Friedrich-Alexander-Universität Erlangen-Nürnberg

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

Master Thesis

Audio Processing Techniques for Analyzing Georgian Vocal Music

submitted by

Sebastian Rosenzweig

submitted

August 31, 2017

Supervisor / Advisor

Prof. Dr. Meinard Müller

Reviewers

Prof. Dr. Meinard Müller

AUDIO LABS

International Audio Laboratories Erlangen
A joint institution of the
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and
the Fraunhofer-Institut für Integrierte Schaltungen IIS.

Fraunhofer IIS

Erklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt.

Erlangen, August 31, 2017

Sebastian Rosenzweig

Acknowledgements

Before diving deeper into audio processing and Georgian vocal music, I would like to express my gratitude to all people who supported me during this thesis project.

First, I would like to thank Prof. Dr. Meinard Müller for being an inspiring mentor and for introducing me to the world of science. The past one and a half years have been a valuable and informative experience for me, which finally encouraged me to start a PhD. Thank you for also giving me the opportunity to contribute to publications at this very early stage.

The initial idea of combining research on audio signal processing and Georgian vocal music was brought to us by Prof. Dr. Frank Scherbaum. He did not only introduce us to Georgian vocal music, Artem Erkomaishvili's recordings, and current ethnomusicological research questions, he also inspired us with his enthusiasm and curiosity. During this thesis project, I had the opportunity to visit Frank Scherbaum and his colleague Daniel Vollmer in Berlin/Potsdam for an interesting introduction to Georgian music and a very helpful tutorial on throat microphones. Thank you for providing us your field recordings, we are curious about working with them in the near future and are looking forward to continuing this fruitful cooperation.

Large parts of this thesis are based on algorithms and code of Jonathan Driedger and the accurate reference annotations by Stefanie Kämmerle. Thank you very much for this excellent basis. Furthermore, I would like to thank Christian Dittmar for providing his code for singing voice detection. A special thanks goes to my office mate Patricio López-Serrano for his helpful advice and proof-reading of this thesis. I also want to thank all other members of Meinard's group for inspiring discussions and useful hints.

This thesis concludes my studies on Communications and Multimedia Engineering. Getting to know people from so many different cultures and backgrounds has been an unforgettable and enriching experience. Thanks to all my new and old friends who made this an incredible journey!

Finally, I would like to thank my family and Doro for their unconditional love and support.

Abstract

Analyzing recorded audio material has become increasingly important in ethnomusicological research. These recordings may contain valuable cues on performance practice - information that is often lost in symbolic music transcriptions. A special case is the current ethnomusicological research on polyphonic vocal music from Georgia, a country located in the Caucasus region of Europe. Research on Georgian vocal music is challenging, since these chants have exclusively been passed down orally for generations, with only few transcriptions available.

In this context, a unique dataset of three-voice chant recordings by the former master chanter Artem Erkomaishvili has become of great interest for ethnomusicological research. The chants were recorded in a three-stage process, leading to a repetitive structure. In this thesis, we address two music information retrieval problems based on this special dataset. First, the goal is to segment these recordings by detecting similar or repeating parts. Second, we consider a task called fundamental frequency estimation, where the goal is to estimate the predominant melody as a basis for transcription or further musical analysis.

The main contributions of this thesis are as follows. First, to segment the recordings, we examine a standard audio matching technique in detail and perform an extensive evaluation on the results. For reference, we also apply a machine learning approach for the same task. Second, we present a fundamental frequency estimation algorithm tailored to to the repetitive structure of the recordings and compare its performance to related algorithms. Third, based on reference annotations, we show how audio processing techniques can support Georgian vocal music research by musical interval analysis.

Zusammenfassung

In der Musikethnologie gewinnt die Analyse von Audiomaterial zunehmend an Bedeutung. Ein Grund hierfür ist, dass Audioaufnahmen wertvolle Informationen zur Aufführungspraxis enthalten können, die in Transkriptionen oft nicht enthalten sind. Ein aktuelles Fallbeispiel ist die musikethnologische Forschung zu polyphoner Vokalmusik aus Georgien. Die Forschung hierzu stellt eine Herausforderung dar, da die Gesänge seit Generationen ausschließlich mündlich weitergegeben werden und wenige Transkriptionen vorhanden sind.

In diesem Kontext ist ein besonderer Datensatz von dreistimmigen Gesangsaufnahmen des einstigen georgischen Meistersängers Artem Erkomaishvili von großem Interesse für die musikethnologische Forschung. Die Aufnahmen wurden in einem dreistufigen Verfahren angefertigt und weisen eine von Wiederholungen geprägte Struktur auf. Basierend auf diesem speziellen Datensatz, beschäftigt sich diese Masterarbeit mit zwei Problemstellungen im Bereich des Music Information Retrievals. Zunächst ist es das Ziel, die Aufnahmen durch Erkennung ähnlicher oder sich wiederholender Teile zu segmentieren. Danach werden Verfahren zur Fundamentalfrequenzschätzung betrachtet, die die dominierende Melodie in Audioaufnahmen bestimmen. Die Ergebnisse können als Grundlage für Transkriptionen oder weitere musikalische Analysen dienen.

Ein Hauptbeitrag dieser Arbeit ist die detaillierte Untersuchung einer Standard-Audio-Matching-Technik zur Segmentierung der wiederholungsbasierten Aufnahmen. Als Referenz für diese Aufgabe wird ein Ansatz aus dem Bereich des Maschinellen Lernens angewendet. Des Weiteren wird ein Algorithmus zur Fundamentalfrequenzschätzung entwickelt, der auf die Wiederholungsstruktur der Aufnahmen zugeschnitten ist, und mit anderen, verwandten Algorithmen verglichen wird. Darüber hinaus wird anhand von Analysen zu musikalischen Intervallen gezeigt, wie Techniken aus der Audiosignalverarbeitung musikethnologische Forschung zu georgischer Vokalmusik unterstützen können.

Contents

Chapter 1

Introduction

Music is an elementary part of every culture. The distinctiveness of each musical work or performance is shaped by social, geographical, and religious influences. The scientific analysis of these complex mixtures of influences in order to "decipher" the main musical ideas and recover the artists' or composers' original intention is the subject of ethnomusicological research. In general, ethnomusicology is defined as the "division of musicology in which special emphasis is given to the study of music in its cultural context" [31]. Starting in the late 19th century as a discipline focusing on comparative studies, nowadays, ethnomusicology has adopted a multitude of disciplines such as the preservation of disappearing music cultures.

In the course of a rising mechanization in the 20th century, audio recordings of musical performances have become increasingly important for ethnomusicologists and researchers in related disciplines [31]. Audio recordings not only constitute an easy method to preserve musical performances, they also enable unlimited reproduction for later analysis. Furthermore, an audio recording can be seen as an "objective" representation of a musical performance, compared to symbolic music transcriptions that may include subjective interpretations. The availability of audio recordings in digital form and the increasing computational capability starting from the late 20th century paved the way for an interdisciplinary research field called *Music Information Retrieval* (MIR). MIR aims at "extending the understanding and usefulness of music data, through the research, development and application of computational approaches and tools" [1]. Thus, researchers combine knowledge from music, computer science and signal processing.

This thesis is grounded at the intersection of music information retrieval and ethnomusicology. More specifically, we examine and develop audio signal processing techniques for analyzing a set of chant recordings by the former Georgian master chanter Artem Erkomaishvili. The recording collection is of great importance for ethnomusicologists, since there exist only few audio recordings of Georgian chants. The dataset also motivates exciting MIR research questions: since the three-voice chants were recorded in a three-stage process, the recordings exhibit a repetitive

structure, which creates an interesting scenario for computational analysis. In this thesis, we focus on two major MIR tasks.

The first task is termed audio segmentation. Although there are various criteria to segment audio recordings, they can be grouped into three main classes [26]: homogeneity-based segmentation detects parts that are similar with respect to a certain musical property such as instrumentation or tempo. Novelty-based segmentation is based on events or sudden changes. Repetition-based segmentation relies on similar or repeating parts. Due to the repetitive structure of Erkomaishvilis recordings, we first focus on repetition-based segmentation and then give a short outlook to homogeneity- and novelty-based segmentation.

The second task is called *fundamental frequency estimation*, which involves "automatically obtaining a sequence of frequency values representing the pitch of the dominant melodic line from recorded music audio signal" [37]. The task can also be seen as an intermediate step for symbolic music transcriptions. In this context, the dataset of Georgian chants provides different levels of difficulty, since the recordings include monophonic as well as more complex polyphonic parts. From an ethnomusicological perspective, the results provide a starting point for various subsequent analyses addressing the performance practice of Georgian vocal music.

1.1 Structure of this Thesis

In Chapter 2, we first elaborate on the musical background with specific focus on Georgia, Georgian vocal music and current ethnomusicological research. Then, we review the well-known Short-Time Fourier Transform, log-frequency spectrograms and the concept of instantaneous frequency estimation, which form the technical background of this thesis.

In Chapter 3, we discuss the segmentation of the given Georgian chant recordings. To this end, we first describe the generation of reference annotations. Secondly, we examine informed and less informed audio matching approaches based on diagonal matching. Finally, we apply a machine-learning algorithm on this task.

Chapter 4 is dedicated to fundamental frequency (F0) estimation. After giving an overview of techniques and challenges, we elaborate on the generation of reference annotations. Then, we present several F0 estimation algorithms and show their properties on selected recordings. Furthermore, we develop a three-stage F0 estimation algorithm tailored to the Georgian chant recordings. In a final step, we evaluate the performance of all algorithms on the dataset.

In Chapter 5, we show how audio processing techniques can support research on Georgian vocal music by interval analysis on the reference F0 annotations. Furthermore, we outline ways to enhance analysis results.

In Chapter 6, we recapitulate the main achievements of this thesis, draw final conclusions and outline future work.

1.2 Main Contributions

The main contributions of this thesis are as follows.

First, we apply audio matching approaches, which exploit the repetitive structure of the Georgian chant recordings. Furthermore, we provide a deeper understanding of diagonal matching by comparing different features and distance measures. As an example for homogeneity- and novelty-based segmentation, we demonstrate how a machine-learning-based singing voice detector can be applied to the same task.

Second, we develop a three-stage F0 estimation algorithm tailored to the repetitive structure of the recordings. Additionally, we provide a detailed performance comparison of multiple standard F0 estimation algorithms on the dataset of Georgian chants.

Third, we outline possible applications to ethnomusicological research by performing musical interval analysis.

Chapter 2

Background

In this chapter, we elaborate on the musical and technical background of this thesis. More specifically, in Section 2.1, we give a brief introduction to Georgian vocal music research and the given set of recordings, which is used in our experiments. In Section 2.2, we introduce concepts and notions of time-frequency representations, which form the basic signal processing tools of this work.

2.1 Georgian Vocal Music Research

In this section, we first give an introduction to Georgia, Georgian vocal music and current ethnomusicological research (Section 2.1.1). Secondly, we describe a collection of Georgian chant recordings of great importance for ethnomusicological research, which we analyze throughout this thesis (Section 2.1.2). Finally, we point out related research in this field (Section 2.1.3).

2.1.1 Georgian Singing Tradition

The description in this section closely follows [30].

Georgia is a country located in the Caucasus region of Eurasia, which has a rich centuries-old polyphonic singing tradition. In particular, Georgian polyphonic singing has been acknowledged as an intangible cultural heritage, which has "regained a place of prominence in the hearts and minds of the public and in the life of the Church" [42].

Despite its small size, Georgia is home to many stylistically very diverse singing traditions, which form an essential part of the cultural identity of this country and which are increasingly receiving the attention of international music lovers, musicians, and scholars alike [46].

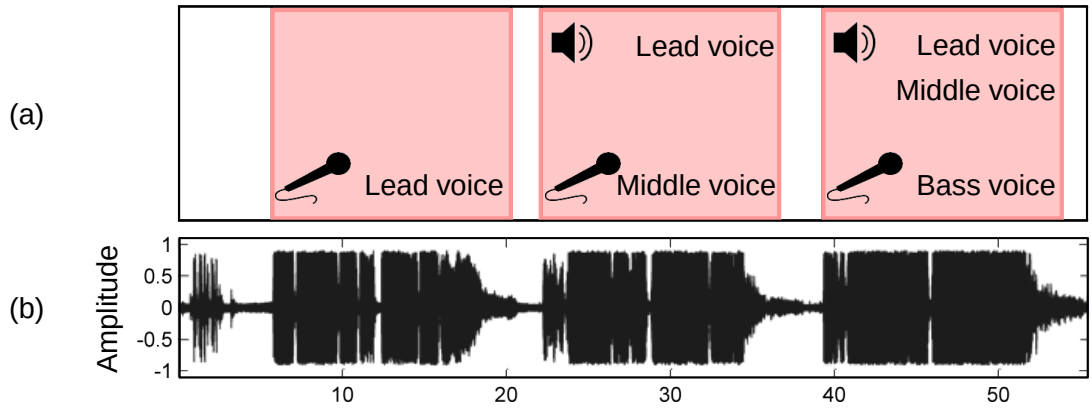

Figure 2.1: Recording of the Georgian chant "Da Sulisatsa" sung by Artem Erkomaishvili (from [30]). **(a)** Three-stage recording process. **(b)** Waveform.

In order to preserve this cultural treasure, the "Commission for Chant Preservation" began notating Georgian chants in the 1860s, which had been passed down orally for many generations. This has resulted in thousands of transcriptions collected between 1880–1920. Due to changing social and political conditions, however, the tradition of how to perform these chants has largely been lost, and ethnomusicologists have started to research on traditional performance practice [42].

The distinctiveness of Georgian vocal polyphonic music in comparison to Western music is based on the abundant use of "dissonances" and on the fact that the music is not tuned to the 12-tone equal-tempered scale. While musicologists have agreed on the non-tempered nature of traditional Georgian vocal music, the particular nature of the traditional Georgian tuning is an ongoing topic of intense and controversial discussions [12, 45, 38].

2.1.2 Recordings by Artem Erkomaishvili

The description in this section closely follows [30].

In the context of the aforementioned preservation activities for Georgian chants, a collection of music recordings performed by Artem Erkomaishvili (1887–1967)—one of the last representatives of the master chanters ("sruligalobelni") of Georgian music—has become of great importance.

Recorded at the Tbilisi State Conservatory in 1966, the aging Erkomaishvili was asked to perform three-voice chants by successively singing the individual voices. At the beginning of each recording, Artem Erkomaishvili announces the name of the chant he is going to perform. After recording the lead voice, one tape recorder was used to playback this first voice while a second tape recorder was used to synchronously record the middle voice. Similarly, playing back the first and second voice, the bass voice was recorded, see Figure 2.1. Note that due to this specific recording procedure, Artem Erkomaishvili begins the middle and bass voice always with a slight

offset against the lead voice.

In this way, Erkomaishvili was able to accompany and embellish his own recordings, yielding a genuine source of original Georgian musical thinking [42]. The resulting collection of 101 audio recordings of $1 - 13$ minutes length is hosted at the Folklore Department of the Tbilisi State Conservatory and has been made available at [43]. The collection comprises various types of chants including hymns for Easter, Christmas, or wedding ceremonies. Despite the poor sound quality, especially in the third part of the recordings due to tape recorder noise, this collection is still of great importance for ethnomusicological research.

While analyzing these chants, the files in the collection have been renamed according to the convention introduced in Appendix A. Furthermore, the files have been transcoded from the original 128 kbit/s MP3 to mono WAV files with a sampling rate of 22 050 Hz. We will use this collection throughout this thesis as an input for various audio processing techniques, which also exploit the three-part structure of the recordings.

2.1.3 Related Research

In summer 2015, another collection of Georgian chants was recorded by Scherbaum et al. during an exploratory field trip to Upper Svaneti/Georgia [40]. The recordings comprise various chants sung by different groups and singers in different villages. A special property of this collection is the variety of recording devices used. Besides a camcorder for video recording and conventional stereo microphones to capture the overall impression, each of the singers was recorded separately with a headset microphone and a larynx/throat microphone.

Throat microphones pick up the vibrations of the singer's throat and convert them to an audio signal. In this way, unwanted environmental noise is not recorded by the microphones, leading to a reasonable voice quality even in loud environments. This effect is especially useful when recording polyphonic vocal music performances, since every throat microphone captures the voice of only one singer while suppressing the other singers' voices. First experiments with throat microphones have been conducted in [39].

At the time of this writing, the recordings were not yet publicly available.

2.2 Time-Frequency Representations

In this section, we introduce the basic notions of time-frequency representations used throughout this thesis. In particular, we focus on the Fourier Transform and its short-time formulation (STFT) in Section 2.2.1. Based on these formulations we derive a spectrogram respresentation with a logarithmically spaced frequency axis in Section 2.2.2. Finally, in Section 2.2.3, we show

how the concept of *Instantaneous Frequency Estimation* can be used to refine the frequency resolution of the previously introduced STFT.

2.2.1 Fourier Transform

The *Fourier Transform* is the basic analysis step in many applications in audio signal processing. It transforms a given time signal into the frequency domain. For a discrete time signal x of length $N \in \mathbb{N}$, the *Discrete Fourier Transform* (DFT) is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) \exp(-2\pi ikn/N) \ , \tag{2.1}$$

for $k \in [0 : N-1]$ according to [26, Section 2.4].

The complex-valued *Fourier coefficients* X describe the magnitude and phase of a sinusoidal with physical frequency

$$F_{\text{coef}}(k) = \frac{k \cdot F_{\text{s}}}{N} \ . \tag{2.2}$$

Figure 2.2 shows different representations of our running example "Da Sulisatsa" (GCH_087_Erkomaishvili.wav) by Artem Erkomaishvili. From the waveform in Figure 2.2a, we can get a rough idea about when a sound event is occurring, but no information about its spectral properties. In the magnitude Fourier spectrum in Figure 2.2b, we see the spectral content of the whole recording, however, the temporal information is lost. In order to retrieve both the temporal and spectral information of a digital audio recording, one uses the discrete *Short-Time Fourier Transform* (STFT). The basic idea behind the STFT is to divide the audio signal into short frames of length N using a suitable window function w and calculate the Fourier transform for each of the frames. Following [26, Section 2.5], this leads to

$$\mathcal{X}(m,k) = \sum_{n=0}^{N-1} x(n+mH)w(n) \exp(-2\pi ikn/N) \ , \tag{2.3}$$

where $m \in \mathbb{Z}$ is the *frame index* and $k \in [0 : K]$ is the *frequency index*. The hopsize $H \in \mathbb{N}$ defines the number of samples between two consecutive frames. The resulting complex time- and frequency-dependent coefficients are associated with the physical time position

$$T_{\text{coef}}(m) = \frac{m \cdot H}{F_{\text{s}}} \tag{2.4}$$

given in seconds and the physical frequency

$$F_{\text{coef}}(k) = \frac{k \cdot F_s}{N} \tag{2.5}$$

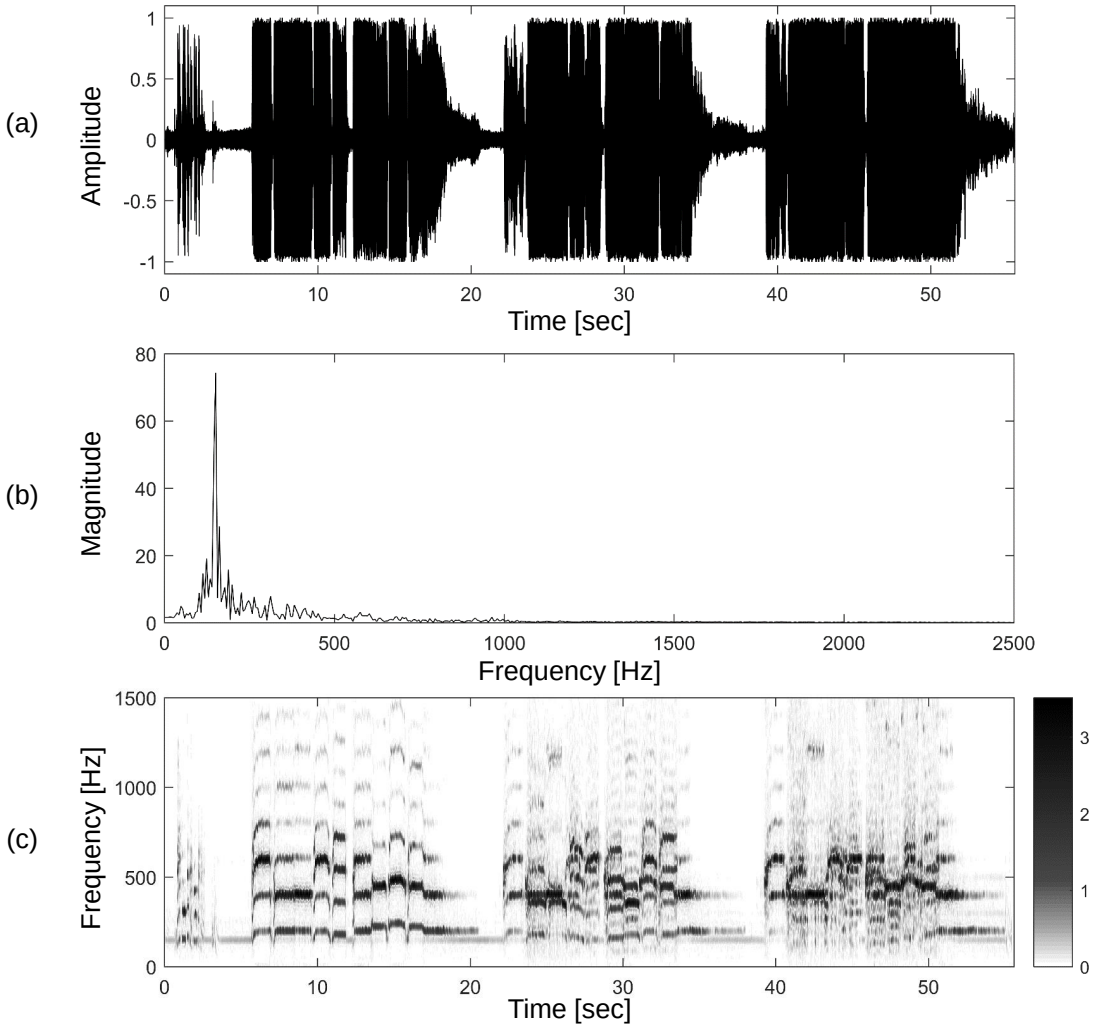

Figure 2.2: Signal representations of "Da Sulisatsa" (And with thy spirit) by Artem Erkomaishvili. **(a)** Waveform. **(b)** Magnitude Fourier spectrum. **(c)** Magnitude spectrogram.

given in Hz. The magnitude of the STFT is referred to as *magnitude spectrogram*:

$$\mathcal{Y}(m,k) := |\mathcal{X}(m,k)| \,. \tag{2.6}$$

Figure 2.2c shows a magnitude spectrogram of our running example with $N = 1024$ Samples and $H = 128$ Samples. Dark regions in the spectrogram reflect coefficients with high magnitude, whereas white regions reflect coefficients with low magnitude. In order to enhance regions in the spectrogram with low magnitude, the spectrogram has been compressed using logarithmic compression defined by

$$\Gamma_\gamma(\mathcal{Y}) := \log(1 + \gamma \cdot \mathcal{Y}) \,, \tag{2.7}$$

with $\gamma = 0.1$.

2.2.2 Log-Frequency Spectrogram

The spectrogram representation introduced in Section 2.2.1 based on the STFT possesses a linearly sampled frequency axis. This means that the distance between the center frequencies of two neighboring frequency bands is constant. However, in order to account for the logarithmic frequency perception of the human ear, it is desirable to have a logarithmically spaced frequency axis. The so called *log-frequency spectrogram* can be calculated in different ways, e.g. using interpolation or pitch filterbanks [25, Chapter 3]. Here, we focus on a rather simple binning technique explained in [26, Section 3.1]. Given a STFT $\mathcal{X}$ of an audio signal, the main idea of this technique is to assign each coefficient $\mathcal{X}(n,k)$, $n \in \mathbb{Z}$ and $k \in [0:K]$, to a MIDI pitch with the center frequency $F_{\text{pitch}}(p)$ that is closest to the frequency $F_{\text{coef}}(k)$. More precisely, we define the set of frequency indices that are pooled in the pitch band p (also called *bin*) as

$$P(p) = \{k : F_{\text{pitch}}(p-0.5) \leq F_{\text{coef}}(k) < F_{\text{pitch}}(p+0.5)\} \,, \tag{2.8}$$

with $p \in [0:127]$. A *pitch-based log-frequency spectrogram* $\mathcal{Y}_{\text{LF}} : \mathbb{Z} \times [0:127] \to \mathbb{R}$ is then given by

$$\mathcal{Y}_{\text{LF}}(n,p) = \sum_{k \in P(p)} |\mathcal{X}(n,k)|^2 \,. \tag{2.9}$$

According to [26, Section 8.2.2.1], it can be shown that equation 2.8 can also be reformulated to

$$P(p) = \{k : \text{Bin}(F_{\text{coef}}(k)) = p\} \,, \tag{2.10}$$

with the binning function

$$\text{Bin}(\omega) = \left\lfloor 12 \cdot \log_2 \left(\frac{\omega}{440} \right) + 69.5 \right\rfloor \,. \tag{2.11}$$

The binning function assigns a given frequency $\omega \in \mathbb{R}$ to a MIDI pitch $p := \text{Bin}(\omega)$. This yields, e.g. for the reference tone A4 $\widehat{=}$ 440 Hz, a MIDI pitch of $p = 69$. Furthermore, an octave is subdivided into twelve bands according to the twelve-tone equal tempered scale. Note that this mapping creates a logarithmic frequency axis with a fixed resolution of 100 cents (one semitone) per bin. When analysing music recordings that do not follow the twelve-tone equal-tempered scale, such as Georgian music, one may want to have a finer logarithmic axis resolution. To this end, we generalize the binning function 2.11 as follows. Given a desired frequency resolution R in cents and a reference frequency ω_{ref}, which is assigned to the bin index 1, the bin assignment is defined by

$$\text{Bin}(\omega) = \left\lfloor \frac{1200}{R} \cdot \log_2 \left(\frac{\omega}{\omega_{\text{ref}}} \right) + 1.5 \right\rfloor \,. \tag{2.12}$$

Note that although this definition allows us to bin the coefficients with an arbitrary resolution,

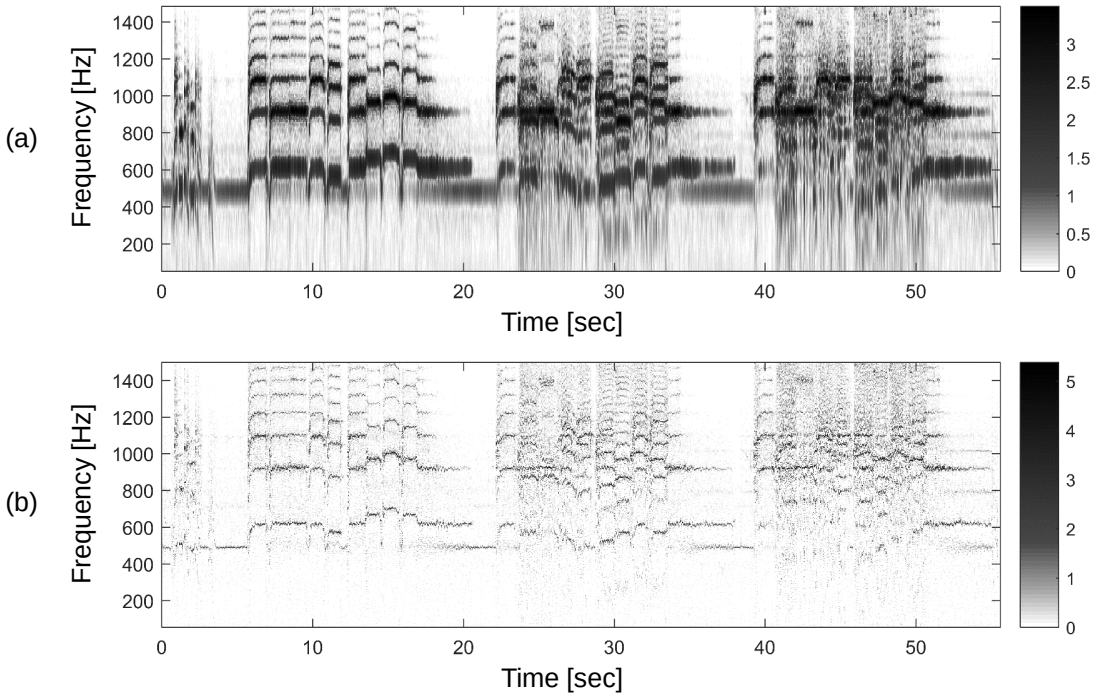

Figure 2.3: Spectrogram representations of "Da Sulisatsa" (And with thy spirit) by Artem Erkomaishvili. **(a)** Log-frequency spectrogram. **(b)** Log-frequency spectrogram with refined frequency resolution.

the overall frequency resolution of the log-frequency spectrogram is still limited by the STFT hopsize and window overlap. However, the frequency resolution of the STFT can be refined, e.g. by using *instantaneous frequency estimation* (see Section 2.2.3).

A log-spectrogram with a resolution of $R = 10$ cents and a reference frequency $\omega_{\text{ref}} = 55$ Hz of our running example "Da Sulisatsa" (And with thy spirit) by Artem Erkomaishvili is shown in Figure 2.3a. Again we applied logarithmic compression with $\gamma = 0.1$. The non-linear expansion of the linearly spaced STFT coefficients results in a clearly visible blurring effect in the lower part of the spectrogram.

2.2.3 Refined STFT Resolution

Depending on the STFT parameters, the frequency resolution of the STFT may not be high enough to capture small variations in melody such as vibrato or glissando. Increasing the window length N would improve the frequency resolution, but would also lead to an undesirable decrease in time resolution. In this section, we introduce a method called *Instantaneous Frequency Estimation*, which makes use of the spectrogram's phase information to increase the frequency resolution without changing the time resolution.

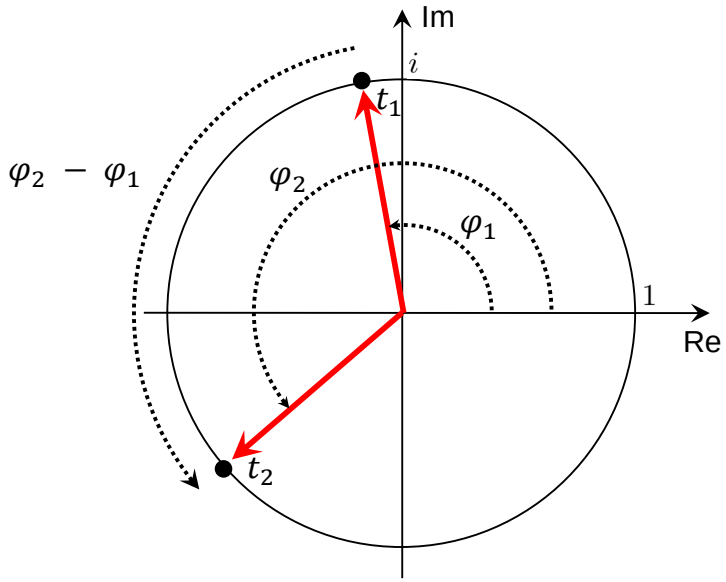

Figure 2.4: Illustration of phase progression in the complex plane (from [26, p. 435]).

First, we recall the definition of the STFT as introduced in Section 2.2.1. Each complex-valued time-frequency bin $\mathcal{X}(n,k)$ can be interpreted as a sinusoidal component with amplitude $|\mathcal{X}(n,k)|$ and phase $\varphi(n,k)$. Assume we are given two adjacent time-frequency bins $\mathcal{X}_1 = \mathcal{X}(n,k)$ and $\mathcal{X}_2 = \mathcal{X}(n+1,k)$, both corresponding to the band with the "coarse" physical frequency $\omega = F_{\text{coef}}(k)$. From the STFT coefficients, we are also given the corresponding phase estimates $\varphi_1 = \varphi(n,k)$, $\varphi_2 = \varphi(n+1,k)$ at the time instances $t_1 = T_{\text{coef}}(n)$, $t_2 = T_{\text{coef}}(n+1)$. The given scenario is visualized in Figure 2.4. Our goal is to use the given time and phase information to estimate the frequency band's "real" frequency $IF(\omega)$.

Given the physical frequency ω, we can determine the phase progression over the course of the interval $\Delta t = t_2 - t_1$:

$$\omega \cdot \Delta t \ .$$

With this and φ_1 we can estimate the phase at time instant t_2:

$$\varphi^{\text{Pred}} = \varphi_1 + \omega \cdot \Delta t \ . \tag{2.13}$$

Now, we can compare the predicted phase φ^{Pred} to the given phase φ_2 and compute the phase error φ^{Err}. Ideally, if both bins correspond to the same "real" frequency, the error will be zero.

$$\varphi^{\text{Err}} = \Psi(\varphi_2 - \varphi^{\text{Pred}}) \ . \tag{2.14}$$

In order to avoid phase ambiguities, we need an unwrapped version of the phase obtained by the *principal argument function* $\Psi : \mathbb{R} \to [-0.5, 0.5)$. With these results, we can correct the "coarse"

phase progression by adding the error phase:

$$\omega \cdot \Delta t + \varphi^{\mathrm{Err}} \ .$$

This gives us the number of oscillations within the interval Δt. Given that Δt is very small (typically a couple of ms), we can finally compute the instantaneous frequency estimates

$$F_{\mathrm{coef}}^{\mathrm{IF}}(n,k) = IF(\omega) = \frac{\omega \cdot \Delta t + \varphi^{\mathrm{Err}}}{\Delta t} = \omega + \frac{\varphi^{\mathrm{Err}}}{\Delta t} \ . \tag{2.15}$$

Similarly to the procedure described in Section 2.2.2, we can bin the refined STFT coefficients $F_{\mathrm{coef}}^{\mathrm{IF}}(n,k)$ onto a logarithmic frequency axis. To this end, we again define the set of frequency indices that are pooled together:

$$\mathrm{P}^{\mathrm{IF}}(n,b) = \{k : \mathrm{Bin}(F_{\mathrm{coef}}^{\mathrm{IF}}(n,k)) = b\} \ , \tag{2.16}$$

with bin index $b \in [1:B]$ and frame index $n \in \mathbb{Z}$. The refined log-frequency spectrogram $\mathcal{Y}_{\mathrm{LF}}^{\mathrm{IF}}$ is derived from this new assignment by setting

$$\mathcal{Y}_{\mathrm{LF}}^{\mathrm{IF}}(n,b) = \sum_{k \in \mathrm{P}^{\mathrm{IF}}(n,b)} |\mathcal{X}(n,k)|^2 \ . \tag{2.17}$$

Figure 2.3b shows a refined log-frequency spectrogram again logarithmically compressed with $\gamma = 0.1$. In comparison with the standard log-frequency spectrogram shown in Figure 2.3b, the resolution, especially in the lower part of the spectrogram, has increased. Furthermore, the vertical structures in the refined spectrogram are much sharper. Further information on instantaneous frequency estimation can be found in [26, Section 8.2].

Chapter 3

Audio Segmentation

Segmentation of audio recordings is a fundamental task in music processing and can serve as a starting point for many subsequent analysis steps such as fundamental frequency estimation or harmony analysis. In this chapter, we analyze the structure of Artem Erkomaishvili's recordings introduced in Section 2.1.2 with specific focus on the repetition of the lead voice. Due to the three-stage recording procedure, the lead voice occurs three times in the recording: the first time as a solo voice, the second time together with the middle voice, and the third time together with middle and bass voices (see Figure 2.1). Our goal is to find the start and end points of all three occurrences of the lead voice in each of the given recordings in an automated fashion. Throughout this chapter, we rely on existing ground truth annotations, which will be described in Section 3.1.

In the last years, many techniques have been developed, which may be suitable to solve this segmentation task. A well-known method for music structure analysis are self-similarity matrices [26, Chapter 4.2]. A good overview about different segmentation techniques is given in [32]. However, we approach this task in two different ways: first, focusing on repetition-based segmentation principles, we interpret the task as an audio retrieval problem in Section 3.2. In this context, we develop an informed matching procedure to find all occurrences. By interpreting the task as a classification problem, our second approach described in Section 3.3, aims to detect all parts where a singing voice is active for subsequent segmentation. To this end, we apply a machine-learning algorithm for singing voice detection, which is implicitly based on homogeneity and novelty segmentation principles. Finally, in Section 3.4, we draw conclusions for both approaches.

Description	Variable	Timestamp in Seconds
Start Lead Voice	t_1^{orig}	5.706122448
End Lead Voice	t_2^{orig}	20.60408163
Start Lead Voice (Loop)	t_3^{orig}	22.11428571
Start Middle Voice	t_4^{orig}	23.60000000
End Middle Voice	t_5^{orig}	37.87619048
End Lead Voice (Loop)	t_6^{orig}	38.0214059
Start Lead Voice (Loop)	t_7^{orig}	39.24897959
Start Middle Voice (Loop)	t_8^{orig}	40.74780045
Start Bass Voice	t_9^{orig}	40.75732426
End Bass Voice	t_{10}^{orig}	55.03433107
End Middle Voice (Loop)	t_{11}^{orig}	55.03673469
End Lead Voice (Loop)	t_{12}^{orig}	55.05306122

Table 3.1: Original annotation with twelve timestamps for "Da Sulisatsa".

3.1 Reference Annotations

Throughout this thesis, we relied on the ground truth annotations for all 101 chants in the collection. The annotations have been produced by a student with some musical background using the publicly available application "Sonic Visualiser" [6]. Originally, every annotation consists of a list of twelve timestamps corresponding to the start and end points of every voice in each recording step. For the first recording step with only one voice present, there exist two timestamps, whereas in the second recording step with two voices present, there exist four timestamps. The same principle applies to the third recording step, resulting in six timestamps. The annotation for our running example "Da Sulisatsa" is illustrated in Table 3.1.

In a subsequent step, we simplify the annotation structure, since we are only interested in the occurrences of the lead voice. In particular, our goal is to divide each recording into three segments, each defined by the start and end point of the lead voice in the current recording step. Consequently, the simplified annotation contains six timestamps: t_{11}, t_{12}, t_{21}, t_{22}, t_{31} and t_{32}, with the first index denoting the segment number and second index denoting start or end position (see Figure 3.1).

For a further simplification step, let us recall the loop-based structure of the given recordings. Ideally, the duration of the lead voice should be the same in all three recording steps, since it is

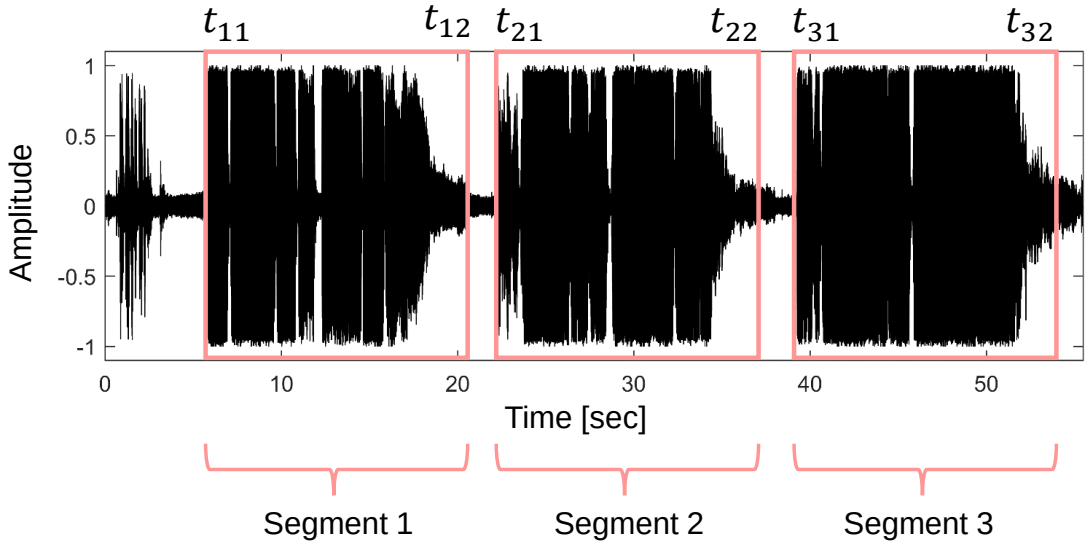

Figure 3.1: Waveform of our running example "Da Sulisatsa" by Artem Erkomaishvili. The segments are marked with red boxes.

simply played back during the recording of middle and bass voice. In practice, due to wow and flutter of the tape recorder, the duration can be slightly shorter or longer, which can easily be proven by looking at Table 3.1:

$$t_2^{\text{orig}} - t_1^{\text{orig}} \neq t_6^{\text{orig}} - t_3^{\text{orig}} \neq t_{12}^{\text{orig}} - t_7^{\text{orig}} \ . \tag{3.1}$$

Furthermore, we also have to account for slight inaccuracies in the reference annotation. However, we neglect these effects in our simplification and require each segment to be of equal length, implying

$$t_{12} - t_{11} = t_{22} - t_{21} = t_{32} - t_{31} \ . \tag{3.2}$$

Given this assumption, individually for each recording, the simplified segment annotation is derived from the original annotation as follows: the start and end points of the lead voice in the first recording step are taken from the original annotation, leading to $t_{11} = t_1^{\text{orig}}$ and $t_{12} = t_2^{\text{orig}}$. From these two timestamps, the segment duration Δt for all three segments can easily be computed by setting

$$\Delta t = t_{12} - t_{11} \ . \tag{3.3}$$

The second segment is defined by $t_{21} = t_1^{\text{orig}}$ and $t_{22} = t_{21} + \Delta t$. In a similar manner, we compute the borders of the third segment as $t_{31} = t_7^{\text{orig}}$ and $t_{32} = t_{31} + \Delta t$. The simplified annotation for our running example "Da Sulisatsa" is illustrated in Table 3.2.

Note that applying these rules to all recordings led to problems with some files, where the end of the third segment overran the length of the corresponding audio file due to the inaccuracies

Description	Variable	Timestamp in Seconds
Start First Segment	t_{11}	5.706122448
End First Segment	t_{12}	20.60408163
Start Second Segment	t_{21}	22.11428571
End Second Segment	t_{22}	37.012244898
Start Third Segment	t_{31}	39.24897959
End Third Segment	t_{32}	54.146938775

Table 3.2: Simplified segment annotation with six timestamps for "Da Sulisatsa".

mentioned above. In these cases, we slightly decreased the segment duration Δt (typically by a couple of milliseconds) in order to fit the end of the third segment to the total length of the recording. This ensures that our constraint formulated in (3.2) is fulfilled in every annotation file.

The simplified/refined segment annotations in CSV format are publicly available and can be downloaded at [29].

3.2 Matching-Based Segmentation

In this section, we interpret the previously defined segmentation task as a standard audio retrieval problem: given a suitable query and a database document, we want to find all occurrences of the query in the database using an audio matching technique.

Two well-known audio matching techniques are *Dynamic Time Warping* and *Diagonal Matching* (see [26][Section 7.2]). Inspired by [20], our approach uses simple diagonal matching, which we will shortly explain in Section 3.2.1. In Section 3.2.2, we outline our informed baseline procedure based on diagonal matching. Subsequently, we discuss suitable feature representations as input for our baseline procedure in Section 3.2.3. The obtained matching curves are evaluated in Section 3.2.4 and the segmentation results are discussed in Section 3.2.5. Finally, in Section 3.2.6, we show how diagonal matching can be used for segmentation with less prior knowledge.

3.2.1 Diagonal Matching

Diagonal Matching is a matching technique to measure the similarity between a query audio fragment $\mathcal{Q}$ and a database recording $\mathcal{D}$. In our description, we follow the notation in [26][Section 7.2.2].

Let us assume we are given a feature sequence $X = (x_1, x_2, ..., x_N)$ of length N corresponding to $\mathcal{Q}$ and a feature sequence $Y = (y_1, y_2, ..., y_M)$ of length M corresponding to $\mathcal{D}$. In order to find out where the query occurs in the database, we shift X over Y and locally compare the subsequences of X and Y with a suitable distance measure $c(x, y)$. Consequently, in regions similar or equal to the query, the distance to the database will be small.

In this work, we use the euclidean distance and the cosine distance as distance measures. The euclidean distance is defined by

$$c_{\text{euclid}}(x, y) := \|x - y\| \ . \tag{3.4}$$

The cosine distance is defined as

$$c_{\text{cosine}}(x, y) := 1 - \frac{\langle x|y\rangle}{\|x\| \cdot \|y\|} \ , \tag{3.5}$$

with $\|x\|$ being the norm of vector x and $\langle x|y\rangle$ being the inner product of x and y. If one of the vectors x or y is 0, $c_{\text{cosine}}(x, y) := 0$. Note that $c_{\text{euclid}} \in \mathbb{R}^+$ whereas $c_{\text{cosine}} \in [0, 1]$. Both distance measures will return values close to 0 if x is similar to y. Conversely, if x and y are not similar, c_{cosine} will return a value close to 1 and c_{euclid} a high positive real number.

Computing the distance measures for all subsequences and all possible shifts, we obtain the matching function or matching curve Δ_{Diag}, which indicates how similar query and database are at a specific shift m:

$$\Delta_{\text{Diag}}(m) := \frac{1}{N} \sum_{n=1}^{N} c(x_n, y_{n+m}) \ , \tag{3.6}$$

with $n \in [1 : N]$ and $m \in [1 : M]$.

The matching function Δ_{Diag} can also be interpreted as the normalized sum of the diagonals of a cost matrix $\mathbf{C} \in \mathbb{R}^{N \times M}$ given by

$$\mathbf{C}(n, m) := c(x_n, y_m) \ , \tag{3.7}$$

which gives diagonal matching its name.

3.2.2 Informed Baseline Procedure

The proposed informed baseline procedure requires the first segment of the recording to be given as a query and the whole recording given as database for diagonal matching. Then, our segmentation task reformulates as follows: given a feature representation of the first segment as a query and the same feature representation of the whole recording as the database, we apply diagonal matching to find all occurrences of the query in the database recording.

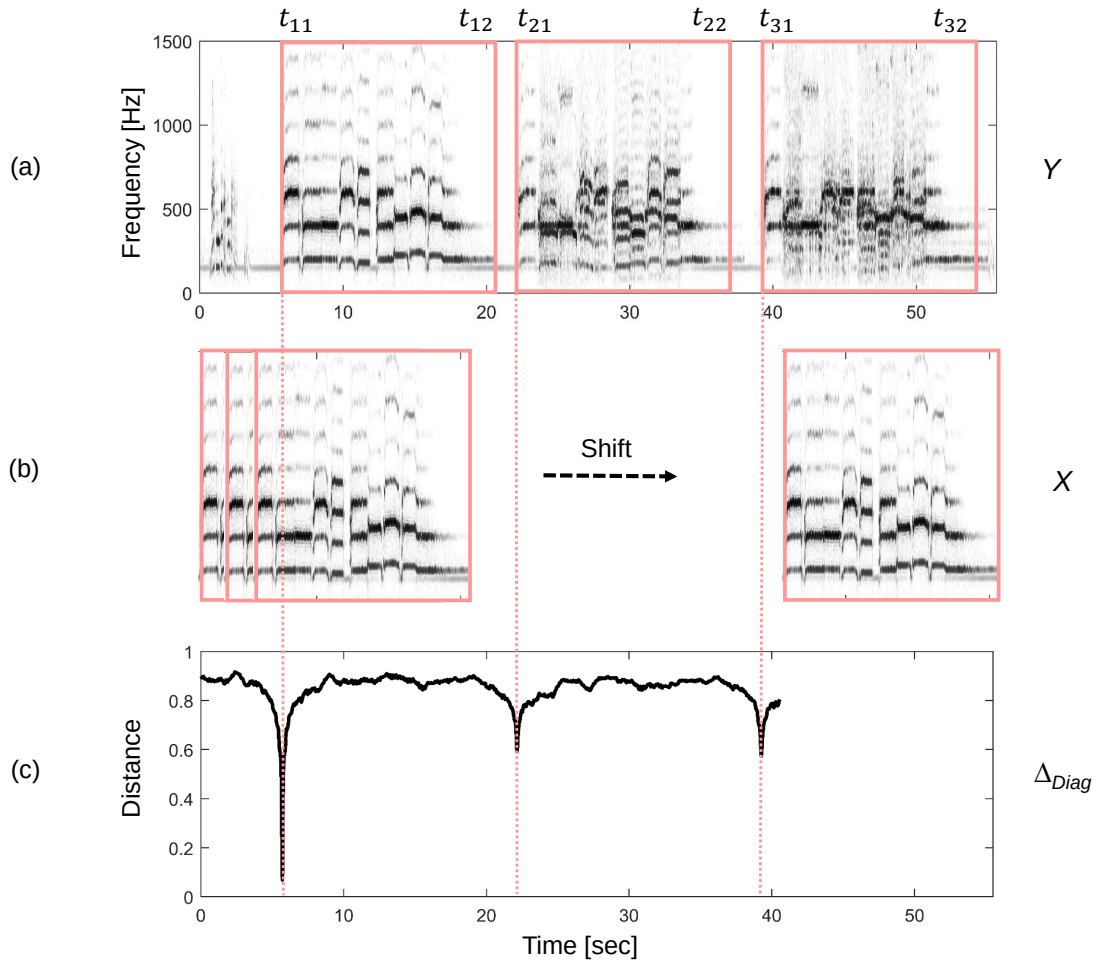

Figure 3.2: Baseline matching procedure illustrated on "Da Sulisatsa". **(a)** Feature representation of database. **(b)** Feature representation of query (first segment) being shifted over the database. **(c)** Matching function obtained by diagonal matching.

According to the diagonal matching procedure described in Section 3.2.1, the feature representation of the query X is shifted over the feature representation of the database Y and locally compared using the given distance measures. The matching process is illustrated in Figure 3.2a. Ideally, the obtained matching function Δ_{Diag} should indicate small distances between query and database at the starting points of the three segments t_{11}, t_{21}, and t_{31}. At all other points in time, Δ_{Diag} should indicate a higher distance between query and database, as depicted in Figure 3.2b. Note that we require the distance to be close to zero at point t_{11}, since the query is originally taken from the recording (for further details, see Section 3.2.3).

In a subsequent step, the estimated starting points of all three segments are obtained by using a suitable peak-picking algorithm. The corresponding end points t_{12}, t_{22} and t_{32} are derived by adding the query length to each of the estimated starting points.

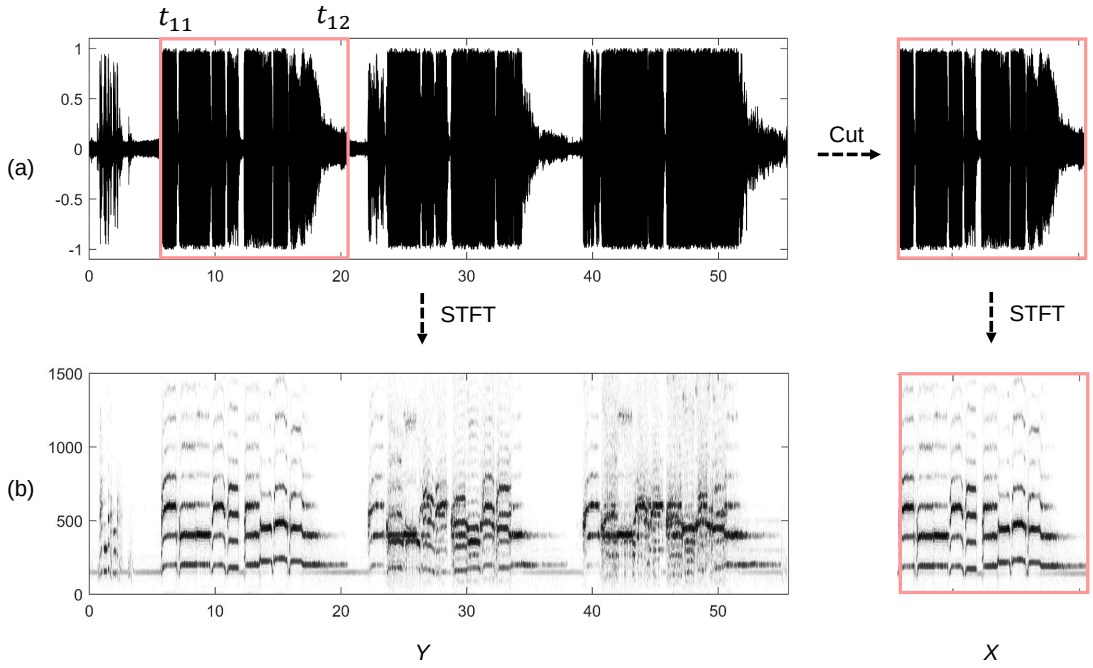

Figure 3.3: Generation of query and database feature representation. **(a)** Query is cut out from the time domain audio recording. **(b)** Query and database are transformed separately to a time-frequency representation.

3.2.3 Feature Representations

For our experiments, we use two types of input feature representations for both database and query, which are generated as follows. First, the query is cut out from the time domain audio file using timestamps t_{11} and t_{12} from the corresponding segment annotation, as illustrated in Figure 3.3a. Afterwards, as shown in Figure 3.3b, query and database are separately transformed to the time-frequency domain using the STFT introduced in 2.2.1. The magnitude of the computed STFT coefficients represents the first feature type. The second feature representation is the refined log-frequency spectrogram representation introduced in Section 2.2.2, which is based on the STFT.

The STFT parameters turned out to have a great impact on the matching curves. Recall that the given collection contains transcoded WAV files with a sampling rate of 22 050 Hz. In our first experiments, we used a STFT with window length $N = 64$ ms and a window overlap of $H = N/2$, as in [11]. Using these settings, the measured overall similarity was very low (see black curve in Figure 3.4). Even the peak at t_{11} was far from zero in the tested feature-distance measure combinations.

The main reason for this observation is the small window length. A larger window length increases the overall similarity, but at the same time broadens the peaks due to blurring (purple curve

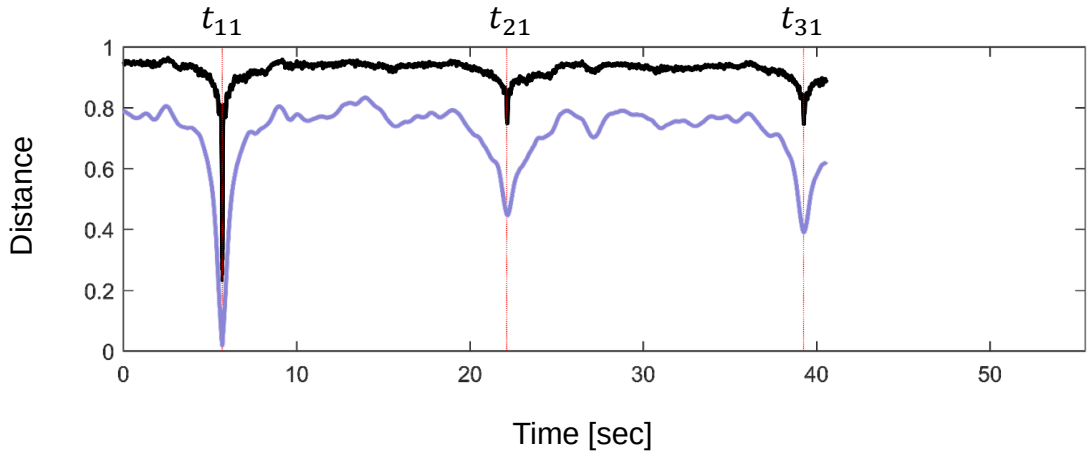

Figure 3.4: Two matching curves generated by diagonal matching based on refined log-frequency spectrograms and the cosine distance measure. The dashed vertical red lines indicate the reference annotation. The black curve was generated with the STFT parameters $N = 1024$ samples and $H = 512$ samples similar to [11]. The purple curve was generated with the STFT parameters $N = 16\,384$ samples and $H = 512$ samples.

in Figure 3.4). After trying several parameter settings, we decided to set the window length to $N = 4096$ samples ≈ 186 ms and the hopsize to $H = N/8 \approx 23$ ms. Note that we do not claim these settings to be optimal in this scenario. However, these settings produced accurate matching curves with good peak quality within reasonable computation time.

3.2.4 Evaluation of Matching Curves

In this section, we evaluate the quality of the peaks depending on the used features and distance measures. Note that we use two feature representations (magnitude spectrogram and the refined log-frequency spectrogram) and two distance measures (euclidean and cosine distance), leading to four different feature-distance measure combinations.

As in [27] and [16], we divide the matching curves into false alarm regions and peak regions. In false alarm regions, the similarity of the query X and the database Y should be low. In order to capture the characteristics of the false alarm regions, we define μ_{F}^{X} and $\min_{\mathrm{F}}^{X}$ as the mean and the minimum of Δ_{Diag} within the false alarm regions. Peak regions contain the true matches (local minima) of the matching curve and their neighborhood. Recall that we expect every matching curve to have three peaks around t_{11}, t_{21} and t_{31}. Since we are given the segment annotations for every file in the collection, we can define the peak neighborhoods by adding a neighborhood parameter κ given in seconds to the left and right of every expected peak position. Furthermore, we define $\min_{\mathrm{T}}^{X}$ to be the minimum/peak value of each peak region.

The above metrics can be combined into two quality measures, which we compute for each of

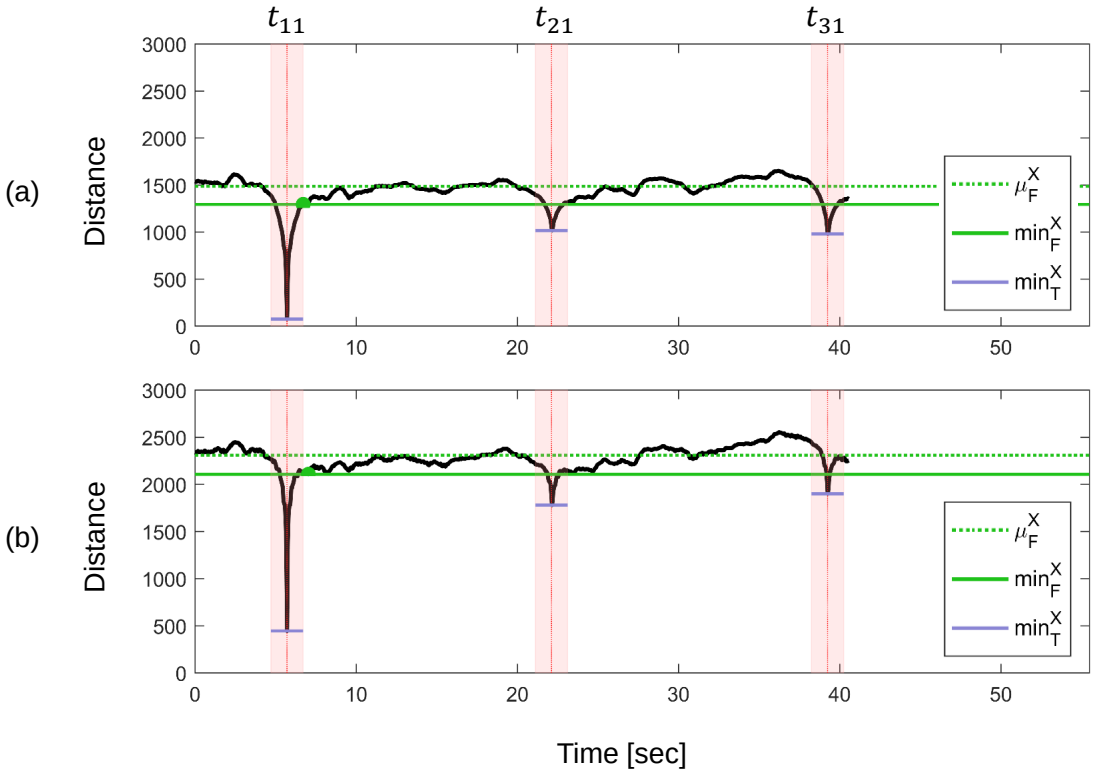

Figure 3.5: Matching curves for "Da Sulisatsa" using the euclidean distance measure. The various evaluation metrics are indicated by horizontal lines. The peak regions defined by $\kappa = 1\,\text{s}$ are marked with light red. The reference positions are marked by vertical red lines. The green dot indicates the position in the matching curve that corresponds to $\min_F^X$. **(a)** Spectrogram-based matching **(b)** Refined log-frequency spectrogram-based matching.

the peak regions separately. We define $\alpha^X := \min_T^X / \min_F^X$ and $\beta^X := \min_T^X / \mu_F^X$. Both quality measures should be close to zero in case of distinct peaks. α^X can be considered as a rather strict measure in our scenario, since it sets the peak values in relation to the minimum of the false alarm regions. β^X can be seen as a rather soft evaluation measure, since it sets the peak values in relation to the mean of the false alarm regions.

The evaluation measures for our running example are visualized as horizontal lines in Figures 3.5 and 3.6. In both figures we set the STFT parameters to $N = 4096\,\text{samples}$, $H = N/8$ and defined the peak regions with $\kappa = 1\,\text{s}$.

Figure 3.5 shows matching curves obtained by diagonal matching using the euclidean distance and magnitude spectrograms in (a) and refined log-frequency spectrograms in (b). Comparing (a) and (b), we notice that using refined log-frequency spectrograms noticeably increases μ_F^X, but at the same time also increases $\min_T^X$. We can also see that in (b) the peak at t_{11} is not close to zero anymore, which can be traced back to the chosen STFT parameters. Furthermore, the peaks are narrower in Figure 3.5b compared to Figure 3.5a, which also leads to an increase in $\min_F^X$.

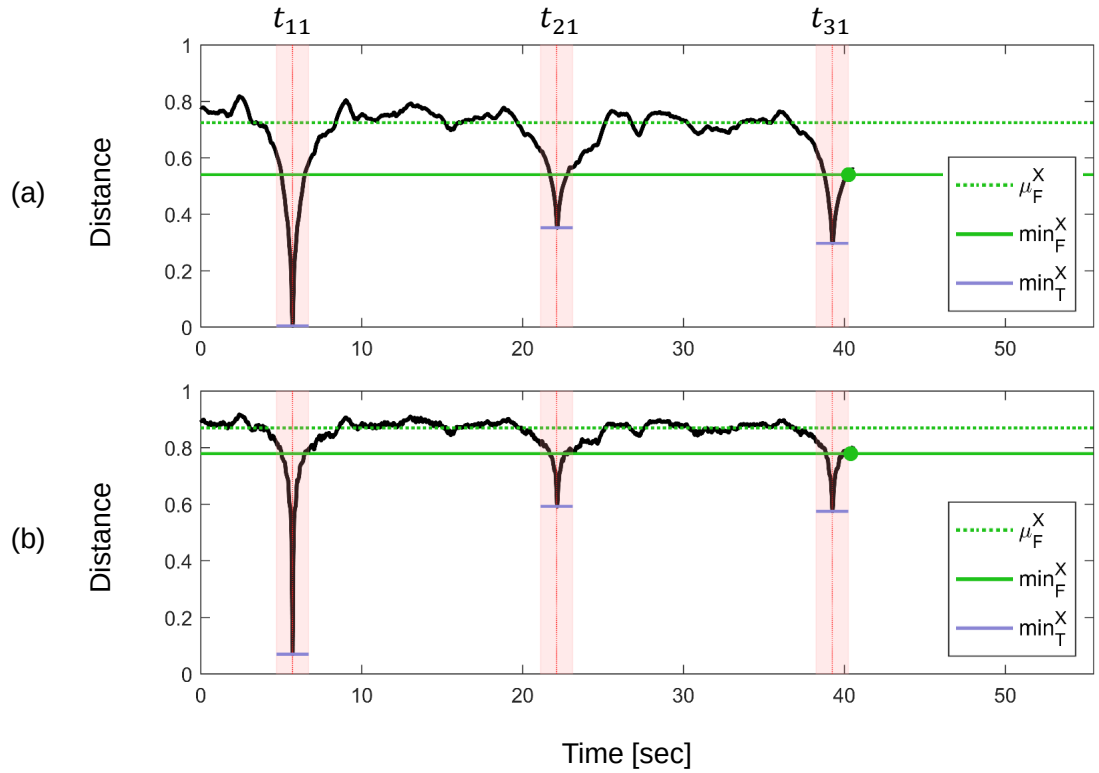

Figure 3.6: Matching curves for "Da Sulisatsa" using the cosine distance measure. The various evaluation metrics are indicated by horizontal lines. The peak regions defined by $\kappa = 1\,\mathrm{s}$ are marked with slight red. The reference positions are marked by vertical red lines. The green dot indicates the position in the matching curve that corresponds to $\min_F^X$. **(a)** Spectrogram-based matching **(b)** Refined log-frequency spectrogram-based matching.

Figure 3.6 shows matching curves obtained by diagonal matching using the cosine distance and magnitude spectrograms in (a) and refined log-frequency spectrograms in (b). Comparing (a) and (b), we notice similar effects regarding means and peak widths as in Figure 3.5. Comparing both figures 3.5 and 3.6, we assert that the cosine distance measure produces more prominent peaks in the shown matching curves than the euclidean distance.

In the following, the observations described above are evaluated on the whole set of recordings in order to see whether they can be generalized to all recordings. Note that the chant "Adide sulo chemo" (GCH_048_Erkomaishvili.wav) was excluded from our evaluation, since the recording is duplicated, resulting in six peaks in the matching curve (see Figure B.3 in the Appendix). The results rounded to two decimals are shown in Tables 3.3, 3.4, 3.5, and 3.6.

The tables show that, as expected, α exhibits higher values than β. Focusing on the results of the first segment, α and β are close to zero throughout all tables. The measures also indicate that the quality of peak two and three corresponding to the beginning of the second and third segment is similar on average, but also depends on the chosen features and distance measures.

Description	$\min_T$	μ_F	$\min_F$	α	β
Peak Region 1	35.20	808.74	711.63	0.05	0.05
Peak Region 2	646.65	808.74	711.63	0.94	0.81
Peak Region 3	637.05	808.74	711.63	0.95	0.81

Table 3.3: Evaluation measures for diagonal matching using spectrogram representations and euclidean distance averaged over whole collection.

Description	$\min_T$	μ_F	$\min_F$	α	β
Peak Region 1	209.59	1217.62	1096.07	0.20	0.18
Peak Region 2	1072.27	1217.62	1096.07	1.01	0.89
Peak Region 3	1076.88	1217.62	1096.07	1.04	0.90

Table 3.4: Evaluation measures for diagonal matching using refined log-frequency spectrogram representations and euclidean distance averaged over whole collection.

Description	$\min_T$	μ_F	$\min_F$	α	β
Peak Region 1	0.00	0.72	0.63	0.01	0.00
Peak Region 2	0.48	0.72	0.63	0.76	0.67
Peak Region 3	0.46	0.72	0.63	0.73	0.64

Table 3.5: Evaluation measures for diagonal matching using spectrogram representations and cosine distance averaged over whole collection.

Description	$\min_T$	μ_F	$\min_F$	α	β
Peak Region 1	0.07	0.88	0.84	0.09	0.08
Peak Region 2	0.73	0.88	0.84	0.86	0.82
Peak Region 3	0.73	0.88	0.84	0.86	0.82

Table 3.6: Evaluation measures for diagonal matching using refined log-frequency spectrogram representations and cosine distance averaged over whole collection.

Comparing the distance measures, α and β indicate for all three peaks that the cosine measure produces better results than the euclidean distance. The main reason for this is the vulnerability of the euclidean distance to changes in dynamics (recording volume/level) that frequently occur in this collection due to the looping structure of the recordings.

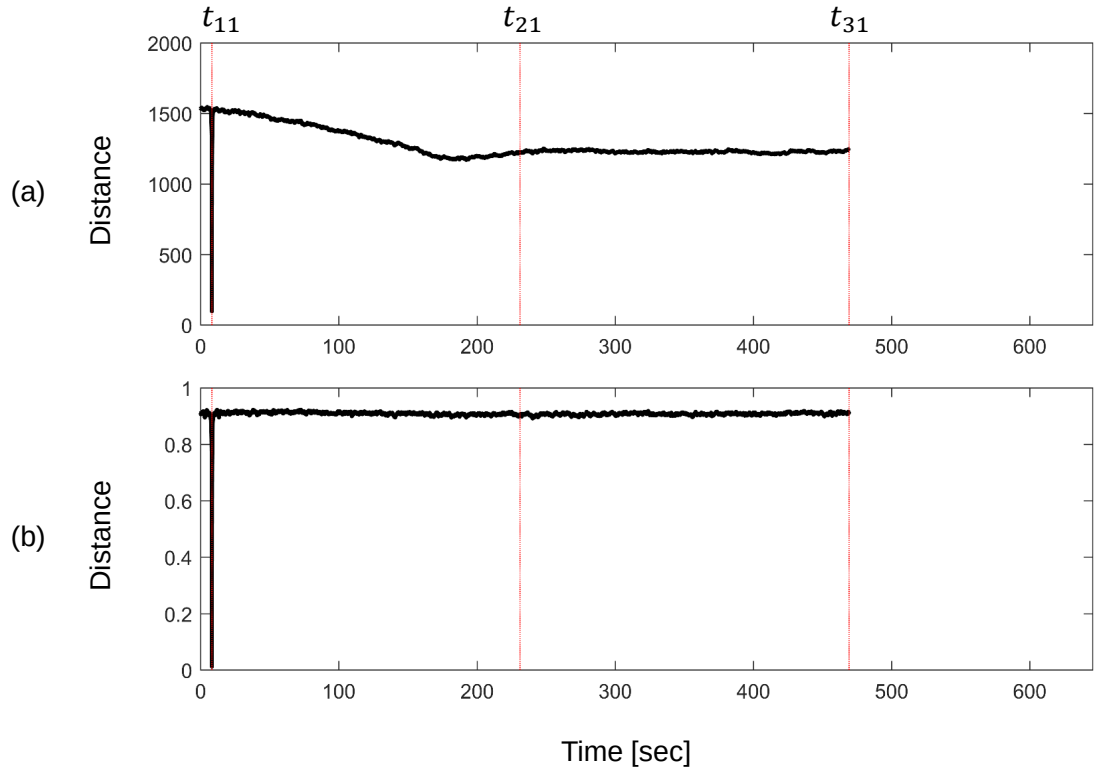

Figure 3.7: Matching curves for the chant "Ats' dzalni tsatani" (GCH_107_Erkomaishvili.wav) where segments two and three could not be detected due to changing tempo and pitch. **(a)** Euclidean distance; a drop in dynamics is visible. **(b)** Cosine distance; the drop in dynamics is not visible due to normalization.

By looking at Figure 3.7a, we see that the matching curve slightly drops in the course of the recording. This effect is compensated in the cosine distance measure due to normalization, which can easily be seen in Figure 3.7b. In general, Figure 3.7 illustrates a worst case scenario (file GCH_107_Erkomaishvili.wav). The velocity of the tape recorder drops during the recording, leading to audible time stretching and pitch shifting artifacts. Under these conditions, it is not possible to detect segments two and three using diagonal matching.

Comparing feature representations, α and β indicate better results for the spectrogram than for the refined log-frequency spectrogram. However, $\min_{\mathrm{F}}^{X}$ and μ_{F}^{X} behave contrarily. This can be explained by looking back at the different representations in Figure 2.2c and Figure 2.3b. Figure 2.2c contains more noise and is rather blurred compared to Figure 2.3b, which increases the general similarity. The refined resolution in Figure 2.3b produces fine spectral lines with lower magnitude and reduces the noise-like components in the spectrogram while decreasing the overall similarity.

3.2.5 Segmentation Results

In this section, we evaluate the accuracy of the peaks in the matching curves of the given recordings. More specifically, we select the three most prominent peaks from every matching curve with a peak-picking algorithm and compare them to the corresponding reference annotations. As a baseline for our peak-picking, we use the matching curves generated by diagonal matching with refined log-frequency representations and the cosine distance, which exhibited the narrowest peaks in the qualitative evaluation in Section 3.2.4.

The peak-picking algorithm works as follows. Assume we are given a matching curve Δ_{Diag} with three distinct peaks generated from a query of length N and a database of length M according to Section 3.2.2. The goal is to find the estimates for t_{11}, t_{21} and t_{31} denoted by $\hat{t}_{11}$, $\hat{t}_{21}$ and $\hat{t}_{31}$. To this end the algorithm first picks the minimum of Δ_{Diag}, which is very likely to correspond to time point t_{11}. In a second step, the algorithm excludes the neighborhood of the peak by setting the values in Δ_{Diag} to the left and to the right to ∞. In our work, we exclude $\varepsilon = N/2$ from each side of the peak. In a subsequent step, the algorithm picks again the minimum of the modified curve. The whole procedure is repeated three times until all three estimates are found. The peak-picking is outlined in Algorithm 1.

Algorithm 1 Peak-picking

Input : Δ_{Diag}, $\varepsilon = N/2$
Output : $\hat{t}_{11}$, $\hat{t}_{21}$, $\hat{t}_{31}$

for $i = 1 : 3$ **do**
  $\hat{t}_{i1} = \min(\Delta_{\text{Diag}})$;
  $\Delta_{\text{Diag}}(\hat{t}_{i1} - \varepsilon : \hat{t}_{i1} + \varepsilon) = \infty$;
end

To evaluate peak accuracy we compute the absolute difference Δt_{i1} between the reference positions t_{i1} and the estimated positions $\hat{t}_{i1}$ with $i \in [1 : 3]$

$$\Delta t_{i1} = |t_{i1} - \hat{t}_{i1}| \,. \tag{3.8}$$

Furthermore, we define a tolerance τ in seconds that defines whether a peak is considered as a match with the reference. In other words, if $\Delta t_{i1} \leq \tau$ we consider the peak as a correct match, if $\Delta t_{i1} > \tau$ we consider the peak as an outlier. We evaluated all matching curves on logarithmically spaced τ-values in the range of 10^{-3} to 10 s as shown in Figure 3.8. For a given τ, the figure shows the number of outliers in percent for each Δt_{i1}. Note that deviations from the reference may also be caused by inaccuracies in the segment annotations, which may lie in the same evaluation range.

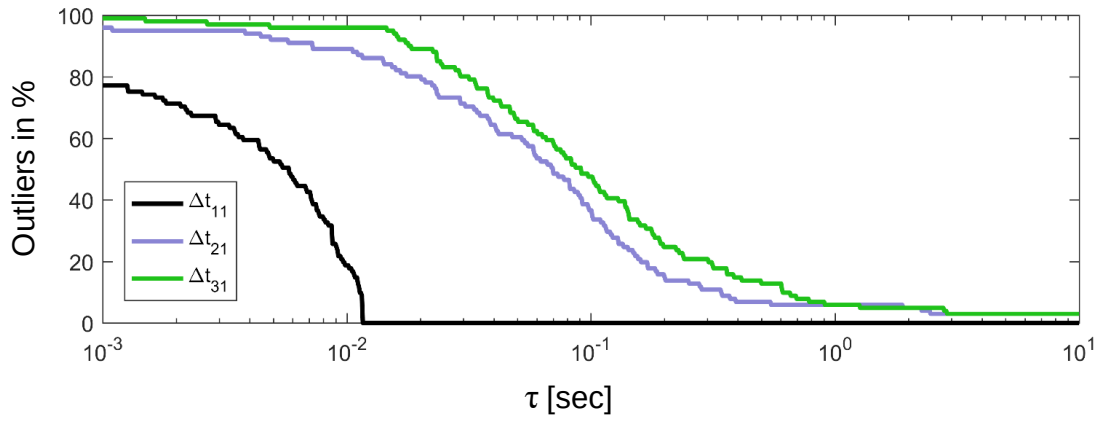

Figure 3.8: Evaluation of peak accuracy for each peak individually on a logarithmic τ-axis. All values of Δt_{i1} that do not lie within the tolerance τ are considered as outliers.

As expected, Figure 3.8 shows a large number of outliers for very small values of τ. However, about 21 % of the Δt_{11} values are even closer to the reference than 10^{-3} s, which is indicated by the black curve starting around 79 % outliers. This is of course of limited significance when regarding the limited annotation accuracy of a human. Estimates of the first peak t_{11} can be considered as the most accurate, since all peaks lie within a maximum tolerance of $0.012 \approx 10^{-2}$ s. For a tolerance of $\tau = 10^0 = 1$ s all curves have reached a low level of outliers. The remaining outliers for Δt_{21} and Δt_{31} of about 4 % can mainly be traced back to tape recorder issues and noise (see Figure 3.7). Matching curves for all recordings where the second and/or third segment could not be found are given in Appendix B.

3.2.6 Towards Blind Segmentation

In this section, we show how diagonal matching can be used to detect the segment borders without any prior knowledge about the first segment. To this end, let us recall the baseline procedure shown in Figure 3.2. In our previous experiments, a feature representation of the first segment was chosen as a query based on the known segment borders.

A more uninformed approach takes the first third of the recording as a query, assuming that the first segment is largely included in the chosen query. A matching curve for this new scenario is shown in Figure 3.9a. The matching curve exhibits again three distinct peaks. The first peak is located at $t = 0$ s, which corresponds to the starting point of our query. All three peaks have an offset of Δt relative to the reference starting positions. Further experiments with diagonal matching for determining this offset, as well as the segment length to find the end points, did not produce reliable results.

A different approach is based on the silence/noise at the beginning of the recording and the

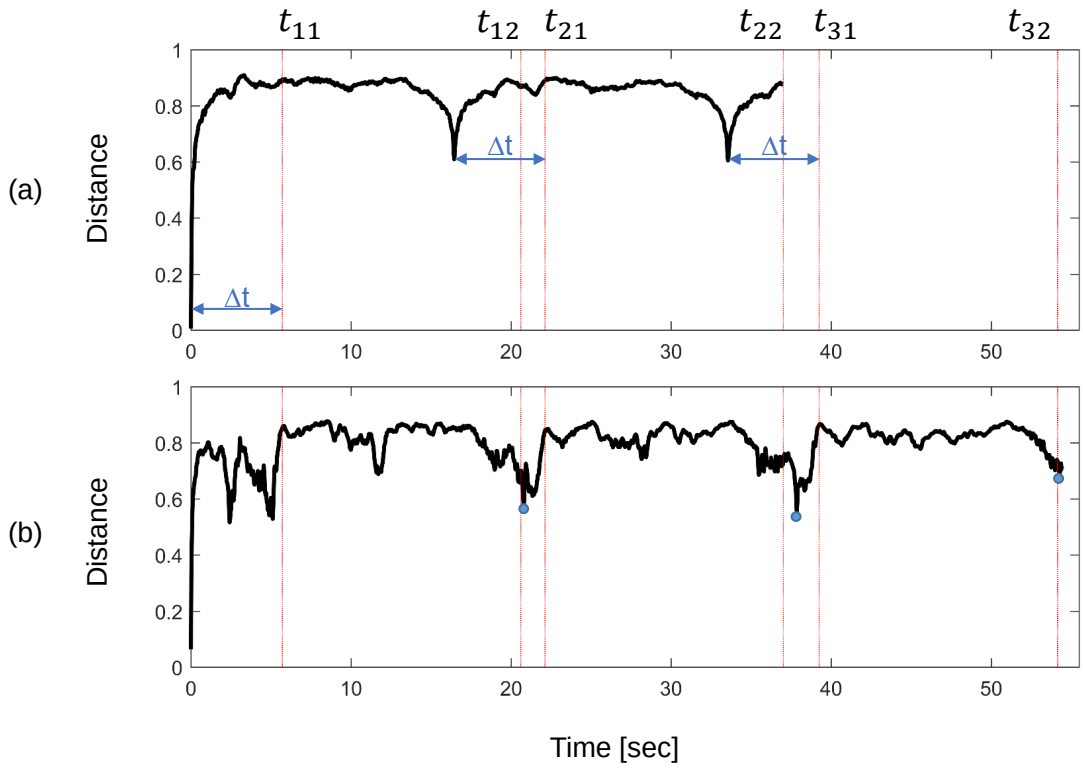

Figure 3.9: Less informed segmentation using diagonal matching. The vertical red lines denote the six reference segment borders. **(a)** Diagonal matching with the first third of the recording as query. **(b)** Diagonal Matching using the first second of the recording as query. The minima between the segments are marked in blue.

pauses in between the segments. This time, we take the first second of the recording as a query, which ideally contains only noise or tape recorder sounds. The resulting matching curve shown in Figure 3.9b. It exhibits local minima at the beginning of the recording and in between the segments (see blue dots). One way to get the final segmentation result would be to divide the obtained matching curve in "segment" and "non-segment" regions with a suitable threshold and then infer the segment borders.

3.3 Classification-Based Segmentation

In this section, we interpret the segmentation task as a classification problem. To this end, recall that each of the segments contains at least one singing voice that is present throughout the whole segment, potentially with short pauses in between. Outside of the segment borders, there is no voice present, except at the beginning, where Artem Erkomaishvili introduces the title of the chant he is going to perform (see Section 2.1.2). However, based on these observations, we consider parts of the recording where a singing voice is present to be located within one

of the three segments. Ideally, these parts cluster in three homogeneous units, reflecting the three-segment structure of the recordings. In this way, we transform the original segmentation problem into a singing voice detection/classification problem.

For the classification into "vocal" and "non-vocal" parts, we apply a machine-learning-based random forest classifier on the recordings, which was originally used for singing voice detection in classical opera recordings by Dittmar et al. [8]. A random forest is a classification method which consists of several binary decision trees. Each decision tree classifies input data based on a randomly assigned (sub-)set of classification criteria. Combining the decisions of all trees leads to the final classification result. Further details on decision trees and machine-learning in general can be found in [2, Chapter 14.4], [15, Chapter 15] and [44, Chapter 7.8].

The baseline procedure for this segmentation approach is explained in Section 3.3.1. Subsequently, we briefly present the classification results in Section 3.3.2.

3.3.1 Baseline Procedure

The baseline procedure closely follows the work of Dittmar et al. [8].

Machine-learning-based classification is typically conducted in three phases, including training the classifier, validation for parameter tuning, and testing with the chosen parameter settings to get final evaluation results.

In a first step, we split the dataset of Georgian chant recordings into a training, validation and test set. More specifically, we randomly assign 70 % of the recordings to the training set (70 tracks), 15 % to the validation set (15 tracks), and the remaining 15 % to the test set (15 tracks). We again exclude GCH_048_Erkomaishvili.wav from our experiment. Furthermore, we slightly bias the random splitting process by requiring our running example "Da Sulisatsa" to be within the test dataset.

In a second step, we compute different low- and mid-level audio features on each of the three subsets. The used feature types are mel-frequency cepstral coefficients, vocal variance, fluctogram variance, spectral contraction variance, spectral flatness mean, and polynomial shape spectral contrast, identically to the ones used and explained in [8]. Furthermore, our random forest is designed to have 128 individual trees. In the first phase, we train each of these trees with a random subset of features of the training set and the corresponding reference segment annotations. In this way, each tree bases its decisions on different classification criteria (feature types).

Applying such a trained random forest on audio feature representations results in a score value between 0 and 1 for each time frame and class. Concatenation of these score values creates a so called decision function, which can also be interpreted as a confidence measure for the classifier decision. Since in our scenario, we only have two classes, the two decision functions are

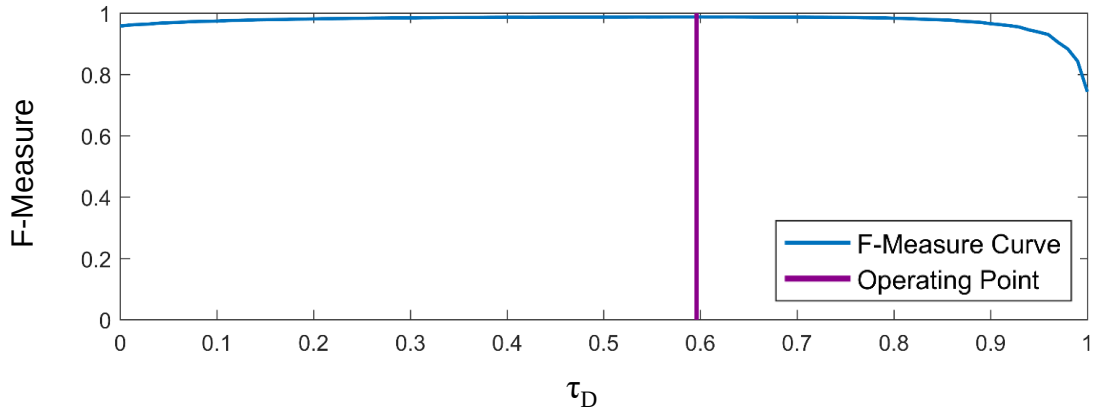

Figure 3.10: F-measure curve for running thresholds τ_{D} on validation dataset.

inversely proportional. In order to get a classification result, we binarize the decision function using a threshold $\tau_{\mathrm{D}} \in [0, 1]$. Only if a score value is larger than τ_{D}, the corresponding frame is considered as "vocal", otherwise as "non-vocal". Instead of simply fixing $\tau_{\mathrm{D}} = 0.5$, we search for a potential value of τ_{D} that delivers a better predictive performance. To this end, we apply the trained random forest classifier on our validation dataset and compare the classification performance for different thresholds.

For the evaluation of classification performance, we use the well-known F-measure [26, Section 4.5.1]. Frames with a positive classification that coincide with an annotated segment in the reference annotation are counted as *true positives* (TP), frames with a negative classification that coincide with an annotated segment are *false negatives* (FN). Frames with a positive classification that do not coincide with an annotated segment are considered *false positives* (FP). These measures are combined in the F-measure by

$$F = \frac{2 \cdot \mathrm{TP}}{2 \cdot \mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \,. \tag{3.9}$$

The resulting F-measure curve for the running threshold is shown in Figure 3.10. The curve is close to 1 for almost all thresholds τ_{D}, which already indicates that the classifier performs well in our segmentation task. The curve reaches its maximum at threshold $\tau_{\mathrm{D}} = 0.596$ with an F-measure of 0.988. We choose this threshold (operating point) for binarizing the decision function in the final test phase, where the trained classifier is applied on the features of the test set.

3.3.2 Classification Results

Figure 3.11a shows the decision function for our running example obtained from applying the trained random forest classifier on the test dataset. The chosen threshold in the validation

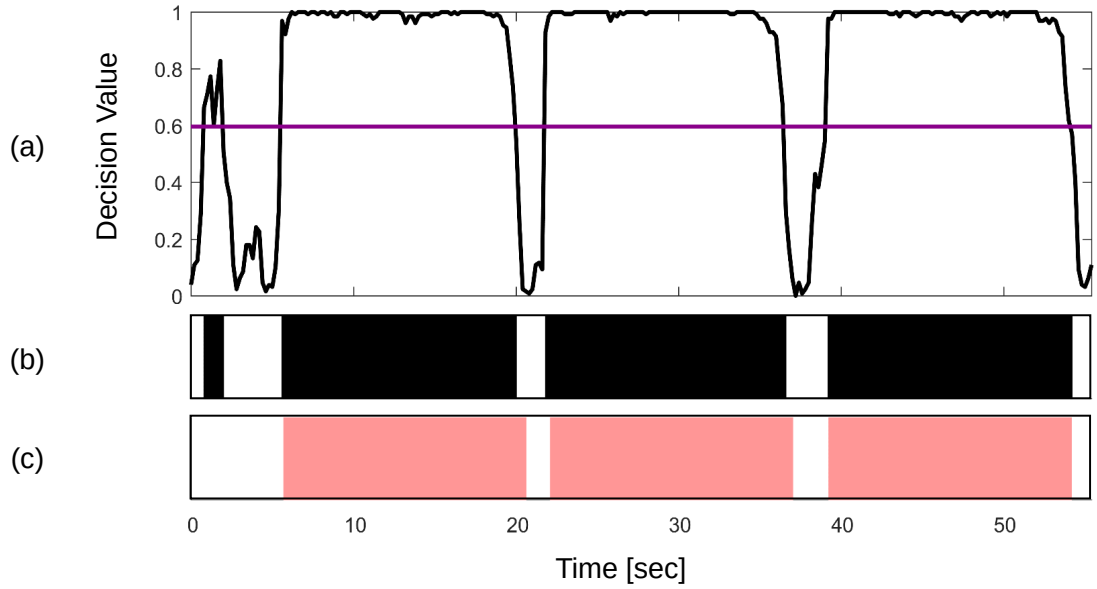

Figure 3.11: Random forest classification. **(a)** Decision function for our running example "Da Sulisatsa". The chosen decision threshold is marked with a solid purple line. **(b)** Activity of automatically detected singing voice. **(c)** Activity of the reference annotation.

phase is again marked with a purple line. The time lines underneath show the activations of our classification approach (Figure 3.11b, black rectangles) and the reference annotations (Figure 3.11c, red rectangles). By having a closer look at the time lines, we see that the activations of our approach form three main homogeneous units which marginally deviate from the reference annotations. The introductory part has also been classified as "vocal", which is formally correct, but is of course an undesired behavior of our classifier for segmentation purposes. In general, we cannot observe any noticeable differences in classification performance between the three segments.

Evaluating the classification results of all 15 items in the test set on the corresponding reference annotations yields an F-measure of 0.947.

3.4 Conclusions and Further Notes

In this chapter, we developed and examined two approaches for segmenting the recordings by Erkomaishvili.

In the informed segmentation scenario, diagonal matching produced good results. The matching curves exhibited accurate and distinct peaks on average. Furthermore, we could show that the quality of the matching curves highly depends on the STFT parameters, feature representations, and distance measures.

Peak maps, which are used in audio fingerprinting (see [26, Section 7.1.2]), can reduce processing time and may also generate more distinct peaks in the matching curves. The proposed methods in Section 3.2.6 are tailored to the recordings of Artem Erkomaishvili and may not generalize to other audio retrieval scenarios. Evaluating these improvements was beyond the scope of this thesis and is left as further research.

The evaluation results for our machine-learning approach indicate that a trained random forest classifier can be suitable for segmenting the given Georgian chant recordings. In this simple scenario, without background music or other interference, the classifier produced accurate results. However, in order to make reliable statements about the classification performance, the baseline procedure described above has to be repeated multiple times (*manifold cross validation*), which was beyond the scope of this thesis. To better understand the classifier and the classification results requires examining the impact of feature types and parameter settings on the decision trees. In this context, it would also be interesting to compare the obtained decision functions with simple, energy-based novelty curves [26, 6.1.1], which should deliver good results for this segmentation task.

One way to eliminate the influence of Erkomaishvili's preface in future classification tasks is to consider only the three largest homogeneous singing voice parts in the binarized decision function. This may also require some post-processing of the curves, e.g. using median filters. With these improvements, the performance of the classifier is expected to become even better.

Chapter 4

Fundamental Frequency Estimation

Another basic task in music processing is fundamental frequency (F0) estimation, where the main goal is to estimate the predominant melody in an audio recording, which can also be seen as an intermediate step for music transcription. Due to the fact that transcriptions are often subjectively influenced, F0 estimation can be a useful tool for ethnomusicologists to gain a more neutral view on the performance practice of Georgian vocal music. In this chapter, we compare multiple F0 estimation algorithms and show how reliable these algorithms perform on the Georgian chant recordings described in Section 2.1.2. To this end, we first give a brief introduction to F0 estimation in Section 4.1. Secondly, in Section 4.2, we explain the generation of reference annotations with specific focus on the interactive graphical user interface (GUI) that we used. In Section 4.3, we describe the metrics for our evaluations. Subsequently, we introduce some standard F0 estimation algorithms based on time domain and enhanced time-frequency domain (*salience*) representations in Section 4.4 and Section 4.5. In this context, we evaluate these algorithms segment-wise on our running example "Da Sulisatsa" to further illustrate their properties. In Section 4.6, we show how a salience-based algorithm can be adapted to the Georgian music scenario. We evaluate all introduced algorithms on the whole dataset in Section 4.7. Finally, we draw conclusions and outline future work in Section 4.8.

4.1 Background

The description in this section follows [30].

Audio recordings (given as waveforms) are complex in the sense that musical parameters such as pitches, note onsets, or note durations are not explicitly given. Furthermore, real-world sounds are far from being simple pure tones with well-defined frequencies. Playing a single note on an instrument may result in a complex sound that contains a mixture of different frequencies

changing over time. Intuitively, such a musical tone can be described as a superposition of pure tones or sinusoids, each with its own frequency of vibration, amplitude, and phase. A *partial* is any of the sinusoids by which a musical tone is described. The frequency of the lowest partial present is called the *fundamental frequency* (abbreviated as F0) of the sound. The *pitch* of a musical tone is usually determined by the fundamental frequency, which is the one created by vibration over the full length of a string or air column of an instrument. For further details, we refer to [26, Chapter 1].

When given an audio recording, one central task in music processing is referred to as *melody extraction*. In general terms, a *melody* (or more generally a *melodic voice*) may be defined as a linear succession of musical tones expressing a particular musical idea. Because of the special arrangement of tones, a melody is perceived as a coherent entity. When performed by a singer or played on an instrument, the melody corresponds to a sequence of frequency values rather than notes. Also, as opposed to a notated symbolic representation, some of the notes may be smoothly connected (e. g., when singing a glissando). Furthermore, one may observe pronounced frequency modulations due to vibrato. Given an audio recording, melody extraction is often understood as the task of estimating the sequence of frequency values that correspond to the main melody [14, 33, 37].

In other words, rather than estimating a sequence of notes, the objective is to determine a sequence of frequency values that correspond to the notes' pitches. Such a frequency path over time, which may also capture continuous frequency glides and modulations, is referred to as a *frequency trajectory*. In particular, one is interested in the fundamental frequency values of a melody's notes. The resulting trajectory is also called the melody's F0 trajectory. For further details, we refer to [26, Chapter 8].

Estimating the fundamental frequency of a quasiperiodic signal, termed *monopitch estimation*, is a long-studied problem with applications in speech processing. For a review of early contributions we refer to [17]. While monopitch estimation is now achievable with a reasonably high accuracy, the problem of *multipitch estimation* with the objective of estimating the fundamental frequencies of concurrent periodic sounds remains very challenging. This particularly holds for music signals, where concurrent notes stand in close harmonic relation.

In this chapter, the F0 estimation task formulates as follows: given the set of Georgian chant recordings, the goal is to estimate the F0 trajectories of lead, middle and bass voice in each of the recordings. In the first segment, the task is a simple monopitch estimation problem, since only the lead voice is present. The task becomes more difficult in the polyphonic second segment where lead and middle voice are present. The F0 estimation task in the third segment is most difficult, since all three voices are present in the recordings (see Figure 2.1). However, throughout all segments, the trajectories may suffer from so-called *octave jumps* or *octave errors*, which result from confusions between the fundamental frequency and higher harmonics.

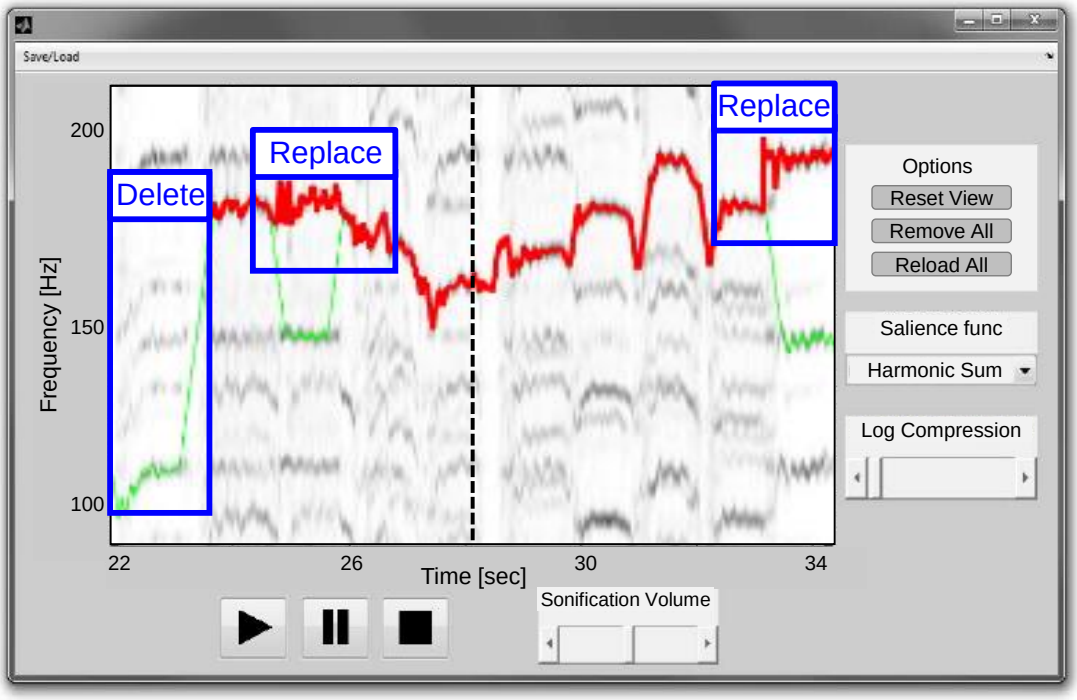

Figure 4.1: Graphical user interface for interactive F0 trajectory estimation (from [30]).

In general, F0 estimation algorithms can be categorized in time domain algorithms, which estimate the F0 trajectories from the time domain signal and salience-based algorithms that estimate the F0 trajectories from an enhanced time-frequency representation. A broad overview about state-of-the-art algorithms and their performances on different datasets is provided by the annual MIREX (*Music Information Retrieval Evaluation eXchange*) challenges [24] and in the article by Salamon et al. [37].

4.2 Generating Reference Annotations

During this thesis project, we relied on manual reference annotations generated with an interactive F0 annotation tool developed by Drieger et al. [10]. In this section, we elaborate on the generation of these reference annotations with specific focus on the GUI, following the description in [30]. Note that the underlying salience-based F0 estimation procedure is explained in Section 4.5.

The GUI allows a user to interactively generate and correct frequency trajectories. Figure 4.1 shows a screenshot of this GUI, which integrates an enhanced time-frequency representation of the audio signal as its central visual element. On top of this representation, a previously specified frequency trajectory can be plotted (green line). The GUI integrates the features of a standard audio player, with buttons for starting, pausing, and stopping the playback of the loaded music

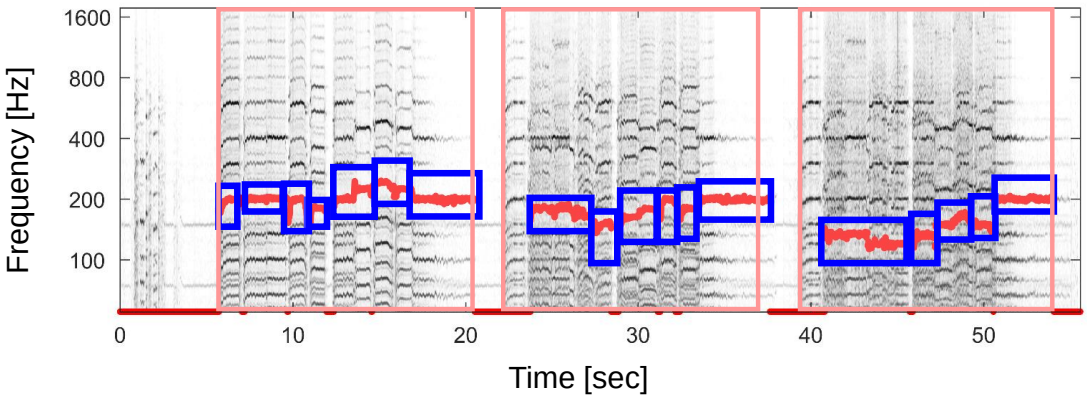

Figure 4.2: Reference F0 trajectory for our running example (red). The constraint regions are marked with blue boxes.

recording at the bottom of the interface. When playing back the music recording, the respective time position is indicated by a vertical dashed playback bar running across the time-frequency representation. This way, salient structures in the visual representation can be directly compared to the auditory cues in the recording. Additionally, the interface allows for playing back a sinusoidal sonification of the specified frequency trajectory (acoustically superimposed with the original audio recording). This allows the user to easily understand the accuracy of the current trajectory simply by listening.

As another important feature, the GUI allows a user to correct a given frequency trajectory. To this end, the rough location of a frequency trajectory can be specified by means of a rectangular box (as indicated by the blue boxes). These boxes are used as constraint regions to recompute frequency trajectories within these regions, while previously specified trajectories within the corresponding time windows are replaced. To allow extremely fine-grained corrections, the user may even use an editing option for drawing free-hand trajectories. To further simplify the tracking process, the user can also visually enhance interesting structures in the time-frequency representation by applying a logarithmic compression. Of course, it is possible to save the current state of the frequency estimation and correction process at any time and to resume the interactive process at a later stage.

Every file in the given recordings was annotated using the above described GUI by a user with some musical background. The user annotated the lead voice in the first, the middle voice in the second and the bass voice in the third recording iteration, leading to F0 trajectories with three parts and non-annotated regions in between. The reference trajectory for our running example including the user-defined constraint regions is shown in Figure 4.2.

In this way, together with the segment annotations described in Section 3.1, we can rely on an extensive dataset similarly structured as other well-known MIR datasets such as MedleyDB [3] or the Orchset [4]. Note that in our dataset, the segment annotations were independently generated

from the F0 annotations. Since the segments only indicate the occurrences of the lead voice, one may also find F0 annotations outside of segment borders. Inside the segments, middle and bass voice start with an offset relative to the lead voice (see Figure 4.2). This is due to the specific recording process, since it took Artem Erkomaishvili some time to "tune" into the playback.

The whole dataset, including the reference F0 annotations, is publicly available at [29].

4.3 Evaluation Measures

In our work, we use the standard evaluation measures for melody extraction algorithms, which were first introduced in [33]. However, our notation follows [37].

Note that from a mathematical point of view, an F0 trajectory is a vector with one frequency value per time frame. Following this, we denote the estimated F0 trajectory vector as $\mathrm{f} \in \mathbb{R}^+$ and the reference trajectory as $\mathrm{f}^* \in \mathbb{R}^+$. Furthermore, we define a reference voicing indicator vector $\mathrm{v}^* \in [0, 1]$, whose τ^{th} elements $v_\tau^* = 1$, where a voice is present ("voiced"), and $\overline{v}_\tau^* = 0$, where no voice present ("unvoiced"). Similarly, we retrieve an estimated voicing indicator vector $\mathrm{v} \in [0, 1]$ from our algorithms. We need these vectors for evaluating algorithms, which automatically detect voiced and unvoiced parts in the recording (*voicing detection*). Using this definition, we require these algorithms to indicate unvoiced frames by setting the estimated F0 trajectory to zero at the respective time frames. For algorithms that do not have a voicing detection, v is filled with ones.

Our first evaluation measure is the *Voicing Recall Rate* ($\mathrm{Rec}_{\mathrm{vx}}$), which is defined as the ratio of frames in v correctly labeled as melody frames to the total number of melody frames in the reference v^*:

$$\mathrm{Rec}_{\mathrm{vx}} = \frac{\sum_\tau v_\tau v_\tau^*}{\sum_\tau v_\tau^*} \,. \tag{4.1}$$

Since $\mathrm{Rec}_{\mathrm{vx}}$ does not capture frames that were incorrectly labeled as voiced in the estimation, we define a second evaluation measure called *Voicing False Alarm Rate* ($\mathrm{FA}_{\mathrm{vx}}$). It is defined by the ratio of frames mistakenly estimated as voiced frames to the number of unvoiced frames in the reference. If an algorithm outputs a value for every time frame (no voicing detection), it will have a perfect recall rate $\mathrm{Rec}_{\mathrm{vx}} = 1$, which would in turn result in a bad false alarm rate $\mathrm{FA}_{\mathrm{vx}} = 1$ if there is at least one time frame in the reference, which is marked as unvoiced.

$$\mathrm{FA}_{\mathrm{vx}} = \frac{\sum_\tau v_\tau \overline{v}_\tau^*}{\sum_\tau \overline{v}_\tau^*} \,. \tag{4.2}$$

Our third evaluation measure is the *Raw Pitch Accuracy* ($\mathrm{Acc}_{\mathrm{pitch}}$), which describes how accurate correctly detected voiced pitches are compared to the reference trajectory. More specifically,

we consider a trajectory value as correct if it deviates at most ε semitones from the reference trajectory value. To this end, the trajectories containing frequency values in Hz are mapped to a logarithmic pitch axis using a mapping function

$$\mathcal{M}(f) = 12 \log_2 \left(\frac{f}{f_{\text{ref}}} \right) , \tag{4.3}$$

where $f_{\text{ref}} = 55\,\text{Hz}$ ($\widehat{=}$ note A1) in this work. The raw pitch accuracy is then defined by

$$\text{Acc}_{\text{pitch}} = \frac{\sum_\tau v_\tau^* \mathcal{T}[\mathcal{M}(f_\tau) - \mathcal{M}(f_\tau^*)]}{\sum_\tau v_\tau^*} , \tag{4.4}$$

where $\mathcal{T}$ is a threshold function defined by:

$$\mathcal{T}[a] = \begin{cases} 1 & \text{if } |a| < \varepsilon \\ 0 & \text{if } |a| \geq \varepsilon \end{cases} . \tag{4.5}$$

Since octave errors are common in F0 estimation algorithms, we introduce a fourth measure called *Raw Chroma Accuracy* ($\text{Acc}_{\text{chroma}}$), which maps the pitch difference onto a single octave before it decides on whether the pitch is accurate or not:

$$\text{Acc}_{\text{chroma}} = \frac{\sum_\tau v_\tau^* \mathcal{T}[\langle \mathcal{M}(f_\tau) - \mathcal{M}(f_\tau^*) \rangle_{12}]}{\sum_\tau v_\tau^*} , \tag{4.6}$$

with the mapping function

$$\langle a \rangle_{12} = a - 12 \lfloor \frac{a}{12} + 0.5 \rfloor . \tag{4.7}$$

Finally, we define a fifth measure *Overall Accuracy* (Acc_{ov}), which combines the performances of the voicing and pitch evaluation tasks. In order to have a good overall accuracy (Acc_{ov} close to 1), an algorithm must correctly detect unvoiced frames and must have a high pitch accuracy for voiced frames:

$$\text{Acc}_{\text{ov}} = \frac{1}{L} \sum_\tau v_\tau^* \mathcal{T}[\mathcal{M}(f_\tau) - \mathcal{M}(f_\tau^*)] + \overline{v}_\tau^* \overline{v}_\tau , \tag{4.8}$$

with L being the total number of frames (voiced + unvoiced frames). Note that in this measure, algorithms without voicing detection may perform worse than algorithms with voicing detection although having the same pitch accuracy.

4.4 Time Domain Algorithms

This section gives a brief overview about the two most popular time domain F0 estimation algorithms YIN (Section 4.4.1) and its further developed variant pYIN (Section 4.4.2). Furthermore, we show their properties by evaluating them on our running example.

4.4.1 YIN

YIN (name stems from oriental philosophy "Yin" and "Yang") is a time domain F0 estimation algorithm based on a modified autocorrelation function (ACF). Given a waveform of a periodic signal, the ACF exhibits peaks at multiples of the period. Instead of simply picking the highest non-zero-lag peak in the ACF, which is prone to errors, the algorithm uses several refinement strategies to increase the estimation accuracy. Among these refinements is a difference function that increases robustness to changes in signal amplitude, a thresholding mechanism to reduce octave jumps and an interpolation algorithm to increase output resolution. The original algorithm does not have a voicing detection, thus computing a single F0 estimate for every time frame regardless of whether a voice is present or not. Further information about YIN can be found in [7].

The F0 trajectory obtained by applying YIN to our running example using an ACF computed within windows of length 1024 samples and a hopsize of 256 samples is visualized in Figure 4.3. In our work, we use a Matlab re-implementation by Grosche et al. used in [28].

The evaluation results for our running example are given in Table 4.1. Note that we evaluate the trajectories only within the three segments of a recording (red squares), since we are only interested in the trajectories of the three voices. As expected, $\mathrm{Rec}_{\mathrm{vx}} = \mathrm{FA}_{\mathrm{vx}} = 1$ in all segments, since YIN does not detect unvoiced frames. In the first segment, YIN performs well in terms of pitch accuracy. Disregarding octave errors, the accuracy slightly increases to $\mathrm{Acc}_{\mathrm{chroma}} = 0.90$. In the second segment, the pitch accuracy drops significantly, mostly due to octave errors ($\mathrm{Acc}_{\mathrm{pitch}} < \mathrm{Acc}_{\mathrm{chroma}}$). In the third segment, the pitch accuracy drops even more, since only the last unison part is detected correctly. A low pitch accuracy together with a low chroma accuracy indicates that the algorithm is distracted by the other two voices being present. In terms of overall accuracy, YIN performs well in the first and equally bad in the second and third segment. Note that YIN is disadvantaged in this measure due to a missing voicing detection.

4.4.2 pYIN

pYIN (*probabilistic* YIN) is a modification of the previously introduced YIN algorithm. Rather than computing a single F0 estimate for each time frame, pYIN first computes multiple F0 candidates. Furthermore, for every candidate, it computes a probability that indicates how likely the estimate is to be the "real" fundamental frequency. In a second step, the algorithm uses a hidden Markov model to find a "smooth" path through the F0 candidates. It also uses a voicing detection algorithm, which marks unvoiced frames by setting them to zero. For further information about pYIN, we refer to [23].

pYIN is used in the publicly available annotation software "Tony" that allows the user to

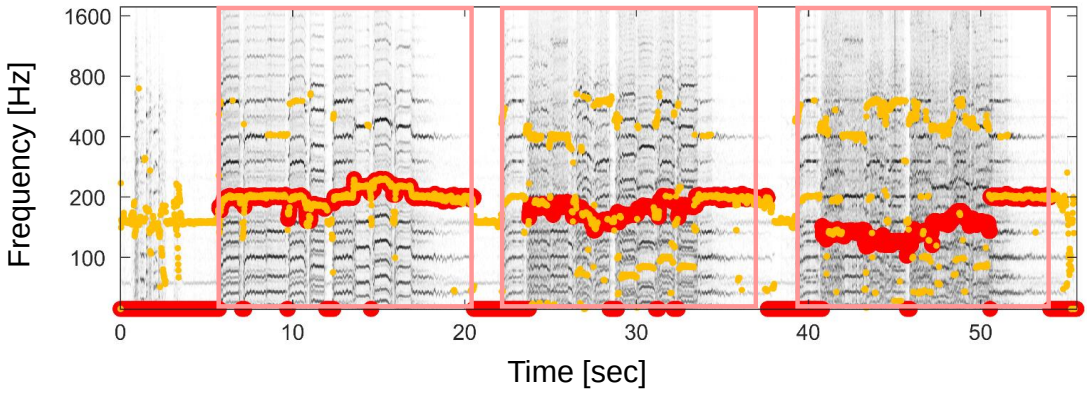

Figure 4.3: F0 trajectory computed with YIN for our running example "Da Sulisatsa" (yellow trajectory). The reference trajectory is visualized in bold red. Zero-frequency values denote unvoiced parts.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	1.00	1.00	0.87	0.90	0.83
Segment 2	1.00	1.00	0.31	0.69	0.26
Segment 3	1.00	1.00	0.30	0.39	0.26

Table 4.1: Evaluation results for YIN for our running example.

graphically correct pre-computed F0 trajectories [21]. The F0 trajectory obtained by applying pYIN on our running example using an ACF computed within windows of length 1024 samples and a hopsize of 256 samples is visualized in Figure 4.4. For the trajectory computation, we used the publicly available batch tool "Sonic Annotator" [5] and the pYIN VAMP plugin by Mauch et al. available at [22].

The evaluation results for our running example are given in Table 4.2. In the first segment, we observe a high recall rate together with a reasonable false alarm rate. Furthermore, pYIN achieves a very high pitch accuracy with (almost) no octave errors ($\text{Acc}_{\text{pitch}} = \text{Acc}_{\text{chroma}}$). In the second segment, the recall rate remains on a high level whereas the false alarm rate drastically increases. The pitch accuracy drops as well, mostly due to jumps to lower octaves. In the third segment, recall and false alarm rate equalize on a high level. Interestingly, the pitch accuracy remains on the same level as in segment two. This time, the amount of octave errors is low. Consequently, the overall accuracy drops throughout the three segments from a high value of 0.88 to a low value of 0.37.

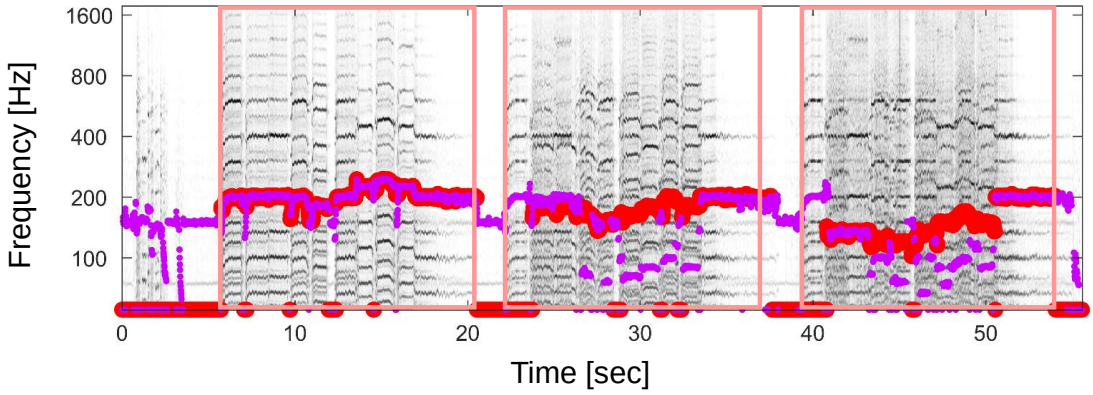

Figure 4.4: F0 trajectory computed with pYIN for our running example "Da Sulisatsa" (purple trajectory). The reference trajectory is visualized in bold red. Zero-frequency values denote unvoiced parts.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	0.98	0.57	0.92	0.92	0.88
Segment 2	0.95	0.82	0.45	0.75	0.39
Segment 3	0.92	0.92	0.45	0.48	0.37

Table 4.2: Evaluation results for pYIN for our running example.

4.5 Salience-Based Algorithms

In this section, we first describe the basic procedure of salience-based F0 estimation algorithms (Section 4.5.1). Subsequently, we explain two algorithms based on this procedure, Melodia (Section 4.5.2) and a re-implementation of Melodia (Section 4.5.3).

4.5.1 Baseline Procedure

The description in this section closely follows [30].

When extracting dominant fundamental frequency information from a complex, possibly polyphonic music recording, most salience-based approaches typically proceed in two steps. In the first step, the audio recording is converted into some kind of time–frequency representation. Then, in the second step, the dominant frequency values are selected for each time position, where one typically introduces temporal continuity conditions and exploits additional knowledge on the expected frequency range. Following this basic approach, we now summarize such a procedure closely following the work by Salamon et al. [36]. The procedure is illustrated for our

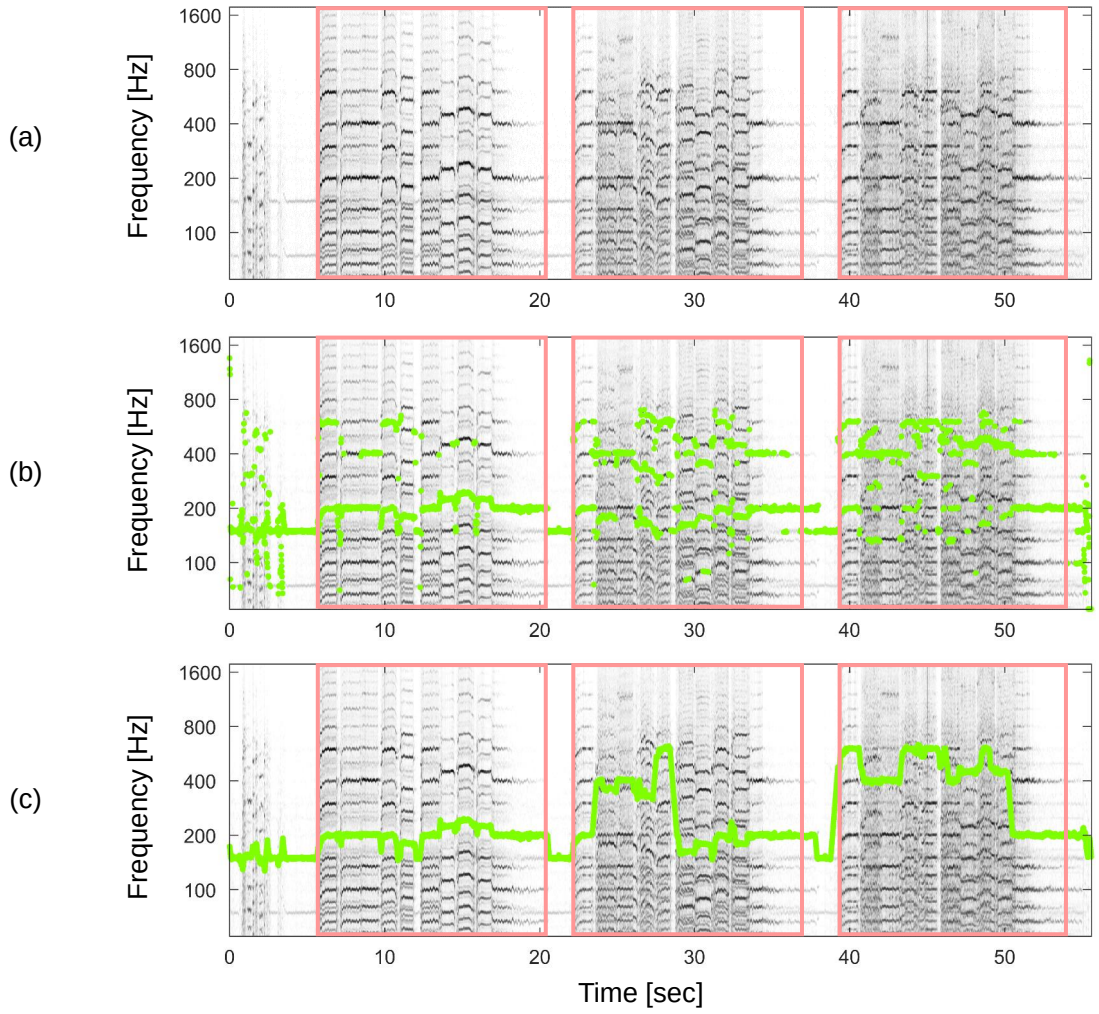

Figure 4.5: Illustration of the F0 trajectory computation for the three-stage recording of Figure 2.1. **(a)** Salience representation (enhanced log-frequency spectrogram). **(b)** Frame-wise F0 trajectory (green line). **(c)** F0 trajectory with continuity constraints.

running example in Figure 4.5.

In the first step, the waveform is converted into a time-frequency representation by applying a suitable STFT as described in Section 2.2.1. Accounting for the fact that the human perception of pitch is logarithmic in frequency, we convert the STFT coefficients to a log-frequency spectrogram according to Section 2.2.3, which also includes a refinement of frequency resolution by instantaneous frequency estimation. When used for extracting fundamental frequency information, the log-frequency spectrogram is typically enhanced to better account for acoustic characteristics that are of perceptual and musical relevance. First, motivated by the observation that spectral components can show extremely small values while still being relevant for a human listener, the spectrogram is logarithmically compressed to balance out the difference between large and small

values.

The second enhancement strategy is based on the observation that a sound event such as a musical tone is associated to a fundamental frequency along with its harmonic partials, which are (approximately) the integer multiples of the fundamental frequency. The multiple appearances of tonal time–frequency patterns can be exploited to enhance a spectrogram representation by jointly considering a frequency and its harmonics forming suitably weighted sums—a technique also called *harmonic summation*, see [14, 19, 36]. The resulting time-frequency representation is often referred to as *salience spectrogram*, since it makes the time-frequency coefficients that are likely to be part of a melody's F0 trajectory more salient (see Figure 4.5a). For further details, we refer to [26, Chapter 8] and [36].

In the second step, the goal is to determine relevant frequency information. Based on the assumption that the melody often correlates to the predominant F0 trajectory, a first strategy is to simply consider the frame-wise maximum of the computed salience representation (see Figure 4.5b). Such a frame-wise approach may lead to a number of temporal discontinuities and random jumps that occur due to confusions between the fundamental frequency and higher harmonics or lower ghost components introduced by the harmonic summation. To balance out the two conflicting conditions of temporal flexibility (to account for possible jumps between notes) and temporal continuity (to account for smoothness properties), one can use a procedure for constructing a frequency trajectory based on *dynamic programming* [34, 25] (see Figure 4.5c). Even though this may be a desirable property most of the time, discontinuities that are the result of abrupt note changes tend to be smoothed out (we elaborate further on this trade-off in the subsequent sections). Furthermore, tracking errors still occur when there are several melodic lines or when there is no melody at all.

4.5.2 Melodia

Melodia is a F0 estimation algorithm developped by Salamon et al. based on the general procedure described above [36]. In this work, we computed F0 trajectories using the Melodia VAMP plugin available at [35]. Based on a STFT with window length $N = 1024$ samples and hopsize $H = 256$ samples, the Melodia trajectory of our running example is visualized in Figure 4.6. As a default setting, the algorithm's temporal continuity constraints restrict the distance in frequency between two consecutive trajectory values (also referred to as *pitch offset*) to 80 cents. Consequently, a large maximum pitch offset allows for large jumps in the trajectory, whereas a small maximum pitch offset creates a rather smooth trajectory. Note that Melodia has a built-in voicing detection, which indicates unvoiced frames by negative values in the trajectory. For better comparison, we set all negative trajectory values to zero.

The evaluation results for Melodia on our running example are given in Table 4.3. In the first

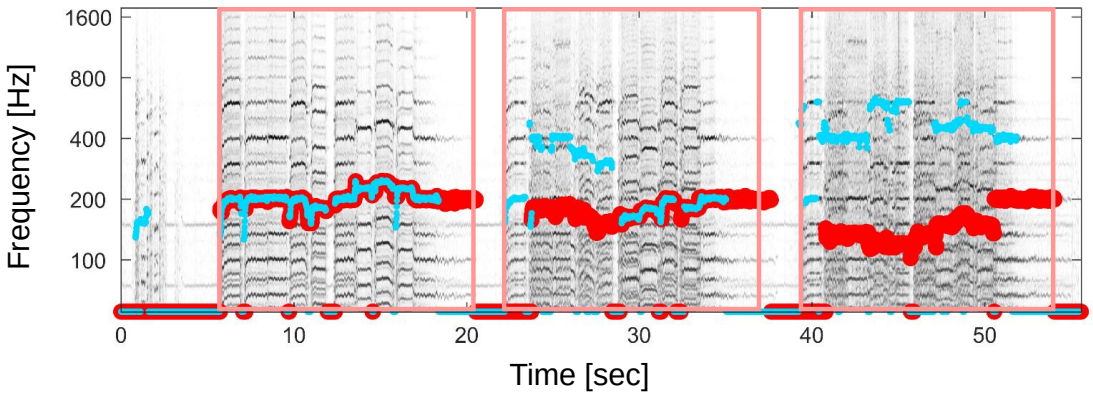

Figure 4.6: F0 trajectory computed with Melodia for our running example "Da Sulisatsa" (light blue trajectory). The reference trajectory is visualized in bold red. Zero-frequency values denote unvoiced parts.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	0.80	0.16	0.94	0.94	0.75
Segment 2	0.70	0.55	0.57	0.79	0.40
Segment 3	0.67	0.85	0.00	0.12	0.02

Table 4.3: Evaluation results for Melodia for our running example.

segment, we observe a high recall value and a very low false alarm rate. For the monopitch estimation task, Melodia achieves a very good pitch accuracy with (almost) no octave errors. In the second segment, the recall rate drops while the false alarm rate increases. In this segment, the trajectory exhibits many jumps to higher octaves, leading to a reasonable pitch accuracy and a good chroma accuracy. The false alarm rate increases again in the third segment. By looking at the third segment in Figure 4.6, one can observe that the Melodia trajectory is far off the reference trajectory. This also reflects in a pitch accuracy of $\text{Acc}_{\text{pitch}} = 0.00$. Remarkably, there are also few octave errors, indicating that Melodia is strongly distracted by spectral contributions of the other two voices. As a consequence, the overall accuracy also drops in the third segment.

4.5.3 Melodia Re-Implementation

As a second algorithm, we examine a re-implementation of Melodia in Matlab by Driedger [9]. In the following, we will denote the algorithm as *DGM*, derived from the surnames of the main contributors Driedger, Grohganz and Müller. Contrary to Melodia, the re-implementation does not have a voicing detection. In this work, we consider two settings for the temporal continuity constraints of DGM. First, we restrict the maximum allowed pitch offset to 10 cents

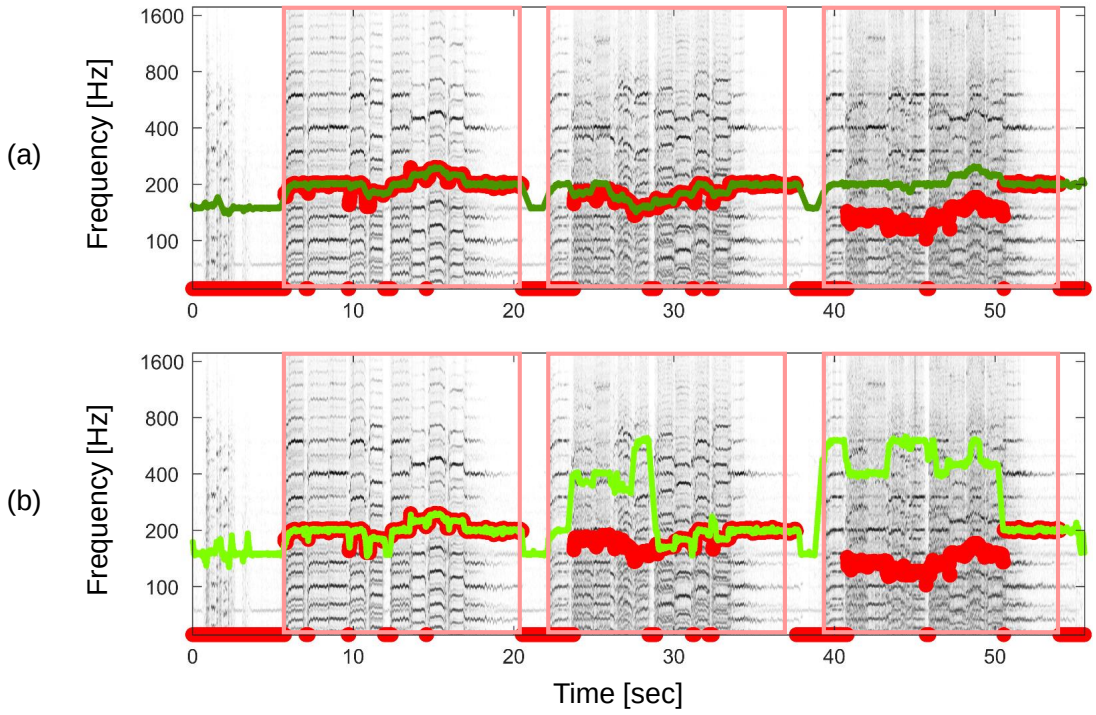

Table 4.4: F0 trajectories computed with DGM for our running example "Da Sulisatsa". The reference trajectory is visualized in bold red. Zero-frequency values denote unvoiced parts. **(a)** Maximum pitch offset: 10 cents (dark green). **(b)** Maximum pitch offset: 50 cents (light green).

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	1.00	1.00	0.94	0.94	0.89
Segment 2	1.00	1.00	0.85	0.85	0.71
Segment 3	1.00	1.00	0.27	0.27	0.23

Table 4.5: Evaluation results for DGM (maximum pitch offset: 10 cents) for our running example.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	1.00	1.00	0.98	0.98	0.93
Segment 2	1.00	1.00	0.57	0.80	0.47
Segment 3	1.00	1.00	0.27	0.29	0.23

Table 4.6: Evaluation results for DGM (maximum pitch offset: 50 cents) for our running example.

(see Figure 4.4a), then we loosen the constraints by setting the allowed pitch offset to 50 cents (see Figure 4.4b). Note that every pitch jump within these intervals has equal weight/probability.

Both figures were generated based on a STFT with window length $N = 1024$ samples and hopsize $H = 256$ samples.

The evaluation results for our running example are given in Tables 4.5 and 4.6. As expected, $\text{Rec}_{\text{vx}} = \text{FA}_{\text{vx}} = 1$ due to a missing voicing detection. In the first segment, both parameter settings deliver very good pitch accuracies and do not suffer from octave jumps. The 10 cents-restricted trajectory follows the reference trajectory rather smoothly in comparison to the 50 cents-restricted trajectory. In the second segment, the differences between the two settings is even more pronounced. While the 10 cents-restricted trajectory smoothly stays on the reference, the less constrained version is able to leave the reference and erroneously jumps to higher octaves. This results in a lower pitch accuracy and a high chroma accuracy. In the third segment, both trajectories are mostly off the reference, only being correct in the final unison part. However, although both settings achieve the same low pitch accuracy, the chroma accuracy for the 50 cents-restricted setting is slightly higher. In terms of overall accuracy, both settings produce similar results in segments one and three and largely differ in the second segment.

4.6 Three-Stage F0 Trajectory Estimation

In this section, based on the DGM implementation introduced above, we propose a refined salience-based F0 estimation algorithm that exploits the three-stage recording process of the given Georgian chant recordings outlined in Figure 2.1. Recall that our goal is to extract the F0 trajectory for the lead voice in the first segment, the F0 trajectory for the middle voice in the second segment, and the F0 trajectory for the bass voice in the third segment. Furthermore, the frequency values for time frames between the sections should remain unspecified.

In the first segment, where only the lead voice is present, we simply use the continuity-constrained F0 tracking procedure as described in Section 4.5.1 to determine the dominant F0 trajectory, see Figure 4.7a. The maximally allowed pitch offset is set to 50 cents. Since we have a monophonic scenario in the first segment, this is expected to produce accurate estimates even though, occasionally one may obtain some octave errors. However, such errors may be reduced by limiting the search range, which is done in the following stages.

In the second segment, the task is to extract the F0 trajectory for the middle voice. In this segment, however, the task is much harder, since the middle voice is superimposed with the lead voice. To reduce octave errors and the confusion between middle and lead voice, we set the previously extracted F0 trajectory from the first segment as an upper limit of the search range in the second segment. In other words, we exclude the time-frequency region that lies above the F0 trajectory of the lead voice, as depicted in Figure 4.7b. Furthermore, we impose a "safety margin" of 50 cents (half a semitone) between the lead and middle voice in order to prevent the

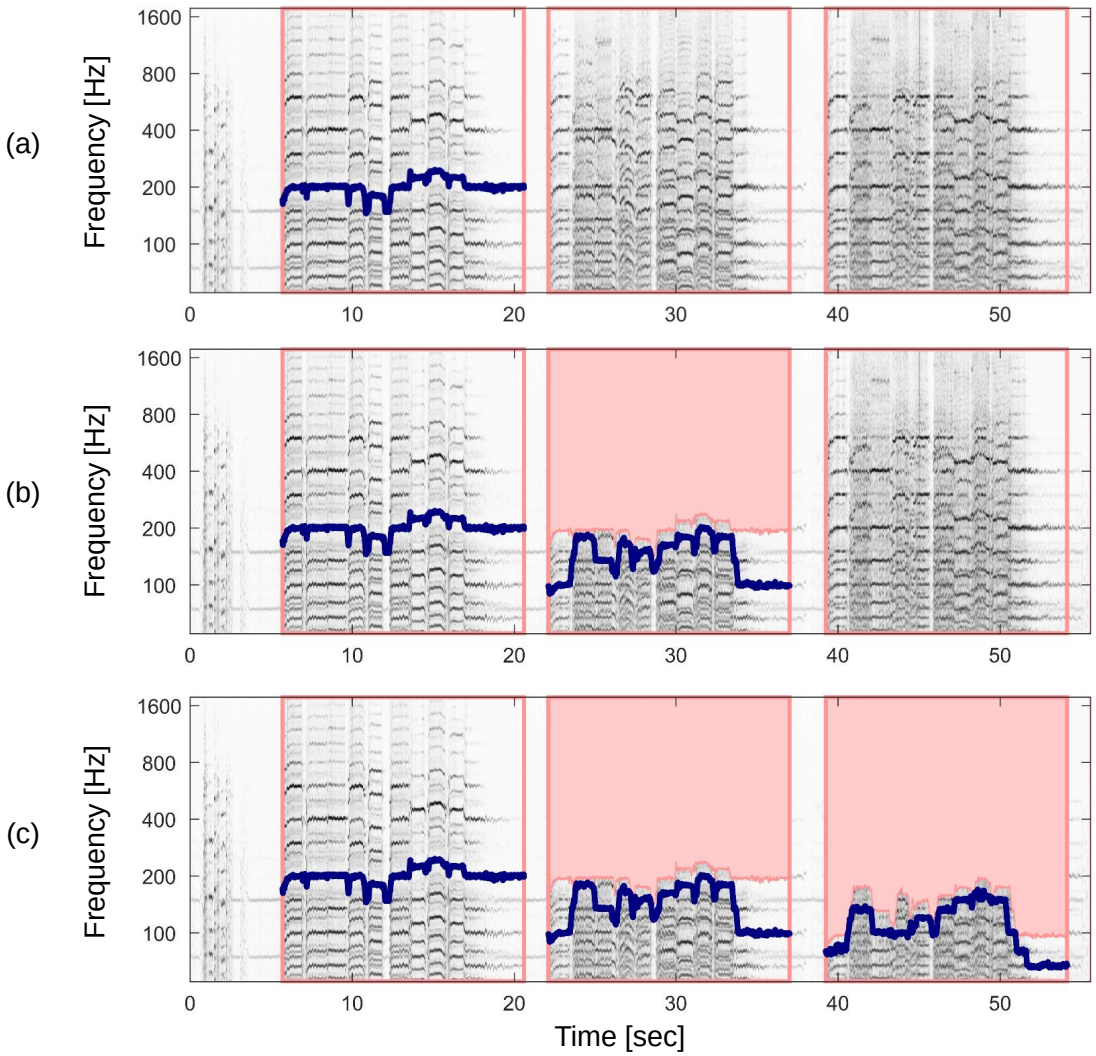

Figure 4.7: Computation of F0 trajectory exploiting the three-step recording process. **(a)** F0 trajectory of the first section (lead voice). **(b)** F0 trajectory of the second section (middle voice) excluding the region above the lead voice trajectory. **(c)** F0 trajectory of the third section (bass voice) excluding the region above the middle voice trajectory.

middle voice trajectory from taking values close to the lead voice trajectory. Subsequently, we apply the continuity-constrained F0 tracking procedure in the remaining region to obtain a F0 trajectory, which largely corresponds to the middle voice.

In the third segment, we proceed in a similar way as in the second step. This time, however, we exclude the time-frequency region that lies above the F0 trajectory of the middle voice and again impose a "safety margin" of 50 cents. The resulting F0 trajectory for our running example is shown in Figure 4.7c.

Note that our approach assumes the middle voice to be sung strictly below the lead voice.

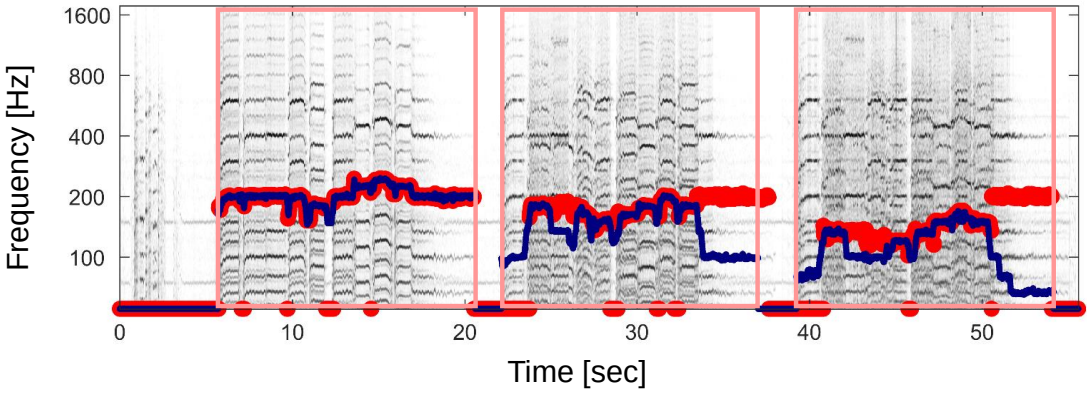

Figure 4.8: F0 trajectory computed with our constrained F0 estimation algorithm for our running example "Da Sulisatsa" (dark blue trajectory). The reference trajectory is visualized in bold red. Zero-frequency values denote unvoiced parts.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	1.00	1.00	0.98	0.98	0.93
Segment 2	1.00	1.00	0.54	0.79	0.45
Segment 3	1.00	1.00	0.44	0.47	0.39

Table 4.7: Evaluation results for the three-stage approach for our running example.

Accordingly, we assume the bass voice to be sung strictly below the middle voice. Although we could observe this behavior in many recordings, this cannot be generalized to Georgian vocal music. Furthermore, our approach disregards the fact that in many Georgian chants, all three voices end on the same pitch ("perfect unison"), thus leading to an error in the estimated frequency value (see end of second and third segment in Figure 4.8).

The evaluation results for our running example are given in Table 4.7. As expected, $\text{Rec}_{\text{vx}} = \text{FA}_{\text{vx}} = 1$ due to a missing voicing detection. In the first segment, the three-stage approach achieves almost perfect pitch accuracy with no octave errors. Note that in the first segment, the evaluation values are equal to the results of the DGM algorithm, since both algorithms are constrained equally in this segment. In the second segment, the estimated trajectory mostly stays on the reference except at the end, where it is forced to jump to a lower octave due to the restrictions mentioned above. This in turn leads to a good chroma accuracy. In the third segment, we observe the same problem with the unison part. Further, we see that errors that are made in early stages (such as around 26 s) propagate through subsequent segments, leading to further estimation errors. Despite these problems, the approach still delivers reasonable results in the third segment.

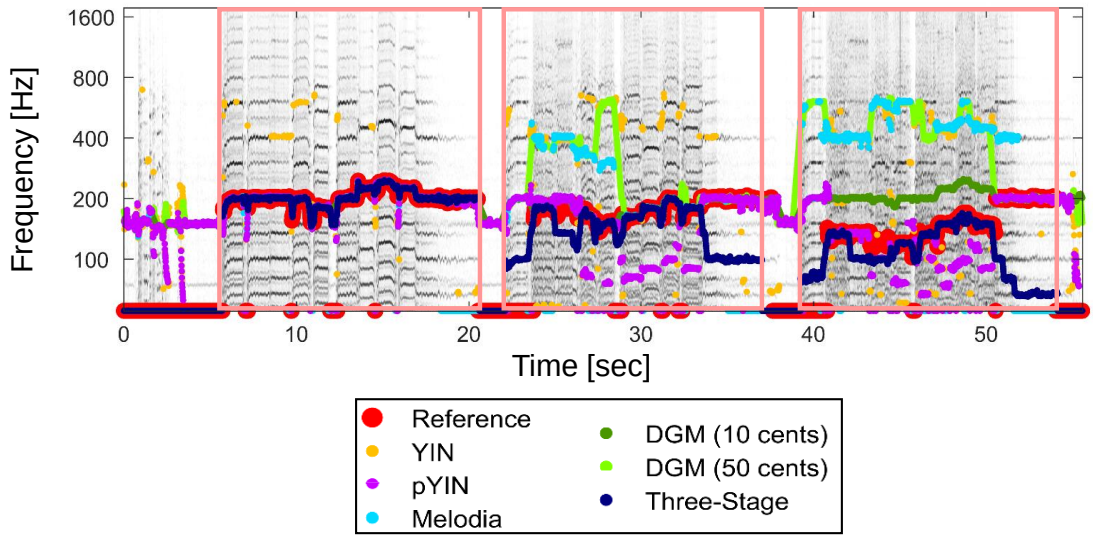

Figure 4.9: F0 trajectories for our running example plotted in one figure.

4.7 Evaluation

In this section, we will evaluate the above introduced F0 estimation algorithms on all given Georgian chant recordings. More specifically, we compare YIN, pYIN, the two parameter settings of DGM, Melodia, and our three-stage algorithm using the evaluation measures defined in Section 4.3. Again, we evaluate the trajectories only within the three segments of a recording. As a reference, Figure 4.9 shows a superposition of the trajectories of all six algorithms together with the reference annotations for our running example "Da Sulisatsa".

The segment-wise evaluation results averaged over all recordings and rounded to two decimals are shown in Tables 4.8-4.13. The corresponding standard deviations are given in brackets. Note that the averaging was conducted with equal weight for each recording, regardless of the recording length. Again, the chant "Adide sulo chemo" (GCH_048_Erkomaishvili.wav) was excluded from our evaluation. The evaluation is based on trajectories computed with a window length of $N = 1024$ samples and a hopsize of $H = 256$ samples. Since changing the default parameter settings of the Melodia VAMP plugin led to unresolvable crashes, we computed the Melodia trajectories with the standard hopsize of $H = 64$ samples and downsampled the trajectories afterwards by a factor of four. For the evaluation of pitch accuracy, we set $\varepsilon = 0.5$ semitone (50 cents).

In the monophonic scenario of the first segment, as expected, all algorithms performed well. The values for the pitch accuracy lie in the range of $0.78 - 0.92$. Interestingly, the 10 cents-constrained DGM algorithm performs worst in this regard, which is assumed to be due to the strong constraint, which does not allow large jumps in the trajectory. The 50 cents-constrained DGM algorithm and

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	1.00 (0.00)	1.00 (0.00)	0.86 (0.08)	0.93 (0.02)	0.77 (0.07)
Segment 2	1.00 (0.00)	1.00 (0.00)	0.61 (0.13)	0.78 (0.11)	0.53 (0.11)
Segment 3	1.00 (0.00)	1.00 (0.00)	0.47 (0.16)	0.64 (0.15)	0.40 (0.14)

Table 4.8: Evaluation results for YIN averaged over all recordings.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	0.97 (0.01)	0.70 (0.14)	0.84 (0.12)	0.92 (0.02)	0.77 (0.11)
Segment 2	0.93 (0.02)	0.76 (0.08)	0.69 (0.14)	0.82 (0.10)	0.59 (0.12)
Segment 3	0.90 (0.03)	0.76 (0.07)	0.64 (0.15)	0.69 (0.13)	0.53 (0.13)

Table 4.9: Evaluation results for pYIN averaged over all recordings.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	0.87 (0.10)	0.13 (0.09)	0.85 (0.07)	0.88 (0.03)	0.75 (0.10)
Segment 2	0.82 (0.08)	0.21 (0.11)	0.77 (0.17)	0.82 (0.11)	0.66 (0.15)
Segment 3	0.63 (0.12)	0.18 (0.15)	0.45 (0.28)	0.67 (0.21)	0.36 (0.19)

Table 4.10: Evaluation results for Melodia averaged over all recordings.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	1.00 (0.00)	1.00 (0.00)	0.78 (0.05)	0.79 (0.05)	0.70 (0.06)
Segment 2	1.00 (0.00)	1.00 (0.00)	0.75 (0.16)	0.78 (0.11)	0.65 (0.14)
Segment 3	1.00 (0.00)	1.00 (0.00)	0.44 (0.32)	0.62 (0.22)	0.38 (0.28)

Table 4.11: Evaluation results for DGM (pitch offset: 10 cents) averaged over all recordings.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	1.00 (0.00)	1.00 (0.00)	0.92 (0.08)	0.95 (0.01)	0.83 (0.07)
Segment 2	1.00 (0.00)	1.00 (0.00)	0.82 (0.18)	0.87 (0.12)	0.70 (0.15)
Segment 3	1.00 (0.00)	1.00 (0.00)	0.42 (0.28)	0.66 (0.22)	0.36 (0.24)

Table 4.12: Evaluation results for DGM (pitch offset: 50 cents) averaged over all recordings.

Description	Rec_{vx}	FA_{vx}	$\text{Acc}_{\text{pitch}}$	$\text{Acc}_{\text{chroma}}$	Acc_{ov}
Segment 1	1.00 (0.00)	1.00 (0.00)	0.92 (0.08)	0.95 (0.01)	0.83 (0.07)
Segment 2	1.00 (0.00)	1.00 (0.00)	0.61 (0.17)	0.84 (0.10)	0.53 (0.15)
Segment 3	1.00 (0.00)	1.00 (0.00)	0.54 (0.15)	0.64 (0.14)	0.46 (0.12)

Table 4.13: Evaluation results for three-stage approach averaged over all recordings.

the three-stage approach perform best in this regard with an average score of 0.92. Furthermore, we observe few octave errors among all algorithms in this segment.

In the second segment, where the scenario becomes polyphonic, the algorithms achieve pitch accuracies in the range of $0.61 - 0.82$ with YIN and the three-stage approach performing worst, and the 50 cents-constrained DGM algorithm performing best. Compared to the first segment, there are noticeably more octave errors except for the 10 cents-constrained DGM algorithm.

In the third segment, where the task is most difficult, all algorithms deliver a reasonable performance. The pitch accuracies of most algorithms lie in the range of 0.42 to 0.47. However, pYIN is an exception with a score of 0.64. One reason for this might be that the frequency resolution of pYIN is not restricted for lower frequencies as for salience-based algorithms. Remarkably, the 10 cents-constrained DGM version achieves a slightly better score in this segment than the 50 cents-constrained DGM version. Except for pYIN, we also observe a high number of octave errors for all algorithms. Compared to the other segments, the standard deviations for the evaluation measures in the third segment are noticeably higher, especially for both DGM variants and Melodia. This indicates strong differences in the performance of the algorithms in the third segment.

As expected, $\text{Rec}_{\text{vx}} = \text{FA}_{\text{vx}} = 1$ for the algorithms without voicing detection. Comparing the voicing detection of pYIN and Melodia, Melodia exhibits a much lower false alarm rate than pYIN at the cost of a lower voicing recall rate. This means that pYIN tends to mark a frame as voiced in most cases, whereas Melodia is more careful in detecting voiced frames. In terms of overall accuracy, pYIN performs better than Melodia in the first and third segment while performing worse in the second segment. Note that algorithms without voicing detection that achieved a high pitch accuracy perform better in this regard.

In a second evaluation procedure, we have a more detailed look at the raw pitch accuracy measure. More specifically, instead of fixing a maximum deviation ε as in our first evaluation, we now run the evaluation with ascending values of ε in the range of 0 cents to 200 cents with a stepsize of 0.1 cents. The resulting curves for each algorithm and segment are shown in Figure 4.10.

Regarding the overall pitch accuracy, we again observe for all algorithms the highest accuracy in

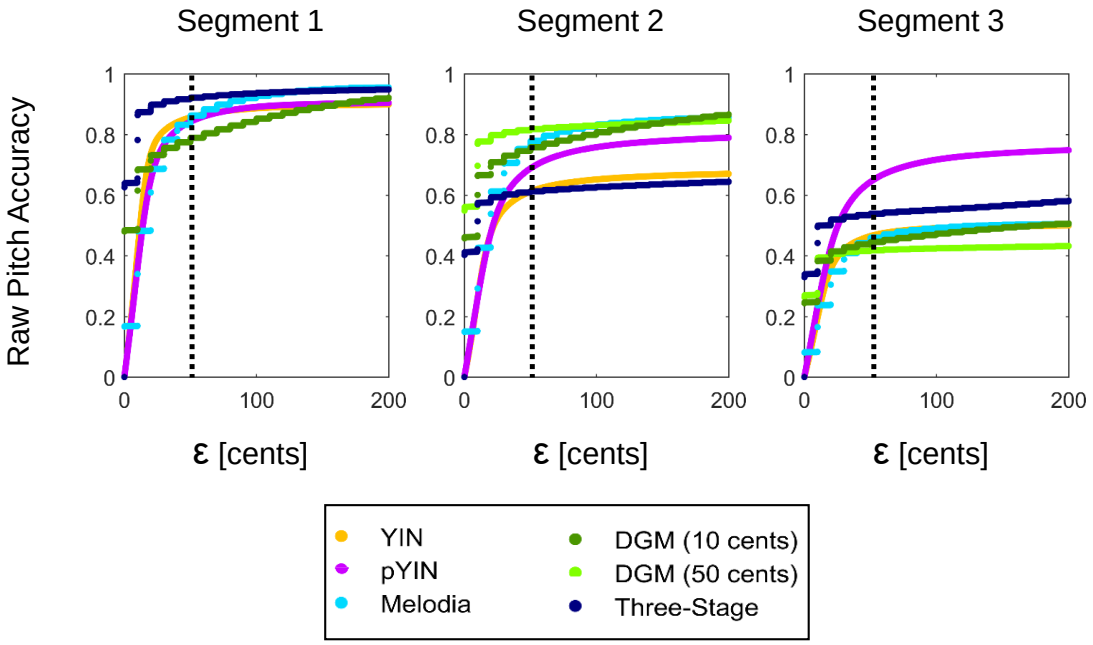

Figure 4.10: Raw pitch accuracy with running threshold ε for all five algorithms averaged over whole dataset. The threshold for our averaged evaluation results ($\varepsilon = 50$ cents) is marked by a dashed black line.

the first segment and the lowest accuracy in the third segment. For low values of $\varepsilon < 50$ cents the raw pitch accuracy drops significantly. For larger values of $\varepsilon > 50$ cents the curves stay on an almost constant level. This indicates that most of the pitch errors lie in the range of 0 cents to 50 cents. Furthermore, there are remarkable differences between the curves' appearances. While the curves for the time domain algorithms are rather smooth, the curves for the salience-based algorithms exhibit stair-like structures, which is due to the limited frequency resolution of the salience-based algorithms.

The performance of the three-stage approach relative to the other algorithms strongly varies. In the first segment, it performs best, together with the underlying 50 cents-constrained DGM algorithm, since both algorithms are equally constrained in this segment. In the second segment, the three-stage approach performs worst (together with YIN) contrary to DGM, which performs best. An improvement over DGM can be observed in segment three, where the three-stage approach performs over average in terms of pitch accuracy. In Appendix C, we visualize examples with low and high average pitch accuracy for the three-stage approach.

4.8 Conclusions and Further Notes

In this chapter, we described the task of fundamental frequency estimation and explained the approaches and properties of different F0 estimation algorithms. Additionally, we introduced a three-stage F0 estimation approach tailored to the three-part structure of the given Georgian chant recordings. In a subsequent step, we evaluated all algorithms on the collection of recordings using standard evaluation measures. The experiments showed that our three-stage approach does not bring improvements in monophonic or simple polyphonic scenarios such as in segments one and two, but can help to increase the performance in more difficult scenarios as in segment three. On average, pYIN delivered the most solid performance throughout all segments, with a surprisingly good result in segment three.

Future research should aim at further improving the three-stage algorithm with specific focus on the unison parts, which are likely to be the reason for comparatively bad evaluation results, especially in the first two segments. One possible refinement would be to relax the constraints at the end of each segment in order to allow for an unison ending. In order to eliminate the influence of unison parts on the evaluation results, one could also restrict the analysis to frames where all three voices differ at least ε cents.

Another interesting study would be to examine the influence of the chosen reference annotations on the evaluation results. In this context, instead of using the manually generated annotations with the introduced GUI, one could think of conducting the experiments with reference trajectories created with the Tony software. Furthermore, our evaluation showed that different continuity constraints for the DGM algorithm strongly influenced the performance. In future work, we will therefore conduct more detailed analyses on this effect.

Chapter 5

Applications to Georgian Vocal Music Research

In this chapter, we will have a closer look at the previously introduced reference F0 annotations. Because of the historical importance of the recordings by Artem Erkomaishvili, these trajectories may serve as a starting point for a whole set of subsequent analysis steps including the analysis of the historical tuning system, transcription-free documentation, harmonic analysis, and quantitative comparison of chants. Additionally, the computational analyses may also be an interesting reference for classic, non-computational ethnomusicological studies. As an example, in Section 5.1, we analyze the sung intervals in the given recordings. Then, we use median filters to smooth the trajectories and show the effect on the interval distributions. In Section 5.3, we again use median filters in a slightly different way to find stable parts within the trajectories. Finally, we draw conclusions and outline further research tasks in Section 5.4.

5.1 Interval Analysis

The analysis in this section follows the idea of Scherbaum [38] and description in Müller et al. [30].

Using our running example "Da Sulisatsa", the interval analysis procedure is illustrated in Figure 5.1. First, the reference F0 trajectories of the lead, middle, and bass voice (see Figure 5.1a) are superimposed (see Figure 5.1b). Note that, in this step, we only consider the F0 trajectories within the three segments given by the segment annotations. In a subsequent step, the intervals (given in cents) between the F0 trajectories of the lead and middle voice, the lead and bass voice, as well as the middle and bass voice are computed for each time position (see Figure 5.1c). Note that the occurrences are given in seconds, indicating how long a certain interval is sung in the

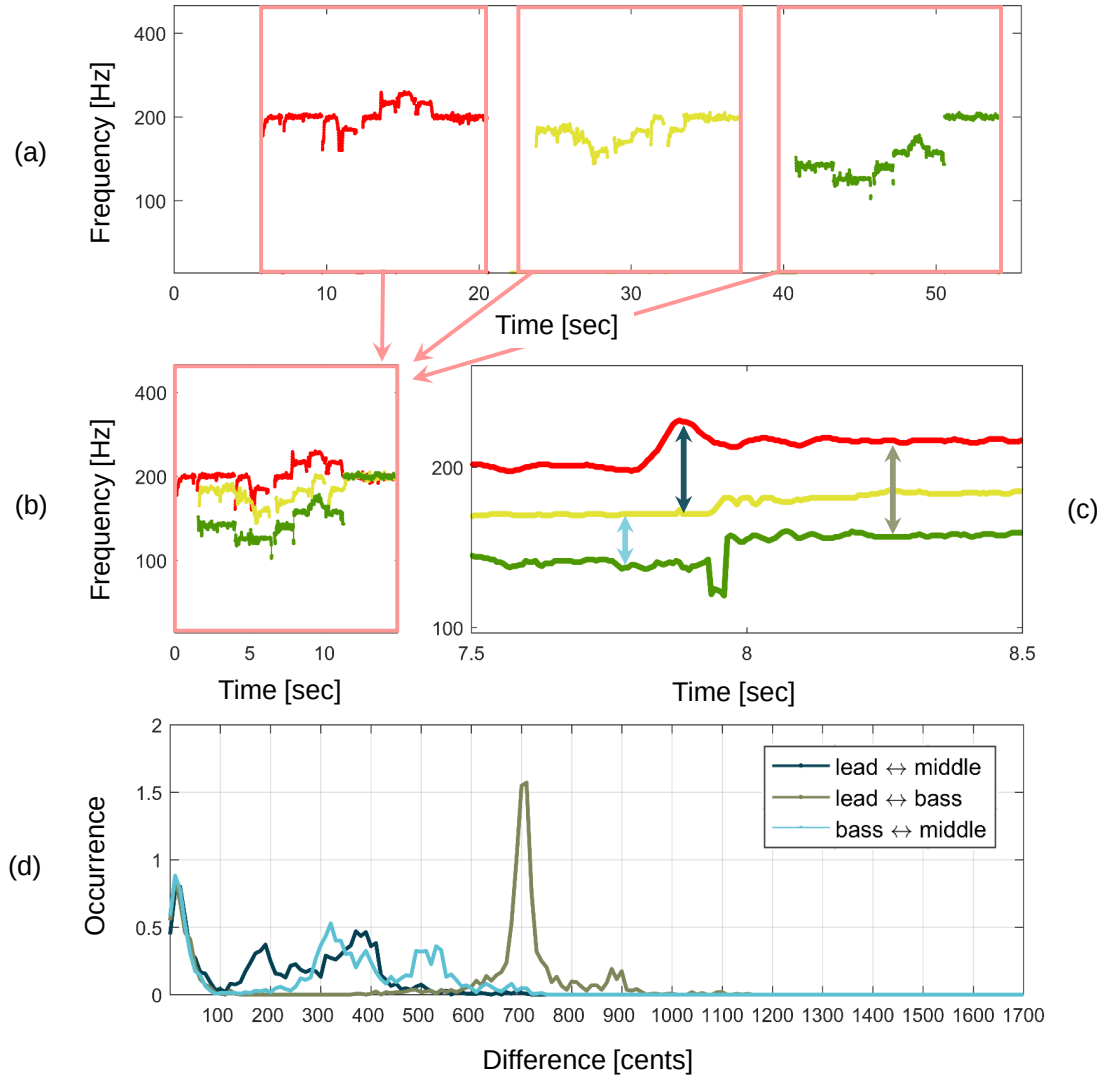

Figure 5.1: **(a)** Reference F0 trajectory for our running example. **(b)** Superposition of the F0 trajectories estimated for the three voices. **(c)** Illustration of the interval computation (using a zoomed section of (b)). **(d)** Resulting interval distribution.

recording by a certain voice pair. The resulting interval distribution is shown in Figure 5.1d.

In our experiments, we computed and averaged such distributions over the 100 recordings by Erkomaishvili (again excluding GCH_048_Erkomaishvili.wav). The three resulting average distributions along with an accumulated distribution (considering all three cases jointly) are shown in Figure 5.2. The accumulated distribution indicates how long a certain interval is sung in an average recording of the given collection. Looking at these distributions, one can make some interesting observations. Disregarding the peaks close to the unison interval (0 cents) and octave interval (1200 cents), the most prominent peak occurs close to the fifth interval (702 cents in just intonation, 700 cents in the 12-tone equal-tempered scale). This reflects the fact that the

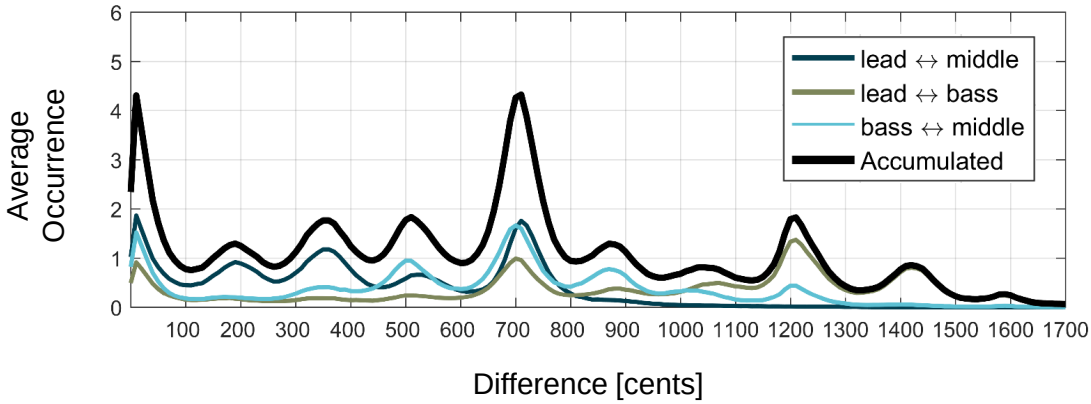

Figure 5.2: Interval distributions averaged over all recordings.

fifth interval plays an important role in Georgian chants and that this interval is sung with high intonation accuracy.

Interestingly, there is another noticeable peak located at about 350 cents. From a Western music perspective, this is an unusual interval, since it lies between the minor third (315.6 cents in just intonation, 300 cents in the 12-tone equal-tempered scale) and the major third (400 cents in the 12-tone equal-tempered scale). The peak may be the result of the non-tempered nature of traditional Georgian vocal music [38]. From a Western music perspective, the role of the third interval in Georgian music seems ambiguous and, in combination with a fifth, evokes in the listener a sound that somehow lies between a minor and major chord. For an extended musical analysis of the Erkomaishvili recordings, we refer to [41]. Our observations agree with the results of the much more detailed study by Scherbaum [38], which was conducted on the larynx-microphone field recordings mentioned in Section 2.1.3. Comparing the interval distribution for our running example with the averaged distribution, we see that "Da Sulisatsa" reflects many characteristics of the averaged distribution, except for the octave interval.

5.2 Trajectory Smoothing

In this section, we add an intermediate filtering step to the interval analysis procedure described above. More specifically, we smooth the F0 trajectories using a median filter before computing the interval distributions. With this refinement, we attempt to enhance the peaky structure of the histograms while reducing the influence of pitch variations or sudden pitch changes.

Median filters are a commonly used tool in image processing for removing salt and pepper or impulse noise [18, Chapter 4.4], while recently, they have also been used in audio signal processing for harmonic/percussive source separation [13]. A median filter replaces a given signal value by the median of all signal values in its neighborhood. Let us again consider an F0 trajectory

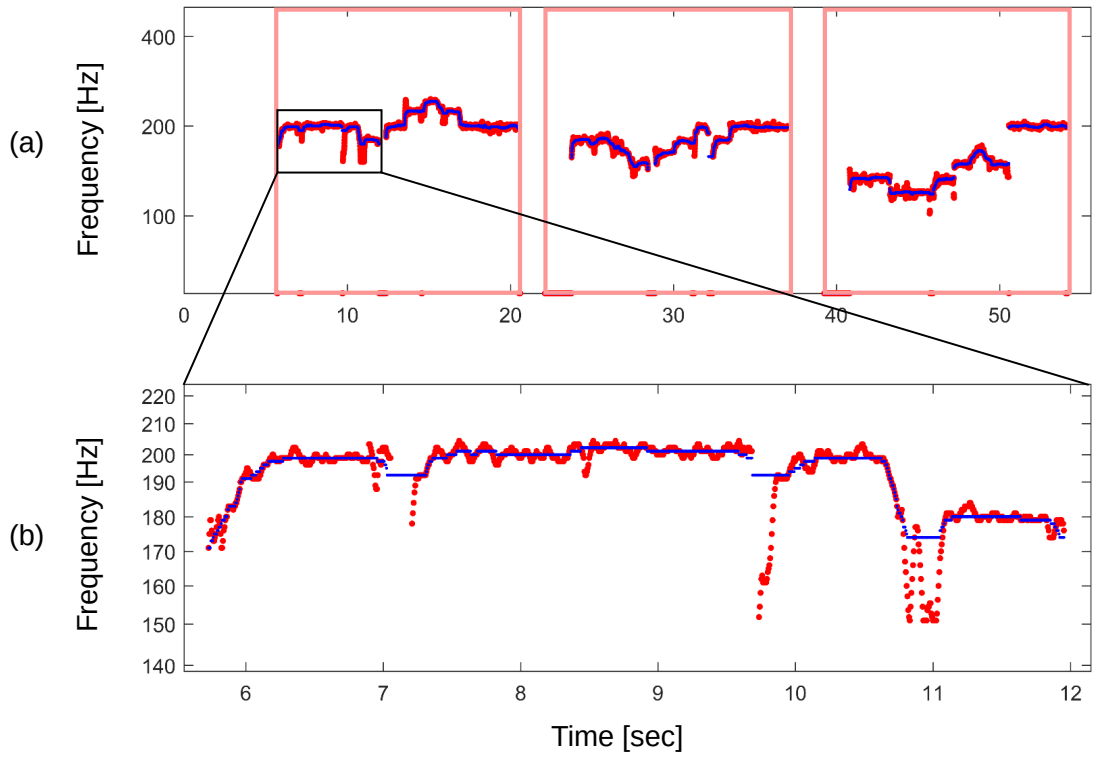

Figure 5.3: Superposition of unfiltered F0 trajectory (red) and median filtered F0 trajectory (blue) of our running example "Da Sulisatsa". **(a)** Whole audio file. **(b)** Detail.

as a vector with one frequency value per time frame. Similar to Section 4.3, let us denote our reference trajectory vector as $\mathrm{f}^*(n)$. Following [13], the median filtered trajectory vector $\mathrm{f}^*_{\mathrm{MF}}(n)$ is then obtained by

$$\mathrm{f}^*_{\mathrm{MF}}(n) = \mathrm{median}\{\mathrm{f}^*(n-k:n+k), k = (l-1)/2\}\,, \tag{5.1}$$

with the median filter length l being an odd number given in frames. In our work, we used a median filter of length 87 frames $\widehat{=}$ 0.5 s.

The unfiltered and filtered reference trajectories for our running example are visualized in Figure 5.3. It is clearly visible that tiny pitch variations and even large pitch drops (as around 11 s) in the trajectory are smoothed out. This is also reflected by more distinct peaks in the accumulated interval distribution visualized in Figure 5.4 (dashed line). Especially the peaks around 0 cents, 700 cents and 1200 cents are significantly enhanced. The peak locations are not (visibly) affected by the filtering operation.

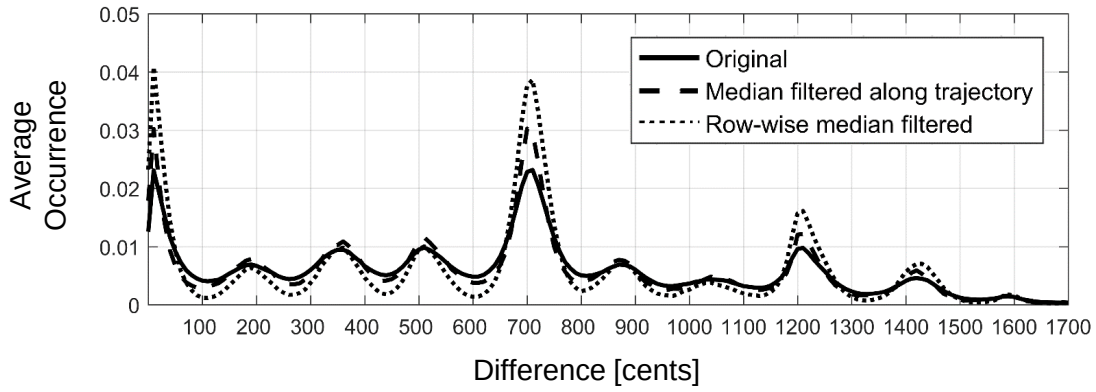

Figure 5.4: Comparison of accumulated interval distributions.

5.3 Detection of Stable Pitches

In a further experiment, we again use median filters, but apply them in a different way. This time, the task is to compute the interval distributions exclusively on stable parts of the trajectories, characterized by similar pitches over a longer period of time. This can also be seen as step towards transcription of the audio recordings. While in the previous section, we filtered along the trajectories, we now slice each trajectory horizontally and filter each of the slices, which we again consider as vectors, separately with a median filter. Looking again at the reference trajectory in Figure 5.3b, ideally, the sparse entries in a slice e.g. at 160 Hz are filtered out, while a slice around 200 Hz with many consecutive entries will keep a filtered version of its entries. In this way, only stable parts, where the trajectory stays in the same slice for a longer time, remain after filtering. The procedure is explained in further detail in the following.

In a first step, the values in the reference trajectory vector are quantized according to a logarithmic frequency axis F_{q}. Note that in this step, the quantization stepsize can be interpreted as a parameter tuning the "strictness" of the stable pitch detection. A fine quantization (strict) results in more slices and less values per slice, hence less values will be indicated as stable after filtering. A coarse quantization (soft), in turn, results in less slices and more values per slice, hence more stable pitches. In our work, we use a quantization stepsize of 50 cents.

In a second step, the goal is to represent the quantized trajectory vector as a matrix. To this end, we construct a matrix where each row corresponds to a frequency value in F_{q} and each column corresponds to a time frame of the quantized trajectory vector (see Figure 5.5). In this way, every element of our quantized trajectory vector can be assigned to one time-frequency bin in our matrix. Time-frequency bins, where the trajectory is active, are set to 1, whereas bins, where the trajectory is inactive, are set to 0.

In a third step, we filter each row of the constructed matrix with a median filter as described

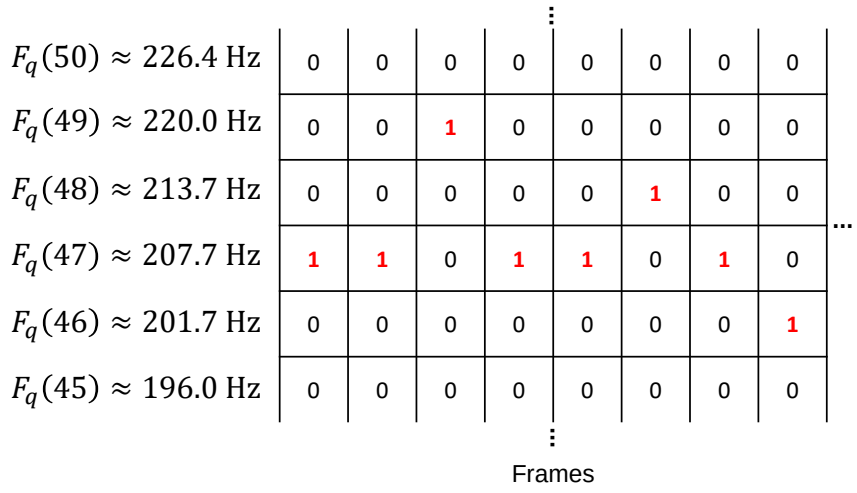

Figure 5.5: Matrix containing trajectory for row-wise median filtering.

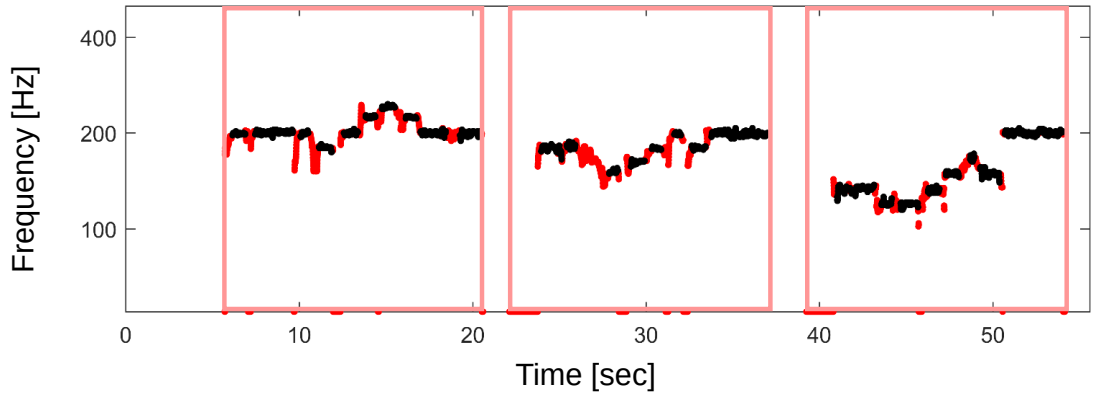

Figure 5.6: Reference F0 trajectory for our running example with stable pitches marked in black.

in Section 5.2. We again use a median filter of length 87 frames $\widehat{=}$ 0.5 s in this experiment. Finally, the resulting filtered matrix can then be transformed back to a trajectory vector by reversing step two. Taking the column-wise maximum of the matrix yields a vector indicating stable pitches by ones and an unstable or non-existent pitches by zeros.

The result of this procedure, using a median filter of length 87 frames $\widehat{=}$ 0.5 s, is shown in Figure 5.6, where the detected stable parts of the reference trajectory are colored in black. Note that "stable" zero-parts of the trajectory are neglected in this visualization. The filtering results for our running example mostly correspond to the auditory impression. However, tones that are perceived as stable, such as the tone at the end of the first segment, may get cut into two or more pieces. Detecting long tones as stable while avoiding short tones to be smoothed out requires a fine tuning of quantization stepsize and filter length, which is not discussed here. The accumulated curve for the row-wise filtered trajectories is marked as a dotted line in Figure 5.4. It is clearly visible that the resulting distribution is even more enhanced than the distribution obtained

by median filtering along trajectories. A significant improvement is again visible at the peaks around 0 cents, 700 cents and 1200 cents. Furthermore, the local minima of the curve are much more distinct compared to the other curves.

5.4 Conclusions and Further Notes

In this chapter, we showed through interval analysis how audio processing techniques can be used to examine the nature of Georgian vocal music. Our analysis results confirmed the non-western-tempered nature of Georgian vocal music. Furthermore, we showed how median filters can be used to enhance the computed interval distributions and obtain stable pitches from F0 trajectories.

An evaluation of the chosen filtering parameters as well as of the effects on the interval distributions was beyond the scope of this thesis and is left for future research. The analyses in this chapter should rather be seen as a starting point or an inspiration for similar studies on the so-called "Georgian sound-scale controversy". Future research in this domain will also require cooperation and exchange of knowledge between signal processing experts and (ethno-)musicologists.

Chapter 6

Summary and Future Work

In this thesis, we developed and examined audio processing techniques for analyzing a historically important set of Georgian chant recordings by the former master chanter Artem Erkomaishvili.

We saw that the segmentation of the recordings (Chapter 3) can be achieved by multiple means. Our audio matching-based approach performs well in a rather artificial scenario where information about the first segment is given. The machine-learning-based approach is more suitable in a real-world scenario where usually no prior knowledge is available. However, this approach needs much data to work properly. In general, we assert that analyzing audio material requires carefully choosing feature representations and parameter settings, since they turned out to have a great impact on the final analysis results. Exhaustive trial and error is often necessary to find "optimal" parameter settings, including trade-offs for time-frequency resolution, or computing time.

Our comparison of fundamental frequency estimation algorithms in Chapter 4 showed that all algorithms perform well in the first monophonic segment but have problems in the second and third polyphonic segments. Based on the performance of our three-stage approach, we can infer the following tendency for salience-based algorithms: The more voices are present in the recording, the more guidance is needed for the algorithms in order to achieve reasonable results. Although our comparison may not suffice to make general statements about the performance of the compared F0 estimation algorithms, it at least indicates strengths and weaknesses of the algorithms and also serves as a starting point for further research.

The interval analysis on the reference F0 annotations conducted in Chapter 5 is only one of many ways how audio processing techniques can support research on Georgian vocal music. In this final chapter, we again want to emphasize the publicly available reference annotations for segmentation and F0 estimation ([29]), which can be a useful starting point for ethnomusicological and MIR studies. With this work, we hope to encourage ethnomusicologists and MIR researchers to further collaborate for research on Georgian vocal music and beyond.

Future research in this domain will focus on F0 estimation on the new dataset of Georgian chants recorded by Scherbaum et al. (see Section 2.1.3). Especially the combination of throat microphone recordings, which contain separate voices, and room microphone recordings, which contain a mix of all voices, forms an interesting scenario for evaluating the performance of F0 estimation algorithms. Preliminary work will include syncing audio and video files, as well as generating processing pipelines and interfaces to facilitate working with the data.

Appendix A

Dataset Description

The original dataset of Georgian chant recordings by Artem Erkomaishvili as available at [43] was renamed according to a unified naming convention in the course of this project. The table presented in this chapter defines the mapping between the original file names and the new naming convention.

The assigned song IDs follow the IDs introduced in [42]. The book contains transcriptions of 118 songs with the corresponding song names given in Georgian and English. Each file of the publicly-available dataset was assigned to one transcription in the book by translating and/or comparing the transcription with the recording.

Empty table fields indicate songs, which were transcribed in the book, but were not included in the publicly-available dataset. The abbreviation *GCH* stands for *Georgian Chant Hymns*, the appendix *Erkomaishvili* for the former master chanter Artem Erkomaishvili introduced in Section 2.1.2.

In addition to renaming, all files of the dataset have been transcoded from the original 128 kbit/s MP3 to mono WAV files with a sampling rate of 22 050 Hz.

ID	Title Georgian	Web File	New Name
001	Qrist'e aghsdga	Artemi_-_Kriste_Agdga.mp3	GCH_001_Erkomaishvili.wav
002	Aghdgomasa shensa	Artemi_-_Agdgomasa_Shensa.mp3	GCH_002_Erkomaishvili.wav
003	Qrist'e aghsdga	Artemi_-_Kriste_Agdga3.mp3	GCH_003_Erkomaishvili.wav
004	Qrist'e aghsdga	Artemi_-_Kriste_Agdga4.mp3	GCH_004_Erkomaishvili.wav
005	Aghdgomisa dghe ars	Artemi_-_Agdgomisa_Dge_Ars.mp3	GCH_005_Erkomaishvili.wav
006	Ganvits'midnet satsnobelni	Artemi_-_Ganvicmidnet_Sacnobelni.mp3	GCH_006_Erkomaishvili.wav
007	Tsani q'ovlad ghirsabit	Artemi_-_Tsani_Kovlad_Girsebit.mp3	GCH_007_Erkomaishvili.wav
008	Movedit da vsvat	Artemi_-_Movedit_Da_Vsvat.mp3	GCH_008_Erkomaishvili.wav
009	Ats' q'ovliturt aghivso	Artemi_-_Ats_Kovliturt.mp3	GCH_009_Erkomaishvili.wav
010	Gushin shentana	Artemi_-_Gushin_Shentana.mp3	GCH_010_Erkomaishvili.wav
011	Saghmrtosa sakhmilavsa zeda	Artemi_-_Sagmrtosa_Sakhmilavsa.mp3	GCH_011_Erkomaishvili.wav
012	Esa ars ts'mida da chinebul dghe	Artemi_-_Ese_Ars_Tsminda.mp3	GCH_012_Erkomaishvili.wav
013	Mamao q'ovlisa mp'q'robelo	Artemi_-_Mamao_Kovlisa_Mpkrobelo.mp3	GCH_013_Erkomaishvili.wav
014	Adidebs suli chemi	Artemi_-_Agdgomis_IXdzilispiris_Chasartavi.mp3	GCH_014_Erkomaishvili.wav
015	Angelozi ghaghadebs	Artemi_-_Angelozi_Gagadebs.mp3	GCH_015_Erkomaishvili.wav
016	Ganatldi, ganatldi	Artemi_-_Ganatldi_Ganatldi.mp3	GCH_016_Erkomaishvili.wav
017	Adidebs suli chemi	Artemi_-_Agdgomis_IXdzlispiri.mp3	GCH_017_Erkomaishvili.wav
018	Angelozi ghaghadebs	Artemi_-_Angelozi_Gagadebs2.mp3	GCH_018_Erkomaishvili.wav
019	Ganatldi, ganatldi	Artemi_-_Ganatldi_Ganatldi2.mp3	GCH_019_Erkomaishvili.wav
020	P'aseki brts'q'invaled	Artemi_-_Paseki_Brtskinvaled_Mshvenieri.mp3	GCH_020_Erkomaishvili.wav
021	Da chven mogvanich'a		
022	Meokhebita ghvtismshobelisata	Artemi_-_Agdgomis_Antiponebi.mp3	GCH_022_Erkomaishvili.wav
023	Gvatskhovnen chven dzeo ghmrtisao		
024	Eklesiasa shina	Artemi_-_Eklesiasa_Shina.mp3	GCH_024_Erkomaishvili.wav
025	Qrist'e aghsdga	Artemi_-_Kriste_Agdga2.mp3	GCH_025_Erkomaishvili.wav
026	Tsiskarsa mstvad movida mariam	Artemi_-_Tsiskarsa_Mstvad.mp3	GCH_026_Erkomaishvili.wav
027	Daghatsatu nebsit tvisit	Artemi_-_Dagatsatu_Nebsit_Tvisit.mp3	GCH_027_Erkomaishvili.wav
028	Shobaman shenmam, ghvtismshobelo		
029	Adidebs suli chemi	Artemi_-_Gvtismshoblis_IXdzlispiris_Chasartavi.mp3	GCH_029_Erkomaishvili.wav
030	Romelman shev mtiebi	Artemi_-_Romelman_Shev.mp3	GCH_030_Erkomaishvili.wav
031	Qrist'es shobasa vadidebdet	Artemi_-_Kristes_Shobasa_Vadidebdet.mp3	GCH_031_Erkomaishvili.wav
032	Sasts'aulita ikhsna eri upalman	Artemi_-_Sastsaulit_Ikhsna_Eri_Tvisi_Upalma.mp3	GCH_032_Erkomaishvili.wav
033	Ghmerto, mokheden monata galobasa	Artemi_-_Gmerto_Mokheden.mp3	GCH_033_Erkomaishvili.wav
034	Kvertkhi ieses dzirisagan	Artemi_-_Kvertkhi_Ieses.mp3	GCH_034_Erkomaishvili.wav
035	Adide, sulo chemo	Artemi_-_Shobis_IXdzlispiris_Chasartavi.mp3	GCH_035_Erkomaishvili.wav
036	Saidumlo, utskho da didebuli	Artemi_-_Saidumlo_Utskho_Da_Didebuli.mp3	GCH_036_Erkomaishvili.wav
037	Saidumlo, utskho da didebuli		
038	Meokhebita ghvtismshobelisata		
039	Gvatskhovnen chven dzeo ghmrtisao		
040	Sashod mtiebisa	Artemi_-_Sashod_Mtiebisa.mp3	GCH_040_Erkomaishvili.wav
041	Shobaman shenmam, qrist'e ghmerto	Artemi_-_Shobaman_Shenman.mp3	GCH_041_Erkomaishvili.wav
042	Qalts'uli dghes arsebad	Artemi_-_Kaltsuli_Dges_Arsebad.mp3	GCH_042_Erkomaishvili.wav
043	Meokhebita ghvtismshobelisata	Artemi_-_Natlisgebis_Antiponebi.mp3	GCH_043_Erkomaishvili.wav
044	Gvatskhovnen chven dzeo ghmrtisao		
045	Kurtkhul ars momavali	Artemi_-_Kurtkheul_Ars_Momavali.mp3	GCH_045_Erkomaishvili.wav
046	Razhams iordanes natel ighe	Artemi_-_Rajams_Iordanes.mp3	GCH_046_Erkomaishvili.wav
047	Raodenta qrist'es mier	Artemi_-_Raodenta_Kriste_Mier.mp3	GCH_047_Erkomaishvili.wav
048	Adide sulo chemo	Artemi_-_Natlisgebis_IXdzlispiris_Chasartavi.mp3	GCH_048_Erkomaishvili.wav
049	Vershemdzlebel vart didebad shenda	Artmi_-_Ver_Shemdzlebel_Vart.mp3	GCH_049_Erkomaishvili.wav
050	Gikharoden mimadlebulo	Artemi_-_Gikharoden_Mimadlebulo.mp3	GCH_050_Erkomaishvili.wav
051	Ghvtismshobelo qalts'ulo, sasoebao	Artemi_-_Mirqmis_IXdzlispiris_Chasartavi.mp3	GCH_051_Erkomaishvili.wav
052	Ts'erilta mier sjulisata	Artemi_-_Tserilta_Mier_Sjulisata.mp3	GCH_052_Erkomaishvili.wav
053	Ats' ganut'eve	Artemi_-_Ats_Ganuteve.mp3	GCH_053_Erkomaishvili.wav
054	Dghes tskhovrebisa chvenisa	Artemi_-_Dges_Tskhovrebisa_Chvenisa.mp3	GCH_054_Erkomaishvili.wav
055	Aghaghe p'iri chemi		
056	Kidobansa nas sjulisasa	Artemi_-_Kidobansa_Mas_Sjulisasa.mp3	GCH_056_Erkomaishvili.wav
057	Ghirsad gabriel qalts'uls akhara	Artemi_-_Girsad_Gabriel.mp3	GCH_057_Erkomaishvili.wav
058	Meokhebita ghvtismshobelisata	Artemi_-_Peristsvalebis_Antiponebi.mp3	GCH_058_Erkomaishvili.wav
059	Gvatskhovnen chven dzeo ghmrtisao		
060	Upalo mogvivline nateli	Artmi_-_Upalo_Mogvivline.mp3	GCH_060_Erkomaishvili.wav
061	Mtasa zeda peri itsvale qrist'e	Artemi_-_Mtasa_Zeda.mp3	GCH_061_Erkomaishvili.wav
062	Adide sulo chemo	Artemi_-_Peristsvalebis_IXdzlispiris_Chaartavi.mp3	GCH_062_Erkomaishvili.wav
063	Shoba sheni ukhrts'nel ars	Artemi_-_Shoba_Sheni_Ukhrtsnel_Ars.mp3	GCH_063_Erkomaishvili.wav
064	Dghes saghmrtoman madlman		
065	Mots'apeta ra ikhiles	Artemi_-_Bzobis_IXdzlispiris_Chasartavi.mp3	GCH_065_Erkomaishvili.wav
066	Ghmerti upali	Artemi_-_Gmerti_Upali.mp3	GCH_066_Erkomaishvili.wav
067	Nateli natlisagan movlinebuli	Artemi_-_Nateli_Natlisagan.mp3	GCH_067_Erkomaishvili.wav
068	Amaghldi didebit qrist'e ghmerto	Artemi_-_Amagldi_Didebit.mp3	GCH_068_Erkomaishvili.wav
069	Adidebs suli chemi	Artemi_-_Amaglebis_IXdzlispiris_Chasartavi.mp3	GCH_069_Erkomaishvili.wav

070	Shev, qalts'ulo	Artemi_-_Shev_Kaltsulo.mp3	GCH_070_Erkomaishvili.wav
071	Razhams mokhvide ghmerti	Artemi_-_Rajams_Mokhvide_Gmerti.mp3	GCH_071_Erkomaishvili.wav
072	Razhams didebulni mots'apeni	Artemi_-_Rajams_Didebulni_Motsapeni.mp3	GCH_072_Erkomaishvili.wav
073	Shvenierman ioseb	Artemi_-_Mshvenierman_Ioseb.mp3	GCH_073_Erkomaishvili.wav
074	Razhams shtakhed saplavad	Artemi_-_Rajams_Shtakhed.mp3	GCH_074_Erkomaishvili.wav
075	Siq'varulman mogiq'vana		
076	Zetsisa mkhedrobata mtavarangelozno	Artmi_-_Zetsisa_Mkhedrobata.mp3	GCH_076_Erkomaishvili.wav
077	T'q'veta ganmatavisuplebelo	Artemi_-_Tkveta_Ganmantavisuplebelo.mp3	GCH_077_Erkomaishvili.wav
078	Barbares ts'midasa p'at'ivs vstsemdet	Artemi_-_Barbares_Tropari.mp3	GCH_078_Erkomaishvili.wav
079	Q'ovelsa qveq'anasa	Artemi_-_Basili_Didis_Tropari.mp3	GCH_079_Erkomaishvili.wav
080	Sit'q'visa ghvtisa	Artemi_-_Sitkvisa_Gvtisa.mp3	GCH_080_Erkomaishvili.wav
081	Motsiquli qrist'esagan gamorcheuli	Artemi_-_Motsikuli_Kristesgan_Gamorcheuli.mp3	GCH_081_Erkomaishvili.wav
082	Dghes mokharul ars eri qartvelta		
083	Ts'inamorbedisa didebulisa	Artemi_-_Tsinamorbedisa_Didebulisa.mp3	GCH_083_Erkomaishvili.wav
084	Mertskhalo mshveniero		
085	Mrts'amsi	Artemi_-_Mrtsamsi.mp3	GCH_085_Erkomaishvili.wav
086	Ts'q'aloba, mshvidoba	Artemi_-_Tskaloba_Mshvidoba.mp3	GCH_086_Erkomaishvili.wav
087	Da sulisatsa	Artemi_-_Da_Sulisatsa.mp3	GCH_087_Erkomaishvili.wav
088	Gvaqvs uplisa mimart	Artemi_-_Gvakvs_Uplisa_Mimart.mp3	GCH_088_Erkomaishvili.wav
089	Ghirs ars da martal	Artemi_-_Girs_Ars_Da_Martal.mp3	GCH_089_Erkomaishvili.wav
090	Ts'mida ars, ts'mida ars	Artemi_-_Tsmindao_Tsmindao.mp3	GCH_090_Erkomaishvili.wav
091	Shen gigalobt	Artemi_-_Shen_Gigalobt.mp3	GCH_091_Erkomaishvili.wav
092	Ghirs ars ch'eshmarit'a	Artemi_-_Girs_Ars.mp3	GCH_092_Erkomaishvili.wav
093	Q'ovelta da q'ovlisatvis	Artemi_-_Kovelta_Da_Kovlisatvis.mp3	GCH_093_Erkomaishvili.wav
094	Mamao chveno	Artemi_-_Mamao_Chveno.mp3	GCH_094_Erkomaishvili.wav
095	Mamao chveno	Artemi_-_Mamao_Chveno2.mp3	GCH_095_Erkomaishvili.wav
096	Shen, upalo		
097	Amin	Artemi_-_Amin.mp3	GCH_097_Erkomaishvili.wav
098	Ert ars ts'mida	Artemi_-_Ert_Ars_Tsminda.mp3	GCH_098_Erkomaishvili.wav
099	Khortsi qrist'esi movighot	Artmi_-_Khortsi_Kristesi.mp3	GCH_099_Erkomaishvili.wav
100	Dideba mamasa da dzesa	Artemi_-_Dideba_Atsda.mp3	GCH_100_Erkomaishvili.wav
101	Upalo romelman q'ovladts'mida	Artmi_-_Upalo_Romelman.mp3	GCH_101_Erkomaishvili.wav
102	Guli ts'mida dabade	Artemi_-_Guli_Tsmida_Da_Romelman.mp3	GCH_102_Erkomaishvili.wav
103	Romelman meeqvsesa dghesa		
104	Romelman metskhresa zhamsa	Artemi_-_Romelman_Metskhresa_Jamsa.mp3	GCH_104_Erkomaishvili.wav
105	Upalo ghaghadvq'av shendami	Artmi_-_Upalo_Gagadvkav.mp3	GCH_105_Erkomaishvili.wav
106	Ts'aremarten lotsva chemi	Artemi_-_Tsaremarten_Lotsva_Chemi.mp3	GCH_106_Erkomaishvili.wav
107	Ats' dzalni tsatani	Artemi_-_Ats_Dzalini_Tsatani.mp3	GCH_107_Erkomaishvili.wav
108	Shendami ikharebs	Artemi_-_Shendami_Ikharebs.mp3	GCH_108_Erkomaishvili.wav
109	Zetsit gamochinebulisa	Artmi_-_Zetsit_Gamochinebulisa.mp3	GCH_109_Erkomaishvili.wav
110	Isp'ola	Artemi_-_Ispola.mp3	GCH_110_Erkomaishvili.wav
111	T'on desp'ot'in	Artemi_-_Ton_Despotin.mp3	GCH_111_Erkomaishvili.wav
112	Kirieleison	Artemi_-_Kirie_Leison.mp3	GCH_112_Erkomaishvili.wav
113	Aqsios	Artemi_-_Aksios.mp3	GCH_113_Erkomaishvili.wav
114	Isaia mkhiarul iq'av	Artemi_-_Esaia_Mkhiarul.mp3	GCH_114_Erkomaishvili.wav
115	Ts'mindano mots'ameno	Artemi_-_Tsmindano_Motsapeno.mp3	GCH_115_Erkomaishvili.wav
116	Dideba shenda qrist'e ghmerto	Artemi_-_Dideba_Shenda_Kriste_Gmerto.mp3	GCH_116_Erkomaishvili.wav
117	Mosvlisa shenisa		
118	Shen khar venakhi		

Appendix B

Diagonal Matching Outliers

In this chapter, we list the matching curves belonging to the recordings where the peak-picking on the matching curves did not succeed in finding the ground truth positions ($\Delta t_{i1} > 10\,\mathrm{s}$, s. Figure 3.8). The figures were created using refined log-frequency representations for database and query based on a STFT with $N = 4096$ samples and $H = N/8$.

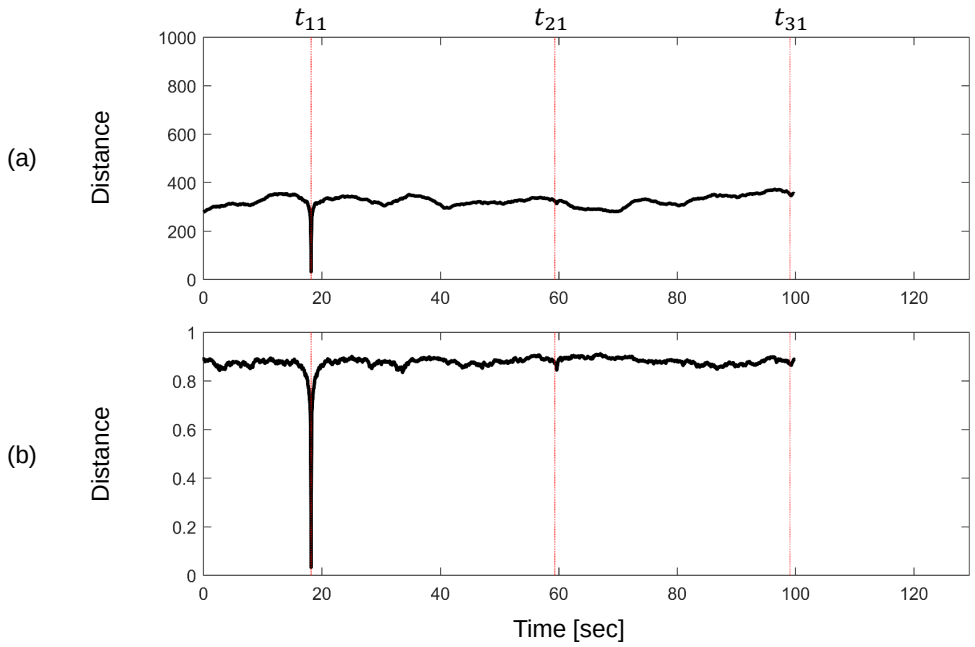

Figure B.1: Matching curve for GCH_001_Erkomaishvili.wav. Lead voice barely audible in second and third segment. **(a)** Euclidean distance.**(b)** Cosine distance.

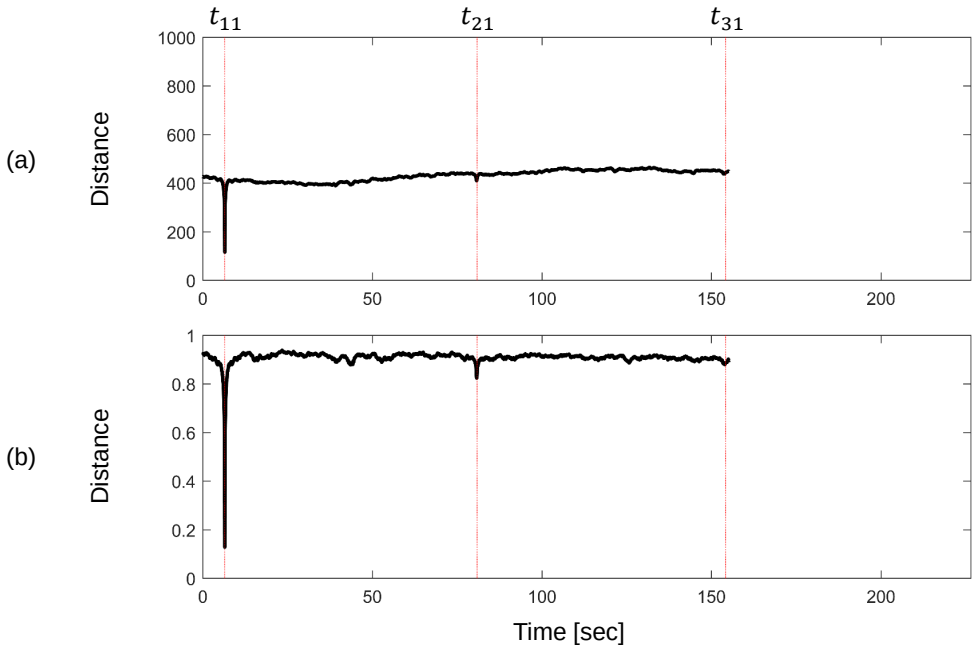

Figure B.2: Matching curve for GCH_015_Erkomaishvili.wav. Lead voice barely audible in third segment. **(a)** Euclidean distance; a slight rise in dynamics is visible. **(b)** Cosine distance.

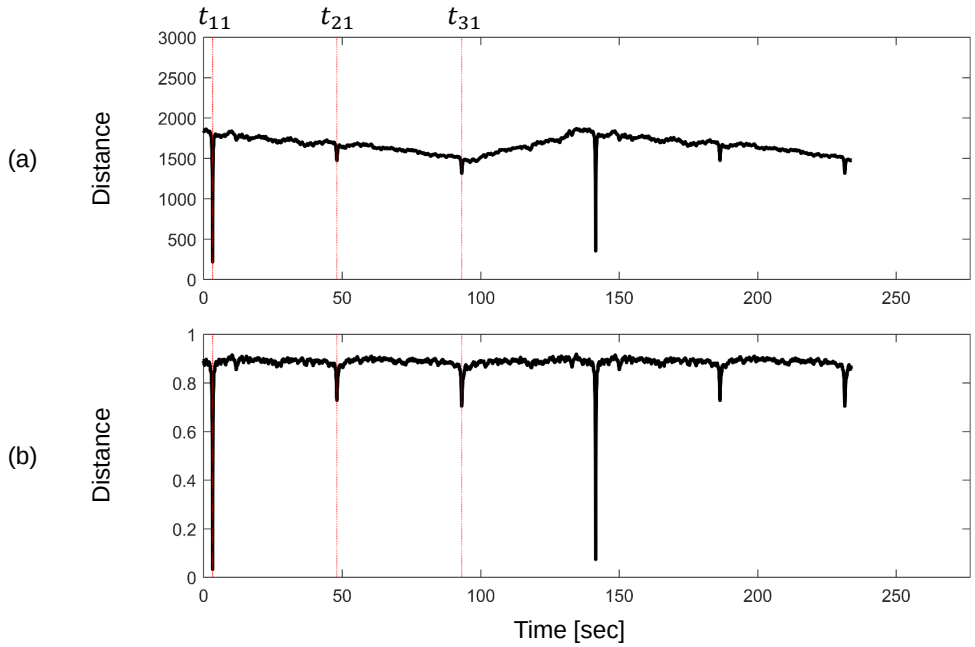

Figure B.3: Matching curve for GCH_048_Erkomaishvili.wav. Recording contains two times the same piece. Presumably, due to similar peak characteristics, the first recording is simply repeated a second time. **(a)** Euclidean distance; fluctuations in dynamics are visible. **(b)** Cosine distance.

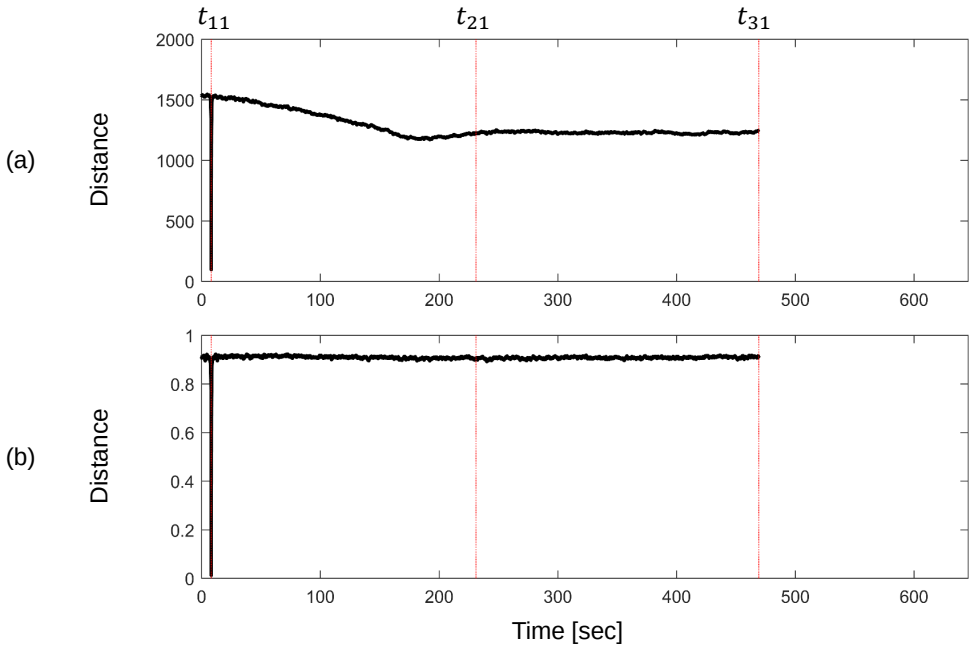

Figure B.4: Matching curve for GCH_107_Erkomaishvili.wav. Decreasing velocity and pitch in the course of the recording. **(a)** Euclidean distance; a drop in dynamics is visible. **(b)** Cosine distance.

Appendix C

F0 Estimation Outliers

In this chapter, we illustrate the performance of our three-stage approach by showing two examples with low average pitch accuracy (Figure C.1) and two examples with high average pitch accuracy (Figure C.2). The examples are based on a STFT with window length $N = 1024$ samples and hopsize $H = 256$ samples. The maximum pitch offset is set to 50 cents.

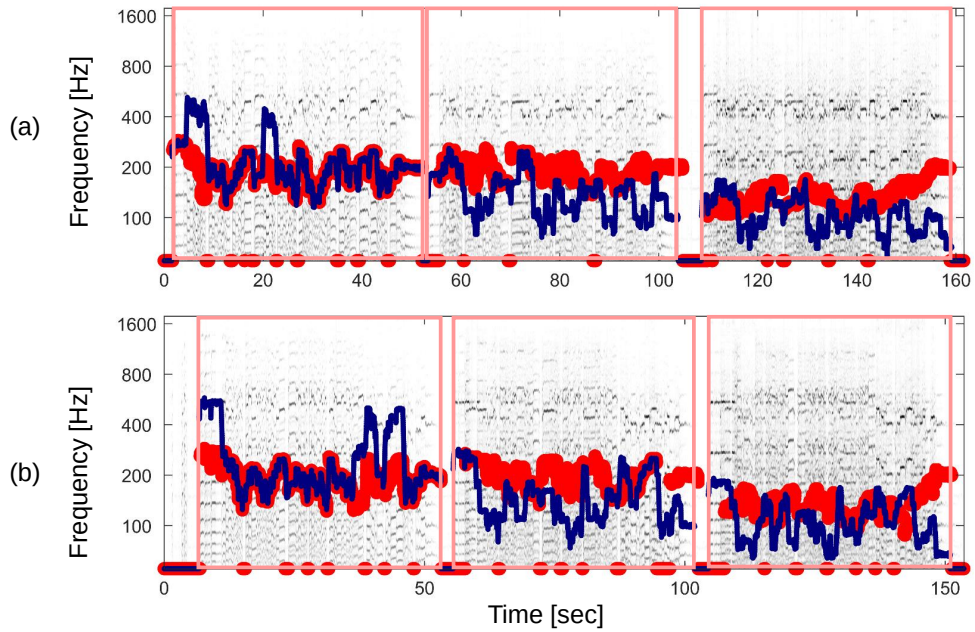

Figure C.1: F0 trajectory generated with three-stage approach (dark blue). The reference trajectory is visualized in bold red. **(a)** GCH_097_Erkomaishvili.wav. **(b)** GCH_093_Erkomaishvili.wav.

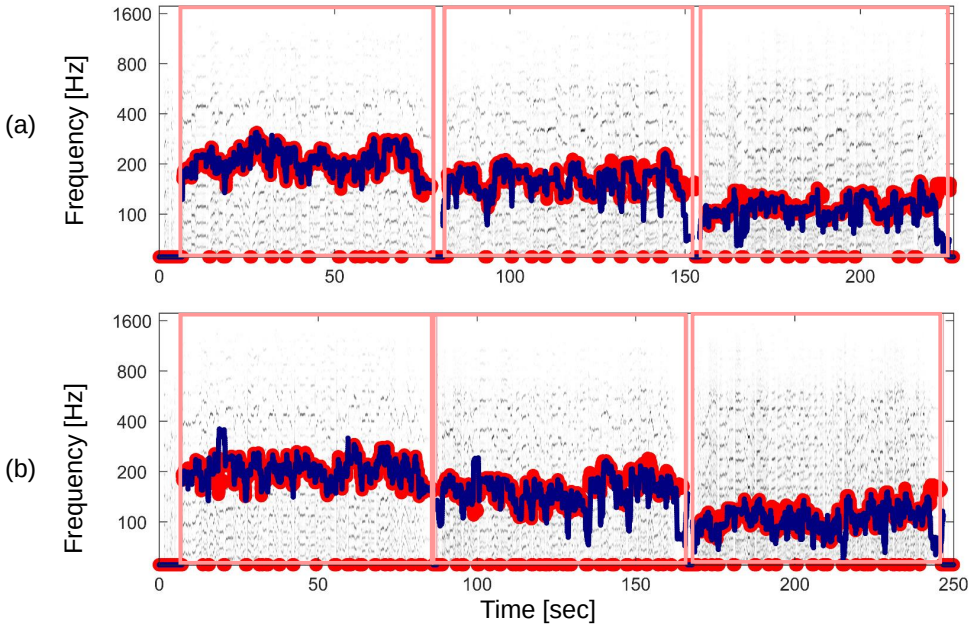

Figure C.2: F0 trajectory generated with three-stage approach (dark blue). The reference trajectory is visualized in bold red. **(a)** GCH_015_Erkomaishvili.wav. **(b)** GCH_016_Erkomaishvili.wav.

Bibliography

[1] J. P. Bello, *Lecture notes in Music Information Retrieval*. `http://www.nyu.edu/classes/bello/MIR_files/1-Introduction.pdf`, last accessed August 29, 2017.

[2] C. M. Bishop, *Pattern recognition and machine learning*, Springer, New York, 2006.

[3] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, *MedleyDB: A multitrack dataset for annotation-intensive MIR research*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Taipei, Taiwan, 2014, pp. 155–160.

[4] J. J. Bosch, R. Marxer, and E. Gómez, *Evaluation and combination of pitch estimation methods for melody extraction in symphonic classical music*, Journal of New Music Research, 45 (2016), pp. 101–117.

[5] C. Cannam, M. O. Jewell, C. Rhodes, M. Sandler, and M. d'Inverno, *Linked data and you: Bringing music research software into the semantic web*, Journal of New Music Research, 39 (2010), pp. 313–325.

[6] C. Cannam, C. Landone, and M. B. Sandler, *Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files*, in Proceedings of the International Conference on Multimedia, Florence, Italy, 2010, pp. 1467–1468.

[7] A. de Cheveigné and H. Kawahara, *YIN, a fundamental frequency estimator for speech and music.*, Journal of the Acoustical Society of America (JASA), 111 (2002), pp. 1917–1930.

[8] C. Dittmar, B. Lehner, T. Prätzlich, M. Müller, and G. Widmer, *Cross-version singing voice detection in classical opera recordings*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), Málaga, Spain, October 2015, pp. 618–624.

[9] J. Driedger, *Processing Music Signals Using Audio Decomposition Techniques*, PhD thesis, Friedrich-Alexander Universität Erlangen-Nürnberg, 2016.

[10] J. Driedger and M. Müller, *Verfahren zur Schätzung der Grundfrequenzverläufe von Melodiestimmen in mehrstimmigen Musikaufnahmen*, in Musikpsychologie – Anwendungsorientierte Forschung, W. Auhagen, C. Bullerjahn, and R. von Georgi, eds., vol. 25 of Jahrbuch Musikpsychologie, Hogrefe-Verlag, 2015, pp. 55–71.

[11] D. P. W. Ellis, *Robust landmark-based audio fingerprinting.* Website `https://labrosa.ee.columbia.edu/matlab/fingerprint/`, last accessed August 29, 2017.

[12] M. Erkvanidze, *The Georgian musical system*, in 6th International Workshop on Folk Music Analysis, Dublin, Ireland, 2016.

[13] D. FitzGerald, *Harmonic/percussive separation using median filtering*, in Proceedings of the International Conference on Digital Audio Effects (DAFx), Graz, Austria, 2010, pp. 246–253.

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

[15] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.*, Springer, 2009.

[16] T. Helten, M. Müller, J. Tautges, A. Weber, and H.-P. Seidel, *Towards cross-modal comparison of human motion data*, in Proceedings of the Annual Symposium of the German Association for Pattern Recognition (DAGM), vol. 6835 of Lecture Notes in Computer Science, Frankfurt, Germany, 2011, Springer, pp. 61–70.

[17] W. Hess, *Pitch Determination of Speech Signals*, Springer-Verlag, Berlin, 1983.

[18] R. Jain, R. Kasturi, and B. G. Schunck, *Machine Vision*, McGraw-Hill, 1995.

[19] A. P. Klapuri, *Multiple fundamental frequency estimation by summing harmonic amplitudes*, in Proceedings of the International Society for Music Information Retrieval Conference (ISMIR), 2006, pp. 216–221.

[20] P. López-Serrano, C. Dittmar, and M. Müller, *Towards modeling and decomposing loop-based electronic music*, in Proceedings of the International Conference on Music Information Retrieval (ISMIR), New York, USA, 2016, pp. 502–508.

[21] M. Mauch, C. Cannam, R. Bittner, G. Fazekas, J. Salamon, J. Dai, J. Bello, and S. Dixon, *Computer-aided melody note transcription using the Tony software: Accuracy and efficiency*, in Proceedings of the International Conference on Technologies for Music Notation and Representation, May 2015.

[22] M. Mauch and S. Dixon, *Vamp plugins*. Website `http://vamp-plugins.org/download.html?platform=win32&search=pyin&go=Go`, last accessed August 29, 2017.

[23] ——, *pYIN: A fundamental frequency estimator using probabilistic threshold distributions*, in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, pp. 7480–7484.

[24] MIREX 2016. Audio Melody Extraction Subtask. `http://www.music-ir.org/mirex/wiki/2016:MIREX2016_Results`, last accessed August 29, 2017.

[25] M. Müller, *Information Retrieval for Music and Motion*, Springer Verlag, 2007.

[26] ——, *Fundamentals of Music Processing*, Springer Verlag, 2015.

[27] M. Müller and S. Ewert, *Towards timbre-invariant audio features for harmony-based music*, IEEE Transactions on Audio, Speech, and Language Processing, 18 (2010), pp. 649–662.

[28] M. Müller, P. Grosche, and F. Wiering, *Automated analysis of performance variations in folk song recordings*, in Proceedings of the International Conference on Multimedia Information Retrieval (MIR), Philadelphia, Pennsylvania, USA, 2010, pp. 247–256.

[29] M. Müller, S. Rosenzweig, J. Driedger, and F. Scherbaum, *Fundamental frequency annotations for Georgian chant recordings*. Website `https://www.audiolabs-erlangen.de/resources/MIR/2017-GeorgianMusic-Erkomaishvili`, last accessed August 29, 2017.

[30] ——, *Interactive fundamental frequency estimation with applications to ethnomusicological research*, in Proceedings of the AES International Conference on Semantic Audio, Erlangen, Germany, 2017.

[31] H. Myers, *Ethnomusicology*, Norton & Company, 1992.

[32] J. Paulus, M. Müller, and A. Klapuri, *Audio-based music structure analysis*, in Proceedings of the International Conference on Music Information Retrieval (ISMIR), Utrecht, The Netherlands, 2010, pp. 625–636.

[33] G. E. Poliner, D. P. Ellis, A. F. Ehmann, E. Gómez, S. Streich, and B. Ong, *Melody transcription from music audio: Approaches and evaluation*, IEEE Transactions on Audio, Speech, and Language Processing, 15 (2007), pp. 1247–1256.

[34] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall Signal Processing Series, 1993.

[35] J. Salamon and E. Gómez, *Melodia - melody extraction vamp plug-in*. Website `https://www.upf.edu/web/mtg/melodia`, last accessed August 29, 2017.

[36] J. Salamon and E. Gómez, *Melody extraction from polyphonic music signals using pitch contour characteristics*, IEEE Transactions on Audio, Speech, and Language Processing, 20 (2012), pp. 1759–1770.

[37] J. Salamon, E. Gómez, D. P. W. Ellis, and G. Richard, *Melody extraction from polyphonic music signals: Approaches, applications, and challenges*, IEEE Signal Processing Magazine, 31 (2014), pp. 118–134.

[38] F. Scherbaum, *On the benefit of larynx-microphone field recordings for the documentation and analysis of polyphonic vocal music*, in 6th International Workshop on Folk Music Analysis, Dublin, Ireland, 2016, pp. 80–87.

[39] F. Scherbaum, W. Loos, F. Kane, and D. Vollmer, *Body vibrations as source of information for the analysis of polyphonic vocal music*, in Proceedings of the International Workshop on Folk Music Analysis, vol. 5, Paris, France, 2015, pp. 89–93.

[40] ——, *On the benefit of larynx-microphone field recordings for the documentation and analysis of polyphonic vocal music*, in Proceedings of the International Workshop on Folk Music Analysis, Dublin, Ireland, 2016, pp. 80–87.

[41] F. Scherbaum, M. Müller, and S. Rosenzweig, *Analysis of the Tbilisi State Conservatory recordings of Artem Erkomaishvili in 1966*, in Proceedings of the 7th International Workshop on Folk Music Analysis, Málaga, Spain, 2017, pp. 29–36.

[42] D. Shugliashvili, *Introduction*, in Georgian Church Hymns, Shemokmedi School, 2014, pp. 23–29.

[43] Tbilisi State Conservatory, Folklore Department, *Artem erkomaishvilis sagaloblebi*. Website `http://www.alazani.ge/old-archives-Artem-Erkomaishvilis-Sagaloblebi-folk-songs-ans59.html`, last accessed August 29, 2017.

[44] S. Theodoridis, *Machine Learning: A Bayesian and Optimization Perspective*, .NET Developers Series, Elsevier Science, 2015.

[45] Z. Tsereteli and L. Veshapidze, *On the Georgian traditional scale*, Tbilisi, Georgia, 2014, pp. 288–295.

[46] R. Tsurtsumia and J. Jordania, *Echoes from Georgia: Seventeen Arguments on Georgian Polyphony*, Nova Science Publishers, 2010.

Curriculum Vitae

Sebastian Rosenzweig was born in Erlangen, Germany. He received his B.Sc. in Mediatechnology from Ilmenau University of Technology in 2015. At the moment, he is a M.Sc. student in Communications and Multimedia Engineering at Friedrich-Alexander University Erlangen-Nürnberg. His research interests are audio signal processing and machine learning, with focus on musical audio.