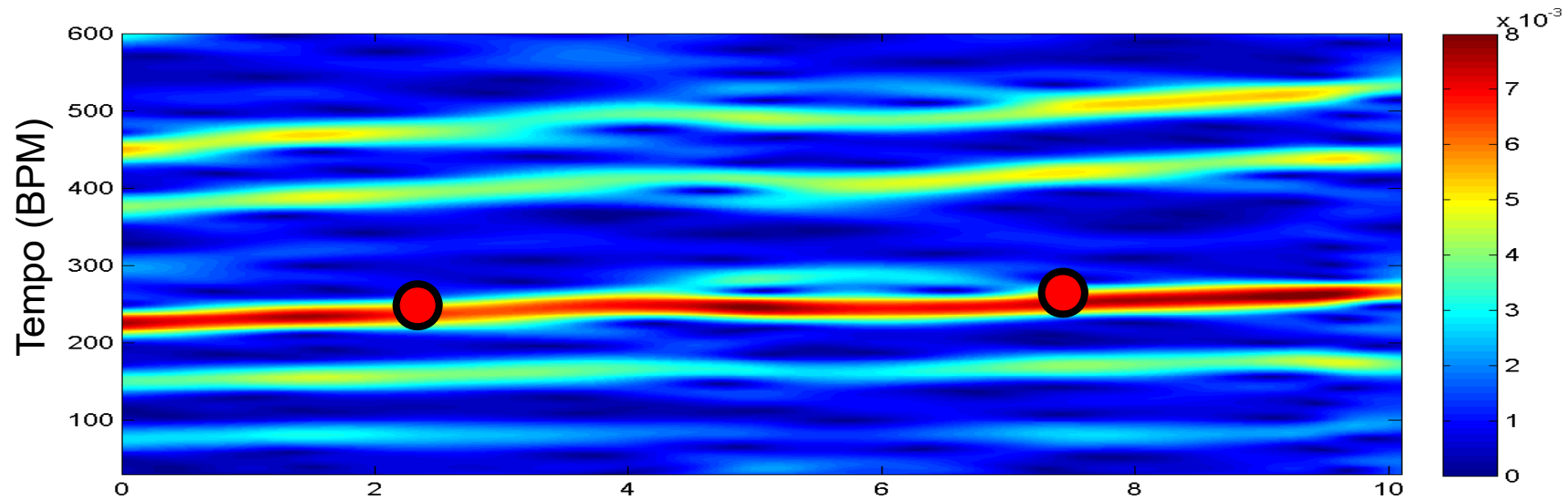


Optimizing local periodicity kernel

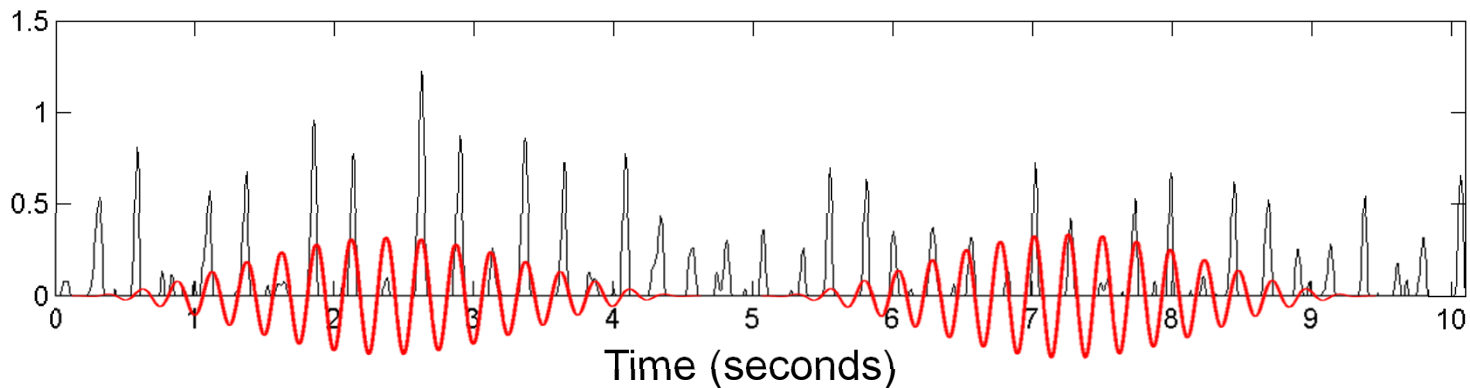


Local Pulse and Tempo Tracking

Fourier temogram (STFT of novelty function)

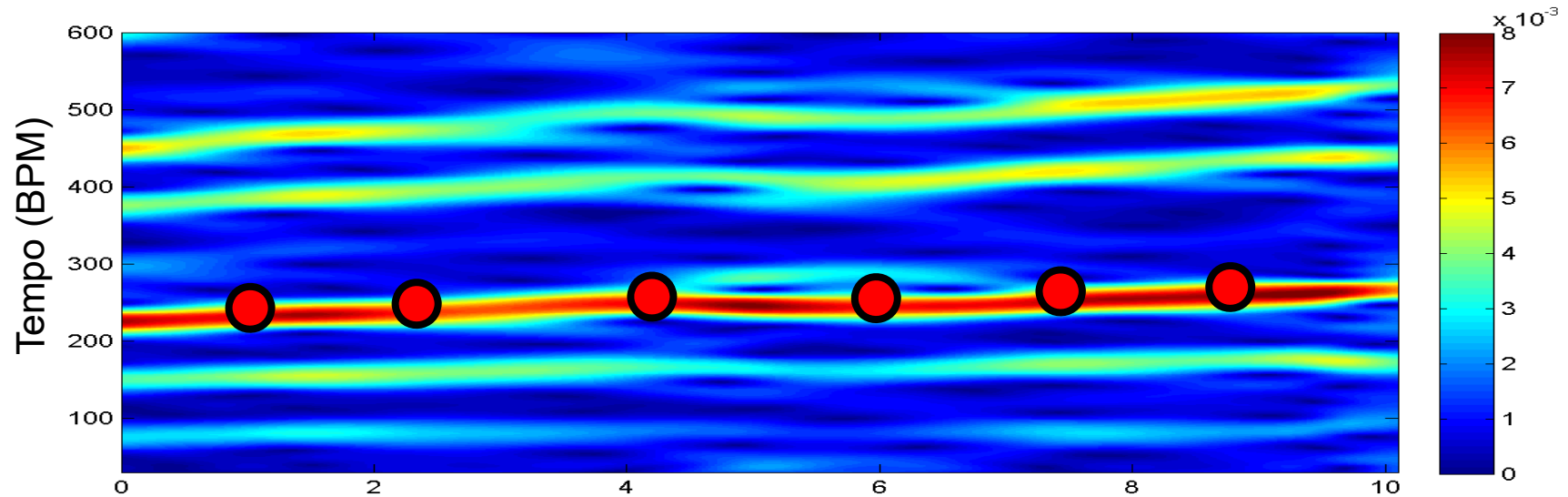


Optimizing local periodicity kernel

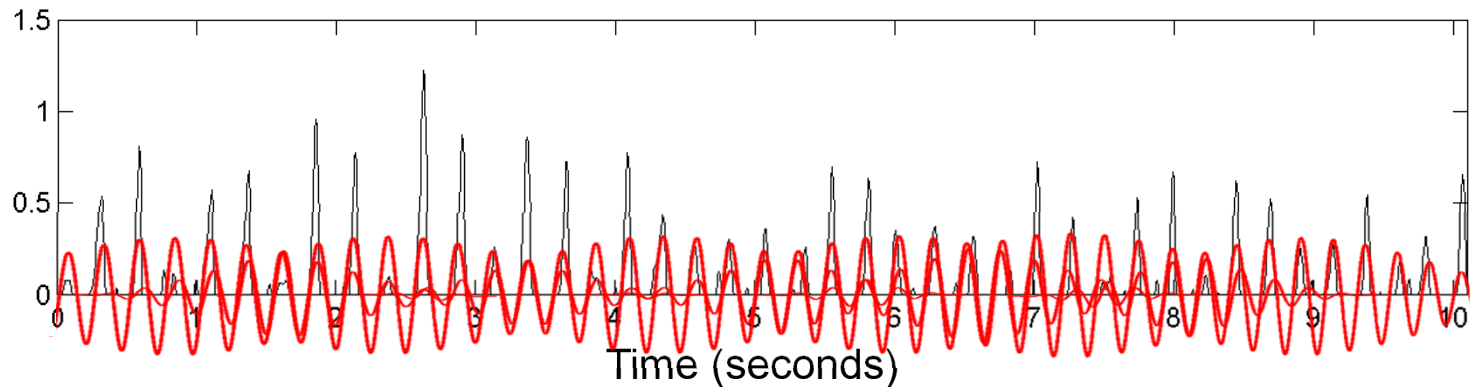


Local Pulse and Tempo Tracking

Fourier temogram (STFT of novelty function)

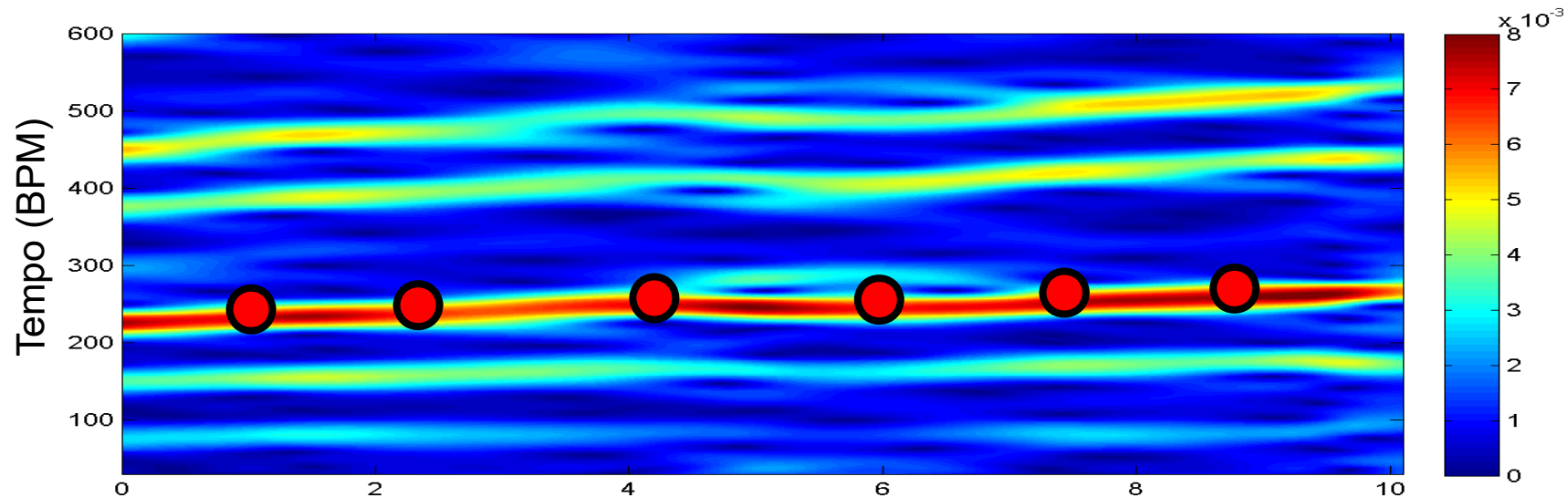


Optimizing local periodicity kernel

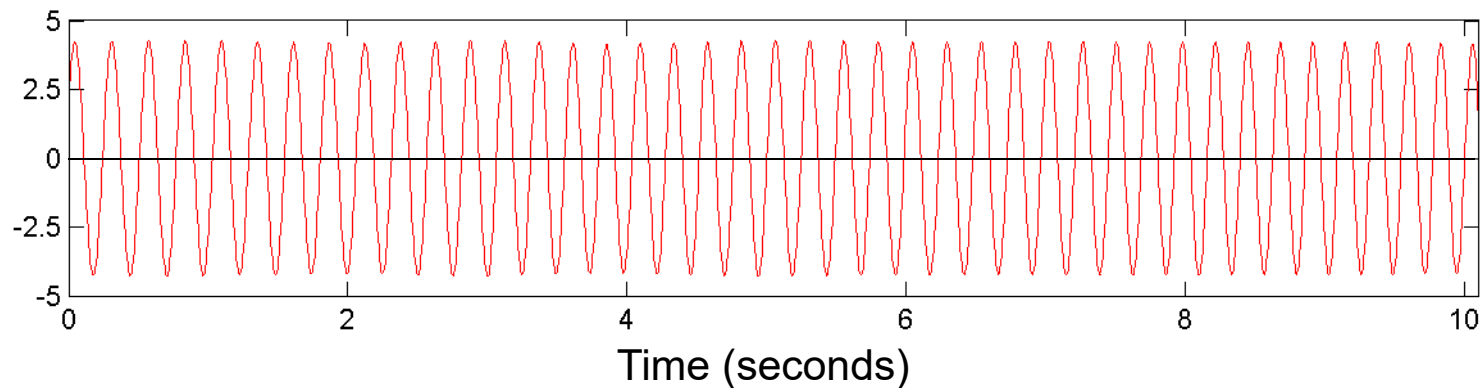


Local Pulse and Tempo Tracking

Fourier temogram (STFT of novelty function)

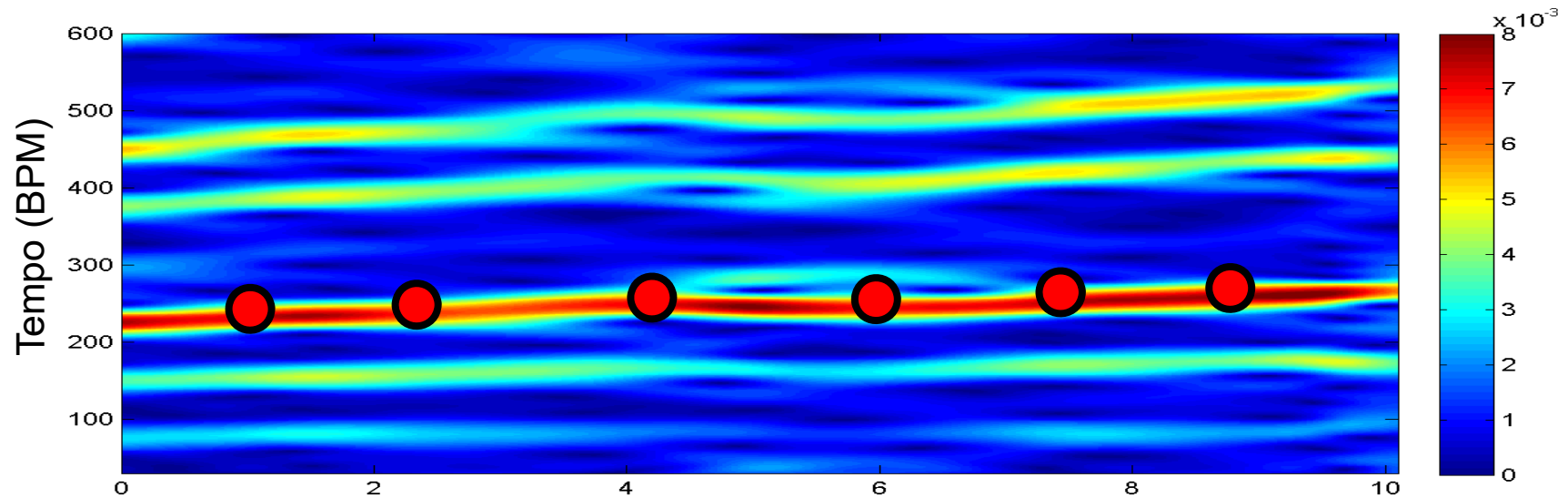


Accumulation of kernels

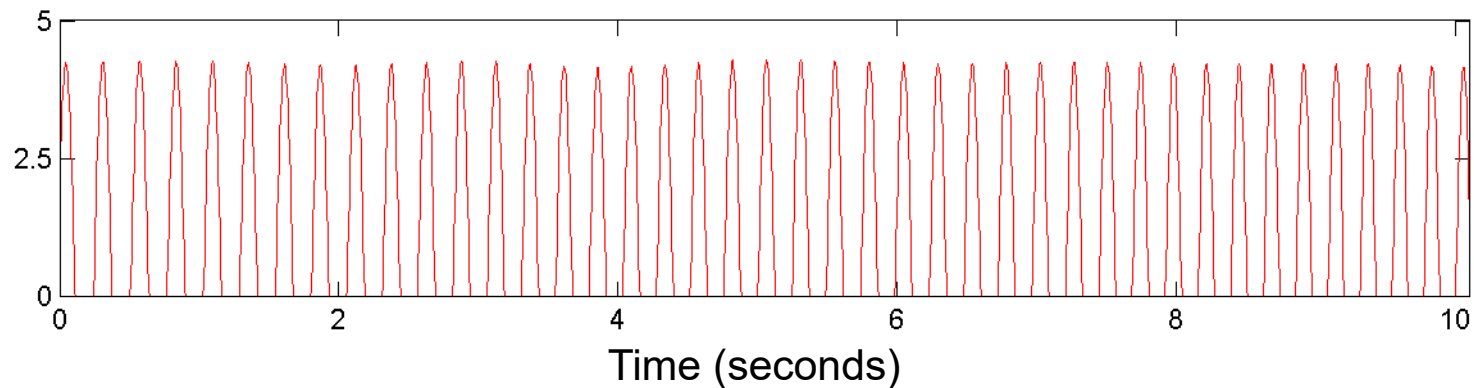


Local Pulse and Tempo Tracking

Fourier temogram (STFT of novelty function)

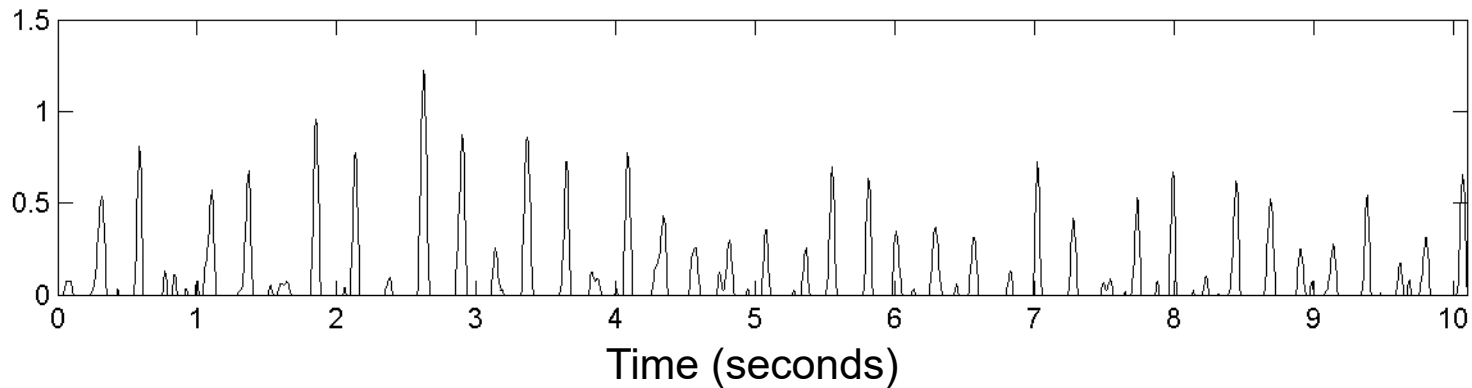


Halfwave rectification

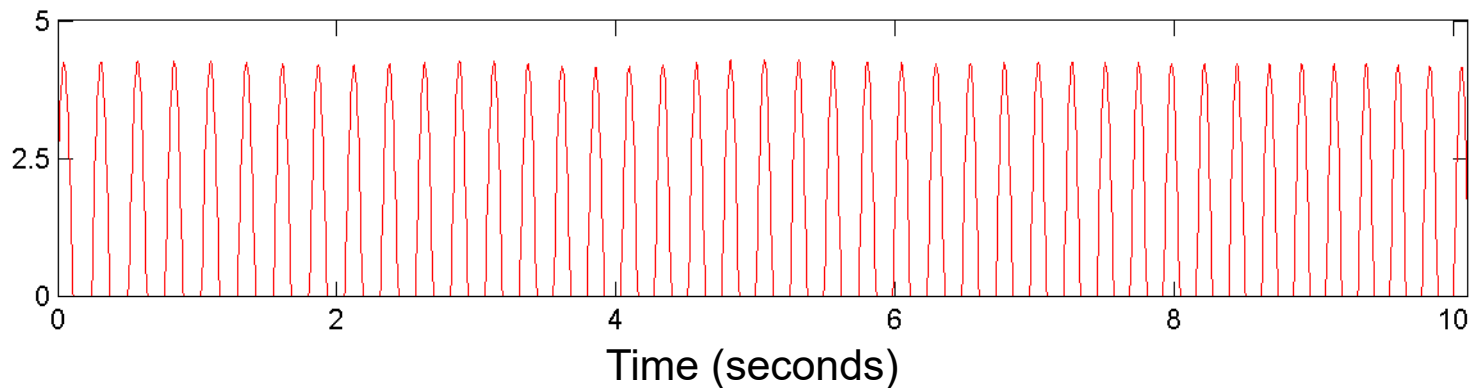


Local Pulse and Tempo Tracking

Novelty Curve



Predominant Local Pulse (PLP)



FMP Notebooks

Structured in 10 parts

- Part B: Basic introductions to
 - Jupyter notebook framework
 - Python programming
 - Other technical concepts underlying these notebooks
- Part 0: Starting notebook
- Part 1 to Part 8: Different music processing scenarios

Part	Title	Notions, Techniques & Algorithms	HTML	IPYNB
	Basics	Basic information on Python, Jupyter notebooks, Anaconda package management system, Python environments, visualizations, and other topics	[html]	[ipynb]
	Overview	Overview of the notebooks (https://www.audiolabs-erlangen.de/FMP)	[html]	[ipynb]
	Music Representations	Music notation, MIDI, audio signal, waveform, pitch, loudness, timbre	[html]	[ipynb]
	Fourier Analysis of Signals	Discrete/analog signal, sinusoid, exponential, Fourier transform, Fourier representation, DFT, FFT, STFT	[html]	[ipynb]
	Music Synchronization	Chroma feature, dynamic programming, dynamic time warping (DTW), alignment, user interface	[html]	[ipynb]
		Similarity matrix, repetition,	[html]	[ipynb]
			[html]	[ipynb]
	Tempo and Beat Tracking	Onset, novelty, tempo, tempogram, beat, periodicity, Fourier analysis, autocorrelation	[html]	[ipynb]
	Content-Based Audio Retrieval	Identification, fingerprint, indexing, inverted list, matching, version, cover song	[html]	[ipynb]
	Musically Informed Audio Decomposition	Harmonic/percussive separation, signal reconstruction, instantaneous frequency, fundamental frequency (F0), trajectory, nonnegative matrix factorization (NMF)	[html]	[ipynb]

Part 6: Tempo and Beat Tracking

Part B: Basics

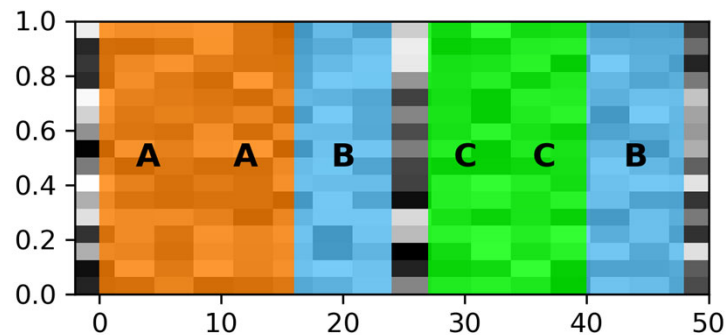
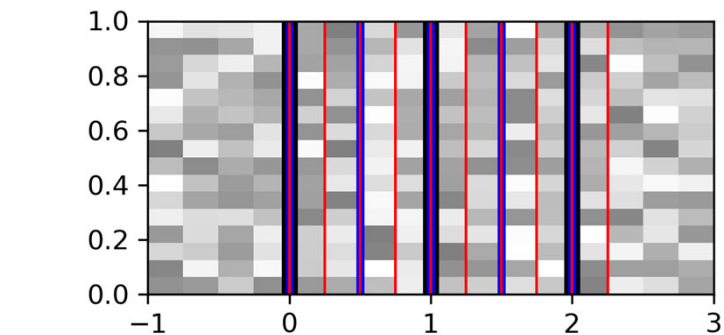
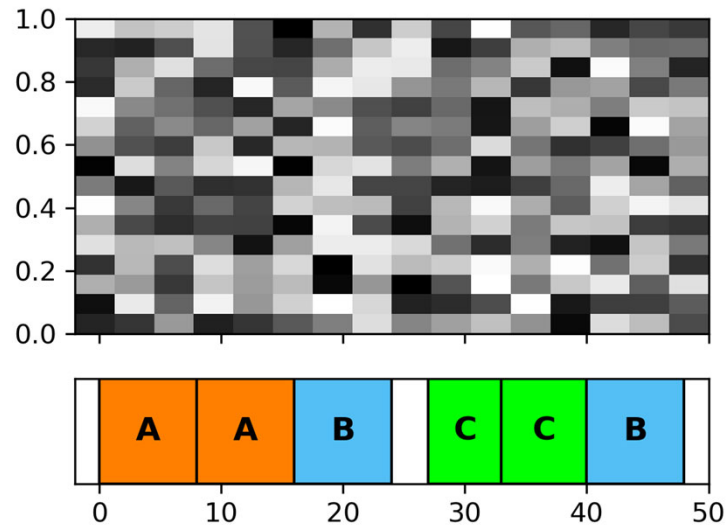
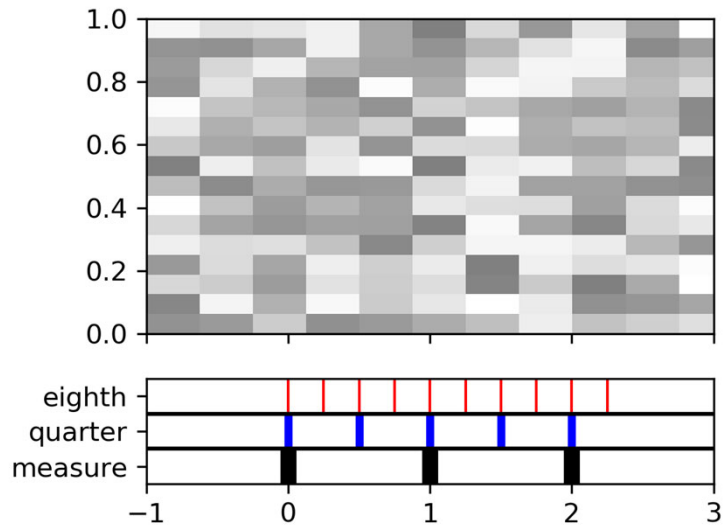


Topic	Description
Get Started	Explanation on how to install and use the FMP notebooks
Installation	Installation of Python using Conda
Jupyter Notebook	Usage of Jupyter notebook framework
Python Basics	Introduction of data types, control structures, and functions
Python Style Guide	Recommendations for programming style
Multimedia	Integration of multimedia objects into notebooks
Python Visualization	Generation of figures and images
Python Audio	Reading and writing audio files
Numba	Acceleration of Python functions via JIT compilation
Annotation Visualization	Visualization of annotations (single value, segments)
Sonification	Sonification methods (onsets, F0 trajectories, pitch, chroma)
libfmp	Library of FMP-specific Python functions
MIR Resources	Links to resources that are useful for MIR

Part B: Basics

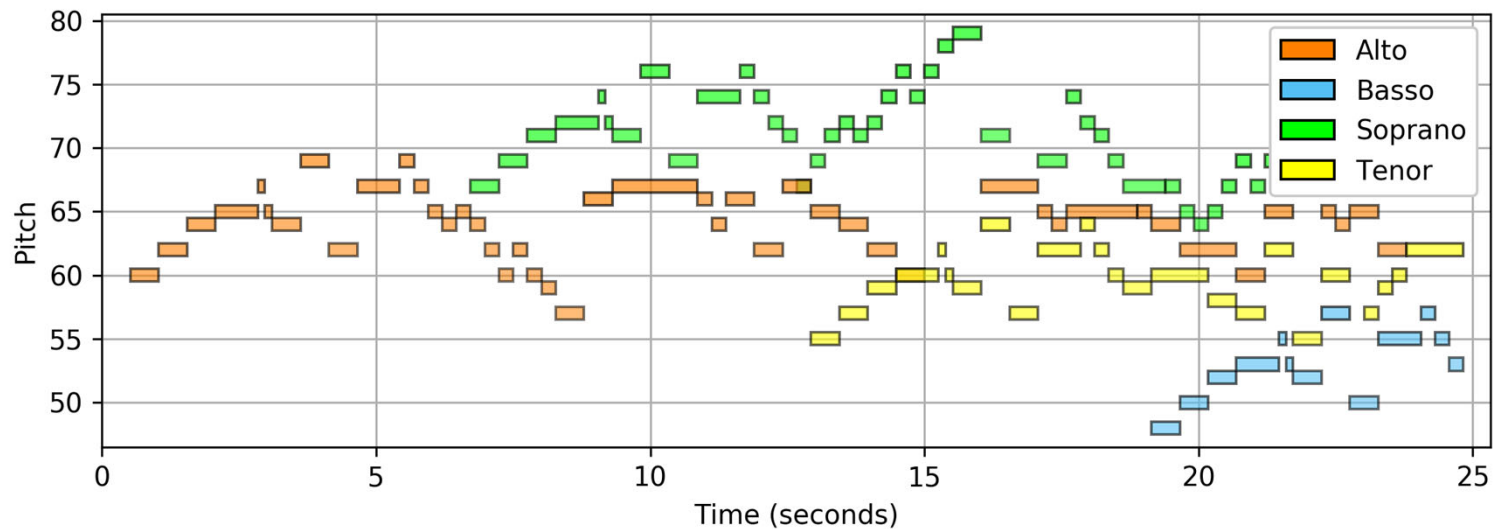
Annotation Visualization

Examples for visualizing annotations of time positions and segments.



Part 1: Music Representations

Symbolic Format: CSV



Visualization of a piano-roll representation
(Fugue BWV 846 by Bach).



Part 1: Music Representations

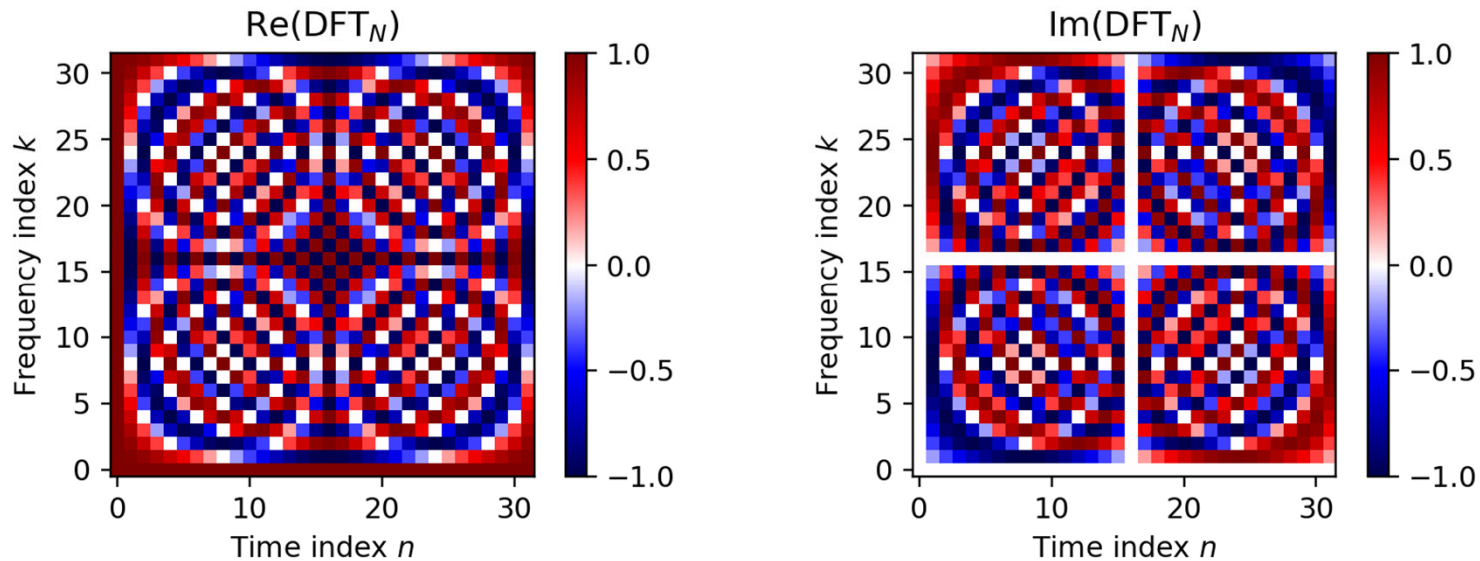
Waves and Waveforms



Videos illustrating the concepts of transverse, longitudinal, and combined waves.

Part 2: Fourier Analysis of Signals

Discrete Fourier Transform (DFT)



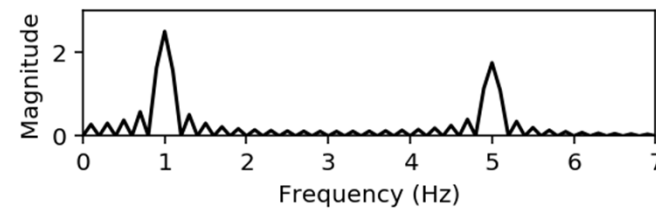
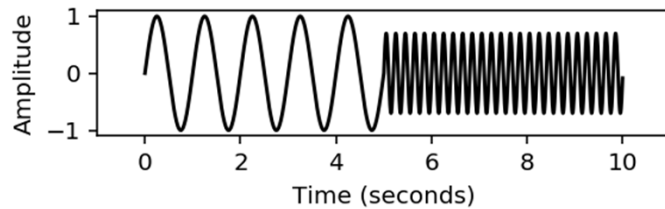
The matrix DFT_N and a visualization of its real and imaginary parts for the case $N = 32$

Part 2: Fourier Analysis of Signals

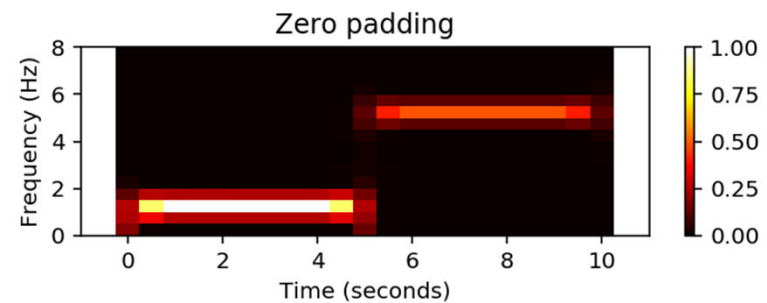
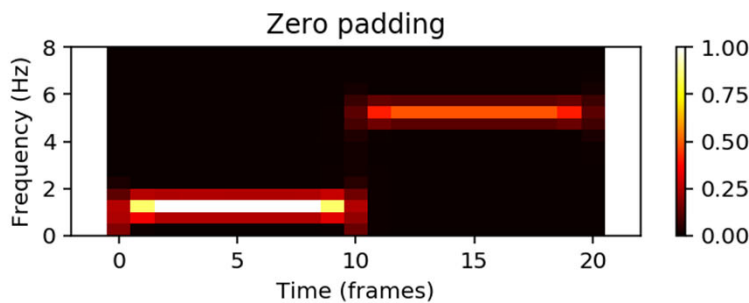
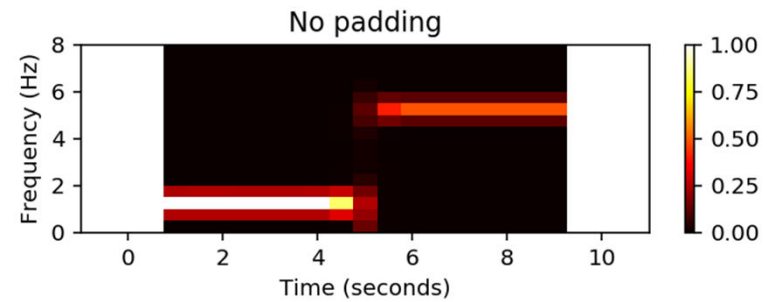
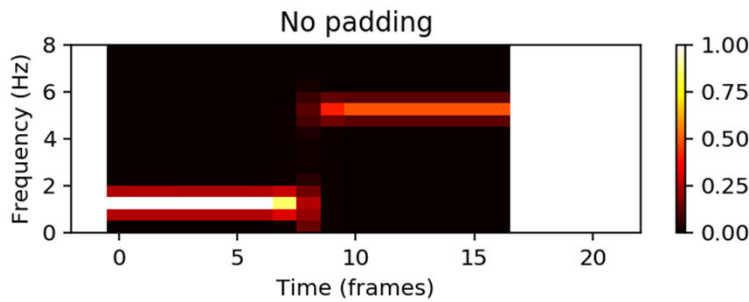


STFT: Padding

Time-domain signal and magnitude Fourier transform.



Magnitude STFT.

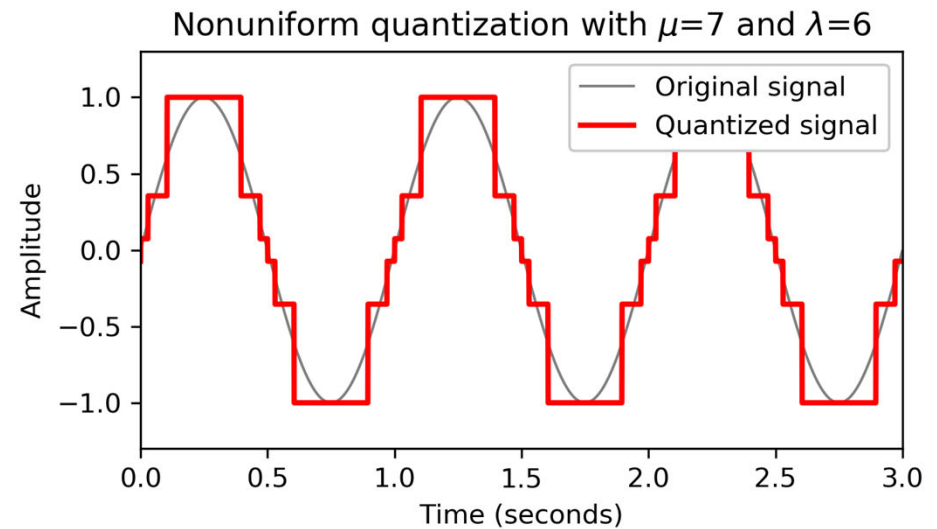
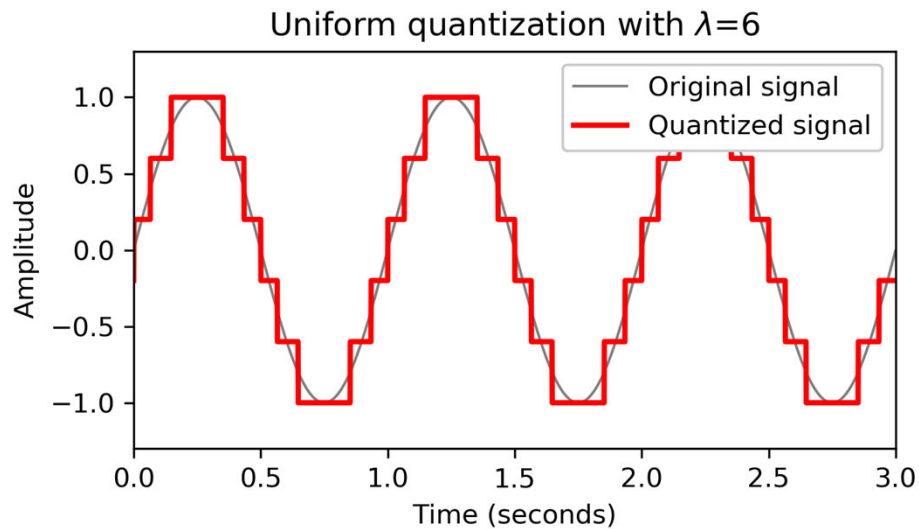


Part 2: Fourier Analysis of Signals



Digital Signals: Quantization

Uniform and nonuniform quantization (based on μ -law encoding) using $\lambda = 6$ quantization levels.

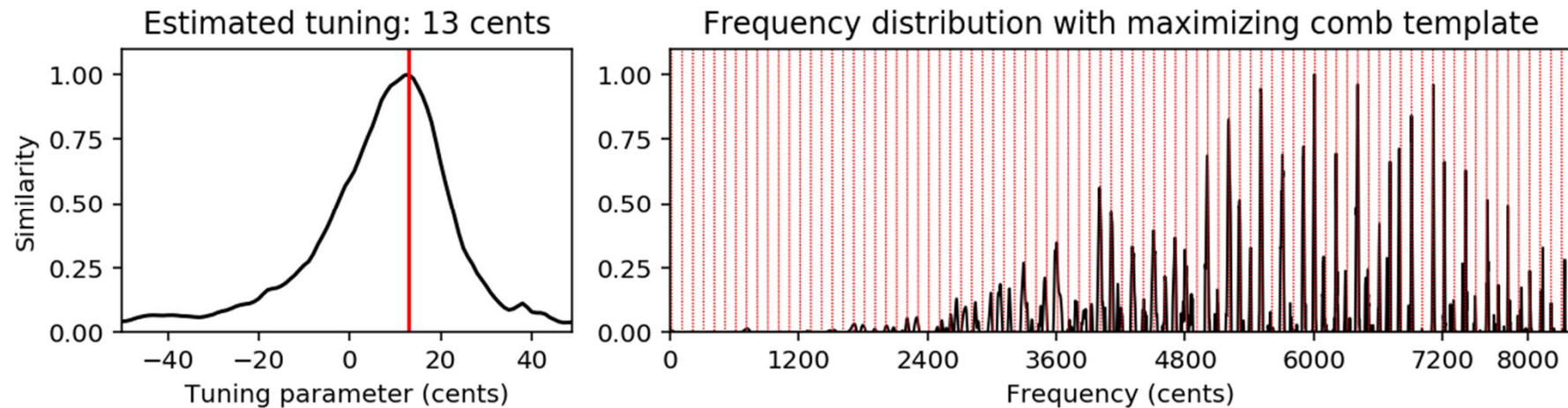


Part 3: Music Synchronization

Transposition and Tuning



Tuning procedure using a comb-filter approach.



A musical score snippet in 2/4 time. The right hand plays a melodic line with dynamics *p*, *p leggiermente*, *cresc.*, and *f*. The left hand plays a harmonic accompaniment. Performance markings include *1*, *3*, *5*, and a first/second ending bracket.

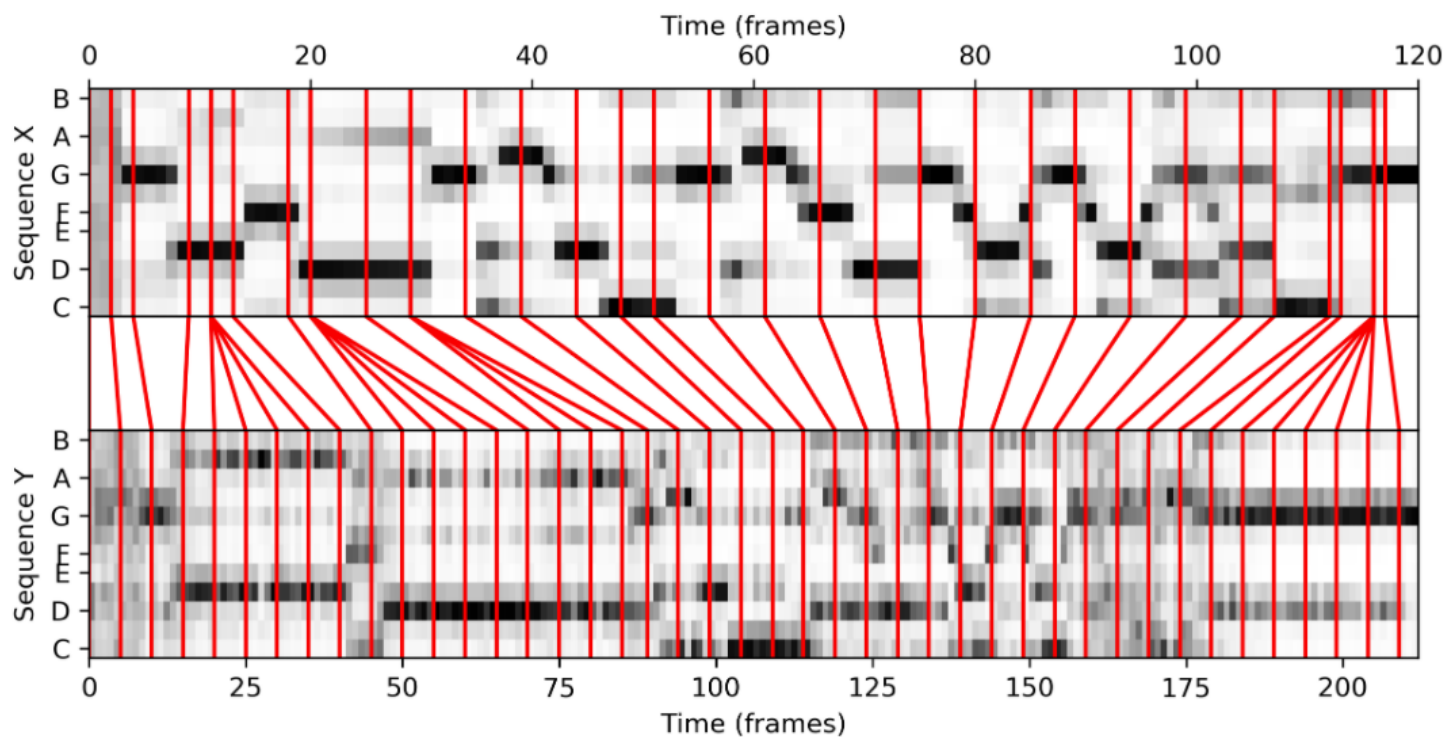


Part 3: Music Synchronization

Music Synchronization



Music synchronization result obtained for two input chromagrams (obtained from two recordings of the beginning of Beethoven's Fifth Symphony).



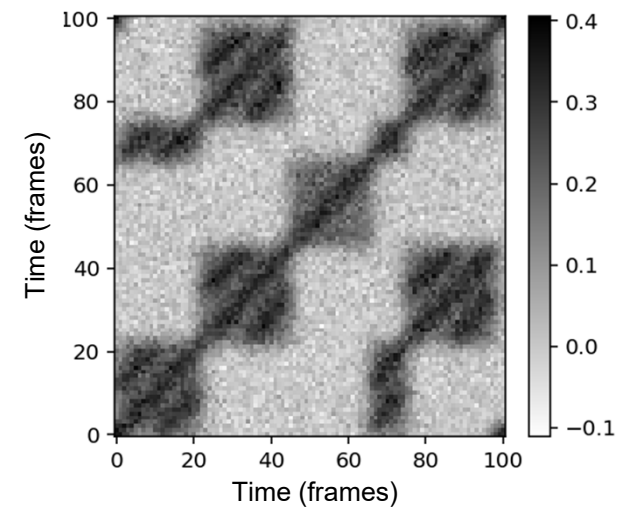
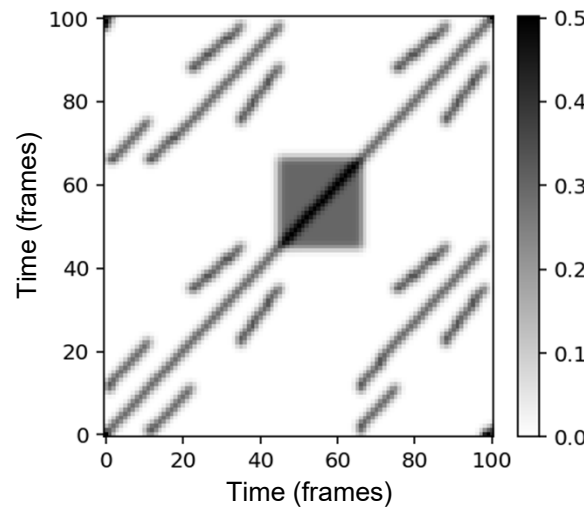
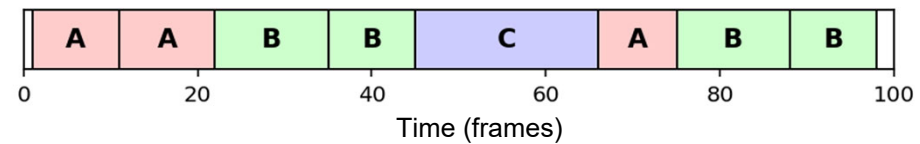
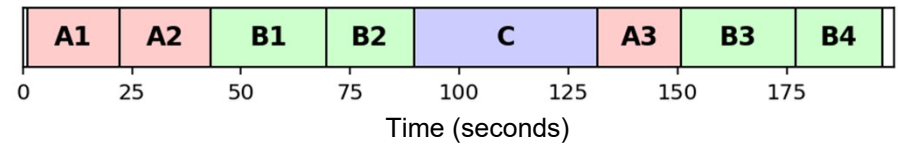
Part 4: Music Structure Analysis

SSM: Synthetic Generation

Structure annotation
and different
synthetically
generated SSMs.



	start	end	label
0	0.00	1.01	
1	1.01	22.11	A1
2	22.11	43.06	A2
3	43.06	69.42	B1
4	69.42	89.57	B2
5	89.57	131.64	C
6	131.64	150.84	A3
7	150.84	176.96	B3
8	176.96	196.90	B4
9	196.90	199.64	

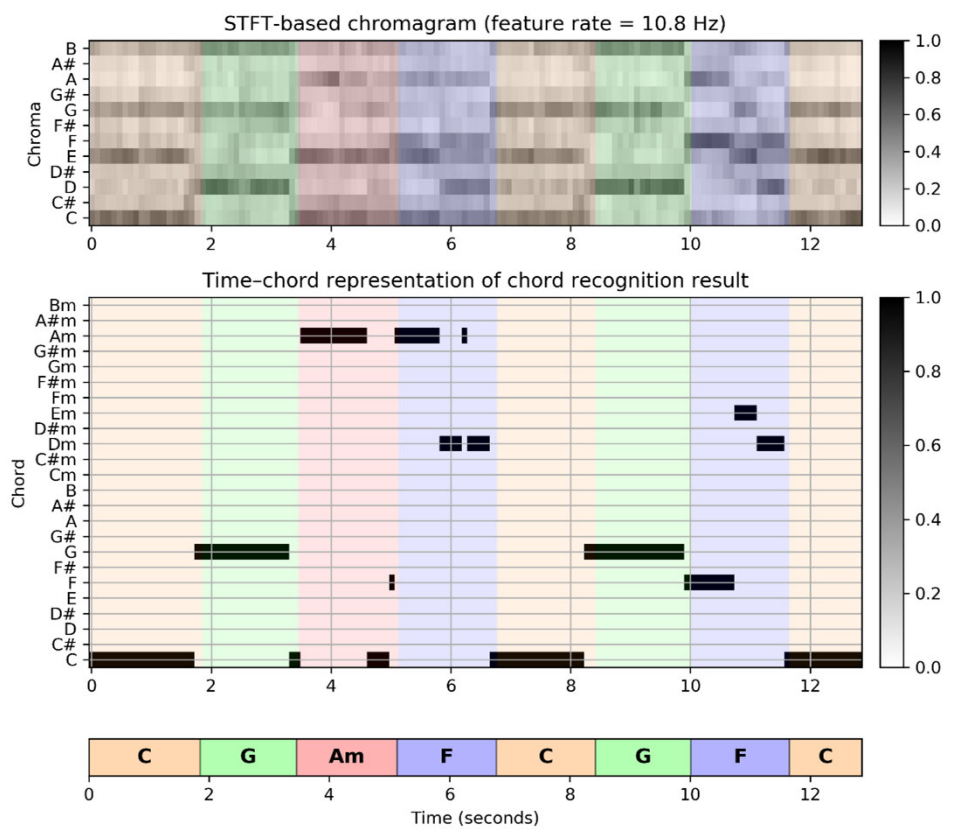


Part 5: Chord Recognition

Template-Based Chord Recognition



Chord recognition task illustrated by the first measures of the Beatles song "Let It Be."



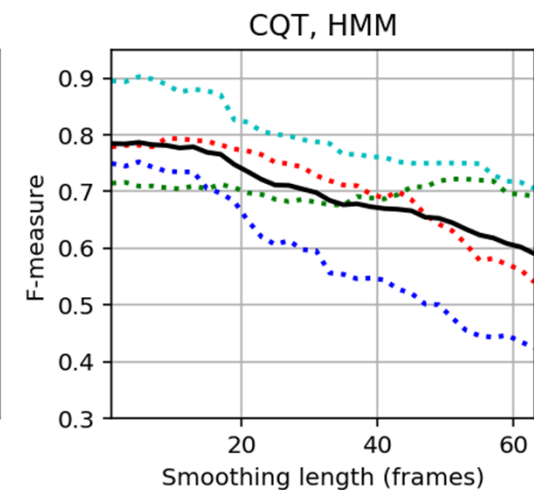
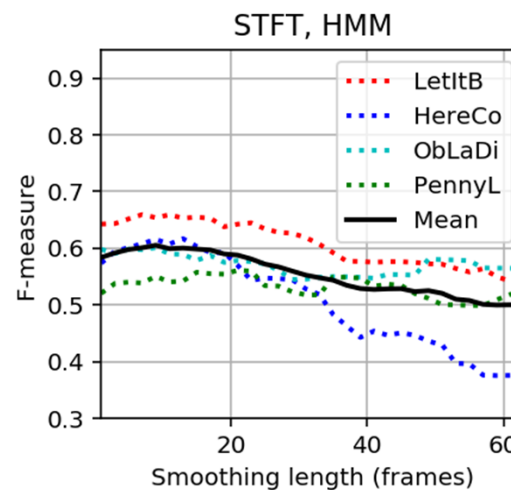
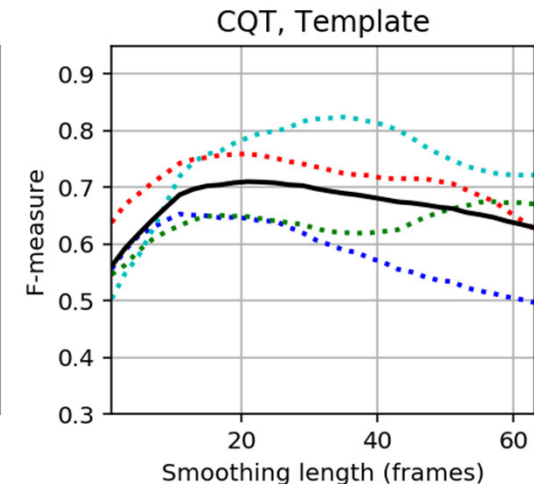
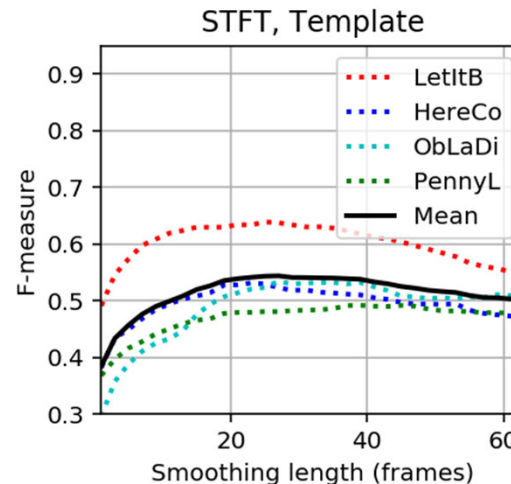
Part 5: Chord Recognition

Experiments: Beatles Collection



Prefiltering experiments for a template-based and an HMM-based chord recognizer applied to two different input chroma representations (STFT, CQT).

The evaluation is performed on the basis of four Beatles songs (LetItB, HereCo, ObLaDi, PennyL).

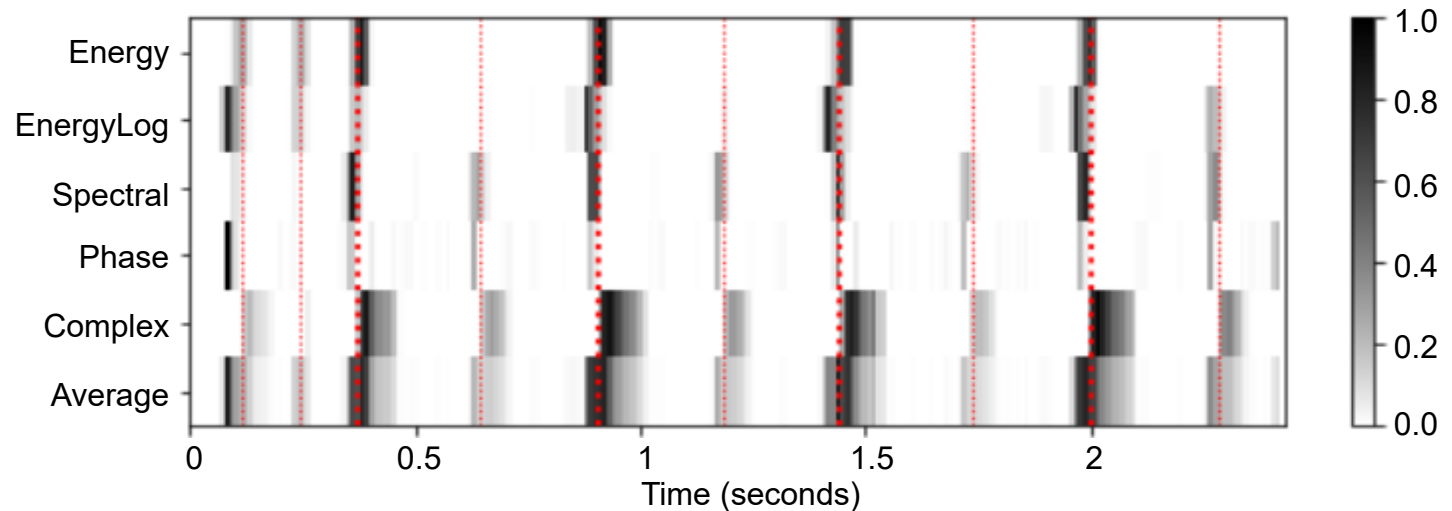
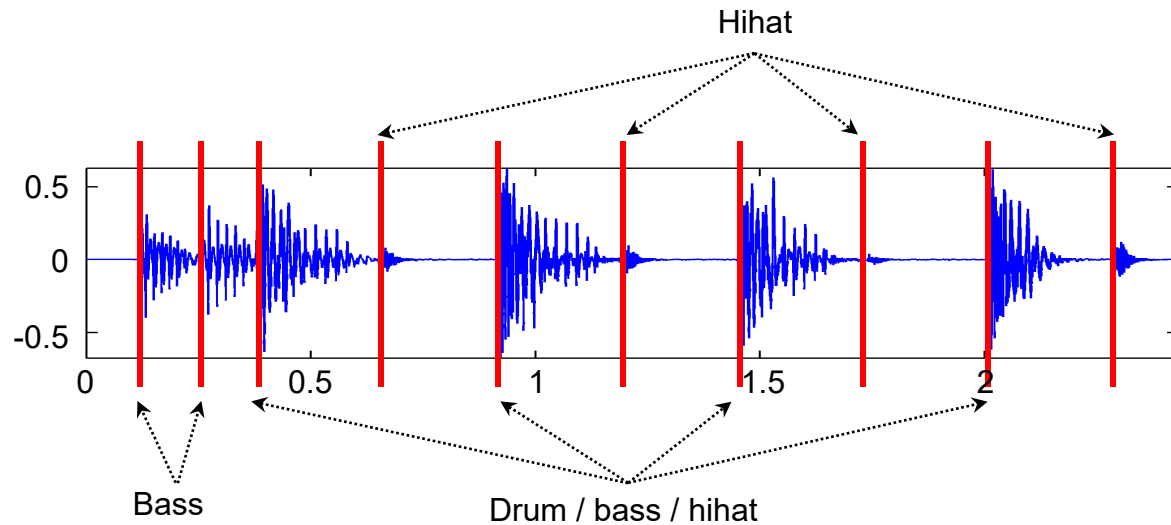


Part 6: Tempo and Beat Tracking

Novelty: Comparison of Approaches



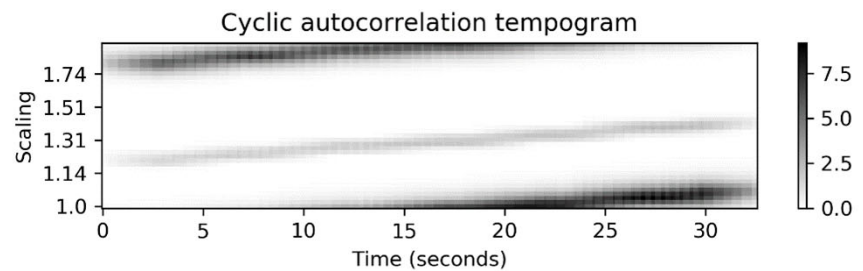
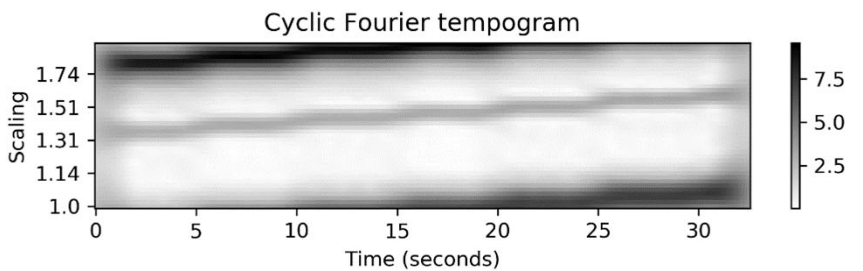
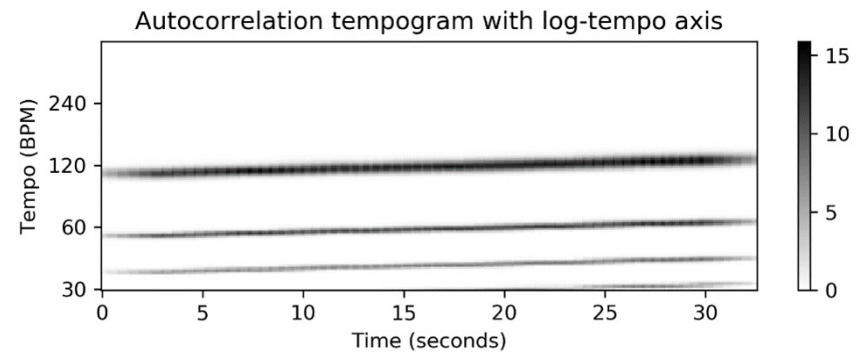
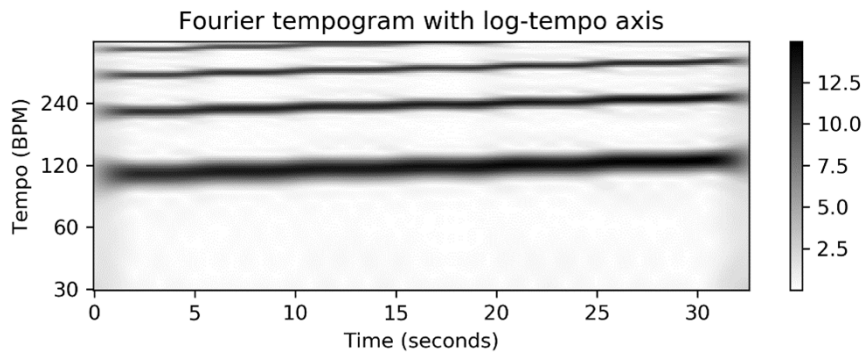
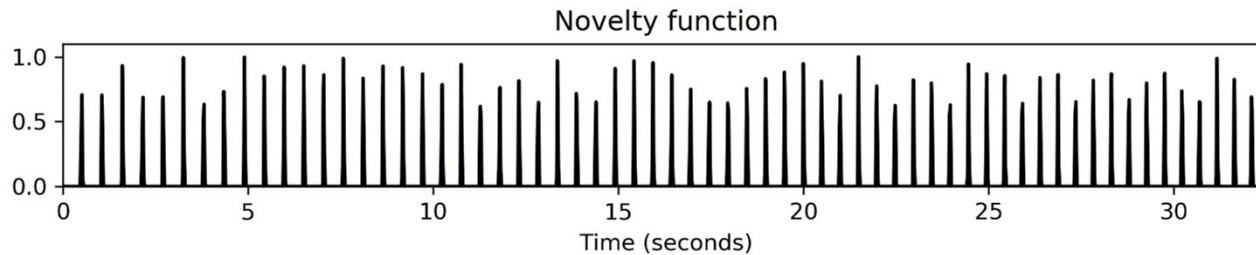
Comparison of novelty detectors using a matrix-based visualization.



Part 6: Tempo and Beat Tracking

Cyclic Tempogram

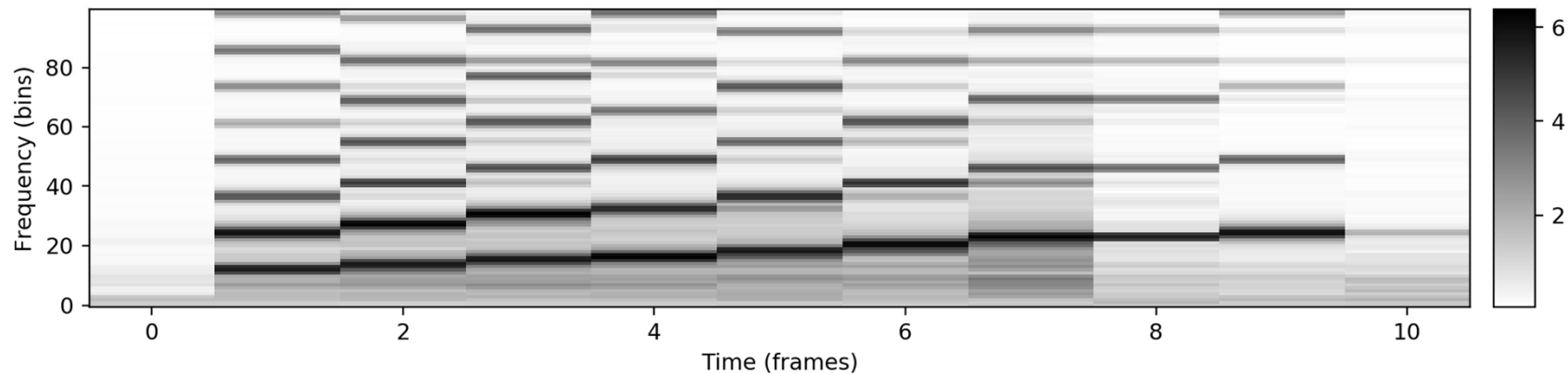
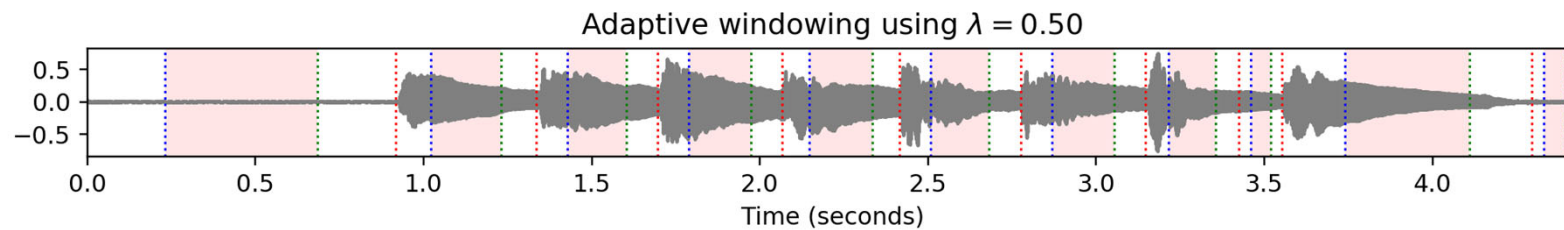
Different tempogram representations of a click track with increasing tempo.



Part 6: Tempo and Beat Tracking

Adaptive Windowing

Example of adaptive windowing using a parameter λ to control the neighborhood's relative size to be excluded.

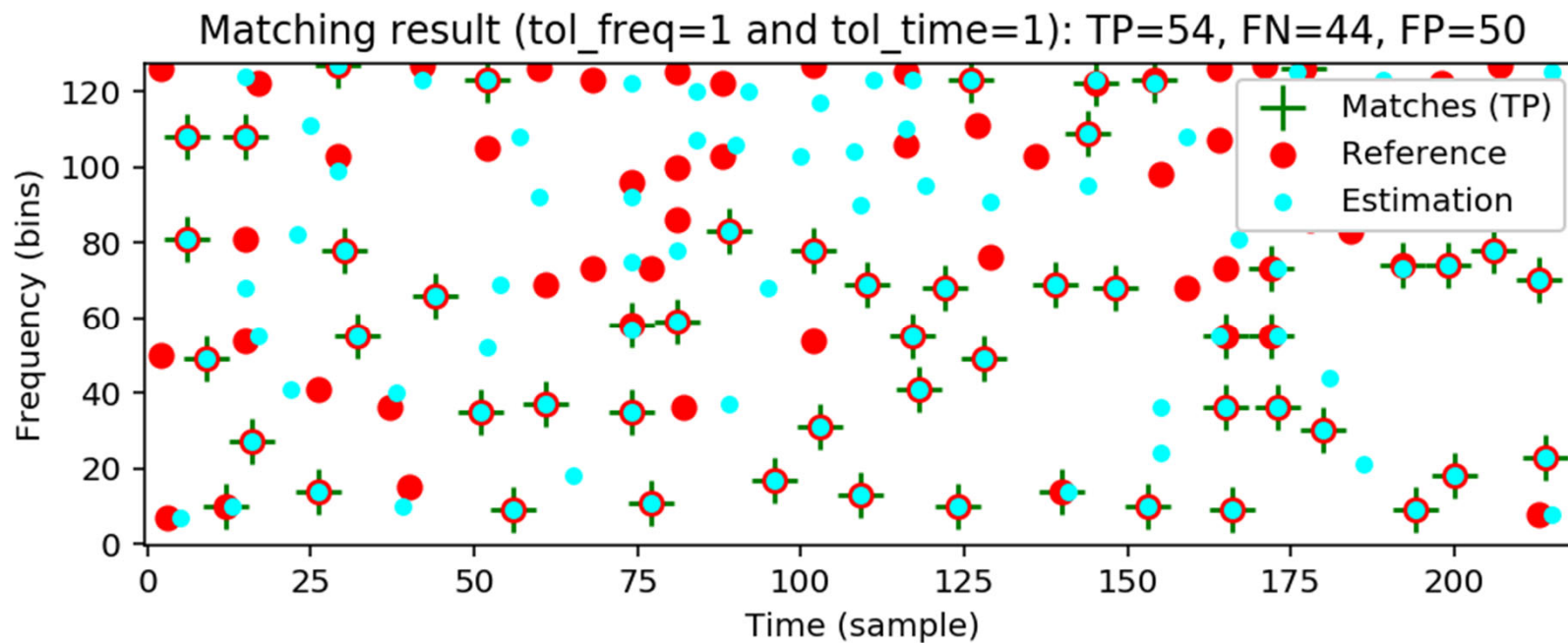


Part 7: Content-Based Audio Retrieval

Audio Identification



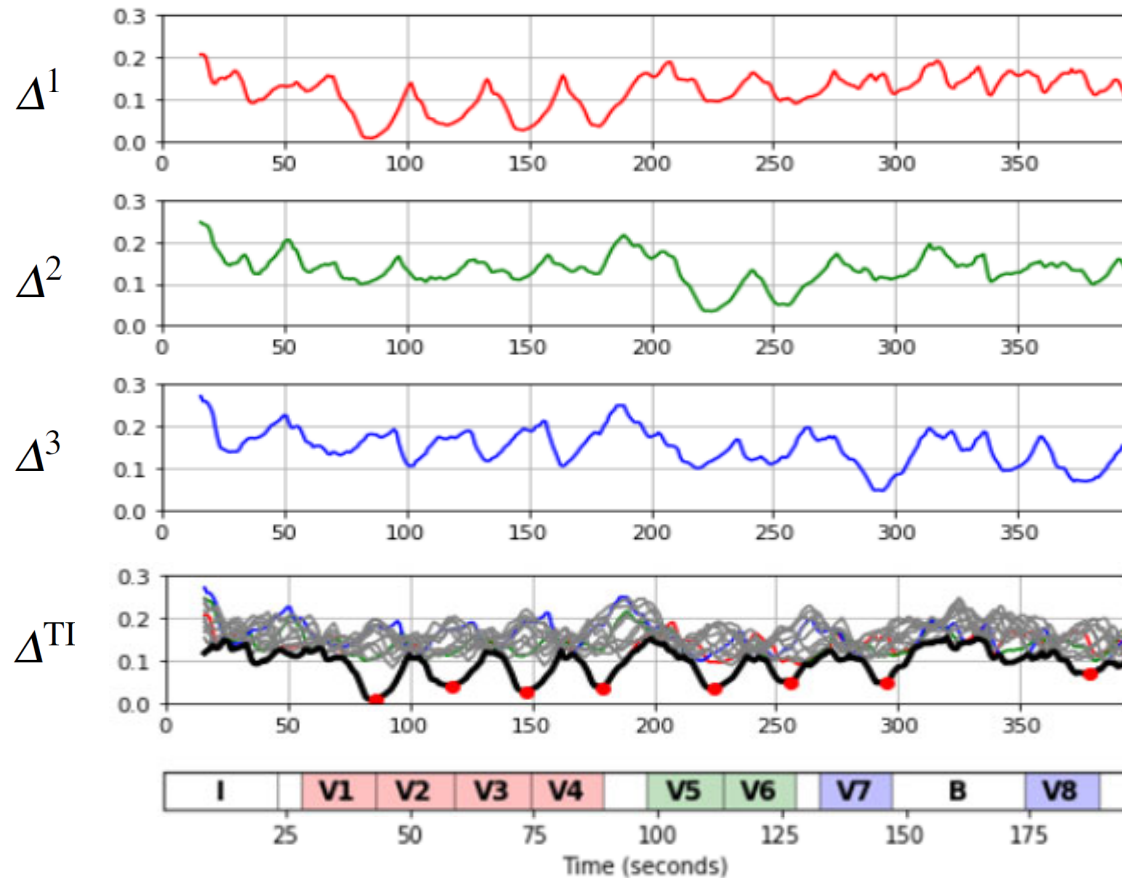
Evaluation measures that indicate the agreement between two constellation maps computed for an original version (Reference) and a noisy version (Estimation).



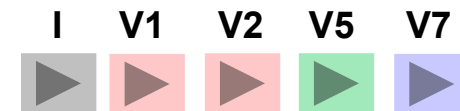
Part 7: Content-Based Audio Retrieval



Audio Matching



Transposition-invariant matching function illustrated by Zager and Evans' song "In the Year 2525."



Part 7: Content-Based Audio Retrieval

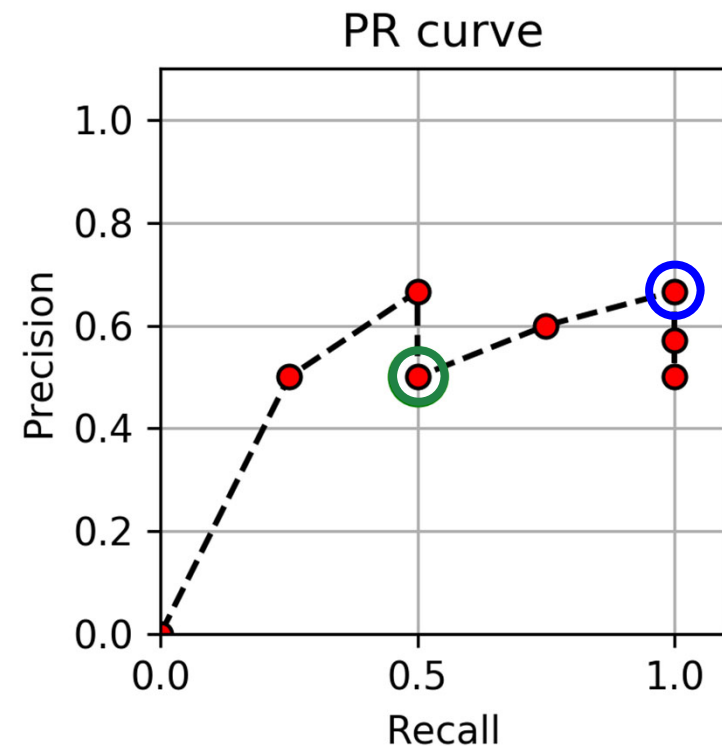
Evaluation Measures



Various evaluation metrics applied to a toy example.

Rank	ID	Score	χ_Q	P(r)	R(r)	F(r)
1	6	3.70	False	0.00	0.00	0.00
2	3	3.60	True	0.50	0.25	0.33
3	4	3.50	True	0.67	0.50	0.57
4	5	3.20	False	0.50	0.50	0.50
5	8	3.10	True	0.60	0.75	0.67
6	2	2.60	True	0.67	1.00	0.80
7	7	1.50	False	0.57	1.00	0.73
8	1	0.70	False	0.50	1.00	0.67

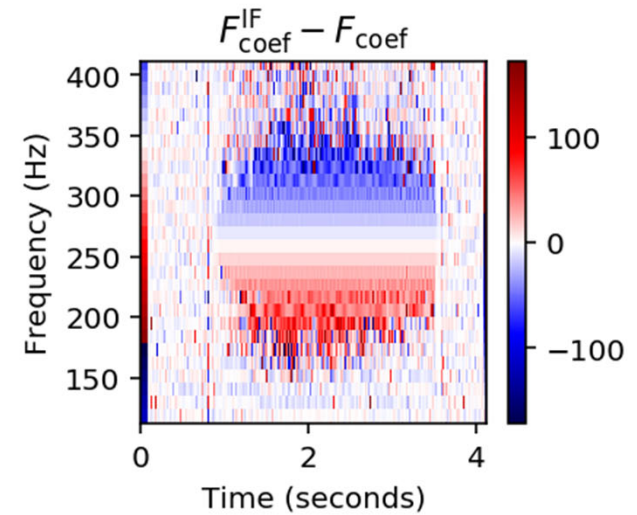
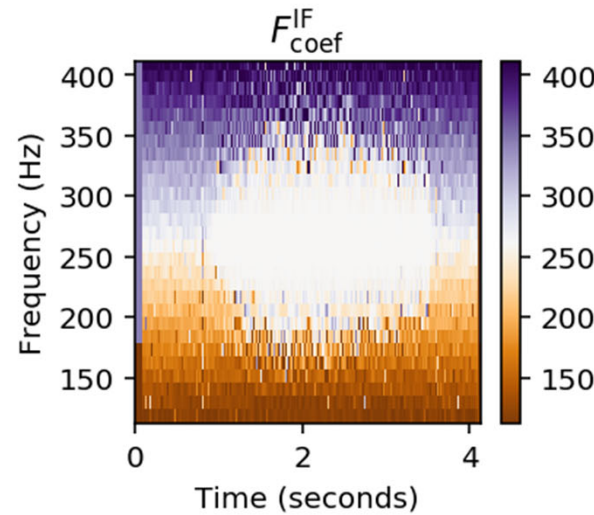
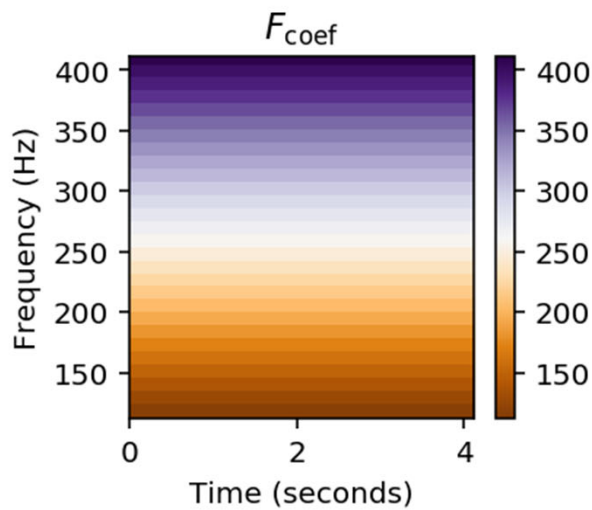
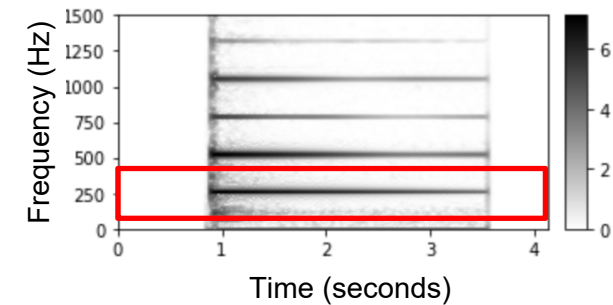
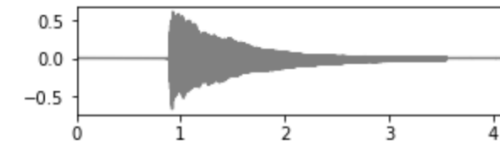
Break-even point = 0.50
F_max = 0.80
Average precision = 0.60833



Part 8: Audio Decomposition

Instantaneous Frequency Estimation

Interpretation of time–frequency bins of an STFT using (frame-dependent) instantaneous frequency values.



Part 8: Audio Decomposition

Fundamental Frequency Tracking

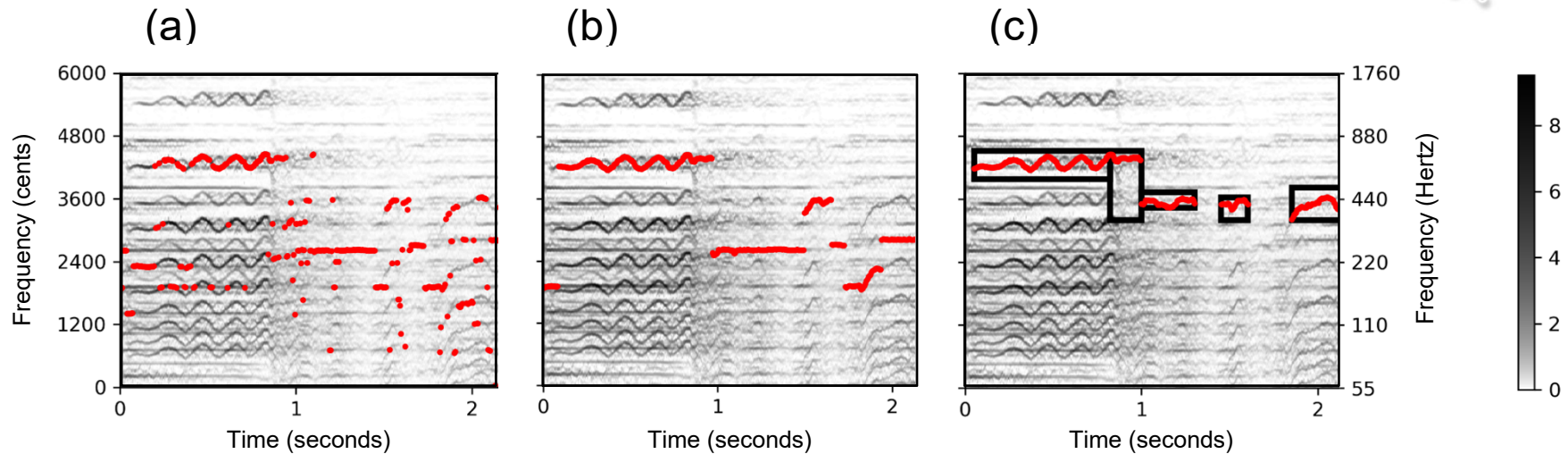


Saliency representation with trajectories computed by

- (a) a frame-wise approach,
- (b) an approach using continuity constraints, and
- (c) a score-informed approach.



Figure 8.10a from [Müller, FMP, Springer 2015]



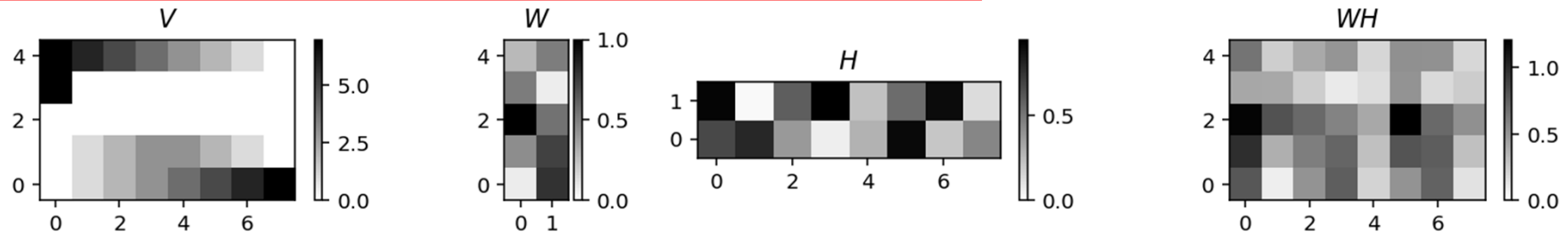
Part 8: Audio Decomposition

Nonnegative Matrix Factorization (NMF)

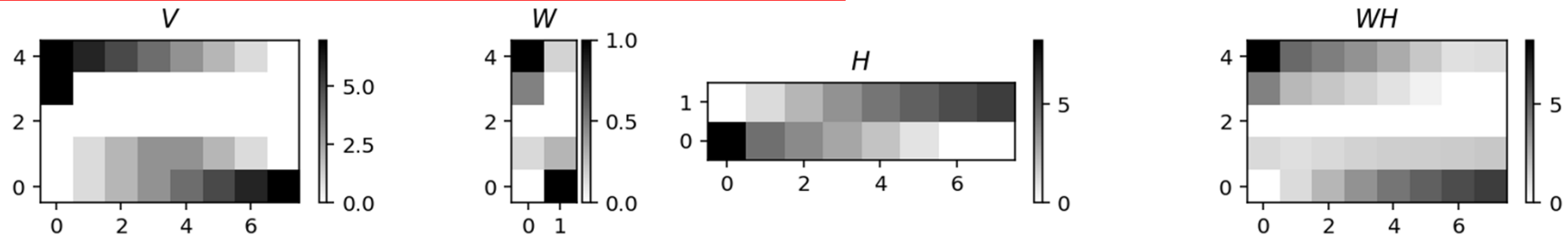
NMF procedure applied to a toy example.



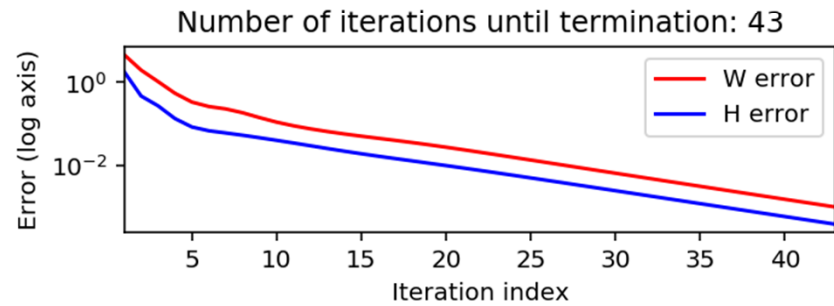
Matrix V and randomly initialized matrices W and H .

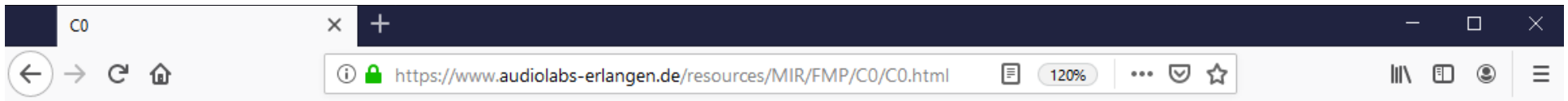


Matrix V and matrices W and H after training.



Error terms over iteration.



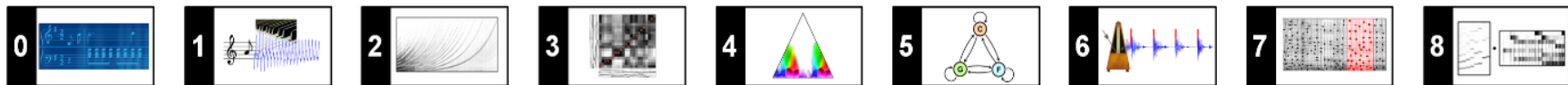


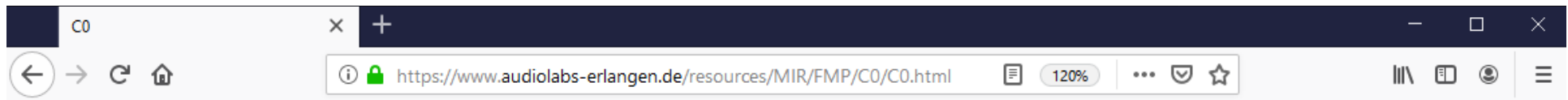
FMP Notebooks

Python Notebooks for Fundamentals of Music Processing



<https://www.audiolabs-erlangen.de/FMP>



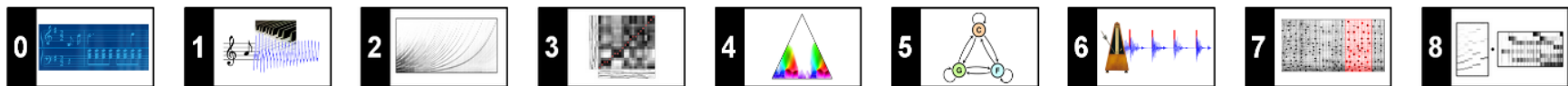


FMP Notebooks

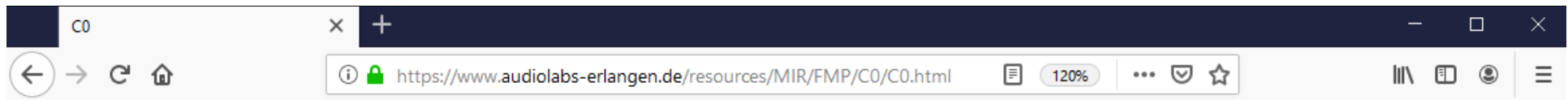
Python Notebooks for Fundamentals of Music Processing



<https://www.audiolabs-erlangen.de/FMP>



Basics + 8 Chapters

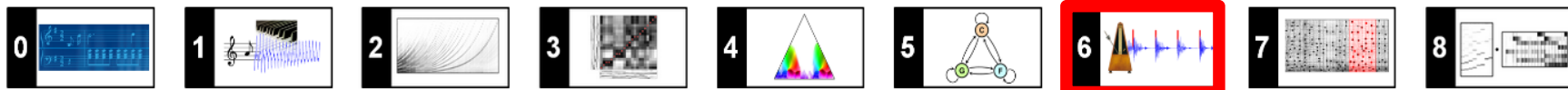


FMP Notebooks

Python Notebooks for Fundamentals of Music Processing

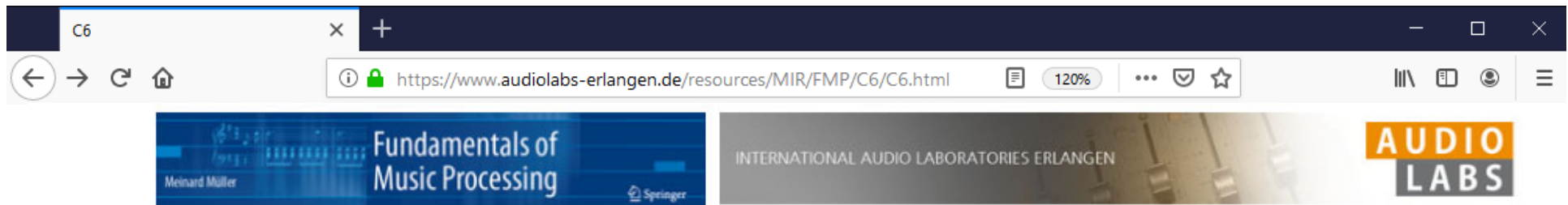


<https://www.audiolabs-erlangen.de/FMP>

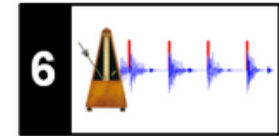


Basics + 8 Chapters

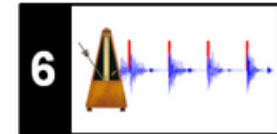
Tempo and Beat Tracking



Tempo and Beat Tracking



Tempo and Beat Tracking

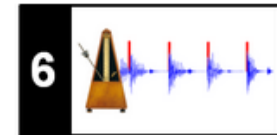


Definition

We assume that we are given a discrete-time novelty function $\Delta : \mathbb{Z} \rightarrow \mathbb{I}$ indicate note onset candidates. The idea of Fourier analysis is to detect local periodicity in novelty curve by comparing it with windowed sinusoids. A high correlation of Δ with a windowed sinusoid indicates a periodicity of the novelty curve (given a suitable phase). This correlation (along with the phase) can be computed via the short-time Fourier transform. To this end, we fix a window function $w : \mathbb{Z} \rightarrow \mathbb{R}$ of length centered at $n = 0$ (e.g., a sampled Hann window). Then, for a frequency parameter $\omega \in \mathbb{R}_{\geq 0}$ and time parameter $n \in \mathbb{Z}$, the complex Fourier coefficient is defined by

$$\mathcal{F}(n, \omega) := \sum_{m \in \mathbb{Z}} \Delta(m) \bar{w}(m - n) \exp(-2\pi i \omega m).$$

Tempo and Beat Tracking



Definition

We assume that we are given a discrete time novelty function $\Delta : \mathbb{Z} \rightarrow \mathbb{I}$ indicate note onset and in novelty curve by computing the correlation of Δ with a window (given a suitable phase). This correlation (along with the phase) is analyzed using the short-time Fourier transform. To this end, we fix a window function $w(m) = 0$ (e.g., a sampled Hann window). Then, for a time parameter $n \in \mathbb{Z}$, the complex Fourier coefficient is defined by

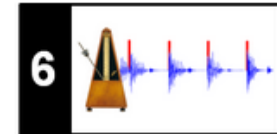
$$\mathcal{F}(n, \omega) := \sum_{m \in \mathbb{Z}} \Delta(m) \bar{w}(m - n) \exp(-2\pi i \omega m).$$

Explanations

Theory

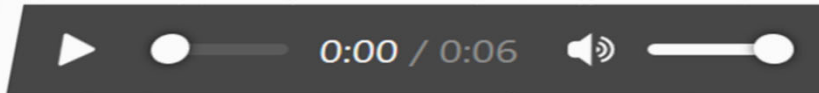
Mathematics

Tempo and Beat Tracking



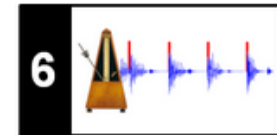
Example: Shostakovich

In the following example, we consider an excerpt of a recording of Dimitri Shostakovich's Suite for Variety Orchestra No. 1. The score version of the excerpt.



We start with a [spectral-based novelty function](#) resampled to F_s^Δ :
Furthermore, we use a window size corresponding to 5 seconds (l)

Tempo and Beat Tracking

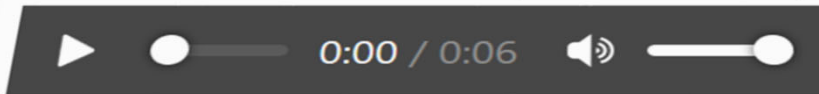


Example: Shostakovich

In the following example, we consider an excerpt from the Suite for Variety Orchestra.

Music Examples

Annotations

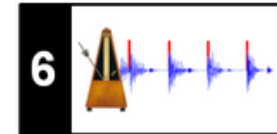


Audio

We start with a [spectral-based novelty function](#) resampled to F_s^Δ . Furthermore, we use a window size corresponding to 5 seconds (l)

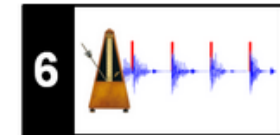
Links

Tempo and Beat Tracking



```
In [2]: def compute_sinusoid_optimal(c, tempo, n, Fs, N
        """Compute windowed sinusoid with optimal p
        Notebook: C6/C6S2_TempogramFourier.ipynb
        Args:
            c: Coefficient of tempogram (c=X(k,n))
            tempo: Tempo parameter corresponding to
            _coef_BPM[k])
            n: Frame parameter of c
            Fs: Sampling rate
            N: Window length
            H: Hop size
```

Tempo and Beat Tracking



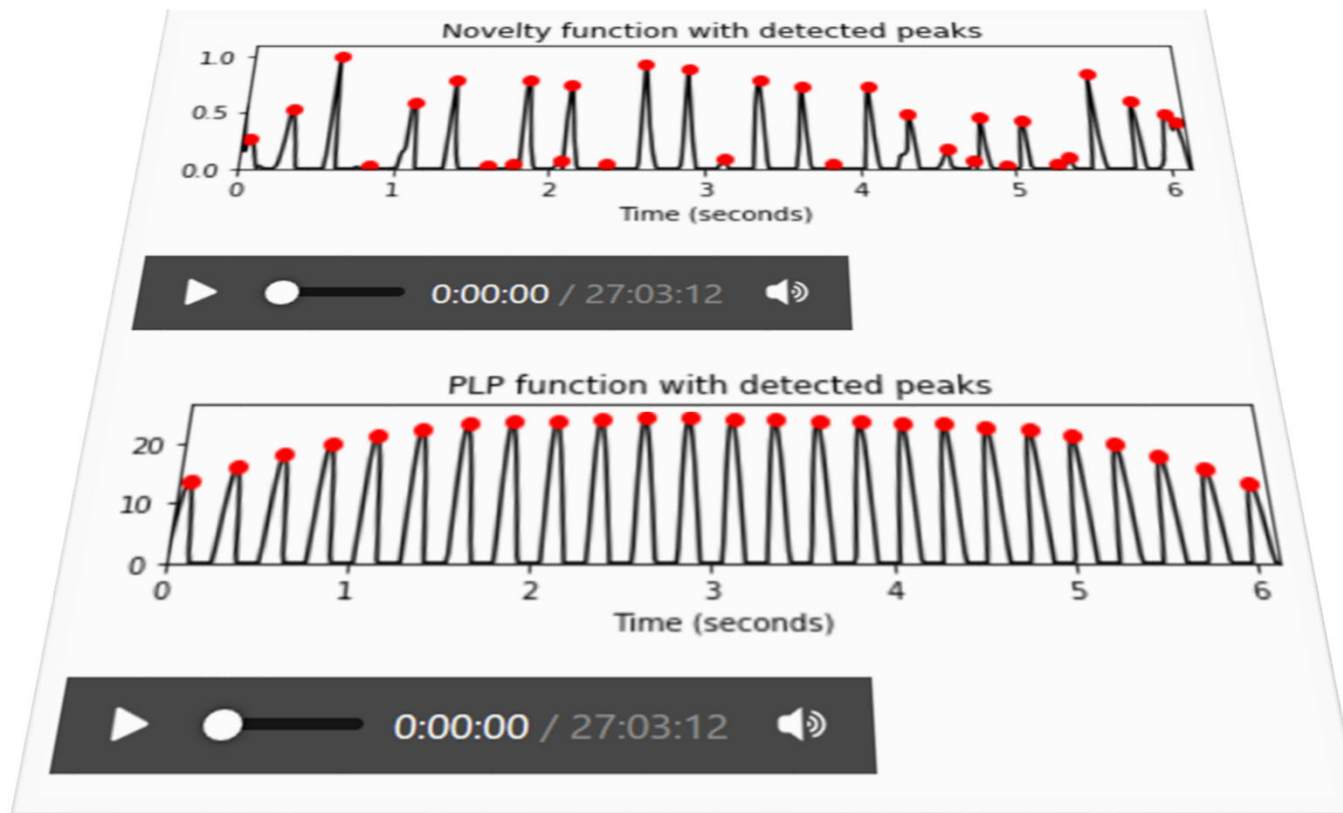
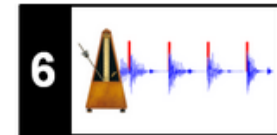
```
In [2]: def compute_sinusoid_coefficient(k, n, Fs, N, H):  
        """Compute windowed sinusoid coefficients for a given tempo parameter.  
        The function returns the coefficient c and the tempo parameter corresponding to the  
        coefficient c. The coefficient c is computed as c = X(k, n) * w(k, n).  
        The tempo parameter is computed as tempo = 60 * (Fs / (N * H * c)).  
        The function also returns the coefficient c and the tempo parameter corresponding to the  
        coefficient c.  
        Parameters  
        c: Coefficient of tempogram (c=X(k,n))  
        tempo: Tempo parameter corresponding to the coefficient c  
        _coef_BPM[k])  
        n: Frame parameter of c  
        Fs: Sampling rate  
        N: Window length  
        H: Hop size
```

Python Code

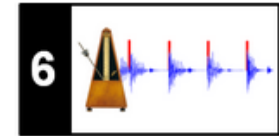
Algorithms

Functions

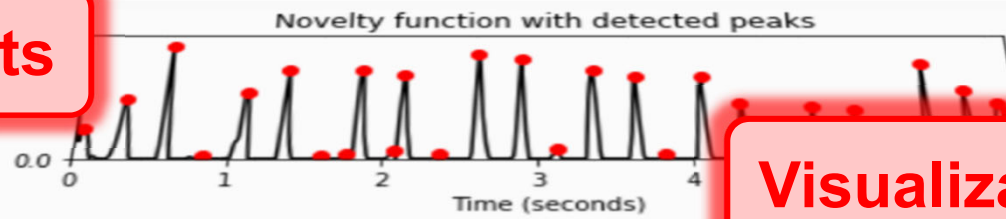
Tempo and Beat Tracking



Tempo and Beat Tracking

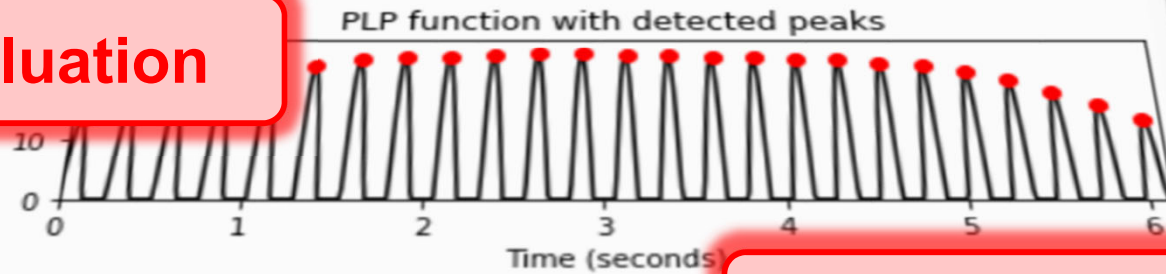


Results



Visualization

Evaluation



Sonification

FMP Notebooks

Teaching

Understanding

Programming

Baselines

Research

Multimedia

The screenshot displays a Jupyter Notebook titled 'C6' from the 'Fundamentals of Music Processing' course. The notebook content is annotated with red boxes:

- Mathematics**: A definition of a discrete-time novelty function $\Delta : \mathbb{Z} \rightarrow \mathbb{R}$ and its Fourier coefficients $\mathcal{F}(n, \omega) := \sum_{m \in \mathbb{Z}} \Delta(m) \bar{w}(m-n) \exp(-2\pi i \omega m)$.
- Theory**: A section titled 'Example: Shostakovich' showing a musical score snippet.
- Music Example**: A video player showing an audio excerpt of a recording.
- Annotations**: A box highlighting the text 'an excerpt of a recording of ... ty Orch'.
- Audio**: A box highlighting the video player's progress bar (0:00 / 0:06).
- Links**: A box highlighting the text 'with a spectral-based novelty function resampled to F_s^{Δ} '.
- Python Code**: A code cell defining a function `compute_sinusoid_optimal`.
- Functions**: A box highlighting the function definition.
- Visualization**: A box highlighting a plot titled 'NOVELTY FUNCTION WITH DETECTED PEAKS'.
- Results**: A box highlighting a plot titled 'PLP function with detected peaks'.
- Evaluation**: A box highlighting the bottom plot's progress bar (0:00:00 / 27:03:12).
- Sonification**: A box highlighting the bottom plot's progress bar.

At the bottom of the notebook interface, there is a navigation bar with icons for different sections, numbered 0 through 8.

References

- Meinard Müller: Fundamentals of Music Processing – Using Python and Jupyter Notebooks. 2nd Edition, Springer, 2021.
<https://www.springer.com/gp/book/9783030698072>
- Meinard Müller and Frank Zalkow: libfmp: A Python Package for Fundamentals of Music Processing. Journal of Open Source Software (JOSS), 6(63): 1–5, 2021.
<https://joss.theoj.org/papers/10.21105/joss.03326>
- Meinard Müller: An Educational Guide Through the FMP Notebooks for Teaching and Learning Fundamentals of Music Processing. Signals, 2(2): 245–285, 2021.
<https://www.mdpi.com/2624-6120/2/2/18>
- Meinard Müller and Frank Zalkow: FMP Notebooks: Educational Material for Teaching and Learning Fundamentals of Music Processing. Proc. International Society for Music Information Retrieval Conference (ISMIR): 573–580, 2019.
<https://zenodo.org/record/3527872#.YOhEQOgzaUk>
- Meinard Müller, Brian McFee, and Katherine Kinnaid: Interactive Learning of Signal Processing Through Music: Making Fourier Analysis Concrete for Students. IEEE Signal Processing Magazine, 38(3): 73–84, 2021.
<https://ieeexplore.ieee.org/document/9418542>

Resources (Group Meinard Müller)

- FMP Notebooks:

<https://www.audiolabs-erlangen.de/FMP>

- libfmp:

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

- synctoolbox:

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

- libtsm:

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

- Preparation Course Python (PCP) Notebooks:

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

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

Resources

- librosa:
<https://librosa.org/>
- madmom:
<https://github.com/CPJKU/madmom>
- Essentia Python tutorial:
https://essentia.upf.edu/essentia_python_tutorial.html
- mirdata:
<https://github.com/mir-dataset-loaders/mirdata>
- open-unmix:
<https://github.com/sigsep/open-unmix-pytorch>
- Open Source Tools & Data for Music Source Separation:
<https://source-separation.github.io/tutorial/landing.html>

