# Automatic Chord Transcription from Audio Using Computational Models of Musical Context

Matthias Mauch

Thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
of the
University of London.

School of Electronic Engineering and Computer Science
Queen Mary, University of London

March 23, 2010

I certify that this thesis, and the research to which it refers, are the product of my own work, and that any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline. I acknowledge the helpful guidance and support of my supervisor, Dr Simon Dixon.

Matthias Mauch

# Abstract

This thesis is concerned with the automatic transcription of chords from audio, with an emphasis on modern popular music. Musical context such as the key and the structural segmentation aid the interpretation of chords in human beings. In this thesis we propose computational models that integrate such musical context into the automatic chord estimation process.

We present a novel dynamic Bayesian network (DBN) which integrates models of metric position, key, chord, bass note and two beat-synchronous audio features (bass and treble chroma) into a single high-level musical context model. We simultaneously infer the most probable sequence of metric positions, keys, chords and bass notes via Viterbi inference. Several experiments with real world data show that adding context parameters results in a significant increase in chord recognition accuracy and faithfulness of chord segmentation. The proposed, most complex method transcribes chords with a state-of-the-art accuracy of 73% on the song collection used for the 2009 MIREX Chord Detection tasks. This method is used as a baseline method for two further enhancements.

Firstly, we aim to improve chord confusion behaviour by modifying the audio front end processing. We compare the effect of learning chord profiles as Gaussian mixtures to the effect of using chromagrams generated from an approximate pitch transcription method. We show that using chromagrams from approximate transcription results in the most substantial increase in accuracy. The best method achieves 79% accuracy and significantly outperforms the state of the art.

Secondly, we propose a method by which chromagram information is shared between repeated structural segments (such as verses) in a song. This can be done fully automatically using a novel structural segmentation algorithm tailored to this task. We show that the technique leads to a significant increase in accuracy and readability. The segmentation algorithm itself also obtains state-of-the-art results. A method that combines both of the above enhancements reaches an accuracy of 81%, a statistically significant improvement over the best result (74%) in the 2009 MIREX Chord Detection tasks.

*für meinen Vater*

# Acknowledgements

I would like to thank my PhD supervisor Simon Dixon for three years of sound advice and friendly guidance, and my secondary supervisors Mark Sandler and Mark Plumbley for additional feedback on my work. I would also like to thank Mark Sandler for letting me join the OMRAS2 project. I have thoroughly enjoyed my time at the Centre for Digital Music.

I would like to thank Andreas Heuer and the *Deutsche Forschungsgemeinschaft* for supporting and funding the first year of my graduate studies in music computing at the University of Rostock.

I am grateful to several people who have given me excellent feedback on my research and my thesis in particular. Thank you, Samer Abdallah, Chris Cannam, Anssi Klapuri, Christophe Rhodes, Matthias Rust, Martin Schmidt, Dan Tidhar. I would particularly like to thank Katy Noland for many helpful comments on my work, especially in the last few months, and more generally for her support.

Many thanks to the systems team at the School for Electronic Engineering and Computer Science at Queen Mary, and in particular Matt Bernstein, for providing help and the computing facilities without which I could not have done the research for this thesis.

My work would not have been possible without Matthew Davies' beat-tracker and beat reference annotations, Christopher Harte's Beatles chord reference annotations, Kevin Murphy's *BNT Toolbox*, and Chris Cannam's *Sonic Visualiser*. Thank you for sharing your work. I would also like to thank the students who helped realise the OMRAS2 Metadata Project.

Further thanks to Amélie Anglade, Daniele Barchiesi, Mathieu Barthet, Mert Bay, Juan Pablo Bello, Emmanouil Benetos, Magdalena Chudy, Alice Clifford, Sharon Cording, Tim Crawford, Ivan Damnjanovic, Mark d'Inverno, Steven Downie, Andreas Ehmann, George Fazekas, Ben Fields, Masataka Goto, Aris Gretsistas, Bas de Haas, Steven Hargreaves, Peyman Heydarian, Sigrun Hoffmann, Hannelore Höft, Ho Huen, Temenushka Ignatova, Kurt Jacobson, Maria Jafari, Petra Kegler, Max Richard Kriesen and family, Sefki Kolozali, Panos Kudumakis, Francesco Lagona, Chris Landone, Mark Levy, David Lewis, Rob Macrae, Michela

Magas, Brian May, Lesley Mearns, Martin James Morrell, Joachim Mühling, Heidemarie Mühling, Daniel Müllensiefen, Tim Murray-Browne, Andrew Nesbit, Elias Pampalk, Enrique Perez Gonzalez, Polina Proutskova, Yves Raimond, Josh Reiss, Christophe Rhodes, Andrew Robertson, Adam Stark, Becky Stewart, Dan Stowell, Chris Sutton, Michael Terrell, Keisuke Toyama, Graham Tudball, Emmanuel Vincent, Beiming Wang, Steve Welburn, Wen Xue, Geraint Wiggins, Theresa Willis, Thomas Wilmering, Betty Woessner, Melissa Yeo, Asterios Zacharakis and Ruohua Zhou and to my mother and father.

This work was funded by the UK Engineering and Physical Sciences Research Council.

# Contents

# List of Tables

# List of Figures

# List of Acronyms

| | |
|---|---|
| 2-TBN | 2-slice temporal Bayesian network |
| ANOVA | analysis of variance |
| BN | Bayesian network |
| CPD | conditional probability distribution |
| CPU | central processing unit |
| DBN | dynamic Bayesian network |
| DFT | discrete Fourier transform |
| FFT | fast Fourier transform |
| HMM | hidden Markov model |
| MCE | minimum classification error |
| MIDI | musical instrument digital interface |
| MIR | music information retrieval |
| MIREX | music information retrieval evaluation exchange |
| NNLS | non-negative least squares (problem) |
| PCP | pitch class profile |
| RCO | relative correct overlap |
| STFT | short-time Fourier transform |
| SVM | support vector machine |
| TC | tonal centroid |
| WAV | waveform audio file format |

# Introduction 1

This thesis is concerned with the automatic transcription of chords from audio. In this chapter we explain the motivations and aim of our work (Sections 1.1 and 1.2) and provide an overview of the material presented in the thesis and its unique contributions (Section 1.3). Section 1.4 concludes the chapter with a list of our own publications relating to the thesis.

## 1.1 Motivation

Automatic chord detection systems analyse a piece of music, either in symbolic form or in the form of digital audio, and output a sequence of labels that describe the chords and chord changes in the piece. Our work focuses on the audio domain. We identify three different motivations for research in automatic chord detection from audio. Firstly, good chord labels of large collections of music are expected to aid music information retrieval tasks such as cover song retrieval. Secondly, since automatic chord detection aims at the imitation of human perception and cognition of harmony, it can be intrinsically motivated as a pure research problem. Thirdly, reliable computational methods can assist musicians and musicologists in the labour-intensive chord transcription task. The rest of this section clarifies these three aspects.

### A Basis for Music Information Retrieval

The interdisciplinary field of *music information retrieval* (*MIR*) research aims at enabling access, comparison and search of music or collections of music by means of computer systems. Much of the data used to access music nowadays is metadata such as information on the title or composer of a piece of music. A growing part of MIR is content-based MIR, i.e. computer systems that base their recommendations at least partly on data that has been extracted automatically from the music itself. So far, mainly low-level features such as mel frequency cepstral coefficients and chroma vectors have been used. Casey et al. (2008) give a comprehensive overview of content-based music information retrieval problems and emphasise the great importance of automatically extracted high-level descriptors such as structure, lyrics, chords and others. In fact, several MIR tasks have successfully used manually extracted chord labels, e.g. for composer retrieval (Ogihara and Li, 2008). Chord labels retrieved from audio have been used for genre classification (Anglade et al., 2009) and cover song retrieval (Bello, 2007). A different flavour of content-based MIR is the use of automatically extracted data for systematic musicological studies, and chord labels can open up new areas of research which otherwise would be too labour-intensive.

#### A Pure Research Problem

The transcription of chords is also interesting as an exercise in computer science or artificial intelligence. We can formulate the pure research question: is it possible to model human perception and reasoning well enough to generate similar chord transcriptions to those produced by highly-trained musicians? The behaviour of computer systems that perform chord transcription could also allow conclusions about the way humans perceive music, which would be worthwhile given how little we understand of human music perception. The research question could be: can methods of machine listening indicate how music is processed in humans?

#### Working towards Reliable Chord Transcription Aimed at Musicians

Automatic chord transcription can assist a human transcriber in performing the task of transcribing musical pieces more accurately and more quickly. The transcription has to be good enough for a musician to play along to the piece with his guitar or other instrument. The combination of two facts makes the automatic transcription of modern popular music and jazz a worthwhile engineering project: this music is so strongly based on chord progressions that often the chords alone provide enough information to perform a recognisable version of a song. On the other hand even these chords are usually published only for relatively few, very popular songs, as an afterthought to the actual recording. This scarcity of supply is contrasted by a great demand for chord transcriptions by (hobby) musicians and a vivid exchange of tens of thousands of home-made chord transcriptions on internet sites like “Ultimate Guitar”[1] and “E-Chords”[2]. However, even on websites with a very active community, it may be hard to find less popular songs, and an automatic tool could provide a welcome alternative.

## 1.2 Research Goal

The three kinds of motivation we have seen are, of course, strongly connected. However, it is the last one that has mainly driven the research for this thesis: methods for reliable chord

[1] http://www.ultimate-guitar.com

[2] http://www.e-chords.com/

transcription for musicians. With this motivation comes the main focus on modern popular music for the reasons described in Section 1.1. The main goal of the work in this thesis is to develop reliable methods for automatic chord transcription from real-world audio, with a focus on modern popular music. We intentionally use the term *chord transcription*, rather than chord extraction or chord detection, because it draws the parallel to the human activity which we measure our work by.

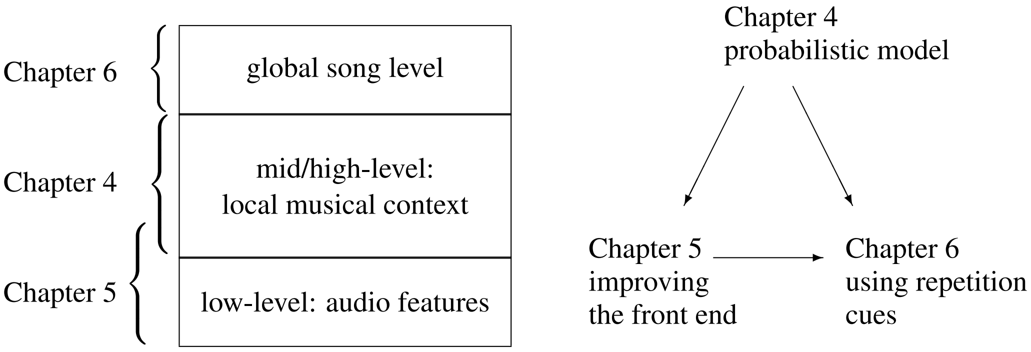
The concentration on this one motivation has a reason. Clearly, the features extracted by our methods can be used for different kinds of music and further processing in an MIR task, as is demonstrated in a recent collaboration (Anglade et al., 2010), but there is evidence suggesting that the optimal feature for retrieval is not always the optimal transcription in musical terms, and vice versa (Noland, 2009, Chapter 6). We believe that developing algorithms with the dual goal of transcription and retrieval performance could have compromised our focus.

The stated goal is a practical one. As a result, this thesis describes a collection of techniques derived from different disciplines that aid automatic chord transcription. We use music theory, theory of music perception, digital signal processing and probability theory, but all with the engineering approach of aiming at improving automatic chord transcription.

## 1.3 Contributions and Thesis Structure

This thesis treats low-level and high-level aspects of automatic chord transcription, and also considers the global song level structure. The main contributions can be found in Chapters 4, 5 and 6. The three chapters cover different levels of musical abstraction, as shown in Figure 1.1a: Chapter 4 describes the core concept: a probabilistic network to model local dependencies between a chord that occurs in a piece of music and its high-level context: the key, the metric position, the bass note, and their connection to the low-level features. Taking this model as a point of departure, Chapter 5 is concerned with the improvement of the interface between the low-level audio features and the higher-level model: the front-end. We compare a statistical learning approach and an approach using an alternative chroma feature. Chapter 6 is concerned with the global song level and proposes a method to improve chord transcription algorithms by exploiting the repetition structure of songs. In Chapter 6 we also present our most successful chord transcription method, which combines the proposed techniques from all three main chapters. The dependencies of the chapters are depicted in Figure 1.1b.

Figure 1.1a shows the abstraction from the audio signal: Chapter 4 defines the central mid/high-level probabilistic model; Chapter 5 contributes improvements in the low-level front



(a) Level of abstraction. (b) Dependencies.

Figure 1.1: Two aspects of the main chapters: the main chapters' contributions, by level of abstraction from the audio signal (Figure 1.1a), and dependencies of the methods proposed in the three main chapters (Figure 1.1b).

end and its representation in the probabilistic model; Chapter 6 explores improvements through repetition cues on the global song level. Figure 1.1b shows dependencies of the methods proposed in the three main chapters: both Chapter 5 and Chapter 6 rely on the probabilistic model proposed in Chapter 4. While the main experiments in Chapter 6 do not depend on the methods in Chapter 5, an additional experiment is conducted that combines the proposed techniques of all three main chapters in Chapter 6.

The following paragraphs summarise each chapter.

**Chapter 1: Introduction**

In this chapter we identify motives for research in chord detection and define the aim of chord transcription from audio. The thesis' main contributions are discussed.

**Chapter 2: Background**

This chapter gives an introduction to chords from the perspective of music theory, including some excursions to music perception, and presents metrics for the evaluation of automatically generated chord sequences. The chapter's focus is the section on related work (Section 2.2), with separate reviews of chord models, the calculation of chromagrams, and musical context models.

**Chapter 3: Beat-synchronous Bass and Treble Chromagrams**

This chapter provides a technical description of the baseline chroma extraction method we use. The three parts of the procedure: note salience computation, chroma wrapping, and beat-synchronisation are discussed in detail.

**Chapter 4: A Model for Chord Transcription**

In this chapter we propose and evaluate the musical probabilistic model for chord extraction that is central to this thesis. The novelty of the approach is that it integrates into a single dynamic Bayesian network (DBN) pieces of musical context that had previously been assessed only separately: key, chord, metric position and bass pitch class are estimated simultaneously. We show that thus increasing the amount of context significantly improves chord transcription performance, and the most complex of the proposed models shows state of the art performance.

**Chapter 5: Improving the Front End through Statistical Training and Approximate Transcription**

This chapter addresses a remaining problem of the DBN presented in Chapter 4: the confusion of some musically unrelated chords. We hypothesise that the problem stems from the low-level front end of the model and propose two different approaches to improve the front-end processing: by statistical learning of chord profiles, and by approximate transcription using a non-negative least squares algorithm. Improvements are achieved in both cases. The approximate transcription approach results in an upward leap in chord extraction accuracy, leading to a significant difference over state-of-the-art methods.

**Chapter 6: Using Repetition Cues to Enhance Chord Transcription**

This chapter introduces a simple technique to use repetition on the global song level to improve existing chord transcription methods. Low-level features are averaged across repetitions in order to emphasise systematic chordal information. We show that the use of the technique results in significantly more accurate chord transcriptions compared to the respective baseline methods discussed in Chapter 4 and Chapter 5. We also discuss that the use of repetition provides a qualitative improvement of the transcription by ensuring that repeated parts share the same chord progression.

**Chapter 7: Conclusions**

This chapter provides a summary of the achievements of this thesis. We end by outlining planned work and, more generally, what we deem worthwhile future work in the area of chord transcription.

## 1.4 Related Publications by the Author

Of the publications listed below, many have influenced the writing of this thesis. Two papers are of particular importance because they are the basis for Chapters 4 and 6. They are marked with asterisks. In both cases, the author was the main contributor to the publications, under supervision by Simon Dixon. Katy Noland's contributions are detailed in Chapter 6.

**Journal Paper**

*[3] Matthias Mauch and Simon Dixon: *Simultaneous Estimation of Chords and Musical Context from Audio*, to appear in IEEE Transactions on Audio, Speech, and Language Processing.

**Peer-Reviewed Conference Papers**

Dan Tidhar, Matthias Mauch and Simon Dixon: *High-Precision Frequency Estimation for Harpsichord Tuning Classification*, to appear in Proceedings of the 2010 IEEE International Conference on Acoustics, Speech and Language Processing, 2010.

*[4] Matthias Mauch, Katy Noland and Simon Dixon: *Using Musical Structure to Enhance Automatic Chord Transcription*, Proceedings of the 10th International Conference on Music Information Retrieval, Kobe, Japan, 2009, pages 231–236.

Matthias Mauch and Simon Dixon: *A Discrete Mixture Model for Chord Labelling*, Proceedings of the 9th International Conference on Music Information Retrieval, Philadelphia, USA, 2008, pages 45–50.

Matthias Mauch, Simon Dixon, Christopher Harte, Michael Casey and Benjamin Fields: *Discovering Chord Idioms Through Beatles and Real Book Songs*, Proceedings of the 8th International Conference on Music Information Retrieval, Vienna, Austria, 2007, pages 255–258.

**Other Publications**

Matthias Mauch, Chris Cannam, Matthew Davies, Simon Dixon, Christopher Harte, Sefki Kolozali, Dan Tidhar and Mark Sandler: *OMRAS2 Metadata Project 2009*[5], late-breaking session at the 10th International Conference on Music Information Retrieval, Kobe, Japan, 2009.

Matthias Mauch, Daniel Müllensiefen, Simon Dixon and Geraint Wiggins: *Can Statistical Language Models be Used for the Analysis of Harmonic Progressions?*, Proceedings of the 10th International Conference on Music Perception and Cognition, Sapporo, Japan, 2008.

[3] basis for Chapter 4

[4] basis for Chapter 6

[5] `http://ismir2009.ismir.net/proceedings/LBD-18.pdf`

### Under Review

Amélie Anglade, Emmanouil Benetos, Matthias Mauch and Simon Dixon: *Improving music genre classification using automatically induced harmony-based rules*, submitted to the Journal of New Music Research.

Dan Tidhar, George Fazekas, Matthias Mauch and Simon Dixon: *TempEst: Automatic Harpsichord Temperament Estimation in a Semantic Web Environment* submitted to the Journal of New Music Research.

## Conclusions

After considering the motivations for automatic chord detection, we have stated the goal of this thesis: to develop reliable methods for automatic chord transcription from real-world audio, with a focus on modern popular music. We have laid out the structure of the thesis and indicated the three chapters that contain its main contributions: based on the novel high-level musical context model (Chapter 4), we explore improvements on the low-level front end (Chapter 5) and the global song level (Chapter 6). The list of publications shows that the computational processing of chords has been our main priority over the last few years, and we have indicated which publications have directly influenced the contents of this thesis.

So far, we have only briefly described the methods we propose to improve chord transcription. The literature review in the following chapter brings together all the information needed to understand how our research goal has motivated the design of our methods, especially the inclusion of high-level musical context into the chord estimation process.

# Background and Related Work 2

A basic understanding of music theory and perception, and the knowledge of related work in computational harmony analysis are prerequisites to the development of new algorithms for chord transcription. This chapter reviews both of these aspects. Section 2.1 provides a music-theoretical definition of chords and introduces related concepts from music theory and music perception. In Section 2.2 we give a survey of related work in the field of computational chord transcription and harmony analysis. Section 2.3 presents techniques that have been used to evaluate automatic chord transcriptions.

## 2.1 Chords in Music Theory and Practice

To develop a basic understanding of what is required from a transcriber of chords—either human being or computer program—it is useful to review some background in music theory and practice, as well as some concepts of music perception and cognition.

### 2.1.1 Pitch and Pitch Class

Pitch is one of the most important concepts in tonal music, and harmony builds upon the human ability to perceive pitch. Klapuri (2006a) defines pitch as follows.

> *Pitch* is a perceptual attribute which allows the ordering of sounds on a frequency-related scale extending from low to high.

An further extensive treatment of musical pitch can be found in (Krumhansl, 1990). Pitch is approximately proportional to log-frequency. Notes of the chromatic scale in equal temperament, which divides an octave[1] into 12 parts, are also spaced linearly in log-frequency, i.e. the fundamental frequency $f_p$ of a note can be described in terms of the fundamental frequency of

[1] Notes that are an *octave* apart have a frequency ratio of $^2/_1$ .

the previous note as

$$f_p = 2^{1/12} f_{p-1}. \tag{2.1}$$

We will use both words, *pitch* as a perceptual quality and *note* as a musical quality, interchangeably. Before we see how chords arise from the combination of pitches, let us consider an important finding of a perceptual study conducted by Shepard (1964, page 159): human beings are able to separately consider *pitch class*, which refers to note names with no octave information, and *pitch height*, which is proportional to the fundamental frequency of a note. In particular they are able to perceive notes that are in octave relation as equivalent, a phenomenon called octave equivalence[2]. For chords, this has a peculiar consequence: "Transposition of one of the voices of a chord through an octave can make very little difference to the sound of the chord" (Parncutt, 1989). Parncutt (1989, Chapter 4) defines perceptual models of pitch and chroma salience, which express the probability that a pitch or pitch class is noticed over a musical element or passage. The perceptual concept of octave equivalence is mirrored by the use of chords in music and music theory. Although for the particular arrangement the voicing may be important, in terms of chord label all note combinations that have the same pitch classes are considered equivalent—with the possible exception of the position of the bass note. We will see how this "vertical" pitch abstraction affects chord syntax in Section 2.1.3. Before doing so let us get a clear idea of what we actually mean by *chord*.

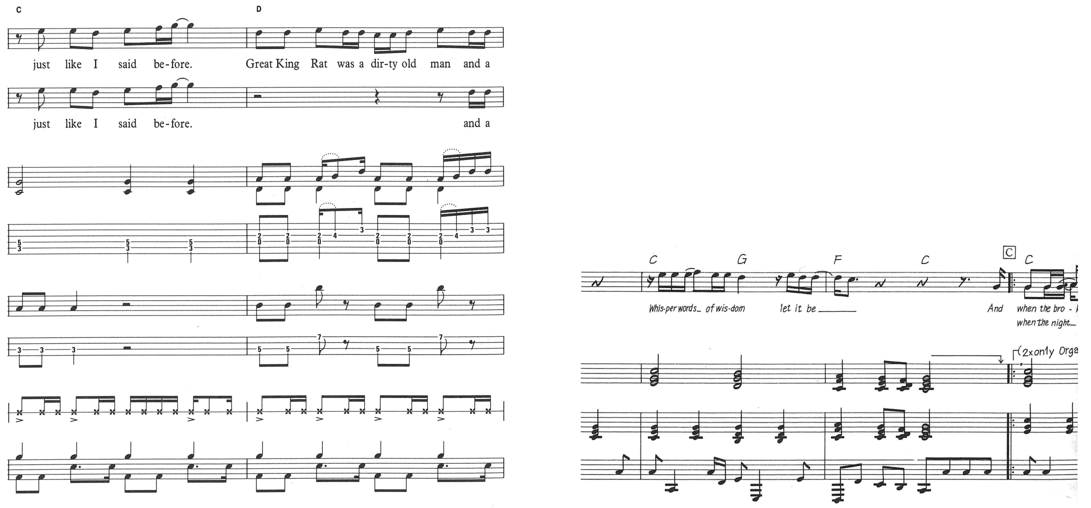
### 2.1.2 Chords and Simultaneities

The Harvard Dictionary of Music (Randel, 2003) defines *chord* as follows:

> *Chord.* Three or more pitches sounded simultaneously or functioning as if sounded simultaneously. [...]

Like many definitions of real-world concepts, this one is rather vague and allows many interpretations. The last part of the definition raises the next question: in a piece of music, what pitches are "functioning as if sounded simultaneously"? It is a well-established fact that the notes of a chord do not necessarily have to be played simultaneously since human beings are able to perform successive interval abstraction (Deutsch, 1969), which allows them to perceptually integrate sequentially played notes, and hence to perceive them as intervals and chords. Ulrich (1977) remarks: "[...] some chords are played with notes omitted. In such cases the sense of

[2] Deutsch and Boulanger (1984) discuss situations in which octave equivalence does not hold. The ability to distinguish between pitch class and pitch height does not exclude the ability to take pitch height into account.


C
D
just like I said be-fore.
Great King Rat was a dir-ty old man and a
just like I said be-fore.
and a
C G F C
C C
Whis-per words of wis-dom let it be
And when the bro-
when the night
2x only Orga

(a) "Great King Rat" (Mercury) taken from *Queen: off the record*, EMI Music Publishing / International Music Publications, 1988.

(b) "Let It Be" (Lennon/McCartney) taken from *The Beatles – Complete Scores*, Hal Leonard Publishing Corporation, 1993.

Figure 2.1: Excerpts from pop music scores. In Figure 2.1a, the thirds of the chords notated are not always sounded: on the last beat of the first bar, no E is present, and over the whole of the second bar, no F♯ is present. Nevertheless the chord annotator has reasonably transcribed the full chords `C` and `D` chords over the staff, using the context knowledge that the segment is in the key of G major. In Figure 2.1b, non-harmony notes occur: the first of the two beats annotated as `F` does indeed contain only notes belonging to that chord (F, A and C). The following two quavers contain non-harmony notes. They could be interpreted as `C/3` and `Dmin7`, respectively.

harmonic history allows the listener or musician to fill in the chord with his ear.". One of many examples is provided in Figure 2.1a. It is informative to compare the definition of *chord* to the definition of a different concept, that of *simultaneity*, taken from the same dictionary:

> *Simultaneity.* Any two or more pitches sounded simultaneously.

This definition is different from the definition of chord in two ways: the minimum number of notes is reduced to two, and—more importantly—strictly requires that the pitches should be sounded simultaneously. What is meant by the word *chord* is usually clear from the context, so the distinction between chords and simultaneities is often treated sloppily when talking about music. However, it has implications when implementing a software system with the goal of transcribing chords rather than simultaneities: chords are not necessarily represented in the low-level content of a signal. This fundamentally distinguishes chord transcription from polyphonic pitch transcription, where we assume that a transcribed note always has a physical counterpart in the audio waveform[3]. This means that a procedure is needed that possesses the human capability

[3] A similar problem occurs in beat-tracking, where beat-times are perceived (like chords) without a clear physical correspondence in the audio data (Davies, 2007).

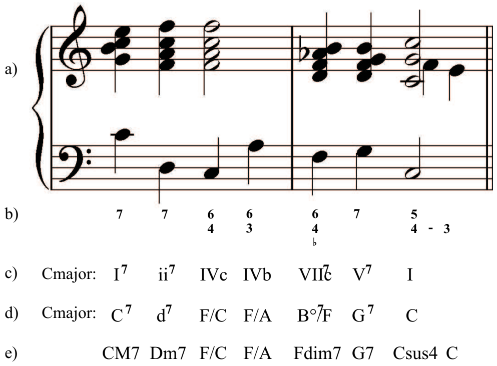

a)
b)
7 7 6 4 6 3 6 4 ♭ 7 5 4 - 3
c) Cmajor: I7 ii7 IVc IVb VII7c V7 I
d) Cmajor: C7 d7 F/C F/A B°7/F G7 C
e) CM7 Dm7 F/C F/A Fdim7 G7 Csus4 C

Figure 2.2: Chord syntax (the image and the following caption text are an excerpt from Harte et al., 2005): a short extract of music with different harmony notations: a) Musical score, b) Figured bass, c) Classical Roman numeral, d) Classical letter, e) Typical Popular music guitar style.

of mentally completing or integrating chords. On the other hand, there may be tones present in the music that are not considered part of the harmony, as is exemplified in Figure 2.1b. They are called *non-harmony notes* (Butterworth, 1999) (also: *embellishing tones* (Randel, 2003, p. 217)). These notes are not recorded in the chord label despite their presence in the audio waveform, as we will see in the following paragraphs.

### 2.1.3 Syntax

There is great variety in the syntax of chord labels. Harte et al. (2005) review different ways of labelling chords in Classical music, jazz and popular music. Examples are given in Figure 2.2. What is common to all forms of chord syntax is that they describe a chord as a chord quality and a bass pitch that indicates which note of the chord is the lowest. An informative transcription needs to include a description of the tonal content (as pitch classes, or degrees from a root note), and the bass note.

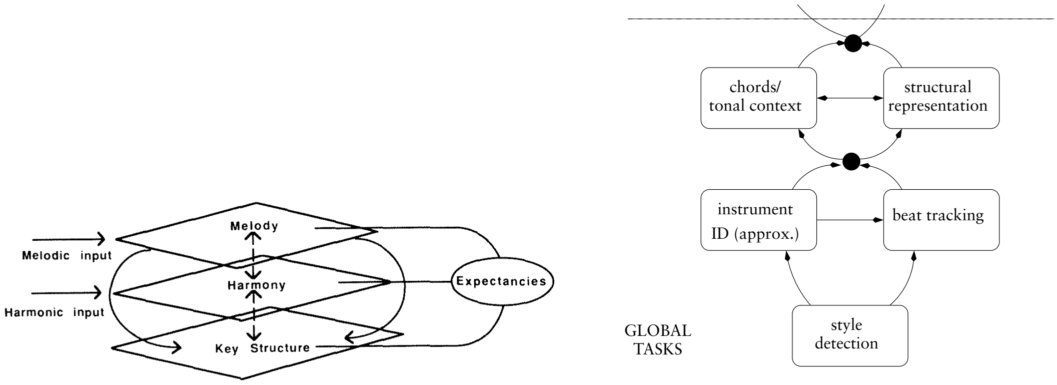
Chord labels represent an abstraction from the sounded music in the two senses we have already considered in Section 2.1.1 and Section 2.1.2: octave information is discarded, and over time, notes are integrated as part of the chord or discarded as non-harmony notes. Note that the bass pitch class, too, is an abstract notion, and, going back to the example in Figure 2.1b, we can see that although the chord `F` implies a bass note of F, other bass notes can occur in the realisation. We will call the bass note implied by the chord the *nominal bass note*.

Harte et al. (2005) devise a *chord syntax* based on that used in jazz and pop music, but with unambiguous definitions, and in a fully textual format that can be easily read by human beings and computers alike. For example, `D:min` is a D minor chord, `F/3` is an F major chord in first inversion, i.e. with an A in the bass (represented as "`/3`": major third). This syntax has been widely used in the MIR community, and it has the convenient property of describing the nominal bass note relative to the chord root, which we will need in later chapters. We use the syntax indicated by typewriter font, and without the colon separating the root note from the chord type shorthand, i.e. `D:min` becomes `Dmin`.

### 2.1.4 Musical Context

In some forms of chord labelling syntax, notably the Roman numeral notation (Figure 2.2) usually used in the analysis of Classical music (Butterworth, 1999), the function of the chord with respect to the current key is explicitly notated. This is not the case in jazz or pop notation, since the chord symbols are intended for live use by musicians: here, for faster realisation, the chord itself is notated explicitly and without an interpretation in terms of the key. This does not mean the relation to the key is lost: in their textbook on jazz harmony theory Wyatt and Schroeder (1998, p. 65) write that "it is necessary [for musicians] to be able to tell what key a [chord] progression belongs to by means of the chords alone". Furthermore, the functional interpretation of the chord in terms of the key may not be clearly defined in jazz and popular music, where chord combinations are handled more freely than in the Baroque and Classical periods, and modulations (key changes) may be only temporary or ambiguous.

Research in music perception has shown that the close link between chords and key in music theory and practice has parallels in human perception and cognition of chords. Harmonic priming studies show (for a review, see Patel, 2003) that human perception of chords is quicker and more accurate if they are harmonically close to their context. Furthermore Thompson (1993) shows that chords are perceived together with key and melody, in a partial hierarchy, in which the three qualities are linked by expectation, see Figure 2.3a. Hainsworth (2003, Chapter 4) conducted a survey among human music transcription experts, and found that these use several musical context parameters during the transcription process. Based on his findings, Hainsworth describes a scheme for automated polyphonic transcription, part of which is depicted in Figure 2.3b: not only is a prior rough chord detection the basis for accurate note transcription, but the chord transcription itself depends on the tonal context and other parameters such as beats, instrumentation and structure. Music theory, perceptual studies, and musicians themselves agree that generally no musical quality can be treated separately from the others.



(a) Music perception: partial hierarchy of melody, harmony, and key, taken from (Thompson, 1993).

(b) Part of a note transcription scheme, taken from (Hainsworth, 2003).

Figure 2.3: Chords in a musical context. Figure 2.3a shows the partial hierarchy which Thompson (1993) derived from perceptual studies. Figure 2.3b shows a part of the human transcription model proposed by Hainsworth (2003) based on a survey among musical experts.

### 2.1.5 Sheet Music and Lead Sheets

Musicians who play music of the *common practice period*, i.e. the Baroque, Classical, and Romanic styles (roughly from the late 1600s to the late 1900s), do not usually play from chord labels because every note is explicitly notated in sheet music, and chords only emerge when they are heard (or read) together. Chord labels are used predominantly as an analysis tool.

An exception is the tradition of *thorough bass* (and its notation, *figured bass*, see Figure 2.2), where a musician is provided with a bass note and a chord label consisting of degree numbers (figures) relative to the bass note. The musician, usually a keyboardist, then improvises the realisation of the music in notes (Randel, 2003, p. 890). The realisation of chord labels in jazz and modern popular music is similar. Often a simple representation called the *lead sheet* (e.g. Rawlins and Bahha, 2005) is used instead of explicitly written out sheet music. A lead sheet typically contains only one staff, on which the melody is notated, complete with lyrics, time signature and key signature (and possibly markups that define the style or feel). Apart from that, the only playing instructions are the chord symbols over the staves and the nominal bass note for the chord (if different from the root note). The players then usually improvise the actual realisation, or play from memory what was arranged during rehearsals. In short, lead sheets describe the song without dictating the arrangement. They are also used by songwriters when submitting their songs for consideration by publishers. The hit songwriter Hirschhorn (2004) recommends "sending a lyric sheet [...], a recording of the song [...], and a lead sheet (handwritten sheet music with melody, chords, and lyric), whenever you submit a song". Lead sheets have also arrived in the computer world mainly through the commercial software "Band in a Box"[4], which automatically generates an accompaniment based on the lead sheet information provided by the user. It is also common, especially among hobby musicians, to play from a text-based format, which contains only the lyrics with chords written above the word they occur closest to, like in this excerpt of the song "Your Latest Trick" (Knopfler):

```
           E
You played robbery with insolence

             F#m                 B         A/C#    B/D#
And I played the blues in twelve bars down Lover's Lane
```

This is the usual way of exchanging chords on the internet, but, according to Hirschhorn (2004), a comparable format is also used by studio musicians: "Some musicians want full lead sheets. The majority of musicians are content with just chords."

We have seen that in the context of chords, numerous other musical and perceptual concepts play a role. We have argued that chord labels are an abstraction of the sounded music, and we have described some formats in which chord labels are communicated by musicians. The next section will show how the complicated concepts surrounding musical chords have been expressed in computational models.

## 2.2 Related Work in Automatic Chord Analysis

In this section we summarise previous work in the field of automatic chord extraction, with some excursions to key extraction methods. We list relevant work ordered by year of publication in Table 2.1 (1999 to 2006), Table 2.2 (2007 to 2008) and Table 2.3 (2008, continued, to 2009). The increasing number of research papers concerned with this subject reflects the popularity of the task and the increase in computational power of modern computers.

Chord extraction methods can differ in many ways. We break down the description of design choices into three different areas, each of which a subsection in this section is dedicated to: Section 2.2.1 deals with chord models, describing different approaches to that part of a chord extraction algorithm which is in direct contact with the audio or symbolic front end. Section 2.2.2 examines different extraction and enhancement techniques for the chromagram, the data model used in the majority of audio chord extraction tasks. Finally, in Section 2.2.3, we turn to higher-level models that have been used to integrate the low-level content—as recognised through the chord model—in order to produce chord transcriptions.

[4] http://www.pgmusic.com

| Year | Paper | Aim | Time-frequency Transform | Window, Frame Size, Hop Size, Sample Frequency | Low-level Processing | Feature Vector | Profile Matching | High-level Model | Output Classes | Evaluation |
|---|---|---|---|---|---|---|---|---|---|---|
| 1999 | Fujishima (1999) | audio chord detection | FFT | ○ | PCP smoothing | PCP | inner product | “Chord Change Sensing” | 324 chords | 27 chords, 1 excerpt |
| 2001 | Su and Jeng (2001) | audio chord | Wavelet | n/a | – | four octaves salience | neural net weights | none | 48 chords | 8 bars |
| 2002 | Pardo (2002) | symbolic chord | n/a | n/a | n/a | PC set | rules | relaxation search | 72 chords | ○ |
| | Pickens and Crawford (2002) | retrieval based on symb. chords | n/a | n/a | n/a | PC set | context score | Markov model (not for chord estimation) | 24 chords | 3000 MIDI pieces |
| 2003 | Raphael and Stoddard (2003) | symbolic chord/key | n/a | n/a | n/a | PC set | learned chord/key prob. | HMM + Viterbi | diatonic chords | examples |
| | Sheh and Ellis (2003) | audio chord | DFT | Hann, 4096, 100 ms, 11025 | n/a | PCP (bin map) | Gaussian | HMM + Viterbi | 84 chords | 2 songs |
| 2004 | Maddage et al. (2004) | audio segmentation, chord, key | ○ | ○ | ○ | PCP | 3-state Gaussian (learned) | HMM + Viterbi + key/beat rules | 48 chords | 10 songs |
| | Yoshioka et al. (2004) | audio chord, key | DFT | Hann, 4096, 1280, 16000 (Goto, 2003) | 8th note quantisation | PCP + bass notes | Mahalanobis dist. | prob. hypothesis search | 48 chords | 7 songs |
| 2005 | Bello and Pickens (2005) | audio chord | Constant Q | ○, 8192, 1024, 11025 | beat-quantisation | PCP (const. Q) | Gaussian | HMM + Viterbi | 24 chords | 28 songs |
| | Cabral et al. (2005) | audio chord | automated | automated | automated | automated | automated | n/a | up to 60 chords | sound samples |
| | Shenoy and Wang (2005) | audio beat/chord/key | ○ | ○ | ○ | ○ | ○ | rules based on key and metric position | 24 chords | 30 songs |
| | Harte and Sandler (2005) | audio chord | Constant Q | ○, 8192, 1024, 11025 | PCP smoothing, tuning | PCP (const. Q) | inner product | median filter on labels | 48 chords | 28 songs |
| | Paiement et al. (2005) | symbolic chord | n/a | n/a | n/a | PCP (calc. from symb.) | function of euclid. dist | static graphical model | many | 52 songs |

Table 2.1: List of work related to chord transcription I: 1999–2006. The character ○ indicates that the information was not disclosed in the respective document. For discussion of the methods, see Section 2.2.

| Year | Paper | Aim | Time-frequency Transform | Window, Frame Size, Hop Size, Sample Frequency | Low-level Processing | Feature Vector | Profile Matching | High-level Model | Output Classes | Evaluation |
|---|---|---|---|---|---|---|---|---|---|---|
| 2006 | Gomez (2006) | audio key | FFT | Blackman-Harris, 4096, 512, 44100 | transient rem., peak picking, tuning, harmonic weighting | PCP (sp. peak map) | correlation | n/a | 24 keys | 1000+ pieces |
| | Peeters (2006) | audio key | FFT | Blackman, 4096, 2048, 11025 | spectr. preproc., tun., med. smooth. | PCP (filter bank) | GMM (learned) | HMM + Viterbi | 24 keys | 302 pieces |
| 2007 | Catteau et al. (2007) | audio chord, key | FFT + log freq. | Hamming, 2048, ○, ○ | silence det., event segmentation | PCP (filter bank) | customized Gaussian | HMM-like + dyn. programming | 48 chords, 24 keys | 10 songs |
| | Burgoyne et al. (2007) | audio chord | DFT | ○, 2048, 1024, 11025 | – | PCP (sp. peak map) | several trained | HMM, CRF | 48 chords | 20 songs |
| | Papadopoulos and Peeters (2007) | audio chord | FFT | Blackman, ○, ○, 11025 | tuning, median smoothing | PCP (bin map) | several theor. and learned | HMM | 48 chords | 110 songs |
| | Rhodes et al. (2007) | symbolic chord | n/a | n/a | n/a | pitch class set | two-stage Dirichlet | Bayesian model selection | 72 chords | 16 songs |
| | Zenz and Rauber (2007) | audio chord/key | Enh. Autocor. | 1024, 512, 22050 | | PCP | “linear distance” | chord change penalty, non-key chord removal | 36 chords | 35 songs |
| 2008 | Sumi et al. (2008) | audio chord/key | ○ | ○ | beat-sync. | PCP | GMM | prob. hypothesis search | 48 chords | 150 songs |
| | Zhang and Gerhard (2008) | audio guitar chord | DFT | 22050, ○, 44100 | ○ | PCP | neural net + voicing constr. | ○/Viterbi | 21 chords | 5 excerpts |
| | Lee and Slaney (2008) | audio chord, key | DFT | ○, 8192, 2048, 11025 | ○ | tonal centroid | GMM (learned) | HMM + Viterbi | 24/36 chords | 30 pieces |
| | Papadopoulos and Peeters (2008) | audio chord, downbeat | DFT + log freq. | Blackman, ○, ○, 11025 | tuning, median smoothing, beat-sync. | PCP (bin map) | correlation | beat/chord HMM + Viterbi | 48 chords | 66 songs |

Table 2.2: List of work related to chord transcription II: 2007 to 2008 (continued in Table 2.3). The character ○ indicates that the information was not disclosed in the respective document. For discussion of the methods, see Section 2.2.

| Year | Paper | Aim | Time-frequency Transform | Window, Frame Size, Hop Size, Sample Frequency | Low-level Processing | Feature Vector | Profile Matching | High-level Model | Output Classes | Evaluation |
|---|---|---|---|---|---|---|---|---|---|---|
| 2008 cont'd | Ryynänen and Klapuri (2008) | audio chord | DFT | Hamming (zero pd.), ○, ○, 44100 | Klapuri salilence fctn. | low and high PCP | inner product | HMM + Viterbi | 24 chords | 110 songs |
| | Scholz and Ramalho (2008) | symbolic chord | n/a | n/a | n/a | partitioned MIDI | utility function | context rules | ○ (many) | 4 pieces |
| | Varewyck et al. (2008) | audio chord similarity | DFT | ○, 8192 (zero-padded), ○, 8000 | peak-picking, pitch-candidate elimination | PCP | cosine similarity | n/a | chroma profiles | 161 30s-excerpts |
| | Weil and Durrieu (2008) | audio chord | Constant Q | ○, ○, 1024, 11025 | melody subtraction | PCP/TC | Gaussian | HMM + Viterbi | 24 chords | 176 songs (MIREX) |
| 2009 | Scholz et al. (2009) | chord language n-gram | n/a | n/a | n/a | n/a | n/a | n-Gram models | 24 chords; 204 chords | 180 songs |
| | Band in a Box 2009 | audio chord | ○ | ○ | ○ | ○ | ○ | ○ | chords, inversions | n/a |
| | Oudre et al. (2009) | audio chord | Constant Q | 8192, 1024, 11025 | tuning, beat-quantisation | PCP (const. Q) | KL divergence | low-pass/median filter | 36 chords | 180 songs |
| | Khadkevich and Omologo (2009b) | audio chord | DFT | Hamming, 2048. 1024, 11025 | tuning, PCP median smoothing | PCP (bin map) | GMM (learned) | HMM, FLM | 24 triads | 180 songs |
| | Noland (2009) | audio key | Constant Q | Hamming, 4096, 512, 2756 | tuning | chord labels (PCP) | chord probability | HMM + Viterbi | 24 keys | 353 pieces |
| | Weil et al. (2009) | audio chord, downbeat | Constant Q | ○, 512, 4096, 11025 | n/a | PCP (const. Q) | Gaussian (learned) | HMM + Viterbi | 24 chords | 278 synth. songs |
| | Weller et al. (2009) | audio chord | ○ | ○ | ○ | PCP | SVMstruct | SVMstruct | 24 chords | 180 songs |
| | Reed et al. (2009) | audio chord | Constant Q | ○, 1024, 2048, 11025 | percussive sound removal, tuning, DFT on chroma | PCP (const. Q) | Gaussian (discr. learning) | HMM | 24 chords | 176 songs (MIREX) |

Table 2.3: List of work related to chord transcription III: 2008 (continued) to 2009. The character ○ indicates that the information was not disclosed in the respective document. For discussion of the methods, see Section 2.2.

### 2.2.1 Chord Models

Chord models describe a chord label in terms of a low-level harmonic descriptors. In the symbolic domain, chord models can work directly on discrete note events, whereas methods working from musical audio work on usually continuous low-level features extracted from audio, a distinction which has led to considerable differences between chord analysis approaches. In the following paragraphs we will review them separately.

#### Symbolic Domain

In his early symbolic work on chord, key and chord function detection Ulrich (1977) represents a chord as a list of semitone distances from the root. For example, a `Dmin` chord in his representation would then be written as `D(3 7)`, encoding the note F as 3, since it is three semitones above the root D, etc. This representation enables him to systematically investigate to what extent chords match notes in a score. It also allows him to represent so called *extended* jazz chords that include notes at distances from the root that are larger than one octave; however, in the examples presented Ulrich confines himself to chords contained in one octave.

Academic approaches to automatic chord estimation mostly assume octave equivalence at least implicitly, i.e. they are based exclusively on pitch class, not pitch height. The rule-based approach to symbolic harmony analysis taken by Temperley and Sleator (1999) has a root model rather than a chord model. Scores for roots are calculated according to a *compatibility rule*, depending on the harmonic relationships of the present notes to the root. The presence of the notes D and A, for example, would generate a high score for a root of `D` because both notes have high (manually tuned) weights in the `D` root model. In a similar way, Sapp (2007) defines scores for roots of chords by their distance in stacked thirds. Though in these two models no chords are directly estimated, they recognise the special role of the chord root.

Pardo (2002) represents chords in "modulo-12 arithmetic" (i.e. pitch classes, not pitches). The pitch class C is represented as 0, and the `Dmin` chord is then represented as `<2 5 9>` (for D, F, A). Similarly, an `Amin` chord would be represented as `<9 0 4>` (for A, C, E). A score for a chord template over a segment is then determined by the number of coinciding notes minus the number of notes that are false positives or false negatives, i.e. only in either score or chord template. In the case of equal scores between chord templates, a tie break rule based on the chord root is applied. In a symbolic chord finding method for Bossa Nova guitar music, Scholz and Ramalho (2008) use a variant of the algorithm proposed by Pardo with more complex jazz chords. It is not clear however how an extended note such as an 11$^{\text{th}}$ would be represented. We assume that they are also mapped to the modulo-12 arithmetic.

Raphael and Stoddard (2003) use a *hidden Markov model* (HMM) to infer chord and key sequences from symbolic MIDI data (key models will be discussed in Section 2.2.3, page 42). In their chord model they distinguish five kinds of pitch classes: the root, the third, the fifth of the triad, then pitch classes that coincide with key scale notes, and finally the remaining pitch classes. The chord model—HMM emissions—then consists of learned probabilities for these five classes (further depending on the metric position). It is worth noting that in this model, a chord will have as many templates as there are keys. This corresponds very well to musical understanding: for example, in the key of C major, the note B may be just a passing note in a `Cmaj` chord, but the note B♭ would indicate that the chord is actually a `C7` chord. In the key of F major, similar situations could be interpreted differently. Raphael and Stoddard also introduce simultaneous modelling of the key through HMMs, which marks a great advance from other models that considered chords and key one after the other. However, since they consider only diatonic chords in a key, a key change is required whenever a chord occurs that is non-diatonic in the present key (more on key models in Section 2.2.3). A similar chord model is used by Rhodes et al. (2007). It factorises the probability of notes given a triad pattern in terms of the relative number of chord notes and the proportion between the three triad notes, both expressed as Dirichlet distributions.

An interesting approach is taken by Paiement et al. (2005), who investigate a way of generating physically motivated substitution probabilities between chords to apply them in a graphical model of jazz chord sequences. They take symbolic chord voicings and generate from them an kind of imitation of MIDI profiles by calculating the strengths of the partials of a note according to a geometric envelope. The profiles are subsequently wrapped to one octave, discarding the pitch height dimension, such that every chord and chord voicing is finally characterised by a 12-dimensional vector of real numbers, not by the notes it contains. As we will see in the following paragraphs, defining chords by such schemata or patterns has been the method of choice for many audio chord models.

### Audio Domain

Chord models for symbolic music do not easily transfer to the audio domain because the musical notes cannot be directly inferred from an audio signal. Estimation of pitch and note onsets and offsets is still an active research field. Multiple pitch detection (see, e.g., Goto and Hayamizu, 1999; Klapuri, 2004) in particular is difficult because a simple frequency spectrum does not only exhibit high energy at the fundamental frequency of a note (i.e. the pitch), but also at related frequencies, upper partials. Additionally, broadband noise generated at instrument transients or drum and cymbal sounds can generate high energy at frequencies that bear no relationship with the pitches played. As a result, transcription methods usually work reliably only in a certain fields of application, like monophonic pitch estimation (for an overview, see Poliner et al., 2007), although quantization to note events is still an unsolved problem. By introducing schemata for tonal centres, Leman (1994) circumvents the detection of notes and directly estimates the tonal centre from the output of an auditory model. Every schema—a tonal centre—has a profile which is directly matched to the continuous data. Fujishima (1999) uses the same kind of paradigm for chords, resulting in the first research paper explicitly concerned with chord transcription from audio. In his work, the schemata modelled are chords, and the features used are pitch class profiles (PCP), also called chroma vectors (Wakefield, 1999), which are produced by wrapping spectral content into one octave. The PCP is then used as a 12-dimensional chromatic representation of the harmonic content of an audio signal. This representation may appear similar to Parncutt's theoretical chroma salience model (page 23), but the PCP represents the physical signal and not the salience of pitch classes in a perceptual (potentially more musical) sense.

The implementation of the feature extraction as used by Fujishima (1999) is straightforward: first, the spectrum $X_k$ of an audio signal $[x]_n$ is computed using the *discrete Fourier transform* (DFT)

$$X_k = \sum_{n=0}^{N_F-1} x_n w_n e^{-\frac{2\pi i}{N_F}kn}, \quad k = 0, \ldots, N_F - 1, \tag{2.2}$$

where $N_F$ is the number of samples in one frame and $w_n$ is a window function that weights the samples to reduce the spectral leakage associated with finite observation intervals (Harris, 1978). The pitch classes from C to B are assigned to the numbers $p = 0, \ldots, 11$. Then, the PCP value for $p$ is computed as the sum of all the power spectrum coefficients closest to an instance of that pitch class,

$$\mathrm{PCP}_p = \sum_{M(m)=p} ||X_m||^2, \tag{2.3}$$

where

$$M(m) = \text{round}\left(12 \log_2 \left(\frac{f_s}{f_{\text{ref}}} \cdot \frac{m}{N_F}\right)\right) \bmod 12,$$

with $f_{\text{ref}}$ being the reference frequency of pitch class $p = 0$, and $f_s$ being the sample frequency. Usually, the spectrum is calculated on overlapping frames over the duration of a piece of music, in a process called *short-time Fourier transform* (STFT). The resulting matrix $(X_{k,m})$, in which the DFT of the $m^{\text{th}}$ frame occupies the $m^{\text{th}}$ column, is called the spectrogram. Much like a spectrogram describes the spectral content of a signal over time, the chromagram matrix $(\text{PCP}_{p,m})$, in which the chroma vector of the $m^{\text{th}}$ frame occupies the $m^{\text{th}}$ column, describes the chroma content over time (Wakefield, 1999). In order to allow for energy differences between frames, every individual chroma vector in the chromagram is normalised. We discuss different ways of generating the chromagram in Section 2.2.2 on page 38.

In order to perform chord recognition on the chromagram Fujishima takes the inner product between predefined 12-dimensional chord patterns and the PCP derived from audio. At every time frame, the chord pattern that returns the highest value is chosen. Fujishima (1999) chooses and test 27 different isolated chord types. For real-world music the chord set is reduced to "triadic harmonic events, and to some extent more complex chords such as sevenths and ninths". Unfortunately, only one short excerpt demonstrates the system's capability of recognising chords from real-world music signals.

We can observe then, that ignoring the upper partials by using theoretical chord templates that only contain the chord notes can result in a working system. Harte and Sandler (2005) too use pitch class templates of a chord and an inner product with chroma vectors as an indicator of the prominence of that chord, but with a reduced set of 48 chords, covering the `maj`, `min`, `dim`, and `aug` triads. It is possible to calculate the fit of such binary chord templates in different ways: Bello and Pickens (2005) model a chord as a 12-dimensional Gaussian distribution, in which they set the means of the chord notes to 1, all others to 0, and furthermore hand-tune a covariance matrix according to an assumed high covariance between chord notes. Due to normalisation, all chromagram values are in the interval $[0, 1]$, and—as in one of our own models (Figure 4.9)—the Gaussian distribution is used as an auxiliary approximation to a heuristic chord score function. Similarly, Catteau et al. (2007) use theoretical chord profiles to build a customized probability distribution with the help of Gaussians, see Figure 2.4. In order to obtain more realistic chord profiles Papadopoulos and Peeters (2007) modify the binary chroma mask to allow for the energy of the upper partials (similar to Paiement et al., 2005, as described on page 33). As a measure of fit between chroma vectors and their 24 `maj` and `min` profiles, Papadopoulos and Peeters use 12-dimensional Gaussians as well as the correlation coefficient (as previously used by Gomez (2006) for key extraction) and find that in their application the

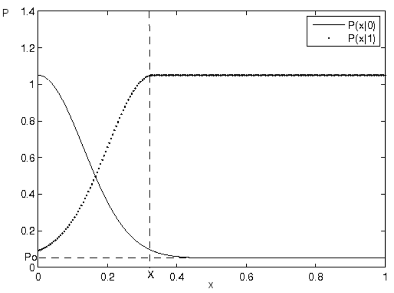


Figure 2.4: Chord model proposed by Catteau et al. (2007): the two distributions used to discriminate between notes appearing or not in a certain chord.

correlation coefficient works best. Oudre et al. (2009) have recently shown that the Kullback-Leibler divergence is a good measure of similarity between chroma vectors and chord profiles of several chord types that are enhanced to take account of harmonics, including not only `maj`, `min`, `dim`, and `aug`, but also dominant chords (`7`) as an extra chord type.

These theoretically-motivated template models have the advantage of being data-independent. From a machine-learning point of view however they may be unsatisfactory despite their good performance in the research described above because they rely on music-theoretical considerations rather than on the physical properties of features such as the chroma vector.

A slightly different, data driven approach is taken by Sheh and Ellis (2003). They too assume that the chroma vectors contain enough information to recognise chords and use multidimensional Gaussians to model chords as chroma templates. However, these templates are now learned from data. Since at the time labelled chord data was not available, they resorted to a half-supervised learning procedure: a hidden Markov model is fed with the unaligned chord progression for a song as well as chromagram data. Then, using the expectation-maximisation algorithm, both chord change times and chord profiles are estimated. Assuming that chord profiles of the same chord type should be identical up to circular permutation in the modulo 12 arithmetic, the chord means and covariance matrices of each chord type are weighted by the number of frames they are estimated from, appropriately rotated, and then added, resulting in only one Gaussian chord profile per chord type. With seven different chord types Sheh and Ellis consider a greater harmonic variety than most later approaches to chord extraction. Sheh and Ellis report recognition scores for two different kinds of experiments: chord alignment (chord sequence is given) and free chord recognition. While it is expected that results for alignment exceed those for recognition, the size of the difference is remarkable, for example 63.2 percentage points (83.3% and 20.1%). We suspect that assuming a single chord profile per chord overfits the data, such that simultaneities with non-harmony notes (as discussed in 2.1.2) are erroneously recognised as different chords.

Similar data-driven approaches have been taken up by other researchers, though the tendency has been to use supervised learning on fully labelled data, enabled by new, hand-labelled data (e.g. Harte et al., 2005). A single 12-dimensional Gaussian is easy to estimate and was used with considerable success in a variety of research papers (Papadopoulos and Peeters, 2007; Weil et al., 2009) not only for chord estimation, but also for key estimation (e.g. Peeters 2006). It is, however, clear that a single Gaussian is a bad match for normalised chromagram values (Burgoyne et al., 2007), and different distribution types have been used to work around this problem: mixture of Gaussians (Lee and Slaney, 2008; Mauch and Dixon, 2008; Khadkevich and Omologo, 2009a,b; Peeters, 2006; Maddage et al., 2004), Dirichlet distributions (Burgoyne et al., 2007), as well as neural networks (Su and Jeng, 2001; Zhang and Gerhard, 2008) and automatically generated feature models (Cabral et al., 2005). Lee and Slaney (2007) report improved performance using a Gaussian mixture model on a six-dimensional linear transform of the chromagram called tonal centroid (TC) (Harte et al., 2006). In all these cases, only relatively few chord classes are learned (mostly 24). It is therefore plausible that treating a chord as one perceptual schema and learning this schema as a chord profile has its limitations: chord types that are similar on average in their acoustic properties can easily be confused, and the surprisingly low performance of a model with many chord types (Sheh and Ellis, 2003), as discussed above, suggests that assuming a generative model with maximum likelihood learning tends to overfit the data.

In contrast, Reed et al. (2009) learn Gaussian chord profiles by using a form of discriminative training: the minimum classification error (MCE). This model still assumes that a chord has one profile, but the profiles in this model depend on each other, they form what could be called a “chord set model” as opposed to 24 separate chord models. Weller et al. (2009) use support vector machines (SVMs) for related discriminative training and achieve state-of-the-art results. It is also possible not to explicitly assume a particular distribution and—as is often done in the theoretically-motivated contexts mentioned above—calculate some distance between a learned chord profile and the data, e.g. the Mahalanobis distance (Yoshioka et al., 2004) (which is however similar to using a Gaussian model), or the correlation distance (Gomez, 2006).

A drawback of all chroma-based models that directly use the spectrum and map it onto one octave is that they assume the information that is lost by discarding the pitch height is not necessary to recognise a chord pattern. Upper partials can corrupt the chroma vector, and in Chapter 5 we find evidence suggesting that this assumption may indeed be problematic.

To conclude the description of chord models, we can observe that symbolic approaches differ greatly from audio-based approaches: while symbolic approaches tend to work on individual note events, audio approaches usually assume that a chord can be expressed by a single template, which is matched to the data in a song. This latter assumption works well in many circumstances, but is flawed from a musical perspective because a chord is a musical construct which develops its meaning over time, as we have argued in Section 2.1.2, and in conjunction with the current tonality. The approach taken by Raphael and Stoddard (2003) is the only one in which the chord model itself changes according to the current key. Finally, the difference in performance between trained models and theoretically-defined models is not clear-cut, and the development of better training methods for chords is an active research topic.

### 2.2.2 Calculation and Enhancement of Chromagrams

The frequency spectrum calculated by the Fourier transform has coefficients at equally-spaced frequencies. We have seen in Section 2.1.1 that pitch, however, is geometrically spaced with respect to frequency (Equation 2.1). Pitch is therefore linear in log-frequency, and in order to calculate any pitch or pitch class representation in the frequency domain, a conversion from frequency to log-frequency is required.

We discussed in Section 2.2.1 on page 34 how chromagrams can be calculated by summing the DFT frequency bins closest to each pitch class. In this approach, the conversion to log-frequency is handled in a discrete mapping directly from the DFT bins. Gomez (2006) uses a similar approach, but detects spectral peaks first and maps only those to the chroma vector, according to their estimated frequency position. An alternative strategy is to perform a log-frequency transform first and then assign the different pitch bins to the appropriate pitch class bin. The most prominent approach is a constant-$Q$ transform (Brown, 1991), often referred to as *the* constant-$Q$ transform. The number $Q = f/\delta f$ is the ratio of note frequency and note bandwidth. Given the desired number $n_{\text{bin}}$ of bins per octave,

$$Q = n_{\text{bin}}/\ln 2 \tag{2.4}$$

can be computed easily[5]. The calculation of the constant-$Q$ transform in the time domain in-

[5] This formula for $Q$ can be derived as follows: Let the frequency $x$ bins higher than a reference frequency $f$ be $g_f(x) = f\ 2^{x/n_{\text{bin}}}$, then the bandwidth $\delta f$ at $f$ is the derivative of $g_f$, evaluated at $x = 0$. Since this derivative is $g'_f(x) = \frac{f \ln 2}{n_{\text{bin}}}\, 2^{x/n_{\text{bin}}}$, the bandwidth is $\delta f = g'_f(0) = \frac{f \ln 2}{n_{\text{bin}}}$, and $Q = f/\delta f = n_{\text{bin}}/\ln 2$.

volves separate windowing for every constant-$Q$ bin, but equivalent windowing in the frequency domain can be implemented efficiently, and requires only a simple matrix multiplication of a kernel matrix with the DFT spectrum. Several key and chord extraction methods have used variants of the constant-$Q$ transform (see Tables 2.1, 2.2 and 2.3). To a similar effect, alternative kernels and filter banks with constant $Q$ (Peeters, 2006; Müller et al., 2009; Sagayama et al., 2004; Catteau et al., 2007) can be used to map the spectrum to the pitch domain. This may be desirable, since the original constant-$Q$ transform requires very long frame sizes for low notes, and the short window in the high frequency range can result in parts of the signal not being considered unless the hop-size is set to a very low value.

None of the methods just mentioned are natively immune to the presence of unwanted noise, the presence of unwanted harmonics of notes, and instrument tuning that differs from the expected pitch. We will review below efforts to find solutions and work-arounds for these problems after a brief consideration of signal processing parameters.

### Signal Processing Considerations

The choice of the DFT frame length $N_F$, and the window function $w$ (Equation 2.2) depend on the application they are used in. The choice of frame length $N_F$ in the DFT has effects in time and frequency resolution. Usually, the DFT is implemented as *fast Fourier transform* (FFT) (Cooley and Tukey, 1965), which dramatically speeds up implementation and only imposes a minor restriction, namely that the frame length $N_F$ be a power of two. Given a sample frequency $f_s$, the greater $N_F$ the better the frequency resolution; the smaller $N_F$ the better the time resolution. Resolving simultaneous sinusoids can become an issue in lower frequency bands, where DFT bins are wide with respect to pitch. For example, if the sampling frequency is $f_s = 11025$Hz and $N_F = 4096$, the distance between two DFT bins is 2.7Hz, more than the difference between the adjacent pitches E1 and F1 (MIDI notes 28 and 29). This is indeed a problem when we require to resolve two simultaneous bass frequencies, which are a semitone or less apart—as happens in the constant $Q$ transform. However, as we will explain in Chapter 3, this is generally not necessary in musical audio because such closely-spaced bass notes do not occur simultaneously. Then, if sinusoids are not very closely-spaced, each can be resolved, and the signal processing literature provides techniques such as quadratic interpolation or the phase vocoder (Zölzer, 2002, Chapters 8 and 10) to detect frequencies with much higher accuracy than the frequency difference between spectral bins. The ability to resolve sinusoids, however, depends on the window function.

Window functions are crucial to the Fourier analysis of finite signals. Harris (1978) compares 23 different families of window shapes according to their theoretical properties, highlighting that there is always a trade-off between *main-lobe* width (should be narrow for higher sinusoid resolution) and *side-lobe height* (should be low for noise robustness). That is why no general conclusion on the optimal window can be made without knowing properties of the data considered. Arguably due to the fact that music data are very heterogeneous, no single best practice windowing function has emerged, and music computing researchers use a range of windows. For example, Gomez and Herrera (2006) use the Blackman-Harris window. Khadkevich and Omologo (2009a) compare three different window shapes, and find that the Blackman window works best for them though the differences between different window types are very small. Papadopoulos and Peeters (2007) also use a Blackman window. Catteau et al. (2007) use a Hamming window, as do Ryynänen and Klapuri (2008). The Hamming window is also the default window in the MATLAB spectrogram function[6]. In many papers the window shape is not reported.

**Transient and Percussive Noise Reduction**

Given a chromagram, a simple way of reducing transient and percussive noise is to smooth the chromagram in the time direction using a finite impulse response lowpass filter (Harte and Sandler, 2005), or a median filter (Khadkevich and Omologo, 2009b). Median filters have also been used one stage earlier, before wrapping a pitch representation to the chroma (Peeters, 2006; Mauch and Dixon, 2008). Gomez (2006) removes frames that have been recognised as transient frames using the method proposed by Bonada (2000). Catteau et al. (2007) and Varewyck et al. (2008) subtract from a spectral representation of the signal the background spectrum, a smooth noise envelope calculated by median smoothing of the spectrum.

Reduction of noise is also a welcome side-effect of beat-synchronising chroma, since short noisy drum sounds and transients are averaged out over the period of one beat. In a chord transcription context Bello and Pickens (2005) are the first to use a beat-synchronous chromagram by averaging frames over a beat. Apart from providing a more principled smoothing (and hence relative reduction of transients and noise), this also results in a data representation in which each chroma "frame" has a musical meaning – it represents one beat. This is very useful when trying to do beat level analysis (e.g. Shenoy and Wang, 2005; Weil et al., 2009).

Reversing the point of view, noise reduction can also be performed by isolating harmonic

[6] `http://www.mathworks.com/access/helpdesk/help/toolbox/signal/spectrogram.html`

sounds. Proposed methods are spectral peak picking (Gomez, 2006) and peak picking in a fine-grained chroma vector (Harte and Sandler, 2005, see also tuning on page 59). Reed et al. (2009) preprocess the audio using harmonic-percussive separation (Ono et al., 2008). We will discuss below harmonic attenuation methods, which, like the pitch salience preprocessing step used by Ryynänen and Klapuri (2008), often include noise reduction at least as a side-effect.

### Harmonics Removal

One of the most difficult problems regarding music transcription in general is the presence of overtones in the sound of all natural instruments (Klapuri, 2006b). Overtones are sinusoidal components of a complex tone above the frequency of the fundamental frequency. Since most instruments have a harmonic set of overtones (i.e. the overtone frequencies are integer multiples of the fundamental frequency) they have been the focus of attention also for chord extraction methods. Though chroma profiles can to some extent be learned from chroma data that still contain overtones, it is desirable that the pitch classes corresponding to true fundamental frequencies dominate the chroma vector in the first place—as they would in the perceptually motivated concept of chroma salience (Parncutt, 1989, Chapter 4). To emphasise fundamental frequencies Gomez (2006) considers not only the spectral peaks at the pitches of the respective pitch class bin, but also at multiple frequencies—harmonics—thereof. The weight of the harmonics decreases exponentially: the $i^{\text{th}}$ harmonic contributes $s^{i-1}$ of its energy, where Gomez chooses $s = 0.6$. A similar approach is taken by Ryynänen and Klapuri (2008) using the pitch salience function developed by Klapuri (2006b) as a preprocessing step. Varewyck et al. (2008) propose a chromagram based on a prior hard transcription step: they detect note candidates by spectral peak picking and calculate for each a salience value based on sub-harmonic summation. Starting from the most salient, they successively eliminate note candidates that are in harmonic relationship to the current note. They show that the chromagrams obtained from this transcription outperform those calculated using several other chromagram calculation methods in terms of correlation with an annotated set of pitch classes.

### Tuning

Human pitch perception is relative to a tuning frequency. The tuning frequency is implicitly contained in the music, and all further tonal analyses are based on being able to decide which pitch a played note can be “quantised” to. The reference frequency often used in music is the A above middle C, whose frequency usually is in the range of 415 to 445 Hz, but most often 440 Hz. For reliable chord estimation, the computer has to be instructed how to determine this reference frequency. Harte and Sandler (2005) use a chromagram with 36 chroma bins, three per pitch class. They use the positions of the peaks in the chromagram to form a histogram over the width of one semitone. The position of the maximum of the histogram is then an estimator of the true tuning. Peeters (2006) defines a range of possible tunings and uses the one that best explains the frequency components found in a song. Dressler and Streich (2007) and Mauch and Dixon (2010) wrap frequency to the interval $[-\pi, \pi)$, where $\pi$ represents a quartertone, and interpret the tuning as an angle. Since the effect of upper partials is negligible for the rough tuning detection needed for harmonic analysis (Gomez, 2006, page 71) all the methods mentioned work well enough for key or chord estimation, and tuning detection can be regarded as a solved problem[7], if the true tuning is within a quartertone of the tuning used in most modern music, 440 Hz. It has been used by several authors of chord and key extraction work, e.g. Gomez (2006); Papadopoulos and Peeters (2007); Mauch and Dixon (2008); Papadopoulos and Peeters (2008); Noland (2009); Reed et al. (2009); Khadkevich and Omologo (2009a); Oudre et al. (2009). In many cases the tuning step can greatly increase the clarity of the data because notes can be clearly attributed to one pitch class.

### 2.2.3 Context Models

The variety of possible note combinations arising during a bar of music is large, even if the bar can be summarised by only one or two chords, as we have seen in Section 2.1. This makes the reverse procedure of estimating the chord behind the set of these played notes a difficult task, and other factors, i.e. musical context, have to be taken into account in order to arrive at a musical solution. This section reviews how musical context such as chord progressions, meter, melody, bass note, and tonality have been used to achieve more reliable automatic chord transcription.

#### Smoothing, Chord Transitions and HMMs

Since noise and embellishments disguise the chords in music, all chord detection methods include strategies to eliminate short term deviations, which otherwise would have resulted in false chord changes being detected. As we have argued in Section 2.1, it is necessary to imitate the horizontal abstraction humans are capable of. One of the most important facts about chords used in chord extraction algorithms is that chords are usually stable over a certain duration (several seconds).

Several segmentation algorithms have been suggested in the symbolic domain that include

[7] If viewed as a problem in its own right, the fine detection of tuning and temperament is much more difficult, but has been successfully implemented for the special case of harpsichord music (Tidhar et al., 2010).

the knowledge that chords “want” to remain stable. Pardo and Birmingham (2001) find the best partitioning for note sequences by first considering all possible transitions, assigning scores to them and then globally finding the best partition using Relaxation Search. In a similar setting, Temperley and Sleator (1999) employ Viterbi-like dynamic programming to obtain smooth chord root transitions. Rhodes et al. (2007) consider every bar of a piece separately and choose the optimal partition pattern according to Bayesian model selection.

In the audio domain, with no score information available, these approaches cannot be directly applied. Instead, every frame of (possibly beat-synchronous) audio is taken as one observation. Fujishima (1999) detects chord boundaries by introducing a procedure called “chord change sensing”, which determines the point of a chord change by “monitoring the direction” of the chroma vectors, which we assume is achieved by calculating the cosine distance. Harte and Sandler (2005) observe a performance increase when using the median filter on frame-wise chord labels to discard short term chord changes: the median filter uses the middle element of the sorted list of chord labels in a window around the current frame. If a chord label occurs in more than half the frames in the window, the median filter will always return it; it may find a different chord label, if the most frequent label in the window occurs in less than half the frames, since it is not guaranteed to occupy the middle of the list anymore. This suggests that the mode filter, which simply picks the most frequent chord label in a window, could be more robust. Oudre et al. (2009) also use median and low-pass filtering: not on the frame-wise output sequence, but on the frame-wise relative score of all chords.

The symbolic domain chord extraction algorithm by Raphael and Stoddard (2003) and—importantly—most audio chord transcription algorithms use probabilistic time-series models in order to obtain smooth chord sequence descriptions (Sheh and Ellis, 2003; Bello and Pickens, 2005; Peeters, 2006; Mauch and Dixon, 2008; Ryynänen and Klapuri, 2008; Weil and Durrieu, 2008; Khadkevich and Omologo, 2009b; Weil et al., 2009; Reed et al., 2009; Yoshioka et al., 2004; Catteau et al., 2007; Papadopoulos and Peeters, 2008; Burgoyne et al., 2007). The predominant model used is the hidden Markov model (HMM), which is known to be an effective tool for spoken language recognition (Manning and Schütze, 1999; Rabiner, 1989) because HMMs model contiguous, non-overlapping events over time. This is why they are an obvious choice for chord labelling, which follows the same paradigm. Since HMMs are not only important in previous approaches, but are also closely related to dynamic Bayesian networks, which we use in our own method (Chapter 4), we will now give a brief overview of the model.

The backbone of an HMM as depicted in Figure 2.5 is a Markov chain of discrete state

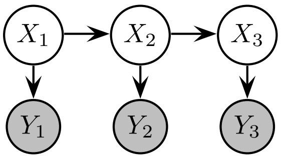


Figure 2.5: Representation of an HMM, taken from (Murphy, 2002). The hidden state variables $X_t$ usually represent an unknown sequence (e.g. chords), while the variables $Y_t$ represent the observed features (e.g. chroma vectors).

variables $X_t \in \{1, \ldots, K\}$, where $K$ is the number of states. In hidden Markov models for chord transcription, each state usually represents a chord. In this model, the state variables are subject to the Markov property,

$$P(X_t \,|\, X_{t-1}, X_{t-2}, \ldots) = P(X_t \,|\, X_{t-1}), \tag{2.5}$$

i.e. the current state directly depends only on the previous state. The specification of the right-hand side in (2.5) is the state transition model, which can be expressed in a stochastic transition matrix $(A_{ij})$, where $A_{ij} = P(X_t = j \,|\, X_{t-1} = i)$. The states (e.g. chords) are not observed directly. Instead we observe features (e.g. chroma vectors), which are modelled as random variables $Y_t$ and are assumed to depend exclusively on the current state, i.e. the observations' distribution is $P(Y_t \,|\, X_t)$. Once the initial state distribution $\pi_i = P(X_1 = i)$ has been specified, the model can be used to infer the most likely state sequence $x^*_{1:T}$ from the observations $y_{1:T}$, i.e.

$$x^*_{1:T} = \arg\max_{x_{1:T}} P(x_{1:T} \,|\, y_{1:T}), \tag{2.6}$$

using the Viterbi algorithm (Rabiner, 1989). The function of the HMM in chord recognition is hence (at least) twofold: the chord models (see 2.2.1) are expressed in the observation distributions, and the nature of chord changes is modelled in the transition matrix. For two given chords, the transition matrix describes the probability of the one following the other. Possibly the most important value is the self-transition value: it describes how likely it is that the chord does not change. The transition matrix can also describe musically relevant transition patterns. Bello and Pickens (2005) learn the transition matrix for every single song using the expectation-maximization algorithm. This works well for the simple song data-base they consider. Collection-level learning for more complex $n$-gram models is demonstrated by Scholz et al. (2009), though not applied to chord recognition. Lee and Slaney (2008) use key-dependent chord transition matrices learned from data. It is not clear however whether the observed improvements are caused by modelling the transition probabilities or whether modelling merely the probability of the chord given the key would have had a similar effect. Khadkevich and Omologo (2009b) find that including longer dependencies and explicit duration modelling can result in a slight increase in accuracy.

There are other strategies to model time series, including Conditional Random Fields (Burgoyne et al., 2007) or dynamic modelling strategies hand-tailored for the task of chord extraction (Yoshioka et al., 2004; Sumi et al., 2008; Catteau et al., 2007). A recent attempt to use the SVNstruct tool (Weller et al., 2009) achieved very good results in the Chord Detection Task of the Music Information Retrieval Evaluation eXchange (MIREX) (Downie et al., 2010). Interestingly, Weller et al. report that increasing the model order from 1 to higher orders does not improve results.

Research by Cemgil et al. (2006) and Leistikov (2006) introduced dynamic Bayesian networks (DBN) to music computing: probabilistic graphical models whose scope is the same as that of HMM. Although their algorithms are not directly concerned with the extraction of chords from audio, they stress the fact that DBNs can natively model higher-level musical qualities more intuitively and efficiently than an HMM. In the late-breaking session of the 2008 ISMIR conference Raczynski[8] presented a DBN for chord and key estimation from symbolic data. As we will see in Chapter 4, DBN can be used to simultaneously integrate many musical context qualities, which have been considered separately before. The following paragraphs give an overview of methods that have used musical context beyond chord transition patterns.

### Key Context

The importance of key for the analysis of chords in music theory and perception (Section 2.1.4) has led several researchers to incorporate key context into their methods (Raphael and Stoddard, 2003; Yoshioka et al., 2004; Maddage et al., 2004; Shenoy and Wang, 2005; Catteau et al., 2007; Lee and Slaney, 2007; Zenz and Rauber, 2007; Reinhard et al., 2008; Lee and Slaney, 2008; Sumi et al., 2008; Khadkevich and Omologo, 2009b). Though the implementations differ, what is common to all is that they use key information to correct chord detection errors. The underlying principle is that, when in a particular key, traditionally only the diatonic chords (i.e. those that can be formed from the notes in the scale of the key) are used, and other chords are less likely. We can distinguish two different approaches to key-aided chord transcription: an

[8] `ismir2008.ismir.net/latebreak/raczynski.pdf`

iterative approach, in which the chord labels are refined in a post-processing step after a previous key recognition step (Maddage et al., 2004; Shenoy and Wang, 2005; Zenz and Rauber, 2007; Reinhard et al., 2008; Khadkevich and Omologo, 2009b), and a simultaneous approach, in which chords and keys are modelled simultaneously in the main recognition step (Raphael, 2005; Catteau et al., 2007). To some degree, the algorithms proposed by Lee and Slaney (2008), Yoshioka et al. (2004) and Sumi et al. (2008) also belong to the latter group, but they assume that the key remains constant throughout a piece, so that a true interaction between key and chord estimation cannot be established. The same is true for (Shenoy and Wang, 2005). Yoshioka et al. argue: "The main difficulty in automatic chord transcription lies in the following mutual dependency of three processes that constitute automatic chord transcription: chord-boundary detection, chord-symbol identification, and key identification. Because of the mutual dependency, these processes are inseparable." In this sense, only the simultaneous approaches can adequately model chord sequences and key, and of those, only the models of Raphael (2005) and Catteau et al. (2007) allow for modulations, i.e. key changes. Both use a probabilistic framework. In the symbolic approach proposed by Raphael and Stoddard (2003), both the observation (pitch class) probabilities and the chord transition probabilities (and hence the chord probabilities) depend on the key, and are learned in an unsupervised way from MIDI data (with some hand-tying of states). In the audio-based approach of Catteau et al. (2007), only the chord change probabilities depend on the current key. Both key and chord transition probabilities are based on the chord distance proposed by Lerdahl (2001), a mixture of pitch-class overlap and root distance.

### Bass Context

A few approaches to chord transcription acknowledge the special role of the bass frequency region (Yoshioka et al., 2004; Sumi et al., 2008; Mauch and Dixon, 2008; Ryynänen and Klapuri, 2008): Ryynänen and Klapuri (2008) use a chroma vector of length 24, in which the first twelve elements represent the chroma over low frequencies (MIDI notes 26 to 49), and the remaining twelve elements represent the high frequencies (MIDI pitches 50 to 73). When trained, a chord profile will naturally include information about the bass note, and about harmonics, which are more likely to be present in the higher frequency range. The remaining approaches employ bass note information more explicitly, by first estimating a bass pitch or pitch class and then modifying the score for every chord according to the bass pitch. This was done by either weighting chord notes differently from non-chord notes (Yoshioka et al., 2004; Mauch and Dixon, 2008) or by applying a learned bass pitch probability (Sumi et al., 2008). Unlike dedicated bass transcription approaches (Ryynänen, 2008), none of the approaches mentioned take into account the temporal development of bass lines. Furthermore, they have not been used to exploit the possibility of discerning different chord inversions, for which the recognition of the bass note is the prerequisite.

#### Metric Position Context

The fact that the likelihood of a chord change is higher at certain metric positions has been used both to improve beat/bar detection (Goto, 2001) and to improve chord transcription (Shenoy and Wang, 2005; Maddage et al., 2004; Papadopoulos and Peeters, 2008; Weil et al., 2009). These algorithms rely on good prior beat detection, and in some cases have used manual beat annotations. The approach taken by Papadopoulos and Peeters (2008) stands out because the downbeat (i.e. the "phase" of the bar beginnings relative to the beat) of the rhythm is estimated at the same time as the chord, and chords and downbeats are treated in a very musical way. The other approaches use separate downbeat estimation (sometimes based on the preliminary chord estimation) and then use the newly found rhythm estimates to post-process the preliminary chord sequence or rerun the algorithm including the new rhythm knowledge. None of the algorithms incorporate a way of dealing with deleted or inserted beats, which restricts their usage to simple songs in which no beats are omitted on the composer's side, and in which the beat-tracker performance is perfect.

Generally, we observe a trend towards more musically-aware models, which seems the only way to imitate the human cognition of harmony. Several ways of integrating musical context into the chord estimation process have been proposed. As yet, the ways in which musical context has been incorporated into chord extraction models have been mainly exploratory, but this is a great advance from the simplistic models that used only chroma information for audio chord detection. The next paragraph reports that musical context models are used in commercial applications too.

### 2.2.4 Commercial Tools

Automatic chord extraction is not a widely-used feature of commercial music software. The company *D'Accord*[9] sells a chord recognition product aimed at guitarists, with fretboard chord visualisation. From the videos available at the website it becomes clear that the automatic chord recognition performance is not state of the art, but the product offers the possibility to add lyrics,

[9] `http://www.daccordmusic.com/`

edit the chord labels and export the final result as a simple lyrics and chords text file. A similar quality and functionality is provided in the program *Chord Pickout*[10]. A more sophisticated approach to chord transcription is built into the newer versions of the commercial software Band in a Box[11]. The extraction itself can be done automatically, but works well only when the user provides bar line positions. The program can display an approximate transcription of the music as discrete note events. We assume that the chord transcription is then generated from this transcription. The user can choose whether he wants the program to estimate one or two chords per bar. The user can also choose a fine tuning, the time signature and key of the piece. The output is clearly not perfect, but good, and has impressive detail (including chord inversions) which generally makes sense musically. The program allows the user to subsequently edit the chord sequence, add lyrics and melody, and output lead sheets. This approach shows that for the analysis of an individual song, a semi-automatic approach can lead to very good results.

## 2.3 Evaluation of Chord Labels

The quality of a chord transcription can be assessed only in relation to the piece of music it transcribes. There are guidelines in the music theory literature of what makes a good chord transcription (e.g. Felts, 2002, Chapter 7), but they are usually based on the assumption that the chord and the chord boundaries have already been recognised and interpreted by a human expert. Automatic chord transcription, at this stage, seeks to imitate the way an experienced human chord annotator would find and segment chords in audio, and further improvement in style of chord labels in ambiguous situations is still uncharted territory. Therefore, chord transcription evaluation has largely concentrated on metrics that focus on the correctness of chord root (with enharmonic equivalence) and type, and the correct chord segmentation granularity.

### 2.3.1 Correct Overlap

In order to assess the quality of an automatic transcription it is common practice to compare it to some reference transcription ("ground truth") provided by a human expert. Not many reference transcriptions are available, and the existing ones (Harte et al., 2005; Mauch et al., 2009a) have been used extensively by researchers and in the Music Information Retrieval Evaluation Exchange (MIREX, Downie et al., 2010) chord detection tasks. The song is partitioned into contiguous segments, of which onset time, offset time and a chord label are transcribed. The chords can then be made available in different formats. The .lab format is human-readable, as

[10] `http://www.chordpickout.com`

[11] `http://www.pgmusic.com/`

in this example[12]

```
35.635 36.711 A
36.711 37.759 D:min
37.759 38.790 A:7
```

where the first and second fields represent onset and offset time, and the third is the chord label in the notation proposed by Harte et al. (2005).

It has been common practice to use the *relative correct overlap* (RCO) with respect to a ground truth annotation as an accuracy measure. It measures the proportion of the duration of a piece or collection of music on which the automatic transcription matches the reference annotation. This is sometimes implemented as relative frame count (e.g. in the MIREX tasks and Sheh and Ellis, 2003),

$$\frac{\text{\# matching frames}}{\text{total \# of frames}} \tag{2.7}$$

or as relative overlap in seconds of the scores (Catteau et al., 2007; Mauch and Dixon, 2008, 2010),

$$\mathsf{RCO} = \frac{\text{summed duration of correct chords}}{\text{total duration}}. \tag{2.8}$$

Equations (2.7) and (2.8) provide the same result up to rounding errors introduced by using the frame-wise approach. Usually, a collection of songs is used to evaluate an algorithm. Two slightly different approaches to do so have emerged, with the past two years' MIREX tasks being examples for each one, respectively. In 2008, the relative overlap measure was calculated for every song, and the final score was obtained by taking the mean over all songs in the collection. In 2009, the strategy changed and the mean of all songs, now weighted by their length, was used, which is equivalent to applying the overlap measure directly on the whole collection.

### 2.3.2 Chord Classes

Manual transcriptions often feature a great amount of detail, particularly extended chords. For example, the reference transcriptions available to us for evaluation (Harte et al., 2005; Mauch et al., 2009a) contain hundreds of different chord symbols, constructed from around 160 chord

[12] taken from "Erbauliche Gedanken Eines Tobackrauchers" (Mauch/Rust), transcription available at `http://isophonics.net/content/reference-annotations`

types. On the other hand, as discussed above, automatic chord transcription methods often have a rather limited chord vocabulary, often only the 24 maj and min chords. Therefore, in order to evaluate algorithms chords are usually organised into a smaller set of chord classes. Typically, these are based on a choice of a set of chords that all other chords are then mapped to, as exemplified in Table 2.4. Often, the 24 maj and min labels are used as classes to map chords to, usually complemented by a class for "no chord". Unfortunately, only some papers explicitly report results evaluated on more detailed chord class sets (Sheh and Ellis, 2003; Papadopoulos and Peeters, 2008), and very few also detail the performance of the algorithms by chord class (Oudre et al., 2009; Mauch and Dixon, 2008, 2010).

| original chord | *by first third (majmin)* | *by root* |
|---|---|---|
| C | C | C |
| Cmin | Cmin | C |
| F♯min | F♯min | F♯ |
| G♭min | F♯min | F♯ |
| D7 | D | D |
| Emin7(♭9) | Emin | E |
| Bdim | Bmin | B |
| N (no chord) | N | N |

Table 2.4: Example chords and their chord classes as classified in the MIREX Chord Detection tasks. The leftmost column shows the chord as given in the manual annotations, the middle column shows the chord as classified by the first third in the chord (25 different labels), and the last column as classified by only considering the root of the chord (13 different labels). Enharmonic equivalence is assumed as symbolised by G♭ root that is converted to F♯.

In general, slight differences in the choice of chord class sets, and—possibly more so—the use of different data sets have rendered the direct comparison of different chord transcription algorithms impossible. The best assessment remains the MIREX Chord Detection task, despite the simplistic chord class set used for evaluation.

### 2.3.3 Segmentation Quality

Correct chord overlap is not the only relevant indicator of the quality of a chord transcription. It is also desirable that the chord boundaries of the automatic transcription segment the song in a way similar to those of the reference annotations. Clearly, if the overlap metric discussed above returns a perfect match, then the segmentation quality will also be perfect[13], and often good overlap will correlate with good segmentation performance. In some cases, however, a tran-

[13] This may not always be true in practice because we do not merge identical consecutive chords in the ground truth.

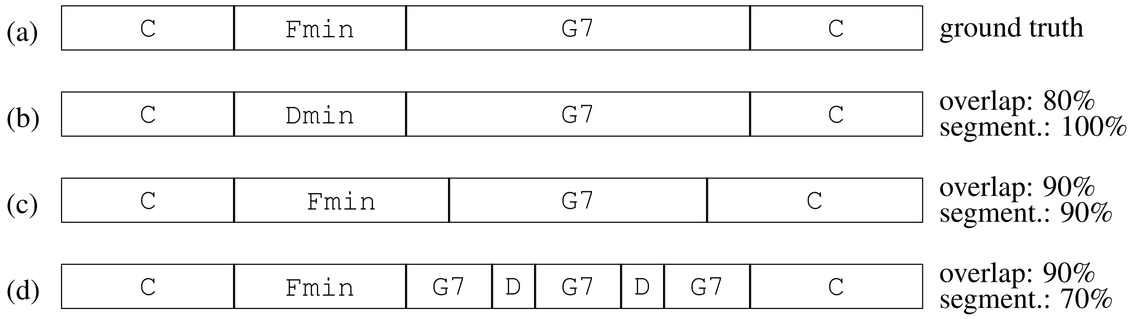


Figure 2.6: Example of overlap metric (Equation (2.8)) and segmentation metric (Equation (2.11)). Against the ground truth (a), transcription (b) has only 80% correct overlap because the second chord is incorrect, but the segmentation is perfect. Transcription (c) has more correct overlap, but a slightly lower segmentation score (90%). Finally, transcription (d) has the same overlap score as transcription (c), but a much lower segmentation score because the transcription is too fragmented.

scription that is good in terms of correct overlap can be quite fragmented (i.e. bad segmentation quality), or a transcription that systematically misinterprets one chord can still have very good segmentation performance. Figure 2.6 provides some examples that illustrate this behaviour.

Chord segmentation quality has received very little attention. Though not explicitly concerned with segmentation similarity, the deletion and insertion counts introduced by Catteau et al. (2007) can give a sense of segmentation quality. The chord change rate, used by Mauch and Dixon (2008) and subsequently by Khadkevich and Omologo (2009a) is quite a coarse measure because it does not take into account the segmentation quality of individual chords, but essentially only expresses that a chord transcription should ideally have as many chord changes as the ground truth.

A metric called *directional Hamming divergence*[14], which has been used in the context of image segmentation (Huang and Dom, 1995) and structural segmentation (Abdallah et al., 2005), takes into account the quality of every single segment. For each element $B_i = [\text{starttime}_i, \text{endtime}_i]$ of a contiguous segmentation $B$, the directional Hamming divergence measures how much of it is *not* overlapped by the maximally overlapping segment of the other segmentation. Then the values over all intervals are summed, i.e. given two segmentations

[14] also called *directional Hamming distance*

$B^0 = (B_i^0)$ and $B = (B_i)$ the directional Hamming divergence is

$$h(B||B^0) = \sum_{i=1}^{N_B} \left( |B_i^0| - \max_j |B_i^0 \cap B_j| \right), \tag{2.9}$$

where $|\cdot|$ is the duration of a segment. It describes how fragmented $B$ is with respect to $B^0$, and that means, if $B^0$ is the ground truth chord segmentation, then $h(B||B^0) \in [0, T]$ is a measure of over-segmentation. Conversely, $h(B^0||B)$ is a measure of the under-segmentation of $B$ with respect to the reference annotation. In both cases, a small value indicates a good transcription. It is usually desirable to know both, and Mauch and Dixon (2010) combined the two measures by taking their mean and normalising by the duration $T$ of the song:

$$\frac{h(B||B^0) + h(B^0||B)}{2T} \in [0, 1]. \tag{2.10}$$

In an online discussion on the MIREX wiki[15] Chris Harte suggested to take (1 minus) the maximum of the two values instead to capture the worse case. This may capture better the overall quality of the segmentation, since it will only be high when neither under-segmentation nor over-segmentation are dominant:

$$H(B, B^0) = 1 - \frac{1}{T} \max\{h(B||B^0), h(B^0||B)\} \in [0, 1]. \tag{2.11}$$

It is desirable that an automatic transcription $B$ have high $H(B, B^0)$ against a ground truth segmentation $B^0$.

The segmentation quality metrics described above can be evaluated without taking the actual chord labels into account. This is a welcome side-effect because such a metric is a true complement to the RCO measure discussed in Section 2.3.1.

### 2.3.4 Comparing Algorithms

When comparing two or more algorithms, an additional problem arises, namely that of determining whether the differences between their overlap, or segmentation scores are significant. While this problem has not received much attention in chord transcription papers, the results of the MIREX tasks provide statistical analysis of significant differences using Friedman rank tests, which we have adopted for evaluation in our own work (Mauch et al., 2009c; Mauch and Dixon, 2010). The data considered for this test are the individual song-wise overlap scores as

[15] http://www.music-ir.org/mirex/2009/index.php/Audio_Chord_Detection

described above. The analysis of variance test (ANOVA) (e.g. Flury, 1997) can be appropriate in many such circumstances, but it is a parametric test that assumes that the data from every algorithm is distributed according to a Gaussian distribution and that the distributions have the same variance. As has been pointed out (Mauch et al., 2009c), these assumptions cannot be maintained for song-wise overlap scores. The appropriate solution is then to consider a non-parametric test. The Friedman test is a non-parametric test based on rank statistics. It works on the song-wise rank of the algorithms under scrutiny and determines whether the mean ranks significantly differ. This has the additional effect of removing the so-called row effects: the differences in difficulty between songs are adjusted for.

## Conclusions

In this chapter we have reviewed the literature on the problem of chord transcription from a musical perspective and in the light of related research in automatic chord detection, and then finished with a survey of evaluation techniques. We saw that existing chord extraction techniques have indeed taken into account some of the aspects that characterise chords in music theory and perception. Automatic chord detection requires making these aspects explicit in terms of algorithms, which leads to these two main problems: finding features that make the physical audio signal processable in musical terms by the computer (low-level processing and chord model), and finding a high-level framework that incorporates the horizontal and vertical abstraction performed by human listeners (context models).

While these two aspects have received a considerable amount of attention by researchers, the work has only just started: chromagrams as they are used today may be superseded by chroma features that show the perceptual salience of pitch classes, rather than a direct transform of the physical signal. Furthermore, research in context modelling has already shown that key, bass, rhythm and other features are indeed important for the recognition of chords, but so far the models do not integrate all these sources of information. Other context qualities have not even been taken into account: the structural segmentation of a song, its genre, melody, phrase structure and instrumentation are still missing from chord detection approaches.

All these directions promise to improve chord transcription, especially when bearing in mind our research goal of providing methods for musically acceptable chord transcriptions for musicians. We believe that driving the research in context modelling is the most interesting and seminal way to progress, and the obvious first step in that direction is to combine the context qualities that have already been used in chord detection. We will present our approach in Chapter 4. Extending the context models to new features is the next step. We present our approach to include structural segmentation in Chapter 6. On the low-level side, the front end, adapting chord models to the data or indeed developing better input features is an equally important part of chord transcription research, and we offer our approach in Chapter 5. First though, in order to get started on high-level modelling, we require baseline low-level features, chromagrams, whose calculation is explained in the following chapter.

# Beat-synchronous Bass and Treble Chromagrams 3

In this chapter we describe how we transform an input audio wave to a baseline chromagram representation that can be processed by the higher-level chord and context models in Chapters 4, 5 and 6, as well as the structural segmentation algorithm presented in Chapter 6. The mentioned chapters should however be accessible without this description.

We have seen in Section 2.2.2 that there are many ways to calculate a chromagram. The approach presented here includes mainly our implementations of standard procedures such as de-noising, summing of harmonics and tuning. Our representation may be special in that it uses a dedicated bass chromagram, which was first introduced in one of our previous publications (Mauch and Dixon, 2008) and independently by Ryynänen and Klapuri (2008). The version described here has been used for our more recent work (Mauch et al., 2009c,b).

The chroma calculation results from three subsequent steps, schematically depicted in Figure 3.1. First, we calculate a note salience representation from the audio (Section 3.1). Obtaining a beat-synchronous chromagram from a note salience representation then involves only a few simple steps. First, we apply pitch-domain windows to the salience in order to distinguish bass and treble frequency regions and wrap the resulting windowed salience to chromagrams (Section 3.2). Then the chroma frames between two beat locations are summarised to yield one chroma vector per beat (Section 3.3).

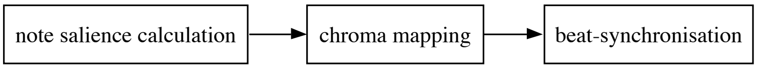


Figure 3.1: Overall view of the chroma generation process.

## 3.1 Note Salience Calculation

Note salience calculation aims at representing which notes are present at every audio frame in a piece of music. We would like to stress that the method presented in this section is not capable of extracting pitch salience like a human would perceive it (Parncutt, 1989). Our approach uses the inner product of the spectrum with harmonic tone patterns and is therefore similar to the harmonic sum spectrum (Noll, 1970). The resulting "salience" will have high values not only at fundamental frequencies, but also at other harmonics of complex tones. We are aware that the two concepts are not the same but will continue to use the word *salience* because we perform some non-linear transformations, and hence describing the results as amplitude or energy would be inappropriate.

Our salience function takes as an input an audio file in WAV[1] format, and outputs a matrix representing the note salience of every note at every time frame. In the following paragraphs we will explain the intermediate steps.

### 3.1.1 Frequency-Domain Transform

The input to the chroma extraction function is a monaural audio waveform at a sample frequency of $f_s = 11025$ Hz. If the audio file under consideration has a higher sample rate, the audio is low-pass filtered and downsampled to $f_s$ using the MATLAB® function `resample`.

We window the input wave using a Hamming window of length $N_F = 4096$ samples at a hop size of 512 samples. The choice of the FFT length $N_F$ is determined by the closest co-occurrence of two sinusoids we require to be resolvable, i.e. we want to find the shortest FFT length that is large enough to meet our frequency resolution requirements. This is mainly an issue in the lower frequency range. We observe that bass notes are generally set apart from higher notes by a wide note interval. In the context of jazz, Lawn and Hellmer (1996, p. 136) recommend that the "C below middle C be considered the lower limit for the lowest pitch in a voicing. Rootless voicings with the bottom tone below this point will sound muddy and interfere with the register occupied by the bass player [...]." Butterworth (1999) makes similar recommendations for Classical music: "3rds low down in the texture sound muddy" . We therefore assume that simultaneous notes occurring in bass frequency regions, they will be spaced some minimum pitch interval apart. We are more conservative than would be necessary according to (Butterworth, 1999) and require only that the note A below middle C can be separated from the semitone below (MIDI notes 56 and 57), and that the note A in the second octave below middle C can be separated from the (full) tone below (MIDI notes 43 and 45).

To ensure computational efficiency and high time resolution we are interested in minimum FFT length that fulfills these requirements. If $f_1$ and $f_2$ are the frequencies of the notes in either

[1] Waveform Audio File in PCM encoding

of the above requirements, Smith (2008) gives a lower bound for the minimum FFT length $M$ as

$$N_F \geq M = K\frac{f_s}{|f_2 - f_1|}, \tag{3.1}$$

where $K$ is a constant that depends on the window function. Smith (2008) also provides the value of $K = 4$ for the Hamming window we decide to use. Both requirements are met if

$$N_F \geq \max\left\{\frac{4 \cdot f_s}{|220 - 220 \cdot 2^{-1/12}|}, \frac{4 \cdot f_s}{|110 - 110 \cdot 2^{-2/12}|}\right\} \approx 3675. \tag{3.2}$$

Hence, our choice of FFT length is $N_{\text{FFT}} = 2^{12} = 4096$, the smallest power of 2 greater than the bound in (3.2). Then, if $x_{k,m}$ is the Hamming-windowed signal in the $m^{\text{th}}$ frame,

$$\begin{gathered} X_{k,m} = \sum_{j=0}^{N_F-1} x_{k,m} e^{-\frac{2\pi i}{N_F}kn} \\ k = 0, ..., N_F/2 - 1, \end{gathered} \tag{3.3}$$

is its discrete Fourier transform. We will use only the amplitude of these signals, and since the Fourier transform is symmetric for real-valued signals we use only the first half of the elements in the spectra described above,

$$\begin{aligned} A_{k,m} =& |X_{k,m}|, \\ & k = 0, ..., N_F/2 - 1. \end{aligned} \tag{3.4}$$

In preparation of the calculation of the cosine distance in Section 3.1.2, we normalise every column of $A_{k,m}$ with respect to the $L_2$ norm, which means that for the $m^{\text{th}}$ column $||A_{\cdot,m}|| = 1$.

### 3.1.2 Preliminary Salience Matrix

We use two collections, $M^s$ and $M^c$, of tone profiles. For a given frequency $f_0$, the collection $M^s$ contains simple tone profiles with one peak at $f_0$, similar to a constant-Q kernel, and $M^c$ contains complex harmonic tone profiles with peaks at integer multiples of $f_0$. We then use these profiles to create two corresponding salience matrices $S^s$ and $S^c$, which in turn will be combined to the preliminary salience matrix $S^{\text{pre}}$ that is passed on to the next step. The following paragraphs explain the details.

In both profile collections $M^s$ and $M^c$, the tones range from MIDI note 21 (A0 $\approx$ 27.5 Hz) to MIDI note 92 (G$\sharp$6 $\approx$ 1661 Hz), which means that we consider six octaves. Two consecutive tones have a frequency ratio of $2^{1/36}$, i.e. the tones are spaced a third of a semitone apart. The rows of the simple tone profile matrix $M^s$ then consist of the amplitude spectrum of a Hamming-windowed sine wave at the fundamental frequency of the $j^{\text{th}}$ tone. Each tone profile, i.e. each row of $M^s$, is subsequently normalised by dividing it by its $L_2$ norm.

Analogously, we compute a second matrix $M^c$ containing amplitude spectrum profiles of complex tones at the same fundamental frequencies as the simple tones above. The complex tones are realised as the sum of four sine waves with frequencies $k \cdot f$ and geometrically modelled (Gomez, 2006) amplitudes $s^{k-1}$ for the $k^{\text{th}}$ partial. The choice of four partials and the parameter $s = 0.9$ are hand-tuned, and testing different values could lead to improvements. In Chapter 5 we propose a different approach to chroma using the same geometric partial model, and test different values of $s$.

Given the two collections of normalised tone profiles $M^s$ and $M^c$, we calculate a preliminary salience for each frame and note using cosine similarity on the STFT magnitude matrix $A$. This results in two preliminary salience matrices

$$S^s = M^s \cdot A \quad \text{and} \quad S^c = M^c \cdot A. \tag{3.5}$$

The product with the complex-tone matrix $S^c$ is similar to applying a harmonic sum spectrum and emphasises pitches that have high amplitudes at multiple frequencies, i.e. at the frequency of harmonics of harmonic tones, and especially at the fundamental frequency. The reason for additionally calculating $S^s$ is that $S^c$ has a high value not only at partials of a harmonic tone, but unfortunately also at sub-harmonics, e.g. at the frequency 220 Hz, if the true note played has a fundamental frequency of 440 Hz. $S_s$ does not suffer from this phenomenon, but from the reverse problem of possibly having high values at pitches that are not harmonics.

Since we are only interested in the harmonic content of a piece of audio, we introduce a next step which serves for broadband spectral noise reduction: we will use only those notes for which both preliminary salience matrices have values that exceed the local standard deviation from the mean, i.e. which are near spectral peaks. To determine which values have high standard deviation we consider the columns $S^s_m$ (and analogously $S^c_m$), and run mean and standard deviation filters on it with a window length of half an octave, i.e. 18 bins, yielding vectors $\mu^s_m$ and $\sigma^s_m$ (and analogously $\mu^c_m$ and $\sigma^c_m$). To form the combined preliminary salience matrix $S^{\text{pre}}$, the element-wise product of $S^s_m$ and $S^c_m$ is taken for bins that are greater than one standard deviation above the mean in both. All others are set to zero.

$$S^{\text{pre}}_{j,m} = \begin{cases} S^s_{j,m} \cdot S^c_{j,m} & \text{if } S^s_{j,m} > \mu^s_{j,m} + \sigma^s_{j,m} \\ & \text{and } S^c_{j,m} > \mu^c_{j,m} + \sigma^c_{j,m}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.6}$$

This preliminary salience matrix still has three bins per semitone. These three give us the possibility to adjust the matrix to the tuning of the piece, as is explained in the following subsection.

### 3.1.3 Tuning and Reduction to Semitones

So far, the rows in $S^{\text{pre}}$ relate to third-of-semitone spaced tones, i.e. three consecutive rows (or bins) relate to one semitone, with the middle bin relating to the respective tone given a standard tuning frequency of 440 Hz. Since pieces of music are not generally tuned to 440 Hz, we compensate for tuning differences (assuming equal temperament) of less than half a semitone by re-adjusting the matrix such that the middle bin corresponds to the estimated semitone frequency. Assuming furthermore that the tuning does not change over the course of a piece of music, we consider the average preliminary salience over all time frames

$$\bar{S} = \frac{1}{T} \sum_{t=1}^{T} S^{\text{pre}}_t. \tag{3.7}$$

Treating $\bar{S}$ itself as a signal, the tuning of the piece now corresponds to the phase angle $\varphi$ of the DFT of $\bar{S}$ at the normalised frequency $\pi/3$, i.e. the song tuning in semitones from the 440 reference tuning is

$$\delta = \frac{\text{wrap}\left(-\varphi - \frac{2\pi}{3}\right)}{2\pi} \in [-0.5, 0.5), \tag{3.8}$$

where wrap is the phase wrap operation to the interval $[-\pi, \pi)$. The estimated tuning frequency is hence $\tau = 440 \cdot 2^{\delta/12}$. We use this information to update the salience matrix $S^{\text{pre}}$ using linear interpolation so that the middle bin of each semitone now corresponds to the semitone frequency in the estimated tuning. The final salience matrix $S$ results from adding, for each time frame, the three bins belonging to one semitone,

$$S_{r,m} = \sum_{i=3r-2}^{3r} S^{\text{pre}}_{i,m}, \quad r = 1, \ldots, 72. \tag{3.9}$$

In the salience matrix $S$, every row corresponds to one semitone from MIDI note 21 to 92, and every column corresponds to one time frame at hop size 512 samples (46 ms). An example of $S$ can be seen at the top of Figure 3.2. The salience matrix can now easily be transformed to a chromagram representation, as we will explain below.

## 3.2 Range-specific Chroma Representations

The underlying reason for considering the frequency regions separately is the special role of the bass note, which we have discussed in Section 2.1.3. That is why the methods presented in Chapters 4, 5 and 6 use separate bass and treble chromagrams. This section describes how we calculate them. For the segmentation algorithm in Chapter 6 we additionally use a "wide" chromagram, which is the sum of the bass and treble chromagrams.

We emphasise bass and treble regions of the salience matrix $S$ in the respective chromagrams by applying the pitch-domain windows shown in Figure 3.3. Every profile is a vector with one element for each semitone. The bass profile $g_b$ vector elements equal 1 in the core bass region (MIDI notes 33–44) and level off linearly to both sides. The same applies to the treble profile $g_t$ (core treble region: MIDI notes 56–68). Where the two profiles overlap, their sum is unity, and the mid-point of their overlap is MIDI note 50, i.e. one octave below middle C. The wide profile $g_w$ is the sum of the treble and bass profiles and hence spans the whole pitch range.

The respective bass, treble and wide chromagrams are easily calculated. The chroma of the $j$th pitch class at frame $m$ is

$$C_{j,m} = \sum_{k=0}^{5} S_{j+12k,m} \ \cdot \ g(j+12k), \tag{3.10}$$

where $g$ is either $g_t$, $g_b$, or $g_w$, as appropriate, and $k$ is the octave index running over the six octaves we consider. These chromagrams could now be used in general applications, but our algorithms require beat-synchronous chroma vectors. This last step of the chroma generation process will be explained below.

## 3.3 Beat-Synchronisation and Normalisation

Beat-synchronisation is the process of summarising frame-wise features that occur between two beats. The beat times that are required in this process can be obtained either manually or automatically, and while most of our methods use automatic beat-tracking, Chapter 6 features some

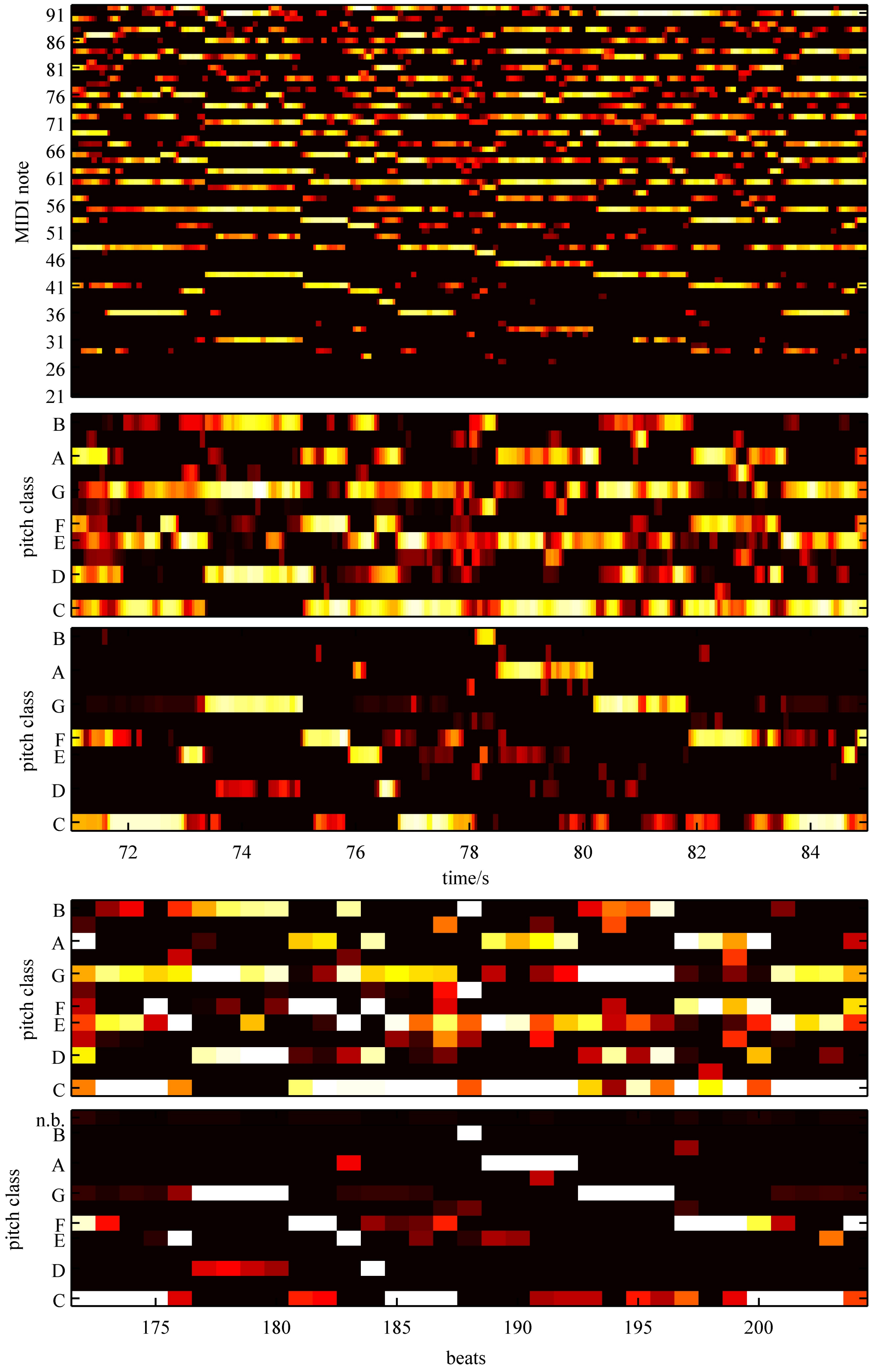


Figure 3.2: Examples of chromagrams, calculated from the song *Let It Be* (Lennon/McCartney). From top to bottom: semitone-spaced salience $S$, treble chroma and bass chroma ($C_{j,m}$), beat-synchronous treble chroma, beat-synchronous bass chroma ($C_{j,t}^{\text{sync}}$). The label "n.b." refers to the "no bass" bin. Lighter shades mean higher salience.

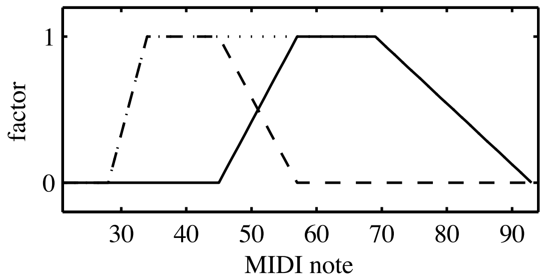


Figure 3.3: Note range profiles: bass profile $g_b$ (dashed line), treble profile $g_t$ (solid), and wide profile $g_w$ (dotted).

experiments with manual beat annotations taken from the OMRAS2 Metadata Project (Mauch et al., 2009a). The beat-tracking algorithm used in the remaining methods was developed by Davies et al. (2009), and we use a version from 2008.

We obtain a single chroma vector for each beat by taking the median (in the time direction) over all the chroma frames $m \in \mathfrak{B}_t = \{m \mid m^{\text{th}} \text{ frame is between the } (t-1)^{\text{th}} \text{ and } t^{\text{th}} \text{ beat}\}$ with centres between two consecutive beat times.

$$C^{\text{sync}}_{j,t} = \operatorname*{median}_{m\in\mathfrak{B}_t} C_{j,m}, \quad j = 0, \dots, 11. \tag{3.11}$$

The bass chroma is extended by a 13th, “no bass” bin. It describes a measure of flatness of the other twelve bins, which is used in the dynamic Bayesian network proposed in Chapter 4:

$$C^{\text{sync}}_{j,t} = \left(\frac{1}{12} \cdot \frac{\sum_{j=0}^{11} C^{\text{sync}}_{j,t}}{\max_j C^{\text{sync}}_{j,t}}\right)^2 \in \left[\frac{1}{144}, 1\right] \tag{3.12}$$

These beat-synchronous bass and treble chroma vectors can then be normalised so as to be independent of the magnitude of the signal. For example, the model presented in Chapter 4 assumes normalisation by the maximum norm, i.e. every bass and treble chroma vector is separately divided by its maximum value.

### Conclusions

In the present chapter we have explained our baseline chroma extraction method. The method combines methods of tuning, noise reduction, emphasis of fundamental frequencies, and beat-synchronisation that are mostly equivalent to similar methods in previous work, if in slightly different variations. As a departure from most other chromagram extraction techniques (with the exception of Ryynänen and Klapuri, 2008), we produce two different chromagrams for different frequency ranges, and a third one that is the sum of the other two.

In the following chapter, the bass and treble chromagrams are used as the input to a high-level model of chords and context, in which we demonstrate the positive effect of including explicit information on the bass range.

# A Musical Probabilistic Model 4

Chords are not simply isolated descriptors of a piece of music. Rather, they become meaningful only when perceived in conjunction with other musical qualities. Among these qualities, it is arguably the key that influences the perception of chords most strongly—and vice versa. The key sets the scene for what happens harmonically, and though the key does not strictly exclude any chord from being played, all chords are interpreted with reference to the current key. This does not mean, however, that the key of a piece is set in stone: a new key arises when the chords can no longer be interpreted as part of the old key, and the hierarchy as discussed in Section 2.1.4 is restored. This kind of inter-dependency is not restricted to chords and keys, but extends to relationships between rhythm, melody, bass and other musical qualities.

The chord transcription approach proposed in this chapter takes inter-dependencies and hierarchies into account and uses them to produce better chord labels. The method is closely linked to one of our previous publications (Mauch and Dixon, 2010). The novelty of the approach is that it integrates in a single graphical model pieces of musical context that had previously been assessed only separately: keys, chords, metric position and bass pitch class can now be estimated *simultaneously* using the efficient inference techniques available for dynamic Bayesian networks. This also means that key changes are tracked and beat omissions and deletions are recognised—parameters which can be used to create lead sheets.

Figure 4.1 shows an overview of our system. In Section 4.1 we motivate our design choices. Section 4.2 details the topology and parameter settings of the novel dynamic Bayesian network. Section 4.3 describes the experiments, and 4.4 provides comparative evaluations of our methods, followed by a discussion and conclusions.

## 4.1 Motivation

While our general goal of providing good transcriptions is simple, we would like to recapitulate the motivation that prompted us to pursue the direction of integrating musical context

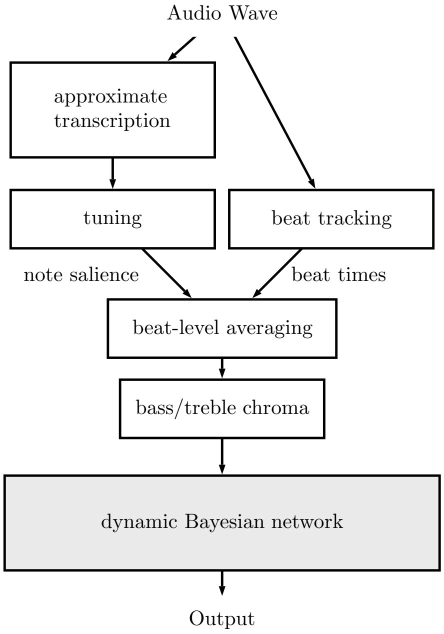


Figure 4.1: A schematic overview of our method (see Section 4.2). White boxes represent the chroma extraction sub-methods described in Chapter 3.

parameters in a probabilistic model. We have argued in Section 2.1 that chord transcription is quite different from polyphonic pitch transcription in that two kinds of abstraction are usually performed by a human listener: notes are integrated into chords over time, and non-harmony notes are discarded; in the pitch dimension, height is largely discarded with the exception of the position of the bass note (relative to the chord root). The abstraction over time is aided mainly by the knowledge of key and rhythm, and vertically, the knowledge of the bass note is essential. To determine the chord label at a particular position within a song, it is generally not enough to consider the chroma or other low-level tonal descriptors at that position. Instead, a multitude of musical qualities are required. In Section 2.2 we have shown that probabilistic models of different musical qualities have been used to improve chord transcription, though usually in isolation. In an attempt to “dig deeper into the music itself” (Downie et al., 2009, page 17) we present a model that integrates several of these approaches and attempts to approximate human music listening better. Importantly, the interdependence of musical parameters should be modelled, and inference on them should be simultaneous. In fact, Raphael calls the inter-dependence of musical qualities we have mentioned in the introduction of this chapter the “chicken and egg problem” (Raphael, 2005, p. 659), and strongly argues for simultaneous estimation for cases in which such inter-dependence arises. *Dynamic Bayesian networks* (DBN) (Murphy, 2002) offer a probabilistic framework to describe the inter-dependence of several parameters, and provide simultaneous inference methods. The next section explains how we make use of this framework to achieve better chord modelling.

## 4.2 Network Model

The foundation of dynamic Bayesian networks are *Bayesian networks* (*BN*s). A BN is a joint distribution of several random variables. It is called a "network" because its dependency structure can be represented using a directed acyclic graph. Every node represents one random variable[1]. A directed edge represents a direct dependency; it points at the node that directly depends on the node from which the edge originates. This duality of the graph and the joint distribution allows very intuitive modelling of several musical qualities as detailed in this section. The requirement of the graph to be acyclic means that there is no dependency "short circuit", so a random variable is never its own descendent.

To model time series with BNs, *dynamic* Bayesian networks (DBNs) are used (Murphy, 2002). A DBN can be thought of as a succession of simple BNs. Like in a HMM, the succession is assumed to be Markovian and time-invariant, i.e. the model can be described recursively by defining only two slices (Boyen and Koller, 1998): one "initial state" slice and one "recursive" slice. Such models are also called *2-slice temporal Bayesian networks* (2-TBN). Note that any DBN could equivalently be modelled as an HMM, comprising the different state variables of the DBN in a single (very large) state variable. As a result, modelling of the adequate HMM is less intuitive and inference can be much slower (Murphy, 2002, page 20–21).

In the proposed DBN topology shown in Figure 4.2 discrete nodes model the states of metric position, key, chord, and bass pitch class. Continuous nodes model bass and treble chroma. Our DBN is a generative model, i.e. some state configuration sequence of the hidden source nodes is assumed to have generated the observed data, which in our case are the bass and treble chromagrams whose calculation we have described in Chapter 3. This assumption allows us to use Bayesian reasoning to infer the state sequence from the data (Leistikov, 2006, p. 96). We use the Bayes Net Toolbox (Murphy, 2001) written in MATLAB, which implements inference methods for DBNs, to model the data and perform the inference. The inference method of our choice is Viterbi decoding, which finds the most probable explanation of the observed data as given in Equation (2.6) in terms of the four layers of discrete nodes.

The definition of the network topology in 2-TBN form is provided by Figure 4.2. To complete the definition of the network the *conditional probability distributions* (CPD) of the

[1] We will use the two expressions *node* and *random variable* interchangeably.

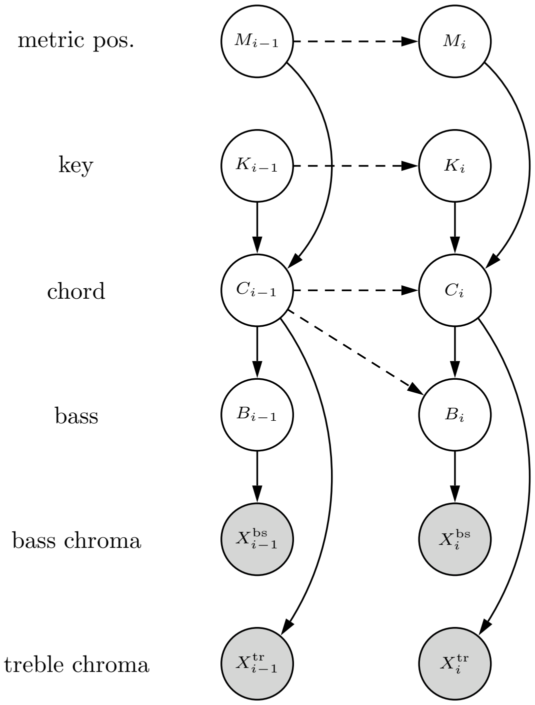


Figure 4.2: Our network model topology, represented as a 2-TBN with two slices and six layers. The clear nodes represent random variables, while the observed ones are shaded grey. The directed edges represent the dependency structure. Intra-slice dependency edges are drawn solid, inter-slice dependency edges are dashed.

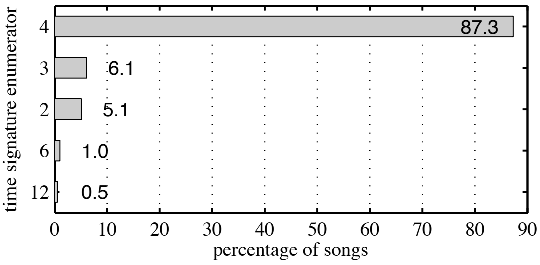


Figure 4.3: Frequency of predominant time signature numerators in the OMRAS2 beat annotations (Mauch et al., 2009a). Time signatures in 4 are by far the most frequent.

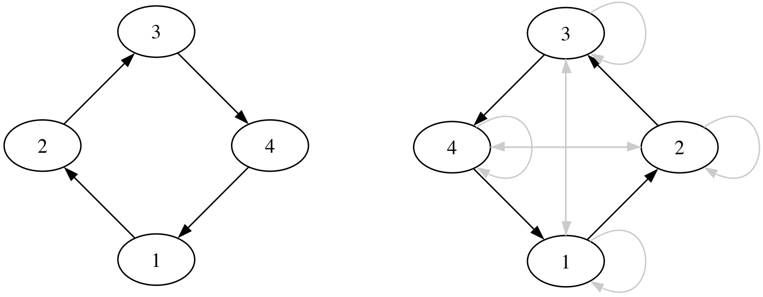
random variables need to be specified, providing a good approximation of how beats, keys, chords and bass interact. Since we do not have any preconception of the initial metric position, key, chord or bass pitch class of a piece, all initial nodes are set to a uniform distribution. In the rest of this section we will be concerned with the details of the CPDs of the recursive nodes on the right hand side of the 2-TBN depicted in Figure 4.2.

Like Leistikov (2006) we map expert musical knowledge onto a probabilistic network. In the process of developing the method, many design choices were made to be able to express this musical knowledge. This chapter focuses on the model itself, and we show that our choices, based on informed considerations of music theory and expert judgement, result in state-of-the-art performance.

### 4.2.1 Metric Position

Western music is usually grouped in bars, each containing a number of beats. The arrangement of strong and weak beats within a bar is described by the *time signature*, denoted usually in a notation similar to fractions: the numerator corresponds to the number of beats per bar (e.g. 3 in the case of $\frac{3}{4}$ time), and the denominator to the note value at the beat level (e.g. 4, a quarter note or crotchet, in the case of $\frac{3}{4}$ time). The 197 manual beat annotations in the OMRAS2 Metadata Project (Mauch et al., 2009a) contain information about the numerator. We found that the predominant time signature numerator in 87.3% of the songs is 4, see Figure 4.3. Songs in *even meter*, i.e. they have a time signature numerator of either 2 or 4, make up 92% of the collection. In the light of this data, we choose to restrict ourselves to modelling only one kind of time signature, namely $\frac{4}{4}$.

A simple model of beat progression would have to express the following straightforward semantics visualised in Figure 4.4a: the first beat (metric position 1) in a bar is followed by the second (metric position 2), and so on, until after the fourth the next bar starts on metric position



(a) simple meter model

(b) implemented meter model

Figure 4.4: Metric position: visualisation of the metric position state transitions as given in (4.1). In the ideal model (a) $\varepsilon = 0$, hence all bars are in $\frac{4}{4}$ meter and all four beats have to be reached before returning to beat 1. The implemented model (b) still assumes a $\frac{4}{4}$ meter, but allows for missing and inserted beats. Black arrows represent a transition probability of $1 - \varepsilon$ ($\varepsilon = 0.05$) to the following beat. Grey arrows represent a probability of $\varepsilon/2$ for either self-transition or transition to the beat after the following, see (4.1).

1. Hence, the node $M_i$ has four states to represent the metric position of the current beat. This is essentially the model used by Papadopoulos and Peeters (2008). This simple model does not take into account occasional inaccuracies in the beat-tracking procedure. Beats may be missed or erroneously inserted. Additionally, we often deal with pieces of music in which beats are intentionally omitted or added to a bar by the songwriter or artist. For robustness against these two sources of irregularity we allow for the small probability $\varepsilon$ of deviation from the normal succession of beats, which we choose to be $\varepsilon = 0.05$. Since in our network topology (Figure 4.2) node $M_i$ depends only on node $M_{i-1}$, the conditional distribution $P(M_i|M_{i-1})$ can be represented as a two-dimensional transition matrix

$$\begin{pmatrix} \varepsilon/2 & \underline{1-\varepsilon} & \varepsilon/2 & 0 \\ 0 & \varepsilon/2 & \underline{1-\varepsilon} & \varepsilon/2 \\ \varepsilon/2 & 0 & \varepsilon/2 & \underline{1-\varepsilon} \\ \underline{1-\varepsilon} & \varepsilon/2 & 0 & \varepsilon/2 \end{pmatrix},$$

in which each row represents a state of $M_{i-1}$, and every column a state of $M_i$. The "usual" transitions are underlined for better readability. The same information can be written as a conditional probability distribution (CPD),

$$P(m_i|m_{i-1}) = \begin{cases} 1-\varepsilon & \text{if } (m_i - m_{i-1}) \bmod 4 = 1, \\ \varepsilon/2 & \text{if } (m_i - m_{i-1}) \bmod 4 \in \{0, 2\}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

Figure 4.4b shows a diagram of the four metric position states with these modified transition probabilities. We will encounter the metric position node again when considering the CPD of the chord node. The key node, which we discuss next, has no direct links to the metric position node.

### 4.2.2 Key

In order to describe the tonal content of a passage of music, chords are complemented by a more general concept relating to the common tonal material that extends over several consecutive chords. This concept is called *key*.

> *Key.* As a principle in music composition, implies adherence, in any passage, to the note-material of one of the major or minor scales–not necessarily a rigid adherence, but a general adherence, with a recognition of the Tonic (or "key-note") of the scale in question as a principal and governing factor in its effect. (Kennedy, 1980)

Together with the findings in the field of music perception that we discussed in Section 2.1.4 on page 26, this suggests that lower-level harmonic content (such as chords) should depend on the current key in a generative model as ours. In their key estimation method, Noland and Sandler (2009) model the key as a hidden variable which "generates" chords, and we follow this general idea. We assume that the modelling of the key and key changes will have two benefits: it improves the stability of the chord estimation by making off-key chords less probable, and the retrieved keys and key changes can be used to set the right key signature at the typesetting stage, for example in a lead sheet.

In our network model (Figure 4.2) the node $K_i$ is a discrete random variable modelling all 24 major and minor keys. In accordance with the observation in the previous paragraph, the key node governs the chord node. In fact, the keys are defined by a scale that contains the note-material predominantly used in that key, but this definition will be used only in the following subsection (4.2.3) to describe the CPD of the chord node. The key node itself depends only on its predecessor in time. Hence, in order to model the key we only need to express the key transition probabilities. Since to the best of our knowledge there is no model of key change

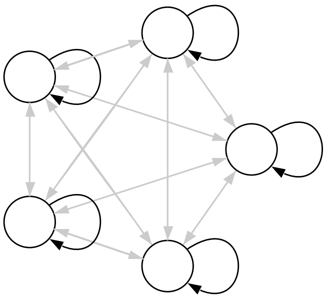


Figure 4.5: Key: state transition graph for keys. This is a simplified representation with only five keys. The key states are fully connected, but transition to a different key (grey) has a low probability of 0.02, see (4.2).

for popular music, we choose to model only the fact that a key is constant over a passage, as illustrated in Figure 4.5. In other words, while making no assumptions of the probability of a target key given a key change, we do model the rate of key change, expressing that at a given beat the key is expected to remain the same with a high probability. We set this probability to 0.98, i.e. we assume that at any beat the key changes with a probability of 0.02,

$$P(k_i|k_{i-1}) = \begin{cases} 0.98 & \text{if } k_{i-1} = k_i, \\ (1-0.98)/23 & \text{otherwise.} \end{cases} \tag{4.2}$$

The way in which the key acts upon the chord is coded into the chord CPD as detailed in the following subsection.

### 4.2.3 Chord

The chord node $C_i$ is the focus point of the DBN as depicted in Figure 4.2 on page 67. Together with the treble chroma node $X_i$ it forms the core part of our method. The discrete CPD of $C_i$ describes the dependencies of the chord node on its predecessor $C_{i-1}$, the metric position node $M_i$ and the key node $K_i$. The inferred state sequence in this node will essentially be the final chord transcription. The complex dependency structure as illustrated in Figure 4.2 allows us to model two vital characteristics of chord occurrence:

1. metric position dependency: a chord change is likely at the beginning of a bar (metric position 1), less likely in the middle of a bar (position 3), and even less likely at the remaining metric positions 2 and 4.

2. key dependency: a chord is more likely the fewer non-key pitch classes it contains. If it

is also a tonic chord in the key, it is yet more likely to occur.

Accordingly, we factorise the conditional probability of chord $c_i$ at the $i^{\text{th}}$ beat given the chord $c_{i-1}$ at the previous beat, the current metric position $m_i$ and the current key $k_i$ as

$$P(c_i|c_{i-1}, m_i, k_i) = P(c_i|c_{i-1}, m_i) \cdot P(c_i|k_i), \tag{4.3}$$

in which the first factor describes the dependency of a chord change on the metric position, and the second factor describes the chord's dependency on the current key. Let us consider the first factor. We translate the above statement on the chord change probability at a particular metric position to numbers in the vector

$$a = \begin{pmatrix} 0.5 \\ 0.1 \\ 0.4 \\ 0.1 \end{pmatrix}. \tag{4.4}$$

The probability of a chord given the previous chord and the current measure is then modelled by the equation

$$P(c_i|c_{i-1}, m_i) = \begin{cases} a_{m_i}/(N_{\mathrm{C}} - 1) & \text{if } c_{i-1} \neq c_i, \\ (1 - a_{m_i}) & \text{otherwise}, \end{cases} \tag{4.5}$$

where $N_{\mathrm{C}}$ is the number of chords. Chord change transitions are favoured at the first and third beats in a measure. Note that no distinction is made for different chord types. Preference for particular chords or chord changes is expressed only though the key model in the second factor of (4.3).

The second factor in (4.3) describes how likely a chord is, conditional on the key. Perceptual chord ratings in a key context are available for `maj`, `min`, and `dim` chords (Krumhansl, 1990, page 171). Though they have been used extensively in key detection research (e.g. Noland, 2009), their use in our computational model is limited because no ratings exist for the more complex chords which we consider, and even in the existing ratings there is a bias towards more consonant chords[2]. Nevertheless, we will use them as a point of reference.

[2] For example, the `Dmin` chord, which is a diatonic chord in C major, has a lower perceptual rating than any `maj` chord, e.g. the chord `F#`.

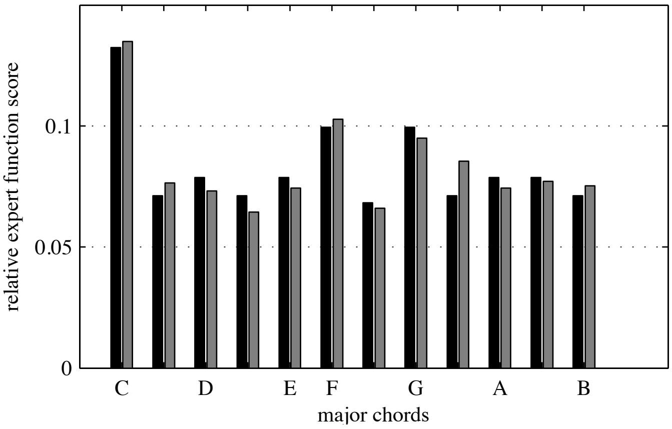


Figure 4.6: Examples of our $f$ chord-key ratings (black) for major chords in a C major key context, compared to the Krumhansl profiles (grey) as described on page 73, both normalised by the $L_1$ norm over the 12 examples given.

Instead of using the ratings directly, we introduce a parametric expert function $f$ to measure how well a chord fits with a key, and adapt its parameter $\nu$ using Krumhansl's perceptual ratings for major chords in a major key context. First of all, however, we notice that the ratings for major chords in a major key range between 4.3 and 6.7, which means they have a high baseline rating of 4.3. To increase the contrast between high and low ratings, we subtract half of the minimum rating from all ratings, yielding twelve ratings $r = (r_1, \ldots, r_{12})$ for the twelve major chords in a C major key context. Then, we need a parametric function whose parameter $\nu$ is going to be adjusted such that they fit the chord ratings:

$$f^\nu(c,k) = \begin{cases} 2e^{-\nu-(\text{\# notes in } c \text{ but not in } k)} & \text{if } c \text{ is tonic in } k, \\ e^{-\nu-(\text{\# notes in } c \text{ but not in } k)} & \text{otherwise.} \end{cases} \tag{4.6}$$

Let the vector $f^\nu = (f_1^\nu, \ldots, f_{12}^\nu)$, with $f_i^\nu = f^\nu(c_i, \text{C major})$, $i = 1, \ldots, 12$, contain the values obtained for the twelve major chords in a C major key context.

Using numerical optimisation, we determine the smoothing parameter $\nu$ by maximising the correlation between $f^\nu$ and the chord ratings vector $r$. Once the parameter $\nu$ is fixed, we use the standard linear regression method to find the parameters $\beta_0$ and $\beta_1$ to minimise the least squares distance $||r - (\beta_1 f^\nu + \beta_0)||$, and the final expert function is then defined for all chords $c$ using the optimal parameters $\nu$, $\beta_0$ and $\beta_1$:

$$f(c,k) = \beta_1 f^\nu(c,k) + \beta_0. \tag{4.7}$$

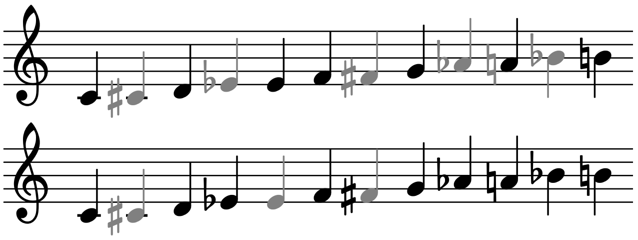


Figure 4.7: The two key types, exemplified with a tonic of C: C major (top) and C minor key scales. Pitch classes in black belong to the key as defined for the purpose of our model, pitch classes in gray do not. Natural, harmonic and melodic minor scales are not distinguished: our minor scale contains all pitch classes appearing in any of these minor scales. Note spelling is not taken into account: enharmonically equivalent pitch classes are treated as equivalent.

Examples are shown in Figure 4.6. The function takes into account how many notes of the chord do not match the current major or minor scale, and whether the chord is a tonic chord. For our model we assume the keys are defined as by a binary vector that indicates which pitch classes are considered part of the key, as illustrated in Figure 4.7. A chord is considered a tonic chord, if in addition to sharing all notes with the current key scale, its root coincides with the tonic of the key. For example, `Gmaj7` is a tonic chord in the key of G major, but `G7` is not. This model has the advantage that the $f$ score does not directly depend on the number of notes in the chord, and that it allows us to calculate scores for any chord defined by a pitch class set.

The $f$ values are then normalised by a constant $\kappa_{k_i}$ such that considering all chords $P(C_i|k_i) = \kappa_{k_i} \cdot f(C_i, k_i)$ is a conditional probability distribution, i.e. for a fixed key $k_i$ the probabilities sum to 1. This finalises the proposed probability distribution of the chord node $C_i$, given its dependencies on the previous chord, the metric position, and the key. The following subsections deal with probability distributions that in turn depend on the chord node.

### 4.2.4 Bass

The bass pitch class plays a crucial role in the recognition of chords, both in classical music and popular music styles. Being at the bottom of the frequency range, it “anchors” the chord and makes the rest of the notes more easily interpretable. For example, knowing the bass note can help disambiguate chords such as `Cmaj6` and `Amin7`, which share the same pitch class set. In particular, distinguishing different inversions of the same chord is impossible without the notion of a bass note. No previous audio chord extraction methods have dealt with this problem.

Bass lines tend to include several different notes over the span of one chord. The role of the bass pitch class of a chord becomes clear if one observes that in popular music the bass note is almost always present on the first beat of a chord. One popular bass player tutorial

(Westwood, 1997) confirms this: among the 207 example bass patterns covering styles Blues & R'n'B, Soul, Motown/Atlantic Records, Funk, and Rock only 20 do *not* start with the bass pitch class. Allowing for some more variation than given in these examples, we estimate that the bass note played and the nominal chord bass note coincide on the first beat of the chord 80% of the time. This behaviour can be modelled, since the bass node $B_i$ depends on both the previous chord node $C_{i-1}$ and the current chord node $C_i$, i.e. we know that if $C_{i-1} \neq C_i$ then the current beat is the first beat of a new chord.

$$P(b_i|c_{i-1} \neq c_i) = \begin{cases} 0.8 & \text{if bass is nominal chord bass,} \\ 0.2/12 & \text{otherwise.} \end{cases} \tag{4.8}$$

As the chord continues, we still expect the "nominal" bass pitch class as the most likely option, but the other pitch classes of the chord may be used as a bass note too. These pitch classes still share the probability 0.8, but the chord pitch class retains double weight among them. This leaves 0.2 for the rarer cases in which the bass can play notes that do not belong to the chord.

$$P(b_i|c_{i-1} = c_i) = \begin{cases} 0.8 \cdot \frac{2}{n_n+1} & \text{if bass is chord bass,} \\ 0.8 \cdot \frac{1}{n_n+1} & \text{if otherwise the bass is in the chord,} \\ 0.2 \cdot \frac{1}{13-n_n} & \text{otherwise,} \end{cases} \tag{4.9}$$

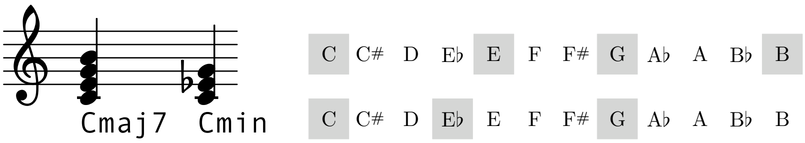
where $n_n$ is the number of notes in the current chord. The bass model presented here goes far beyond the one presented by Yoshioka et al. (2004) and our own previous method (Mauch and Dixon, 2008) in that it gives special weight to the chord bass note, rather than preferring just any chord note or only the root note. Specifically, it enables the recognition of chord inversions.

### 4.2.5 Chroma

The chroma nodes $X_i$ and $X_i^{\mathrm{b}}$ provide models of the chroma features. Unlike the nodes previously discussed, they are continuous nodes because the 12 elements of the chroma vector represent relative salience (see Chapter 3), which can assume any value between zero and unity. The BNT toolbox provides a Gaussian node class for the modelling of continuous variables, which we use to describe both treble and bass chroma.

#### Treble Chroma

We employ a data-independent way of modelling chords such as the ones shown in Figure 4.8 and define a chord by the pitch classes it contains. In the example of the `Cmin` chord in Figure 4.8, these are C, E♭, and G. This is reflected in the treble chroma emission model. As



(a) musical notation

(b) binary pitch classes

Figure 4.8: Chord examples: `Cmaj7` and `Cmin` chords in musical notation and a binary pitch class representation. The shaded squares in (b) denote the pitch classes belonging to the chord. To obtain the same chord type with a different root, the chord is "rolled" (circular shift).

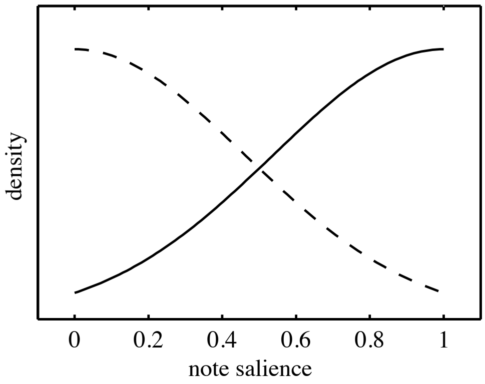


Figure 4.9: Treble chroma node: distribution of single elements of the 12-dimensional Gaussian, monotonically increasing curve for chord pitch classes, monotonically decreasing curve (dashed) for non-chord pitch classes.

has been explained in Chapter 3, the chroma features $x_i \in [0, 1]$ are normalised by the maximum norm, so high values will be close to one, and—ideally—low values will be close to zero. Hence, the probability density $P(X_i|c_i)$ of the chroma node given a chord should monotonically *increase* with any of the chord pitch class saliences increasing (C, E♭, and G in the case of a `Cmin` chord). It should monotonically *decrease* with any of the non-chord pitch class saliences increasing. In a manner very similar to Bello and Pickens (2005) and Catteau et al. (2007) (see also Figure 2.4 in this thesis) we model this behaviour as a 12-dimensional Gaussian random variable: the mean vector has ones at elements representing the chord pitch classes, and zeros at the elements representing non-chord pitch classes. We choose to use a diagonal covariance matrix and set the variances in all dimensions to $\sigma^2 = 0.2$. Figure 4.9 shows the marginal probability density distribution over the interval $[0, 1]$ for a single dimension for the case in which this dimension corresponds to a chord note and a non-chord note, respectively. Note that due to the chroma normalisation, a flat chroma vector will contain only ones. Therefore, we define `N` (no chord) as including all pitch classes.

**Bass Chroma**

The bass chroma is modelled in much the same way as the treble chroma, by a multidimensional Gaussian vector. Its number of dimensions is $13 = 12 + 1$, with 12 dimensions representing the bass pitch classes `C` through `B`, and the thirteenth representing "no bass note" (*cf.* Chapter 3). Since the bass is defined by just one note, every profile has only one element for which the mean value is set to $1$ (rather than 3 or 4 in the case of chords), while the others are set to $0$. Usually only one bass note is played at any time, which implies that the pitch class played will more often have a normalised salience of 1, and the other pitch classes will have saliences close to zero. We choose a lower variance value of $\sigma^2 = 0.1$.

The description of the chroma nodes completes the definition of the DBN. We have established musically-informed conditional probability distributions for metric position, key, chord, bass pitch class, and the corresponding chroma emission nodes. Note that while modelling essential properties of popular music in $\frac{4}{4}$ time, the CPDs described in this section do not explicitly suppress or encourage particular key, chord or bass note transitions. The experiments in Section 4.4 show that the individual components of the system lead to improved chord transcriptions of recordings of popular songs.

## 4.3 Experimental Setup and Chord Label Mapping

We conduct several experiments to investigate the overall performance of our model and the influence of choice of chord set, metric position, bass note, and key. We use the song collection from the 2009 MIREX Audio Chord Detection subtask[3], which contains 210 songs (174 by the Beatles, and 18 by Queen and Zweieck, respectively), making it the largest audio-aligned chord test set used to date. A list of all songs can be found in Appendix B.

### 4.3.1 Configurations and Inference

We consider 10 different DBN configurations by varying two parameters: the network topology and the size of the chord vocabulary. Since the DBN described in Section 4.2 is modular, the network topology can be changed by disabling selected nodes. We consider these four network topologies of ascending complexity:

***plain*** In the *plain* model, the modelling of metric position, key, and bass pitch class is disabled; chord duration is modelled as similar to our previous work (Mauch and Dixon, 2008) as a negative binomial distribution[4] with shape parameter 2, and scale parameter 1/3,

[3] `http://www.music-ir.org/mirex/2009/index.php/Audio_Chord_Detection`

[4] the discrete analogue of a gamma distribution

corresponding to an expected chord duration of 4 beats.

***M*** In the *metric* model (*M*), metric position is fully modelled as described in Section 4.2; the bass and key nodes are disabled.

***MB*** In the *metric-bass* model (*MB*), the bass pitch class node and the bass chroma node are additionally enabled.

***MBK*** The *metric-bass-key* model (*MBK*) is the entire model as described in 4.2.

The treble chroma node and the chord node are always enabled. As a second variable, the following three different chord vocabularies are tested:

***majmin*** consists of the 24 `maj` and `min` chords as well as the "no chord" symbol `N` (25 chords).

***inv*** additionally contains the chord inversions `maj/3` and `maj/5` (49 chords).

***full*** contains chords of types `maj`, `min`, `maj/3`, `maj/5`, `maj6`, `7`, `maj7`, `min7`, `dim`, `aug`, and, the "no chord" class `N` (121 chords).

The *full* vocabulary is the largest chord vocabulary tested on a substantial collection of musical audio. We infer the most likely state sequence for the enabled discrete nodes using the Viterbi algorithm as provided by the `find_mpe` function in the BNT Toolbox, which we slightly modified for better memory usage. All calculations are performed in MATLAB on a shared computer running CentOS 5.3 with 8 Xeon X5570 cores at 2.93GHz, 24 GB RAM. The song "Ticket to Ride" (Lennon/McCartney) as performed by the Beatles has a typical pop single play time of 190 seconds and takes 131 seconds to process (21 seconds CPU time). Memory consumption peaks at 6 GB. Longer songs can take considerably more time to process, and much more memory. "I Want You (She's So Heavy)" (Lennon/McCartney) is the longest song in the test set (467 seconds) and takes 354 seconds to calculate (135 seconds CPU time). Memory consumption peaks at 15 GB.

### 4.3.2 Chord Label Mapping for Evaluation

To measure the relative correct overlap (RCO) of a transcription with respect to a ground truth annotation, we use the metric

$$\mathsf{RCO} = \frac{\text{summed duration of correct chords}}{\text{total duration}}, \tag{4.10}$$

which is equivalent (up to rounding) to the measure used in the 2009 MIREX Chord Detection tasks (see also Section 2.3.1). Note that if a collection is concerned, RCO is the duration of correctly annotated chords in the whole collection divided by the length of the whole collection. This is different from taking the mean of the song-wise RCO. The reason for our choice is that it is easily comparable with the results from the 2009 MIREX Chord Detection tasks.

We have already discussed the necessity of mapping chords to chord classes in Section 2.3.2. To evaluate our methods on different levels of detail we use two chord class sets: one is relatively coarse, and equivalent to that used in the 2009 MIREX Chord Detection task; the other one distinguishes many more chords, which enables us to analyse results in more detail. The reader will notice that the names of the following definitions correspond to the names used for two of the sets of chords we model in the DBN. This is because the different chord models directly correspond.

***majmin*** is equivalent to the mapping used in the MIREX Chord Detection Subtask[5]. It uses as a basis the 24 `maj` and `min` chord as well as the "no chord" symbol `N`. Except for a few unclassifiable chords (0.3%), all other chords are then mapped to these 25 chord classes. Usually this is decided by the first interval above the root: chords with a major third are mapped to the respective `maj` class; chords with minor thirds are mapped to the respective `min` class.

***full*** contains 121 chord classes with types `maj`, `min`, `maj/3`, `maj/5`, `maj6`, `7`, `maj7`, `min7`, `dim`, `aug` and, again, the "no chord" class `N`. They are chosen such that many chords can sensibly be mapped to these classes without changing the chord's function. Note that now, the chord symbols `maj` and `min` will unite fewer chords under their label, since chords such as the *dominant* chords (`7` chords) will be mapped to a class of their own.

All chords that cannot be mapped to any will be mapped to the "unknown" class and always considered wrongly estimated. A detailed account of all mappings and some statistics on chord type occurrence can be found in Appendix A. We will refer to the RCO metric together with the *majmin* chord class set as *MIREX-style evaluation*.

[5] `http://www.music-ir.org/mirex/2009/index.php/Audio_Chord_Detection`

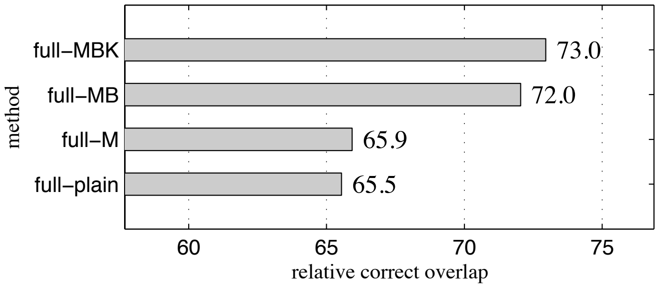


Figure 4.10: Songwise relative correct overlap for different algorithms with *full* chord vocabulary, using the MIREX-style *majmin* evaluation.

## 4.4 Results

This section is organised as follows. In Subsection 4.4.1 we compare our method to the current state of the art using MIREX-style evaluation and investigate which components improve performance with statistical significance. In Subsection 4.4.2 we report in detail the chord confusion behaviour in our best-performing model. In Subsection 4.4.3 we compare chord segmentation results between our methods. The performance in terms of key extraction is discussed in 4.4.4. Section 4.4.5 features some examples of the output of our methods.

### 4.4.1 Relative Correct Overlap

The MIREX-style relative correct overlap (RCO) is a good benchmark for comparing our algorithm to others. The song-wise original MIREX task results are freely available[6], which allows us to test the significance of the observed differences using the Friedman test as explained in Section 2.3.4.

Several versions of our algorithm (Table 4.1) have a relative correct overlap of 72%. The best configuration is *full-MBK*, reaching 73%, which means it performs better than the best performances in the 2008 MIREX pretrained Audio Chord Detection task (our own submission (Mauch et al., 2009b) at 71%). The best-performing algorithm in the train-test task was submitted by Weller et al. (2009) and scored 74%. Figure 4.10 shows the RCO results for the four different algorithms using the *full* chord dictionary. In order to determine whether the difference between the method proposed by Weller et al. (2009) and our *full-MBK* method is significant, we use a one-way ANOVA test and a Friedman ANOVA test on the song-wise relative overlaps. Both tests return $p$-values much larger than 0.05 (0.31 and 0.17, respectively), which indicates that there is no significant difference between the results: we cannot decide which method is better. In particular, since the Friedman test is based on the ranking within rows (performance

[6] `http://www.music-ir.org/mirex/2009/results/chord`

| | RCO score in % | | | |
|---|---|---|---|---|
| chord set | *plain* | *M* | *MB* | *MBK* |
| *majmin* | 65.7 | 66.5 | 70.9 | 72.1 |
| *inv* | n/a | n/a | 71.0 | 72.0 |
| *full* | 65.5 | 65.9 | 72.0 | **73.0** |

(a) MIREX-style RCO score

| | RCO score in % | | | |
|---|---|---|---|---|
| chord set | *plain* | *M* | *MB* | *MBK* |
| *majmin* | (56.4) | (57.0) | (60.3) | (61.3) |
| *inv* | n/a | n/a | (56.9) | (57.8) |
| *full* | 54.6 | 55.1 | 55.8 | 56.7 |

(b) RCO score using the *full* chord class set

Table 4.1: Overlap score results: (a) the MIREX-style RCO against (b) the more detailed RCO using the *full* chord class set. The results of models that do not output as many classes as are tested are given in brackets. Since they never transcribe complex chords, their performance is high only due to their better performance on more common chords.

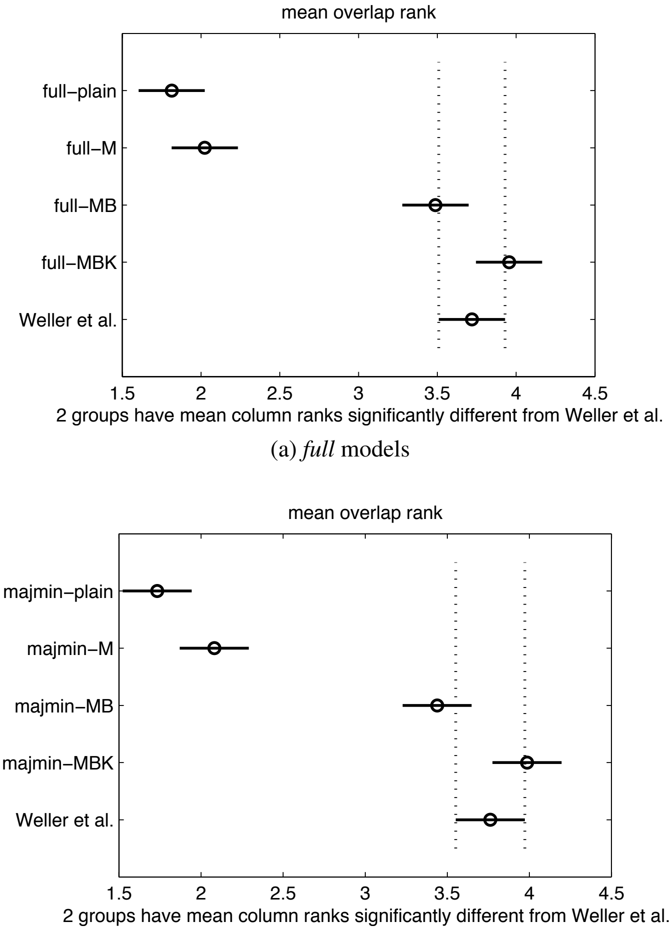
of songs), this means that the algorithms do not consistently rank differently given a song.

The use of more different chord types in the *full* chord vocabulary does not decrease the MIREX score, and in some cases goes along with an improvement in accuracy. Hence for further evaluations we consider only the configurations using *full*, the largest chord vocabulary. As can be seen from Table 4.1a, each of the added system components (metric position, bass, and key) has increased the overlap score. We would like to assess which of these improvements can be considered significant. We perform a Tukey-Kramer multiple comparison test at a confidence level of 95%, based on the Friedman analysis of variance, over the four configurations with *full* chord and also include the results by Weller et al. for comparison. Figure 4.11 shows the results of this test: except for the step from the *plain* model to the *M* model, each additionally added node achieves a significant improvement. We conclude that bass and key modelling significantly contribute to better chord labelling in our model.

### 4.4.2 Performance by Chord Type and Chord Confusion

The performance of specific chord types is relevant to understand the system. In order to show the performance of individual chords, we use the chord class set *full*, i.e. all the 121 different chords modelled in the *full-MBK* model are discerned. Here the performance is naturally much lower (57% overlap). Figure 4.12 shows the performance of the eleven individual chord types. The `maj` and `min` triads are by far the best-modelled and still have 69% and 61% relative correct overlap. All other chord classes behave much worse, ranging between 12% and 37%. While this may look discouraging at first, a closer look at the chord confusion will show that many of the errors are “well-behaved”, which explains the good performance in the MIREX-style evaluation.

Tables 4.2, 4.3, and 4.4 show eleven chord confusion matrices, one for every chord type in the *full*-MBK model as described in Section 4.2. The rows represent the different chord types,



(a) *full* models

(b) *majmin* models

Figure 4.11: Friedman significance analysis based on the song-wise RCO rank of the four configurations with (a) *full* chords and (b) *majmin* chords. In each case the MIREX results of Weller et al. are given for comparison. Where confidence intervals overlap, the difference between methods cannot be considered significant. We can see that our best model actually ranks higher on average than Weller's, if not significantly so. See discussion in Section 4.4.1.

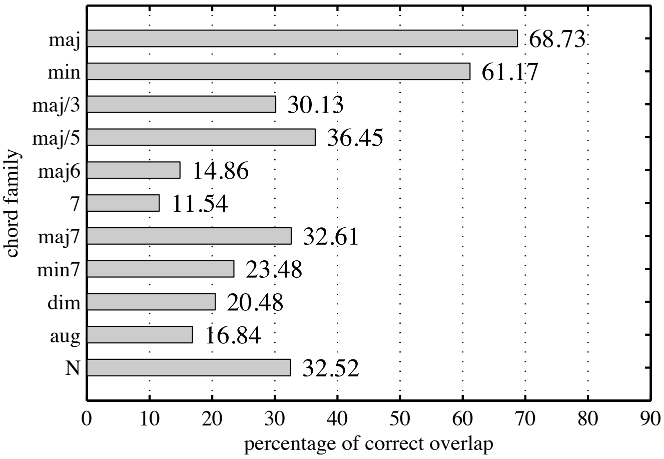


Figure 4.12: Relative overlap for individual chords in the *MBK* setting with *full* chord dictionary.

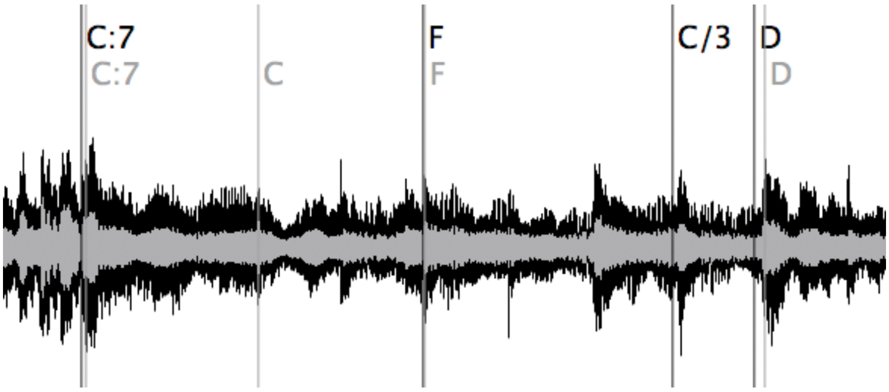


Figure 4.13: Excerpt of *Something* (Lennon/McCartney), displayed in the free software *Sonic Visualiser*. The first (black) line of chords is the ground truth transcription, the lines below (grey chord symbols) are our automatic transcription, using full chords, metric position, bass, and key (*full-MBK*).

while the columns represent relative chord root note difference. This means that the correct chord is in the zero difference column in the row that shares its chord type name. The confusion values are given as percentages, rounded to the nearest integer. Confusion values $< 0.5$ are not printed for clarity. The confusion behaviour of the chords `maj/3` and `maj/5`—the inversions of the `maj` chord—can be observed in Table 4.2: for both chord types the highest score is indeed concentrated in the respective correct bin. Most of the confusions are then dispersed on the `min` or `maj` chords with which they share the respective bass note (i.e. `4-min` and `7-maj`), and the `maj` chord of the same root. Both confusion types are thus easily explained. We have to acknowledge that, although the confusion of `0-maj/3` chords as `4-min` chords is not very high, it is likely to be due to the insufficient handling of harmonic overtones in the chroma extraction procedure, since the chord's bass note may have a strong third harmonic (creating salience at the interval of a fifth from the bass pitch class), which makes it hard to distinguish from the `4-min` chord. There is clearly some scope for improvement. It is worth noting that the confusion with the `maj` chord of the same root has no influence on the MIREX score, since both root and chord type are correctly recognised. Similarly, a human reader of the transcription is likely to forgive this kind of error, especially since conversely very few `maj` chords in root position are transcribed as inversions. This is a phenomenon which also affects the other chords with additional sixths and sevenths, i.e. `maj6`, `7`, `maj7`, and `min7`, as can be seen in Table 4.3. The dominant chord `7` has a low recall of only 12%, but confusion is heavily concentrated on the `maj` of the same root (50%). Similar observations can be made for the three remaining chord types in Table 4.3, if not always as pronounced. Again, these confusions make sense musically, since transcribing a non-triadic chord with its base triad is acceptable (and rewarded in the MIREX-style evalutation). However it is a characteristic that has not wittingly been modelled: the maj and min triads "attract" chords with more complex ground truth. This may have a number of reasons. In the case of the dominant 7 chord we can attempt a qualitative explanation in musical terms: in Figure 4.13 an excerpt of the song "Something" (Lennon/McCartney), is displayed as loaded from an automatically created XML file into Sonic Visualiser (grey). For comparison we have additionally loaded the ground truth annotations (black). Note that while the ground truth correctly annotates the first two full bars of the example as C7, our method switches back to Cmaj in the second bar. This happens because in the second bar the flat seventh that turns a Cmaj chord into a C7 is not present, but the annotator has made the musical choice of extending the chord over both bars. Our model does not take into account this kind of semantics.

A different behaviour can be found in Table 4.4, where the confusion of dim and aug triads as well as the "no chord" symbol N are shown. Here, musically acceptable confusion accounts only for a relatively small part of chord confusions (e.g. the dim chord is confused with other dim chords shifted by 3, 6 or 9 semitones). Otherwise confusion is widely spread, and indicates that the chord chroma model we used has its limitations, especially when it comes to modelling aug chords. Here, similar to the maj/3 chord, the confusion is likely to have been caused by the chromagram, which does not sufficiently take into account harmonics that remain in the chromagram. Another reason for low recall of aug chords may be their shorter individual duration. The results of the N chord are hard to interpret, since the N symbol in annotations does not necessarily imply that there are no harmonic sounds. Hence, the confusions might at times make musical sense, but investigating this would be a substantial musicological task on its own. Generally, of course, we would like to boost the performance of "no chord" detection.

The confusion behaviour in the chord transcriptions often makes musical sense. The maj and min triads expose good recall behaviour, and their extensions maj6, 7, maj7, and min7 are transcribed consistently as the corresponding base triad or indeed correctly. The performance on the "no chord" symbol N and the dim and aug chords is less satisfactory and suggests that better chord modelling, and in particular better chromagrams, could further improve results (see Chapter 5).

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | **69** | | 1 | | | 1 | | 3 | | | | |
| min | 2 | | | | 1 | | | 1 | | 1 | | |
| maj/3 | 1 | | | | | | | | | | | |
| maj/5 | 6 | | | | | 1 | | | | | | |
| maj6 | 1 | | | | | | | | | | | |
| 7 | 2 | | | | | | | | | | | |
| maj7 | 2 | | | | | | | | | | | |
| min7 | | | | | | | | | | | | |
| dim | | | | | | | | | | | | |
| aug | | | | | | | | | | | | |
| N | | | | | | | | | | | | |

(a) maj chord confusion

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 7 | | | 4 | | 1 | | 2 | | | 1 | |
| min | **61** | | | | | 1 | | 3 | | | | |
| maj/3 | | | | 2 | | | | | | | | |
| maj/5 | 1 | | | 1 | | 1 | | | | | | |
| maj6 | | | | 2 | | | | | | | | |
| 7 | 1 | | | | | | | | | | | |
| maj7 | | | | | | | | | | | | |
| min7 | 6 | | | | | | | | | | | |
| dim | | | | | | | | | | | | |
| aug | | | | | | | | | | | | |
| N | | | | | | | | | | | | |

(b) min chord confusion

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 12 | | | | 3 | 3 | | 6 | | | | |
| min | | | 1 | | 22 | | | | | 2 | | |
| maj/3 | **30** | | | | | | | | | | | |
| maj/5 | 2 | | | | | | | | | | | |
| maj6 | 1 | | | | | | | | | | | |
| 7 | 1 | | | | 1 | | | | | | | |
| maj7 | 1 | | | | | 1 | | | | | | |
| min7 | | | | | 2 | | | | | | | |
| dim | | | | | 5 | | | | | | | |
| aug | | | | | 1 | | | | | | | |
| N | | | | | | | | | | | | |

(c) maj/3 chord confusion

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 14 | | | | | | | 30 | | | | |
| min | | | | | 2 | | | 3 | | 2 | | |
| maj/3 | 1 | | | | | | | | | | | |
| maj/5 | **37** | | | | | | | | | | | |
| maj6 | 1 | | | | | | | 2 | | | | |
| 7 | | | | | | | | | | | | |
| maj7 | | | | | | | | | | | | |
| min7 | | | | | | | | 1 | | 1 | | |
| dim | | | | | | | 1 | | | | | |
| aug | | | | | | | | | | | | |
| N | | | | | | | | | | | | |

(d) maj/5 chord confusion

Table 4.2: Chord confusion table I: chord confusions for four different chord types, maj, min, maj/3, maj/5. The columns represent semitone difference from the root of the reference chord (modulo 12). The confusion values are given as percentages, rounded to the nearest integer. Confusion values $< 0.5$ are not printed for clarity. The percentage of the correct chord is printed bold.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 63 | | | | | 1 | | 1 | | | | |
| min | 1 | | | | 1 | | | | | 3 | | |
| maj/3 | 1 | | | | | | | 1 | | | | |
| maj/5 | 9 | | | | | | | | | | | |
| maj6 | **15** | | | | | | | | | | | |
| 7 | 1 | | | | | | | | | | | |
| maj7 | 1 | | | | | | | | | | | |
| min7 | | | | | | | | | | 1 | | |
| dim | | | | | | | | | | | | |
| aug | | | | | | | | | | | | |
| N | | | | | | | | | | | | |

(a) maj6 chord confusion

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 50 | | | | | 2 | | 2 | | | 2 | |
| min | 3 | | | | 1 | 1 | | 7 | | 1 | 1 | |
| maj/3 | 1 | | | | | | | | | | | |
| maj/5 | 4 | | | | | 2 | | | | | | |
| maj6 | 1 | | | | | | | | | | 1 | |
| 7 | **12** | | | | | | | | | | | |
| maj7 | 1 | | | | | | | | | | | |
| min7 | 1 | | | | | | | 1 | | | | |
| dim | | | | | 2 | | | 1 | | | | |
| aug | 1 | | | | | | | | | | | |
| N | | | | | | | | | | | | |

(b) 7 chord confusion

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 27 | | 2 | | | | | 7 | | | | 1 |
| min | | | | | 11 | | | | | 1 | | |
| maj/3 | 7 | | | | | | | 1 | | | | |
| maj/5 | 2 | | | | | | | 1 | | | | |
| maj6 | 1 | | | | | | | 1 | | | | |
| 7 | | | | | | | | | | | | |
| maj7 | **33** | | | | | | | 1 | | | | |
| min7 | | | | | | | | | | | | |
| dim | | | | | | | | | | | | |
| aug | | | | | | | | | | | | |
| N | | | | | | | | | | | | |

(c) maj7 chord confusion

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 6 | | | 5 | | 1 | | | 1 | | 5 | |
| min | 32 | | | | | | | 4 | | | | |
| maj/3 | | | | 6 | | | | | 1 | | | |
| maj/5 | | | | 5 | | 1 | | | | | | |
| maj6 | | | | 2 | | | | | | | 1 | |
| 7 | 1 | | | | | | | | | | | |
| maj7 | | | | | | | | | | | | |
| min7 | **24** | | | | | | | 1 | | | | |
| dim | | | | | | | | | | | | |
| aug | | | | | | | | | | | 1 | |
| N | | | | | | | | | | | | |

(d) min7 chord confusion

Table 4.3: Chord confusion table II: chord confusions for four different chord types, maj6, 7, maj7, min7. Confusion values $< 0.5$ are not printed for clarity. The percentage of the correct chord is printed bold.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 5 | | 1 | 1 | | | 5 | | | 9 | 2 | 1 |
| min | 3 | | 1 | 9 | | | | | | | | |
| maj/3 | 1 | | | | | | | | 5 | | | |
| maj/5 | | | | | | 1 | 2 | | 1 | | | 1 |
| maj6 | 1 | | | | | | 1 | | | | | |
| 7 | 1 | | 1 | | | | | | | | | |
| maj7 | | | 1 | | | | | | | | | |
| min7 | 2 | | | 1 | | | | | | | | |
| dim | **21** | | | 6 | | | 2 | | | 7 | | |
| aug | 1 | | | | | | | | | | | |
| N | | | | | | | | | | | | |

(a) dim chord confusion

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 44 | | | | 1 | 1 | | | | | | |
| min | 4 | 1 | | | 1 | 4 | | | | | | |
| maj/3 | 2 | | | | 1 | | | | 3 | | | |
| maj/5 | | | 1 | | 1 | 2 | | | 1 | | | |
| maj6 | | | | | | | | | | | | |
| 7 | 4 | | | | | | | | | | | |
| maj7 | 1 | | | | | | | | | | | |
| min7 | | | | | | | | | | | | |
| dim | 1 | | | | | | | | | | | |
| aug | **16** | | | | 3 | | | | 5 | | | |
| N | | | | | | | | | | | | |

(b) aug chord confusion

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 25 | | | | | | | | | | | |
| min | 11 | | | | | | | | | | | |
| maj/3 | 5 | | | | | | | | | | | |
| maj/5 | 9 | | | | | | | | | | | |
| maj6 | 2 | | | | | | | | | | | |
| 7 | 2 | | | | | | | | | | | |
| maj7 | 3 | | | | | | | | | | | |
| min7 | 1 | | | | | | | | | | | |
| dim | 7 | | | | | | | | | | | |
| aug | 3 | | | | | | | | | | | |
| N | **32** | | | | | | | | | | | |

(c) N confusion matrix

Table 4.4: Chord confusion table III: chord confusions for two different chord types, dim, aug, and the "no chord" symbol N (since N has not root note, all confusions are given in the 0 column). Confusion values $< 0.5$ are not printed for clarity. The percentage of the correct chord is printed bold.

| | segmentation score | | | |
|---|---|---|---|---|
| chord set | *plain* | *M* | *MB* | *MBK* |
| *majmin* | 0.743 | 0.759 | 0.780 | 0.781 |
| *inv* | n/a | n/a | 0.781 | **0.782** |
| *full* | 0.721 | 0.711 | 0.781 | 0.779 |

Table 4.5: Segmentation score. Generally there is a tendency of the more complex models to perform better. The metric model increases segmentation performance only in the case of the *majmin* configurations.

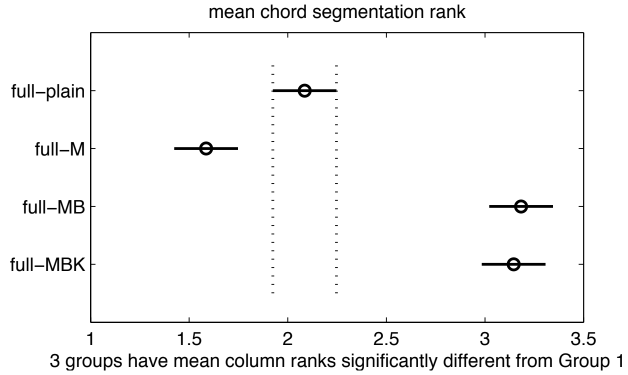
### 4.4.3 Chord Segmentation Quality

False over-segmentation can make a chord transcription much harder to read, and under-segmentation necessarily results in wrongly annotated chords. In both cases the transcription is less useful, even if the overlap measure discussed above (Section 4.4.1) is moderate to good. In fact, chord segmentation quality on its own can point to a good chord estimation, regardless of the correct overlap.

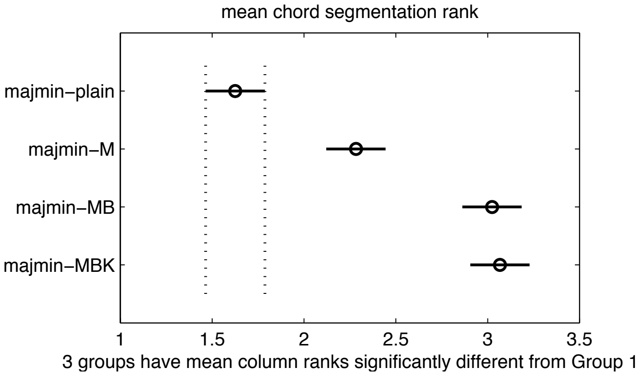
We evaluate the segmentation quality according to the $H$ measure discussed in Section 2.3.3, given in Equation (2.11), on all *full* chord versions. Table 4.5 shows the segmentation scores for all ten configurations. For the models using *majmin* and *full* chord dictionaries we have also illustrated the results of the Tukey-Kramer multiple comparison with a confidence level of 95%, see Figure 4.14. Modelling the bass results in a significantly improved segmentation score. Modelling the key does not greatly influence the segmentation score. The influence of meter modelling is ambiguous. While it significantly increases the segmentation score with respect to the *plain* model for the configurations using the *majmin* chord dictionary (Figure 4.14b), it significantly decreases the segmentation score for the configurations using the *full* chord dictionary (Figure 4.14a). We interpret these results as follows: bass modelling, and to some degree meter modelling, provide means of finding chord change positions at a level of granularity more closely related to manual annotations.

### 4.4.4 Key Estimates

In order to automatically generate a score of a piece of tonal music, it is essential to have key information. The proposed model *full-MBK* estimates the key dynamically, and simultaneously with the chords, so key changes will also be tracked, and can be used to insert a new key signature in a musical score (see Figure 4.18). Since we have the beat-wise data, we use an overlap measure similar to the one used for chord recognition, based on the key ground truth from the OMRAS2 Metadata Project (Mauch et al., 2009a). The results are very encouraging:



(a) *full* models



(b) *majmin* models

Figure 4.14: Chord segmentation.

77% of all key regions are detected correctly[7]. It is common to measure key detection accuracy by main key. We derive the main key as the one that occupies the largest amount of time in the piece, according to the ground truth and automatic extraction respectively. The portion of songs for which the main keys match is 80%, which compares very well with other key extraction algorithms. In 90% of all cases the correct main key is part of the automatic transcription.

### 4.4.5 Examples

Our system can automatically generate LilyPond[8] source files. Figures 4.15, 4.17, and 4.18 show excerpts of lead-sheet-like engravings (compiled from the LilyPond source files) using only the output of our *full-MBK* method. In Figure 4.15 the lead sheet makes use of key, chord inversion, and metric information to provide a detailed notation that matches the official version from Mercury et al. (1992) depicted in Figure 4.16.

As remarked above, the dynamic modelling of metric position and key enables us to automatically transcribe time and key signature changes. Even if "Back In The USSR" (Lennon/McCartney) is mainly in $\frac{4}{4}$ meter, after the second verse, two extra beats are inserted. The DBN inference compensates by remaining in the same metric position state twice, leading to two measures in 5/4, as can be seen in Figure 4.17. Unfortunately, in this particular case the correct key A major has not been recognised, but rather a closely related key—the correct key's subdominant D major. An example of very good key recognition is shown in Figure 4.18: the key correctly changes from E♭ major to B major and back. Note that the knowledge of the key also enables us to pitch-spell correctly, i.e. had the whole song been transcribed as E♭ major, then the `G♯min` chord just after the first key change would have been transcribed as `A♭min`. The inferred bass pitch class, written as a note in the bass clef staff, can be used as a rough guideline only. For a proper transcription (Ryynänen and Klapuri, 2008), additional octave information as well as onset and finer-grained duration information would be required.

## 4.5 Discussion

We have briefly noted in Section 4.3.1 that the memory needed to run our most complex DBNs can exceed 10GB. This prohibits their usage on today's personal computers, which usually have around 2GB of RAM. However, there are a number of ways in which the memory size can be reduced. Implementing the model in a programming language that passes values by reference

[7] There are a few pieces annotated as modal. We convert them to the major/minor scale they share the third degree with, i.e. mixolydian is taken as major, dorian is taken as minor, etc. This is common practice, see (Noland, 2009, page 69).

[8] http://lilypond.org/web/

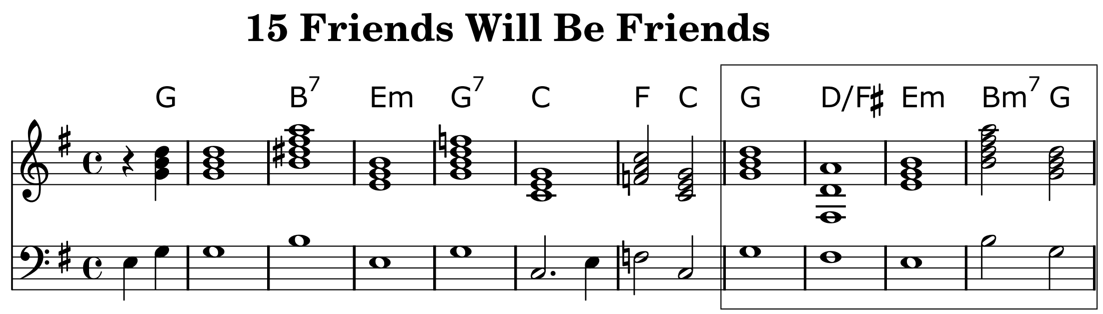


Figure 4.15: Excerpt of an automatic output of our algorithm using the *full-MBK* model for "Friends Will be Friends" (Deacon/Mercury). In the second bar of the four bars marked with a box, the `Dmaj` chord is correctly identified as being in first inversion (`D/3`, here D/F♯). The key signature of G major is also correct. The notes in the treble clef staff replicate the information of the chord symbols, the bass notes are the ones inferred from the system. Bass and chord lengths are appropriately merged or divided at bar lines in a post-processing step. Music engraving by LilyPond.

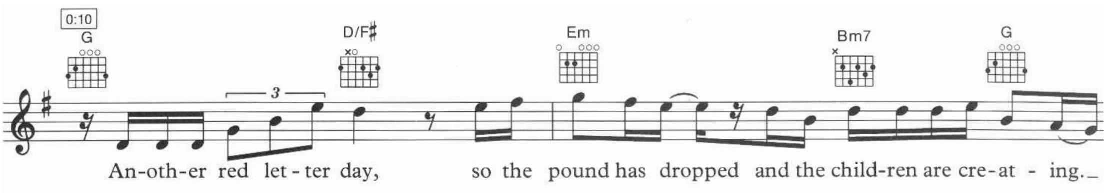

0:10
G
D/F♯
Em
Bm7
G
3
An-oth-er red let-ter day, so the pound has dropped and the child-ren are cre-at - ing._

Figure 4.16: Pop music lead-sheet: excerpt of *Friends Will Be Friends* (Deacon/Mercury) taken from (Mercury et al., 1992). Chords are represented both by chord labels and the corresponding guitar fingering. The number in a box denotes the physical time. The bass is represented only implicitly in the chord labels.

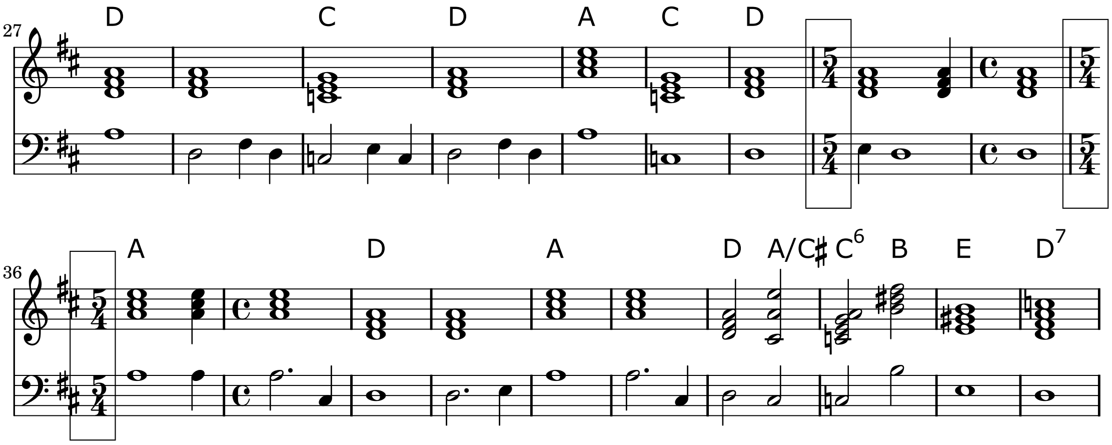


Figure 4.17: Time signature change: excerpt of an automatic output of our algorithm using the *full-MBK* model for "Back In The USSR" (Lennon/McCartney). The metric position node enables inference of time signature changes (marked with boxes) to account for inserted/deleted beats. In this case the inserted beats are part of the composition. Music engraving by LilyPond.

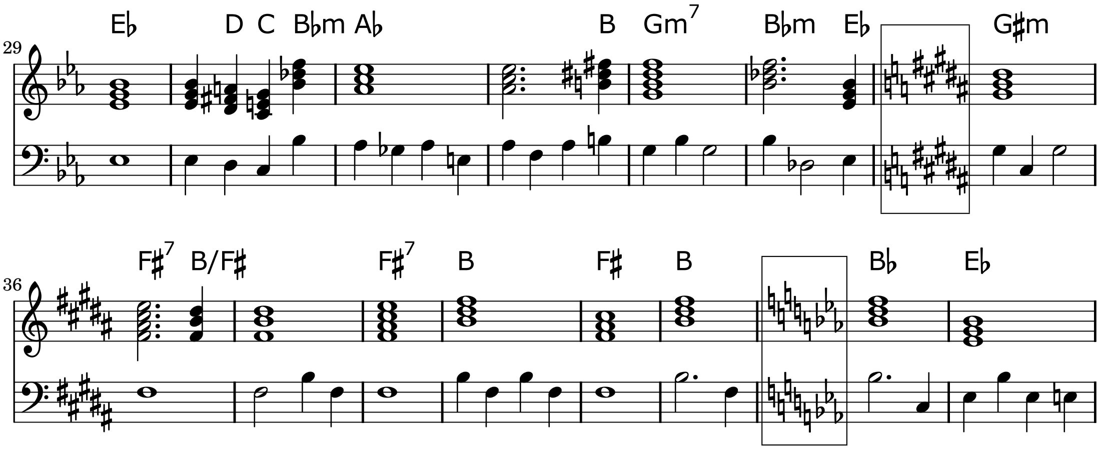


Figure 4.18: Key signature change: excerpt of an automatic output of our algorithm using the *full-MBK* model for "Good Old-Fashioned Lover Boy" (Mercury). The key node enables inference of key signature changes (marked with boxes). Music engraving by LilyPond.

(instead of copying) and has better memory management could alleviate this practical problem. Speech recognition models also have a large number of states in similar probabilistic models, so borrowing memory-saving techniques could be a way to further increase practical usability. For example, while we want to keep modelling the key dynamically, we believe it is safe to remove or "prune" the $n$ least likely keys or even chords from the model. We also show in Chapter 6 how memory consumption can be greatly reduced without significant performance loss by considering structural segments of a song instead of the whole song.

One striking observation is that the modelling of meter has helped improve the chord segmentation measure only for one of two cases. Informal tests show that a metric position model in which missing two beats in one bar is allowed (compare Equation 4.1) leads to the expected improvement for the models with *full* chord dictionary too. In future work, we would like to study this parameter because—as we have seen in the examples—having the metric position available allows the creation of lead sheets. Furthermore, implementation of time signatures other than $\frac{4}{4}$ is analogous, and preliminary experiments have shown that it is possible to track even time signature changes within one song.

### Conclusions

We have presented a novel musically-informed dynamic Bayesian network for the automatic extraction of chord transcriptions from musical audio. While being a manually-tuned expert model, it is not a rule based model; rather, it reflects the natural inter-dependence of these entities by simultaneous inference of metric position, key, chord, and bass pitch class. With 121 chords, the model provides a higher level of detail than previous approaches, including chord inversions of major chords. Sophisticated bass pitch class modeling acknowledges the special position of the bass at the first beat of the chord.

The comprehensive context model of the proposed *full-MBK* compensates for deleted or inserted beats, detects key changes and infers the nominal bass note for `maj` chords. These capabilities are essential for the creation of lead sheets. We have provided several examples of lead sheets created from our fully automatic transcription.

The proposed *full-MBK* method achieves a state-of-the-art correct overlap score of 73%, and outperforms all systems tested in the 2009 MIREX task for pretrained chord detection. In the train-test evaluation task the algorithm proposed by Weller et al. (2009) scores better than our *full-MBK* model in terms of relative correct overlap, but does not do so significantly. We compared 10 different variants of our algorithm and show that bass and key modelling cumulatively improve the method's performance with statistical significance. The greatest enhancement is achieved by bass modelling. The key model does not only aid the correct identification of chords, but also performs well in its own right by correctly identifying 80% of the songs' main keys. The relative overlap of correctly recognised keys is 77%. The high number of recognised chord types provides new musical information, without decreasing the performance of the method.

As a complement to the correct overlap evaluation method, we have used a metric for chord segmentation quality to show how well the locations and granularity of chord changes resemble those of the ground truth. Our results show a significant improvement in segmentation quality due to bass modelling, and for some models also for metric position modelling. Our best model achieves a segmentation measure of 0.782.

Our evaluation of the confusion behaviour of the system has shown that—although the model behaves as expected most of the time—chords that do not contain the fifth degree above the bass note (`maj/3`, `maj/5`, `aug`, `dim`) perform worse as those that do. We attribute this mainly to the insufficient reduction of harmonics in the chromagrams. Therefore, we believe that a better front end would help recognise those chords more reliably.

In the following chapter we investigate two ways of improving the front end of our model presented here: by learning chroma profiles, and by applying a soft transcription approach to produce better chromagrams.

# 5 Improving the Front End

An important part of a chord transcription method is the interface between the audio and a high-level model, usually referred to as the front end. Since the front end input features of our DBN model are chroma vectors, we want to investigate whether changes in the way they are modelled in the high-level model or changes in the chroma vector itself can result in better chord recognition. Recall that one of the main weaknesses of the model presented in Chapter 4 is the chord confusion between chords that are musically dissimilar. This was especially noticeable in chords that have no fifth above the bass note (`maj/3`, `aug`, `dim`): these were often falsely recognised as chords that do have a fifth above the bass note but are otherwise similar[1]. Our interpretation of this result is that the bass note's third harmonic, which causes a spectral peak an octave and a fifth above the bass note, often leads to an incorrect interpretation of these chords. More generally, the presence of harmonics is not negligible. The most common chords (`maj`, `min`) suffer less from this problem, since the first few harmonics of the bass note coincide with chord notes. However, for good chord transcription, rare chords should also be recognised correctly as they often add decisive detail to a song. Therefore, we believe that raising recognition rates of rare chords while maintaining high accuracy of more frequent chords would greatly benefit an automatic transcription—and not only for overall accuracy in the MIREX sense. Our main efforts shall be directed at finding a better fit between the high-level model and the low-level features to accommodate the presence of harmonics. Without changing the overall model topology as presented in Chapter 4, we can follow two different paradigms: firstly, we can modify the model by training its parameters, and secondly, we can modify the chroma generation procedure such that the model assumptions are met more closely.

Let us consider first the problem of training model parameters. Due to the complexity of the model presented in Chapter 4, the number of parameters that need to be estimated to fully train the DBN is large and requires a large amount of training data. For example, the chord node

[1] For example, `Cmaj/3` recognised as `Emin`, see Table 4.2.

$C_i$ (see Figure 4.2) alone has more than a million parameters: $121 \times 4 \times 24 \times 121 = 1405536$ (for the previous chord, current metric position, current key and current chord). Even when considering tying of parameters, the available ground truth data is still comparatively scarce. We do have audio-synchronised ground truth of chords and keys, and additionally metric position data (Mauch et al., 2009a) for a large subset of the songs used in the 2009 MIREX chord detection task. However, the key change ground truth is not sufficient for training since key changes are rare in songs, and no aligned ground truth is available for bass notes. Nonetheless—especially in the light of the results in Chapter 4 mentioned above—it seems appropriate to learn a subset of the DBN parameters: the chroma nodes, which connect the high-level model with the chromagrams, and the chord transition rate given the metric position. The aligned chord and metric position ground truth data can be used to train these parts of the model. We describe how we proceed in Section 5.1.

The alternative paradigm, adapting the data properties to the model assumptions, is necessarily concerned with the low-level features too. The model assumption which was not met by the chroma features described in Chapter 3 is the absence of upper partials of the notes present in the signal. In Section 5.2 we describe a simple procedure using idealised note profiles to find a note activation pattern that matches a given log-frequency spectrum most closely in the least-squares sense, using a non-negative least squares (NNLS) algorithm. This note activation pattern, a kind of approximate transcription similar to the pitch salience proposed by Klapuri (2006b, 2009), is then mapped to bass and treble chroma vectors and can directly be used in the proposed DBN of Chapter 4. The results of several variants of both approaches will be given in Section 5.3, followed by a discussion in Section 5.4. We also show that the mediocre performance of the model in terms of the “no chord” label can be improved by changing the means of the Gaussian chroma node from 1 to 0.5.

## 5.1 Training Chord Profiles as Gaussian Mixtures

As we have mentioned above, we do not have sufficient ground truth data at our disposal to fully train the DBN presented in Chapter 4. Furthermore, the experiments with trained language models, e.g. Khadkevich and Omologo (2009b), do not suggest great improvements from training chord sequences. In informal experiments, we found that using the relative frequencies of the chord types as prior probabilities in our model led to results extremely biased towards the most frequent `maj` chords. Learning the chroma profiles from the chord labels seems, then, an adequate way of improving the labelling process, because the harmonics will be automatically

| norm | cov. matrix | GMM components | | |
|---|---|---|---|---|
| | | 1 | 3 | 6 |
| $L_1$ | diagonal | 8.17 | 19.98 | 16.14 |
| | full | 5.71 | 20.55 | 17.99 |
| $L_2$ | diagonal | 8.12 | 18.52 | 16.08 |
| | full | 8.62 | **22.67** | 21.69 |
| max | diagonal | 8.82 | 18.04 | 15.59 |
| | full | 9.76 | 19.99 | 19.07 |
| standardise | diagonal | 8.25 | 13.09 | 14.96 |
| | full | 7.60 | 19.56 | 17.79 |

Table 5.1: Raw (no high-level model) beat-wise chord recall percentage, based on the 121 chords in the *full* chord class set (defined on page 78), modelled using different types of chroma normalisation and Gaussian mixture models.

included in the chord model. Since Gaussian mixture models are a standard way of training chroma data (e.g. Papadopoulos and Peeters, 2007) we choose to follow that approach too.

There are however many ways of doing so, and we perform preliminary experiments using the 121 different chord classes of the *full* chord class set, which we have already used in Chapter 4 (page 78), to determine which chroma normalisation procedure ($L_1$ norm, $L_2$ norm, maximum norm, or standardisation) and which number of Gaussian mixture components (1, 3, or 6) are best suited to our chroma data. We would also like to know whether to use full covariance matrices or diagonal ones. As is common practice, we assume transposition invariance (e.g. the profiles of `Cmin` and `Fmin` are identical up to transposition), and transpose all chromagrams such that their root note coincides with the first bin (Sheh and Ellis, 2003; Lee and Slaney, 2008; Mauch and Dixon, 2008; Ryynänen and Klapuri, 2008). In this way we estimate only one chord model for each of the 11 chord classes, which is especially important in the case of rarer chord classes like `aug` for which otherwise the training data would not suffice. The "no chord" label is trained by using the respective beat chroma vectors twelve times in all possible rotations. Our measure of quality is the beat-wise recall on the raw Bayesian classification of each beat, i.e. for every beat we calculate the density of each of the trained *Gaussian mixture model* (GMM) distributions, and the chord model with the highest density wins. The results can be seen in Table 5.1. The variant using $L_2$ norm and a GMM with three mixture components and full covariance matrices performs best. In the following, we will use these parameters to estimate the treble chroma node. To estimate the bass chroma, we only need to estimate 12 bass note chroma patterns and one "no chord" pattern. Using the argument that the first beat of chords usually features the respective bass note (see Section 4.2.4, page 74), to estimate the bass note chroma we use the first beats of all chords that feature this bass note. The bass chroma model will also differ from the treble chroma model in that we will use only a single Gaussian, which we deem appropriate due to the lower complexity of the bass note.

The results of the new models with learned chroma profiles are described in Section 5.3. The following section introduces our second approach to improving the front end: a different chroma calculation method.

## 5.2 Chroma as Pitch Class Activation from Approximate Non-Negative Least Squares Transcription

This section is concerned with a novel application of the non-negative least squares (NNLS) algorithm to the generation of better chromagrams. Let us consider the difference between what information a chromagram usually contains and what a perfect chromagram for musicological purposes would be.

Chromagrams are usually a transform (often linear) of some kind of spectrum onto twelve bins that correspond to the twelve pitch classes. Our own baseline approach from Chapter 3 is a variant of this model. While the chromagrams generated this way are highly correlated to the true pitch class set, they do not actually contain information about which pitch classes have been played, but rather about the energy across the spectrum. As has been discussed in Chapter 2.2, such chromagrams are hence corrupted by noise and upper partials, though attempts have been made to attenuate both (e.g. Gomez, 2006, Chapter 3). A perfect chromagram, for our purposes and for musicological usage, would simply be an automatically generated pitch class set: it would preserve only the pitch classes of the musical notes present at a certain time in a piece of music, with any unwanted spectral information removed, much like in Parncutt's perceptually motivated chroma tally or chroma probability. Perhaps more generally, we could call this ideal *pitch class activation*.

Furthermore, when concerned with harmony analysis the relative energy of notes present is of little use: what matters is which notes (or pitch classes) are present and which are not. Klapuri (2009) proposes a visualisation method that transforms spectral energy to a salience representation in which spectral peaks that are also fundamental frequencies have a value close to unity, and others close to zero. This note activation likelihood could be used as input for harmony extraction tools. It is however a one step transform, in which the note saliences are independent of each other.

In a different class of approaches to (approximate) transcription, the spectrum (or a log-frequency spectrum) is considered a sum of note profiles in a dictionary, and an algorithm is used to find a note activation pattern that best explains the spectrum (e.g. Abdallah and Plumbley, 2004), with some constraints. This approach differs in that it involves iterative re-weighting of the note activation values. To our knowledge, such a procedure has not been used to generate chromagrams or otherwise conduct further automatic harmony analysis. Our proposed method can be broken down into the calculation of a log-frequency spectrogram (Subsection 5.2.1), different ways of preprocessing the log-frequency spectrum (Subsection 5.2.2), and finally the application of the NNLS (Subsection 5.2.3). We call the chroma resulting from this procedure *NNLS chroma*.

### 5.2.1 Log-frequency Spectrum

We map the 2048 bins of the magnitude spectrum onto bins whose centres are linearly-spaced in log frequency, i.e. they correspond to pitch (Peeters, 2006; Müller et al., 2009; Sagayama et al., 2004; Catteau et al., 2007). As in our baseline chroma approach (see Chapter 3, page 57), these pitch bins are spaced a third of a semitone apart. The mapping is performed as a matrix multiplication of the chromagram with a transform matrix, similar to a constant-$Q$ kernel, which can be calculated in advance. The resulting log-frequency spectrum is then adjusted to the tuning as described in Chapter 3.

As motivated in in Chapter 3, we use the discrete Fourier transform with a frame length of 4096 samples on audio downsampled to 11025 Hz. The main problem in mapping the spectrum to a log-frequency representation is that in the low frequency range several log-frequency bins may fall between two DFT bins, while in high frequency regions the reverse is true. We use cosine interpolation on both scales for its simplicity despite providing smoother results than linear interpolation: first we upsample the DFT spectrum to a highly over-sampled frequency representation, and then we map that intermediate representation to the desired log-frequency representation. We estimate the intermediate representation $M_f$ at frequency $f$ as

$$M_f = \sum_{i=0}^{N_\mathrm{F}} h(f, f_i) X_i, \tag{5.1}$$

where $X_i$ is the $i^{\text{th}}$ FFT bin, and

$$h(f, f_i) = \begin{cases} \frac{1}{2}\cos\left(\frac{2\pi(f-f_i)}{f_s/N_F}\right) + \frac{1}{2}, & \text{if } |f_i - f| < f_s/N_F \\ 0 & \text{otherwise,} \end{cases} \tag{5.2}$$

where $N_F$ is the frame length and $f_s$ is the sampling frequency. We use frequencies $f$ spaced by $1/40$ of the DFT bandwidth $\delta f = f_s/N_F$. We choice of this very high oversampling rate is not problematic in terms of performance since the calculation is only performed only once to generate the mapping matrix, and not at runtime. We proceed analogously to map the intermediate representation into the log-frequency domain, only that the band-width term $\delta f$, which in Equation (5.2) was constant, $\delta f = f_s/N_F$, now becomes linear with respect to frequency. In fact, since in our case we have $n_{\text{bin}} = 36$ bins per octave, we compute $Q$ using (2.4) as

$$Q = n_{\text{bin}}/\ln 2 = 36/\ln 2 \approx 51.94,$$

and the bandwidth term is $\delta f(f) = f/Q \approx f/51.94$ (see the notes on constant-$Q$ transforms in Section 2.2.2). Hence, the transform from the intermediate representation to the log-frequency domain is

$$Y_k = \sum_f h_l(f, f_k) M_f, \tag{5.3}$$

where

$$h_l(f, f_k) = \begin{cases} \frac{1}{2}\cos\left(\frac{2\pi(f-f_k)}{\delta f(f)}\right) + \frac{1}{2}, & \text{if } |f_k - f| < \delta f(f) \\ 0 & \text{otherwise} \end{cases} \tag{5.4}$$

Using Equations (5.1) and (5.3), the transform can then be calculated directly as

$$Y_k = \sum_f h_l(f, f_k) \sum_{i=0}^{N_{\text{F}}} h(f, f_i) X_i, \tag{5.5}$$

and it can thus be implemented as a matrix multiplication, which is computationally efficient. This calculation is performed on all frames of a spectrogram, yielding a log-frequency spectrogram $Y = (Y_{k,m})$. The tuning of the piece is now estimated from the phase shift of the sum of the frames of the spectrogram in exactly the same way as previously described in Equation (3.8) on page 59. Then the matrix is updated via linear interpolation, such that the centre bin of every semitone corresponds to the tuning frequency. The updated log-frequency spectrogram $Y$ has 256 $^1\!/_3$-semitone bins, and is hence much smaller than the original spectrogram. The reduced size enables us to model it efficiently as a sum of idealised notes, as will be explained in Subsection 5.2.3.

### 5.2.2 Pre-processing the Log-frequency Spectrum

Since the NNLS dictionary, which will be introduced in the next subsection, does not have explicit noise profiles, it seems natural to perform a noise reduction step on the log-frequency spectrogram $Y$ prior to the approximate transcription procedure. The noise reduction procedure we choose is subtraction of the background spectrum (Catteau et al., 2007): we calculate the running mean $\mu_{k,m}$ at every note bin $Y_{k,m}$ with a note Hamming-weighted neighbourhood window spanning half an octave around the note. The values at the edges of the spectrogram, where the full window is not available are set to the value at the closest bin that is covered. Then, the $\mu_{k,m}$ is subtracted from $Y_{k,m}$, and negative values are discarded. This results in the new spectrum

$$Y^{\text{SUB}}_{k,m} = \begin{cases} Y_{k,m} - \mu_{k,m} & \text{if } Y_{k,m} - \mu_{k,m} > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{5.6}$$

We observed that additionally dividing by the respective running standard deviation $\sigma_{k,m}$, a procedure similar to spectral whitening (e.g. Klapuri, 2006b), produces visually more pleasing chromagrams. The two procedures, subtraction of the background spectrum and division by the standard deviation, amount to a local standardisation: the data has mean 0 and standard deviation 1. Here also, negative values are discarded. This results in the new spectrum

$$Y^{\text{STD}}_{k,m} = \begin{cases} \frac{Y_{k,m} - \mu_{k,m}}{\sigma_{k,m}} & \text{if } Y_{k,m} - \mu_{k,m} > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{5.7}$$

Although the process—in particular the division—clearly destroys connections to the physical properties of the signal, the underlying motivation is that spectral amplitude differences in the music are evened out, and the influence of noise and timbre is hence reduced. Whether this has an impact on the chord extraction performance in our model will be considered in Section 5.3. Note that the background spectrum and the running standard deviation can both be calculated very efficiently using FFT-based convolution.

### 5.2.3 Note Dictionary and Non-negative Least Squares

In order to decompose a log-frequency spectral frame into the notes it has been generated from, we need two basic ingredients: a note dictionary, describing the assumed profile of (idealised) notes, and an inference procedure to determine the note activation patterns that result in the closest match to the spectral frame.

Applying Equation (5.3), we generate a dictionary of idealised note profiles in the log-frequency domain with geometrically decaying overtone amplitudes

$$a_k = s^{k-1} \tag{5.8}$$

where the parameter $s \in (0, 1)$ influences the spectral shape: the smaller the value of $s$, the weaker the higher partials. Gomez (2006) uses the parameter $s = 0.6$ for her chroma generation, and we have used $s = 0.9$ (if in a slightly different context, see Section 3.1.2). We will test both possibilities. We add a third possibility motivated by the fact that resonant frequencies of musical instruments are fixed (Rossing, 1990), and hence partials of notes with higher fundamental frequency are less likely to correspond to a resonance: here, $s$ is linearly spaced between $s = 0.9$ for the lowest note and $s = 0.6$ for the highest note. In each of the three cases, we create tone patterns over seven octaves, with twelve tones per octave: a set of 84 tone profiles. Note that we do not model three semitones here because we assume that the notes we want to transcribe are approximately in tune. The fundamental frequencies of these tones range from A0 (at 27.5 Hz) to G♯6 (at approximately 3322 Hz). Every note profile is normalised such that the sum over all the bins equals unity. Together they form a matrix $\mathcal{T}$, in which every column corresponds to one tone. We assume now that the individual frames of the log-frequency spectrogram $Y$ are generated approximately as a linear combination

$$Y_{\cdot,m} \approx \mathcal{T} z \tag{5.9}$$

of the 84 tone profiles. The problem is to find a tone activation pattern $z$ that minimises the Euclidian distance

$$||Y_{\cdot,m} - \mathcal{T} z|| \tag{5.10}$$

between the linear combination and the data, with the constraint $z \geq 0$, i.e. all activations

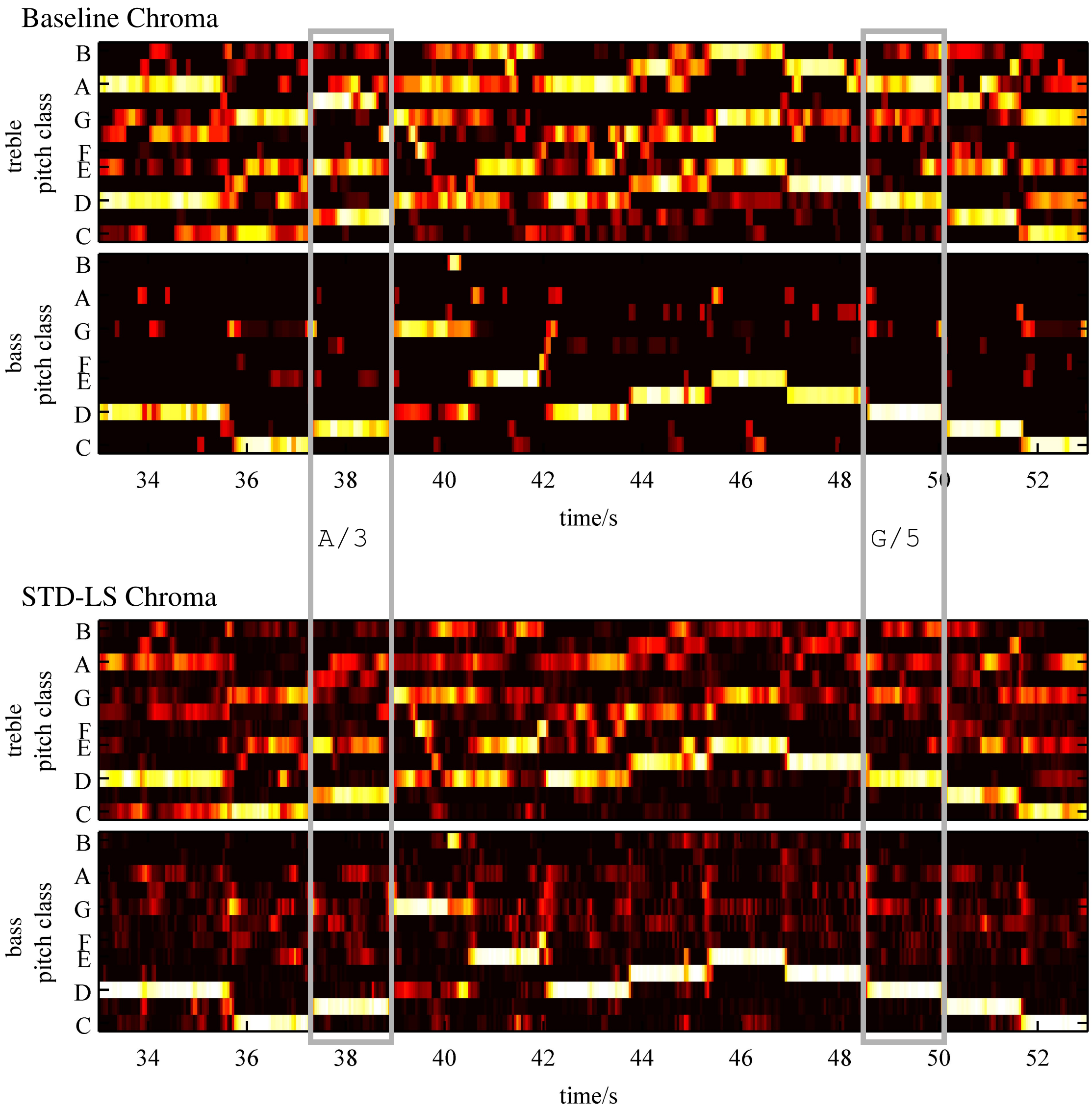


Figure 5.1: The baseline chroma (top) and the new NNLS-based chroma representation (STD-LS) proposed in this chapter, illustrated using the song “Friends Will Be Friends” (Deacon/Mercury). Two main differences are highlighted: in the left box, the true chord is `A/3`, i.e. it has the pitch classes A, C♯, and E, and a C♯ in the bass. Due to the strong third (and sixth etc.) partial of the bass note, the upper chromagram falsely suggests the presence of a G♯. The new chroma calculation method attenuates this tendency. Similarly, in the right box, the `G/5` chord is unrecognisable in the baseline chroma representation, because the bass note D generates harmonics at A. In the proposed chroma representation, this is attenuated.

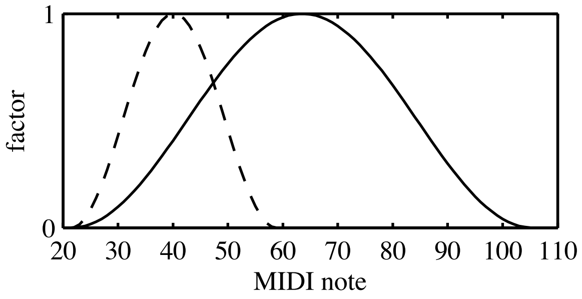


Figure 5.2: Bass and treble profiles: compared to Figure 3.3 the treble profile is now spread over the whole note range with an emphasis in the mid range, while the bass profile occupies a similar region to the one shown in Figure 3.3. The wider treble profile is used so that all active pitch classes are included.

must be non-negative[2]. This is a well-known mathematical problem called the non-negative least squares (NNLS) problem (Lawson and Hanson, 1974). To find a solution, we use the MATLAB implementation[3] of an algorithm proposed by Lawson and Hanson (1974). Note that since we do not aim to perform actual transcription, sparseness constraints (Abdallah and Plumbley, 2004) are less important. The effect that the NNLS approach has on chromagrams is illustrated in Figure 5.1 by comparing them to the baseline chromagrams.

Unlike the salience calculations performed in Chapter 3, these tone activation patterns may feature a certain pitch class only in lower frequency regions—ideally just like it is played. The nearly mutually exclusive note separation between bass and treble chroma (see Figure 3.3) makes less sense in this setup, since we want the treble chroma to represent the chord type and hence all active pitch classes including the bass pitch classes, whereas the bass chroma will be used to determine the chord inversion. We therefore choose different profiles (see Figure 5.2), in which the bass profile remains in the low tone range, but the treble profile encompasses the whole note spectrum, with an emphasis on the mid range. Beat-synchronisation and normalisation are performed exactly as expained in Section 3.2.

## 5.3 Experiments and Results

We calculated chord transcriptions using the inference algorithm and data as described in Section 4.3.1. The methods using trained chroma and chord change probabilities were evaluated in five-fold cross-validation on the 2009 MIREX data set. We use the best-performing, most complex model from Chapter 4 for the evaluation, with the *full* chord alphabet. Since we are

[2] The non-negativity is a reasonable assumption, since notes played by physical instruments necessarily have non-negative amplitudes.

[3] http://www.mathworks.com/access/helpdesk/help/techdoc/ref/lsqnonneg.html

| chord set | trained *MBK* | *MBK* |
|---|---|---|
| *majmin* | 72.4 | 72.1 |
| *inv* | 69.6 | 72.0 |
| *full* | 70.6 | 73.0 |

(a) MIREX-style evaluation using the *majmin* chord class set

| chord set | trained *MBK* | *MBK* |
|---|---|---|
| *majmin* | (61.0) | (61.3) |
| *inv* | (51.1) | (57.8) |
| *full* | 41.8 | 56.7 |

(b) evaluation using the *full* chord class set

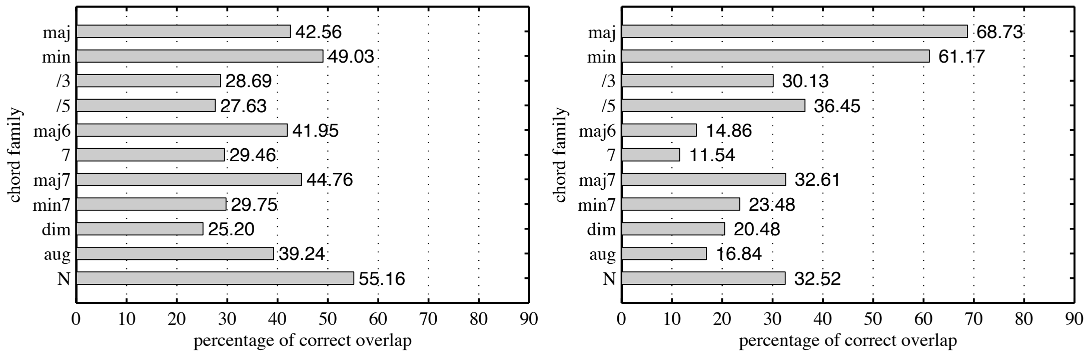
Table 5.2: Models with trained chroma: overall relative correct overlap. In Table 5.2b, the results of models that do not output as many classes as are tested are given in brackets. Since they never transcribe complex chords, their performance is high only due to their better performance on more common chords.

interested also in whether learning less detailed data influences the result, we also learn and test using smaller chord class sets (chord alphabets), namely *majmin* and *inv* (see Section 2.3.2).

### 5.3.1 Models with Trained Chroma

The MIREX-style results of the trained models are shown in Table 5.2. We can observe that overall accuracy is good but not exceptional. The trained *majmin-MBK* model has a slightly increased accuracy of 72.4% compared to the baseline model (*majmin-MBK* at 72.1%), but accuracy in the full model has decreased by two percentage points. Table 5.2b shows that evaluating by the *full* chord class set, all trained models perform worse. Figure 5.3 provides an indication of the reason why this has happened: accuracy for rare chords such as `aug`, `dim`, and `maj6`, has risen steeply (compare with Figure 4.12). The "no chord" label `N` in particular has greatly improved and now has the highest recall. The figure shows that, the accuracy is indeed reasonably balanced over all chord types. While this is a positive result and shows that the training has indeed improved the recognition of chords that were badly recognised by the baseline model, the more frequent chords `maj` and `min` have worse recall figures compared to the baseline method. Since these chords are far more frequent than the chords for which recognition has improved, overall recognition has decreased. To analyse the nature of the errors better, in Subsection 5.3.3 we will examine some typical chord confusion matrices and compare them to those retrieved using the NNLS chroma.

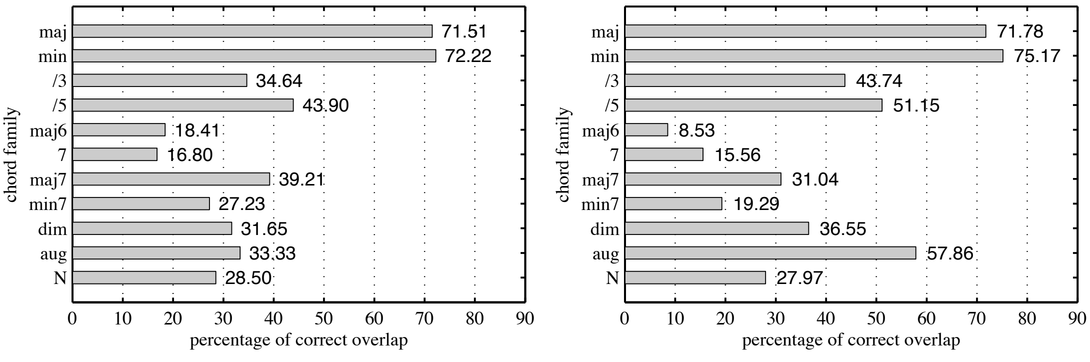
Note that while giving a better MIREX-style result, the trained *majmin-MBK* model is not actually capable of conveying musically detailed data. It recognises well all the chords subsumed under the `maj` label, but should a musician want to replicate the song from the chord labels given, it would be possible only for very simple songs that do not feature other chord labels. This is, of course, also true for the un-trained *majmin* models in Chapter 4, and previous chord recognition methods presented in Chapter 2.2. In that light, the *full* models would seem



(a) trained *full-MBK*

(b) un-trained baseline *full-MBK*

Figure 5.3: Model with trained chroma and the baseline model for comparison: relative overlap for the individual chord types, using *MBK* models with trained chroma nodes (Section 5.1).



(a) STD-0.6

(b) STD-LS

Figure 5.4: Methods with NNLS chroma: relative overlap for the individual chord types, using the *full-MBK* model proposed in Chapter 4.

like the better choice because they offer more musical detail.

### 5.3.2 Methods Using NNLS Chroma

We compare the use of NNLS chroma retrieved from three different versions of log-frequency spectra discussed in Section 5.2.2:

**O** the unprocessed, original log-frequency spectrum $Y$ (Equation 5.5),

**SUB** the log-frequency spectrum $Y^{\text{SUB}}$, after subtraction of the background spectrum (Equation 5.6), and

**STD** the standardised log-frequency spectrum $Y^{\text{STD}}$ (Equation 5.7).

The second variable we test is the partial roll-off parameter $s$ from Equation (5.8): it is either 0.6 for all notes, 0.9 for all notes, or linearly spaced (LS) between 0.9 and 0.6. We refer to the

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| `maj` | 16 | | | | | | | 26 | | | | |
| `min` | | | | | 1 | | | 2 | | 1 | | |
| `maj/3` | 2 | | | | | | | | | | | |
| `maj/5` | 44 | | | | | | | | | | | |
| `maj6` | 1 | | | | | | | 2 | | | | |
| `7` | 1 | | | | | | | | | | | |
| `maj7` | 1 | | | | | | | | | | | |
| `min7` | | | | | | | | | | 1 | | |
| `dim` | | | | | 1 | | | | | | | |
| `aug` | | | | | | | | | | | | |
| `N` | | | | | | | | | | | | |

Table 5.3: `maj/5` chord confusion in the *full-MBK* model with NNLS chroma (STD-0.6). The columns represent semitone difference from the root of the reference chord (modulo 12). The confusion values are given as percentages, rounded to the nearest integer. Confusion values $< 0.5$ are not printed for clarity. The percentage of the correct chord is printed bold.

combinations by linking the parameter names; for example, the combination of the standardised log-frequency spectrum with roll-off parameter $s = 0.9$ would be named STD-0.9. All nine variants of the NNLS chroma are used for inference with the DBN proposed in Chapter 4. At 79% the MIREX-style score of the STD-0.6 model performs best, and around five percentage points better than the equivalent model using the baseline chroma (Chapter 4). It also performs significantly better (ANOVA value is $p < 0.001$) than the algorithm proposed by Weller et al. (2009) which at 74% was the best-performing algorithm in the 2009 MIREX Chord Detection task (train/test). As we can see from Table 5.4a, the methods using no standardisation or only subtraction of the background spectrum perform considerably worse than the best method; the variants O-0.6 and SUB-0.6 are still better than the baseline method.

The evaluation using the *full* chord class set offers a more differentiated view. Let us consider Figure 5.4a: compared to the baseline method's results displayed in Figure 5.3b every individual chord type achieves higher performance, the only exception being the "no chord" label, whose performance decreases by about four percentage points. The increased performance of the other chords is particularly impressive for `aug` chords (16 percentage points), `min` and `dim` chords (both 11 percentage points), and `/5` and `maj7` chords (both seven percentage points). Consider, for example, the confusion of `/5` chords in Table 5.3: not only has the accuracy risen (44%, was 36%), but also the unwanted confusion with the 7-`maj` chord has been reduced from 30% to 26%. Our aim to improve in particular the chords which do not feature a fifth above the bass note has been achieved.

| log-freq. | parameter $s$ | | |
|---|---|---|---|
| spectrum | $s = 0.6$ | $s = 0.9$ | LS |
| O | 74.8 | 66.5 | 70.8 |
| SUB | 73.5 | 65.9 | 69.7 |
| STD | **78.8** | 74.2 | 76.7 |

(a) MIREX-style evaluation using the *majmin* chord class set

| log-freq. | parameter $s$ | | |
|---|---|---|---|
| spectrum | $s = 0.6$ | $s = 0.9$ | LS |
| O | 52.6 | 45.4 | 49.2 |
| SUB | 56.8 | 50.8 | 53.9 |
| STD | 61.1 | 60.6 | **61.8** |

(b) evaluation using the *full* chord class set

Table 5.4: Relative correct overlap for the methods using NNLS chroma (all extracted using the *full-MBK* model from Chapter 4). In all experiments the NNLS chroma using the standardised log-frequency spectrum achieves the highest scores. In the MIREX-style evaluation, the highest result comes from the standardised log-frequency spectrum with parameter $s = 0.6$, in the evaluation on the *full* chord class set, the LS parameter setting achieves better relative correct overlap, but still using standardisation.

We compare some of the confusion matrices of the STD-LS method to those of the model with trained chroma nodes in the following subsection.

### 5.3.3 A Comparison of Chord Confusion Matrices

The overall accuracy does not always provide a comprehensive picture of the performance of an algorithm. We can demonstrate more subtle advantages and disadvantages of the two competing approaches discussed in this chapter by considering selected confusion matrices. We have already seen that the accuracy of `maj` and `min` chords was only moderate in the trained models, but no chord types had very low accuracy (Figure 5.3). On the other hand, the methods using NNLS chroma had good overall performance in the MIREX sense due to several very accurately-recognised chord types, but at the price of a few chord types with low accuracy (Figure 5.4a).

Let us first examine the confusion matrices of the `maj` chord as given in Table 5.5: here, the method using the NNLS chroma (STD-LS) obviously performs better. The errors of the trained model are widely spread over incorrect chords with the same root. Unfortunately, chords that have a simple structure are often falsely recognised as more complex chords, for example 0-`maj` as 0-`7`[4]. This can falsely suggest a different function of the chord, in the case of the `7` chord, a dominant function (Rawlins and Bahha, 2005). Another possible explanation is missing detail in the reference annotations.

It is then interesting to find out what happens in the converse case, the confusion table of the `7` chord (Table 5.6). Clearly, the NNLS chroma has only 16% correctly transcribed `7` chords

[4] We have listened to several songs and found that this tends to happen when the melody features a non-chord note that briefly makes the chromagram “look” like the respective wrong chord.

compared to 29% correctly transcribed by the trained model. This means that the trained model will more often correctly indicate a dominant function of a chord. The transcriptions using the NNLS chroma do not tend to make "un-musical mistakes" as much as were done by the trained chroma in the case of the maj chord: here, the distribution of falsely recognised 7 chords is concentrated on the maj chord of the same root. Although not ideal, this is musically acceptable because the maj chord is a subset of the 7 chord and *may* have the same function.

The models using NNLS chroma inherit from the baseline model the tendency to transcribe a complex chord as its less complex relative, described in Section 4.4.2, but with boosted recognition rates. This is musically more acceptable than falsely transcribing a chord as more complex than the ground truth. The balanced performance of the trained models on both simple and complex chords is marred because it tends to do just that. Apparently, the implicit bias towards simpler chords induced by the key node in our DBN (Section 4.2.2) does not suffice to prevent these false classifications. We discuss other implications and possible solutions to this problem in Section 5.4.

### 5.3.4 An Additional Experiment

The good recognition results for the "no chord" label in the model using the trained chroma profiles encouraged us to investigate why the recognition of these had been worse in the baseline models. We decided to take a closer look at where it had been correctly recognised, and where it had not. Both the baseline method from Chapter 4 and the methods using NNLS chroma usually failed to detect the "no chord" segments in the middle of a piece. It was only the silent parts at the beginnings and ends of pieces, and their immediate surroundings that were correctly transcribed as N. Clearly, the N chords do not behave as we assumed in Section 4.2.5 (see page 76): the corresponding treble chroma vector is not flat enough, and in the normalisation process we therefore obtain chroma values that are not necessarily situated close to unity. Changing the mean from unity to a lower value in the Gaussian distribution of N chord model in the treble chroma node (see Subsection 4.2.5) arose as a simple way to remedy this behaviour. We chose a value of 0.5 and tested a thus modified *full-MBK* model with the new chroma features (STD-0.6). As we had hoped, the model maintained a high recognition rate for the chords, but also recognised 70% of the duration of N labels (see Figure 5.5), which is considerably better than without the modification (28%), but also better than the trained model (55%). Here the improvement of individual chord recognition is true for all chord types, including the "no chord" label N. The overall MIREX-style result is 80% correct overlap (*full* chord class set 63%).

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | **72** | | 1 | | | 2 | | 2 | | | | |
| min | 3 | | | | | 1 | | 1 | | 2 | | |
| maj/3 | 1 | | | | | | | | | | | |
| maj/5 | 5 | | | | | 3 | | | | | | |
| maj6 | 1 | | | | | | | | | | | |
| 7 | 3 | | | | | | | | | | | |
| maj7 | | | | | | | | | | | | |
| min7 | | | | | | | | | | | | |
| dim | | | | | | | | | | | | |
| aug | | | | | | | | | | | | |
| N | | | | | | | | | | | | |

(a) NNLS chroma (STD-LS)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | **43** | | 1 | | | 1 | | 1 | | | | |
| min | 4 | | | | | 1 | | 1 | | 1 | | |
| maj/3 | 3 | | | | | | | | | | | |
| maj/5 | 6 | | | | | 4 | | | | | | |
| maj6 | 8 | | | | | | | | | | | |
| 7 | 8 | | | | | | | | | | | |
| maj7 | 7 | | | | | 1 | | | | | | |
| min7 | 2 | | 1 | | | | | | | 1 | | |
| dim | | | | | | | | | | | | |
| aug | 2 | | | | | | | | | | | |
| N | 1 | | | | | | | | | | | |

(b) trained chroma (*full* chord dictionary)

Table 5.5: Comparison of maj chord confusion. The columns represent semitone difference from the root of the reference chord (modulo 12). The confusion values are given as percentages, rounded to the nearest integer. Confusion values $< 0.5$ are not printed for clarity. The percentage of the correct chord is printed bold.

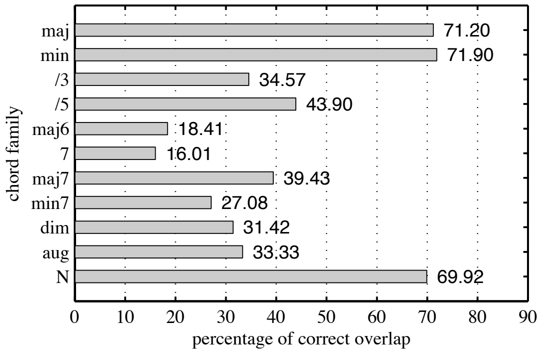


Figure 5.5: Method with NNLS chroma (STD-0.6) and the DBN with a minor modification as described in Subsection 5.3.4: relative overlap for the individual chord types.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 52 | | | | | 2 | | 1 | | | 1 | |
| min | 4 | | | | | 2 | | 5 | | 1 | | |
| maj/3 | 1 | | | | | | | | | | | |
| maj/5 | 2 | | | 1 | | 2 | | | | | | |
| maj6 | | | | | | | | | | | | |
| 7 | **16** | | | | | | | | | | | |
| maj7 | | | | | | | | | | | | |
| min7 | 1 | | | | | | | | | | | |
| dim | | | | | 1 | | | | | | | |
| aug | 1 | | | | | | | | | | | |
| N | | | | | | | | | | | | |

(a) NNLS chroma (STD-LS)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| maj | 28 | | | | | 1 | | | | | 1 | |
| min | 6 | | | | | 1 | | 2 | | | | |
| maj/3 | 2 | | | | | | | | 1 | | | |
| maj/5 | 2 | | | 1 | | 2 | | | | | 1 | |
| maj6 | 3 | | | | | | | | | | 1 | |
| 7 | **29** | | | | | | | | | | | |
| maj7 | 2 | | | | | 1 | | | | | | |
| min7 | 4 | | | | | | | 2 | | | | |
| dim | | | | | 1 | | | | | | | |
| aug | 2 | | | | | | | | | | | |
| N | 1 | | | | | | | | | | | |

(b) trained chroma (*full* chord dictionary)

Table 5.6: Comparison of 7 chord confusion. The columns represent semitone difference from the root of the reference chord (modulo 12). The confusion values are given as percentages, rounded to the nearest integer. Confusion values $< 0.5$ are not printed for clarity. The percentage of the correct chord is printed bold.

## 5.4 Discussion and Future Work

The most puzzling result presented in this chapter is the lack of overall performance increase due to statistical learning. This may certainly be due to the way we performed the training, i.e. despite using standard techniques, we did not carefully examine the interactions between all components in our model. As demonstrated by the MIREX performance of the algorithm proposed by Weller et al. (2009), a trained algorithm can perform better than our trained models presented in this chapter, and the approach of discriminative learning and the use of support vector machines may be superior to ours. It is however the only model we know of that outperforms even our baseline models presented in the previous chapter, and performs worse than our best model, using the new chroma by means of NNLS and our expert model. This may be a hint at what we believe are three principle shortcomings of the trained chroma profile model. Firstly, as the definition of a chord makes clear (page 32), a chord can develop over time, which implies that a single chord profile will not suffice to model it closely. Secondly, the chord surface features for the same nominal chord may differ substantially with the key. The only chord model that uses this information is that proposed by Raphael and Stoddard (2003) for symbolic data. Thirdly, occurrences of notes may have effects that are not modelled in any model to date. For example, an occurrence of the 7th in one beat of a `maj` chord stretching a bar can turn the chord into a dominant `7` chord. We have seen a similar situation in the example of the song "Something" in Chapter 4 (page 4.13). But an occurrence of the 6th on one beat in a similar situation would often be interpreted as a non-harmony note, and the note would have to be longer to turn the chord into a `maj6` chord.

We believe that in order to properly learn chord models, these important characteristics of chords will have to be taken into account. This may not be possible with the chord models that have been used for chord transcription (including ours) because they fuse into one chord profile what really consists of two very distinct components: the musical chord model (choice of notes given the chord, key and other factors) and the audio feature model (the surface given the choice of notes). In other words, a note (or pitch class) will be modelled by its probability of sounding in a chord model, and by the effect it has on the audio given that it is sounded. We argue that for further improvements a departure towards a more realistic model with these distinct sub-components will be vital.

Our findings provide evidence to support the intuition that the information which is lost by mapping the spectrum to a chroma vector cannot be recovered completely: therefore it is necessary to perform note transcription or calculate a note activation pattern *before* mapping the spectrum to a chroma representation (as we did in this chapter) or directly use spectral features as the input to higher-level models, which ultimately may be the more principled solution.

These new directions provide all sorts of possibilities for future work, and especially two-component (musical + physical) chord models are an exciting perspective. There is also scope in comparing how other approximate transcription methods influence chord analysis, whether through an intermediate chroma mapping or directly on some spectral representation. Finally, the combination of transcription techniques and statistical learning could provide further improvements to chord extraction, if the topology of the chord models were more adequate.

## Conclusions

In this chapter we have presented two substantially different approaches with the aim of improving the front-end interface of our DBN proposed in Chapter 4: statistical training of the chroma nodes in the model, and an enhanced chroma extraction technique that performs a preliminary transcription step to match the model better. The best results were achieved by one of the methods using the enhanced NNLS chroma extraction technique, reaching 79% MIREX accuracy (80% with an additional minor modification of the DBN). This is a significant improvement over the state of the art (MIREX 2009 Chord Detection Task). We have shown that statistical learning of chroma can boost the recognition rate of individual chord types. In our implementation, this came at the cost of lower overall accuracy because many frequent chords were misclassified.

We have compared in detail confusion matrices obtained from NNLS chroma and the trained chroma models, and have found that while the trained variants can be better for the annotation of some particular chords, they tend to go wrong in less musically accepted ways than the NNLS models. These inherit the musical "conservativeness" from the baseline method, and often provide acceptable (simpler) approximations to the true chord.

We have also discussed some more conceptual issues, arguing that in the future the general approach to chord models will have to model what notes are played, and how these affect the surface features separately. The success of our NNLS chroma has also given us confidence that the information lost in simple chroma mapping is not negligible, and that further research should go into using approximate transcription methods for chord extraction. We have derived several areas of future work, that extend the approximate transcription approach presented in this chapter and combine it with statistical learning.

This chapter has considered some methods that improve chord transcription results through changes in the low-level processing front end of the baseline model presented in Chapter 4. In the following chapter we draw attention to the highest organisational level in a piece of music: the form. We will investigate its importance for chord transcription, and, in particular, what gains in transcription performance can be achieved through knowledge of repeated sections in a piece of music.

# 6 Using Repetition Cues to Enhance Chord Transcription

In this chapter we propose a further step towards unified music analysis by using a global feature of music to enhance chord estimation: the repetition of song segments. The work is based on a published collaboration with Katy Noland (Mauch et al., 2009c), in which she contributed background knowledge on structural segmentation, further technical ideas and analysis of results. The proposed technique allows us to generate more authentic chord transcriptions than previously possible with automatic methods. This is achieved by averaging the low-level features of repeated sections of a piece of music, thus attenuating non-systematic deviations and noise. The structural segmentations needed in this process are provided either by manual annotation or by a novel structural segmentation method tailored to the task. We show that in both cases the improvements are twofold: due to the cleaner features, chord extraction performance increases significantly; furthermore, repeated segments of a song are transcribed with identical chord progressions (unless the context around the segment suggests otherwise), a characteristic that would be expected from a manually transcribed lead sheet.

We have already seen how musical context has been used to improve chord recognition (Section 2.2.3 and Chapter 4). However, none of the previous methods take into account the global structure of a piece of music. In fact, hidden Markov models, and related dynamical models such as our DBN (Chapter 4), are intrinsically ill-suited to modelling global structures since the direct temporal dependencies are assumed to be Markovian, i.e. only local direct dependencies are allowed (Equation (2.5) on page 44). On the other hand, Dannenberg (2005) shows that a music computing algorithm (beat tracking) can be greatly improved by the use of knowledge about the musical structure. To our knowledge this principle has not yet been applied to chord estimation.

Since much of musical structure is defined by repetition, which is a core principle in music (Huron, 2006, p. 229), exploiting repetition structures seems to be a valid starting point for research in this direction. Moreover, in popular songs in particular, a repeated *verse-chorus* format is common, in which the chord sequence is the same in all sections of the same type. In lead sheets then—for better readability—these sections would usually only be notated once, with repeats indicated. Our method mirrors this improvement by assigning the same chord progression to repeated sections. In addition, having found repeating sections, we have available several instances of a given chord sequence from which to estimate the chords, so we expect an improvement in estimation accuracy. We demonstrate the improvements in readability and accuracy using manually-annotated descriptions of the musical structure, and show that the improvement can also be achieved using an automatic structure annotation algorithm tailored to the task.

In Section 6.1 we describe work related to structural segmentation. In Section 6.2 we present a new segmentation technique that is tailored to our task of finding repeated chord sequences. In Section 6.3 we describe the technique that integrates this repetition information into the chord transcription process. In the results section 6.4 we first give examples of chord estimation with and without the segmentation technique (Section 6.4.1), and then present quantitative chord estimation results (Section 6.4.2). In Section 6.5 we discuss our findings.

## 6.1 Related Work on Structural Segmentation and Motivation

Previous automatic music structure extraction techniques are usually based on either of two ways of defining a structural segment: firstly a structural segment can be defined as a segment in which some parameter of interest, for example timbre, remains stable; secondly it can be defined as a segment that has a characteristic sequence of events, for example a chord progression.

Methods of the first kind primarily search for section boundaries, indicated by a sudden change in the feature of interest. This can be timbre (Aucouturier et al., 2005; Levy and Sandler, 2008; Abdallah et al., 2005; Chu and Logan, 2000), spectral evolution (Peeters et al., 2002), rhythm (Jensen et al., 2005), or combinations of features (Jensen, 2007; Maddage, 2006). A common approach is to cluster together frames that are similar, then label contiguous similar frames as a segment. Often several clustering methods are used in series, including hidden Markov models (Peeters et al., 2002; Chu and Logan, 2000; Abdallah et al., 2005; Levy and Sandler, 2008), k-means (Peeters et al., 2002) and simulated annealing (Abdallah et al., 2005). For chord transcription, timbre and rhythm are not very important, and often a repeated verse will intentionally differ in terms of timbre and rhythm from the first verse. The chord progression is usually unchanged between different verses (or choruses etc.).

Hence, an approach that searches for repeated progressions (Rhodes and Casey, 2007; Bartsch and Wakefield, 2005; Müller and Kurth, 2007; Goto, 2003; Lu et al., 2004; Paulus and Klapuri, 2006) is more appropriate for our purposes. Methods using this paradigm rely on a self-similarity matrix (Foote, 1999), which is a symmetric, square matrix that contains a measure of the similarity between every pair of frames. Repeated sections appear as parallel diagonal lines, and can be extracted with some post-processing, such as the application of a low pass filter to reduce noise (Bartsch and Wakefield, 2005) or erosion and dilation techniques to eliminate noise (Lu et al., 2004), followed by a thresholding operation to find highly similar sequences of frames. Recently, Paulus and Klapuri (2009) have shown that the state-based and sequence-based approaches can be combined, and Peeters and Deruty (2009) have proposed a multi-dimensional annotation format for musical segments.

Since these approaches are usually aimed at structural segmentation for its own sake, they do not restrict segments to be of equal length. For instance, if the third chorus in a song is extended by one bar compared to the previous ones, a general structural segmentation algorithm should ideally recognise all three as *chorus*. This kind of variation, however, could not be handled by our current algorithm, as we will see in Section 6.3. We require repetitions that approximately match with respect to similarity and *exactly* match with respect to length in beats. Only then will the chromagrams of the respective repeats neatly fit onto each other. Using beat-synchronous chromagrams is essential for two reasons: firstly, our baseline chord transcription model from Chapter 4 requires beat-synchronous chroma as its input, and secondly, the beat-tracking can adjust for changing tempos between segments of a song.

This special requirement has led us to develop a new variation of a sequence-based segmentation algorithm which is similar to algorithms proposed by Ong (2006) and Rhodes and Casey (2007) and detects approximately repeated chroma sequences of equal length in beats. The next section describes this algorithm.

## 6.2 Segmentation Method

In a song, we call a chord sequence that describes a section such as the verse or chorus a *segment type*. Any segment type may occur one or more times in a song and we call each occurrence a *segment instance*. To make use of segment repetition as part of the chord estimation process, we rely on segment types whose instances are not only harmonically very similar, but also have the same length in beats.

Our automatic segmentation method has two main steps: finding approximately repeated beat-synchronous chroma sequences in a song (Section 6.2.1), and deciding which of these sequences are indeed segments, using a greedy algorithm (Section 6.2.2). The section ends with a description of the process of obtaining similar structural segmentation based on hand-annotated audio-aligned segment names (Section 6.2.3).

### 6.2.1 Finding Approximately Repeated Chroma Sequences

We calculate the Pearson correlation coefficients between every pair of the *wide* beat-synchronous chroma vectors described in Section 3.3. More precisely, given two chroma vectors $\mathbf{c}$ and $\mathbf{e}$ we calculate the Pearson correlation coefficient

$$r = \frac{\sum(c_i - \bar{c})(e_i - \bar{e})}{(12-1)\cdot s_c s_e} \in [-1, 1], \tag{6.1}$$

where $s_c$ and $s_e$ are the standard deviations of the elements of the respective chroma vectors. The matrix of correlation coefficients is a beat-wise self-similarity matrix $R = (r_{ij})$ of the whole song. This is similar to the matrix of cosine distances used by Ong (2006). In the similarity matrix, parallel diagonal lines indicate repeated sections of a song. In order to eliminate short term noise or deviations we run a median filter of length 5 (typically just more than one bar) diagonally over the similarity matrix. This step ensures that *locally* some deviation is tolerated.

We perform a search of repetitions over all diagonals in the matrix over a range of lengths. We assume a minimum length of $m_1 = 28$ beats and a maximum length of $m_M = 128$ beats for a segment, leading to a very large search space. We minimise the number of elements we have to compare by considering as section beginnings only those beats that have a correlation $r$ greater than a threshold $t_r$, and assuming that section durations are quantised to multiples of four beats. We found that a value of $t_r = 0.65$ worked well.

To assess the similarity of a segment of length $l$ starting at beat $i$ to another one of the same length starting at $j$ we consider the diagonal elements

$$D_{i,j,l} = (r_{i,j}, r_{i+1,j+1}, \ldots, r_{i+l,j+l}) \tag{6.2}$$

of the matrix $\mathcal{R}$. If the segments starting at $i$ and $j$ are exactly the same, then $D_{ij}$ will be a vector of ones (and vice versa), hence we can characterise a perfect match by

$$\min\{D_{i,j,l}\} = 1. \tag{6.3}$$

In practice however, repeated segments are rarely identical. To accommodate variation arising in a practical situation, we relax the requirement (6.3) by using the empirical $p$-quantile function[1] instead of the minimum (which is the 0-quantile), and choosing a segment threshold $t_s$ lower than unity. For our purposes, the triple $(i, j, l)$ hence describes an approximate repetition, if

$$\text{quantile}_p\{D_{i,j,l}\} > t_s. \tag{6.4}$$

The two parameters $p = 0.1$ and $t_s = 0.6$ are chosen empirically. They express that we allow 10% of the beats in $D_{i,j,l}$ to have values $\leq 0.6$. In future work we would like to learn these values from the ground truth data. For every beat index $i$ and every length $l$ we obtain at least the trivial beat index $j = i$ that fulfills the requirement (6.4), and possibly others. If two segments of those found overlap, the one with the higher value of $\text{quantile}_p\{D_{i,j,l}\}$ is chosen. The remaining set of indices $j$ is then added to a list $\mathcal{L}$ of repetition sets, if it has more than one element, i.e. if it actually describes at least one repetition.

Each of the repetition sets in $\mathcal{L}$ represents a potential segment type, and its elements represent the start beats of instances of that segment type. However, there are typically many more repetition sets than there are segment types. In the following subsection we describe how we decide which repetition sets become segment types.

### 6.2.2 A Greedy Structure-finding Algorithm

To find repetition sets relating to actual segment types we use the heuristic of a music editor who tries to make a concise transcription to make orientation in the score easier, and to save space (or paper): the space saved transcribing repeated sections only once will be the number of repetitions (minus 1) multiplied by the length (in beats) of the repeated material. If $l$ is the length of a segment in a particular repetition set, and it occurs $n_r$ times, then the score is $l \times (n_r - 1)$. For example, a sequence of length 32 occurring three times would "save" $32 \times (3 - 1) = 64$ beats, while a sequence of length 48 occurring twice would save only $48 \times (2 - 1) = 48$ beats. This score can be easily calculated for all the repetition sets. The repetition set with the highest score in $\mathcal{L}$ will be selected to represent a segment type. If more than one repetition set yields the top score, then the one with the highest mean of $\text{quantile}_p\{D_{i,j,l}\}$ is chosen. All repetition sets whose segment instances overlap with the top-scoring repetition set are removed from $\mathcal{L}$. This procedure is applied to the remaining repetition sets until no repetition sets are left in $\mathcal{L}$.

[1] http://www.mathworks.com/access/helpdesk/help/toolbox/stats/quantile.html

At the end of the process there will generally be parts of the song that are not assigned to any segment type because they have not been detected as a repetitions. Each of these segments will then be treated as a separate segment type with only one segment instance.

The algorithm described above has performed well as a structural segmentation algorithm in its own right, with very good results at the 2009 MIREX Structural Segmentation task[2].

### 6.2.3 Manual Structural Segmentation

We use manual structural segmentation annotations (Mauch et al., 2009a) of 192 songs by The Beatles and Zweieck[3]. The annotators were instructed to always annotate segment boundaries at bar boundaries, but chose the segment boundaries according to their own perception. The basis for all Beatles annotations were Pollack's song analyses (Pollack, 1995). A song usually contains several segment types, some of which have multiple instances. We combined several segment instances into one segment type if they had the same manually assigned label and also the same length in beats.

The automatic and manual segmentation techniques presented in this section are the basis for a modified chord extraction technique, which will be described in the next section.

## 6.3 Using Repetition Cues in Chord Extraction

We use structural segmentation to combine several instances of a segment type in a song and then infer a single chord sequence from the combination. As a baseline we use the *full-MBK* chord transcription method presented in Chapter 4, which extracts chords from beat-synchronous treble and bass chromagrams, using a dynamic Bayesian network. In order to integrate the knowledge of repeating segments, we update the beat-synchronous bass and treble chromagrams by averaging the respective beats of all segment instances of the same segment type. For example, assume that we have a segment type with two segment instances which start at beats $s_1$ and $s_2$, respectively. The two beat-synchronous chroma vectors $C^{\text{sync}}_{s_1+t-1}$ and $C^{\text{sync}}_{s_2+t-1}$ describe the $t^{\text{th}}$ beat in the respective instances. Both will then be replaced by their arithmetic mean, i.e.

$$C^{\text{new}}_{s_1+t-1} = C^{\text{new}}_{s_2+t-1} = \frac{C^{\text{sync}}_{s_1+t-1} + C^{\text{sync}}_{s_2+t-1}}{2}. \tag{6.5}$$

[2] Five algorithms were ranked according to the frame pair clustering F-measure (Levy and Sandler, 2008), and our algorithm achieved the best score (0.60). `http://www.music-ir.org/mirex/2009/results/MIREX2009ResultsPoster1.pdf`

[3] In order to have a homogeneous test set in this chapter, we decided not to use the Queen songs used in previous chapters. The reason is that, as described in Section 6.4, we will test manually extracted beat annotations, which were not available for the Queen songs.

Both treble and bass chromagrams are processed in this fashion. Trivially, the chroma vectors in segments for which no repetition is detected remain unchanged.

This is a simple way of sharing chroma information between segment instances of the same segment type. Non-systematic chroma information such as noise and off-pitch melody notes which happen in only one of the instances is relatively reduced, while systematic chordal information which is present in all instances is retained. The updated chromagram can now be processed as proposed in Chapter 4, by using the Viterbi algorithm in the DBN.

Recall, however, our remark in Section 4.4 that memory consumption for this task was very high. As an alternative way of processing the signal we therefore propose to process every segment instance separately. This leads to a major reduction in memory consumption, since usually the songs' longest sections are much shorter than half their length, and the memory complexity of inference in DBN is linear in the sequence length (Murphy, 2002, page 8).

First, in order not to deal with very short segments (less than 12 beats) between two segment instances, the end-boundary of the first one is moved to coincide with the start-boundary of the second instance. Any short segment is thus appended to the previous segment. Then inference is performed in the usual way, but separately on each segment instance, with an additional neighbourhood of 8 beats on both sides: this neighbourhood is meant to provide additional context for the segments and to avoid mis-classification of boundary chords. Then, to form a final contiguous transcription from the individual segment transcriptions we use the transcribed states from the core part of the individual segment instances (i.e. without the neighbourhood), and re-assemble them.

In general, transcriptions of beats that share exactly the same chroma (by the process we have just described) will be more similar than the transcriptions of the same beats would have been without using repetition cues. Note however that there are temporal dependencies between beats, which potentially exist over longer stretches of audio, and it may in some cases happen that these beats are transcribed with different chord transcriptions.

## 6.4 Experiments and Results

We conducted experiments on a subset of the 2009 MIREX test set for which chord, segmentation and beat annotations were available (Mauch et al., 2009a). This subset consists of 197 songs by The Beatles and Zweieck. For the Queen songs we additionally use to evaluate our methods in Chapters 4 and 5, beat transcriptions are not yet available. We compare the influence on combinations of the following three parameters: type of segmentation (automatic, manual or none[4]), type of beat tracking (automatic, manual), and the segment-wise inference described in Section 6.3 (on or off). This leads to the following ten different configurations, including the baseline configuration as proposed in Chapter 4:

**autobeat-autoseg** automatic beats, automatic segmentation, song-wise inference

**autobeat-autoseg-segwise** automatic beats, automatic segmentation, segment-wise inference

**autobeat-manseg** automatic beats, manual segmentation, song-wise inference

**autobeat-manseg-segwise** automatic beats, manual segmentation, segment-wise inference

**autobeat-noseg** automatic beats, no segmentation (baseline), song-wise inference

**manbeat-manseg** manual beats, manual segmentation, song-wise inference

**manbeat-manseg-segwise** manual beats, manual segmentation, segment-wise inference

**manbeat-autoseg** manual beats, automatic segmentation, song-wise inference

**manbeat-autoseg-segwise** manual beats, automatic segmentation, segment-wise inference

**manbeat-noseg** manual beats, no segmentation, song-wise inference

We use manual segmentation annotations because they are expected to be more reliable than automatic ones, and hence they are expected to work even if the automatic ones do not. This reasoning applies to the manual beats too: if in a segment instance the automatic beat-tracker misses a beat or falsely detects one, the segmentation algorithm may fail to recognise it as a repeated segment because no other similar segment will have the same length. This could corrupt an otherwise positive outcome, and so we wanted to exclude that possibility by testing with manually annotated beats too. Though we are ultimately interested in the fully automatic method, we test the concept of our algorithm by using ideal segmentation and beat detection to get an upper bound on performance increase.

The rest of this section will explain the effects of the proposed methods on the chord transcriptions in two different ways. Firstly, we use four song examples to provide the reader with a qualitative idea of the effect of using repetition information in the proposed way (Section 6.4.1). This will help to explain the increase in quantitative performance and performance differences between the configurations, reported in Section 6.4.2.

[4] i.e. without updating the chromagram (Section 6.3)

### 6.4.1 Qualitative Evaluation of Four Examples

We have chosen four songs to demonstrate the effect of our method. Since for practical use the fully automatic autobeat-autoseg method is most relevant, all examples will feature this configuration: the first three examples will highlight its behaviour compared to the baseline method (autobeat-noseg), and in the fourth example we compare it to the autobeat-manseg method.

The four figures 6.1, 6.2, 6.3 and 6.4 are all structured in the same way: the top part provides an overview of the whole song, displaying the manual segmentation, the automatic segmentation (on automatically extracted beats), and then two horizontal bars each of which displays in black the times in the song where the chord transcription is correct for each of the two configurations, respectively. The bottom part then shows chord transcriptions of an interesting excerpt of that song: the ground truth transcription and the two automatic transcriptions.

**Example 1: "It Won't Be Long" (Lennon/McCartney) performed by The Beatles**

The MIREX-style evaluation for this song increased by 8.7 percentage points as a result of using repetition cues. Let us first compare manual and automatic segmentation in Figure 6.1a. Evidently, they correspond well, but the automatic segmentation algorithm has summarised the "verse" and "chorus" parts into the larger "part A". The first chorus is not preceded by a verse, and is therefore classified as not repeated by the automatic method ("part n1"). The last chorus in the manual transcription is shorter, and leads into the "outro", whereas the automatic segmentation finds that the outro actually has similar chords to the end of the chorus, so it puts the section boundary later, which is reasonable. The black bars indicate where the transcriptions using autobeat-autoseg and the baseline (autobeat-noseg) were correct according to the MIREX-style RCO score. One can clearly see that some white gaps in the baseline bar have been closed due to averaging of information between segment instances of "part A". This improvement is particularly evident between times 100s and 120s. We will consider this excerpt in more detail.

Figure 6.1b shows three chord transcriptions between times 100s and 120s. The baseline transcription is much worse than the transcription using repetition cues; in particular, many `E` chords were transcribed as `B` chords. This may be caused by strong bass notes on the pitch B, which repeatedly appear in the `E` chords. The autobeat-autoseg transcription however, does not suffer from this and recognises the excerpt completely correctly, using the information from other instances of "part A" that previously appeared in the piece. In this song, the desired improvement in accuracy due to shared information between instances of the same segment type has indeed been realised.

**Example 2: “A Hard Day’s Night” (Lennon/McCartney) performed by The Beatles**

This song is another example in which the proposed method autobeat-autoseg produces better chord transcriptions than the baseline method (+12.3 percentage points): the black bar corresponding to autobeat-autoseg in Figure 6.2a shows that more chords are recognised if segmentation information is used. Of course, this can also backfire in some places, and if we consider the segment transcribed as “part B”, we can observe that its first instance (around 50s) is actually transcribed better using the baseline method: the transcription of that part has been influenced by “bad” chromagrams from the second instance of “part B” (around 110s). In Figure 6.2b we can more clearly observe what happened at 54s: the `Emin` chord was falsely transcribed as `Cmaj7`. Generally, the method worked though and provided an overall better transcription of the song. In the next example we will see that this is not always the case.

**Example 3: “Got To Get You Into My Life” (Lennon/McCartney) performed by The Beatles**

Due to erroneous segmentation, the MIREX-style score for this song decreased by 6.5 percentage points. It is very instructive to look at the automatic segmentation. Until around 110s it is a very good match for the manual transcription. After that, however, something goes wrong: instead of recognising the refrain, “part B” is recognised as having two instances. These are not in fact repetitions, and contain different chord sequences. This causes the chord recognition to do badly, and one can see that a gap is introduced in the bar showing the correct regions of the chord transcription using segmentation (autobeat-autoseg). The more detailed Figure 6.2b shows that all chord changes have been dropped in favour of a contiguous chord `G`, the predominant chord in the second instance of (the falsely recognised) “part B”. This kind of false segmentation seems to occur relatively rarely.

**Example 4: “Liebesleid” (Kreisler) performed by Zweieck**

Finally, we encounter an example in which we compare two transcriptions that use segmentation, one automatic (autobeat-autoseg) and one manual (autobeat-manseg). Though generally the use of manual segmentation worked well (as we will see in Section 6.4.2), we want to highlight one pitfall that occurred in the extraction of the song “Liebesleid”: since the manual segmentations were not strictly meant to be referring to the chord sequence of a song segment, the person transcribing decided to summarise several segment instances with two different chord sequences with the label “bridge”. In the transcription, then, all of the chromagrams belonging to these were averaged, resulting in incorrect transcriptions of the two segment instances around 55s to 80s, those in fact transcribed as “part B” by the automatic segmentation algorithm. The detail of Figure 6.4b highlights the remarkable effect, showing completely incorrect chord labels. Conversely, some segment instances that do in fact have a common chord sequence were transcribed with different labels (“bridge” and “chorus”, towards the end of the song), where the automatic segmentation procedure correctly recognised the repetition (“part A”).

We can see that while the manual segmentation may make perfect sense in terms of how humans perceive the song, it can still be detrimental to our method if the judgement is not based on similarity of chord progressions. The automatic method does judge repetition strictly by chroma similarity and is hence less susceptible to this kind of error.

For all songs that have repeated parts—almost all—it is not only the improved chord accuracy that makes the transcriptions more helpful to a musician. It also results in more natural transcriptions because chord progressions that are repeated are transcribed identically, so could be used to generate compact lead-sheets with each segment type written exactly once.

We have demonstrated how segmentation can help create consistent and hence more readily readable chord transcriptions. In the following paragraphs we will examine their overall performance.

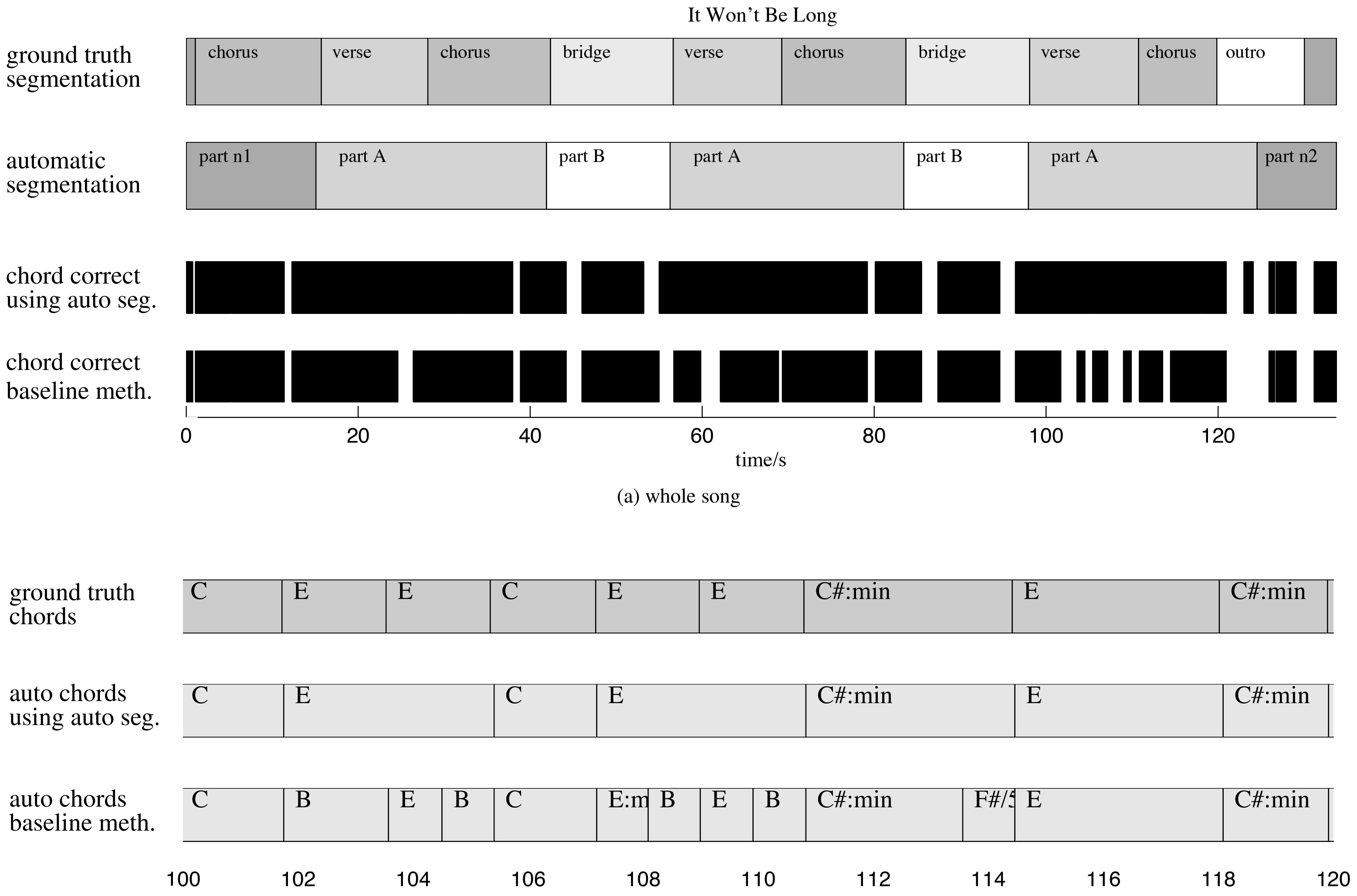
### 6.4.2 Quantitative Results

We compare the ten different combinations arising from two different beat annotations (manual and automatic) and three different segmentation annotations (manual, automatic, and none). At the end we will present an additional experiment using the fully automatic configuration autobeat-autoseg with the NNLS chromagrams we introduced in Chapter 5.

As in the previous chapters, we use two different kinds of chord class sets to examine the accuracy of the automatically extracted chord transcriptions (see Section 2.3.2): the *majmin* chord class set, which results in an evaluation equivalent to that used in the 2009 MIREX Chord Detection task, and the more detailed *full* chord class set, which distinguishes as many chord classes as we transcribe in the *full-MBK* method.

**Effect of Using Repetition Cues**

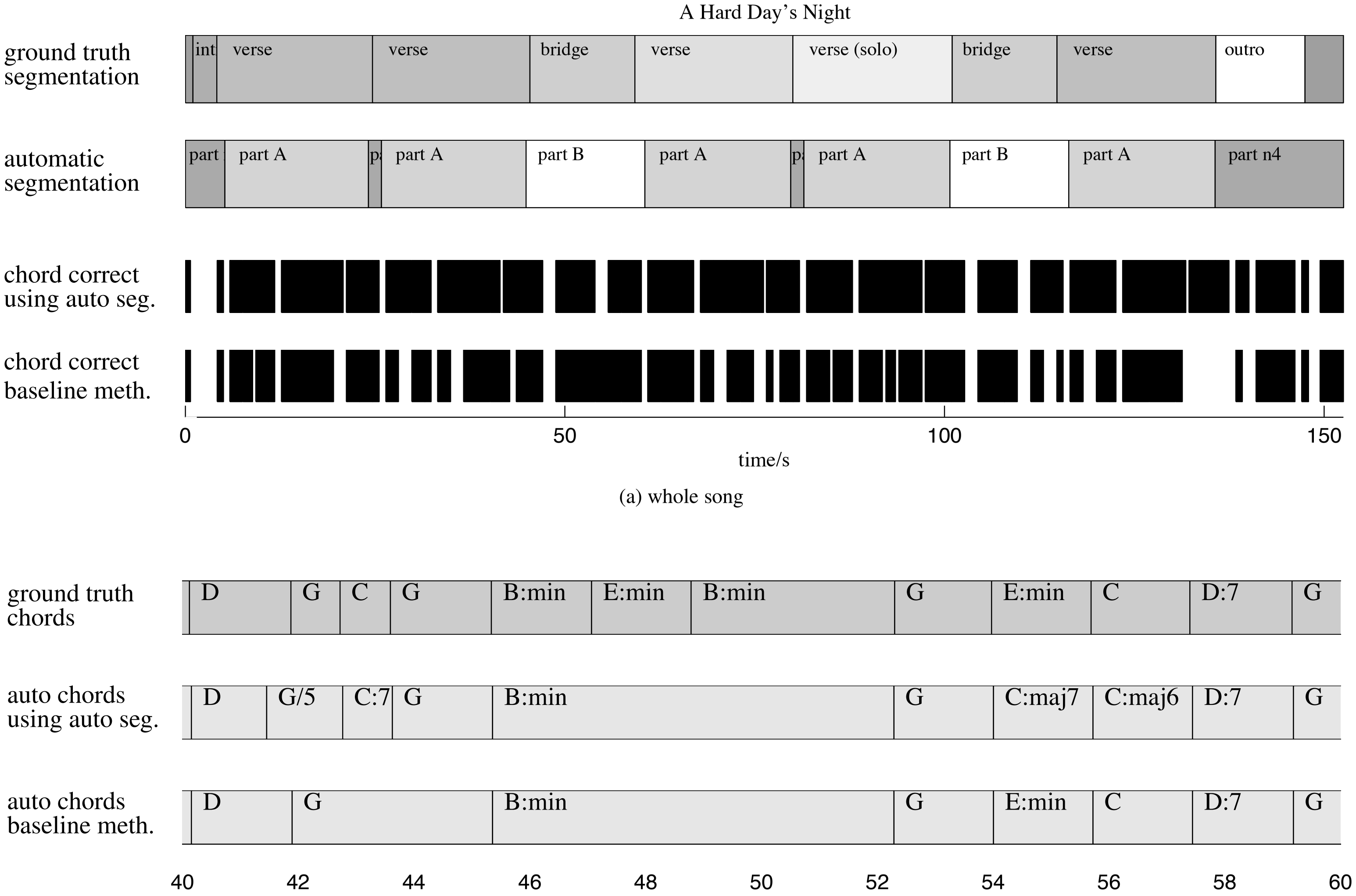
Let us first consider the effect of using the proposed method of integrating repetition information. As Figure 6.5 illustrates, all methods employing repetition information—manual or automatic—have higher MIREX-style scores than the baseline methods without segmentation. The highest score of 75.7% is achieved by the manbeat-autoseg algorithm. In fact, the Tukey-



(a) whole song

(b) excerpt between 100s and 120s

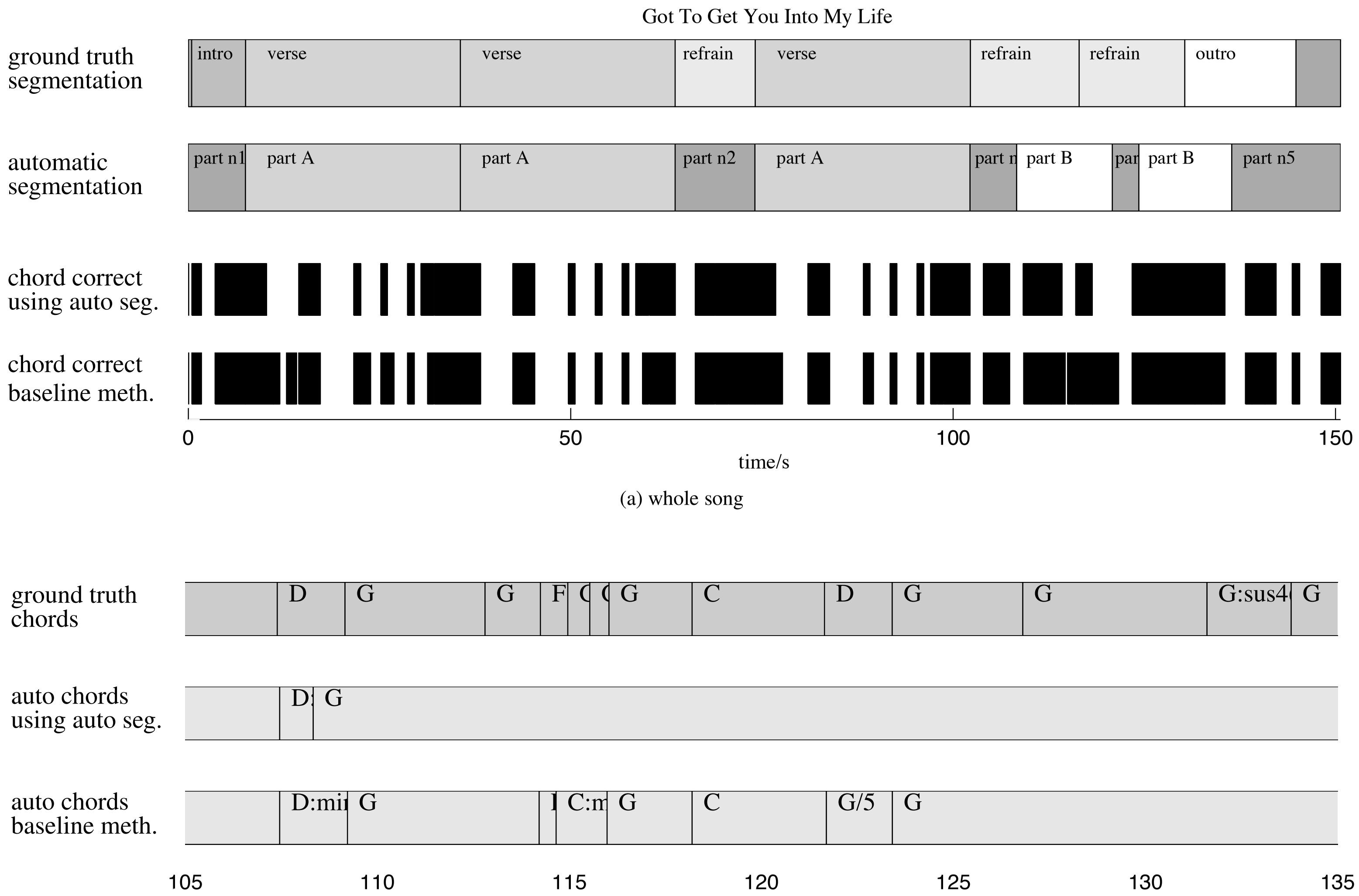
Figure 6.1: “It Won’t Be Long”. Effect of using the repetition information (see discussion in Section 6.4.1): comparing the fully-automatic autobeat-autoseg method to the baseline method that does not use repetition information. (a) the two bottom bars are black at times where the chord has been recognised correctly (using MIREX-style evaluation) over the whole song. The two top bars display the manual segmentation (for reference) and the automatic segmentation used to obtain the autobeat-autoseg results. (b) manually-annotated and automatically-extracted chord sequences for an excerpt of the song.



(a) whole song

(b) excerpt between 40s and 60s

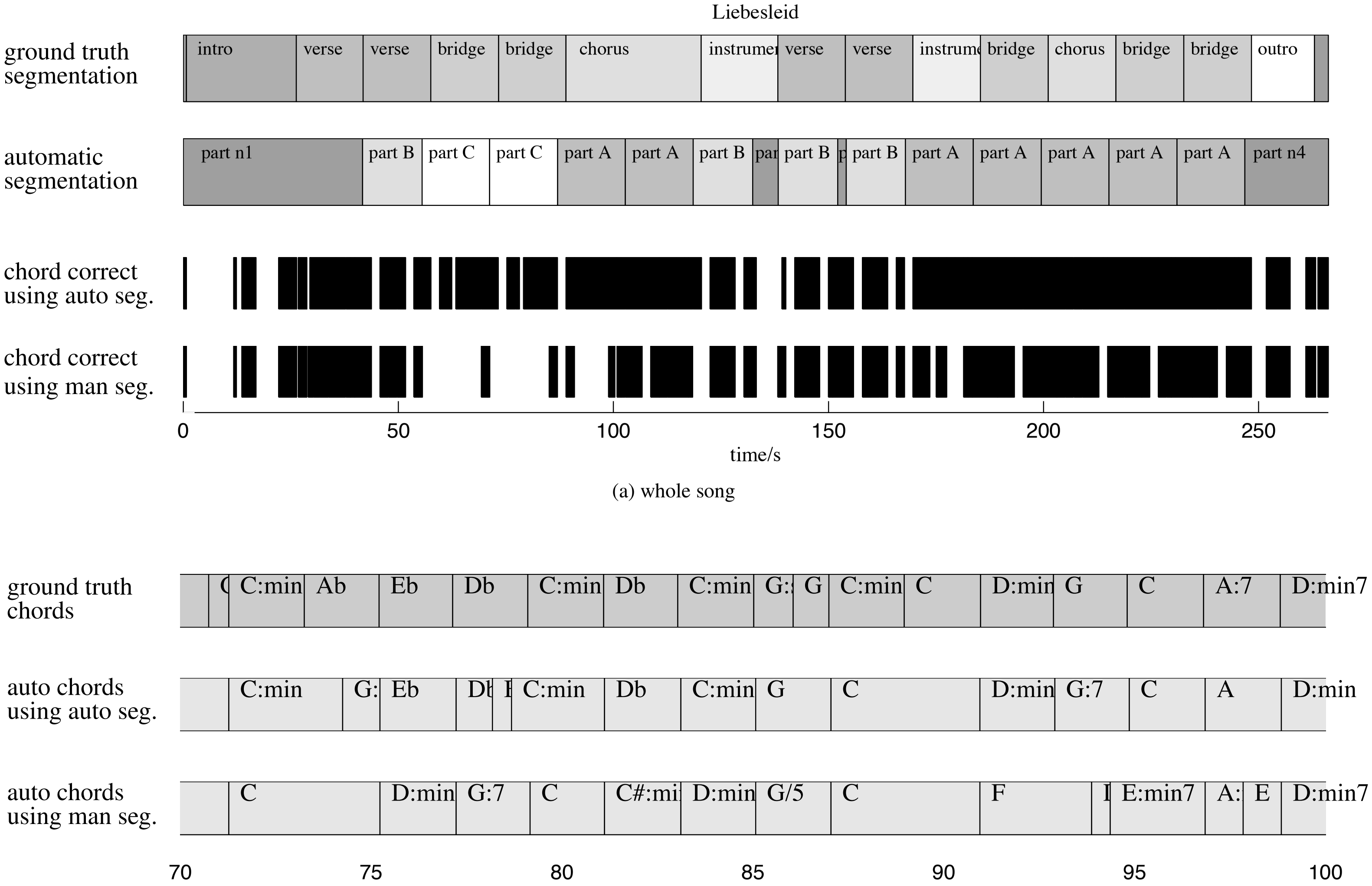
Figure 6.2: "A Hard Day's Night". Effect of using the repetition information: comparing the fully-automatic autobeat-autoseg method to the baseline method that does not use repetition information. (a) the two bottom bars are black at times where the chord has been recognised correctly (using MIREX-style evaluation) over the whole song. The two top bars display the manual segmentation (for reference) and the automatic segmentation used to obtain the autobeat-autoseg results. (b) manually-annotated and automatically-extracted chord sequences for an excerpt of the song.



(a) whole song

(b) excerpt between 105s and 135s

Figure 6.3: “Got To Get You Into My Life”. Effect of using the repetition information: comparing the fully-automatic autobeat-autoseg method to the baseline method that does not use repetition information. In (a) the two bottom bars are black at times where the chord has been recognised correctly (using MIREX-style evaluation) over the whole song. The two top bars display the manual segmentation (for reference) and the automatic segmentation used to obtain the autobeat-autoseg results. (b) manually-annotated and automatically-extracted chord sequences for an excerpt of the song.



(a) whole song

(b) excerpt between 70s and 100s

Figure 6.4: “Liebesleid”. Effect of using the repetition information: comparing the fully-automatic autobeat-autoseg method to the automatic method using manual segmentation (autobeat-manseg). (a) the two bottom bars are black at times where the chord has been recognised correctly (using MIREX-style evaluation) over the whole song. The two top bars display the manual segmentation used to obtain the autobeat-manseg results and the automatic segmentation used to obtain the autobeat-autoseg results. (b) manually-annotated and automatically-extracted chord sequences for an excerpt of the song.

| configuration | | normal | segwise |
|---|---|---|---|
| manbeat | autoseg | 75.7 | 75.4 |
| | manseg | 74.6 | 74.7 |
| | noseg | 73.4 | n/a |
| autobeat | autoseg | 75.2 | 75.1 |
| | manseg | 74.4 | 74.5 |
| | noseg | 73.0 | n/a |

(a) MIREX-style RCO

| configuration | | normal | segwise |
|---|---|---|---|
| manbeat | autoseg | 59.3 | 59.1 |
| | manseg | 57.2 | 57.2 |
| | noseg | 55.9 | n/a |
| autobeat | autoseg | 59.0 | 58.8 |
| | manseg | 57.7 | 57.4 |
| | noseg | 56.3 | n/a |

(b) RCO using the *full* chord class set

Table 6.1: Effects of using repetition cues on transcription accuracy.

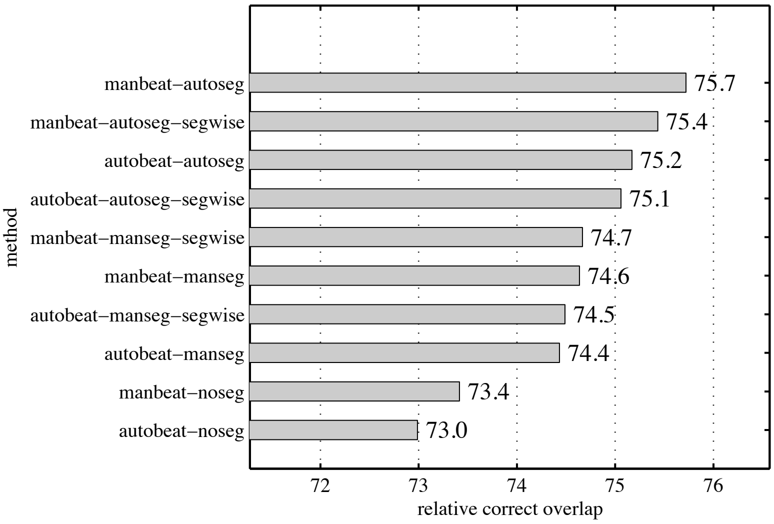


Figure 6.5: MIREX-style relative correct overlap (RCO) of all 10 tested configurations. All models using repetition cues perform better than the baseline models.

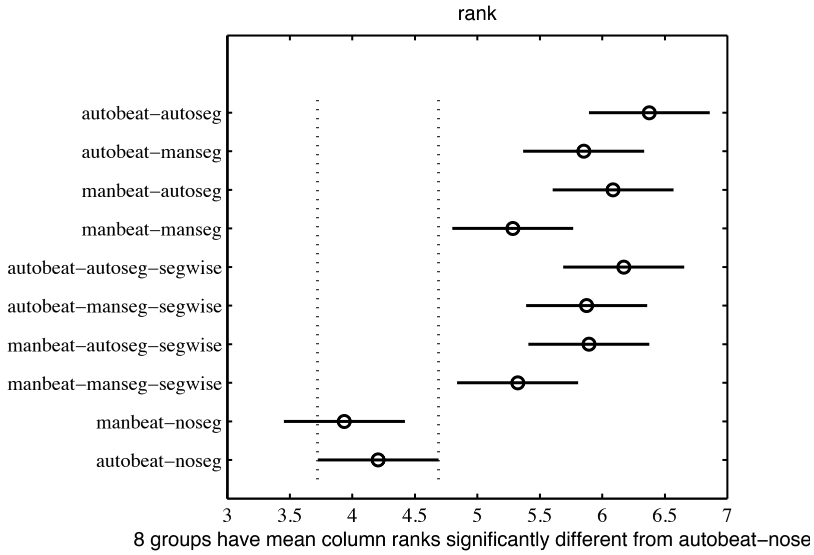
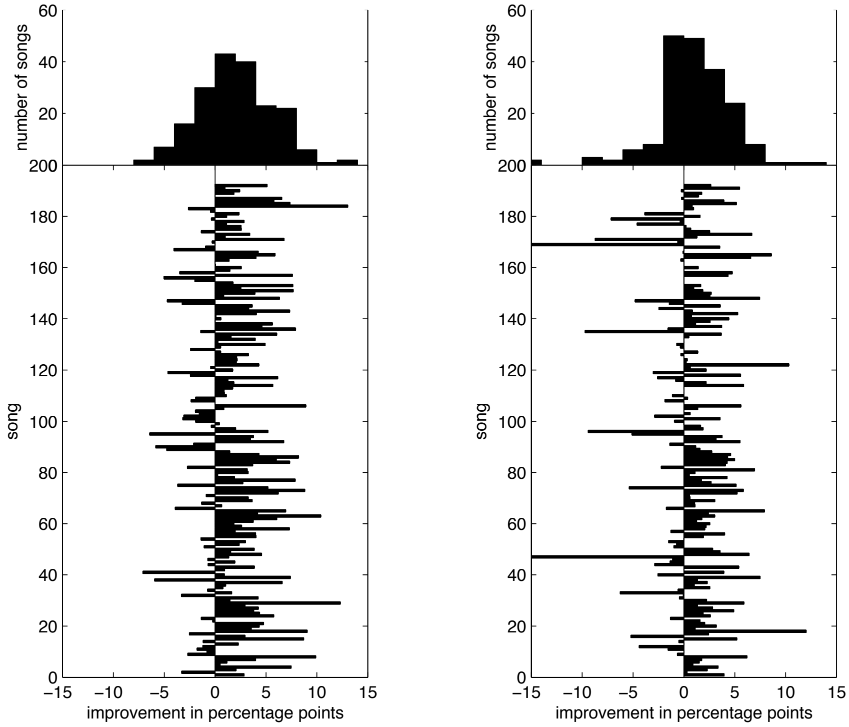


Figure 6.6: Tukey-Kramer multiple comparison test (at 95% confidence) of the results of all 10 configurations based on the Friedman test and MIREX-style evaluation.

Kramer multiple comparison test shows that all methods using segmentation result in a significant improvement over the baseline methods (Figure 6.6). There is also a difference between automatic and manual segmentation, as shown in Table 6.1a shows: whether we consider the "normal" column or the "segwise" column, the automatic segmentation performs better than the manual segmentation for both automatic and manual beats. This is also true for the results using evaluation on the *full* chord class set listed in Table 6.1b. However, the differences are not significant at 95% confidence in terms of the Tukey-Kramer multiple comparison test, neither for MIREX-style evaluation (see Figure 6.6) nor for evaluation on the *full* chord class set. However, we had expected manual segmentation to outperform our automatic algorithm, and it is indeed very encouraging to see that our automatic segmentation algorithm has performed at least as well as the manual segmentation for our use case.

To explain why manual segmentation does not actually perform better than automatic segmentation we can refer to Example 4 in Section 6.4.1 (also, Figure 6.4): since the manual segmentation annotations do not strictly assign exactly the segments that are harmonic repeats to the same segment type, it is likely that, as a result, segments were added erroneously. The automatic method is tailored to the task of finding harmonic repetitions and is conservative enough to avoid this kind of misclassification.

An alternative way to look at the improvements is offered by Figure 6.7. For simplicity we have restricted our scope to two pairwise comparisons, which both show the improvements



(a) autobeat-autoseg against autobeat-noseg

(b) manbeat-manseg against manbeat-noseg

Figure 6.7: Song-wise improvement in RCO for the methods using segmentation cues over the respective baseline methods. The lower part of the figures shows the performance difference per song, and the upper part summarises the same information in a histogram. Using autobeat-autoseg improves performance on 71% of songs compared to autobeat-noseg (Figure 6.7a); manbeat-manseg improves RCO scores for 63% of songs compared to the manbeat-noseg method (Figure 6.7b).

resulting from segmentation information. Here we can see the performance improvement on a song-wise basis. For example, Figure 6.7a allows us to see immediately why the difference between using and not using segmentation is significant: the autobeat-autoseg method performs better than the autobeat-noseg method on the majority of songs (71%). Similarly Figure 6.7b shows the song-wise improvement of manbeat-manseg over manbeat-noseg, where the method using repetition cues increases performance in 63% of the songs.

Our main hypothesis—that repetition cues can be used to improve chord transcription—has been confirmed with statistical significance. Next, we consider the difference between manual and automatic beat extraction to find out whether the more practicable automatic approach yields significantly lower results.

#### Effect of Automatic Beat-tracking against Manual Annotations

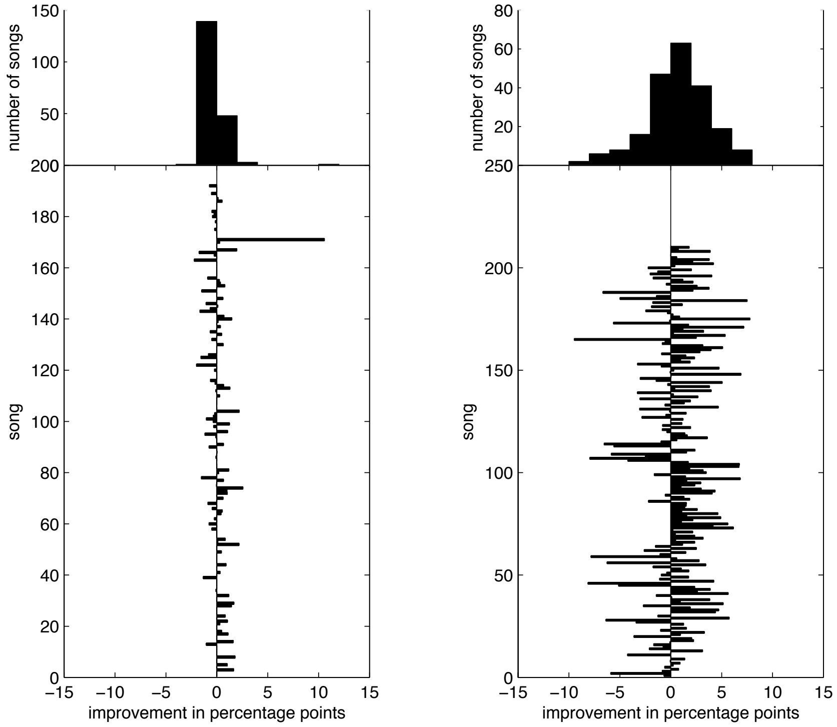
Manual beat annotation is very labour-intensive, and hence being able to use automatic beat-tracking considerably increases the usability of an algorithm. Tables 6.1a and 6.1b (on page 129) show that the use of manual beats increases performance in some cases, as could have been expected. In fact, this result is more intuitive than that presented in our previous publication (Mauch et al., 2009c), in which we found the manual beat detection to result in worse chord transcriptions, and the results presented here have probably benefited from the use of revised and corrected manual beat transcriptions (Mauch et al., 2009a). However, the differences are slight and do not prove to be significant (Figure 6.6). For the kind of popular music our test set contains, automatic beat-tracking is of no significant disadvantage compared to manual beat annotations.

#### Effect of Segment-wise Inference

Like automatic beat-tracking, the segment-wise inference has a practical advantage: it reduces the memory requirements of our algorithms. It is therefore encouraging that no significant difference can be found between the methods using inference by segment compared to their "normal" inference counterpart. Even looking at individual songs, the differences are generally small. Consider, for example, the improvement of autobeat-autoseg over the method autobeat-autoseg-segwise as seen in Figure 6.8: differences are random and small, with the exception of one song ("One After 909", Lennon/McCartney). Here, in the case of segment-wise inference, the key context was missing, and produced a faulty key estimate on some parts, whereas in the normal inference the key estimate helped stabilise the chord transcription (see Chapter 4). The result suggests that segment-wise inference will not usually be detrimental in the chord analysis of popular songs. This is a very welcome result because memory consumption can be lowered.

### 6.4.3 An Additional Experiment

The work in this chapter and the work in Chapter 5 are based on the chord extraction method proposed in Chapter 4. Since we observed improvement in accuracy in both, it is interesting to see whether combining the two techniques—a different chroma and the use of repetition information—improves results further. We take the method presented in Section 5.3.4 and perform inference as previously done in this chapter using the autobeat-autoseg configuration (with normal, song-wise inference). The result is the best obtained so far, with 80.7% RCO according to the MIREX-style *majmin* evaluation, and 64.5% with the more detailed *full* chord class eval-



(a) autobeat-autoseg over autobeat-autoseg-segwise

(b) using NNLS chroma: improvement of autobeat-autoseg over autobeat-noseg

Figure 6.8: Song-wise improvement in RCO. Figure (a) shows the relatively small effect of segment-wise inference: improvement of the method autobeat-autoseg over the method autbeat-autoseg-segwise is small. Figure 6.8b displays the differences between two algorithms using the NNLS chroma introduced in Chapter 5: here also, using repetition cues (autobeat-autoseg) increases performance (see discussion in Section 6.4.3). The upper parts of the figures are histograms summarising the song-wise differences.

uation. Here too, the method using segmentation is significantly different from the one without (Section 5.3.4), according to the Friedman test on the song-wise results ($p < 0.01$, for both evaluations). The song-wise differences for these two experiments are shown in Figure 6.8b.

## 6.5 Discussion and Future Work

The method of averaging the low-level content between repeated sections can be seen as a global-level modelling of the song. While the results are encouraging, the procedure is sequential, and, unlike the more elegant DBN, in which several musical parameters mutually interact, no information from the chord/key extraction is fed back to the segmentation. Developing a "grand" model that achieves this would involve a radical redesign of the baseline model, but there are a number of possible modifications that are easier to implement. For example, rather than simply averaging the chromagram chunks that belong to repeated segments, one could use the one that yields the highest likelihood given the model[5]. The chunks could also be averaged in different ways, perhaps by using the median instead of the mean.

The procedure presented here can also be looked at as a modular algorithm, in which the actual implementation of the components matters little, as long as they work well. We have already shown that both manual segmentation and our automatic segmentation algorithm work well in the framework, and it would be surprising if others could not. This is to say, our method is not tied to our segmentation and chord extraction modules, and we expect that the method still achieves an improvement if other modules, possibly from third parties, are plugged in.

We have shown that for many songs, our method brings about improvement, and that the idea is indeed a valid way of achieving better chord transcription. In a fully-automatic environment, however, the pitfalls highlighted in the examples may sometimes hinder performance. Additionally, slight changes between two choruses, for example, may be intended by the songwriter or artist interpreting the song, and a fully automatic method may not detect them. In a semi-automatic application of chord transcription, these limitations would be removed: a user could easily adjust section boundaries, and dissociate a segment instance from a segment type, if needed. Once section boundaries are adjusted, our method could infer the chord transcription, and a lead sheet representation with appropriate repeat signs could easily be produced.

### Conclusions

We have proposed a novel algorithm to improve an existing chord transcription method by averaging low level features over the song segments whose chroma progressions are approximately

[5] We would like to thank Prof. Sagayama for this suggestion.

the same. To be able to apply this method in a fully automatic chord transcription procedure, we also developed a structural segmentation algorithm tailored to the task.

Using manually annotated segments, we have shown that the method does indeed significantly improve chord extraction accuracy for the evaluation used in the MIREX tasks as well as for a more detailed chord evaluation method. The improvement is statistically significant, with a typical improvement of around 1.5 percentage points. The improvement is also observed, and significant, when using the automatic segmentation algorithm, which shows that the principle can be applied to real systems. Here the improvement on the collection level is usually higher than 2 percentage points. The automatic segmentation algorithm ranked first in the 2009 MIREX Structural Segmentation task as a general structural segmentation method.

We have illustrated the benefits of the method and differences between manual and automatic segmentation. The examples show how non-repeating incorrect chord fragments are removed by the averaging process, and why on some songs, chord transcription deteriorated.

We have described a property which further increases the quality of the chord transcriptions and will be necessary for the production of concise lead sheets: the chords in repeated segments will be transcribed identically or nearly identically, since they are provided with the same low-level input data. In a lead sheet, for example, a repeated chorus would then be written only once with repeats indicated.

We have shown that the use of automatic beat-tracking does not lead to significantly worse chord transcription results. We have also shown that inferring the chords on separate structural segments, not on the whole song, does not significantly affect recognition rates. This is a welcome result because the memory usage of our algorithms is proportional to the length of the observed feature sequence. As a consequence, memory usage can be reduced.

Finally, combining the new technique with the proposed DBN from Chapter 4 and the proposed NNLS chroma from Chapter 5 yields a score of 81% on the 2009 MIREX dataset, which is significantly higher than any other known algorithm, including the ones in this thesis.

# Conclusions 7

In this thesis we have discussed several novel methods for automatic chord transcription from audio that span a wide range of abstraction levels. All methods were developed with express emphasis on their musical relevance.

In order to provide low-level features that reflect the notes played—rather than a spectral transform including harmonically irrelevant partials and noise—we have proposed an approximate transcription method using non-linear least squares. In order to integrate high-level features such as metric position, key, chord, and bass note, we have presented a dynamic Bayesian network that treats all of these as hidden variables, and models two low level features as observed variables: bass and treble chromagrams. Time series models such as DBNs model only local dependencies. In order to overcome this limitation, we have introduced an algorithm that uses repetition cues to feed information of the global song structure into the chord estimation process. We have shown that every proposed method increases the accuracy of chord transcription with statistical significance, and our best methods outperform the state of the art by a large margin.

In the rest of this chapter we will summarise the main achievements (Section 7.1) and then conclude this thesis by exploring a range of possible directions for future work (Section 7.2).

## 7.1 Summary

The overriding principle of our algorithms is to integrate as much musical context as possible into the transcription process. Especially Chapter 4 and Chapter 6 show that musical qualities beyond low-level harmonic features can contribute to a better chord transcription.

In Chapter 4 we have presented a novel musically-informed dynamic Bayesian network for the automatic extraction of chord transcriptions from musical audio. This model is the basis for all other methods we have proposed in this thesis. The model is a manually-tuned expert model. Although this aspect of the model resembles a rule-based approach, one important aspect of the model is different from rule-based approaches: our method simultaneously infers metric position, key, chord, and bass pitch class, thus reflecting the inter-dependencies of these musical qualities.

Ours is the first model to combine these musical qualities into a single model. As a consequence of the higher amount of detail, we can obtain information beyond chord transcription: the method detects bar boundaries and is able to compensate for deleted or inserted beats; the method detects keys and key changes; sophisticated bass pitch class modeling uses the special position of the bass at the first beat of the chord and allows the method to detect the nominal bass note of a chord (for `maj` chords). The time signature and key signature changes are essential features for the creation of lead sheets, and we have shown several examples of lead sheets created from our fully automatic transcription. With 121 chords, the model provides a higher level of chord detail than has been realised in previous approaches. This provides higher accuracy, without decreasing the performance of the method.

The proposed *full-MBK* method achieves a state-of-the-art correct overlap score of 73%, and outperforms all systems tested in the 2009 MIREX task for pretrained chord detection. In the train-test evaluation task the algorithm proposed by Weller et al. (2009) scores better than our *full-MBK* model in terms of relative correct overlap, but does not do so significantly. We compared 10 different variants of our algorithm and show that bass and key modelling cumulatively improve the method's performance in terms of correct overlap with statistical significance. The greatest enhancement is achieved by bass modelling.

As a complement to the correct overlap evaluation method, we have used a metric for chord segmentation quality to show how well the locations and granularity of chord changes resemble those of the ground truth. Our results show a significant improvement in segmentation quality due to bass modelling, and—in some circumstances—also for metric position modelling. Our best model achieves a segmentation measure of 0.782.

The key model does not only aid the correct identification of chords, but also performs well in its own right by correctly identifying 80% of the songs' main keys. The relative overlap of correctly recognised keys is 77%.

In Chapter 5 we address a remaining problem in the front end of the DBN: the confusion of chords due to the influence of upper partials on the chromagram. We propose two substantially different approaches, namely: statistical training of the chroma nodes in the model, and an enhanced chroma extraction technique based on a prior transcription step using a non-negative least squares (NNLS) algorithm.

The best results were achieved by one of the methods using the enhanced NNLS chroma extraction technique, reaching 79% MIREX-style accuracy (80% with an additional minor modification of the DBN). This is a significant improvement over the state of the art (MIREX 2009 Chord Detection Task). We have also shown that statistical learning of chroma can boost the recognition rate of individual chord types. In our implementation, this came at the cost of lower overall accuracy because many frequent chords were misclassified. The detailed comparison of confusion matrices between the two approaches shows that while the trained variants can be better for the annotation of some particular chords, they tend to generate errors in less musically acceptable ways than the NNLS models do. These inherit the musical "conservativeness" from the baseline method, and often provide acceptable (simpler) approximations to the true chord.

Chapter 6 has two distinct contributions. We introduce a method that uses the global song-level repetition structure to share information between repeated segments. We also provide an algorithm that automatically extracts the repetition structure from a beat-synchronous chroma representation. The segmentation algorithm ranked first in the 2009 MIREX Structural Segmentation task as a general structural segmentation method, and an earlier version of the proposed chord extraction algorithm ranked first in the 2009 MIREX Chord Detection (pretrained) task.

Using manually annotated segment and beat annotations, we proved that the use of repetition cues significantly improves recognition accuracy. The fully automatic method, using automatic beat-tracking and the novel automatic segmentation method, also yielded significantly better results than the baseline method, and often also better than the methods using manual segmentation. The best result was achieved by the method using manual beat annotations and automatic segmentation. According to the MIREX-style evaluation metric the fully automatic method performs significantly better than the best method entering the 2009 MIREX Chord Detection tasks.

We also report the welcome result that two other parameters do not result in significant differences: the use of automatic beat-tracking, and the use of segment-wise inference (instead of inference over the whole song). Both have practical implications: the use of automatic beat-tracking greatly increases the ease of use of the methods; the use of segment-wise inference can reduce the memory requirements of the algorithm.

Finally, combining the new technique with the proposed DBN from Chapter 4 and the proposed NNLS chroma from Chapter 5 yields a score of 81% on the 2009 MIREX dataset, which is significantly higher than any other known algorithm, including the ones presented in this thesis. The ranking in Table 7.1 allows a quick comparison of a selection of methods in

| method | RCO in % |
|---|---|
| *full-plain* (Chapter 4) | 65.5 |
| *full-MBK* (Chapter 4) | 73.0 |
| Weller et al. (MIREX 2009) | 74.2 |
| autobeat-autoseg (Chapter 6) | 75.2 |
| STD-0.6 (Chapter 5) | 78.8 |
| autobeat-autoseg with STD-0.6 (Section 6.4.3) | **80.7** |

Table 7.1: Ranked MIREX-style relative correct overlap of selected fully automatic methods.

terms of the MIREX-style RCO score. We have included the best-performing methods from Chapters 4, 5 and 6, as well as results of the basic *full-plain* model (Chapter 4) and the highest score in the 2009 MIREX Chord Detection tasks (Weller et al., 2009).

## 7.2 Directions for Future Work

In the process of working on the research for this thesis, and the writing of the thesis itself, many exciting new research ideas have arisen but had to be left aside in favour of implementing and testing the methods presented. Here, we would like to mention a selection of ideas for future work in the context of chord transcription. Since we will not be able to follow all directions mentioned here ourselves, we hope that other researchers can draw some inspiration, or even insights, from the following paragraphs.

**Implementation of the Proposed Methods as a Vamp Plugin**

Most immediately, we will concentrate our efforts on making the proposed methods available to the public. One possibility of doing so is an implementation in C++ as a *Vamp plugin*[1], which would make our methods available to all users of Vamp hosts such as the open source multi-platform software *Sonic Visualiser*[2]. This aim seems attainable, since for inference in DBNs there is already an open source C++ package[3]. The usage of the C++ programming language is also expected to alleviate the problem of high memory usage because parameters can be passed by reference, in contrast to pass by value (i.e. copying) in MATLAB.

**Refinement of our DBN**

Perhaps the most serious shortcoming of our DBN is its inability to deal with time signatures other than $\frac{4}{4}$. As we have already indicated, preliminary experiments have shown that other time signatures such as $\frac{3}{4}$ can be implemented and even time signature changes can be tracked.

[1] `http://vamp-plugins.org/`

[2] http://www.sonicvisualiser.org/

[3] Mocapy++ `http://sourceforge.net/projects/mocapy/`

An algorithm with this ability would be substantially more general. Additionally, we would like to test whether the minor change in the metric position model we discussed in Chapter 4 will provide the expected significant improvement of metric modelling over the *plain* model (page 77). We would also like to revisit the bass model, and find a solution which uses only 12 bass chroma bins, without the additional flatness measure in the 13th bin. Informal experiments indicate that this is possible.

### Separation of Music Model and Sound Model

Based on the results of Chapter 5 we have already discussed the idea that chords should be modelled not as a single profile, but as a dual model, in which one part is concerned with the musical note events (or pitch class events) conditional on a chord label, and the other one with the physical properties of a feature conditional on the note events. This is a step towards a more realistic model whose implementation is easily conceivable within the DBN paradigm. For example, twelve binary pitch class nodes could depend on the chord node (and, ideally, also the key node). Each binary pitch class node would in turn generate a chroma salience value, modelled as a Gaussian. The essential characteristic of this kind of model is the intermediate pitch class layer[4], which is removed from the physical realisation of the pitch classes. A possible topology of one slice of such a model is displayed in Figure 7.1. This resembles a chord labelling approach from symbolic data with an audio transcription front-end, but here, the transcription would be a "soft" transcription, such as our approximate NNLS transcription. Using other third party methods may have similar or even better effect on our chord transcriptions, as we will discuss below.

### Replace Modules in Our Methods with Other Components

Our implementations of algorithms described in this thesis show only a few ways of realising the underlying concepts. We believe that at least some of the techniques are more generally useful. In the discussion sections of Chapters 5 and 6 we have already stated that it would be interesting to replace parts of our algorithms with third party methods: using other transcription approaches such as the visualisation function proposed by Klapuri (2009) as a front end instead of our NNLS approximate trancription, and using other sequence-based segmentation algorithms (Chapter 6) such as the one proposed by Rhodes and Casey (2007).

[4] similar to the subchords from one of our previous papers (Mauch and Dixon, 2008)

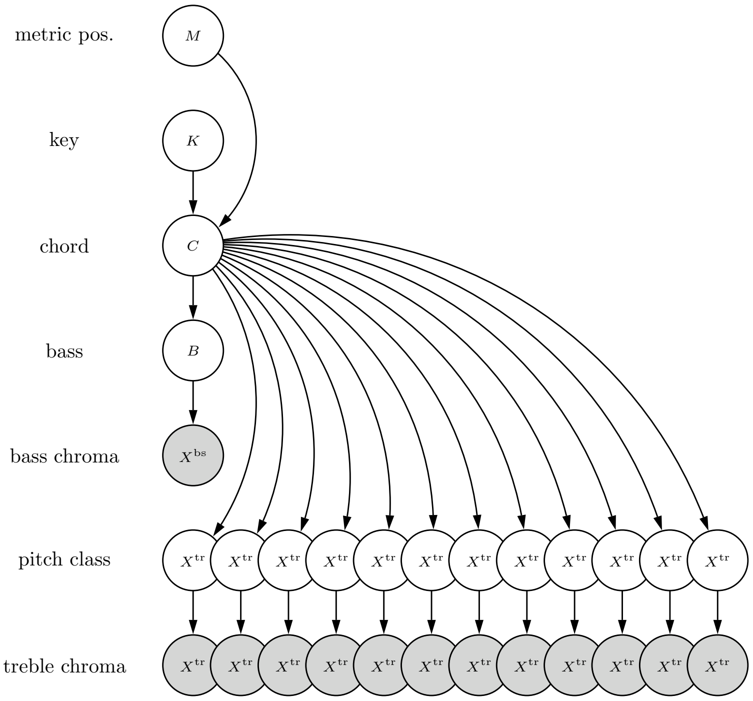


Figure 7.1: Separation of music model and sound model. In this hypothetical model the 12 binary pitch class nodes depend on the current chord, and the treble chroma is split into 12 one-dimensional chroma nodes, each modelled by a one-dimensional Gaussian. Independence of the chroma bins is assumed, i.e. the chroma needs to stem from a transcription approach, like, for example, our NNLS chroma.

### Reducing Redundancies in Models

Nearly all chord extraction algorithms assume, quite rightly, that chord types do not change with the root note (except for transposition). The same is true for the keys' relationships to chords. In existing probabilistic models, including ours, every chord type is modelled twelve times, instead of once. That means that a large part of these models is redundant. It seems that there should be a way to reduce these redundancies in order to make more parsimonious models of music. These could not only be far more elegant, but also more memory-efficient.

### Tatum-Synchronous Features

The use of beat-synchronous chroma is essential to our musical context models. It allows us to describe bars and bass note behaviour, and to extract repetitions of meaningful lengths. One drawback of the beat-synchronous paradigm is that chords or other musical events that are not quantised to the beat will be detected with a systematic error. One of many examples of this occurs in the song "You Won't See Me" (Lennon/McCartney) as shown in Figure 7.2: the chord A starts on the anticipated third beat of the bar. A purely beat-based algorithm is by definition not able to transcribe the chord change time correctly. There are many possible approaches to solve this problem. If we assume that the two chords before and after the change are detected correctly, a post-processing step on *tatum* level (the fastest regular pulse train, usually quavers) could adjust the chord change. It may be more principled to infer the correct harmonic rhythm from the tatum-synchronous features, for example by means of a pre-processing step that groups tatum-synchronous features into beat-synchronous features. We expect that this change could benefit chord transcription performance for heavily-syncopated styles like modern rock music and jazz.

### Integrating Segmentation

We have already indicated in the discussion of Chapter 6 that a "grand model" including both the high-level model of metric-position, key, chord and bass, and a repetition model could be a worthwhile project. In contrast to the algorithm we have proposed, information could flow back and forth between the chord extraction and repetition extraction, in a manner similar to the information flow between key and chords exploited in our DBN. Since, however, the model without the repetition part already requires large amounts of memory, it is not clear whether the implementation of an additional layer in the existing DBN would be computationally feasible. It is also not clear whether a DBN could express the semantics needed for repetition at all. An alternative technique may be to employ probabilistic context-free grammars.

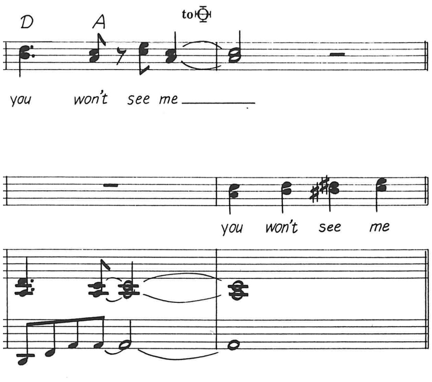


Figure 7.2: Excerpt from “You Won’t See Me” (Lennon/McCartney), taken from *The Beatles – Complete Scores*, Hal Leonard Publishing Corporation, 1993. The chord change from D to A is syncopated (anticipated), and happens between the second and third beat of the bar.

### Statistical Learning

The parameters for the segmentation method proposed in Section 6 are manually set. Since we have at our disposal manual segmentation data, we could learn these parameters from the data. We expect this to be relatively straight-forward because it mainly involves finding a good threshold for what is a “similar” beat in terms of chroma correlation, and what is not. Statistical learning in the chord estimation methods is far more complex. We have not used much statistical learning in the chord transcription methods because of the relative scarcity of training data. Another problem is that the current methods do not always model actual concepts; the chord profile, for example, has no counterpart in the real world. We hypothesise that this is why none of the previous statistical learning approaches, including ours presented in Chapter 5, have reached the levels of accuracy we have presented using the NNLS chroma in Chapter 5 and 6. However, we believe that more realistic models like the pitch class model described above (see Figure 7.1) could be trained more effectively.

### Other Music Domains and Style-Dependent Inference and Testing

While our focus has been on chord transcription of popular music, we have already mentioned in Section 2.1.3 that in Baroque music the figured bass notation has played a role similar to that of lead sheet chords in pop and jazz music. The paradigm is the same: a label represents a bass note and the chord note degrees relative to this note. Although some of the properties of the model will change, for example the chord names and qualities, and the metric position modelling may have to be relaxed, the overall structure of our DBN model is well suited to this task and could be adapted to figured bass transcription.

In jazz music, improvements due to repetition cues are likely to be significant: while the extraction of chords in jazz is expected to be more difficult than in rock music, as a result of improvisation and more complex chord types, the repetition of segment types, “choruses”, is often more rigid.

We expect that there is not one best chord transcription algorithm for all music, even within one genre. Testing existing algorithms against editorial metadata such as genre or artist could reveal in more detail the strengths of different existing methods. For example, it is not hard to imagine high correlation between good performance of one of our algorithms with happy love songs, whereas bluesy, sad love songs—due to their use of blue notes—may perform much worse.

Hence, customising chord transcription methods to cater for different musical styles or genres seems a natural step. The survey of transcription practice conducted by Hainsworth (2003) also supports this idea (see Figure 2.3b on page 27). For a given piece of music, an algorithm or model can then be selected according to the retrieved metadata. In contrast to chord transcriptions, metadata like the name of the performer and genre of pieces of music are easily obtainable. They are often encoded in the ID3 tags of MP3 files, or otherwise they can be retrieved from databases like *MusicBrainz*[5] and *Last.fm*[6].

Perhaps, style could also be detected dynamically, in a similar way to the key model in our DBN. Here, the main issue is finding the right model parameters for a certain style (or mood etc.) of music. We expect that, with better chord models, this could be achieved through unsupervised learning.

**Semi-Automatic Chord Transcription**

We have seen in Chapter 6 that using repetition cues helped produce better chord transcriptions in many cases, but not always. This is likely to be true for many techniques that result in improved collection-level results. We expect that human beings, even those who themselves may not be expert enough to perform a complete chord transcription, would be able to discover mistakes in an automatic transcription. A computer program that enables such a user to make explicit the errors he has spotted could then use this information for a re-estimation of the transcription. By using probabilistic models like our DBN, rich ways of manipulating the output can be introduced, as in the following scenario. First, the user could correct possible beat-tracking errors. The next step could be to correct segmentation errors. Even if the user does not know what chord was played, he could suggest that he perceives a chord change at a certain

[5] `http://musicbrainz.org/`

[6] `http://www.last.fm/`

beat, and the computer program would re-estimate the sequence given the manually annotated chord change. Many other ways of intervention are possible, and could help even a novice user obtain a good chord transcription for songs on which a fully automatic algorithm fails.

Applications in music transcription, computational musicology, and content-based music information retrieval provide strong motivation to develop ever better music computing systems. All are likely to benefit from computational methods that can imitate aspects of human listening to obtain a multi-faceted, more complete representation of music.

# Appendix

# Chord Class Mapping A

Table A.1 and Figure A.1 display what proportion of the MIREX 2009 collection the chord classes occupy. Table A.2 contains a list with all chord mappings used.

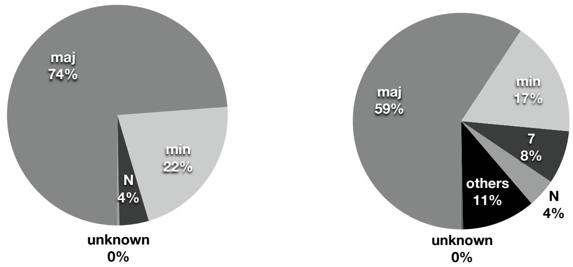
| chord type | relative<br>duration (%) |
|---|---:|
| `maj` | 59.5 |
| `min` | 17.3 |
| `7` | 8.0 |
| `N` | 4.3 |
| `min7` | 3.3 |
| `maj/5` | 2.2 |
| `maj/3` | 1.9 |
| `dim` | 1.0 |
| `maj6` | 0.9 |
| `maj7` | 0.8 |
| `aug` | 0.6 |
| unknown | 0.3 |

(a) *full*

| chord type | relative<br>duration (%) |
|---|---:|
| `maj` | 73.8 |
| `min` | 21.6 |
| `N` | 4.3 |
| unknown | 0.3 |

(b) *majmin*

Table A.1: The relative durations of chord class types in the two chord class sets used for evaluation, expressed as percentages.



(a) *majmin* (b) *full*

Figure A.1: Pie charts visualising the information given in Table A.1: the relative duration of chord classes in the collection. It is evident that the dominance of the `maj` chords is reduced in favour of more detailed chords.

| original chord | *majmin* | *full* |
|---|---|---|
|  | maj | maj |
| min | min | min |
| 7 | maj | 7 |
| min7 | min | min7 |
| N | N | N |
| /5 | maj | maj/5 |
| maj | maj | maj |
| /3 | maj | maj/3 |
| maj7 | maj | maj7 |
| sus4 | maj | maj |
| min/5 | min | min |
| maj6 | maj | maj6 |
| 9 | maj | 7 |
| min/b3 | min | min |
| /2 | maj | maj |
| aug | maj | aug |
| /7 | maj | maj |
| dim | min | dim |
| (1) | unknown | unknown |
| min/b7 | min | min7 |
| /b7 | maj | maj |
| 6 | unknown | unknown |
| min6 | min | min |
| dim7 | min | dim |
| 7(#9) | maj | 7 |
| /4 | maj | maj |
| (1,5) | maj | maj |
| /6 | maj | maj |
| 7/3 | maj | maj/3 |
| min/6 | min | min |
| maj7/3 | maj | maj/3 |
| min7/b7 | min | min7 |
| min/4 | min | min |
| maj9 | maj | maj7 |
| hdim7 | min | dim |
| min9 | min | min7 |
| sus4(b7) | maj | maj |
| maj(9) | maj | maj |
| maj/9 | maj | maj |
| 7/b7 | maj | 7 |
| min/2 | min | min |
| maj6/5 | maj | maj/5 |
| 7/5 | maj | maj/5 |
| minmaj7/b3 | min | min |
| min(*b3) | min | min |
| maj(9)/9 | maj | maj |
| sus4/5 | maj | maj/5 |
| sus2 | maj | maj |
| min7/4 | min | min7 |
| min(9) | min | min |
| sus4(2) | maj | maj |
| min(6) | min | min |
| dim/b3 | min | dim |
| /9 | maj | maj |
| (1,4) | maj | maj |
| min/7 | min | min |
| maj6/2 | maj | maj |
| hdim7/b7 | min | dim |
| min7/b3 | min | min7 |
| maj(2) | maj | maj |
| min6/5 | min | min |
| 9(*3) | maj | 7 |
| min7(4)/b7 | min | min7 |
| maj(4) | maj | maj |
| 6/2 | unknown | unknown |
| maj(*3) | maj | maj |
| minmaj7 | min | min |
| min7(9) | min | min7 |
| maj7(9) | maj | maj7 |
| maj(9)/3 | maj | maj/3 |
| maj(#4)/5 | maj | maj/5 |
| 7/2 | maj | 7 |
| min7(4) | min | min |
| maj7/5 | maj | maj/5 |
| maj(2)/2 | maj | maj |
| maj(*1)/#1 | maj | maj |
| 9(11) | maj | 7 |
| minmaj7/5 | min | min |
| min7(2,*b3,4) | min | min7 |
| min6/6 | min | min |
| maj6/3 | maj | maj/3 |
| maj(b9) | maj | maj |
| maj(*5) | maj | maj |
| dim7/b3 | min | dim |
| dim/b5 | min | dim |
| 7(b9) | maj | 7 |
| 7(13) | maj | 7 |
| /b3 | maj | maj |
| sus4(9) | maj | maj |
| sus4(2)/2 | maj | maj |
| sus2(b7) | min | min |
| min7(*5,b6) | min | min7 |
| min(2) | min | min |
| min(*5) | min | min |
| maj/2 | maj | maj |
| maj(*1)/5 | maj | maj/5 |
| maj(#11) | maj | maj |
| dim7/b9 | min | dim |
| dim7/5 | min | dim |
| 7sus | unknown | unknown |
| /#4 | maj | maj |
| (6) | unknown | unknown |
| min7/5 | min | min7 |
| min7(4)/5 | min | min7 |
| min7(*b3) | min | min7 |
| min6/b3 | min | min |
| min/3 | min | min |
| min(b6)/5 | min | min |
| min(9)/b3 | min | min |
| maj6/b7 | maj | maj6 |
| maj/5 | maj | maj/5 |
| maj/3 | maj | maj/3 |
| maj(9)/6 | maj | maj |
| maj(9)/5 | maj | maj/5 |
| hdim7/b3 | min | dim |
| dim7/2 | min | dim |
| aug/3 | maj | aug |
| aug/#5 | maj | aug |
| 9/5 | maj | maj/5 |
| 7/b3 | maj | 7 |
| 7/b2 | maj | maj |
| 7/#5 | maj | 7 |
| 4 | unknown | unknown |
| /b6 | maj | maj |
| (1,5,9) | min | min |
| (1,4,b7) | maj | 7 |
| (*5) | unknown | unknown |
| sus4/4 | maj | maj |
| sus | maj | maj |
| min7/7 | min | min7 |
| min7(*5)/b7 | min | min7 |
| min(4) | min | min |
| min(*b3)/5 | min | min |
| min(*5)/b7 | min | min |
| min(*3)/5 | min | min |
| maj9(*7) | maj | maj7 |
| maj7/7 | maj | maj7 |
| maj7/2 | maj | maj7 |
| maj7(*b5) | maj | maj7 |
| maj7(*5) | maj | maj7 |
| maj6(9) | maj | maj6 |
| maj(13) | maj | maj |
| maj(11) | maj | maj |
| dim7/7 | min | dim |
| dim/b7 | min | dim |
| dim/5 | min | dim |
| aug(9,11) | maj | aug |
| 9(*3,11) | maj | 7 |
| 7(*5,13) | maj | 7 |
| (b6) | unknown | unknown |
| (b3,5) | min | min |
| (7) | unknown | unknown |
| (5) | unknown | unknown |
| (4,b7,9) | unknown | unknown |
| (3) | unknown | unknown |
| (1,b7)/b7 | unknown | unknown |
| (1,b7) | unknown | unknown |
| (1,b3,4)/b3 | unknown | unknown |
| (1,b3)/b3 | unknown | unknown |
| (1,b3) | unknown | unknown |
| (1,4,b5) | unknown | unknown |
| (1,2,5,b6) | unknown | unknown |
| (1,2,4) | unknown | unknown |

Table A.2: All chord types in the song collections and their chord class mappings for the *majmin* and *full* chord class sets.

# Song Collection B

All ground truth song collections used in this thesis are subsets of the collection published in the OMRAS2 Metadata Project 2009 (Mauch et al., 2009a). For all songs we have audio-aligned reference chord and key annotations as well as annotations of musical structural segmentation. Of the 225 reference chord transcriptions, 210 were used in the MIREX Chord Detection tasks, and for ease of comparison we use the same 210 songs (174 by The Beatles, 18 by Queen and 18 by Zweieck). A full list, is given on the following pages. For the evaluation in Chapter 6, we additionally need manual beat labels. At this moment, we do not have at our disposal the beat annotations for the Queen songs in the collection mentioned above, and hence the number of songs reduces to 192 for those experiments.

$$\left\{\begin{array}{c}\text{OMRAS2 Metadata:}\\ \text{225 Songs}\\ \text{(Beatles, Queen,}\\ \text{Zweieck, Carole King)}\end{array}\right\} \supset \left\{\begin{array}{c}\text{Chapters 4 and 5:}\\ \text{210 Songs}\\ \text{(Beatles, Queen,}\\ \text{Zweieck)}\end{array}\right\} \supset \left\{\begin{array}{c}\text{Chapter 6:}\\ \text{192 Songs}\\ \text{(Beatles,}\\ \text{Zweieck)}\end{array}\right\}$$

## B.1 Queen

A Kind Of Magic

Bicycle Race

Bohemian Rhapsody

Crazy Little Thing Called Love

Don't Stop Me Now

Fat Bottomed Girls

Friends Will Be Friends

Good Old-Fashioned Lover Boy

Hammer To Fall

I Want To Break Free

Play The Game

Save Me

Seven Seas Of Rhye

Somebody To Love

We Are The Champions

We Will Rock You

Who Wants To Live Forever

You're My Best Friend

## B.2 The Beatles

Across the Universe

Act Naturally

A Day In The Life

A Hard Day's Night

All I've Got To Do

All My Loving

All You Need Is Love

And I Love Her

And Your Bird Can Sing

Anna (Go To Him)

Another Girl

Any Time At All

Ask Me Why

A Taste Of Honey

Baby It's You

Baby's In Black

Baby You're A Rich Man

Back in the USSR

Because

Being For The Benefit Of Mr. Kite!

Birthday

Black Bird

Blue Jay Way

Boys

Can't Buy Me Love

Carry That Weight

Chains

Come Together

Cry Baby Cry

Dear Prudence

Devil In Her Heart

Dig a Pony

Dig It

Dizzy Miss Lizzy

Doctor Robert

Don't Bother Me

Do You Want To Know A Secret

Drive My Car

Eight Days a Week

Eleanor Rigby

Everybody's Got Something To Hide Except Me and My Monkey

Everybody's Trying to Be My Baby

Every Little Thing

Fixing A Hole

Flying

For No One

For You Blue

Get Back

Getting Better

Girl

Glass Onion

Golden Slumbers

Good Day Sunshine

Good Morning Good Morning

Good Night

Got To Get You Into My Life

Happiness is a Warm Gun

Hello Goodbye

Help!

Helter Skelter

Here Comes The Sun

Here, There And Everywhere

Her Majesty

Hold Me Tight

Honey Don't

Honey Pie

I Am The Walrus

I Don't Want to Spoil the Party

If I Fell

If I Needed Someone

I'll Be Back

I'll Cry Instead

I'll Follow the Sun

I'm a Loser

I Me Mine

I'm Happy Just To Dance With You

I'm Looking Through You

I'm Only Sleeping

I'm So Tired

I Need You

In My Life

I Saw Her Standing There

I Should Have Known Better

It's Only Love

It Won't Be Long

I've Got A Feeling

I've Just Seen a Face

I Wanna Be Your Man

I Want To Tell You

I Want You

I Will

Julia

Kansas City- Hey, Hey, Hey, Hey

Let It Be

Little Child

Long Long Long

Love Me Do

Lucy In The Sky With Diamonds

Maggie Mae

Magical Mystery Tour

Martha My Dear

Maxwell's Silver Hammer

Mean Mr Mustard

Michelle

Misery

Money

Mother Nature's Son

Mr. Moonlight

No Reply

Norwegian Wood (This Bird Has Flown)

Not A Second Time

Nowhere Man

Ob-La-Di, Ob-La-Da

Octopus's Garden

Oh! Darling

One After 909

Penny Lane

Piggies

Please Mister Postman

Please Please Me

Polythene Pam

P. S. I Love You

Revolution 1

Rock and Roll Music

Rocky Raccoon

Roll Over Beethoven

Run For Your Life

Savoy Truffle

Sexy Sadie

Sgt. Pepper's Lonely Hearts Club Band

Sgt. Pepper's Lonely Hearts Club Band (Reprise)

She Came In Through The Bathroom Window

She Said She Said

She's Leaving Home

Something

Strawberry Fields Forever

Sun King

Taxman

Tell Me What You See

Tell Me Why

The End

The Fool On The Hill

The Long and Winding Road

The Night Before

There's A Place

The Word

Things We Said Today

Think For Yourself

Ticket To Ride

Till There Was You

Tomorrow Never Knows

Twist And Shout

Two of Us

Wait

What Goes On

What You're Doing

When I Get Home

When I'm Sixty-Four

While My Guitar Gently Weeps

Why Don't We Do It In The Road

With A Little Help From My Friends

Within You Without You

Words of Love

Yellow Submarine

Yer Blues

Yesterday

You Can't Do That

You Like Me Too Much

You Never Give Me Your Money

You Really Got A Hold On Me

You're Going To Lose That Girl

Your Mother Should Know

You've Got To Hide Your Love Away

You Won't See Me

## B.3 Zweieck

Akne

Andersrum

Blass

Duell

Erbauliche Gedanken Eines Tobackrauchers

Es Wird Alles Wieder Gut, Herr Professor

Ich Kann Heute Nicht

Jakob Und Marie

Liebesleid

Mr Morgan

Paparazzi

Rawhide

Santa Donna Lucia Mobile

She

Spiel Mir Eine Alte Melodie

Tigerfest

Zuhause

Zu Leise Für Mich

# Index

# Bibliography

S. Abdallah, K. Noland, M. Sandler, M. Casey, and C. Rhodes. Theory and evaluation of a Bayesian music structure extractor. In *Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005, London, UK*, pages 420–425, 2005.

Samer A. Abdallah and Mark D. Plumbley. Polyphonic music transcription by non-negative sparse coding of power spectra. In *Proceedings of the 5th International Conference on Music Information Retrieval, ISMIR 2004, Barcelona, Spain*, 2004.

Amélie Anglade, Raphael Ramirez, and Simon Dixon. Genre classification using harmony rules induced from automatic chord transcriptions. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 669–674, 2009.

Amélie Anglade, Emmanouil Benetos, Matthias Mauch, and Simon Dixon. Improving music genre classification using automatically induced harmony-based rules. *submitted to Journal of New Music Research*, 2010.

Jean-Julien Aucouturier, François Pachet, and Mark Sandler. The way it sounds: Timbre models for analysis and retrieval of polyphonic music signals. *IEEE Transactions of Multimedia*, 2005.

Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *IEEE Transactions on Multimedia*, 7(4), February 2005.

Juan P. Bello. Audio-based cover song retrieval using approximate chord sequences: Testing shifts, gaps, swaps and beats. In *Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR 2007, Vienna, Austria*, 2007.

Juan P. Bello and Jeremy Pickens. A Robust Mid-level Representation for Harmonic Content in Music Signals. In *Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005, London, UK*, pages 304–311, 2005.

Jordi Bonada. Automatic technique in frequency domain for near-lossless time-scale modification of audio. In *Proceedings of the International Computer Music Conference*, pages 396–399, 2000.

Xavier Boyen and Daphne Koller. Tractable inference for complex stochastic processes. In *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 33–42, 1998.

Judith C. Brown. Calculation of a constant $Q$ spectral transform. *Journal of the Acoustical Society of America*, 89(1), 1991.

John Ashley Burgoyne, Laurent Pugin, Corey Kereliuk, and Ichiro Fujinaga. A cross-validated study of modelling strategies for automatic chord recognition in audio. In *Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR 2007, Vienna, Austria*, pages 251–254, 2007.

Anna Butterworth. *Harmony in Practice*. The Associated Board of Royal Schools of Music Publishing, 1999.

Giordano Cabral, François Pachet, and Jean-Pierre Briot. Automatic X traditional descriptor extraction: the case of chord recognition. In *Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005, London, UK*, pages 444–449, 2005.

M. Casey, R. Veltkamp, M. Goto, M. Leman, C. Rhodes, and M. Slaney. Content-Based Music Information Retrieval: Current Directions and Future Challenges. *Proceedings of the IEEE*, 96(4):668–696, 2008.

Benoit Catteau, Jean-Pierre Martens, and Marc Leman. A probabilistic framework for audio-based tonal key and chord recognition. In Reinhold Decker and Hans-Joachim Lenz, editors, *Proceedings of the 30th Annual Conference of the Gesellschaft für Klassifikation e.V., Freie Universität Berlin, March 8-10, 2006*, pages 637–644, 2007.

A Taylan Cemgil, Hilbert J. Kappen, and David Barber. A generative model for music transcription. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(2):679–694, 2006.

Stephen Chu and Beth Logan. Music summary using key phrases. Technical report, HP Cambridge Research Laboratory, April 2000. Available at `http://www.hpl.hp.com/techreports/Compaq-DEC/CRL-2000-1.html`, accessed 11.05.09.

J.W. Cooley and J.W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19:297–301, 1965.

Roger B. Dannenberg. Toward automated holistic beat tracking, music analysis, and understanding. In *Proceedings of the 6th International Conference on Music Information Retrieval*, London, 2005.

Matthew Davies. *Towards Automatic Rhythmic Accompaniment*. PhD thesis, Queen Mary University of London, London, UK, August 2007. URL `http://www.elec.qmul.ac.uk/digitalmusic/papers/2007/Davies07-phdthesis.pdf`.

Matthew E. P. Davies, Mark D. Plumbley, and Douglas Eck. Towards a musical beat emphasis function. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2009.

Diana Deutsch. Music recognition. *Psychological Review*, 76:300–307, 1969.

Diana Deutsch and Richard C. Boulanger. Octave equivalence and the immediate recall of pitch sequences. *Music Perception*, 2(1):40–51, 1984.

J. Stephen Downie, Donald Byrd, and Tim Crawford. Ten years of ISMIR: Reflections on challanges and opportunities. In Keiji Hirata, George Tzanetakis, and Kazuyoshi Yoshii, editors, *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 13–18, 2009.

J. Stephen Downie, Andreas F. Ehmann, Mert Bay, and M. Cameron Jones. The Music Information Retrieval Evaluation eXchange: Some observations and insights. In Z.W. Raś and A.A. Wieczorkowska, editors, *Advances in Music Information Retrieval*, pages 93–115. Springer-Verlag Berlin/Heidelberg, 2010.

K. Dressler and Sebastian Streich. Tuning frequency estimation using circular statistics. In *Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR 2007, Vienna, Austria*, pages 357–360, 2007.

Randy Felts. *Reharmonization techniques*. Music Reference Series, Arranging: reharmonization. Berklee Press, 2002.

Bernhard Flury. *A First Course in Multivariate Statistics*. Springer, 1997.

Jonathan Foote. Visualizing music and audio using self-similarity. In *Proceedings of the 7th ACM International Conference on Multimedia (Part 1)*, pages 77–80, 1999.

Takuya Fujishima. Real time chord recognition of musical sound: a system using Common Lisp Music. In *Proceedings of the International Computer Music Conference (ICMC)*, pages 464–467, 1999.

E. Gomez and P. Herrera. The song remains the same: identifying versions of the same piece using tonal descriptors. In *Proceedings of the 7th International Conference on Music Information Retrieval, ISMIR 2006, Victoria, Canada*, 2006.

Emilia Gomez. *Tonal Description of Audio Music Signals*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2006.

Masataka Goto. An Audio-based Real-time Beat Tracking System for Music With or Without Drum-sounds. *Journal of New Music Research*, 30(2):159–172, 2001.

Masataka Goto. A chorus-section detecting method for musical audio signals. In *Proceedings of the 2003 IEEE Conference on Acoustics, Speech and Signal Processing*, pages 437–440, 2003.

Masataka Goto and S. Hayamizu. A real-time music scene description system: Detecting melody and bass lines in audio signals. In *Working Notes of the IJCAI-99 Workshop on Computation Auditory Scene Analysis*, pages 31–40, 1999.

Steven W. Hainsworth. *Techniques for the Automated Analysis of Musical Audio*. PhD thesis, University of Cambridge, 2003.

Fredric J. Harris. On the use of windows for harmonic analysis with the discrete Fourier transform. *Proceedings of the IEEE*, 66(1):51–83, January 1978.

Christopher Harte and Mark Sandler. Automatic chord identifcation using a quantised chromagram. In *Proceedings of 118th Convention*. Audio Engineering Society, 2005.

Christopher Harte, Mark Sandler, Samer A. Abdallah, and Emilia Gómez. Symbolic representation of musical chords: A proposed syntax for text annotations. In *Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005, London, UK*, pages 66–71, 2005.

Christopher Harte, Mark Sandler, and Martin Gasser. Detecting harmonic change in audio. In *Proceedings of the 1st ACM workshop on Audio and music computing multimedia*, 2006.

Joel Hirschhorn. *The complete idiot's guide to songwriting*. Marie Butler-Knight, 2nd edition, 2004.

Qian Huang and B. Dom. Quantitative methods of evaluating image segmentation. In *Proceedings of the International Conference on Image Processing, 1995*, volume 3, pages 53–56, 1995.

David Huron. *Sweet Anticipation: Music and the Psychology of Expectation*. MIT Press, 2006.

Kristoffer Jensen. Multiple scale music segmentation using rhythm, timbre, and harmony. *EURASIP Journal on Advances in Signal Processing*, 2007.

Kristoffer Jensen, Jieping Xu, and Martin Zachariasen. Rhythm-based segmentation of popular Chinese music. In *Proceedings of the 6th International Conference on Music Information Retrieval*, London, 2005.

Michael Kennedy. *The Concise Oxford Dictionary of Music*. Oxford University Press, third edition, 1980.

Maksim Khadkevich and Maurizio Omologo. Phase-change based tuning for automatic chord recognition. In *Proceedings of the 12th International Conference on Digital Audio Effects (DAFx-09), Como, Italy*, 2009a.

Maksim Khadkevich and Maurizio Omologo. Use of hidden Markov models and factored language models for automatic chord recognition. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, 2009b.

A. Klapuri. Automatic music transcription as we know it today. *Journal of New Music Research*, 33(3):269–282, September 2004.

Anssi Klapuri. *Introduction to Music Transcription*, chapter 1. Springer Science + Business Media, 2006a.

Anssi P. Klapuri. Multiple fundamental frequency estimation by summing harmonic amplitudes. In *Proceedings of the 7th International Conference on Music Information Retrieval, ISMIR 2006, Victoria, Canada*, pages 216–221, 2006b.

Anssi P. Klapuri. A method for visualizing the pitch content of polyphonic music signals. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 615–620, 2009.

C. L. Krumhansl. *Cognitive Foundations of Musical Pitch.* Oxford University Press, 1990.

Richard Lawn and Jeffrey L. Hellmer. *Jazz: theory and practice*. Alfred Publishing, 1996.

C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems*, chapter 23. Prentice-Hall, 1974.

Kyogu Lee and Malcolm Slaney. Acoustic Chord Transcription and Key Extraction From Audio Using Key-Dependent HMMs Trained on Synthesized Audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):291–301, February 2008.

Kyogu Lee and Malcolm Slaney. A Unified System for Chord Transcription and Key Extraction Using Hidden Markov Models. In *Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR 2007, Vienna, Austria*, 2007.

Randal J. Leistikov. *Bayesian Modeling of Musical Expectations via Maximum Entropy Stochastic Grammars*. PhD thesis, Department of Music, Stanford University, June 2006.

Marc Leman. Schema-based tone center recognition of musical signals. *Journal of New Music Research*, 23(2):169–204, 1994.

Fred Lerdahl. *Tonal Pitch Space*. Oxford University Press, 2001.

Mark Levy and Mark Sandler. Structural segmentation of musical audio by constrained clustering. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(2):318–326, 2008.

Lie Lu, Muyuan Wang, and Hong-Jiang Zhang. Repeating pattern discovery and structure analysis from acoustic music ... In *Proceedings of the 6th ACM SIGMM international workshop on multimedia information retrieval*, 2004.

N. C. Maddage, C. Xu, M. S. Kankanhalli, and X. Shao. Content-based music structure analysis with applications to music semantics understanding. In *Proceedings of the 12th annual ACM international conference on Multimedia*, 2004.

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

Christopher D. Manning and Hinrich Schütze. *Foundations of Natural Language Processing*. MIT Press, 1999.

M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler. OMRAS2 metadata project 2009. In *Late-breaking session at the 10th International Conference on Music Information Retrieval, Kobe, Japan*, 2009a. `http://ismir2009.ismir.net/proceedings/LBD-18.pdf`.

Matthias Mauch and Simon Dixon. A Discrete Mixture Model for Chord Labelling. In *Proceedings of the 9th International Conference on Music Information Retrieval, ISMIR 2008, Philadelphia, USA*, pages 45–50, 2008.

Matthias Mauch and Simon Dixon. Simultaneous estimation of chords and musical context from audio. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(2), February 2010.

Matthias Mauch, Katy Noland, and Simon Dixon. MIREX submissions for audio chord detection (no training) and structural segmentation. In *MIREX Submission Abstracts*. 2009b. URL `http://music-ir.org/mirex/2009/results/abs/ACD_SS_mauch.pdf`.

Matthias Mauch, Katy C. Noland, and Simon Dixon. Using musical structure to enhance automatic chord transcription. In *Proceedings of the 10th International Conference on Music Information Retrieval, Kobe, Japan*, pages 231–236, 2009c.

F. Mercury, J. Deacon, B. H. May, and R. M. Taylor. *Greatest Hits II: Top Line and Chorus*. Queen Music Ltd./EMI Music Publishing, 1992.

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

Meinhard Müller, Sebastian Ewert, and Sebastian Kreuzer. Making chroma features more robust to timbre changes. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1877–1880, 2009.

Kevin P. Murphy. The Bayes Net Toolbox for Matlab. *Computing Science and Statistics*, 33(2): 1024–1034, 2001.

Kevin P Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, University of California, Berkeley, 2002.

Katy Noland. *Computational Tonality Estimation: Signal Processing and Hidden Markov Models*. PhD thesis, Queen Mary University of London, 2009.

Katy Noland and Mark Sandler. Influences of signal processing, tone profiles, and chord progressions on a model for estimating the musical key from audio. *Computer Music Journal*, 33(1), 2009.

A. M. Noll. Pitch determination of human speech by the harmonic product spectrum, the harmonic sum spectrum, and a maximum likelihood estimate. *Proceedings of the Symposium on Computer Processing in Communications*, 13:208–214, 1970.

M. Ogihara and T. Li. $n$-gram chord profiles for composer style representation. In *Proceedings of the 9th International Conference on Music Information Retrieval, ISMIR 2008, Philadelphia, USA*, pages 671–676, 2008.

Bee Suan Ong. *Structural Analysis and Segmentation of Music Signals*. PhD thesis, Universitat Pompeu Fabra, 2006.

N. Ono, K. Miyamoto, H. Kameoka, and S. Sagayama. A real-time equalizer of harmonic and percussive components in music signals. In *Proceedings of the 9th International Conference on Music Information Retrieval, ISMIR 2008, Philadelphia, USA*, pages 139–144, 2008.

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

Jean-François Paiement, Douglas Eck, and Samy Bengio. A probabilistic model for chord progressions. In *Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005, London, UK*, pages 312–319, 2005.

Hélène Papadopoulos and Geoffroy Peeters. Large-scale Study of Chord Estimation Algorithms Based on Chroma Representation and HMM. In *International Workshop on Content-Based Multimedia Indexing*, pages 53–60, 2007.

Hélène Papadopoulos and Geoffroy Peeters. Simultaneous estimation of chord progression and downbeats from an audio file. In *Proceedings of the 2008 ICASSP Conference*, pages 121–124, 2008.

Bryan Pardo. Algorithms for chordal analysis. *Computer Music Journal*, 26(2):27–49, 2002.

Bryan Pardo and William P. Birmingham. Following a musical performance from a partially specified score. In *Proceedings of the Multimedia Technology and Applications Conference*, 2001.

Richard Parncutt. *Harmony*. Springer Verlag Berlin, 1989.

A.D. Patel. Language, music, syntax and the brain. *Nature Neuroscience*, 6(7):674–681, 2003.

Jouni Paulus and Anssi Klapuri. Music structure analysis by finding repeated parts. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*, pages 59–68, 2006.

Jouni Paulus and Anssi Klapuri. Music structure analysis using a probabilistic fitness measure and a greedy search algorithm. *IEEE Trans. Audio, Speech, and Language Processing*, 17 (6):1159–1170, 2009.

G. Peeters, A. La Burthe, and Xavier Rodet. Toward automatic music audio summary generation from signal analysis. In *Proceedings of the 3rd International Conference on Music Information Retrieval*, Paris, 2002.

Geoffroy Peeters. Chroma-based estimation of musical key from audio-signal analysis. In *Proceedings of the 7th International Conference on Music Information Retrieval, ISMIR 2006, Victoria, Canada*, 2006.

Geoffroy Peeters and Emmanuel Deruty. Toward music structure annotation. In *International Society for Music Information Retrieval Conference, Late-Breaking Session*. 2009. `http://ismir2009.ismir.net/proceedings/LBD-7.pdf`.

Jeremy Pickens and Tim Crawford. Harmonic models for polyphonic music retrieval. In *CIKM '02: Proceedings of the 11th International Conference on Information and Knowledge Management*, pages 430–437, New York, NY, USA, 2002. ACM Press.

G. Poliner, D. Ellis, A. F. Ehmann, E. Gómez, S. Streich, and B. Ong. Melody transcription from music audio: approaches and evaluation. *IEEE Trans. Audio, Speech, and Language Processing*, 15(4):1247–1256, May 2007.

Alan W. Pollack. Notes on... series, 1995. Available at `http://www.recmusicbeatles.com`.

Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Don Michael Randel. *The Harvard dictionary of music*. Harvard University Press, 4 edition, 2003.

C. Raphael and J. Stoddard. Harmonic analysis with probabilistic graphical models. In *Proceedings of the 4th International Conference on Music Information Retrieval, ISMIR 2003, Baltimore, USA*, 2003.

Christopher Raphael. A graphical model for recognizing sung melodies. In *Proceedings of the 6th International Conference on Music Information Retrieval, ISMIR 2005, London, UK*, pages 658–663, 2005.

Robert Rawlins and Nor Eddine Bahha. *Jazzology*. Hal Leonard Corporation, 2005.

J.T. Reed, Yushi Ueda, S. Siniscalchi, Yuki Uchiyama, Shigeki Sagayama, and C.-H. Lee. Minimum classification error training to improve isolated chord recognition. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, pages 609–614, 2009.

Johannes Reinhard, Sebastian Stober, and Andreas Nürnberger. Enhancing chord classification through neighbourhood histograms. In *International Workshop on Content-Based Multimedia Indexing, CBMI*, pages 33–40. 2008.

Christophe Rhodes and Michael Casey. Algorithms for determining and labelling approximate hierarchical self-similarity. In *Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR 2007, Vienna, Austria*, pages 41–46, 2007.

Christophe Rhodes, David Lewis, and Daniel Müllensiefen. Bayesian Model Selection for Harmonic Labelling. In *Proceedings of the International Conference on Mathematics and Computation in Music*, May 2007.

Thomas D. Rossing. *The Science of Sound*. Addison-Wesley Publishing Company, 2 edition, 1990.

Matti P. Ryynänen. *Automatic Transcription of Pitch Content in Music and Selected Applications*. PhD thesis, Tampere University of Technology, December 2008.

Matti P. Ryynänen and Anssi P. Klapuri. Automatic Transcription of Melody, Bass Line, and Chords in Polyphonic Music. *Computer Music Journal*, 32(3):72–86, 2008.

Shigeki Sagayama, Keigo Takahashi, Hirokazu Kameoka, and Takuya Nishimoto. Specmurt anasylis: A piano-roll-visualization of polyphonic music signal by deconvolution of log-frequency spectrum. In *Proceedings of the 2004 ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing*, 2004.

Craig Sapp. Computational chord-root identification in symbolic musical data: Rationale, methods, and applications. In Walter B. Hewlett, Eleanor Selfridge-Field, and Edmund Correia, editors, *Tonal Theory for the Digital Age*. Center for Computer Assisted Research in the Humanities at Stanford University, 2007.

Ricardo Scholz and Geber Ramalho. COCHONUT: Recognizing complex chords from midi guitar sequences. In *Proceedings of the 9th International Conference on Music Information Retrieval, ISMIR 2008, Philadelphia, USA*, pages 27–32, 2008.

Ricardo Scholz, Emmanuel Vincent, and Frédéric Bimbot. Robust modeling of musical chord sequences using probabilistic n-grams. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2009.

Alexander Sheh and Daniel Ellis. Chord segmentation and recognition using EM-trained hidden Markov models. In Holger H. Hoos and David Bainbridge, editors, *Proceedings of the 4th International Conference on Music Information Retrieval, ISMIR 2003, Baltimore, USA*, 2003.

Arun Shenoy and Ye Wang. Key, chord, and rhythm tracking of popular music recordings. *Computer Music Journal*, 29(3):75–86, September 2005.

Roger Shepard. Circularity in judgements of relative pitch. *Journal of the Acoustical Society of America*, 36:2346–2353, 1964.

Julius O. Smith. *Spectral Audio Signal Processing, October 2008 Draft*. online book, 2008. `https://ccrma.stanford.edu/~jos/sasp/`.

B. Su and S. Jeng. Multi-timbre chord classification using wavelet transform and self-organized map neural networks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3377–3380, 2001.

K. Sumi, K. Itoyama, K. Yoshii, K. Komatani, T. Ogata, and H.G. Okuno. Automatic chord recognition based on probabilistic integration of chord transition and bass pitch estimation.

In *Proceedings of the 9th International Conference on Music Information Retrieval, ISMIR 2008, Philadelphia, USA*, 2008.

David Temperley and Daniel Sleator. Modeling Meter and Harmony: A Preference-Rule Approach. *Computer Music Journal*, 25(1):10–27, 1999.

William Forde Thompson. Modeling perceived relationships between melody, harmony, and key. *Perception & Psychophysics*, 53(1):13–24, 1993.

Dan Tidhar, Matthias Mauch, and Simon Dixon. High precision frequency estimation for harpsichord tuning classification. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2010.

John Wade Ulrich. The analysis and synthesis of jazz by computer. In *Proceedings of the 5th International Joint Conference on Artifcial Intelligence. Los Altos*, 1977.

Matthias Varewyck, Johan Pauwels, and Jean-Pierre Martens. A novel chroma representation of polyphonic music based on multiple pitch tracking techniques. In *Proceedings of the 16th ACM International Conference on Multimedia*, pages 667–670, 2008.

Gregory H. Wakefield. Mathematical representation of joint time-chroma distributions. In *Proceedings of SPIE*, volume 3807, 1999.

Jan Weil and J. L. Durrieu. An HMM-based audio chord detection system attenuating the main melody. In *MIREX Submission Abstracts*. 2008. `http://www.music-ir.org/mirex/2008/abs/Mirex08_AudioChordDetection_Weil_Durrieu.pdf`.

Jan Weil, J. L. Durrieu, and Gaël Richard. Automatic generation of lead sheets from polyphonic music signals. In *Proceedings of the 10th International Society for Music Information Retrieval Conference (ISMIR 2009)*, 2009.

Adrian Weller, Dan Ellis, and Tony Jebara. Structured prediction models for chord transcription of music audio. In *MIREX Submission Abstracts*. 2009. `http://www.cs.columbia.edu/~jebara/papers/icmla09adrian.pdf`.

Paul Westwood. *Bass Bible: a world history of styles and techniques*. AMA Verlag, 1997.

Keith Wyatt and Carl Schroeder. *Harmony and Theory: A Comprehensive Source for All Musicians*. Hal Leonard Corporation, 1998.

Takuya Yoshioka, Tetsuro Kitahara, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. Automatic chord transcription with concurrent recognition of chord symbols and boundaries. In *Proceedings of the 5th International Conference on Music Information Retrieval, ISMIR 2004, Barcelona, Spain*, pages 100–105, 2004.

Veronika Zenz and Andreas Rauber. Automatic chord detection incorporating beat and key detection. In *Proceedings of the 2007 IEEE International Conference on Signal Processing and Communications (ICSPC 2007)*, 2007.

X. Zhang and D. Gerhard. Chord Recognition using Instrument Voicing Constraints. In *Proceedings of the 9th International Conference on Music Information Retrieval, ISMIR 2008, Philadelphia, USA*, 2008.

Udo Zölzer, editor. *DAFX - Digital Audio Effects*. John Wiley Sons, 2002.