Saarland University
Faculty of Natural Sciences and Technology I
Department of Computer Science

Master's Thesis

Determining Tempo Characteristics of Expressive Music Recordings: An Algorithmic Approach

submitted by

Andi Scharfstein

submitted

May 11, 2009

Supervisor / Advisor

Priv.-Doz. Dr. Meinard Müller

Reviewers

Priv.-Doz. Dr. Meinard Müller
Prof. Dr. Michael Clausen

Acknowledgements

During the time of working on this thesis, I have enjoyed the support of a great number of people. I would like to express my gratitude for this fact by explicitly mentioning some of these people who were directly or indirectly involved in helping me succeed in this undertaking.

First of all, greatly deserved thanks go to Meinard Müller for offering me such an interesting topic, and for the exceptional care he provided as my supervisor. It has been a great pleasure operating in the friendly atmosphere of his workgroup! A large share of the credit for this is of course due to my direct collaborator (and unofficial advisor) Verena Konz—our technical discussions never failed to help me gain insight into the problem at hand, and our excursions to the vending machine provided much-needed relief in stressful times. In the same vein, I'd like to extend my thanks to Thomas Helten for his faithful attendance at our little jogging breaks, as well as for the vastly helpful proofreading feedback he provided.

One former member of Meinard's workgroup contributed much time and effort as well: Sebastian Ewert (University of Bonn) provided substantial support in the creation of Chapter 4 by making available a complete framework for the generation and quantitative evaluation of tempo curve performance data. His help and expertise (in this as well as in other matters) are greatly appreciated.

Of course, personal friends have been very supportive as well. Stefan Winterstein has been a mainstay among my proofreaders and invariably gave detailed and insightful feedback, especially on early drafts of the work. Fabian Sobotka and Martin Schreiber both did great proofreading work and were ready to encourage and motivate me whenever necessary, and I feel lucky to count them among my friends. The same goes for Harm-Friedrich Steinmetz and Jan Christoph—Jan in particular deserves special mention as a great motivating factor in my quest to complete the work on time; I don't think I could have done it without the strength of his example. Radu Curticapean managed to pick out an astonishing amount of typing errors at the last minute, for which I am particularly grateful.

Last but definitely not least, I'd like to thank all members of my family for their patience during this period and the preceding years of study. I consider myself fortunate to have them; their support means a lot to me and has never been taken for granted.

Contents

Chapter 1

Introduction

We do not wish any so-called interpretations of our music; just play the notes; add nothing, and take nothing away.

–Aaron Copland paraphrasing "modern" compositional attitude,
WHAT TO LISTEN FOR IN MUSIC *(1939)*

The history of Western classical music is rich with examples that document the importance of a performer's skill and artistry for conveying musical visions: Johann Sebastian Bach was famous for his feats of improvisation, not for his reconciliation of harmonic and contrapuntal concerns. Clara Schumann enjoyed greater contemporary success than her husband Robert, as she was the better performer of the two. Niccolò Paganini was able to build a whole career on virtuosity alone. Even the sporadic extreme counter-reaction to this fact (such as in the above quote) only serves to underline the relevance of the subject.[1] Yet, in the field of music information retrieval, the topic focusing on this particular aspect—performance analysis—has only recently begun to receive the attention it therefore deserves [Gab03, Wid02, WDG$^+$03, LG03, Wid05, Sap07, Sap08]. Part of the problem is the inadequacy of current methods for the determination of such elementary performance attributes as tempo or dynamic shaping: Automated processes are too unreliable and error-prone for widespread usage, and most current performance analysis studies fall back on manual feature annotation. In cases where manual annotation is not an option, another common approach is to use special-purpose hardware such as *player pianos* to transcribe symbolic information about the music at the time it is played. Of course, neither of these procedures generalize to settings where such data is not available or manual annotation is considered too labor-intensive.

In this thesis, we present an alternative approach to the extraction and processing of attributes of music data which relate to its agogical aspects, in particular the tempo. This approach does not rely on manually generated data or symbolic transcriptions of specific performances, but instead employs a general alignment technique known as *dynamic time warping* (DTW). The main contribution of this thesis lies in the development of methods to exploit the implicit tempo information contained in the so-called *warping path* produced by the DTW technique to generate data suitable for subsequent musical analysis. For this, we mostly stay confined to the domain of piano music produced in the Western classical period. This is due to several reasons: It provides a large pool of available performance data (e.g. CD recordings) that lends itself reasonably well to automatic processing with DTW; it is not restricted to musically

[1]In the context of the quote, Copland disagrees with the expressed sentiment (calling it a "nonrealistic attitude on the part of the composers") and quickly goes on to dispel the notion that an interpretation-less performance is at all possible, or even desirable.

trivial “academic” examples but makes use of real-world data; and by employing a player piano, we were able to generate custom data pairs of acoustic and symbolic representations of the same piece of music that facilitate a proper evaluation of our techniques.

The thesis is structured deliberately in such a way that it follows the complete process of performance analysis chronologically: Beginning at feature design and extraction, going over the processing steps necessary to compute more complex data, and finally closing with the musical analysis and evaluation of this data. The main focus and contribution of the work lies in the middle part of this process. Specifically, we explore several techniques for the automatic computation of tempo parameters from a piece of music. These can be visualized in the form of a *tempo curve* that plots the tempo of the piece for each point in time, as e.g. in Figure 1.1. The depicted curve shows the tempo for an excerpt from Schubert’s “Winterreise” as computed by the approach presented in this work; one can clearly see the artistic shaping of the two phrases, in particular a slow-down at each phrase ending. Note that tempo here refers to a highly localized view of the piece where we may even compute the tempo of individual notes, unlike e.g. with beat trackers that only estimate an average overall tempo of a piece.

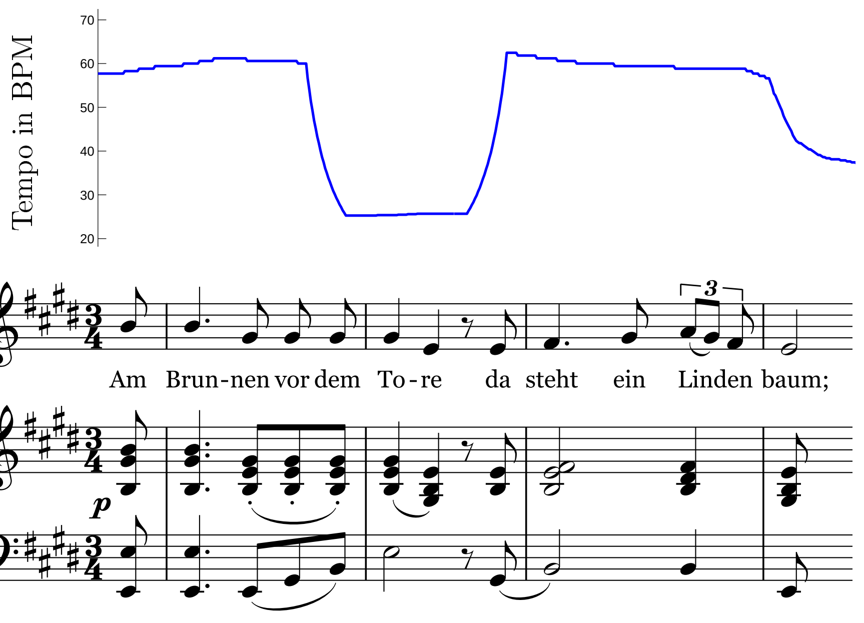

Figure 1.1: Franz Schubert: Winterreise D911, “Der Lindenbaum” (Excerpt)

The detailed organization of the thesis is as follows:

Chapter 2 introduces basic terminology and a set of features and methods to extract these features from different music representations. It also presents the DTW alignment procedure and explains how the features form the basis of the alignment that is gained by this procedure.

Chapter 3 contains some thoughts on tempo and tempo measurement principles and shows how the warping path computed by the DTW technique may serve as the defining characteristic of such attributes of a musical piece. Three techniques are discussed that attempt to compute tempo data on the basis of the warping path using different approaches.

Chapter 4 gives two evaluations of the techniques presented in the third chapter: The first focuses on a quantitative data analysis for an objective performance measure, the second illustrates the kind of results that can be obtained from applying these techniques by discussing some tempo curves generated for selected musical examples.

Chapter 5 has an overview of several related works that use tempo curve information (or similar data) for further analysis steps. These works can be classified roughly according to two main goals: One group focuses on commonalities of different performances, the other is concerned with their respective differences.

Chapter 6 concludes the thesis by giving a summary of what has been achieved, and identifies some opportunities for further research based on the results obtained in this work.

Chapter 2

Music Synchronization

In fact the kind of music [Havelock Vetinari] really liked was the kind that never got played. It ruined music, in his opinion, to torment it by involving it on dried skins [...] and lumps of metal hammered into wires and tubes. It ought to stay written down, on the page, in rows of little dots and crotchets, all neatly caught between lines. Only there was it pure.

–Terry Pratchett, SOUL MUSIC *(1994)*

This chapter lays the foundation for the subsequent discussion concerning the main contribution of this thesis. Following a short overview of the possible forms of music representation relevant for this work, we introduce a number of descriptors of such music representations called *features* which capture essential attributes of the data while discarding irrelevant (or representation-specific) information. We also demonstrate how these features are generated from the data available, and how they are processed afterwards. The presentation order of the features is chronological in that the feature produced from a particular processing step forms the input for the next step (see Fig. 2.5 for an overview of the complete processing pipeline).

After the feature processing pipeline has been shown in full, we explain how to make use of the computed features by including them in an alignment procedure known as *dynamic time warping*. This procedure is presented in a general fashion first, then an examination of how it is accommodated for the specific needs of *music synchronization* follows in a second step. Music synchronization here refers to the process of finding the semantic correlations between two different interpretations of the same musical piece.

The most important concept introduced in this chapter is the *warping path*, which appears in the discussion of dynamic time warping. It will be used as the basis for all computations in the following chapters, in particular with regard to the tempo curves introduced in Chapter 3. The warping path stores all correlation information gained in the synchronization process.

2.1 Music Representations

The first step in understanding how to process music in an automated manner is to get a clear view on the different formats in which music may be represented.[2] In this work, we will focus on three different representations. Each of these has its individual strong points, which in turn necessitate individual processing approaches. They can be classified according to the relative degree of abstractness by which the music they embody is represented:

[2]This chapter mostly follows [Mül07] in the presentation of the different formats, features and algorithms.

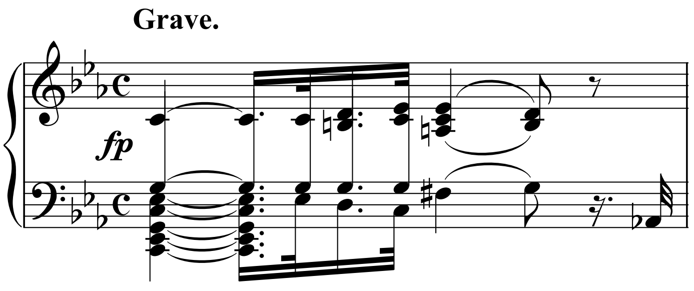

Figure 2.1: Ludwig van Beethoven: Sonata Pathétique, beginning of the first movement (**PathBeg**)

Score representation. The most "purely" symbolic representation among the three can be regarded as a sort of reference or *ground truth* against which more concrete realizations of the same piece may be measured in order to gain information about specifics of a particular performance.

MIDI representation. A kind of hybrid that (more or less successfully) attempts to incorporate both abstract information about a piece and concrete information about a specific realization of the piece, while still remaining on a symbolic level.

Audio representation. The audio representation favors concrete information over symbolic data, opting to capture a data set that allows nearly perfect reconstruction of a specific interpretation of a piece while completely ignoring semantic correlations of that interpretation to symbolic representations of the same piece. This enables extraction of very nuanced interpretational features, but makes it difficult to give a semantically meaningful analysis of the data.

While the MIDI representation of music data exists by its nature only in a digital format, both of the other representations have analog origins. Since we are mainly interested in automated music processing, we will briefly discuss how to convert "real-world" score sheets and physical audio data into their digital counterparts. Following this, we will assume that our representations occur as digital versions only (unless explicitly stated otherwise). This discussion will take place as we look at the details and idiosyncrasies of each representation in turn.

2.1.1 Score Representation

A *musical score* (in analog contexts also referred to as *sheet music*) contains a formalized description of a particular piece of music. In its typical non-digitized form, it consists of graphical and textual information on note pitches and lengths as well as meta-information about loudness, tempo and other musical attributes of individual notes or sections of the piece. This information is organized according to periodic structural divisions, so that each group of notes of a certain overall length is contained in a *bar* (or *measure*). As an example, Figure 2.1 depicts the first bar of the first movement of Beethoven's Sonata Pathétique. The same excerpt will be used to illustrate the other concepts introduced in this chapter wherever possible. It will be referred to as **PathBeg**, for "Pathétique beginning". We also introduce **PathExp** as a shorthand for the complete exposition of this piece (measures 1–132).

The score may prescribe the overall tempo of a piece in an absolute measure of *beats per minute*, but is not required to do so. In fact, composers commonly resort to relative tempo markings like *Andante*, *Presto* or *Lento* instead, since this allows them to account for the performer's artistic freedom in the interpretation of the piece. Terms like those mentioned denote loosely defined tempo ranges that depend on a performer's musical intuition and experience to choose a specific tempo for an actual realization of the piece. This dependency is true for other musical attributes as well, e.g. local tempo, dynamics, articulation and all agogical aspects (like use of *Rubato*). Hence, most of these attributes must either be notated in relative terms or even left implicit. *Dynamics* (instructions pertaining to loudness) are always notated in relative terms such as *mezzoforte* or *pianissimo*, and even explicit articulation instructions like *staccato* will require artistic interpretation for a concrete performance.

The high abstractness degree of score data as compared to the other two representations accounts for both the main advantage and the main disadvantage of this representation. Extracting musically meaningful information from a digitized score sheet may be nearly trivial (such as with pitch and onset information for specific notes), or at least feasible (e.g. searching for and locating recurring motivic ideas inside a particular score), which is typically not the case for the other representations. On the other hand, the score was never meant to be able to represent interpretation-specific information, but rather to indicate to the performer how an interpretation should "sound". While this is of course perfectly legitimate, not being able to account for concrete performance data nevertheless disqualifies this representation from being used as a basis for performance analysis tasks like the computation of complex timing information of an interpretation.

Converting an analog representation of a score into its digital version is not a trivial feat. We will refrain from dealing with this issue here (see e.g. [Mül07] for a more complete discussion), but simply explain how the digital version might be represented. The underlying assumption is that the score may always be digitized manually, if everything else fails.

The most trivial digital representation of a score sheet is the graphical equivalent of the analog version in the form of some bitmap image format—in this case, nothing is gained by the digitization process, since the graphical representation is not well-suited for automatic processing. What is needed instead is a semantically meaningful symbolic representation of the score. One such representation format is defined by the MusicXML standard [Rec09] which (as the name suggests) is a Document Type Definition for XML documents that contain musical data. XML documents prepared according to this definition can easily be converted to a plethora of other formats and representations, including a graphical score sheet rendition, MIDI output and proprietary formats for commercial scorewriters such as *Sibelius* or *Finale*—even more esoteric output targets like e.g. *Braille Music* are easily achieved. While widespread adoption of this standard has been relatively slow, a clearly superior or more popular format definition has not yet emerged.

Note that even the minimal example given in Figure 2.2 (which produces a staff containing one whole note, an a$'$) is quite verbose. The reason for this is that MusicXML aims to store semantic information about the music as well as the layout information necessary to produce an æsthetically pleasing graphical output. Other formats focus on more specific goals, e.g. LilyPond [Lil09] offers good score layouting capabilities, while Humdrum [Hum09] seems more concerned with providing a data format that is well-suited for musical post-processing.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE score-partwise PUBLIC "-//Recordare//DTD MusicXML 2.0 Partwise//EN"
                                "http://www.musicxml.org/dtds/partwise.dtd">
<score-partwise>
  <part-list>
    <score-part id="P1">
      <part-name>MusicXML Example</part-name>
    </score-part>
  </part-list>
  <part id="P1">
    <measure number="1">
      <attributes>
        <divisions>1</divisions>
        <key>
          <fifths>0</fifths>
        </key>
        <time>
          <beats>4</beats>
          <beat-type>4</beat-type>
        </time>
        <clef>
          <sign>G</sign>
          <line>2</line>
        </clef>
      </attributes>
      <note>
        <pitch>
          <step>A</step>
          <octave>4</octave>
        </pitch>
        <duration>4</duration>
        <type>whole</type>
      </note>
    </measure>
  </part>
</score-partwise>
```

Figure 2.2: MusicXML example code

2.1.2 MIDI Representation

Musical Instrument Digital Interface, MIDI for short, was created as a protocol for the exchange of musical information between electronic instruments. It defines a large variety of messages, starting with basics such as describing notes in terms of their pitch and onset/release times with regard to a global time stream, and going further to include meta information and control messages such as “Distort pitch by a specific factor”. Since its conception in 1981, it has become the *de facto* standard used (or at least supported) by almost any application (including hardware) that deals with the transmission or processing of symbolic music data.

For automated music processing, it is useful to gather the protocol messages which make up a single piece of music into one file, which can then be distributed or modified. The MIDI standard provides for this with the SMF (*Standard MIDI File*) file format. In the following, when we refer to the MIDI representation of a piece, we will usually mean the respective SMF.

One major shortcoming of MIDI in comparison to an audio stream is its lack of support for the representation of timbral attributes of a sound. On the other hand, the kind of information which *is* supported is highly useful for a large variety of performance analysis tasks. Of particular importance for this context are two commands: *note-on* and *note-off*, which together serve to define the relative length of a note, as well as its position in the global time stream. This time stream is defined in terms of *ticks* or *clock pulses*, which are sent out periodically during the duration of the piece. The MIDI protocol is designed in such a way that these ticks measure musical time units instead of absolute time units such as seconds: Note lengths are defined in terms of the number of clock ticks passed. The respective unit of measurement is called *pulses per quarter note* (PPQN), and the PPQN for a single piece is specified by the user in the MIDI file header. Tempo of a piece is then controlled by changing the clock pulse frequency. Hence, the absolute length of a time interval can be determined by examining PPQN and tick frequency for that segment. Typical PPQN values are 480 and 960; in general, is is given by $24 \cdot 2^n$ and $30 \cdot 2^n$ for $n \in \{2, ..., 6\}$.

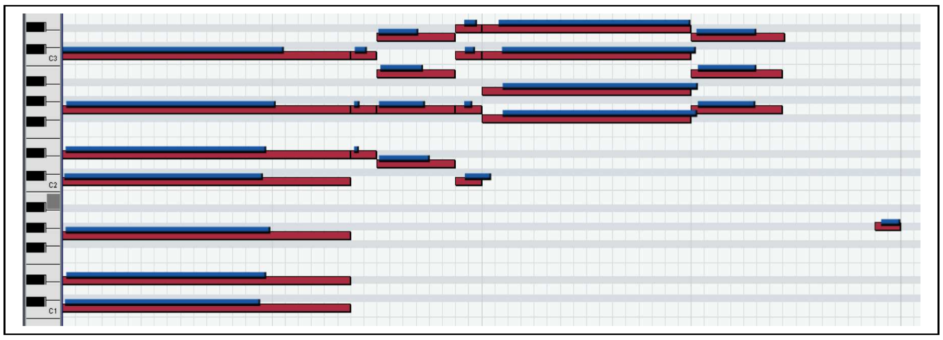

Figure 2.3: **PathBeg**, MIDI piano roll representation. Reference data generated from the score is shown in red, a specific interpretation of the piece is overlayed in blue.

As we have just seen, the onset/offset times of a note in MIDI representation is explicit and very specific in its nature. This allows a very nuanced encoding of tempo changes in the performance of a piece, as opposed to sheet music representations where such subtleties are mostly lost in the notation. The difference can be seen in the so-called *piano roll* representation, which is often employed for the visualization of MIDI data (Fig. 2.3): Each column represents a discrete musical time interval, usually divided at least on the beat level.[3] Pitches are distributed vertically, such that each row corresponds to a semitone step on the twelve-tone equal tempered scale. Notes are displayed as horizontal bars in this structure. The example shows two different versions of the beginning of Beethoven's Pathétique—in red, a "ground-truth" version generated as a straightforward translation of the score representation of the same segment into MIDI data, and superimposed in blue, a real-life interpretation of the same segment. We can plainly see the discrepancy between the different versions, and it is in fact exactly this difference which will be used in later chapters of this thesis to work out

[3]Intuitively, the *beat* of a piece of music corresponds to the points in time where a listener might tap his foot to keep in rhythm with the music. For details, see Section 3.1.1.

the artistic idiosyncrasies of a specific interpretation with regard to tempo/timing attributes. From the example it is also apparent that the heightened specificity of MIDI comes with a loss in semantic expressiveness: Note timings no longer have musical connotations (e.g. "first note of a new measure"), but float "freely" in the global time stream.[4]

The other main attribute dimension (besides timing) we may be interested in concerns the dynamics of the piece. MIDI does not have a direct way of encoding loudness, in the same way it does not feature a method of representing the timbre of a sound. Instead, it provides a command for specifying the *velocity* of a note, which corresponds to the intensity with which the note should be "played" (i.e., synthesized) by the output device. MIDI reserves seven bits for this information, so 128 different velocity values can be represented. This information is not displayed in the piano roll representation of MIDI data.

2.1.3 Audio Representation

The audio (or *waveform*) representation of music data is the most concrete of the three formats presented here, it contains no abstract musical information at all. The name "waveform" is derived from the graphical representation of such a signal, where pressure changes in a carrier medium (usually air) are plotted over time (Fig. 2.4 shows an interpretation of the beginning of the Sonata Pathétique). For a *pure tone*, this plot will yield a sinusoidal oscillation around the zero reference (which is just the pressure of the carrier medium in an unexcited state). The maximum distance between reference and waveform in this context is called the *amplitude* a of the signal, while the time difference between two consecutive repetitions of an oscillation is called the *period* T. The *frequency* f of the signal is defined as the inverse of the period, $f = \frac{1}{T}$. When we talk about the *pitch* of a sound, we usually refer to the acoustic property of this sound that corresponds to this frequency.

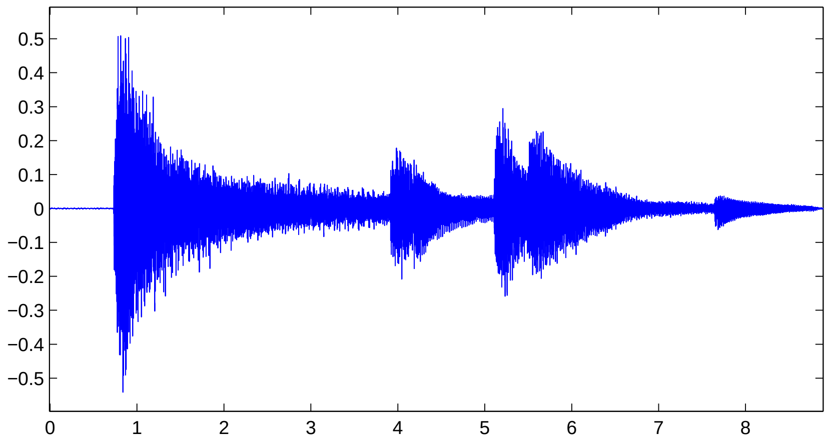

Figure 2.4: **PathBeg**, waveform representation, time in seconds

[4]Pitches cease to carry musical information about the key in which the piece is notated as well: G♭ and F♯ for example are both represented as MIDI pitch 42. This is legitimated by the concept of enharmonic equivalence in the equal tempered scale (see e.g. [Mic08], p. 85).

In the analog case, all of these parameters are measured continuously. However, for a digital context one must define a suitable quantization scheme when discretizing the data. In this work, we follow the *Pulse Code Modulation* (PCM) scheme laid out in the "Red Book" standard defined for CDDA (*Compact Disc Digital Audio*), the most popular format in use today. CDDA uses a sampling rate of 44.1 kHz (i.e. 44,100 samples per second are taken) at a resolution of 16 bit—hence, it is possible to distinguish 65,536 different "pressure points" corresponding to the relative oscillation of the waveform. This is sufficiently detailed for the average human listener not to notice any differences to an actual performance of the discretized data (although several alternative schemes with different sample and bit rates have been proposed, e.g. *Direct Stream Digital* for the *Super Audio CD* or PCM at 96 kHz/24 bit for *DVD-Audio*).

For the purposes of performance analysis, the audio representation offers an interesting mix of properties: On one hand, the sheer quantity of interpretations only available in a waveform representation (e.g. as a CD recording) is a convincing argument already to make this a primary analysis target. On the other hand, the complete lack of symbolic information in this format poses serious problems when trying to extract musically meaningful data. Unsolved problems in the domain of *music information retrieval* from audio data (MIR) include the determination of the *fundamental frequency* of a sound,[5] automatic separation of different instruments or voices in a polyphonic context (the so-called *source separation* problem) and alignment of the recording to a score sheet, although progress has been made to varying degrees in either of these topics. The *alignment* or *synchronization* problem is of particular interest, since its solution offers a way out of this conundrum: Aligning waveform and score sheet would combine concrete and symbolic information about the data in one meta-format that would lend itself to a great variety of analysis tasks. Section 2.4 introduces one possible approach for tackling this problem.

2.2 Pitch- and Chroma-Based Audio Features

As we have learned in the preceding section, the representations of music data we are interested in have vastly different attributes, making direct comparisons between them impossible. Yet even if the representation is the same for two interpretations of a piece, direct comparisons will often yield unsatisfactory results if they operate on a purely "syntactical" level (e.g., bit-level comparison of the waveforms). Instead, it would be desirable to have an overarching concept of similarity (i.e., a *similarity metric*) that has a semantic meaning. Such a concept should abstract from the format of the data, such that it can be used to compare two pieces of music regardless of how they are represented.

In trying to define a similarity metric, one has to keep in mind several aspects that need to be reconciled with each other in order for the metric to be useful. For example, melodic similarity would be a musically meaningful metric, but in a polyphonic context requires strong assumptions in order to solve the prerequisite *source separation* problem (cf. [Bur08]). Hence, the metric should be computationally feasible. It is also important to keep in mind robustness

[5]A sound is typically made up by a series of *overtones* (or *harmonics*) that grant it a unique texture or tone color. By definition, the fundamental frequency corresponds to the first of these overtones.

of the metric, so that it does not break even in the face of differently orchestrated interpretations of the same piece (e.g. Liszt's piano arrangement of Beethoven's Fifth Symphony vs. the original setting), or when the two performances use different instrument tunings (which is usually the case for so-called *historically informed* or *authentic* performances[6]). One metric that fulfills the requirements of robustness, computability and, above all, has a straightforward musical interpretation has been proposed by Müller in [Mül07]. The idea is to determine the *harmonic similarity* of two pieces, but since this is not an explicit attribute of a piece in any representation, it introduces a number of features that approximate said measurement. We will briefly present these features in the context of waveform analysis before discussing how they are used to provide a musically meaningful metric to a music synchronization algorithm in Section 2.4. Figure 2.5 provides a rough overview of the audio feature extraction process we will traverse in the course of this discussion.

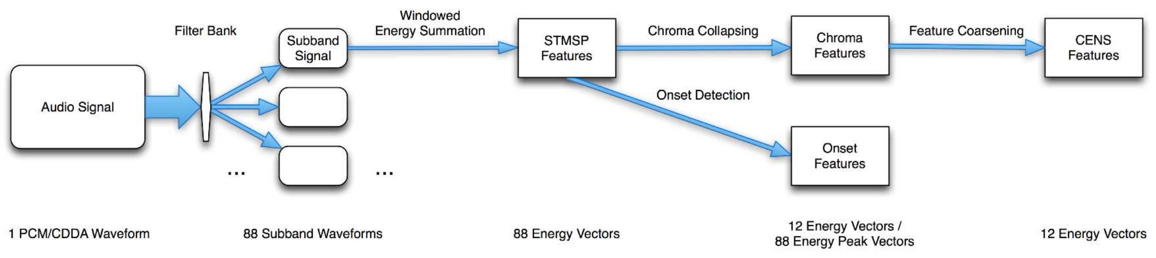

Figure 2.5: An overview of the feature extraction pipeline

2.2.1 Pitch Features

The first step in the feature extraction pipeline deals with the decomposition of the audio signal into groups of frequency components, which are determined according to their association to pitches of the standardized equal tempered scale. Since this scale is designed primarily with regard to human perception of sound, it takes into account the well-known logarithmic nature of this perception. This nature is revealed in the fact that for any pitch s with frequency f, the pitch with frequency $2^{n-1} \cdot f$ will be perceived as being n times higher than s (e.g. A6 at 1760 Hz is perceived as being three times higher than A4 at 440 Hz). The interval between two consecutive such pitches is called an *octave*,[7] and it is clear that octaves (and indeed any musical interval between two notes) span different frequency ranges, depending on the respective base frequency. Any "musical" frequency grouping must necessarily take this into account. In our case, this is handled by an adaptive window size in the grouping of frequency ranges to specific pitches—as the absolute frequency decreases, so does the size of the respective window that determines if a specific frequency still belongs to an individual pitch.[8] On the technical side, this is implemented by an array of bandpass filters of varying size and sample

[6]Also called *period performances*, these use a standard pitch of $a' = 415\,\text{Hz}$ for early music instead of the modern standard of $a' = 440\,\text{Hz}$, corresponding to a lowering by one semitone.

[7]The name "octave" is derived from the range of eight notes which are contained in this musical interval on the diatonic scale (counting both ends). There is another interesting phenomenon concerning the perception of sound called *octave equivalence* which we will encounter in Section 2.2.3.

[8]In other words: Low pitches have smaller windows than high pitches, since there is less space between two low pitches than there is between two high pitches.

rate. Figure 2.6 shows a graphical representation of this where the varying interval sizes for a fixed sample rate are particularly well visible.

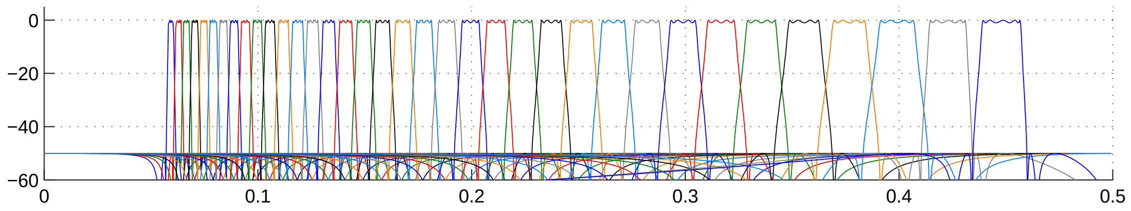

Figure 2.6: A sample array of filters with their respective magnitude responses in dB (reproduced from [Mül07] by permission)

Dealing with the technical details involved with the concrete realization of this design would stretch the scope of this work too much, so the interested reader is referred to the original monograph instead. Intuitively, after the frequency decomposition, one ends up with an array of 88 different signals corresponding to the contributions of each of the 88 pitches produced by a typical modern piano.

2.2.2 Local Energy (STMSP) Features

Decomposing an input signal into frequency groups corresponding to specific pitches enables us to measure the individual contributions each of the pitches makes to the overall signal. What is not yet clear is the unit of measurement for these contributions. Since we are looking for a measure closely correlated to the loudness of a certain signal, we choose *local energy* as measured by the *short-time mean-square power* (STMSP) of this signal. For a signal x_n of one specific subband n and some sampling points k taken from a time window of a (small) fixed

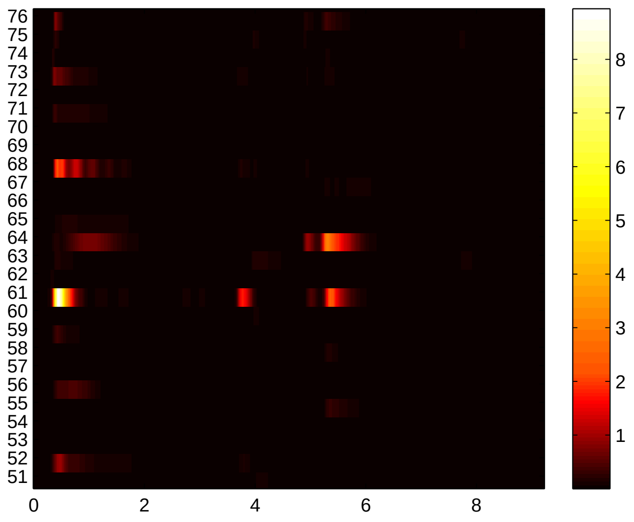

Figure 2.7: Selected STMSP features for **PathBeg**, time in seconds

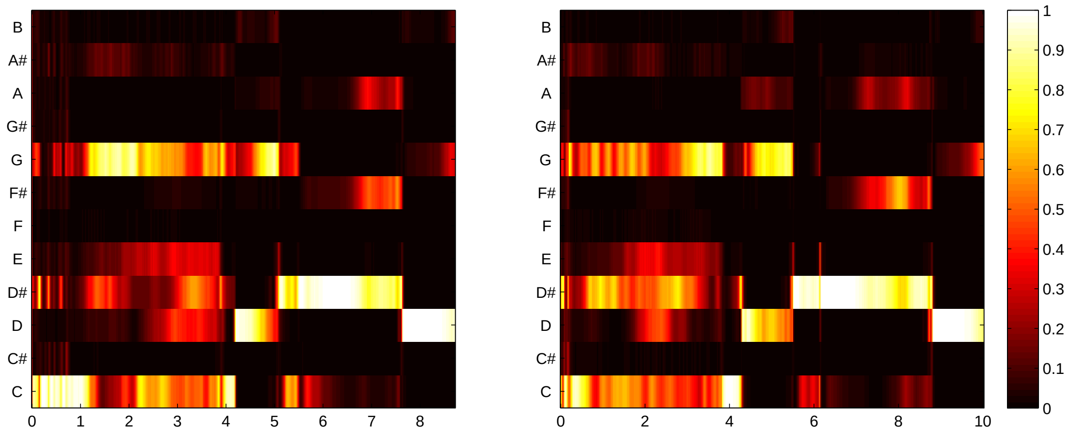

Figure 2.8: Chroma features for two different **PathBeg** versions, time in seconds

size, this is defined as $\sum_k |x_n(k)|^2$. The size of the time window is typically chosen somewhere in the order of a few milliseconds, the sampling rate in the order of about 88 Hz. The choice of both parameters depends on the original sample rate of the respective subband. They are defined in such a way that individual subbands yield comparable results in the STMSP computation. This step is performed multiple times while the time window is shifted across the whole of the input signal. The result is a sequence of individual STMSP features.

The example (Fig. 2.7) shows a time-pitch plot for the beginning of Beethoven's Sonata Pathétique, our running example. Note that only the most prominent pitches show up at all—this is due to the fact that individual pitch contributions are usually not significant with regard to the overall picture, and one has to do some more work to extract musically meaningful data from this basic feature.

2.2.3 Chroma Features

The next step in the extraction pipeline is concerned with a reduction of the feature space generated by pitch energy extraction to a 12-dimensional space that is suitable for harmonic post-processing. To understand precisely how this is done (and why it is a valid processing step), one has to consider the perceptual phenomenon known as *octave equivalence*: In human hearing, any two tones separated by a distance of one (or more) octave(s) are considered to "sound" alike, i.e. they are perceived as the same tone played in different pitch registers (for details see [Mic08], p. 21). This enables the classification of pitches along two dimensions, the first being the respective "note color" of the pitch as perceived by the human listener (which is the attribute of the sound that does not change across different octaves), and the second being an indication of the register in which the note is sounded. Usually, the first aspect is referred to as *chroma* (Greek for "color") and the second as *tone height*. As an example, consider the pitch A4: its chroma is designated by the "A", and the respective register or tone height is defined by the number 4. The entirety of pitches of a single chroma is sometimes also called a *pitch class*.

As we learn that each octave of the equal tempered scale consists of twelve notes corresponding to twelve different chroma, it becomes clear how the feature space reduction works: The information on single pitches is simply collapsed across octaves, such that one ends up with an indicator of the contribution all notes of a specific chroma make to the original signal, irrespective of the octave in which they are played. This is in accordance with the traditional theory of harmony which states that pitch register is largely irrelevant for the harmonic classification of a single chord. The collapsing step simply consists of adding up all individual note contributions (i.e., their STMSP) of the same chroma for each element of the STMSP feature sequence, so the computational effort required for this transformation is linear in the number of these elements. The transformation result, a vector containing a sequence of 12-dimensional features, is called *chroma representation* of the audio signal. Figure 2.8 depicts a sample output of this step, where the names of the chroma vectors have already been annotated.[9]

2.2.4 CENS Features

Conceptually, the chroma feature already gives a good approximation of the goal we were trying to achieve with our feature design, namely extracting the harmonic characteristics of a piece. Technically however, there is still some work left to do in order to make the feature more robust and invariant with regard to certain data attributes that should be discarded when focusing on harmonic similarity. This is done in the final phase of the feature extraction pipeline with the computation of *chroma energy normalized statistics* (CENS) features.

The computation of CENS features is done in multiple stages and is thus best understood in terms of being a pipeline itself, contained in the larger and more general feature extraction pipeline. The CENS processing steps are as follows:

1. **Normalization.** Each feature vector of the chroma representation is normalized to a range in the interval $[0, 1]$. In cases of near-silence, the uniform distribution is substituted for the actual vector to avoid introducing statistical noise.
2. **Quantization.** The normalized sectors are quantized according to a logarithmic binning function $b : [0, 1] \rightarrow \{0, 1, 2, 3, 4\}$. This coarsens the resolution to make the measure insensitive to local variations.
3. **Convolution.** The quantized vectors are convolved with a Hann window of a specific size to lessen the impact of local errors in the extracted features.
4. **Downsampling.** The resulting feature sequence is downsampled by a specific factor to further coarsen the resolution.
5. **Normalization.** Finally, the single vectors are again normalized with respect to the Euclidean norm.

The resulting CENS features are robust enough to ignore local variations in timbre, articulation and other performance-specific attributes, while correlating very strongly to the harmonic progression of a piece (Fig. 2.9). They form the basis of a "globally-oriented" alignment procedure which will be introduced in Section 2.4—in this context, globally-oriented should

[9]Note the enharmonic spelling of E$\flat$ as D$\sharp$.

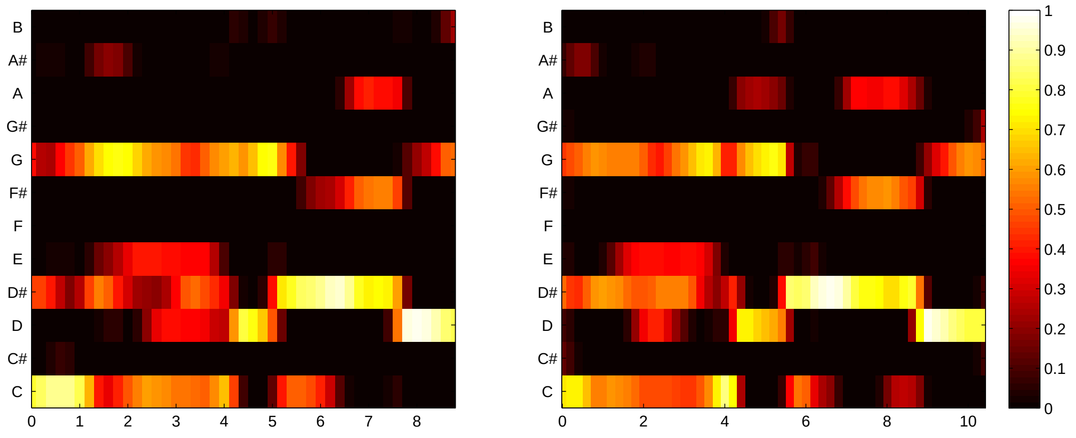

Figure 2.9: CENS features for the two **PathBeg** versions shown in Figure 2.8, time in seconds

be taken to mean that this feature is not suitable for the alignment of very subtle note-to-note correspondences, since it is designed to suppress this level of detail. We will see a potential source of “locally-oriented” alignment information in the next section.

2.2.5 Onset Features

CENS features are one example for the extraction of useful information from STMSP data on a particular audio stream. Another interesting feature that the STMSP representation gives rise to is concerned with the detection of *note onsets*. This information is especially useful for the synchronization (or alignment) of two audio streams on a very high resolution level, which of course is essential for the purposes of performance analysis. Onset information can be extracted from STMSP features of an audio stream by applying the following observation: For many instruments, sound generation is characterized by a sudden increase in energy, followed by a gradual decline (also called *attack* and *decay* phases of the sound). This sudden energy increase is especially pronounced in the pitch band corresponding to the fundamental frequency of the sound (and, to a lesser extent, in its first few harmonics). Hence, measuring increases in energy in a specific pitch band may give pointers to locations where a note corresponding to this pitch is played. This measurement can be performed comparatively easily by computing the first-order difference of consecutive entries of the STMSP curve x defined by $x'(n) := x(n) - x(n-1), n \in \mathbb{Z}$. This difference is then *half wave rectified*, a process that essentially “cuts away” negative values of the curve and sets them to zero, leaving only positive values for further processing. Finding *peaks* (local maxima) in this so-called *onset signal* then gives good indicators to locations of potential note onsets.

In the case of MIDI data, note onsets can be extracted trivially by direct examination of the data stream, without needing to resort to STMSP features. We will exploit this fact later on to arrive at very precise onset measurements that can be used to facilitate analysis of a single specific performance of a piece.

2.3 Dynamic Time Warping (DTW)

In the preceding section, we encountered some features which allow us to make musically meaningful assertions about a piece after their extraction. Comparing two feature sets extracted from different interpretations of the same piece allows us to make an additional observation: We can see how the different performances are interrelated and try to find out the main similarities and differences between them. When focusing the comparison on a specific section of the piece, however, one needs to be careful: There is no absolute timing reference that can be used to locate such a section in a performance, hence it is possible that one artist arrives at a specific section at e.g. 240 seconds into the interpretation, while another artist may need 270 seconds to get to the same section. The process of computing an association time frame that can be used to locate semantically equivalent sections of two pieces is referred to as *alignment* of the pieces. This can be done accurately and efficiently using a technique called *Dynamic Time Warping* (DTW), which is well-known and widely used. This section presents this technique; in the next section, a discussion follows on its relation to performance analysis.

Stated in general terms, the objective of DTW is to align two time-dependent sequences $X := (x_1, x_2, ..., x_N)$ of length $N \in \mathbb{N}$ and $Y := (y_1, y_2, ..., y_M)$ of length $M \in \mathbb{N}$. The content of these sequences consists of equidistantly sampled features taken from some fixed *feature space* $\mathcal{F}$. To measure the similarity of two features, a *local cost measure* (or *local distance measure*) c is employed, with $c : \mathcal{F} \times \mathcal{F} \rightarrow \mathbb{R}_{\geq 0}$. This distance measure should be small (indicating a low cost) when comparing two similar features, and high in the opposite case. To obtain an optimal alignment, one has to compare all feature pairs. Storing the results of these comparisons in a *cost matrix* $C \in \mathbb{R}^{N \times M}$ with $C(n,m) := c(x_n, y_m)$, one can imagine the optimal alignment as a path running from lower left to upper right of the matrix along

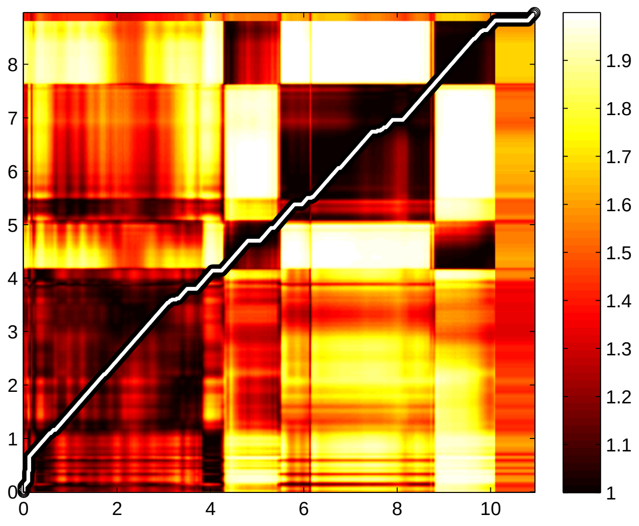

Figure 2.10: Cost matrix and warping path for the two **PathBeg** versions of Figure 2.9, time in seconds

Vector	Stored Features													
X	a		b	c_1	c_2	d_1	d_2	e	f	g		h		
Y	a_1	a_2	b	c		d		e	f	g_1	g_2	h_1	h_2	h_3

Table 2.1: Two feature vectors with semantically associated frames grouped together

Reference Vector	Warping Path Assignments													
X	1	1	2	3	4	5	6	7	8	9	9	10	10	10
Y	1	2	3	4	4	5	5	6	7	8	9	10	11	12

Table 2.2: An optimal warping path for table 2.1

the "valley" of minimal cost (see Figure 2.10).

A formalization of the concept of an alignment yields the following definition: A *warping path* of length $L \in \mathbb{N}$ between two sequences X and Y of length N and M, respectively, is a sequence $p = (p_1, ..., p_L)$ with $p_l = (n_l, m_l) \in [1 : N] \times [1 : M]$ for $l \in [1 : L]$ which satisfies the following conditions:[10]

i) *Boundary condition*: $p_1 = (1, 1)$ and $p_L = (N, M)$.

ii) *Step size condition*: $\forall\, l \in [1 : L - 1] : p_{l+1} - p_l \in \{(1, 0), (0, 1), (1, 1)\}$.

Intuitively, the warping path defines a mapping between the two feature sequences such that features x_{n_l} and y_{m_l} correspond to the same semantic unit. Table 2.2 illustrates how such a mapping might look like for two sequences X and Y as defined in table 2.1. In this sample case, $\mathcal{F} = \{a, b, c, d, e, f, g, h\}$, $N = 10$ and $M = 12$. It is easy to see from the table that the first element of X corresponds to the first two elements of Y, and so on. We will sometimes refer to the entries of the warping path by their association to a specific sequence, for instance the elements $n_l,\ l \in [1 : L]$ might be referred to as "X entries" of the warping path. Likewise, the row containing such entries may be referred to as "X row" of the warping path.

The conditions ensure that the mapping is complete in the sense that no element of either X or Y is neglected. Note that the step size condition implies monotonicity of the sequences $n_1, n_2, ..., n_L$ and $m_1, m_2, ..., m_L$, a fact sometimes made explicit in a separately stated *monotonicity condition*.

The total cost $c_p(X, Y)$ of a warping path p between X and Y using cost measure c is defined as $c_p(X, Y) := \sum_{l=1}^{L} c(x_{n_l}, y_{m_l})$. The path having minimal total cost over all possible warping paths is called *optimal warping path p**. The *DTW distance* between X and Y is then defined by $DTW(X, Y) := c_{p*}(X, Y)$.

Computation of the Warping Path

A naïve implementation of a DTW computation algorithm might simply compute all possible warping paths between X and Y and then pick the one with minimal total cost. However, this

[10]We define the shorthand $[a : b] := \{a, ..., b\}$ for $a, b \in \mathbb{N}$.

approach takes time exponential in N and M, so it is computationally infeasible. One can do better by observing that DTW exhibits the properties of *overlapping subproblems* and *optimal substructure* (since every globally optimal warping path must necessarily be locally optimal), making it well suited for a *dynamic programming* approach with a time complexity of $\mathcal{O}(NM)$. For the implementation of such an approach, we define an $N \times M$ matrix D called *accumulated cost matrix* as follows: $D(n,m) := DTW((x_1,...,x_n),(y_1,...,y_m))$, i.e. every *cell* of this matrix contains the cost of a "partial" warping path between some prefixes of X and Y.

D satisfies the following identities:

$$\begin{aligned}
D(n,1) &= \sum_{k=1}^{n} c(x_k, y_1) \text{ for } n \in [1:N] \\
D(1,m) &= \sum_{k=1}^{m} c(x_1, y_m) \text{ for } m \in [1:M] \\
D(n,m) &= \min\{D(n-1,m-1),\ D(n-1,m),\ D(n,m-1)\} + c(x_m, y_m) \\
&\quad \text{for } 1 < n \le N \text{ and } 1 < m \le M
\end{aligned}$$

Hence, $DTW(X,Y) = D(N,M)$.[11] This implies that D can be computed recursively, starting at the upper right of the matrix at location (N,M) and working downward in a stepwise manner till one arrives at the lower left $(1,1)$, the base case. Reversing the process to work iteratively in a bottom-up fashion from $(1,1)$ towards (N,M), one can cut down on memory space requirements while preserving the computation complexity of $\mathcal{O}(NM)$.

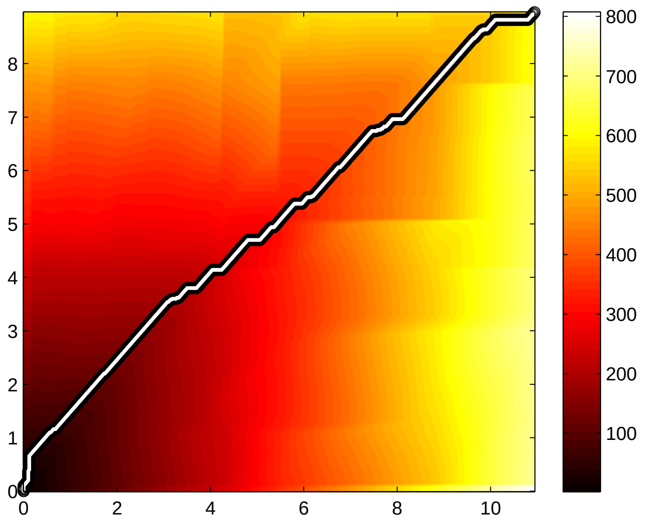

Figure 2.11: Accumulated cost matrix for the two **PathBeg** versions of Figure 2.9, time in seconds

Using D, the optimal warping path $p* = (p_1,..,p_L)$ can be computed by tracing the lowest cost path between $(1,1)$ and (N,M). Here we set $p_1 := (1,1)$, $p_L := (N,M)$ and in case

[11]See [Mül07] for proofs and a detailed discussion.

$p_l = (n, m)$ for $l \in [2 : L]$, then

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

See Figure 2.11 for a visualization of the basic idea. There is a large variety of possible optimizations for the computation of the warping path, and we have barely begun to scratch the surface. A more in-depth discussion is given in [Mül07].

2.4 DTW in Music Synchronization

To make use of the DTW algorithm for the purpose of *music synchronization*, one just needs to define a suitable feature space $\mathcal{F}$ and cost measure c. Perhaps not surprisingly, we choose the CENS features introduced in Section 2.2.4, so $\mathcal{F} = \{v \in [0,1]^{12} \mid \|v\|_2 = 1\}$. For the cost measure, we define c_α by $c_\alpha := 1 - \langle x, y \rangle + \alpha$ for some offset $\alpha \in \mathbb{R}_{\geq 0}$. The offset is necessary to account for areas of little harmonic change where the CENS features cannot accurately distinguish "good" and "bad" paths any longer. Without the offset, the path's behavior in these cases would be essentially random, since movement inside such a region does not incur high costs—including it predisposes the warping path towards the geometric optimum, which is the most reasonable alternative in such a case. Since the CENS features are normalized with regard to the Euclidean norm, $\langle x, y \rangle$ is equivalent to the cosine of the angle between x and y.

While the design described above is sufficient for a good intuition of a harmony-based alignment procedure, of course such a method would not yield very convincing results for our goal of extracting detailed timing information from the warping path. In practice, the harmony-based alignment is used as a basis for further refinements, for example using onset features as introduced in 2.2.5. Recently, this has been done by Ewert, Müller and Grosche [EMG09]. Their approach integrates chroma-based and onset features on the cost matrix level and affords good alignment accuracy while preserving the robustness gained by the use of chroma-based features. In the remainder of this work, we will suppose synchronization data generated by such an algorithm, without going into the specifics of warping path refinement necessary to obtain higher-quality alignments.

In the preceding sections, the presentation has focused on the alignment of audio/audio pairs, since this is perhaps the most challenging problem. Alignment of MIDI/MIDI or MIDI/audio pairs can be performed by using the same procedure as described before, except that STMSP and onset features can be extracted from the MIDI directly since they are represented in a symbolic manner. For the first, one only has to read the relevant parameters from the respective note onset/offset messages and convert them to a suitable STMSP representation. Onset features are even easier to extract, since they correspond exactly to the note onset timings already present in said messages. Finally, the alignment of score/MIDI or score/audio pairs can be reduced to MIDI/MIDI or MIDI/audio alignments by generating a standard MIDI file from the score. This functionality is provided by default for all the toolkits that deal with the processing of digital score data, so this does not pose any technical problems.[12]

[12]Even though some notational ambiguities have to be resolved, see Section 2.1.1.

Chapter 3

Tempo Curves

The notes I handle no better than many pianists. But the pauses between the notes—ah, that is where the art resides.

–Arthur Schnabel, 1958

The previous chapter introduced the technical infrastructure necessary for the synchronization of two interpretations of a piece of music. This chapter builds on that material buy demonstrating how to use alignment information for the computation of tempo information. However, it will first be necessary to discuss what exactly is meant by "tempo information".

The chapter begins with the introduction of some of the different hierarchies (or *levels*) that determine the structural layout of a piece. We then show how these levels form the basis of a formal definition of the tempo and associated tempo curve of a piece, and discuss three algorithms for its computation using alignment data as input. This forms the presentation of the principal conceptual contribution of the work. Finally, we briefly touch on related topics such as dynamics curves.

3.1 Measuring Tempo

Listening to a piece of music is always a subjective experience, and no two persons have precisely the same thoughts or emotions when witnessing a specific performance of a piece. Even so, there are certain characteristics of a musical piece that transcend subjectivity and can be claimed to be universal, and among the most important of those is its tempo. The feeling of pulse and rhythm is one of the central defining characteristics of nearly all Western music up to (and mostly including) the 20^{th} century, and thus measuring the tempo of a piece as accurately as possible is an obvious goal of (automated) music processing.

Conceptually, a tempo curve is the natural result of such a measurement process; it plots the tempo of a piece over the time span in which it is played (Fig. 3.1). Implementing this idea, however, is harder than it first appears. Since "tempo" is hard to define in absolute terms, one has to find a proper reference against which to measure deviations (e.g. to determine a piece's tempo in BPM, one could use the beat indicated by the piece's time signature as a reference). Even the very process of measurement is not as well-defined as one would wish: Which musical properties characterize the tempo, and exactly how precisely can they be measured before getting drowned in statistical noise? In the following, we will explore some of the possibilities in an attempt to answer these questions.

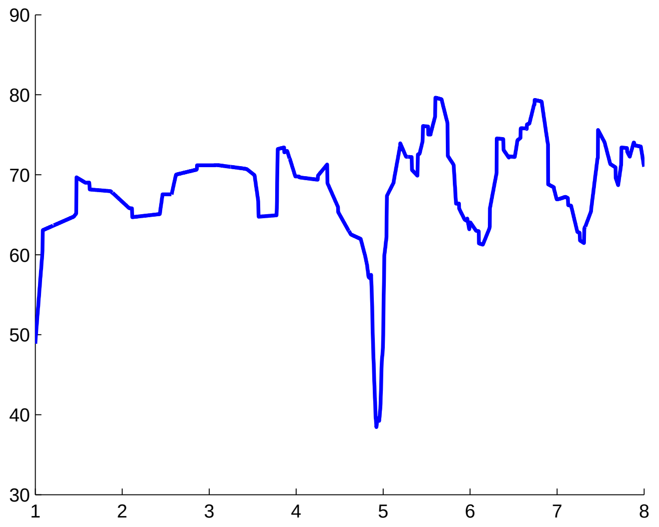

Figure 3.1: A sample tempo curve for the first seven measures of **PathExp**, time in measures and tempo in BPM. Phrase structuring and a temporary slow-down at a particularly challenging passage in measure four are clearly visible.

3.1.1 Metrical Hierarchies in Tempo

Unlike visual media such as paintings, music is an inherently sequential art form. It cannot exist outside a temporal reference frame, hence the relation between the development of musical concepts on the one hand and the passage of time on the other hand is one of the main expressive parameters which composer and performer manipulate for artistic purposes. However, since it is quite easy to "get lost" in a completely free stream of time, Western classical music regularly employs rigid structures for the organization and sectioning of musical episodes in time, which are observed and typically emphasized by the artist in a musical performance [Cla87]. We have already seen the basic organizational structure called a *measure* (or *bar*) in Section 2.1.1. There are other divisions possible on several levels that highlight different musical entities, as can be seen in Fig. 3.2. Accordingly, different division levels may serve as the foundation of different tempo levels that can conceivably be measured.

Beat level. The beat forms the most basic building block of larger periodic structures such as individual measures (inversely, it can also be regarded as a refinement of the sectioning imposed on the piece by the bar structure). The beat provides a steady and regular pulse as indicated by the time signature of the piece, e.g. for a piece in 3/4 time, there will always be three beats (of quarter note length) per bar. Finer subdivisions based on the beat are referred to as beat level (or *mensural level*) structures as well, e.g. divisions on the level of eighth or sixteenth notes in a quarter beat context. If the tempo of a piece does not change over time, sectioning on this level can be done by determining the length of one beat and slicing the piece into time segments of that length. Here, the main advantage of a beat level division is periodicity—each time segment can be relied on to have the same length. This is also its main disadvantage: If the time segment is too large, valuable information may be lost in the tempo measurement.

Note level. The note level division separates single consecutive notes from each other, regardless of their respective length. A division on this level has the advantage of automatic adaptability to the "best" resolution available to capture a musical segment, although this comes at the cost of lost periodicity—while the beat is guaranteed to occur at well-known intervals throughout a piece (barring time signature changes), for the note level division there is no such assurance. Due to the adaptive resolution, tempo measurements done on the note level are usually more sensitive than measurements done on the beat level. This means that they are better at spotting subtle tempo nuances while at the same time being more susceptible to measurement errors or processing artifacts.

Motif level. A division on the motif level makes great musical sense, although it relies on the assumption that motives exist in the piece and can be readily obtained either by analysis or prior knowledge (which is not the case in general).[13] However, if such a division is possible, timing information based on it may reveal a great deal of information about either the piece or the performing artist.

Phrase level. The same considerations as for the motif level hold for the phrase level, they merely differ in their respective structural level—phrases obviously belong to a more global structural context than simple motives.

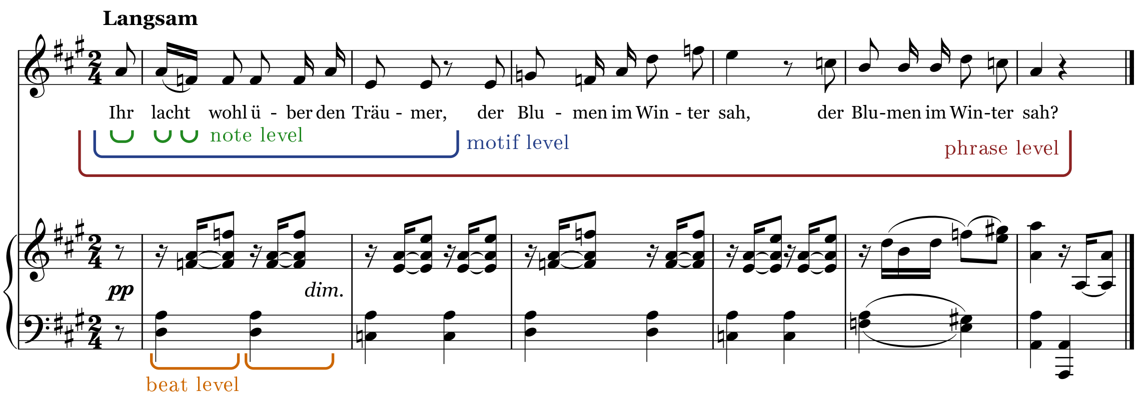
Langsam
Ihr lacht wohl ü - ber den Träu - mer, der Blu - men im Win - ter sah, der Blu-men im Win-ter sah?
note level
motif level
phrase level
pp
dim.
beat level

Figure 3.2: Franz Schubert: Winterreise D911, "Frühlingstraum" (Excerpt)

The differences between the structural hierarchies already point to a related issue, which concerns the contrast between *global tempo*, *local tempo* and *local timing*. Figure 3.3 illustrates how these are laid out with respect to each other: The global tempo refers to the complex layout of a piece as envisioned by the composer and typically spans a wide number of bars (though it still very much depends on the performer to realize the composer's ideas). In contrast, the term "local timing" is used to describe tempo variations on a very small scale (in the order of at most a few hundred milliseconds). We are mainly interested in the local tempo, which concerns small-scale tempo variations whose range typically encompasses a small number of individual notes at the most. These variations can be due to various factors, including artistic phrase shaping and realization of composer instructions such as fermatas [FGW08].

[13] As an additional difficulty, definitions of motives and phrases are always subjective and thus open for debate.

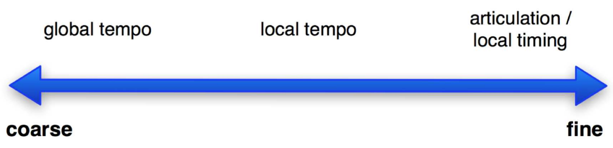

Figure 3.3: Different resolution levels of tempo measurement

From a performance analysis viewpoint, tempo and timing measures serve different purposes: Information about the global tempo of a piece is much more useful for assertions about the piece itself (i.e., the composition) than the timing measurement, although it may also be used to analyze a specific performance. On the other hand, a performer's artistic timing is strictly tied to a specific interpretation and thus best suited for analysis tasks that aim to compare and relate a number of different performances (possibly of the same piece). Here, the focus is mainly on gaining information about the performers' idiosyncratic playing styles. The local tempo lies somewhere in the middle ground and can thus be used for both of these purposes.

In the following, any unqualified use of the term "tempo" should be taken to refer to the local tempo, which is the main focus of this work. When necessary, the term "global tempo" will be used explicitly to distinguish the two.

3.1.2 A Reference Frame for Tempo

As has been already mentioned, the tempo of a piece does not exist in an "absolute" space, it always needs a reference frame to be meaningful (i.e., the piece must always get slower, faster or stay the same in relation to something else). The implicit frame of reference for a human listener is usually an idealized beat produced by his imagination—he picks up on the pulse of a piece, and then (based on his musical intuition) extrapolates from it to estimate the onset times of the next couple of notes. If these notes arrive at an earlier or later point in time than expected, the listener notices a change in tempo. While we may take this frame of reference for granted since it corresponds so closely to our subjective experience, it is in fact somewhat arbitrary: What if the composer indicated in the score that he wanted the piece to slow down, yet the performer maintained a steady tempo? An unsuspecting listener would not notice anything unusual (unless he was familiar with a faithful recording of the piece), but if one chose the composer's given tempo as a basis for judging a performance, the steadily held tempo would register as a speed-up.

One might argue that the BPM measure offers absolute information about a piece's tempo, but even this is not completely true: There are a multitude of different possible ways to notate a piece of music that can be argued to be essentially equivalent (e.g. using 6/8 time instead of 3/4 time), but would produce different BPM results—after all, the 6/8 time signature implies double the number of beats as 3/4 time, even though the duration of a single such beat is then only half as long. This means that BPM information becomes meaningful only in the context of a known time signature that can act as a reference frame.

Given this understanding, the first thing one needs to do in measuring the tempo of a piece is to pick a sensible frame of reference. A natural choice for this is a fixed number of BPM tied to a specific time signature as described above, since this corresponds closely to the regular listening experience. To avoid problems where different reference frames (e.g. composer's tempo indication vs. listener's expectation) would yield different results for the tempo curve, we assume that the only tempo changes in our input data are of a local nature; that is, we do not permit that time signature changes or novel tempo markings occur in the passage we are analyzing. This assumption is not overly restricting since such markings normally indicate structural changes as well, e.g. the beginning of a whole new section. In such a case, it is reasonable from a musical standpoint to analyze that segment separately.

The concrete data used as a reference is a Standard MIDI File that is produced directly from the score. Its tempo is fixed by obtaining the composer's tempo marking of the relevant passage, which is converted to a specific BPM value. As discussed in Section 2.1.1, this is an inexact science: Tempo is usually indicated with a deliberately loosely defined term that is open for interpretation. This ambiguity must be resolved at the time the MIDI is produced. For this, we simply pick a likely tempo from the possible range of options. Since we are interested in relative values rather than absolute ones, we accept that this may slightly skew the concrete computed values upwards or downwards—the only relevant concern is that the shape of the tempo curve be preserved, which is the case here. Note that even though this is not a primary goal of our work, we may still compute accurate absolute BPM values if the MIDI is set to a precise reference tempo. We refer to the generated MIDI file as the *reference*; the musical interpretation we want to analyze is called either *performance* or simply *interpretation*.

3.1.3 Extraction Methods of Tempo Features

After the reference has been established, we still have to measure the tempo of the actual performance. This process can be split into two steps: The extraction of certain features from reference and interpretation, and the comparison of these features. The features used for this closely relate to the various levels discussed in the preceding section—in fact, for tempo curves it is sufficient to use onset features that capture the point in time when an event on such a level happens. As an example, features on the note level consist of note onset times as presented in Section 2.2.5. There are several different ways of obtaining these features:

Automatic annotation. Trying to automate the process of feature extraction is an obvious (but challenging) idea in performance analysis. To date, no algorithm is known to produce results which are as accurate as can be achieved by manual annotation, although error margins may be small enough for certain applications [Dix01, Dix07]. This work tries to slightly improve this state.

Manual annotation. Human intervention is the most labor-intensive way of collecting features, but also among the most accurate. It is usually done on the beat level, often using a special-purpose tool (e.g. the *Sonic Visualiser* [Son09]) that displays and plays the waveform and lets the user graphically place the onsets in this representation. One useful course of action is to take the output of an automated analysis and adjust it manually to the desired degree of accuracy, thereby minimizing required human effort while maintaining high data quality. In previous work, this method has enjoyed heightened attention because results produced by other approaches were often not satisfactory [Wid02, WDG$^+$03, Sap07, Sap08].

"Direct" annotation. By use of specialized equipment, one can capture onset times during the actual recording of a piece of music. One example of this is the so-called *player piano*, a computer-monitored piano that generates symbolic (MIDI) data when it is played. Such data can be used as a basis for manual annotations, e.g. [Wid02, WDG$^+$03]. The advantage of this approach is that it produces the best data that can be gained (since symbolic onsets correlate perfectly to physical onset times), the obvious disadvantage is that special-purpose hardware needs to be used during the recording of the piece. In particular, there is no way of adapting this approach to work on existing recordings, so the huge amount of data available e.g. on CD recordings cannot be analyzed using this approach.

It is worth pointing out that the method used for obtaining tempo features is essentially irrelevant with regard to the performance analysis steps that follow feature extraction. This means that an approach that operates on beat-level tempo features will take as input any such feature set, regardless of whether it was produced manually or automatically. Hence, one can choose the best feature extraction method available for the development of such an algorithm—even if obtaining such a feature set is not feasible for regular usage, the algorithm will work just fine with differently generated features (provided that they meet reasonably high quality standards).

The alignment process described in Section 2.4 can be regarded as an automated annotation of the interpretation by the data given in the reference file. This includes note onsets, offset and possibly dynamics information—even song lyrics may be incorporated in the reference MIDI.

3.1.4 Tempo Feature Comparison

The last step in tempo measurement is actually the easiest. After the required features have been extracted, measuring the tempo of the piece amounts to a straightforward correlation of these features and measurement of the difference in onset time between interpretation and reference. For example, consider a short piece of music where only four note onsets occur (Table 3.1). Each onset is designated by an individual letter corresponding to a musical note, length of the notes is indicated by splitting single letters into multiple versions of the same letter, distinguished by their indices.

In the reference, the onsets occur at times 1, 2, 3 and 4 (given by their respective index into the data sequence) while in the interpretation, the onsets occur at times 1, 3, 5 and 7. We can observe that the time difference between two consecutive onsets is constant in the two versions, and that it is 1 in the reference and 2 in the interpretation. Thus, the translation factor between reference and interpretation is $\frac{1}{2}$, and we can conclude that the interpretation is played half as fast as the reference. This forms the basis of our understanding of the term *tempo*: It is the progression factor of time units in the reference vs. the progression factor of time unit in the interpretation. By using a known BPM value for the progression of time units in the reference (where a time unit is defined as the duration of one beat), we are then able to compute absolute BPM values for the tempo of the performance as well.

3.2 Warping Path Based Tempo Curves

We are now ready to introduce the approach taken in this thesis. Its basic idea is to make use of the fact that an alignment between reference and actual performance—i.e., a warping path—can be regarded as a description of the performance's tempo structure. This facilitates analysis of the warping path to build a tempo curve from the information it contains. To see how this is done, consider again Table 3.1. An optimal warping path for this example is depicted in Table 3.2.

Intuitively, one can "read off" the ratio of time progression in the reference vs. time progression in the interpretation by looking at the respective length of semantically corresponding music segments in the two data streams: In this example, each progress by one time unit in the reference corresponds to a progression by two time units in the interpretation. This is reflected in the warping path by the fact that each individual index into the reference must occur twice to fit its tempo to the tempo of the interpretation. Hence, we can again conclude that the translation factor between reference and interpretation is $\frac{1}{2}$, i.e. half-tempo.

Notice that in the plot of this warping path (Fig. 3.4), the *gradient* of the idealized warping path (which is gained by averaging over the values of the actual warping path) is precisely two—the inverse of the translation factor. We will now formalize this intuitive understanding.

3.2.1 Sliding Window Computation of Tempo Curves

Let $p = (p_1, ..., p_L)$ be a warping path of length L between two sequences X and Y of length N and M as defined in Section 2.3, where X is the reference of a given piece of music and Y its respective interpretation. We define the *extended warping path* p^{ext} for all $l \in \mathbb{Z}$ by padding the regular warping path p at the locations where it was not explicitly computed, using the assumption that reference and interpretation have equivalent tempo there:

$$p_l^{\text{ext}} := \begin{cases} p_l = (n_l, m_l) & \text{if } l \in [1 : L] \\ (l, l) & \text{if } l < 1 \\ (N + l - L, M + l - L) & \text{if } l > L \end{cases}$$

For the following definitions, we will assume that the warping path is always padded like this to avoid special treatment of "boundary cases".

The computation of the tempo curve basically works by looking at each element of the reference sequence, determining the length of the entries of the warping path semantically corresponding to this element both in the reference row and the interpretation row, and computing the quotient between both of these lengths. However, such an element-wise computation is extremely unstable in terms of robustness against alignment errors and artifacts. Therefore, we also introduce an averaging window of size w that defines a broader range of elements of the reference that are included in this examination. The tempo curve is then determined not by finding semantically corresponding entries to a specific element $x \in X$, but rather to a range of w such elements centered around x. We formalize this by looking at the computation of one particular entry of the tempo curve in detail.

Reference	e	c	d	g				
Interpretation	e_1	e_2	c_1	c_2	d_1	d_2	g_1	g_2

Table 3.1: Two schematic pieces of music, note onsets marked red

Associated Vector	Warping Path Assignments							
Reference	1	1	2	2	3	3	4	4
Interpretation	1	2	3	4	5	6	7	8

Table 3.2: An optimal warping path for Table 3.1

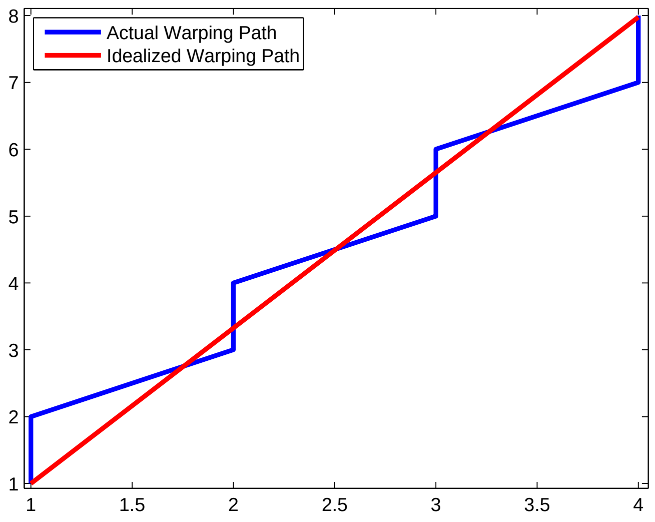

Figure 3.4: Visualization of the warping path of Table 3.2 (notice different scaling for horizontal reference and vertical interpretation, time given in indices into Table 3.1)

Let $n \in [1 : N]$ be an arbitrary but fixed index into the reference data sequence X and $w \in \mathbb{N}_{>0}$ the size of an averaging window. The *tempo curve* induced by the warping path p with respect to w is a function $\tau : [1 : N] \to \mathbb{R}_{\geq 0}$ that is defined by

$$\tau_w(n) \;:=\; \frac{(b+1)-a}{(m_{l_b}+1)-m_{l_a}}$$

Here, l_a and l_b are computed according to the following definitions:

$$a := n - \left\lfloor \frac{w-1}{2} \right\rfloor$$

$$b := n + \left\lceil \frac{w-1}{2} \right\rceil$$

$$l_a := \max\{l \in \mathbb{Z} \mid n_l = a\}$$

$$l_b := \max\{l \in \mathbb{Z} \mid n_l = b\}$$

This means that a and b define the indices of the outer limits of the window w in the reference sequence, i.e. the elements of the reference sequence included in this examination step are given by the range $[x_a : x_b]$. The middle element of this range is always x_n. Although some of the elements of this range may be non-existent (for $a < 1$ or $b > N$), their respective indices are still included in the extended warping path. Hence, τ is well-defined on the full domain $[1 : N]$ for arbitrary window sizes. Just as a and b define the indices of window limits in the reference sequence X, l_a and l_b represent the indices of window limits in the extended warping path p_l^{ext}. In the case of ambiguities (when a or b occur multiple times in the reference row of the warping path), their definition is designed to always pick the largest possible index still denoting an occurrence of a or b, respectively. Other possible choices would have included the smallest such index, or the index of the middle element of the respective range. Since the impact of this choice is negligible with respect to overall accuracy, the only important point here is that for any ambiguous case, exactly one index is picked that always remains the same whenever that case is evaluated. This is the case with the maximum used in this definition.

Note that the tempo curve is defined in terms of the reference sequence, so the "resolution" of this sequence (i.e. the length of the features that make up its individual elements) determines the resolution (or precision) of the tempo curve as well. Since $w = (b + 1) - a$, we can also formulate the definition of τ as follows:

$$\tau_w(n) \;:=\; \frac{w}{(m_{l_b} + 1) - m_{l_a}}$$

In $b + 1$ and $m_{l_b} + 1$, the addition of "one" is necessary to account for the last element of the sequence we are inspecting which would not be counted otherwise—e.g., even if $a = b$ and $m_{l_a} = m_{l_b}$, we are still examining exactly one element of reference sequence and warping path.

Consider an example: We will evaluate the warping path depicted in Fig. 3.4 at $n = 3$ with a window size $w = 3$. In this case, $a = 2$ and $b = 4$. As $\{l \in [1 : L] \mid n_l = 2\} = \{3, 4\}$ and $\{l \in [1 : L] \mid n_l = 4\} = \{7, 8\}$, $l_a = 4$ and $l_b = 8$, according to the definition. Notice how the ambiguous borders are resolved in both cases. Next, we evaluate $\tau_3(3) = \frac{(4+1)-2}{(8+1)-4} = \frac{3}{5}$. The difference of this result to the "ideal" tempo of $\frac{1}{2}$ is due to a general drawback of this approach: Operating on the actual warping path is in general not equivalent to working on an ideal warping path, and an important point of our contribution is to alleviate this problem by smoothing (averaging) over various values to still arrive at adequate tempo curves.

In the algorithmic computation of the tempo curve, we will iteratively "slide" the window w over all indices $n \in [1 : N]$ into the reference sequence to compute all entries of the tempo curve. Consequentially, we refer to the class of algorithms presented here as *sliding window* algorithms. The different evaluation techniques that distinguish these algorithms from each other are based on some of the different feature levels from Section 3.1.1: The first technique sections the warping path into equal-length snippets of a fixed time, correlating to the beat level. The second technique introduces uneven sectioning based on a note-level division of the music data, and the third technique tries to unify time based and note based approach.

A major advantage of the warping path based sliding window approach is that it can be used in conjunction with the DTW method presented in Section 2.3. This allows us to exploit advanced music alignment techniques to automatically generate tempo curves without the need for manual intervention. Since DTW is designed to work with a wide variety of features, we are not limited to any particular feature resolution level but can use whatever features necessary to build an accurate warping path. On the other hand, we can still decide to evaluate the warping path on any precision level necessary for a specific performance analysis purpose—one could even perform an analysis of the symbolic reference data stream to determine which timings fall on heavy beat times, melodic highlights, deceptive cadences and so on, and can use this knowledge to selectively evaluate the tempo at such points in time.

3.2.2 Fixed Window Size Warping Path Evaluation

The most straightforward approach to tempo curve computation consists of a literal algorithmic implementation of the formal description given in Section 3.2.1. The corresponding code is presented in Algorithm 3.1. It expects only the extended warping path and the window size w as input. Remember that n_l designates elements of the reference, while m_l is used for elements of the interpretation.

The algorithm slides and centers the window of size w over every element n of the reference sequence (lines 2–8), positioning the respective borders exactly as in the formal definition of the procedure. While the presentation of the algorithm in this work remains faithful to that definition, the actual implementation has a slightly different structure that avoids potentially costly operations such as the set comprehension of lines 5–6. The call to ExtendWarpingPath extends p by the number of entries needed to accommodate a window of size w during the execution of the algorithm.

Since w is fixed as a parameter to the algorithm, we call this algorithm the *fixed window* approach to tempo curve computation (FW for short). Notice that the window size is the only parameter that can be manipulated in this approach. Early experiments showed that a subsequent additional averaging over multiple entries of the tempo curve did not yield significantly better results for the FW algorithm. Figures 3.5 and 3.6 may serve as a preliminary example of the output produced by the FW algorithm before we come to a more detailed analysis in Chapter 4.

Changing the window size obviously changes the outcome of the computation: Plotted are two tempo curves generated by different parameter settings (blue) against a synthesized *ground*

Algorithm 3.1: Tempo curve computation based on the fixed window technique

```
Input: warping path $p_l = (n_l, m_l)$ $(l \in L)$, window size $w \in \mathbb{N}_{>0}$
Output: tempo curve $\tau$
1 $p \leftarrow$ ExtendWarpingpath($p$, $w$);
2 for $n \leftarrow 1$ to $N$ do
3     $a \leftarrow n - \lfloor \frac{w-1}{2} \rfloor$;
4     $b \leftarrow n + \lceil \frac{w-1}{2} \rceil$;
5     $l_a \leftarrow \max\{l \in \mathbb{Z} \mid n_l = a\}$;
6     $l_b \leftarrow \max\{l \in \mathbb{Z} \mid n_l = b\}$;
7     $\tau(n) \leftarrow \frac{w}{m_{l_b} - m_{l_a} + 1}$;
8 end
```

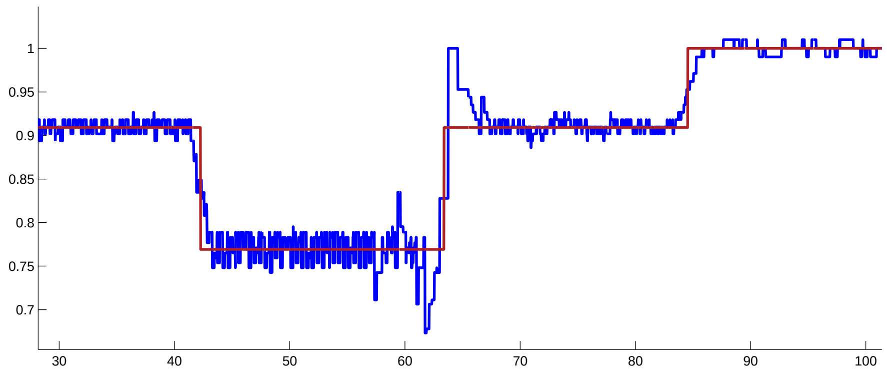

Figure 3.5: Results of the FW algorithm for $w = 2\,\mathrm{s}$, time in seconds

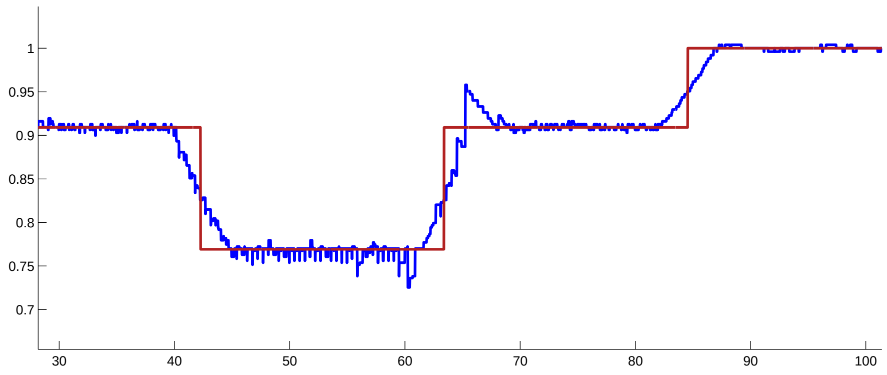

Figure 3.6: Results of the FW algorithm for $w = 5\,\mathrm{s}$, time in seconds

truth reference (which will be discussed in detail in Section 4.1, shown here in red). As per the definition of the tempo curve, tempo is given here not in BPM but in relation to a reference tempo: We start a bit slower than the reference, slow down to about 3/4 of the original speed, then speed up again until we have reached the original tempo at $t = 85\,\mathrm{s}$. Two characteristics of the algorithm are immediately apparent: Fluctuations in the generated curve become less pronounced as the window size increases, but sensitivity to changes lessens in turn (see e.g. adaptation of the curve to a new tempo at $t = 41\,\mathrm{s}$). Another visible effect is caused by the centering of the smoothing window which results in an anticipation of tempo changes even before they have started happening. Although we give the size of the window w in terms of seconds in these examples (and will continue to do so for reasons of intuitiveness), we actually mean by that a size with respect to the feature rate used in the warping path computation that corresponds to three seconds of the reference audio data.

Viewing the tempo curve at a higher resolution, one can see a plateau effect where certain values have a much higher probability of appearing in the tempo curve than other values (Fig. 3.7). Furthermore, these seem to cancel each other out in their fluctuation around the ground truth tempo curve. This phenomenon is due to the stepwise, “integer” nature of the warping

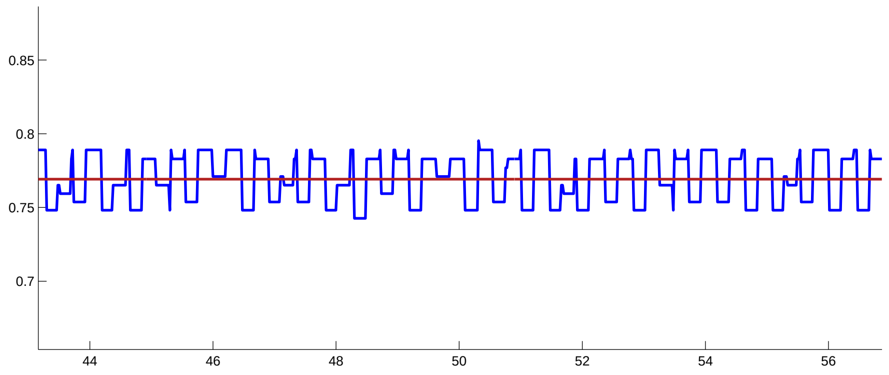

Figure 3.7: Higher-resolution view of FW algorithm results for $w = 2\,\mathrm{s}$, time in seconds

path that prescribes a maximum resolution to the computed value. We will discuss all of these aspects in greater detail in Chapter 4.

3.2.3 Adaptive Window Size Warping Path Evaluation

As already mentioned in Section 3.1.1, segmenting and sampling the warping path according to periodic intervals has the disadvantage that it does not adapt optimally to the musical attributes of the data. A division that takes the distribution of the notes into account circumvents this problem: The *adaptive window size* algorithm (AW for short, see Algorithm 3.2) is one of the possible implementations of this idea. Instead of computing the gradient between two points in time that are always w reference time units apart, it computes gradients (or *slopes*) only at the points in time between two distinct note onsets. If these onsets are consecutive, such an interval is referred to as *interonset interval* (IOI).

The basic idea behind this approach is to acknowledge that note onsets are the main source of tempo data available for performance analysis processing. This is especially true for Western classical piano music, but also for pieces with different orchestrations (note onsets may be some orders of magnitude harder to extract in such contexts, but this is an active research field [GME09]). Choosing to neglect arguably less important aspects such as note offsets or pedaling, we can claim that measuring note onset information is sufficient to reconstruct the tempo of a piece by correlating each measured onset to the respective onset in the reference as described in Section 3.1.4. Consequently, computing the gradient between note onsets tries to use the full amount of tempo information available from these onsets while discarding any alignment artifacts that occur inside the region of an IOI.

For our theoretical discussion, we need to update the computation of the borders of the averaging window. These are aligned to note onsets, which we model as a set of indices into the reference sequence $O \subseteq [1 : N]$. For a number of K onsets, we define $O := \{o_1, ..., o_K\}$, with $1 \leq o_1 < o_2 < ... < o_{K-1} < o_K \leq N$ (that is, we regard this set essentially as an ordered list). Furthermore, without loss of generality we require that $o_1 = 1$ and $o_K = N$ (if this is not the case, extend O by inserting $o_0 = 1$ and $o_{K+1} = N$). This ensures proper window alignment in the boundary cases (and also implies that $K = L$). Similar to the extended warping path, we define an extended onset list O^{ext} for all $k \in \mathbb{Z}$ as follows:

$$o_k^{\text{ext}} := \begin{cases} o_k & \text{if } k \in [1:K] \\ k & \text{if } k < 1 \\ N - K + k & \text{if } k > K \end{cases}$$

Simply put, this pads O with evenly-spaced onsets on both sides, which again enables us to extend our averaging window beyond the boundaries of 1 and N, even when it must be aligned to note onsets. We are now ready to define how the window edges are computed in the AW case.

Let $w_{\text{ioi}} \in \mathbb{N}_{>0}$ be the size of a window indicating the number of interonset intervals that should be included in an averaging step. We first examine the case where $w_{\text{ioi}} = 1$, i.e. we are averaging between two consecutive onsets. Then, for $k \in [1 : K - 1]$, we have

$$\tau_{w_{\text{ioi}}}(o_k) \;:=\; \frac{(b+1) - a}{(m_{l_b} + 1) - m_{l_a}}$$

In this case, the following definitions are used for the window edges:

$$\begin{aligned} a &:= o_k \\ b &:= o_{k+1} \\ l_a &:= \max\{l \in \mathbb{Z} \mid n_l = a\} \\ l_b &:= \max\{l \in \mathbb{Z} \mid n_l = b\} \end{aligned}$$

Hence, this case is analogous to the FW approach—the only differences are in the location of the window edges, and the "placement" of the computed value in the tempo curve. Where the computed gradient between two locations a and b was formerly placed in their arithmetic middle $n = a + \lfloor \frac{b-a}{2} \rfloor$ due to the centered window, it is now placed simply at $o_k = a$, the location of the first onset.

This only defines the tempo for locations where onsets are present, so we have to interpolate the tempo at places where this is not the case. To do this, first of all set $\tau(K) := 1$. Now we can perform simple linear interpolation between all known onset locations: Let $n \in [1 : N] \setminus O$ be an arbitrary but fixed location where the tempo curve is not yet defined, and $k \in [1 : K-1]$ the index of the onset which immediately precedes it, i.e. $o_k < n < o_{k+1}$. Then,

$$\tau_{w_{\text{ioi}}}(n) \;:=\; \tau(o_k) + \frac{\tau(o_{k+1}) - \tau(o_k)}{o_{k+1} - o_k} \cdot (n - o_k)$$

Let us now examine the case where $w_{\text{ioi}} > 1$. Actually, we shall see that the definition is general enough to accommodate $w_{\text{ioi}} = 1$ as well, so let $w_{\text{ioi}} \in \mathbb{N}_{>0}$ in the following. The indices of the window edges are then given by this definition:

$$\begin{aligned} c &:= k - \left\lfloor \frac{w_{\text{ioi}}}{2} \right\rfloor \\ d &:= k + \left\lceil \frac{w_{\text{ioi}}}{2} \right\rceil \\ a &:= o_c \\ b &:= o_d \end{aligned}$$

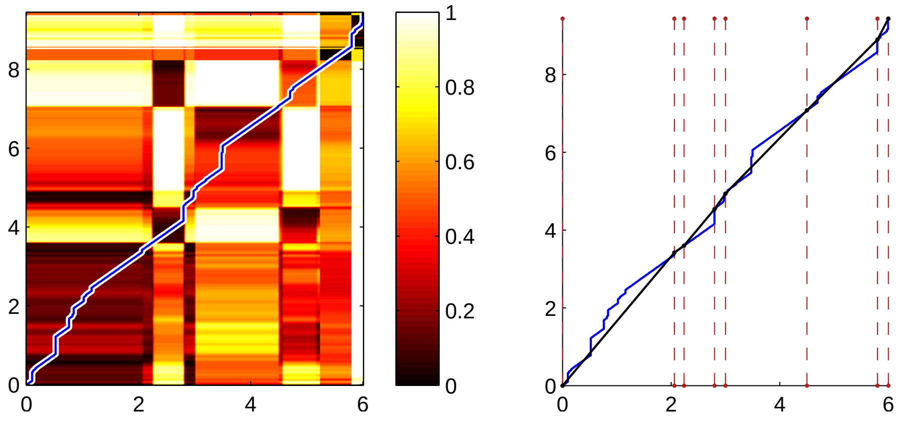

Figure 3.8: Warping path (blue) for **PathBeg** in cost matrix context and with interonset interpolation (onsets red, interpolation black), time in seconds

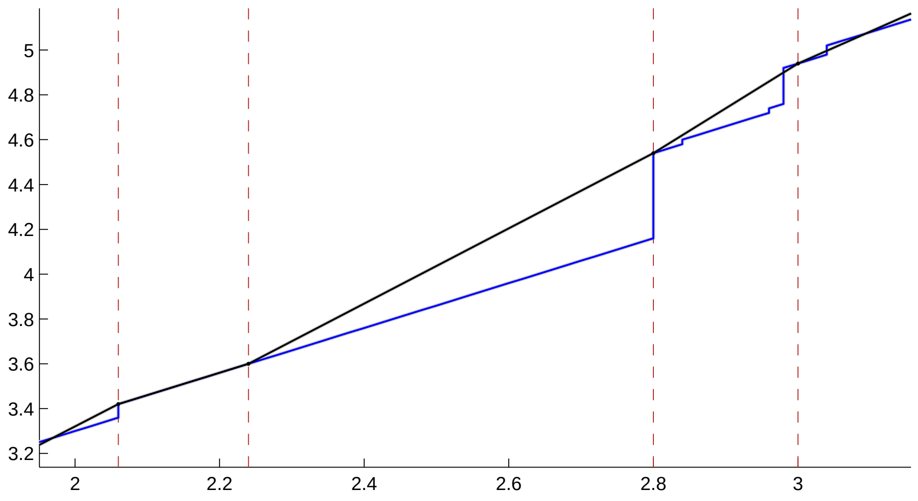

Figure 3.9: High-resolution view of the results of interonset interpolation, legend as above

Here, c and d index into the list of onsets O. All other definitions remain as above; notice that in case $w_{\mathrm{ioi}} = 1$, we have in fact $a = o_k$ and $b = o_{k+1}$. This definition can also be used to describe the FW case by setting $O := [1 : N]$. For such an O, this definition of τ becomes equivalent to the old definition of the FW case for $w = w_{\mathrm{ioi}}$.

Figure 3.9 depicts an example of the combination of warping path (blue), onset information (vertical red lines) and the idealized warping path (black) that is used by the AW approach for the computation of the tempo curve (i.e., an interonset interpolation of the actual warping path). Figure 3.8 shows how the original warping path was obtained from its respective cost matrix, demonstrating clearly the correspondence between artifacts in the warping path that need to be smoothed out and regions of harmonic stagnancy in the original piece.

Algorithm 3.2: Tempo curve computation based on the adapting window technique

```
Input: warping path $p_l = (n_l, m_l)(l \in L)$, onsets $O = \{o_1, ..., o_K\}$, window size $w_{\text{ioi}} \in \mathbb{N}_{>0}$
Output: tempo curve $\tau$
1  $p \leftarrow$ ExtendWarpingpath($p$, $w_{\text{ioi}}$);
2  $O \leftarrow$ ExtendOnsets($O$, $w_{\text{ioi}}$);
3  for $k \leftarrow 1$ to $K$ do
4      $c \leftarrow k - \lfloor \frac{w_{\text{ioi}}}{2} \rfloor$;
5      $d \leftarrow k + \lceil \frac{w_{\text{ioi}}}{2} \rceil$;
6      $a \leftarrow o_c$;
7      $b \leftarrow o_d$;
8      $l_a \leftarrow \max\{l \in \mathbb{Z} \mid n_l = a\}$;
9      $l_b \leftarrow \max\{l \in \mathbb{Z} \mid n_l = b\}$;
10     $\tau_{\text{onsets}}(o_c) \leftarrow \frac{(b+1)-a}{(m_{l_b}+1)-m_{l_a}}$;
11 end
12 $\tau \leftarrow$ Interpolate($\tau_{\text{onsets}}$, $O$);
```

The structure of the algorithmic implementation (Alg. 3.2) is not largely different from the theoretic outline. The main work is done in lines 3–11: We iterate over all onsets and compute the gradients between them, paying respect to the averaging window defined by w_{ioi}. After this step, the tempo curve τ is defined exactly at the places where an onset occurred, hence we call the intermediate result τ_{onsets}. For a definition on the full domain, we still have to interpolate the values in between onsets. This is done in line 17 by calling the auxiliary function Interpolate. This function computes a linear interpolation as described, but of course other interpolation methods could be used here as well.

In the case of the FW algorithm, smoothing of the tempo curve was done by choosing a window of larger size. This is the case here as well, but the window size can no longer be controlled directly: it is computed implicitly from the number of IOIs that are included in the averaging step (lines 4–5). This has direct implications for the impact of the averaging: Areas of the piece with high onset density are affected less than areas where only a small number of note onsets occur. As in the case of the FW algorithm, the window for this computation is centered around one specific location of the tempo curve o_k. The computed value is just the average gradient between the two onsets o_c and o_d. Keep in mind that due to different IOI lengths, the centering may be biased to one side: If $w_{\text{ioi}} = 3$, the IOI to the left of o_k (of length $o_k - o_{k-1}$) may be significantly shorter or longer than the IOI to the right of o_k (which has length $o_{k+2} - o_{k+1}$).

As a result of this, the absolute size of the averaging window w depends a lot on the tempo characteristics of the musical passage encompassed by o_c and o_d. Generally speaking, fast and complex passages with small note lengths will cause it to shrink, while slow and simple passages will yield a much larger window. This is the case as e.g. three half-notes will normally take a longer time to play than three sixteenth-notes, even though the number $w_{\text{ioi}} = 3$ stays fixed. The results of this unpredictability can be observed in Figures 3.10 and 3.11 which show two sample outputs of the AW algorithm. Here, in the region from $t = 30$ to $t = 70$ there are relatively few but longer notes (an average of approximately 3.4 onsets/s), while in the region from $t = 70$ to $t = 85$, there are relatively more shorter notes (approx. 9.5 onsets/s on average).

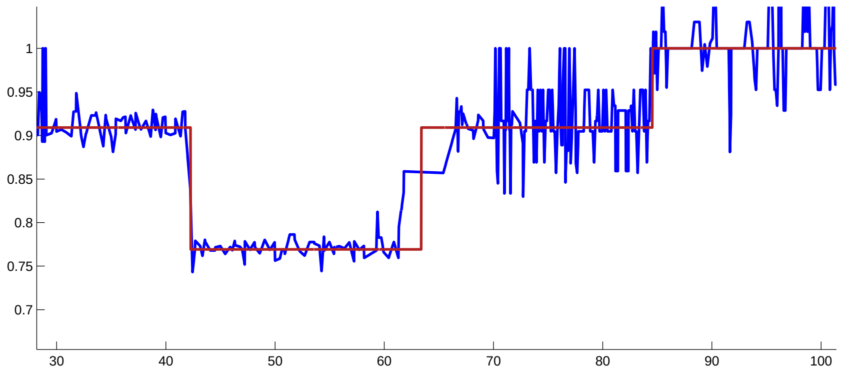

Figure 3.10: Results of the AW algorithm for $w_{\text{ioi}} = 3$, time in seconds

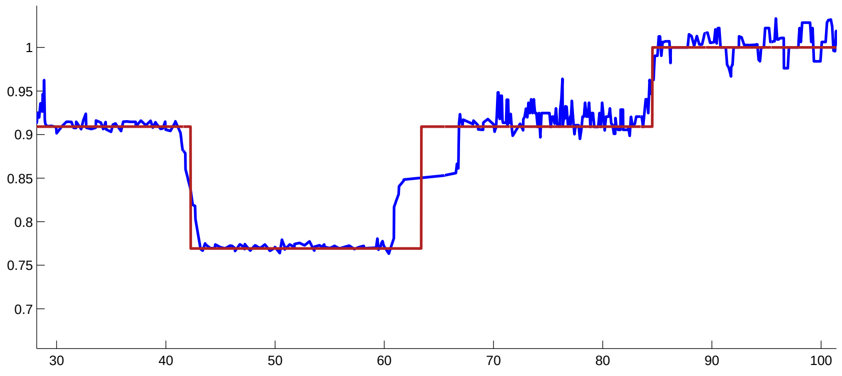

Figure 3.11: Results of the AW algorithm for $w_{\text{ioi}} = 9$, time in seconds

As a result, the smoothing is noticeably stronger in the first region than in the second one. This is especially apparent in the region of tempo change at $t = 62$—here, only one note onset occurs in a region over four seconds long, resulting in an extremely broad averaging window. Notice how in contrast, the change-over to a new tempo at $t = 41$ is processed remarkably fast.

From the difference between Figures 3.10 and 3.11, one can see the importance of the smoothing step for the AW algorithm. Jitter from alignment artifacts dominates the picture from $t = 70$ onwards in Figure 3.10. Though still far from perfect, the results become much better for a larger w_{ioi} (Fig. 3.11).

3.2.4 Fixed Window Size Evaluation on Corrected Warping Paths

We have now seen two different approaches: The basic FW algorithm just sampled the warping path at evenly spaced intervals, the AW algorithm introduced sampling at onset locations and IOI smoothing. This section presents a third approach that tries to combine the two previously discussed algorithms. The main idea of this hybrid approach (which will be referred to as FWC

Algorithm 3.3: SmoothWarpingpath, smoothing of warping path entries by onsets

```
Input: warping path $p_l = (n_l, m_l)(l \in L)$, onsets $O = \{o_1, ..., o_K\}$
Output: smoothed warping path $p$
1  for $k \leftarrow 1$ to $K$ do
2      $a \leftarrow o_k$;
3      $b \leftarrow o_{k+1}$;
4      $l_a \leftarrow \max\{l \in \mathbb{Z} \mid n_l = a\}$;
5      $l_b \leftarrow \max\{l \in \mathbb{Z} \mid n_l = b\}$;
6      $x \leftarrow \frac{m_{l_b} - m_{l_a}}{n_{l_b} - n_{l_a}}$;
7      for $i \leftarrow 0$ to $n_{l_b} - n_{l_a}$ do
8          $p_{l_a+i} \leftarrow (n_{l_a} + i,\ m_{l_a} + \mathrm{round}(x \cdot i))$
9      end
10 end
11 $p \leftarrow$ FillGaps($p$);
```

in the following, for *fixed window corrected*) is to perform FW sampling on a smoothed (or *corrected*) warping path, where smoothing is done by computing gradients between consecutive onset locations, similar to the AW approach. The implementation is quite straightforward: The warping path is re-computed analogously to the method presented in the AW approach (with $w_{\text{ioi}} = 1$ fixed since smoothing is only done inside IOI regions), and the results are exported as a new warping path (Alg. 3.3). This corrected warping path is then used as input for the FW algorithm (Alg. 3.1).

The FillGaps function called in line 11 of Algorithm 3.3 merely ensures that the step-size condition of the warping path is always met. Up to that line, this may not have been the case due to rounding in the computation of the interpolated values (line 8)—e.g., for $x = 1.4$, in step $i = 1$ the respective value in the interpolated warping path would be computed as $m_{l_a} + \mathrm{round}(1.4 \cdot 1) = m_{l_a} + 1$, but in the subsequent step $i = 2$ it would become $m_{l_a} + \mathrm{round}(1.4 \cdot 2) = m_{l_a} + 3$. Hence, the value $m_{l_a} + 2$ would be skipped, violating the step-size condition. FillGaps detects such violations and fills in the missing values.

The results produced by this algorithm are a marked improvement over both of the previous approaches: Fig. 3.12 no longer exhibits any sections dominated by alignment errors. Due to the relatively broad time window $w = 5$, the transition to a new tempo is not as quick as e.g. in the AW case. However, such sudden and severe transitions are rather unusual in the music domain we are interested in, so this effect is in fact appropriate.

3.2.5 Interpretation-scaled Tempo Curves

Until now, the time axis of a plot was always scaled with regard to the reference. This is important for the conceptual understanding of tempo curves and for the comparison of curves generated from different performances, but inconvenient when working in a real-world performance analysis context. Here, one would like to determine the tempo of a given interpretation at a specific point in time t by consulting this interpretation's tempo curve at t and using the respective value. A naïve (but working) approach would be to recompute the desired

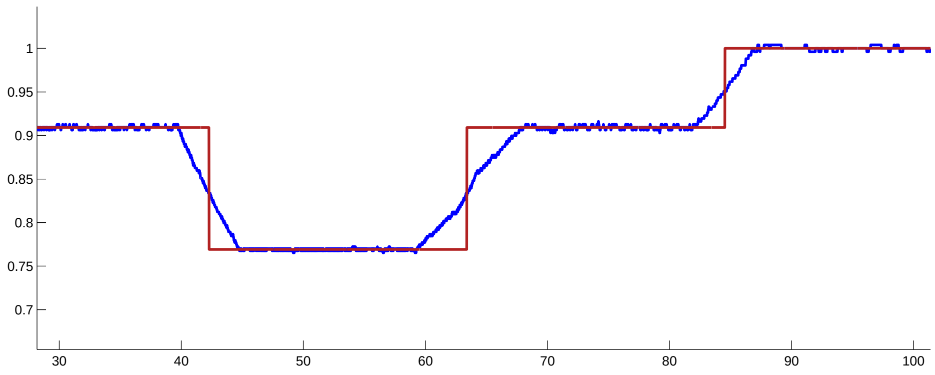

Figure 3.12: Results of the FWC algorithm for $w = 5$, time in seconds

tempo curve with reference and interpretation flipped, resulting in an “inverted” tempo curve. Maintaining the horizontal scale, this curve could then be inverted again along the vertical axis to arrive at the desired interpretation-scaled tempo curve. However, there are certain disadvantages to this approach: Onset information is guaranteed to be available and accurate for the reference, but this is not necessarily the case for the interpretation. Consequently, the AW and FWC approaches may not be used for the recomputation.

Furthermore, the computational effort required for a fresh computation of the inverted tempo curve from scratch are not strictly speaking necessary, since the existing data already contains all necessary information to construct rescaled tempo curves. The idea with this rescaling approach is simply to “warp” the tempo curve using the established warping path.

Algorithm 3.4 describes how this is done: Basically, one just has to establish evaluation points in the tempo curve and then interpolate between the values of these points according to the interpretation data stream instead of the reference data stream. Evaluation points can be chosen according to note onsets, or just be set to $[1 : N]$ for the case of FW tempo curves.

In the algorithm, the search for the relevant evaluation points of the original tempo curve is performed in line 3. The values at these points are then entered into a new curve (line 4). We keep track of the interpretation-scaled onsets by updating a set O_{rescaled} that stores this

Algorithm 3.4: RESCALE, rescaling of tempo curves to performance tempo

Input: tempo curve τ, onsets $O = \{o_1, ..., o_K\}$, warping path $p_l = (n_l, m_l)$ $(l \in L)$
Output: rescaled tempo curve τ_{rescaled}

```
1 $O_{\text{rescaled}} \leftarrow \{\}$;
2 for $k \leftarrow 1$ to $K$ do
3     $l_k \leftarrow \max\{l \in [1 : L] \mid n_l = o_k\}$;
4     $\tau_{\text{rescaled}}(m_{l_k}) \leftarrow \tau(o_k)$;
5     $O_{\text{rescaled}} \leftarrow O_{\text{rescaled}} \cup \{m_{l_k}\}$;
6 end
7 $\tau_{\text{rescaled}} \leftarrow$ INTERPOLATE($\tau_{\text{rescaled}}, O_{\text{rescaled}}$);
```

information (line 5). The algorithm as presented relies on the assumption that interpretation-scaled onsets are unique; if we define $l_i := \max\{l \in [1 : L] \mid n_l = o_i\}$, we can state this requirement as $\forall i, j \in [1 : K] : o_i \neq o_j \Rightarrow m_{l_i} \neq m_{l_j}$. However, this assumption is not necessarily met by the warping path. To implement line 4 correctly, one would therefore need to compute $\tau(m_{l_k})$ by taking the average over all $\tau(o_i)$ where $(o_i, m_{l_k}) \in p$. For didactic purposes, we have chosen to retain the simple presentation of the algorithm that is easier to digest.

Figure 3.13 shows the result of a rescaling transformation: the time scale is changed according to the length of the piece, but the tempo values are maintained. Notice that regions of the interpretation where the tempo is comparatively slow take "longer" in the rescaled tempo curve than in the original curve (e.g. at $t = 35\,\text{s}$ in the interpretation-scaled curve), and vice versa for faster passages.

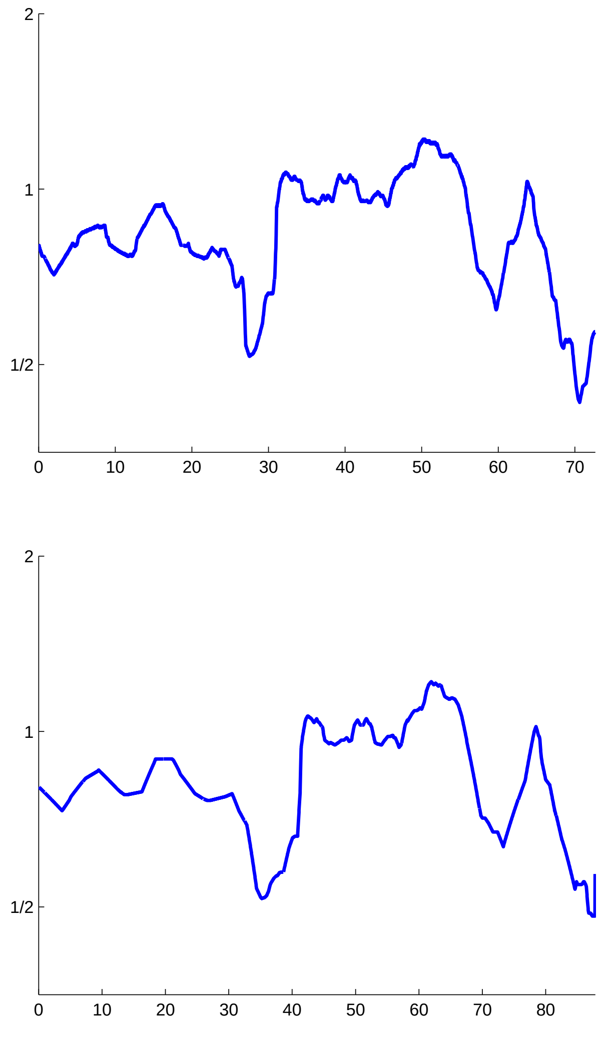

Figure 3.13: Comparing a regular tempo curve (top) against a rescaled tempo curve (bottom), time in seconds

3.3 Dynamics Curves

As mentioned before, a useful way of looking at the alignment process between reference and interpretation is to regard it as an automated annotation of the interpretation by the data provided in the reference. Such annotations facilitate extraction of multiple kinds of performance characteristics, not just the tempo. In particular, targeting levels other than the beat level for such extractions becomes feasible with the introduction of note-level annotations. Since automated annotations directly benefit from any improvements to the accuracy of the DTW alignment algorithm, algorithms build on this basis are likely to yield better results over time.

One example for how automatically generated annotations can be exploited in the computation of dynamics curves is shown in Figure 3.14. Here, the dynamics of two different interpretations of the same piece are plotted according to the time axis of the reference instead of the time axis of the performances. This is made possible by using the annotations of the performances to compute a kind of inverse rescaling of a regular dynamics curve: Where the original rescaling procedure presented in the previous section translated from a reference-scaled tempo curve to an interpretation-scaled curve, the rescaling used in this case translates from interpretation time to reference time instead. The resulting reference-scaled dynamics curves are useful in comparing multiple performances with each other, since their time axes can thus be normalized to the reference. The example shown in Figure 3.14 demonstrates that clear correlations can be seen in such a direct comparison; Section 4.2.2 gives a closer look at how such dynamics curves relate to the performance analysis process.

Information about the dynamics of a recording is computed in the following way: The STMSP features for all pitch subbands of the input signal at a specific point in time t are summed up, with the result energy_t being the energy of the whole signal at this point. The dynamics curve at point t is then defined by $\log_2(\text{energy}_t + 1)$.

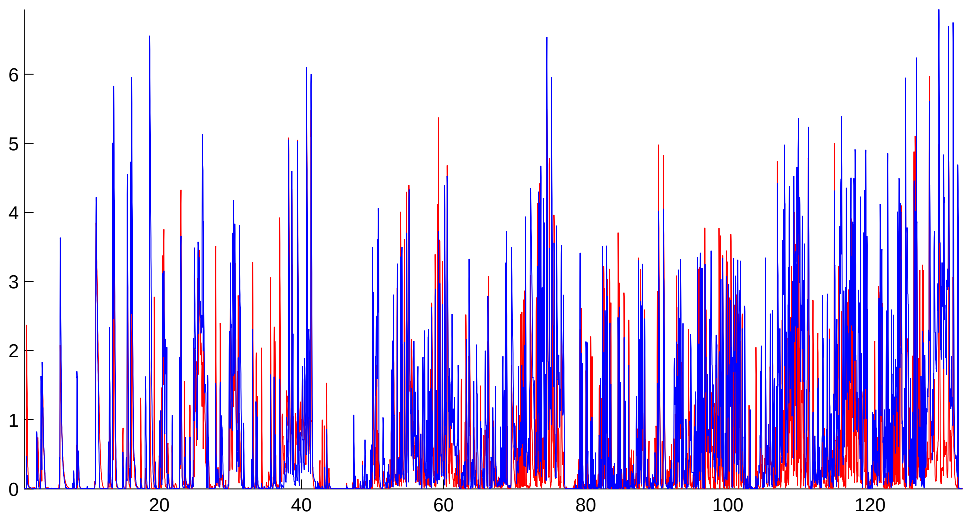

Figure 3.14: Dynamics curves for two interpretations of **PathExp**, time in measures

3.4 Chapter Summary

The present chapter introduced the main contribution of this work: Three algorithmic methods to automatically compute the tempo attributes of an expressive musical recording, under a well-grounded definition of "tempo". The remainder of this work is concerned with establishing data on the performance of these algorithms: How reliable they can be expected to be, which technique delivers the best results, how tempo information generated by these algorithms looks like and how it can be used for performance analysis.

We continue by presenting an evaluation on the techniques that incorporates both quantitative and qualitative aspects.

Chapter 4

Evaluation

The pleasure we obtain from music comes from counting, but counting unconsciously. Music is nothing but unconscious arithmetic.

–Gottfried Wilhelm Leibniz (1712), quoted in Oliver Sacks,
THE MAN WHO MISTOOK HIS WIFE FOR A HAT *(1985)*

Any discussion of a new approach for the computation of tempo curves for expressive music recordings would of course be incomplete without a proper evaluation of its effectiveness. We divide this evaluation of our approach into two distinct parts: The first part tries to *quantify* the performance of the three techniques using measurements aiming for maximum objectivity. The second part is a deliberately subjective *qualitative* analysis that uses individual examples to illustrate some aspects of the various techniques. The two parts of the evaluation are complementary to each other—taken together, they should convey a fairly complete perspective of the advantages and shortcomings of the approach presented in this work.

4.1 Evaluating Against Ground Truth Data

In order to be able to define an objective way of measuring the effectiveness of our techniques, we first need to know exactly what output we are trying to achieve. Computing the magnitude of deviations from that "ideal" goal is then a good way of establishing a quantifiable performance measure. Figure 4.1 outlines the process that realizes this idea: Basically, we create a number of artificial interpretations for which an "ideal" ground truth[14] tempo curve is known, compute a regular tempo curve for each of these interpretations, and compare these computed curves against the ground truth. The detailed steps are as follows:

Step 1. Generate a number of reference MIDIs from a representative set of scores covering several different music genres. Synthesize one or more tempo curves for each of these references according to a certain parameter set, and use the synthetic curves to warp (or *distort*) the reference MIDIs. Create an audio representation from these warped MIDIs using a high-quality synthesizer. This results in a number of *artificial interpretations* that have the tempo characteristics of the synthetic curves, i.e. the synthetic tempo curves act as *ground truth* tempo curves for the respective artificial interpretations. These artificial interpretations are stored as wave files.

[14]The term "ground truth" is derived from remote sensing applications such as cartography and satellite imagery and describes data of a known good quality that can be used for measurement/calibration purposes.

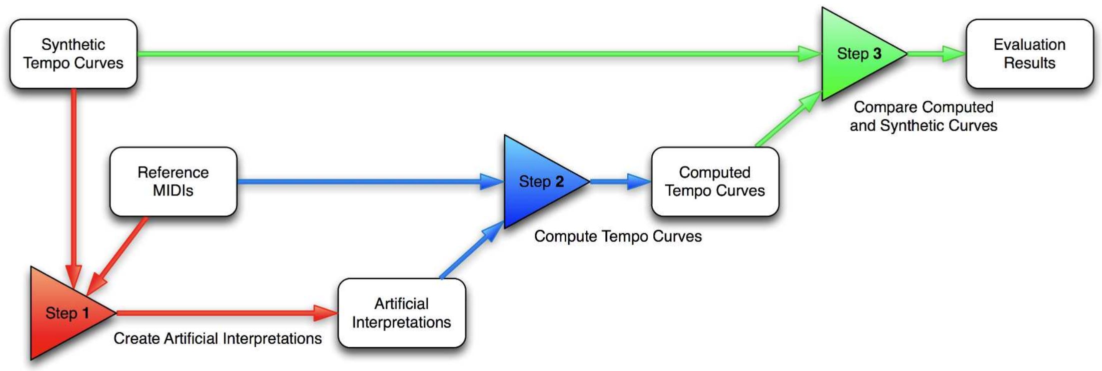

Figure 4.1: Schematic outline of the ground truth evaluation process

Step 2. Using the reference/artificial interpretation pairings from the first step, compute tempo curves for these interpretations with all three techniques presented in this work (for numerous settings of w and w_{ioi}). The only difference between a regular use case of our algorithms and this run is that the input data used here is synthetic.

Step 3. Compare the tempo curves computed in the second step with the synthetic tempo curves generated in the first step. Ideally, these would be identical, but in reality there will of course be differences. Using some kind of distance metric, measure the respective deviations of the computed curve from the desired ground truth.

From this process, a set of measurements is obtained that describe the performance of the three algorithms over a range of several possible parameter settings. We will now show the settings used for the generation of the evaluation data presented in this work, then proceed to introduce a suitable distance metric and present the actual obtained results.

4.1.1 Evaluation Scenarios

For our evaluation, we produced data on a selection of 15 pieces from the RWC database by Goto et. al [GHNO02]. These pieces were chosen with the intention of representing three different major musical fields: Five pieces were taken from the class of Western classical piano music, five pieces contained classical music not focused on the piano (mainly orchestral), and five pieces served as exemplary pop/jazz works. The individual choices are listed in Table 4.1, along with their respective RWC ID for ease of reference. Since the synchronization algorithm at the foundation of our approach depends on the availability of reliable onset information for precise alignments, we expected the accuracy of computed tempo curves to be higher for the data where this was the case, which concerns the piano pieces in particular.

To get a broad spectrum of analysis data, we formulated five different scenarios that presented challenges of varying degree of difficulty to our techniques. We deliberately included scenarios that incorporated somewhat realistic assumptions about the tempo attributes of a given piece as well as scenarios representing unrealistic stress tests designed to expose the limits of our approach.

RWC ID	Comp./Interp.	Piece	Instrumentation
C025	Bach	Fuge, C-Major, BWV 846	Piano
C028	Beethoven	Op. 57, 1st Mov. (Appassionata)	Piano
C031	Chopin	Etude Op. 10, No. 3 (Tristesse)	Piano
C032	Chopin	Etude Op. 25, No. 2 (The Bees)	Piano
C029	Schumann	Reverie (Träumerei)	Piano
C003	Beethoven	Op. 67, 1st Mov. (Fifth Symphony)	Orchestra
C015	Borodin	String Quartett No. 2, 3rd Mov.	Strings
C022	Brahms	Hungarian Dance No. 5	Orchestra
C044	Rimski-Korsakov	Flight of the Bumblebee	Flute/Piano
C044	Schubert	Op. 89, No. 5 (Der Lindenbaum)	Voice/Piano
J001	Nakamura	Jive	Piano
J038	HH Band	The Entertainer	Big Band
J041	Umitsuki Quartet	Friction	Sax/Bass/Perc.
P031	Nagayama	Moving Round and Round	Electronic
P093	Burke	Sweet Dreams	Voice/Guitar

Table 4.1: Pieces used for quantitative technique evaluations

In a general sense, a scenario was characterized by three attributes which described strength and frequency of the tempo variations allowed in that scenario. In a more specific sense, each scenario consisted of a number of artificial interpretations which were produced according to these attributes. The synthetic tempo curves necessary for the production of these interpretations were generated in the following way: According to the range of allowed variation, a random number generator picked several tempo indicators that prescribed a piece's tempo at distinct points in time. The tempo was then interpolated between these points to arrive at a relatively smooth tempo curve. To account for the inherent variation of the randomized process, three different synthetic tempo curves were produced for each piece in this way—the subsequent evaluation then computed data points for each of the curves individually, and returned an averaged result. Each scenario included all 15 files, making the total number of artificial interpretations and associated tempo curves of a single scenario 45, respectively.

The three parameters that guided synthesis of a scenario's tempo curve were as follows:

Interpolation method. Two different interpolation models were used. *Linear interpolation* performed a gradual change between two tempi to mimic ritardandi and accelerandi, while the *step-function interpolation* method maintained constant tempo over a certain amount of time, but then performed a sudden jump to another region of constant tempo. This situation arises in regular scores e.g. in the case of fermatas that register as sudden slow-downs in the span of a single note in the tempo curve.

Interpolation interval length. This interval describes the duration of one segment of the reference that would be warped according to a constant tempo in the case of step-function interpolation, or to a constant acceleration/deceleration in the case of linear interpolation. Two different durations were used for this, one of 5 s and one of 10 s.

Interpolation range. The range of acceptable tempi concerns the output boundaries of the randomization algorithm, which were given in terms of the original tempo. Again, two

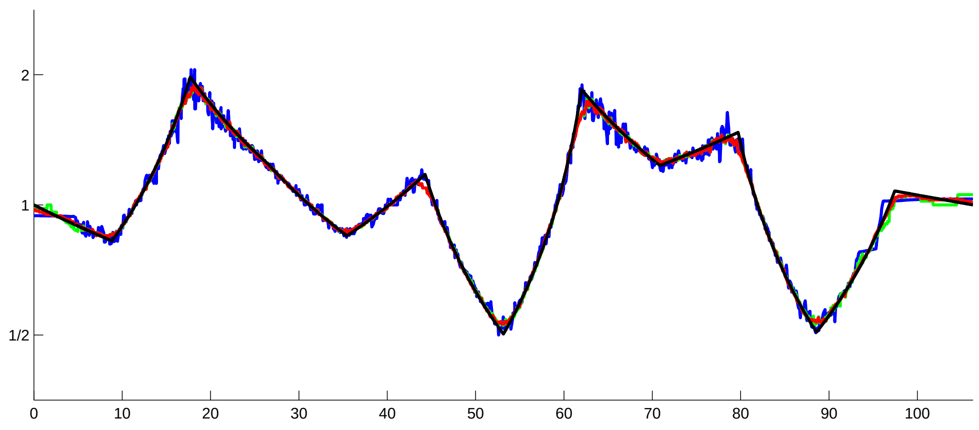

Figure 4.2: Linearly interpolated ground truth tempo curve (black) approximated by FW (green), AW (blue) and FWC (red) techniques, time in seconds ($w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 10$)

possible specifications were used: half to double the original tempo, and quarter to four times the original tempo (allowing tempo changes of up to a factor of 16).

The five scenarios used for the actual evaluation were generated according to the following parameter settings:

Scenario	Interpolation method	Interpolation interval	Interpolation range
1	Linear	10 s	1/2 to 2
2	Linear	5 s	1/2 to 2
3	Linear	10 s	1/4 to 4
4	Step-function	10 s	1/2 to 2
5	Step-function	10 s	1/4 to 4

The scenarios were ordered according to expected quality of performance, with Scenario 1 being the easiest and Scenario 5 the hardest for the algorithms to process. Figure 4.2 illustrates how a specific ground truth tempo curve (plotted in black) might look like for the first scenario.[15] To give a feeling for the relative performance that can be expected for such a scenario, the output of all techniques is plotted against this ground truth tempo curve as well. In this rather benevolent example, the approximation of all three algorithms stays mostly true to the expected output. In comparison, output for the fifth scenario (Fig. 4.3) seems less accurate, even though this particular example is still essentially well-behaved. Note the different tempo scaling for the two figures.

4.1.2 Evaluation Metric

The metric used to measure the "distance" of computed curve deviations from the ground truth tempo is motivated by the idea of relating such a distance to the reference tempo

[15]The curve displays a small distortion compared to a fully linear curve. This is due to the specific process used to generate the image and has no bearing on the fact that the performed interpolation was indeed linear.

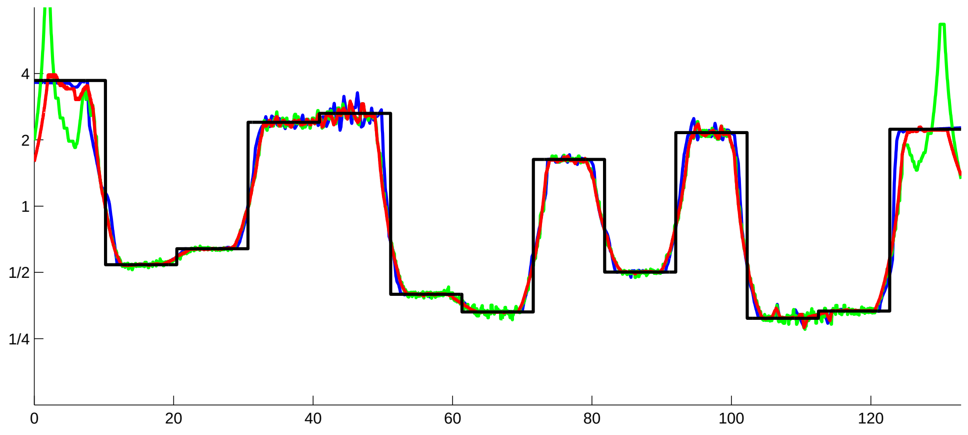

Figure 4.3: Step-function interpolated ground truth curve (black) approximated by FW (green), AW (blue) and FWC (red) techniques, time in seconds ($w = 4\,\mathrm{s}$, $w_{\mathrm{ioi}} = 10$)

laid down by the ground truth. The goal is to group deviations by *scale* rather than by absolute *value*. For example, assume that the ground truth tempo curve was a simple constant distortion of the original tempo by the factor two, i.e. the artificial interpretation has double the tempo of the reference. Further assume that differently computed tempo curves yield different approximations of the ground truth: In one case, the tempo is estimated to be constantly 1, in the other case the result comes out as a constant 4. This means that the two computations estimate the tempo of the artificial interpretation to be equivalent to the original tempo or four times the original tempo, respectively. However, in relation to the actual (ground truth tempo) of 2, both approximations have the same error ratio: The first computation underestimates this actual tempo by a factor of two, the second one overestimates it by the same factor. Since neither of these estimations has a qualitative difference over the other, the distance measure should assign the same error value to both of these flawed approximations.

To fulfill this requirement, we define a distance measure δ as follows: Let $N \in \mathbb{N}$ be the length of a feature sequence describing a given musical piece, $n \in [1 : N]$ an arbitrary but fixed index into this sequence, $g : [1 : N] \to \mathbb{R}_{\geq 0}$ a ground truth tempo curve of an (artificial) interpretation of this piece and $\tau : [1 : N] \to \mathbb{R}_{\geq 0}$ a tempo curve computed for the same interpretation of the piece. Then, the distance measure $\delta : [1 : N] \to \mathbb{R}_{\geq 0}$ between g and τ is a function defined by

$$\delta_g^\tau(n) := \left|\log_2\left(\frac{\tau(n)}{g(n)}\right)\right| \cdot 100$$

Here, dividing the computed tempo curve value by the ground truth value achieves the desired effect of measuring error scale rather than error value. Taking the logarithm of the resulting value has two different purposes: The first is to emphasize small-scale deviations from the ground truth tempo and lessen the impact of outliers, the second is to adjust the computed values to the graphical plots of the tempo curves that use a logarithmic tempo scale as well. Since deviations of the computed curve from the ground truth tempo curve turn out to be seldom larger than by a factor of two (i.e. half or double the ground truth tempo), and the binary logarithm takes on an almost linear shape in the interval $[0.5, 2]$, this does not affect the computed values too much. The sign of the computed value is discarded since we are not interested in the respective nature of the deviation. Lastly, the computed value is scaled up slightly since most measurements fell into the range between 0.01 and 0.20. This became inconvenient to display, so the values were translated to 1 and 20, respectively. The result approximates a measure of deviation in percent of the original tempo,[16] so a value of $\delta_g^\tau(n) = 2$ can be taken to mean that τ deviates from g in the order of 2% at point n—for a sample tempo of 120 BPM, a deviation in the order of 2.4 BPM.

The result of evaluating δ_g^τ at all points $n \in [1 : N]$ is a data sequence describing pointwise deviations of the computed tempo curve from the given ground truth tempo. Three characteristics of this result sequence are of particular interest: The mean value, the maximum value and the standard deviation of the complete data set. Of these, the mean represents "overall performance quality" of the evaluated technique, the maximum indicates outlier values that may be the result of synchronization errors resulting in a faulty warping path, and the standard deviation can be taken as a reliability measure of the technique—the higher the standard deviation, the less confidence can be placed in the technique's tempo estimate for a given point in time. Result tables are given in terms of these three indicators. Here, results for individual files are reproduced in full in the appendix (Tables A.1–A.16), but will be shown in an abridged version for the discussion of this chapter. In particular, the abridged table contains only average values for the different instrumental classes and an overall average. Maximum values are left out in the abridged table, since they are indicative of exceptional outliers rather than the more interesting regular behavior of the techniques.

4.1.3 Evaluation Results

In the following, we present and discuss the evaluation results for each of the five scenarios individually. For each scenario, abridged result tables will show typical evaluation data, with the full tables reproduced in the appendix. In all tables, the quoted value for the parameter w designates the input for both the FW and the FWC technique, so that their individual results are directly comparable (i.e., improvements from FW to FWC technique inside the context of one table are always due to IOI correction of the warping path).

Scenario 1

The first scenario uses a conservative configuration for tempo variations and can thus be said to be somewhat benevolent. This does not mean that it's not representative of "real-world" settings—in fact, the assumption that tempo changes occur every ten seconds and may range within a factor of up to 4 in relation to a previous tempo is valid for a large number of cases. The only thing that is excluded here are sudden changes of the tempo, as in the case of e.g. fermatas.

Tables A.1–A.4 show the complete evaluation results of this scenario for ever larger settings of w and w_{ioi}. While the settings of Table A.1 represent a very moderate smoothing configuration,

[16]The result would be precisely equivalent to such a percentage if we set $\delta_g^\tau(n) := \left|\frac{\tau(n)}{g(n)} - 1\right| \cdot 100$ for $\tau(n) \geq g(n)$ and $\delta_g^\tau(n) := \left|\frac{g(n)}{\tau(n)} - 1\right| \cdot 100$ otherwise. This is obviously not as elegant as the logarithmic solution.

Results by instrumental class	**FW**		**AW**		**FWC**	
	mean	std	mean	std	mean	std
Average over piano	5.66	10.42	5.50	9.07	3.25	6.24
Average over non-piano	4.17	5.20	5.91	8.48	3.22	4.17
Average over jazz/pop	3.67	5.10	6.80	10.78	3.20	4.70
Average over all	4.50	6.90	6.07	9.44	3.22	5.04

Table 4.2: Results for Scenario 1, $w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 10$

Results by instrumental class	**FW**		**AW**		**FWC**	
	mean	std	mean	std	mean	std
Average over piano	4.90	8.33	6.19	9.11	3.19	5.98
Average over non-piano	3.55	4.39	4.65	5.70	2.89	3.78
Average over jazz/pop	3.15	4.31	4.53	6.29	2.81	4.10
Average over all	3.87	5.68	5.12	7.03	2.96	4.62

Table 4.3: Results for Scenario 1, $w = 4\,\mathrm{s}$, $w_{\mathrm{ioi}} = 20$

Table A.4 shows the effects of employing much stronger adjustments. Here and in the other scenarios, the values of w and w_{ioi} are chosen in such a way that one table shows optimal results for the scenario, and the other tables show results resulting from suboptimal settings of w and w_{ioi}. This is done in order to give a feeling for the dimension of change that can be expected when experimenting with different parameter settings in different scenarios. For the suboptimal results, the parameters were chosen by modifying the "optimal" parameters until a clear trend could be identified in the new result table. In general, this meant larger variations in the case of w_{ioi} than for w.

Optimal settings for the first scenario are found in Tables A.2 and A.3. Abridged versions of these are given in Tables 4.2 and 4.3. Overall, the FW/FWC techniques produce their best results for $w = 4$, with single files (like the second Chopin Etude **C032**) performing better for $w = 3$. The AW technique profits from the high settings of w_{ioi} in Table 4.3, but is generally outperformed by the other two techniques. Of the three techniques, FWC does best, which was to be expected since it is the most sophisticated. The somewhat disappointing performance of the AW technique is most probably due to the fact that in regions of high note density, the smoothing is not strong enough; however, setting the w_{ioi} parameter to a higher value introduces too much blur in other regions. Here, the evaluation shows that the current approach of using a fixed parameter for w_{ioi} is not flexible enough.

One surprising finding is that this scenario does not exhibit the expected advantage of piano music over other styles. This seems to be due to two factors: On one hand, there is a very good alignment quality for the non-piano and jazz/pop pieces that keeps tempo curve errors quite low. For such "harmless" distortions as used in this scenario, the harmonic progressions in these pieces seem to be sufficient for the alignment algorithm to produce very accurate results. On the other hand, the C-Major Fugue from the Well-Tempered Clavier (**C025**) suffers from a comparatively bad performance over all techniques. This raises the suspicion that the fault lies with the synchronization, which is confirmed by the tempo curve generated

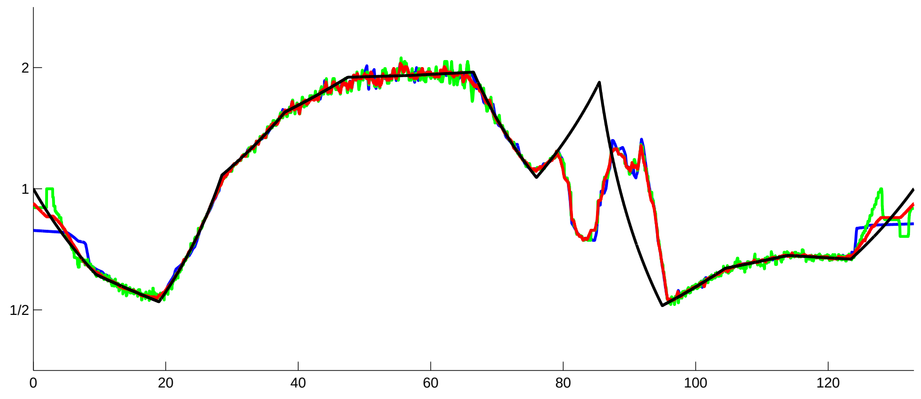

Figure 4.4: A synchronization error affecting tempo curve computation, time in seconds

for the first of the three synthetic distortions used for this piece in this scenario (Fig. 4.4): Here, a synchronization error becomes clearly visible in the region from $t = 88\,\text{s}$ to $t = 97\,\text{s}$. The probable cause of this negative effect seems to be the rapid tempo change in this region that switches from a nearly maximal tempo to the lowest possible value.

The good performance of single files for a window size of $w = 3$ for the FWC case is likely due to an inverse effect: Here, exceptionally good alignments allow the obtaining of high-quality results (e.g. Rimski-Korsakov, **C044**, mean error 1.39 for $w = 3$ and 1.66 for $w = 4$), for which a smaller window size is better suited as it preserves the accuracy of the synchronization without blurring the regions of tempo changes.

Overall, the results seem very good in this scenario, with Table 4.3 representing a kind of ideal case. The effect of broader window size in the FW/FWC cases seems to be advantageous up to a size of about $w = 4$, after which the detrimental effects of slower adaptability to new tempi seem to dominate the result. A window size of $w = 3$ produces better results only for special cases. The AW technique exhibits an overall poorer performance, while still maintaining a comparatively high quality level. Outliers are most pronounced in the AW case, indicating a certain brittleness of the design that is also reflected in a slightly higher standard deviation. The FWC technique seems most robust in the face of the comparatively easy challenges posed by this scenario, and clearly benefits from the IOI interpolation as can be seen by the difference to the "plain-vanilla" FW technique.

The tempo variations contained in this setting are representative of many real-world scenarios, hence the good performance of the techniques serve as validation that usage of the FWC technique is appropriate to derive tempo estimations of fair accuracy in these cases.

Scenario 2

The second scenario introduces more frequent tempo changes into an otherwise still benevolent setting: The distance between regions of different acceleration is now 5 s instead of 10 s. The result data (Tables A.5–A.7) confirms the expected effects of this: Overall quality is still very high, with a slightly lower baseline due to a greater number of tempo change locations.

Results by instrumental class	**FW**		**AW**		**FWC**	
	mean	std	mean	std	mean	std
Average over piano	6.85	10.24	7.36	9.99	4.55	6.84
Average over non-piano	5.81	7.72	7.22	9.25	4.95	6.91
Average over jazz/pop	4.98	6.96	6.98	9.96	4.48	6.36
Average over all	5.88	8.31	7.19	9.73	4.66	6.70

Table 4.4: Results for Scenario 2, $w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 12$

Best results are obtained by smaller window sizes than in the previous example, with optimal performance at $w = 3$ and $w_{\mathrm{ioi}} = 12$ (Table 4.4). This is an obvious and expected result of shortening the interpolation interval length, since the new setting demands higher adaptability of the algorithms to a new tempo that can only be gained by smaller window sizes. For the FWC technique, results for the piano pieces are comparatively better than in the first scenario, as the Bach Fugue **C025** is now processed correctly—this is especially pronounced for $w = 2$, where the algorithm can profit from the high quality of available onset information. Still, synchronization results are good enough for the other styles (jazz/pop in particular) that there is no clear advantage for any of the three classes in the general case.

Scenario 3

The third scenario poses the first great challenge to the three techniques. Even though the interpolation interval is scaled back to 10 s, the interpolation range of 1/4 to 4 allows for accelerations/decelerations of up to factor 16. This proves too much for a proper alignment: The results of Tables A.8–A.11 show that most pieces are affected by more or less serious synchronization errors. Only three pieces are exempt, with the piano pieces finally profiting from better availability of onset information—two of the three pieces are of this class (Bach **C025** and Chopin **C032**), with the third piece (Rimski-Korsakov **C044**) also being partly arranged for the piano. This last piece also had consistently excellent performance in Scenarios 1 and 2, which implies that it is especially well suited for the employed alignment procedure. Performance improves in all instrumental classes for relatively strong smoothing configurations (Table 4.5). However, this improvement is relative: Synchronization errors dominate the overall result, which has a low baseline that cannot be much polished, even using extreme averaging parameters (Table A.11).

Figure 4.5 illustrates how these numbers translate into "real-life" performance. Aside from synchronization errors ranging from negligible to catastrophic (e.g. for the FW technique, which shows a maximum deviation of 721.95 at $t = 120\,\mathrm{s}$), there are a number of other interesting phenomena to be observed. We can identify two locations of high acceleration followed by abrupt deceleration ($t = 74\,\mathrm{s}$ and $t = 103\,\mathrm{s}$), which the AW technique averages out in both cases to produce results significantly below the tempo peak. The same happens to the FW/FWC techniques in the second case, but in the first one, the tempo change provokes a greater confusion. In this instance, the change seems to be registered at a slightly earlier place in time than when it actually happens.

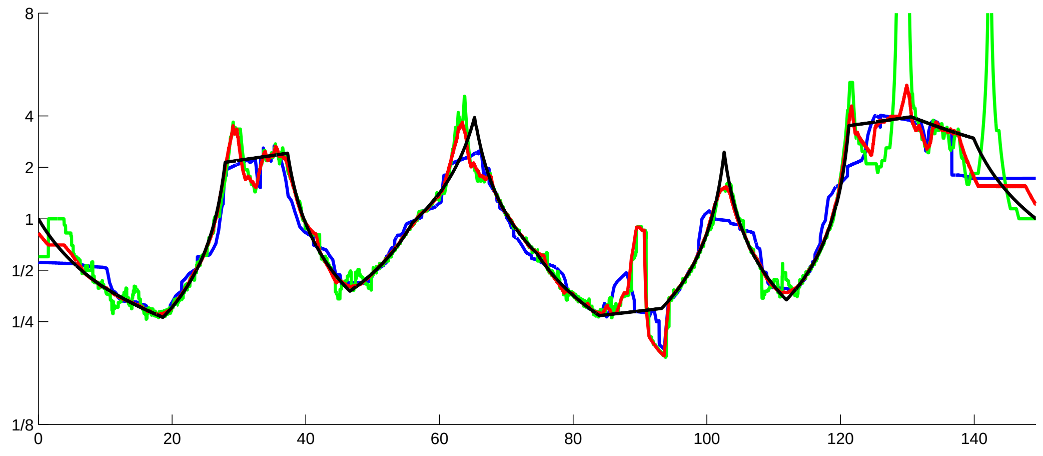

Figure 4.5: Sample result for Scenario 3, Schumann **C029**, $w = 3$, $w_{\text{ioi}} = 12$, time in seconds

Results by instrumental class	**FW** mean	std	**AW** mean	std	**FWC** mean	std
Average over piano	19.12	32.14	21.94	31.73	16.44	28.38
Average over non-piano	21.92	35.03	23.93	35.84	20.88	34.58
Average over jazz/pop	27.45	40.26	29.57	42.48	26.95	40.25
Average over all	22.83	35.81	25.15	36.68	21.42	34.40

Table 4.5: Results for Scenario 3, $w = 4\,\text{s}$, $w_{\text{ioi}} = 20$

Another "strange" occurrence takes place at $t = 30\,\text{s}$: Here, it seems that the FW/FWC tempo curves overshoot the actual tempo, but then correct this error by underestimating the tempo of the following two seconds until the approximations converge at the actual ground truth curve again. Such a pattern occurs when an important synchronization event (e.g. a note onset or a harmonic change) is aligned to an earlier place in time than when it actually occurs, while the information in the surrounding context is processed accurately. For example, in a setting where a note onset occurs every three beats, the region $[1 : 9]$ might be evaluated in such a way that onsets are not placed at times 1, 4 and 7 (as would be accurate), but at times 1, 3 and 7 instead. The distance between first and second onset is then shortened (with respect to the actual distance), and the distance between second and third onset is lengthened. This would cause the tempo in the first region to be overestimated by a factor of 1/3, and the tempo in the second region to be underestimated by the same factor. The resulting tempo curve would then resemble the tempo curve gained for the Schumann piece at $t = 30\,\text{s}$. Similar scenarios are of course possible for the reverse case as well.

A relatively consistent phenomenon is the mangling of beginning and end of a piece by all three algorithms. This is due to three factors: First of all, lack of data in these regions,[17] secondly the inadequacy of the assumption that tempo is constantly 1 at regions not defined by

[17] Which is a problem since tempo is always dependent on context—after all, what is the tempo of a single note? At the beginning and the end, this context information is simply absent.

the warping path (indeed, results seem to be more convincing for ground truth tempo curves where beginning and end happen to coincide with a tempo of 1), and thirdly the behavior of the DTW synchronization algorithm in these boundary cases, which sometimes leaves a little room for improvement.

Scenario 4

The fourth scenario again limits the maximal factor of tempo change to 4, but instead allows very rapid periods of change followed by regions of constant tempo. Contrary to our expectations, this scenario actually produced better results than the previous setting since it did not provoke such a great number of synchronization errors. The limiting factor here was again the size of the averaging window, with the algorithms needing to adapt to changes quicker than in the otherwise comparable first scenario. The nature of these changes in this scenario leads to an overestimation of the overall error: Figure 4.6 shows the error curve for a specific file of the scenario together with the corresponding mean error. Tempo curve results for this file are shown in figure 4.7. It can be seen that even though the error baseline is much smaller than the mean error, the great deviations caused by the rapid tempo changes inhibit better results in this case.

Tables A.12–A.15 show results that lie somewhere between Scenarios 2 and 3 in terms of overall quality. Window sizes of $w = 2$ and $w = 3$ prove optimal for the FW/FWC algorithms, and

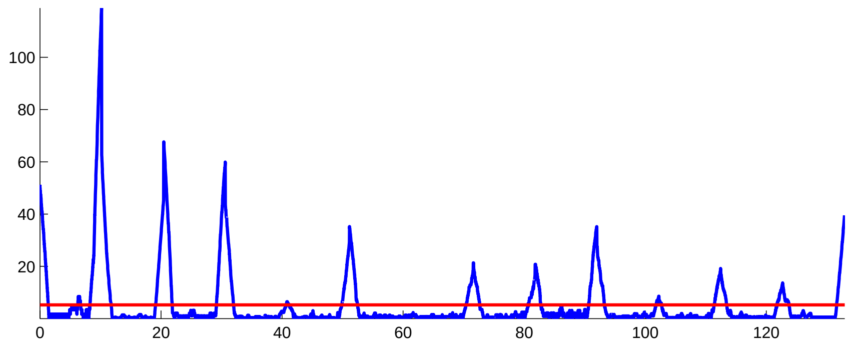

Figure 4.6: Mean error (red) vs. actual error curve δ_g^τ (blue) for Scenario 4, Bach **C025**, τ computed using the FWC technique ($w = 3$), time in seconds

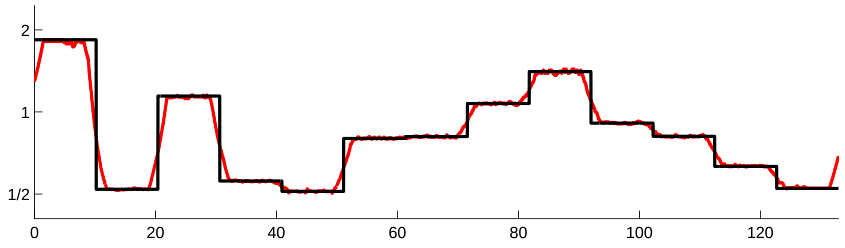

Figure 4.7: Ground truth tempo curve (black) and FWC tempo curve (red) for Scenario 4, Bach **C025**, $w = 3$, time in seconds

Results by instrumental class	**FW** mean	std	**AW** mean	std	**FWC** mean	std
Average over piano	13.55	22.57	10.17	18.46	9.26	17.13
Average over non-piano	10.29	15.16	9.31	14.59	8.95	14.37
Average over jazz/pop	10.39	16.24	11.42	18.99	9.67	15.99
Average over all	11.41	17.99	10.30	17.35	9.29	15.83

Table 4.6: Results for Scenario 4, $w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 12$

Results by instrumental class	**FW** mean	std	**AW** mean	std	**FWC** mean	std
Average over piano	54.49	71.30	51.70	62.05	46.60	62.39
Average over non-piano	49.97	62.91	50.31	62.37	47.85	61.37
Average over jazz/pop	55.72	64.93	55.71	66.00	54.69	64.91
Average over all	53.39	66.38	52.57	63.47	49.71	62.89

Table 4.7: Results for Scenario 5, $w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 20$

the AW technique produces best results for $w_{\mathrm{ioi}} = 12$ (Table 4.6). The other visible trends are repetitions of phenomena reported in earlier scenarios: The FWC technique continues to outperform the other techniques, larger window sizes improve performance in cases of bad synchronization and impair performance in cases of good synchronization, and performance on piano pieces is (non-significantly) better than for other styles.

Scenario 5

In the final scenario, the performance finally breaks down in full. No piece can be synchronized without errors; Figure 4.8 illustrates why this is the case. As can be seen there, the extreme tempo variations on a very small time range do not permit satisfactory synchronization results. We only reproduce one result table (Table 4.7) to demonstrate the output range that is to be expected in such a scenario. Not much can be said here, other than that any amount of smoothing is of course wasted on a faulty synchronization—results of the tempo curve estimation will be better once these errors can be removed.

4.1.4 Evaluation Summary

The five scenarios that were evaluated demonstrate that in general, the presented techniques (and the FWC approach in particular) work well for the extraction of tempo characteristics of a piece of music. The three most realistic Scenarios 1, 2 and 4 feature mean error rates averaging between 2–10, which is accurate enough for MIR applications concerned with phrase-level tempo attributes. Using the FWC technique with a window size of $w = 3$ seems to be the best compromise between result precision and robustness against synchronization artifacts—if the alignment is known to be unreliable, a choice of $w = 4$ may be better suited, and if high alignment qualities can be expected, $w = 2$ enables obtaining of higher-precision results.

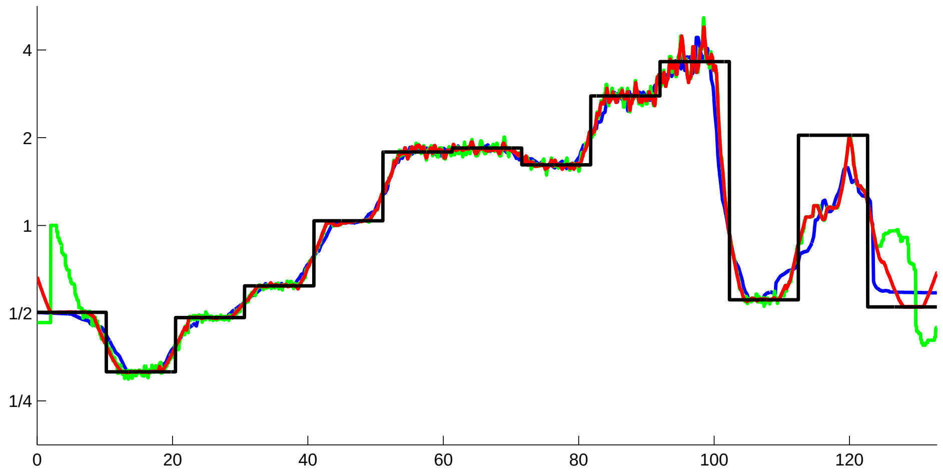

Figure 4.8: Sample result for Scenario 5, Bach **C025**, $w = 3$, $w_{\mathrm{ioi}} = 20$, time in seconds

In case of significant synchronization errors, the techniques do not estimate the tempo reliably. In our settings, such errors were provoked mostly by drastic tempo distortions that do not occur in practice, but even for simple cases such as in the first scenario, the DTW procedure occasionally computed faulty warping paths that resulted in a deteriorated performance. For the practical usage of the presented approach, it will be necessary to compute estimations of the alignment quality at the time of music synchronization in order to be able to judge during which passages of a piece the tempo estimations might become unreliable.

Of the three techniques, the FWC technique is an easy choice as the best candidate for practical employment: In all five scenarios, it produced the best results both in terms of precision and robustness. The AW technique may have merit as a basis for further developments that operate on note-level structures; the FW technique is primarily useful as a didactic device for the subsequent introduction of the FWC approach.

4.2 Evaluating Selected Musical Examples

Although the quantitative evaluation of the presented techniques indicated that they were well suited to be used for performance analysis purposes, their practical benefit has not yet been demonstrated. This section showcases several different examples which illustrate that the approach can indeed be used to derive musically interesting statements about specific performances. For this, we compute performance curves for various interpretations of Western classical piano pieces that were either taken from off-the-shelf recordings or produced for our internal database, and discuss how they relate to musical properties of the respective pieces. Since the FWC technique has been shown to yield the best results during the quantitative evaluation phase, all qualitative evaluations will be done using this technique.

Performance analysis techniques can often be classified as either being concerned with *similarities* or *commonalities* of playing style in the interpretations of different artists (or even in different performances by the same artist), or focusing on *systematic differences* between various interpretations of the same piece (usually done by different artists). In the first case, the main research goal is to discover universal rules that govern playing style and musical interpretation of a composer's intentions; in the second case, the goal lies in identifying the unique traits of an artist's playing style that distinguish his performances from those by different artists. Examples of both kinds of research will be discussed in Chapter 5.

Our qualitative evaluation is divided in a similar manner. We begin by showing similarities across different interpretations of the same piece that have straightforward musical explanations, in order to demonstrate that our approach yields the expected results in such a case.

4.2.1 Evaluation Focusing on Common Interpretational Traits

Although each artist has his own idiosyncratic playing style that can be instantly recognizable in the case of an established performer who has had the time and experience to develop a unique artistic identity, they all speak the same musical language. Any score written in a specific style calls for an interpretation that does justice to the musical demands and expectations of that style, and this will be reflected in the respective performer's playing style. The range of this musical expectations extends from fundamentals like phrasing and shaping of musical developments to agogical aspects like the interpretation of indicators such as "staccato". In this section, we will focus on the more basic aspects that have clear groundings in the musical score which will be reproduced alongside the respective tempo curve.

Robert Schumann: Kinderszenen op. 15 no. 7, "Träumerei"

In the first example, we will discuss the first eight measures of the popular "Träumerei" from Schumann's "Kinderszenen". Here, three different interpretations were analyzed with respect to their relation to the score (Fig. 4.9). Two of the interpretations were taken from regular CD recordings, while a third one was produced by a member of our workgroup with a strong musical grounding (Verena Konz). First of all, notice that the basic shapes of the tempo curve have a surprisingly high degree of correlation. Even though their tempo baseline differs, the different artists have chosen similar ways of illustrating the structure of the score. The first musical point of rest reached by the performers is the subdominant B chord of the second measure, which coincides with a temporary melodic climax (the soprano voice's f′′). Accordingly, the tempo curve shows a deceleration in the second measure for all three interpretations.

To underline the stronger sense of action conveyed by the eighth movements of measure three, all pianists choose a faster pace that reaches or exceeds their previous maximum tempo. This heightened sense of movement comes to a rest on the dominant C chord of measure 4, which is in turn resolved in the next measure by the tonic F. Two of the three interpretations emphasize this striving towards the resolution by giving the bass movement leading to the tonic root note a distinct accelerando/ritardando shape.

The harmonic and melodic climax of this excerpt of the piece is reached in measure 6, where all three performers take some time to let the listener appreciate the suspense created by the A^7 chord that leads into the d chord of measure 7. To create an adequate feeling of closure at the end of the completed musical thought, all interpretations follow the given phrasing

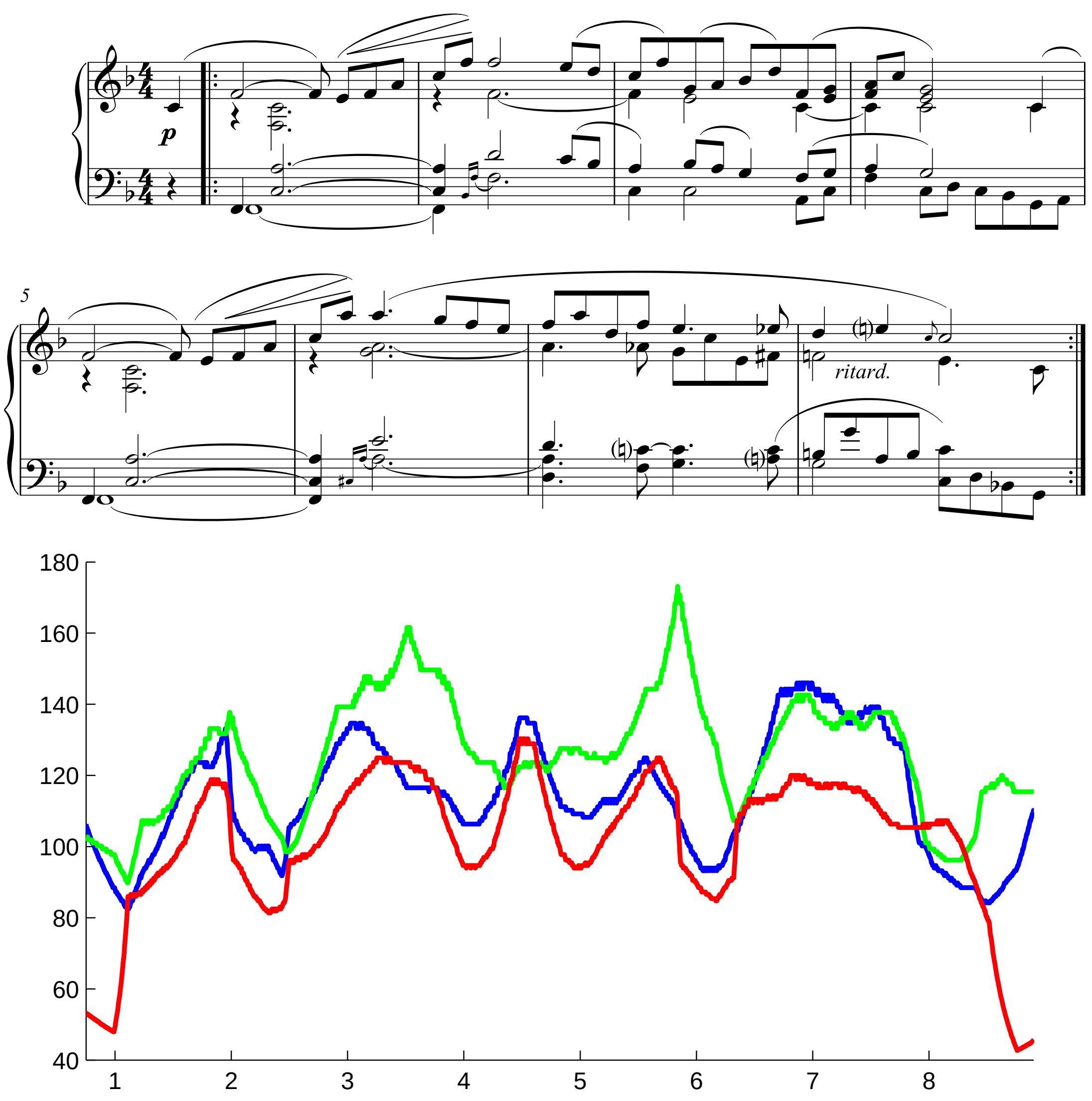

Figure 4.9: Three interpretations of Robert Schumann: Kinderszenen op. 15, “Träumerei”, time in measures, tempo in BPM, $w = 2$

direction that implies a somewhat uniform tempo from the middle of measure 6 until the final target of the tonic F in measure 8 is reached. The ritardando of measure 8 is approached by all performers a bit earlier than indicated, beginning at the end of measure 7. This may be a result of prolonging the sense of unresolved tension on the diminished seventh chord that precedes the final measure.

The last measure shows one difference between the two CD recordings and our “custom-made” recording: Both regular recordings gather speed again to launch into a new rendition of the repeat section, while the third interpretation did not perform the repetition and so comes to a full stop on measure 8.

Overall, this example demonstrates that the shape of the tempo curves results from the performer’s interpretation of the musical meaning of the score, as should be expected. The phenomena visible in all three interpretations have clear explanations that in the majority of all cases can be deduced from the score, and do not appear to be the result of arbitrary processes of chance. Moreover, the three tempo curves bear a strong resemblance to each other, giving credence to the claim that the harmonic and structural demands of this piece dominate individual artistic playing style in this example.

Ludwig van Beethoven: Sonata No. 8 op. 13 “Pathétique”

An example which illustrates that our approach is capable of uncovering unexpected similarities between different performances of a piece is given in Figure 4.10. Here, the interpretations were performed by two students of a common piano class at the Saar Academy of Music. Their actual tempo shaping is not relevant in this context, hence the accompanying score is not shown here (it is, however, reproduced in Figure 4.17). Striking about the two interpretations is their choice of nearly identical *absolute* tempo for measures 5–10. We can only speculate about the reason for this—they may have practiced together, or they may have received similar instructions by a common teacher. However, their tempo phrasing seems too much alike to dismiss this result as a mere coincidence.[18] Discovering such similarities could potentially be automated by using the distance measure δ on pairwise combinations of such interpretations and searching for regions where the results fall below a certain threshold.

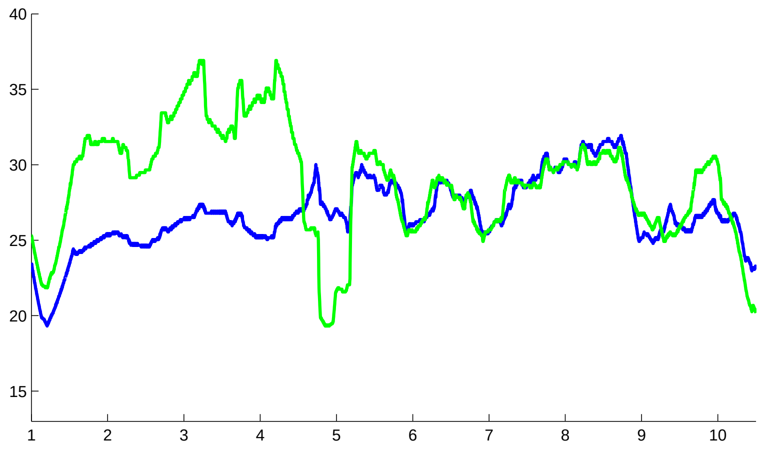

Figure 4.10: Identical phrasing by two pianists: **PathBeg**, time in measures, tempo in BPM, $w = 3$

A second example from the same score illustrates how performers handle the transition from one region of thematic material to the next. Such a change occurs in the Beethoven sonata e.g. in measures 49/50 (Fig. 4.11). Here, the second theme of the sonata’s first movement is introduced for the first time. Beethoven modulates from the first theme in c minor to the second theme in e♭ minor by moving from A♭ major (measure 39) to B♭ major (measure 42),

[18]Another case that turned up during our analysis seemed too perfect to be true: Here, the two tempo curves where almost exactly identical over the whole course of the piece. Exploration of this quickly revealed that the same recording had found its way into our database twice by accident, the two only differing by sample rate. Of course, this explanation can be ruled out for the interpretations of Figure 4.10.

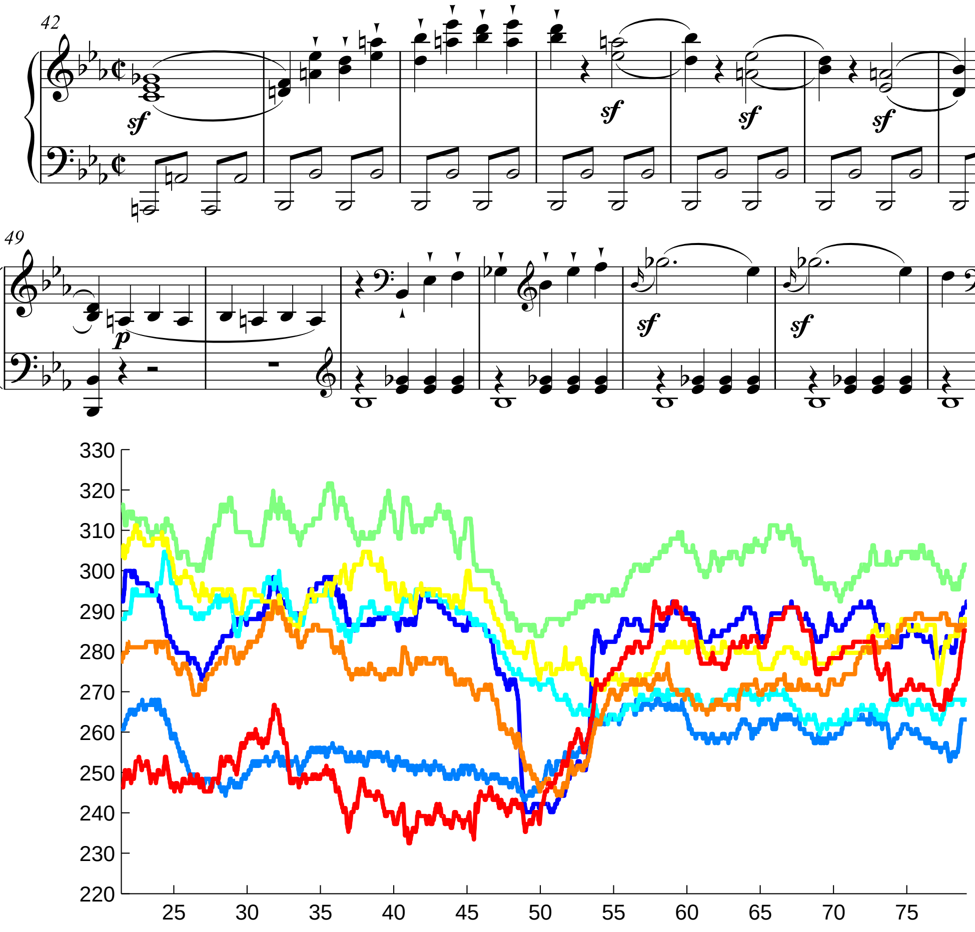

Figure 4.11: Thematic change in **PathExp** (M. 42–55) reflected in tempo curves of various interpretations (M. 25–75), time in measures, tempo in BPM

the dominant of the new key of the second theme. In measure 51, the dominant resolves into the temporary new tonic for the first time, and the second theme is stated.

The tempo curves which were computed from several interpretations performed by students of the piano class mentioned earlier on clearly show that all performers recognized this as an important structural event. Depending on their initial tempo, they approached the passage differently: Those who had performed the preceding section in a fast tempo had a general tendency to slow down, in three cases even resting for a short while on the first beat of measure 49 before continuing in a slightly slower tempo than before. On the other hand, performers who displayed a slow initial tempo used this transition to speed up their interpretation, in some cases even arriving at a higher tempo rate than reached by their fellow students who had a faster initial tempo.

The commonality across the different interpretations here does not lie in a similar tempo structure that was chosen by all, but rather in the fact that this structural event was taken by all performers as a reason to change their initial tempo. This indicates a similar musical understanding of the structure of the score, even if the individual interpretation of how this structure could best be conveyed to the listener differed from artist to artist.

Franz Schubert: Winterreise D911 No. 5, "Der Lindenbaum"

A rich source of examples both for similarities and systematic differences among interpretations can be found in the large variety of recordings available of Schubert's Winterreise (for voice and piano), e.g. in the Lied "Der Lindenbaum". One case in point can be made for the ending of this piece that shows nearly uniform shaping among many commercially available interpretations (Fig. 4.12). Here, the emotional affect of the song seems to call for a specific musical interpretation that is universally understood and answered. The calmness suggested by the lyrics of measures 72–76 is reflected by a ritardando that comes to a resting point at the end of the phrase in measure 76. After the sung part of the piece has ended, the pianist plays

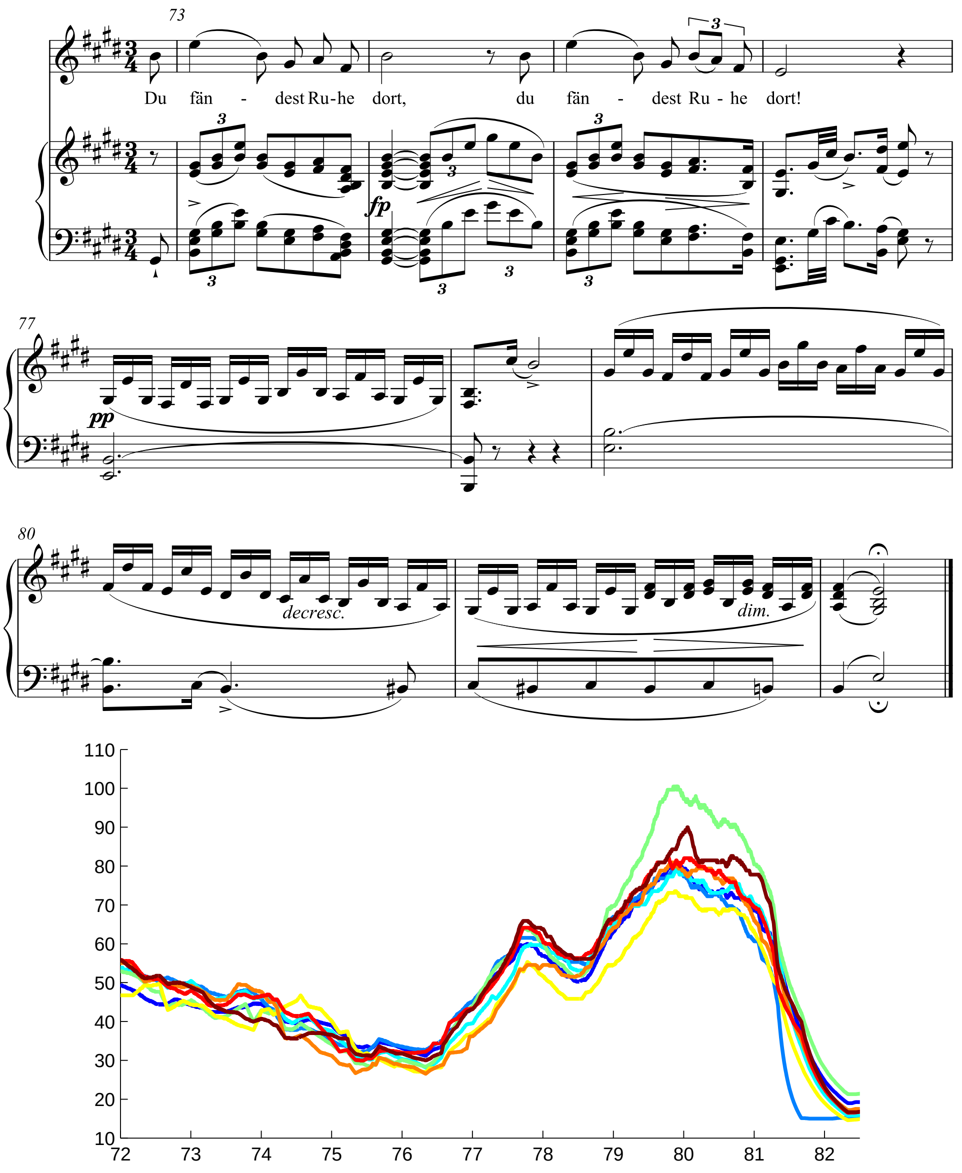

Figure 4.12: Consistent ending forms for Franz Schubert: Winterreise D911, "Der Lindenbaum" (M. 72–82), time in measures, tempo in BPM

a few measures of what could be called musical afterthoughts that recall the agitation felt by the song's protagonist as he is thinking of the lime tree's rustling branches, symbolized by the characteristic triplet figures. In accordance with this interpretation, all pianists accelerate the tempo until they come to a temporary rest on measure 78, then draw a wider phrase that climaxes at an even higher tempo till the final fermata is reached.

4.2.2 Evaluation Focusing on Systematic Differences in Interpretation

As was already partly illustrated by the "Pathétique" example of the preceding section, the individual understanding of a piece's musical layout or a specific musical notion can differ vastly between different artists. However, the most interesting cases are not found in the lone exception to an otherwise unquestioned rule of performance (although these are of course relevant in their own right), but rather in systematic differences between two or more classes of interpretational thought. Simply put, in such a case there is one group of artists who decide on one course of action, and another group of artists that chooses a different approach. Both of these choices may be valid interpretations of the musical source material, but they highlight the personal preferences and sensibilities of the performer.

Franz Schubert: Winterreise D911 No. 5, "Der Lindenbaum"

The same recordings that were already used to demonstrate commonalities among multiple interpretations of Schubert's "Lindenbaum" of course also exhibit sections where different interpretations make use of different approaches. One such section can be found directly at the beginning of the piece: Figure 4.13 shows how the phrase shaping of the first seven measures is performed in these recordings. Beginning in measure 4, two classes of performers can be distinguished: The first class paints measures 4–6 as a big phrase with a roughly uniform tempo that ends with a cadential move to f♯ minor in measures 6–7.[19] In contrast, performers of the second class introduce a relatively early tempo relaxation in measures 4–5 and maintain the slower tempo throughout measures 6 and 7. There is no clearly discernible musical reason for this pattern, other than that the four artists who implement it obviously feel differently about the implications of this part of the composition than the first class of performers.

Another section of this song that exhibits a similar pattern of differing phrase shapings can be explained more directly by the lyrics and musical images used in the score. Figure 4.14 shows the relevant tempo curve, as well as the score providing context information. The lyrics of this passage change from a melancholy "Hier find'st du deine Ruh" (M. 42-44) to an aggressive "Die kalten Winde bliesen mir grad ins Angesicht" (M. 45–49).[20] Schubert has painted the "cold winds" of this image by using triplets that move up and down in a chromatic fashion. Most performers emphasize the protagonist's anguish and bitterness by choosing a faster tempo for the whole passage that comes to an end in measures 57–58. However, two artists instead seem to focus on the swelling and subsiding of the blowing wind, which they realize by playing in a similarly fluctuating manner. Since this gives the passage a comparatively quieter feel, the

[19] A harmonically rather unusual ending. For E major, f♯ is the supertonic chord, i.e. the subdominant parallel.
[20] Translations: "Here you'll find your rest" and "The cold winds were blowing straight into my face."

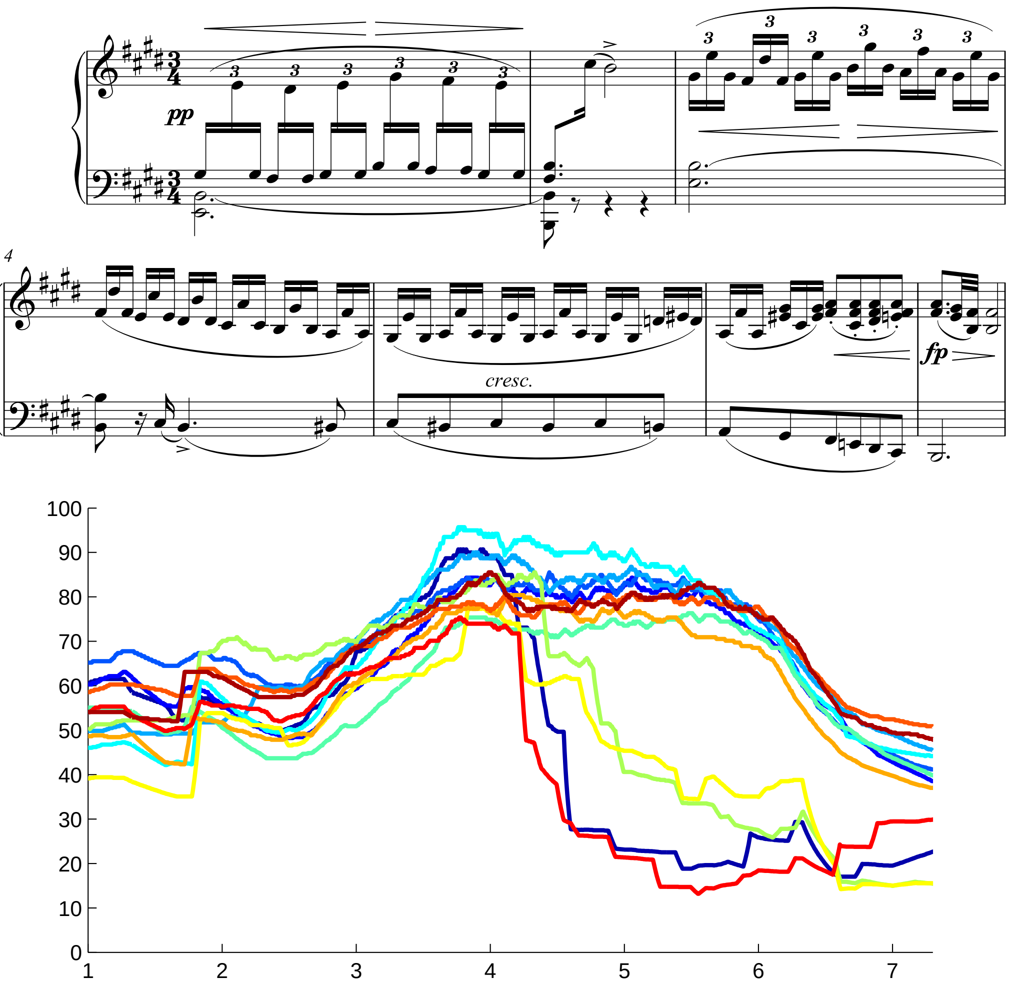

Figure 4.13: Systematic interpretation differences for Franz Schubert: Winterreise D911, "Der Lindenbaum" (M. 1–7), time in measures, tempo in BPM

need for a grand gesture of returning calm is obviated for these interpretations. Instead of resting on the fermata of measure 58 as the other performers do, they continue in their regular tempo, only to be joined by the other artists again in measure 60.

Ludwig van Beethoven: Sonata No. 8 op. 13 "Pathétique"

A different kind of deviation from the expected paths of phrase shaping emerged from the recordings taken of Beethoven's "Pathétique" sonata, performed by art students from the local Saar Academy of Music. Here, we found some tempo changes in performances by single students that did not fit into their general tempo forms. Exploration of these deviations quickly showed that these were involuntary: Perhaps due to a lack of preparation time, single students did not produce "ideal" recordings, but made mistakes during their play that were reflected in the tempo curve.

Figure 4.15 shows two examples of this for the complete **PathExp**. In one interpretation,

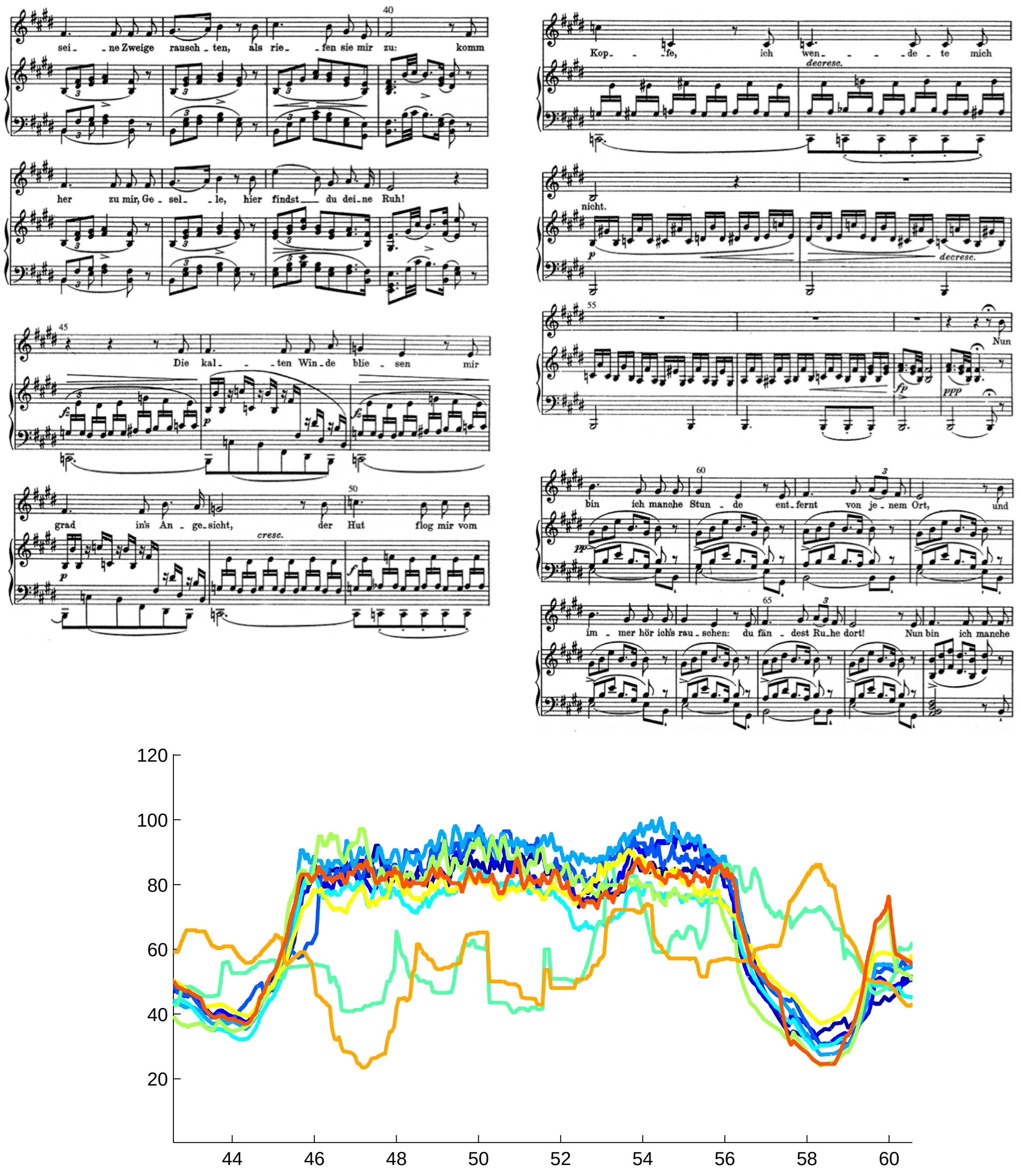

Figure 4.14: Two performances deviating from common interpretatoric consensus for Franz Schubert: Winterreise D911, "Der Lindenbaum" (M. 44–60), time in measures, tempo in BPM

errors occur at measures 40, 50, 70 and 90; the other performance displays one such error near measure 80. These errors show in the tempo curve as sudden drops in tempo that last for a short period of time before the performance returns to its initial tempo.

Indeed, the typical error episode audible in the actual recordings confirms this pattern: The performer plays a wrong note or chord, halts for a short time, then plays a correct version of the passage and continues as normal. While this example of course does not fall into the usual realm of performance analysis concerns, the ability to quickly locate such accidents may nevertheless be very useful in a didactic context, e.g. in a piano lesson performed on a player

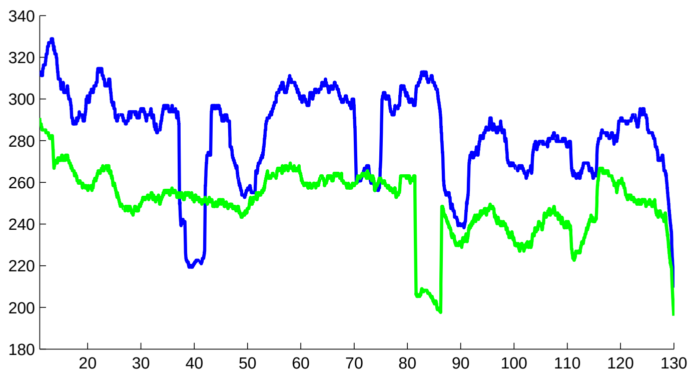

Figure 4.15: Identifying problematic passages in student interpretations of **PathExp**, time in measures, tempo in BPM

piano that automatically monitors the student's playing.

Of course, other examples from the Sonata Pathétique may display more "conventional" deviations. Figure 4.17 shows the tempo of four interpretations of the first ten measures of this work (selected from the performances of art students mentioned in the previous example), and

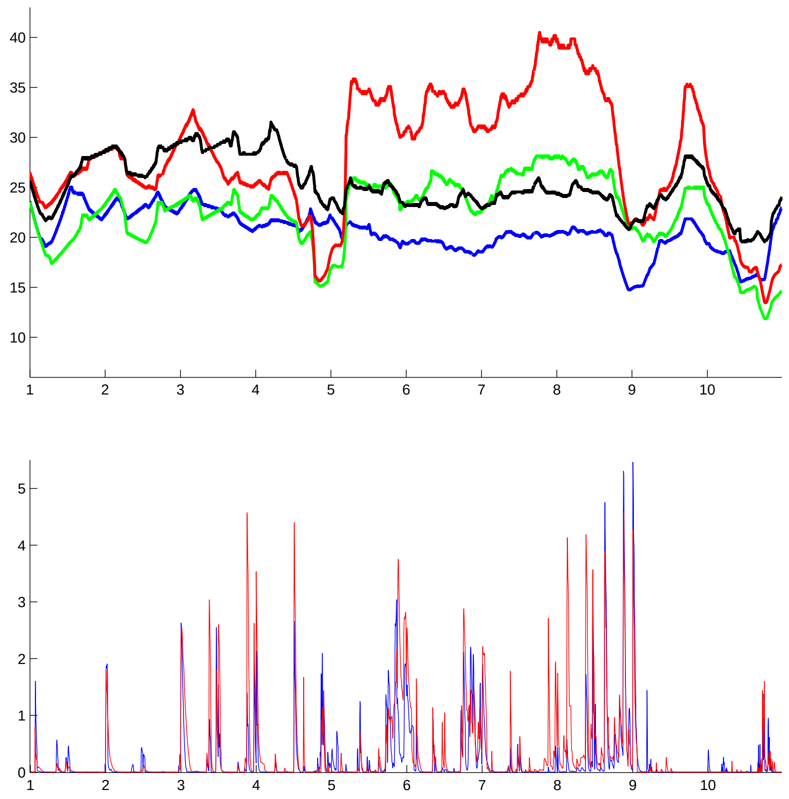

Figure 4.16: Tempo curve (top) and dynamics curve (bottom) for performances of **PathExp** (M. 1–10), time in measures, tempo in BPM, dynamics as in Section 3.3

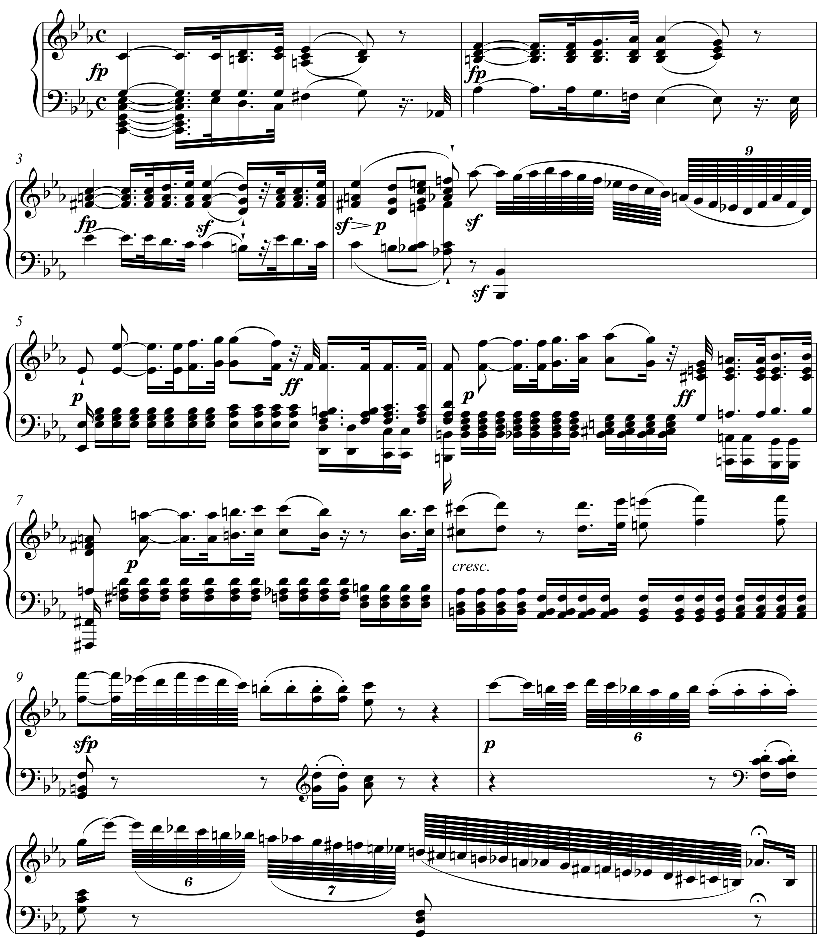

Figure 4.17: Score for **PathExp**, measures 1–10)

also gives dynamics for two of these interpretations (red and blue, colors are consistent across tempo and dynamics curves). Here, the dynamics curves are computed as $\log_2(\text{energy}_t + 1)$ and then scaled to the reference time axis as described in Section 3.3.

The corresponding score of the segment is shown in Figure 4.17. As can be expected, the dynamic directives given by the score are implemented by both artists, albeit in slightly different shapes. Both performances exhibit a gradual crescendo that climaxes in the multiple *subito forte* directives of measure 4. This measure marks the end of the introduction of the first theme of this movement, which is reflected in the score by the textural change in the left-hand accompaniment starting in measure 5 as well as the cadential move to E♭ major (the first appearance of a “pure” major chord in the piece). Appropriately enough, the tempo curves of almost all performances exhibit a slow-down at this juncture.

The next goal for the artists seems to be the *subito forte piano* of measure 9, which is preceded by a *crescendo* in the eighth measure that is clearly visible in the two dynamics curves. Again, the tempo curves confirm the structural importance of this measure (which, not coincidentally, contains the highest note of the whole segment as well as a deceptive cadence) for the artists, all of whom rest on the first beat of the measure for a short time before continuing. The performers choose different ways of approaching this goal: Beginning from measure 5, one artist displays a marked speed-up with respect to the other performances. The same performance differs from the other performances in its choice of dynamics as well. In measure 5, it starts out softer than the other performance shown, but quickly builds up in energy and reaches a significantly louder fortissimo than the other performance in measures 6 and 7, which is then only intensified in measure 8. Here, it is interesting to note that both performances display a quieter fortissimo in measure 7 than in measure 6, although their reasons for this are not entirely clear.

In measure 9, both performances have reached their climax and return to the *piano* dynamics of the beginning of the piece. The tempo is quickened a bit for the last measure, but must of course slow down for the final fermata before the beginning of the development section (measure 10). Even though the "fast" performance again reaches a higher tempo in the middle of measure 9 than all other performances (trying to match this measure to its former high tempo, in all likelihood), the fermata is performed in the same tempo chosen by the other artists. This suggests that the artist's divergence from the path chosen by the other performers was of a temporary nature, and that the following section may be performed in a manner similar to the one chosen by different interpretations.

In summary, the analysis of the the dynamics of a performance in conjunction with its respective tempo curve may yield results that are not readily apparent from either of the curves alone. In this example, the tempo and dynamics information suggest that the "fast" performer had a heightened sense of suspense and development towards measure 9 in mind, which is the natural affective result of a joint increase in tempo and loudness. Such a claim can be stated with a much higher degree of confidence when it is substantiated by multiple sources of information than when only one such source is available.

4.3 Chapter Summary

The present chapter has demonstrated the potential and the limit both of theoretical and practical ability of the proposed techniques to capture the essence of a piece's temporal structure. The results are indicative of a good overall performance, in particular when high-quality alignment information between reference and interpretation is available.

The FWC technique was put to pragmatic use in the exemplary evaluation of several musical interpretations of pieces of Western classical music, and produced convincing results that were analyzed using standard musicological criteria. Here, it was shown that the generated tempo curves reflected artistic intentions in the musical shaping of phrases and the highlighting of structurally important events of a piece. In one case, this was substantiated by additional evaluation of dynamics curves generated by making use of the available alignment information. This example in particular demonstrated that the annotation information automatically created as a byproduct of the alignment process can provide useful assistance in tackling tasks involving evaluation or analysis of performance data.

The following chapter extends the results obtained in this chapter by presenting various techniques from related literature dealing with the automated processing of tempo and dynamics information (i.e. data such as produced by our techniques).

Chapter 5

Performance Analysis

I'll play it first and tell you what it is later.

–Miles Davis, 1963

As illustrated by the examples given in Section 4.2, the tempo curves generated by our techniques can be used as a basis for deriving musical statements about specific performances. The field of study concerned with the automation of such analysis processes is appropriately called *performance analysis*. In this chapter, we present some interesting techniques of this field that rely on tempo curve information for the computation of a number of related performance attributes. According to the structural division already mentioned in Section 4.2, these techniques are grouped according to the main focus of their studies, which lay either in the commonalities or the differences among a number of interpretations of one or more pieces.

5.1 Research Focusing on Common Interpretational Traits

One interesting approach that falls into the former category tries to derive elementary *rules of performance* that capture basic principles every performer adheres to [Wid02].[21] This is done without falling back on domain knowledge; instead, the rules are induced empirically from a large data set of piano music using machine learning methods. The employed data set is created specifically for this undertaking: A recording of 13 complete Mozart piano sonatas (about four hours of music overall), performed on a player piano. This enables using direct onset annotation on the note level as input for the learning algorithm.

An example for a rule generated by this algorithm might look like this (in fact, this rule was one of 17 rules produced and accepted for the final result set):

```
Context:
    Two consecutive notes
Precondition:
    Second note has same pitch as first note
Action:
    Play the first note 'staccato'
```

[21] Even though this approach uses data by a single pianist only.

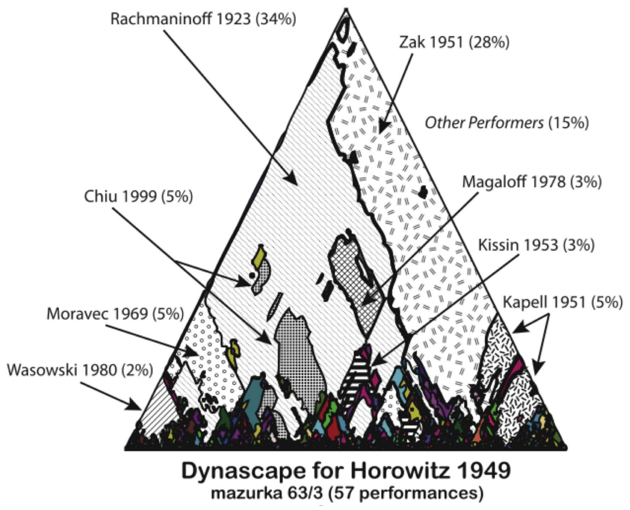

Figure 5.1: Sample Dynascape for Horowitz: Chopin Mazurka 63/3, 1949 performance (reproduced from [Sap08])

The authors call this the "temporal separation" rule, in that two notes of equal pitch become easier to separate for the listener after application of the rule. An empirical evaluation of this approach shows that the rules seem to capture basic performance principles very well. In fact, they even perform better on some test data than on the training data, although the test data consists of Chopin pieces, while the training was done on Mozart sonatas.

A different approach focusing on the comparative analysis of multiple performances at once uses an innovative visualization method encoding similarity aspects of these performances [Sap07, Sap08]. This visualization is called *scape plot*. The name derives from landscape paintings, where, according to the author, "the interesting parts lie somewhere in the middle-ground".

One example plot is depicted in Figure 5.1. It shows the correspondence of loudness features of one particular recording of a piece to other recordings of the same piece (of course, different features such as tempo can be plotted as well). For this so-called *dynascape*, the reference recording was done by Horowitz in 1949. The plot shows the closest matches for Horowitz's dynamic choices in different segmentations of the original—e.g., the top point corresponds to the overall best match, while points in the middle of the plot correspond to matches of segments that have half the length of the complete piece.

While the reader is referred to the original paper for a detailed explanation of the segmentation procedure and results, one can intuitively conclude from Figure 5.1 that the first half of the reference recording is best matched by a Rachmaninoff recording of this piece, while the second half is better matched by Zak (1951).

The information used for scape plot generation is also evaluated to yield a non-visual, computational similarity metric for different performances of a given piece. Empirical testing of this metric shows that it successfully ranks different interpretations of one piece by the same artist as very similar. In the same work, the author introduces an interesting concept: *Residual tempo*, the result of subtracting a smoothed tempo curve from a high-resolution tempo curve.

Thus, the residual tempo describes the local, small-scale variations of a player without the influences of larger-scale phrasing concerns.

5.2 Research Focusing on Systematic Differences Between Interpretations

One technique that concentrates on systematic differences between artists (even across different pieces) was developed in an attempt to formally specify the basic *musical gestures* individual artists are prone to use [WDG$^+$03, Wid05]. Musical gestures here are defined in a three-dimensional space of tempo-loudness variations over time. This movement is visualized in the so-called *performance worm* (Fig. 5.2), where variations along the horizontal axis indicate tempo changes and variations on the vertical axis indicate changes in loudness [LG03]. Progress along the performance's time dimension is indicated by including depth information: Recent information is displayed very clearly, while older data points begin to blur and become smaller as if fading into the distance. Thus, the "worm" seems to move towards the viewer.

In order to get the necessary data for a successful classification of salient aspects which define an artist's playing style, a large collection of over 500 professional CD recordings was annotated at the beat level using a semi-automatic approach. This data was then analyzed to derive performance worms for the individual pieces, and the worms were divided into small segments of about two bars length. After normalization of these segments, they were clustered according to their shape, such that 24 *prototype shapes* emerged. The concluding stage consisted of rating all artists according to their frequency of use of each of these shapes.

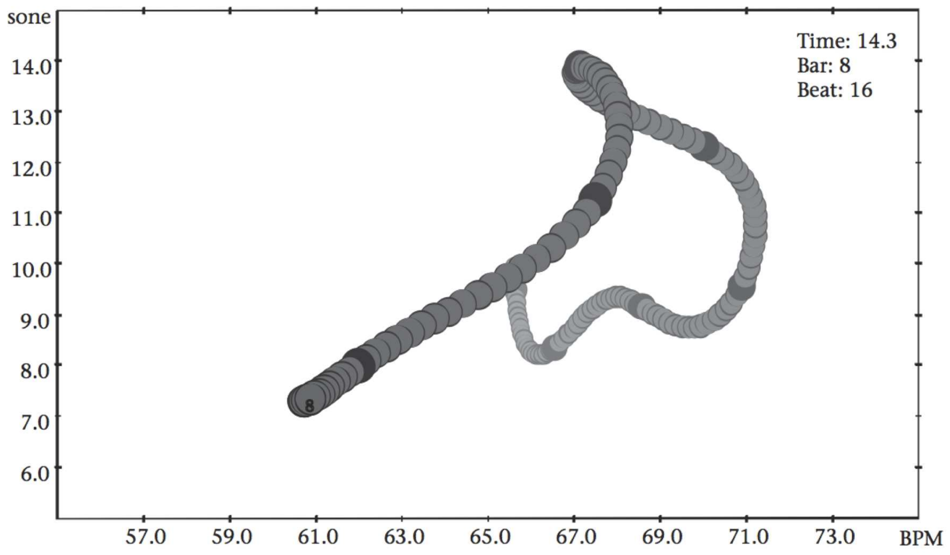

Figure 5.2: Performance Worm for Horowitz: Robert Schumann, Kinderszenen op. 15, "Von fremden Ländern und Menschen", measures 1–8 (reproduced from [WDG$^+$03])

The results show that this method is useful for computing a musically meaningful clustering of artists by using their respective rating. Another idea, the derivation of "typical gestures" of individual performers by means of searching for especially discriminating combinations of prototype shapes,[22] does not seem to produce convincing results: The outcomes are affected heavily by artifacts stemming from the chosen processing approach. Here, single musical shapes are codified as short strings of letters, where each letter designates a specific prototype shape. Frequency analysis of these strings is performed on a purely lexical basis. However, this neglects similarity relations between single shapes, and so the results do not lead to any discovery of musically meaningful gestures.

[22]Meaning shape combinations that are employed particularly often by a single artist, but neglected by other artists.

Chapter 6

Summary

I like talking about ideas. I find them terribly interesting.

–Brian Eno

This chapter briefly summarizes the main points of the work, showing what has been achieved and how it relates to the field of performance analysis as a whole. We then discuss some opportunities for future work that might build on the results established here.

6.1 Contributions

In this work, we were concerned with the automatic extraction of tempo information from expressive music recordings. Our goal in particular was to explore the potential of automatic performance annotations gained by an alignment of symbolic MIDI data with concrete audio recordings. In the course of this investigation, we have introduced three different techniques for the automatic computation of tempo curves which describe the tempo structure of such a recording. For this, we have shown how to synchronize two pieces of music using the DTW algorithm, and explained how the alignment information obtained in this way can be used as a basis for further algorithmic processing. The algorithms we presented built upon three ideas: The first technique (FW) uses a fixed window size to compute average slopes of the warping path. The second technique (AW) computes such slopes of the warping path between onset locations. Finally, the third technique (FWC) uses onset locations to correct the warping path and remove synchronization artifacts, and then apply the FW approach on the corrected warping path to compute the tempo curve.

We have given an evaluation of the different techniques which strongly suggested that of the presented approaches, the third approach delivers the best results; best in terms both of accuracy and robustness. However, all three algorithms are vulnerable to synchronization errors that may distort the warping path. We have shown the conditions under which such errors are likely to occur, and the conditions under which the techniques work well. For the latter case, we have given an explicit recommendation for practical deployment of the approach: FWC with a setting of $w = 3$. Using such a configuration, we explored some musical examples and found that the technique produced results that were in accordance with our musical intuition.

The current state of the art in performance analysis research is to generate tempo and dynamics data on an interpretation using semi-automatic annotation for feature extraction. This data can be processed in a variety of ways, some of which we presented in an overview of related work. This work has introduced and discussed an attempt to extract tempo data from an interpretation in a fully automatic fashion, using only the performance and a digital representation of the score as a reference. Evaluation results indicate good performance, which makes this technique a valid and worthwhile target of further studies.

6.2 Future Work

While the performance of the FWC technique was satisfactory for benign cases, synchronization errors dominated results in the quantitative analysis of scenarios that put harder strains on the alignment procedure. The research opportunities arising from this are twofold: The obvious approach would be to try to increase synchronization quality, or barring that, at least to give an estimation of synchronization accuracy. This estimate could be used to regulate the window size of the tempo curve computation technique dynamically, increasing it in regions of low synchronization accuracy and decreasing it for results of higher precision. This way, quality of the computed tempo curve could be improved due to feedback from the synchronization algorithm.

However, feedback can be given in the inverse direction as well: Since we know that extreme tempo variations are quite unrealistic and thus improbable, synchronization can likely be improved by incorporating information about the locations of the tempo curve where such extreme variations occur. This gives the algorithm an opportunity to reconsider the computed warping path and possibly correct the synchronization error. On the technical side, this could be implemented in the DTW computation phase by using a modified cost measure that penalized high variability in the tempo curve, or even earlier during feature computation. Since extraction and processing of features is computationally costly, it would be reasonable to recompute features for single segments only where one suspected a synchronization failure. For this recomputation step, both higher and lower feature resolutions may produce better results, depending on the specific setting: Higher resolutions may incorporate information that was previously blurred out, while lower resolutions may discard artifacts that hindered proper synchronization.

A different potentially worthwhile approach might lie in further experimentation with the AW algorithm. While performance of this algorithm has not been on par with the FWC approach, it too might profit from a dynamic window size. As in the case discussed above, the respective window size might be determined by synchronization error estimations—however, if such information should not be available, an interesting possibility would be to design a heuristic based on note frequency in a specific time interval. For example, if two notes are played within the time interval of one second, synchronization for this interval may be better than if ten notes were played in the same interval. Observation of potential correlations between error rate and tempo curve accuracy could help to confirm or reject this hypothesis. Another related point is the integration of a greater variety of features into the current approaches (in particular note offsets and pedalling) that could be used to yield results of even higher precision.

In the current approach, linear interpolation has been used exclusively for the computation of tempo curves. A different venue to explore might lie in the choice of different interpolation schemes, such as with polynomial or spline interpolation. One could also try to smooth the warping path using techniques other than interonset correction, e.g. Gaussian smoothing or Fourier smoothing. However, this would neglect the valuable information gained by determining note onset locations, so a combination of these approaches might be worth looking into.

As has already been mentioned, tempo curves are not the only information that can be extracted from a performance using a reference score alignment as a basis. Dynamics curves have been briefly mentioned but should be investigated in greater detail. Making use e.g. of note offset features, one could try to determine the agogics of an interpretation.

Chapter 5 already discussed some possible applications where tempo curves are processed in analysis steps of higher musical abstraction. Of course, there is great potential here, and not all possible applications can be listed. One exciting prospect is the ability to automatically correlate tempo data with semantic events of the score—melodic highlights, harmonic surprises, even phrase structures. Semantically motivated recommender systems are another option: “People who liked the playing style of artist X may also enjoy artist Y”. This can be a great advantage when the meta-data used for such applications today is not available, e.g. in the case of an artist who is not yet established on the market. Even recognizing individual artists by their playing style may become feasible in the future, although much work will have to be invested to let this vision be implemented in a real-world system.

Appendix A

Result Tables

This chapter presents the full tables of the ground truth evaluation results discussed in Section 4.1. As mentioned there, the quoted value for parameter w in each table designates the input for both the FW and the FWC technique, so that one can directly read off the respective impact of interonset correction of the warping path for the respective scenario.

RWC ID (Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
	mean	max	std	mean	max	std	mean	max	std
C025 (Bach, piano)	12.11	320.59	23.83	9.04	254.69	18.09	6.80	117.72	13.32
C028 (Beethoven, piano)	10.54	308.48	18.53	17.39	320.53	25.23	6.74	234.49	13.46
C031 (Chopin, piano)	11.80	400.39	29.22	9.05	210.47	13.86	5.17	104.78	8.19
C032 (Chopin, piano)	5.86	132.06	12.85	9.76	134.19	10.73	2.72	28.97	3.34
C029 (Schumann, piano)	21.28	512.58	47.80	6.16	72.17	8.50	5.22	54.57	7.88
Average over piano	12.32	334.82	26.45	10.28	198.41	15.28	5.33	108.10	9.24
C003 (Beethoven, orchestra)	18.67	406.22	24.80	31.40	354.81	38.17	14.19	253.67	18.49
C015 (Borodin, strings)	11.95	227.24	15.43	17.91	337.82	26.63	10.00	227.08	13.32
C022 (Brahms, orchestra)	6.48	55.75	6.97	12.79	164.61	16.35	4.26	51.15	5.34
C044 (Rimski-Korsakov, flute/piano)	2.66	25.92	2.62	11.78	131.70	12.29	2.43	22.44	2.42
C048 (Schubert, voice/piano)	13.58	485.63	26.82	10.70	262.11	16.09	6.00	125.89	9.31
Average over non-piano	10.67	240.15	15.33	16.92	250.21	21.91	7.37	136.05	9.77
J001 (Nakamura, piano)	5.90	74.73	7.58	6.82	122.30	8.47	3.04	33.42	3.31
J038 (HH Band, big band)	8.75	150.97	12.35	15.61	280.38	23.09	6.85	134.11	10.07
J041 (Umitsuki Quart., sax/bass/perc.)	7.10	124.40	9.73	19.02	268.37	24.15	6.27	110.09	8.77
P031 (Nagayama, electronic)	9.74	197.52	14.13	30.86	340.32	36.15	9.26	190.17	13.70
P093 (Burke, voice/guitar)	13.56	465.04	26.72	26.70	398.93	42.19	12.09	353.76	23.37
Average over jazz/pop	9.01	202.53	14.10	19.80	282.06	26.81	7.50	164.31	11.84
Average over all	10.66	259.17	18.63	15.67	243.56	21.33	6.74	136.15	10.29

Table A.1: Results for Scenario 1, $w = 1\,\mathrm{s}$, $w_{\mathrm{ioi}} = 2$

RWC ID (Comp./Interp., Instr.)	**FW** mean	max	std	**AW** mean	max	std	**FWC** mean	max	std
C025 (Bach, piano)	7.11	79.98	13.74	6.14	69.69	12.13	4.82	54.07	10.41
C028 (Beethoven, piano)	4.32	115.06	8.14	6.98	261.04	13.29	3.28	113.09	7.48
C031 (Chopin, piano)	5.07	127.01	10.81	4.24	59.41	7.06	2.84	37.30	4.94
C032 (Chopin, piano)	2.84	35.19	5.01	3.92	32.37	5.44	1.83	20.95	2.73
C029 (Schumann, piano)	8.97	120.66	14.42	6.20	46.40	7.44	3.49	36.08	5.64
Average over piano	5.66	95.58	10.42	5.50	93.78	9.07	3.25	52.30	6.24
C003 (Beethoven, orchestra)	7.24	75.18	8.83	11.84	222.06	15.91	5.91	71.31	7.36
C015 (Borodin, strings)	4.63	64.59	5.84	6.77	207.45	12.45	4.05	56.03	5.28
C022 (Brahms, orchestra)	2.58	19.78	2.69	3.44	39.59	4.10	1.85	16.58	2.15
C044 (Rimski-Korsakov, flute/piano)	1.42	17.38	1.93	3.09	56.64	3.70	1.39	17.12	1.90
C048 (Schubert, voice/piano)	4.97	56.53	6.70	4.43	78.37	6.24	2.88	41.87	4.19
Average over non-piano	4.17	46.69	5.20	5.91	120.82	8.48	3.22	40.58	4.17
J001 (Nakamura, piano)	2.41	28.90	3.15	2.26	43.09	2.67	1.52	18.31	2.01
J038 (HH Band, big band)	3.51	35.53	4.36	4.53	74.97	6.80	2.94	30.41	3.98
J041 (Umitsuki Quart., sax/bass/perc.)	2.99	40.93	4.23	5.33	132.45	8.01	2.70	34.89	4.00
P031 (Nagayama, electronic)	4.29	58.85	6.47	12.89	339.23	20.04	4.12	58.01	6.43
P093 (Burke, voice/guitar)	5.15	56.39	7.27	9.01	276.67	16.38	4.71	56.38	7.07
Average over jazz/pop	3.67	44.12	5.10	6.80	173.28	10.78	3.20	39.60	4.70
Average over all	4.50	62.13	6.90	6.07	129.30	9.44	3.22	44.16	5.04

Table A.2: Results for Scenario 1, $w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 10$

RWC ID (Comp./Interp., Instr.)	**FW** mean	max	std	**AW** mean	max	std	**FWC** mean	max	std
C025 (Bach, piano)	6.37	69.25	12.43	7.33	70.99	12.76	4.71	54.65	10.02
C028 (Beethoven, piano)	3.75	77.14	6.78	5.60	135.54	9.69	3.01	75.89	6.43
C031 (Chopin, piano)	4.24	62.80	7.65	5.02	45.98	7.04	2.70	32.77	4.67
C032 (Chopin, piano)	2.91	28.97	4.62	3.67	39.63	6.34	2.14	25.01	3.30
C029 (Schumann, piano)	7.24	72.14	10.18	9.30	55.69	9.71	3.37	34.30	5.46
Average over piano	4.90	62.06	8.33	6.19	69.56	9.11	3.19	44.53	5.98
C003 (Beethoven, orchestra)	5.80	67.07	6.96	7.60	84.79	9.36	4.81	64.72	5.86
C015 (Borodin, strings)	3.94	38.37	4.69	5.52	106.46	7.24	3.53	37.57	4.31
C022 (Brahms, orchestra)	2.35	19.52	2.64	3.07	21.20	3.20	1.86	18.09	2.40
C044 (Rimski-Korsakov, flute/piano)	1.67	20.98	2.62	1.79	20.27	1.89	1.66	20.71	2.61
C048 (Schubert, voice/piano)	4.01	39.78	5.04	5.24	46.66	6.80	2.61	31.64	3.73
Average over non-piano	3.55	37.14	4.39	4.65	55.88	5.70	2.89	34.55	3.78
J001 (Nakamura, piano)	2.25	23.94	2.89	2.65	34.76	3.69	1.60	21.67	2.38
J038 (HH Band, big band)	3.16	31.51	3.87	3.46	31.67	4.00	2.73	27.93	3.67
J041 (Umitsuki Quart., sax/bass/perc.)	2.72	38.06	3.99	3.46	57.82	5.13	2.49	34.32	3.84
P031 (Nagayama, electronic)	3.73	46.75	5.75	7.83	111.68	10.60	3.61	46.15	5.73
P093 (Burke, voice/guitar)	3.87	35.88	5.04	5.23	81.89	8.00	3.62	33.62	4.87
Average over jazz/pop	3.15	35.23	4.31	4.53	63.56	6.29	2.81	32.74	4.10
Average over all	3.87	44.81	5.68	5.12	63.00	7.03	2.96	37.27	4.62

Table A.3: Results for Scenario 1, $w = 4\,\mathrm{s}$, $w_{\mathrm{ioi}} = 20$

RWC ID	(Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
		mean	max	std	mean	max	std	mean	max	std
C025	(Bach, piano)	6.22	64.41	11.36	9.21	72.73	13.05	5.17	58.92	10.29
C028	(Beethoven, piano)	3.79	53.99	5.99	5.76	114.58	9.55	3.38	54.90	5.94
C031	(Chopin, piano)	3.98	48.91	5.89	6.63	56.82	8.72	3.15	32.71	4.98
C032	(Chopin, piano)	3.82	34.99	5.38	3.82	41.12	6.75	3.44	34.87	5.13
C029	(Schumann, piano)	5.82	36.90	6.72	13.59	72.32	12.59	3.88	39.43	5.92
Average over piano		4.73	47.84	7.07	7.80	71.52	10.13	3.80	44.17	6.45
C003	(Beethoven, orchestra)	5.02	49.87	5.84	6.77	74.23	8.01	4.29	47.97	5.13
C015	(Borodin, strings)	3.83	41.08	4.73	6.53	73.35	7.62	3.62	40.68	4.64
C022	(Brahms, orchestra)	2.87	26.65	3.87	3.69	25.61	3.98	2.59	26.35	3.80
C044	(Rimski-Korsakov, flute/piano)	2.73	30.27	4.37	1.50	16.28	1.71	2.74	30.21	4.36
C048	(Schubert, voice/piano)	3.81	32.47	4.56	6.62	54.61	8.35	3.13	34.99	4.36
Average over non-piano		3.65	36.07	4.67	5.02	48.82	5.93	3.27	36.04	4.46
J001	(Nakamura, piano)	2.80	31.03	3.85	3.60	33.52	4.63	2.37	30.86	3.77
J038	(HH Band, big band)	3.52	33.68	4.55	3.85	31.92	4.33	3.29	33.64	4.50
J041	(Umitsuki Quart., sax/bass/perc.)	3.12	39.28	4.76	3.27	45.36	4.67	2.97	38.20	4.69
P031	(Nagayama, electronic)	4.14	44.81	6.18	5.63	76.86	7.53	4.08	45.03	6.17
P093	(Burke, voice/guitar)	3.79	35.06	4.79	4.23	36.71	5.12	3.67	35.06	4.73
Average over jazz/pop		3.47	36.77	4.83	4.12	44.87	5.26	3.28	36.56	4.77
Average over all		3.95	40.23	5.52	5.65	55.07	7.11	3.45	38.92	5.23

Table A.4: Results for Scenario 1, $w = 6\,\mathrm{s}$, $w_{\mathrm{ioi}} = 30$

RWC ID	(Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
		mean	max	std	mean	max	std	mean	max	std
C025	(Bach, piano)	7.19	98.00	12.94	5.36	51.19	7.04	3.69	37.85	5.27
C028	(Beethoven, piano)	6.10	93.07	8.82	8.44	239.31	14.04	4.47	77.37	7.11
C031	(Chopin, piano)	7.84	269.76	18.18	6.05	73.06	8.69	4.37	62.36	7.03
C032	(Chopin, piano)	4.83	82.38	9.86	5.40	54.86	7.67	3.73	47.72	7.00
C029	(Schumann, piano)	13.10	116.40	18.49	8.24	74.16	9.98	5.38	61.58	8.38
Average over piano		7.81	131.92	13.66	6.70	98.52	9.48	4.33	57.38	6.96
C003	(Beethoven, orchestra)	11.09	156.97	14.30	14.61	278.33	19.32	9.22	105.67	11.59
C015	(Borodin, strings)	7.85	164.92	11.35	10.27	342.02	16.49	7.08	170.75	10.79
C022	(Brahms, orchestra)	4.50	41.84	5.42	5.39	83.34	7.14	3.26	39.09	4.31
C044	(Rimski-Korsakov, flute/piano)	2.47	22.01	2.98	4.28	45.60	4.97	2.33	22.42	2.90
C048	(Schubert, voice/piano)	8.09	129.00	13.26	6.47	96.21	8.72	4.74	97.01	9.53
Average over non-piano		6.80	102.95	9.46	8.20	169.10	11.33	5.33	86.99	7.82
J001	(Nakamura, piano)	4.07	56.37	6.51	3.70	73.22	4.83	2.58	25.22	3.48
J038	(HH Band, big band)	5.70	65.89	7.36	7.08	107.01	10.65	4.66	54.38	6.60
J041	(Umitsuki Quart., sax/bass/perc.)	5.80	146.29	11.83	7.97	242.86	13.57	5.19	107.81	10.40
P031	(Nagayama, electronic)	5.16	41.76	5.47	14.89	278.75	20.44	4.86	40.86	5.34
P093	(Burke, voice/guitar)	7.05	65.28	9.29	9.87	273.43	16.38	6.58	63.89	8.99
Average over jazz/pop		5.56	75.12	8.09	8.70	195.05	13.18	4.77	58.44	6.96
Average over all		6.72	103.33	10.40	7.87	154.22	11.33	4.81	67.60	7.25

Table A.5: Results for Scenario 2, $w = 2\,\mathrm{s}$, $w_{\mathrm{ioi}} = 8$

RWC ID	(Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
		mean	max	std	mean	max	std	mean	max	std
C025	(Bach, piano)	6.52	62.70	10.27	6.60	57.38	8.44	4.08	43.12	5.71
C028	(Beethoven, piano)	5.37	66.42	7.16	7.65	181.11	12.39	4.41	58.60	6.41
C031	(Chopin, piano)	6.51	101.66	10.77	6.98	76.37	9.86	4.47	55.32	6.82
C032	(Chopin, piano)	5.09	65.54	8.50	4.83	54.76	7.76	4.48	46.67	7.41
C029	(Schumann, piano)	10.75	81.94	14.51	10.74	77.95	11.49	5.32	58.35	7.86
Average over piano		6.85	75.65	10.24	7.36	89.51	9.99	4.55	52.41	6.84
C003	(Beethoven, orchestra)	8.64	101.04	10.63	11.42	199.13	13.82	7.57	84.91	9.36
C015	(Borodin, strings)	6.49	89.18	8.64	9.47	241.75	13.33	6.16	90.20	8.50
C022	(Brahms, orchestra)	4.01	32.45	4.66	4.57	66.58	5.55	3.23	31.71	4.08
C044	(Rimski-Korsakov, flute/piano)	3.11	30.24	4.20	3.20	26.77	3.41	3.04	30.08	4.18
C048	(Schubert, voice/piano)	6.78	94.62	10.49	7.45	96.35	10.11	4.76	76.31	8.41
Average over non-piano		5.81	69.50	7.72	7.22	126.12	9.25	4.95	62.64	6.91
J001	(Nakamura, piano)	4.01	51.63	5.72	4.19	58.96	5.43	3.10	33.90	4.44
J038	(HH Band, big band)	4.93	45.60	6.08	5.97	76.97	8.11	4.29	40.38	5.71
J041	(Umitsuki Quart., sax/bass/perc.)	5.44	113.62	10.72	6.33	164.15	11.71	4.99	89.07	9.56
P031	(Nagayama, electronic)	4.80	43.58	5.35	10.60	189.51	13.78	4.63	43.99	5.31
P093	(Burke, voice/guitar)	5.72	54.28	6.91	7.82	94.00	10.79	5.39	51.42	6.77
Average over jazz/pop		4.98	61.74	6.96	6.98	116.72	9.96	4.48	51.75	6.36
Average over all		5.88	68.97	8.31	7.19	110.78	9.73	4.66	55.60	6.70

Table A.6: Results for Scenario 2, $w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 12$

RWC ID	(Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
		mean	max	std	mean	max	std	mean	max	std
C025	(Bach, piano)	6.89	60.35	9.59	8.48	68.25	10.41	5.26	51.85	6.88
C028	(Beethoven, piano)	5.91	57.66	7.38	7.47	184.30	11.85	5.25	56.90	7.16
C031	(Chopin, piano)	6.71	70.03	9.07	8.14	76.90	10.89	5.35	59.37	7.39
C032	(Chopin, piano)	6.12	56.27	8.52	5.10	54.75	8.63	5.81	52.50	8.28
C029	(Schumann, piano)	9.75	72.76	12.61	12.99	86.69	13.16	5.94	59.49	8.08
Average over piano		7.08	63.41	9.43	8.44	94.18	10.99	5.52	56.02	7.56
C003	(Beethoven, orchestra)	8.05	80.52	9.39	10.24	114.43	11.52	7.38	72.09	8.68
C015	(Borodin, strings)	6.78	84.94	8.51	9.88	159.13	12.17	6.56	85.69	8.67
C022	(Brahms, orchestra)	4.56	35.73	5.35	4.63	52.65	4.89	4.08	35.57	5.12
C044	(Rimski-Korsakov, flute/piano)	4.45	38.49	5.86	2.74	23.21	2.95	4.41	37.94	5.86
C048	(Schubert, voice/piano)	6.74	81.17	9.63	8.63	95.47	11.18	5.47	70.99	8.53
Average over non-piano		6.11	64.17	7.75	7.23	88.98	8.54	5.58	60.46	7.37
J001	(Nakamura, piano)	4.88	47.69	6.44	4.90	57.49	6.22	4.28	43.26	5.92
J038	(HH Band, big band)	5.43	47.48	6.41	5.92	59.16	7.28	4.99	47.10	6.33
J041	(Umitsuki Quart., sax/bass/perc.)	6.08	93.69	10.28	5.82	142.64	10.75	5.74	74.32	9.44
P031	(Nagayama, electronic)	5.38	52.30	6.39	8.68	101.83	9.80	5.27	52.30	6.39
P093	(Burke, voice/guitar)	5.86	44.86	6.80	6.86	61.83	8.55	5.64	45.92	6.67
Average over jazz/pop		5.53	57.20	7.27	6.44	84.59	8.52	5.19	52.58	6.95
Average over all		6.24	61.60	8.15	7.37	89.25	9.35	5.43	56.35	7.29

Table A.7: Results for Scenario 2, $w = 4\,\mathrm{s}$, $w_{\mathrm{ioi}} = 16$

RWC ID (Comp./Interp., Instr.)	**FW** mean	max	std	**AW** mean	max	std	**FWC** mean	max	std
C025 (Bach, piano)	9.32	129.97	15.17	6.66	80.98	10.09	4.90	74.42	8.26
C028 (Beethoven, piano)	17.20	542.30	39.63	21.58	475.24	47.11	13.61	517.29	36.30
C031 (Chopin, piano)	24.92	313.20	46.98	20.32	236.52	36.62	20.09	303.86	41.87
C032 (Chopin, piano)	7.73	119.86	16.50	10.41	117.44	17.06	5.13	72.36	10.09
C029 (Schumann, piano)	62.71	672.71	103.29	47.90	304.08	61.59	46.67	396.93	70.33
Average over piano	24.38	355.61	44.32	21.37	242.85	34.49	18.08	272.97	33.37
C003 (Beethoven, orchestra)	32.61	455.72	48.42	38.55	478.94	58.05	28.77	435.07	46.11
C015 (Borodin, strings)	22.93	533.19	45.46	23.15	388.05	39.26	19.85	334.75	34.88
C022 (Brahms, orchestra)	19.30	123.60	27.78	22.14	229.22	35.12	17.67	121.30	27.25
C044 (Rimski-Korsakov, flute/piano)	2.60	31.20	3.20	6.44	63.96	8.24	2.30	30.28	3.01
C048 (Schubert, voice/piano)	53.15	640.39	86.62	49.25	431.62	80.49	48.59	603.79	82.80
Average over non-piano	26.12	356.82	42.29	27.91	318.36	44.23	23.43	305.04	38.81
J001 (Nakamura, piano)	15.09	178.90	31.44	14.66	191.09	30.83	13.07	179.17	30.58
J038 (HH Band, big band)	14.89	139.19	22.87	17.31	279.79	28.54	13.45	139.05	22.69
J041 (Umitsuki Quart., sax/bass/perc.)	44.50	397.81	58.57	48.27	419.48	60.86	43.84	397.71	58.25
P031 (Nagayama, electronic)	52.81	346.64	65.48	61.84	392.52	67.47	52.42	346.10	65.58
P093 (Burke, voice/guitar)	23.18	493.03	44.29	28.60	541.15	52.45	22.29	419.56	43.87
Average over jazz/pop	30.09	311.11	44.53	34.14	364.80	48.03	29.01	296.32	44.19
Average over all	26.86	341.18	43.71	27.81	308.67	42.25	23.51	291.44	38.79

Table A.8: Results for Scenario 3, $w = 2\,\mathrm{s}$, $w_{\mathrm{ioi}} = 8$

RWC ID (Comp./Interp., Instr.)	**FW** mean	max	std	**AW** mean	max	std	**FWC** mean	max	std
C025 (Bach, piano)	7.82	80.13	12.25	7.24	72.47	10.75	4.47	55.34	7.04
C028 (Beethoven, piano)	13.72	402.44	31.13	18.50	477.71	41.88	11.74	374.75	30.38
C031 (Chopin, piano)	22.16	308.71	42.68	20.31	163.83	33.77	18.97	248.50	38.68
C032 (Chopin, piano)	6.36	105.20	13.23	8.95	111.15	16.35	4.88	63.17	9.88
C029 (Schumann, piano)	54.20	721.95	87.02	48.67	305.10	57.92	44.26	401.58	65.67
Average over piano	20.85	323.68	37.26	20.73	226.05	32.13	16.87	228.67	30.33
C003 (Beethoven, orchestra)	27.29	267.65	42.65	33.83	466.15	53.36	25.14	256.86	42.19
C015 (Borodin, strings)	18.30	467.90	35.56	20.39	379.35	34.58	16.44	240.85	29.78
C022 (Brahms, orchestra)	17.59	116.29	26.18	19.73	213.11	31.25	16.61	115.08	25.83
C044 (Rimski-Korsakov, flute/piano)	2.36	26.72	3.42	4.64	46.91	5.74	2.25	27.06	3.39
C048 (Schubert, voice/piano)	49.53	592.25	78.96	48.16	380.38	75.86	46.78	497.78	77.05
Average over non-piano	23.01	294.16	37.35	25.35	297.18	40.16	21.44	227.53	35.65
J001 (Nakamura, piano)	14.11	181.14	30.54	14.78	181.85	30.65	12.91	180.27	30.33
J038 (HH Band, big band)	12.92	121.70	20.77	14.54	176.81	23.45	11.91	119.02	20.52
J041 (Umitsuki Quart., sax/bass/perc.)	42.63	351.31	56.64	45.41	377.11	58.89	42.11	351.30	56.50
P031 (Nagayama, electronic)	51.05	353.29	62.75	58.13	384.23	66.27	50.87	352.88	62.72
P093 (Burke, voice/guitar)	19.90	259.56	38.82	24.03	333.43	44.00	19.24	258.50	38.80
Average over jazz/pop	28.12	253.40	41.90	31.38	290.69	44.65	27.41	252.39	41.77
Average over all	24.00	290.41	38.84	25.82	271.31	38.98	21.91	236.19	35.92

Table A.9: Results for Scenario 3, $w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 12$

RWC ID (Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
	mean	max	std	mean	max	std	mean	max	std
C025 (Bach, piano)	7.19	63.07	10.95	10.22	100.37	15.47	4.73	53.69	7.44
C028 (Beethoven, piano)	12.36	321.83	27.47	16.44	462.16	35.83	10.93	321.04	27.10
C031 (Chopin, piano)	21.26	263.44	39.60	22.07	166.24	34.35	18.84	226.94	36.64
C032 (Chopin, piano)	6.42	88.68	12.64	8.46	98.99	16.65	5.33	76.60	10.57
C029 (Schumann, piano)	48.35	551.74	70.03	52.53	272.87	56.33	42.38	406.05	60.13
Average over piano	19.12	257.75	32.14	21.94	220.13	31.73	16.44	216.87	28.38
C003 (Beethoven, orchestra)	25.01	246.51	40.66	29.54	443.10	46.46	23.49	245.68	40.53
C015 (Borodin, strings)	16.70	250.94	30.76	19.77	249.28	30.76	15.48	198.76	28.53
C022 (Brahms, orchestra)	17.37	109.90	25.78	18.32	150.66	26.52	16.61	109.77	25.63
C044 (Rimski-Korsakov, flute/piano)	2.77	37.98	4.80	3.28	29.65	4.05	2.71	38.16	4.82
C048 (Schubert, voice/piano)	47.75	379.61	73.14	48.74	329.73	71.38	46.11	376.98	73.38
Average over non-piano	21.92	204.99	35.03	23.93	240.48	35.84	20.88	193.87	34.58
J001 (Nakamura, piano)	14.28	178.94	30.38	15.67	172.10	30.62	13.46	176.90	30.41
J038 (HH Band, big band)	12.27	116.37	20.12	13.49	140.52	21.11	11.51	113.46	19.97
J041 (Umitsuki Quart., sax/bass/perc.)	42.07	287.70	54.91	43.36	350.61	57.27	41.66	287.26	54.90
P031 (Nagayama, electronic)	50.07	274.93	59.09	54.96	344.35	64.56	49.96	274.65	59.16
P093 (Burke, voice/guitar)	18.58	222.80	36.82	20.37	230.90	38.83	18.18	223.66	36.83
Average over jazz/pop	27.45	216.15	40.26	29.57	247.69	42.48	26.95	215.19	40.25
Average over all	22.83	226.30	35.81	25.15	236.10	36.68	21.42	208.64	34.40

Table A.10: Results for Scenario 3, $w = 4\,\mathrm{s}$, $w_{\mathrm{ioi}} = 20$

RWC ID (Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
	mean	max	std	mean	max	std	mean	max	std
C025 (Bach, piano)	9.13	83.08	13.49	15.03	113.30	19.91	7.96	83.92	13.00
C028 (Beethoven, piano)	12.66	167.73	22.85	16.39	346.97	32.98	12.09	167.27	22.78
C031 (Chopin, piano)	22.24	175.85	34.90	25.55	177.70	36.34	21.15	171.42	34.78
C032 (Chopin, piano)	8.51	114.69	15.23	8.42	97.97	17.10	8.38	113.45	15.16
C029 (Schumann, piano)	44.16	297.26	52.33	59.91	293.09	57.41	42.23	285.68	52.40
Average over piano	19.34	167.72	27.76	25.06	205.81	32.75	18.36	164.35	27.62
C003 (Beethoven, orchestra)	24.68	224.28	39.79	28.47	266.32	42.67	23.88	224.42	39.80
C015 (Borodin, strings)	17.36	219.35	29.47	22.63	210.74	32.99	17.00	218.85	29.59
C022 (Brahms, orchestra)	19.24	121.15	27.45	19.00	120.04	25.87	18.90	121.09	27.50
C044 (Rimski-Korsakov, flute/piano)	5.92	62.59	9.51	2.82	32.44	3.94	5.90	63.13	9.52
C048 (Schubert, voice/piano)	47.31	314.58	67.02	49.74	310.09	67.71	46.77	311.05	67.28
Average over non-piano	22.90	188.39	34.65	24.53	187.93	34.64	22.49	187.71	34.74
J001 (Nakamura, piano)	17.53	162.76	31.58	18.32	172.02	31.66	17.19	162.67	31.68
J038 (HH Band, big band)	14.04	133.04	21.87	14.51	126.23	21.38	13.76	133.73	21.85
J041 (Umitsuki Quart., sax/bass/perc.)	42.62	249.58	52.73	42.65	298.61	55.28	42.36	249.65	52.79
P031 (Nagayama, electronic)	50.56	196.71	55.17	53.23	334.47	63.49	50.54	196.58	55.21
P093 (Burke, voice/guitar)	19.62	206.72	35.75	19.38	214.48	36.36	19.41	206.33	35.78
Average over jazz/pop	28.87	189.76	39.42	29.62	229.16	41.63	28.65	189.79	39.46
Average over all	23.70	181.96	33.94	26.40	207.63	36.34	23.17	180.62	33.94

Table A.11: Results for Scenario 3, $w = 7\,\mathrm{s}$, $w_{\mathrm{ioi}} = 30$

RWC ID (Comp./Interp., Instr.)	**FW** mean	max	std	**AW** mean	max	std	**FWC** mean	max	std
C025 (Bach, piano)	26.02	543.55	57.16	13.16	153.34	25.38	13.51	258.37	24.84
C028 (Beethoven, piano)	16.38	466.21	30.03	12.83	357.34	25.18	10.64	315.51	21.32
C031 (Chopin, piano)	18.32	518.24	43.32	7.66	151.73	14.47	7.70	241.97	14.89
C032 (Chopin, piano)	13.76	372.64	43.23	6.14	112.71	9.63	4.80	109.09	9.77
C029 (Schumann, piano)	29.36	376.35	53.78	9.78	116.77	18.65	8.01	103.24	14.03
Average over piano	20.77	455.40	45.50	9.92	178.38	18.66	8.93	205.63	16.97
C003 (Beethoven, orchestra)	25.58	501.63	34.71	22.35	358.00	29.87	19.85	320.51	25.39
C015 (Borodin, strings)	17.83	267.29	22.47	15.11	282.92	22.61	15.22	227.21	19.33
C022 (Brahms, orchestra)	12.02	371.17	25.49	7.81	108.74	11.58	7.45	103.70	10.86
C044 (Rimski-Korsakov, flute/piano)	5.44	84.05	9.14	6.76	103.51	9.22	4.54	82.74	8.17
C048 (Schubert, voice/piano)	21.40	485.06	40.91	7.98	185.51	15.21	8.99	183.20	15.26
Average over non-piano	16.46	341.84	26.55	12.00	207.74	17.70	11.21	183.47	15.80
J001 (Nakamura, piano)	10.15	488.16	21.54	5.04	192.31	10.82	5.06	103.52	8.80
J038 (HH Band, big band)	15.66	464.87	31.79	12.06	171.43	18.49	11.98	151.62	18.12
J041 (Umitsuki Quart., sax/bass/perc.)	11.63	262.42	20.07	12.34	237.20	19.70	10.42	248.34	17.96
P031 (Nagayama, electronic)	18.85	418.99	28.87	29.40	422.94	40.77	17.96	384.71	28.36
P093 (Burke, voice/guitar)	19.94	472.18	35.28	19.12	369.89	32.62	17.04	304.40	28.29
Average over jazz/pop	15.24	421.32	27.51	15.59	278.75	24.48	12.49	238.52	20.31
Average over all	17.49	406.19	33.19	12.50	221.62	20.28	10.88	209.21	17.69

Table A.12: Results for Scenario 4, $w = 1\,\mathrm{s}$, $w_{\mathrm{ioi}} = 6$

RWC ID (Comp./Interp., Instr.)	**FW** mean	max	std	**AW** mean	max	std	**FWC** mean	max	std
C025 (Bach, piano)	22.30	472.55	48.88	13.92	137.80	26.47	13.27	155.57	24.88
C028 (Beethoven, piano)	11.45	280.50	19.31	10.08	326.10	20.24	8.84	211.44	17.62
C031 (Chopin, piano)	13.10	282.97	30.63	8.29	100.82	15.10	6.89	104.48	12.75
C032 (Chopin, piano)	11.45	436.25	34.16	5.29	114.77	10.23	5.66	110.45	12.80
C029 (Schumann, piano)	18.93	427.97	36.15	11.81	107.82	20.49	7.66	96.86	13.70
Average over piano	15.44	380.05	33.83	9.88	157.46	18.51	8.46	135.76	16.35
C003 (Beethoven, orchestra)	16.86	178.16	21.12	17.58	312.54	23.62	14.23	150.67	18.85
C015 (Borodin, strings)	12.14	126.83	16.39	12.76	282.60	19.61	11.06	121.85	15.47
C022 (Brahms, orchestra)	8.16	109.91	13.25	5.95	99.00	10.35	6.13	94.62	10.68
C044 (Rimski-Korsakov, flute/piano)	5.37	82.36	10.78	4.82	85.31	7.92	5.00	82.36	10.65
C048 (Schubert, voice/piano)	13.04	153.30	18.35	7.69	120.53	14.34	7.73	106.38	13.67
Average over non-piano	11.11	130.11	15.98	9.76	179.99	15.17	8.83	111.18	13.86
J001 (Nakamura, piano)	7.17	101.52	12.53	5.13	151.52	11.64	5.21	103.41	10.82
J038 (HH Band, big band)	10.87	185.57	17.89	9.61	104.63	15.25	9.38	121.40	15.41
J041 (Umitsuki Quart., sax/bass/perc.)	8.91	161.37	15.61	9.58	218.77	17.49	8.33	156.80	15.13
P031 (Nagayama, electronic)	14.23	179.13	22.15	22.62	410.39	33.93	13.79	176.53	22.09
P093 (Burke, voice/guitar)	12.99	197.43	18.55	13.83	246.75	22.23	11.90	180.85	17.71
Average over jazz/pop	10.84	165.00	17.34	12.15	226.41	20.11	9.72	147.80	16.23
Average over all	12.46	225.05	22.38	10.60	187.96	17.93	9.01	131.58	15.48

Table A.13: Results for Scenario 4, $w = 2\,\mathrm{s}$, $w_{\mathrm{ioi}} = 10$

RWC ID	(Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
		mean	max	std	mean	max	std	mean	max	std
C025	(Bach, piano)	19.94	226.24	33.76	14.35	139.46	26.61	14.03	143.65	25.84
C028	(Beethoven, piano)	10.61	249.83	17.55	9.66	316.82	19.17	9.16	221.23	16.93
C031	(Chopin, piano)	11.27	207.03	20.02	8.69	97.75	15.30	7.51	105.09	13.36
C032	(Chopin, piano)	10.78	150.33	20.74	5.36	114.22	10.69	7.36	109.39	15.10
C029	(Schumann, piano)	15.17	144.08	20.80	12.77	108.90	20.54	8.24	98.98	14.40
Average over piano		13.55	195.50	22.57	10.17	155.43	18.46	9.26	135.67	17.13
C003	(Beethoven, orchestra)	14.17	135.33	18.06	16.20	267.96	21.53	12.72	128.70	16.94
C015	(Borodin, strings)	11.23	121.94	15.98	12.42	232.36	18.73	10.60	120.05	15.50
C022	(Brahms, orchestra)	7.97	94.72	12.92	5.81	95.26	10.35	6.90	93.00	12.37
C044	(Rimski-Korsakov, flute/piano)	6.59	81.93	12.69	4.34	80.79	7.96	6.37	81.47	12.70
C048	(Schubert, voice/piano)	11.49	110.14	16.13	7.80	108.49	14.40	8.16	106.58	14.35
Average over non-piano		10.29	108.81	15.16	9.31	156.97	14.59	8.95	105.96	14.37
J001	(Nakamura, piano)	7.60	101.23	13.17	5.50	122.36	12.25	6.34	104.67	12.71
J038	(HH Band, big band)	10.21	109.80	15.55	9.37	106.33	15.18	9.33	107.92	15.04
J041	(Umitsuki Quart., sax/bass/perc.)	8.88	119.74	15.53	8.99	221.54	16.93	8.48	121.16	15.33
P031	(Nagayama, electronic)	13.87	136.68	21.33	20.35	409.51	31.16	13.60	134.79	21.38
P093	(Burke, voice/guitar)	11.37	114.69	15.63	12.89	181.09	19.40	10.61	114.95	15.49
Average over jazz/pop		10.39	116.43	16.24	11.42	208.17	18.99	9.67	116.70	15.99
Average over all		11.41	140.25	17.99	10.30	173.52	17.35	9.29	119.44	15.83

Table A.14: Results for Scenario 4, $w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 12$

RWC ID	(Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
		mean	max	std	mean	max	std	mean	max	std
C025	(Bach, piano)	19.06	135.59	29.72	15.44	127.43	26.93	15.08	135.25	26.71
C028	(Beethoven, piano)	11.02	130.43	17.16	9.21	272.77	17.94	10.01	129.60	16.95
C031	(Chopin, piano)	11.04	96.26	16.90	9.80	94.30	16.13	8.60	102.22	14.25
C032	(Chopin, piano)	11.63	111.69	18.88	6.18	117.03	12.34	9.29	111.08	16.96
C029	(Schumann, piano)	14.18	112.33	18.11	15.67	101.49	22.03	9.31	98.53	15.27
Average over piano		13.39	117.26	20.16	11.26	142.60	19.08	10.46	115.34	18.03
C003	(Beethoven, orchestra)	13.38	124.89	17.52	14.97	149.04	19.33	12.47	124.37	16.73
C015	(Borodin, strings)	11.42	122.49	16.51	12.74	150.97	18.03	10.98	122.29	16.18
C022	(Brahms, orchestra)	8.78	93.75	14.05	5.94	89.46	10.98	8.08	92.48	13.87
C044	(Rimski-Korsakov, flute/piano)	8.13	81.98	14.14	4.21	81.36	8.56	7.96	81.80	14.17
C048	(Schubert, voice/piano)	11.28	108.78	16.30	8.71	108.78	15.31	9.11	106.36	15.48
Average over non-piano		10.60	106.38	15.70	9.31	115.92	14.44	9.72	105.46	15.29
J001	(Nakamura, piano)	8.69	103.80	14.52	6.43	116.12	13.50	7.76	103.87	14.36
J038	(HH Band, big band)	10.88	102.96	15.66	9.46	108.80	15.11	10.23	104.49	15.53
J041	(Umitsuki Quart., sax/bass/perc.)	9.76	118.61	16.52	8.66	225.72	16.55	9.47	117.05	16.38
P031	(Nagayama, electronic)	14.26	133.20	21.95	18.01	304.43	27.32	14.09	132.81	21.99
P093	(Burke, voice/guitar)	11.24	113.43	15.65	11.30	123.25	16.63	10.68	113.48	15.58
Average over jazz/pop		10.97	114.40	16.86	10.77	175.66	17.82	10.45	114.34	16.77
Average over all		11.65	112.68	17.57	10.45	144.73	17.11	10.21	111.71	16.69

Table A.15: Results for Scenario 4, $w = 4\,\mathrm{s}$, $w_{\mathrm{ioi}} = 16$

RWC ID (Comp./Interp., Instr.)	**FW**			**AW**			**FWC**		
	mean	max	std	mean	max	std	mean	max	std
C025 (Bach, piano)	51.08	348.22	56.59	47.49	211.05	47.02	41.86	213.55	45.03
C028 (Beethoven, piano)	37.08	392.19	58.52	37.77	439.19	62.00	35.05	390.64	58.60
C031 (Chopin, piano)	60.67	408.90	78.94	55.17	272.65	68.18	53.88	287.83	70.60
C032 (Chopin, piano)	35.41	500.49	69.95	29.66	271.07	57.14	27.55	274.42	52.95
C029 (Schumann, piano)	88.20	642.16	92.49	88.41	320.59	75.89	74.65	395.91	84.78
Average over piano	54.49	458.39	71.30	51.70	302.91	62.05	46.60	312.47	62.39
C003 (Beethoven, orchestra)	44.95	386.81	60.31	47.83	374.92	63.63	42.84	389.13	60.06
C015 (Borodin, strings)	56.19	515.18	75.43	59.66	473.41	71.78	52.18	392.47	68.08
C022 (Brahms, orchestra)	48.80	273.32	52.54	47.57	378.17	55.99	47.64	273.54	52.83
C044 (Rimski-Korsakov, flute/piano)	30.71	173.44	35.65	29.45	192.95	35.00	30.34	172.91	35.74
C048 (Schubert, voice/piano)	69.22	601.92	90.64	67.06	375.06	85.44	66.24	492.55	90.14
Average over non-piano	49.97	390.13	62.91	50.31	358.90	62.37	47.85	344.12	61.37
J001 (Nakamura, piano)	28.10	262.31	49.39	28.92	267.13	50.89	26.30	259.86	49.66
J038 (HH Band, big band)	34.84	273.86	50.79	34.66	271.74	51.45	33.60	273.86	50.46
J041 (Umitsuki Quart., sax/bass/perc.)	67.83	398.69	73.67	66.99	397.32	74.30	66.72	399.84	73.28
P031 (Nagayama, electronic)	91.91	315.15	73.82	92.95	342.43	76.03	91.84	314.18	73.91
P093 (Burke, voice/guitar)	55.93	366.57	76.99	55.01	362.56	77.35	55.00	362.89	77.24
Average over jazz/pop	55.72	323.32	64.93	55.71	328.24	66.00	54.69	322.12	64.91
Average over all	53.39	390.61	66.38	52.57	330.02	63.47	49.71	326.24	62.89

Table A.16: Results for Scenario 5, $w = 3\,\mathrm{s}$, $w_{\mathrm{ioi}} = 20$

Appendix B

Source Code

In this chapter, the headers of selected Matlab [Mat09] functions created during the writing of this thesis are reproduced. The headers contain information about the name of the described function and its input/output behavior. The organization of the headers is according to chronological usage order of the respective function in the tempo curve computation process, starting with feature extraction and continuing with curve computation procedures and auxiliary functions.

Feature Extraction

The `file_to_feature` function is used as a wrapper for several low-level functions that perform feature extraction or loading of precomputed features. In the case of MIDI files, `dirname` and `filename` locate the specific file; for wave files, `filename` denotes the name of the file in question, but `dirname` indicates the directory where precomputed features are stored.

Sample usage:
`[f_pitch, f_peaks] = file_to_feature('features', 'pathetique.wav');`

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: file_to_feature
% Version: 1.0
% Date: 10.10.2008
% Programmer: Meinard Mueller, Verena Konz
%
% Description:
%   Load or compute features for audio and MIDI files
%
% Input:
% - dirname: Directory where the file or features are located
% - filename: Name of the file for which to load/compute features
% - parameter
%          .win_len: Window length used for STMSP feature generation
%          .win_res: Window resolution
%
% Output:
```

```
% - f_pitch: Pitch features (STMSP)
% - f_peaks: Energy peaks for onset computation
% - f_onsets: Precise onsets (only generated in case of MIDI input data)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Warping Path Computation

The `feature_to_warpingpath` function uses methods such as described in [Mül07] and [GME09] to compute warping paths from pitch and onset features extracted in a previous step.

Sample usage:

```
warpingpath = feature_to_warpingpath(f_pitch_reference, f_peaks_reference,
                           f_pitch_interpretation, f_peaks_interpretation);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: feature_to_warpingpath
% Version: 1.0
% Date: 20.03.2009
% Programmer: Meinard Mueller, Sebastian Ewert
%
% Description:
%   Compute a warping path from pitch and onset features using multiscale
%   DTW with DLNCO features
%
% Input:
% - f_pitch_1: Pitch features (STMSP) of the reference audio stream
% - f_peaks_1: Onset features of the reference audio stream
% - f_pitch_2: Pitch features (STMSP) of the performance audio stream
% - f_peaks_2: Onset features of the performance audio stream
%
% Output:
% - warpingpath: A regular warping path between reference and
%   interpretation
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Tempo Curve Computation (FW Approach)

As the name suggests, `warpingpath_to_tempocurve_SlidingWindow` computes a tempo curve from a warping path using a sliding window of fixed size. This size is determined by `parameter.windowSize_sec`. The resulting tempo curve is scaled to the reference time axis (i.e. it has length equal to `warpingpath(end, 1)`).

Sample usage:

```
parameter.windowSize_sec = 3;
```

```
tempocurveFW = warpingpath_to_tempocurve_SlidingWindow(warpingpath,
                                                        parameter);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: warpingpath_to_tempocurve_SlidingWindow
% Version: 1.0
% Date: 6.4.2009
% Programmer: Meinard Mueller, Verena Konz
%
% Description:
%   Compute tempo curve from warping path using a fixed-width sliding
%   window approach
%
% Input:
% - warpingpath: A regular warping path between reference and
%   interpretation of some piece of music
% - parameter
%           .vis_warpingpath: If true, visualize warping path together
%                              with generated tempo curve
%           .vis_tempocurve: If true, plot the generated tempo curve
%           .windowSize_sec: Determines the window size of the averaging
%                            window for curve computation
%
% Output:
% - tempocurve: A regular, reference-scaled tempo curve
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Tempo Curve Computation (AW Approach)

Similar to the previous function, the `warpingpath_to_tempocurve_onsets` function computes a tempo curve from a warping path, this time using onset information to adapt the window size. Hence, the window size is here determined by `parameter.windowSizeIOI`. Again, the resulting tempo curve is scaled to the reference time axis.

Sample usage:

```
[f_pitch, f_peaks, f_onsets] = file_to_feature('midi', 'pathetique.mid');
parameter.windowSizeIOI = 12;
tempocurveAW = warpingpath_to_tempocurve_onsets(warpingpath, f_onsets,
                                                parameter);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: warpingpath_to_tempocurve_onsets
% Version: 1.0
% Date: 6.5.2009
% Programmer: Andi Scharfstein, Verena Konz
```

```
%
% Description:
%   Compute tempo curve from warping path using an adaptive window aligned
%   to note onset timings
%
% Input:
% - warpingpath: A regular warping path between reference and
%   interpretation of some piece of music
% - onsets: A vector of all unique frames of the warping path which contain
%   at least one note onset event.
% - parameter
%            .vis_warpingpath: If true, visualize warping path together
%                              with generated tempo curve
%            .vis_tempocurve: If true, plot the generated tempo curve
%            .windowSizeIOI: Determines how many interonset intervals
%                            should be included in the curve averaging
% Output:
% - tempocurve: A regular, reference-scaled tempo curve
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Tempo Curve Computation (FWC Approach)

As described in Section 3.2.4, the FWC approach relies on performing the FW technique on a smoothed warping path. The function `smooth_warpingpath` performs this smoothing on an interonset level.

Sample usage:

```
smoothedWarpingpath = smooth_warpingpath(warpingpath, f_onsets);
tempocurveFWC = warpingpath_to_tempocurve_SlidingWindow(smoothedWarpingpath,
                                                        parameter);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: smooth_warpingpath
% Version: 1.0
% Date: 16.01.2009
% Programmer: Andi Scharfstein, Verena Konz
%
% Description:
%   Perform IOI smoothing of a given warping path by re-computing the path
%   between onset locations as a straight line.
%
% Input:
% - warpingpath: A regular warping path between two pieces of music
% - onsets: A vector of all times (reference time axis) which contain at
%   least one note onset event in the reference
```

```
%
% Output:
% - smoothedWarpingpath: A warping path of the same dimension as the input
%   warping path, but with linear slopes between onset locations
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Onset Index Transformation

The conversion between onset times that are given relative to the reference time axis and indices into the warping path that locate the occurrences of these onsets if performed by the auxiliary function `compute_warped_onsets`. The function can be characterized by the equivalence `onsets == warpingpath(compute_warped_onsets(onsets, warpingpath), 1)`. The parameter `invertWarping` may be set to `true` to indicate that the onsets relate to the *interpretation* instead of the reference. In this case, the onsets will occur at `warpingpath(n, 2)` instead of at `warpingpath(n, 1)`.

Sample usage:

```
dtwOnsets = compute_warped_onsets(f_onsets, warpingpath);
dtwOnsetsInterpretation = compute_warped_onsets(f_onsets, warpingpath, 1);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: compute_warped_onsets
% Version: 1.0
% Date: 16.01.2009
% Programmer: Andi Scharfstein
%
% Description:
%   Convert absolute onset information into a representation relative to a
%   given warping path (i.e., determine indices of the onsets in this path)
%
% Input:
% - onsets: Onset information for the reference audio stream
% - warpingpath: The warping path of reference audio and comparison
% - invertWarping: If true, compute indices of onsets with respect to the
%   interpretation column of the warping path instead of the reference
%   column
%
% Output:
% - dtwOnsets: The indices of warping path entries where the reference
%   column (or the interpretation column if invertWarping is true)
%   corresponds to a note onset event
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Curve Interpolation

The interpolation function simply connects the points defined by the pairs (onsets(i), values(i)). The length of the resulting curve is equivalent to onsets(end). In case the first onset is not located at position 1, the vectors are extended by including $(1,1)$ before interpolation.

Sample usage:
```
interpolation = interpolate_between_onsets(values, f_onsets);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: interpolate_between_onsets
% Version: 1.0
% Date: 16.01.2009
% Programmer: Andi Scharfstein, Verena Konz
%
% Description:
%   Simple linear interpolation between onsets using a set of given values.
%
% Input:
% - values: Vector of values used to interpolate between consecutive onsets
% - onsets: Onset times between which to interpolate (length of values and
%   onsets vectors should be exactly equal)
%
% Output:
% - tempocurve: A tempocurve where intermediate values between onsets have
%   been filled out according to 'values' parameter. Here, values(i) will
%   be used as the value for onsets(i)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Curve Rescaling

The rescaling procedure translates from a reference-scaled tempo curve to an interpretation-scaled tempo curve. For this, it "warps" the original tempo curve according to warpingpath.

Sample usage:
```
tempocurveInterpretation = rescale_tempocurve(tempocurve, f_onsets,
                                              warpingpath);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: rescale_tempocurve
% Version: 1.0
% Date: 20.03.2009
% Programmer: Andi Scharfstein
%
```

```
% Description:
% Rescales standard tempo curves on the time axis to fit the interpretation
% they are describing, i.e. rescaled_tempocurve(x) is the tempo of
% interpretation at interpretation time point x instead of reference time
% point x.
%
% Input:
%   - tempocurve: A regular (reference-scaled) tempo curve
%   - reference_onsets: The onsets of the reference. If these are unknown,
%     just use (1:warpingpath(end,1))' to set onsets at every frame
%   - warpingpath: A regular warping path
%
% Output:
%   - rescaled_tempocurve: The rescaled tempo curve
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Tempo Curve Visualization

Plotting of tempo curves can be done comfortably using the `visualize_tempocurve` function. It is highly flexible: Calls of the form `visualize_tempocurve(tempocurve)` as well as calls of the form `visualize_tempocurve({tempocurve1 tempocurve2})` are supported, and there is a great variety of parameter settings available. Scaling of the temporal axis is controlled by the settings of `parameter.plotInMeasures` together with `parameter.startMeasure` and `parameter.endMeasure`, while scaling of the tempo value axis depends on the settings of `parameter.plotInBPM` and `parameter.referenceBPM`. If the tempo is plotted in BPM, the vertical axis is scaled linearly; otherwise, it is scaled logarithmically.

Sample usage:

```
% Linear curve plot
parameter.plotInBPM = 1;
parameter.referenceBPM = 33;
visualize_tempocurve(tempocurve, parameter);

% Multiple curves plot, time axis in measures
parameter.plotInMeasures = 1;
parameter.endMeasure = 55;
% parameter.startMeasure = 1; (this is the implicit default value)
visualize_tempocurve({tempocurve1 tempocurve2 tempocurve3}, parameter);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: visualize_tempocurve
% Version: 1.0
% Date: 04.02.2009
% Programmer: Andi Scharfstein, Verena Konz
%
```

```
% Description:
%   A function for visualizing one or more tempo curves. If you want to
%   display more than one curve, wrap them in a cell array.
%
% Input:
% - tempocurve: One or more vectors containing tempo information
%   (wrap in cell array to show multiple curves at once).
% - parameter
%           .plotInBPM: If true, plot the curve in BPM instead of factors
%           .referenceBPM: Needed to compute the curve (factor 1 becomes
%                          parameter.referenceBPM) if plotInBPM is true
%           .plotInMeasures: If true, use measures for the time axis
%                            instead of seconds or frames
%           .startMeasure: Optional, denotes the first measure if
%                          plotInMeasures is true
%           .endMeasure: Needed to compute the measures if plotInMeasures
%                        is true
%           .holdFigure: If true, use gcf instead of creating a new figure
%           .scaleToSeconds: If true, plot time axis in seconds instead of
%                            frames
%           .lineStyle: Sets the line style of the plot
%           .lineWidth: Sets the line width of the plot
%           .win_res: Used to scale the plot to seconds if scaleToSeconds
%                     is true
%
% Output:
% - figureHandle: The handle of the figure that was created for
%   visualization of the tempo curve(s).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Warping Path Visualization

As for tempo curves, there also exists a specialized function for plotting warping paths. Unsurprisingly, this is the `visualize_warpingpath` function. It supports calls similar in structure to `visualize_tempocurve`, in particular both `visualize_warpingpath(warpingpath)` and `visualize_warpingpath({warpingpath1 warpingpath2})` are possible. Optional parameters include `parameter.dtwOnsets` and `parameter.tempocurve` that can be used to include vertical onset lines or a tempo curve in the plot, respectively. Onsets must be given in their "warped" version as indices to the respective warping path. If one of those parameters is set, the `warpingpath` argument must be a single path and may not contain a cell array.

Sample usage:

```
visualize_warpingpath({warpingpath1 warpingpath2});

tempocurve = warpingpath_to_tempocurve_onsets(warpingpath, f_onsets,
                                              parameter);
```

```
parameter.tempocurve = tempocurve;
parameter.dtwOnsets = compute_warped_onsets(f_onsets, warpingpath);
visualize_warpingpath(warpingpath, parameter);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Name: visualize_warpingpath
% Version: 1.0
% Date: 15.01.2009
% Programmer: Andi Scharfstein, Verena Konz
%
% Description:
%   A function for visualizing one or more warping paths. If you want to
%   display more than one warping path, wrap your paths in a cell array.
%
% Input:
% - warpingpath: One or more regular warping paths (wrap in cell array to
%   plot multiple warping paths).
% - parameter
%            .dtwOnsets: Optional, plots vertical lines at onset positions.
%                        Must be given in terms of warping path indices,
%                        not in terms of a reference time axis as usual
%            .tempocurve: Optional, but must be given if dtwOnsets is set.
%                         A regular tempo curve computed from the given
%                         warping path is plotted against this path
%            .scaleToSeconds: If true, plot time axis in seconds instead of
%                             frames
%            .win_res: Used to scale the plot to seconds if scaleToSeconds
%                      is true
%
% Output:
% - figureHandle: The handle of the figure that was created for
%   visualization of the warping path(s).
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Bibliography

[Bur08] Juan José Burred. From sparse models to timbre learning: New methods for musical source separation. PhD Thesis, 2008.

[Cla87] Eric F. Clarke. Levels of structure in the organization of musical time. *Contemporary Music Review*, 2:211–238, 1987.

[Dix01] Simon Dixon. Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research*, 30:39–58, 2001.

[Dix07] Simon Dixon. Evaluation of the audio beat tracking system BeatRoot. *Journal of New Music Research*, 36:39–50, 2007.

[EMG09] Sebastian Ewert, Meinard Müller, and Peter Grosche. High resolution audio synchronization using chroma onset features. In *Proc. IEEE ICASSP, Taipei, Taiwan*, 2009.

[FGW08] Sebastian Flossmann, Maarten Grachten, and Gerhard Widmer. Experimentally investigating the use of score features for computational models of expressive timing. In *ICMPC Proceedings*, 2008.

[Gab03] Alf Gabrielsson. Music Performance Research at the Millennium. *Psychology of Music*, 31:221–272, 2003.

[GHNO02] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. RWC Music Database: Popular, Classical and Jazz Music databases. In *ISMIR Proceedings*, pages 287–288, 2002.

[GME09] Peter Grosche, Meinard Müller, and Sebastian Ewert. Combination of onset-features with applications to high-resolution music synchronization. In *DAGA Proceedings*, 2009.

[Hum09] Humdrum. The humdrum toolkit, *Software for Music Research.* `http://musiccog.ohio-state.edu/Humdrum/`, Retrieved 31.03.2009.

[LG03] Jörg Langner and Werner Goebl. Visualizing expressive performance in tempo-loudness space. *Computer Music Journal*, 27(4):69–83, 2003.

[Lil09] LilyPond. LilyPond, *an automated music engraving system.* `http://lilypond.org/web/about/`, Retrieved 31.03.2009.

[Mat09] The MathWorks. Matlab R2009a. `http://www.mathworks.com/products/matlab/`, Retrieved 31.03.2009.

[Mic08] Ulrich Michels. *dtv-Atlas Musik.* Deutscher Taschenbuch Verlag, 2008.

[Mül07] Meinard Müller. *Information Retrieval for Music and Motion.* Springer, 2007.

[Rec09] Recordare. MusicXML, *A universal translator for common Western musical notation.* `http://recordare.com/xml.html`, Retrieved 31.03.2009.

[Sap07] Craig Stuart Sapp. Comparative analysis of multiple musical performances. In *ISMIR Proceedings*, pages 497–500, 2007.

[Sap08] Craig Stuart Sapp. Hybrid numeric/rank similarity metrics. In *ISMIR Proceedings*, 2008.

[Son09] Sonic Visualiser. `http://www.sonicvisualiser.org/`, Retrieved 31.03.2009.

[WDG$^+$03] Gerhard Widmer, Simon Dixon, Werner Goebl, Elias Pampalk, and Asmir Tobudic. In search of the Horowitz factor. *AI Magazine*, 24(3):111–130, 2003.

[Wid02] Gerhard Widmer. Machine discoveries: A few simple, robust local expression principles. *Journal of New Music Research*, 31(1):37–50, 2002.

[Wid05] Gerhard Widmer. Musikalisch intelligente Computer – Anwendungen in der klassischen und populären Musik. *Informatik Spektrum*, Oct.:363–368, 2005.