Signal Processing Methods for Music Synchronization, Audio Matching, and Source Separation

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Sebastian Ewert

aus

Werl

Bonn, März 2012

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn

1. Gutachter:	Prof. Dr. Meinard Müller
2. Gutachter:	Prof. Dr. Michael Clausen
Tag der Promotion:	

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Statement under Oath

I confirm under oath that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Bonn, ______________________________ ______________________________
(Datum / Date) (Unterschrift / Signature)

Abstract

The field of music information retrieval (MIR) aims at developing techniques and tools for organizing, understanding, and searching multimodal information in large music collections in a robust, efficient and intelligent manner. In this context, this thesis presents novel, content-based methods for music synchronization, audio matching, and source separation.

In general, *music synchronization* denotes a procedure which, for a given position in one representation of a piece of music, determines the corresponding position within another representation. Here, the thesis presents three complementary synchronization approaches, which improve upon previous methods in terms of robustness, reliability, and accuracy. The first approach employs a late-fusion strategy based on multiple, conceptually different alignment techniques to identify those music passages that allow for reliable alignment results. The second approach is based on the idea of employing musical structure analysis methods in the context of synchronization to derive reliable synchronization results even in the presence of structural differences between the versions to be aligned. Finally, the third approach employs several complementary strategies for increasing the accuracy and time resolution of synchronization results.

Given a short query audio clip, the goal of *audio matching* is to automatically retrieve all musically similar excerpts in different versions and arrangements of the same underlying piece of music. In this context, chroma-based audio features are a well-established tool as they possess a high degree of invariance to variations in timbre. This thesis describes a novel procedure for making chroma features even more robust to changes in timbre while keeping their discriminative power. Here, the idea is to identify and discard timbre-related information using techniques inspired by the well-known MFCC features, which are usually employed in speech processing.

Given a monaural music recording, the goal of *source separation* is to extract musically meaningful sound sources corresponding, for example, to a melody, an instrument, or a drum track from the recording. To facilitate this complex task, one can exploit additional information provided by a musical score. Based on this idea, this thesis presents two novel, conceptually different approaches to source separation. Using score information provided by a given MIDI file, the first approach employs a parametric model to describe a given audio recording of a piece of music. The resulting model is then used to extract sound sources as specified by the score. As a computationally less demanding and easier to implement alternative, the second approach employs the additional score information to guide a decomposition based on non-negative matrix factorization (NMF).

Keywords: Audio processing, music information retrieval, music synchronization, alignment, audio matching, source separation, non-negative matrix factorization, parametric models.

Acknowledgements

This thesis has been carried out at the department of computer science III, University of Bonn, under the supervision of Meinard Müller (Saarland University and Max-Planck-Institut Informatik). During the first two years I have been funded by the German Research Foundation (DFG CL 64/6-1) working on the project Armada*: Automatisierte Erschließung von Musikdokumenten unter Ausnutzung verschiedener Darstellungsformen.*

Many people supported me during the preparation of this thesis, either through helpful discussions, valuable advice, assistance of any kind, hard work or simply trust. First of all, I am indebted to Meinard Müller and Michael Clausen who essentially shaped my understanding of good science. Thank you for the motivating and inspiring time. I would also like to thank my current and former colleagues at the University of Bonn and the Max-Planck-Institut Informatik in Saarbrücken for helpful discussions and the pleasant working atmosphere: Andreas Baak, Rolf Bardeli, David Damm, Christian Fremery, Harald Grohganz, Peter Grosche, Thomas Helten, Nanzhu Jiang, Verena Konz, Frank Kurth, and Verena Thomas. Furthermore, I would like to express my gratitude to our technical support group, in particular Thomas Fuchs, Peter Lachart, and Walter Steiner, and our team of secretaries, in particular Martina Doelp and Heidi Georges-Hecking. I am also thankful to Sabine Budde and Ellen Fries for taking care of all organizational issues during my stays in Saarbrücken. Last but not least, I would like to thank my family, Manfred and Simona Ewert and Anna Bezgubenko, for their support, love, and patience with me.

Contents

Chapter 1

Introduction

During the last decade, worldwide digitization efforts have resulted in a tremendous growth of digital music collections, which comprise music-related documents of various types and formats. In particular for Western classical music, a piece of music is often associated with multiple audio recordings, several editions of sheet music, or various symbolic score representations (MusicXML, Lilypond, MIDI). Each type of representation describes a piece of music from a very specific perspective. As a visual representation, sheet music specifies the most important musical parameters of a piece in a compact and human readable form. This way, sheet music serves as a basis for musicians to create a performance and for musicologists to study a piece in terms of harmonic, rhythmic, and formal aspects. Computer-readable symbolic score representations are typically employed in music production systems to control the way synthesizers and samplers create sound. Finally, audio recordings describe physical properties of sound and capture the aural impression of a listener during a musical performance.

A major goal in the field of *music information retrieval* (MIR) is to develop techniques that allow users to conveniently access, explore, and experience music in all its different facets. In particular, as discussed in [119], combining different types of representations allows for novel possibilities to interact with the music. For example, the regular playback of a CD recording can be enriched by automatically presenting the corresponding score for the underlying piece of music. Additionally highlighting the current audio playback position in the score discloses the semantic link between both representations and allows for navigating in the recording in an intuitive and visual way. Furthermore, combined with a Google-like search engine, the user can further explore the entire music collection. For example, after selecting a few bars in a score or a short passage from a CD recording, the system would present a ranked list of passages from the collection, which are similar to the query in terms of harmonic, rhythmic, or timbral aspects. Moreover, an interface might allow the user to intuitively select arbitrary note groups in the score, which are automatically emphasized or attenuated in the recording in real-time. This way, the user can easily identify and focus on the constituent parts of a piece of music by highlighting notes corresponding to an instrument, a specific motif, a musical voice, or the left or the right hand of a piano piece. One goal of this thesis is to introduce or improve techniques underlying such a future multimodal music player. Key challenges are to organize, understand, and structure the vast amount of content in modern digital music collections and to identify semantic relationships across the various music representations and formats. In this context, the thesis presents fully automated content-based methods which extract structured information directly from the raw input data without the need for any manual intervention. For the application scenario described above, three types of methods are of particular importance: methods for music synchronization, audio matching, and source separation.

Music synchronization denotes a procedure which, for a given position in one representation of a piece of music, determines the corresponding position within another representation. In computing such alignments, recent approaches assume that the versions to be aligned correspond to each other with respect to their global and local structure. However, in real-world scenarios, this assumption is often violated [48, 120]. For example, for a popular song there often exist various structurally different album, radio, or extended versions. Or, in classical music, different recordings of the same piece may exhibit omissions of repetitions or significant differences in solo cadenzas, in ornamentation, or in the interpretation of trills and arpeggios. In this thesis, two novel approaches are presented that allow for automatically identifying the reliable parts of alignment results. Instead of relying on one single strategy, the idea of the first approach is to employ a late-fusion method that combines several types of conceptually different alignment strategies within an extensible framework. Looking for consistencies and inconsistencies across the synchronization results, the method automatically classifies the alignments locally as reliable or critical. Focusing on structural differences between the versions to be aligned, the idea of the second approach is to perform a single structure analysis for both versions simultaneously. Such a joint structure analysis reveals the repetitions within and across the two versions and discloses the reliable alignment parts.

After identifying the reliable alignment parts, one typically employs classical music synchronization techniques to align corresponding passages on a finer temporal level. In this context, strategies based on chroma features have turned out to yield robust alignment results, see [40] for a discussion. Such chroma-based approaches typically yield a reasonable synchronization quality that suffices for music browsing and retrieval applications. However, the alignment may not be accurate enough to capture fine nuances in tempo and articulation as needed in applications such as performance analysis or audio editing. Other synchronization strategies yield a higher accuracy for certain classes of music by incorporating onset information, but suffer from a high computational complexity and a lack of robustness. This thesis introduces several strategies on various conceptual levels to increase the time resolution and quality of the synchronization result without sacrificing robustness and efficiency. First, novel audio features are introduced that combine the temporal accuracy of onset features with the robustness of chroma features. Then, it is shown how these features can be used within an efficient and robust multiscale synchronization framework.

A second group of methods considered in this thesis is often subsumed under the notion of *audio matching*. Based on the query-by-example paradigm, these methods allow for creating Google-like search services for music documents. More exactly, given a short query clip, the goal of audio matching is to automatically retrieve all excerpts from all recordings within a given music collection that musically correspond to the query. In this context, chroma-based features have turned out to be a powerful mid-level representation. In particular, their main strength lies in their robustness to variations in timbre and instrumentation as they appear in different interpretations, cover songs, and arrangements of a piece of music, see also [121]. This thesis describes a novel procedure that further enhances chroma features by significantly boosting the degree of timbre invariance without degrading the features' discriminative power. The underlying idea is based on the generally accepted observation that the lower mel-frequency cepstral coefficients (MFCCs) are closely related to timbre. Now, instead of focusing on the lower coefficients as done for many MIR applications, they are discarded in the proposed method such that only the upper coefficients remain. Furthermore, using a pitch scale instead of a mel scale allows for projecting the remaining coefficients onto the twelve chroma bins. As the thesis will show, the resulting CRP (chroma DCT-reduced log pitch) features outperform various state-of-the art chroma features in the context of audio matching and retrieval applications.

A third group of methods considered in this thesis are methods for the decomposition of a monaural audio recording into musically meaningful sound sources, a task often referred to as *source separation.* To extract sources corresponding to a melody, a bassline, or an instrument track, classical methods exploit specific spectral and temporal properties of the given target source. For example, a melody is often characterized by its dominance in dynamics, while a given instrument can typically be identified by its specific overtone energy distribution. To create acoustically appealing separation results, one has to consider the complex physical properties of the sources, their delicate interaction in polyphonic mixtures, as well as acoustic properties of the recording environment. To facilitate this difficult task one can employ additional note information provided by a musical score or a MIDI file to guide the separation process. Based on this concept, this thesis introduces two novel, conceptually different approaches for separating arbitrary note groups from polyphonic audio recordings. While both approaches leverage the synchronization techniques developed in this thesis to robustly align given score and audio representations, they build on fundamentally different signal models. More precisely, given a MIDI file (representing the score) and an audio recording (representing an interpretation) of a piece of music, the idea of the first approach is to parameterize the spectrogram of the audio recording by exploiting the MIDI information. Closely following [41,42], the underlying model is based on the concept of note-wise spectrograms each describing the part of a spectrogram that can be attributed to a given note event. After initializing the model with note events provided by the MIDI file, all model parameters are adapted using efficient numerical optimization techniques such that the model spectrogram approximates the audio spectrogram as accurately as possible. The resulting parameterization then allows for constructing an audio recording considering only the notes of the target source. The second approach is based on the non-negative matrix factorization (NMF) framework, which can be used to decompose a magnitude spectrogram into a set of template (column) vectors and activation (row) vectors in an unsupervised manner. To better control this decomposition, NMF has previously been extended using prior knowledge and parametric models. Following [44], this thesis presents such an extension that employs the available score information to guide the NMF learning and decomposition process. Opposed to previous methods, the main idea is to impose constraints on both the template as well as the activation side. As experiments show, using such double constraints results in musically meaningful decompositions similar to some parametric approaches, while being computationally less demanding and easier to implement. Furthermore, as this thesis will show, additional onset constraints can be incorporated in a straightforward manner without sacrificing robustness.

Besides the introduction of new computational approaches to score-informed source separation, one further goal of this thesis is to explore novel application scenarios based on these techniques. So far, most methods have been developed for the purpose of extracting individual instrument tracks from audio recordings. To further illustrate the potential of score-informed source separation techniques, this thesis investigates two novel use cases. As a first application, a novel voice or note equalizer is presented, which allows a user not only to emphasize or attenuate whole instrument tracks but also specific note groups played by different or the same instrument. Here, a group of notes might correspond, for example, to a motif, a voice, or a melody. Moreover, by incorporating these concepts into a previously proposed multimodal music player it is demonstrated how a user-friendly interface might be devised. Given an audio recording and a scanned score image for a piece of music, the interface allows a user to select a staff in the score in an intuitive way. The corresponding group of notes is then separated or enhanced in the audio recording in real-time. The goal is to enrich the overall listening experience, in particular for classical music where every musician interprets a piece differently. Here, highlighting central musical elements allows for perceiving these differences much more intensely. In a second use case, instead of aiming at producing acoustically appealing separation results, score-informed source separation techniques are employed for analysis purposes. Given a MIDI file and an audio recording for a piece of music, the task consists of estimating an intensity for each MIDI note event as occurring in the recording. On the one hand, this enables a user to visually analyze and compare different interpretations of a piece in terms of dynamics on a note-level. On the other hand, it allows for enriching a given score-like MIDI representation with performance-specific subtleties.

1.1 Contributions of this Thesis

Overall, the main contributions of this thesis can be summarized as follows:

- A novel partial music synchronization method based on late-fusion principles allowing for a reliable partial synchronization of music data across different versions and domains. Here, the main idea is to look for consistencies and inconsistencies across various alignments obtained from conceptually different synchronization strategies.

- A novel partial music synchronization method based on a strategy of performing a joint structural analysis to detect the repetitive structure within and across different versions of the same musical work. Here, a core component is a structure analysis procedure that can cope with relative tempo differences between repeating segments.

- Various refinement strategies for global music synchronization including a new procedure based on a novel class of onset-based audio features in combination with standard chroma features. The resulting approach significantly improves the synchronization accuracy while preserving the robustness and efficiency of previously described procedures.

- A new type of chroma feature, which shows a higher degree of robustness to changes in timbre than conventional chroma features. These novel CRP features significantly improve the performance in matching and classification applications, where invariance to instrumentation and tone color is required.

- A score-informed source separation approach employing a parametric model that describes the spectrogram of a given recorded performance as a sum of note-event spectrograms. Efficient numerical optimization techniques are used to adapt the model parameters for a given audio recording.

- An extended NMF variant that exploits available score information to guide the source separation process. Based on the idea of simultaneously constraining both the template vectors as well as the activations, the method yields results similar to some state-of-the-art parametric approaches while being computationally less demanding and easier to implement.

- Novel application scenarios demonstrating the applicability and potential of score-informed source separation techniques. Here, the thesis considers a novel voice equalizer as well as a note intensity analysis task in the context of performance analysis.

1.2 Structure of this Thesis

Overall, the thesis is subdivided into three parts corresponding to the three general MIR tasks outlined in the introduction. These parts are organized in such a way that concepts, methods and strategies developed in one part also lay important foundations for the subsequent ones. The first part relates to the audio matching and retrieval task. Here, we first describe chroma features in Chapter 2, which will be important throughout the entire thesis. Then, in Chapter 3, we introduce the novel procedure which allows for significantly boosting the timbre invariance of chroma features. In the second part of the thesis, we focus on music synchronization techniques. We start in Chapter 4 with an overview of standard alignment techniques, which underlie most previous synchronization approaches. After that, we describe in Chapter 5 the novel late-fusion based music synchronization approach and in Chapter 6 the new synchronization method based on music structure analysis. Then, in Chapter 7, we introduce various new strategies for enhancing the accuracy of methods for global music synchronization. In the third part of the thesis, we then turn to score-informed audio processing methods. We begin in Chapter 8 with a comprehensive overview of previously proposed methods. Then, in Chapter 9, we describe the score-informed source separation approach based on a parametric model. The synchronization techniques developed in the second part are of particular importance in this context. We demonstrate the potential of the parametric model discussing two application scenarios: voice equalization and note intensity estimation. Finally, we introduce in Chapter 10 the novel score-informed NMF variant and conclude the thesis in Chapter 11 giving an outlook on future challenges.

1.3 Publications Related To This Thesis

Parts of this thesis have been previously published. In the following, publications by the author related to this thesis are listed in chronological order.

[40] Sebastian Ewert and Meinard Müller. Refinement strategies for music synchronization. In *Proceedings of the International Symposium on Computer Music Modeling and Retrieval (CMMR)*, volume 5493 of *Lecture Notes in Computer Science*, pages 147–165, Copenhagen, Denmark, 2008.

[120] Meinard Müller and Sebastian Ewert. Joint structure analysis with applications to music annotation and synchronization. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 389–394, Philadelphia, Pennsylvania, USA, 2008.

[123] Meinard Müller, Sebastian Ewert, and Sebastian Kreuzer. Making chroma features more robust to timbre changes. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009.

[47] Sebastian Ewert, Meinard Müller, and Peter Grosche. High resolution audio synchronization using chroma onset features. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009.

[49] Sebastian Ewert, Meinard Müller, Daniel Müllensiefen, Michael Clausen, and Geraint Wiggins. Case study "Beatles Songs" – What can be learned from unreliable music alignments? In Eleanor Selfridge-Field, Frans Wiering, and Geraint A. Wiggins, editors, *Knowledge representation for intelligent music processing*, number 09051 in Dagstuhl Seminar Proceedings. Schloss Dagstuhl - Leibniz Center for Informatics, Germany, 2009.

[46] Sebastian Ewert, Meinard Müller, and Roger B. Dannenberg. Towards reliable partial music alignments using multiple synchronization strategies. In *Proceedings of the International Workshop on Adaptive Multimedia Retrieval (AMR), Lecture Notes in Computer Science (LNCS)*, volume 6535, pages 35–48, Madrid, Spain, 2009.

[121] Meinard Müller and Sebastian Ewert. Towards timbre-invariant audio features for harmony-based music. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):649–662, 2010.

[45] Sebastian Ewert, Meinard Müller, and Michael Clausen. Score-informed audio parametrization. International Society for Music Information Retrieval (ISMIR) latebreaking session, 2010. Available online: `http://www.ismir.net`.

[41] Sebastian Ewert and Meinard Müller. Estimating note intensities in music recordings. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 385–388, Prague, Czech Republic, 2011.

[42] Sebastian Ewert and Meinard Müller. Score-informed voice separation for piano recordings. In *Proceedings of the International Society for Music Information Retrieval Conference(ISMIR)*, pages 245–250, Miami, USA, 2011.

[122] Meinard Müller and Sebastian Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *Proceedings of the International Society for Music Information Retrieval Conference(ISMIR)*, pages 215–220, Miami, USA, 2011.

[44] Sebastian Ewert and Meinard Müller. Using score-informed constraints for NMF-based source separation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Kyoto, Japan, 2012.

[43] Sebastian Ewert and Meinard Müller. Score informed source separation. In Meinard Müller, Masataka Goto, and Markus Schedl, editors, *Multimodal Music Processing (to appear)*, Dagstuhl Follow-Ups. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2012.

[48] Sebastian Ewert, Meinard Müller, Verena Konz, Daniel Müllensiefen, and Geraint Wiggins. Towards cross-domain harmonic analysis of music. *IEEE Transactions on Multimedia (to appear)*, 2012.

The journal article [121] significantly extends the findings from the conference paper [123] and gives a more detailed discussion of the influence of various parameters. Furthermore, the journal article [48] describes the methods presented in [46] in more detail including new experiments to demonstrate the role of the individual parameters. Publication [40] extends the synchronization approach presented in [47] by introducing efficient multiscale implementations and further refinement strategies based on interpolation techniques. Publication [49] presents a case study illustrating the challenges for and potential of partial synchronization techniques as published in [48] and [120].

Additional Publications

The following publications by the author are also related to the subject of the thesis but are not further considered in the following.

[125] Meinard Müller, Verena Konz, Andi Scharfstein, Sebastian Ewert, and Michael Clausen. Towards automated extraction of tempo parameters from expressive music recordings. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 69–74, Kobe, Japan, 2009.

[52] Christian Fremerey, Michael Clausen, Sebastian Ewert, and Meinard Müller. Sheet music-audio identification. In *Proceedings of the International Society for Music Information Retrieval Conference(ISMIR)*, pages 645–650, Kobe, Japan, 2009.

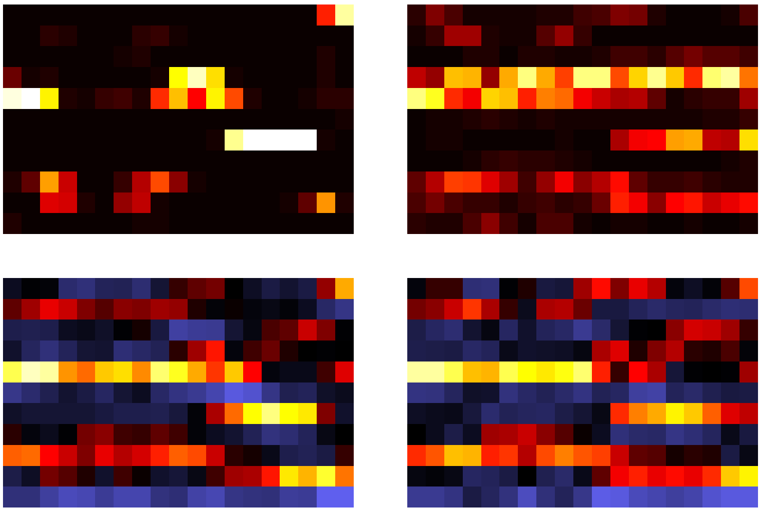
[92] Verena Konz, Meinard Müller, and Sebastian Ewert. A multi-perspective evaluation framework for chord recognition. In *Proceedings of the International Society for Music Information Retrieval Conference(ISMIR)*, pages 9–14, Utrecht, The Netherlands, 2010.

[119] Meinard Müller, Michael Clausen, Verena Konz, Sebastian Ewert, and Christian Fremerey. A multimodal way of experiencing and exploring music. *Interdisciplinary Science Reviews (ISR)*, 35(2):138–153, 2010.

[24] David Damm, Harald Grohganz, Frank Kurth, Sebastian Ewert, and Michael Clausen. SyncTS: Automatic synchronization of speech and text documents. In *Proceedings of the AES International Conference Semantic Audio*, pages 98–107, Ilmenau, Germany, 2011.

Part I

Feature Design



Chapter 2

Chroma Features

It is a well-known phenomenon that human perception of pitch is periodic in the sense that two pitches are perceived as similar in "color" if they differ by an octave. Based on this observation, a pitch can be separated into two components, which are referred to as *tone height* and *chroma*, see [169]. Assuming the equal-tempered scale, the chromas correspond to the set $\{\mathrm{C}, \mathrm{C}^\sharp, \mathrm{D}, \ldots, \mathrm{B}\}$ that consists of the twelve pitch spelling attributes[1] as used in Western music notation. Thus, a chroma feature is represented by a 12-dimensional vector $x = (x(1), x(2), \ldots, x(12))^\top$, where $x(1)$ corresponds to chroma C, $x(2)$ to chroma $\mathrm{C}^\sharp$, and so on. In the feature extraction step, a given audio signal is converted into a sequence of chroma features each expressing how the short-time energy of the signal is spread over the twelve chroma bands. Identifying pitches that differ by an octave, chroma features show a high degree of robustness to variations in timbre and closely correlate to the musical aspect of harmony. This is the reason why chroma-based audio features, sometimes also referred to as pitch class profiles, are a well-established tool for processing and analyzing music data [5, 60, 116]. For example, basically every chord recognition procedure relies on some kind of chroma representation [7, 16, 57, 70, 110, 139, 140, 168, 183]. Also, chroma features have become the de facto standard for tasks such as audio structure analysis [13, 14, 26, 136, 144, 145] as well as music synchronization and alignment [75, 82, 116, 131], which will be covered in more detail in Part II of this thesis. Furthermore, chroma features have turned out to be a powerful mid-level feature representation in content-based audio retrieval tasks such as cover song identification [38, 164] or audio matching [95, 128].

There exist many approaches for the computation of chroma-based audio features. For example, the conversion of an audio recording into a chroma representation (or chromagram) may be performed by using a short-time Fourier transform in combination with binning strategies [5], by using a constant-Q transform [10], or by employing a suitable multirate filter bank [116]. Furthermore, the properties of chroma features can be significantly changed by introducing suitable pre- and post-processing steps modifying spectral, temporal, and dynamical aspects. This chapter gives an overview of several commonly used chroma variants while discussing the role of the most important parameters that can be used to modify the features' characteristics. On overview of the entire feature process-

[1] Note that in the equal-tempered scale different pitch spellings such as $\mathrm{C}^\sharp$ and $\mathrm{D}^\flat$ refer to the same chroma.

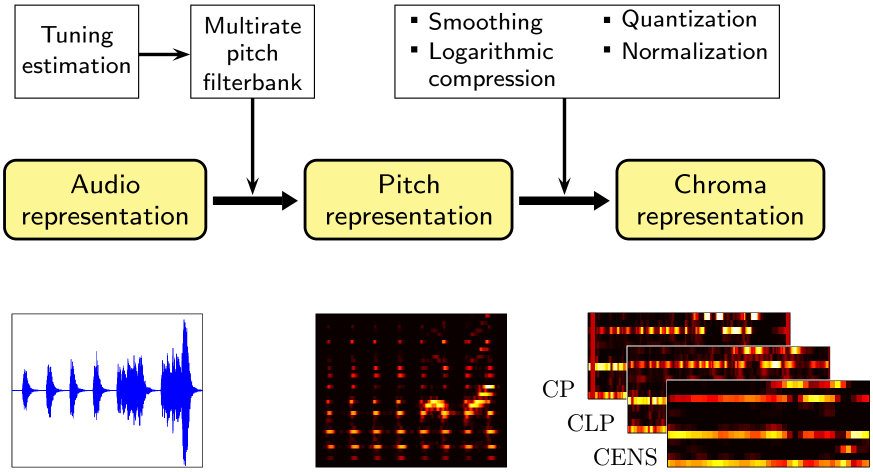


Figure 2.1. Overview of the feature extraction pipeline.

ing pipeline is given in Figure 2.1. The individual feature representations are illustrated in Figure 2.3 using an audio recording of the first six measures of Op. 100, No. 2 by Friedrich Burgmüller. MATLAB implementations of the chroma features discussed in this chapter as well as of several other chroma variants have been made available to the public in form of a chroma toolbox, which has recently been released under a GNU-GPL license, see also Appendix A.

2.1 Pitch Representation

As basis for the chroma feature extraction, one first decomposes a given audio signal into 88 frequency bands with center frequencies corresponding to the pitches $A0$ to $C8$ (MIDI pitches $p = 21$ to $p = 108$). As mentioned above, this decomposition can be derived in different ways, for example, by suitably pooling Fourier coefficients obtained from one or several spectrograms [5,38,60], by using a constant-Q transform [10] or multirate filter bank techniques [116, 128]. To obtain a sufficient spectral resolution for the lower frequencies, one either needs a low sampling rate or a large temporal window. In this thesis, a constant Q multirate filter bank is employed using a sampling rate of 22050 Hz for high pitches, 4410 Hz for medium pitches, and 882 Hz for low pitches, see [116] for details. Each filter is implemented using an eighth-order elliptic filter with 1 dB passband ripple and 50 dB rejection in the stopband. To separate the adjacent notes, a Q factor of $Q = 25$ is used and a transition band that has half the width of the passband. The employed pitch filters possess a relatively wide passband, while still properly separating adjacent notes thanks to sharp cutoffs in the transition bands, see Figure 2.2a. Actually, the pitch filters are robust to deviations of up to ±25 cents[2] from the respective note's center frequency. However, these properties are at the expense of large phase distortions and group delays. Since this thesis considers only audio signals that are entirely known prior to computations, one

[2] The *cent* is a logarithmic unit to measure musical intervals. The semitone interval of the equally-tempered scale equals 100 cents.

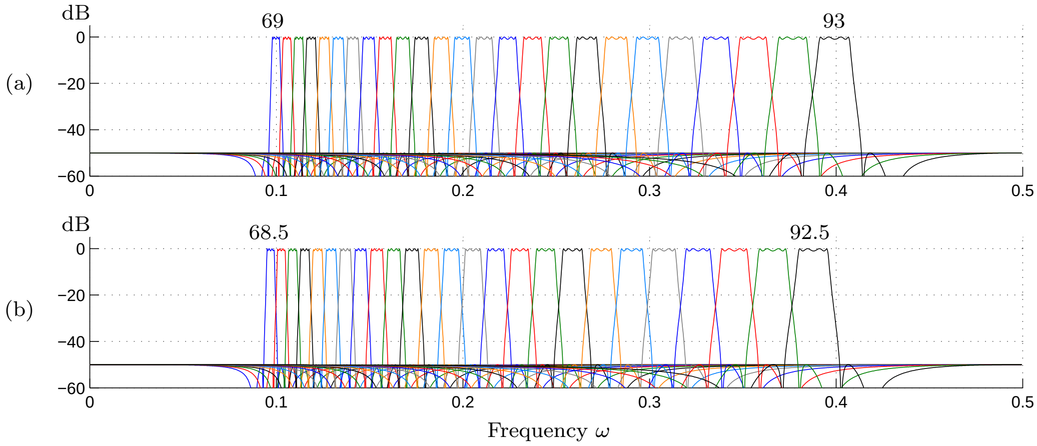


Figure 2.2. Magnitude responses in dB for some of the pitch filters of the multirate pitch filter bank used for the chroma computation. **(a)** Filters corresponding to MIDI pitches $p \in [69 : 93]$ (with respect to the sampling rate 4410 Hz). **(b)** Filters after applying a shift of half a semitone ($\sigma = \frac{1}{2}$).

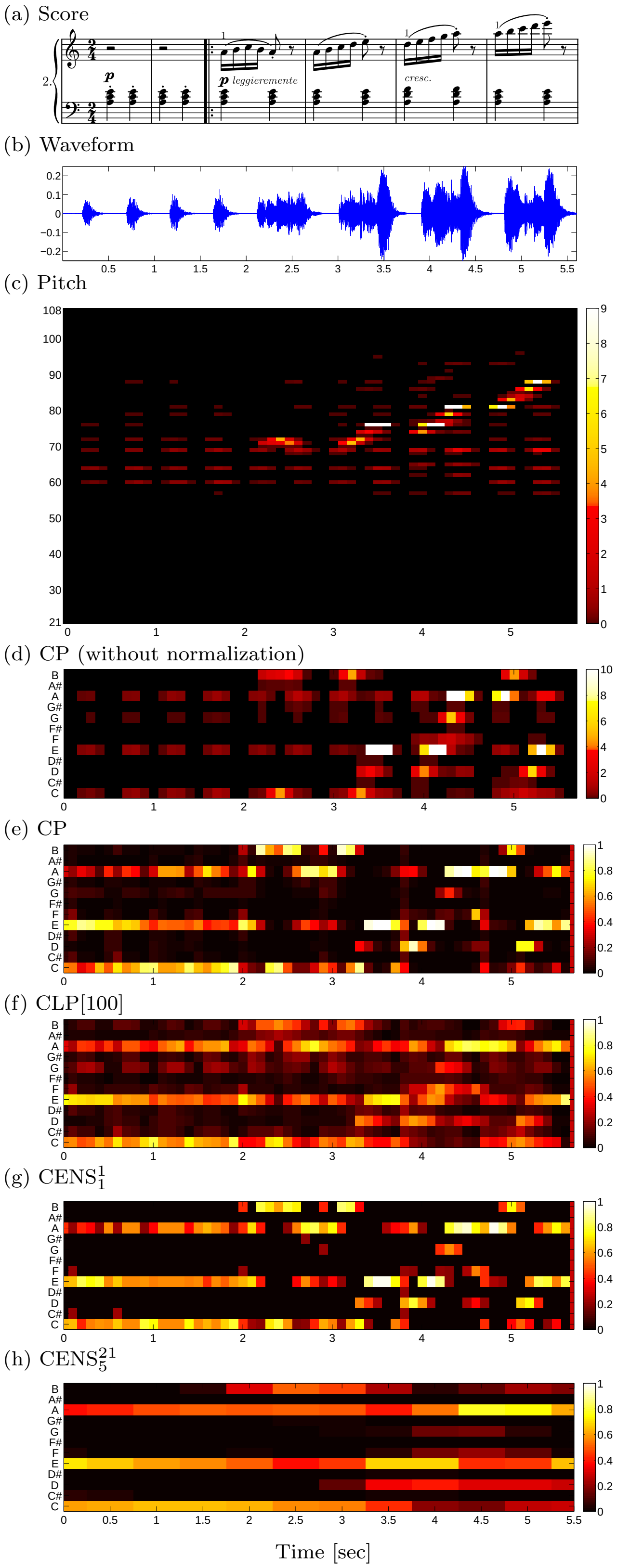
can apply the following trick: After filtering in the forward direction, the filtered signal is reversed and run back through the filter. The resulting output signal has precisely zero phase distortion and a magnitude modified by the square of the filter's magnitude response. Further details may be found in standard text books on digital signal processing such as [149]. In the next step, for each of the 88 pitch subbands, the short-time mean-square power (i. e., the samples of each subband output are squared) is computed using a rectangular window of a fixed length and an overlap of 50 %. For example, using a window length of 200 milliseconds leads to a feature rate of 10 Hz (10 features per second). The resulting Pitch features measure the short-time energy content of the audio signal within each pitch subband, see Figure 2.3c for an illustration and [116] for further details.

2.2 Tuning

To account for the global tuning of a recording, one needs to suitably shift the center frequencies of the subband-filters of the multirate filter bank. To this end, one can compute an average spectrogram vector and derive an estimate for the tuning deviation by simulating the filterbank shifts using weighted binning techniques similar to [60]. The methods in this thesis employ six different multirate filter banks corresponding to a shift of $\sigma \in \left\{0, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}\right\}$ semitones, respectively, see Figure 2.2 for an illustration. From these filter banks, the most suitable one is chosen according to the estimated tuning deviation.

2.3 CP Feature

From the pitch representation, one obtains a chroma representation simply by adding up the corresponding values that belong to the same chroma. For example, to compute



(a) Score

(b) Waveform

(c) Pitch

(d) CP (without normalization)

(e) CP

(f) CLP[100]

(g) CENS_1^1

(h) CENS_5^{21}

Figure 2.3. Score and various feature representations for an audio recording of the first four measures of Op. 100, No. 2 by Friedrich Burgmüller.

the entry corresponding to chroma C, one adds up values corresponding to the musical pitches C1, C2, ..., C8 (MIDI pitches $p = 24, 36, \ldots, 108$). For each window, this yields a 12-dimensional vector $x = (x(1), x(2), \ldots, x(12))^\top$, where $x(1)$ corresponds to chroma C, $x(2)$ to chroma C$^\sharp$, and so on. The resulting features are referred to as *Chroma-Pitch* and are denoted by CP in the following, see Figure 2.3d for our Burgmüller example. Recall that the timbre of a sound strongly relates to the energy distribution in the harmonics. Therefore, due to the octave equivalence, chroma features show a high degree of robustness to variations in timbre. Furthermore, chroma features account for the close octave relationship in both melody and harmony as prominent in Western music and are ideal for the analysis of music that is characterized by a prominent harmonic progression, see [5].

2.4 Normalization

To achieve invariance in dynamics, one can normalize the features with respect to some suitable norm. While there are various norms to choose from, this thesis only considers the ℓ^p-norm defined by $\|x\|_p := \left(\sum_{i=1}^{12} |x(i)|^p\right)^{1/p}$ for a given chroma vector $x = (x(1), x(2), \ldots, x(12))^\top$ and a natural number $p \in \mathbb{N}$. To avoid random energy distributions occurring during passages of very low energy (e. g., passages of silence before the actual start of the recording or during long pauses), a chroma vector x is replaced by the uniform vector of norm one in case $\|x\|_p$ falls below a certain threshold. Note that the case $p = 2$ yields the Euclidean norm and the case $p = 1$ the Manhattan norm. If not specified otherwise, all chroma vectors in this thesis are normalized with respect to the Euclidean norm, see also Figure 2.3e.

2.5 CLP Features

To account for the logarithmic sensation of sound intensity [199], one often applies a logarithmic amplitude compression when computing audio features [143]. For example, similar to μ-law compression [79], each energy values e of the pitch representation is replaced by the value $\log(\eta \cdot e + 1)$, where η is a suitable positive constant. Then, the chroma values are computed as explained in Section 2.3. The resulting features, which depend on the compression parameter η, are referred to as *Chroma-Log-Pitch* and are denoted by CLP[η] in the following, see Figure 2.3f. Note that a similar flattening effect can be achieved by spectral whitening techniques, where the pitch subbands are normalized according to short-time variances in the subbands [60, 90].

2.6 CENS Features

Adding a further degree of abstraction by considering short-time statistics over energy distributions within the chroma bands, one obtains CENS (Chroma Energy Normalized Statistics) features, which constitute a family of scalable and robust audio features. These features have turned out to be very useful in audio matching and retrieval applications [95, 128]. In computing CENS features, each chroma vector is first normalized with respect to the ℓ^1-norm thus expressing relative energy distribution. Then, a quantization is applied based on suitably chosen thresholds. Here, choosing thresholds in a logarithmic fashion introduces some kind of logarithmic compression as above, see [128] for details. In a subsequent step, the features are further smoothed over a window of length $w \in \mathbb{N}$ and downsampled by a factor of d, see Section 2.7. The resulting features are normalized with respect to the ℓ^2-norm and denoted by CENS^w_d, see also Figure 2.3g and Figure 2.3h for illustrations.

2.7 Smoothing

As already mentioned in Section 2.6, one can further process the various chroma variants by applying smoothing and downsampling operations. For example, subsequent vectors of a feature sequences can be averaged using a sliding window of size w (given in frames) and then downsampled by a factor d. Starting with CP, CLP[η], and CENS, the resulting features are denoted by CP^w_d, $\mathrm{CLP}[\eta]^w_d$, and CENS^w_d, respectively. Even though being a simple strategy, smoothing can have a significant impact on the features' behavior within a music analysis tasks. For example, as reported in [128], the temporal blurring of CENS features makes audio matching more robust to local tempo variations. Furthermore, using the parameters w and d, one obtains a computationally inexpensive procedure to simulate tempo changes on the feature level. To illustrate this, consider the following example. Suppose, we start with a chroma representation having a feature rate of 10 Hz. Then using $w = 41$ and $d = 10$, one obtains one chroma vector per second, each covering roughly 4100 ms of the original audio signal. Now, using $w = 53$ (instead of $w = 41$) and $d = 13$ (instead of $d = 10$) results in a temporally scaled version of the features sequence simulating a tempo change of $10/13 \approx 0.77$. Such tempo change strategies have been applied successfully in the context of audio indexing [95].

2.8 Chroma Toolbox

The feature extraction components described in Sections 2.1-2.7 as well as several extensions have been implemented as part of a MATLAB chroma toolbox, which can be freely obtained from the website [17] under a GNU-GPL license. Appendix A gives an overview of the main functions along with the most important parameters.

Chapter 3

Timbre-Invariant Audio Features

One main goal of content-based music analysis and retrieval is to reveal semantically meaningful relationships between different music excerpts contained in a given data collection. Here, the notion of similarity used to compare different music excerpts is a delicate issue and largely depends on the respective application. In particular, for detecting harmony-based relations, chroma features as described in Chapter 2 have turned out to be a powerful mid-level representation for comparing and relating music data in various realizations and formats [5, 60, 116]. Fusing pitches that differ by an octave, chroma features show a high degree of robustness to variations in timbre and are well-suited for the analysis of Western music which is characterized by a prominent harmonic progression [5]. This chapter presents a novel method for making chroma features even more robust to changes in timbre. At the same time, their discriminative power – as needed in matching applications – is preserved. Here, the general idea is to discard timbre-related information similar to that expressed by certain mel-frequency cepstral coefficients (MFCCs). More precisely, recall that the mel-frequency cepstrum is obtained by taking a discrete cosine transform (DCT) of a log power spectrum on the logarithmic mel scale [30]. A generally accepted observation is that the lower MFCCs are closely related to the aspect of timbre [3, 178]. Therefore, intuitively spoken, one should achieve some degree of timbre-invariance when discarding exactly this information. As one main contribution, this chapter shows how this idea can be combined with the concept of chroma features by first replacing the nonlinear mel scale with a nonlinear pitch scale. Then, a DCT is applied to the logarithmized pitch representation to obtain *pitch-frequency cepstral coefficients* (PFCCs). After keeping only the upper coefficients and applying an inverse DCT, the resulting pitch vectors are finally projected onto twelve-dimensional chroma vectors. These vectors are referred to as CRP (c̲hroma DCT-r̲educed log p̲itch) features. The technical details of this procedure are described in Section 3.1. The gist of these novel features is illustrated by Figure 3.3, which shows two different types of chromagrams for two musically related audio excerpts that differ significantly in instrumentation. Note that the two chromagrams based on conventional chroma features (Figure 3.3a and b) look rather different, whereas the chromagrams based on the novel CRP features (Figure 3.3c and d) are quite similar thus indicating the boost towards timbre invariance.

CRP audio features constitute a valuable tool in all retrieval, matching, and classification applications where one is interested in blending out musical details related to timbre and instrumentation. In particular, this chapter demonstrates the potential of this concept by means of the audio matching scenario based on the query-by-example paradigm as introduced in [128]. Given a short query audio clip, the goal is to automatically retrieve all excerpts from all recordings within a given audio collection that musically correspond to the query. Here, one typically has to cope with variations in timbre and instrumentation as they appear in different interpretations, cover songs, and arrangements of a piece of music. As another contribution, the audio matching procedure is used to systematically evaluate the matching and separation capabilities of different types of audio features. Among others, various quality measures are introduced that indicate how well semantically correct matches are separated from spurious matches. As it turns out, these quality measures are also good indicators for the degree of timbre invariance exhibited by the respective feature type. In a series of experiments the novel CRP features are systematically compared with previously suggested chroma features, which also reveals the role of the various parameters and measures involved in the feature computation. Among others, this chapter investigates the role of the feature rate and the number of coefficients to be pruned as well as the influence of amplitude compression and spectral whitening. Furthermore, we discuss two different cost measures used to compare the resulting features including the binary shift measure introduced in [164]. As one main result, it is shown that the proposed procedure is conceptually different to previous feature enhancement strategies in the sense that it yields a significant boost towards timbre invariance independent of a particular choice of parameters and measures. To allow for a straightforward reproduction of these experiments, a reference implementation of CRP features has been made available in a MATLAB toolbox, see also Appendix A.

As a final contribution, the DCT-based reduction step in the proposed procedure is analyzed in detail. As it turns out, the most dominant of the upper PFCCs capture interpretable pitch periodicities, whereas PFCCs surrounding the dominant ones account for different phases. We show that a reduction based on only a few relevant DCT basis vectors along with suitable phase-shifted duplicates results in a similar feature enhancement as using the entire range of upper DCT basis vectors, see Section 3.4. This observation reveals the musical meaning of certain pitch-frequency cepstral coefficients.

Section 3.1 starts with the main contribution by introducing the novel CRP features and by describing in detail the involved signal processing steps. Then, Section 3.2 gives a short description of the audio matching application, which also lays the foundation for various quality measures used to compare and evaluate the different feature types. Next, in Section 3.3, a series of experiments is presented discussing the influence of various parameters on the quality of the resulting CRP features. Finally, in Section 3.4, the principles are investigated that underlie the boost towards timbre invariance for harmony-based Western classical music. Conclusions and prospects on future work are given in Section 3.5. A discussion of related work and further references are given in the respective sections.

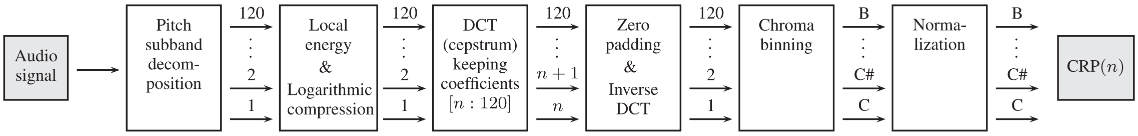

Audio signal
Pitch subband decom-position
120
2
1
Local energy & Logarithmic compression
120
2
1
DCT (cepstrum) keeping coefficients [n : 120]
120
$n+1$
n
Zero padding & Inverse DCT
120
2
1
Chroma binning
B
C#
C
Norma-lization
B
C#
C
CRP(n)

Figure 3.1. Overview of the steps in the computation of the CRP (chroma DCT-reduced log pitch) features.

3.1 Feature Design

This section describes the proposed enhancement procedure that allows for increasing the robustness of chroma features to changes in timbre and instrumentation while keeping their discriminative power. To this end, we combine and modify various techniques known from the design of chroma features and mel-frequency cepstral coefficients (MFCCs) in a novel way. In Section 3.1.1, we review MFCCs and then, in Section 3.1.2, go into the technical details of the proposed procedure. Finally, Section 3.1.3 reports on a first baseline experiment conducted on systematically generated audio material.

3.1.1 MFCC features

In some sense, MFCC features, which are closely related to the aspect of timbre, can be considered as kind of complementary to chroma features. Originally, MFCCs were developed for speech processing applications [30, 150] and have then found their way into the music domain [103], where they have been used for various music analysis tasks including genre classification [182] and musical instrument recognition [39]. In most implementations, the mel-frequency cepstrum is obtained in the following way. First, the power spectrum of the signal is computed using a short-time Fourier transform. Then, to account for properties of the human auditory system, the resulting coefficients are pooled into 20 to 40 nonlinearly spaced frequency bins along the perceptually motivated mel frequency scale [150]. Similarly, a musically motivated frequency scale is used in [108]. Finally, after taking the logarithm on the bin values, a discrete cosine transform (DCT) is applied to yield the MFCCs. A generally accepted observation is that the lower MFCCs are closely related to the aspect of timbre [3, 178]. Therefore, intuitively spoken, one should achieve some degree of timbre-invariance when discarding exactly this information. This is the basic idea of the enhancement procedure to be described next.

3.1.2 CRP Feature Computation

We now describe in detail all steps needed to compute the novel CRP audio features. For an overview of these steps, see Figure 3.1. Instead of using a mel-frequency scale, CRP features are based on a pitch-frequency scale. To this end, we use the filter bank approach described in Section 2.1 to compute a pitch representation of the audio signal, which can be summarized as follows. First, we decompose the audio signal into 88 frequency bands with center frequencies corresponding to the MIDI pitches $p = 21$ to $p = 108$. Next, we compute the short-time mean-square power (local energy) for each of the 88 squared subbands using

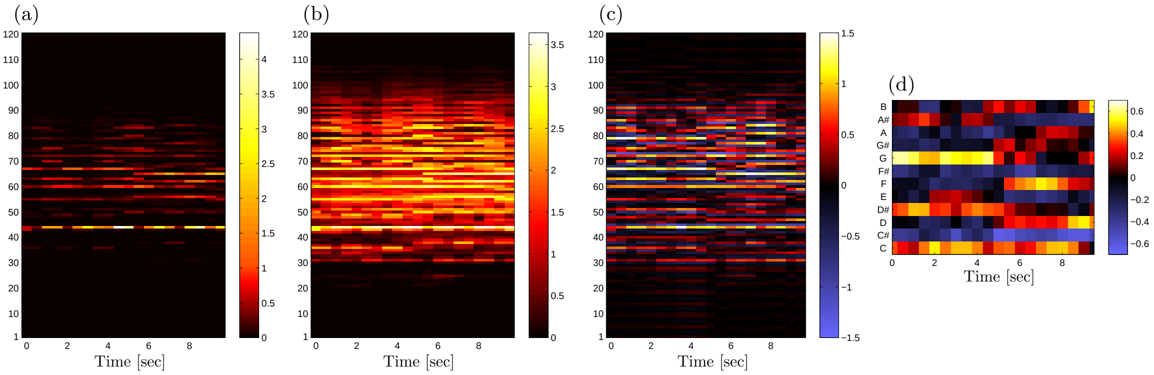


Figure 3.2. Various feature representations of the passage E_3 (trombone part in the Yablonsky recording of the Shostakovich Waltz) illustrating the steps in the CRP feature computation. **(a)** Pitch representation. **(b)** Pitch representation after the logarithmic compression. **(c)** Pitch representation after the DCT reduction step keeping coefficients [55 : 120]. **(d)** CRP(55) features.

a rectangular window of a fixed length and an overlap of 50%. For example, a window length corresponding to 1 second leads to a feature rate of 2 Hz (2 features per second). In Section 3.3.5, we discuss the role of the feature rate in more detail. As a result, we obtain a sequence of 88 dimensional feature vectors where the entries correspond to MIDI pitches $p = 21$ to $p = 108$. For later usage, we extend each such vector by suitably adding zeros (20 at the beginning and 12 at the end) to obtain a 120 dimensional feature vector where the entries now correspond to MIDI pitches $p = 1$ to $p = 120$. For an illustration, see Figure 3.2a.

To obtain a conventional chroma representation or chromagram (Chroma-Pitch), one adds up the corresponding values of the pitch representation that belong to the same chroma yielding a 12-dimensional vector for each analysis window, see Figure 3.3 for an illustration. For the proposed audio features, we further process the pitch representation before doing the chroma binning. The steps are similar to the ones in the computation of MFCCs. First, the pitch representation is logarithmized, see Figure 3.2(b). Here, we replace each entry e by the value $\log(\eta \cdot e + 1)$, where η is a suitable positive constant. Similar to CLP[η] features, such a logarithmic compression is conducted to account for the logarithmic sensation of sound intensity [103, 199] and was also used in a similar way in [91]. The role of the parameter η, which is set to $\eta = 100$ in most of the following experiments, is discussed in Section 3.3.4.

Next, we apply a discrete cosine transform (DCT) to each of the 120-dimensional logarithmized pitch vectors resulting in 120 coefficients, which are referred to as *pitch-frequency cepstral coefficients* (PFCCs). The PFCCs have a similar interpretation as the MFCCs. In particular, the lower coefficients are related to timbre as observed by various researchers, see, e. g., [3, 178]. Now, our goal of achieving timbre-invariance is the exact opposite of the goal of capturing timbre. Therefore, we discard the information given by the lower $n-1$ PFCCs for a parameter $n \in [1 : 120]$ by setting them to zero while leaving the upper PFCCs unchanged. Each resulting 120-dimensional vector is then transformed by the inverse DCT to yield an enhanced 120-dimensional pitch vector, see Figure 3.2(c). The role of the parameter n is discussed in Section 3.3.3. Furthermore, in Section 3.4, we analyze

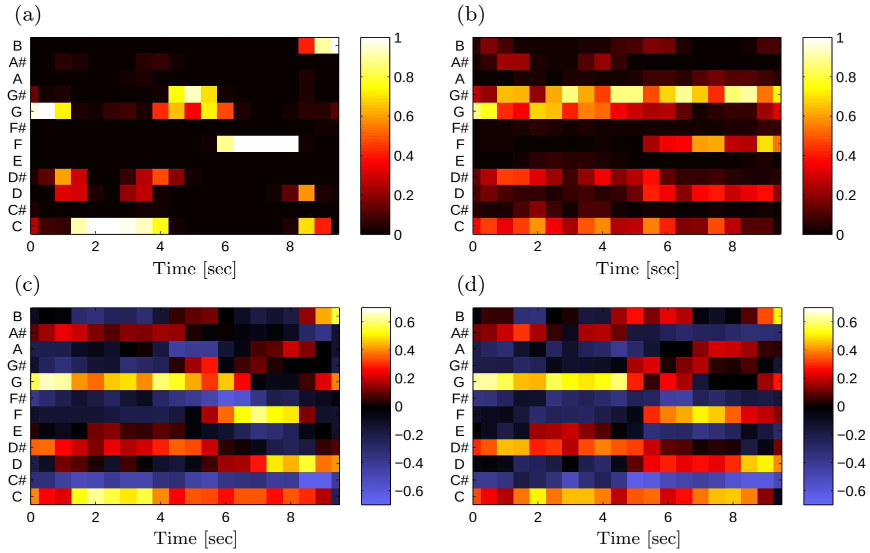


Figure 3.3. Various chromagrams of the passages E_1 (clarinet) and E_3 (trombone) in the Yablonsky recording of the Shostakovich Waltz. **(a)/(b)**: Conventional chromagram of E_1/E_3. **(c)/(d)**: CRP(55) chromagram of E_1/E_3. All chroma vectors are normalized w.r.t. the Euclidean norm.

the reduction step in detail and derive a musically meaningful explanation responsible for the final enhancement.

In the last stage, the entries of each enhanced pitch vector are projected onto the twelve chroma bins to yield a 12-dimensional chroma vector. Finally, the chroma vectors are normalized with respect to the Euclidean norm to have unit length. The resulting audio features are referred to as CRP(n) (chroma DCT-reduced log pitch) features, see Figure 3.2(d). A reference MATLAB implementation of CRP features is available as part of the chroma toolbox, which has been made freely available on a website[1], see also Appendix A.

In the experiments to be described, it is shown that the resulting CRP features have indeed gained a significant amount of robustness to changes in timbre and instrumentation. As a first illustrative example, we consider the second Waltz of the Jazz Suite No. 2 by Shostakovich, which also serves as running example in the subsequent sections. The theme of this piece occurs four times played in four different instrumentations (clarinet, strings, trombone, tutti). Furthermore, there are also significant differences between the four themes with respect to secondary voices. In the considered recording of this piece by Yablonsky, the four occurrences of the theme are referred to as E_1 (5-26), E_2 (39-59), E_3 (129-149), and E_4 (160-180), where the brackets indicate the start and end times in seconds of the respective passage. Figure 3.3(a) and (b) show conventional chromagrams of the passages E_1 (theme played by clarinet) and E_3 (theme played by trombone), respectively. Note that the two chromagrams strongly deviate from each other due to large differences in instrumentation and voicing. Contrary, the corresponding two CRP(55) chromagrams as shown in (c) and (d) of Figure 3.3 coincide to a much larger degree.

[1] www.mpi-inf.mpg.de/resources/MIR/chromatoolbox/

3.1.3 Baseline Experiments on Chord Chroma Classes

To illustrate the boost of robustness achieved by the CRP features, we next discuss a baseline experiment conducted on systematically generated audio material. For the moment, we fix certain parameters using a feature rate of 2 Hz, setting $\eta = 100$ in the logarithmic compression, and considering only the case $n = 55$ in CRP(n). Section 3.3 gives a detailed analysis of these parameters reporting on extensive experiments based on real audio material.

We compare the resulting CRP(55) features with various publicly available implementations of state-of-the-art chroma feature types including three implementations (Chroma-IF, Chroma-P, Chroma-E) by Ellis[2], one implementation (Chroma-QM) developed at the Centre for Digital Music, Queen Mary, University of London[3], as well as one implementation (Chroma-MIR) contained in the MIR toolbox[4]. These chroma variants are based on similar concepts as the ones described in Chapter 2 but differ in some details. The Chroma-E implementation is based on a Gaussian weighted pooling of magnitude spectrum coefficients. Its extension, Chroma-P, additionally implements a simple spectral peak picking to reduce spectral noise. In the more complex Chroma-IF variant, spectral regions of uniform instantaneous frequency are estimated to separate tonal components from noise. The instantaneous frequency information is also used to account for tuning differences. The fourth implementation, Chroma-MIR, is derived from the magnitude spectrum using a decibel scale. The Chroma-QM implementation uses the magnitude of the constant-Q transform as described in [10]. For further details and applications of the various chroma variants, see [10, 12, 38, 60, 97]. Furthermore, we use the conventional chroma features (Chroma-Pitch) obtained from the pitch representation as described in Section 2.3. For all chroma implementations, similar parameters settings and rates were used, see Appendix B. Furthermore, all chroma features were normalized with respect to the Euclidean norm.

To indicate the degree of timbre-invariance of the various chroma implementations, the following baseline experiment was conducted. First, a MIDI file was created containing all possible single pitches (1-chords), duads (2-chords) and triads (3-chords) within a fixed octave. This resulted in $12 + \binom{12}{2} + \binom{12}{3} = 220$ chords. The MIDI file was then synthesized in 24 different ways using eight different instruments each playing the file in three different octaves. Here, the software Cubase was used in combination with a high quality sample library with a size of more than 50GB. Fixing a specific feature type, each of the resulting 24 audio files was converted into a chromagram. Next, for each of the 220 chords a class was formed consisting of 24 chroma vectors—one representative chroma vector from each of the 24 realizations of the respective chord. The classes are referred to as chord chroma classes. The distance between two normalized chroma vectors was computed using $1 - \langle \cdot, \cdot \rangle$ (also referred to as cosine distance). Note that the entries of CRP features may be negative, so that the distance between two normalized CRP vectors lies in the range $[0, 2]$.

Now, disregarding timbre and dynamics, any two chroma vectors within a chord chroma class are considered as similar, whereas two chroma vectors from different classes are considered as dissimilar. To measure the degree of timbre invariance of a given feature

[2] http://www.ee.columbia.edu/~dpwe/resources/matlab/chroma-ansyn/

[3] http://www.vamp-plugins.org

[4] http://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/

Feature type	μ_{I}	σ_{I}	μ_{O}	σ_{O}	δ
Chroma-IF	0.299	0.193	0.654	0.188	0.457
Chroma-P	0.174	0.133	0.464	0.160	0.374
Chroma-E	0.168	0.129	0.452	0.159	0.373
Chroma-MIR	0.107	0.078	0.268	0.137	0.398
Chroma-QM	0.124	0.098	0.396	0.146	0.313
Chroma-Pitch	0.232	0.194	0.749	0.161	0.309
CRP(55)	**0.078**	**0.069**	**1.002**	**0.131**	**0.077**

Table 3.1. Quality of several feature types in the experiments on chord chroma classes.

type, we compute the distances between any two chroma vectors that belong to the same chord chroma class. Let μ_{I} be the mean and σ_{I} the standard deviation over the resulting $220 \cdot \binom{24}{2}$ distances. Note that μ_{I} should be small in the case that the feature type has a high degree of timbre invariance. Similarly, let μ_{O} be the mean and σ_{O} the standard deviation over the distances of any two chroma vectors from different chord chroma classes. Note that μ_{O} should be large to indicate a high discriminative power of a feature type. Finally, the quotient $\delta := \mu_{\mathrm{I}}/\mu_{\mathrm{O}}$ can be formed which expresses the within-class distance μ_{I} relative to the across-class distance μ_{O}. Note that a small value of δ is desirable in view of our evaluation.

Table 3.1 shows the values μ_{I}, μ_{O}, and δ for various feature types. Note that for CRP(55) features the within-class distance ($\mu_{\mathrm{I}} = 0.078$) is much smaller while the across-class distance ($\mu_{\mathrm{O}} = 1.002$) is much larger than for all other conventional chroma types. This clearly demonstrates that CRP features differ fundamentally from previous chroma types. As shown in Section 3.3, the boost of timbre invariance can also be observed when using real audio material.

3.2 Application: Audio Matching

The identification and retrieval of semantically related music data is of major concern in the field of music information retrieval. Loosely speaking, one can distinguish between two different scenarios. In the *global* matching scenario one compares and relates entire instances (on the document level) of a piece of music such as entire audio recordings or MIDI files. For example, in *cover song identification* the goal is to identify all performances of the same piece by different artists with varying interpretations, styles, instrumentation, and tempos [38, 164]. In the *local* matching scenario one compares and relates different subsegments contained in the same or in different instances of a piece. For example, in *audio matching* the goal is to automatically retrieve all passages (subsegments) from all audio documents that musically correspond to a given query excerpt [128]. Of course, the two scenarios seamlessly merge into each other. For example, Serrà et al. [164] use a local matching strategy for global document retrieval. The quality of the respective matching procedure depends on various factors including the underlying feature representation, the cost measure used to compare two feature vectors, as well as the distance function used to relate the various feature sequences.

In this chapter, we study the behavior of the proposed feature enhancement strategy within the audio matching scenario. In Section 3.2.1, we define the distance function that underlies the matching procedure and provides a powerful tool for compactly assessing the matching capability of the used feature type. Then, in Section 3.2.2, we derive various quality measures from the distance function, which turn out to be good indicators for the degree of timbre invariance exhibited by the respective feature type.

3.2.1 Distance Function

Let Q be a query a clip (typically a short audio excerpt) and let $(D_1, D_2, \ldots, D_N)$ be a collection of database documents (typically a large number of audio recordings). To simplify things, we assume that we have only one large database document D by concatenating $D_1, \ldots, D_N$, where we keep track of document boundaries in a supplemental data structure. The goal of audio matching is to find all subsegments or passages within D that are similar to Q.

The first step of the audio matching procedure is to transform the query and the database document into suitable feature sequences $X = (X(1), X(2), \ldots, X(K))$ with $X(k) \in \mathcal{F}$ for $k \in [1:K] := \{1, 2, \ldots, K\}$ and $Y = (Y(1), Y(2), \ldots, Y(L))$ with $Y(\ell) \in \mathcal{F}$ for $\ell \in [1:L]$, respectively. Here, $\mathcal{F}$ denotes the underlying feature space. For example, in the case of normalized chroma features one has $\mathcal{F} = [0,1]^{12}$. Furthermore, let $c : \mathcal{F} \times \mathcal{F} \to \mathbb{R}$ denote a *cost measure* on $\mathcal{F}$. If not stated otherwise, we revert to the cost measure $1 - \langle \cdot, \cdot \rangle$ (which is the cosine measure for normalized vectors). In Section 3.3, we also consider a binary shift measure similar to the one as introduced in [164]. As basis for the matching procedure, we use a distance function that locally compares the query sequence X with subsequences of the database sequence Y. More precisely, we define a distance function $\Delta : [1:L] \to \mathbb{R} \cup \{\infty\}$ between X and Y using dynamic time warping (DTW):

$$\Delta(\ell) := \frac{1}{K} \min_{a \in [1:\ell]} \Big(\mathrm{DTW}\big(X\,,\, Y(a:\ell)\big) \Big), \tag{3.1}$$

where $Y(a:\ell)$ denotes the subsequence of Y starting at index a and ending at index $\ell \in [1:L]$. Furthermore, $\mathrm{DTW}(X, Y(a:\ell))$ denotes the DTW distance between X and $Y(a:\ell)$ with respect to the cost measure c, which will be explained in more detail in Part II. To avoid degenerations in the DTW alignment, we use the modified step size condition with step sizes $(2,1)$, $(1,2)$, and $(1,1)$ (instead of the classical step sizes $(1,0)$, $(0,1)$, and $(1,1)$). Note that the distance function Δ can be computed efficiently using dynamic programming.

The interpretation of Δ is as follows: a small value $\Delta(\ell)$ for some $\ell \in [1:L]$ indicates that the subsequence of Y starting at frame a_ℓ (with $a_\ell \in [1:\ell]$ denoting the minimizing index in (3.1)) and ending at frame ℓ is similar to X. To determine the best match between Q and D, one simply has to look for the index $\ell_0 \in [1:L]$ minimizing Δ. Then the best match is the audio clip corresponding to the feature subsequence $(Y(a_{\ell_0}), \ldots, Y(\ell_0))$. The value $\Delta(\ell_0)$ is also referred to as the *cost* of the match. To look for the second best match, we exclude a neighborhood around the index ℓ_0 from further consideration to avoid large overlaps with the best match. For the subsequent experiments, we exclude half the query length to the left and right by setting the corresponding Δ-values to ∞. To find

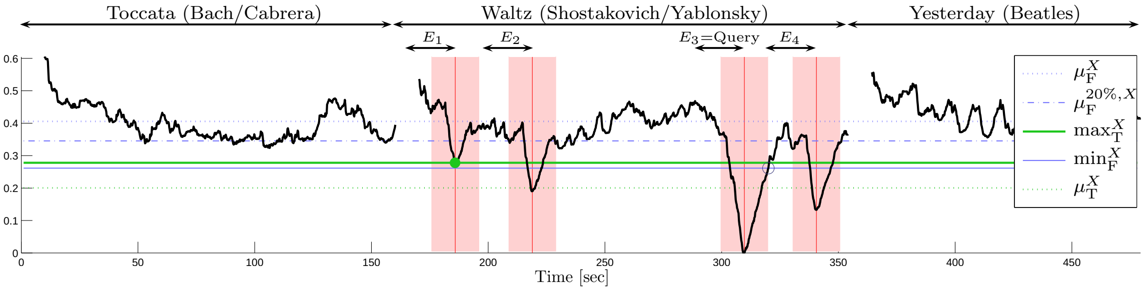


Figure 3.4. Distance function with respect to the query E_3 for a database sequence corresponding to audio recordings of three different pieces (Bach Toccata played by Cabrera, Shostakovich Waltz conducted by Yablonsky, Yesterday by the Beatles). Indices corresponding to the four true matches are indicated by the four vertical red lines. The false alarm region consists of all indices outside the neighborhoods that are indicated by light red. The various quality measures are indicated by the horizontal lines. The green dot and the blue circle indicate the positions in the distance function that correspond to $\max_{\mathrm{T}}^{X}$ and $\min_{\mathrm{F}}^{X}$, respectively.

subsequent matches, the above procedure is repeated until a certain number of matches is obtained or a specified distance threshold is exceeded. Note that the extracted matches can be naturally ranked according to their cost.

We illustrate the definition of Δ by means of our Shostakovich example introduced in Section 3.1.2. We consider three different database documents that refer to audio recordings of three different pieces (Bach Toccata played by Cabrera, Shostakovich Waltz conducted by Yablonsky, Yesterday by the Beatles). First, we transform the three audio recordings into suitable feature sequences, which are concatenated to form a single database feature sequence Y. Furthermore, using the passage E_3 (trombone) from the Yablonsky recording as query, we derive a query feature sequence X for the query E_3. Figure 3.4 shows the resulting distance function Δ. Within the three documents, there are four semantically correct matches, namely the passages E_1, E_2, E_3, and E_4 within the Waltz. Indeed, these four passages are revealed by four local minima of Δ. However, note that due to the above mentioned differences in timbre, some of these local minima are not well developed and have relatively large Δ-values such as the one corresponding to E_1. This is problematic as will be detailed in the next section. For example, iteratively extracting matches as described above, E_3, E_4, and E_2 appear as the top three matches. However, the next match is a false positive match (corresponding to the index 320 next to the right neighborhood boundary of E_3), before E_1 is identified as the fifth match.

3.2.2 Quality Measures

In view of the audio matching application, the following two properties of Δ are of crucial importance. First, the semantically correct matches (in the following referred to as *true matches*) should correspond to local minima of Δ close to zero thus avoiding false negatives. We capture this property by defining μ_{T}^{X} and $\max_{\mathrm{T}}^{X}$ to be the average and maximum of Δ, respectively, over all indices that correspond to the local minima of the true matches for a given query X. Second, Δ should be well above zero outside a neighborhood of the desired local minima thus avoiding false positives. Recall from Section 3.2.1 that we use half the query length to the left and right to define such a neighborhood. The region outside these neighborhoods is referred to as *false alarm region.* We then define μ_{F}^{X} and $\min_{\mathrm{F}}^{X}$ to be the average respective minimum of Δ over all indices within the false alarm region. For our Shostakovich example shown in Figure 3.4, these values are indicated by suitable horizontal lines. In order to separate the true matches from spurious matches, it is clear that μ_{T}^{X} and $\max_{\mathrm{T}}^{X}$ should be small whereas μ_{F}^{X} and $\min_{\mathrm{F}}^{X}$ should be large. We express these two properties within a single number, respectively, by defining the quotients $\alpha^X := \mu_{\mathrm{T}}^{X}/\mu_{\mathrm{F}}^{X}$ and $\gamma^X := \max_{\mathrm{T}}^{X} / \min_{\mathrm{F}}^{X}$.

In view of a good separability, α^X and γ^X should be close to zero. In the case $\gamma^X < 1$, all true matches appear as the top most matches. Contrary, $\gamma^X > 1$ indicates that at least one false positive match appears before all true matches are retrieved. Note that the quality measure γ^X is rather strict in the sense that one single outlier (either a true match of high cost or a spurious match of low cost in the false alarm region) may completely corrupt the value of γ^X. Contrary, the quality measure α^X is rather soft in the sense that despite of having a low value α^X one may obtain a large number of false positive matches. As a trade-off between the two quality measures α^X and γ^X, we introduce a third quality measure β^X. To this end, we sort the indices within the false alarm region by increasing cost and define $\mu_{\mathrm{F}}^{p\%,X}$ to be the average Δ only over the lower p% of the indices, $p \in [0, 100]$, see also Figure 3.4. Note that for $p = 100$, one simply obtains $\mu_{\mathrm{F}}^{p\%,X} = \mu_{\mathrm{F}}^{X}$. Finally, we define $\beta^X := \mu_{\mathrm{T}}^{X}/\mu_{\mathrm{F}}^{p\%,X}$. In the experiments, we used $p = 1$ considering only 1% of the indices within the false alarm region. Note that β^X is a much better measure for indicating possible false positive matches than α^X while being more robust to outliers than γ^X. In Section 3.3, we apply these quality measures on the basis of a carefully selected set of queries and a manually annotated collection of audio recordings in order to determine the degree of timbre invariance of various features types.

3.3 Experiments

Section 3.1.3 reported on a first baseline experiment using systematically generated audio material. This section presents a series of experiments based on real audio recordings to indicate how CRP features behave in comparison to previously introduced chroma features as well as to explore the role of various parameters. First, in Section 3.3.2, it is shown that CRP features outperform various publicly available state-of-the-art chroma features [12,38, 97] with regard to timbre invariance. Then, we discuss the dependency of the CRP features' quality on the number of coefficients to be pruned in the reduction step (Section 3.3.3), on the value of the constant used in the logarithmic compression (Section 3.3.4), and on the feature rate (Section 3.3.5). Only recently, Serrà et al. [164] have introduced a novel binary shift measure for comparing chroma features. In Section 3.3.6, we show that CRP features also yield significant quality improvements in combination with this novel cost measure. Finally, we investigate the effect of the CRP features on precision and recall values in the context of the audio matching application (Section 3.3.7). Altogether, the experiments show that the proposed enhancement strategy yields a significant boost towards timbre invariance independent of a particular choice of parameters and measures.

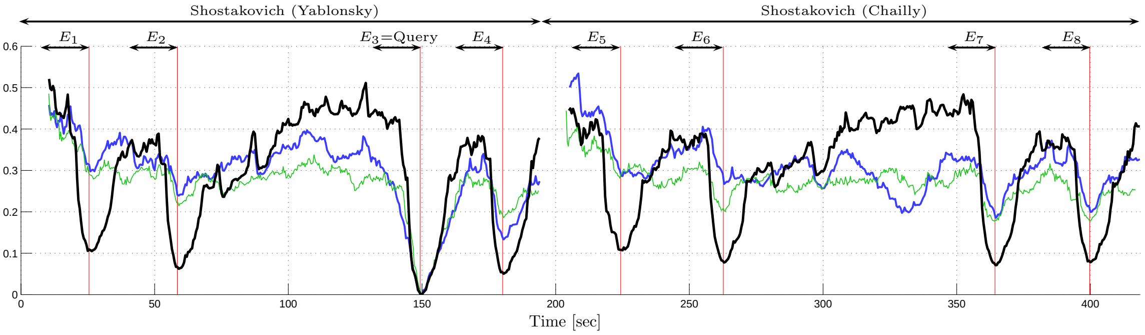


Figure 3.5. Several distance functions shown for two recordings (Yablonsky, Chailly) of the Shostakovich example using the excerpt E_3 as query. The following feature types were used: Chroma-IF (thin green), Chroma-Pitch (blue) and CRP(55) (bold black). For the query, there are 8 annotated excerpts (true matches).

3.3.1 Experimental Setup

For evaluating and comparing various types of chroma features, we employ a collection of audio recordings compiled by the author, which comprises harmony-based music of various genres. Here, the objective was to include music material that, on the one hand, contains a large number of harmonically related excerpts, which, on the other hand, reveal significant differences in timbre and instrumentation. For one thing, the collection contains pieces such as the Waltz by Shostakovich or the Bolero by Ravel, where a theme is repeated in different instrumentations. For another thing, for each piece there are at least two different versions such as different arrangements or cover songs. For example, on the classical music side, the collection contains an orchestra version as well as a piano version of the first movement of Beethoven's Fifth Symphony, Brahms' Hungarian dance No. 5, or Wagner's Prelude of the Meistersinger. On the popular music side, there are the original version and at least one cover song of pieces by the Beatles, Queen, Genesis, Indigo Girls, and Gloria Gaynor. Altogether, the collection consists of 32 recordings amounting to 166 minutes of music, see Appendix B.

A total of 101 audio excerpts with an average length of 30 seconds were carefully selected, which are used as queries in the following matching experiments. The data collection was then manually annotated by specifying all relevant matches (referred to as true matches, see Section 3.2.1) for each of the queries. At this point, it should be emphasized that the main object of these experiments is to assess the degree of timbre invariance and the discriminative power of the various chroma features. In other words, we are interested in evaluating the underlying features and use the matching procedure only for the purpose of comparing features. Therefore, we employ a controlled and manageable database with a clear notion of true matches, where the true matches represent various kinds of variations with regard to timbre and instrumentation. For each query X, we compute the values μ_{T}^X, μ_{F}^X, $\mu_{\mathrm{F}}^{1\%,X}$, $\min_{\mathrm{F}}^X$, and $\max_{\mathrm{T}}^X$ as well as the quality measures α^X, β^X, and γ^X using the entire collection as the database documents, see Section 3.2.2. Averaging over all 101 queries, we obtain the corresponding numbers denoted by μ_{T}, μ_{F}, $\mu_{\mathrm{F}}^{1\%}$, $\min_{\mathrm{F}}$, and $\max_{\mathrm{T}}$, as well as α, β, and γ. Note that α is not the quotient of μ_{T} and μ_{F}, but the average of the α^X. Analogously, this also holds for β and γ.

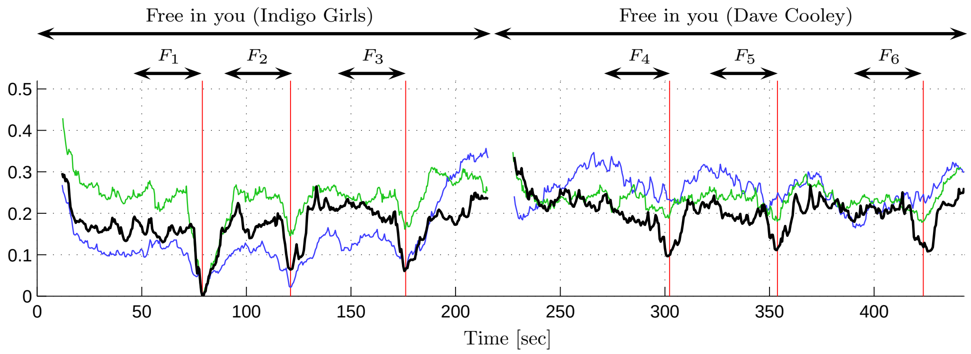


Figure 3.6. Different distance functions using the excerpt F_3 as query. Only the part of the database is shown that consists of two versions (original by Indigo Girls, cover by Dave Cooley) of the piece "Free in you". Altogether, there are 6 true matches denoted by F_1 to F_6. The following feature types were used: Chroma-IF (thin green), Chroma-Pitch (blue) and CRP(55) (bold black).

3.3.2 Comparison between Feature Types

We compare the CRP(n) features for various parameters $n \in [1 : 120]$ with various state-of-the-art chroma types using the same parameter settings as described in Section 3.1.3 (feature rate of 2 Hz, $\eta = 100$, feature vectors of Euclidean norm 1). Before giving a systematic evaluation, we illustrate the matching capability of different feature types by means of our Shostakovich example, see Section 3.1.2. The database contains two different recordings (Yablonsky, Chailly) of the Waltz with 8 annotated excerpts corresponding to the theme, where E_1 to E_4 denote the corresponding excerpts in the Yablonsky and E_5 to E_8 in the Chailly recording. Now, using E_3 (trombone) as query, one has eight true matches. Using conventional chroma features such as Chroma-IF or Chroma-Pitch most of the expected local minima are not significant or not even existing (e. g., E_5), see Figure 3.5. Now, using CRP(n) features, one obtains for all eight true matches (even for E_5) much more concise local minima, see the black curve of Figure 3.5. This demonstrates that the particular choice of a feature type has a significant impact on the final matching quality. A similar effect can be noticed in Figure 3.6, which shows the distance function for the song "Free in you" by the Indigo Girls and a cover version of the same piece by Dave Cooley. In the original version, the voice is accompanied by an acoustic guitar and some moderate percussion, whereas in the cover song there are additional voices, percussion is much more dominant, and the guitar is replaced by distorted electronic synthesizer effects. Also for this popular music example, using conventional chroma features (Chroma-IF, Chroma-Pitch) results in a distance function without well-defined local minima for the true matches (especially for the cover version). On the contrary, using CRP(n) features leads to local minima at the positions of the true matches that are clearly separated from the false alarm region.

Table 3.2 shows different quality measures for six types of conventional chroma features and for the novel CRP(n) features for selected parameters $n \in [1 : 120]$. For example, using the conventional chroma features Chroma-P, the average cost of the true matches is $\mu_\mathrm{T} = 0.042$, whereas the average distance in the false alarm region is $\mu_\mathrm{F} = 0.132$. The

	μ_{T}	μ_{F}	α	μ_{T}	$\mu_{\mathrm{F}}^{1\%}$	β	$\max_{\mathrm{T}}$	$\min_{\mathrm{F}}$	γ
Chroma-IF	0.150	0.313	0.487	0.150	0.198	0.767	0.177	0.162	1.098
Chroma-P	0.042	0.132	0.320	0.042	0.072	0.610	0.055	0.060	0.954
Chroma-E	0.045	0.133	0.346	0.045	0.075	0.642	0.059	0.062	0.979
Chroma-MIR	0.032	0.078	0.415	0.032	0.042	0.834	0.040	0.035	1.234
Chroma-QM	0.065	0.147	0.457	0.065	0.090	0.750	0.080	0.075	1.087
Chroma-Pitch	0.120	0.433	0.282	0.120	0.219	0.568	0.164	0.171	0.985
CRP(35)	0.110	0.671	0.167	0.110	0.287	0.397	0.147	0.223	0.691
CRP(55)	**0.092**	**0.626**	**0.150**	**0.092**	**0.244**	**0.388**	**0.124**	**0.187**	**0.693**
CRP(75)	0.076	0.550	0.143	0.076	0.172	0.459	0.106	0.136	0.840
CRP(95)	0.044	0.472	0.099	0.044	0.081	0.546	0.066	0.058	1.180

Table 3.2. Overview over the various quality measures for different types of chroma features (feature rate ≈ 2 Hz, $\eta = 100$).

average quotient amounts to $\alpha = 0.320$[5]. Other conventional chroma features exhibit a larger average cost for the true matches such as $\mu_{\mathrm{T}} = 0.120$ for Chroma-Pitch. However, in this case the average distance within the false alarm region also increases remarkably amounting to $\mu_{\mathrm{F}} = 0.433$. As a result, the average quotient of $\alpha = 0.282$ for Chroma-Pitch is lower thus expressing a higher discrimination capability than the one for Chroma-P. Now, looking at the quality measures for the novel CRP(n) features, one can recognize a significant improvement. For example, in the case $n = 55$ one obtains $\alpha = 0.150$, which is nearly half of $\alpha = 0.282$ obtained from Chroma-Pitch. In other words, the discrimination capability of CRP(55) features is nearly twice as good as in the case of Chroma-Pitch.

According to the measure α, the CRP(n) features seem to perform best for the parameter $n = 95$ among all selected parameters listed in Table 3.2. However, looking at the γ-measure, one obtains $\gamma = 1.180$ for $n = 95$, which is much worse than $\gamma = 0.693$ for $n = 55$. As already noted in Section 3.2.2, the α-measure does not warrant a clear separation between true matches and spurious matches. In contrast, the γ-measure yields an explicit separation distance, but may be corrupted by a single outlier. In the following, the main focus is on the β-measure using only the lower $p = 1\%$ of the indices in the false alarm region, which constitutes a suitable compromise between the α- and γ-measure, see Section 3.2.2.

With respect to the β-measure, the CRP(55) features with $\beta = 0.388$ perform best among all listed feature types. For the best conventional feature (Chroma-Pitch), one already has $\beta = 0.568$. The difference between CRP(35) and CRP(55) is not significant. Here, the parameter $n = 55$ may be preferable since less coefficients are needed to yield the same discriminative power. In the next section, we investigate the role of the reduction parameter $n \in [1:120]$ in more detail.

3.3.3 Dependency on DCT Reduction

In the last section, we have compared and discussed the discrimination capability of various conventional chroma features and of CRP(n) features for selected parameters $n \in [1:120]$.

[5] Recall that the values are obtained by averaging over all queries. The average quotient $\alpha = 0.320$ does not coincide with the quotient of the averages $\mu_{\mathrm{T}}/\mu_{\mathrm{F}} = 0.318$.

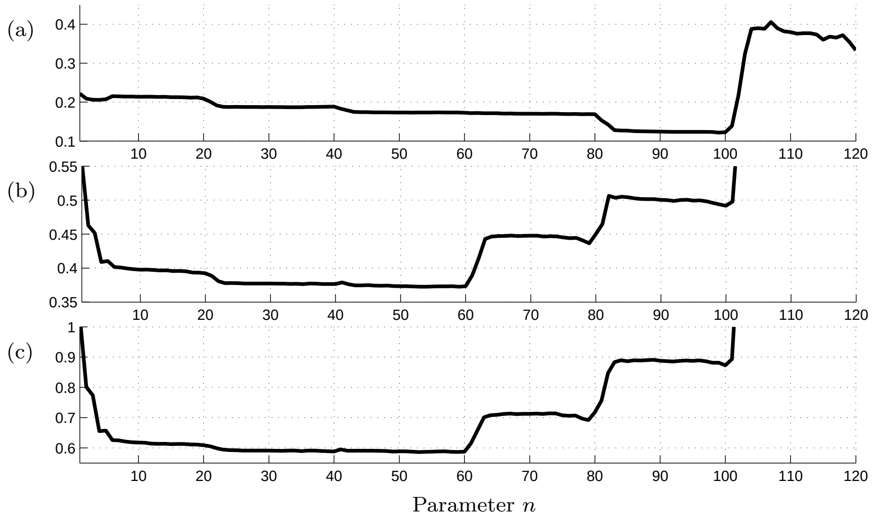


Figure 3.7. Influence of the parameter $n \in [1:120]$ (horizontal axis) on the performance measures **(a)** α, **(b)** β, and **(c)** γ. The range of each vertical axis has been limited to show more details of the relevant parts of the respective curve.

We now look closer at the role of this parameter, which determines the number of PFCCs to be pruned in the reduction step, see Section 3.1.2. To this end, we compute the quality measures α, β, and γ in dependence of $n \in [1:120]$. The resulting curves are shown in Figure 3.7. The curve for α may indicate that the discriminative power, in average, is optimal for parameters $n \in [83:99]$. However, as already discussed in Section 3.3.2, a low α-measure does not warrant a clear separation between true matches and spurious matches. More meaningful indicators are the β- and γ-measures. Here, the corresponding curves show that one obtains the best separation between true and spurious matches for parameters $n \in [23:59]$. In the following experiments, we use the parameter $n = 55$, which exhibits low values with respect to all three quality measures. Actually, in Section 3.4, we will discuss the musical meaning of a small number of dominant PFCCs, which also explains the “jumps” in the curves of Figure 3.7. These findings can then be used to further reduce the number of coefficients without a degradation of the discriminative power.

3.3.4 Dependency on Logarithmic Compression

It is a well-known fact that loudness is perceived in a logarithmic fashion [199]. Therefore, after a suitable decomposition of the audio signal, one often applies a logarithmic energy or amplitude compression. For example, such a step is involved in the computation of MFCCs [103] or in deriving onset signals as used for beat tracking and meter analysis [91]. In Section 3.1.2, we employed such a compression step after the subband decomposition replacing each entry e in the resulting pitch representation by the value $\log(\eta \cdot e + 1)$. To investigate the role of the positive constant η, we compute CRP(55) features using different constants η and derived the corresponding quality measures α, β, and γ. From these measures, which are listed in Table 3.3, we conclude that for any choice of η between 10 and 1000 one obtains features of a similar quality. In the following experiments, we therefore use the value $\eta = 100$. Similar findings are reported by Klapuri et al. [91].

η	μ_{T}	μ_{F}	α	μ_{T}	$\mu_{\mathrm{F}}^{1\%}$	β	$\max_{\mathrm{T}}$	$\min_{\mathrm{F}}$	γ
1	0.114	0.654	0.179	0.114	0.279	0.414	0.155	0.219	0.718
10	0.098	0.642	0.156	0.098	0.258	0.389	0.133	0.199	0.686
100	**0.092**	**0.626**	**0.150**	**0.092**	**0.244**	**0.388**	**0.124**	**0.187**	**0.693**
1000	0.089	0.606	0.151	0.089	0.234	0.397	0.120	0.180	0.705
10000	0.089	0.583	0.157	0.089	0.226	0.412	0.119	0.173	0.733
100000	0.087	0.557	0.162	0.087	0.216	0.425	0.116	0.166	0.758

Table 3.3. Influence of the parameter η used in the logarithmic compression on the quality of CRP(55) features.

	μ_{T}	μ_{F}	α	μ_{T}	$\mu_{\mathrm{F}}^{1\%}$	β	$\max_{\mathrm{T}}$	$\min_{\mathrm{F}}$	γ
10 Hz	0.121	0.642	0.191	0.121	0.315	0.384	0.159	0.253	0.633
5 Hz	0.107	0.636	0.171	0.107	0.291	0.374	0.143	0.229	0.637
2 Hz	**0.092**	**0.626**	**0.150**	**0.092**	**0.244**	**0.388**	**0.124**	**0.187**	**0.693**
1 Hz	0.083	0.619	0.138	0.083	0.197	0.443	0.112	0.147	0.827
0.5 Hz	0.074	0.625	0.123	0.074	0.154	0.513	0.106	0.110	1.061

Table 3.4. Influence of the feature rate on the quality of CRP(55) features.

Another approach used for dynamics compression is referred to as *spectral whitening*. The author implemented a version of the whitening procedure similar to [90] locally normalizing the pitch subbands obtained from the filterbank decomposition according to short-time variances of the subbands. Actually, this procedure is related to the logarithmic amplitude compression as both flatten (or whiten) the spectral energy distribution. Indeed, using spectral whitening instead of logarithmic compression did not have a significant impact on the various quality measures. Therefore, in the following, we only consider the algorithmically simpler logarithmic compression.

3.3.5 Dependency on Feature Rate

Next, we investigate the influence of the features rate on the final quality of CRP(n) features. Table 3.4 only reports on the results for the parameter $n = 55$ as other feature types were found to show a similar behavior. Recall from Section 3.1.2 that the final feature rate can be adjusted by modifying the size of the rectangular window used to compute the local energies in the pitch subbands. With respect to the β- and γ-measure, the resulting CRP features perform almost equally well for feature rates ranging from 10 Hz down to 2 Hz. However, further decreasing the feature rate results in noticeable degradations. For example, one has $\beta = 0.374$ for 5 Hz and a slightly higher $\beta = 0.388$ for 2 Hz, whereas the value significantly drops to $\beta = 0.443$ for 1 Hz. In the following experiments, we therefore revert to the feature rate of 2 Hz. In comparison to higher features rates, 2 Hz features not only possess a comparable quality, but also keep the data at a manageable size, thus making the subsequent steps in the matching procedure more efficient.

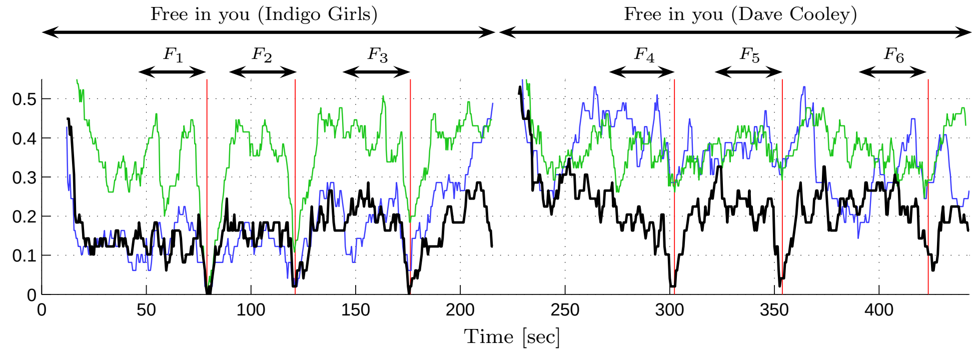


Figure 3.8. Different distance functions using the excerpt F_3 as query with respect to the binary shift measure c_{bs}, continuing the example from Figure 3.6. The following feature types were used: Chroma-IF (thin green), Chroma-Pitch (blue) and CRP(55) (bold black).

3.3.6 Dependency on Cost Measure

So far, we have used the cosine measure as cost measure to compare two chroma vectors. Only recently, Serrà et al. [164] have introduced a novel binary cost measure that only assumes two values. Basically, the idea is to consider all cyclically shifted versions of the two vectors to be compared [61]. Then, the two original chroma vectors are regarded as similar (binary cost measure assumes the value 0) if they best correlate without any shift relative to each other, otherwise they are regarded as dissimilar (binary cost measure assumes the value 1). This cost measure has turned out to be suitable in global matching tasks such as cover song identification [164]. A similar concept considering minimizing shift indices has been introduced in the context of music structure analysis, see [118].

In this section, we first define a binary cost measure similar to [164], which we refer to as *binary shift measure*. Then, we show that CRP features also yield significant quality improvements in combination with this novel cost measure. In the following, all chroma vectors are assumed to be normalized with respect to the Euclidean norm. As in Section 3.2.1, let $\mathcal{F} = [0,1]^{12}$ denote the feature space and $c : \mathcal{F} \times \mathcal{F} \to \mathbb{R}$ the cosine measure. We define the *cyclic shift* $\sigma : \mathcal{F} \to \mathcal{F}$ by

$$\sigma((v(1), v(2), \ldots, v(12))) := (v(2), \ldots, v(12), v(1))$$

for a chroma vector $v = (v(1), \ldots, v(12)) \in \mathcal{F}$. By iteratively applying σ, one obtains σ^i, $i \in \mathbb{N}_0$, where i is referred to as the *shift index*. Obviously, $\sigma^{12} = \sigma^0$ is the identity on $\mathcal{F}$. Now, when comparing two chroma vectors $v, w \in \mathcal{F}$, one first computes the minimizing shift index:

$$\mathrm{msi}(v, w) := \mathrm{argmin}_{i \in [0:11]} \Big(c(v, \sigma^i(w)) \Big).$$

Then, the binary shift measure $c_{\mathrm{bs}} : \mathcal{F} \times \mathcal{F} \to \{0, 1\}$ is defined by

$$c_{\mathrm{bs}}(v, w) := \begin{cases} 0 & \text{for } \mathrm{msi}(v, w) = 0, \\ 1 & \text{for } \mathrm{msi}(v, w) \neq 0. \end{cases}$$

We now repeat the computation of the quality measures α, β, and γ, where we use the binary shift measure c_{bs} instead of c. The result is shown in Table 3.5. There are several

	μ_{T}	μ_{F}	α	μ_{T}	$\mu_{\mathrm{F}}^{1\%}$	β	$\max_{\mathrm{T}}$	$\min_{\mathrm{F}}$	γ
Chroma-IF	0.171	0.528	0.327	0.171	0.283	0.598	0.233	0.215	1.081
Chroma-P	0.069	0.558	0.124	0.069	0.264	0.256	0.121	0.189	0.633
Chroma-E	0.087	0.552	0.159	0.087	0.275	0.312	0.142	0.203	0.689
Chroma-MIR	0.089	0.525	0.171	0.089	0.222	0.408	0.142	0.148	1.135
Chroma-QM	0.151	0.531	0.289	0.151	0.286	0.527	0.215	0.211	1.071
Chroma-Pitch	0.091	0.583	0.160	0.091	0.231	0.431	0.164	0.152	1.220
CRP(35)	0.011	0.572	0.019	0.011	0.126	0.115	0.029	0.060	1.710
CRP(55)	**0.011**	**0.570**	**0.019**	**0.011**	**0.123**	**0.097**	**0.031**	**0.057**	**1.246**
CRP(75)	0.029	0.574	0.052	0.029	0.140	0.227	0.076	0.061	3.371
CRP(95)	0.058	0.580	0.101	0.058	0.139	0.416	0.108	0.052	6.694

Table 3.5. Overview over the quality of various feature types employing the binary shift measure in the matching process.

interesting observations. First, it is striking that for all features types the α-measures with respect to c_{bs} are much lower than the ones with respect to c (compare Table 3.5 and Table 3.2). Second, also using c_{bs} as cost measure, the CRP(n) features by far outperform conventional chroma features with respect to the α- and β-measure. Again, the parameter $n = 55$ leads to very good overall results. For example, one has $\beta = 0.097$ for CRP(55) using c_{bs}, which yields the lowest β-value among the listed feature types.

At first sight surprisingly, the behavior of the γ-measure is quite different. Here, when using c_{bs} instead of c, conventional chroma features seem to be superior to CRP features. This can be explained as follows. Recall from Section 3.2.2 that the γ-measure suffers in the sense that a single outlier may completely corrupt the value of γ. Now, the binary shift measure c_{bs} assuming only the two values zero and one is a rather coarse measure compared to the cosine measure c. As a consequence, the distance function Δ typically decreases in regions that are harmonically related to the query (but it may even increase in regions that are harmonically unrelated to the query). On the positive side, this generally lowers the cost of true matches. On the negative side, this often produces a few (not necessarily many) false positive matches of quite low cost. These false positive matches corrupt the γ-measure, but do not have such a large effect on the β-measure. This phenomenon is also illustrated by Figure 3.8 (continuing the Indigo Girls example shown in Figure 3.6), where the binary shift measure c_{bs} is employed instead of c. Note that the c_{bs}-based distance functions yield a much better average separation (almost all true matches have a cost very close to zero) than the c-based counterparts. However, in particular in the original version, the c_{bs}-based distance function dangerously approaches zero at some positions within the false alarm regions.

3.3.7 Effect on Precision and Recall

To indicate the potential of the CRP features for music retrieval applications, we investigate the effect of the proposed enhancement strategy in terms of precision and recall values. In the following experiment, we use the queries and the manually annotated database from Section 3.3.1. The annotations constitute the ground truth on the exact positions of the true (relevant) matches for each query. Now, for a fixed feature type, we compute the

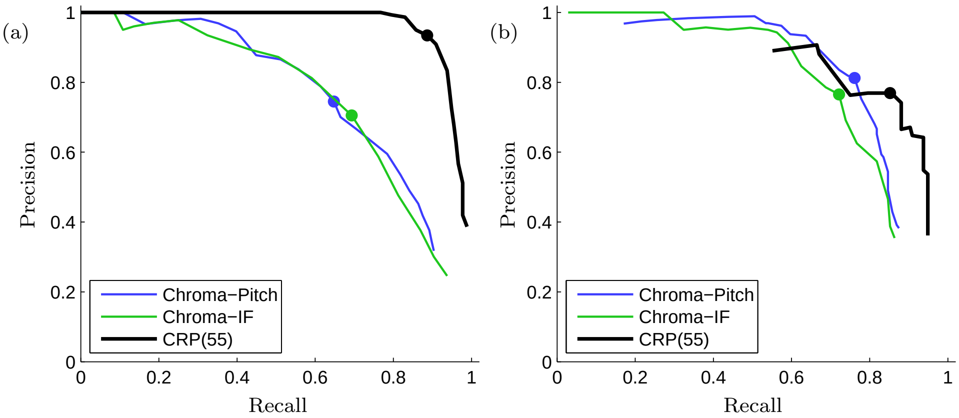


Figure 3.9. Quality of several feature types in terms of precision (vertical axis) and recall (horizontal axis) values. **(a)** PR-diagrams when using the cosine measure c. **(b)** PR-diagrams when using the binary shift measure c_{bs}. The dot within a PR-diagram indicates the respective maximal F-value $\mathrm{F}_{\max}$.

distance function Δ for each of the queries. Then, for a given positive distance threshold τ, we subsequently derive all matches having a cost below τ as described in Section 3.2.1. Using the ground truth information, we then compute the precision value P_τ and the recall value R_τ for the set of retrieved matches. From these values one obtains the F-measure $\mathrm{F}_\tau := \frac{2\cdot\mathrm{P}_\tau\mathrm{R}_\tau}{\mathrm{P}_\tau+\mathrm{R}_\tau}$. Starting with a threshold τ close to zero and increasing it little by little, one obtains a family of precision (P) and recall (R) values, which can be graphically visualized by a PR-diagram.

Figure 3.9(a) shows three representative PR-diagrams for two conventional chroma features (Chroma-IF, Chroma-Pitch) and for the CRP(55) features. As the diagrams indicate, one obtains much better PR-values using the enhanced CRP features than in the case of conventional chroma features. A good indicator for this is the maximal F-value $\mathrm{F}_{\max} := \max_\tau(\mathrm{F}_\tau)$, which is indicated by a dot within the respective PR-diagram in Figure 3.9. In our experiments, one obtains $\mathrm{F}_{\max} = 0.70$ and $\mathrm{F}_{\max} = 0.69$ for the conventional chroma features Chroma-IF and Chroma-Pitch, respectively. On the other hand, one obtains $\mathrm{F}_{\max} = 0.91$ for the CRP(55) features, which is an improvement of more than 30% over the conventional features.

Finally, Figure 3.9(b) shows the corresponding PR-diagrams using the binary shift measure c_{bs} instead of the cosine measure c. Also in this case, the CRP(55) features still outperform the conventional features, in particular with regard to recall. However, as explained in Section 3.3.6, there tend to be a notable number of false positives when using c_{bs}, which is also reflected in the PR-diagrams. For example, when using CRP(55) features in combination with c_{bs}, there are already quite a number of false positive matches having cost zero. This experiment also indicates that the binary shift measure, even though being a very powerful tool in global matching scenarios, tends to be too coarse for local matching scenarios, see Section 3.2.

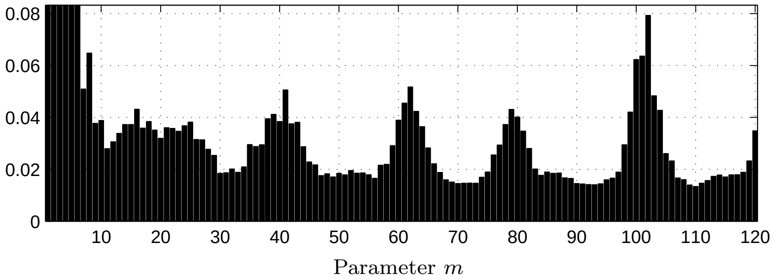


Figure 3.10. Entries $\bar{y}(m)$ of the vector $\bar{y}$ for $m \in [1:120]$ (horizontal axis). The value $\bar{y}(m)$ indicates the average absolute correlation of the normalized pitch vectors of the database recordings with the DCT basis vector $\mathbf{c}_m$.

3.4 Detailed Analysis

This section gives a detailed analysis of the DCT-based reduction step, which plays a central role in the CRP enhancement procedure. In Section 3.4.1, we show that for harmony-based music the upper PFCCs are dominated by a few dominating coefficients. As it turns out, these coefficients correspond to pitch periodicities that allow for a musically meaningful interpretation (Section 3.4.2). Furthermore, we show that the PFCCs surrounding the dominating ones account for different phases or pitch transpositions (Section 3.4.3).

3.4.1 Relation of DCT Basis Vectors to Pitch Vectors

For CRP features, the logarithmized pitch vectors are transformed by means of a discrete cosine transform (DCT). This transform is represented by an orthogonal 120 × 120 matrix denoted by DCT_{120}, where the m^{th} row of DCT_{120} can be thought of as a 1-sampled cosine function of frequency $freq(m) = \frac{m-1}{2\cdot 120}$, $m \in [1:120]$. In the following, this vector is denoted by $\mathbf{c}_m$ and referred to as the m^{th} DCT basis vector. The period of $\mathbf{c}_m$ is given by $period(m) = \frac{1}{freq(m)}$. Now, computing the matrix-vector product $y = \mathrm{DCT}_{120} \cdot x$ for a pitch vector $x \in \mathbb{R}^{120}$, the m^{th} coefficient $y(m)$ of y expresses to which degree x and $\mathbf{c}_m$ correlate.

To get some hints on a possible semantic meaning of the DCT basis vectors, the following experiment was conducted. First, logarithmized pitch vectors as described in Section 3.3 were computed for each audio recording of the database (Section 3.3.1). After normalizing each of these vectors with respect to the Euclidean norm, a DCT was applied to obtain PFCC vectors. Then, each entry of these coefficient vectors was replaced by its absolute value. Finally, all resulting vectors were averaged over the entire database to obtain a single 120-dimensional vector, say $\bar{y}$. This vector is shown (in a horizontal form) in Figure 3.10. The entry $\bar{y}(m)$ can be interpreted to represent the average absolute correlation of the normalized pitch vectors with the DCT basis vector $\mathbf{c}_m$.

The lower PFCCs, which are related to loudness and timbre, are left unconsidered in the CRP features, and we also disregard them in the following analysis. As revealed by Figure 3.10, some of the upper PFCCs indicate that certain DCT basis vectors show

DCT basis vector	$\mathbf{c}_{21}$	$\mathbf{c}_{41}$	$\mathbf{c}_{61}$	$\mathbf{c}_{81}$	$\mathbf{c}_{101}$	$\mathbf{c}_{120}$
frequency	0.083	0.167	0.250	0.333	0.417	0.496
period	12	6	4	3	2.4	≈ 2

Table 3.6. Frequency and period for selected DCT basis vectors.

a strikingly high average correlation with the pitch vectors. This particularly holds for all DCT basis vectors $\mathbf{c}_m$ with $m \in \mathcal{S} := \{41, 61, 81, 101, 120\}$. Actually, as seen from Table 3.6, all these basis vectors are 12-periodic (or nearly 12-periodic in the case $m = 120$).

3.4.2 Musical Meaning of Dominating DCT Basis Vectors

The dominance of the 12-periodic DCT basis vectors does not come all of a sudden, but originates from certain musical properties of the underlying audio material. We now give some explanations for this dominance. Recall that the CRP enhancement strategy is based on the pitch-frequency scale, which has a much closer relation to harmony-based music than the mel-frequency scale. In our setting, the DCT basis vectors capture certain periodicities of a pitch vector along the 120-dimensional pitch scale. The 12-periodicity is strongly connected to the octave interval that plays a crucial role in musical sounds and harmony-based music [5]. First, playing a musical note on an instrument typically produces a sound involving several frequencies known as harmonics, where the harmonics are integer multiples of the fundamental frequency. Since many of the harmonics are in an octave relationship, a pitch vector computed from a musical sound typically contains some quasi-periodic patterns of period 12. Second, Western music is often based on the use of specific *chords*, i. e., pattern of notes that are played simultaneously. Typical examples are major and minor 3-chords or major seventh 4-chords. Also the tonic-dominant relationship plays a fundamental role in Western harmony-based music. This implies the importance of certain pitch intervals including the fifth (pitch distance 7), the fourth (pitch distance 5), the major third (pitch distance 4), the minor third (pitch distance 3), and the octave (pitch distance 12).

Because of these two reasons—the nature of harmonics and the nature of harmony-based music—the pitch vectors derived from such music recording often exhibit quasi-periodic patterns, which are captured by the dominant DCT coefficients. To illustrate this fact, we discuss the result of some experiments similar to the one described in Section 3.4.1. Instead of using pitch vectors from real audio recordings, the experiment employs constructed sets of pitch vectors that correspond to specific harmonic chords. For example, the C-major chord is represented by a pitch vector where all entries are set to one that correspond either to pitch class C, E, or G; all other entries were set to zero. Other chords are represented in the same fashion. Now, using the set of pitch vectors covering all major chords, we compute the average absolute correlation vector $\bar{y}$. The resulting vector $\bar{y}$, which is shown Figure 3.11(a), clearly exhibits the dominance of $\mathbf{c}_m$ for $m = 61$, $m = 81$, and $m = 101$. Here, the basis vector $\mathbf{c}_{61}$ accounts for the major third (distance 4) and $\mathbf{c}_{81}$ for the minor third (distance 3). Interestingly, the basis vector $\mathbf{c}_{101}$ of period 2.4 accounts for the tonic-dominant relationship, which is based on a fifth (distance 7) and a fourth (distance 5) to the next octave. Here, note that twice the period 2.4 picks up

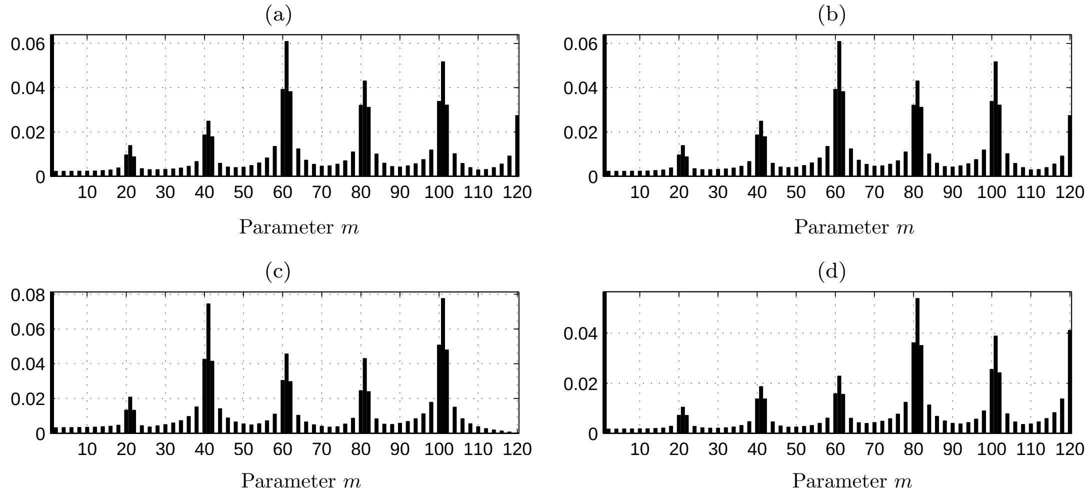


Figure 3.11. Average absolute correlation of the pitch vectors of various sets with the DCT basis vectors $\mathbf{c}_m$ for $m \in [1:120]$. **(a)** Major 3-chords. **(b)** Minor 3-chords. **(c)** Tonic/dominant 2-chords. **(d)** Major seventh 4-chords.

	μ_{T}	μ_{F}	α	μ_{T}	$\mu_{\mathrm{F}}^{1\%}$	β	$\max_{\mathrm{T}}$	$\min_{\mathrm{F}}$	γ
CRP(55)	0.092	0.626	0.150	0.092	0.244	0.388	0.124	0.187	0.693
CRP($\mathcal{S}$)	0.105	0.667	0.158	0.105	0.249	0.459	0.145	0.199	0.799
CRP$^{\mathrm{sin}}$($\mathcal{S}$)	0.112	0.673	0.169	0.112	0.290	0.397	0.150	0.225	0.696
CRP($\overline{\mathcal{S}}$)	0.110	0.677	0.165	0.110	0.292	0.387	0.148	0.227	0.682

Table 3.7. Quality measures for various CRP variants.

the fourth (distance $4.8 \approx 5$) and three times the period 2.4 picks up the fifth (distance $7.2 \approx 7$). Similar experiments were conducted with a set of minor 3-chords, a set of tonic-dominant 2-chords, and a set of major-seventh 4-chords, see (b)-(d) of Figure 3.11. For example, Figure 3.11(d) reveals a striking dominance of $\mathbf{c}_{81}$ of period 3, which indeed reflects the importance of the minor third in seventh chords. Finally, the basis vector $\mathbf{c}_{120}$ of approximate period 2 accounts for the dominance of whole steps (distance 2) in harmonic chords. Finally, we emphasize that the 12-periodic basis vectors $\mathbf{c}_m$ for $m \in \mathcal{S} = \{41, 61, 81, 101, 120\}$ additionally account for the octave relationship. This not only explains the musical importance of these basis vectors but also the "jumps" in the curves of Figure 3.7 at the corresponding index positions.

3.4.3 Phase Shift Simulation by DCT Basis Vectors

So far, we have seen that the DCT basis vectors $\mathbf{c}_m$ for $m \in \mathcal{S}$ capture the musically important pitch periodicities. At this point, one may assume that a reduction using only the few dominating DCT basis vectors may yield a similar enhancement than using the entire range of upper PFCCs. To investigate this assumption, we conduct the following experiment. In the construction of the CRP features, we only keep the five PFCCs corresponding to the set $\mathcal{S}$ and discard the other 115 PFCCs by setting them to zero. We then apply the inverse DCT, the chroma binning, and the normalization as before, see

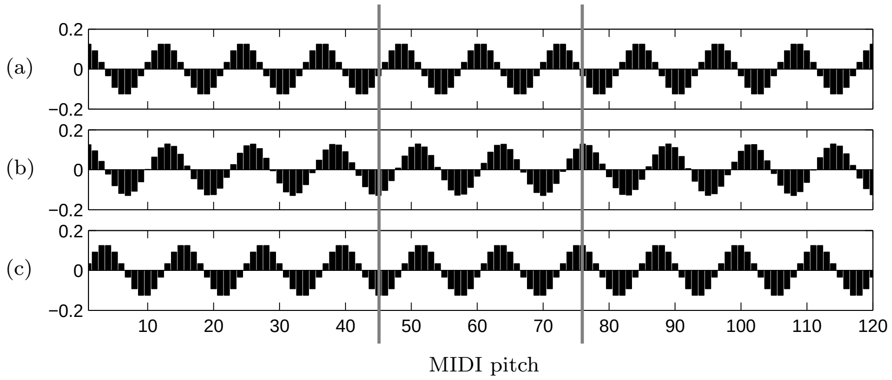


Figure 3.12. **(a)** DCT basis vector $\mathbf{c}_{21}$. **(b)** DCT basis vector $\mathbf{c}_{20}$. **(c)** Phase-shifted version of $\mathbf{c}_{21}$ shifted by $\pi/2$. Note that the vectors shown in (b) and (c) nearly coincide in the part between the two gray vertical lines.

Figure 3.1. The resulting features are referred to as CRP($\mathcal{S}$) features. Table 3.7 shows the results of our various quality measures for the CRP($\mathcal{S}$) features. Even though the CRP($\mathcal{S}$) features achieve some improvement over conventional chroma features (cf. Table 3.2), there is a significant degradation in the β- and γ-measures compared to CRP(55) features. For example, one has $\beta = 0.388$ for CRP(55) features, whereas $\beta = 0.459$ for CRP($\mathcal{S}$) features (Table 3.7).

The main reason for this degradation can be explained as follows. First, recall that a DCT basis vector is obtained by sampling a suitable cosine function of certain frequency and phase. Using only one DCT basis vector of a fixed phase for a specific pitch periodicity, one is not able to deal with phase shifts, which can be interpreted as pitch transpositions in our scenario. For example, the DCT basis vector $\mathbf{c}_{101}$ is able to capture periodicities stemming from a C-major chord, but has difficulties in capturing the same periodicities in the case of a D-major chord. One can deal with phase shifts as is done in Fourier analysis [116]: one simply complements each DCT basis vectors by an additional phase-shifted (shifted by $\pi/2$) duplicate. In our scenario, we introduce an additional phase-shifted basis vector $\mathbf{s}_m$ for each DCT basis vector $\mathbf{c}_m$, $m \in \mathcal{S}$. Here, $\mathbf{s}_m$ is obtained by sampling a sine function that corresponds to the cosine function used to obtain $\mathbf{c}_m$. Now, we project the pitch vectors onto the space spanned by the ten basis vectors $\mathbf{c}_m$ and $\mathbf{s}_m$, $m \in \mathcal{S}$. (Before, we only used the five vectors $\mathbf{c}_m$.) Then, we continue with the usual chroma binning and normalization to obtain features denoted by CRP$^{\text{sin}}$($\mathcal{S}$). As shown by Table 3.7, the CRP$^{\text{sin}}$($\mathcal{S}$) features exhibit much better β- and γ-measures than the CRP($\mathcal{S}$) features and qualitatively come up to the original CRP(55) features.

Finally, we investigate how this phase shift information is recovered in the case of the purely cosine-based CRP(n) features. Looking at Figure 3.10 and Figure 3.11, one can notice that the dominating PFCCs corresponding to the DCT basis vectors $\mathbf{c}_m$, $m \in \mathcal{S}$, are flanked at both sides by further relevant PFCCs. Exemplarily, we look at $\mathbf{c}_{21}$ and its adjacent basis function $\mathbf{c}_{20}$, see (a) and (b) of Figure 3.12. The two underlying cosine functions differ only slightly in their frequency. As a consequence, $\mathbf{c}_{20}$ behaves like a phase-shifted version of $\mathbf{c}_{21}$ in the middle part of the pitch scale. In this part, the vector $\mathbf{c}_{20}$ nearly coincides with $\mathbf{s}_{21}$, cf. (b) and (c) of Figure 3.12. In other words, phase-shifts in the middle part of the pitch scale are simulated by DCT basis vectors with a slightly changed frequency. This property is particularly important in view of real-world music recordings, where most of the energy is concentrated in the middle part of the pitch scale. We close our discussion by a final experiment, which reinforces these explanations. Here, we use in the reduction step the set $\overline{\mathcal{S}} := \{40 : 42, 60 : 62, 80 : 82, 100 : 102, 119 : 120\}$ instead of the set $\mathcal{S}$. The resulting features, which are denoted as $\mathrm{CRP}(\overline{\mathcal{S}})$ features, indeed exhibit a similar β- and γ-measure as the $\mathrm{CRP}^{\mathrm{sin}}(\mathcal{S})$ and the CRP(55) features, see Table 3.7.

3.5 Conclusions

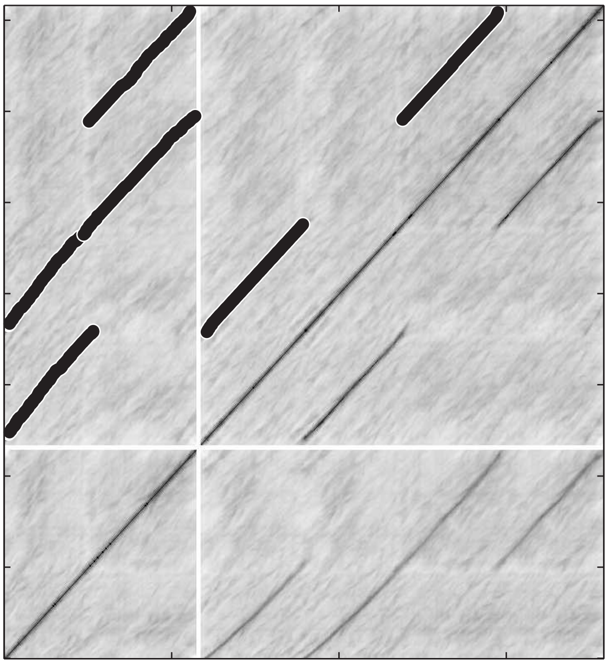
This chapter introduced a novel enhancement procedure for significantly increasing the robustness of conventional chroma features to changes in timbre and instrumentation. Here, the main ideas were first to compute cepstral coefficients based on a pitch-frequency scale, second to discard the lower PFCCs, and third to deduce from the remaining upper PFCCs the chroma-based CRP features. As it turned out, the upper PFCCs are dominated by only a few coefficients that reflect harmonically relevant pitch-periodicities as prominent in Western music. Revealing the musical meaning of certain PFCCs not only puts the procedure in a nutshell, but also allows for further reducing the number of PFCCs without a degradation of the discriminative power of the resulting CRP features.

Extensive experiments showed that the proposed enhancement strategy yields a significant boost towards timbre invariance independent of a particular choice of parameters and measures. Using the novel CRP features, one can significantly improve the performance in all those matching and classification applications, where one wants to be invariant with regard to instrumentation and tone color. Exemplarily, this was shown for an audio retrieval application, where precision and recall values substantially increased when using CRP features instead of conventional chroma features. For the future, it will be interesting to apply CRP features also for other MIR tasks such as cover song identification [38, 164], structure analysis [65, 127, 145, 157], and cross-domain music matching [54, 147]. Other authors already reported on good results using CRP and PFCC features for structure analysis [6], chord recognition [15, 81], instrument classification [76], and music synchronization [101].

Generally, the direct comparison of audio features as well as the assessment of the features' properties is a difficult and time-consuming problem. Here, as a further conceptual contribution of this chapter, the proposed evaluation framework constitutes a powerful tool for comparing and studying the behavior of audio features in a compact form and systematic way. Using a DTW-based distance function, this chapter introduced various quality measures that express separation and matching capabilities on the basis of real-world music material. Note that the musical meaning of the measures very much depend on the particular choice of the underlying audio material. For the evaluation, music recordings were carefully selected and annotated to obtain quality measures that indicate the degree of timbre invariance exhibited by the respective feature type. By suitably changing the audio material and the annotations, the framework can easily be adjusted to also facilitate the evaluation of audio features with regard to other musical aspects such as timbre (not timbre-invariance), rhythm, or melodic similarity.

Part II

Music Synchronization



Chapter 4

Alignment Methods

As a result of massive digitization efforts, there is an increasing number of relevant digital documents for a single musical work comprising audio recordings, MIDI files, digitized sheet music, music videos, and various symbolic representations. In order to coordinate the multiple information sources related to a given musical work, various alignment and synchronization procedures have been proposed with the common goal to automatically link several types of music representations, see [1,21,28,29,31,55,75,96,116,129–131,152, 154,163,164,174,181,187]. Potential applications of synchronization techniques include automatic score following [20, 21, 28, 29, 36, 163], audio segmentation [152], cover song identification [164] and cross-modal music retrieval [75, 96, 129]. In general terms, *music synchronization* denotes a procedure which, for a given position in one representation of a piece of music, determines the corresponding position within another representation. Depending upon the respective data formats, one distinguishes between various synchronization tasks [2, 116]. For example, *audio-audio* synchronization [31, 131, 181] refers to the task of time aligning two different audio recordings of a piece of music. These alignments can be used to jump freely between different interpretations, thus affording efficient and convenient audio browsing. The goal of *score-audio* and *MIDI-audio* synchronization [2,28,130,154,174] is to coordinate note and MIDI events with audio data. The result can be regarded as an automated annotation of the audio recording with available score and MIDI data. A recently studied problem is referred to as *sheet music-audio* synchronization [96], where the objective is to link regions (given as pixel coordinates) within the scanned images of given sheet music to semantically corresponding physical time positions within an audio recording. Such linking structures can be used to highlight the current position in the scanned score during playback of the recording. Similarly, the goal of *lyrics-audio* synchronization [55, 89, 129] is to align given lyrics to an audio recording of the underlying song. For an overview of related alignment and synchronization problems, see also [29, 116, 119].

Computational and algorithmic challenges for music synchronization approaches arise in particular from the complexity and diversity of music data. For example, the music representations to be aligned might differ in terms of the local tempo, the instrumentation, or the overall dynamics. Variations in the musical structure or the polyphony and differences in the realization of trills and arpeggios add further complexity to the already difficult task. In this chapter, we summarize basic alignment techniques that underlie most state-of-the-art synchronization approaches. It is one goal of this chapter to give a unifying view on these approaches while highlighting the conceptual differences. In chapters 5, 6, and 7, we will develop novel methods based on the techniques presented in this chapter and improve upon them in terms of accuracy, efficiency, reliability, and robustness.

Most alignment and synchronization procedures proceed in three basic steps. In the first step, the data streams to be aligned are converted to a suitable feature representation. Then, a local cost measure is used to compare features from the two streams. Finally, based on this comparison, the actual synchronization result is computed using a suitable alignment strategy. For synchronizing several versions or representations of a piece of music, chroma-based features and contextual cost measures have proven to be suitable tools, which are reviewed in Section 4.1. Then, in the remainder of this chapter, we focus on the third step and describe three conceptually different alignment strategies: dynamic time warping (Section 4.3), a recursive version of the Smith-Waterman algorithm (Section 4.4), and partial matching (Section 4.5). While these three approaches share similar algorithmic roots (dynamic programming) and possess a close mathematical modeling (Section 4.2), they produce fundamentally different types of alignments, see also Section 4.6. For a discussion of these differences we consider in the following the case of MIDI-audio synchronization. As a running example, we use a MIDI version and an audio recording for the song ‘And I love her’ by the Beatles. However, our findings also hold for other cases such as audio-audio synchronization.

4.1 Feature Representation and Cost Measure

To compare different versions of a same song, we convert both representations into a common mid-level representation. Depending on the type of this representation, the comparison can be based on musical properties such as harmony, rhythm or timbre. In the subsequent discussion, we employ normalized 12-dimensional chroma features as described in Chapter 2 with a temporal resolution of 2 Hz (2 features per second). Such feature rates have also turned out to be suitable for related tasks such as audio matching [128] and cover song detection [164]. Let $V := (v^1, v^2, \dots, v^N)$ and $W := (w^1, w^2, \dots, w^M)$ be two chroma feature sequences. To relate two chroma vectors, we use the cosine distance defined by $c(v^n, w^m) = 1 - \langle v^n, w^m \rangle$ for normalized vectors. By comparing the features of the two sequences in a pairwise fashion, one obtains an $(N \times M)$-*cost matrix* C defined by $C(n, m) := c(v^n, w^m)$, see Figure 4.1a. Each tuple (n, m) is called a *cell* of the matrix. To increase the robustness of the overall alignment procedure, it is often beneficial to also include the local temporal evolution of the features in order to enhance the structural properties of a cost matrix. To this end, Foote [51] proposed to average the cost values from a number of consecutive frames and to use that as the new cost value. This results in a smoothing effect of C. Müller and Kurth [126] extended these ideas by suggesting a contextual distance measure that allows for handling local tempo variations in the underlying audio recording. The enhancement procedure can be thought of as a multiple filtering of C along various directions given by gradients in a neighborhood of the gradient $(1, 1)$, see [116, 126] for further details. We denote the smoothed cost matrix again by C. The degree of smoothing depends on a parameter λ, which specifies the number of consecutive frames taken into account for the filtering. The role of this parameter will be discussed in Section 5.3. For an example see Figure 4.1c.

4.2 General Concepts

We now introduce some common mathematical notations that are shared by all three alignment procedures to be discussed. Generally, an *alignment* between the feature sequences $V := (v^1, v^2, \ldots, v^N)$ and $W := (w^1, w^2, \ldots, w^M)$ is regarded as a set $\mathcal{A} \subseteq [1:N] \times [1:M]$, where $[1:N]$ is a shorthand for $\{1, 2, \ldots, N\}$. Here, each cell $\pi = (n, m) \in \mathcal{A}$ encodes a correspondence between the feature vectors v^n and w^m. By ordering its elements lexicographically $\mathcal{A}$ takes the form of a sequence, i. e., $\mathcal{A} = (\pi_1, \ldots, \pi_L)$ with $\pi_\ell = (n_\ell, m_\ell)$, $\ell \in [1:L]$. Additional constraints on the set ensure that only musically meaningful alignments are permitted. We say that the set $\mathcal{A}$ is *monotonic* if

$$n_1 \leq n_2 \leq \ldots \leq n_L \text{ and } m_1 \leq m_2 \leq \ldots \leq m_L.$$

Similarly, we say that $\mathcal{A}$ is *strictly monotonic* if

$$n_1 < n_2 < \ldots < n_L \text{ and } m_1 < m_2 < \ldots < m_L.$$

Note that the monotonicity condition reflects the requirement of faithful timing: if an event in V precedes a second one this also should hold for the aligned events in W. A strictly monotonic set $\mathcal{A}$ will also be referred to as *match*, denoted by the symbol $\mathcal{M} = \mathcal{A}$. To ensure certain continuity conditions, we introduce step-size constraints by requiring

$$\gamma_{\ell+1} - \gamma_\ell \in \Sigma$$

for $\ell \in [1:L-1]$, in which Σ denotes a set of admissible step sizes. A typical choice is $\Sigma = \Sigma_1 := \{(1,1), (1,0), (0,1)\}$ or $\Sigma = \Sigma_2 := \{(1,1), (2,1), (1,2)\}$. A set $\mathcal{A}$ that fulfills the step-size condition is also referred to as *path* denoted by the symbol $\mathcal{P} = \mathcal{A}$. Note that when using Σ_1 the set $\mathcal{A}$ also becomes monotonic allowing a relatively high degree of flexibility in the alignment path. Using Σ_2 instead typically results in more restricted alignments with additional slope constraints, which, on the positive side, often introduces a higher degree of robustness. As final constraint, the boundary condition

$$\gamma_1 = (1,1) \text{ and } \gamma_L = (N, M),$$

ensures in combination with a step-size condition the alignment of V and W as a whole. If both the step-size as well as the boundary condition hold for a set $\mathcal{A}$, then $\mathcal{A}$ will be referred to as *global path* (or *warping path*) denoted by $\mathcal{G}$. Finally, a monotonic set $\mathcal{A}$ is referred to as *family of paths*, denoted by $\mathcal{F}$, if there exist paths $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_K$ with $\mathcal{F} = \mathcal{A} = \bigcup_{k \in [1:K]} \mathcal{P}_k$.

4.3 Dynamic Time Warping

If it is known a-priori that the two sequences to be aligned correspond to each other globally then a global path is the correct alignment model. Here, classical *dynamic time*

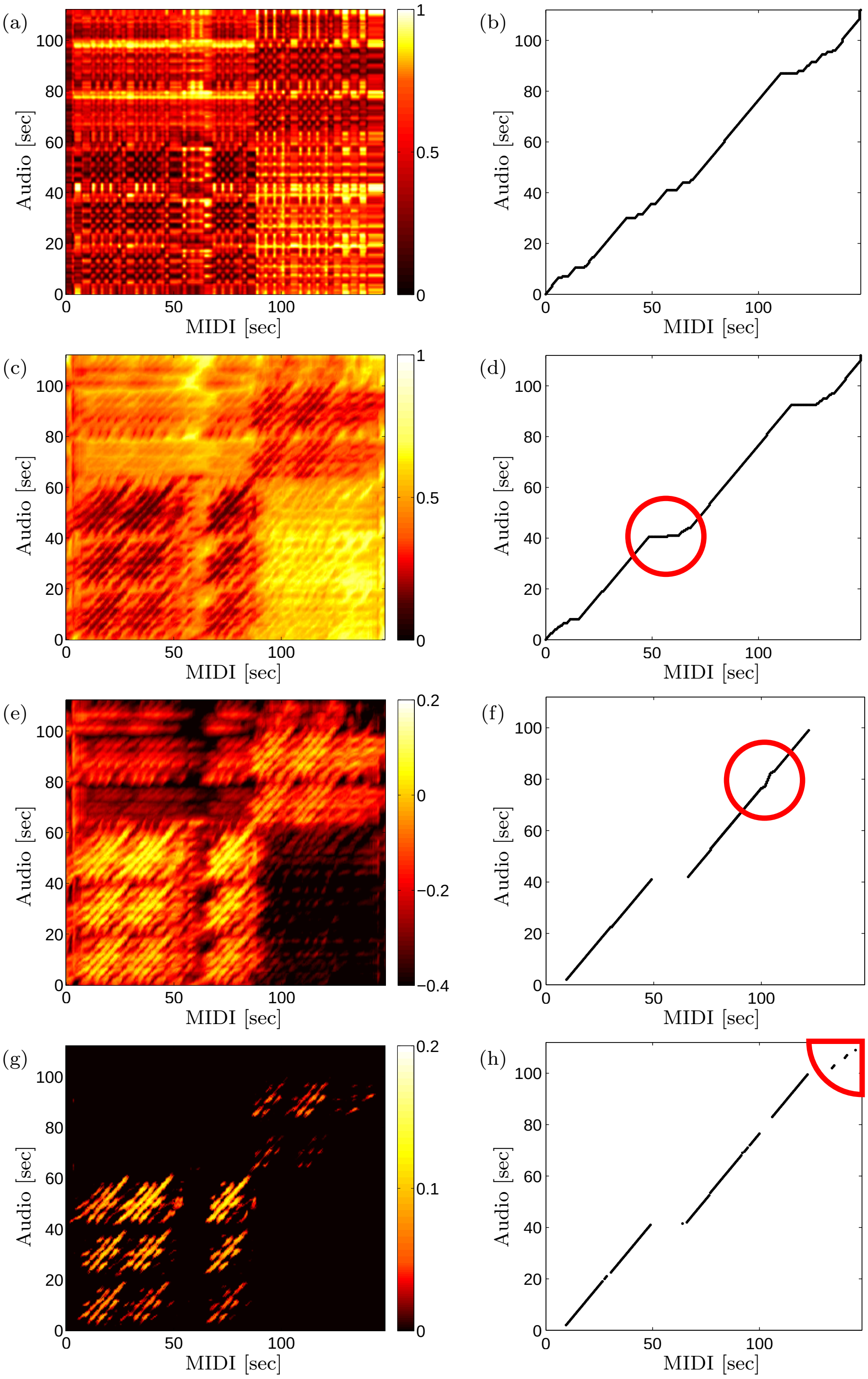


Figure 4.1. Several techniques for the alignment of an audio recording (vertical axis) and a MIDI version (horizontal axis) of the song 'And I love her' by the Beatles. The marked regions are further discussed in the text. **(a)** Chroma-based cost matrix. **(b)** Optimal global path obtained via DTW based on the chroma cost matrix. **(c)** Smoothed cost matrix C using $\lambda = 12$. **(d)** Optimal global path obtained via DTW based on matrix C. **(e)** Score matrix S. **(f)** Family of paths obtained via Smith-Waterman based on matrix S. **(g)** Thresholded score matrix $S_{\geq 0}$. **(h)** Optimal match obtained via partial matching based on matrix $S_{\geq 0}$.

warping (DTW), which has originally been used to compare different speech patterns in automatic speech recognition [150], can be used to compute a global path. In this context, the *cost* of an alignment $\mathcal{A}$ is defined as $c_{\mathcal{A}}(V, W) := \sum_{\ell=1}^{L} C(n_\ell, m_\ell)$. Then, after fixing a set of admissible step-sizes Σ, DTW yields an optimal global path having minimal cost among all possible global paths. More exactly, the *DTW distance* between V and W is defined as

$$DTW(V, W) := \min\{c_{\mathcal{G}}(V, W) \mid \mathcal{G} \text{ is a global path between } V \text{ and } W\}$$

An *optimal global path* $\mathcal{G}^*$ is a global path having minimal cost, i.e. $c_{\mathcal{G}^*}(V, W) = DTW(V, W)$. To compute an optimal global path, a brute force method could consider each possible global path. However, since the number of possible global paths exponentially increases in N and M, this is not feasible in practice. Using dynamic programming techniques, one can reduce the computational complexity to $O(NM)$. To this end, we define the prefix sequences $V(1 : n) := (v_1, \ldots, v_n)$ with $n \in \{1, \ldots, N\}$ and $W(1 : m) := (w_1, \ldots, w_m)$ with $m \in \{1, \ldots, M\}$ for V and W, respectively. Then, we can define the *accumulated cost matrix* $D \in \mathbb{R}^{N \times M}$ via $D(n, m) := DTW(V(1 : n), W(1 : m))$. Using step size condition Σ_1, one can show [116] that D can be computed in $O(NM)$ arithmetic operations using the following recursion:

$$D(n, m) := \min \begin{cases} D(n-1, m) + C(n, m), \\ D(n, m-1) + C(n, m), \\ D(n-1, m-1) + C(n, m) \end{cases}$$

The recursion is well-defined using the definitions $D(1, 1) := C(1, 1)$, $D(n, 1) := \sum_{k=1}^{n} C(k, 1)$ for $n \in \{1, \ldots, N\}$ and $D(1, m) := \sum_{k=1}^{m} C(1, k)$ for $m \in \{1, \ldots, M\}$. For other step size conditions, such as Σ_2, matrix D can be computed in a similar way [116]. Starting with $D(N, M)$, one can compute an optimal global path $\mathcal{G}^*$ by tracking the minimizing argument in the recursive definition of D. For more details on DTW in a musical context, see [116].

For the subsequent discussion, we use $\mathrm{A}(s, t)$ to refer to the segment in the audio recording starting at s seconds and terminating at t seconds. Similarly, $\mathrm{M}(s, t)$ refers to a MIDI segment. So listening to $\mathrm{M}(55, 65)$ of the song 'And I love her' (used throughout Figure 4.1) reveals a short bridge in the song. However, in the particular audio recording used here the bridge is omitted. Since DTW always aligns the sequences as a whole we find a musically inappropriate alignment between $\mathrm{A}(40, 42)$ and $\mathrm{M}(48, 65)$, see also the marked region in Figure 4.1d. A similar observation can be made at the beginning and the end of the optimal global path. Here, the intro and outro in the audio recording deviate strongly from those in the MIDI version.

4.4 Recursive Smith Waterman

In general, using DTW in the case that elements in one sequence do not have suitable counterparts in the other sequence is problematic. In particular, in the presence of structural differences between the two sequences, this typically leads to misalignments. Therefore, if it is known a-priori that the two sequences to be aligned only partially correspond to each other, a path or a family of paths allows for a more flexible alignment than a global path.

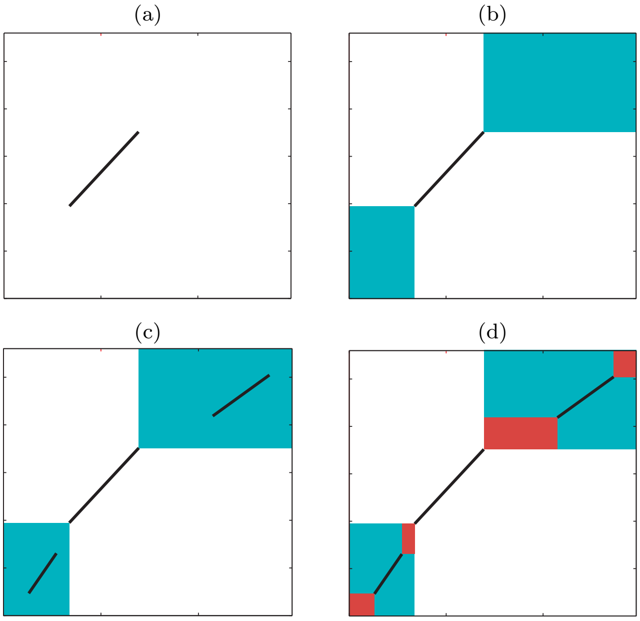


Figure 4.2. First steps of our recursive Smith-Waterman variant. **(a)** Optimal path $\mathcal{P}$ derived via classical Smith-Waterman. **(b)** Submatrices defined via $\mathcal{P}$. **(c)** Result after the first recursion. Optimal paths have been derived from the submatrices. **(d)** New submatrices for the next recursive step are defined.

To align two sequences that correspond only locally to each other, one can deploy the *Smith-Waterman algorithm*—a well-known technique originally used in biological sequence analysis [146, 173]. In the music context, this algorithm has also been successfully used for the task of cover song identification [164]. Instead of using the concept of a cost matrix with the goal of finding a cost-minimizing alignment, one now uses the concept of a *score matrix* with the goal to find a score-maximizing alignment. To obtain a score matrix S from a cost matrix C, we fix a threshold $\tau > 0$ and set $S = \tau - C$. Figure 4.1e shows a score matrix derived from the cost matrix shown in Figure 4.1c. The *score* of an alignment $\mathcal{A}$ is defined as $\sum_{\ell=1}^{L} S(n_\ell, m_\ell)$. Then, after fixing a set of admissible step-sizes Σ, the Smith-Waterman algorithm computes an optimal path having maximal score among all possible paths. Cells of $\mathcal{A}$ having negative score are often referred to as gaps, where one considers gap openings and gap extensions. Typically, such gaps are further penalized by introducing additional *gap-penalty parameters* [146, 164]. In our setting, for simplicity, we use a single gap parameter γ for openings as well as extensions. Then, this parameter can be realized by a subtraction of γ from all negative entries in the score matrix S. Using dynamic programming techniques, the Smith-Waterman algorithm can be implemented similarly to DTW. To this end, we define an *accumulated score matrix* $T \in \mathbb{R}^{N\times M}$, exemplarily using the step size condition Σ_1, as

$$T(n,m) := \max \begin{cases} T(n-1,m) + \delta_\gamma(S(n,m)), \\ T(n,m-1) + \delta_\gamma(S(n,m)), \\ T(n-1,m-1) + \delta_\gamma(S(n,m)), \\ 0 \end{cases}$$

where $\delta_\gamma : \mathbb{R} \to \mathbb{R}$ is defined as

$$\delta_\gamma(x) = \max \begin{cases} x, & x \geq 0 \\ x - \gamma & x < 0 \end{cases} .$$

The recursion is well-defined using the definitions $T(1,1) := \max\{S(1,1), 0\}$, $T(n,1) := \max\{T(n-1,1) + \delta_\gamma(S(n,1)), 0\}$ for $n \in \{1, \ldots, N\}$ and $T(1,m) := \max\{T(1,m-1) + \delta_\gamma(S(1,m)), 0\}$ for $m \in \{1, \ldots, M\}$. For step size conditions other than Σ_1, matrix T can be computed in a similar way [164]. Starting with the entry in T having the maximal value, one can compute an optimal path using backtracking techniques similar to DTW.

The original Smith-Waterman algorithm only delivers a single alignment path, which is often not enough to encode a suitable alignment. Therefore, we now introduce a novel recursive variant of the Smith-Waterman algorithm. First, we derive an optimal path $\mathcal{P}$ as described above, see Figure 4.2a. Then, we define two submatrices in the underlying score matrix S, see Figure 4.2b. The first matrix is defined by the cell $(1,1)$ and the starting cell of $\mathcal{P}$, and the second matrix by the ending cell of $\mathcal{P}$ and the cell (N,M). For these submatrices, we call the Smith-Waterman algorithm recursively to derive another optimal path for each submatrix, see Figure 4.2c. These new paths define new submatrices on which Smith-Waterman is called again, see Figure 4.2d. This procedure is repeated until either the score of an optimal path or the size of a submatrix is below a given threshold. This results in a monotonic alignment set in form of a family of paths $\mathcal{F}$. Figure 4.1f shows a family of two paths derived from the score matrix in Figure 4.1e using our recursive Smith-Waterman variant. Using this method, the missing bridge in the audio as well as the different intros and outros in the audio and MIDI version are detected and, in this example, the recursive Smith-Waterman approach avoids the misalignment of the DTW case (Figure 4.1d).

While this example highlights some of the strengths of the Smith-Waterman algorithm, it also illustrates one of its weaknesses. Listening to A$(75, 83)$ and M$(99, 107)$ reveals a solo improvisation which differs in the audio and MIDI versions, so they should not be aligned. Also, the corresponding area in the score matrix shows negative values. However, the Smith-Waterman algorithm aligns these two segments as part of the second path, see marked region in Figure 4.1f. The reason is that Smith-Waterman always tries to find the path with maximum score, where even a small number (relative to the total length of the path) of gaps are tolerated.

4.5 Partial Matching

As a third approach, we now describe a *partial matching* strategy, which gives the least constrained alignment [2, 116, 146]. Here, similar to the Smith-Waterman approach, the goal is to find an alignment with maximal score. However, in this case we require that the alignment is a match (strictly monotonous alignment) without imposing any further step size conditions. Therefore, opposed to a score-maximizing path, there are no cells of negative score in a score-maximizing match. Thus, negative scores can be ignored completely and we therefore use the rectified version $S_{\geq 0}$, in which every negative entry in S is replaced by zero (see Figure 4.1g). Again, a score-maximizing match can be computed efficiently using dynamic programming. To this end, we define a matrix U as

$$U(n,m) := \max \begin{cases} U(n-1,m), \\ U(n,m-1), \\ U(n-1,m-1) + S_{\geq 0}(n,m) \end{cases}$$

and $U(0,0) := U(n,0) := U(0,m) := 0$ for $n \in \{1,\ldots,N\}$ and $m \in \{1,\ldots,M\}$. Then a score-maximizing match $\mathcal{M}^*$ can be computed using the following algorithm, see [116] for further details.

```
Initialization:   $n := N$, $m := M$, $\mathcal{M}^* := \emptyset$
Procedure:        while $(n > 0)$ and $(m > 0)$ do
                      if $D(n,m) = D(n,m-1)$ then
                          $m := m - 1$
                      elseif $D(n,m) = D(n-1,m)$ then
                          $n := n - 1$
                      else $\mathcal{M}^* := \mathcal{M}^* \cup \{(n,m)\}$, $n := n-1$, $m := m-1$
                  return $\mathcal{M}^*$
```

Figure 4.1h shows an example of an optimal match computed via partial matching, based on the matrix shown in Figure 4.1g. Here, the misalignment of the solo segments A(75, 83) and M(99, 107) found in the Smith-Waterman case is not present. So partial matching, not enforcing any step-size or continuity conditions on the alignment, yields a more flexible alignment than the Smith-Waterman approach. However, in turn, this flexibility can also lead to spurious, inappropriate and fragmented alignments, as can be seen in segments A(101, 110) and M(127, 147), see marked region in Figure 4.1h.

4.6 Concluding Remarks

In summary, one may think of two extremes: on the one hand, DTW relies on strong model assumptions, but works reliably in the case that these assumptions are fulfilled; on the other hand, partial matching offers a high degree of flexibility, but may lead to alignments being locally misguided or split into many fragments. The Smith-Waterman approach lies in between these two extremes.

Furthermore, alignment problems as discussed in this chapter are closely related to tasks such as automated accompaniment [25,153] and score following [19,20,29,36]. However, alignment strategies often employed in these fields such as hidden Markov models (HMMs) [138] and other graphical models [20,154] are not further considered in the following. Such probabilistic methods usually require training data consisting of several different versions of the underlying audio material to identify statistical properties of the sound. Because only one audio version is assumed to be available in the following, such methods have not been incorporated. Further discussion about the use of graphical models in alignment scenarios can be found in [84,138,154].

Chapter 5

Late-Fusion-Based Partial Synchronization

Even though recent music synchronization approaches can handle significant variations in tempo, dynamics, and instrumentation, most of them rely on the assumption that the two versions to be aligned correspond to each other with respect to their global structure. However, using real-world data, these idealized conditions are often not met. For example, in popular music one often encounters structurally different album and radio edits as well as extended versions and remixes. Live or cover versions may contain improvisations, additional solos, and other deviations from the original song [164]. Poor recording conditions, interfering screams and applause, or distorted instruments may introduce additional serious degradations in the audio recordings. On the other side, MIDI and other symbolic descriptions often convey only a simplistic view of a musical work, where, e. g., certain voices or drum patterns are missing. Furthermore, symbolic data as obtained from optical music recognition is often corrupted by recognition errors.

In this chapter, we address the problem of *reliable partial music synchronization* with the goal to automatically identify those passages within the given music representations that allow for a reliable alignment. Given two different representations of the same piece, the idea is to use several types of conceptually different synchronization strategies to obtain an entire family of temporal alignments. Now, consistencies over the various alignments indicate a high reliability in the encoded correspondences, whereas inconsistencies reveal problematic passages in the music representations to be aligned. Based on this automated local classification of the synchronization results, we segment the music representations into passages, which are then further classified as *reliable* and *critical*. Here, the reliable passages have a high confidence of being correctly aligned with a counterpart, whereas the critical passages are likely to contain variations and artifacts. The reliable passages can then be used as anchors for subsequent improvements and refinements of the overall synchronization result. Conversely, this automated validation is also useful in revealing the critical passages, which often contain the semantically interesting and surprising parts of a representation.

The remainder of this chapter is organized as follows. In Section 5.1, we describe the proposed method. Then, we introduce in Section 5.2 a suitable quality measure, which is employed in Section 5.3 to investigate the role of the parameters used in the overall procedure. After that, we demonstrate the practical applicability of the proposed technique in Section 5.4. Here, we formulate the idea of a cross-version analysis for comparing and/or combining analysis results from different representations. As an example, we apply this idea in the context of harmonic analysis. In particular, we automatically evaluate MIDI-based chord labeling procedures using annotations given for corresponding audio recordings.

5.1 Consistency Alignment

As illustrated by the examples given in Chapter 4, alignments computed using previous methods may contain satisfying as well as misguided parts. Therefore, with no definite a-priori knowledge about the input data, none of the alignment methods presented in Chapter 4 can in general guarantee a reliable and musically meaningful alignment. However, if several strategies with different design goals yield locally similar alignment results, then there is a high probability that these results are musically meaningful. Based on this simple idea, we present now a novel late-fusion approach that combines several alignment procedures in order to identify passages that can be reliably synchronized. To allow for a direct comparison to the alignment methods described in Chapter 4, we consider here again the case of MIDI-audio synchronization and discuss the proposed method using the same running example, i.e., the song 'And I love her' by the Beatles. For other cases, such as audio-audio synchronization, the proposed method can be applied in a similar fashion.

Given a MIDI-audio pair for a song, we start by computing an optimal global path using DTW, a family of paths using recursive Smith-Waterman, and an optimal match using partial matching as described in Chapter 4. Here, we choose $\Sigma = \Sigma_1$ in the DTW alignment, since this leads to more flexibility in cases where the assumption of global correspondence between the sequences is violated. The Smith-Waterman procedure can better deal with local deviations in the two sequences to be aligned. Therefore it does not require the flexibility offered by Σ_1 and we can choose the more robust $\Sigma = \Sigma_2$. Partial matching does not incorporate any step-size constraints. In a next step, we convert each alignment into a binary matrix having the same size as the cost matrix C. Here, a cell in the matrix is set to one if it is contained in the corresponding alignment, and zero otherwise (actually, in Figure 4.1 the three alignments are already represented in this way). Next, we combine the three alignments using a late-fusion strategy to compute a kind of soft intersection. To this end, we augment the binary matrices by additionally setting every cell in the binary matrices to one that is within a neighborhood of an alignment cell (see Figure 5.1a-c). Without such a tolerance small differences between the individual alignments would lead to empty intersections. In the following, we use a neighborhood corresponding to one second. Here, experiments have shown that changing the neighborhood size within reasonable limits does not have a significant impact on the final results. In a last step, we derive an intersection matrix by setting each matrix cell to one that is one in all three augmented binary matrices (see Figure 5.1d).

The intersection matrix can be thought of as a rough indicator for areas in the cost matrix where the three alignment strategies agree. However, this matrix does not encode an

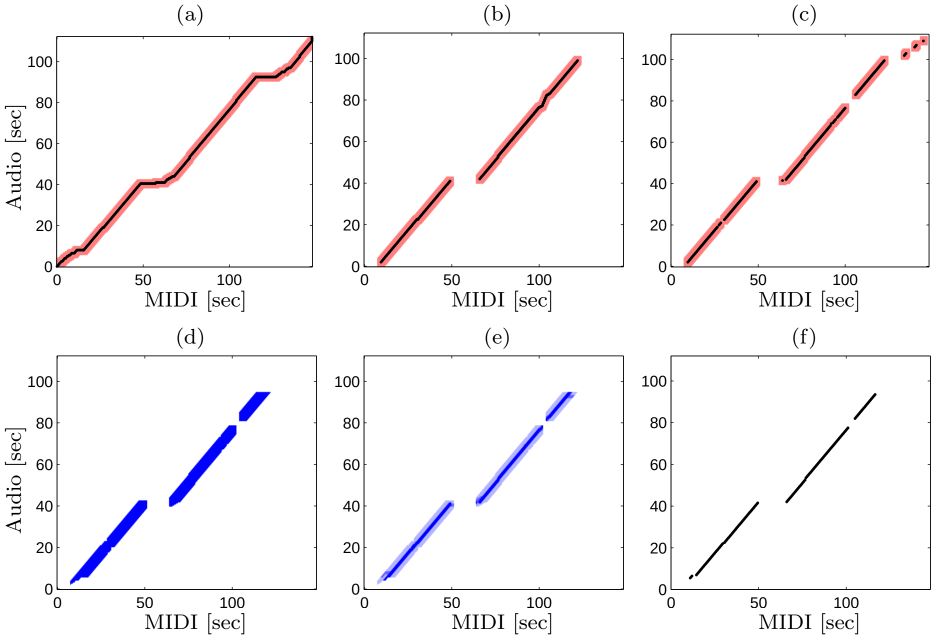


Figure 5.1. Steps in the proposed method continuing the example shown in Figure 4.1. **(a)-(c)** Alignment (black) and corresponding augmented binary matrix (red and white) for the optimal global path (DTW), family of paths (Smith-Waterman) and the optimal match (partial matching). **(d)** Intersection matrix derived from (a)-(c). **(e)** Weighted intersection matrix. **(f)** Consistency alignment $\mathcal{C}$.

alignment that is constrained by any of the conditions described in Section 4.2. Therefore, to derive a final alignment result from this matrix, we first weight the remaining cells in the intersection matrix according to how often they are contained in one of the original three alignments (Figure 5.1e). Then, interpreting the weighted matrix as a score matrix, we use partial matching to compute an optimal match, $\mathcal{C}$, referred to as the *consistency alignment* (Figure 5.1f).

In the following, we call a segment in the audio recording (in the MIDI version) *reliable* if it is aligned via $\mathcal{C}$ to a segment in the MIDI version (in the audio recording). Similarly, we call a segment *critical* if it is not aligned. Here, A(3, 39), A(39, 76) and A(83, 95) as well as M(8, 45), M(63, 99) and M(106, 117) are examples of reliable segments in the audio recording and in the MIDI version, respectively, see Figure 5.1f. However, the automatic detection of critical sections can also be very useful, as they often contain musically interesting deviations between two versions. For example, consider the critical segment M(45, 63). As discussed in Chapter 4, this segment contains the bridge found in the MIDI that was omitted in the audio recording. Here, the proposed method automatically revealed the inconsistencies between the MIDI version and the audio recording. The differences between the audio and the MIDI version in the intro, outro and solo segments have also been detected. Here, using multiple alignment strategies leads to a more robust detection of critical segments than using just a single approach. The reasons why a segment is classified as critical can be manifold and constitute an interesting subject for a subsequent musical analysis, which would be beyond the scope of this thesis. In this context, however, this novel approach provides support for such an analysis.

5.2 Evaluation Setup

To systematically evaluate the performance of the proposed procedure, we use 60 pieces from the classical and 60 pieces from the popular music collection of the RWC music database [66]. For each piece, RWC supplies high-quality MIDI-audio pairs that globally correspond to each other. To obtain a ground-truth alignment for each MIDI-audio pair we employ a high-resolution global synchronization approach (described in more detail in Chapter 7) and manually check the results for errors.

To simulate typical musical and structural differences between the two versions, we severely distort and modify the MIDI versions as follows. Firstly, we temporally distort each MIDI file by locally speeding up or slowing down the MIDI up to a random amount between ±50%. In particular, we change the tempo continuously within segments of 20 seconds of length, and add abrupt changes at segment boundaries to simulate musical tempo changes (ritardandi, accelerandi, fermata). Secondly, we structurally modify each MIDI file by replacing several MIDI segments (each having a length of 30 to 40 seconds) by concatenations of short 2 second snippets taken from random positions within the same MIDI file. In doing so, the length of each segment remains the same. These modified segments do not correspond to any segment in the audio anymore. However, because they are taken from the same piece, the snippets are likely to be harmonically related to the replaced content. Here, the idea is to simulate a kind of improvisation that fits into the harmonic context of the piece, but that is understood as musically different between the audio and the MIDI version (similar to the differences in $\mathrm{A}(75, 83)$ and $\mathrm{M}(99, 107)$, discussed in Chapter 4). Finally, we employ the ground-truth alignment between the original MIDI and the audio. Keeping track of the MIDI modifications, we derive a ground-truth alignment between the modified MIDI and the audio, in the following referred to as $\mathcal{A}^*$. To present even more challenges to the alignment approaches, we employ a second dataset with more *strongly modified* MIDI versions. Here, we not only distort and replace randomly chosen MIDI segments as described above, but insert additional MIDI snippet segments. These additional structural modifications make the synchronization task even harder.

For a given modified MIDI-audio pair, let $\mathcal{A}$ denote an alignment obtained using one of the synchronization strategies described above. To compare $\mathcal{A}$ with the ground-truth alignment $\mathcal{A}^*$, we introduce a quality measure that is based on precision and recall values, while allowing some deviation controlled by a given tolerance parameter $\varepsilon > 0$. The precision of $\mathcal{A}$ with respect to $\mathcal{A}^*$ is defined by

$$\mathrm{P}(\mathcal{A}) = \frac{|\{\gamma \in \mathcal{A} \,|\, \exists \gamma^* \in \mathcal{A}^* : ||\gamma - \gamma^*||_2 \leq \varepsilon\}|}{|\mathcal{A}|}$$

and the recall of $\mathcal{A}$ with respect to $\mathcal{A}^*$ is defined by

$$\mathrm{R}(\mathcal{A}) = \frac{|\{\gamma^* \in \mathcal{A}^* \,|\, \exists \gamma \in \mathcal{A} : ||\gamma - \gamma^*||_2 \leq \varepsilon\}|}{|\mathcal{A}^*|}.$$

Here, $||\gamma - \gamma^*||_2$ denotes the Euclidean norm between the elements $\gamma, \gamma^* \in [1:N] \times [1:M]$. In the experiments, a tolerance parameter ε corresponding to one second was used. Finally,

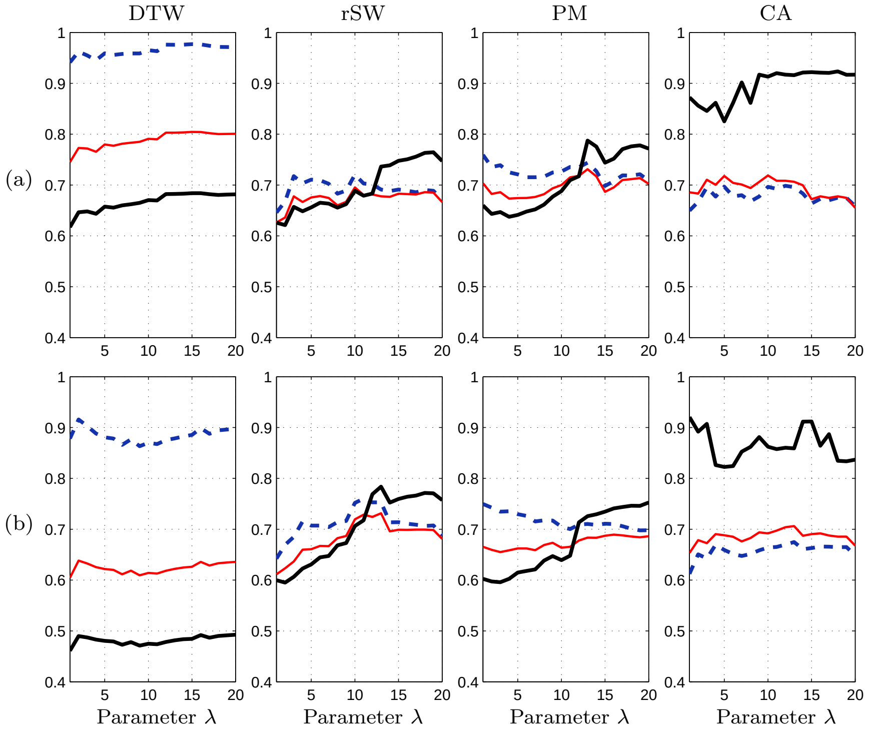


Figure 5.2. Effect of the smoothing parameter λ on the alignment accuracy of the DTW, rSW, PM, and CA procedures leaving the remaining parameters fixed ($\tau = 0.2$, $\gamma = 1$). Horizontal axis: λ. Vertical axis: Precision (bold black), F-measure (red), Recall (dashed blue). **(a)** Results using modified MIDI-audio pairs. **(b)** Results using strongly modified MIDI-audio pairs.

the F-measure is defined by

$$\mathrm{F}(\mathcal{A}) := \frac{2\mathrm{P}(\mathcal{A})\mathrm{R}(\mathcal{A})}{\mathrm{P}(\mathcal{A}) + \mathrm{R}(\mathcal{A})}.$$

5.3 Experiments

In a first experiment, we investigate the influence of the smoothing parameter λ on the performance of dynamic time warping (DTW), the recursive variant of the Smith-Waterman approach (rSW), partial matching (PM), and the proposed consistency alignment (CA). The parameter specifies the number of consecutive features taken into account for the smoothing. On the one hand, increasing λ emphasizes the structural properties of a cost matrix as discussed in Section 4.1 and is often a requirement to yield an overall robust synchronization result. On the other hand, smoothing can lead to a gradual loss of temporal accuracy in the alignment.

Figure 5.2 shows the average precision (bold black), recall (dashed blue), and F-measure (red) for all four alignment procedures using increasing values for λ in combination with fixed values for the other parameters ($\tau = 0.2$, $\gamma = 1$). Here, we used the modified MIDI-audio pairs in Figure 5.2a and the strongly modified pairs in Figure 5.2b. For

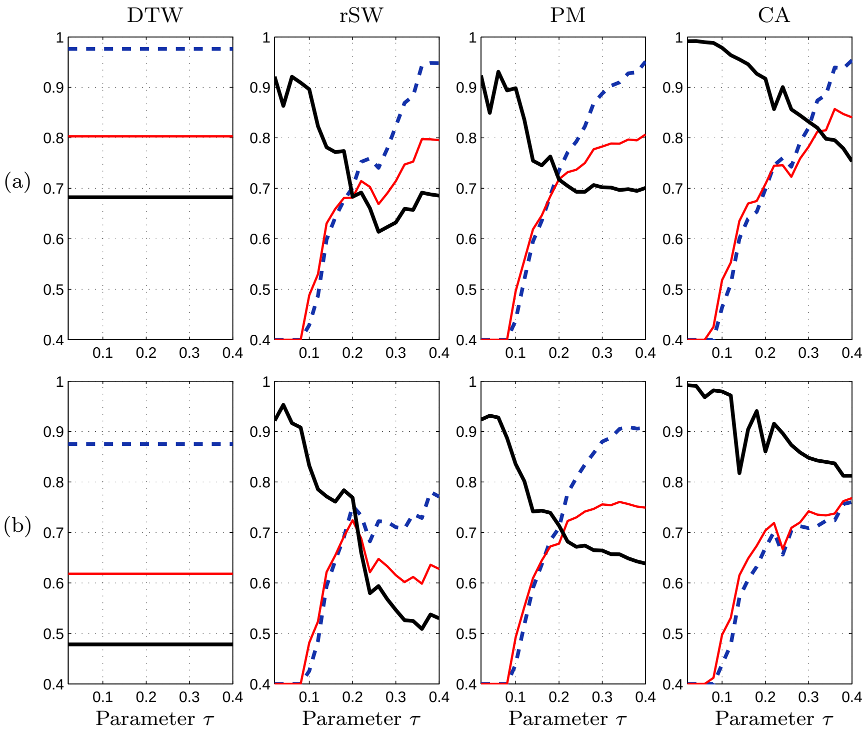


Figure 5.3. Effect of the threshold parameter τ on the alignment accuracy of the DTW, rSW, PM, and CA procedures leaving the remaining parameters fixed ($\lambda = 12$, $\gamma = 1$). Horizontal axis: τ. Vertical axis: Precision (bold black), F-measure (red), Recall (dashed blue). **(a)** Results using modified MIDI-audio pairs. **(b)** Results using strongly modified MIDI-audio pairs.

computational reasons, we computed the average only over a subset of ten classical and ten pop pieces from the original dataset. Here, looking at the results for DTW, rSW, and PM reveals that increasing λ leads to a higher precision. This indicates an enhanced robustness for all three procedures. However, if the smoothing is applied strongly the average recall slightly drops indicating the gradual loss of temporal accuracy. Furthermore, the DTW procedure only yields a rather low average precision for the strongly modified MIDI-audio pairs. Here, the reason is the boundary condition forcing DTW to align both versions as a whole, even if there are locally no musically meaningful correspondences. Still, DTW offers a very high recall value meaning that the correct alignment is often a true subset of the DTW alignment. This property is exploited by the consistency alignment which is often able to extract the correct parts of the DTW alignment thus yielding a very high overall precision. Looking at the CA results reveals that the proposed procedure yields a high precision with competitive F-measure and recall values for $\lambda \in [9\colon 15]$. In the following, we set $\lambda = 12$ which corresponds to 6 seconds using a feature rate of 2 Hz.

In a second experiment, we analyze the role of the threshold parameter τ. This parameter controls which cells in the cost matrix C become positive score entries in the matrices S and $S_{\geq 0}$, see Section 4.4. Figure 5.3 shows the results for varying τ while fixing the other parameters ($\lambda = 12$, $\gamma = 1$). Apart from that, the same experimental setup is used as in the previous experiment. Note, that the DTW procedure does not dependent on τ thus its results are constant in Figure 5.3. As the experiment shows, using very small values for

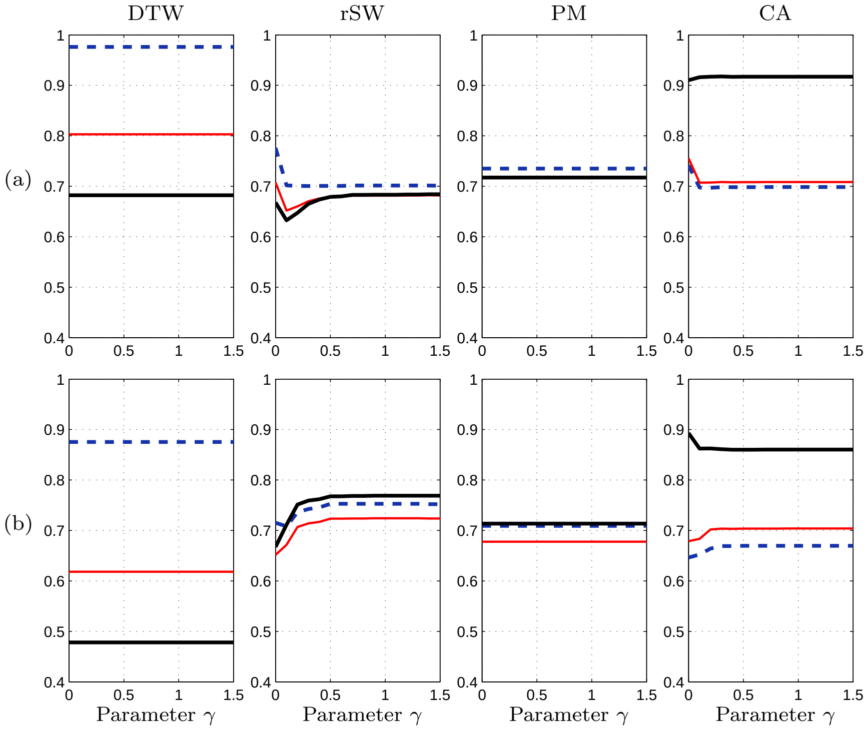


Figure 5.4. Effect of the parameter γ on the alignment accuracy of the DTW, rSW, PM, and CA procedures leaving the remaining parameters fixed (λ=12, τ=0.2). Horizontal axis: γ. Vertical axis: Precision (bold black), F-measure (red), Recall (dashed blue). **(a)** modified data. **(b)** strongly modified data. The DTW and PM procedures are not affected by this parameter.

τ, only very similar feature sequences are aligned and both the rSW and PM procedures are able to produce alignments with a high precision. However, this is only possible at the cost of having a very low recall as many correct alignment paths are missed. The break-even point for both procedures is near 0.2. For this value, the proposed consistency alignment yields a recall similar to rSW and PM but the precision is significantly higher. Overall, since the increase in F-measure is noticeable for all procedures up until 0.2 and diminishes beyond, we use $\tau = 0.2$ in the following. This value was also found to deliver reasonable results in the context of audio matching [128].

In a third experiment, we inspect the influence of the gap-penalty parameter γ. This parameter controls the fragmentation level of the alignment resulting from rSW. Here, we found that the influence of this parameter is less significant compared to the other parameters. Still, the experiment indicated that using some penalty for the gaps is needed for rSW to yield a robust alignment in our scenario, see Figure 5.4. Here, choosing γ between 0.5 and 2 yielded very similar results. In the following, we set $\gamma = 1$.

In general, the consistency alignment could be computed using an arbitrary combination of alignment procedures. In a fourth experiment, we investigate the alignment accuracy for all possible combinations of the DTW, rSW, and PM procedures (Figure 5.5). All free parameters are fixed for the experiment ($\lambda = 12$, $\tau = 0.2$, $\gamma = 1$). A first interesting observation is that all three individual procedures (Figure 5.5a-c) only yield a rather low average precision thus none of them can guarantee a meaningful alignment on its own. Combining

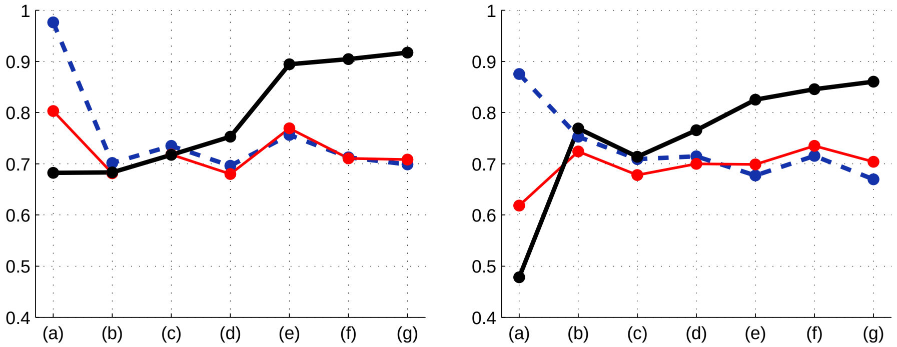


Figure 5.5. Effect of using different combinations of alignment procedures to compute the consistency alignment on the alignment accuracy. The parameter settings are fixed ($\lambda = 12$, $\tau = 0.2$, $\gamma = 1$). Vertical axis: Precision (bold black), F-measure (red), Recall (dashed blue). Horizontal axis: **(a)** DTW. **(b)** rSW. **(c)** PM. **(d)** rSW/PM. **(e)** DTW/PM. **(f)** DTW/rSW. **(g)** DTW/rSW/PM. **Left:** Results using modified MIDI-audio pairs. **Right:** Results using strongly modified MIDI-audio pairs.

(a) Classical Music modified

	P	R	F
DTW	0.68	0.99	0.81
rSW	0.84	0.90	0.87
PM	0.86	0.93	0.89
CA	0.91	0.90	0.90

(b) Classical Music strongly modified

	P	R	F
DTW	0.52	0.96	0.68
rSW	0.81	0.89	0.85
PM	0.83	0.93	0.87
CA	0.90	0.87	0.88

(c) Popular Music modified

	P	R	F
DTW	0.66	0.96	0.78
rSW	0.66	0.65	0.64
PM	0.70	0.71	0.70
CA	0.89	0.64	0.69

(d) Popular Music strongly modified

	P	R	F
DTW	0.47	0.87	0.61
rSW	0.62	0.63	0.61
PM	0.66	0.70	0.67
CA	0.89	0.58	0.65

Table 5.1. Average precision (P), recall (R) and F-measure (F) for the DTW, rSW, PM, and CA procedures using four different datasets with fixed parameters settings (λ=12, τ=0.2, and γ=1).

any two of the procedures results in a noticeable gain in precision (Figure 5.5d-f). In particular, including DTW is important for a high precision, see Figure 5.5e-f. Finally, the proposed combination of all three methods yields the highest precision (Figure 5.5g). As expected, the recall is slightly lower here, but is still on a competitive level.

In a final experiment, we determine the results for each alignment procedure separately for each available dataset. In Table 5.1, we consider the full classical and popular music datasets (120 recordings in total) using modified and strongly modified MIDI-audio pairs. Here, comparing the results for the modified and the strongly modified MIDI-audio pairs reveals that all procedures are able to cope quite well with the additional structural differences used in the strongly modified pairs. For example, the precision / recall for rSW slightly decrease from 0.84/0.9 (Table 5.1a) to 0.81/0.89 (Table 5.1b), respectively. Only DTW, again being forced to align the versions as a whole, shows a significant drop in precision. Furthermore, comparing the results for the classical and the popular music dataset

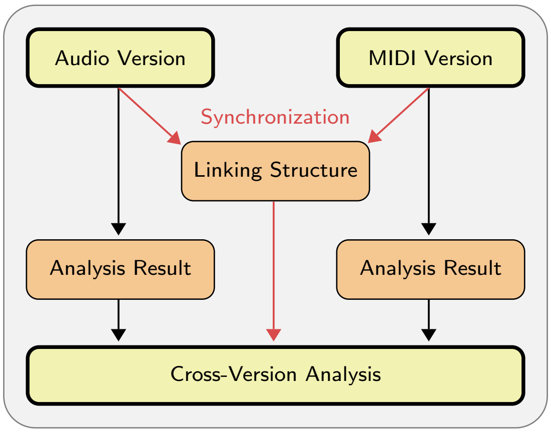


Figure 5.6. Cross-version music analysis based on synchronization techniques.

shows much lower values for the latter. Here, the underlying reason is that popular music tends to be highly repetitive. Combined with structural differences, this often leads to a higher confusion in the alignment. This is also reflected in Table 5.1 where most precision and recall values are significantly lower for the popular music dataset. For example, precision / recall for rSW decrease from 0.84/0.9 (Table 5.1a) to 0.66/0.65 (Table 5.1c), respectively. On the contrary, this is not the case for the consistency alignment which achieves a high precision of 0.89 also for the popular music dataset. Again, the recall is still on a competitive level.

In summary, as the experiments illustrate, the consistency alignment is able to deliver alignments with a high precision in combination with a competitive recall. Furthermore, the proposed late-fusion procedure is less dependent on the employed parameter settings or on the given dataset compared to the other individual alignment procedures. In the next section, we demonstrate the applicability of the proposed method in the context of harmonic analysis.

5.4 Application: Cross Version Harmonic Analysis

A musical work can be described in various ways using different representations. Symbolic formats (e. g., MusicXML, MIDI, Lilypond) conventionally describe a piece of music by specifying important musical parameters like pitch, rhythm, and dynamics. Interpreting these parameters as part of a musical performance leads to an acoustical representation that can be described by audio formats encoding the physical properties of sound (e. g., WAV, MP3). Depending on the type of representation, some musical properties are directly accessible while others may be implicit or even absent. For example, extracting pitch information from a MIDI file is straightforward, while extracting the same information from an audio file is a nontrivial task. On the other hand, while timbre and other complex musical properties are richly represented in an audio recording, the corresponding options in a MIDI file are very limited. Such differences between music representations allow for conceptually very different approaches to higher-level music analysis tasks such as melody extraction or structure analysis. Typically, each approach has intrinsic domain-specific strengths and weaknesses.

The main conceptual contribution of this section is the idea of a *cross-version analysis* for comparing and/or combining analysis results from different domains. The main idea is to employ the music synchronization techniques developed in this chapter to temporally align music representations across the different domains (see Figure 5.6). In general, a cross-version approach presents many varied opportunities to compare methods across different domains or to create methods that unite the domain-specific strengths while attenuating the weaknesses. This section presents an instance of such a cross-version analysis procedure, considering the task of automated chord labeling. Here, the objective is to induce the harmonic structure of a piece of music. The output of a chord labeling process is a sequence of chord labels with time stamps, either in musical time (i. e., in bars and beats) or in physical time measured in seconds. Because chord progressions describe the structure of a piece in a very musical and compact way, they often form the basis of musicological analyses and further automatic music processing applications. In particular, we demonstrate the cross-version approach by evaluating two state-of-the-art MIDI-based chord labelers using a ground truth originally created for audio recordings. Using the synchronization techniques, we can compare chord labels obtained from different procedures (automated or manual) and from different music representations (MIDI or audio). Having a unified view of the analysis results not only allows for an automated evaluation of the various analysis procedures but also deepens the understanding of the algorithms' behavior and the properties of the underlying music material. In the following, a brief overview of available chord labeling methods is given including a short description of two state-of-the-art symbolic domain methods. Then, we discuss the evaluation of these two methods while demonstrating how a cross-version visualization greatly deepens the understanding of the analysis results.

5.4.1 Automatic Chord Labeling

In the literature, most chord labeling procedures focus on chord labeling from audio data. Many of these procedures follow a two-step approach. In a first stage, chroma features (see Section 2) are extracted from an audio file in a frame-wise fashion. Then, a statistical model is applied to the sequence of chroma vectors that optimizes the match between specific chord templates and local sections of the chromagram. Typical statistical models applied as part of this second stage are hidden Markov models [99, 168] or more general graphical models [110]. Additional modeling constraints or auxiliary information can further improve chord labeling accuracy. These include the prior identification of the fundamental frequency or root note of each chord before the chromagram is estimated [159], information about the metrical structure [141], information about the musical structure [107], or the musical context [110]. Current state of the art chord labeling programs from audio have reached an identification accuracy of up to 80% as measured by the time overlap between predicted and ground truth chord labels, see [113].

Several procedures have been proposed that make use of symbolic music data. Early models such as those by [192] and [112] were designed to perform music-theoretic harmonic analyses (roman numeral analyses) from symbolic music input. Identifying chords in harmonic context (key) was one component within these music-analytic procedures. Both Winograd's and Maxwell's procedures are rule-based and rely heavily on knowledge of Western music theory, designed for the use with Western art music. [176] proposed key identification, chord labeling, and harmonic analysis procedures from a similar perspective. These procedures were implemented by Sleator and Temperley as part of the Melisma Music Analyzer [170]. In [155], the authors presented a hidden Markov model that uses symbolic MIDI data as input and produces a harmonic analysis of a musical piece including key and roman numerals labeling. [162] describes a chord labeling system for MIDI guitar sequences that is based on the symbolic chord labeler proposed by [142]. However, to be applicable in a jazz or latin music context the chord labeling system in [162] is specifically designed for the recognition of more complex chords. Their procedure is based on a hybrid mixture of pattern-matching techniques, harmonic context rules, and rules based on stylistic knowledge, and the resulting system is thus somewhat specific to their chosen task.

Even in this very short literature summary, a trend becomes apparent, moving away from rule-based and style-specific chord labeling systems that use explicit, built-in expert knowledge, towards data-driven and statistical reasoning approaches that learn and adapt to musical data from arbitrary styles. In the following, we evaluate two current chord-labeling systems based on Bayesian statistical frameworks, which have proven to be very successful in many areas of computational music processing. The *Melisma* system [176] employs a combination of preference rules (pitch spelling and chord root identification, see [100]) and Bayesian methods (chord and key estimation, see [177]). The system can produce chord labels as well as a roman numeral analysis, an analysis describing the relation between a chord and the key of the segment the chord is part of. For the evaluation below we use the information about chord root, mode (major, minor, unspecified) and fifth (perfect, diminished, unspecified) as well as the onset and offset times. This leads to three possible chord classes, namely major, minor, and diminished.

In contrast to *Melisma*, the second approach evaluated in the following employs Bayesian techniques for all parts of the underlying harmony model [158]. After incorporating all relevant musical parameters into a single model, a Bayesian model selection strategy is employed to choose the most likely chord. Overall, six different chord classes are considered by this approach, namely major, minor, diminished, augmented, sus2, and sus4. We refer to this approach in the following as *RLM*.

5.4.2 Evaluation

Exploiting the availability of multiple versions of a given piece of music, we have suggested the general concept of a cross-version analysis for comparing and/or combining analysis results across the versions. We now exemplarily apply this concept in the context of harmony analysis. In particular, we automatically evaluate the two MIDI-based chord labelers *RLM* and *Melisma* from Section 5.4.1, whose performance has not been clear so far, since no ground truth labels have been available on a larger scale.

Experimental Setup

In the following evaluation we exploit the audio data chord annotations provided by Christopher Harte, who manually annotated all 180 songs of the 12 Beatles studio albums [71]. Harte's annotations are generally accepted as the de-facto standard for evaluating audio-based chord labeling methods. Transferring these annotations from the acoustic to the symbolic domain allows for an efficient reuse of the existing ground truth for the evaluations of symbolic chord labelers. Furthermore, having a common set of ground truth across all available musical domains presents a starting point to identify exactly those positions in a piece where a method relying on one music representation has the advantage over another method, and to investigate the underlying musical reasons.

The evaluation dataset consists of 112 songs out of the 180 songs. For these 112 songs not only an audio recording with annotated chord labels is available, but also a corresponding MIDI version. Given a MIDI file and a corresponding audio recording, we start our evaluation by computing a MIDI-audio alignment. Because the MIDI versions often differ significantly, at local level, from the audio recordings, we cannot simply employ global synchronization techniques. Therefore, we employ the proposed consistency alignment, as described in Section 5.1, which identifies those sections that can be aligned reliably. Using the linking information provided by the alignment, we compute for each MIDI beat the corresponding position in the audio version. Using this linking information, we then transfer the audio-based chord labels to the MIDI version. If more than one audio chord label exists in the audio segment associated with a MIDI beat, we simply choose the predominant chord label as MIDI annotation. As the result, we obtain a beat-wise chord label annotation for the MIDI version.

For our evaluation, we compare the transferred ground truth annotations to the automatically generated chord labels obtained from *Melisma* and *RLM* on the basis of the 12 major and the 12 minor chords. Therefore, using the interval comparison of the triad as used for MIREX 2010 [113], all ground truth chord labels are mapped to one of these 24 chords. Here, both a seventh chord and a major seventh chord are mapped to the corresponding major chord. However, augmented, diminished or other more complex chords cannot be reduced to either major or minor and therefore are omitted from the evaluation.

Visualization

Using synchronization techniques allows for visualizing different chord recognition results simultaneously for multiple versions. Such cross-version visualizations turn out to be a powerful tool for not only analyzing the chord label results but also for better understanding the underlying music material [92]. We introduce this visualization concept by means of a concrete example shown in Figure 5.7. Here, the chord labels generated by *Melisma* and *RLM* are visualized along with the transferred ground truth annotations using a common MIDI time axis given in beats (horizontal axis). The vertical axis represents the 24 major and minor chords, starting with the 12 major chords and continuing with the 12 minor chords. Associated with each beat is a black entry representing the ground truth chord label that we transferred to the MIDI files. For example, in Figure 5.7, a G major chord label is assigned to beat 50. The colored entries in the figure are used to indicate

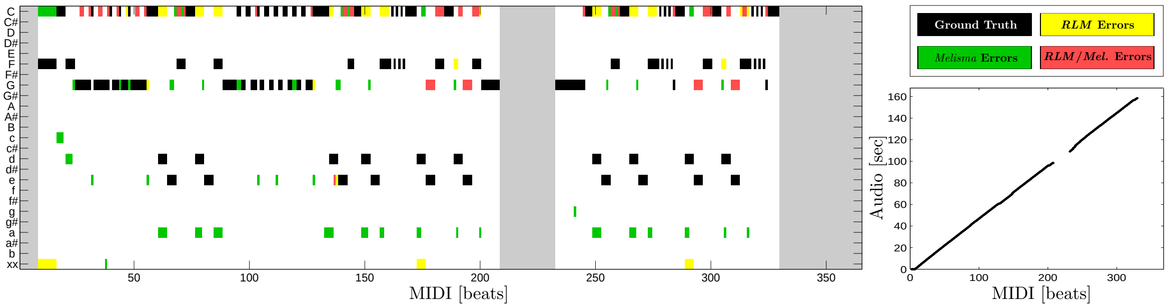


Figure 5.7. Cross-version chord evaluation for the song *Getting Better*. **Left:** Overlay of two MIDI-based chord labeling results (*Melisma* and *RLM*) and manually generated audio-based chord labels. **Right:** Consistency alignment (horizontal axis specifies MIDI time in beats and vertical axis specifies audio time in seconds).

where the two automatic chord labelers differ from the manual annotation. Here, yellow and green entries indicate that *RLM* and *Melisma* differ from the manual annotation, respectively. For example, in the beginning of the song the green entries show that *Melisma* detected a C major chord, while the ground truth specified an F major chord. If a chord labeler generated a chord label that cannot be reduced to either major or minor, then this is indicated by a colored entry in the 'xx' row. For example, in the beginning of the song *RLM* detected a complex chord corresponding to a yellow entry in the 'xx' row. Sometimes, both automatic chord labelers differ from the ground truth, but agree on the same chord label. Such consistent deviations from the ground truth are marked in red. An example can be found around beat 200, where both automatic chord labelers specify a C major chord instead of an F major chord in the ground truth. Furthermore, areas in the figure with a gray background indicate beats for that no ground truth is available. For example, in Figure 5.7, this can be observed between beat 210 and 230. Here, our consistency alignment, given on the right in the figure, shows that this section in the MIDI file could not be reliably aligned to a corresponding section in the audio. Furthermore, a ground truth annotation might also be unavailable for a beat if the chord label at that position is irreducible to major or minor—for example, if the chord label specifies an augmented chord.

Overall, our visualization allows for the identification of two different classes of inconsistencies. On the one hand, red entries in the visualization reveal positions, where the two chord labelers consistently differ from the ground truth. Here, the reason for the error may be of extrinsic or musical nature, independent of the specific chord labeler. On the other hand, yellow and green entries indicate intrinsic errors of the respective chord labeler. Thus, our visualization constitutes a useful tool to identify interesting or problematic passages in the audio recording.

Quantitative Evaluation

We now quantitatively evaluate the two MIDI-based chord labelers. Table 5.2 presents the results for all 112 pieces in our database. For each song, the precision values of *Melisma* and *RLM* are listed. Here, precision indicates the percentage of the manually annotated

		Prec	Prec
Piece	AC	*RLM*	*Melisma*
AcrossTheUniverse	72	82	79
ADayInTheLife	90	79	83
AHardDaysNight	99	90	76
AllIveGotToDo	96	96	93
AllMyLoving	97	90	61
AllYouNeedIsLove	84	88	85
AndILoveHer	83	77	75
AndYourBirdCanSing	98	80	73
AnnaGoToHim	91	86	78
AnotherGirl	97	98	60
AnyTimeAtAll	75	94	91
BabyItsYou	84	99	93
BabysInBlack	93	94	92
BabyYoureARichMan	76	85	74
BackInTheUSSR	97	85	44
Because	81	84	83
BeingForTheBenefitOfMrKit	92	81	75
Birthday	99	74	65
BlueJayWay	26	100	100
Boys	51	79	19
CantBuyMeLove	94	73	82
ComeTogether	0	–	–
DearPrudence	92	70	64
DigaPony	99	62	33
DoctorRobert	99	76	60
DontPassMeBy	95	96	96
DoYouWantToKnowASecret	70	94	72
DriveMyCar	75	83	65
EightDaysAWeek	99	92	74
EleanorRigby	98	72	78
EverybodysGotSomethingToH	85	69	60
EverybodysTryingToBeMyBab	95	71	85
FixingAHole	84	69	82
ForNoOne	90	87	80
GetBack	42	58	64
GettingBetter	83	60	52
Girl	83	92	92
GoodDaySunshine	82	85	55
GotToGetYouIntoMyLife	21	53	56
HelloGoodbye	65	84	78
Help	99	90	73
HelterSkelter	0	–	–
HereComesTheSun	96	95	80
HereThereAndEverywhere	91	85	82
HoneyDont	98	93	76
HoneyPie	93	80	42
IAmTheWalrus	95	81	55
IDontWantToSpoilTheParty	99	94	84
IfIFell	96	100	57
IllBeBack	86	72	70
IllCryInstead	99	85	89
IllFollowTheSun	98	77	84
ImALoser	0	–	–
ImHappyJustToDanceWithYou	0	–	–
INeedYou	97	99	91
InMyLife	97	90	75
ISawHerStandingThere	98	83	87
IShouldHaveKnownBetter	98	77	69
ItsOnlyLove	21	96	83
ItWontBeLong	93	95	69
IWannaBeYourMan	91	61	42
IWantYou	89	61	52
LetItBe	89	96	89
LovelyRita	90	90	80
LoveMeDo	42	56	65
LucyInTheSkyWithDiamonds	70	63	71
MagicalMysteryTour	87	88	65
MaxwellsSilverHammer	99	89	76
Michelle	91	69	67
Money	56	38	11
MotherNaturesSon	97	83	82
NoReply	80	77	81
NorwegianWoodThisBirdHasF	96	17	79
NowhereMan	83	94	90
ObLaDiObLaDa	98	100	88
OhDarling	95	19	76
PennyLane	98	93	77
PleasePleaseMe	92	90	74
PSILoveYou	91	91	92
Revolution1	79	80	53
RockAndRollMusic	98	91	66
RollOverBeethoven	72	85	72
RunForYourLife	72	98	84
SavoyTruffle	92	87	37
SgtPeppersLonelyHeartsClu	44	80	60
ShesLeavingHome	98	69	61
Something	89	82	81
StrawberryFieldsForever	0	–	–
Taxman	100	67	81
TellMeWhatYouSee	99	82	81
TheContinuingStoryOfBunga	90	97	87
TheEnd	96	74	40
TheFoolOnTheHill	77	92	60
TheLongAndWindingRoad	98	75	68
TheNightBefore	98	97	81
TheresAPlace	65	87	81
ThingsWeSaidToday	99	61	69
TicketToRide	90	76	90
TwistAndShout	86	96	82
WhatGoesOn	99	97	14
WhenIGetHome	57	97	79
WhenImSixtyFour	97	89	81
WhileMyGuitarGentlyWeeps	91	82	76
WhyDontWeDoItInTheRoad	100	97	90
WithALittleHelpFromMyFrie	99	91	79
YellowSubmarine	97	90	83
Yesterday	97	83	70
YouCantDoThat	75	59	59
YouNeverGiveMeYourMoney	78	92	74
YourMotherShouldKnow	77	88	82
YouveGotToHideYourLoveAwa	97	96	96
YouWontSeeMe	78	89	39
Average over all 112 songs	86	82	72

Table 5.2. Results of the cross-version chord evaluation for *RLM* and *Melisma*. The four columns indicate the song title, the alignment coverage (AC), as well as the precision (Prec) for the two methods.

beats correctly classified by the respective chord labeler. Also, the alignment coverage (AC), which specifies the percentage of the MIDI version that has been aligned to the respective audio version, is listed.

As can be seen from Table 5.2, the precision of *RLM*, averaged over all 112 songs, is 82%, whereas that of *Melisma* is only 72%. Using Bayesian model selection, *RLM* seems to be more data adaptive and performs better in our experiments than *Melisma*, depending on some hard-coded parameters. Furthermore, *Melisma* is tuned towards classical music, whereas *RLM* focuses on popular music, which might be advantageous with regard to the Beatles dataset.

Chapter 6

Structure-Oriented Partial Synchronization

In general, music synchronization is a challenging task as different versions of the same piece of music may exhibit significant variations in terms of tempo, instrumentation, and dynamics. However, as discussed in Chapter 4 and Chapter 5, structural differences have been found to be particularly problematic for most state-of-the-art methods. To complement the generally applicable partial synchronization method described in Chapter 5, we present in this chapter a new method, which focuses on structural variations between the versions to be aligned.

A basic idea to deal with structural differences is to combine methods from music structure analysis and music alignment. In a first step, one may partition the two versions to be aligned into musically meaningful segments. Here, one can use methods from automated structure analysis [27, 65, 127, 145, 157] to derive similarity clusters that represent the repetitive structure of the two versions. In a second step, the two versions can then be compared on the segment level with the objective to match musically corresponding passages. Finally, each pair of matched segments can be synchronized using global alignment strategies such as DTW. In theory, this seems to be a straightforward approach. In practice, however, one has to deal with several problems due to the variability of the underlying data. In particular, the automated extraction of the repetitive structure constitutes a delicate task in case the repetitions reveal significant differences in tempo, dynamics, or instrumentation. Furthermore, errors in the structural analysis may be aggravated in the subsequent segment-based matching step leading to strongly corrupted synchronization results.

The key idea of the method presented in this chapter is to perform a single, joint structure analysis for both versions to be aligned, which provides richer and more consistent structural data than in the case of two separate analyses. The resulting similarity clusters not only reveal the repetitions within and across the two versions, but also induce musically meaningful partial alignments between the two versions. Section 6.1 describes the proposed procedure for a joint structure analysis. Then, it is shown how the joint structure can be used for deriving a musically meaningful partial alignment between two audio recordings with structural differences, see Section 6.2. Furthermore, as described in

Section 6.3, the proposed procedure can be applied for automatic annotation of a given audio recording by partially available MIDI data.

6.1 Joint Structure Analysis

The problem of automated partial music synchronization has been introduced in [117], where the idea is to use the concept of path-constrained similarity matrices to enforce musically meaningful partial alignments. The approach presented here carries this idea even further by using cluster-constraint similarity matrices, thus enforcing structurally meaningful partial alignments.

The objective of a joint structure analysis is to extract the repetitive structure within and across two different music representations referring to the same piece of music. Each of the two versions can be an audio recording, a MIDI version, or a MusicXML file. The basic idea of how to couple the structure analysis of two versions is very simple. First, one converts both versions into common feature representations and concatenates the resulting feature sequences to form a single long feature sequence. Then, one performs a common structure analysis based on the long concatenated feature sequence. To make this strategy work, however, one has to deal with various problems. First, note that basically all available procedures for automated structure analysis have a computational complexity that is at least quadratic in the input length. Therefore, efficiency issues become crucial when considering a single concatenated feature sequence. Second, note that two different versions of the same piece often reveal significant local and global tempo differences. Recent approaches to structure analysis such as [65, 145, 157], however, are built upon the constant tempo assumption and cannot be used for a joint structure analysis. Allowing also tempo variations between repeating segments makes the structure analysis problem a much harder problem [27, 127]. We now summarize the approach used in this chapter closely following [127].

Given two music representations, we transform them into suitable feature sequences $V := (v^1, v^2, \dots, v^N)$ and $W := (w^1, w^2, \dots, w^M)$, respectively. To reduce different types of music data (audio, MIDI, MusicXML) to the same type of representation, we again make use of chroma-based audio features as discussed in Chapter 2. In the subsequent discussion, we employ CENS features with a temporal resolution of 1 Hz. Next, we define the sequence X of length $K := N + M$ by concatenating the sequences V and W:

$$X := (x^1, x^2, \dots, x^K) := (v^1, \dots, v^N, w^1, \dots, w^M).$$

Fixing a suitable local similarity measure — here, we use the inner product — the $(K \times K)$-*joint similarity matrix* $\mathcal{S}$ is defined by $\mathcal{S}(i, j) := \langle x^i, x^j \rangle$, $i, j \in [1 : K]$. Each tuple (i, j) is called a *cell* of the matrix. Furthermore, recall from Chapter 4 that a *path* in this matrix is a sequence $p = (p_1, \dots, p_L)$ with $p_\ell = (i_\ell, j_\ell) \in [1 : K]^2$, $\ell \in [1 : L]$, satisfying $p_{\ell+1} - p_\ell \in \Sigma$, where Σ denotes a set of admissible step sizes. In the following, we use $\Sigma = \Sigma_2 = \{(1, 1), (1, 2), (2, 1)\}$.

As an illustrative example, we consider two different audio recordings of the Aria of the Goldberg Variations BWV 988 by J.S. Bach, in the following referred to as *Bach example.*

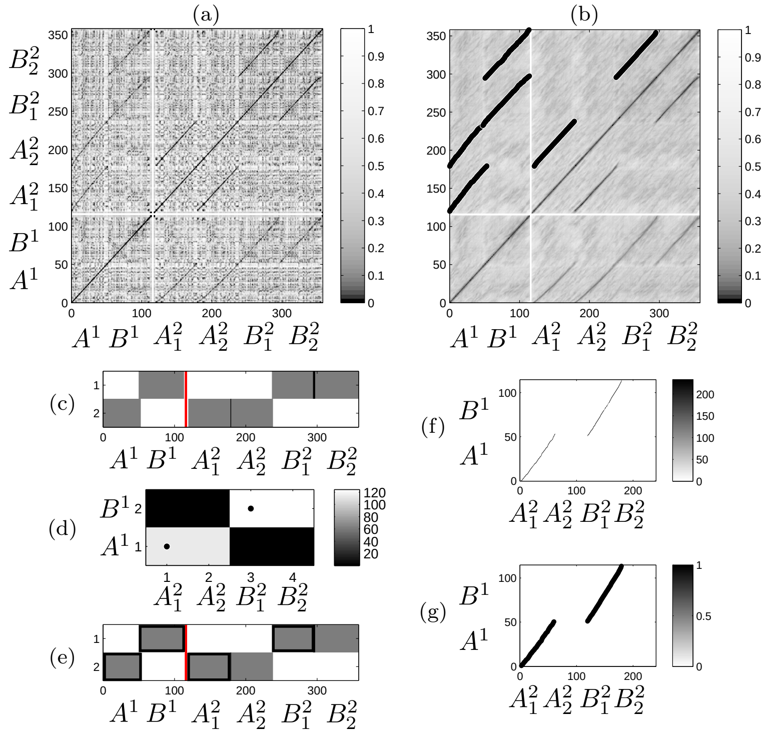


Figure 6.1. Joint structure analysis and partial synchronization for two structurally different versions of the Aria of the Goldberg Variations BWV 988 by J.S. Bach. The first version is played by G. Gould (musical form A^1B^1) and the second by M. Perahia (musical form $A^2_1A^2_2B^2_1B^2_2$). **(a)** Joint similarity matrix $\mathcal{S}$. **(b)** Enhanced matrix and extracted paths. **(c)** Similarity clusters. **(d)** Segment-based score matrix $\mathcal{M}$ and match (black dots). **(e)** Matched segments. **(f)** Matrix representation of matched segments. **(g)** Partial synchronization result.

The first version with a duration of 115 seconds is played by Glen Gould without repetitions (corresponding to the musical form A^1B^1) and the second version with a duration of 241 seconds is played by Murray Perahia with repetitions (corresponding to the musical form $A^2_1A^2_2B^2_1B^2_2$). For the feature sequences hold $N = 115$, $M = 241$, and $K = 356$. The resulting joint similarity matrix is shown in Figure 6.1a, where the boundaries between the two versions are indicated by white horizontal and vertical lines.

In the next step, the path structure is extracted from the joint similarity matrix. Here, the general principle is that each path of low cost running in a direction along the main diagonal (gradient $(1, 1)$) corresponds to a pair of similar feature subsequences. Note that relative tempo differences in similar segments are encoded by the gradient of the path (which is then in a neighborhood of $(1, 1)$). To ease the path extraction step, we enhance the path structure of $\mathcal{S}$ using the smoothing technique described in Section 4.1. The paths can then be extracted by a robust and efficient greedy strategy as described in [116, Chapter 7], see Figure 6.1b. Here, because of the symmetry of $\mathcal{S}$, one only has to consider the upper left part of $\mathcal{S}$. Furthermore, we prohibit paths to cross the boundaries between the two versions. As a result, each extracted path encodes a pair of musically similar segments, where each segment entirely belongs either to the first or to the second version. To determine the global repetitive structure, the proposed procedure incorporates a one-

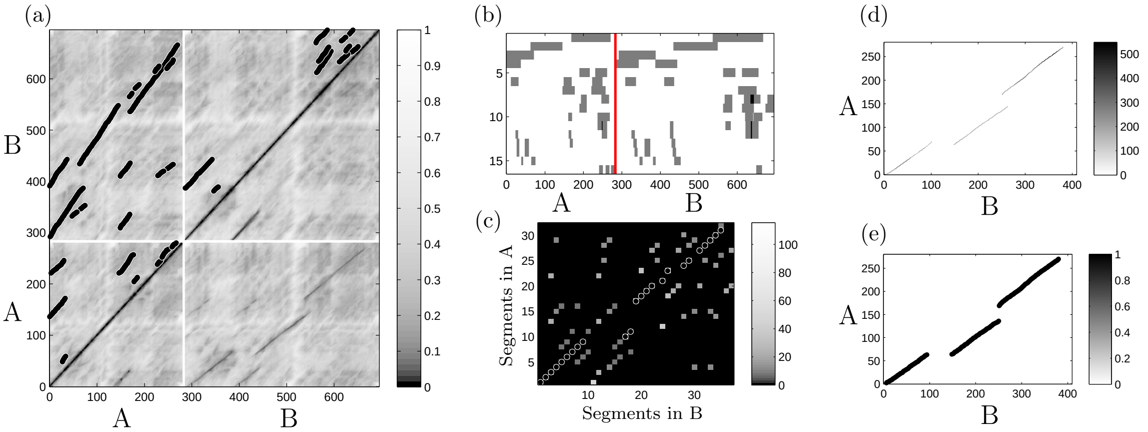


Figure 6.2. Joint structure analysis and partial synchronization for two structurally modified versions of Beethoven's Fifth Symphony Op. 67. The first version is a MIDI version and the second one an audio recording by Bernstein. **(a)** Enhanced joint similarity matrix and extracted paths. **(b)** Similarity clusters. **(c)** Segment-based score matrix $\mathcal{M}$ and match (indicated by black dots). **(d)** Matrix representation of matched segments. **(e)** Partial synchronization result.

step transitivity clustering procedure, which balances out the inconsistencies introduced by inaccurate and incorrect path extractions. For further details, see [116, 127].

Altogether, we obtain a set of similarity clusters. Each similarity cluster in turn consists of a set of pairwise similar segments encoding the repetitions of a segment within and across the two versions. Figure 6.1c shows the resulting set of similarity clusters for our Bach example. Both of the clusters consist of three segments, where the first cluster corresponds to the three B-parts B^1, B^2_1, and B^2_2 and the second cluster to the three A-parts A^1, A^2_1, and A^2_2. The joint analysis has several advantages compared to two separate analyses. First note that, since there are no repetitions in the first version, a separate structure analysis for the first version would not have yielded any structural information. Second, the similarity clusters of the joint structure analysis naturally induce musically meaningful partial alignments between the two versions. For example, the first cluster shows that B^1 may be aligned to B^2_1 or to B^2_2. Finally, note that the delicate path extraction step often results in inaccurate and fragmented paths. Because of the transitivity step, the joint clustering procedure balances out these flaws and compensates for missing parts to some extend by using joint information *across* the two versions.

On the downside, a joint structural analysis is computationally more expensive than two separate analyses. Therefore, in the structure analysis step, the proposed procedure employs a relatively low feature resolution of 1 Hz. This resolution may then be increased in the subsequent synchronization step (Section 6.2) and annotation application (Section 6.3). The current MATLAB implementation can easily deal with an overall length up to $K = 3000$ corresponding to more than forty minutes of music material. (In this case, the overall computation time adds up to 10-400 seconds with the path extraction step being the bottleneck, see [127]).

Another drawback of the joint analysis is that local inconsistencies across the two versions may cause an over-fragmentation of the music material. This may result in a large number of incomplete similarity clusters containing many short segments. As an example, we consider a MIDI version as well as a Bernstein audio recording of the first movement of Beethoven's Fifth Symphony Op. 67, which were both structurally modified by removing some sections. Figure 6.2a shows the enhanced joint similarity matrix and Figure 6.2b the set of joint similarity clusters. Note that some of the resulting 16 clusters contain semantically meaningless segments stemming from spuriously extracted path fragments. At this point, one could try to improve the overall structure result by a suitable postprocessing procedure. This itself constitutes a difficult research problem and is not in the scope of this chapter. Instead, we introduce a procedure for partial music alignment, which has some degree of robustness to inaccuracies and flaws in the previously extracted structural data.

6.2 Partial Synchronization

In [117], an approach for partial music synchronization has been described. Here, the idea is to first construct a path-constrained similarity matrix, which a priori constricts possible alignment paths to a semantically meaningful choice of admissible cells. Then, in a second step, a path-constrained alignment can be computed using standard matching procedures based on dynamic programming.

We now carry this idea even further by using the segments of the joint similarity clusters as constraining elements in the alignment step. To this end, we consider pairs of segments, where the two segments lie within the same similarity cluster and belong to different versions. More precisely, let $\mathcal{C} = \{C_1, \dots, C_R\}$ be the set of clusters obtained from the joint structure analysis. Each similarity cluster C_r, $r \in [1 : R]$, consists of a set of segments (i. e., subsequences of the concatenated feature sequence X). Let $\alpha \in C_r$ be such a segment. Then let $\mathcal{L}(\alpha)$ denote the length of α and $c(\alpha) := r$ the cluster affiliation. Recall that α either belongs to the first version (i. e., α is a subsequence of U) or to the second version (i. e., α is a subsequence of V). We now form two lists of segments. The first list $(\alpha_1, \dots, \alpha_I)$ consists of all those segments that are contained in some cluster of $\mathcal{C}$ and belong to the first version. The second list $(\beta_1, \dots, \beta_J)$ is defined similarly, where the segments now belong to the second version. Both lists are sorted according to the start positions of the segments. (In case two segments have the same start position, we break the tie by also considering the cluster affiliation.) We define a segment-based $I \times J$-score matrix $\mathcal{M}$ by

$$\mathcal{M}(i,j) := \begin{cases} \mathcal{L}(\alpha_i) + \mathcal{L}(\beta_j) & \text{for } c(\alpha_i) = c(\beta_j), \\ 0 & \text{otherwise,} \end{cases}$$

$i \in [1 : I]$, $j \in [1 : J]$. In other words, $\mathcal{M}(i,j)$ is positive if and only if α_i and β_j belong to the same similarity cluster. Furthermore, $\mathcal{M}(i,j)$ depends on the lengths of the two segments. Here, the idea is to favor long segments in the synchronization step. For an illustration, we consider the Bach example of Figure 6.1, where $(\alpha_1, \dots, \alpha_I) = (A^1, B^1)$ and $(\beta_1, \dots, \beta_J) = (A_1^2, A_2^2, B_1^2, B_2^2)$. The resulting matrix $\mathcal{M}$ is shown in Figure 6.1d. For another more complex example, see Figure 6.2c.

Next, we compute a segment-based match using matrix $\mathcal{M}$. Recall from Section 4.2 that a match is a sequence $\mathcal{M} = (\pi_1, \dots, \pi_L)$ with $\pi_\ell = (i_\ell, j_\ell) \in [1 : I] \times [1 : J]$ for $\ell \in [1 : L]$

satisfying $1 \leq i_1 < i_2 < \ldots < i_L \leq I$ and $1 \leq j_1 < j_2 < \ldots < j_L \leq J$. Note that a match with respect to $\mathcal{M}$ induces a partial assignment of segment pairs, where each segment is assigned to at most one other segment. The *score* of a match $\mathcal{M}$ with respect to $\mathcal{M}$ is then defined as $\sum_{\ell=1}^{L} \mathcal{M}(i_\ell, j_\ell)$. To compute a score-maximizing match, one can employ the partial matching procedure discussed in Section 4.5. In the Bach example, the score-maximizing match μ is given by $\mu = ((1,1),(2,3))$. In other words, the segment $\alpha_1 = A^1$ of the first version is assigned to segment $\beta_1 = A_1^2$ of the second version and $\alpha_2 = B^1$ is assigned to $\beta_3 = B_1^2$.

In principle, the score-maximizing match μ constitutes our partial music synchronization result. To make the procedure more robust to inaccuracies and to remove cluster redundancies, we further clean the synchronization result in a post-processing step. To this end, we convert the score-maximizing match μ into a sparse path-constrained similarity matrix $\mathcal{S}^{\text{path}}$ of size $N \times M$, where N and M are the lengths of the two feature sequences V and W, respectively. For each pair of matched segments, we compute an alignment path using a global synchronization algorithm [47], which will be discussed in more detail in Chapter 7. Each cell of such a path defines a non-zero entry of $\mathcal{S}^{\text{path}}$, where the entry is set to the length of the path (thus favoring long segments in the subsequent matching step). All other entries of the matrix $\mathcal{S}^{\text{path}}$ are set to zero. Figure 6.1f and Figure 6.2d show the resulting path-constrained similarity matrices for the Bach and Beethoven example, respectively. Finally, we apply the procedure as described in [117] using $\mathcal{S}^{\text{path}}$ (which is generally much sparser than the path-constrained similarity matrices as used in [117])to obtain a purified synchronization result, see Figure 6.1g and Figure 6.2e.

To evaluate the proposed synchronization procedure, experiments similar to [117] were conducted. In one experiment, synchronization pairs were formed consisting of two different versions of the same piece. Each pair consists either of an audio recording and a MIDI version or of two different audio recordings (interpreted by different musicians possibly in different instrumentations). After manually labeling musically meaningful sections in all versions, the pairs were modified by randomly removing or duplicating some of the labeled sections, see Figure 6.3. The partial synchronization result computed by the proposed algorithm was analyzed by means of its path components. A path component is said to be *correct* if it aligns corresponding musical sections. Similarly, a match is said to be *correct* if it covers (up to a certain tolerance) all semantically meaningful correspondences between the two versions (this information is given by the ground truth) and if all its path components are correct. The algorithm was tested on more than 387 different synchronization pairs resulting in a total number of 1080 path components. As a result, 89% of all path components and 71% of all matches were correct (using a tolerance of 3 seconds).

The results obtained using the proposed segment-based synchronization approach are qualitatively similar to those reported in [117]. However, there is one crucial difference in the two approaches. In [117], the authors use a combination of various ad-hoc criteria to construct a path-constrained similarity matrix as basis for their partial synchronization. In contrast, the proposed approach uses only the structural information in form of the joint similarity clusters to derive the partial alignment. Furthermore, the availability of structural information within and across the two versions allows for recovering missing relations based on suitable transitivity considerations. Thus, each improvement of the

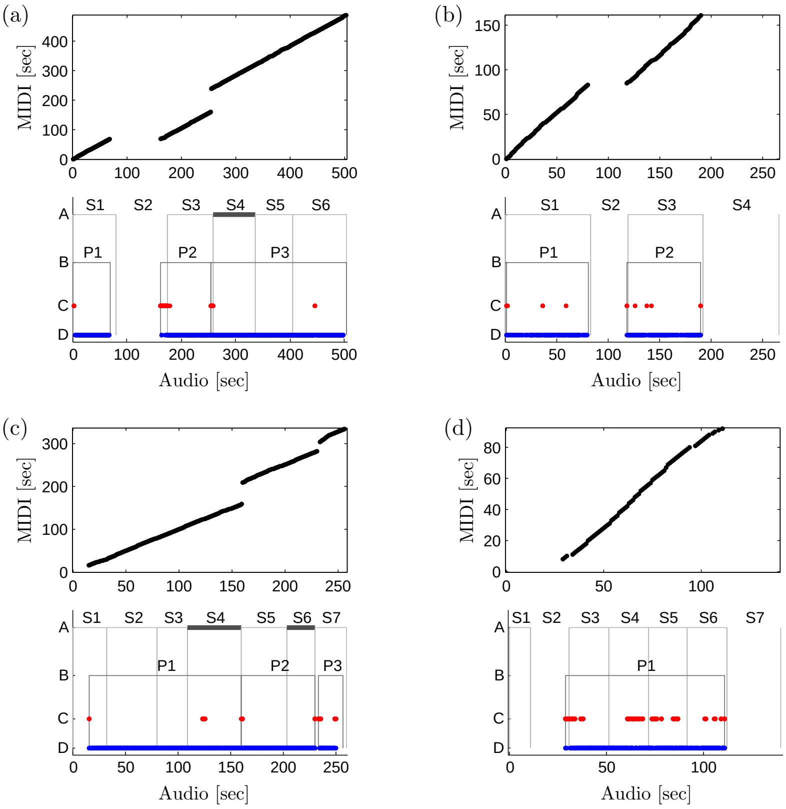


Figure 6.3. Partial synchronization results for various MIDI-audio synchronization pairs. The top figures show the final path components of the partial alignments and the bottom figures indicate the ground truth (Row A), the final annotations (Row B), and a classification into correct (Row D) and incorrect annotations (Row C), see text for additional explanations. The pieces are specified in Table 6.1. **(a)** Haydn (RWC C001), **(b)** Schubert (RWC C048, distorted), **(c)** Burke (P093), **(d)** Beatles ("Help!", distorted).

structure analysis will have a direct positive effect on the quality of the synchronization result, see

6.3 Audio Annotation

The synchronization of an audio recording and a corresponding MIDI version can be regarded as an automated annotation of the audio recording by means of the explicit note events given by the MIDI file. Often, MIDI versions are used as a kind of score-like symbolic representation of the underlying musical work, where redundant information such as repetitions are not encoded explicitly. This is a further setting with practical relevance where two versions to be aligned have a different repetitive structure (an audio version with repetitions and a score-like MIDI version without repetitions). In this setting, one can use the proposed segment-based partial synchronization to still obtain musically adequate audio annotations.

In the following, one of the experiments is summarized, which has been conducted on the basis of synchronization pairs consisting of structurally equivalent audio and MIDI versions.[1] In a first step, corresponding audio and MIDI versions were globally aligned using the synchronization procedure described in Chapter 7. These alignments were taken as ground truth for the audio annotation. Similar to the experiment in Section 6.2, musically meaningful sections of the MIDI versions were manually labeled and some of these sections were randomly removed or duplicated. Figure 6.3a illustrates this process by means of the first movement of Haydn's Symphony No. 94 (RWC C001). **Row A** of the bottom part shows the original six labeled sections S1 to S6 (warped according to the audio version). In the modification, S2 was removed (no line) and S4 was duplicated (thick line). Next, the modified MIDI was partially aligned with the original audio recording as described in Section 6.2. The resulting three path components of our Haydn example are shown in the top part of Figure 6.3a. Here, the vertical axis corresponds to the MIDI version and the horizontal axis to the audio version. Furthermore, **Row B** of the bottom part shows the projections of the three path components onto the audio axis resulting in the three segments P1, P2, and P3. These segments are aligned to segments in the MIDI thus being annotated by the corresponding MIDI events. Next, these partial annotations were compared with the ground truth annotations on the MIDI note event level. We say that an alignment of a note event to a physical time position of the audio version is correct in a *weak* (*strong*) sense, if there is a ground truth alignment of a note event of the same pitch (and, in the strong case, additionally lies in the same musical context by checking an entire neighborhood of MIDI notes) within a temporal tolerance of 100 ms. In our Haydn example, the weakly correct partial annotations are indicated in **Row D** and the incorrect annotations in **Row C**.

The other examples shown in Figure 6.3 give a representative impression of the overall annotation quality. Generally, the annotations are accurate—only at the segment boundaries there are some larger deviations. This is due to the employed path extraction procedure, which often results in “frayed” path endings. Here, one may improve the results by correcting the musical segment boundaries in a postprocessing step based on cues such as changes in timbre or dynamics. A more critical example (Beatles example) is shown in Figure 6.3d, where two sections (S2 and S7) are removed from the MIDI file and temporally distorted the remaining parts. In this example, the MIDI and audio version also exhibit significant differences on the feature level. As a result, an entire section (S1) has been left unannotated leading to a relatively poor rate of 77% (74%) of correctly annotated note events with respect to the weak (strong) criterion.

Finally, Table 6.1 shows further rates of correctly annotated note events for some representative examples. Additionally, the table also gives results for using significantly temporally distorted MIDI files (locally up to ±20%). Note that most rates only slightly decrease (e. g., for the Schubert piece, from 97% to 95% with respect to the weak criterion), which indicates the robustness of the proposed annotation procedure to local tempo differences. Further results as well as audio files of sonifications have been made available online[2].

[1] Most of the audio and MIDI files were taken from the RWC music database [66]. Note that for the classical pieces, the original RWC MIDI and RWC audio versions are not aligned.

[2] `http://www-mmdb.iai.uni-bonn.de/projects/partialSync/`

Composer	**Piece**	**RWC**	**Original**		**Distorted**	
			weak	**strong**	**weak**	**strong**
Haydn	Symph. No. 94, 1st Mov.	C001	98	97	97	95
Beethoven	Symph. Op. 67, 1st Mov.	C003	99	98	95	91
Beethoven	Sonata Op. 57, 1st Mov.	C028	99	99	98	96
Chopin	Etude Op. 10, No. 3	C031	93	93	93	92
Schubert	Op. 89, No. 5	C048	97	96	95	95
Burke	Sweet Dreams	P093	88	79	74	63
Beatles	Help!	—	97	96	77	74
Average			96	94	91	87

Table 6.1. Examples for automated MIDI-audio annotation (most of files are from the RWC music database [66]). The columns show the composer, the piece of music, the RWC identifier, as well as the annotation rate (in %) with respect to the weak and strong criterion for the original MIDI and some distorted MIDI.

Overall, in this chapter, we have introduced the strategy of performing a joint structural analysis to detect the repetitive structure within and across different versions of the same musical work. As a core component for realizing this concept, we have discussed a structure analysis procedure that can cope with relative tempo differences between repeating segments. Furthermore, it was demonstrated how joint structural information can be used to deal with structural variations in synchronization and annotation applications. One main message of this chapter is that automated music structure analysis is closely related to partial music alignment and annotation applications. Hence, improvements and extensions of current structure analysis procedures to deal with various kinds of variations is of fundamental importance for future research.

Chapter 7

High Resolution Music Synchronization

As we have seen throughout Part II of this thesis, automated music synchronization constitutes a challenging research field since one has to account for a multitude of musical aspects. For example, the music representations to be aligned might differ regarding the data format, (e.g., score, MIDI, PCM), the genre, (e.g., pop music, classical music, jazz), the instrumentation, (e.g., orchestra, piano, drums, voice), or in terms of the tempo, articulation, dynamics, structure, polyphony, or ornamentation. In the last two chapters, we introduced methods that robustly identify passages across various representations of a piece of music that can be aligned in a meaningful way. As a last step, these approaches aligned corresponding passages using classical global music synchronization methods. In this chapter, we will concentrate further on this final alignment step.

In the design of *global synchronization* algorithms, one has to deal with a delicate trade-off between robustness, temporal resolution, alignment quality, and computational complexity. For example, global music synchronization strategies based on chroma features [28] typically lead to robust alignment results with a reasonable synchronization quality. While such alignments might be accurate enough for browsing and retrieval applications, they are often too coarse to capture fine nuances in tempo and articulation as required in performance analysis [190] or audio editing [28]. To obtain alignment results of higher accuracy, other synchronization approaches propose the use of onset-based information for certain classes of music [130, 174], but suffer from a high computational complexity and a lack of robustness. Dixon et al. [31] describe an online approach to audio synchronization. Even though the proposed algorithm is very efficient, it cannot guarantee that the optimal DTW alignment is actually computed. Müller et al. [131] present a more robust, but very efficient offline approach, which is based on a multiscale strategy.

Focusing on the global synchronization scenario, we introduce in this chapter several strategies on various conceptual levels to increase the time resolution and quality of the synchronization result without sacrificing robustness and efficiency. First, we introduce a new class of audio features that inherit the robustness from chroma-based features and the temporal accuracy from onset-based features (Section 7.1). Then, in Section 7.2, we show how these features can be used within an efficient and robust multiscale synchro-

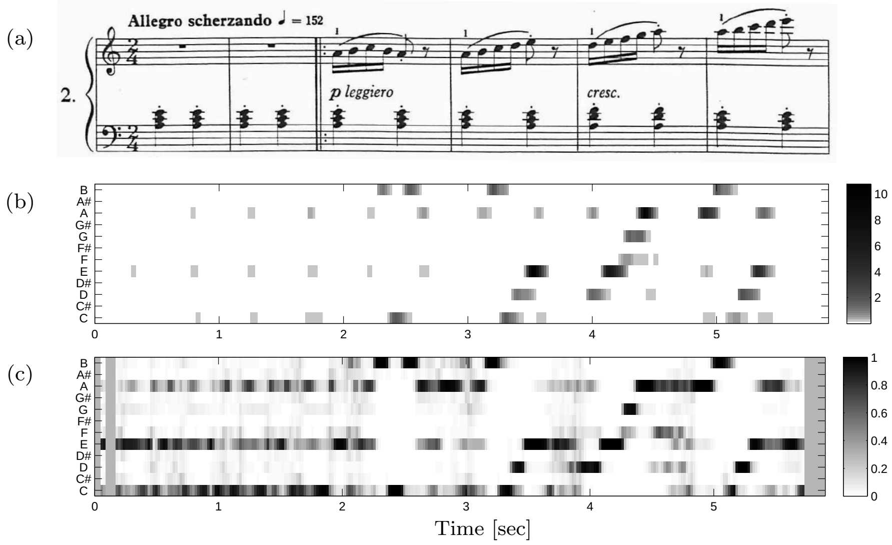


Figure 7.1. **(a)** First six measures of Burgmüller, Op. 100, Etude No. 2 (`Burg2`, see Table 7.1). **(b)** Chroma representation of a corresponding audio recording. Here, the feature resolution is 50 Hz (20 ms per feature vector). **(c)** Normalized chroma representation.

nization framework. Finally, for further improving the alignment quality, we introduce an interpolation technique that refines the given alignment path in some time consistent way (Section 7.3). In Section 7.4, various experiments based on polyphonic Western music are summarized, and we discuss the results indicating the respective improvements of the proposed refinement strategies. Even though we focus on MIDI-audio and audio-audio synchronization, similar techniques may be applied to other synchronization tasks as well. We summarize our findings in Section 7.5 and conclude this part of the thesis with a discussion of recently proposed global synchronization approaches in Section 7.6.

7.1 Robust and Accurate Audio Features

In this section, we introduce a new class of so-called DLNCO (decaying locally adaptive normalized chroma-based onset) features that indicate note onsets along with their chroma affiliation. These features possess a high temporal accuracy, yet being robust to variations in timbre and dynamics. With chroma features being describes in Chapter 2, we start with a summary of onset features. The novel DLNCO features are then described in Section 7.1.2.

In the following, the first six measures of the Etude No. 2, Op. 100, by Friedrich Burgmüller will serve us as our running example, see Figure 7.1a. For short, we will use the identifier `Burg2` to denote this piece, see Table 7.1. Figures 7.1b and 7.1c show a chroma representation and a normalized chroma representation, respectively, of an audio recording of `Burg2`. Because of their invariance, chroma-based features are well-suited for music synchronization leading to robust alignments even in the presence of significant variations between different versions of a musical work, see [75, 131].

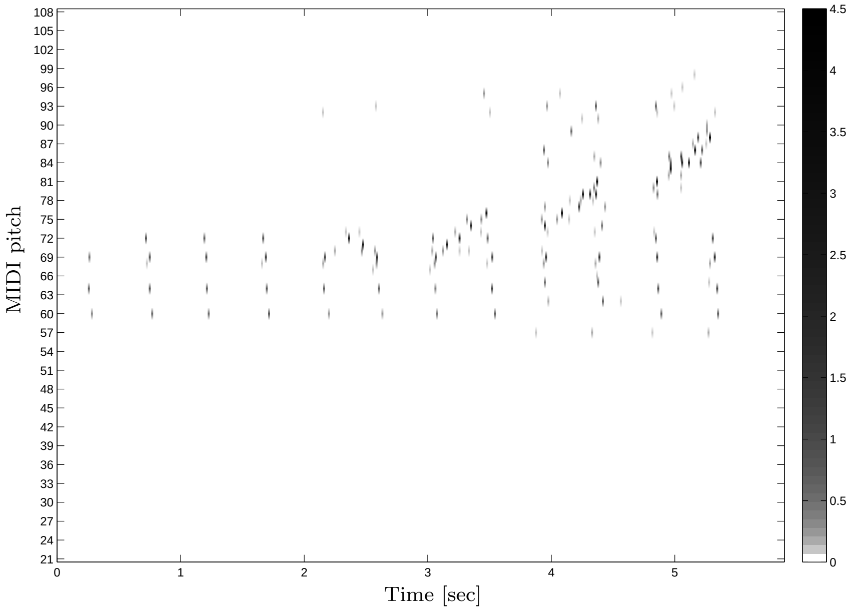


Figure 7.2. Onset representation of `Burg2`. Each rectangle represents an onset feature specified by pitch (here, indicated by the MIDI note numbers given by the vertical axis), by time position (given in seconds by the horizontal axis), and by a color-coded value that correspond to the height of the peak. Here, for the sake of visibility, a suitable logarithm of the value is shown.

7.1.1 Onset Features

We now describe a class of highly expressive audio features that indicate note onsets along with their respective pitch affiliation. For details, see also [116, 130]. Note that for many instruments such as the piano or the guitar, there is a sudden energy increase when playing a note (attack phase). This energy increase may not be significant relative to the entire signal's energy, since the generated sound may be masked by the remaining components of the signal. However, the energy increase relative to the spectral bands corresponding to the fundamental pitch and harmonics of the respective note may still be substantial. This observation motivates the following feature extraction procedure.

First the audio signal is decomposed into 88 subbands corresponding to the musical notes A0 to C8 (MIDI pitches $p = 21$ to $p = 108$) of the equal-tempered scale. This can be done by a high-quality multirate filter bank that properly separates adjacent notes as discussed in Section 2.1. Then, 88 local energy curves are computed by convolving each of the squared subbands with a suitable window function. Finally, for each energy curve the first-order difference is calculated (discrete derivative) and half-wave rectified (positive part of the function remains). The significant peaks of the resulting curves indicate positions of significant energy increase in the respective pitch subband. An onset feature is specified by the pitch of its subband and by the time position and height of the corresponding peak.

Figure 7.2 shows the resulting onset representation obtained for our running example `Burg2`. Note that the set of onset features is sparse while providing information of very high temporal accuracy. (The implementation used in this chapter has a maximal pitch dependent resolution of $2 - 10$ ms.) On the downside, the extraction of onset features is a delicate problem involving fragile operations such as differentiation and peak picking. Furthermore, the feature extraction only makes sense for music with clear onsets (e. g., piano music) and may yield no or faulty results for other music (e. g., soft violin music).

7.1.2 DLNCO Features

We now introduce a new class of features that combine the robustness of chroma features and the accuracy of onset features. The basic idea is to add up those onset features that belong to pitches of the same pitch class. To make this work, we first evenly split up the time axis into segments or frames of fixed length (In our experiments, we use a length of 20 ms). Then, for each pitch, we add up all onset features that lie within a segment. Note that due to the sparseness of the onset features, most segments do not contain an onset feature. Since the values of the onset features across different pitches may differ significantly, we take a suitable logarithm of the values, which accounts for the logarithmic sensation of sound intensity. For example, in our experiments, we use $\log(5000 \cdot v + 1)$ for an onset value v. Finally, for each segment, we add up the logarithmic values over all pitches that correspond to the same chroma. For example, adding up the logarithmic onset values that belong to the pitches A0,A1,. . .,A7 yields a value for the chroma A. The resulting 12-dimensional features will be referred to as *CO (chroma onset) features*, see Figure 7.3a.

The CO features are still very sensitive to local dynamic variations. As a consequence, onsets in passages played in piano may be marginal in comparison to onsets in passages played in forte. To compensate for this, one could simply normalize all non-zero CO feature vectors. However, this would also enhance small noisy onset features that are caused by mechanical noise, resonance, or beat effects thus leading to a useless representation, see Figure 7.3b. To circumvent this problem, we employ a locally adaptive normalization strategy. First, we compute the norm for each 12-dimensional CO feature vector resulting in a sequence of norms, see Figure 7.3c (blue curve). Then, for each time frame, we assign the local maxima of the sequence of norms over a time window that ranges one second to the left and one second to the right, see Figure 7.3c (red curve). Furthermore, we assign a positive threshold value to all those frames where the local maximum falls below that threshold. The resulting sequence of local maxima is used to normalize the CO features in a locally adaptive fashion. To this end, we simply divide the sequence of CO features by the sequence of local maxima in a point-wise fashion, see Figure 7.3d. The resulting features are referred to as *LNCO (locally adaptive normalized CO) features.* Intuitively, LNCO features account for the fact that onsets of low energy are less relevant in musical passages of high energy than in passages of low energy.

In summary, the octave identification makes LNCO features robust to variations in timbre. Furthermore, because of the locally adaptive normalization, LNCO features are invariant to variations in dynamics and exhibit significant onset values even in passages of low energy. Finally, the LNCO feature representation is sparse in the sense that most feature vectors are zero, while the non-zero vectors encode highly accurate temporal onset information.

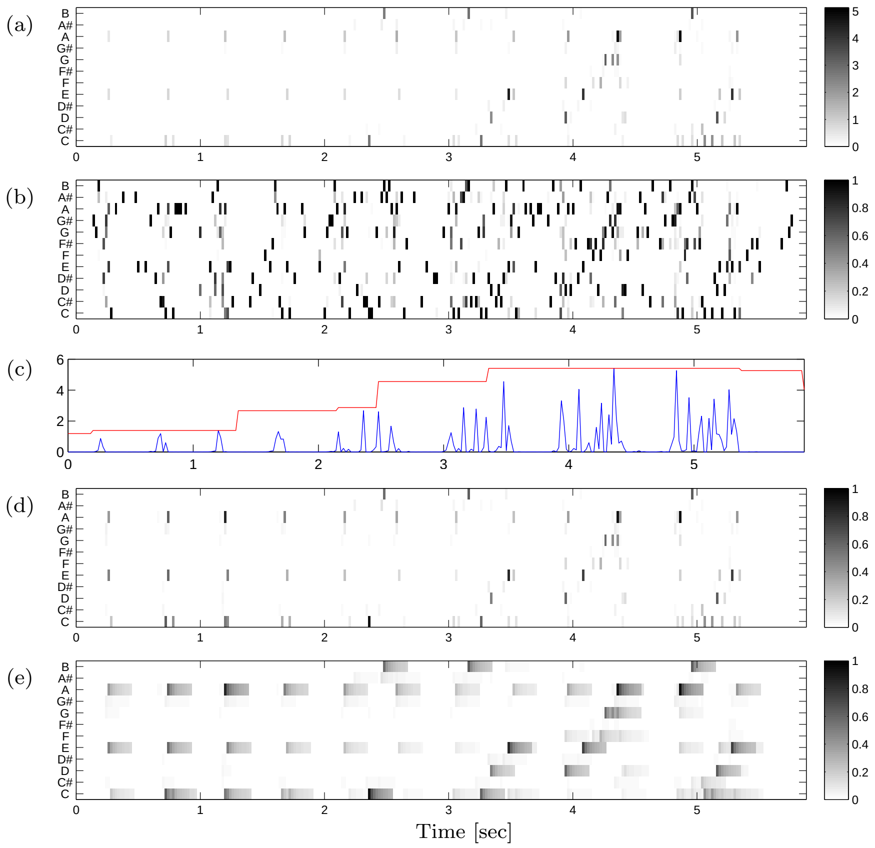


Figure 7.3. **(a)** Chroma onset (CO) features obtained from the onset representation of Figure 7.2. **(b)** Normalized CO features. **(c)** Sequence of norms of the CO features (blue, lower curve) and sequence of local maxima over a time window of ±1 second (red, upper curve). **(d)** Locally adaptive normalized CO (LNCO) features. **(e)** Decaying LNCO (DLNCO) features.

In view of synchronization applications, we further process the LNCO feature representation by introducing an additional temporal decay. To this end, each LNCO feature vector is copied n times (in the following experiments we use $n = 10$) and the copies are multiplied by decreasing positive weights starting with 1. Then, the n copies are arranged to form short sequences of n consecutive feature vectors of decreasing norm starting at the time position of the original vector. The overlay of all these decaying sequences results in a feature representation, which we refer to as *DLNCO (decaying LNCO) feature* representation, see figures 7.3e and 7.6a. The benefit of these additional temporal decays will become clear in the synchronization context, see Section 7.2.1. Note that in the DLNCO feature representation, one does not lose the temporal accuracy of the LNCO features—the onset positions still appear as sharp left edges in the decays. However, spurious double peaks, which appear in a close temporal neighborhood within a chroma band, are discarded. By introducing the decay, as we will see later, one loses sparseness while gaining robustness.

As a final remark of this section, we emphasize that the opposite variant of first computing chroma features and then computing onsets from the resulting chromagrams is not as successful as our strategy. As a first reason, note that the temporal resolution of the pitch energy curves is much higher ($2 - 10$ ms depending on the respective pitch) then for the

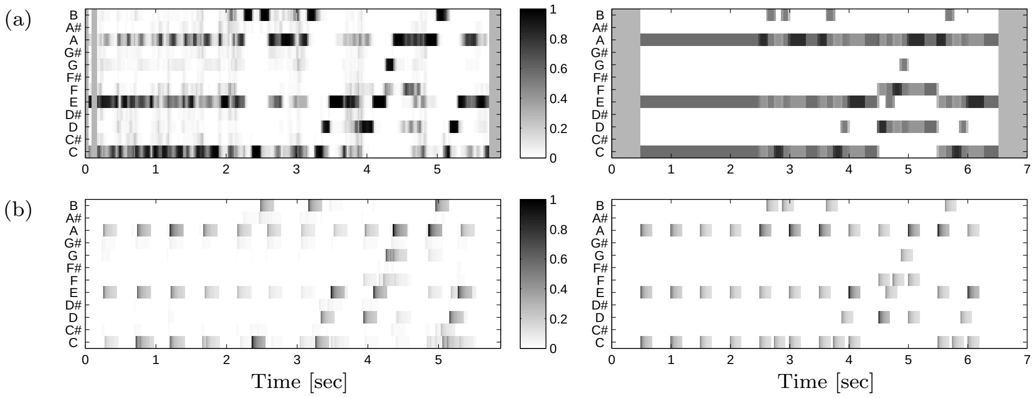


Figure 7.4. **(a)** Sequences of normalized chroma features for an audio version (left) and MIDI version (right) of `Burg2`. **(b)** Corresponding sequences of DLNCO features.

chroma features (where information across various pitches is combined at a common lower temporal resolution) thus yielding a higher accuracy. As a second reason, note that by first changing to a chroma representation one may already loose valuable onset information. For example, suppose there is a clear onset in the C3 pitch band and some smearing in the C4 pitch band. Then, the smearing may overlay the onset on the chroma level, which may result in missing the onset information. However, by first computing onsets for all pitches separately and then merging this information on the chroma level, the onset of the C3 pitch band will become clearly visible on the chroma level.

7.2 Synchronization Algorithm

In this section, we show how the proposed DLNCO features can be used to significantly improve the accuracy of previous chroma-based strategies without sacrificing robustness and efficiency. First, in Section 7.2.1, we introduce a combination of cost matrices that suitably captures harmonic as well as onset information. Then, in Section 7.2.2, we discuss how the new cost matrix can be plugged in an efficient multiscale music synchronization framework by using an additional alignment layer.

7.2.1 Local Cost Measures and Cost Matrices

In the following, we consider the case of MIDI-audio synchronization. Other cases such as audio-audio synchronization may be handled in the same fashion. Similar to most synchronization algorithms [28, 31, 75, 130, 131, 174, 181], we employ in the following a variant of the standard DTW algorithm as discussed in Section 4.3. For an illustration, we refer to Figure 7.5, which shows various cost matrices along with optimal global alignment paths.

The final synchronization result heavily depends on the type of features used to transform the music data streams and the local cost measure used to compare the features. We now introduce three different cost matrices, where the third one is a simple combination of the

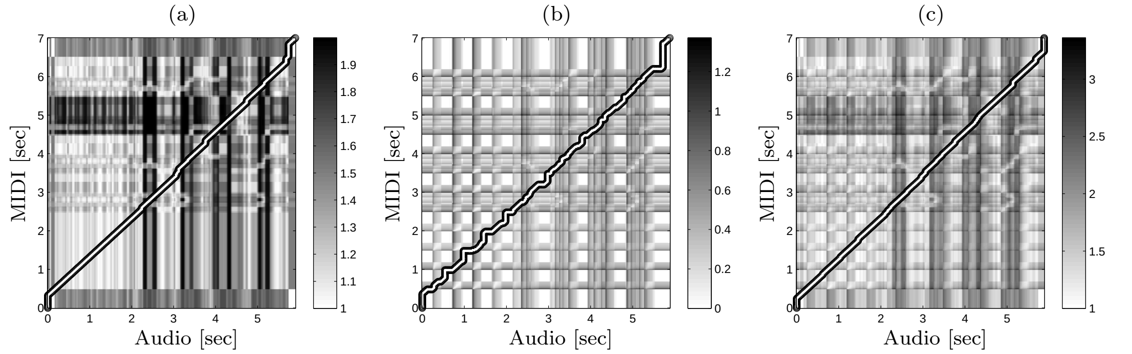


Figure 7.5. **(a)** Cost matrix $C_{\mathbf{chroma}}$ using normalized chroma features and the local cost measure $c_{\mathbf{chroma}}$. The two underlying feature sequences are shown Figure 7.4a. A cost-minimizing alignment path is indicated by the white line. **(b)** Cost matrix $C_{\mathbf{DLNCO}}$ with cost-minimizing alignment path using DLNCO features and $c_{\mathbf{DLNCO}}$. The two underlying feature sequences are shown Figure 7.4b. **(c)** Cost matrix $C = C_{\mathbf{chroma}} + C_{\mathbf{DLNCO}}$ and resulting cost-minimizing alignment path.

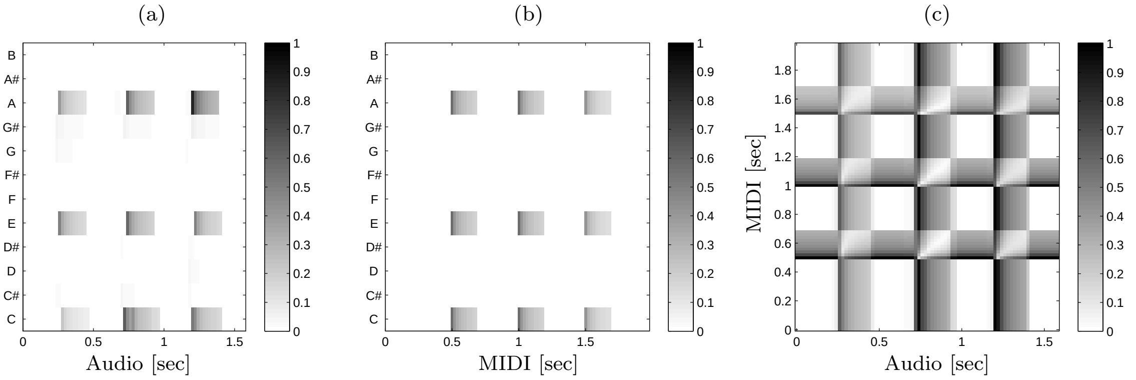


Figure 7.6. Illustration of the effect of the decay operation on the cost matrix level. A match of two onsets leads to a small corridor within the cost matrix that exhibits low costs and is tapered to the left (where the exact onsets occur). **(a)** Beginning of the audio DLNCO representation (left part of Figure 7.4b). **(b)** Beginning of the MIDI DLNCO representation (right part of Figure 7.4b). **(c)** Resulting section of $C_{\mathbf{DLNCO}}$, see Figure 7.5b.

first and second one. The first matrix is a conventional cost matrix based on normalized chroma features. Note that these features can be extracted from audio representations, as described in Section 2, as well as from MIDI representations, as suggested in [75]. Figure 7.4a shows normalized chroma representations for an audio recording and a MIDI version of `Burg2`, respectively. To compare two normalized chroma vectors v and w, we use the cost measure $c_{\mathbf{chroma}}(v,w) := 2 - \langle x, y\rangle$. Note that $\langle v, w\rangle$ is the cosine of the angle between v and w since the features are normalized. The offset 2 is introduced to favor diagonal directions in the DTW algorithm in regions of uniformly low cost, see [131] for a detailed explanation. The resulting cost matrix is denoted by $C_{\mathbf{chroma}}$, see Figure 7.5a.

The second cost matrix is based on DLNCO features as introduced in Section 7.1.2. Again, one can directly convert the MIDI version into a DLNCO representation by converting the MIDI note onsets into pitch onsets. Figure 7.4b shows DLNCO representations for an audio recording and a MIDI version of Burg2, respectively. To compare two DLNCO feature vectors, v and w we now use the Euclidean distance $c_{\mathbf{DLNCO}}(v, w) := \|v - w\|$. The resulting cost matrix is denoted by $C_{\mathbf{DLNCO}}$, see Figure 7.5b. At this point, we need to make some explanations. First, recall that each onset has been transformed into a short vector sequence of decaying norm. Using the Euclidean distance to compare two such decaying sequences leads to a diagonal corridor of low cost in $C_{\mathbf{DLNCO}}$ in the case that the directions (i. e., the relative chroma distributions) of the onset vectors are similar. This corridor is tapered to the lower left and starts at the precise time positions of the two onsets to be compared, see Figure 7.6c. Second, note that $C_{\mathbf{DLNCO}}$ reveals a grid like structure of an overall high cost, where each beginning of a corridor forms a small needle's eye of low cost. Third, sections in the feature sequences with no onsets lead to regions in $C_{\mathbf{DLNCO}}$ having zero cost. In other words, only significant events in the DLNCO feature sequences take effect on the cost matrix level. In summary, the structure of $C_{\mathbf{DLNCO}}$ regulates the course of a cost-minimizing alignment path in event-based regions to run through the needle's eyes of low cost. This leads to very accurate alignments at time positions with matching chroma onsets.

The two cost matrices $C_{\mathbf{chroma}}$ and $C_{\mathbf{DLNCO}}$ encode complementary information of the two music representations to be synchronized. The matrix $C_{\mathbf{chroma}}$ accounts for the rough harmonic flow of the two representations, whereas $C_{\mathbf{DLNCO}}$ exhibits matching chroma onsets. Forming the sum $C = C_{\mathbf{chroma}} + C_{\mathbf{DLNCO}}$ yields a cost matrix that accounts for both types of information. Note that in regions with no onsets, $C_{\mathbf{DLNCO}}$ is zero and the combined matrix C is dominated by $C_{\mathbf{chroma}}$. Contrary, in regions with significant onsets, C is dominated by $C_{\mathbf{DLNCO}}$, thus enforcing the cost-minimizing alignment path to run through the needle's eyes of low cost. Note that in a neighborhood of these eyes, the cost matrix $C_{\mathbf{chroma}}$ also reveals low costs due to the similar chroma distribution of the onsets. In summary, the component $C_{\mathbf{chroma}}$ regulates the overall course of the cost-minimizing alignment path and accounts for a robust synchronization, whereas the component $C_{\mathbf{DLNCO}}$ locally adjusts the alignment path and accounts for highly temporal accuracy.

7.2.2 Multiscale Implementation

Note that the time and memory complexity of DTW-based music synchronization linearly depends on the product $N \cdot M$ of the lengths N and M of the feature sequences to be aligned. For example, having a feature resolution of 20 ms and music data streams of 10 minutes of duration, results in $N = M = 30000$ making computations infeasible. To overcome this problem, we adapt an efficient multiscale DTW (MsDTW) approach as described in [131]. The idea is to calculate an alignment path in an iterative fashion by using multiple resolution levels going from coarse to fine. Here, the results of the coarser level are used to constrain the calculation on the finer levels, see Figure 7.7.

In a first step, we use the chroma-based MsDTW as described in [131]. In particular, we employ an efficient MsDTW implementation in C/C++ (used as a MATLAB DLL), which is based on three levels corresponding to a feature resolution of 1/3 Hz, 1 Hz, and 10 Hz, respectively. For example, our implementation needs less than a second (not including the feature extraction, which is linear in the length of the pieces) on a standard PC for

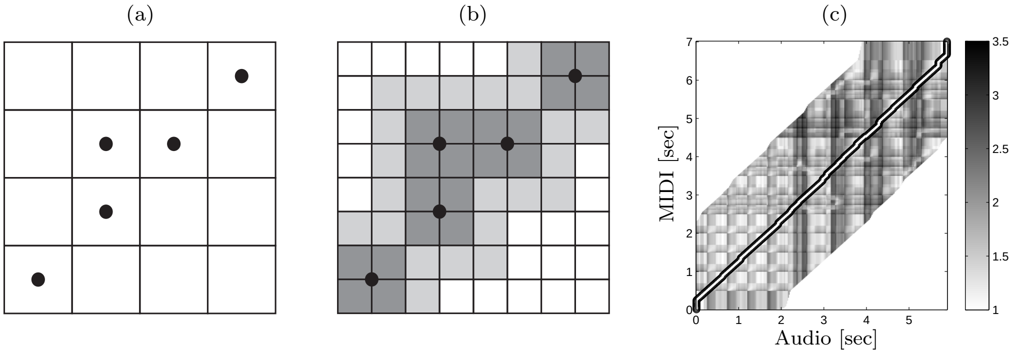


Figure 7.7. Illustration of multiscale DTW. **(a)** Optimal alignment path (black dots) computed on a coarse resolution level. **(b)** Projection of the alignment path onto a finer resolution level with constraint region (dark gray) and extended constraint region (light gray). **(c)** Constraint region for `Burg2`, cf. Figure 7.5c. The entries of the cost matrix are only computed within the constraint region. The resulting MsDTW alignment path indicated by the white line coincides with the DTW alignment path shown in Figure 7.5c.

synchronizing two music data streams each having a duration of 15 minutes of duration. The MsDTW synchronization is robust leading to reliable, but coarse alignments, which often reveal deviations of several hundreds of milliseconds.

To refine the synchronization result, we employ an additional alignment level corresponding to a feature resolution of 50 Hz (i. e., each feature corresponds to 20 ms). On this level, we use the cost matrix $C = C_{\mathbf{chroma}} + C_{\mathbf{DLNCO}}$ as described in Section 7.2.1. First, the resulting alignment path of the previous MsDTW method (corresponding to a 10 Hz feature resolution) is projected onto the 50 Hz resolution level. The projected path is used to define a tube-like constraint region, see Figure 7.7b. As before, the cost matrix C is only evaluated within this region, which leads to large savings if the region is small. However, note that the final alignment path is also restricted to this region, which may lead to incorrect alignment paths if the region is too small [131]. As our experiments showed, an extension of two seconds in all four directions (left, right, up, down) of the projected alignment path yields a good compromise between efficiency and robustness. Figure 7.7c shows the resulting extended constraint region for our running example `Burg2`. The relative savings with respect to memory requirements and running time of our overall multiscale procedure increases significantly with the length of the feature sequences to be aligned. For example, our procedure needs only around $3 \cdot 10^6$ of the total number of $15000^2 = 2.25 \cdot 10^8$ matrix entries for synchronizing two versions of a five minute piece, thus decreasing the memory requirements by a factor of 75. For a ten minute piece, this factor already amounts to 150. The relative savings for the running times are similar.

7.3 Resolution Refinement through Interpolation

A synchronization result is encoded by an alignment path, which assigns the elements of one feature sequence to the elements of the other feature sequence. Note that each feature refers to an entire analysis window, which corresponds to a certain time range rather than a single point in time. Therefore, an alignment path should be regarded as an assignment

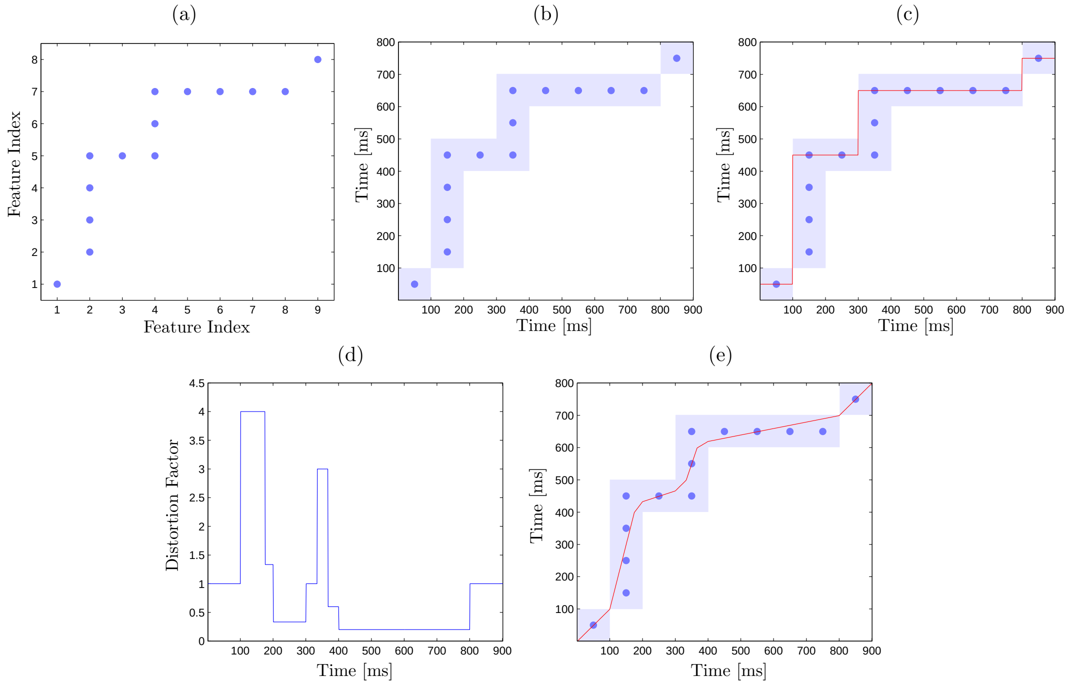


Figure 7.8. **(a)** Alignment path assigning elements of one feature sequence to elements of the other feature sequence. The elements are indexed by natural numbers. **(b)** Assignment of time ranges corresponding to the alignment path, where each feature corresponds to a time range of 100 ms. **(c)** Staircase interpolation path (red line). **(d)** Density function encoding the local distortions. **(e)** Smoothed and strictly monotonic interpolation path obtained by integration of the density function.

of certain time ranges. Furthermore, an alignment path may not be strictly monotonic in its components, i. e., a single element of one feature sequence may be assigned to several consecutive elements of the other feature sequence. This further increases the time ranges in the assignment. As illustration, consider Figure 7.8, where each feature corresponds to a time range of 100 ms. For example, the fifth element of the first sequence (vertical axis) is assigned to the second, third, and fourth element of the second sequence (horizontal axis), see Figure 7.8a. This corresponds to an assignment of the range between 400 and 500 ms with the range between 100 and 400 ms, see Figure 7.8b. One major problem of such an assignment is that the temporal resolution may not suffice for certain applications. For example, one may want to use the alignment result in order to temporally warp CD audio recordings, which are typically sampled at a rate of 44,1 kHz.

To increase the temporal resolution, one usually reverts to interpolation techniques. Many of the previous approaches are based on simple staircase paths as indicated by the red line of Figure 7.8c. However, such paths are not strictly monotonic and reveal abrupt directional changes leading to strong local temporal distortions. To avoid such distortions, one has to smooth the alignment path in such a way that both of its components are strictly monotonic increasing. To this end, Kovar et al. [93] fit a spline into the alignment path and enforce the strictness condition by suitably adjusting the control points of the splines. In the following, we introduce a novel strictly monotonic interpolation function that closely reflects the course of the original alignment path. Recall that the original alignment path

ID	Comp./Interp.	Piece	RWC ID	Instrument
Burg2	Burgmüller	Etude No. 2, Op. 100	–	piano
BachFuge	Bach	Fuge, C-Major, BWV 846	C025	piano
BeetApp	Beethoven	Op. 57, 1st Mov. (Appasionata)	C028	piano
ChopTris	Chopin	Etude Op. 10, No. 3 (Tristesse)	C031	piano
ChopBees	Chopin	Etude Op. 25, No. 2 (The Bees)	C032	piano
SchuRev	Schumann	Reverie (Träumerei)	C029	piano
BeetFifth	Beethoven	Op. 67, 1st Mov. (Fifth)	C003	orchestra
BorString	Borodin	String Quartett No. 2, 3rd Mov.	C015	strings
BrahDance	Brahms	Hungarian Dance No. 5	C022	orchestra
RimskiBee	Rimski-Korsakov	Flight of the Bumblebee	C044	flute/piano
SchubLind	Schubert	Op. 89, No. 5 (Der Lindenbaum)	C044	voice/piano
Jive	Nakamura	Jive	J001	piano
Entertain	HH Band	The Entertainer	J038	big band
Friction	Umitsuki Quartet	Friction	J041	sax,bass,perc.
Moving	Nagayama	Moving Round and Round	P031	electronic
Dreams	Burke	Sweet Dreams	P093	voice/guitar

Table 7.1. Pieces of music with identifier (ID) contained in our test database. For better reproduction of our experiments, we used pieces from the RWC music database [63, 66].

encodes an assignment of time ranges. The basic idea is that each assignment defines a local distortion factor, which is the proportion of the ranges' sizes. For example, the assignment of the range between 400 and 500 ms with the range between 100 and 400 ms, as discussed above, defines a local distortion factor of 1/3. Elaborating on this idea, one obtains a density function that encodes the local distortion factors. As an illustration, we refer to Figure 7.8d, which shows the resulting density function for the alignment path of Figure 7.8a. Then, the final interpolation path is obtained by integrating over the density function, see Figure 7.8e. Note that the resulting interpolation path is a smoothed and strictly monotonic version of the original alignment path. The continuous interpolation path can be used for arbitrary sampling rates. Furthermore, as we will see in Section 7.4, it also improves the final synchronization quality.

7.4 Experiments

In this section, we report on synchronization experiments, which have been conducted on a corpus of harmony-based Western music. To allow for a reproduction of these experiments, we use pieces from the RWC music database [63, 66]. In the following, we consider 16 representative pieces, which are listed in Table 7.1. These pieces are divided into three groups, where the first group consists of six classical piano pieces, the second group of five classical pieces of various instrumentations (full orchestra, strings, flute, voice), and the third group of five jazz pieces and pop songs. Note that for pure piano music, one typically has concise note attacks resulting in characteristic onset features. Contrary, such information is often missing in string or general orchestral music. To account for such differences, we report on the synchronization accuracy for each of the three groups separately.

To demonstrate the respective effect of the different refinement strategies on the final synchronization quality, we evaluate eight different synchronization procedures. The first procedure (MsDTW) is the MsDTW approach as described in [131], which works with a feature resolution of 10 Hz. The next three procedures are all refinements of the first procedure working with an additional alignment layer using a feature resolution of 50 Hz. In particular, we use in the second procedure (Chroma 20ms) normalized chroma features, in the third procedure (DLNCO) only the DLNCO features, and in the fourth procedure (Chroma+DLNCO) a combination of these features, see Section 7.2.1. Besides the simple staircase interpolation, we also refine each of these four procedures via smooth interpolation as discussed in Section 7.3. Table 7.2, which will be discussed later in detail, indicates the accuracy of the alignment results for each of the eight synchronization procedures.

To automatically determine the accuracy of our synchronization procedures, we use pairs of MIDI and audio versions for each of the 16 pieces listed in Table 7.1. Here, the audio versions were generated from the MIDI files using a high-quality synthesizer. Thus, for each synchronization pair, the note onset times in the MIDI file are perfectly aligned with the physical onset times in the respective audio recording (Only for our running example `Burg2`, a MIDI version was manually aligned with a corresponding real audio recording). In the first step of the evaluation process, the MIDI files were randomly distorted. To this end, the MIDI files were split up into N segments of equal length (in our experiment we used $N = 20$). Each segment was then stretched or compressed by a random factor within an allowed distortion range (in our experiments we used a range of ±30%). We refer to the resulting MIDI file as the *distorted MIDI file* in contrast to the original *annotation MIDI file*. In the second evaluation step, the distorted MIDI file and the associated audio recording were synchronized. The resulting alignment path was used to adjust the note onset times in the distorted MIDI file to obtain a third MIDI file referred to as *realigned MIDI file*. The accuracy of the synchronization result can now be determined by comparing the note onset times of the realigned MIDI file with the corresponding note onsets of the annotation MIDI file. Note that in the case of a perfect synchronization, the realigned MIDI file exactly coincides with the annotation MIDI file.

For each of the 16 pieces (Table 7.1) and for each of the eight different synchronization procedures, the corresponding realigned MIDI file was computed. One can then calculate the mean value, the standard deviation, as well as the maximal value over all note onset differences comparing the respective realigned MIDI file with the corresponding annotation MIDI file. Thus, for each piece, we obtain 24 statistical values, which are shown in Table 7.2. Actually, all experiments were repeated with five different randomly distorted MIDI files and all statistical values are averaged over these five repetitions. For example the value 73 in the first row of Table 7.2 means that for the piece `Burg2` the difference between the note onsets of the realigned MIDI file and the annotation MIDI file was in average 73 ms when using the MsDTW synchronization approach in combination with a staircase interpolation. In other words, the average synchronization error of this approach is 73 ms for `Burg2`.

We start the discussion of Table 7.2 by looking at the values for the first group consisting of six piano pieces. Looking at the averages of the statistical values over the six pieces, one can observe that the MsDTW procedure is clearly inferior to the other procedures. This is by no surprise, since the feature resolution of MsDTW is 100 ms compared to the resolution of 20 ms used in the other approaches. Nevertheless the standard deviation and maximal deviation of MsDTW is small relative to the mean value indicating the robustness of this approach. Using 20 ms chroma features, the average mean values decreases from 100 ms (MsDTW) to 51 ms (Chroma 20 ms). Using the combined features, this value fur-

		staircase			smooth		
ID	Procedure	mean	std	max	mean	std	max
Burg2	MsDTW	73	57	271	71	65	307
	Chroma 20ms	49	43	222	50	48	228
	DLNCO	31	20	94	21	17	73
	Chroma+DLNCO	**28**	**16**	**77**	**18**	**14**	**61**
BachFuge	MsDTW	97	55	319	55	41	223
	Chroma 20ms	34	34	564	27	33	554
	DLNCO	20	30	318	18	27	296
	Chroma+DLNCO	**18**	**15**	**96**	**14**	**12**	**81**
BeetApp	MsDTW	116	102	1197	77	94	1104
	Chroma 20ms	62	58	744	54	58	757
	DLNCO	136	318	2323	131	318	2335
	Chroma+DLNCO	**37**	**41**	**466**	**29**	**40**	**478**
ChopTris	MsDTW	115	76	1041	72	62	768
	Chroma 20ms	66	69	955	57	64	754
	DLNCO	30	68	1318	22	68	1305
	Chroma+DLNCO	**31**	**34**	**539**	**22**	**33**	**524**
ChopBees	MsDTW	108	79	865	59	71	817
	Chroma 20ms	41	49	664	30	47	625
	DLNCO	20	14	104	12	9	95
	Chroma+DLNCO	**22**	**24**	**366**	**13**	**21**	**355**
SchuRev	MsDTW	93	95	887	66	77	655
	Chroma 20ms	51	80	778	46	72	567
	DLNCO	98	261	1789	94	264	1841
	Chroma+DLNCO	**22**	**38**	**330**	**15**	**36**	**315**
Average over piano examples	MsDTW	100	77	763	67	68	646
	Chroma 20ms	51	56	655	44	54	581
	DLNCO	56	119	991	50	117	991
	Chroma+DLNCO	**26**	**28**	**312**	**19**	**26**	**302**
BeetFifth	MsDTW	194	124	1048	142	116	952
	Chroma 20ms	128	98	973	116	96	959
	DLNCO	254	338	2581	241	338	2568
	Chroma+DLNCO	**128**	**99**	**1144**	**116**	**98**	**1130**
BorString	MsDTW	157	110	738	118	106	734
	Chroma 20ms	88	68	584	79	68	576
	DLNCO	275	355	2252	268	356	2233
	Chroma+DLNCO	**91**	**57**	**682**	**82**	**56**	**675**
BrahDance	MsDTW	104	62	385	64	54	470
	Chroma 20ms	58	54	419	50	54	427
	DLNCO	31	52	567	26	52	556
	Chroma+DLNCO	**24**	**22**	**185**	**17**	**20**	**169**
RimskiBee	MsDTW	99	48	389	50	32	196
	Chroma 20ms	51	17	167	41	17	155
	DLNCO	31	23	183	22	19	160
	Chroma+DLNCO	**37**	**17**	**108**	**27**	**15**	**91**
SchubLind	MsDTW	124	73	743	78	59	549
	Chroma 20ms	66	57	718	55	50	509
	DLNCO	79	175	1227	70	173	1206
	Chroma+DLNCO	**41**	**36**	**406**	**31**	**34**	**387**
Average over various instrumentation examples	MsDTW	136	83	661	90	73	580
	Chroma 20ms	78	59	572	68	57	525
	DLNCO	134	189	1362	125	188	1345
	Chroma+DLNCO	**64**	**46**	**505**	**55**	**45**	**490**
Jive	MsDTW	97	105	949	58	93	850
	Chroma 20ms	44	61	686	34	59	668
	DLNCO	23	38	638	17	37	632
	Chroma+DLNCO	**22**	**18**	**154**	**14**	**15**	**158**
Entertain	MsDTW	100	67	579	66	58	492
	Chroma 20ms	52	44	407	45	46	414
	DLNCO	93	204	1899	85	204	1887
	Chroma+DLNCO	**40**	**65**	**899**	**31**	**64**	**889**
Friction	MsDTW	94	81	789	58	75	822
	Chroma 20ms	47	67	810	39	67	815
	DLNCO	44	120	2105	37	117	2106
	Chroma+DLNCO	**30**	**55**	**810**	**23**	**55**	**819**
Moving	MsDTW	114	76	497	76	64	473
	Chroma 20ms	77	51	336	68	50	343
	DLNCO	127	216	1443	124	217	1432
	Chroma+DLNCO	**53**	**45**	**284**	**46**	**43**	**275**
Dreams	MsDTW	136	105	659	115	106	674
	Chroma 20ms	97	94	702	91	95	673
	DLNCO	73	103	692	71	103	702
	Chroma+DLNCO	**43**	**57**	**429**	**40**	**58**	**434**
Average over jazz/pop examples	MsDTW	108	87	695	75	79	662
	Chroma 20ms	63	63	588	55	63	583
	DLNCO	72	136	1355	67	136	1352
	Chroma+DLNCO	**38**	**48**	**515**	**31**	**47**	**515**
Average over all examples	MsDTW	114	82	710	77	73	630
	Chroma 20ms	63	59	608	55	58	564
	DLNCO	85	146	1221	79	145	1214
	Chroma+DLNCO	**42**	**40**	**436**	**34**	**38**	**428**

Table 7.2. Alignment accuracy for eight different synchronization procedures (MsDTW, Chroma 20 ms, DLNCO, Chroma+DLNCO with staircase and smooth interpolation, respectively). The table shows for each of the eight procedures and for each of 16 pieces (Table 7.1) the mean value, the standard deviation, and the maximal value over all note onset difference of the respective realigned MIDI file and the corresponding annotation MIDI file. All values are given in milliseconds.

ther decreases to 26 ms (Chroma+DLNCO). Furthermore, using the smooth interpolation instead of the simple staircase interpolation further improves the accuracy, for example, from 100 ms to 67 ms (MsDTW) or from 26 ms to 19 ms (Chroma+DLNCO). Another interesting observation is that the pure DLNCO approach is sometimes much better (e. g. for `ChopBees`) but also sometimes much worse (e. g. for `BeetApp`) than the Chroma 20ms approach. This shows that the DLNCO features have the potential for delivering very accurate results but also suffer from a lack of robustness. It is the combination of the DLNCO features and chroma features which ensures robustness as well as accuracy of the overall synchronization procedure.

Next, we look at the group of the five classical pieces of various instrumentations. Note that for the pieces of this group, opposed to the piano pieces, one often has no clear note attacks leading to a much poorer quality of the onset features. As a consequence, the synchronization errors are on average higher than for the piano pieces. For example, the average mean error over the second group is 136 ms (MsDTW) and 134 ms (DLNCO) opposed to 100 ms (MsDTW) and 56 ms (DLNCO) for the first group. However, even in the case of missing onset information, the synchronization task is still accomplished in a robust way by means of the harmony-based chroma features. The idea of using the combined approach (Chroma+DLNCO) is that the resulting synchronization procedure is at least as robust and exact as the pure chroma-based approach (Chroma 20 ms). Table 7.2 demonstrates that this idea is realized by the implementation of the combined synchronization procedure. Similar results are obtained for the third group of jazz/pop examples, where the best results were also delivered by the combined approach (Chroma+DLNCO).

At this point, one may object that one typically obtains better absolute synchronization results for synthetic audio material (which was used to completely automate our evaluation) than for non-synthetic, real audio recordings. We therefore included also the real audio recording `Burg2`, which actually led to similar results as the synthesized examples. Furthermore, our experiments on the synthetic data are still meaningful in the relative sense by revealing relative performance differences between the various synchronization procedures. Finally, we also generated MIDI-audio alignments using real performances of the corresponding pieces (which are also contained in the RWC music database). These alignments were used to modify the original MIDI files to run synchronously to the audio recordings. Generating a stereo file with a synthesized version of the modified MIDI file in one channel and the audio recording in the other channel, we have acoustically examined the alignment results. The acoustic impression supports the evaluation results obtained from the synthetic data. The stereo files have been made available on a website[1].

For the experiments of Table 7.2, a distortion range of ±30% was used, which is motivated by the observation that the relative tempo difference between two real performances of the same piece mostly lies within this range. In a second experiment, we investigate the dependency of the final synchronization accuracy on the size of the allowed distortion range. To this end, the mean values of the synchronization error were calculated for each of the 16 pieces using different distortion ranges from ±10% to ±50%. Table 7.3 shows the resulting vales for two of the eight synchronization procedures described above, namely MsDTW and Chroma+DLNCO both post-processed with smooth interpolation. As one may expect, the mean error values increase with the allowed distortion range. For example,

[1] `http://www.mpi-inf.mpg.de/resources/MIR/SyncRWC60/`

		Distortion range				
ID	Procedure	±10%	±20%	±30%	±40%	±50%
Burg2	MsDTW	48	53	65	85	94
	Chroma+DLNCO	**15**	**16**	**19**	**17**	**22**
BachFuge	MsDTW	44	49	52	62	67
	Chroma+DLNCO	**11**	**12**	**13**	**15**	**15**
BeetApp	MsDTW	53	68	75	96	170
	Chroma+DLNCO	**22**	**25**	**29**	**36**	**98**
ChopTris	MsDTW	57	64	72	75	82
	Chroma+DLNCO	**18**	**19**	**21**	**22**	**29**
ChopBees	MsDTW	51	54	57	60	67
	Chroma+DLNCO	**11**	**12**	**13**	**14**	**18**
SchuRev	MsDTW	50	58	64	77	85
	Chroma+DLNCO	**11**	**14**	**12**	**13**	**22**
Average over piano examples	MsDTW	51	58	64	76	94
	Chroma+DLNCO	**15**	**16**	**18**	**20**	**34**
BeetFifth	MsDTW	119	126	141	143	184
	Chroma+DLNCO	**101**	**106**	**113**	**113**	**145**
BorString	MsDTW	86	97	109	118	153
	Chroma+DLNCO	**75**	**78**	**82**	**84**	**101**
BrahDance	MsDTW	52	58	66	70	81
	Chroma+DLNCO	**13**	**15**	**18**	**19**	**25**
RimskiBee	MsDTW	49	47	52	53	56
	Chroma+DLNCO	**25**	**26**	**26**	**28**	**28**
SchubLind	MsDTW	69	73	78	99	91
	Chroma+DLNCO	**28**	**28**	**31**	**35**	**35**
Average over various instrumentation examples	MsDTW	75	80	89	97	113
	Chroma+DLNCO	**48**	**51**	**54**	**56**	**67**
Jive	MsDTW	44	62	50	63	77
	Chroma+DLNCO	**12**	**13**	**14**	**14**	**15**
Entertain	MsDTW	47	53	62	78	94
	Chroma+DLNCO	**21**	**25**	**30**	**36**	**44**
Friction	MsDTW	44	48	54	70	82
	Chroma+DLNCO	**14**	**17**	**22**	**28**	**37**
Moving	MsDTW	61	63	75	127	871
	Chroma+DLNCO	**33**	**39**	**47**	**59**	**732**
Dreams	MsDTW	71	84	114	142	178
	Chroma+DLNCO	**24**	**28**	**39**	**52**	**85**
Average over jazz/pop examples	MsDTW	53	62	71	96	260
	Chroma+DLNCO	**21**	**24**	**30**	**38**	**183**
Average over all examples	MsDTW	59	66	74	89	152
	Chroma+DLNCO	**27**	**30**	**33**	**37**	**91**

Table 7.3. Dependency of the final synchronization accuracy on the size of the allowed distortion range. For each of the 16 pieces and each range, the mean values of the synchronization errors are given for the MsDTW and Chroma+DLNCO procedure both post-processed with smooth interpolation. All values are given in milliseconds.

the average mean error over all 16 pieces increases from 59 ms to 152 ms for the MsDTW and from 27 ms to 91 ms for the combined procedure (Chroma+DLNCO). However, the general behavior of the various synchronization procedures does not change significantly with the ranges and the overall synchronization accuracy is still high even in the presence of large distortions. As an interesting observation, for one of the pieces (Moving) the mean error exploded from 59 ms to 732 ms (Chroma+DLNCO) when increasing the range from ±40% to ±50%. Here, a manual inspection showed that, for the latter range, a systematic synchronization error happened. Here, for an entire musical segment of the piece, the audio version was aligned to a similar subsequent repetition of the segment in the distorted MIDI version. However, note that such strong distortion (±50% corresponds to the range of having half tempo to double tempo) rarely occurs in practice and only causes problems for repetitive music.

7.5 Conclusions

In this chapter, we have discussed various refinement strategies for global music synchronization. Based on a novel class of onset-based audio features in combination with previous chroma features, we introduced a new synchronization procedure that can significantly improve the synchronization accuracy while preserving the robustness and efficiency of previously described procedures. For the future, one could further extend the proposed synchronization framework by including various features types that also capture local rhythmic information [94] and that detect even smooth note transitions as often present in orchestral or string music [198].

7.6 Further Notes

We conclude the second part of the thesis with an overview of further synchronization methods described in the literature. For a discussion of earlier approaches such as [25] we refer to [84, 116]. In the following, we concentrate on more recent approaches proposed in the last years.

In this thesis, we focused on the offline music synchronization scenario meaning that all versions to be aligned are entirely available before the actual alignment. However, there are also several synchronization approaches operating in an online fashion where one of the versions is recorded in real-time [18–20, 36, 104, 114]. In [20], the author employs a hybrid between a hidden Markov model and a hidden semi-Markov model to identify the current tempo and score position in real-time. The tempo estimation is based on a set of oscillators. Their resonance frequency is chosen to capture the tempo at several metrical levels (compare also [161]). Designed to align single-instrument polyphonic audio with a given score, the approach relies on low level features derived from the spectrogram to compare the different music representations. Similar to automatic accompaniment, the system was employed to automatically trigger acoustic sounds in real-time as part of a human-computer performance, see [19]. In [36], an online synchronization approach is presented based on a state-space model. In an alignment context, state-space models are conceptually similar to hidden Markov models (HMMs). However, a major difference to HMMs, which employ a finite set of hidden states to compute an alignment, is that state-space models often employ a continuous, infinite set of states. Here, a change of states is not described by a transition within a graphical model (HMM) but by a (continuous) function that maps the current state to a new state. While in HMMs a sequence of states that optimally explains a given sequence of observations can always be computed using the Viterbi algorithm [150], such concepts can not be directly applied to state-space models. In cases where the state mapping function is linear, Kalman filtering can be used to find the optimal state sequence. In other cases, one has to revert to approximative solutions. One of these techniques is particle filtering, which is also called sequential Monte Carlo sampling. For more details, see also the tutorial on particle filters in [32]. In [36], the use of particle filtering allows for estimating the current score position and tempo as continuous variables. Another system which partly employs particle filtering was presented in [18, 114]. In [18], a MIDI-audio alignment is computed in real-time using a hierarchical hidden Markov model which combines particle filtering techniques with a procedure similar to Viterbi decoding. The method relies on accurate training material as well as knowledge, which instruments are active in a given recording. In [114], an improved version of this approach is presented, which additionally allows for audio-audio synchronization. In [104], a windowed variant of dynamic time warping is presented, which allows for synchronizing music representations in near real-time. Here, the idea is to employ standard DTW in a first step to compute a small fraction of the optimal global path using only a part of the cost matrix as defined by a window. Based on the computed local alignment, the window is moved such that another fraction of the optimal global path can be computed in the next step. Heuristics for path cost estimation further reduce the number of evaluated cells. While the procedure significantly reduces the computational costs, it does not guarantee that the optimal global path is actually computed.

Also in the offline synchronization context, several methods have been proposed in recent years [83–85, 109, 132, 134]. In [84], a novel offline synchronization approach is presented based on a conditional random field (CRF). While a CRF is a discrete graphical model similar to an HMM, CRFs allow for the formulation of more general graph structures. In particular, a major difference is that a CRF can employ several observation feature vectors to model the local likelihood of a specific state. In an HMM, the local state likelihood can only depend on a single observation vector. From a conceptual point of view, CRFs employ ideas similar to the cost matrix smoothing techniques described in Section 4.1. To allow for an efficient computation of the alignment result, the authors propose in [84] a hierarchical decoding method (first described in [83]), which is conceptually similar to the multiscale DTW idea described in Section 7.2. In [85] similar concepts as proposed in [84] are presented based on the dynamic Bayesian network (DBN) formalism.

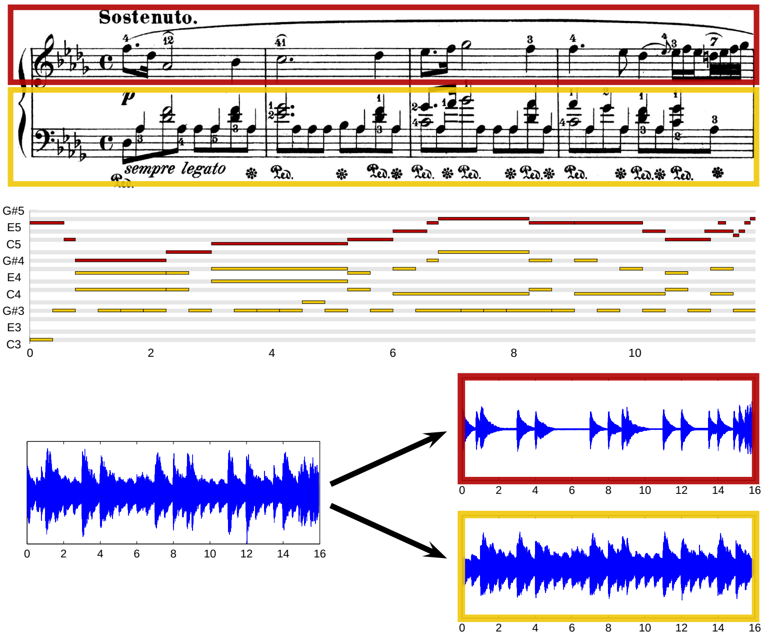
In [132], a novel DTW-based synchronization approach is presented, which additionally employs non-negative matrix factorization (NMF) to post-process the computed alignment results (NMF will be discussed in more detail in Chapter 10). The general idea is to employ a fixed, pre-trained set of NMF-template vectors to analyze the audio signal with the goal to detect sudden pitch-wise energy-increases in the resulting NMF activities. Interpreting these energy-increases as potential onset positions allows then for further refining the onset positions for individual note events. This way, the approach can correctly align arpeggios, which are often notated as concurrent chord notes in a given score. However, in cases where onsets are hard to detect, this weakly constrained post-processing technique often results in misaligning note events with falsely detected onset positions. In [134], an improved version of this approach is presented. Here, the NMF step is only used to identify anchor notes which are assumed to be correctly aligned. Note events between anchor notes are further refined based on interpolation techniques that additionally take the rhythmic information provided by the score into account. In [109], an offline synchronization technique based on a Bayesian hidden Markov model is presented. To compare a given score with the audio, the alignment procedure employs parameters describing the timbre, the fundamental frequency, and the volume related to note events. The synchronization is then performed by iteratively updating these parameters as well as the alignment. While the general approach is promising, the reported results seem to be below the state-of-the-art.

As we have already seen in Part I, the choice of features is important for many tasks in the field of music information retrieval. Also in the context of music synchronization, several feature modifications have been proposed to improve the overall synchronization quality [18, 80, 86, 133]. In [80] and [86], the authors have employed training material to learn a linear mapping that can be used for pre-processing pitch features to enhance the synchronization accuracy. A difference between both approaches is that the authors in [80] incorporate a mapping to derive chroma features from the pitch features, while the authors in [86] employ the enhanced pitch features directly within their synchronization approach. In [18], the authors use non-negative matrix factorization (NMF) to improve the expressiveness of their features. To this end, pitch-wise template vectors are learned in a preprocessing step for each possible instrument. At runtime, the audio signal is analyzed using the pre-learnt template dictionary, which allows for a more precise assessment of the active pitches in a given time frame. However, a disadvantage is that one needs to know which instruments appear in a given recording. In [133], a similar procedure is proposed but instead of learning the templates instruments-wise, they are averaged over the entire training database.

In recent years, much efforts have been directed towards systems that allow for synchronizing music representations across various domains [34, 52–56, 89, 96, 105, 106, 111, 179]. In [89], the goal is to align lyrics given as text files to corresponding audio recordings. The approach combines rhythm and structure analysis methods [62] to link lyrics passages to corresponding audio segments. The method proposed in [55, 56] employs source separation techniques (PreFEst approach [64]) to improve the robustness of the lyrics alignment. In a first step, the main melody is extract from a given recording, which corresponds to the singing voice in most pop songs. The separation result is then synchronized with the textual lyrics using a method similar to regular text-to-speech alignment techniques. In [111] the approach is extended by additionally incorporating automatic methods for harmonic analysis. Here, chord information linked to the lyrics is exploited to stabilize the alignment accuracy. In [96], a method is presented for linking pixel positions in a scanned score image to their corresponding positions in a given audio recording. The approach employs optical music recognition (OMR) techniques to derive an intermediate MIDI-like score representation, which is then synchronized with the audio file. In [52–54] several extensions to this approach are presented. In [54], the authors introduce a novel procedure for mapping single scanned pages of sheet music to corresponding audio clips contained in a given collection of audio recordings. Corresponding score pages and audio segments are aligned similar to [96]. An enhanced version of this approach is presented in [52]. In [53], the approach is further extended using a variant of dynamic time warping, referred to as JumpDTW, which incorporates information about potential jump and repeat position as provided by the score. This way, the scanned score and a corresponding audio recording can be reliably synchronized even if the number of repetitions is unknown for certain parts of the score. A similar approach was also presented in [34] in the context of lead-sheet/audio synchronization. Here, semi-improvised jazz recordings are aligned to lead-sheets which only specify essential musical elements such as the melody and the harmony. As potential jump and repeat positions are not specified as part of a lead-sheet the authors employ manual annotations of these positions. In [105], the authors present a novel approach which synchronizes guitar tablatures with corresponding audio recordings in real-time. Here, the idea is to combine real-time chord recognition techniques [175] with the windowed variant of dynamic time warping proposed in [104]. This alignment technique is also used in [106] to synchronize internet streaming video with high quality audio versions. Finally, building on top of the system introduced in [96], the authors present in [179] a system which allows for a synchronized playback of audio, video, score, and lyrics data.

Part III

Score-Informed Audio Processing



Chapter 8

Score-Informed Source Separation

The decomposition of a mixture of superimposed acoustic sound sources into its constituent components, a task also known as *source separation*, is one of the central research topics in digital audio signal processing [33, 37, 64, 72, 137, 183]. For example, in speech signal processing, an important task is to separate the voice of a specific speaker from a mixture of conversations of multiple speakers and background noises ("Cocktail party scenario"), see for example [87]. Also in the field of musical signal processing, there are many related issues that are commonly subsumed under the notion of source separation. In the musical context, a source might correspond to a melody, a bassline, a drum track, or an instrument track. To extract such sources various elaborate processing and analysis methods have been developed, which has led to significant improvements for tasks such as instrument recognition [72], harmonic analysis [183], or melody estimation [37]. Most of these methods exploit certain spectral and temporal properties of the sound sources to be extracted. For example, the melody is often the leading voice characterized by its dominance in dynamics and by its temporal continuity [9,33]. Or the track of a bass guitar may be identified by specifically looking at the lower part of the frequency spectrum [64]. Furthermore, when extracting the drum track, one often relies on the assumption that the other sources are of harmonic nature. Then, one can exploit the fact that percussive elements (vertical spectral structures) are fundamentally different from harmonic elements (horizontal spectral structures) [137]. Last but not least, a human singing voice can often be distinguished from other musical sources because of the presence of vibrato and portamento (sliding voice) effects [156].

In the last years, also multimodal, score-informed source separation strategies have been employed where one assumes the availability of a score representation along with the music recording. Here, the score provides valuable information in two respects. On the one hand, pitch and timing of note events provide a rough guidance during the separation process. On the other hand, the score provides a natural way of specifying what and how sound sources are to be separated. For example, in [74] the score's natural partition into instrument tracks is exploited to extract each individual instrument from a given audio recording, see Figure 8.1a for an illustration. Here, the score provides additional cues on the sources' spectral and temporal properties. In [78], it was further demonstrated that this concept can also be incorporated into an intuitive and easy-to-use interface. Here, the

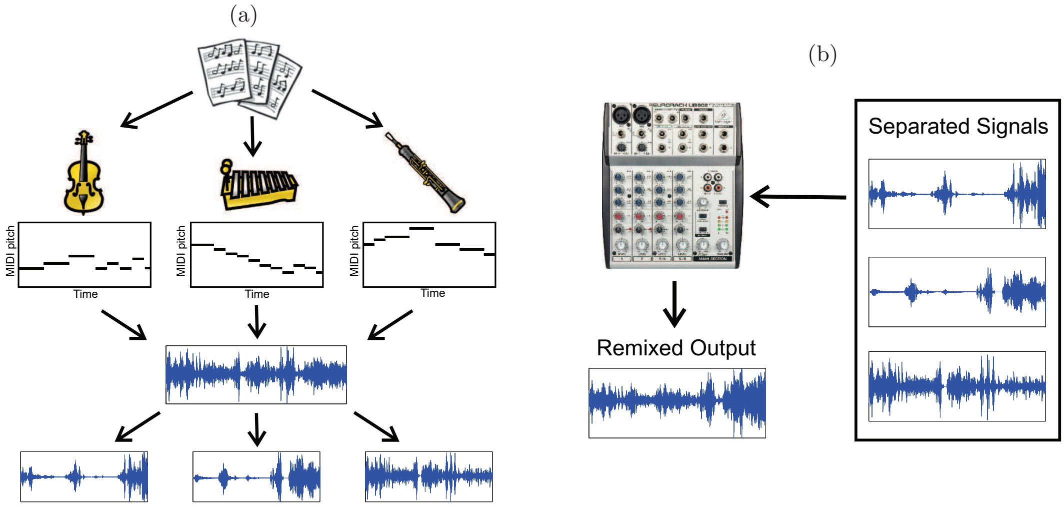


Figure 8.1. Score-informed source separation: **(a)** Instrument tracks as specified by a given score are employed for the separation of instrument sounds from a polyphonic audio recording (figure inspired by [74]). **(b)** Separated signals corresponding to instrument tracks can be remixed by the user in real-time (figure inspired by [78]).

user can adjust the volume of each instrument in real-time using an interactive equalizer, see Figure 8.1b. As an introduction to the task of score-informed source separation, we start the final part of this thesis with a comprehensive overview of available methods and strategies. Rather than focusing on technical details, we will highlight conceptual differences between the individual approaches. In the upcoming Chapter 9 and Chapter 10, we then introduce novel computational approaches to score-informed source separation demonstrating in detail how the score information is employed to guide the separation process.

8.1 Overview of Available Approaches

In the context of score-informed source separation, music synchronization methods as introduced in Part II of this thesis are of particular importance thus we will discuss their use in the following explicitly. In particular, we distinguish two groups of separation approaches. Methods in the first group employ robust and accurate synchronization techniques and consider or even account for typical differences between the score and a given interpretation, for example, in terms of structure, ornamentation, the interpretation of trills and arpeggios as well as additional and missed notes. By adjusting the onset position and duration of each MIDI event, these approaches use the computed synchronization result to transform the original *score-like MIDI file* to a *synchronized MIDI file*, which runs synchronously to the audio, see Figure 8.2. Methods in the second group simply assume that *perfectly synchronized MIDI files* are available. This assumption, however, is often not realistic in real-world scenarios. The practical applicability of these approaches is therefore often hard to assess.

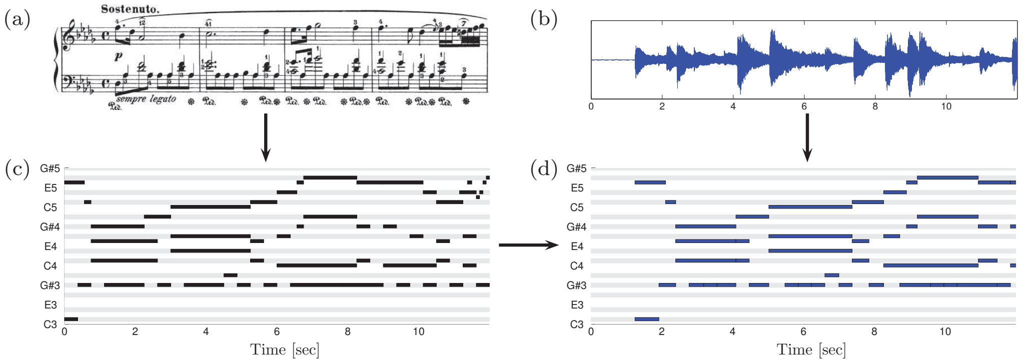


Figure 8.2. Music synchronization for a score and an audio recording of Chopin's Op. 28 No. 15: **(a)** Musical score. **(b)** Audio recording of an interpretation taken from the SMD database [124]. **(c)** Score-like MIDI file generated from the score shown in (a). **(d)** Synchronized MIDI file.

Early score-informed source separation approaches adopt score and MIDI information only for evaluation purposes, for example, to investigate the influence of a pitch estimation step in a complex separation system [148]. One of the first approaches focusing on the conceptual benefits of incorporating score information was proposed in [166]. Here, the task consists in separating a single instrument specified by a given score-like MIDI file from a polyphonic music recording. The main idea is based on designing a filter, which in some sense optimally extracts the instrument from the recording. To compute the MIDI-audio synchronization, the authors refer to a procedure previously proposed in [165]. While presenting a novel application idea, this early work has several conceptual limitations. First of all, the proposed filter design procedure models all non-target sound sources as Gaussian noise. Therefore, in cases where the target instrument is accompanied by other instruments, this assumption is obviously violated. Furthermore, the proposed method assumes that the score provides an exact specification of the fundamental frequency for the target instrument for each analysis frame. This assumption is not realistic, since the score usually provides only high-level note information of the piece of music without specifying tuning or small pitch deviations of the respective music recording.

Subsequently proposed systems were not subject to such strict limitations. In [195], the authors integrate score information into a system for blind source separation previously described in [194] (an extended version was presented in [193]). Here, the goal is to extract individual instruments from a music recording, which then enables a user to create new music by remixing the extracted sound sources. In this approach, stereo information is employed in a first step to determine for each analysis frame the number of concurrent sources. Frames identified to contain only a single source are used as cues in the consecutive pitch-tracking step to support the separation in frames with multiple sources. The authors incorporate score information into this process as a rough guidance for the pitch-tracking. The underlying MIDI-audio alignment is based on a procedure proposed by Hu et al. [75]. A technical limitation of the approach is its dependency on reliable stereo information to identify the sources. This is problematic for many commercial studio productions, where spatial information contained in the stereo recordings is often corrupted by digital effect filters and virtual room acoustics. Furthermore, the influence of the alignment step is hard to assess from the experimental results, as the method is only evaluated on a dataset consisting of four second snippets of synthetically created MIDI sonifications.

While score information is used in [195] mostly as an add-on to an existing source separation system, Han and Raphael presented in [68, 69] a model that completely relies on available score data. In their contribution, the authors aim at removing the soloist from orchestral music recordings to generate recordings that can be used as a basis for automated accompaniment systems [29]. Relying on score information at an early stage of their algorithmic pipeline allowed for innovative computational concepts. On the one hand, the method represents a given input spectrogram as a compound of note-event based models. This allows for effectively using the score information to specify the temporal and spectral extent in which a note-event is permitted to be active. On the other hand, the score is used to identify the instruments occurring in a given music recording. This way, some instrument-dependent model parameters such as overtone energy distributions can simply be learnt from monophonic training material in advance and fixed afterwards. A benefit of this approach is that the parameter estimation process becomes efficient (as only a small set of parameters needs to be adjusted) and robust (as unreasonable parameter values are prevented by the model). However, a drawback is that the model can be imprecise, in particular when the training instruments differ strongly from the ones used in the given recording.

Roughly at the same time, Itoyama et al. presented a system, which explored novel application scenarios based on score-informed source separation [78]. This system allows a user to adjust the volume of each instrument in a polyphonic music recording in real-time. To this end, the system separates the individual instrument tracks in a preprocessing step as follows. In a first step, a MIDI synthesizer is employed to create one audio representation for each of the instrument tracks contained in a given MIDI file. This audio data is used as prior knowledge to initialize a note-based spectrogram model. Next, the model parameters are adapted to a given audio recording by minimizing a Kullback-Leibler distance between the given spectrogram and the model spectrogram. Here, to allow only musically meaningful values for the model parameters, strong deviations from the initial values set in the first step are penalized. In a final step, the spectrogram model is employed to isolate the individual instrument tracks as specified by the MIDI file. Technically, the model is based on the harmonic-temporal-structured clustering (HTC) model proposed in [88]. To control the influence of their percussion related submodel on the remaining system, the authors have to resort to smoothing and regulation techniques [77], which further increase the complexity of the system. Furthermore, alignment issues are not considered in this approach, hence it is not clear how the system behaves in real-world scenarios starting with score-like MIDI files.

Using MIDI-synthesized audio material for initialization purposes was also proposed by Gansemann et al. in [58,59]. Given a MIDI file and an audio recording for a piece of music, the approach starts by sonifying the MIDI instrument tracks using a wavetable synthesizer similar to [78]. In a next step, probabilistic latent component analysis (PLCA) [167] is employed to identify the most important spectral components for each sonification. Here, PLCA is a probabilistic formulation of the well-known non-negative matrix factorization (NMF) method, which will be discussed in more detail in Chapter 10. In a last step, the instrument-wise spectral components are used as initialization and additional knowledge for a prior-based PLCA analysis of the original audio recording [172]. The results of this final analysis are subsequently used to extract each instrument from the original recording. Incorporating an alignment procedure by Turetsky and Ellis [181], the authors aim at using full-length score-like MIDI files as they can be found in real-world scenarios. While this approach presents a novel computational concept, the approach suffers from several weaknesses. Similar to all approaches relying on synthetic audio material as prior knowledge, this method's separation quality depends on the spectral similarity between the MIDI instruments and the actual target instruments. Moreover, this method also requires that the MIDI instruments have a similar tuning as the instruments in the given audio recording. For large tuning deviations, the separation quality might be significantly reduced.

An alternative way of using MIDI information for initialization purposes was presented in [74]. Here, instead of generating synthetic audio, the MIDI file is used to directly instruct the underlying spectrogram model when a given instrument is active with a certain pitch. This way, the separation performance does not depend on the quality of an underlying MIDI synthesizer. However, as a drawback, no expectations about the spectral shape of an instrument are incorporated, which may lead to a less robust separation process. As a novel contribution, the method employs a parametric NMF variant [73], which significantly enhances the modeling accuracy for instruments with vibrato and glissando. A technical limitation of this model is that all harmonic sounds in an analysis frame are assumed to be a compound of stationary sinusoidals. To evaluate the instrument separation quality of this approach, the authors neglect the alignment step and employ synthetic MIDI sonifications of Bach, Beethoven and Boccherini pieces.

As demonstrated by Duan and Pardo in [35], the separation step can be performed in a low-delay real-time fashion. To this end, the authors replace the usually employed offline synchronization step by an online approach [36], which aligns a given MIDI file and a corresponding audio recording in real-time, a task often referred to as score-following [20, 29]. For each analysis frame, their separation system first estimates the exact fundamental frequency of each pitch using the aligned MIDI file as a guidance. In a next step, each pitch is extracted using a harmonic mask and assigned to one of the instruments as specified by the MIDI file. To make this process feasible in real-time, the mask is computed using a fixed overtone model, which is not adapted to a given recording.

Overall, while source separation has been a field of research for decades, using score information to guide the separation process is a relatively recent approach. As demonstrated by the contributions discussed in this section, score guidance allows for novel and innovative applications of source separation techniques. Furthermore, the additional musical cues provided by the score often allow for a gain in separation quality, which is difficult to achieve otherwise. Moreover, robust music synchronization techniques allow for using score-informed source separation methods in real-world scenarios, where usually no perfectly aligned MIDI file is available.

Chapter 9

Audio Parameterization

In general, separating sound sources from polyphonic music recordings requires an understanding of many musical and technical aspects. For example, one has to account for the complexity of musical sound sources, the interaction and superposition of such sources in polyphonic mixtures, room acoustics, and recording conditions. Additionally, in many studio productions, numerous digital effect filters are applied to the recording thus making the task even more complex. However, although being extremely difficult, source separation is mostly pursued in a blind fashion, where as little prior knowledge as possible is used. A natural idea to facilitate the separation process is to incorporate additional musical cues, for example, in the form of available musical score data. Based on this idea, this chapter presents a novel approach for separating musically meaningful sound sources from polyphonic audio recordings. More precisely, given a MIDI file and an audio recording of a piece of music, the idea is to employ a parametric model that describes a spectrogram as a sum of note-event spectrograms. Here, each note-event spectrogram describes the part of a spectrogram that can be attributed to a specific note event. The proposed method starts by initializing the pitch, onset, and duration parameters in the spectrogram model using the note events provided by the MIDI file. In the second step, we adapt the onset and duration parameters by aligning the note events with their corresponding occurrences in the audio using the high-resolution music synchronization approach described in Chapter 7. In the third step, we iteratively modify model parameters related to the acoustic representation of a note event such that the model spectrogram approximates the audio spectrogram as accurately as possible.

To investigate the separation quality of the proposed method we consider two use cases (Section 9.2 and Section 9.3). In particular, we investigate how an instrument equalizer as described in Chapter 8 can be extended to a more general voice or note equalizer. Here, instead of just having the possibility to emphasize or attenuate entire instrument tracks a user can freely choose which group of notes should be in the focus. For example, a user might highlight a melody, a specific motif, the left or the right hand of a piano score, or an entire staff. To demonstrate this concept, a prototypical implementation of a novel interface is presented using the multimodal music player presented in [22, 23] as a basis. This interface shows a score aligned to the audio where the user can click on individual staffs corresponding to the left or the right hand of a piano piece. The system then separates or enhances the corresponding group of notes in real-time.

In a second use case, we investigate how score-informed source separation techniques can be employed in music *analysis* tasks. This is in contrast to most previous methods, which usually have the goal to create audible separation results. In particular, given a MIDI file and an audio recording for a piece of music, we employ the techniques developed in the following to analyze how intense or how loud each MIDI note event occurs in the audio recording. In this way, a comparison of dynamics across different interpretations of a piece is no longer limited to the measure- or the chord-level, but can even be performed on the basis of single notes. Furthermore, extracting such performance-specific subtleties allows for enriching a given score-like MIDI representation.

9.1 Parametric Model

To describe an audio recording of a piece of music using a parametric model, one has to consider many musical and acoustical aspects [72, 78]. For example, parameters are required to encode the pitch as well as the onset position and duration of note events. Further parameters might encode tuning aspects, the timbre of specific instruments, or amplitude progressions. In this section, we describe our model and show how its parameters can be estimated by an iterative method.

Let $X \in \mathbb{C}^{K\times N}$ denote the spectrogram and $Y = |X|$ the magnitude spectrogram of a given music recording. Furthermore, let $\mathcal{S} := \{\mu_s \mid s \in [1:S]\}$ denote a set of note events as specified by a MIDI file representing a musical score. Here, each note event is modeled as a triple $\mu_s = (p_s, t_s, d_s)$, with p_s encoding the MIDI pitch, t_s the onset position and d_s the duration of the note event. Our strategy is to approximate Y by means of a model spectrogram $Y_\lambda^{\mathcal{S}}$, where λ denotes a set of free parameters representing acoustical properties of the note events. Based on the note event set $\mathcal{S}$, the model spectrogram $Y_\lambda^{\mathcal{S}}$ will be constructed as a superposition of note-event spectrograms Y_λ^s, $s \in [1:S]$. More precisely, we define $Y_\lambda^{\mathcal{S}}$ at frequency bin $k \in [1:K]$ and time frame $n \in [1:N]$ as

$$Y_\lambda^{\mathcal{S}}(k,n) := \sum_{\mu_s \in \mathcal{S}} Y_\lambda^s(k,n), \tag{9.1}$$

where each Y_λ^s denotes the part of $Y_\lambda^{\mathcal{S}}$ that is attributed to μ_s. Each Y_λ^s consists of a component describing the amplitude or activity over time and a component describing the spectral envelope of a note event. More precisely, we define

$$Y_\lambda^s(k,n) := \alpha_s(n) \cdot \varphi_{\tau,\gamma}(\omega_k, p_s), \tag{9.2}$$

where ω_k denotes the frequency in Hertz associated with the k-th frequency bin. Furthermore, $\alpha_s \in \mathbb{R}_{\geq 0}^N$ encodes the activity of the s-th note event. Here, we set $\alpha_s(n) := 0$, if the time position associated with frame n lies in $\mathbb{R} \setminus [t_s, t_s + d_s]$. The spectral envelope associated with a note event is described using a function $\varphi_{\tau,\gamma} : \mathbb{R} \times [1 : P] \to \mathbb{R}_{\geq 0}$, where $[1:P]$ with $P = 127$ denotes the set of MIDI pitches. More precisely, to describe the frequency and energy distribution related to a specific note event with MIDI pitch $p \in [1:P]$,

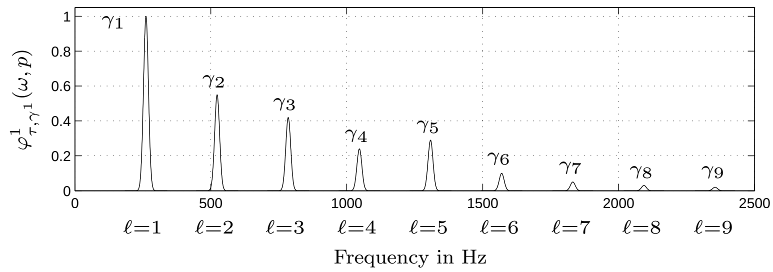


Figure 9.1. Illustration of the spectral envelope function $\varphi^1_{\tau,\gamma^1}(\omega,p)$ for $p = 60$ (middle C), $\tau = 0$ and some example values for the parameter γ^1.

the function $\varphi_{\tau,\gamma}$ depends on a parameter $\tau \in [-0.5, 0.5]^P$ related to the tuning and a parameter γ related to the energy distribution over the first L partials. In the following, we consider two different spectral envelope models and corresponding functions φ. Given a frequency ω in Hertz and $\gamma^1 \in [0,1]^L$, we define the envelope function for the first model

$$\varphi^1_{\tau,\gamma^1}(\omega,p) := \sum_{\ell\in[1:L]} \gamma^1_\ell \cdot \kappa(\omega - \ell \cdot f(p+\tau_p)), \tag{9.3}$$

where the function $\kappa : \mathbb{R} \to \mathbb{R}_{\geq 0}$ is a suitably chosen Gaussian centered at zero, which is used to describe the shape of a partial in frequency direction, see Figure 9.1. Furthermore, $f : \mathbb{R} \to \mathbb{R}_{\geq 0}$ defined by $f(p) := 2^{(p-69)/12} \cdot 440$ maps the pitch to the frequency scale. To account for non-standard tunings, we use the parameter τ_p to shift the fundamental frequency upwards or downwards by up to half a semitone. The spectral envelope function for the second model is defined as

$$\varphi^2_{\tau,\gamma^2}(\omega,p) := \sum_{\ell\in[1:L]} \gamma^2_{\ell,p} \cdot \kappa(\omega - \ell \cdot f(p+\tau_p)), \tag{9.4}$$

where $\gamma^2 \in [0,1]^{L\times P}$.

These two models differ significantly in the way they describe the spectral envelope associated with a note event. In the first model, one assumes that the overtone energy distribution is independent of the pitch (parameter γ^1 does not depend on p). So the energy distribution, for example, for a middle C is assumed to be the same as for a middle E. The second model implements exactly the opposite idea assuming that the partials' energy distribution strongly depends on the pitch. Therefore, the parameter γ^2 has a dependency on p. Both models have advantages and disadvantages. On the one hand, using an individual description of the energy distribution for each pitch as in the second model allows for a fine grained representation of the spectral envelope. On the other hand, the parameter estimation process might not be robust in some cases, for example, for pitches with many overlapped partials. Consider for example a recording of a sequence of C-major chords. Here, the second and the fourth overtone of the C overlap with overtones belonging to the E and the G. For these overtones, the second model cannot resolve how much energy belongs to each of the three pitches, because they always occur together. The first model, however, assumes that the overtone energy distribution is the same for all three pitches. Here, one can exploit that the second and fourth overtone of the E and the second overtone of the G are not overlapped by any other overtones. This way, information about

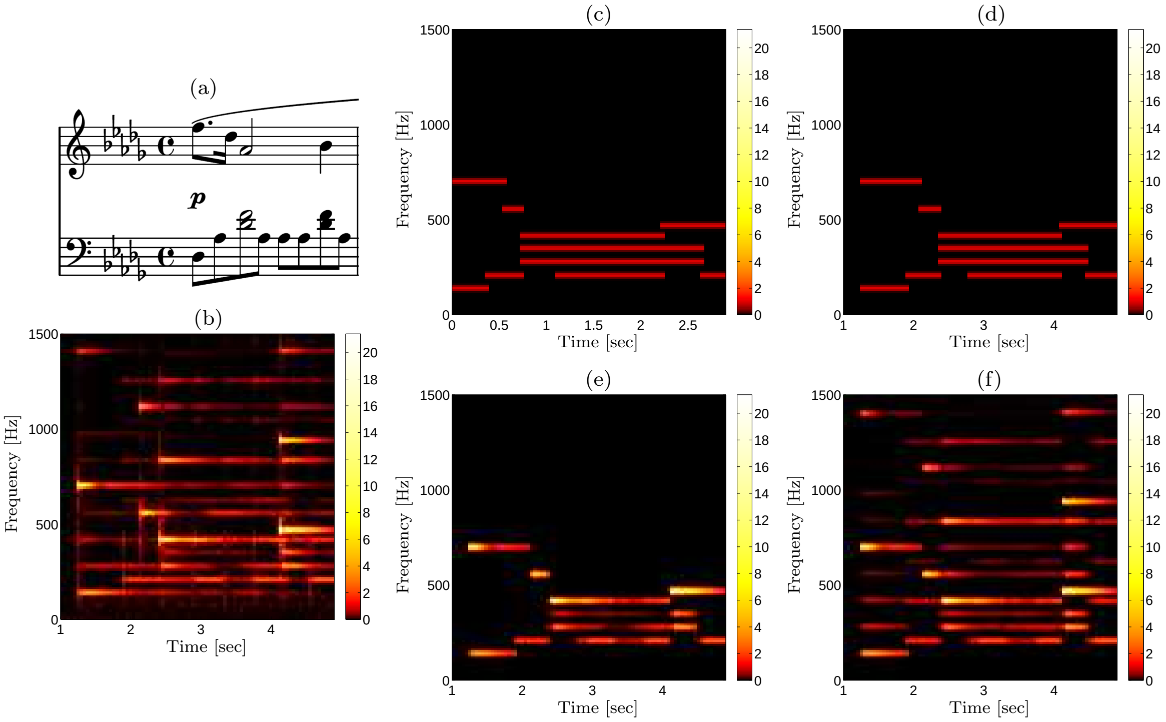


Figure 9.2. Illustration of the first iteration of the parameter estimation procedure using the first measure of Chopin's Prélude "Raindrop" (Op. 28 No. 15). **(a)** Score for the first measure. **(b)** Audio spectrogram Y to be approximated. **(c)-(f)** Model spectrogram Y_λ after certain parameters are estimated. **(c)** Parameter $\mathcal{S}$ is initialized with MIDI note events. **(d)** Note events in $\mathcal{S}$ are synchronized with the audio recording. **(e)** Activity α and tuning parameter τ are estimated. **(f)** Partials' energy distribution parameter γ is estimated.

the spectral envelope of the E and the G can be used transitively to resolve overlapped partials of the C.

Altogether, $\lambda := (\alpha, \tau, \gamma)$ denotes the set of free parameters with $\alpha := \{\alpha_s \mid s \in [1:S]\}$ and $\gamma := \gamma^1$ for the first spectral envelope model and $\gamma := \gamma^2$ for the second. Note, that the number of free parameters is kept low by sharing the parameters τ and γ across the individual note events given by $\mathcal{S}$. Here, a low number allows for an efficient parameter estimation process as described below and additionally prevents model over-fitting.

Now, finding a meaningful parameterization of Y can be formulated as the following optimization task:

$$\lambda^* = \underset{\lambda}{\operatorname{argmin}} \|Y - Y_\lambda^{\mathcal{S}}\|_F, \tag{9.5}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. In the following, we illustrate the individual steps in our parameter estimation procedure in Figure 9.2 using an audio recording of Chopin's Prélude "Raindrop" (Op. 28 No. 15) taken from the SMD database [124]. Here, the given audio spectrogram (Figure 9.2b) is gradually approximated by our model (Figure 9.2c-9.2f).

9.1.1 Initialization and Adaption of Note Timing Parameters

To initialize our model, we exploit the available MIDI information represented by $\mathcal{S}$. For the s-th note event $\mu_s = (p_s, t_s, d_s)$, we set $\alpha_s(n) := 1$ if the time position associated with frame n lies in $[t_s, t_s + d_s]$ and $\alpha_s(n) := 0$ otherwise. Furthermore, we set $\tau_p := 0$, $\gamma_1^1 := \gamma_{1,p}^2 := 1$ and $\gamma_\ell^1 := \gamma_{\ell,p}^2 := 0$ for $p \in [1:P], \ell \in [2:L]$. An example model spectrogram $Y_\lambda^{\mathcal{S}}$ using spectral envelope model φ^2 after the initialization is given in Figure 9.2c.

Next, we need to adapt and refine the model parameters to approximate the given audio spectrogram as accurately as possible. This parameter adaption is simplified when the MIDI file is assumed to be perfectly aligned to the audio recording as in [78]. However, in most practical scenarios such a MIDI file is not available. Therefore, in our approach, we employ the high resolution music synchronization approach as described in Chapter 7. Using the resulting alignment, we determine for each note event the corresponding position in the audio recording and update the onset positions and durations in $\mathcal{S}$ accordingly. After the synchronization, the note event set $\mathcal{S}$ remains unchanged during all further parameter estimation steps. Figure 9.2d shows an example model spectrogram after the synchronization step.

9.1.2 Estimation of Model Parameters

To estimate the parameters in λ, we look for (α, τ, γ) that minimize the function $d(\alpha, \tau, \gamma) := \|Y - Y_{(\alpha,\tau,\gamma)}^{\mathcal{S}}\|_F$, thus minimizing the distance between the audio and the model spectrogram. Additionally, we need to consider range constraints for the parameters. For example, τ is required to be an element of $[-0.5, 0.5]^P$. To approximately solve this constraint optimization problem, we employ a variant of the trust region based interior points method described in [11]. To this end, we fix two parameters and minimize d regarding the third. For example, to get a better estimate for α, we fix τ and γ and minimize $g(\cdot, \tau, \gamma)$. This process is repeated until all three parameters converge. Figure 9.2e and f illustrate the first iteration of our parameter estimation. Here, Figure 9.2e shows the spectrogram described by our model after the estimation of the tuning parameter τ and the activity parameter α. Figure 9.2f shows the spectrogram model after the estimation of the energy distribution parameter γ^2. Here, one can observe that our model focuses on the harmonic parts of a spectrogram while mainly ignoring the noisy percussive elements.

The main ideas behind [11] can be summarized as follows. Let $h : \mathbb{R}^d \to \mathbb{R}$ be a function to be minimized and $x \in \mathbb{R}^d$. Then the method computes the first and second derivative of h at position x and derives a quadratic approximation of h using the Taylor series. The Taylor approximation of h is assumed to be meaningful only in a neighborhood of x, the so called trust region. Instead of minimizing h, the method then minimizes the Taylor approximation within the trust region yielding a new value x^*. If $h(x^*) > h(x)$, then the trust region was too large and the approximation of h was insufficient. In this case, the trust region is decreased and the process is repeated for x. If $h(x^*) \leq h(x)$, then x is set to x^* and the process is repeated until x converges. Possible parameter range constraints are considered in [11] by a barrier approach. Here, the basic idea is to reformulate the function h by including penalty terms that get increasingly large when the value for a parameter is invalid.

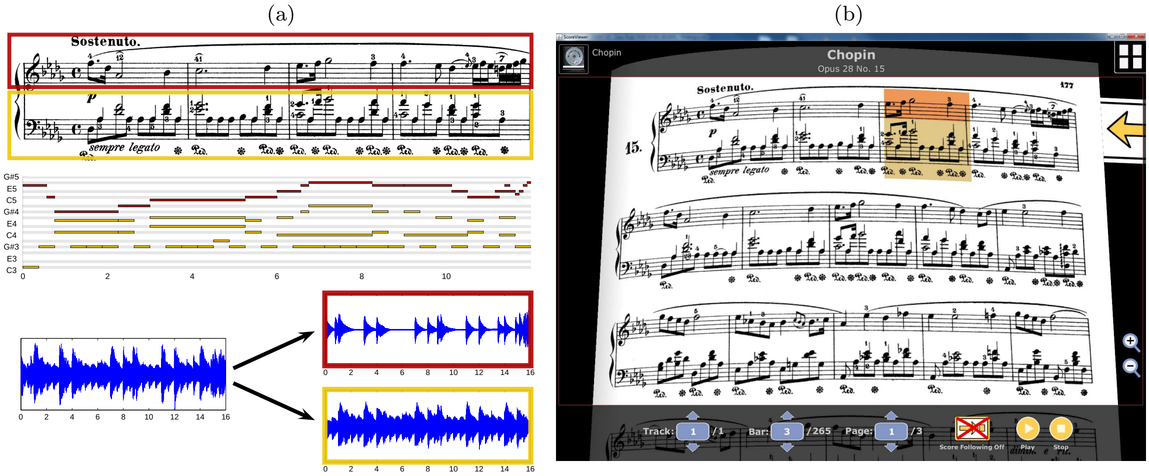


Figure 9.3. Score-informed voice separation: **(a)** Decomposition of a piano recording into two sound sources corresponding to the left and right hand as specified by a musical score. Shown are the first four measures of Chopin's Prélude "Raindrop" (Op. 28 No. 15). **(b)** Prototypical implementation of a voice equalizer based on the multimodal music player proposed in [22]. By selecting a staff/hand in the scanned score image the corresponding group of notes is separated/enhanced in real-time.

9.2 Application: Voice Equalizer

In this section, we employ the parametric model presented in Section 9.1 to develop an automated method for the decomposition of a monaural piano recording into sound sources corresponding to the left and the right hand as specified by a score, see Figure 9.3a. Played on the same instrument and often being interleaved, the two sources share in this case many spectral properties. As a consequence, classical source separation techniques that rely on statistical differences between the sound sources are not directly applicable. To make the separation process feasible, we exploit the fact that a musical score is available for many pieces and use our score-informed parametric model to approximate the spectrogram of the given piano recording. Characterizing which parts of the spectrogram belong to a given note event, the model is then employed to decompose the spectrogram into parts related to the left hand and to the right hand. While we restrict the task in this section to the left/right hand scenario, the proposed method is sufficiently general to isolate any kind of voice (or group of notes). In particular, a note group might correspond to a melody, a motif, an accompaniment track, or any other set of notes as specified by the user or by some labeling of the score.

As an application, the goal is to extend the idea of an instrument equalizer as presented in [78] and discussed in Chapter 8 to a generally applicable voice equalizer. Here, rather than being limited to emphasizing some instrument tracks the user can more freely decide which musical elements he or she wants to highlight and to focus on. Integrating these ideas into the multimodal music player proposed in [22] we demonstrate that such a voice equalizer can be used in an intuitive and user-friendly way. In its original form the player highlights during audio playback the corresponding position in the score. Integrating the proposed system, the player was extended such that by selecting a staff in the scanned score image the corresponding group of notes is separated or enhanced in real-time, see Figure 9.3b.

This way, a listener can easily select and focus on specific musically meaningful elements of a piece, which might be hard to recognize in a regular audio recording. In the following sections, we now describe the techniques underlying the proposed voice equalizer. Using the left-right hand scenario as an example, we describe in Section 9.2.1 how our parametric model can be employed to extract arbitrary note groups from a given audio recording. In Section 9.2.2, we then indicate the separation quality of this approach in systematic experiments.

9.2.1 Separation Process

Given a MIDI file and an audio recording for a piece of music, we first estimate parameters λ such that the model spectrogram $Y_\lambda^{\mathcal{S}}$ approximates the original spectrogram Y as precisely as allowed by the model. To this end, we first initialize our model with the note events provided by the MIDI file and adapt the model parameters iteratively as described in Section 9.1. In a next step, we employ information derived from the model to decompose the original audio spectrogram into separate channels or voices. To this end, we exploit that $Y_\lambda^{\mathcal{S}}$ is a compound of note-event spectrograms Y_λ^s. With $\mathcal{T} \subset \mathcal{S}$, we define $Y_\lambda^{\mathcal{T}}$ as

$$Y_\lambda^{\mathcal{T}}(k,n) := \sum_{\mu_s \in \mathcal{T}} Y_\lambda^s(k,n). \tag{9.6}$$

Then $Y_\lambda^{\mathcal{T}}$ approximates the part of Y that can be attributed to the note events in $\mathcal{T}$. One way to obtain an audible separation result could be to apply a spectrogram inversion directly to $Y_\lambda^{\mathcal{T}}$. However, to yield an overall robust approximation result the proposed model does not attempt to capture every possible spectral nuance in Y. Therefore, an audio recording deduced directly from $Y_\lambda^{\mathcal{T}}$ would miss these nuances and would consequently sound rather unnatural. Instead, we revert to the original spectrogram again and use $Y_\lambda^{\mathcal{T}}$ only to extract suitable parts of Y. To this end, we derive a *separation mask* $M^{\mathcal{T}} \in [0,1]^{K \times N}$ from the model which encodes how strongly each entry in Y should be attributed to $\mathcal{T}$. More precisely, we define

$$M^{\mathcal{T}} := \frac{Y_\lambda^{\mathcal{T}}}{Y_\lambda^{\mathcal{S}} + \varepsilon}, \tag{9.7}$$

where the division is understood entry-wise. The small constant $\varepsilon > 0$ is used to avoid a potential division by zero. Furthermore, ε prevents that relatively small values in $Y_\lambda^{\mathcal{T}}$ lead to large masking values, which would not be justified by the model. For our experiments, we set $\varepsilon = 10^{-2}$.

For the separation, we apply $M^{\mathcal{T}}$ to a magnitude spectrogram via

$$\hat{Y}^{\mathcal{T}} := M^{\mathcal{T}} \circ Y, \tag{9.8}$$

where $\circ$ denotes entry-wise multiplication (Hadamard product). The resulting $\hat{Y}^{\mathcal{T}}$ is referred to as *estimated magnitude spectrogram.* Here, using a mask for the separation allows for preserving most spectral nuances of the original audio. In a final step, we apply a spectrogram inversion to yield an audible separation result. Here, a commonly used

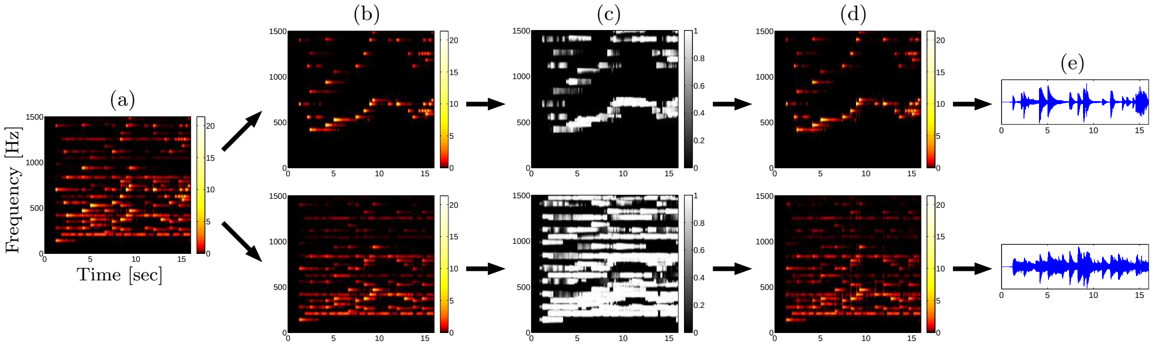


Figure 9.4. Illustration of our voice separation process continuing the example shown in Figure 9.3a. **(a)** Model spectrogram $Y_\lambda^{\mathcal{S}}$ after the parameter estimation. **(b)** Derived model spectrograms $Y_\lambda^{\mathcal{L}}$ and $Y_\lambda^{\mathcal{R}}$ corresponding to the notes of the left and the right hand. **(c)** Separation masks $M^{\mathcal{L}}$ and $M^{\mathcal{R}}$. **(d)** Estimated magnitude spectrograms $\hat{Y}^{\mathcal{L}}$ and $\hat{Y}^{\mathcal{R}}$. **(e)** Reconstructed audio signals $\hat{x}^{\mathcal{L}}$ and $\hat{x}^{\mathcal{R}}$.

approach is to combine $\hat{Y}^{\mathcal{T}}$ with the phase information of the original spectrogram X in a first step. Then, an inverse FFT in combination with an overlap-add technique is applied to the resulting spectrogram [72]. However, this can lead to clicking and ringing artifacts in the resulting audio recording. Therefore, we apply a spectrogram inversion approach originally proposed by Griffin and Lim in [67]. The method attenuates the inversion artifacts by iteratively modifying the original phase information. The resulting $\hat{x}^{\mathcal{T}}$ constitutes our final separation result referred to as *reconstructed audio signal (relative to $\mathcal{T}$).*

Next, we transfer these techniques to our left/right hand scenario. Each step of the full separation process is illustrated by Figure 9.4. First, we assume that the score is partitioned into $\mathcal{S} = \mathcal{L} \dot{\cup} \mathcal{R}$, where $\mathcal{L}$ corresponds to the note events of the left hand and $\mathcal{R}$ to the note events of the right hand. Starting with the model spectrogram $Y_\lambda^{\mathcal{S}}$ (Figure 9.4a) we derive the model spectrograms $Y_\lambda^{\mathcal{L}}$ and $Y_\lambda^{\mathcal{R}}$ using Eqn. (9.6) (Figure 9.4b) and then the two masks $M^{\mathcal{L}}$ and $M^{\mathcal{R}}$ using Eqn. (9.7) (Figure 9.4c). Applying the two masks to the original audio spectrogram Y, we obtain the estimated magnitude spectrograms $\hat{Y}^{\mathcal{L}}$ and $\hat{Y}^{\mathcal{R}}$ (Figure 9.4d). Finally, applying the Griffin-Lim based spectrogram inversion yields the reconstructed audio signals $\hat{x}^{\mathcal{L}}$ and $\hat{x}^{\mathcal{R}}$ (Figure 9.4e).

9.2.2 Experiments

This section describes systematically conducted experiments to illustrate the potential of the proposed method. For the evaluation, we use a database consisting of seven representative pieces from the Western classical music repertoire, see Table 9.1. Using only freely available audio and score data allows for a straightforward replication of the following experiments. In particular, the database contains uninterpreted score-like MIDI files from the Mutopia Project[1](MUT), high-quality audio recordings from the Saarland

[1] `http://www.mutopiaproject.org`

Composer	Piece	MIDI	Audio 1	Audio 2	Identifier
Bach	BWV875-01	MUT	Synthetic	SMD	Bach875
Beethoven	Op031No2-01	MUT	Synthetic	SMD	Beet31No2
Beethoven	Op111-01	MUT	Synthetic	EA	BeetOp111
Chopin	Op028-01	MUT	Synthetic	SMD	Chop28 − 01
Chopin	Op028-04	MUT	Synthetic	SMD	Chop28 − 04
Chopin	Op028-15	MUT	Synthetic	SMD	Chop28 − 15
Chopin	Op064No1	MUT	Synthetic	EA	Chop64No1
Chopin	Op066	MUT	Synthetic	SMD	Chop66

Table 9.1. Pieces and audio recordings (with identifier) used in our experiments.

Identifier	SNR	SNR	SNR	SNR	SNR	SNR
	($Y^{\mathcal{L}}$,$\hat{Y}^{\mathcal{L}}$)	($Y^{\mathcal{R}}$,$\hat{Y}^{\mathcal{R}}$)	($Y^{\mathcal{L}}$,$\hat{Y}^{\mathcal{L}}$)	($Y^{\mathcal{R}}$,$\hat{Y}^{\mathcal{R}}$)	($Y^{\mathcal{L}}$,Y)	($Y^{\mathcal{R}}$,Y)
	pre-aligned		distorted			
Bach875	11.24	12.97	11.17	12.89	-1.99	3.03
Beet31No2	12.65	10.38	12.47	10.23	1.24	-0.09
BeetOp111	13.21	12.26	12.92	11.99	0.16	0.97
Chop28-01	10.52	13.96	10.43	13.84	-3.38	4.48
Chop28-04	17.63	10.48	17.58	10.45	8.65	-7.55
Chop28-15	17.79	13.35	17.56	13.18	3.48	-2.47
Chop64No1	12.93	11.86	12.60	11.55	-0.06	1.31
Chop66	11.61	11.17	11.46	11.03	-0.41	2.01
Average	**13.45**	**12.05**	**13.27**	**11.90**	**0.96**	**0.21**

Table 9.2. Experimental results using ground truth data consisting of synthesized versions of the pieces in our database. For these experiments the spectral envelope model φ^2 was employed.

Music Database[2](SMD) as well as digitized versions of historical gramophone and vinyl recordings from the European Archive[3](EA).

In a first step, we indicate the quality of proposed method quantitatively using synthetic audio data. To this end, we use the Mutopia MIDI files to create two additional MIDI files for each piece using only the notes of the left and the right hand, respectively. Using a wave table synthesizer, we can then generate audio recordings from these MIDI files which are used as ground truth separation results in the following experiment. We denote the corresponding magnitude spectrograms by $Y^{\mathcal{L}}$ and $Y^{\mathcal{R}}$, respectively. For our evaluation we use a quality measure based on the signal-to-noise ratio (SNR)[4]. More precisely, to compare a reference magnitude spectrogram $Y_R \in \mathbb{R}_{\geq 0}^{K \times N}$ to an approximation $Y_A \in \mathbb{R}_{\geq 0}^{K \times N}$we define

$$\mathrm{SNR}(Y_R, Y_A) := 10 \cdot \log_{10} \frac{\sum_{k,n} Y_R(k,n)^2}{\sum_{k,n} (Y_R(k,n) - Y_A(k,n))^2}.$$

The second and third column of Table 9.2 show SNR values for all pieces using the spectral envelope model φ^2. Here, the ground truth is compared to the estimated spectrogram for the left and the right hand. For example, the left hand SNR for Chop28 − 15 is 17.79 whereas the right hand SNR is 13.35. The reason the SNR being higher for the left hand

[2] http://www.mpi-inf.mpg.de/resources/SMD/

[3] http://www.europarchive.org

[4] Even though SNR values are often not perceptually meaningful, they at least give some tendencies on the quality of separation results.

than for the right hand is that the left hand is already dominating the mixture in terms of overall loudness. Therefore, the left hand segregation is per se easier compared the right hand segregation. To indicate which hand is dominating in a recording, we additionally give SNR values comparing the ground truth magnitude spectrograms $Y^{\mathcal{L}}$ and $Y^{\mathcal{R}}$ to the mixture magnitude spectrogram Y, see column six and seven of Table 9.2. For example for $\texttt{Chop28} - \texttt{15}$, $\mathrm{SNR}(Y^{\mathcal{L}}, Y) = 3.48$ is much higher compared to $\mathrm{SNR}(Y^{\mathcal{R}}, Y) = -2.47$ thus revealing the left hand dominance.

Using synthetic data, the audio recordings are already perfectly aligned to the MIDI files. To further evaluate the influence of the music synchronization step, we randomly distort the MIDI files by splitting them into 20 segments of equal length and by stretching or compressing each segment by a random factor within an allowed distortion range (in our experiments we used a range of ±50%). The results for these distorted MIDI files are given in column four and five of Table 9.2. Here, the left hand SNR for $\texttt{Chop28} - \texttt{15}$ decreases only moderately from 17.79 (pre-aligned MIDI) to 17.56 (distorted MIDI), and from 13.35 to 13.18 for the right hand. Similarly, the average SNR also decreases moderately from 13.45 to 13.27 for the left hand and from 12.05 to 11.90 for the right hand, which indicates that our synchronization works robustly in these cases. The situation in real world scenarios becomes more difficult, since here the note events of the given MIDI may not correspond one-to-one to the played note events of a specific recording. An example will be discussed in the next paragraph, see also Figure 9.5.

As mentioned before, signal-to-noise ratios and similar measures cannot capture the perceptual separation quality. Therefore, to give a realistic and perceptually meaningful impression of the separation quality, a website[5] is provided with audible separation results as well as visualizations illustrating the intermediate steps in the proposed procedure. Here, only real, non-synthetic audio recordings from the SMD and EA databases were used to illustrate the performance of the proposed method in real world scenarios. Listening to these examples does not only allow to quickly get an intuition of the method's properties but also to efficiently locate and analyze local artifacts and separation errors. For example, Figure 9.5 illustrates the separation process for `BeetOp111` using an interpretation by Egon Petri (European Archive). As a historical recording, the spectrogram of this recording (Figure 9.5c) is rather noisy and reveals some artifacts typical for vinyl recordings such as rumbling and cranking glitches. Despite these artifacts, the proposed model approximates the audio spectrogram well (w.r.t. to the euclidean norm) in most areas (Figure 9.5d). Also the resulting separation results are plausible, with one local exception. Listening to the separation results reveals that the trills towards the end of the first measure were assigned to the left instead of the right hand. Investigating the underlying reasons shows that the trills are not correctly reflected by the given MIDI file (Figure 9.5b). As a consequence, our score-informed approach cannot model this spectrogram area correctly as can be observed in the marked areas in Figures 9.5c and 9.5d. Applying the resulting separation mask (Figure 9.5e) to the original spectrogram leads to the trills being misassigned to the left hand in the estimated magnitude spectrogram as shown in Figure 9.5f.

Overall, we presented in this section a novel method for the decomposition of a monaural audio recording into musically meaningful voices. Here, the goal was to extend the idea

[5] `http://www.mpi-inf.mpg.de/resources/MIR/2011-ISMIR-VoiceSeparation/`

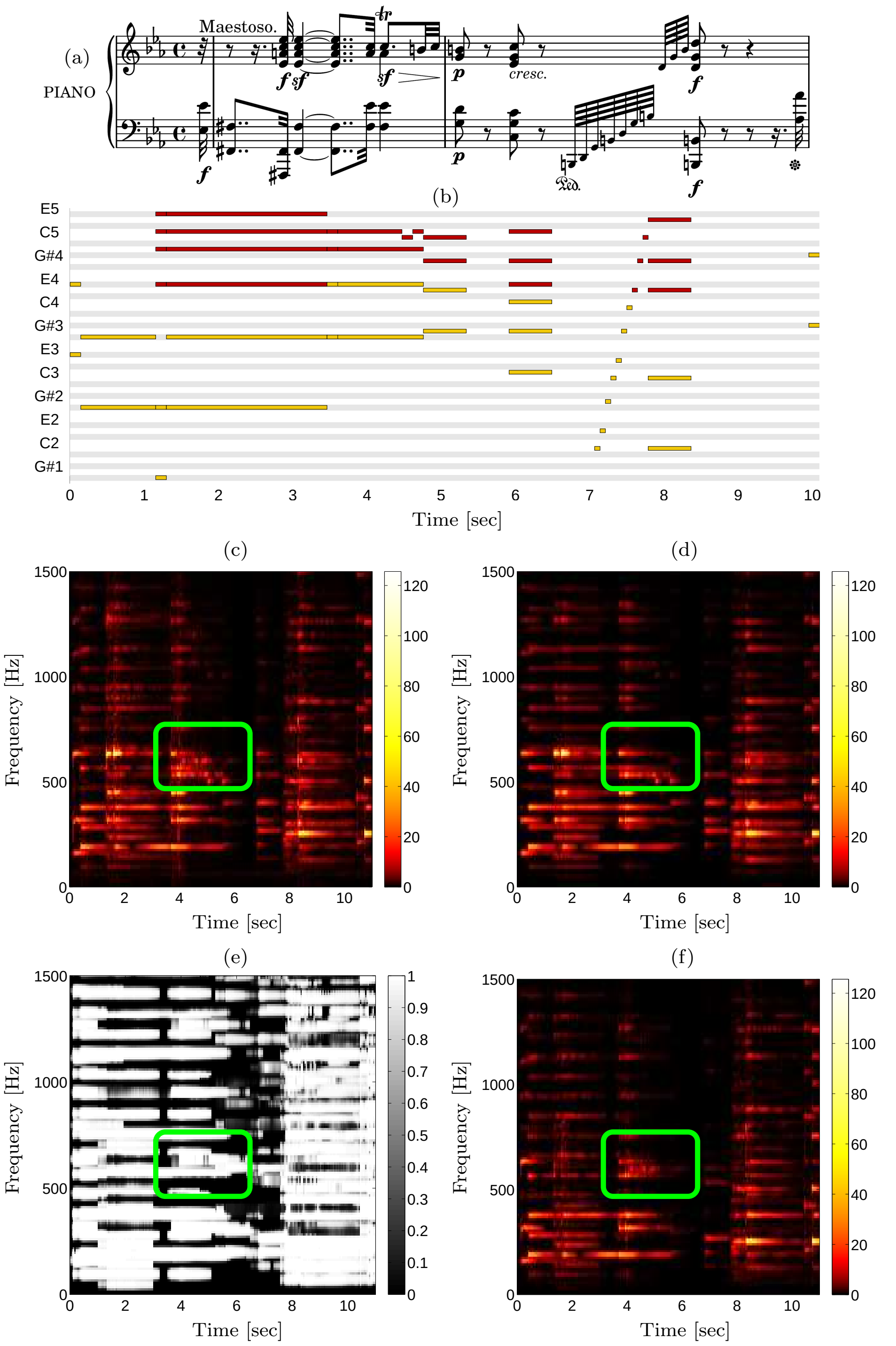


Figure 9.5. Illustration of the separation process for `BeetOp111`. **(a)** Score corresponding to the first two measures. **(b)** MIDI representation (Mutopia Project). **(c)** Spectrogram of an interpretation by Petri (European Archive). **(d)** Model spectrogram after parameter estimation. **(e)** Separation mask $M^{\mathcal{L}}$. **(f)** Estimated magnitude spectrogram $\hat{Y}^{\mathcal{L}}$. The area corresponding to the fundamental frequency of the trills in measure one is indicated using a green rectangle.

of an instrument equalizer to a voice equalizer which does not rely on statistical properties of the sound sources and which is able to emphasize or attenuate even single notes played by the same instrument. Instead of relying on pre-aligned MIDI files, the proposed score-informed procedure directly addresses alignment issues using high-resolution music synchronization techniques thus allowing for an adoption in real world scenarios. Initial experiments showed good results using synthetic as well as real audio recordings. In the future, it would be interesting to extend this approach with an onset model while avoiding the drawbacks discussed in [78], see also Chapter 8.

9.3 Application: Note Intensity Estimation

The score of a piece of music basically specifies note parameters such as the pitch, the onset position and the duration. Musical nuances beyond the score are subject to the interpretation by a musician. For example, timings and dynamics (intensities) are not taken as fixed constants and offer a musician the artistic freedom to form a piece of music in his or her own way. Also parameters referring to the timbre are often strongly influenced by the musician. Capturing these musical nuances is important for many different fields in music signal processing. For example, it allows for an automated analysis of differences between several interpretations of a piece of music, as done in the field of performance analysis [160,191]. A compact description of the nuances might also lead to more efficient compression approaches or to higher quality in applications of source separation.

To further demonstrate the potential of the proposed parametric model, we focus in this section on the estimation of note intensities in recordings of polyphonic piano music. In particular, given a MIDI file (representing the score) and an audio recording (representing an interpretation) of a piece of music, the idea is to parameterize the spectrogram of the audio recording using the proposed model in a first step. Then, we exploit that the model is based on note-event spectrograms each describing the part of a spectrogram that can be attributed to a specific note event. This way, we can derive the individual note intensities by analyzing the note-event spectrograms described by the model. A visual representation of these intensity values (Figure 9.6) provides support for a subsequent analysis of the music material. For example, such a visualization allows a user to identify and analyze dynamics-related differences between several interpretations of a piece on a note-level. Furthermore, describing musical properties beyond the score, the intensity values can be used to enrich a given MIDI representation with performance-specific subtleties. Overall, the presented approach demonstrates that score-informed source separation techniques can also be employed for analysis purposes and are useful beyond the goal of creating appealing separation results.

After describing how the proposed parametric model can be employed for estimating note intensities, we discuss some systematic experiments conducted on a database of piano recordings. However, because of a lack of annotated ground truth data, evaluating the quality of estimated note intensities is a challenging task itself. In the following experiments, we use audio recordings obtained by a Yamaha Disklavier. Equipped with optical sensors and electromechanical devices, such pianos allow for recording the key movements along with the acoustic audio data. On the one hand, the key movement data can be used

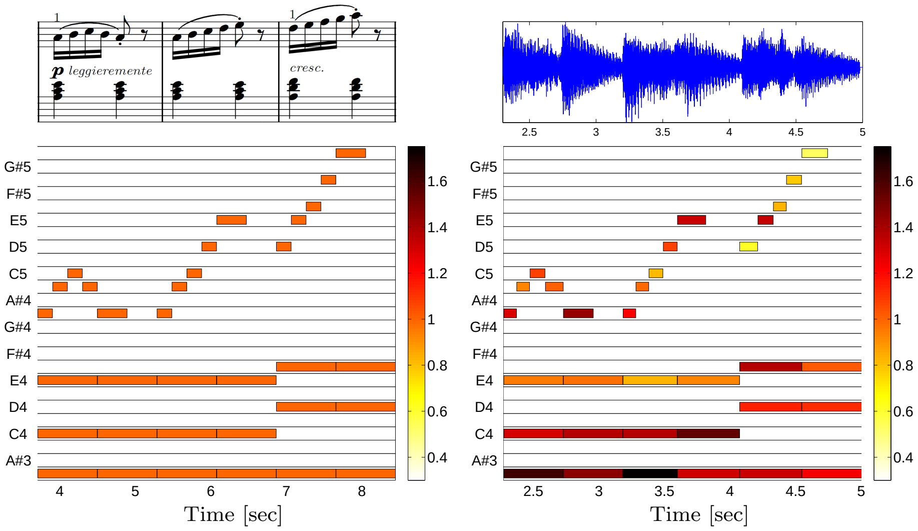


Figure 9.6. Illustration of our note intensity estimation procedure using three measures of Burgmüller, Op. 100, Etude No. 2 as an example. The color encodes the note intensity. **Left:** Note events taken from a MIDI generated from score data. The note intensities are constant for all notes. **Right:** Note events derived from our model approximating an audio recording.

to derive expected note intensities as described below. On the other hand, we can use the proposed procedure to estimate the note intensities using the audio data. Comparing the expected with the estimated note intensities allows for a first assessment of the estimation quality using real-world data.

9.3.1 Note Intensity Estimation

Again, given a MIDI file and a corresponding audio recording for a piece of music, we start by estimating parameter values, such that the audio spectrogram Y is approximated by $Y_\lambda^{\mathcal{S}}$ as precisely as allowed by the model. In particular, the model describes the note-event spectrograms Y_λ^s, which can be used to derive an intensity value for each note event as follows. The energy related to the s-th note event in frame n is given by

$$E_s(n) := \sum_{k\in[1:K]} Y_\lambda^s(k,n)^2.$$

To describe the intensity of a note event by a single value, we define

$$\mathcal{I}(s) := \max_{n\in[1:N]} E_s(n)^{0.3}.$$

Here, the exponent 0.3 is used so that the note intensity roughly approximates the perceived loudness of the human auditory system [50, chapter 8]. The result of our method is illustrated in Figure 9.6. Here, the left half of Figure 9.6 shows note events from a MIDI file generated from score data. The note intensities are constant for all note events. The right half shows the result of our procedure, where note timings and intensities are estimated from an audio recording.

9.3.2 Experiments

For our evaluation, we employ a database consisting of Disklavier recordings of various pieces from the Western classical music repertoire[6]. For each audio recording, the Disklavier automatically generates a MIDI file, which can be regarded as a kind of ground truth annotation. For example, the MIDI onset-positions and durations correspond closely to those in the audio recordings. Particularly interesting for our evaluation are the velocity values, which symbolically encode the dynamics in a MIDI file. Here, a translation of the symbolic velocity values to physical note intensities would allow for a simple way to evaluate our method. However, this translation is not trivial for polyphonic music because of the complex interaction of multiple sound sources and other acoustical effects. To approximate such a translation, a dictionary was built up that maps a pitch and a velocity to a physical note intensity. In a first step, training data was used consisting of single note events played at several velocities. In a second step, the dictionary was refined using five pieces from our database. These pieces are marked with a star in the result table. Initial experiments using monophonic recordings revealed that the dictionary is capable of estimating the note intensity with an accuracy of about five to ten percent.

Given a piece from our database, we can use the dictionary in combination with the pre-aligned annotation MIDI file to estimate the intensity for each note event. We denote the resulting values by $(\mathcal{I}_{\mathrm{dic}}(s))_{s\in[1:S]}$, where $[1:S]$ denotes the index set for the note events. Then, ignoring the given velocity values, we use the proposed method to estimate the individual note intensities using the MIDI and the audio data. The resulting values are denoted by $(\mathcal{I}(s))_{s\in[1:S]}$. To compensate for different overall recording levels, we only consider relative note intensities in our evaluation. To this end, we normalize the S-dimensional vectors $\mathcal{I}_{\mathrm{dic}}$ and $\mathcal{I}$ with regard to the Euclidean norm and compare them in terms of a percentage error defined by

$$\mathrm{PE} := \left(100 \cdot \left|\frac{\tilde{\mathcal{I}}(s) - \tilde{\mathcal{I}}_{\mathrm{dic}}(s)}{\tilde{\mathcal{I}}_{\mathrm{dic}}(s)}\right|\right)_{s\in[1:S]}$$

where $\tilde{\mathcal{I}}$ and $\tilde{\mathcal{I}}_{\mathrm{dic}}$ denote the normalized versions of $\mathcal{I}$ and $\mathcal{I}_{\mathrm{dic}}$, respectively. Using the spectral envelope model φ^1, the mean and standard deviation of PE are given in the third and fourth column of Table 9.3. For example, for Bach's BWV849-02 one gets a mean percentage error of 9.3 with a standard deviation of 5.5. The average over all pieces is 16.9 with a standard deviation of 9.3. With an intrinsic error of five to ten percent induced by the estimated dictionary, this indicates a reasonable estimation quality.

To further evaluate the influence of the music synchronization step, we randomly distort the pre-aligned MIDI files by splitting them into 20 segments of equal length and by stretching or compressing each segment by a random factor within an allowed distortion range (in the experiments we use a range of ±50%). The results are shown in the fifth and sixth column of Table 9.3. Here, the average error for Bach's BWV849-02 increases only moderately from 9.3 (pre-aligned MIDI) to 9.5 (distorted MIDI). Similarly, the average error also increases moderately from 16.9 to 17.2, which indicates that our synchronization works robustly in most cases.

[6] All files are part of the SMD database [124], which can be obtained from the website `http://www.mpi-inf.mpg.de/resources/SMD/`.

Composer	**Piece**	**Proposed**		**Proposed**		**Baseline**		**Baseline**	
		pre-aligned		distorted		pre-aligned		distorted	
		Mean	STD	Mean	STD	Mean	STD	Mean	STD
Bach	BWV849-01*	9.3	5.3	9.5	5.5	31.5	25.9	31.9	26.5
Bach	BWV849-02	9.3	5.5	9.5	5.7	28.7	23.9	29.3	24.5
Bach	BWV871-01	11.0	6.2	11.4	6.3	27.5	21.4	28.2	21.7
Bach	BWV871-02	7.7	5.1	7.8	5.2	24.6	20.0	25.0	20.3
Bach	BWV875-01	13.9	6.7	14.1	6.9	31.6	26.2	32.0	26.8
Bach	BWV875-02	8.3	4.9	8.5	5.0	28.2	25.4	28.8	26.1
Beethoven	Op027No1-01	12.5	7.1	13.0	7.2	39.4	28.0	40.5	28.6
Beethoven	Op027No1-02*	10.3	6.5	10.6	6.7	35.2	24.4	35.9	24.7
Beethoven	Op027No1-03	13.6	7.3	14.0	7.5	45.6	32.0	46.6	33.3
Beethoven	Op031No2-01	16.1	8.7	16.5	8.9	36.1	29.9	36.9	30.4
Beethoven	Op031No2-02	27.2	14.5	27.8	14.7	38.6	27.5	39.2	27.8
Beethoven	Op031No2-03	13.2	8.1	13.5	8.2	34.9	29.4	35.4	30.1
Brahms	Op010No1*	13.8	7.3	14.0	7.5	38.9	29.3	39.4	29.8
Brahms	Op010No2	13.6	7.9	14.2	8.0	41.3	32.6	42.5	33.8
Chopin	Op010-03	25.2	13.0	25.4	13.2	35.1	31.0	35.5	32.2
Chopin	Op010-04	25.0	13.2	25.8	13.6	36.0	34.2	36.9	35.1
Chopin	Op026No1	22.6	13.2	22.9	13.5	34.1	34.8	34.5	35.2
Chopin	Op026No2	23.6	14.2	23.8	14.6	34.7	33.2	35.0	33.8
Chopin	Op028-01	22.9	11.4	23.6	11.9	37.7	25.8	38.6	26.7
Chopin	Op028-03	19.0	12.2	19.3	12.6	33.4	36.8	33.9	38.0
Chopin	Op028-04	19.5	11.6	20.2	12.0	29.6	29.2	30.4	30.3
Chopin	Op028-11	18.8	9.1	19.0	9.3	25.6	23.3	25.9	23.5
Chopin	Op028-15	18.0	9.2	18.7	9.4	24.7	19.3	25.4	19.5
Chopin	Op028-17	22.1	10.7	22.9	11.0	31.1	24.9	32.0	25.4
Chopin	Op029*	20.1	11.6	20.8	11.9	32.7	34.9	33.6	35.9
Chopin	Op048No1	26.0	11.2	26.2	11.3	39.0	34.0	39.4	35.3
Chopin	Op066	22.4	13.5	22.7	13.8	31.0	37.3	31.4	38.6
Haydn	Hob017No4*	14.8	8.1	15.4	8.3	44.5	32.7	45.8	33.4
Rachman.	Op039No1	15.5	9.0	16.0	9.2	36.6	28.0	37.6	28.6
Skryabin	Op008No8	10.1	5.6	10.4	5.8	24.5	21.5	25.1	21.8
Average		**16.9**	**9.3**	**17.2**	**9.5**	**33.8**	**28.6**	**34.4**	**29.3**

Table 9.3. Estimation quality of our proposed method (using spectral envelope model φ^1) and a baseline. Shown are the mean and standard deviation of the percentage errors PE and $\mathrm{PE}_{\mathrm{base}}$. The results for using pre-aligned and temporally distorted MIDI files are listed separately. The stars indicate pieces, that have been used to refine our note intensity dictionary.

To get a better understanding of these numbers, we next conduct a simple baseline experiment. Our baseline method starts by computing the magnitude spectrogram for a given audio recording. Then, exploiting the onset, duration and pitch information given by a synchronized MIDI file, the method locates the spectrogram bins that are related to the first five partials of a given note event. To locate the partials correctly, we incorporate simple heuristics to estimate the fundamental frequency for each pitch. Using only the located bins, the baseline method then computes the energy in each time frame and derives a note intensity from the maximum of these energy values. In some sense, this method roughly represents what is possible without using a sophisticated overtone model. We denote the resulting intensity values by $(\mathcal{I}_{\mathrm{base}}(s))_{s\in[1:S]}$. After normalizing $\mathcal{I}_{\mathrm{base}}$, we compare $\mathcal{I}_{\mathrm{base}}$ and $\mathcal{I}_{\mathrm{dic}}$ in terms of a percentage error $\mathrm{PE}_{\mathrm{base}}$ as described above. The results of our baseline experiments using both pre-aligned and distorted Disklavier MIDI files are shown in columns seven and eight as well as nine and ten of Table 9.3, respectively. Using pre-aligned MIDI files, the error for Bach's BWV849-02 is 28.7, which is over three times higher compared to an error of 9.3 for the proposed method. Furthermore, the average error when using the baseline method is with 33.8 twice as high as the error when using the proposed method. Overall, the baseline experiments indicate that the proposed method indeed yields note intensities with a reasonable estimation quality.

Overall, we have discussed in this section a first method for the estimation of note intensities in music recordings. While such an estimation is a very challenging task in general, experiments on polyphonic piano music revealed measurable advantages of the proposed method over a given baseline. However, for the future there are still several challenges to be solved. For example, the definition of a suitable ground truth for note intensities is an open problem. Here, one has to consider complex acoustical effects which strongly depend on the recording conditions. In particular, the room acoustics and the interaction of multiple sound sources with the resonance body of an instrument are important factors. However, this cannot be achieved with a simple dictionary based ground truth. A better ground truth is also a requirement for a more detailed analysis of the capabilities and limitations of the proposed procedure, and the underlying model in particular. A first manual inspection already revealed that the procedure tends to incorrectly model note events with very low pitches. Here, other spectral representations or multi-resolution spectrograms should be considered that offer a higher frequency resolution for lower pitches. Furthermore, the procedure could be improved by incorporating perceptually oriented methods to assess the intensity or loudness of a note event [50, 115].

Chapter 10

Score-Informed Non-Negative Matrix Factorization

In recent years, methods for separating musically meaningful sound sources from monaural music recordings have been applied to many music processing tasks. For example, techniques to extract individual instrument tracks have been incorporated into approaches for instrument recognition [72] or instrument-wise equalization [78]. A lot of these techniques rely on some variant of non-negative matrix factorization (NMF) [98], or on an equivalent formulation such as probabilistic latent component analysis (PLCA) [167]. Here, the idea is to decompose the magnitude spectrogram of a given recording into a set of template (column) vectors and activation (row) vectors. However, as discussed in more detail below, template vectors learnt by NMF-based approaches are often hard to interpret and lack explicit semantics. To obtain musically meaningful vectors, the original NMF can be modified such that each template vector is associated with a single musical pitch. To this end, many approaches specify the template vectors using a parametric model. For example, the template vectors in [72] are described by a source/filter model, in [73] by harmonic atoms, in [180] by comb filtered spectral vectors, and in [78, 196] by spectrally and/or temporally localized Gaussians. Also the approach described in Chapter 9 can be interpreted in this way. On the one hand, a parametric model allows for a straightforward integration of musical knowledge. For example, in [196] the authors extend their harmony-based approach with a percussive component exploiting the typical spectral shape around onsets. Furthermore, allowing only solutions that are valid within the model, the parametric approaches offer a high degree of robustness. On the other hand, the parameter estimation and the resulting spectrogram approximation can be inaccurate in the case that some model assumptions are violated. Additionally, the parameter estimation process is often computationally expensive.

In this chapter, we discuss a novel method that combines the efficiency and flexibility of classic NMF with advantages of parametric approaches. The method is based on a strategy originally presented in [151]. Here, the underlying idea is to enforce a harmonic structure for the template vectors by setting those entries to zero that are not in a neighborhood of an expected partial. Then, using multiplicative update rules guarantees that these constraints remain valid during the subsequent learning process. We extend this idea in several ways.

First, opposed to previous methods, we simultaneously constrain the template vectors as well as the activations. In this way, instead of just specifying *what* is expected we additionally specify *when* something is expected. The use of these double constraints becomes possible by exploiting available score information in form of a MIDI file. Here, we again rely on the high-resolution synchronization techniques developed in Chapter 7 to align the MIDI file with a given recording. As a second extension, we exploit the robustness gained by the double constraints to integrate template vectors that represent percussive elements such as onsets. As experiments show, these double constraints in combination with the onset template vectors stabilize the resulting separation results and lead to an increased overall separation quality. Altogether, the proposed method combines the expressive power of some parametric approaches with the efficiency of classic NMF, while still being easy to implement.

The remainder of this chapter is organized as follows. In Section 10.1, we describe the classical NMF framework and introduce the novel score-informed variant using double constraints. In Section 10.2, we employ the proposed NMF model for the separation of note groups (e. g. the left or right hand) from monaural piano recordings. Then, in Section 10.3, we report on systematic experiments, where we compare the performance of the proposed NMF model with the parametric model described in Chapter 9. Conclusions and prospects on future work are given in Section 10.4. Further related work is discussed in the respective sections.

10.1 Score-Informed Constraints in NMF

Non-negative matrix factorization (NMF) has turned out to be a powerful tool for modeling, analyzing and separating the constituent parts of polyphonic music recordings. NMF variants form the basis of methods for pitch estimation [8,171], source separation [186], and pattern and motive identification [188]. In this section, we show how the classical NMF framework can be extended in a straightforward way using available score data. As we will see, the basic idea is to replace the standard NMF initialization without changing the established and computationally efficient NMF learning process. This way, a musically meaningful factorization structure can be enforced, which stabilizes NMF-based source separation.

10.1.1 Non-Negative Matrix Factorization

In classic non-negative matrix factorization, one approximates a spectral representation of a given recording by a product of two non-negative matrices. More exactly, given a magnitude spectrogram $V \in R_{\geq 0}^{M \times N}$ of a music recording, NMF seeks to find non-negative matrices $W \in R_{\geq 0}^{M \times K}$ and $H \in R_{\geq 0}^{K \times N}$ such that $V \approx W \cdot H$, see Figure 10.1a. In this context, the columns of W are often referred to as *template vectors* and the rows of H as the corresponding *activations*. As an example, Figure 10.1b shows an example factorization for a recording of Chopin's Op. 28 No. 15. Here, the free parameter K is set to the number of pitches that occur in the corresponding part of the piece. In this case, the activation

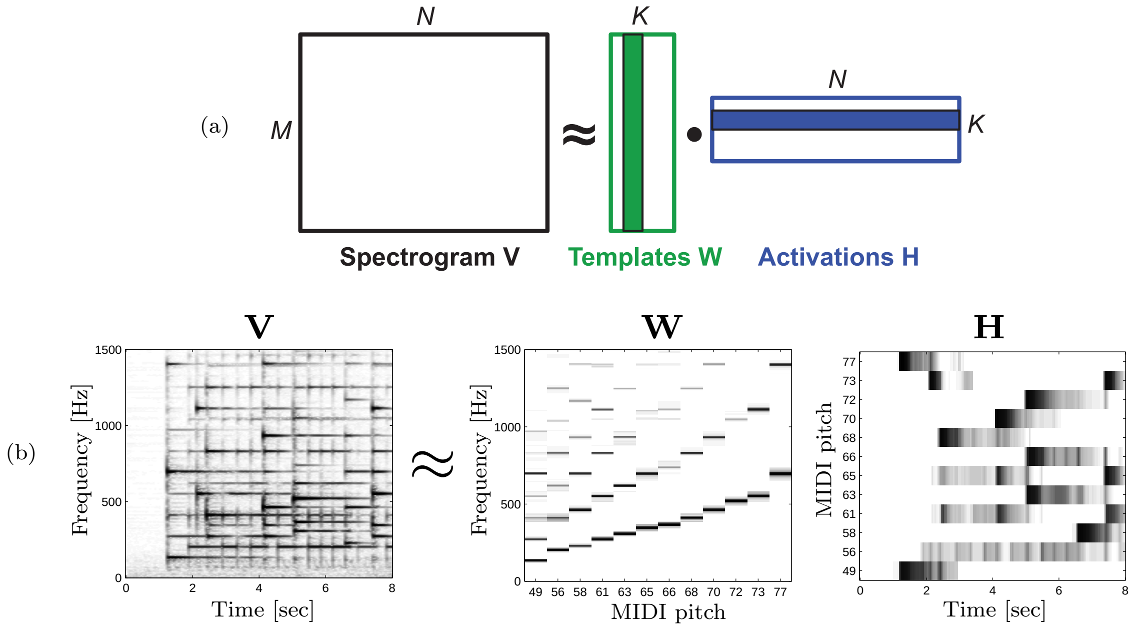


Figure 10.1. Non-negative matrix factorization (NMF). **(a)** A given non-negative matrix V is approximated as a product of two non-negative matrices W and H typically having a much smaller rank. **(b)** Example factorization of a magnitude spectrogram for an audio recording of Chopin's Op. 28 No. 15 taken from the SMD database [124].

matrix H is similar to a pianoroll representation and shows when these pitches become active.

In the classical approach for computing such a factorization, one employs some form of gradient descent to minimize a distance measure $D(V, W \cdot H)$ with respect to W and H, where D is typically based on the Euclidean norm or a variant of the Kullback-Leibler divergence, see [98]. However, to account for the non-negativity constraints for W and H, one usually has to resort to rather complex optimization algorithms [135]. As an easy-to-implement alternative, Lee and Seung proposed multiplicative update rules, which are derived from gradient descent by choosing a specific step size [98]. Using the popular Kullback-Leibler variant as a distance measure, these rules can be written as

$$H_{kn} \leftarrow H_{kn} \frac{\sum_i W_{ik} V_{in}/(WH)_{in}}{\sum_j W_{jk}} \quad \text{and} \quad W_{mk} \leftarrow W_{mk} \frac{\sum_i H_{ki} V_{mi}/(WH)_{mi}}{\sum_j H_{kj}},$$

where $m \in [1 : M]$, $n \in [1 : N]$, and $k \in [1 : K]$. For vectorized programming languages such as Matlab it is useful to express these rules in matrix notation:

$$H \leftarrow H \circ \frac{W^\top \cdot (\frac{V}{W \cdot H})}{W^\top \cdot J} \quad \text{and} \quad W \leftarrow W \circ \frac{(\frac{V}{W \cdot H}) \cdot H^\top}{J \cdot H^\top},$$

where the $\cdot$ operator denotes the usual matrix product, the $\circ$ operator denotes the Hadamard product (point-wise multiplication), $J \in \mathbb{R}^{M \times N}$ denotes the all-one matrix, and the division is understood point-wise. These multiplicative update rules have several interesting properties. First, the Kullback-Leibler distance measure is non-increasing under these

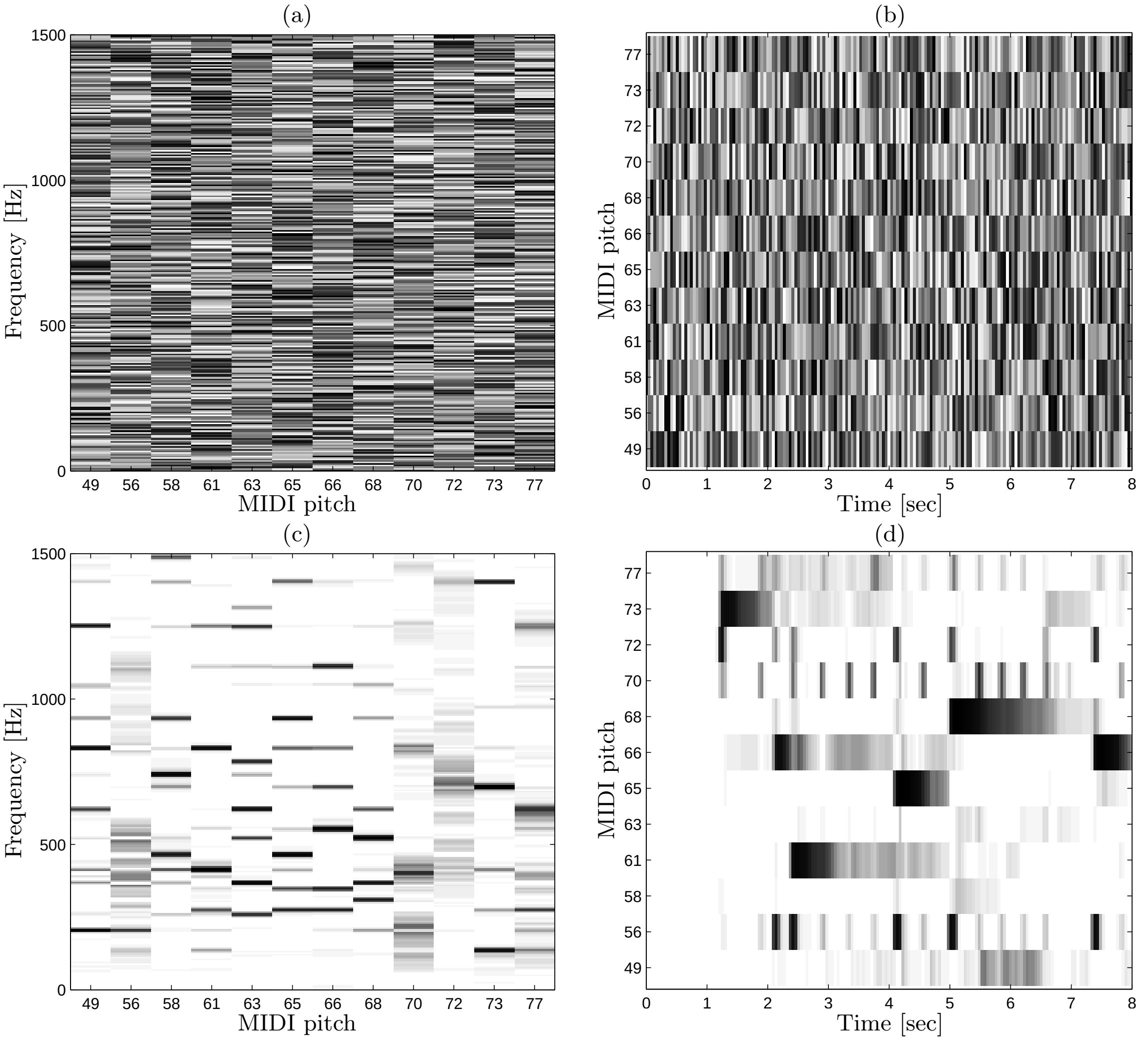


Figure 10.2. Classical NMF factorization for the magnitude spectrogram shown in Figure 10.1. **(a)** Random initialization of W. **(b)** Random initialization of H. **(c)** Learnt W. **(d)** Learnt H.

rules[1]. Furthermore, initializing W and H with non-negative random values, these rules guarantee that W and H remain non-negative during the entire learning process.

In general, however, NMF factorizations computed in this classical way cannot be as easily interpreted as the example shown in Figure 10.1b. For example, Figure 10.2 shows a factorization based on the classical NMF algorithm for the magnitude spectrogram shown in Figure 10.1b. Here, the initialization of W and H with random values does not lead to a musically meaningful structure in the computed factorization. Furthermore, the free parameter K is usually set according to simple rules of thumb that usually do not account for any musical prior knowledge. As a result, the factorization often becomes completely unpredictable and lacks clear musical semantics.

[1] As pointed out by several authors [4,102,197], however, multiplicative rules do not guarantee in general convergence to a local minimum of the employed distance measure.

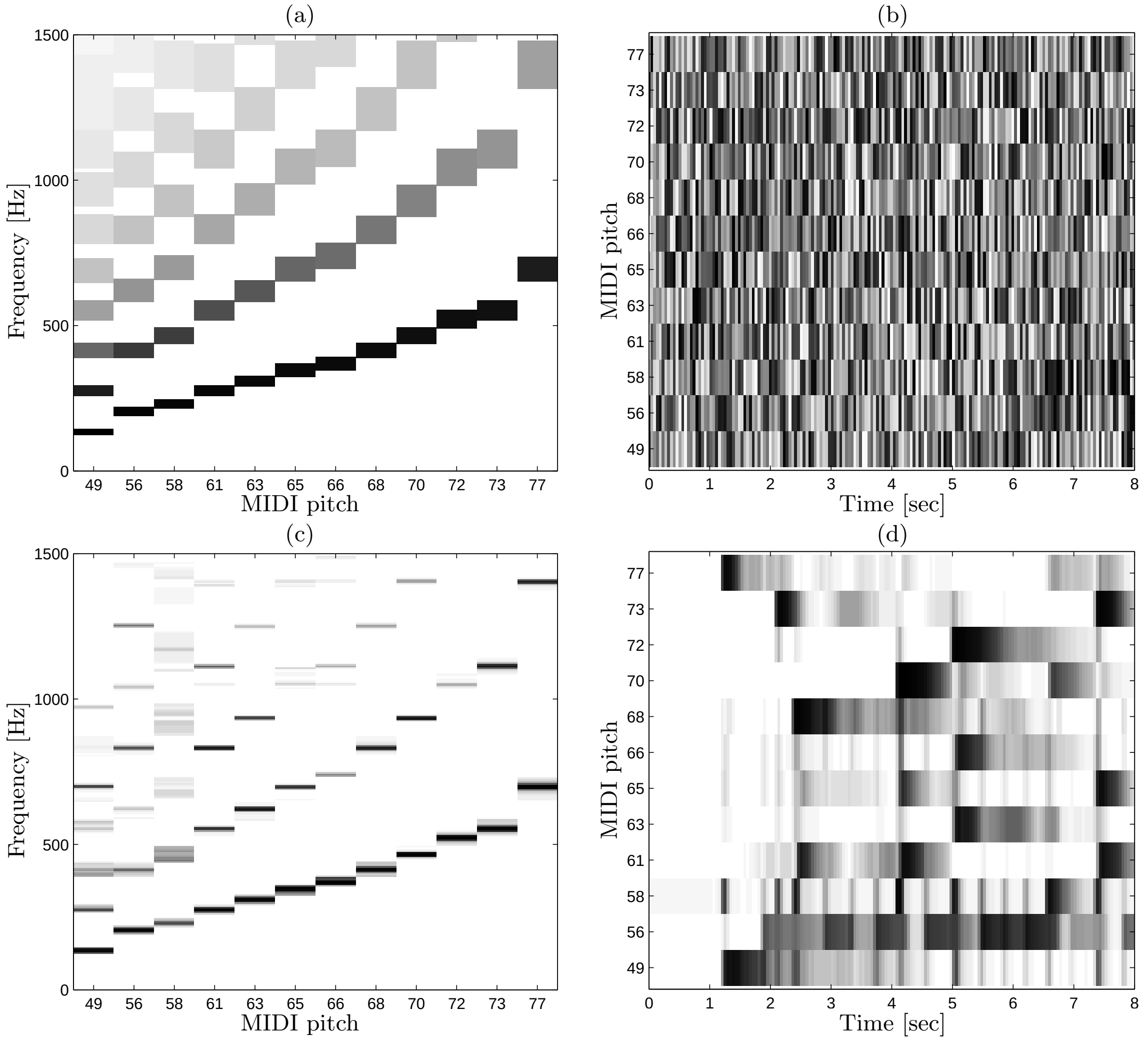


Figure 10.3. NMF factorization resulting from harmonic initialization of the template vectors for the magnitude spectrogram shown in Figure 10.1. **(a)** Harmonic initialization of W. **(b)** Random initialization of H. **(c)** Learnt W. **(d)** Learnt H.

10.1.2 Constraints in NMF

Another important property of multiplicative update rules is that zero-valued entries remain zero during the entire learning process. Combined with musically informed initialization schemes, this yields a straightforward way to enforce a specific structure of a factorization as proposed in [151, 185]. Here, one first creates one template vector for each possible MIDI pitch. Then, a harmonic structure is imposed by inserting zero-valued entries into the template initialization at positions where no partial is expected for a given pitch, see Figure 10.3a. The remaining entries are initialized according to a simplified overtone model. As we see in Figure 10.3c, the learning process based on multiplicative rules not only retains this harmonic structure but further refines it such that each template vector has a clear pitch association. This is a significant gain in structure compared to the unpredictable results computed via standard NMF as shown in Figure 10.2. However, looking at the resulting factorization in Figure 10.3c/d reveals that template vectors are

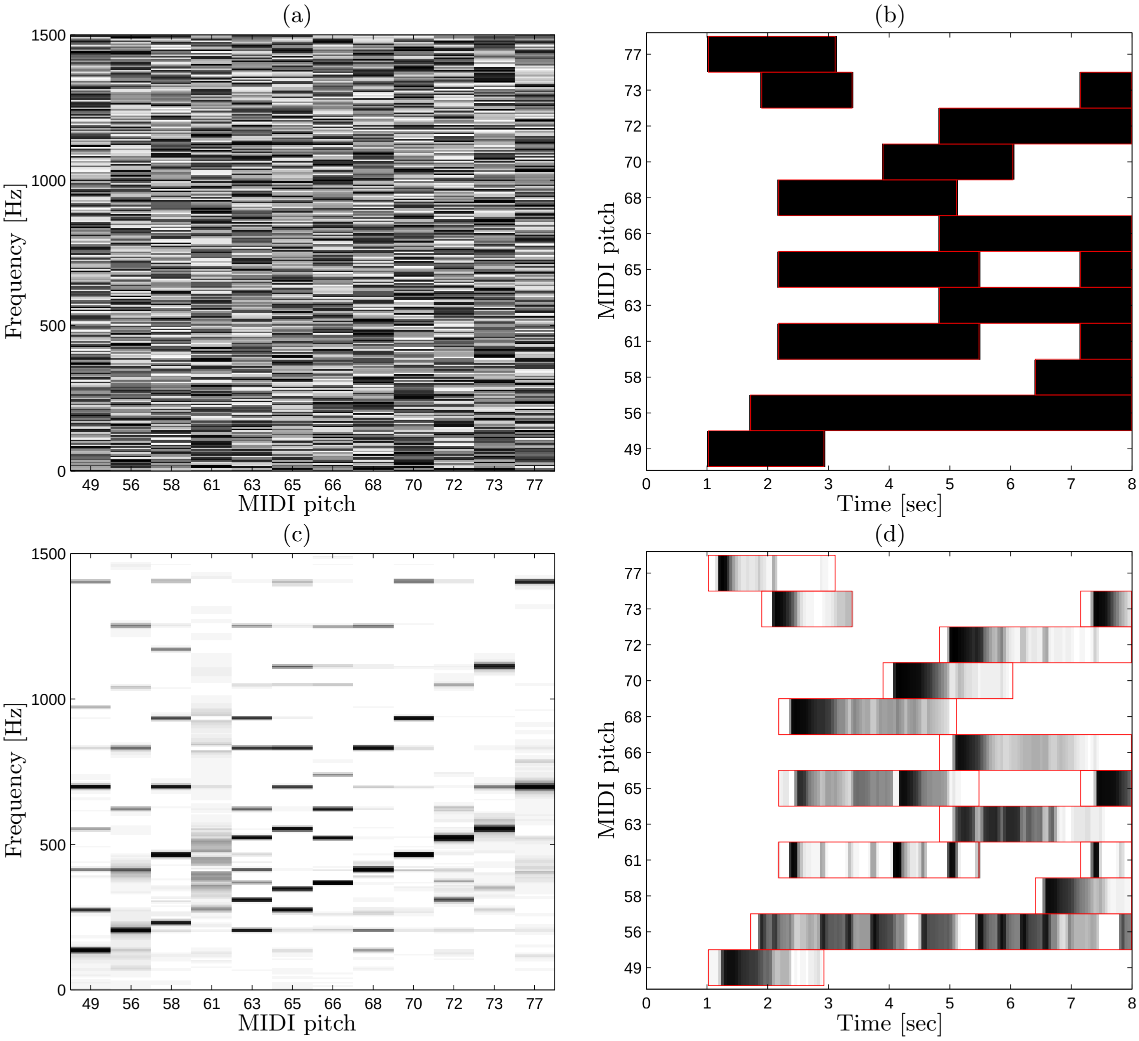


Figure 10.4. NMF factorization resulting from harmonic initialization of the template vectors for the magnitude spectrogram shown in Figure 10.1. **(a)** Random initialization of W. **(b)** Score-informed initialization of H. **(c)** Learnt W. **(d)** Learnt H.

still often “misused”, for example to represent onsets. This becomes particularly apparent in the template for MIDI pitch 58, where energy is distributed over a larger number of frequency bands compared to the other templates (Figure 10.3c). Here, instead of representing harmonic components of the spectrum, the template is misused to explain parts of the broadband energy distribution related to onsets. This is also reflected by the short-term intensity bursts in the corresponding activation row (Figure 10.3d). In the following, we refer to this initialization strategy as I_W.

Alternatively, another possibility is to constrain the activations instead of the template vectors. To this end, one can mark suitable regions in H where a given pitch is expected while setting the remaining entries to zero, see Figure 10.4b. This results in a similar factorization as the one using I_W. However, the results depend strongly on the input data. In our example, several pitches appear only in groups of two, such that the corresponding template vectors tend to be mixtures of those pitches. However, such conditions usually

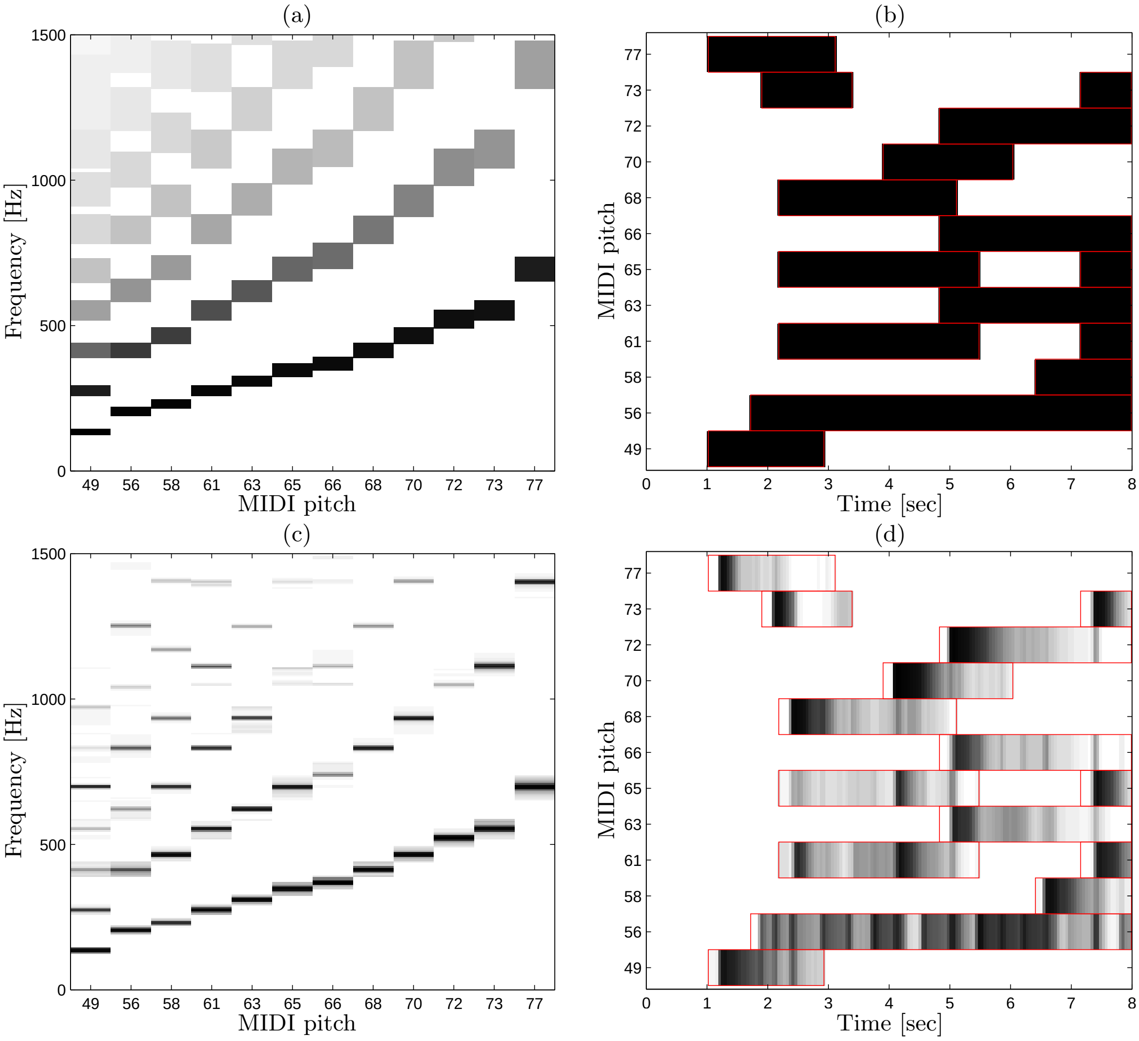


Figure 10.5. NMF factorization resulting from harmonic initialization of the template vectors and score-informed activation constraints for the magnitude spectrogram shown in Figure 10.1. **(a)** Harmonic initialization of W. **(b)** Score-informed initialization of H. **(c)** Learnt W. **(d)** Learnt H.

do not occur when using more extensive audio material instead of just short snippets. We refer to this initialization strategy in the following as I_{H}.

Opposed to previous methods, the main idea in this chapter is to constrain *both* the template vectors and the activities, see Figure 10.5. As to be expected, such double constraints lead to an increased stability and robustness of the factorization. While this will be experimentally shown in Section 10.3, it can also be observed in our example, see Figure 10.5. Here, almost all template vectors have a well-defined harmonic structure. We refer to this combined strategy as I_{WH}. Furthermore, the robustness of I_{WH} even allows for introducing additional template vectors dedicated to describe onsets. This further stabilizes the factorization and leads to even more meaningful template vectors, see Figure 10.6. In the next subsection, we describe this strategy, referred to as $I^{\mathrm{O}}_{\mathrm{WH}}$, in more detail.

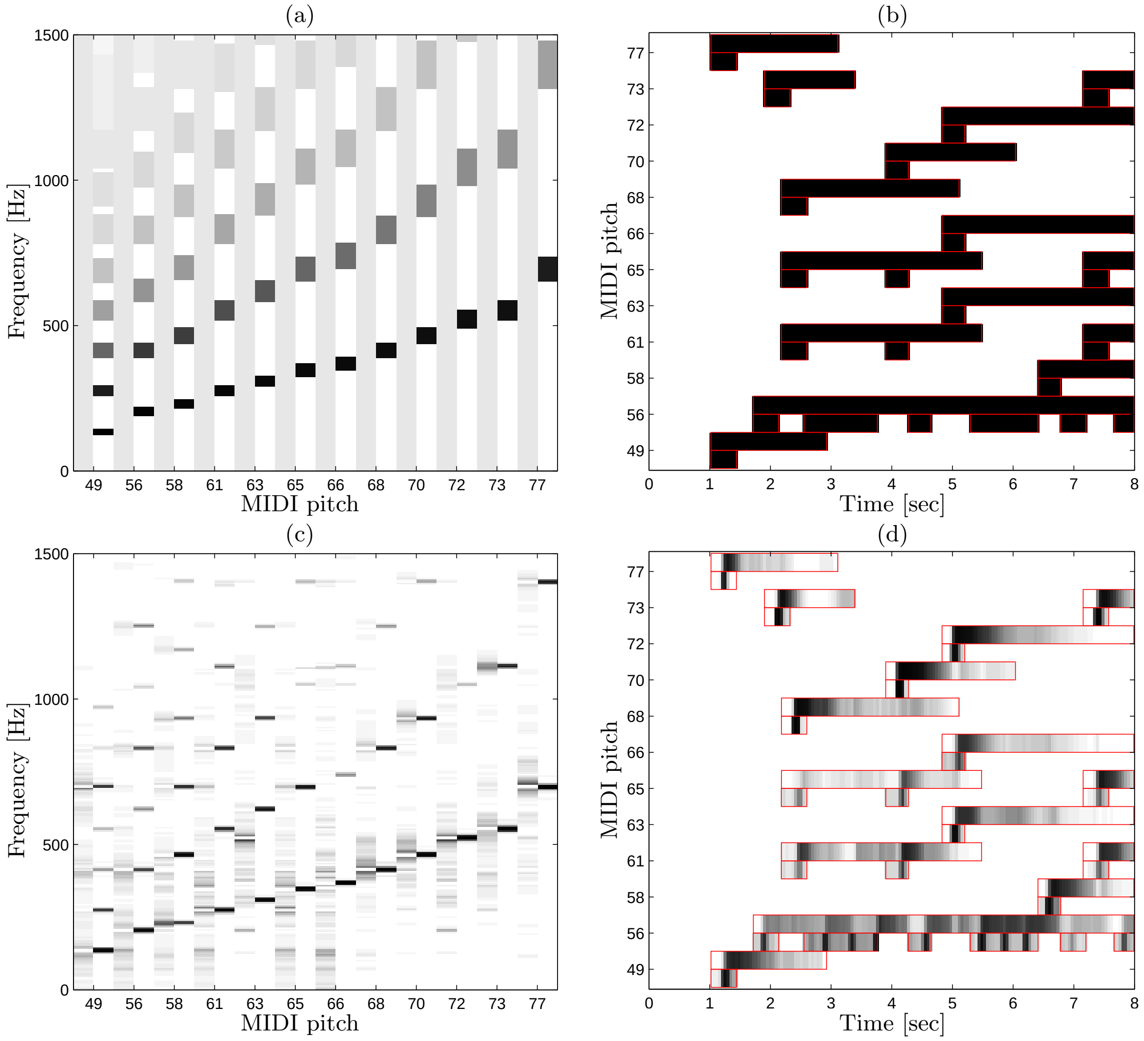


Figure 10.6. Extended NMF model with additional onset templates for the magnitude spectrogram shown in Figure 10.1. **(a)** Initialization of harmonic and onset template vectors in W. **(b)** Score-informed initialization of the corresponding activations in H. **(c)** Learnt W. **(d)** Learnt H.

10.1.3 Proposed Method

Overall, to use strategy $I_{\mathrm{WH}}^{\mathrm{O}}$, one needs to suitably initialize onset and harmony template vectors as well as their activations. After that, only the standard NMF updates rules have to be applied. For the harmony template vectors, the proposed procedure essentially follows [151]. To this end, each vector is assigned to a pitch and then initialized such that only areas around the partials are non-zero. We choose the size of these areas relatively generous in order to be flexible in dealing with potential inharmonicities of the recorded instrument or non-standard tunings. More exactly, the area for the n-th partial of pitch p corresponds to the frequency range $(n \cdot f(p-\phi), n \cdot f(p+\phi))$, where ϕ is a parameter in semitones to control the size of these areas (we use $\phi = 1$ in the following experiments). Here, $f : \mathbb{R} \to \mathbb{R}_{\geq 0}$ defined by $f(p) := 2^{(p-69)/12} \cdot 440$ maps the pitch to the frequency scale. Furthermore, since the lower partials usually carry most of the energy, we set all entries in the n-th area to $1/n^2$, see Figure 10.6a. In a next step, we initialize the onset template vectors. Here, opposed to many other approaches, we take into account that the spectral shape for onsets is for many instruments (including the piano) not the same as for white noise but depends on the respective pitch with the energy being concentrated around the partials. Therefore, we use one onset vector for each pitch. Contrary to the harmonic templates, we do not enforce here any spectral constraints but initialize the onset templates uniformly and let the learning process derive their shape. To compensate for this lack of constraints, we apply more rigid restrictions on the activation side.

Next, to meaningfully initialize the activations, the proposed method exploits available score information given in the form of a MIDI file. Here, instead of unrealistically assuming that a perfectly aligned MIDI file is available (as it is done in many of the previously described score-informed source separation methods), we employ the high-resolution music synchronization approach introduced in Chapter 7 to determine for each MIDI note event its corresponding position in the audio recording [47]. To impose the activation constraints, we essentially initialize H to look like a piano roll representation of the synchronized MIDI file. Starting with the activations for the harmony template vectors, we extract a pitch, as well as an onset and offset position from each MIDI event. Then, we set the corresponding entries in H to 1, while all remaining entries are set to zero. To account for possible alignment inaccuracies that occur using automatic synchronization procedures, we relax these constraints to some degree. To this end, we additionally set all entries in H in a $\mathrm{tol}_{\mathrm{on}}$-neighborhood around onsets and in a $\mathrm{tol}_{\mathrm{off}}$-neighborhood around offsets to 1 (in our experiments we use $\mathrm{tol}_{\mathrm{on}} = 0.2$ seconds and $\mathrm{tol}_{\mathrm{off}} = 1$ second). Then, in a final step, we initialize the activations for the onset template vectors. Here, we place more strict constraints by only setting entries in a $\mathrm{tol}_{\mathrm{on}}$-neighborhood around the MIDI onset positions to 1, Figure 10.6b.

Comparing the $I^{\mathrm{O}}_{\mathrm{WH}}$ factorization (Figure 10.6) to the others (Figures 10.2-10.5), we see that the harmonic vectors of $I^{\mathrm{O}}_{\mathrm{WH}}$ have the clearest harmonic structure with most partials being very sharp in frequency direction. Here, a reason is that the percussive broadband energy is now well-described by the onset vectors, such that onsets have significantly less disturbing influence on the harmonic vectors. Furthermore, most onset vectors are activated in an impulse-like manner at the start of note events, which indeed indicates their use for representing onsets. Overall, making use of double constraints, the initialization strategy $I^{\mathrm{O}}_{\mathrm{WH}}$ allows for computing musically meaningful factorizations including a dedicated representation of onsets. It combines the expressive power of some parametric approaches with the efficiency of classic NMF, while still being easy to implement. Furthermore, as shown in the next section, it is robust regarding smaller alignment errors as well as regarding potential inharmonicities of an instrument or non-standard tunings.

10.2 Separation Process

Next, we employ the NMF factorization based on the $I^{\mathrm{O}}_{\mathrm{WH}}$ strategy to separate note groups such as a melody line, the staff of the right hand, a specific motive, or the accompaniment from a given recording. The only requirement is that the notes to be considered are somehow specified by the user or by some labeling of the score. Similar to the methods presented in Chapter 9, we consider here the task of separating the left from the right

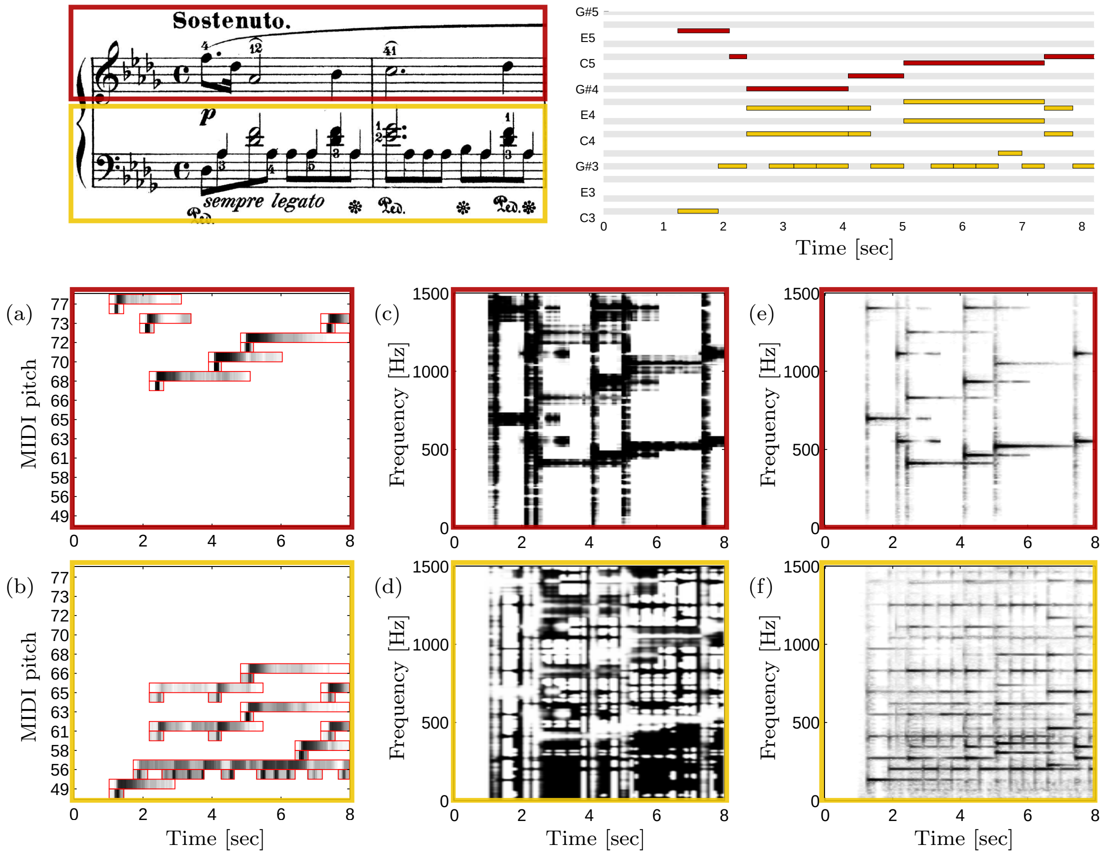


Figure 10.7. Illustration of the separation process for the left and the right hand. **(a)**/**(b)**: Partition of the activation matrix H (Figure 10.6d) into H_L and H_R. **(c)**/**(d)**: Masking matrices M_L and M_R. **(e)**/**(f)**: Separated spectrograms.

hand staff as specified by a given score as an illustrating example, see Figure 10.7a. While staffs do not always correspond to musically meaningful note groups, it demonstrates how note groups could be easily specified in a natural way.

For the separation, we exploit that every non-zero entry in H is associated with a specific note event, see Figure 10.6d. Therefore, we can partition H into two new matrices H_L and H_R, which contain either the activations for the left or the right hand, see Figure 10.7a/b. A straightforward way to create an audible separation result could be to multiply these two matrices with the template matrix W, shown in Figure 10.6c, and to invert the resulting spectrogram. As a tool for approximating the original magnitude spectrogram, however, NMF-based models usually do not capture every spectral nuance in a given recording. Therefore, the resulting audio recording would sound rather unnatural.

An alternative to this direct sonification is commonly referred to as masking. Similar to the separation process described in Section 9.2, one first derives masking matrices via

$$M_L := \frac{WH_L}{WH+\epsilon} \qquad \text{and} \qquad M_R := \frac{WH_R}{WH+\epsilon},$$

where the division is understood point-wise and ϵ is a small positive constant to avoid a potential division by zero, see Figure 10.7c/d. M_L and M_R have the same size as the original spectrogram V and, having values between 0 and 1, indicate how strongly each entry in V belongs to either the left or the right hand. Multiplying these masking matrices point-wisely with V, one obtains a separated spectrogram for the left and the right hand, see Figure 10.7e/f. Finally, to obtain the separated audio signals, one applies an inverse DFT in combination with an overlap-add technique to the separated spectrograms. The necessary phase information is provided by the original spectrogram. This way, masking-based separation allows for preserving most spectral details of the original recording, which is important to create acoustically appealing results. However, by filtering the original audio data, masking may also retain more non-target spectrogram components compared to a direct sonification.

10.3 Experiments

In this section, we discuss systematically conducted experiments, which illustrate the potential of the proposed method. To this end, we employ a database consisting of ten pieces from the Western classical music repertoire. The database consists of four Bach pieces (mainly inventions) and six Chopin pieces (mainly preludes and mazurkas). For each piece, the database contains an uninterpreted score-like MIDI file from the Mutopia Project[2] as well as a corresponding audio recording. Here, we use either high-quality audio recordings from the Saarland Music Database (SMD)[3] or digitized versions of historical recordings from the Piano Society project[4]. In total, the database contains 24 minutes of music with each recording having a length between 30 seconds and 5 minutes.

In a first step, we indicate the quality of the proposed method quantitatively using synthetic audio data. To this end, we use the Mutopia MIDI files to create two additional MIDI files for each piece using only the notes of the left and the right hand, respectively. Using a wave table synthesizer, we then generate audio recordings from these MIDI files which are used as ground truth separation results in the following. A linear mix of these two recordings serves as input for all evaluated separation approaches. For the experiment, we compute a magnitude spectrogram of the mix and derive a factorization with one of the methods discussed in Section 10.1. Then, we employ the masking-based separation procedure as described in Section 10.2 to obtain separated audio signals for the left and the right hand, respectively.

To assess the quality of a separation result, we employ version 3.0 of the BSSEVAL toolkit [184] to compute *signal-to-distortion (SDR)* values. Figure 10.8 shows SDR values for the initialization strategies I_{H}, I_{W}, I_{WH} and $I_{\mathrm{WH}}^{\mathrm{O}}$ separately for the left and the right hand as well as an average for both hands. All values are averaged over the ten pieces in our database. Note that the SDR values for the right hand are consistently higher than those for the left hand. Here, the main reason is that the right hand often contains the main melody and is therefore played louder (level difference is 1.64 dB on average). As a consequence, there is more energy related to this hand in the mixture making the separation easier. Furthermore, we see in Figure 10.8 that the strategies I_{H} and I_{W},

[2] `http://www.mutopiaproject.org`

[3] `http://www.mpi-inf.mpg.de/resources/SMD/`

[4] `http://pianosociety.com`

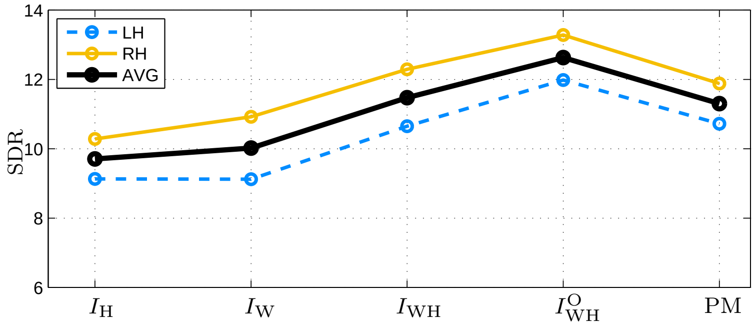


Figure 10.8. Evaluation results given in SDR values for the left (LH) and the right hand (RH) as well as an average of both hands (AVG).

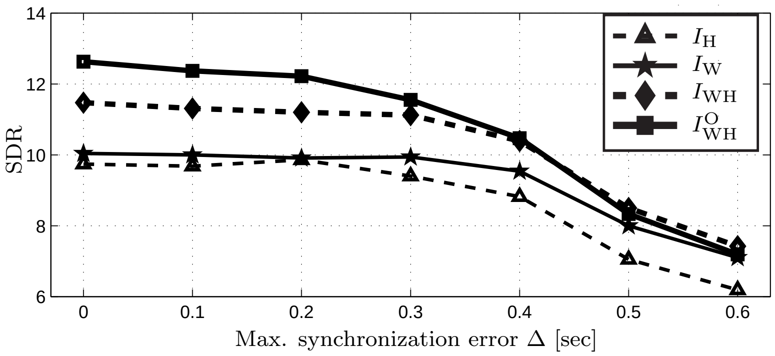


Figure 10.9. Evaluation results given in SDR values for several separation approaches varying the synchronization error.

which initialize only one of the matrices, yield the lowest SDR values. Combining the two strategies (I_{WH}) we see a significant SDR-gain of almost 1.5 dB. Finally, integrating onset information leads to another substantial gain of 1.2 dB for the strategy $I^{\mathrm{O}}_{\mathrm{WH}}$. Here, the dedicated representation of the percussive sounds leads to a more coherent representation of the harmonic parts and consequently to a better separation quality.

To additionally indicate how a typical parametric model (PM) behaves in our scenario, we also include SDR values for a the approach presented in Chapter 9, here using the spectral envelope model φ^2, which models the overtone energy distribution for each pitch separately. Similar to I_{WH}, this approach only models the harmonic part of a recording, i. e. no onset model is included, and, indeed, the average SDR values for both approaches are almost identical (11.3 dB and 11.47 dB SDR, respectively). However, the NMF factorization, using only simple matrix operations, can be computed more efficiently than the parameter estimation required for PM and additionally is easier to implement. For example, to process the whole database consisting of 24 minutes of music, the Matlab implementation used in these experiments takes about 6 minutes on an Intel W3530 for the synchronization (5.5 minutes for the chroma feature extraction and 20 seconds for the high-resolution synchronization), and another 6 minutes for the factorizations using 100 NMF iterations. Using more optimized implementations [189], both values could be reduced even further. This is in contrast to the parametric approach, which roughly runs in real-time (excluding the synchronization step). Other previously proposed approaches even require several hours to process this amount of data. Furthermore, the straightforward integration of an onset model allows for a significant SDR-gain for the $I_{\mathrm{WH}}^{\mathrm{O}}$ strategy over PM.

Since the MIDI files were perfectly aligned to their sonifications in the first experiment, we now investigate how the synchronization accuracy affects the separation performance. To this end, we simulate a low accuracy alignment by shifting each note event randomly by $\pm\Delta$ seconds. Figure 10.9 gives the averaged SDR values for the four initialization strategies and varying values for Δ. Here, we see that all approaches are relatively stable as long as the synchronization error is not larger than $\mathrm{tol}_{\mathrm{on}}$ (200 ms in our experiments). Beyond $\mathrm{tol}_{\mathrm{on}}$, all SDR values drop significantly. However, it should be noted that even with very inaccurate onset information the strategy $I_{\mathrm{WH}}^{\mathrm{O}}$ stays on a similar level as I_{WH} demonstrating its overall robustness.

While signal-to-distortion ratios and similar evaluation measures can be used to illustrate some general tendencies, they often do not capture the perceptual separation quality. Therefore, audible separation results are additionally provided on a website[5]. Here, real, non-synthetic audio recordings from the SMD and Piano Society databases are used to give a realistic and perceptually meaningful impression of the quality of the proposed method in real world scenarios.

10.4 Conclusions

In this chapter, we have introduced an extended NMF variant that exploits available score information to guide the factorization. Based on the idea of simultaneously constraining both the template vectors as well as the activations, the method yields similar results as a the parametric approach presented in Chapter 9. These results are further improved by integrating template vectors dedicated to representing onsets. In the future, one could further extend the idea of double constraining to integrate further model assumptions into the NMF framework. Furthermore, it would be interesting to apply the proposed framework to other types of music.

[5] http://www.mpi-inf.mpg.de/resources/MIR/ICASSP2012-ScoreInformedNMF

Chapter 11

Conclusions and Outlook

In recent years many techniques have been developed in the field of music information retrieval (MIR) that allow users to discover and experience music in large music collections in an intuitive and efficient manner. In this thesis, we have focused on three central MIR tasks: music synchronization, audio matching, and source separation. Although these tasks seem to have little in common at first sight, we have seen that strategies, methods, and concepts developed for one task also played a central role for the other tasks.

In the first part of the thesis, we focused on the design of chroma-based audio features, which were important throughout the entire thesis. In retrieval applications such as audio matching, these features are particularly powerful as they offer a high degree of invariance to changes in timbre. As one of the main results of the first part of this thesis, we showed how this robustness can be increased even further. More precisely, inspired by techniques used in the context of MFCC features, we proposed a novel method that identifies and discards timbre-related information as part of the feature computation. We demonstrated the effectiveness of this method using audio matching as an example application. Furthermore, initial results presented in [6, 15, 81, 101] suggest that the proposed CRP features might also be useful for other applications. However, one should not expect that simply replacing chroma features by CRP features will lead to measurable performance gains in all applications. In particular, CRP features were designed for Western tonal music that is characterized by a prominent harmonic progression. For pieces that are dominated by percussive sounds, CRP features might fail to capture the harmonic elements. Therefore, in the future, it might be interesting to incorporate methods for the separation of harmonic and percussive sounds as a preprocessing step into the feature computation [137]. Furthermore, while the investigations in Section 3.4 have been conducted to analyze the principles underlying the boost towards timbre invariance, they might also indicate another potential research direction. Here, we have seen that CRP features are dominated by only a few DCT coefficients. This might be interesting in the context of large scale indexing applications, which benefit strongly from low-dimensional feature representations, both in terms of efficiency and retrieval quality. Moreover, it would be interesting in the future to further investigate the potential of PFCC features in music analysis tasks. Using a pitch- instead of a mel-scale, PFCC features might be better suited to capture musical timbre than classical speech-oriented MFCC features. Initial results presented in [76] indeed indicate a performance gain resulting from the use of PFCC features in a genre recognition and instrument classification task.

In the second part of the thesis, we then addressed the task of music synchronization. Here, the chroma features discussed in the first part were of central importance. Similar to the design of features, we had to account for various musical variabilities in the synchronization process. To cope with structural differences between the versions to be aligned, we proposed two general strategies. On the one hand, we introduced a late-fusion approach which employs several conceptually different alignment strategies to identify reliable alignment parts. On the other hand, we presented a procedure which combines structure analysis methods with music synchronization techniques to identify corresponding passages within and across different versions of the same musical work. As we have seen, both approaches achieve meaningful alignment results even in the presence of significant structural differences. Compared to classical global synchronization, however, structure-aware partial synchronization constitutes a much more challenging research problem. Global music synchronization methods only have to find the best in a set of possible alignments. Partial synchronization methods additionally have to decide which of these alignments are actually valid and meaningful. As we have seen, this leads eventually to the decision whether two given passages should be called similar or not. Finding an answer to this question solely based on audio features can be a rather ill-defined problem in some cases. So overall, music synchronization in the presence of structural differences is still far from being solved. Further challenges arise in particular from differences in polyphony, for example when aligning a melody to a version that additionally contains an accompaniment track. A basic idea to deal with such differences could be to employ classical source separation techniques to extract some musical sound sources in a first step and to align them individually with the score in a second step. However, in this way, the valuable additional information provided by the score would be useless during the separation process. As an alternative, it might be more beneficial to investigate novel extensions to score-informed source separation. In particular, given a score and an audio recording for a piece of music, one could relax the assumption that the score specifies a note event if and only if there is a corresponding note event occurring in the audio recording. Overall, this might lead to interesting questions both in the context of music synchronization and score-informed source separation.

After addressing the task of partial music synchronization, we focused on classical music synchronization where the versions to be aligned are assumed to globally correspond to each other. Here, we have seen that the alignment accuracy of classical chroma-based approaches often does not suffice to capture fine nuances in tempo. However, such level of detail is of central importance for tasks such as performance analysis or score-informed source separation. To further refine the synchronization accuracy, our strategy was to combine the established chroma features with novel, temporally precise features that indicate onset positions for each chroma separately. As we have seen, the proposed system inherits the robustness of chroma-based approaches and yields high-precision alignments whenever clear onset information is available. However, for recordings in which onsets are less pronounced (string quartets, symphonic pieces), the proposed method was limited to the accuracy of classical chroma-based approaches. To further stabilize the synchronization process in such cases, it would be interesting to investigate new ways to consider the temporal context during the alignment. In particular, beat tracking and tempo estimation techniques as well as features describing the local rhythmic information [94] might be useful to identify potential onset positions.

In the third part, we then employed the synchronization methods developed in the second part to facilitate a very challenging MIR task: musical source separation. Here, we took advantage of the fact that large music collections often contain multiple versions or representations of a given piece of music. Using synchronization techniques allowed us to align the various sources of information and to utilize them as a guidance during the separation process. In particular, we have pursued two approaches. On the one hand, we presented a parametric spectrogram model, where the underlying idea was to iteratively estimate musical or acoustical parameters such that the parametric model accurately explains a given spectrogram. As a second strategy, we introduced a score-informed variant of non-negative matrix factorization (NMF) with the idea to introduce constraints for both the NMF templates and their corresponding activities. This even allowed us to incorporate template vectors representing percussive elements such as onsets. For piano music, both approaches led to plausible separation results in most cases. Furthermore, we have seen that the computationally efficient NMF model yielded a similar separation quality as the more complex parametric model. Additionally, the NMF model allowed for a straightforward integration of onset information, which gave an additional gain in separation quality. However, we have not yet answered the question whether these findings also hold for other types of music. For example, for very complex pieces with a large number of overlapping sound sources, the NMF-based model might not be as robust as the parametric approach. The latter offers a stricter enforcement of structure in its signal model. Additionally, it might also be easier to integrate musical knowledge into a parametric approach or to capture instrument-specific characteristics such as a vibrato or a specific amplitude progression. To further enhance the separation quality and robustness of the NMF model for other types of music, a promising research direction could be to transfer more concepts and ideas known from parametric models to the NMF model using ideas similar to the double constraints as presented in Chapter 10.

Furthermore, it would also be interesting to investigate whether the proposed score-informed source separation methods could be applied in real-time scenarios. To this end, one could try to combine the online synchronization approaches discussed in Section 7.6 with the methods presented in Part III. To the best of the author's knowledge, this scenario has only been considered so far in [35]. However, the authors employ a fixed model for the overtone energy distribution, which might be too inflexible for general music recordings. Therefore, it would be interesting to see how adaptive overtone models as presented in this thesis perform in such a scenario.

In real-time scenarios, efficiency and computational complexity are particularly important aspects. While the parameter estimation process as presented in Chapter 9 employs elaborate numerical optimization methods, it might not be efficient enough for real-time usage. An interesting research direction therefore could be to take the mathematical properties of the parametric model more into account to design a high performance parameter estimation process.

Based on the techniques developed in Part III, we further presented novel applications that demonstrated the potential of score-informed source separation. First, we showed how a previously proposed instrument equalizer can be extended to a general voice equalizer which allows for highlighting (or attenuating) arbitrary note groups in a given audio recording. Here, we have seen that the availability of score information not only stabilizes the separation process but also allows for a natural and user-friendly way of specifying the voices or note groups to be separated. Furthermore, we demonstrated how score-informed source separation methods can be employed for analysis purposes. In particular, given a polyphonic piano recording, we estimated intensity values capturing the loudness of individual note events as they occur in the recording. Our evaluation results showed a significant improvement for the proposed method over a given baseline. For the future, it will be interesting to explore many more potential applications of score-informed source separation. For example, the proposed voice equalizer could be further extended such that the interface not only allows for separating note events but additionally to edit them in many different ways. Here, one could integrate modules for shifting the pitch, changing the loudness, modifying the timbre, altering the note length, or inserting and deleting note events. Exploiting the score information, all of this could be incorporated into an intuitive interface hiding most technical details from the user.

Furthermore, score-informed source separation could also be employed in novel applications in the context of performance analysis. For example, instead of analyzing dynamics in piano recordings as presented in this thesis, one could also analyze the vibrato intensity in violin recordings or the use of picking and strumming techniques in guitar recordings. In this context, score-informed source separation techniques constitute a valuable supporting tool for capturing and analyzing various performance-specific subtleties beyond the score.

Part IV

Appendix

Appendix A

Chroma Toolbox

The feature extraction components described in Chapter 2 and Chapter 3 have been implemented in a *chroma toolbox*, which is freely available at the well-documented website [17] under a GNU-GPL license. Table A.1 gives an overview of the main MATLAB functions along with the most important parameters. Note that there are many more parameters not discussed in this thesis. However, for all parameters there are default settings so that none of the parameters need to be specified by the user.

To demonstrate how the toolbox can be applied, we now discuss the code example[1] shown in Table A.2. Our example starts with a call to the function `wav_to_audio`, which is a simple wrapper around MATLAB's `wavread.m` and converts the input WAV file into a mono version at a sampling rate of 22050 Hz. Furthermore, the struct `sideinfo` is returned containing meta information about the WAV file. In line 3, the audio data is processed by `estimateTuning`, which computes an appropriate filter bank shift σ for the recording. Next, in lines 5–9, Pitch features are computed. Here, the struct `paramPitch` is used to pass optional parameters to the feature extraction function. If some parameters or the whole struct are not set manually, then meaningful default settings are used. This is a general principle throughout the toolbox. For the pitch computation, `winLenSTMSP` specifies the window length in samples. Here, 4410 together with a sampling frequency of 22050 Hz results in a window length corresponding to 200ms of audio. Using half-overlapped windows leads to a feature rate of 10 Hz. The filterbank shift is specified in line 6 using the output of `estimateTuning`. Furthermore, an internal visualization is activated using the parameter `visualize`. Then, a call to `audio_to_pitch_via_FB` results in a $120 \times N$-matrix `f_pitch` that constitutes the Pitch features, where N is the number of time frames and the first dimension corresponds to MIDI pitches. Actually, only the bands corresponding to MIDI pitches 21 to 108 are computed and the values of the other bands are set to zero. Furthermore, details on the feature configuration are appended to the `sideinfo` struct. Using `sideinfo` to store all relevant meta information related to the feature processing pipeline constitutes a second general principle in our toolbox.

In lines 11–31, various chroma representations are derived from the pitch features. First, in lines 11–14, CP features are computed. Then, activating the logarithmic compression

[1] This example is also contained in the toolbox as function `demoChromaToolbox.m`.

Filename	Main parameters	Description
`wav_to_audio.m`	–	Import of WAV files and conversion to expected audio format.
`estimateTuning.m`	pitchRange	Estimation of the filterbank shift parameter σ.
`audio_to_pitch_via_FB.m`	winLenSTMSP	Extraction of pitch features from audio data.
`pitch_to_chroma.m`	applyLogCompr, factorLogCompr $\widehat{=}\ \eta$	Derivation of CP and CLP features from Pitch features.
`pitch_to_CENS.m`	winLenSmooth $\widehat{=}\ w$, downsampSmooth $\widehat{=}\ d$	Derivation of CENS features from Pitch features.
`pitch_to_CRP.m`	coeffsToKeep $\widehat{=}\ n$, factorLogCompr $\widehat{=}\ \eta$	Derivation of CRP features from Pitch features.
`smoothDownsampleFeature.m`	winLenSmooth $\widehat{=}\ w$, downsampSmooth $\widehat{=}\ d$	Post-processing of features: smoothing and downsampling.
`normalizeFeature.m`	p	Post-processing of features: ℓ^p-normalization (default: $p = 2$).
`visualizePitch.m`	featureRate	Visualization of pitch features.
`visualizeChroma.m`	featureRate	Visualization of chroma features.
`visualizeCRP.m`	featureRate	Specialized version of visualizeChroma for CRP features.
`generateMultiratePitchFilterbank.m`	–	Generation of filterbanks (used in `audio_to_pitch_via_FB.m`).

Table A.1. Overview of the MATLAB functions contained in the chroma toolbox [17].

using `applyLogCompr`, CLP[100] features are computed in lines 16–20. The compression level is specified in line 17 by the parameter `factorLogCompr`, which corresponds to the parameter η introduced in Section 2.5. Next, in lines 22–26, CENS_5^{21} features are computed. Here, the parameters `winLenSmooth` and `downsampSmooth` correspond to the parameters w and d explained in Section 2.7, respectively. Finally, in lines 28–31, CRP(55) features are computed, where the parameter n of Section 3.1.2 corresponds to the lower bound of the range specified by `coeffsToKeep`, see line 28. Finally, the use of the function `smoothDownsampleFeature` is demonstrated, where in lines 33–34 the parameters w and d are specified as for the CENS computation. At the end of our example, we visualize the smoothed CRP features using the function `visualizeCRP`.

```
filename='Systematic_Chord-C-Major_Eight-Instruments.wav';
[f_audio,sideinfo]=wav_to_audio('','data_WAV/',filename);
shiftFB=estimateTuning(f_audio);

paramPitch.winLenSTMSP=4410;
paramPitch.shiftFB=shiftFB;
paramPitch.visualize=1;
[f_pitch,sideinfo]=...
    audio_to_pitch_via_FB(f_audio,paramPitch,sideinfo);

paramCP.applyLogCompr=0;
paramCP.visualize=1;
paramCP.inputFeatureRate=sideinfo.pitch.featureRate;
[f_CP,sideinfo]=pitch_to_chroma(f_pitch,paramCP,sideinfo);

paramCLP.applyLogCompr=1;
paramCLP.factorLogCompr=100;
paramCLP.visualize=1;
paramCLP.inputFeatureRate=sideinfo.pitch.featureRate;
[f_CLP,sideinfo]=pitch_to_chroma(f_pitch,paramCLP,sideinfo);

paramCENS.winLenSmooth=21;
paramCENS.downsampSmooth=5;
paramCENS.visualize=1;
paramCENS.inputFeatureRate=sideinfo.pitch.featureRate;
[f_CENS,sideinfo]=pitch_to_CENS(f_pitch,paramCENS,sideinfo);

paramCRP.coeffsToKeep=[55:120];
paramCRP.visualize=1;
paramCRP.inputFeatureRate=sideinfo.pitch.featureRate;
[f_CRP,sideinfo]=pitch_to_CRP(f_pitch,paramCRP,sideinfo);

paramSmooth.winLenSmooth=21;
paramSmooth.downsampSmooth=5;
paramSmooth.inputFeatureRate=sideinfo.CRP.featureRate;
[f_CRPSmoothed,featureRateSmoothed]=...
    smoothDownsampleFeature(f_CRP,paramSmooth);
parameterVis.featureRate=featureRateSmoothed;
visualizeCRP(f_CRPSmoothed,parameterVis);
```

Table A.2. Code example.

Appendix B

Technical Details CRP Evaluation

Chroma-IF
Chroma-P
Chroma-E
	`chromagram_IF(f_audio,sr,fftlen,nbin,f_ctr,f_sd)`
	`chromagram_P(f_audio,sr,fftlen,nbin,f_ctr,f_sd)`
	`chromagram_E(f_audio,sr,fftlen,nbin,f_ctr,f_sd)`
	`sr` = 22050
	`fftlen` = 16384
	`nbin` = 12
	`f_ctr` = $2^{((60-69)/12)} \cdot 440 = 262$
	`f_sd` = 1.5
	Version: 2007-04-21 14:03:14 (last site update)
	
	As the hopsize was fixed at fftlen/4 we threw away every second feature vector from the output resulting in a feature rate of 2.7 Hz (2.7 features per second).

Chroma-Mir	`mirchromagram(filename,'Frame',1,0.5)` Version: 1.1
Chroma-QM	`sonic-annotator -t chroma.n3 -w csv filename` `step_size` = 11025 `block_size` = 16384 `sample_rate` = 22050 `bpo` = 12 `minpitch` = 21 `maxpitch` = 108 `normalization` = 1 `tuning` = 440 Version: 1.6

Table B.1. Configuration of the reference chroma implementations used in Chapter 3: Each feature extractor was configured to produce similar output as the baseline implementation (Chroma-Pitch), which produces 12 dimensional chroma vectors with a feature rate of 2 Hz and using the pitch range between MIDI Pitch 21 to 108. If the configurability of a feature extractor was limited we tried to approximate these settings as closely as possible. After feature extraction all features were normalized using the Euclidean norm.

Composer/ Artist	**Title / Opus**	**Query**	**Rec.**	**Occ.**
Brahms	Hungarian Dances 05	Part A	2	6
Shostakovich	JazzSuite2 6 Waltz2	Main Theme	2	8
Ravel	Bolero	Main Theme	2	18
Bach	BWV565 toccataOrgan	Beginning	3	3
Schumann	Traeumerei	Main Theme	3	9
Wagner	Meistersinger Prelude	Beginning	2	2
Beethoven	Op. 67 (5th Symphony)	Secondary theme	5	15
Coldplay	In my place	Stanza + Stanza + Chorus	2	4
Indigo Girls	Free in you	Stanza + Chorus	2	6
Genesis	That's all	Chorus	3	9
Queen	We are the champions	Stanza	3	6
Beatles	Yesterday	Stanza + Chorus	2	4
Gloria Gaynor	I will survive	Main theme	1	11
Total			**32**	**101**

Table B.2. Database used in the experimental section of Chapter 3. As the number of query occurrences varied between the pieces we averaged the results per piece in a first step and then over the whole database in a second step. This way, pieces with many occurrences do not dominate the results presented in Chapter 3.

Bibliography

[1] Vlora Arifi, Michael Clausen, Frank Kurth, and Meinard Müller. Score-to-PCM music synchronization based on extracted score parameters. In *Proceedings of 2nd International Symposium on Computer Music Modeling and Retrieval (CMMR)*, pages 193–210, Esbjerg, Denmark, 2004.

[2] Vlora Arifi, Michael Clausen, Frank Kurth, and Meinard Müller. Synchronization of music data in score-, MIDI- and PCM-format. *Computing in Musicology*, 13:9–33, 2004.

[3] Jean-Julien Aucouturier and Francois Pachet. Improving timbre similarity: How high's the sky. *Journal of Negative Results in Speech and Audio Sciences*, 1, 2004.

[4] Roland Badeau, Nancy Bertin, and Emmanuel Vincent. Stability analysis of multiplicative update algorithms for non-negative matrix factorization. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2148–2151, Prague, Czech Republic, 2011.

[5] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(1):96–104, 2005.

[6] Juan Pablo Bello. Measuring structural similarity in music. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2013–2025, 2011.

[7] Juan Pablo Bello and Jeremy Pickens. A robust mid-level representation for harmonic content in music signals. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 304–311, London, UK, 2005.

[8] Nancy Bertin, Roland Badeau, and Emmanuel Vincent. Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):538–549, 2010.

[9] Albert S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. MIT Press, 1990.

[10] Judith C. Brown. Calculation of a constant Q spectral transform. *Journal of the Acoustical Society of America*, 89(1):425–434, 1991.

[11] Richard H. Byrd, M. E. Hribar, and Jorge Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 1999.

[12] Chris Cannam, Christian Landone, Mark Sandler, and Juan Pablo Bello. The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 324–327, Victoria, Canada, 2006.

[13] Wei Chai. *Automated Analysis of Musical Structure*. PhD thesis, MIT MediaLab, 2005.

[14] Wei Chai. Semantic segmentation and summarization of music: methods based on tonality and recurrent structure. *IEEE Signal Processing Magazine*, 23(2):124–132, 2006.

[15] Taemin Cho and Juan P. Bello. A feature smoothing method for chord recognition using recurrence plots. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 651–656, Miami, USA, 2011.

[16] Taemin Cho, Ron J. Weiss, and Juan Pablo Bello. Exploring common variations in state of the art chord recognition systems. In *Proceedings of the Sound and Music Computing Conference (SMC)*, pages 1–8, Barcelona, Spain, 2010.

[17] Chroma Toolbox. `http://www.mpi-inf.mpg.de/~mmueller/chromatoolbox/`, Retrieved 14.12.2010.

[18] Arshia Cont. Realtime audio to score alignment for polyphonic music instruments using sparse non-negative constraints and hierarchical HMMS. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 5, Toulouse, France, 2006.

[19] Arshia Cont. ANTESCOFO: Anticipatory synchronization and control of interactive parameters in computer music. In *Proceedings of the International Computer Music Conference (ICMC)*, North Irland, Belfast, 2008.

[20] Arshia Cont. A coupled duration-focused architecture for real-time music-to-score alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):974–987, 2010.

[21] Arshia Cont, Shlomo Dubnov, and David Wessel. Realtime multiple-pitch and multiple-instrument recognition for music signals using sparse non-negativity constraints. In *10th International Conference on Digital Audio Effects*, pages 85–92, Bordeaux, France, 2007.

[22] David Damm, Christian Fremerey, Frank Kurth, Meinard Müller, and Michael Clausen. Multimodal presentation and browsing of music. In *Proceedings of the 10th International Conference on Multimodal Interfaces (ICMI)*, pages 205–208, Chania, Crete, Greece, 2008.

[23] David Damm, Christian Fremerey, Verena Thomas, Michael Clausen, Frank Kurth, and Meinard Müller. A digital library framework for heterogeneous music collections—from document acquisition to cross-modal interaction. *International Journal on Digital Libraries: Special Issue on Music Digital Libraries*, 2011, to appear.

[24] David Damm, Harald Grohganz, Frank Kurth, Sebastian Ewert, and Michael Clausen. SyncTS: Automatic synchronization of speech and text documents. In *Proceedings of the AES International Conference Semantic Audio*, pages 98–107, Ilmenau, Germany, 2011.

[25] Roger B. Dannenberg. An on-line algorithm for real-time accompaniment. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 193–198, 1984.

[26] Roger B. Dannenberg and Masataka Goto. Music structure analysis from acoustic signals. In David Havelock, Sonoko Kuwano, and Michael Vorländer, editors, *Handbook of Signal Processing in Acoustics*, volume 1, pages 305–331. Springer, New York, NY, USA, 2008.

[27] Roger B. Dannenberg and Ning Hu. Pattern discovery techniques for music audio. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 63–70, Paris, France, 2002.

[28] Roger B. Dannenberg and Ning Hu. Polyphonic audio matching for score following and intelligent audio editors. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 27–34, San Francisco, USA, 2003.

[29] Roger B. Dannenberg and Christopher Raphael. Music score alignment and computer accompaniment. *Communications of the ACM, Special Issue: Music information retrieval*, 49(8):38–43, 2006.

[30] Steven B. Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980.

[31] Simon Dixon and Gerhard Widmer. MATCH: A music alignment tool chest. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, London, GB, 2005.

[32] Arnaud Doucet and Adam M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. In Dan Crisan and Boris Rozovskii, editors, *The Oxford Handbook of Nonlinear Filtering*, pages 656–705. Oxford University Press, 2011.

[33] Karin Dressler. An auditory streaming approach for melody extraction from polyphonic music. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 19–24, Miami, USA, 2011.

[34] Zhiyao Duan and Bryan Pardo. Aligning semi-improvised music audio with its lead sheet. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 513–518, Miami, FL, USA, 2011.

[35] Zhiyao Duan and Bryan Pardo. Soundprism: An online system for score-informed source separation of music audio. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1205–1215, 2011.

[36] Zhiyao Duan and Bryan Pardo. A state space model for online polyphonic audio-score alignment. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 197–200, Prague, Czech Republic, 2011.

[37] Jean-Louis Durrieu, Gaël Richard, Bertrand David, and Cédric Févotte. Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE Transactions on Audio, Speech and Language Processing*, 18(3):564–575, 2010.

[38] Daniel P.W. Ellis and Graham E. Poliner. Identifying 'cover songs' with chroma features and dynamic programming beat tracking. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 4, Honolulu, Hawaii, USA, 2007.

[39] Antti J. Eronen and Anssi P. Klapuri. Musical instrument recognition using cepstral coefficients and temporal features. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 2, pages II753–II756, Istanbul, Turkey, 2000.

[40] Sebastian Ewert and Meinard Müller. Refinement strategies for music synchronization. In *Proceedings of the International Symposium on Computer Music Modeling and Retrieval (CMMR)*, volume 5493 of *Lecture Notes in Computer Science*, pages 147–165, Copenhagen, Denmark, 2008.

[41] Sebastian Ewert and Meinard Müller. Estimating note intensities in music recordings. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 385–388, Prague, Czech Republic, 2011.

[42] Sebastian Ewert and Meinard Müller. Score-informed voice separation for piano recordings. In *Proceedings of the International Society for Music Information Retrieval Conference(ISMIR)*, pages 245–250, Miami, USA, 2011.

[43] Sebastian Ewert and Meinard Müller. Score informed source separation. In Meinard Müller, Masataka Goto, and Markus Schedl, editors, *Multimodal Music Processing (to appear)*, Dagstuhl Follow-Ups. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2012.

[44] Sebastian Ewert and Meinard Müller. Using score-informed constraints for NMF-based source separation. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Kyoto, Japan, 2012.

[45] Sebastian Ewert, Meinard Müller, and Michael Clausen. Score-informed audio parametrization. International Society for Music Information Retrieval (ISMIR) latebreaking session, 2010. Available online: `http://www.ismir.net`.

[46] Sebastian Ewert, Meinard Müller, and Roger B. Dannenberg. Towards reliable partial music alignments using multiple synchronization strategies. In *Proceedings of the International Workshop on Adaptive Multimedia Retrieval (AMR), Lecture Notes in Computer Science (LNCS)*, volume 6535, pages 35–48, Madrid, Spain, 2009.

[47] Sebastian Ewert, Meinard Müller, and Peter Grosche. High resolution audio synchronization using chroma onset features. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009.

[48] Sebastian Ewert, Meinard Müller, Verena Konz, Daniel Müllensiefen, and Geraint Wiggins. Towards cross-domain harmonic analysis of music. *IEEE Transactions on Multimedia (to appear)*, 2012.

[49] Sebastian Ewert, Meinard Müller, Daniel Müllensiefen, Michael Clausen, and Geraint A. Wiggins. Case study "Beatles Songs" – What can be learned from unreliable music alignments? In Eleanor Selfridge-Field, Frans Wiering, and Geraint A. Wiggins, editors, *Knowledge representation for intelligent music processing*, number 09051 in Dagstuhl Seminar Proceedings. Schloss Dagstuhl - Leibniz Center for Informatics, Germany, 2009.

[50] Hugo Fastl and Eberhard Zwicker. *Psychoacoustics, Facts and Models*. Springer, 2007 (3rd Edition).

[51] Jonathan Foote. Visualizing music and audio using self-similarity. In *Proceedings of the ACM International Conference on Multimedia*, pages 77–80, Orlando, FL, USA, 1999.

[52] Christian Fremerey, Michael Clausen, Sebastian Ewert, and Meinard Müller. Sheet music-audio identification. In *Proceedings of the International Society for Music Information Retrieval Conference(ISMIR)*, pages 645–650, Kobe, Japan, 2009.

[53] Christian Fremerey, Meinard Müller, and Michael Clausen. Handling repeats and jumps in score-performance synchronization. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 243–248, Utrecht, The Netherlands, 2010.

[54] Christian Fremerey, Meinard Müller, Frank Kurth, and Michael Clausen. Automatic mapping of scanned sheet music to audio recordings. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pages 413–418, Philadelphia, USA, 2008.

[55] Hiromasa Fujihara, Masataka Goto, Jun Ogata, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Automatic synchronization between lyrics and music cd recordings based on viterbi alignment of segregated vocal signals. In *IEEE International Symposium on Multimedia (ISM)*, pages 257–264, Los Alamitos, CA, USA, 2006.

[56] Hiromasa Fujihara, Masataka Goto, Jun Ogata, and Hiroshi G. Okuno. LyricSynchronizer: Automatic synchronization system between musical audio signals and lyrics. *IEEE Journal of Selected Topics in Signal Processing*, 5(6):1252–1261, 2011.

[57] Takuya Fujishima. Realtime chord recognition of musical sound: A system using common lisp music. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 464–467, Beijing, 1999.

[58] Joachim Ganseman, Paul Scheunders, Gautham J. Mysore, and Jonathan S. Abel. Evaluation of a score-informed source separation system. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 219–224, Utrecht, The Netherlands, 2010.

[59] Joachim Ganseman, Paul Scheunders, Gautham J. Mysore, and Jonathan S. Abel. Source separation by score synthesis. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 462–465, New York, USA, 2010.

[60] Emilia Gómez. *Tonal Description of Music Audio Signals*. PhD thesis, UPF Barcelona, 2006.

[61] Masataka Goto. A chorus-section detecting method for musical audio signals. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 437–440, Hong Kong, China, 2003.

[62] Masataka Goto. SmartMusicKIOSK: Music listening station with chorus-search function. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST)*, pages 31–40, 2003.

[63] Masataka Goto. Development of the rwc music database. In *Proceedings of the International Congress on Acoustics (ICA)*, pages 553–556, 2004.

[64] Masataka Goto. A real-time music-scene-description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication (ISCA Journal)*, 43(4):311–329, 2004.

[65] Masataka Goto. A chorus section detection method for musical audio signals and its application to a music listening station. *IEEE Transactions on Audio, Speech and Language Processing*, 14(5):1783–1794, 2006.

[66] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. RWC music database: Popular, classical and jazz music databases. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2002.

[67] Daniel W. Griffin and Jae S. Lim. Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 32(2):236–243, 1984.

[68] Yushen Han and Christopher Raphael. Desoloing monaural audio using mixture models. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 145–148, Vienna, Austria, 2007.

[69] Yushen Han and Christopher Raphael. Informed source separation of orchestra and soloist. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 315–320, Utrecht, The Netherlands, 2010.

[70] Christopher Harte and Mark Sandler. Automatic chord identification using a quantised chromagram. In *Proceedings of the 118th AES Convention*, Barcelona, Spain, 2005.

[71] Christopher Harte, Mark Sandler, Samer Abdallah, and Emilia Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 66–71, London, GB, 2005.

[72] Toni Heittola, Anssi P. Klapuri, and Tuomas Virtanen. Musical instrument recognition in polyphonic audio using source-filter model for sound separation. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 327–332, Kobe, Japan, 2009.

[73] Romain Hennequin, Roland Badeau, and Bertrand David. Time-dependent parametric and harmonic templates in non-negative matrix factorization. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, pages 246–253, Graz, Austria, 2010.

[74] Romain Hennequin, Bertrand David, and Roland Badeau. Score informed audio source separation using a parametric model of non-negative spectrogram. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 45–48, Prague, Czech Republic, 2011.

[75] Ning Hu, Roger B. Dannenberg, and George Tzanetakis. Polyphonic audio matching and alignment for music retrieval. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, NY, USA, 2003.

[76] Ying Hu and Guizhong Liu. Dynamic characteristics of musical note for musical instrument classification. In *Proceedings of the IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pages 1–6, Xi'an, Shaanxi, China, 2011.

[77] Katsutoshi Itoyama, Masataka Goto, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Integration and adaptation of harmonic and inharmonic models for separating polyphonic musical signals. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages I–57–I–60, Hawaii, USA, 2007.

[78] Katsutoshi Itoyama, Masataka Goto, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Instrument equalizer for query-by-example retrieval: Improving sound source separation based on integrated harmonic and inharmonic models. In *Proceedings of the International Conference for Music Information Retrieval (ISMIR)*, pages 133–138, Philadelphia, USA, 2008.

[79] ITU. *Pulse code modulation (PCM) of voice frequencies (ITU-T Recommendation G.711)*. International Telecommunications Union, 1993.

[80] Özgür Izmirli and Roger B. Dannenberg. Understanding features and distance functions for music sequence alignment. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 411–416, Utrecht, Netherlands, 2010.

[81] Nanzhu Jiang, Peter Grosche, Verena Konz, and Meinard Müller. Analyzing chroma feature types for automated chord recognition. In *Proceedings of the Audio Engineering Society Conference (AES)*, Ilmenau, Germany, 2011.

[82] Cyril Joder, Slim Essid, and Gaël Richard. A comparative study of tonal acoustic features for a symbolic level music-to-score alignment. In *Proceedings of the 35nd IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Dallas, USA, 2010.

[83] Cyril Joder, Slim Essid, and Gaël Richard. An improved hierarchical approach for music-to-symbolic score alignment. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 39–45, Utrecht, Netherlands, 2010.

[84] Cyril Joder, Slim Essid, and Gaël Richard. A conditional random field framework for robust and scalable audio-to-score matching. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(8):2385–2397, 2011.

[85] Cyril Joder, Slim Essid, and Gaël Richard. Hidden discrete tempo model: A tempo-aware timing model for audio-to-score alignment. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 397–400, Prague, Czech Republic, 2011.

[86] Cyril Joder, Slim Essid, and Gaël Richard. Optimizing the mapping from a symbolic to an audio representation for music-to-score alignment. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 121–124, New Paltz, NY, USA, 2011.

[87] Cyril Joder, Felix Weninger, Florian Eyben, David Virette, and Björn Schuller. Real-time speech separation by semi-supervised nonnegative matrix factorization. In *Proceedings of the International Conference on Latent Variable Analysis and Signal Separation (LVA/ICA)*, Tel Aviv, Israel, 2012.

[88] Hirokazu Kameoka, Takuya Nishimoto, and Shigeki Sagayama. Harmonic-temporal-structured clustering via deterministic annealing EM algorithm for audio feature extraction. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 115–122, London, GB, 2005.

[89] Min-Yen Kan, Ye Wang, Denny Iskandar, Tin Lay Nwe, and Arun Shenoy. LyricAlly: Automatic synchronization of textual lyrics to acoustic music signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):338–349, 2008.

[90] Anssi P. Klapuri. Multipitch analysis of polyphonic music and speech signals using an auditory model. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):255–266, 2008.

[91] Anssi P. Klapuri, Antti J. Eronen, and Jaakko Astola. Analysis of the meter of acoustic musical signals. *IEEE Transactions on Audio, Speech and Language Processing*, 14(1):342–355, 2006.

[92] Verena Konz, Meinard Müller, and Sebastian Ewert. A multi-perspective evaluation framework for chord recognition. In *Proceedings of the International Society for Music Information Retrieval Conference(ISMIR)*, pages 9–14, Utrecht, The Netherlands, 2010.

[93] Lucas Kovar and Michael Gleicher. Flexible automatic motion blending with registration curves. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation (ACM-SCA)*, pages 214–224, Aire-la-Ville, Switzerland, 2003.

[94] Frank Kurth, Thorsten Gehrmann, and Meinard Müller. The cyclic beat spectrum: Tempo-related audio features for time-scale invariant audio identification. In *Proceedings of the 7th International Conference on Music Information Retrieval (ISMIR)*, pages 35–40, Victoria, Canada, 2006.

[95] Frank Kurth and Meinard Müller. Efficient index-based audio matching. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):382–395, 2008.

[96] Frank Kurth, Meinard Müller, Christian Fremerey, Yoon ha Chang, and Michael Clausen. Automated synchronization of scanned sheet music with audio recordings. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 261–266, Vienna, Austria, 2007.

[97] Olivier Lartillot and Petri Toiviainen. MIR in Matlab (II): A toolbox for musical feature extraction from audio. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 127–130, Vienna, Austria, 2007.

[98] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *Proceedings of the Neural Information Processing Systems (NIPS)*, pages 556–562, Denver, CO, USA, 2000.

[99] Kyogu Lee and Malcolm Slaney. Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):291–301, 2008.

[100] Fred Lerdahl and Ray Jackendoff. *A Generative Theory of Tonal Music.* MIT Press, 1983.

[101] Cynthia C.S. Liem and Alan Hanjalic. Expressive timing from cross-performance and audio-based alignment patterns: An extended case study. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 519–524, Miami, USA, 2011.

[102] Chih-Jen Lin. On the convergence of multiplicative update algorithms for nonnegative matrix factorization. *IEEE Transactions on Neural Networks*, 18:1589–1596, 2007.

[103] Beth Logan. Mel frequency cepstral coefficients for music modeling. In *Proceedings of the International Symposium on Music Information Retrieval (ISMIR)*, Plymouth, Massachusetts, 2000.

[104] Robert Macrae and Simon Dixon. Accurate real-time windowed time warping. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 423–428, Utrecht, Netherlands, 2010.

[105] Robert Macrae and Simon Dixon. A guitar tablature score follower. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 725–726, Singapore, 2010.

[106] Robert Macrae, Joachim Neumann, Xavier Anguera, Nuria Oliver, and Simon Dixon. Real-time synchronisation of multimedia streams in a mobile device. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, Barcelona, Spain, 2011.

[107] Namunu C. Maddage. Automatic structure detection for popular music. *IEEE Multimedia*, 13(1):65–77, 2006.

[108] Namunu C. Maddage, Changsheng Xu, Mohan S. Kankanhalli, and Xi Shao. Content-based music structure analysis with applications to music semantics understanding. In *Proceedings of the ACM International Conference on Multimedia*, pages 112–119, New York, NY, USA, 2004.

[109] Akira Maezawa, Hiroshi G. Okuno, Tetsuya Ogata, and Masataka Goto. Polyphonic audio-to-score alignment based on bayesian latent harmonic allocation hidden markov model. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 185–188, Prague, Czech Republic, 2011.

[110] Matthias Mauch and Simon Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1280–1289, 2010.

[111] Matthias Mauch, Hiromasa Fujihara, and Masataka Goto. Integrating additional chord information into HMM-based lyrics-to-audio alignment. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):200–210, 2012.

[112] John H. Maxwell. An expert system for harmonic analysis of tonal music. In *Understanding Music with AI*, pages 335–353. MIT Press, 1992.

[113] MIREX 2010. Audio Chord Estimation Subtask. `http://www.music-ir.org/mirex/wiki/2010:Audio_Chord_Estimation`, Retrieved 17.09.2010.

[114] Nicola Montecchio and Arshia Cont. A unified approach to real time audio-to-score and audio-to-audio alignment using sequential Montecarlo inference techniques. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 193–196, Prague, Czech Republic, 2011.

[115] Brian C. J. Moore, Brian R. Glasberg, and Thomas Baer. A model for the prediction of thresholds, loudness, and partial loudness. *Journal of the Audio Engineering Society*, 45(4):224–240, 1997.

[116] Meinard Müller. *Information Retrieval for Music and Motion.* Springer Verlag, 2007.

[117] Meinard Müller and Daniel Appelt. Path-constrained partial music synchronization. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 65–68, Las Vegas, Nevada, USA, 2008.

[118] Meinard Müller and Michael Clausen. Transposition-invariant self-similarity matrices. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, pages 47–50, Vienna, Austria, 2007.

[119] Meinard Müller, Michael Clausen, Verena Konz, Sebastian Ewert, and Christian Fremerey. A multimodal way of experiencing and exploring music. *Interdisciplinary Science Reviews (ISR)*, 35(2):138–153, 2010.

[120] Meinard Müller and Sebastian Ewert. Joint structure analysis with applications to music annotation and synchronization. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 389–394, Philadelphia, Pennsylvania, USA, 2008.

[121] Meinard Müller and Sebastian Ewert. Towards timbre-invariant audio features for harmony-based music. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):649–662, 2010.

[122] Meinard Müller and Sebastian Ewert. Chroma Toolbox: MATLAB implementations for extracting variants of chroma-based audio features. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 215–220, Miami, FL, USA, 2011.

[123] Meinard Müller, Sebastian Ewert, and Sebastian Kreuzer. Making chroma features more robust to timbre changes. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1869–1872, Taipei, Taiwan, 2009.

[124] Meinard Müller, Verena Konz, Wolfgang Bogler, and Vlora Arifi-Müller. Saarland music data (SMD). In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR): Late Breaking session*, 2011.

[125] Meinard Müller, Verena Konz, Andi Scharfstein, Sebastian Ewert, and Michael Clausen. Towards automated extraction of tempo parameters from expressive music recordings. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 69–74, Kobe, Japan, 2009.

[126] Meinard Müller and Frank Kurth. Enhancing similarity matrices for music audio analysis. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 437–440, Toulouse, France, 2006.

[127] Meinard Müller and Frank Kurth. Towards structural analysis of audio recordings in the presence of musical variations. *EURASIP Journal on Advances in Signal Processing*, 2007(1), 2007.

[128] Meinard Müller, Frank Kurth, and Michael Clausen. Audio matching via chroma-based statistical features. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 288–295, 2005.

[129] Meinard Müller, Frank Kurth, David Damm, Christian Fremerey, and Michael Clausen. Lyrics-based audio retrieval and multimodal navigation in music collections. In *Proceedings of the 11th European Conference on Digital Libraries (ECDL)*, pages 112–123, Budapest, Hungary, 2007.

[130] Meinard Müller, Frank Kurth, and Tido Röder. Towards an efficient algorithm for automatic score-to-audio synchronization. In *Proceedings of the 5th International Conference on Music Information Retrieval (ISMIR)*, pages 365–372, Barcelona, Spain, 2004.

[131] Meinard Müller, Henning Mattes, and Frank Kurth. An efficient multiscale approach to audio synchronization. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 192–197, Victoria, Canada, 2006.

[132] Bernhard Niedermayer. Improving accuracy of polyphonic music-to-score alignment. In *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, pages 585–590, Kobe, Japan, 2009.

[133] Bernhard Niedermayer. Towards audio to score alignment in the symbolic domain. In *Proceedings of the Sound and Music Computing Conference (SMC)*, pages 77–82, Porto, Portugal, 2009.

[134] Bernhard Niedermayer and Gerhard Widmer. A multi-pass algorithm for accurate audio-to-score alignment. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 417–422, Utrecht, The Netherlands, 2010.

[135] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer (Springer Series in Operations Research and Financial Engineering), 2006.

[136] Bee Suan Ong. *Structural Analysis and Segmentation of Music Signals*. PhD thesis, University Pompeu Fabra, Barcelona, Spain, 2007.

[137] Nobutaka Ono, Kenichi Miyamoto, Hirokazu Kameoka, and Shigeki Sagayama. A real-time equalizer of harmonic and percussive components in music signals. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 139–144, Philadelphia, Pennsylvania, USA, 2008.

[138] Nicola Orio, Serge Lemouton, and Diemo Schwarz. Score following: State of the art and new developments. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, pages 36–41, Montreal, Canada, 2003.

[139] Laurent Oudre, Yves Grenier, and Cédric Févotte. Template-based chord recognition: Influence of the chord types. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR)*, Kobe, Japan, 2009.

[140] Hélène Papadopoulos and Geoffroy Peeters. Large-scale study of chord estimation algorithms based on chroma representation and HMM. In *Content-Based Multimedia Indexing (CBMI)*, pages 53–60, 2007.

[141] Hélène Papadopoulos and Geoffroy Peeters. Simultaneous estimation of chord progression and downbeats from an audio file. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 121–124, 2008.

[142] Brian Pardo and William Birmingham. The chordal analysis of tonal music. Technical report cse-tr-439-01, University of Michigan, Dept. of Electrical Engineering and Computer Science, 2001.

[143] Jouni Paulus and Anssi P. Klapuri. Measuring the similarity of rhythmic patterns. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 150–156, Paris, France, 2002.

[144] Jouni Paulus, Meinard Müller, and Anssi P. Klapuri. Audio-based music structure analysis. In *Proceedings of the 11th International Conference on Music Information Retrieval (ISMIR)*, pages 625–636, Utrecht, The Netherlands, 2010.

[145] Geoffrey Peeters. Sequence representation of music structure using higher-order similarity matrix and maximum-likelihood approach. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 35–40, Vienna, Austria, 2007.

[146] Pavel A. Pevzner. *Computational Molecular Biology: An Algorithmic Approach*. MIT Press, 2000.

[147] Jeremy Pickens, Juan Pablo Bello, Giuliano Monti, Tim Crawford, Matthew Dovey, Mark Sandler, and Don Byrd. Polyphonic score retrieval using polyphonic audio. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Paris, France, 2002.

[148] Mark D. Plumbley, Samer A. Abdallah, Juan Pablo Bello, Mike E. Davies, Giuliano Monti, and Mark B. Sandler. Automatic music transcription and audio source separation. *Cybernetics and Systems*, 33(6):603–627, 2002.

[149] John G. Proakis and Dimitris G. Manolakis. *Digital Signal Processsing.* Prentice Hall, 1996.

[150] Lawrence Rabiner and Bing-Hwang Juang. *Fundamentals of Speech Recognition.* Prentice Hall Signal Processing Series, 1993.

[151] Stanislaw Andrzej Raczynski, Nobutaka Ono, and Shigeki Sagayama. Multipitch analysis with harmonic nonnegative matrix approximation. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 381–386, 2007.

[152] Christopher Raphael. Automatic segmentation of acoustic musical signals using hidden markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:360–370, 1998.

[153] Christopher Raphael. A probabilistic expert system for automatic musical accompaniment. *Journal of Computational and Graphical Statistics*, 10(3):487–512, 2001.

[154] Christopher Raphael. A hybrid graphical model for aligning polyphonic audio with musical scores. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 387–394, Barcelona, Spain, 2004.

[155] Christopher Raphael and Josh Stoddard. Functional harmonic analysis using probabilistic models. *Computer Music Journal*, 28(3):45–52, 2004.

[156] Lise Regnier and Geoffroy Peeters. Singing voice detection in music tracks using direct voice vibrato detection. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1685–1688, Taipei, Taiwan, 2009.

[157] Christophe Rhodes and Michael A. Casey. Algorithms for determining and labelling approximate hierarchical self-similarity. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 41–46, Vienna, Austria, 2007.

[158] Christophe Rhodes, David Lewis, and Daniel Müllensiefen. Bayesian model selection for harmonic labelling. In *Proceedings of the International Conference on Mathematics and Computation in Music (MCM). Revised Selected Papers (Communications in Computer and Information Science)*, pages 107–116. Springer, 2009.

[159] Matti Ryynänen and Anssi P. Klapuri. Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music Journal*, 32(3):72–86, 2008.

[160] Craig Stuart Sapp. Comparative analysis of multiple musical performances. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 497–500, Vienna, Austria, 2007.

[161] Eric D. Scheirer. Tempo and beat analysis of acoustical musical signals. *Journal of the Acoustical Society of America*, 103(1):588–601, 1998.

[162] Ricardo Scholz and Geber Ramalho. COCHONUT: Recognizing complex chords from MIDI guitar sequences. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 27–32, 2008.

[163] Diemo Schwarz, Nicola Orio, and Norbert Schnell. Robust polyphonic midi score following with hidden Markov models. In *International Computer Music Conference (ICMC)*, Miami, Florida, US, 2004.

[164] J. Serrà, E. Gómez, P. Herrera, and X. Serra. Chroma binary similarity and local alignment applied to cover song identification. *IEEE Transactions on Audio, Speech and Language Processing*, 16:1138–1151, 2008.

[165] Shai Shalev-Shwartz, Shlomo Dubnov, Nir Friedman, and Yoram Singer. Robust temporal and spectral modeling for query by melody. In *Proceeding of the International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 331–338, Tampere, Finland, 2002.

[166] Adiel Ben Shalom, Shai Shalev-Shwartz, Michael Werman, and Shlomo Dubnov. Optimal filtering of an instrument sound in a mixed recording using harmonic model and score alignment. In *Proceedings of the International Computer Music Conference (ICMC)*, Miami, USA, 2004.

[167] Madhusudana Shashanka, Bhiksha Raj, and Paris Smaragdis. Probabilistic latent variable models as nonnegative factorizations (article id 947438). *Computational Intelligence and Neuroscience*, 2008.

[168] Alexander Sheh and Daniel P.W. Ellis. Chord segmentation and recognition using EM-trained hidden Markov models. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 185–191, Baltimore, USA, 2003.

[169] Roger N. Shepard. Circularity in judgments of relative pitch. *Journal of the Acoustic Society of America*, 36(12):2346–2353, 1964.

[170] Daniel Sleator and David Temperley. The Melisma Music Analyzer. `http://www.link.cs.cmu.edu/music-analysis/`, 2003.

[171] Paris Smaragdis and Judith C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 177–180, 2003.

[172] Paris Smaragdis and Gautham J. Mysore. Separation by humming: User guided sound extraction from monophonic mixtures. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 69–72, New Paltz, NY, USA, 2009.

[173] Temple F. Smith and Michael S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[174] Ferréol Soulez, Xavier Rodet, and Diemo Schwarz. Improving polyphonic and poly-instrumental music to score alignment. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 143–148, Baltimore, USA, 2003.

[175] Adam M. Stark and Mark D. Plumbley. Real-time chord recognition for live performance. In *Proceedings of the International Computer Music Conference (ICMC)*, Montreal, Canada, 2009.

[176] David Temperley. *The Cognition of Basic Musical Structures.* MIT Press, 2001.

[177] David Temperley. *Music and Probability.* MIT Press, 2007.

[178] H. Terasawa, M. Slaney, and J. Berger. The thirteen colors of timbre. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 323–326, 2005.

[179] Verena Thomas, Christian Fremerey, David Damm, and Michael Clausen. SLAVE: a Score-Lyrics-Audio-Video-Explorer. In *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR 2009)*, pages 717–722, 2009.

[180] Steven K. Tjoa, Matthew C. Stamm, W. Sabrina Lin, and K. J. Ray Liu. Harmonic variable-size dictionary learning for music source separation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 413–416, Dallas, TX, USA, 2010.

[181] Robert J. Turetsky and Daniel P.W. Ellis. Ground-truth transcriptions of real music from force-aligned MIDI syntheses. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 135–141, Baltimore, USA, 2003.

[182] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.

[183] Yushi Ueda, Yuuki Uchiyama, Takuya Nishimoto, Nobutaka Ono, and Shigeki Sagayama. HMM-based approach for automatic chord detection using refined acoustic features. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5518–5521, Dallas, USA, 2010.

[184] Emmanuel Vincent, Rémi Gribonval, and Cédric Févotte. Performance measurement in blind audio source separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(4):1462–1469, 2006.

[185] Tuomas Virtanen. *Sound Source Separation in Monaural Music Signals*. PhD thesis, Tampere University of Technology, 2006.

[186] Tuomas Virtanen. Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *IEEE Transactions on Audio, Speech and Language Processing*, 15(3):1066–1074, 2007.

[187] Ye Wang, Min-Yen Kan, Tin Lay Nwe, Arun Shenoy, and Jun Yin. Lyrically: automatic synchronization of acoustic musical signals and textual lyrics. In *Proceedings of the ACM International Conference on Multimedia*, pages 212–219, New York, NY, USA, 2004.

[188] Ron J. Weiss and Juan Pablo Bello. Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 123–128, Utrecht, The Netherlands, 2010.

[189] Felix Weninger, Alexander Lehmann, and Björn Schuller. OpenBliSSART: Design and evaluation of a research toolkit for blind source separation in audio recognition tasks. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1625–1628, Prague, Czech Republic, 2011.

[190] Gerhard Widmer. Using ai and machine learning to study expressive music performance: project survey and first report. *AI Communications*, 14(3):149–162, 2001.

[191] Gerhard Widmer, Simon Dixon, Werner Goebl, Elias Pampalk, and Asmir Tobudic. In search of the Horowitz factor. *AI Magazine*, 24(3):111–130, 2003.

[192] Terry Winograd. Linguistics and the computer analysis of tonal harmony. *Journal of Music Theory*, 12:2–49, 1968.

[193] John Woodruff and Bryan Pardo. Using pitch, amplitude modulation, and spatial cues for separation of harmonic instruments from stereo music recordings (article id 86369). *EURASIP Journal on Advances in Signal Processing*, 2007.

[194] John Woodruff and Bryan Pardo. Active source estimation for improved source separation. Technical Report NWU-EECS-06-01, Electrical Engineering and Computer Science Department, Northwestern University, 2006.

[195] John Woodruff, Bryan Pardo, and Roger B. Dannenberg. Remixing stereo music with score-informed source separation. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 314–319, 2006.

[196] Jun Wu, Emmanuel Vincent, Stanislaw Andrzej Raczynski, Takuya Nishimoto, Nobutaka Ono, and Shigeki Sagayama. Multipitch estimation by joint modeling of harmonic and transient sounds. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 25–28, Prague, Czech Republic, 2011.

[197] Shangming Yang and Zhang Yi. Convergence analysis of non-negative matrix factorization for BSS algorithm. *Neural Processing Letters*, 31:45–64, 2010.

[198] Wei You and Roger B. Dannenberg. Polyphonic music note onset detection using semi-supervised learning. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, Vienna, Austria, 2007.

[199] Eberhard Zwicker and Hugo Fastl. *Psychoacoustics, facts and models.* Springer Verlag, New York, NY, US, 1990.

Index

Curriculum Vitae

Name:	Sebastian Ewert	
Date of Birth:	13. October 1980	
Place of Birth:	59457 Werl, Germany	
Martial Status:	Single	
School Education:	1987-1991	Primary school, Marien Grundschule Werl-Büderich
	1991-2000	Grammar school, Marien Gymnasium Werl
	05/2000	Abitur (higher education entrance qualification)
Civilian Service:	2000-2001	Mariannen-Hospital Werl
University Career:	2001	Studies of computer science and mathematics at the University of Bonn
	2003	Pre-Diploma in computer science
	12/2007	Graduation in computer science (Diploma)
Experience:	2008-2010	Research position funded by the German Research Foundation, Multimedia Signal Processing Group, University of Bonn
	since 2010	Assistant position, Multimedia Signal Processing Group, University of Bonn

Sebastian Ewert