Melody Extraction from Polyphonic Music Signals

Justin J. Salamon

TESI DOCTORAL UPF / 2013

Thesis advisors

Dr. Emilia Gómez Gutiérrez
Dr. Xavier Serra i Casals
Department of Information and Communication Technologies
Universitat Pompeu Fabra, Barcelona, Spain

Dissertation submitted to the Department of Information and Communication Technologies of Universitat Pompeu Fabra in partial fulfilment of the requirements for the degree of

DOCTOR PER LA UNIVERSITAT POMPEU FABRA,

with the mention of European Doctor.

Music Technology Group (http://mtg.upf.edu), Dept. of Information and Communication Technologies (http://www.upf.edu/dtic), Universitat Pompeu Fabra (http://www.upf.edu), Barcelona, Spain.

To Judi, Ruben, Guy, Hedi and Amalia.

Acknowledgements

What an incredible journey this has been. And like every good journey, it is marked not only by the paths I have taken or the things I have accomplished, but also, and perhaps most importantly, by the people who have accompanied me along the way. This particular journey began a few years ago, when Xavier Serra welcomed me into the Music Technology Group. I am extremely grateful to him for giving me this opportunity, and for accompanying me in this journey ever since. Even before starting at the MTG, I was interested in the computational aspects of melody, melodic similarity and retrieval (perhaps it was the combined effect of being a jazz-fusion guitarist and a computer scientist). Once at the MTG, it very quickly became evident to me that if I wanted to explore this area, there was no better person to do it with than Emilia Gómez. Now, as I near the end of this journey, I can say I feel extremely fortunate to have had Emilia to guide me through it. Thank you so much for your guidance, advice, friendship and support.

During most of my time at the MTG, I have had the privilege of sharing an office with Perfecto Herrera, Joan Serrà and Enric Guaus. Being able to debate my ideas with them and receive comments and suggestions regarding my work has been invaluable. They have also been very tolerant of my victory dance whenever a paper of mine was accepted for publication. After a few years in sunny Barcelona, I also got the chance to work for several months in autumnal Paris with Axel Röbel and Geoffroy Peeters. This was a great learning opportunity, and I would like to thank them and the entire Sound Analysis/Synthesis team at IRCAM for welcoming me there and making it a highly enriching experience.

One of the great things about working at the MTG is the opportunity to collaborate with a fantastic group of colleagues. It was a great pleasure to work and co-author with (in addition to Emilia, Xavier, Joan and Perfe) Jordi Bonada, Sankalp Gulati, Bruno Rocha, Dmitry Bogdanov, Jose Ricardo Zapata, Gerard Roma, Oscar Mayor and Nicolas Wack. I would also like to thank many other past and present members of the MTG who have been an important part of my stay here, both academically and socially. In no particular order, I would like to thank Martin Haro, Ferdinand Fuhrmann, Cyril Laurier, Ricard Marxer, Saso Musevic, Jordi Janer, Agustín Martorell, Sergi Jordà, Carles F. Juliá, Daniel Gallardo, Sebastián Mealla, Mathieu Bosi, Mohamed Sordo, Frederic Font, Juanjo Bosch, Sergio Giraldo, Marco Marchini, Panos Papiotis, Alastair Porter, Alvaro Sarasua, Marti Umbert, Gopala Krishna Koduri, Sertan Şentürk, Ajay Srinivasamurthy, Rafael Caro, Graham Coleman, Piotr Holonowicz, Ines Salselas, Merlijn Blaauw, Stefan Kersten, Esteban Maestre, Waldo Nogueira, Zacharias Vamvakousis, Rafael Ramírez, Pedro Cano, Oscar Celma, Maarten de Boer, Paul Brossier, Roberto Toscano, Hendrik Purwins, Fernando Villavicencio, Vincent Akkermans, Marcos Alonso, Amaury Hazan, Owen Meyers and Alfonso Pérez. Thanks Uri, Vassilis, Elena, Paqui, Marcelo, Yotam, Alex and Mike for helping me kick-start this journey in the best way possible. A special thanks to Alba Rosado, Cristina Garrido, Sonia Espí and Lydia Garcia for really knowing their stuff. Sorry if I have left anyone out!

I have also had the pleasure of collaborating with researchers outside the MTG, including (in addition to Geoffroy and Axel) Dan Ellis, Gaël Richard, Julián Urbano, Martin Rohrmeier and the flamenco team: Francisco Gómez, Aggelos Pikrakis, Joaquín Mora, José Miguel Díaz-Báñez, Francisco Escobar, Sergio Oramas, Francisco Cañadas, Pablo Cabañas and Pedro Vera. In addition, I would like to thank the following researchers for some very interesting conversations, feedback and willingness to help: Karin Dressler, Jean-Louis Durrieu, Masataka Goto, Anssi Klapuri, Matti Ryynänen, Rui Pedro Paiva, Benoit Fuentes, Jyh-Shing Roger Jang and Meinard Müller.

But this journey was not only an academic endeavour, and over the years and I have met some very special people who have helped make Barcelona my home. They include the people of Santa Anna 37 (Uri, Jordi, Leslie and April), Muntaner 24, and the members of my band Terapia de Groove. A special thanks to Uri (Oriol Nieto) for helping with the translation of the abstract, and to Shaul Tor for always being there for a consult. Last but not least, I would like to say thank you to my family in Israel and the UK – Judi (who also proof read parts of this thesis), Ruben, Guy and Hedi for their endless love and support. And to Amalia, my family here.

This thesis has been carried out at the Music Technology Group of Universitat Pompeu Fabra (UPF) in Barcelona, Spain between Sep. 2008 and Aug. 2011 and between Jan. 2012 and July. 2013, and at the Sound Analysis/Synthesis research team of the Institut de Recherche et Coordination Acoustique/Musique (IRCAM) in Paris, France between Sep. 2011 and Dec. 2011. This work has been supported by the Programa de Formación del Profesorado Universitario (FPU) of the Spanish Ministry of Education and by the European Commission, FP7 (Seventh Framework Programme), ICT-2011.1.5 Networked Media and Search Systems, grant agreement No. 287711 (MIRES).

Abstract

Music was the first mass-market industry to be completely restructured by digital technology, and today we can have access to thousands of tracks stored locally on our smartphone and millions of tracks through cloud-based music services. Given the vast quantity of music at our fingertips, we now require novel ways of describing, indexing, searching and interacting with musical content. In this thesis we focus on a technology that opens the door to a wide range of such applications: automatically estimating the pitch sequence of the melody directly from the audio signal of a polyphonic music recording, also referred to as melody extraction. Whilst identifying the pitch of the melody is something human listeners can do quite well, doing this automatically is highly challenging. We present a novel method for melody extraction based on the tracking and characterisation of the pitch contours that form the melodic line of a piece. We show how different contour characteristics can be exploited in combination with auditory streaming cues to identify the melody out of all the pitch content in a music recording using both heuristic and model-based approaches. The performance of our method is assessed in an international evaluation campaign where it is shown to obtain state-of-the-art results. In fact, it achieves the highest mean overall accuracy obtained by any algorithm that has participated in the campaign to date. We demonstrate the applicability of our method both for research and end-user applications by developing systems that exploit the extracted melody pitch sequence for similarity-based music retrieval (version identification and query-by-humming), genre classification, automatic transcription and computational music analysis. The thesis also provides a comprehensive comparative analysis and review of the current state-of-the-art in melody extraction and a first of its kind analysis of melody extraction evaluation methodology.

Resumen

La industria de la música fue una de las primeras en verse completamente reestructurada por los avances de la tecnología digital, y hoy en día tenemos acceso a miles de canciones almacenadas en nuestros dispositivos móviles y a millones más a través de servicios en la nube. Dada esta inmensa cantidad de música al nuestro alcance, necesitamos nuevas maneras de describir, indexar, buscar e interactuar con el contenido musical. Esta tesis se centra en una tecnología que abre las puertas a nuevas aplicaciones en este área: la extracción automática de la melodía a partir de una grabación musical polifónica. Mientras que identificar la melodía de una pieza es algo que los humanos pueden hacer relativamente bien, hacerlo de forma automática presenta mucha complejidad, ya que requiere combinar conocimiento de procesado de señal, acústica, aprendizaje automático y percepción sonora. Esta tarea se conoce en el ámbito de investigación como "extracción de melodía", y consiste técnicamente en estimar la secuencia de alturas correspondiente a la melodía predominante de una pieza musical a partir del análisis de la señal de audio. Esta tesis presenta un método innovador para la extracción de la melodía basado en el seguimiento y caracterización de contornos tonales. En la tesis, mostramos cómo se pueden explotar las características de contornos en combinación con reglas basadas en la percepción auditiva, para identificar la melodía a partir de todo el contenido tonal de una grabación, tanto de manera heurística como a través de modelos aprendidos automáticamente. A través de una iniciativa internacional de evaluación comparativa de algoritmos, comprobamos además que el método propuesto obtiene resultados punteros. De hecho, logra la precisión más alta de todos los algoritmos que han participado en la iniciativa hasta la fecha. Además, la tesis demuestra la utilidad de nuestro método en diversas aplicaciones tanto de investigación como para usuarios finales, desarrollando una serie

de sistemas que aprovechan la melodía extraída para la búsqueda de música por semejanza (identificación de versiones y búsqueda por tarareo), la clasificación del estilo musical, la transcripción o conversión de audio a partitura, y el análisis musical con métodos computacionales. La tesis también incluye un amplio análisis comparativo del estado de la cuestión en extracción de melodía y el primer análisis crítico existente de la metodología de evaluación de algoritmos de este tipo.

Resum

La indústria musical va ser una de les primeres a veure's completament reestructurada pels avenços de la tecnologia digital, i avui en dia tenim accés a milers de cançons emmagatzemades als nostres dispositius mòbils i a milions més a través de serveis en xarxa. Al tenir aquesta immensa quantitat de música al nostre abast, necessitem noves maneres de descriure, indexar, buscar i interactuar amb el contingut musical. Aquesta tesi es centra en una tecnologia que obre les portes a noves aplicacions en aquesta àrea: l'extracció automàtica de la melodia a partir d'una gravació musical polifònica. Tot i que identificar la melodia d'una peça és quelcom que els humans podem fer relativament fàcilment, fer-ho de forma automàtica presenta una alta complexitat, ja que requereix combinar coneixement de processament del senyal, acústica, aprenentatge automàtic i percepció sonora. Aquesta tasca es coneix dins de l'àmbit d'investigació com a "extracció de melodia", i consisteix tècnicament a estimar la seqüència de altures tonals corresponents a la melodia predominant d'una peça musical a partir de l'anàlisi del senyal d'àudio. Aquesta tesi presenta un mètode innovador per a l'extracció de la melodia basat en el seguiment i caracterització de contorns tonals. Per a fer-ho, mostrem com es poden explotar les característiques de contorns combinades amb regles basades en la percepció auditiva per a identificar la melodia a partir de tot el contingut tonal d'una gravació, tant de manera heurística com a través de models apresos automàticament. A més d'això, comprovem a través d'una iniciativa internacional d'avaluació comparativa d'algoritmes que el mètode proposat obté resultats punters. De fet, obté la precisió més alta de tots els algoritmes proposats fins la data d'avui. A demés, la tesi demostra la utilitat del mètode en diverses aplicacions tant d'investigació com per a usuaris finals, desenvolupant una sèrie de sistemes que aprofiten la melodia extreta per a la cerca de música per semblança

(identificació de versions i cerca per taral·larà), la classificació de l'estil musical, la transcripció o conversió d'àudio a partitura, i l'anàlisi musical amb mètodes computacionals. La tesi també inclou una àmplia anàlisi comparativa de l'estat de l'art en extracció de melodia i la primera anàlisi crítica existent de la metodologia d'avaluació d'algoritmes d'aquesta mena.

Contents

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations

Abbreviation	Description
ACF	Autocorrelation function
ADC	Audio description contest
AME	Audio melody extraction
ANOVA	Analysis of variance
ASA	Auditory scene analysis
CASA	Computational auditory scene analysis
EMS	Expected mean square
FFT	Fast Fourier transform
GMM	Gaussian mixture model
HMM	Hidden Markov model
HPCP	Harmonic pitch class profile
HPSS	Harmonic-percussive sound separation
IF	Instantaneous frequency
ISMIR	International society for music information retrieval conference
MAP	Mean average precision
MaRR	Mean averaged reciprocal rank
MIDI	Musical instrument digital interface
MIR	Music information retrieval
MIREX	Music information retrieval evaluation exchange

continued on the next page...

Abbreviation	Description
MRFFT	Multi-resolution fast Fourier transform
MRR	Mean reciprocal rank
QBH	Query-by-humming
RR	Reciprocal rank
STFT	Short-time Fourier transform

Mathematical symbols

Symbol	Description
$\mathcal{A}$	Set of algorithms
$\hat{a}$	Estimated amplitude
$\mathcal{B}$	Frequency mapping function from Hz to salience function bin
b	Index
$\mathcal{C}$	Frequency mapping function from Hz to cents
C_l	Contour length
$C_{\bar{p}}$	Contour pitch mean
C_{σ_p}	Contour pitch deviation
$C_{\Sigma s}$	Contour total salience
$C_{\bar{s}}$	Contour mean salience
C_{σ_s}	Contour salience deviation
C_r	Contour pitch range
C_v	Contour vibrato presence
c_{tuning}	Tuning deviation in cents
d	Distance between FFT bin and maximum of fitted parabola
$\mathcal{E}$	Magnitude threshold function
$\hat{\mathbf{f}}$	Estimated frequency sequence
$\mathbf{f}^*$	Ground truth frequency sequence
f	Frequency in Hz
$\hat{f}$	Estimated frequency in Hz
f_0	Fundamental frequency in Hz
f_c	Frequency in cents
f_{ref}	Reference frequency for Hz to cents conversion
f_S	Sampling frequency in Hz
f_{tuning}	Tuning frequency in Hz
$\mathcal{G}$	Harmonic weighting function

continued on the next page...

Symbol	Description
G_{pmax}	Global highest melody pitch
G_{pmin}	Global lowest melody pitch
G_r	Global melody pitch range
g	Harmonic weighting function
H	Hop size used in STFT
h	Index
$\hbar$	Pitch histogram peak
I	Number of detected spectral peaks
i	Index
j	Imaginary unit
k	Index
$\mathcal{L}$	Likelihood
L	Total number of frames in a melody pitch sequence
l	Index
$\mathcal{M}$	"Melodiness" index
M	Window length for STFT
$\mathcal{N}$	Multivariate normal distribution
N	FFT length
N_h	Number of harmonics used in harmonic summation
N_c	Contour length
N_F	Number of contour features
n	Index
$\mathcal{P}$	Melody mean pitch trend
P	Fourier transform of p
p	Contour pitch sequence in cents
$\mathcal{Q}$	Set of songs
r_u	Rank of single correct target for song u in ordered song list
$r_{i\sim u}$	Rank of i^{th} correct target for song u in ordered song list
R_e	Energy recall
R_p	Peak recall
$\mathcal{S}$	Salience function
S_y	Pitch score function for monophonic signals
S'_x	Pitch score function for polyphonic signals
S_1	Relative melody salience compared to top salience peak
S_3	Relative melody salience compared to top 3 salience peaks
S^+	Set of predominant salience peaks
S^-	Set of weak salience peaks

continued on the next page...

Symbol	Description
s	Contour salience sequence
$\mathcal{T}$	Pitch distance threshold function
t	Time
u	Song index
U	Song collection size
$\mathcal{V}$	Version set cardinality
V_r	Vibrato rate
V_e	Vibrato extent
V_c	Vibrato coverage
$\mathbf{v}$	Estimated voicing indicator vector
$\mathbf{v}^*$	Ground truth voicing indicator vector
v	Voicing indicator, element of $\mathbf{v}$
$\bar{v}$	Unvoicing indicator
W	Autocorrelation analysis window length
W_{Hann}	Hann window kernel
w	windowing function
$\mathbf{x}$	Pitch contour feature vector
x	Polyphonic audio signal
X	Fourier transform of x
X_m	Normalised magnitude spectrum of x
X_{dB}	Normalised magnitude spectrum of x in dB
y	Monophonic melody audio signal
$\hat{y}$	Separated melody audio signal
α	Harmonic weighting parameter
β	Magnitude compression parameter
Γ	Relevance function
γ	Magnitude threshold
Δ	Difference (error)
δ	Distance in semitones
ε	Residual
ζ	Contour type
η	Accompaniment audio signal
θ	Multivariate normal distribution parameters
κ	FFT bin offset
Λ	Ordered song list
λ	Power parameter used in Box-Cox transform

continued on the next page...

Symbol	Description
$\boldsymbol{\mu}$	Mean vector
μ	Mean value
ν	Voicing parameter
ρ	Pitch histogram distance feature
$\boldsymbol{\Sigma}$	Covariance matrix
σ	Standard deviation
σ^2	Variance
$\widehat{\sigma}^2$	Estimated variance component
τ	Index
τ_σ	Weighting factor for salience peak filtering
τ_+	Weighting factor for salience peak filtering
τ_v	Voicing threshold
Φ	Dependability index
$\widehat{\Phi}$	Estimated dependability index
ϕ	Phase spectrum
φ	Score
Ψ	Principal argument function
ψ	Precision of ordered song list
$\overline{\psi}$	Average precision of ordered song list

Chapter 1

Introduction

1.1 Motivation

> *"Melody, defined as pitched sounds arranged in musical time in accordance with given cultural conventions and constraints, represents a universal human phenomenon traceable to prehistoric times."* (Ringer, 2013)

The human "melodic impulse" manifests itself in a wide range of aspects in our lives, starting with an infants first cry, social intercourse and communication, and of course music. Even without going back to prehistoric times, it is clear that melody has had an important role in our experience of music throughout modern history, and continues to do so today. Melodic motifs stand at the core of many Western music compositions, as well as non-Western music traditions such as Indian classical music and orally transmitted music traditions such as flamenco. When it comes to modern popular music, the melody can be of particular importance, to such an extent that many songs are directly composed with a "melody plus accompaniment" paradigm in mind. As listeners, we often reproduce the melody when we wish to identify a specific musical piece:

> *"It is melody that enables us to distinguish one work from another. It is melody that human beings are innately able to reproduce by singing, humming, and whistling. It is melody that makes music memorable: we are likely to recall a tune long after we have forgotten its text.".* (Selfridge-Field, 1998)

The importance of the melody phenomenon as a part of the human musical experience makes it a highly interesting topic of study, also from a computational point of view. Building computational systems which imitate human behaviour can provide new insights and increase our understanding of a given phenomenon, even when the underlying system implementation differs from its physiological human counterpart (Sun, 2008). In this thesis, we focus on a specific aspect of our interaction with melodies – our ability to identify the pitch sequence that constitutes the melody of a piece when listening to a music recording. To be specific, this thesis deals with the automatic extraction of the pitch sequence of the melody directly from the audio signal of a polyphonic music recording, henceforth *melody extraction*. The problem is illustrated in Figure 1.1: given the audio signal of a polyphonic music recording, we want the system to return a sequence of fine grained pitch values which describe the pitch of the melody at every given moment in time.

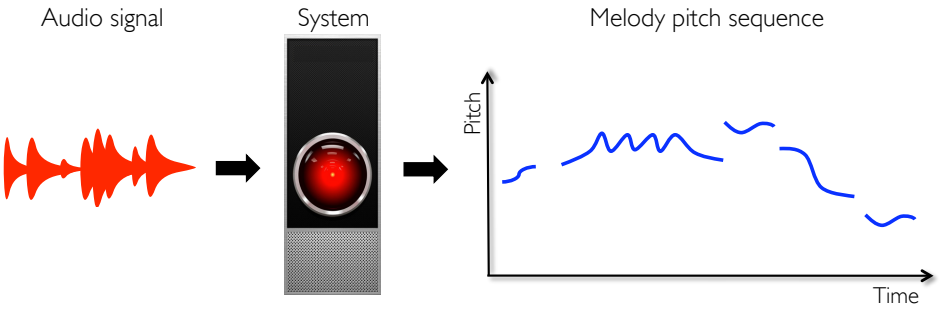

Figure 1.1: Melody extraction problem illustration: going from the audio signal of a polyphonic music recording to a sequence of values representing the pitch of the melody at every given moment in time.

Whilst valuable insights may be gained simply by building such a system, our main motivation for studying automatic melody extraction is more practical in nature. Even though scores have been published for many musical pieces, they represent but a fraction of the amount of recorded music material now in existence. If we are able to automatically obtain a description of the melody of a piece of music directly from its audio recording, we immediately open the door to a wide range of applications that can exploit this particular kind of information. Starting with end users, a clear application area is music retrieval. Automatically compiling large databases of melodies would allow users to search for a song by singing or humming part

of its melody (Salamon et al., 2013b). We could also automatically compare melodies of different recordings to detect whether they are renditions of the same musical piece (Foucard et al., 2010; Salamon et al., 2013b), which could be used for retrieval purposes ("find me other versions of this song") or for resolving copyright disputes. By analysing the characteristics of the obtained melodies we could train a system to identify the musical genre of a song based on its melody, thus helping users to automatically organise their music collections (Salamon et al., 2012c). Additional end-user applications include music de-soloing for the automatic generation of karaoke accompaniment (Durrieu et al., 2009) and singer identification (Mesaros et al., 2007).

Automatically obtaining a representation of the melody also has a wide range of applications in computational music analysis. Melody extraction is a necessary first step for automatic transcription into score notation (Gómez et al., 2012). It is also a precursor for automating tasks such as intonation analysis (Koduri et al., 2012b) and melodic motif and pattern analysis (Pikrakis et al., 2012). Extracting the melody of a song could also be used as an intermediate step towards the derivation of semantic labels from music signals (Serra et al., 2007). Finally, melody extraction also has a variety of uses outside the realm of research, such as electroacoustic composition and music education.

1.2 Some important definitions

This thesis deals with aspects of melody, pitch, and polyphonic music. As such, it is essential that we have a clear understanding of these terms and how they will be used throughout this dissertation.

1.2.1 Melody

Definition

Our primary topic of interest is the concept of melody – extracting the melody, characterising it, using it for other applications. To do so, we must first have a clear understanding of *what is the melody?* Going back to the quotation at the beginning of this chapter, we see that Ringer (2013) defines the melody as:

> *"pitched sounds arranged in musical time in accordance with given cultural conventions and constraints."*

From this definition it is evident that melody, a musicological concept, may not have a clear-cut definition that suits all the purposes in which we may wish to refer to it. Rather, melody is a concept based partly on the judgement of human listeners (Poliner et al., 2007), and as such, we can expect to find different definitions of melody depending on the context in which they are proposed. If we take a look at melody from a historical point of view, we see that the very definition and purpose of melody has continually evolved. In the mid-15th century, melody (*melus*) was identified with *cantus*, associating melody with song. In the 18th century, Rameau considered melody to be a product of harmony, whilst Rousseau argued melody was autonomous. In the 19th century, Hegel considered harmony and melody to be

> *"one compact whole, and a change in one necessarily involves a change in the other".*

For Helmholtz, melody was the expression of motion in music, expressed

> *"in such a manner that the hearer may easily, clearly, and certainly appreciate the character of that motion by immediate perception".*

Hanslick saw melody as the *"archetypal configuration of beauty"*, whilst Wagner postulated:

> *"an ordered series of quasi-intellectual, unfulfilled speech-sounds – indirectly representative, concentrated as image but not yet as immediate, inevitably true expression . . . directly addressed to feeling, unerringly vindicated and fulfilled".*

Later in the 20th century, Thurstone, a student of non-Western music, defined 'the essence of melody' as

> *"unity in the perception of pitch variation".*

Even nowadays, the debate on how to define melody is still open. Alongside Ringer's definition we can find others such as *"a combination of a pitch series and a rhythm having a clearly defined shape"* (Solomon, 1997) or *"musical sounds in a pleasant order and arrangement"* (Wordsmyth, 2013).

What is important to realise, is that this is not only a problem for musicologists. As scientists, if we wish to develop a system which can extract the melody from a music recording, and importantly, if we wish to be able to quantitatively and objectively evaluate the performance of our system, we too must have a clear definition of melody. Faced with this problem, the Music Information Retrieval (MIR) research community, and more specifically those researchers working on melody extraction, have proposed several definitions of their own (more on MIR in Section 1.4.1). Importantly, these definitions are designed to offer a pragmatic solution to the problem, in the form of simplified definitions which can be more directly translated into a signal processing problem.

One of the first to tackle melody extraction was Masataka Goto (Goto & Hayamizu, 1999). In this work, melody is defined as a

> *"series of notes* [*which*] *is more distinctly heard than the rest"*.

Gómez et al. (2003) provide an interesting discussion regarding the definition of melody. For the purposes of their work, they eventually consider melody as a *pitch sequence* with a series of attributes. Paiva et al. (2006) define melody as

> *"the dominant individual pitch line in a musical ensemble"*.

One of the most commonly referenced definitions in the literature is the one proposed by Poliner et al. (2007), who note that

> *"roughly speaking, the melody is the single (monophonic) pitch sequence that a listener might reproduce if asked to whistle or hum a piece of polyphonic music, and that a listener would recognize as being the 'essence' of that music when heard in comparison"*.

Whilst these definitions can be considered less ambiguous (which also means more limited) compared to those proposed by musicologists, we see that they still include a strong dependency on human perception. In his doctoral thesis, Durrieu (2010) attempts to reduce this dependency to a minimum by searching for a definition which relies, as much as possible, on the signal itself and its production, rather than its perception. He does so by breaking down the aforementioned definition proposed by Paiva et al. (2006), interpreting it in the following way:

- *music ensemble*: the signal is played by one or more instruments, leading to a possibly polyphonic mixture. There are no specific constraints on the ensemble, which can include both percussive and harmonic instruments.

- *pitched*: meaning unpitched percussive sounds are excluded as a possible source of the melody. Durrieu's work focuses on human voice as the lead instrument, but the melody need not be constrained to the human voice – other lead instruments (e.g. saxophone or violin) should also be considered.

- *individual*: the melody is defined to be monophonic, and played by a single instrument. Furthermore, both Paiva et al. (2006) and Durrieu (2010) restrict the melody to being played by the same instrument throughout the audio recording being analysed (so for example a guitar solo in a song with a lead singer would not be considered part of the melody).

- *line*: the melody is expected to exhibit a certain degree of smoothness (in time and pitch). This would be in accordance with voice leading rules derived from perceptual principles too (Huron, 2001).

- *dominant*: the melody should be, in some sense, more "dominant" compared to the other instruments in the mixture. This is where perception unavoidably comes back into play – it is the human listener who perceives some things as more dominant than others, and this process inevitably includes a degree of subjectivity. Noting this, Durrieu (2010) proposes to follow Goto (2000), who primarily links predominance to the energy and pitch range of the lead instrument compared to the rest of the ensemble.

Whilst this definition is clearly somewhat restrictive, and by no means encompasses everything that one might consider as melody, it is also very clear, and allows objective evaluation. The only thing we need to do a priori is define which instrument in the musical mixture is the lead instrument. This can be done with relatively little controversy by focusing on music material that has a clear lead instrument, such as pop and rock music with a lead singer, the "head" section of vocal and instrumental jazz, Indian classical music or opera arias from classical corpora. For this reason, we believe it is the most appropriate definition to use also in this thesis. It is important to be aware that the restrictiveness of the definition will

also limit the type of musical material our algorithms will be able to handle. Nonetheless, even this more restrictive definition still encompasses an immense corpus of music, and perhaps more importantly, it is absolutely necessary if we want to develop and evaluate a working system without getting stuck at the definition stage. The only considerable difference in our definition compared to that proposed by Durrieu (2010) is the aspect of dominance, which, as shall be seen in later chapters, we model using a set of pitch contour characteristics which include, but are not limited to, loudness and pitch range.

Representation

Now that we have a clear definition of melody itself, we need to define precisely the representation of the melody we are interested in extracting. Earlier we described the desired output as a sequence of values representing the pitch of the melody at every given moment in time. But how is this pitch sequence represented exactly, and why? In his now seminal work, Goto (2004) proposes that an appropriate description should be:

- An intuitive description that can be easily obtained by untrained listeners.
- A basic description which trained musicians can use as a basis for higher-level music understanding.
- A useful description that facilitates the development of various practical applications.

Goto also notes that whilst an easy solution would be to adopt the terminology of existing discrete symbol systems (e.g. Western score notation), such systems fail to express non-symbolic properties such as the expressiveness of a musical performance (e.g. vibrato). For this reason, he argues that a mid-level representation is needed, more abstract than the raw waveform but with richer detail compared to symbolic notation. Based on the above criteria, Goto proposes to represent the melody as a sequence of fundamental frequency (henceforth f_0) values. Each value corresponds to the pitch of the melody at a specific moment in time, and the values are sampled at a high rate, in this way ensuring we capture even the most subtle changes in the pitch of the melody over time. The difference between the waveform representation, f_0 sequence representation and symbolic notation representations is illustrated in Figure 1.2 where we plot the same melody

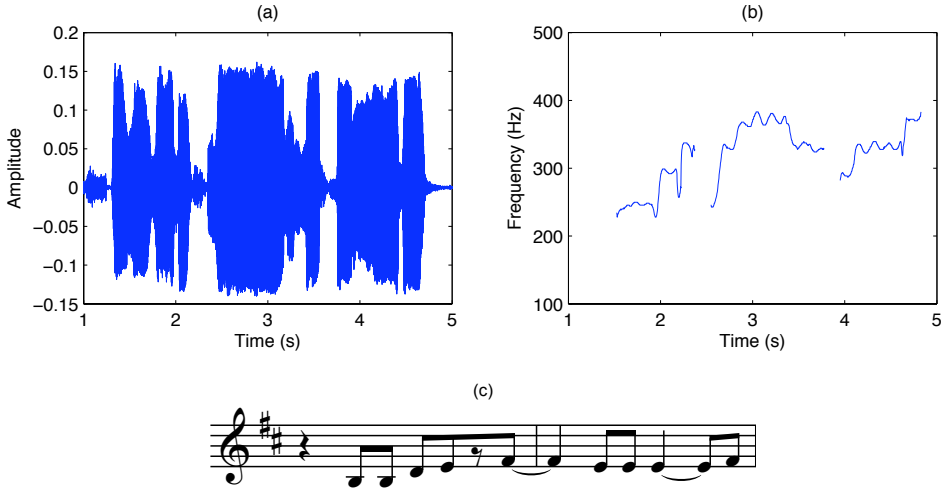

Figure 1.2: An illustration of different melody representations with an increasing degree of abstraction: (a) raw waveform, (b) f_0 sequence and (c) symbolic notation.

(a segment of "Ain't no mountain high enough" by Marvin Gaye) using the three different representations (plots (a), (b) and (c) respectively). We see that the mid-level representation in Figure 1.2 (b) satisfies the first two requirements specified by Goto: the representation can be intuitively understood by untrained listeners as "pitch height over time", and it resembles the pitch curve they would produce when singing the melody themselves. It is also useful for trained musicians who could use it as a basis for deriving the transcription in Figure 1.2 (c). Unlike the transcription however, the mid-level representation also captures the expressiveness of the specific performance, and effects such as glissando, overshoot and vibrato are clearly visible in the plot. In Chapter 6 we will demonstrate how this mid-level representation also satisfies the third requirement, that of facilitating the development of practical applications.

1.2.2 Pitch and fundamental frequency

In most of the definitions of melody presented above, and more specifically in the musicological definition proposed by Ringer (2013) and the pragmatic definition proposed by Paiva et al. (2006) adopted here, the melody is defined as a sequence of *pitched* sounds. It is thus clear that for our definition of melody to be complete we must define pitch as well.

Baines & Temperley (2013) define pitch as *"a basic dimension of musical sounds, in which they are heard to be high or low"*. They further explain that *"the subjective sense of pitch is closely correlated with frequency (number of sound vibrations per second)"*. A similar definition is provided by Klapuri & Davy (2006): *"pitch is a perceptual attribute which allows the ordering of sounds on a frequency-related scale extending from high to low"*. Hartmann (1996) proposes an operational definition: *"a sound has a certain pitch if it can be reliably matched by adjusting the frequency of a sine wave of arbitrary amplitude"*. It thus becomes clear that pitch, whilst related to frequency, is a perceptual phenomenon resulting from auditory processing in the human brain (de Cheveigné, 2005). How then can we relate pitch to a scientific quantity that we can estimate using a computer? The physical term which is most closely related to pitch is the *fundamental frequency* (f_0). For periodic (or nearly periodic) sounds, the f_0 is defined as the inverse of the period of the waveform (Klapuri & Davy, 2006). Unlike pitch, which is the product of processes of auditory perception, the f_0 is a physical quantity that can be measured and quantified. Whilst the perceived pitch of a tone may differ in some cases from the perceived pitch of a sine wave whose frequency is set to the measured f_0 of the tone (Hartmann, 1996), for periodic signals it is often the case that the two correspond very well (de Cheveigné, 2005; Klapuri & Davy, 2006). In practice, the f_0 is the most correlated measurable physical counterpart to pitch. Going back to the mid-level representation discussed in the previous section, it now becomes clear why Goto proposes to represent the pitch of the melody as a sequence of f_0 values. Indeed, in the music processing literature "pitch" and "f_0" are often used almost synonymously. In this thesis, for the sake of consistency with the terminology established in the MIR literature, we will often use the terms "pitch" and "f_0" interchangeably (e.g. one of the standard measures used for evaluating melody extraction algorithms is called "Raw Pitch Accuracy", even though what is actually measured is the f_0 estimation accuracy). Additionally, we will consider f_0 sequences as accurate representations of pitch sequences. For further reading on the relationship between pitch and frequency, and the possible auditory processes that underlie the perception of pitch, the reader is referred to de Cheveigné (2005); Hartmann (1996); Rasch & Plomp (1999); Terhardt (1974).

1.2.3 Polyphony

Given our definitions of melody, pitch and f_0, we now have a clear notion of the pitch sequence we are interested in estimating. What remains to be

defined is the type of musical mixtures from which we intend to extract this sequence. In Western musicology, *musical texture* refers to the way in which melodic, rhythmic and harmonic materials are combined in a composition. Often, texture is considered in terms of the number of parts in a composition and their relationship with one another. Traditionally, texture is divided into three[1] categories (Copeland, 1957):

- Monophonic: music consisting of a single, unaccompanied melodic line.
- Homophonic: music consisting of a principle melodic line and chordal accompaniment.
- Polyphonic: music consisting of multiple (two or more) relatively independent melodic lines.

However, in the MIR research literature this classification is commonly narrowed down to two classes, defined as follows:

- Monophonic: music consisting of a single, unaccompanied melodic line (this definition remains unchanged).
- Polyphonic: any musical mixture in which two or more instruments may sound simultaneously.

That is, a single distinction is made between musical content containing a single line (and hence a single sound source in most cases) and music containing several sound sources, where all non-monophonic textures are grouped under the single title of polyphonic music. This distinction stems from the significantly different signal processing techniques required to analyse each of these two classes, where the polyphonic case is often more complex compared to its monophonic counterpart (more on this in Section 1.3). Following this reasoning and for the sake of consistency with the literature, in this thesis we use the term polyphonic to refer to all non-monophonic music. In practice, the musical material addressed in this dissertation will be either homophonic or polyphonic.

[1]Other slightly less commonly used categories include *biphony* where one line sustains a constant drone note and a second line creates a more elaborate melody above it, and *hetrophony* where two or more voices simultaneously perform variations of the same melody.

1.2.4 Melody extraction

Now that we have defined both the pitch sequence we are interested in estimating and the musical material from which we wish to estimate it, we can finally provide an accurate definition of the melody extraction task as it will be treated in this thesis:

Melody extraction: *fundamental frequency estimation of a single predominant pitched source from polyphonic music signals with a lead voice or instrument.*

Note that this definition also implicitly includes the requirement to estimate the time intervals in which the melody is present in the recording.

1.3 The challenge

For a human listener, the task of melody extraction might seem almost trivial – many of us can sing the melodies of our favourite songs even without any musical training. Those with musical training can even transcribe a melody into musical notation. However, when we try to automate this task, it turns out to be highly challenging. The complexity of the task is twofold: first, the signal representation of polyphonic music is composed of the superposition of the sound waves produced by all instruments in the recording, and much of the time these instruments play simultaneously. When considering the spectral content of the signal, the frequency components of different sources superimpose making it very hard to attribute specific energy levels in specific frequency bands to the notes of individual instruments. This is further complicated by mixing and mastering techniques which can add reverberation (thus blurring note onsets and offsets and increasing the overlap of sound sources) or apply dynamic range compression (thus reducing the difference between soft and loud sources, increasing interference). Second, even if we manage to obtain a pitch-based representation of the audio signal, we still need to determine which pitches belong to the predominant melody and which are merely accompaniment. This in turn entails three main challenges – determining when the melody is present and when it is not (referred to as *voicing detection*), ensuring the estimated pitches are in the correct octave (i.e. avoiding *octave errors*), and selecting the correct melody pitch when there is more than one note sounding simultaneously. In Figure 1.3 we illustrate the difficulty of the

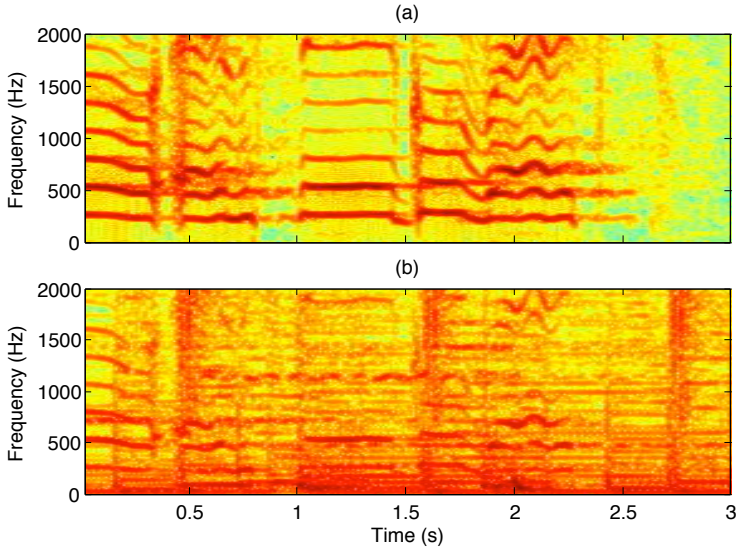

Figure 1.3: Two spectrograms that illustrate how the harmonic content of the melody signal in plot (a) is hard to estimate once it is superimposed with that of the accompaniment, producing the final polyphonic mixture in plot (b).

task by plotting the spectrogram of a melody source on its own and then the spectrogram of the full polyphonic mixture containing both the melody and accompaniment instruments.

1.4 Scientific context

The challenge of melody extraction lies at the intersection of several areas of research (Figure 1.4). In the sections below we provide a brief introduction to each of these areas and their relationship to melody extraction.

1.4.1 Music information retrieval

Music was the first mass-market industry to be completely restructured by digital technology, starting with the compact disc, and leading to today's situation where typical consumers may have access to thousands of tracks stored locally on their smartphone or music player, and millions of tracks instantly available through cloud-based music services. Given these vast numbers of songs, we now require novel ways of describing, indexing, searching and interacting with music. In addition to industry bodies engaged in

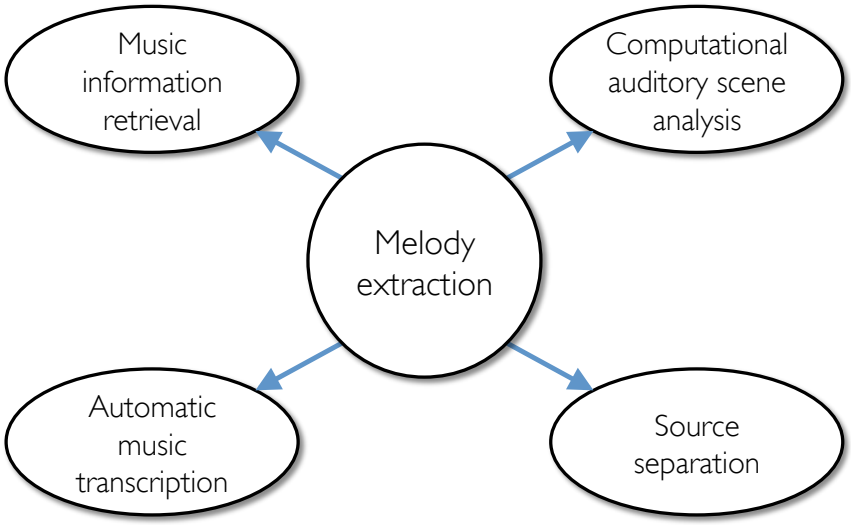

Figure 1.4: Research areas related to melody extraction: music information retrieval, computational auditory scene analysis, automatic music transcription and source separation.

recording, aggregating and disseminating music and their target audience of end users, the abundance of digital music also offers new opportunities for music professionals including performers, teachers and musicologists.

This new reality has led to the rise of an increasingly-active interdisciplinary research field, Music Information Retrieval (MIR)[2], a convergence point for experts from a varied range of long-established disciplines such as musicology, signal processing, psychoacoustics, information science, computer science and statistics (Casey et al., 2008a; Gouyon et al., 2008; Orio, 2006). The goal of MIR research is to develop computational methods for addressing the wealth of possible scenarios for interacting with music in the digital era, including the understanding, description, organisation and retrieval of musical content, both in symbolic form (e.g. digitised sheet music or MIDI

[2]The earliest article known to use the acronym MIR (for "musical information retrieval") was in fact published almost fifty years ago (Kassler, 1966), and early efforts to build music databases for music content retrieval can be traced back to the 1980's (Pope, 1986). Still, the majority of MIR research is concentrated in the last two decades, and MIR started to consolidate as a research field with the turn of the millennium and the appearance of conferences such as the International Symposium on Music Information Retrieval (ISMIR), nowadays the International Society for Music Information Retrieval Conference.

files) and in the form of recorded audio. The latter case, in which information is obtained by directly analysing the audio signal of music recordings, is often referred to as content-based MIR (Gouyon et al., 2008). Content-based MIR involves both audio content description, i.e. the automatic extraction of meaningful information from the audio signal, and audio content exploitation, which is the utilisation of the obtained description for various applications such as content-based search and retrieval (Cano, 2007; Serrà, 2011), recommendation (Bogdanov et al., 2013) and transformation (Verfaille et al., 2006). Audio content description ranges from the extraction of low-level temporal, spectral and cepstral features (Peeters, 2004), through tonal and rhythmic information (Gómez, 2006a; Gouyon, 2008) and automatic transcription (Bello, 2003; Klapuri & Davy, 2006), all the way to the extraction of higher-level concepts such as genre (Scaringella et al., 2006) or mood (Laurier, 2011).

As the number of researchers working on MIR grew, so did the need for proper means of evaluating and comparing the performance of different algorithms. In 2004, the first Audio Description Contest (ADC) was hosted by the Music Technology Group at Universitat Pompeu Fabra in Barcelona, Spain. This initiative was followed by the Music Information Retrieval Evaluation eXchange (MIREX) campaign for evaluating MIR technologies (Downie, 2008), and has resulted in a series of well-organised and well-supported international evaluations in the years since.

Melody extraction is a specific goal of content-based MIR. As shall be seen in Chapter 2, this topic has received substantial attention from the audio signal processing and MIR research communities. It is one of the tasks evaluated annually in MIREX, and the wide range of applications that can be built on top of it, both for research and commercial purposes, positions it as a central problem in MIR research.

1.4.2 Computational auditory scene analysis

When we listen to the sounds of our environment, we are able to distinguish the different sounds produced by different processes, even though they may all be sounding simultaneously. For example, during a walk in the park we can distinguish the sound of a bird from that of a car passing by from that of a child playing. Indeed, we humans seem to be able to perform this "segregation" even though the soundwave reaching our ears is the single superposition of the waveforms produced by all the different sound sources around us. This perceptual process, of organising and segmenting sounds

in time and frequency and assigning them to real-world objects (or *auditory streams*), is often referred to as "Auditory Scene Analysis" (ASA), a term coined by psychologist Albert Bregman (Bregman, 1990). Bregman describes parsing the auditory scene as a grouping problem in (at least) two dimensions: across time (sequential integration) and across the spectrum (simultaneous integration). Within these two categories different grouping principles are defined. A similar thing occurs when we listen to music, where we are often able to distinguish and follow a single instrument within a polyphonic mixture even though all instrument soundwaves are superimposed (cf. Section 1.3). Bregman refers to this as "Music Scene Analysis".

If we try to imitate the human ability to make sense of the auditory scene using computers, we enter the field of "Computational Auditory Scene Analysis" (CASA; Ellis, 1996; Wang & Brown, 2006). Here the goal is to build systems that use the principles of auditory scene analysis in order to segregate and identify the objects contributing to a sound recording, usually either environmental or musical. CASA has an important role in MIR, and many ASA-based and ASA-inspired algorithms have been proposed (cf. Goto, 2006, for a review of some algorithms). If we consider melody extraction in an ASA context, the melody (as we have defined it) is an auditory stream, and our goal is to segregate the f_0 of this stream from the rest of the mixture. Huron (2001) showed that Western voice-leading rules can be derived from the perceptual principles established by ASA. It thus stands to reason that we should be able to exploit these principles for designing a melody extraction algorithm. The melody extraction method presented in this thesis is not strictly ASA-based, nor is it our goal to design a CASA algorithm for melody extraction. Nonetheless, our work does draw from ASA, as we exploit some of the grouping principles defined by ASA for the design of our algorithm. Since we do not strictly implement each principle as defined by Bregman, we shall refrain from defining our method as ASA-based or as a CASA algorithm. But as shall be seen, certain parts of the algorithm are, most definitely, inspired by the ASA grouping principles of simultaneous and sequential integration.

1.4.3 Automatic music transcription

Another important category melody extraction falls into is music transcription. Music transcription refers to the analysis of an acoustic music signal for producing a parametric representation of the signal (Bello, 2003; Ryynänen & Klapuri, 2008a). Music transcription is not restricted to the melody, and

also addresses the transcription of other lines (e.g. the bass line) as well as the harmony (e.g. in the form of chords). Apart from being an attractive end goal by itself, music transcription has many application areas including MIR, music processing (transcribe, transform and re-synthesise), human-computer interaction (e.g. score typesetting, music oriented games), music equipment (e.g. music-synchronous light effects, interactive accompaniment systems) and musicological analysis (Klapuri & Davy, 2006).

One of the earliest music transcription problems to be addressed was that of monophonic pitch (f_0) estimation, which has a long research tradition. Early approaches were adopted primarily from the speech processing literature (Hess, 1983). Since then, various approaches specifically tailored for f_0 estimation in monophonic music signals have been proposed (see Gómez et al., 2003 for a review). More recently, algorithms have also been proposed for estimating the f_0 of multiple concurrent instruments in polyphonic recordings (*multi-pitch estimation*; Klapuri, 2003). For a detailed review the reader is referred to Klapuri (2004). Melody extraction differs from both monophonic and multi-pitch estimation in two important ways. Unlike monophonic pitch estimation, in melody extraction we are dealing with polyphonic material and the challenges it entails (cf. Section 1.3). Unlike multi-pitch estimation, melody extraction requires the identification of the specific voice that carries the melody within the polyphony, but does not involve estimating the remaining pitches.

Nowadays, music transcription usually refers to the process of abstracting all the way to a symbolic representation which includes the pitch, onset time, duration, and source of each sound in the mixture (Klapuri & Davy, 2006). This highlights an important distinction between melody extraction and music transcription: the goal of melody extraction is not to obtain an abstract symbolic representation, but rather a more fine-grained mid-level representation (cf. Section 1.2.1). Melody extraction thus constitutes the first step, or "front-end", in a complete melody transcription system. This step would then be followed by a pitch segmentation and quantisation process in order to obtain a score-like representation of the melody. In Chapter 6 we will demonstrate how the melody extraction algorithm proposed in this dissertation can be successfully combined with such a note segmentation algorithm to produce a complete symbolic transcription of the melody. For further reading on music transcription the reader is referred to Bello (2003); Klapuri (2004); Klapuri & Davy (2006); Ryynänen (2008) and references therein.

1.4.4 Source separation

Source separation, sometimes also referred to as source segregation or sound separation, is the task of recovering the waveforms of the individual sound sources that were mixed together in a sound recording. A classic example is the "cocktail party problem", where we have several people talking simultaneously in a room (for example in a cocktail party[3]) and we are interested in capturing what a specific person is saying. Given a recording of the conversation, the goal of source separation would be to recover the signal corresponding to the waveform produced by the person we are interested in, or in more ambitious cases to recover the individual signals of all speakers. In general, the task is harder when there are fewer channels in the observed mixture compared to the number of sources we wish to separate (in this case the problem is *underdetermined*). When no information is available about the sources or the mixing conditions, the problem is referred to as blind source separation.

Music source separation is the application of source separation techniques to music audio signals. Given a music mixture (usually mono or stereo), the goal is to separate the signal into several coherent components. This could be the separation of the lead instrument from the accompaniment, separation into harmonic and percussive instruments, or indeed the separation of all individual instruments in the mixture. Source separation algorithms are commonly based on decomposition methods such as independent component analysis (Vincent & Deville, 2010) or matrix factorisation techniques such as non-negative matrix factorisation (Smaragdis & Brown, 2003). The reader is referred to the work by Durrieu (2010); Plumbley et al. (2002); Vincent (2004); Vincent et al. (2012); Virtanen (2006) for a comprehensive review of audio source separation methods.

Melody extraction can be posed as a source separation problem. If we are able to separate the signal of the melody source from the rest of the mixture, estimating the f_0 sequence of the melody from this signal can be less complex compared to estimating it from the original mixture. With the advances in audio source separation, several melody extraction algorithms based on source separation have been proposed (cf. Chapter 2). But using source separation for melody extraction has its limitations. For example, in some methods the separation is performed by defining a model of the sound production mechanism of the melody source (Durrieu, 2010). Other

[3]Nowadays this term is perhaps somewhat anachronistic, and something in the lines of "the bar problem" would probably be more meaningful.

methods make assumptions about the musical accompaniment (e.g. that it has a repeating pattern; Liutkus et al., 2012; Rafii & Pardo, 2013). In both cases, such assumptions limit the musical material that the algorithm can process. Additionally, source separation methods are often (though not always) computationally expensive. In the following section we describe the approach for melody extraction used in this thesis, namely, that of *understanding without separation*.

1.4.5 Understanding without separation

Unlike source separation, the goal of melody extraction is not to recover the signal of the melody source, but rather to estimate the f_0 sequence corresponding to the pitch of the melody. This means that whilst source separation is a possible approach for melody extraction, it is not a prerequisite. Indeed, research on auditory perception suggests that our ability to segregate and understand sounds does not rely on the complete separation of sounds by some internal mechanism (Bregman, 1995). The problem could in fact be turned on its head: a melody extraction algorithm could be used to obtain the f_0 sequence of the melody in order to guide a source separation algorithm in separating the signal of the melody.

In this dissertation we adopt the *auditory scene description* philosophy for melody extraction proposed by Goto (2004), which states that source separation is not necessary for melody extraction. That is, we assume that the melody f_0 sequence can be estimated without having to first separate the waveform of the melody source from the mix. This approach, of *understanding without separation* (Scheirer, 2000), is not limited to melody extraction and has been argued for in various studies on machine listening and music understanding (Ellis, 1996; Herrera et al., 2000; Scheirer, 1996, 1999). The distinction between sound separation and sound understanding is discussed at length by Scheirer (2000) in his doctoral thesis, who notes that "it is unlikely that human listeners maintain multiple independent time-domain signals as an intermediate representation of complex signals". He also notes the practical advantage of adopting a non-separation approach, since "less time must be spent on achieving high-quality synthesis of output sounds". Furthermore, Scheirer (2000) explains that

> "the advantage of the understanding-without-separation approach is most apparent in the case when one constituent signal destroys information in another through masking or cancellation.

> In a sound-separation system, it is very difficult to deal with this situation properly, since the obliterated sound must be invented wholesale from models or a priori assumptions. In a separationless approach, the required action is one of making feature judgements from partial evidence, a problem that is treated frequently in the pattern recognition and artificial intelligence literature."

Models for machine listening (and more specifically melody extraction) can be either separation-based or separationless. In the former, source separation is first applied to the mixture to discover the constituent sounds that comprise it, and then the separated sounds are analysed. Emphasis is put on being able to reconstruct the exact constituent sounds that were combined to create the sound mixture. In the latter, the sound mixture is analysed to discover the *features* of the sounds that comprise it, which can then be used to infer higher-level information about the mixture. As Scheirer notes, "the goal [of separationless sound understanding] is to describe the sound scene at a sufficient level of detail for solving problems". In the case of melody extraction, the feature we are interested in is the f_0 sequence of the melody. The difference between the sound separation and sound understanding approaches for melody extraction is illustrated in Figure 1.5, based on (Scheirer, 2000).

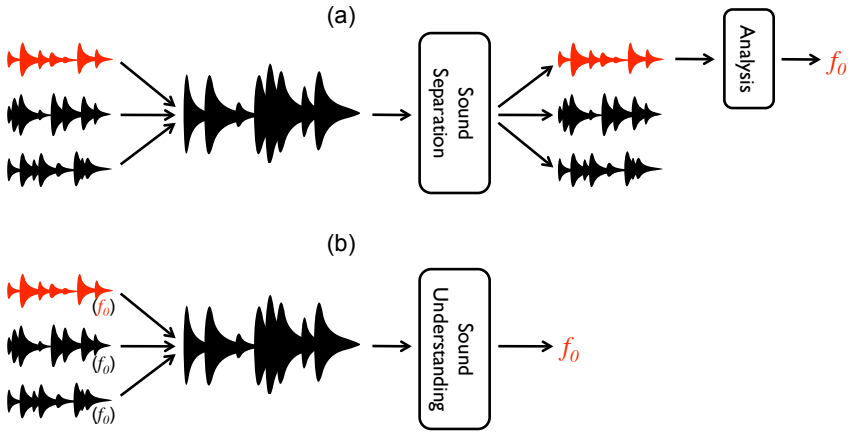

Figure 1.5: Separation versus understanding: (a) a sound separation approach to melody extraction, (b) a sound understanding approach to melody extraction. The melody signal is highlighted in red.

By now it should be clear that the melody extraction algorithm that will be presented in this thesis follows the approach illustrated in plot (b). Indeed, the plot can be considered as a more detailed version of the illustration in Figure 1.1 presented at the very beginning of this chapter.

1.5 Summary of contributions

This thesis contributes to the field of content-based music information retrieval, with specific contributions in the areas of automatic music analysis, retrieval and organisation. The main contributions of the thesis can be summarised as follows:

Scientific contributions

- A discussion of the musical concept of "melody", both from a historical and an engineering perspective, leading to a clear problem statement and definition of what we refer to by melody extraction.
- A detailed review of the current state of the art in automatic melody extraction, including a discussion of the main strategies for melody extraction, a comparison of the different techniques applied in the main processing blocks of different algorithms, a comparison of extraction accuracy and a study of the evolution of performance in the past decade. It also includes a case study which highlights the main challenges and sources of confusion for melody extraction algorithms.
- A comparative evaluation of different signal processing techniques and parameter settings for sinusoid extraction and pitch salience computation and their optimisation for the specific task of melody extraction.
- A novel method for melody extraction based on the creation and characterisation of pitch contours. An extension of the proposed approach based on the statistical modelling of melodic pitch contour feature distributions.
- A study of the current challenges in melody extraction evaluation, including a statistical analysis of the most commonly used evaluation datasets based on generalisability theory. To the best of the author's knowledge this is the first study to examine evaluation methodology for melody extraction.

- A large study of the different application areas for the melody extraction method presented in this thesis, and the presentation of novel methods in several domains, namely: version identification, query-by-humming, genre classification, tonic identification in Indian classical music and melodic transcription with a focus on accompanied flamenco singing.

Technical contributions

- MTG-QBH, a dataset of sung melodies which can be used (inter alia) to develop and evaluate query-by-humming systems. The dataset is freely available online[4] and includes detailed metadata files.

- MELODIA - Melody Extraction vamp plug-in: an implementation of the melody extraction algorithm proposed in this thesis, in the form of a vamp[5] plug-in. The plug-in is available for all three major operating systems (Windows, Linux and OSX) and can be freely downloaded for non-commercial purposes[6]. The plug-in can be used for detailed analysis and visualisation of the predominant melody of an audio recording using Sonic Visualiser[7], including the visualisation of the intermediate steps of the algorithm. It can also be used for batch processing of large music collections using Sonic Annotator[8]. Since its announcement in October 2012, MELODIA has been downloaded over 4,000 times (4,400 as of June 15$^{\text{th}}$ 2013) by researchers, educators, artists and hobbyists from all over the world (cf. Appendix A).

The melody extraction algorithm proposed in this thesis was evaluated in the Audio Melody Extraction task of the MIREX international evaluation campaign, and was shown to obtain the highest mean overall accuracy to date (June 2013) for the current datasets used for melody extraction evaluation. As far as the author is aware, at the time of writing this thesis the algorithm has been (or is being) used in at least 7 Master's theses, 5 ongoing doctoral theses (not including this one) and 2 European funded research projects (cf. Appendix A). The outcomes of the research carried out in this thesis have been published in a number of peer-reviewed journals and

[4] http://mtg.upf.edu/download/datasets/mtg-qbh
[5] http://vamp-plugins.org/
[6] http://mtg.upf.edu/technologies/melodia
[7] http://www.sonicvisualiser.org/
[8] http://www.omras2.org/sonicannotator

international conference proceedings. A full list of the author's publications is provided in Appendix B.

1.6 Goals and thesis outline

The work in this thesis is driven by two hypotheses. First, that melody extraction can be performed successfully using an understanding without separation approach. More specifically, that the characteristics of the pitch contour of the melody can be exploited for the identification of the melodic line within the polyphony, and consequently for the extraction of its f_0 sequence. Thus, the first and primary goal of this thesis is to develop a method for automatic melody extraction that is based on the definition and exploitation of a set of characteristics which differentiate the melodic pitch contour from the remaining tonal sources in a polyphonic music recording. As noted earlier, it is not our intention to design an algorithm that strictly imitates the human auditory system, and rather our objective is to produce an algorithm that can extract the melody with as high an accuracy as possible. The algorithm should be able to extract the melody from polyphonic music by directly analysing the audio signal, without using any external information about the specific piece being analysed such as its musical genre, instrumentation, or the nature of the source playing the melody. Whilst it is not a strict requirement, we will also aim to keep the algorithm as conceptually simple as possible, avoiding any black-boxes in the design, with the intention that every processing step should be easily interpretable and musically meaningful.

The second hypothesis driving the work in this dissertation is that the mid-level representation of the melody obtained from the polyphonic music signal using melody extraction can be highly useful for developing applications for music description, retrieval and organisation. Consequently, the second goal of this thesis is to develop a set of prototype applications that exploit the output of the melody extraction method proposed here. In this way, we aim to exemplify some of the many use-cases of melody extraction, and demonstrate that the method proposed in this dissertation, which is based on exploiting features of the melodic contour for its identification and extraction, can be used successfully in real-world applications.

The structure of the remainder of this thesis is as follows. We start by providing the scientific background for the work presented in this dissertation in Chapter 2. We explain how to go from monophonic pitch tracking

to melody extraction, review a large set of melody extraction algorithms and compare them in terms of algorithmic design and performance. We also provide a case study to better understand the types of errors made by melody extraction algorithms. The following two chapters correspond to the first two blocks (Chapter 3) and the final two blocks (Chapter 4) of the melody extraction method proposed in this dissertation. To help follow the thesis, a block diagram of the complete algorithm is provided here (Figure 1.6). In Chapter 3 we describe the processing steps applied in order to extract sinusoidal components from the audio signal, and how these are then used to compute a *salience function* – a representation of pitch salience over time. This includes a quantitative comparison of different signal processing techniques and the optimisation of the parameters of the proposed salience function. In Chapter 4 we describe how the proposed salience function is used as part of a novel melody extraction algorithm based on the creation and characterisation of pitch contours. We also describe a modified version of the algorithm based on a statistical model of the feature distributions of melodic pitch contours. The chapter includes a thorough evaluation of the algorithm, and discusses the results obtained both in the MIREX international evaluation campaign and in our own experiments.

In Chapter 5 we present a critical study of the current evaluation methodology used for evaluating and comparing melody extraction algorithms, focusing on the MIREX campaign. We consider three aspects of evaluation: the annotation process, the duration of the audio excerpts and the size and contents of the music collections used for evaluation. To show that the output of the melody extraction algorithm proposed in this thesis can be used successfully in real-world applications, in Chapter 6 we present a set of systems that we have developed on top of the proposed algorithm. Specifically, we present and evaluate systems in four application areas: music similarity and retrieval, genre classification, tonic identification in Indian classical music and automatic melodic transcription with a focus on flamenco music. Finally, in Chapter 7 we provide a summary of the work presented in this dissertation, and discuss some future perspectives.

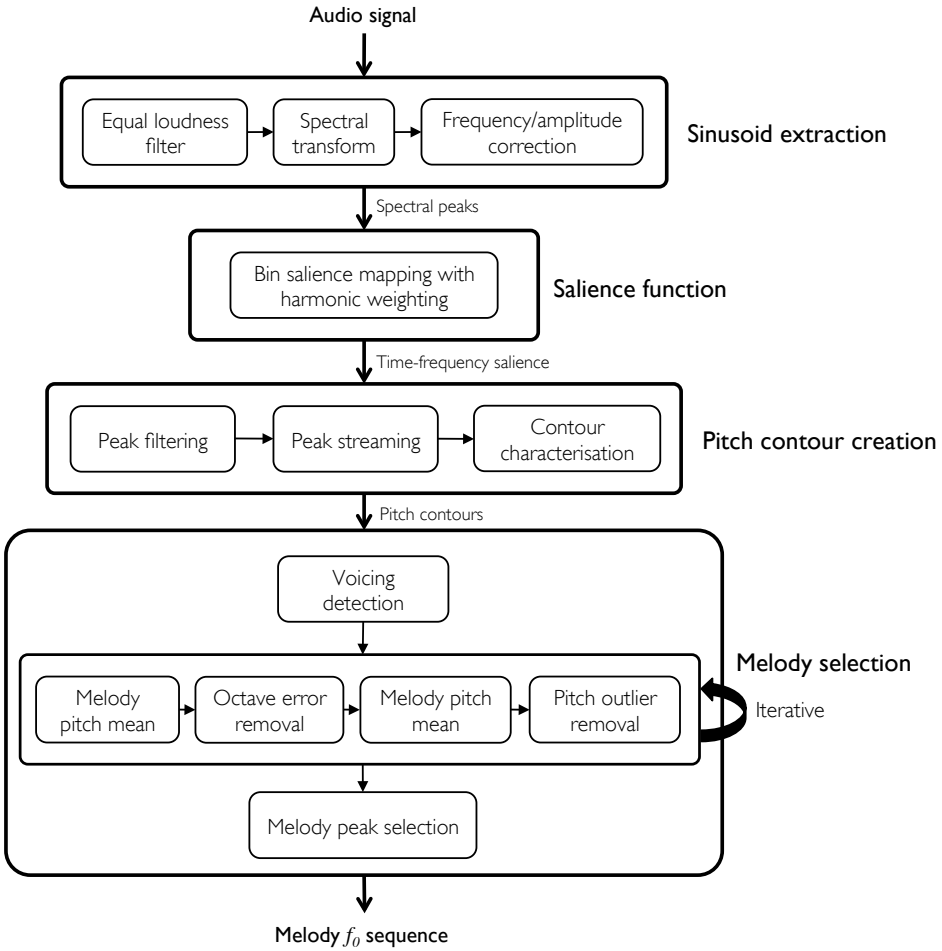

Figure 1.6: Block diagram of the complete melody extraction algorithm proposed in this dissertation, consisting of four main blocks: sinusoid extraction, salience function, contour creation and melody selection. The first two blocks are presented in Chapter 3 and the final two blocks are presented in Chapter 4.

Chapter 2

Scientific Background

2.1 Introduction

In this chapter we provide the relevant scientific background for the work presented in the following chapters of this dissertation. We start by discussing algorithms for monophonic pitch tracking (Section 2.2), and consider their links and differences compared to melody extraction algorithms. In the subsequent sections of this chapter, which are largely based on Salamon et al. (2013a), we provide a detailed overview of the current state of the art in melody extraction. This includes a review of algorithmic approaches (Section 2.3), evaluation methodology (Section 2.4) and performance (Section 2.5). Then, we provide a case study where we examine the actual output of a melody extraction algorithm, in this way highlighting some of the most common errors made by melody extraction algorithms and identifying their possible causes (Section 2.6). The chapter is concluded with a summary of the main concepts and issues presented (Section 2.7).

As explained in Section 1.6, in Chapter 6 we present a set of applications based on the melody extraction algorithm proposed in this thesis. Since each of these applications comes from a (related yet different) field of study within MIR, it makes sense to provide some background information on each of these fields too. For the sake of readability, we have opted to place this background information directly in Chapter 6, in the introduction section of each application.

2.2 From pitch estimation to melody extraction

Melody extraction is strongly linked to pitch (f_0) estimation, which has a long research tradition. As mentioned in the previous chapter, early approaches for pitch estimation in music dealt with the estimation of the f_0 of monophonic music recordings, and were adopted from the speech processing literature (Hess, 1983). Since then, various approaches specifically tailored for f_0 estimation in monophonic music signals, also referred to as *pitch tracking*, have been proposed. In the following sections we provide a brief overview of monophonic pitch trackers and their link to melody extraction. It is beyond the scope of this thesis to provide a comprehensive review of the large number of monophonic pitch trackers that have been proposed to date, and the reader is referred to de Cheveigné & Kawahara (2002); Gómez et al. (2003); Klapuri (2000) and references therein for further coverage of this topic.

2.2.1 Monophonic pitch trackers

Broadly speaking, monophonic pitch trackers can be classified into two groups based on their processing domain: algorithms operate either in the *time domain*, i.e. the pitch is estimated directly from the audio signal, or in the *frequency domain*, i.e. the pitch is estimated from a spectral representation of the audio signal obtained using a transform such as the short-time Fourier transform (STFT; Smith III, 2013)[1].

Time-domain algorithms

Time-domain algorithms compute the f_0 of a sound by attempting to estimate the periodicity of the audio signal from its waveform (i.e. time-domain) representation. One of the earliest and simplest techniques proposed is to compute the *zero crossing rate* of the signal (i.e. the number of time the signal crosses the 0-level), but the approach's high sensitivity to noise means it is rarely used for f_0 estimation in practice. Some early approaches are based on models of the auditory system and attempt to compute the "perceived pitch" (Terhardt, 1979; Terhardt et al., 1982). Others are based on parallel processing where the signal is processed into several impulse trains from which several periodicity estimates are computed and finally re-combined to produce a final estimate (Gold & Rabiner, 1969; Rabiner & Schafer,

[1] Some algorithms, such as the autocorrelation-based approach, can be expressed in both domains.

1978). A popular set of approaches is based on computing the autocorrelation of the audio signal (Medan et al., 1991; Talkin, 1995), which can also be computed in the frequency domain (Klapuri, 2000). The maximum of the Autocorrelation Function (ACF) corresponds to the f_0 for periodic signals. Still, whilst they are fairly robust to noise, ACF-based methods are sensitive to formants and spectral peculiarities (Klapuri, 2000), and are likely to produce frequency halving errors. In an attempt to reduce the error rate of ACF-based methods, de Cheveigné & Kawahara (2002) propose a series of modifications to the ACF approach, most notably replacing the use of autocorrelation with a cumulative mean-normalised difference function. This method, called the YIN algorithm, is still amongst the most popular time-domain techniques for monophonic pitch tracking in use today.

Frequency-domain algorithms

Frequency-domain algorithms use a spectral representation of the signal to estimate its f_0. One of the first algorithms implemented on a computer was based on *cepstrum* analysis (Noll, 1967). The (power) cepstrum is defined as the inverse Fourier transform of the logarithm of the power spectrum of the signal. For a periodic signal, a strong peak appears at the location corresponding to the lag of the period of the signal, from which the f_0 can be computed. A different way to exploit the spectrum is to compute its autocorrelation. Complex (i.e. non-sinusoidal) periodic signals have a periodic magnitude spectrum. Autocorrelation can be used to find the period of the magnitude spectrum, which corresponds to the f_0 of the signal (Lahat et al., 1987). Another set of frequency-domain algorithms is based on *harmonic matching*: the peaks of the magnitude spectrum are matched against the expected locations of the harmonics of a candidate f_0 (Maher & Beauchamp, 1994; Piszczalski & Galler, 1979). The degree of the match is evaluated with a fitness measure, according to which the best f_0 candidate is selected. Finally, Klapuri (2000) proposes a bandwise processing algorithm, in which the spectrum is divided into bands and the f_0 is estimated in each band separately by computing the likelihood of every pitch candidate as the summation of its harmonic amplitudes. Finally the per-band estimates are recombined to yield a global estimate. This provides greater robustness against inharmonicity (since for narrow enough bands we can assume that the spectral intervals between partials are constant), as well as robustness in the case of badly corrupted signals where only some segment of the spectrum is usable.

2.2.2 From monophonic to polyphonic signal processing

To understand the link (and the differences) between pitch tracking and melody extraction, it is instructive to consider melody extraction algorithms as elaborations of monophonic pitch trackers. A monophonic pitch tracker, whether it is a time-domain or frequency-domain approach, can generally be described as a method that takes an audio signal $y(t)$ and calculates a function $S_y(f_\tau, \tau)$ evaluated across a range of candidate pitch frequencies f that indicates the relative score or likelihood of the pitch candidates at each time frame τ. The local estimates of period are then typically subject to sequential constraints, for instance via dynamic programming. Thus, the estimated sequence of pitch values $\hat{\mathbf{f}}$, represented as a vector with one value for each time frame, is derived as:

$$\hat{\mathbf{f}}_{\text{monophonic}} = \arg\max_{\mathbf{f}} \sum_{\tau} S_y(f_\tau, \tau) + C(\mathbf{f}) \tag{2.1}$$

where f_τ is the τ^{th} element of $\mathbf{f}$, and $C(\mathbf{f})$ accounts for the temporal constraints. As noted earlier, a common choice for $S_y(f_\tau, \tau)$ is an autocorrelation function such as:

$$S_y(f, \tau) = r_{yy}(\frac{1}{f}; \tau) = \frac{1}{W} \int_{\tau - W/2}^{\tau + W/2} y(t) y(t + \frac{1}{f}) dt \tag{2.2}$$

where W is the length of the autocorrelation analysis window.

In melody extraction, we move from analysing a monophonic signal (which is assumed to contain a single periodicity) to a polyphonic signal. Thus, the observed signal $x(t)$ now consists of a target monophonic melody signal $y(t)$ with added accompaniment "noise" $\eta(t)$:

$$x(t) = y(t) + \eta(t). \tag{2.3}$$

There are two principle paths to extending monophonic trackers to succeed in such conditions: we could design a new pitch candidate scoring function which can robustly process polyphonic signals, so it continues to reflect the desired pitch even in the presence of other periodicities. We refer to methods based on this strategy as *salience based* melody extraction approaches:

$$\hat{\mathbf{f}}_{\text{salience}} = \arg\max_{\mathbf{f}} \sum_{\tau} S'_x(f_\tau, \tau) + C'(\mathbf{f}) \tag{2.4}$$

where S'_x is the new pitch *salience function* – a time-frequency representation of pitch salience, calculated over the mixed signal x. There are many

different approaches for calculating the salience function (cf. Section 2.3.1). For instance, some functions compute the salience of a candidate frequency f as the weighted sum of its harmonics:

$$S'_x(f_\tau, \tau) = \sum_{h=1}^{N_h} g(f_\tau, h)|X(h \cdot f_\tau, \tau)| \tag{2.5}$$

where N_h is the number of harmonics in the summation, $g(f_\tau, h)$ is a harmonic weighting function (Ryynänen & Klapuri, 2008a), and $X(f, \tau)$ is the short-time Fourier transform,

$$X(f, \tau) = \int_{-W/2}^{W/2} w(t)x(\tau + t)e^{-j2\pi ft}dt \tag{2.6}$$

where $w(t)$ is a windowing function. Note that in Equation 2.4 we now use $C'(\mathbf{f})$ to represent the temporal constraints instead of $C(\mathbf{f})$, since for the polyphonic case this is a far more complex problem: even with a salience function designed for processing polyphonic audio signals, there is no guarantee that the frequency of the melody will always be found at the maximum of the function. As shall be seen in Section 2.3.1, this is addressed by considering several peaks of the salience function at each time frame (not just the maximum) in combination with *tracking* techniques such as Viterbi decoding, tracking agents or clustering.

Alternatively, we could attempt to decompose the mixed signal into separate sources, at least one of which, $\hat{y}(t)$, is dominated by the melody signal to a degree that makes it suitable for a largely unmodified pitch tracker. We refer to methods based on this strategy as *source separation based* melody extraction approaches:

$$\hat{\mathbf{f}}_{\text{separation}} = \arg\max_{\mathbf{f}} \sum_{\tau} S_{\hat{y}}(f_\tau, \tau) + C'(\mathbf{f}) \tag{2.7}$$

where $\hat{y}(t)$ is estimated using decomposition or matrix factorisation techniques (cf. Section 2.3.2). Whilst salience based and source separation based approaches are the two most common types of melody extraction algorithms found in the literature, alternative approaches have also been proposed, as shall be seen in Section 2.3.3.

2.3 Melody extraction: the state of the art

Since its initiation in 2005, over 50 melody extraction algorithms have been submitted to the Music Information Retrieval Evaluation eXchange

(MIREX; Downie, 2008). In this annual campaign, different algorithms are evaluated against the same set of music collections in order to obtain a quantitative comparison between methods and assess the accuracy of the current state of the art in melody extraction. We believe MIREX is a good point of reference for this review, given that the large majority of melody extraction algorithms that have had an impact on the research community have participated in MIREX at some point. Many of the earliest published methods for melody extraction (as it is defined in this thesis), such as the algorithms proposed by Goto (2000); Goto & Hayamizu (1999); Marolt (2004); Paiva et al. (2004), were also subsequently evaluated in MIREX (in most cases the authors submitted an updated version of the algorithm based on the same principles, e.g. Goto, 2004; Paiva et al., 2006), meaning they are also reflected in this review. For further details on early melody extraction algorithms the interested reader is referred to Gómez et al. (2006); Klapuri (2004); Klapuri & Davy (2006) and references therein.

In Table 2.1 we provide a summary of the characteristics of a selection of 16 representative algorithms out of all the submissions to MIREX since 2005. To do so, we have attempted to break down the extraction process into a series of steps which are common to most algorithms. Since some authors submitted several algorithms over the years, we have opted to include only their most recent (published) contribution, as in most cases it represents the latest version in the evolution of a single approach. If a certain step is not included in an algorithm (or otherwise not mentioned by the authors) a '-' is placed in the table. We put 'N/A' when a step is not relevant to the method (e.g. Poliner & Ellis, 2005, determine the melody directly from the power spectrum and hence a multipitch representation of the audio signal is not relevant to their approach). Finally we note that some algorithms (namely those by Durrieu et al., 2010, and Tachibana et al., 2010b) cannot be broken down into the same steps as the rest of the approaches. This is indicated by fusing the columns of some steps in the table for these algorithms.

By inspecting the table, we can learn about the variety of algorithmic approaches that have been employed for melody extraction. Lets start with the last column of the table titled 'Approach Type': here we have attempted to classify all the algorithms based on their underlying approach. We see that most of the approaches fall into one of the two main categories introduced earlier: *salience based* and *source separation based*. Some approaches, however, do not fit into either category: a *data driven* approach in which the power spectrum is fed directly into a machine learning algorithm which attempts to classify the melody frequency based on the observed spectrum at

Algorithm [MIREX year]	Preprocessing	Spectral Transform and Processing	Multipitch Represent. (salience function)	Tracking	Voicing	Approach Type
Paiva et al. (2006) [2005]	-	Auditory model + autocorrelation peaks	Summary correlogram	Multipitch trajectories + note deletion	Salience valleys	Salience based
Marolt (2004) [2005]	-	STFT + SMS harmonics plus noise	EM fit to tone models	Fragments + fragment clustering	Loudness filter	
Goto (2004) [2005]	Bandpass filter	Multirate filterbank + IF-based peak selection	EM fit to tone models	Tracking agents	-	
Cancela (2008) [2008]	-	Constant-Q + high pass filter + log power normalisation	Harmonicity map	Contour tracking + weighting + smoothing	Adaptive threshold	
Ryynänen & Klapuri (2008a) [2008]	-	STFT + spectral whitening	Harmonic summation	Note event HMM + global HMM	Silence model	
Dressler (2011a) [2009]	-	MRFFT + IF peak correction + magnitude threshold	Pairwise comparison of spectral peaks	Streaming rules	Dynamic threshold	
Rao & Rao (2010) [2009]	-	High resolution FFT + main-lobe magnitude matching	SMS + TWM	Dynamic programming	NHC threshold	
Salamon & Gómez (2012) [2011]	Equal loudness filter	STFT + IF peak correction	Harmonic summation	Contour tracking + contour filtering	Salience distribution	
Jo et al. (2010) [2011]	-	STFT with varying window length	Harmonic summation	Stable candidates + rule based selection	Implicit	
Arora & Behera (2013) [2012]	-	STFT + log spectrum + peak selection	IFT of log spectrum	Harmonic cluster tracking + cluster score	Harm. sum. threshold	
Hsu & Jang (2010a) [2010]	Harm./perc. sound sep.	MRFFT + vocal partial discrimination	Normalised sub-harmonic summation	Global trend + dynamic programming	Classification	Salience based + source sep. preprocessing
Yeh et al. (2012) [2012]	Harm./perc. sound sep.	MRFFT + vocal partial discrimination	Normalised sub-harmonic summation	Trend estimation + HMM	-	
Durrieu et al. (2010) [2009]	Source/filter model for melody source separation			Viterbi smoothing	Energy threshold	Source separation
Tachibana et al. (2010b) [2011]	Two stage harmonic/percussive sound separation			Dynamic programming	Signal/noise ratio thresh.	
Poliner & Ellis (2005) [2006]	Downsample to 8kHz	STFT + limit to 2kHz + normalise magnitude	N/A	Support Vector Machine classifier	Energy threshold	Data driven
Sutton (2006) [2006]	Semitone att. + bandpass	N/A	N/A	HMM combination of monophonic pitch trackers	Confidence HMM	Monophonic

Table 2.1: Algorithm architecture of 16 melody extraction algorithms that have participated in MIREX between 2005 and 2012.

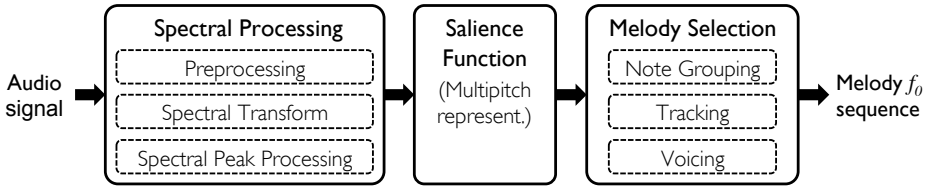

Figure 2.1: Block diagram of salience based melody extraction algorithms.

each frame; and a *monophonic* approach, in which the output of monophonic pitch trackers (even though they were not designed to handle polyphonic signals) is combined to produce a final estimate.

Note that whilst melody extraction includes detecting both sung melodies and melodies played by lead instruments, many algorithms are developed particularly for singing voice extraction. The reason for this is twofold: first, there is a large body of popular music with sung melodies, which makes vocal melody extraction commercially attractive. Second, the singing voice has unique characteristics which are different from most instruments (Sundberg, 1987), and algorithms can exploit these unique features to identify the melody more accurately.

2.3.1 Salience based approaches

As evident in Table 2.1, the largest set of approaches are those based on time-frequency representations of pitch salience (a salience function). The general architecture of these approaches, with possible sub-steps, is depicted in Figure 2.1.

Preprocessing

As a first step, some approaches apply some type of preprocessing, normally a filter to enhance the frequency content where we expect to find the melody: Goto (2004) applies a band pass filter between 261.6 Hz and approximately 4 kHz, whilst Salamon & Gómez (2012) apply a perceptually motivated equal loudness filter (Moore, 2003; Robinson & Dadson, 1956). Some approaches use source separation to enhance the melody signal before it is further processed: Hsu & Jang (2010a) and Yeh et al. (2012) use a technique originally designed for harmonic-percussive sound separation (HPSS) adapted to perform melody-accompaniment separation (cf. Section 2.3.2).

Spectral transform and processing

Next, the signal is chopped into time frames and a transform function is applied to obtain a spectral representation of each frame. The most straight forward approach is to apply the short-time Fourier transform (STFT), with a window length typically between 50 and 100ms (Arora & Behera, 2013; Marolt, 2004; Rao & Rao, 2010; Ryynänen & Klapuri, 2008a; Salamon & Gómez, 2012). Such a window length usually provides sufficient frequency resolution to distinguish different notes whilst maintaining adequate time resolution to track pitch changes in the melody over short time periods. Still, some approaches attempt to overcome the time-frequency resolution limitation inherent to the Fourier transform by applying a multiresolution transform such as a multirate filterbank (Goto, 2004), the constant-Q transform (Brown, 1991; Cancela, 2008) or the multi-resolution FFT (MRFFT; Dressler, 2006; Hsu & Jang, 2010a; Yeh et al., 2012). In general, these transforms use larger windows at low frequencies (where we require greater frequency resolution to resolve close notes) and smaller windows at higher frequencies (where we need high temporal resolution to track rapidly changing harmonics). Salamon et al. (2011) compared between the STFT and the MRFFT and showed that there was no statistically significant difference between using one transform over the other for their melody extraction algorithm (cf. Chapter 3). Nonetheless, since each step in a melody extraction system is highly sensitive to the output of the preceding step, it is possible that some algorithms do benefit from using multiresolution transforms. Finally, we note that some methods use transforms designed to emulate the human auditory system (Moore, 2003) such as the model used by Paiva et al. (2006).

After applying the transform, most approaches only use the spectral peaks for further processing. Apart from detecting the peaks themselves, different peak processing techniques may be applied: some methods filter peaks based on magnitude or sinusoidality criteria in an attempt to filter out peaks that do not represent harmonic content or the lead voice (Goto, 2004; Hsu & Jang, 2010a; Marolt, 2004; Rao & Rao, 2010; Yeh et al., 2012). Other approaches apply spectral magnitude normalisation in an attempt to reduce the influence of timbre on the analysis – Cancela (2008) and Arora & Behera (2013) take the log spectrum and Ryynänen & Klapuri (2008a) (who use the whole spectrum, not just the peaks) apply spectral whitening. Finally, Dressler (2006) and Salamon & Gómez (2012) obtain more accurate frequency and amplitude estimates for each spectral peak by computing its instantaneous frequency from the phase spectrum.

Salience function

At the core of salience based algorithms lies the multipitch representation, i.e. the salience function. This function provides an estimate of the salience of each possible pitch value (within the range where we expect to find the melody) over time. An example of the output of a salience function (proposed by Salamon & Gómez, 2012) is depicted in Figure 2.2. The peaks of this function are taken as possible candidates for the melody, which are further processed in the next stages. Different methods can be used to obtain a salience function: most approaches use some form of harmonic summation, by which the salience of a certain pitch is calculated as the weighted sum of the amplitude of its harmonic frequencies (Cancela, 2008; Hsu & Jang, 2010a; Jo et al., 2010; Ryynänen & Klapuri, 2008a; Salamon & Gómez, 2012; Yeh et al., 2012). Goto (2004) and Marolt (2004) use expectation maximisation (Dempster et al., 1977) to fit a set of tone models to the observed spectrum. The estimated maximum a posteriori probability of the tone model whose f_0 corresponds to a certain pitch is considered to be the salience of that pitch. Other approaches include two-way mismatch (Maher & Beauchamp, 1994) computed by Rao & Rao (2010), summary autocorrelation used by Paiva et al. (2006) and pairwise analysis of spectral peaks as done by Dressler (2011b).

As evident in Figure 2.2, the salience function approach has one main undesirable effect – the appearance of "ghost pitches" whose f_0 is an exact multiple (or sub-multiple) of the f_0 of the actual pitched sound. This effect can lead to what is commonly referred to as *octave errors*, in which an algorithm selects a pitch value which is exactly one octave above or below the correct pitch of the melody. This type of error can be observed in example (c) of Figure 2.6, which shall be discussed in greater detail in Section 2.6. Different algorithms adopt different strategies to reduce the number of octave errors they produce. Some algorithms, such as the ones by Cancela (2008) and Dressler (2011a) attempt to directly reduce the number of ghost pitches present in the salience function. Dressler does this by examining pairs of spectral peaks which potentially belong to the same harmonic series and attenuating the result of their summation if there are many high amplitude spectral peaks whose frequencies lie between the pair being considered. Cancela attenuates the harmonic summation supporting a certain f_0 if the mean amplitude of spectral components at frequencies $2k \cdot f_0$, $3k \cdot f_0/2$ and $3k \cdot f_0$ is above the mean of the components at frequencies $k \cdot f_0$ (this will attenuate ghost pitches whose f_0 is 1/2, 2/3 or 1/3 of the real f_0). Klapuri (Klapuri, 2003, 2004) proposes a method for reducing octave er-

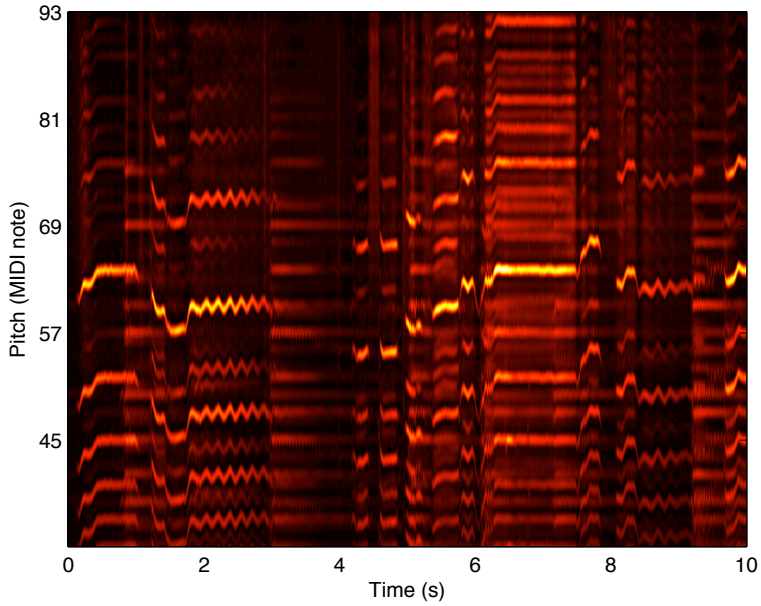

Figure 2.2: Example of the output of a salience function for an excerpt of vocal jazz (cf. example (a) in Figure 2.6) computed using the algorithm proposed by Salamon & Gómez (2012).

rors based on spectral smoothness. In this technique, designed for resolving the amplitudes of overlapping partials, the amplitude of each peak in the salience function is recalculated after smoothing the spectral envelope of its corresponding harmonic frequencies. Peaks representing octave errors will have an irregular envelope (compared to a smoother envelope for real notes) and thus will be attenuated by this process. An alternative strategy for coping with octave errors is proposed by Paiva et al. (2006) and Salamon & Gómez (2012), who first group the peaks of the salience function into pitch contours and then determine which contours are actually ghost contours and remove them. The underlying idea is that once the salience peaks are grouped into contours, detecting duplicate contours becomes easier since they have identical trajectories one octave apart. Determining which of the two is the ghost contour is done using criteria based on contour salience and the overall pitch continuity of the melody. Finally, we note that practically all methods reduce octave errors non-explicitly by penalising large jumps in pitch during the tracking stage of the algorithm.

Tracking

Given the peaks of the salience function, the remaining task is to determine which peaks (i.e. pitches) belong to the melody. This is one of the most crucial stages of each algorithm and, interestingly, it is also perhaps the most varied step where practically every algorithm uses a different approach. Most approaches attempt to directly track the melody from the salience peaks, though some (namely Cancela, 2008; Marolt, 2004; Paiva et al., 2006 and Salamon & Gómez, 2012) include a preliminary grouping stage where peaks are grouped into continuous pitch contours (also referred to as 'fragments' or 'trajectories') out of which the melody is later selected. This grouping is usually performed by tracking sequential peaks based on time, pitch and salience continuity constraints inspired by auditory streaming cues (Bregman, 1990). Given the pitch contours (or salience peaks if no grouping is applied), a variety of tracking techniques have been proposed to obtain the final melody sequence: Marolt (2004) uses clustering, whilst Goto (2004) and Dressler (2011a) use heuristic-based tracking agents. Ryynänen & Klapuri (2008a) and Yeh et al. (2012) use HMMs, whilst Rao & Rao (2010) and Hsu & Jang (2010a) use dynamic programming. Finally, Paiva et al. (2006) and Salamon & Gómez (2012) take a different approach – instead of tracking the melody, they attempt to remove all the pitch contours (or notes) that do not belong to the melody.

Voicing

An important part of melody extraction which is sometimes overlooked is *voicing detection*. That is, determining when the melody is present and when it is not. The voicing detection step of an algorithm is usually applied at the very end, though exceptions do exist: for instance, Salamon & Gómez (2012) use a threshold based on the salience distribution of pitch contours in the entire piece to remove non-salient contours before proceeding to filter out other non-melody contours. A common approach is to use a fixed or dynamic per-frame salience-based threshold, as done by Paiva et al. (2006), Marolt (2004), Cancela (2008), Dressler (2011a), Rao & Rao (2010) and Arora & Behera (2013). Alternative strategies include the algorithm by Ryynänen & Klapuri (2008a) which incorporates a silence model into the HMM tracking part of the algorithm, and the algorithm by Hsu & Jang (2010a) which uses timbre based classification to determine the presence or absence of the human voice.

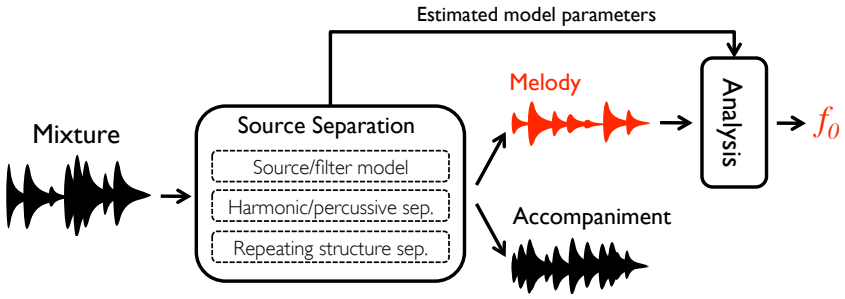

Figure 2.3: Block diagram of source-separation based melody extraction algorithms.

2.3.2 Source separation based approaches

An alternative strategy to salience based melody extraction is to use source separation to isolate the melody source from the mixture. A block diagram illustrating some of the possible strategies for melody extraction using source separation is provided in Figure 2.3. This class of approaches is the most recent of the ones included in Table 2.1, and has gained popularity in recent years following the advances in audio source separation research. Whilst there is a large body of research on melody and lead voice source separation (cf. the work by Durrieu et al., 2011, 2010; Hsu & Jang, 2010b; Huang et al., 2012; Li & Wang, 2007; Liutkus et al., 2012; Ozerov et al., 2007; Rafii & Pardo, 2013; Vembu & Baumann, 2005 and references therein), such algorithms are usually evaluated using measures based on signal-to-noise ratios, and only few have been evaluated in terms of estimating the frequency sequence of the melody, as is our goal here.

Two of the methods in Table 2.1 are source separation based – those of Durrieu et al. (2010) and Tachibana et al. (2010b). Durrieu et al. (2010) model the power spectrogram of the signal as the instantaneous sum of two contributions: the lead voice and the accompaniment. The contribution of the lead voice is represented with a source/filter model, and the contribution of the accompaniment as the sum of an arbitrary number of sources with distinct spectral shapes. Two different representations are proposed for the source/filter model: a Smooth Instantaneous Mixture Model (SIMM) and a Smooth Gaussian Scaled Mixture Model (SGSMM). The former represents the lead instrument (or voice) as the instantaneous mixture of all possible notes, whilst the latter is more realistic in that it only allows one source/filter couple to be active at any moment, albeit computationally

heavier. In both cases, the model parameters are estimated using an expectation maximisation framework. Once the model parameters are estimated, the final melody sequence is obtained using the Viterbi algorithm to find a smooth trajectory through the model parameters (which include the f_0 of the source). Voicing detection is done by first using Wiener filtering (Benaroya et al., 2006) to separate the melody signal based on the estimated model parameters, and then computing the energy of this signal at every frame to determine an energy threshold for frames where the melody is present.

It is worth commenting that in some cases, such as in the case of the SIMM variant of the aforementioned approach, the classification of the approach as salience based or source separation based is not so clear cut. Here we have chosen to classify it as a source separation based approach since, unlike other salience based algorithms, it is an analysis-by-synthesis method (i.e. the signal model parameters are estimated by generating an estimate of the signal and minimising the error with respect to the observed signal). This is different from the more "analytical" salience based approaches where the salience is computed directly from the observed signal in a "lossy" process which can not be inverted back into an audio signal. Still, Durrieu et al. (2010) obtain the f_0 sequence of the melody using Viterbi decoding of the model (which could be viewed as a salience function) and not by running a pitch tracker on the separated signal, and so the approach could be viewed from a salience-based perspective as well (indeed, it bears certain similarities with Goto's PreFEst system; Goto, 2004).

The approach proposed by Tachibana et al. (2010b) is quite distinct. It is based on exploiting the temporal variability of the melody compared to more sustained chord notes. To do so, they make use of the Harmonic-Percussive Sound Separation (HPSS) algorithm (Ono et al., 2010). The algorithm was originally designed to separate harmonic from percussive elements in a sound mixture by separating sources that are smooth in time (harmonic content) and sources that are smooth in frequency (percussive content). By changing the window length used for the analysis, the algorithm can be used to separate "sustained" (i.e. chord) sounds from "temporally variable" (melody + percussive) sounds. Once the chords (i.e. accompaniment) are removed, the algorithm is run again, this time in its original form in order to remove percussive elements. After these two passes, the melody in the resulting signal should be significantly enhanced. The melody frequency sequence is obtained directly from the spectrogram of the enhanced signal using dynamic programming by finding the path which maximises the max-

imum a posteriori probability of the frequency sequence, where the probability of a frequency given the spectrum is proportional to the weighted sum of the energy at its harmonic multiples, and transition probabilities are a function of the distance between two consecutive frequency values. Voicing detection is done by setting a threshold on the (Mahalanobis) distance between the two signals produced by the second run of the HPSS algorithm (the melody signal and the percussive signal).

In Table 2.1 we see that some authors attempt to combine salience based and source separation approaches. Here, source separation is used as a preprocessing step to attenuate the accompaniment signal, and then a salience function is computed from the processed signal. Both Hsu & Jang (2010a) and Yeh et al. (2012) use the HPSS algorithm mentioned earlier to perform this preprocessing, but rather than estimate the melody directly from the spectrum of the separated signal as done by Tachibana et al. (2010b), they use the separated signal to compute a salience function and proceed similarly to other salience based approaches.

Finally, we briefly describe some singing-voice source separation algorithms here. As mentioned earlier, whilst these methods have not been evaluated in terms of melody extraction, they could be used to build melody extraction systems by combining them with a monophonic pitch tracking algorithm which estimates the melody f_0 sequence from the separated voice signal, or by using them as a preprocessing step similar to the aforementioned approaches by Hsu & Jang (2010a) and Yeh et al. (2012). Two separation methods already described above are the source/filter model proposed by Durrieu et al. (2010) and the HPSS method employed by Tachibana et al. (2010b). A different strategy for separating the lead voice is to exploit the fact that the music accompaniment often has a repetitive structure, whilst the voice contains more variation. Huang et al. (2012) exploit this by assuming that the spectrogram of the accompaniment can be modelled by a low-rank matrix, and the spectrogram of the voice by a sparse matrix. They use robust principal component analysis (Candès et al., 2009) to factorise the spectrogram of the signal into the desired voice and accompaniment matrices. A different way of exploiting repetition is proposed by Rafii & Pardo (2013) – they first compute the repetition period of the accompaniment using autocorrelation applied to the spectrogram of the mixture. By computing the median of the spectrograms of consecutive repetitions, they obtain a spectrogram which contains only the repeating signal (i.e. the accompaniment). This spectrogram is used to derive a time-frequency mask which is then used to separate the voice from the accompaniment. This ap-

proach was extended by Liutkus et al. (2012) to work on full songs (where the repetition period can change between verse and chorus) by searching for local periodicities in a song, and again by Rafii & Pardo (2012) who propose to compute a self-similarity matrix to better identify repeating segments in a song and exploit this information to apply the algorithm to local windows of the signal. Rafii & Pardo (2013) also present some experiments on combining their approach with existing pitch trackers to perform melody extraction, though the dataset and evaluation measures used do not allow a direct comparison with the results presented later on in this chapter. Still, it is quite probable that the number of source-separation-based melody extraction algorithms participating in MIREX will increase in the future as source separation techniques become faster and more accurate.

2.3.3 Alternative approaches

Whilst most melody extraction methods are either salience or source separation based, some very different strategies have been proposed as well. The first to appear in Table 2.1 is the data driven approach by Poliner & Ellis (2005). Rather than handcraft knowledge about musical acoustics into the system (e.g. in the form of a salience function based on harmonic summation), they propose to use machine learning in order to train a classifier to estimate the melody note directly from the power spectrum. As a preprocessing step they downsample the audio to 8 kHz, and use the STFT to obtain a spectral representation. Bins corresponding to frequencies above 2 kHz are discarded and the magnitude of the remaining bins is normalised over a short time period to reduce the influence of different instrument timbres. The resulting 256 feature vector is used to train a support vector machine classifier using training data where each frame is labelled with one of 60 possible output classes corresponding to 60 MIDI notes spanning five octaves. Voicing detection is done by means of a global threshold based on the magnitude squared energy found between 200 and 1800 Hz.

Another completely different strategy is the one proposed by Sutton (2006). Rather than design an algorithm to handle polyphonic audio signals, he computes the pitch sequences returned by two different monophonic pitch estimators and then combines them using an HMM. The underlying assumption is that whilst monophonic pitch trackers are not designed to handle audio where there is more than one pitch present at a time (normally leading to a large degree of estimation errors), by combining the output of different trackers a more reliable result could be obtained.

2.4 Evaluation: measures and music collections

As explained earlier, melody extraction algorithms are expected to accomplish two goals: estimate the correct pitch of the melody (pitch estimation), and estimate when the melody is present and when it is not (voicing detection). The output of a melody extraction algorithm typically includes two columns, the first with timestamps at a fixed interval (e.g. for MIREX a 10 ms interval is used), and the second with f_0 values representing the algorithm's pitch estimate for the melody at each timestamp (i.e. at each analysis frame). For every frame the algorithm also indicates whether it estimates the melody to be present or absent in that frame (this is typically indicated in a third output column or by returning an f_0 value with a negative sign for frames where the melody is estimated to be absent). In this thesis, we shall use the term *voiced* to refer to frames where the melody is present, and *unvoiced* to refer to frames where the melody is not present. This is the standard terminology used in the melody extraction literature, and it is used regardless of whether the melody is sung by a human voice or played by an instrument. It is important to note the difference between the use of the terms *voicing*, *voiced* and *unvoiced* in the melody extraction literature (and this thesis) and their use in other fields such as linguistics and speech processing, where the term unvoiced (or *voiceless*) is usually used to refer to sounds pronounced without the vocal folds vibrating. For evaluation purposes, algorithms can report a pitch even for frames where they estimate the melody to be absent (i.e. unvoiced frames). This allows us to evaluate an algorithm's pitch estimation accuracy independently of its voicing detection accuracy. In other words, even if an algorithm wrongly estimates a voiced frame as unvoiced, it can still provide a pitch estimate for this frame, thus ensuring that its pitch estimation accuracy is not biased by voicing detection errors.

To evaluate the performance of an algorithm for a given audio excerpt, we compare the algorithm's output with the excerpt's *ground truth*. The ground truth file has the same format as the output file, and contains the correct sequence of f_0 values representing the melody of the excerpt. The ground truth is produced by running a monophonic pitch tracker on the solo melody track of the excerpt (meaning we require access to the multitrack recording of every song we use for evaluation). Using a graphical user interface such as SMSTools[2] or WaveSurfer[3], the output of the monophonic pitch tracker is

[2] http://mtg.upf.edu/technologies/sms
[3] http://www.speech.kth.se/wavesurfer/

manually inspected and corrected if necessary. Given the ground truth file, an algorithm is evaluated by comparing its output on a per-frame basis to the ground truth. For unvoiced frames in the ground truth, the algorithm is expected to indicate that it has detected the absence of melody. For voiced frames, the algorithm is expected to return a frequency value matching the one in the ground truth. An algorithm's frequency estimate is considered correct if it is within 50 cents (i.e. half a semitone) of the value in the ground truth.

2.4.1 Measures

Based on this per-frame comparison, we compute five global measures which assess different aspects of the algorithm's performance for the audio excerpt in question. These measures were first used in MIREX 2005 (Poliner et al., 2007), and have since become the de facto set of measures for evaluating melody extraction algorithms.

Let the algorithm's estimated melody pitch frequency sequence be represented by a unidimensional vector $\mathbf{f}$, and the ground truth frequency sequence by $\mathbf{f}^*$ (we use $\mathbf{f}$ rather than $\hat{\mathbf{f}}$ to represent the estimated melody sequence in this section to simplify the notation in the formulae below). Let us also define a voicing indicator vector $\mathbf{v}$, whose τ^{th} element $v_\tau = 1$ when the algorithm estimates the τ^{th} frame to be voiced (i.e. it estimates the melody is present in this frame), with corresponding ground truth $\mathbf{v}^*$. For the sake of clarity, we also define an "unvoicing" indicator $\bar{v}_\tau = 1 - v_\tau$. Recall that an algorithm may report an estimated melody pitch ($f_\tau > 0$) even for times where it reports the frame to be unvoiced ($v_\tau = 0$). The measures are then defined as follows:

- **Voicing Recall Rate**: the proportion of frames labelled as voiced in the ground truth that are estimated as voiced by the algorithm:

$$\text{Rec}_{\text{vx}} = \frac{\sum_\tau v_\tau v_\tau^*}{\sum_\tau v_\tau^*} \tag{2.8}$$

- **Voicing False Alarm Rate**: the proportion of frames labelled as unvoiced in the ground truth that are mistakenly estimated as voiced by the algorithm:

$$\text{FA}_{\text{vx}} = \frac{\sum_\tau v_\tau \bar{v}_\tau^*}{\sum_\tau \bar{v}_\tau^*} \tag{2.9}$$

- **Raw Pitch Accuracy**: the proportion of voiced frames in the ground truth for which f_τ is considered correct (i.e. within half a semitone of the ground truth f_τ^*):

$$\text{Acc}_{\text{pitch}} = \frac{\sum_\tau v_\tau^* \mathcal{T}\left[\mathcal{C}(f_\tau) - \mathcal{C}(f_\tau^*)\right]}{\sum_\tau v_\tau^*} \tag{2.10}$$

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

$$\mathcal{T}[a] = \begin{cases} 1 & \text{if } |a| < 50 \\ 0 & \text{if } |a| \geq 50 \end{cases} \tag{2.11}$$

  and $\mathcal{C}$ is a function that maps a frequency value in Hz to a perceptually motivated axis where every semitone is divided into 100 *cents*, and so a frequency can be expressed as the real-valued number of cents above an arbitrary reference frequency f_{ref}:

$$\mathcal{C}(f) = 1200 \log_2 \left(\frac{f}{f_{\text{ref}}} \right). \tag{2.12}$$

  Throughout this thesis we use $f_{\text{ref}} = 55$ Hz (piano note A1). Note that since 100 cents = 1 semitone, the 50 cent threshold in Equation 2.11 is equivalent to half a semitone.

- **Raw Chroma Accuracy**: same as the raw pitch accuracy, except that both the estimated and ground truth f_0 sequences are mapped onto a single octave. This gives a measure of pitch accuracy which ignores octave errors, a common error made by melody extraction systems (cf. Section 2.6):

$$\text{Acc}_{\text{chroma}} = \frac{\sum_\tau v_\tau^* \mathcal{T}\left[\langle \mathcal{C}(f_\tau) - \mathcal{C}(f_\tau^*) \rangle_{1200}\right]}{\sum_\tau v_\tau^*} \tag{2.13}$$

  Octave equivalence is achieved by taking the difference between the cent-scale pitch values modulo 1200 (one octave), where

$$\langle a \rangle_{1200} = a - 1200 \lfloor \frac{a}{1200} + 0.5 \rfloor. \tag{2.14}$$

- **Overall Accuracy**: this measure combines the performance of the pitch estimation and voicing detection tasks to give an overall performance score for the system. It is defined as the proportion of all frames correctly estimated by the algorithm, where for unvoiced frames this

means the algorithm labelled them as unvoiced, and for voiced frames the algorithm both labelled them as voiced and provided a correct f_0 estimate for the melody (i.e. within half a semitone of the ground truth):

$$\text{Acc}_{\text{overall}} = \frac{1}{L} \sum_{\tau} v_\tau^* \mathcal{T} \left[\mathcal{C}(f_\tau) - \mathcal{C}(f_\tau^*) \right] + \bar{v}_\tau^* \bar{v}_\tau \tag{2.15}$$

where L is the total number of frames.

Note that all measures go from 0 (worst case) to 1 (best case), with the exception of the voicing false alarm rate FA_{vx} where 0 represents the best case and 1 the worst case. The performance of an algorithm on an entire music collection for a given measure is obtained by averaging the per-excerpt scores for that measure over all excerpts in the collection.

2.4.2 Music collections

Over the years, different research groups have contributed annotated music collections for evaluating melody extraction in MIREX. The limited amount of multitrack recordings freely available, and the time-consuming annotation process, mean that most of these collections are relatively small compared to those used in other MIR tasks. The collections currently used for evaluation in MIREX, which have remained fixed since 2009, are described below.

ADC2004

This collection was compiled and annotated by Emilia Gómez, Beesuan Ong and Sebastian Streich of the Music Technology Group, Universitat Pompeu Fabra, for the Audio Description Contest (ADC) held in conjunction with the 2004 ISMIR conference in Barcelona (Cano et al., 2006). It consists of 20 excerpts of roughly 20 s in the genres of pop, jazz and opera. The excerpts include real audio recordings (12), excerpts where the melody is produced by a singing-voice synthesiser (4) and excerpts synthesised from MIDI files (4). 8 of the real recordings are vocal and 4 are instrumental with the melody played by a saxophone. Originally half of the excerpts were available for training and the other half were kept private for testing. Since 2005 all 20 excerpts are publicly available for training. The total play time of the collection is 369 s.

MIREX05

This collection, compiled and annotated by Graham Poliner and Daniel P. W. Ellis of the Laboratory for the Recognition and Organization of Speech and Audio (LabROSA) at Columbia University, contains 25 excerpts of 10–40 s duration in the genres of rock, R&B, pop, jazz and solo classical piano. The excerpts include real audio recordings and audio synthesised from MIDI files. All excerpts are kept private by the MIREX organisers for testing purposes. The total play time of the collection is 686 s.

INDIAN08

This collection, compiled and annotated by Vishweshwara Rao and Preeti Rao of the Indian Institute of Technology Bombay, consists of 4 one-minute-long excerpts of north Indian classical vocal performances. Each performance includes singing voice (male or female), tanpura (an Indian drone instrument which is played continually in the background), harmonium (a secondary melodic instrument) and tablas (pitched percussion). Each excerpt was mixed twice, each time with differing amounts of accompaniment, resulting in a total of 8 audio clips. The total play time of the collection is 501 s.

MIREX09 (-5dB), MIREX09 (0dB) and MIREX09 (+5dB)

The MIREX09 collection was compiled by Chao-Ling Hsu and Jyh-Shing Roger Jang of the Multimedia Information Retrieval laboratory at the National Tsing Hua University, Taiwan. It consists of 374 excerpts of Chinese pop karaoke recordings, i.e. amateur singing with synthesised karaoke accompaniment. The melody and accompaniment of each excerpt were mixed at three different melody-to-accompaniment ratios: -5 dB, 0 dB and 5 dB, resulting in three test collections – MIREX09 (-5dB), MIREX09 (0dB) and MIREX09 (+5dB) respectively. The total play time of each of the three collections is 10,022 s.

2.5 Performance

2.5.1 Extraction accuracy

In Figure 2.4 we present the results obtained by the 16 algorithms in Table 2.1 for the MIREX evaluation collections. Note that some algorithms only participated in MIREX before all of the collections were added, meaning we

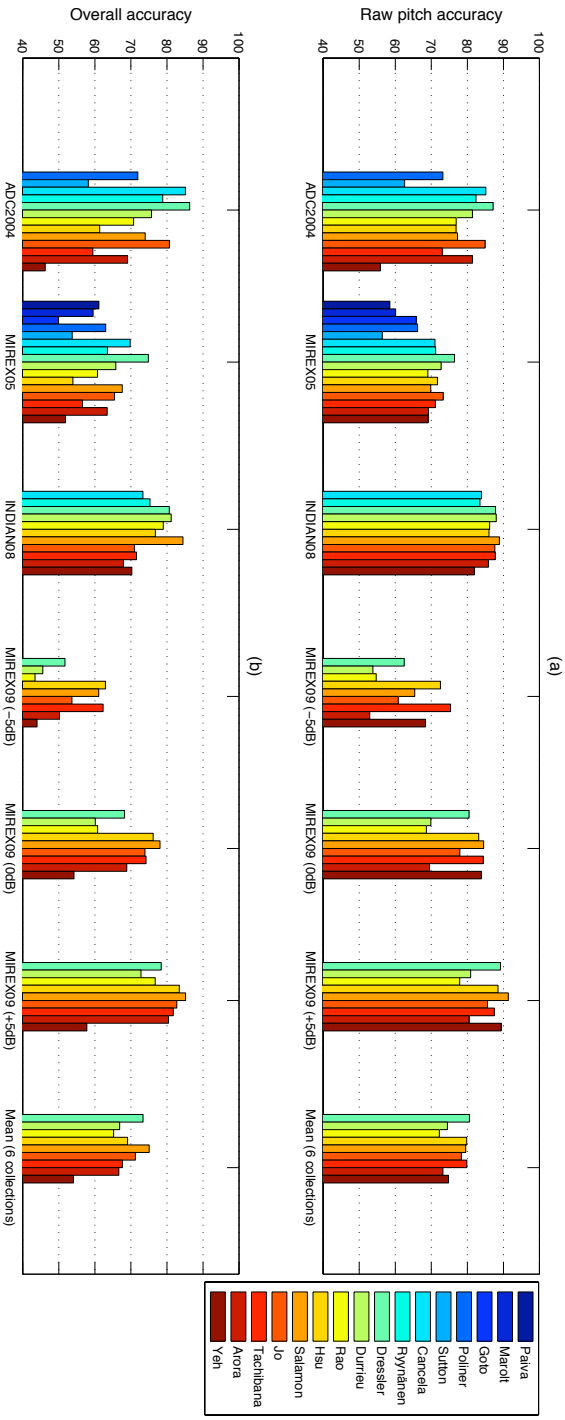

Figure 2.4: (a) Raw pitch accuracy and (b) Overall accuracy obtained in MIREX by the 16 melody extraction algorithms in Table 2.1.

only have partial results for these algorithms. We only compute the mean (represented by the rightmost group of bars in plots (a) and (b) of Figure 2.4) for algorithms that have been evaluated on all six collections. To get a general idea of the performance of these algorithms we focus on two evaluation measures – the raw pitch accuracy (Figure 2.4 (a)) and the overall accuracy (Figure 2.4 (b)). The former tells us how well an algorithm tracks the pitch of the melody, and the latter combines this measure with the efficiency of the algorithm's voicing detection, meaning the voicing-related measures are (to an extent) also reflected in this measure. Starting with the raw pitch accuracy, the first thing we note is that the accuracy of all algorithms varies depending on the collection being analysed. Whilst some collections are generally harder for all approaches (e.g. MIREX09 (-5dB) where the accompaniment is louder and masks the melody), in general the variability in performance is not homogeneous. This highlights the advantages and disadvantages of different approaches with respect to the music material being analysed. For instance, we see that Dressler's method outperforms all others for the ADC2004 and MIREX05 collections, which contain a mixture of vocal and instrumental pieces, but does not for the other collections where the melody is always vocal. On the one hand this means that her approach is generalisable to a wider range of musical material, but on the other hand we see that approaches that take advantage of specific features of the human voice (e.g. Tachibana et al., 2010b or Salamon & Gómez, 2012) can do better on vocal melodies. We also see that the HPSS melody enhancement applied by Hsu & Jang (2010a); Tachibana et al. (2010b) and Yeh et al. (2012) is particularly advantageous when the melody source is relatively weak compared to the accompaniment (the MIREX09 (-5dB) collection). Finally, we note that for the MIREX05 collection the raw pitch accuracy has improved gradually from 2005 to 2009, after which it has remained relatively unchanged (more on the evolution of performance in Section 2.5.2). Overall, we see that the average raw pitch accuracy over all collections lies between 70–80%.

Turning over to the overall accuracy, we see that performance goes down compared to the raw pitch accuracy for all algorithms, since voicing detection is now factored into the results. Note that the results for Goto (2004) and Yeh et al. (2012) are artificially low since these methods do not include a voicing detection step. The importance of this step depends on the intended use of the algorithm. For example, if we intend to use it as a first step in a transcription system, it is very important that we do not include notes that do not belong to the melody in our output. On the other hand,

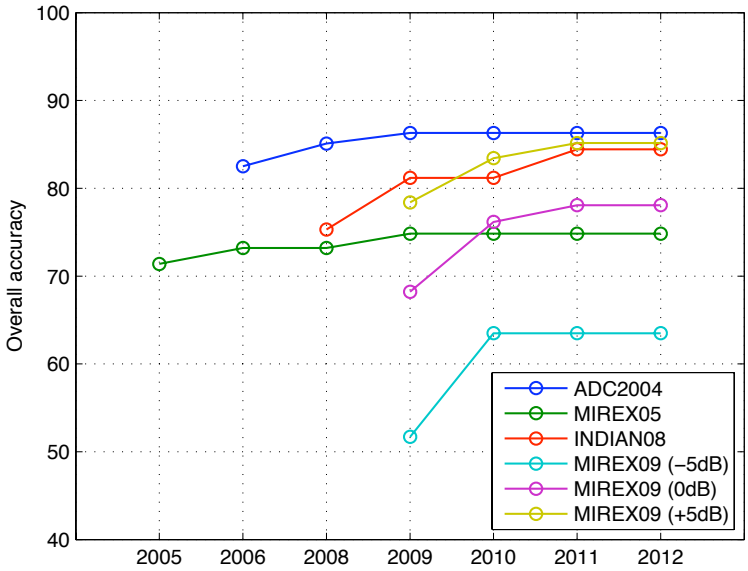

Figure 2.5: Evolution of the best overall accuracy result over the years for the six MIREX collections.

similarity-based applications which rely on matching algorithms that can handle gaps in the alignment of melodic sequences may be less sensitive to voicing mistakes. If we look at the average results over all 6 collections, we see that the algorithms obtaining the best overall accuracy are those that obtain good raw pitch accuracy combined with an effective voicing detection method. Generally, we see that the average overall accuracy results lie between 65% and 75% for the best performing algorithms. Whilst this clearly indicates that there are still many challenges remaining, this degree of accuracy is in fact good enough for new applications to be built on top of melody extraction algorithms (cf. Chapter 6).

2.5.2 Are we improving?

In the previous section we noted that for some collections performance has not improved much over the last 3-4 years. In Figure 2.5 we present the evolution of the overall accuracy obtained for the six MIREX collections over the years. For each collection, we plot the best overall accuracy result obtained up to a given year (i.e. for 2008 we plot the best result obtained up to 2008, for 2009 the best result obtained up to 2009, etc.). Indeed, our previous observation seems to be confirmed – for the two earliest collections

Algorithm	Runtime (minutes)
Dressler	24
Rao	26
Cao1	28
Cao2	33
Wendelboe	132 (~2 h)
Hsu1	344 (~6 h)
Durrieu2	524 (~9 h)
Hsu2	578 (~10 h)
Tachibana	1,468 (~1 d)
Joo	3,726 (~2 d)
Cancela	4,677 (~3 d)
Durrieu1	23,040 (~16 d)

Table 2.2: Total runtime of algorithms participating in MIREX 2009.

(ADC2004 and MIREX05), we observe a steady improvement in results from 2005 to 2009, after which performance does not improve. For the more recent collections (INDIAN08 and the three MIREX09 collections) we see a gradual improvement up to 2011; in 2012 no algorithm outperformed its predecessors for any of the collections. This highlights an important limitation of the MIREX evaluation campaign – since the collections are kept secret, it is very hard for researchers to learn from the results in order to improve their algorithms.

2.5.3 Computational cost

As a final aspect of performance, we consider the computational cost of melody extraction algorithms. Depending on the intended application, we may have limited resources (e.g. time, computing power) and this can influence our decision when choosing which algorithm to use. Whilst deriving O-notation complexity estimates is too complicated for some of the algorithms, a simpler yet informative means of comparing computational cost is by comparing algorithm runtime. In Table 2.2 we provide the runtime values (in minutes) for all the melody extraction algorithms that participated in MIREX 2009[4] (runtime information for subsequent years is not publicly available). Note that this list includes seven (some being earlier

[4] http://www.music-ir.org/mirex/wiki/2009:Audio_Melody_Extraction_Results

versions) of the algorithms considered in Table 2.1 (Dressler, Rao, Durrieu, Hsu, Tachibana, Jo (Joo) and Cancela). The value reported is the time it took each algorithm to process all six music collections used in MIREX. Before considering the results we must include a word of caution: since the emphasis in MIREX is on accuracy and not on runtime, the time and skill spent on optimising the code may differ considerably between implementations (in addition to the actual programming language used). As such, the values in Table 2.2 can only be considered as very rough estimates. Still, even if we only consider the order of magnitude, we see that the difference in runtime can be quite dramatic, ranging from minutes all the way up to several days. Generally, we observe that algorithms involving source separation techniques (which are often implemented as iterative matrix operations) tend to be significantly more computationally complex than salience based approaches.

2.6 Case study

To better understand the challenges of melody extraction and the types of errors afflicting melody extraction algorithms, we now take a closer look at the actual output of a melody extraction algorithm for some musical excerpts. For conciseness, we limit ourselves to one state-of-the-art algorithm (Salamon & Gómez, 2012), but the types of errors we observe (and the challenges they represent) are common to all methods.

In Figure 2.6 we provide the output of the algorithm for three excerpts in the genres of (a) vocal jazz, (b) pop music and (c) opera. Each plot has two panes: in the top pane, we display a log-frequency spectrogram, showing the complex pattern of harmonics associated with these polyphonic musical signals. In the bottom pane we display the final melody line estimated by the algorithm (blue) overlaid on top of the ground truth annotation (red).

Before we can interpret different types of errors in the plots, it is useful to know what a correct extraction looks like, provided in plot (a). We see that the blue (estimated) and red (ground truth) melody sequences overlap almost perfectly, and there are practically no frames where only one sequence is present. The perfect overlap means the pitch estimation of the algorithm is correct. The fact that there are no frames where only one sequence is present indicates we have not made any voicing detection mistakes – a red sequence on its own would mean we wrongly estimated the frame as unvoiced when the melody is actually present. A blue sequence

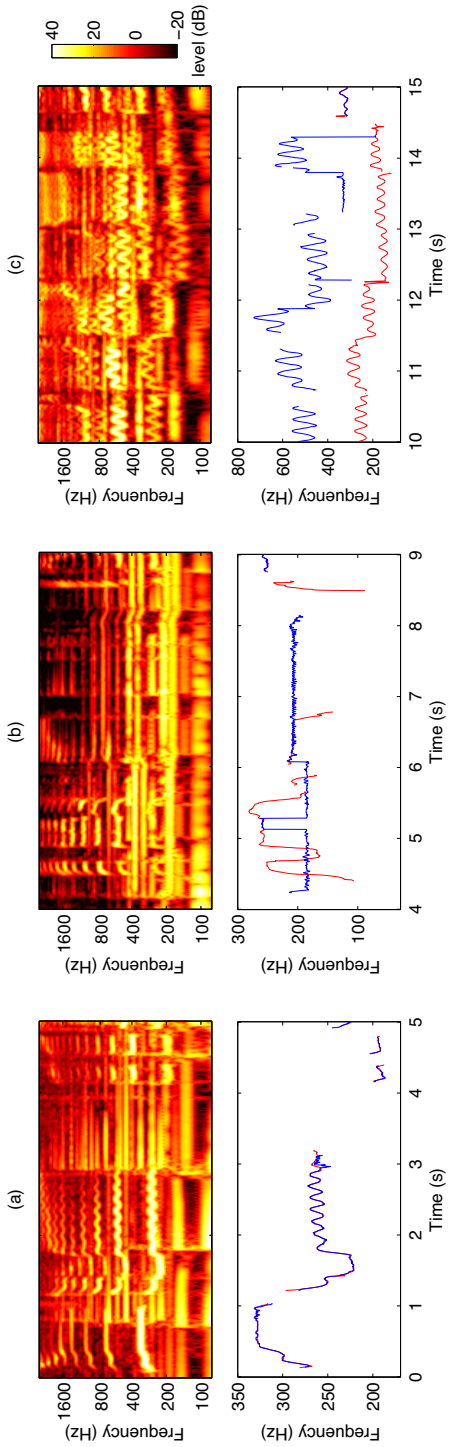

Figure 2.6: Case study examples. Top pane: log-frequency spectrogram. Bottom pane: extracted melody (Salamon & Gómez, 2012; in blue) and ground truth (red). Excerpt: (a) vocal jazz, (b) pop and (c) opera.

on its own would mean a case of voicing false alarm, that is, a frame where we mistakenly included some other pitched source in the melody when the melody is in fact not present in that frame. In plot (a) we see that the algorithm correctly estimates the pitch of the lead singer whilst excluding the notes of the piano chord played between seconds 3 and 4.

In plot (b) we provide an example which contains both pitch errors (seconds 4 to 7) and voicing errors (seconds 7 to 9). The excerpt is taken from a pop song whose arrangement includes a lead singer, guitar accompaniment and backing vocals. Here, the source of both types of errors are the backing vocals, who sing a stable pitch in the same range as the melodic line of the lead singer. As a result, the algorithm mistakenly tracks the backing vocals, resulting in a wrong pitch estimate (up to second 7) followed by a voicing false alarm, since the backing vocals continue after the lead singer has paused.

Finally, in plot (c) we provide an example where the algorithm makes octave errors. In this excerpt, taken from an opera aria sung by a male singer, the pitch class of the melody is correctly estimated but in the wrong octave (one octave above the actual pitch of the singer). Here, the octave errors most likely stem from the actual singing technique used by the singer. Unlike pop or jazz singers, classical singers are trained to produce a highly resonant sound (allowing them to be heard over the orchestra). In the low frequencies this resonance results in the second harmonic often having a larger amplitude than the fundamental frequency (as seen in the spectrogram in the top pane of plot (c) between seconds 10 and 12), and in the high frequencies the appearance (especially in male singers) of a clear formant around 3 kHz (the "singer's formant"; Kreiman & Sidtis, 2011). Combined, these phenomena can cause the algorithm to give more weight to $2f_0$ than to f_0 (f_0 being the correct fundamental frequency). The increased salience at double the true f_0 combined with the relatively low pitch range of the melody (algorithms often bias the tracking against low frequencies) result in the algorithm tracking the melody one octave above the correct pitch, thus producing the observed octave errors.

2.7 Conclusion

In this chapter we have provided the scientific background for the main topic of this thesis: melody extraction from polyphonic music signals. We started by discussing algorithms for monophonic pitch tracking, which can

be considered as the precursors of melody extraction algorithms. We noted that in order to go from monophonic pitch tracking to melody extraction several considerable changes have to be made, including the design of a new salience function that can handle polyphonic signals, and the employment of more advanced tracking techniques in order to identify the melodic line within the polyphony. Two main strategies for melody extraction were identified: salience based methods and source separation based methods.

Next, we provided a detailed review of algorithmic design by considering 16 of the most relevant algorithms submitted to the MIREX evaluation campaign since its initiation in 2005. We noted the great diversity of approaches and signal processing techniques applied, and that the majority of approaches proposed to date are salience based. We discussed the different techniques applied and the overall structure of salience based approaches, which can be broken down into preprocessing, spectral transform, salience function, tracking and voicing. We also described several source separation based approaches (including systems that have not yet been evaluated for melody extraction), and two completely different strategies, one based on classification and the other on combining monophonic pitch trackers. We described the evaluation measures most commonly used to assess melody extraction algorithms, the music collections used in MIREX, and the extraction results obtained for these collections. We saw that the best performing algorithms obtain a raw pitch accuracy between 70–80% and an overall accuracy between 65–75%. We also saw that whilst performance has not improved much for some of the earlier collections, overall performance has improved gradually over the years. We saw that whilst both salience base and separation based approaches yield comparable results, there can be considerable differences in algorithm runtime, and that separation based techniques are often more computationally complex. Finally, by means of a case study we examined the different types of errors made by melody extraction algorithms (pitch estimation errors, octave errors and voicing errors) and identified their possible causes.

Chapter 3

Sinusoid Extraction and Salience Function

3.1 Introduction

In this chapter, based on Salamon et al. (2011), we describe the first two blocks of the melody extraction algorithm proposed in this thesis: *sinusoid extraction* and *salience function*, depicted in Figure 3.1. As noted in Section 1.6, the proposed algorithm is salience-based. That is, the melody pitch sequence is extracted from a time-frequency representation of pitch salience, calculated by means of a salience function. The first two blocks of the algorithm are concerned with obtaining this representation. In the first block, sinusoid extraction, we detect and estimate the frequency and amplitude of the sinusoidal components of the audio signal. In the second block, these components are used to compute the salience function.

As seen in Chapter 2, a variety of approaches have been proposed for extracting sinusoidal components and then computing a salience function, including the application of different preprocessing techniques, spectral transforms, spectral post-processing and salience functions. Whilst the melody extraction algorithms discussed in Chapter 2 have been evaluated and compared in terms of melody extraction performance, their overall complexity makes it hard to determine the effect of the first processing steps of each algorithm on the final result. For this reason, in this section we do not only describe the processing steps used in our algorithm, but also provide a comparative evaluation of the different approaches and parameter values we have considered for each step. Whilst some of the techniques mentioned

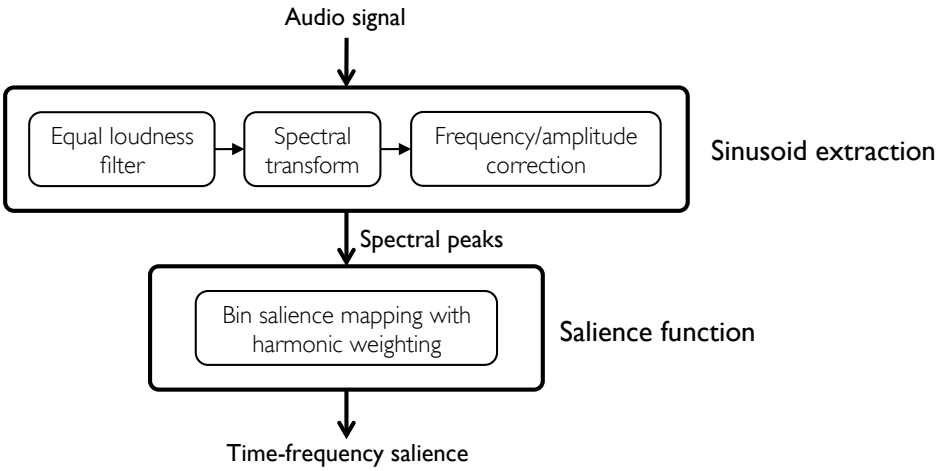

Figure 3.1: First two blocks of the algorithm: sinusoid extraction and salience function.

further down have been compared in isolation before (e.g. Cancela et al. (2009) compare different spectral transforms), our goal is to evaluate them in the specific context of melody extraction. For this purpose, a special evaluation framework, datasets and evaluation measures have been developed. In Section 3.2 we describe the different alternative we have considered for extracting sinusoidal components (spectral peaks), and in Section 3.3 we propose a parametrised salience function based on harmonic summation. In Section 3.4 we describe our methodology for evaluating each of the two blocks, where for the first this involves a comparative evaluation of different processing techniques, and for the second the optimisation of the salience function parameters based on a set of proposed evaluation measures. Then in Section 3.5 we present and discuss the evaluation results for each block, and finally in Section 3.6 we provide a summary of the work and results presented in this chapter.

3.2 Sinusoid extraction

The first block of the algorithm involves obtaining sinusoidal components (spectral peaks) from the audio signal. Goto (2004) refers to this part of a melody extraction algorithm as the *front end*. Different methods have been proposed for obtaining the spectral peaks in the context of melody extraction, usually with two common goals in mind – first, extracting the

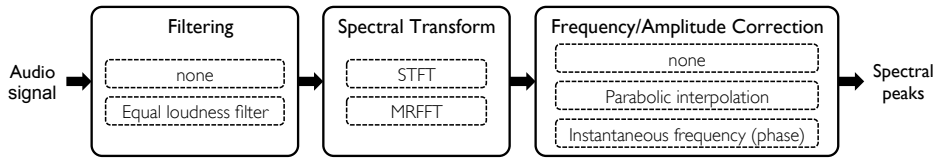

Figure 3.2: Alternative analysis techniques for sinusoid extraction.

spectral peaks as accurately as possible in terms of their frequency and amplitude. Second, enhancing the amplitude of spectral peaks likely to belong to the melody source whilst suppressing the amplitude of background peaks. Our approach divides this process into three steps: filtering, spectral transform and frequency/amplitude correction. The techniques considered at each step are summarised in Figure 3.2.

3.2.1 Filtering

We start by filtering the time-domain signal in order to enhance parts of the spectrum where the frequency content of the melody is more likely to reside. To do this we apply an *equal loudness filter* (Robinson, 2013; Vickers, 2001) which enhances frequencies to which the human listener is more perceptually sensitive (and attenuates those to which they are not). This is done by taking a representative average of the equal loudness curves (Robinson & Dadson, 1956) and filtering the signal by its inverse. The filter is implemented as a 10th order infinite impulse response (IIR) filter cascaded with a 2nd order Butterworth high pass filter, following the implementation proposed by Robinson (2013). Our hypothesis is that this filter is well suited for melody extraction, as it enhances mid-band frequencies where we expect to find the melody, and attenuates low-band frequencies where low pitched instruments (e.g. the bass) can be found.

3.2.2 Spectral transform

We consider two different transforms for obtaining a spectral representation of the signal: the STFT and the Multi-Resolution FFT (MRFFT) proposed by Dressler (2006). In this way we evaluate the effect of using a multi-resolution transform compared to the STFT, in which a fixed window length is used for the analysis (making it a "single resolution" transform).

Short-time Fourier transform (single resolution)

The discrete STFT can be defined as follows:

$$X_l(k) = \sum_{n=0}^{M-1} w(n)x(n+lH)e^{-j\frac{2\pi}{N}kn}, \tag{3.1}$$

$$l = 0, 1, \ldots \text{ and } k = 0, 1, \ldots, N-1$$

where $x(n)$ is the time signal, $w(n)$ the windowing function, l the frame number, M the window length, N the FFT length and H the hop size. We use the Hann windowing function with a window size of 46.4 ms, a hop size of 2.9 ms and a $\times 4$ zero padding factor. For data sampled at $f_S = 44.1$ kHz this gives $M = 2048$, $N = 8192$ and $H = 128$. The relatively small hop size (compared to other MIR tasks; Peeters, 2004) is selected to facilitate more accurate f_0 tracking at the latter stages of the algorithm (cf. Section 4.2).

Given the FFT of a single frame $X(k)$, peaks are selected by finding all the local maxima k_m of the normalised magnitude spectrum $X_m(k)$:

$$X_m(k) = 2\frac{|X(k)|}{\sum_{n=0}^{M-1} w(n)}. \tag{3.2}$$

Peaks with a magnitude more than 80 dB below the highest spectral peak in an excerpt are not considered.

Multi-resolution FFT

We implemented the MRFFT proposed by Dressler (2006). The MRFFT is an efficient algorithm for simultaneously computing the spectrum of a frame using different window sizes, thus allowing us to choose which window size to use depending on whether we require high frequency resolution (larger window size) or high time resolution (smaller window size). The algorithm is based on splitting the summations in the FFT into smaller summations which can be combined in different ways to form frames of varying sizes, and performing the windowing in the frequency domain by convolution. The resulting spectra all have the same FFT length N (i.e. smaller windows are zero padded) and use the Hann windowing function. For further details about the algorithm the reader is referred to Dressler (2006).

In our implementation we set $N = 8192$ and $H = 128$ as with the STFT so that they are comparable. We compute four spectra $X_{256}(k)$, $X_{512}(k)$,

$X_{1024}(k)$ and $X_{2048}(k)$ with respective window sizes of $M = 256$, 512, 1024 and 2048 samples (all windows are centred on the same sample). Then, local maxima (peaks) are found in each magnitude spectrum within a set frequency range as in Dressler (2006), using the largest window (2048 samples) for the first six critical bands of the Bark scale (Zwicker, 1961; 0–630 Hz), the next window for the following five bands (630–1480 Hz), the next one for the following five bands (1480–3150 Hz) and the smallest window (256 samples) for the remaining bands (3150–22050 Hz). The peaks from the different windows are combined to give a single set of peaks at positions k_m, and (as with the STFT) peaks with a magnitude more than 80 dB below the highest peak in an excerpt are not considered.

3.2.3 Frequency and amplitude correction

Given the set of local maxima (peaks) k_m, the simplest approach for calculating the frequency and amplitude of each peak is to directly use its spectral bin location and FFT magnitude (as detailed in equations 3.3 and 3.4 below). This approach is limited by the frequency resolution of the FFT. In order to address this limitation various correction methods have been developed to achieve a higher frequency precision, and consequently a better amplitude estimate as well. Keiler & Marchand (2002) conducted a survey of such methods, evaluating them on artificial, monophonic stationary sounds. In order to evaluate these techniques for melody extraction, we perform a similar evaluation using real-world, polyphonic, quasi-stationary sounds (as is the case in melody extraction). We consider three of the methods discussed in Keiler & Marchand (2002):

Plain FFT with no post-processing

Given a peak at bin k_m, its sine frequency and amplitude are calculated as follows:

$$\hat{f} = k_m \frac{f_S}{N} \tag{3.3}$$

$$\hat{a} = X_m(k_m) \tag{3.4}$$

Note that the frequency resolution is limited by the size of the FFT, in our case the frequency values are limited to multiples of $f_S/N = 5.38$ Hz. This also results in errors in the amplitude estimation as it is quite likely for the true peak location to fall between two FFT bins, meaning the detected peak is actually lower (in magnitude) than the true magnitude of the sinusoidal component.

Parabolic interpolation

This method improves the frequency and amplitude estimation of a peak by taking advantage of the fact that in the magnitude spectrum of most analysis windows (including the Hann window), the shape of the main lobe resembles a parabola in the dB scale. Thus, we can use the bin number and magnitude of the peak together with those of its neighbouring bins to estimate the position (in frequency) and amplitude of the true maximum of the main lobe, by fitting them to a parabola and finding its maximum. Given a peak at bin k_m, we define:

$$A_1 = X_{\text{dB}}(k_m - 1), A_2 = X_{\text{dB}}(k_m), A_3 = X_{\text{dB}}(k_m + 1), \tag{3.5}$$

where $X_{\text{dB}}(k) = 20\log_{10}(X_m(k))$. The frequency difference in FFT bins between k_m and the true peak of the parabola is given by:

$$d = 0.5\frac{A_1 - A_3}{A_1 - 2A_2 + A_3}. \tag{3.6}$$

The corrected peak frequency and amplitude (this time in dB) are thus given by:

$$\hat{f} = (k_m + d)\frac{f_S}{N}, \tag{3.7}$$

$$\hat{a} = A_2 - \frac{d}{4}(A_1 - A_3). \tag{3.8}$$

Note that following the results of Keiler & Marchand (2002), we do not actually use equation 3.8 to estimate the amplitude, but rather we use equation 3.11 given below (with the value of d used as the bin offset $\kappa(k_m)$), which according to the authors gives better results.

Instantaneous frequency using phase vocoder

This technique uses the phase spectrum $\phi(k)$ to calculate the peak's *instantaneous* frequency (IF) and amplitude, which serve as more accurate estimates of its true frequency and amplitude. The IF is computed from the phase difference $\Delta\phi(k)$ of successive phase spectra using the phase vocoder method (Flanagan & Golden, 1966) as follows:

$$\hat{f} = (k_m + \kappa(k_m))\,\frac{f_S}{N}, \tag{3.9}$$

where the bin offset $\kappa(k)$ is calculated as:

$$\kappa(k) = \frac{N}{2\pi H}\Psi\left(\phi_l(k) - \phi_{l-1}(k) - \frac{2\pi H}{N}k\right), \tag{3.10}$$

where Ψ is the principal argument function which maps the phase to the $\pm\pi$ range.

The instantaneous magnitude is calculated using the peak's spectral magnitude $X_m(k_m)$ and the bin offset $\kappa(k_m)$ as follows:

$$\hat{a} = \frac{1}{2}\frac{X_m(k_m)}{W_{\text{Hann}}\left(\frac{M}{N}\kappa(k_m)\right)}, \tag{3.11}$$

where W_{Hann} is the Hann window kernel:

$$W_{\text{Hann}}(z) = \frac{1}{2}\frac{\text{sinc}(z)}{1-z^2}, \tag{3.12}$$

and $\text{sinc}(z)$ is the normalised sinc function. To achieve the best phase-based correction we use $H = 1$, by computing at each hop (of 128 samples) the spectrum of the current frame and of a frame shifted back by one sample, and using the phase difference between the two.

3.3 Salience function design

3.3.1 Introduction

The extracted spectral peaks are used to compute a salience function – a representation of pitch salience over time. The peaks of this function represent possible f_0 candidates for the melody. Before we explain how we compute our salience function, it is worth considering first what features would make a salience function useful for melody extraction. In order to facilitate the extraction of the melody f_0 sequence, the salience function should:

- Be computed over a relevant frequency range. Our function should be limited to the frequency range in which we expect to find the f_0 of the melody.
- Highlight harmonic sources. We are only interested in the tonal (i.e. pitched) content of the signal. The salience function should thus emphasise harmonic sound sources. In particular, the human voice

is a highly harmonic sound source which produces a large number of harmonic partials (Sundberg, 1987), so highlighting such sources should help in identifying the melody when it is sung (which is often the case in Western popular music as well as some non-Western music traditions such as flamenco, Indian classical music or Beijing opera singing).

- Highlight perceptually predominant sources. Whilst it is not always so, it is often the case that the melody source is louder than the accompaniment (especially in Western popular music). Thus, we would like the salience function to emphasise perceptually loud sources.

To achieve these goals, we propose a salience function based on *harmonic summation*. That is, the salience of a given frequency is computed as the sum of the weighted energies found at integer multiples (harmonics) of that frequency. As discussed in Chapter 2, salience functions based on harmonic summation have been used successfully for melody extraction, multi-f_0 estimation and tonal analysis in a variety of algorithms (e.g. Cancela, 2008; Gómez, 2006b; Klapuri, 2006; Ryynänen & Klapuri, 2008a). Harmonic summation can be linked to the ASA principle of *simultaneous integration* (or *perceptual fusion*) proposed by Bregman (1990): when many frequencies are heard at the same time, the ASA system must determine whether they belong to the same sound source, or whether they originate from different sources. This perceptual allocation is based on the probable relations between frequency components when they are produced by the same sound source. As Bregman explains, one of these relations is belonging to the same harmonic series. That is, when a set of frequencies belong to a harmonic series, we are more likely to perceive them as a single, complex, tone. Approaches based on harmonic summation exploit this by estimating the perceived salience of tone as the sum of the energies of the frequencies that contribute to our perception of the tone. With the correct parametrisation, this scheme satisfies our latter two requirements for a salience function – by searching for harmonic series we will be highlighting harmonic sources, and by summing their energy the salience will be biased towards louder sources. To satisfy the first requirement, we simply need to define the frequency range for which the function will be computed: the f_0 of the human singing voice typically ranges from 78 Hz (D$^\sharp$2) to 1046 Hz (C6) if we consider both male and female voices (Kob et al., 2011). Thus, if we consider a range of 5 octaves from 55 Hz (A1) to 1760 Hz (A6) we are guaranteed to cover the

frequency range of practically all sung melodies[1]. In the case of instrumental melodies, the possible frequency range depends on the characteristics of the instrument, and some instruments can go beyond (lower or higher) the aforementioned range. In practice, however, this range is sufficient for many instrumental melodies too (in fact it covers all the melodies in all the test collections that have been annotated for evaluating melody extraction to date), and so for practical reasons we do not consider f_0 values outside this range.

3.3.2 Definition

We propose a salience function that is a modified, parametrised version of the Harmonic Pitch-Class Profile (HPCP) function proposed by Gómez (2006b) for tonal description of polyphonic audio. Rather than search for energies at integer multiples of a candidate f_0 (as done by Klapuri (2006) for example), the salience is computed as a *sub-harmonic summation* (Hermes, 1988). Here, the energy f of every detected spectral peak is mapped (with a weighting scheme) to the frequencies of which f could be a harmonic partial, i.e. f/h where $h = 1, 2, 3 \ldots N_h$ is an integer representing the harmonic number[2] of f with respect to a candidate $f_0 = f/h$. By only using spectral peaks for the summation, we discard spectral energies which can be less reliable due to masking or noise. Using the peaks also allows us to apply the aforementioned frequency correction step which should improve the frequency accuracy of the salience function. Thus, the important factors affecting the salience computation are the number of harmonics considered N_h and the weighting scheme used. In the salience function described below, we control these factors using a set of parameters which need to be optimised for the specific goal of melody extraction.

As explained above, our salience function covers a pitch range of five octaves from 55 Hz to 1.76 kHz. We quantise this range into 600 bins ($b = 0 \ldots 599$) on a cent scale (cf. Section 2.4.1), so that the distance between f_0 candidates can be measured in the same way we perceive the distance between pitches. This gives us a resolution of 10 cents per bin (or 10 bins per semitone), which is sufficient for our intended applications. Given a frequency $\hat{f}$ in Hz,

[1] The Guinness Book of Records lists the highest demanded note in the classical repertoire as G6 in "Popoli di Tessaglia", K. 316, a concert aria by W. A. Mozart, composed for Aloysia Weber.

[2] There is no consensus in the literature for numbering harmonics: some use $h = 1$ to refer to the f_0, whilst others use $h = 1$ to refer to the subsequent frequency in the harmonic series ($2f_0$). In this thesis the former numbering scheme is used throughout.

its corresponding bin $\mathcal{B}(\hat{f})$ is calculated as:

$$\mathcal{B}(\hat{f}) = \left\lfloor \frac{1200 \log_2 \left(\frac{\hat{f}}{55} \right)}{10} + 0.5 \right\rfloor . \tag{3.13}$$

Let the frequencies and linear amplitudes of the spectral peaks returned by the sinusoid extraction block be represented by $\hat{f}_i$ and $\hat{a}_i$ respectively ($i = 1 \ldots I$, where I is the number of peaks found). We then define the salience function $\mathcal{S}(b)$ as:

$$\mathcal{S}(b) = \sum_{h=1}^{N_h} \sum_{i=1}^{I} \mathcal{E}(\hat{a}_i) \mathcal{G}(b, h, \hat{f}_i) \hat{a}_i^{\beta}, \tag{3.14}$$

where β is a magnitude compression parameter, $\mathcal{E}(\hat{a}_i)$ is a magnitude threshold function, and $\mathcal{G}(b, h, \hat{f}_i)$ is a function that defines the weighting scheme. The magnitude threshold function is defined as:

$$\mathcal{E}(\hat{a}_i) = \begin{cases} 1 & \text{if } 20 \log_{10}(\hat{a}_{\max}/\hat{a}_i) < \gamma, \\ 0 & \text{otherwise,} \end{cases} \tag{3.15}$$

where $\hat{a}_{\max}$ is the magnitude of the highest spectral peak in the frame and γ is the maximum allowed difference (in dB) between $\hat{a}_i$ and $\hat{a}_{\max}$. The weighting function $\mathcal{G}(b, h, \hat{f}_i)$ defines the weight given to a peak with amplitude $\hat{a}_i$, when it is considered as the h^{th} harmonic of the f_0 corresponding to bin b:

$$\mathcal{G}(b, h, \hat{f}_i) = \begin{cases} \cos^2(\delta \frac{\pi}{2}) \alpha^{h-1} & \text{if } |\delta| \leq 1, \\ 0 & \text{if } |\delta| > 1, \end{cases} \tag{3.16}$$

where

$$\delta = \frac{|\mathcal{B}(\hat{f}_i/h) - b|}{10} \tag{3.17}$$

is the distance in semitones between the sub-harmonic frequency $\hat{f}_i/h$ and the centre frequency of bin b, and α is a harmonic weighting parameter. Put simply, Equations 3.16 and 3.17 mean that for every spectral peak the salience function adds a $\cos^2$ lobe centred on each of the peak's sub-harmonic frequencies ($\hat{f}_i/h$), where the magnitude of each lobe is weighted according the harmonic number h. This avoids potential problems that could arise due to the quantisation of the frequency range into bins, and also accounts for inharmonicities. In the following sections we describe how we examine the effect of each of the salience function parameters on the resulting salience

Analysis Config.	Filtering	Spectral Transform	Frequency/Amplitude Correction
1	none	STFT	none
2			Parabolic
3			IF (phase)
4		MRFFT	none
5			Parabolic
6			IF (phase)
7	Eq. loudness	STFT	none
8			Parabolic
9			IF (phase)
10		MRFFT	none
11			Parabolic
12			IF (phase)

Table 3.1: Analysis configurations for sinusoid extraction.

representation, with the goal of selecting the parameter combination most suitable for a salience function targeted at melody extraction. The parameters studied are the weighting parameters α and β, the magnitude threshold γ and the number of harmonics N_h used in the summation. An example of the salience representation produced by the proposed salience function is provided in Figure 2.2 of Chapter 2.

3.4 Evaluation methodology

The evaluation is split into two parts. First, we evaluate the different analysis techniques for extracting sinusoids using a similar methodology to that employed by Keiler & Marchand (2002). The combination of different approaches at each step (filtering, transform, correction) gives rise to 12 possible analysis configurations, summarised in Table 3.1. In the second part, we evaluate the effect of the analysis configuration and salience function parameters on the resulting salience representation. In the remainder of this section we describe the experimental setup, ground truth and evaluation measures used for each part of the evaluation.

Genre	Excerpts	Tot. Melody Frames	Tot. Ground Truth Peaks
Opera	5	15,660	401,817
Pop/Rock	3	11,760	769,193
Instrumental Jazz	4	16,403	587,312
Bossa Nova	2	7,160	383,291

Table 3.2: Ground truth for sinusoid extraction and salience function evaluation.

3.4.1 Sinusoid extraction evaluation

Ground truth

Starting with a multi-track recording, the ground truth is generated by analysing the melody track on its own as done by Bonada (2008) to produce a per-frame list of f_0 + harmonics (up to the Nyquist frequency) with frequency and amplitude values. The output of the analysis is then re-synthesised using additive synthesis with linear frequency interpolation and mixed together with the rest of the tracks in the recording. The resulting mix is used for evaluating the different analysis configurations by extracting spectral peaks at every frame and comparing them to the ground truth. In this way we obtain a melody ground truth that corresponds perfectly to the melody in the mixture, whilst being able to use real music as opposed to artificial mixtures.

As we are interested in the melody, only voiced frames are used for the evaluation (i.e. frames where the melody is present). Furthermore, some of the melody peaks will be masked in the mix by the spectrum of the accompaniment, where the degree of masking depends on the analysis configuration used. The measured frequency and amplitude of peaks detected at bins where the melody is masked by the background will depend on the background spectrum, and hence should not be counted as successfully detected melody peaks. To account for this, we compute the spectra of the melody track and the background separately, using the analysis configuration being evaluated. We then check for each peak extracted from the mix whether the melody spectrum is masked by the background spectrum at the peak location (a peak is considered to be masked if the spectral magnitude of the background is greater than that of the melody for the corresponding bin), and if so the peak is discarded. The evaluation material is composed of excerpts from real-world recordings in various genres, summarised in Table 3.2.

Measures

We base our measures on the ones used by Keiler & Marchand (2002), with some adjustments to account for the fact that we are only interested in the spectral peaks of the melody within a polyphonic mixture. At each frame, we start by checking which peaks detected by the algorithm correspond to peaks in the ground truth (melody peaks). A peak is considered a match if it is within 21.5 Hz (equivalent to 1 FFT bin without zero padding) from the ground truth. If more than one match is found, we select the peak closest in amplitude to the ground truth. Once the matching peaks in all frames are identified, we compute the following measures:

- **Peak recall** R_{p}: The total number of melody peaks found by the algorithm in all frames divided by the total number of peaks in the ground truth.
- **Energy recall** R_{e}: The sum of the energy (amplitude) of all melody peaks found by the algorithm divided by the total energy of the peaks in the ground truth.

Given the matching melody peaks, we can compute the frequency estimation error Δf_{c} and the amplitude estimation error Δa_{dB} of each peak. Note that since we are using polyphonic material the amplitude error may not reflect the accuracy of the method being evaluated, and we compute it for completeness. The errors are measured in cents and dB respectively, and averaged over all peaks of all frames to give $\overline{\Delta f_{\mathrm{c}}}$ and $\overline{\Delta a_{\mathrm{dB}}}$. A potential problem with $\overline{\Delta f_{\mathrm{c}}}$ is that the mean may be dominated by peaks with very little energy (especially at high frequencies), even though their effect on the harmonic summation later on will be insignificant. For this reason we define a third measure $\overline{\Delta f_{\mathrm{w}}}$, which is the mean frequency error in cents where each peak's contribution is weighted by its amplitude, normalised by the amplitude of the highest peak in the ground truth of the same frame. The normalisation ensures the weighting is independent of the volume of each excerpt[3]. The frequency and amplitude estimation errors are summarised below:

- **Mean amplitude error (dB)** $\overline{\Delta a_{\mathrm{dB}}}$: average amplitude estimation error for all detected melody peaks.

[3]Other weighting schemes were tested and shown to produce very similar results.

- **Mean frequency error (cents)** $\overline{\Delta f_{\mathrm{c}}}$: average frequency estimation error for all detected melody peaks.
- **Weighted frequency error (cents)** $\overline{\Delta f_{\mathrm{w}}}$: average frequency estimation error for all detected melody peaks weighted by the normalised peak amplitude.

3.4.2 Salience function evaluation

In the second part of the evaluation we take the spectral peaks produced by each of the 12 analysis configurations and use them to compute the salience function with different parameter configurations. The output of the salience function is then evaluated in terms of its usefulness for melody extraction using the ground truth and the measures detailed below.

Ground truth

We use the same evaluation material we generated for evaluating the sinusoid extraction block. The first spectral peak in every row of the ground truth represents the melody f_0, and is used to evaluate the frequency accuracy of the salience function as explained below.

Measures

We evaluate the salience function in terms of two aspects – frequency accuracy and melody salience, where by melody salience we refer to the predominance of the melody compared to the other pitched elements represented in the output of the salience function. For this purpose we have devised four measures, computed on a per-frame basis and finally averaged over all frames.

We start by selecting the peaks of the salience function. The salience peak closest in frequency to the ground truth f_0 is considered the melody salience peak. We can then calculate the frequency error of the salience function Δf_{m} as the difference in cents between the frequency of the melody salience peak and the ground truth f_0. The mean over all frames is denoted $\overline{\Delta f_{\mathrm{m}}}$.

To evaluate the predominance of the melody we propose three measures: the first is the rank R_{m} of the melody salience peak amongst all salience peaks in the frame, which ideally should be 1. Rather than report the rank directly we compute the reciprocal rank $RR_{\mathrm{m}} = 1/R_{\mathrm{m}}$ which is less sensitive to outliers when computing the mean over all frames $\overline{RR_{\mathrm{m}}}$. The

second measure is the relative salience S_1 of the melody peak, computed by dividing the salience of the melody peak by that of the highest peak in the frame. The average over all frames is denoted $\overline{S_1}$. The third metric, S_3, is the same as S_1 only this time we divide the salience of the melody peak by the mean salience of the top 3 peaks of the salience function. In this way we can measure not only whether the melody salience peak is the highest, but also whether it stands out from the other peaks of the salience function and by how much. As before, the average over all frames is denoted $\overline{S_3}$. The measures are summarised below:

- $\overline{\Delta f_{\mathrm{m}}}$: Mean melody frequency estimation error.
- $\overline{RR_{\mathrm{m}}}$: Mean reciprocal rank of the melody salience peak amongst all peaks of the salience function.
- $\overline{S_1}$: Mean salience ratio of the melody salience peak and the top peak in the salience function.
- $\overline{S_3}$: Mean salience ratio of the melody salience peak and the top 3 peaks of the salience function.

3.5 Results

The results are presented in two stages. First we present the results for the sinusoid extraction, and then the results for the salience function. In both stages, every measure is computed for each of the 12 possible analysis configurations summarised earlier in Table 3.1.

3.5.1 Sinusoid extraction results

We start by examining the results obtained when using all the evaluation material (i.e. all genres), provided in Table 3.3. The best result in each column is highlighted in bold. Recall that R_{p} and R_{e} should be maximised whilst $\overline{\Delta a_{\mathrm{dB}}}$, $\overline{\Delta f_{\mathrm{c}}}$ and $\overline{\Delta f_{\mathrm{w}}}$ should be minimised. We see that regardless of the filtering and transform used, both parabolic and phase based correction provide a statistically significant (N-way analysis of variance followed by a Tukey range test with $\alpha = 0.001$) improvement in frequency accuracy (i.e. lower $\overline{\Delta f_{\mathrm{c}}}$ values), with the phase based method providing just slightly better results. The benefit of using frequency correction is further accentuated when considering $\overline{\Delta f_{\mathrm{w}}}$. As expected, there is no significant difference between the amplitude error $\overline{\Delta a_{\mathrm{dB}}}$ when correction is applied and when it is not, as the error is dominated by the spectrum of the background.

Config.	R_{p}	R_{e}	$\overline{\Delta a_{\mathrm{dB}}}$	$\overline{\Delta f_{\mathrm{c}}}$	$\overline{\Delta f_{\mathrm{w}}}$
1	**0.62**	**0.88**	3.03	3.17	8.77
2	**0.62**	**0.88**	3.02	2.89	7.20
3	**0.62**	**0.88**	3.02	**2.88**	6.91
4	0.29	0.84	1.43	5.21	9.60
5	0.29	0.84	1.43	4.75	7.99
6	0.31	0.85	1.46	4.35	7.40
7	0.55	**0.88**	2.79	3.47	8.10
8	0.55	**0.88**	2.78	3.16	6.69
9	0.54	**0.88**	2.78	3.13	**6.45**
10	0.27	0.83	**1.41**	5.63	9.04
11	0.27	0.83	**1.41**	5.13	7.58
12	0.27	0.84	1.45	4.84	7.03

Table 3.3: Sinusoid extraction results for all genres.

When considering the difference between using the STFT and MRFFT, we first note that there is no improvement in frequency accuracy (i.e. smaller frequency error) when using the MRFFT (for all correction options), as indicated by both $\overline{\Delta f_{\mathrm{c}}}$ and $\overline{\Delta f_{\mathrm{w}}}$. This suggests that whilst the MRFFT might be advantageous for certain types of data (cf. results for opera in Table 3.4), when averaged over all genres the method does not provide a significant improvement in frequency accuracy. When we turn to examine the peak and energy recall, we see that the STFT analysis finds more melody peaks, however, interestingly both transforms obtain a similar degree of energy recall. This implies that the MRFFT, which generally finds less peaks (due to masking caused by wider peak lobes at higher frequencies), still finds the most important melody peaks. Whether this is significant or not for melody extraction should become clearer in the second part of the evaluation when examining the salience function results.

Next, we observe the effect of applying the equal loudness filter. We see that peak recall is significantly reduced, but that energy recall is maintained. This implies that the filter does not attenuate the most important melody peaks. If, in addition, the filter attenuates some background peaks, the overall effect would be that of enhancing the melody. As with the spectral transform, the significance of this step will become clearer when evaluating the salience function results.

Finally, we provide the results obtained for each genre separately in Table

Genre	Config.	R_p	R_e	$\overline{\Delta a_\mathrm{dB}}$	$\overline{\Delta f_\mathrm{c}}$	$\overline{\Delta f_\mathrm{w}}$
Opera	2	**0.73**	0.83	3.74	3.97	7.48
	6	0.59	**0.93**	1.15	**3.66**	6.50
	11	0.53	0.92	**1.08**	3.88	**5.91**
Jazz	3	**0.57**	**0.96**	2.20	**2.33**	6.23
	9	0.56	**0.96**	2.18	2.36	**5.75**
	10	0.20	0.84	**1.57**	7.88	10.95
Pop/Rock	2	**0.54**	**0.84**	3.08	**3.05**	7.71
	3	**0.54**	0.83	3.08	**3.05**	7.43
	9	0.46	**0.84**	2.89	3.37	**6.83**
	11	0.17	0.73	**1.86**	6.73	8.97
Bossa Nova	2	**0.76**	0.91	3.17	**1.95**	5.75
	8	0.56	**0.92**	2.74	2.32	5.48
	9	0.56	**0.92**	2.74	2.36	**5.30**
	10	0.29	0.86	**1.33**	4.19	8.00

Table 3.4: Sinusoid extraction results per genre.

3.4 (for conciseness only configurations which obtain the best result for at least one of the measures are included). We can see that the above observations hold for the individual genres as well. The only interesting difference is that for the opera genre the MRFFT gives slightly better overall results compared to the STFT. This can be explained by the greater pitch range and deep vibrato which often characterise the singing in this genre. The MRFFT's increased time resolution at higher frequencies means it is better at estimating the rapidly changing harmonics present in opera singing.

3.5.2 Salience function results

Analysis configuration

We start by examining the effect of the analysis configuration on the salience function. In Figure 3.3 we plot the results obtained for each measure using each of the 12 configurations (cf. Table 3.1). For comparability, the salience function is computed using the same (optimal) parameter values $\alpha = 0.8$, $\beta = 1$, $\gamma = 40$ dB and $N_h = 20$ for all analysis configurations (the optimal parameter values are discussed further down). Configurations that only differ in the filtering step are plotted side by side. The values of measures

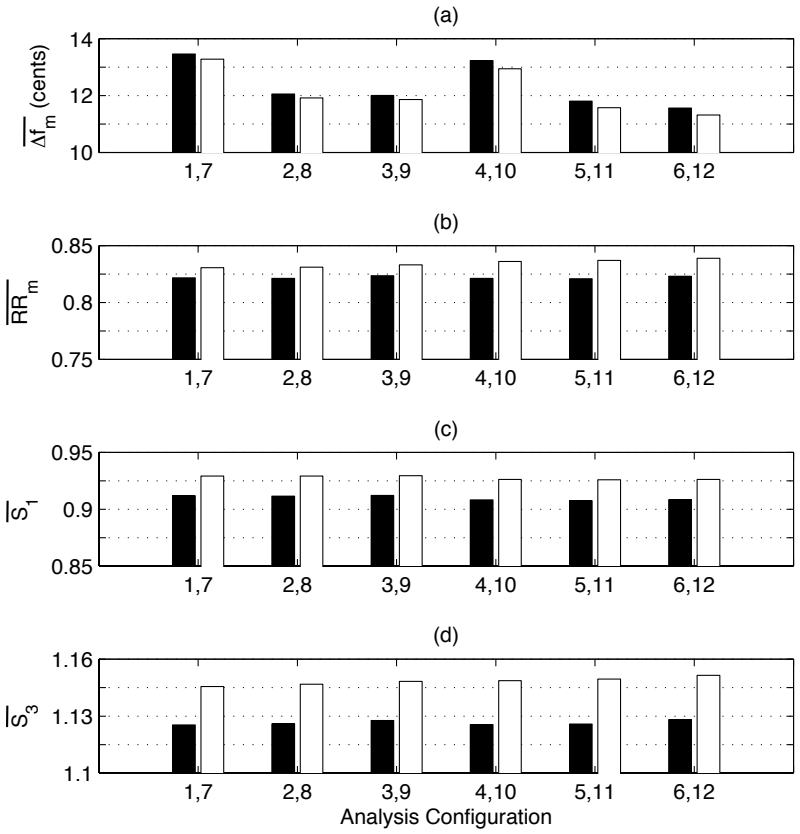

Figure 3.3: Salience function design, analysis configuration results. Each bar represents an analysis configuration, where white bars are configurations which apply equal loudness filtering. Recall that $\overline{\Delta f_{\mathrm{m}}}$ should be minimised whilst $\overline{RR_{\mathrm{m}}}$, $\overline{S_1}$ and $\overline{S_3}$ should be maximised.

$\overline{\Delta f_{\mathrm{m}}}$, $\overline{RR_{\mathrm{m}}}$, $\overline{S_1}$ and $\overline{S_3}$ are displayed in plots (a), (b), (c) and (d) of Figure 3.3 respectively.

The first thing we see is that for all measures, the results are always improved when equal loudness filtering is applied. This confirms our previous stipulation that the filter enhances the melody by attenuating non-melody spectral peaks. It can be explained by the filter's enhancement of the mid-band frequencies which is where the melody is usually present, and the attenuation of low-band frequencies where we expect to find low pitched instruments such as the bass.

Next, we examine the frequency error $\overline{\Delta f_{\mathrm{m}}}$ in plot (a) of Figure 3.3. We see that there is a (significant) decrease in the error when either of the two correction methods (parabolic interpolation or phase vocoder) are applied, as evident by comparing configurations 1, 7, 4, 10 (no correction) to the rest (where correction is applied). Though the error using phase based correction is slightly lower, the difference between the two correction methods was not statistically significant. Following these observations, we can conclude that both equal loudness filtering and frequency correction are beneficial for melody extraction.

Finally we consider the difference between the spectral transforms. Interestingly, the MRFFT now results in just a slightly lower frequency error than the STFT. Whilst it is not possible to determine the exact cause from the data, a possible explanation could be that whilst the overall frequency accuracy for melody spectral peaks is not improved by the MRFFT, the improved estimation at high frequencies is beneficial when we do the harmonic summation (the harmonics are better aligned). Another possible cause is the greater masking of spectral peaks, which could remove non-melody peaks interfering with the summation. When considering the remaining measures, the STFT gives slightly better results for $\overline{S_1}$, whilst there is no statistically significant difference between the transforms for $\overline{RR_{\mathrm{m}}}$ and $\overline{S_3}$. All in all, we see that using a multi-resolution transform provides only a marginal improvement (less than 0.5 cents) in terms of melody frequency accuracy, suggesting it might not necessarily provide significantly better results in a complete melody extraction system based on the salience function proposed here.

Salience function parameter configuration

As explained in section 3.3, in addition to the analysis configuration used, the salience function is influenced by four main parameters – the weighting parameters α and β, the energy threshold γ and the number of harmonics N_h. To find the best parameter combination for each analysis configuration and to study the interaction between the parameters, we performed a grid search of these four parameters using several representative values for each parameter: $\alpha = 1, 0.9, 0.8, 0.6$, $\beta = 1, 2$, $\gamma = \infty$, 60 dB, 40 dB, 20 dB, and $N_h = 4, 8, 12, 20$. This results in 128 possible parameter combinations which were used to compute the salience function measures for each of the 12 analysis configurations. Initially, we plotted a graph for each metric with a data point for each of the 128 parameter combinations, for the 12 analysis configurations (cf. Appendix C). At first glance it was evident that for all

analysis and parameter configurations the results were consistently better when $\beta = 1$, thus only the 64 parameter configurations in which $\beta = 1$ shall be considered henceforth. Also, in the previous section we saw that equal loudness filtering and frequency correction are important, whilst the type of correction and transform used do not affect the results significantly. Thus, in this section we will focus on configuration 9, which applies equal loudness filtering and uses the STFT transform with phase-based frequency correction[4].

In Figure 3.4 we plot the results obtained for the four measures using configuration 9 with each of the 64 possible parameter configurations ($\beta = 1$ in all cases) for the salience function. The first 16 data points represent configurations where $\alpha = 1$, the next 16 where $\alpha = 0.9$ and so on. Within each group of 16, the first 4 have $N_h = 4$, the next 4 have $N_h = 8$ etc. Finally within each group of 4, each data point has a different γ value from ∞ down to 20 dB.

We first examine the effect of the peak energy threshold γ, by comparing individual data points within every group of 4 (e.g. comparing peaks 1–4, 29–32 etc.). We see that (for all measures) there is no significant difference for the different values of the threshold except for when it is set to 20 dB for which the results degrade. That is, unless the filtering is too strict, filtering relatively weak spectral peaks seems to neither improve nor degrade the results (though it can speed up the computation since we process less peaks in the harmonic summation).

Next we examine the effect of N_h, by comparing different groups of 4 data points within every group of 16 (e.g. 17–20 vs 25–28). With the exception of the configurations where $\alpha = 1$ (1–16), for all other configurations all measures are improved the more harmonics we consider. As the melody in our evaluation material is primarily human voice (which tends to have many harmonic partials), this makes sense. We can explain the decrease for configurations 1–16 by the lack of harmonic weighting ($\alpha = 1$) which results in a great number of fake peaks with high salience at integer/sub-integer multiples of the true f_0.

Finally, we examine the effect of the harmonic weighting parameter α. Though it has a slight effect on the frequency error, we are primarily interested in its effect on melody salience as indicated by $\overline{RR_{\mathrm{m}}}$, $\overline{S_1}$ and $\overline{S_3}$. For all three metrics, no weighting (i.e. $\alpha = 1$) never produces the best

[4]Configurations 8, 11 and 12 result in similar graphs.

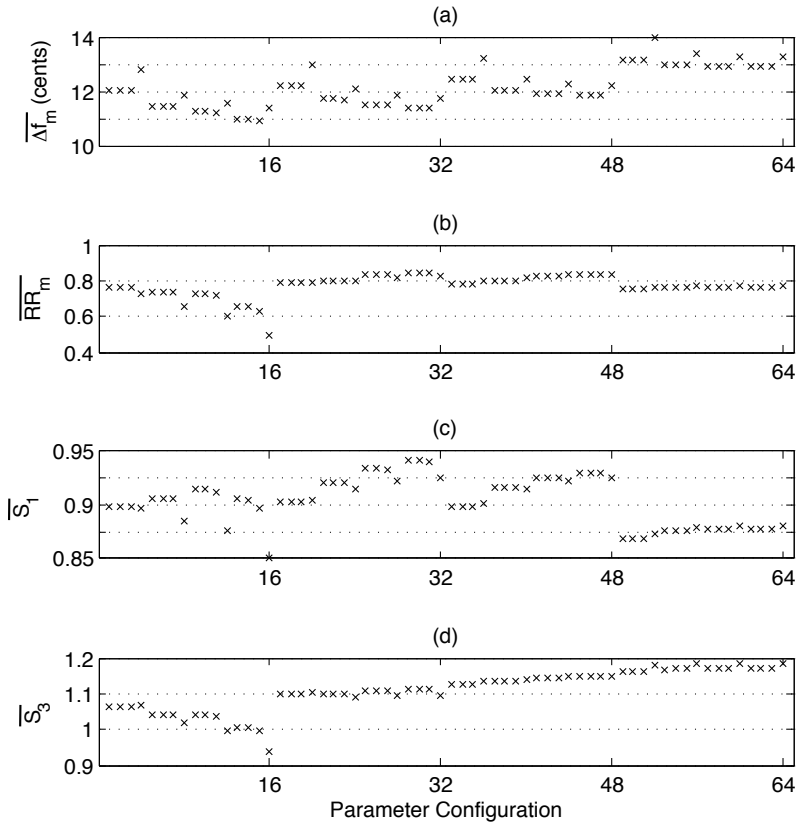

Figure 3.4: Salience function design, results by parameter configuration.

results. For $\overline{RR_{\mathrm{m}}}$ and $\overline{S_1}$ we get best performance when α is set to either 0.9 or 0.8. Interestingly, $\overline{S_3}$ increases continually as we decrease α. This implies that even with weighting, fake peaks at integer/sub-integer multiples (which are strongly affected by α) are present. This means that regardless of the configuration used, systems which use salience functions based on harmonic summation should include a post-processing step to detect and discard octave errors.

In Figure 3.5 we plot the measures as a function of the parameter configuration once more, this time for each genre separately (using analysis configuration 9). Interestingly, opera, jazz and bossa nova behave quite similarly to each other and to the overall results. For pop/rock however we generally get slightly lower results, and there is a greater sensitivity to the

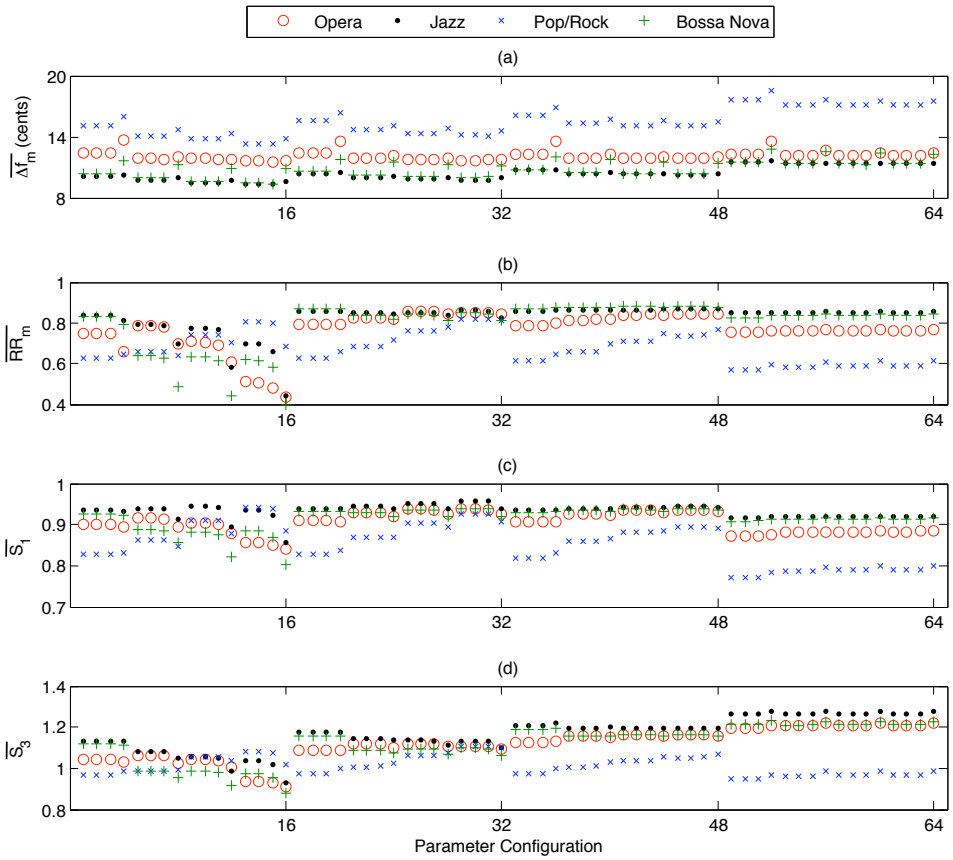

Figure 3.5: Per genre results by parameter configuration. Genres are labelled by their first letter – Opera, Jazz, Pop/Rock and Bossa Nova.

parameter values. This is most likely due to the fact that the accompaniment is more predominant in the excerpts we use for this genre, making it harder for the melody to stand out. This in turn results in a greater number of predominant peaks in the salience function that represent background instruments rather than octave errors of the melody. Consequently, $\overline{S_3}$ no longer favours the lowest harmonic weighting and, as with $\overline{RR_{\mathrm{m}}}$ and $\overline{S_1}$, is maximised when $\alpha = 0.8$ or 0.9. Following the above analysis, we can identify the combination of salience function parameters that gives the best overall results across all four metrics as $\alpha = 0.8$ or 0.9 (we use 0.8), $\beta = 1$, $N_h = 20$ and $\gamma \geq 40$ dB (we use $\gamma = 40$ dB to improve the efficiency of the harmonic summation computation).

3.6 Conclusion

In this chapter the first two blocks of our proposed melody extraction method were described and evaluated – sinusoid extraction and salience function. Different analysis techniques were compared for sinusoid extraction and it was shown that spectral peak estimation accuracy is improved when frequency/amplitude correction is applied. Two spectral transforms (single and multi-resolution) were compared and shown to perform similarly in terms of melody energy recall and frequency accuracy. A salience function based on harmonic summation was proposed alongside its key parameters. Different combinations of analysis techniques (i.e. analysis configurations) were evaluated in terms of the salience representation they produce. It was shown that equal loudness filtering and frequency correction both result in significant improvements in the salience representation, whilst the difference between the alternative frequency correction methods or the single/multi-resolution transforms is marginal. Henceforth, the analysis configuration used for the melody extraction method presented in this dissertation (unless mentioned otherwise) will be configuration 9 (equal loudness filtering + STFT + phase-based frequency/amplitude correction). Finally, the effect of the different parameters of the salience function on the resulting salience representation was studied, and an overall optimal parameter configuration for melody extraction using the proposed salience function was chosen ($\alpha = 0.8$, $\beta = 1$, $N_h = 20$ and $\gamma = 40$ dB).

Chapter 4

Melody Extraction by Contour Characterisation

4.1 Introduction

In this chapter we propose a novel salience-based melody extraction algorithm (Salamon & Gómez, 2012), which makes use of the salience function described in the previous chapter. The algorithm is centred on the creation and characterisation of *pitch contours* – time continuous sequences of f_0 candidates (salient pitches) that are grouped using heuristics inspired by auditory streaming cues such as harmonicity, pitch continuity and exclusive allocation (Bregman, 1990) . We define a set of musical features which are automatically computed for each contour. By studying the feature distributions of melodic and non-melodic contours we are able to define rules for distinguishing between the contours that form the melody and contours that should be filtered out. Combining these rules with voice leading principles (Huron, 2001), novel techniques are developed for addressing the challenges discussed in Chapter 2 – voicing detection, avoiding octave errors and distinguishing the pitch contours that belong to the melody from those of the accompaniment.

The idea of f_0 candidate grouping (or tracking) was previously introduced in studies such as the ones by Cancela (2008) and Dressler (2009). ASA inspired grouping principles have been employed in melody extraction systems based on source separation (Lagrange et al., 2008), as well as in the work of Paiva et al. (2006) where pitch contours are first segmented into notes out of which the melody is selected. Whilst the structure of our algorithm is some-

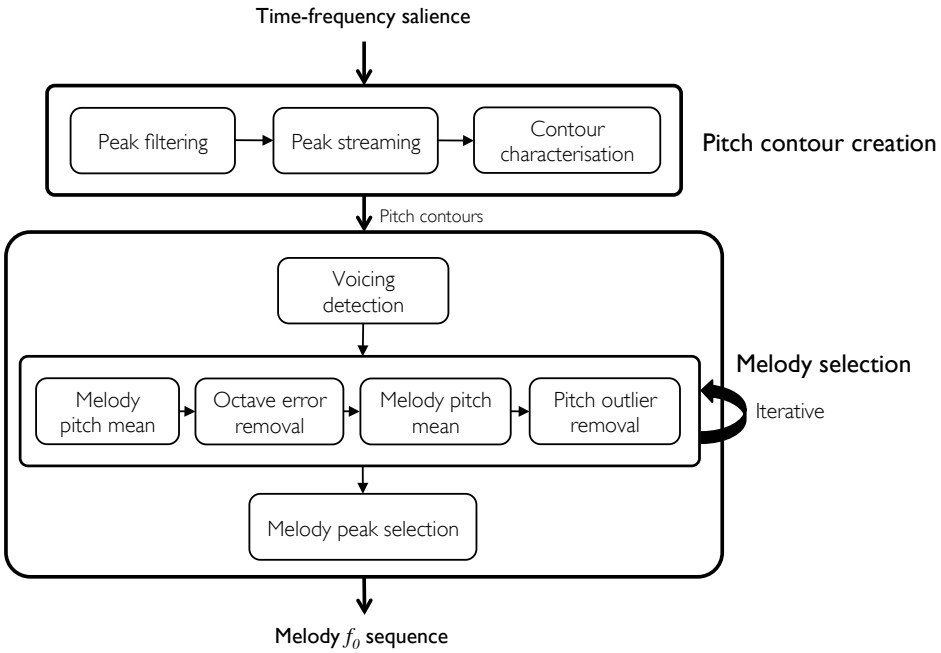

Figure 4.1: Final two blocks of the system: pitch contour creation and melody selection.

what similar to the one proposed by Paiva et al. (2006), the method differs in several important ways. To begin with, a wider set of contour characteristics beyond the basic pitch height, length and mean salience is considered. The method does not require segmentation into notes, and makes use of contour features that would be lost during pitch quantisation such as vibrato and pitch deviation. Furthermore, these features are exploited using new techniques following the study of contour feature distributions.

The algorithm is described in section 4.2. A method for the generation and characterisation of pitch contours is proposed, and a set of pitch contour features is defined. The distributions of the contour features are studied, leading to novel heuristics for voicing detection, octave error minimisation and melody selection. A block diagram of the steps described in Section 4.2 (which correspond to the final two blocks of the complete system depicted in Figure 1.6) is provided in Figure 4.1.

In Section 4.3 we propose a modification of the melody extraction algorithm described in Section 4.2, in which we replace the heuristic-based melody

selection block with a selection method based on a statistical model of pitch-contour features (Salamon et al., 2012b). Through the use of a statistical model we explore the possibility of exploiting contour features in a more automated manner – the key advantage of this approach being that the model can be easily updated whenever we want to incorporate new features into the algorithm.

Two versions of the model-free algorithm (Section 4.2) were submitted to the MIREX international evaluation campaign, in 2010 and 2011. In section 4.4 we provide and discuss the results of these comparative evaluations. In addition to the MIREX evaluations we evaluate specific aspects of the the algorithm, including a qualitative error analysis, a study of the influence of different algorithmic components on performance and a glass ceiling analysis to help determine the current limitations of the approach and propose directions for future work. Finally, we evaluate and contrast the algorithm with the model-based version proposed in Section 4.3.

4.2 Melody extraction using contour characteristics

4.2.1 Creating pitch contours (peak streaming)

Given the output of the salience function, its maxima (peaks) at each frame are taken as candidates for the melody f_0 (recall that peaks of the salience function represent salient pitches). At this point, some melody extraction methods attempt to track the melody directly from the set of available peaks (Goto, 2004; Rao & Rao, 2010). Our approach however is based on the idea that further information (which can be exploited to select the correct melody pitch) can be extracted from the data by first grouping the peaks into pitch contours – time and pitch continuous sequences of salience peaks. Each contour has a limited time span corresponding roughly to a single note in the shortest case or a short phrase in the longest. Though f_0 grouping, or *peak streaming*, for melody extraction has been proposed previously (Cancela, 2008; Paiva et al., 2006), in this work the characterisation of pitch contours is explored in new ways, resulting in original solutions to the challenges mentioned in Section 1.3.

Before the streaming process is carried out, we first filter out non-salient peaks to minimise the creation of "noise" contours (non-melody contours). The filtering process is carried out in two stages: first, peaks are filtered on a per-frame basis by comparing their salience to that of the highest peak in

the current frame. Peaks below a threshold factor τ_+ of the salience of the highest peak are filtered out. In the second stage the salience mean $\mu_{\mathcal{S}}$ and standard deviation $\sigma_{\mathcal{S}}$ of all remaining peaks (in all frames) are computed. Peaks with salience below $\mu_{\mathcal{S}} - \tau_\sigma \cdot \sigma_{\mathcal{S}}$ are then filtered out, where τ_σ is a factor that determines the degree of deviation below the mean salience accepted by the filter. The first filter ensures we only focus on the most predominant pitch candidates at each frame, whilst the second, a precursor to our voicing detection method, removes peaks in segments of the song which are generally weaker (and hence more likely to be unvoiced). This filtering has an inherent trade-off – the more peaks we filter out the fewer noise contours will be created (thus improving the detection of unvoiced sections of the song and the correct selection of melody contours), however we also increase the risk of filtering out salience peaks which belong to the melody (henceforth *melody peaks*). The selection of optimal values for τ_+ and τ_σ is discussed further down.

The remaining peaks are stored in the set S^+, whilst the peaks that were filtered out are stored in S^-. The peaks are then grouped into contours in a simple process using heuristics inspired by auditory streaming cues (Bregman, 1990). We start by selecting the highest peak in S^+ and add it to a new pitch contour. We then track forward in time by searching S^+ for a salience peak located at the following time frame (time continuity cue) which is within 80 cents (pitch continuity cue) of the previously found peak. A matching peak is added to the pitch contour and removed from S^+ (exclusive allocation principle, i.e. a single piece of auditory input can contribute to the mental description of only one sound at any given moment). This step is repeated until no further matching salience peaks are found. During the tracking we must ensure that short time gaps in the pitch trajectory do not split what should be a single contour into several contours. To do so, once no matching peak is found in S^+, we allow the tracking to continue for a limited amount of frames using peaks from S^-. The underlying assumption is that melody peaks whose salience is temporarily masked by other sources will be stored in S^-, and tracking them allows us to stay on the correct trajectory until we find a peak in S^+. If the gap length exceeds 100 ms (the selection of thresholds and parameter values is discussed further down) before a peak from S^+ is found the tracking is ceased. We then go back to the first peak of the contour and repeat the tracking process backwards in time. Once the tracking is complete we save the contour and the entire process is repeated until there are no peaks remaining in S^+.

To select the best parameters for the contour creation (τ_+, τ_σ, the maximum

allowed pitch distance between consecutive peaks and the maximum allowed gap duration during tracking), we compared contours generated from different excerpts to the excerpts' melody ground truth and evaluated them in terms of pitch accuracy (distance in cents between the ground truth and the contours) and voicing (i.e. whether the contours overlap perfectly with the ground truth or are otherwise too long or too short). This process was repeated in a grid search until the parameters which resulted in the most accurate peak tracking were found ($\tau_+ = 0.9$, $\tau_\sigma = 0.9$, maximum pitch distance = 80 cents and maximum gap duration = 100 ms). For τ_+ and τ_σ we also measured the amount of melody peaks (and non-melody peaks) before and after the filtering. This analysis revealed that as τ_+ is increased the number of non-melody salience peaks drops dramatically, whilst the number of melody peaks reduces very gradually. Using the selected parameter values the number of non-melody peaks is reduced by 95% whilst melody peaks are reduced by less than 17% (and this loss can be recovered by the gap tracking). The result is that the percentage of melody peaks out of the total number of peaks goes up on average from 3% initially to 52% after filtering. The quality of contour formation is discussed in Section 4.4.6.

4.2.2 Pitch contour characterisation

Once the contours are created, the remaining challenge is that of determining which contours belong to the melody. To do so, we define a set of contour characteristics which will be used to guide the system in selecting melody contours. Similarly to other systems, we define features based on contour pitch, length and salience. However, by avoiding the quantisation of contours into notes (a step applied by Paiva et al., 2006) we are able to extend this set by introducing features extracted from the pitch trajectory of the contour, namely its pitch deviation and the presence of vibrato. Note that whilst Paiva et al. (2006) also keep a non-quantised version of each contour for use at a later stage of their algorithm, they do not exploit it to compute additional contour features. Furthermore, as shall be seen in Section 4.2.3, we use not only the feature values directly but also their distributions.

Every pitch contour is represented by two discrete series $p(n)$ and $s(n)$, $n = 1 \ldots N_c$. The former contains the frequency (in cents) of every pitch value (salience peak) in the contour, and the latter its corresponding salience value. The time difference between consecutive values in both series, which is determined by the hop size H used for the spectral analysis (cf. Section

3.2.2), is 2.9 ms ($H = 128$ for a sampling rate of $f_S = 44.1$ kHz), and the frequency resolution of the values in $p(n)$ is determined by the resolution of the salience function, in our case 10 cents. Then, for every pitch contour we define and compute the following characteristics:

- **Pitch mean** $C_{\bar{p}} = \frac{1}{N_c} \sum_{n=1}^{N_c} p(n)$: the mean pitch height of the contour.
- **Pitch deviation** $C_{\sigma_p} = \sqrt{\frac{1}{N_c} \sum_{n=1}^{N_c} (p(n) - C_{\bar{p}})^2}$: the standard deviation of the contour pitch.
- **Total salience** $C_{\Sigma s} = \sum_{n=1}^{N_c} s(n)$: the sum of the salience of all peaks comprising the contour.
- **Mean salience** $C_{\bar{s}} = \frac{1}{N_c} C_{\Sigma s}$: the mean salience of all peaks comprising the contour.
- **Salience deviation** $C_{\sigma_s} = \sqrt{\frac{1}{N_c} \sum_{n=1}^{N_c} (s(n) - C_{\bar{s}})^2}$: the standard deviation of the salience of all peaks comprising the contour.
- **Length** $C_l = N_c \cdot \frac{H}{f_S}$: the length (duration) of the contour in seconds, where H and f_S are the hop size (128) and sampling frequency (44100) used for the spectral analysis respectively.
- **Vibrato presence** C_v: whether the contour has vibrato or not (true/false). Vibrato is automatically detected by the algorithm using a method based on Herrera & Bonada (1998): we apply the FFT to the contour's pitch sequence $p(n)$ (after subtracting the mean) and check for a prominent peak in the expected frequency range for human vibrato (5–8 Hz; Sundberg, 1995).

In Figure 4.2 we provide examples of contours created for excerpts of different musical genres, where the contours which form the melody are highlighted in bold. By observing these graphs we can propose contour characteristics that differentiate the melody from the rest of the contours: vibrato, greater pitch variance (in the case of human voice), longer contours, a mid-frequency pitch range and (though not directly visible in the graphs) greater salience. These observations concur with voice leading rules derived from perceptual principles (Huron, 2001). To confirm our observations, we computed the feature distributions for melody and non-melody contours using the representative dataset described further down in Section 4.4.1. Note

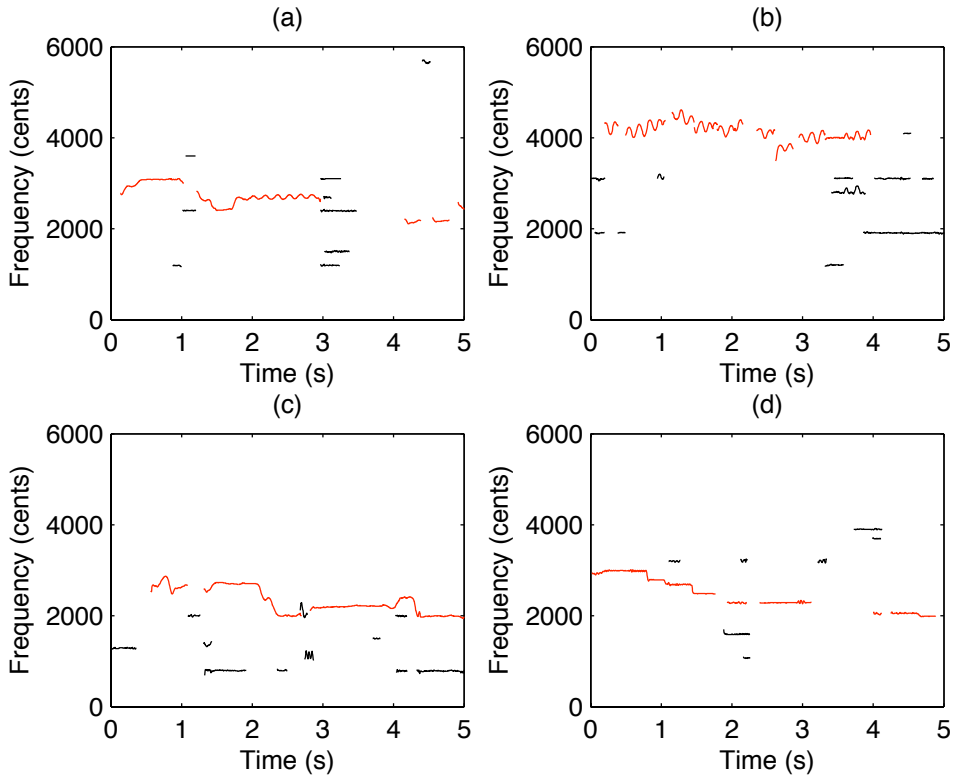

Figure 4.2: Pitch contours generated from excerpts of (a) vocal jazz, (b) opera, (c) pop and (d) instrumental jazz. Melody contours are highlighted in bold.

that in most (but not all) of the excerpts in this dataset the melody is sung by a human voice. The resulting distributions are provided in Figure 4.3, where for each feature we plot the distribution for melody contours (solid red line) and non-melody contours (dashed blue line). In plots (c), (d) and (e) the feature values are normalised by the mean feature value for each excerpt. We see that the above observations are indeed evident in the feature distributions. Additionally, for vibrato presence we found that 95% of all contours in which vibrato was detected were melody contours.

In the Section 4.2.3 we explain how we take advantage of these distributions to derive a set of heuristics for selecting the pitch contours that belong to the melody. A possible concern is that an accompanying instrument could display a certain characteristic (e.g. pitch height) which fits one of the melodic contour feature distributions. However, as shall be seen further

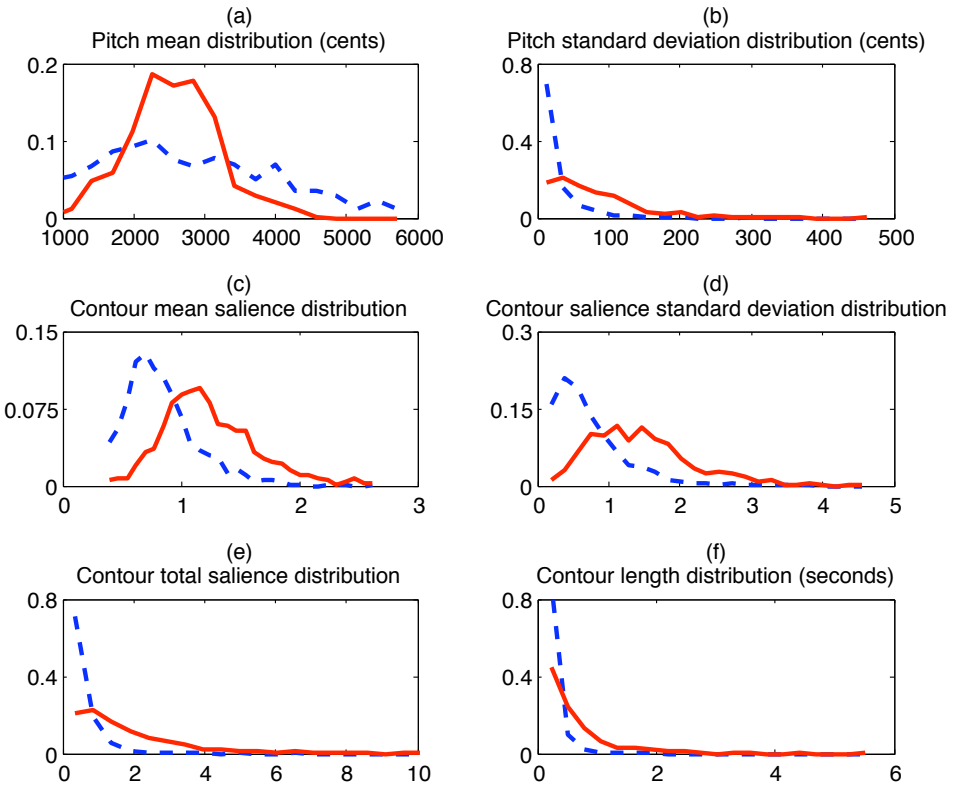

Figure 4.3: Pitch contour feature distributions: (a) Pitch mean, (b) Pitch std. dev., (c) Mean salience, (d) Salience std. dev., (e) Total salience, (f) Length. The red solid line represents the distribution of melody contour features, the blue dashed line represents the distribution of non-melody contour features.

down, by considering this expanded set of contour characteristics we can avoid selecting contours of accompanying instruments even if they exhibit a certain melodic characteristic. For example, a contour produced by an accompanying violin with vibrato may still be discarded due to its pitch height. Finally, we note that basing our algorithm on pitch contours gives us the possibility of introducing new contour features in the future. Furthermore, these features could be exploited for other MIR tasks such as genre classification (Salamon et al., 2012c; cf. Chapter 6) or singing style characterisation (Kroher, 2013).

4.2.3 Melody selection

We now turn to describe how the melody is chosen out of all the contours created in the previous step of the algorithm. Rather than attempt to pick out the contours that belong to the melody, we pose this task as a contour filtering problem, where our goal is to filter out all non-melody contours. As seen in the block diagram in Figure 4.1, this process is comprised of three steps: (1) voicing detection, (2) octave error minimisation and pitch outlier removal, and (3) final melody selection. In the first two steps the contour characteristics are used to filter out non-melody contours, and in the final step the melody frequency at each frame is selected out of the remaining contours.

Voicing detection

Voicing detection is the task of determining when the melody is present and when it is not. For example in plot (a) of Figure 4.2 the melody is present between seconds 0–3 and 4–5, but not between seconds 3–4 where non-melody contours are found (in fact they are the notes of a piano chord). To filter out these contours we take advantage of the contour mean salience distribution provided in plot (c) of Figure 4.3. Even though the mean salience distributions for melody and non-melody contours are not perfectly separated, we see that by setting a threshold slightly below the average contour mean salience of all contours in the excerpt $\overline{C_{\bar{s}}}$, we can filter out a considerable amount of non-melody contours with little effect on melody contours. We define the following voicing threshold τ_v based on the distribution mean $\overline{C_{\bar{s}}}$ and its standard deviation $\sigma_{C_{\bar{s}}}$:

$$\tau_v = \overline{C_{\bar{s}}} - \nu \cdot \sigma_{C_{\bar{s}}}. \tag{4.1}$$

The parameter ν determines the lenience of the filtering – a high ν value will give more false positives (i.e. false melody contours) and low value more false negatives (i.e. filter out melody contours). The sensitivity of the system to the value of ν is discussed in Section 4.4.4. When developing the voicing filter we also considered using the contour total salience $C_{\Sigma s}$ instead of the mean salience $C_{\bar{s}}$ in the equation above, but the latter was found to give better results. This is likely due to the bias of the contour total salience towards longer contours, which is not beneficial at this stage as we risk removing short melody contours. At a later stage length will be exploited to guide the system when a choice must be made between alternative concurrent contours.

In the previous section we also noted that if the system detected vibrato in a contour, it was almost certainly a melody contour. Furthermore, in plot (b) of Figure 4.3 we see that there is a sudden drop in non-melody contours once the pitch deviation goes above 20 cents, and once the deviation is greater than 40 cents the probability of a contour being a non-melody contour is less than 5%. We use this information to tune our voicing filter, by giving greater "immunity" to contours where vibrato was detected ($C_v = true$) or whose pitch deviation is above 40 cents ($C_{\sigma_p} > 40$). This is accomplished by weighting the salience mean $C_{\bar{s}}$ of a contour where $C_v = true$ or $C_{\sigma_p} > 40$ by a positive factor τ_{vib} or τ_{dev} respectively before comparing it to τ_v. In this way we ensure that contours which have relatively low salience but strong melodic characteristics are not filtered out at this stage. The values of τ_{vib} and τ_{dev} were empirically set to 3 and 1.5 respectively, though setting them to higher values (effectively completely preventing such contours from being filtered at this stage) produces similar results.

Octave errors and pitch outliers

One of the main sources of errors in melody extraction algorithms is the selection of a harmonic multiple/sub-multiple of the correct melody f_0 instead of the true f_0, commonly referred to as octave errors. Various approaches have been proposed for the minimisation of octave errors, usually performed directly after the calculation of the salience function and on a per-frame basis (e.g. Cancela, 2008; Klapuri, 2009; cf. Section 2.3.1). When we consider a single frame in isolation, determining whether two salience peaks at a distance of one octave from each other represent two separate sources or whether they are both the result of the same source (with one peak being a multiple of the other) can prove a difficult task. On the other hand, once we have tracked the pitch contours, detecting the presence of octave duplicates becomes a relatively straight forward task, as these manifest themselves as contours with practically identical trajectories at a distance of one octave from each other. In practice, to compare contour trajectories we compute the distance between their pitch values on a per-frame basis for the region in which they overlap, and compute the mean distance over this region. If the mean distance is within 1200 ± 50 cents (i.e. 1 octave $\pm$ half a semitone), the contours are considered octave duplicates. An example of octave duplicates can be observed in plot (b) of Figure 4.2 between seconds 3–4 where the correct contour is at approximately 4000 cents and the duplicate at about 2800 cents.

We propose a method for octave error minimisation that takes advantage

of temporal information in two ways. First, as mentioned above, we use the temporal grouping of salience peaks into pitch contours in order to detect octave duplicates by comparing contour trajectories. Second, we use the relationship between neighbouring contours (in time) to decide which of the duplicates is the correct one. Our approach is based on two assumptions: first, that most (though not all) of the time the correct contour will have greater salience than its duplicate (the salience function parameters were optimised to this end). Second, that melodies tend to have a continuous pitch trajectory avoiding large jumps, in accordance with voice leading principles (Huron, 2001).

The first assumption is implemented in a fairly straight forward manner: if one of the duplicates is considerably weaker than the other (defined as having less than half the total salience), it is highly likely that it is a "ghost contour" (an artefact of the salience function), and so we remove it and keep the contour with greater salience. When both contours have similar salience values, we consider the principle of pitch continuity. To take advantage of the pitch continuity which often characterises melodies, we iteratively calculate a *melody pitch mean* $\mathcal{P}(n)$ – a pitch trajectory that represents the overall trend of the melody's pitch height over time. When octave duplicates are encountered, the assumption is that the contours directly before and after the duplicates will pull $\mathcal{P}(n)$ towards the contour in the correct octave, since it would result in a more continuous melody. Thus, the contour closest to $\mathcal{P}(n)$ is selected as the correct contour and the other is discarded. Whilst this rule was found to work well in many cases, one exception had to be considered: if both contours are relatively far from $\mathcal{P}(n)$ (i.e. roughly half an octave or more), they both represent a large jump in the melody (e.g. one could represent a large jump up whilst the other a large jump down), in which case proximity to $\mathcal{P}(n)$ might not be a good indicator of which contour is the right one to keep. In such cases, we go back to the first assumption, and keep the contour with greater total salience $C_{\Sigma s}$.

We also exploit $\mathcal{P}(n)$ to remove *pitch outliers* – contours that are more than one octave above or below the pitch mean $\mathcal{P}(n)$. Filtering outliers ensures there are no large jumps in the melody (continuity assumption), and may also filter out contours at unvoiced sections of the piece that were not captured by the voicing detection step. The distance between a contour and $\mathcal{P}(n)$ is computed in the same way the distance between two contours is computed: by averaging the per-frame distances between them. The complete process can be summarised as follows:

1. Calculate $\mathcal{P}(n)$ at each frame as the weighted mean of the pitch values of all contours present in the frame.
2. Smooth $\mathcal{P}(n)$ using a 5-second sliding mean filter (the length was determined empirically) with a hop size of 1 frame. This limits the rate at which the melody pitch trajectory can change, thus ensuring continuity and avoiding large jumps.
3. Detect pairs of octave duplicates and for each pair remove the contour furthest from $\mathcal{P}(n)$, unless it is considerably more salient in which case remove the weaker contour.
4. Recompute $\mathcal{P}(n)$ using the remaining contours, following Steps 1-2.
5. Remove pitch outliers by deleting contours at a distance of more than one octave from $\mathcal{P}(n)$.
6. Recompute $\mathcal{P}(n)$ using the remaining contours, following Steps 1-2.
7. Repeat Steps 3-6 twice more, each time starting with all contours that passed the voicing detection stage, but using the most recently computed melody pitch mean $\mathcal{P}(n)$. The number of iterations was chosen following experimentation that suggested that this was sufficient for obtaining a good approximation of the true trajectory of the melody. In the future the fixed iteration number could be replaced with a stabilisation criterion.
8. Pass the contours remaining after the last iteration to the final melody selection stage.

It was found that the mean pitch trend $\mathcal{P}(n)$ computed in Step 1 most closely approximates the true trajectory of the melody when each contour's contribution is weighted by its total salience $C_{\Sigma s}$. This biases the mean towards contours which are salient for a longer period of time, which is desirable since such contours are more likely to belong to the melody, as evident from the distributions in Figure 4.3 (e) and (f). Furthermore, in plot (a) of the same figure we see that the majority of melodic pitch contours lie between 2000 and 3000 cents (relative to 55 Hz, i.e. between 175–311 Hz), and that there are virtually no contours whose pitch mean lies below 1000 cents (98 Hz). In order to incorporate this information, we bias the calculation of $\mathcal{P}(n)$ against low-pitched contours (below 98 Hz) by further weighting them by a penalty factor proportional to their mean pitch height

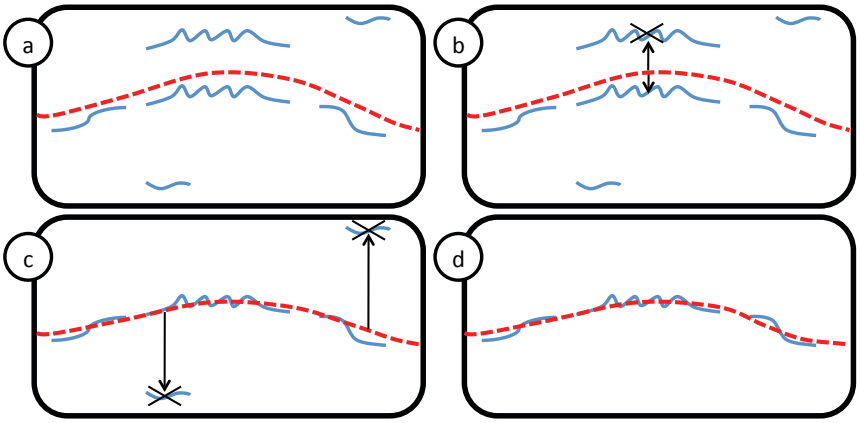

Figure 4.4: Removing octave duplicates and pitch outliers. (a) Steps 1-2: the initial smoothed melody mean pitch trend $\mathcal{P}(n)$ is computed (dashed red line). (b) Step 3: an octave duplicate is detected and removed. (c) Steps 4-5: $\mathcal{P}(n)$ is recomputed and two pitch outliers are removed. (d) Step 6: $\mathcal{P}(n)$ is recomputed.

(i.e. the contribution of a contour with mean frequency $f_c < 1000$ cents to $\mathcal{P}(n)$ is further weighted by $0.3f_c/1000$). Even without the empirical evidence provided in plot (a) of Figure 4.3, one could argue that this is a fairly intuitive step, and indeed penalising low-frequency pitch content is incorporated (using different techniques) in many melody extraction algorithms (e.g. Cancela, 2008; Dressler, 2011a; Goto, 2004).

An illustration of running steps 1-6 is provided in Figure 4.4. In plot (a) we start with a set of contours, together with the smoothed mean pitch trend $\mathcal{P}(n)$ (Steps 1-2) represented by the dashed red line. In the next plot (b), octave duplicates are detected, and the duplicate farther from the mean trend $\mathcal{P}(n)$ is removed (Step 3). Next (c) the mean trend $\mathcal{P}(n)$ is recomputed (Step 4), and pitch outliers are detected and removed (Step 5). Finally $\mathcal{P}(n)$ is recomputed once more (Step 6), displayed in plot (d) together with the remaining contours.

Final melody selection

In the final step of the algorithm we need to select the peaks that belong to the melody out of the remaining contours (recall that each peak represents an f_0 candidate). Whilst in other approaches this step often involves fairly complicated peak tracking using streaming rules or note transition models, in our algorithm these considerations have already been taken into account

by the contour creation, characterisation and filtering process. This means that often there will only be one peak left to choose. When there is still more than one contour present in a frame, the melody is selected as the peak belonging to the contour with the highest total salience $C_{\Sigma s}$. If no contour is present the frame is regarded as unvoiced.

Since it is possible that the voicing detection step will have erroneously removed contours from sections of the song where the melody is in fact present, we would also like to provide pitch estimates for frames the algorithm has estimated as unvoiced (in this way enabling us to evaluate the algorithm's raw pitch and chroma accuracies independently of its voicing recall and false alarm rates, cf. Section 2.4.1). To do so we go back to the original set of pitch contours (before any contour filtering was applied) and for every unvoiced frame we select the pitch belonging to the most salient contour present in that frame as the melody. To avoid large discontinuities in the final output, the pitch values selected for the unvoiced frames are octave shifted to be as close to $\mathcal{P}(n)$ (which was computed previously) as possible.

In Figure 4.5 we provide an example of the complete melody selection process for the excerpt previously featured in plot (a) of Figure 4.2. In plot (a) of Figure 4.5 we show all the contours created from the polyphonic recording (in blue), in plot (b) the remaining contours after filtering (in blue) with the final melody mean pitch trend $\mathcal{P}(n)$ indicated by the thick red line, and finally in plot (c) the final estimated melody sequence (thin red line) on top of the ground truth (thick black line). Further examples of melodies extracted using the method described in this section are provided in Figures 4.6 and 4.7. In each plot in these figures we display the melody extracted from a different audio excerpt (thin red line) on top of the ground truth for that excerpt (thick black line). The original audio recordings from which the melody pitch sequences in Figures 4.5, 4.6 and 4.7 were extracted can be listened to on the thesis companion website[1]. The website also includes synthesised versions of the extracted pitch sequences so that they can be listened to in comparison with the original recording.

[1] http://www.justinsalamon.com/phd-thesis

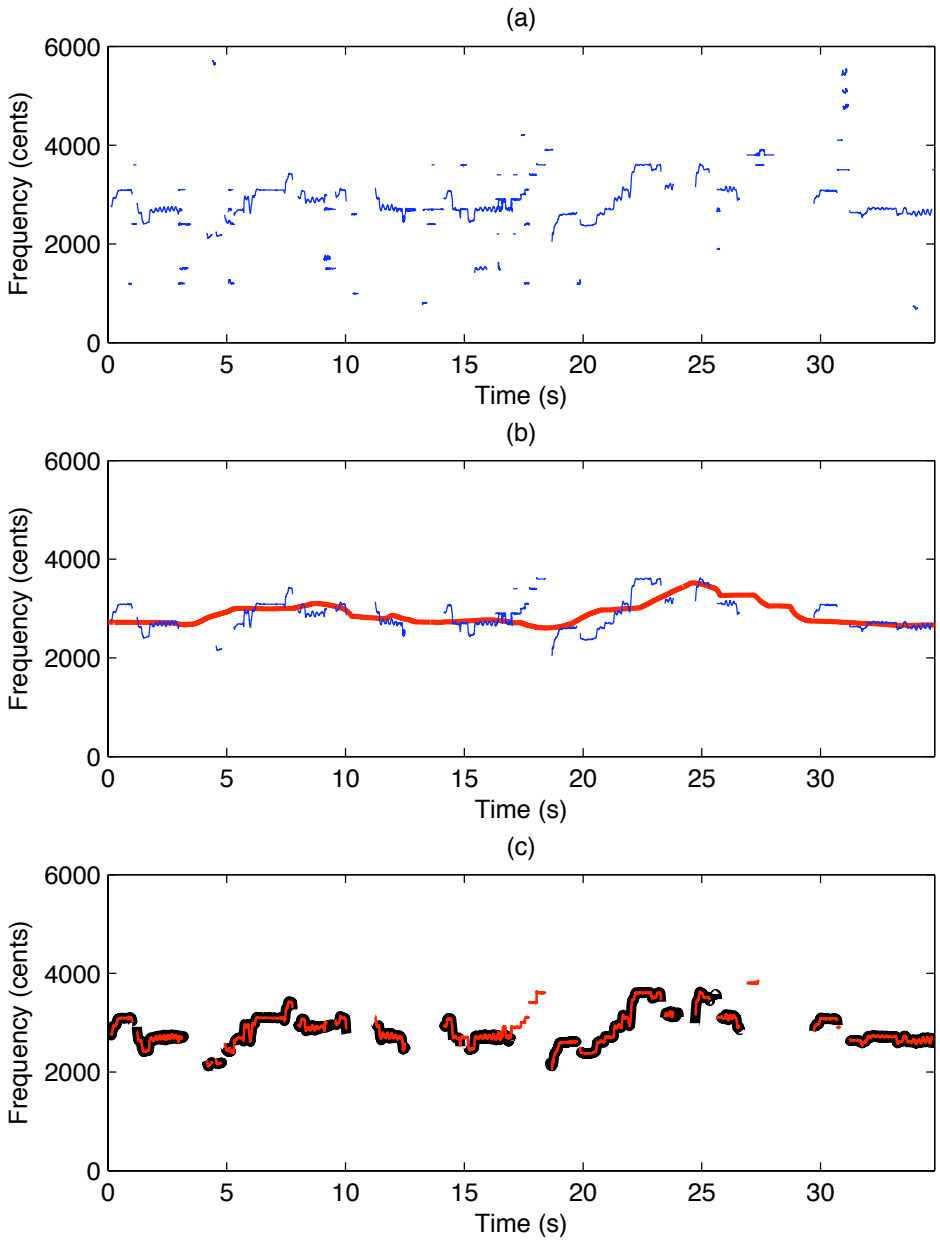

Figure 4.5: Melody selection for an excerpt of vocal jazz: (a) all pitch contours created from the polyphonic recording (in blue), (b) the remaining contours after filtering (in blue) and the melody mean pitch trend (thick red line), (c) final extracted melody (thin red line) on top of the ground truth (thick black line).

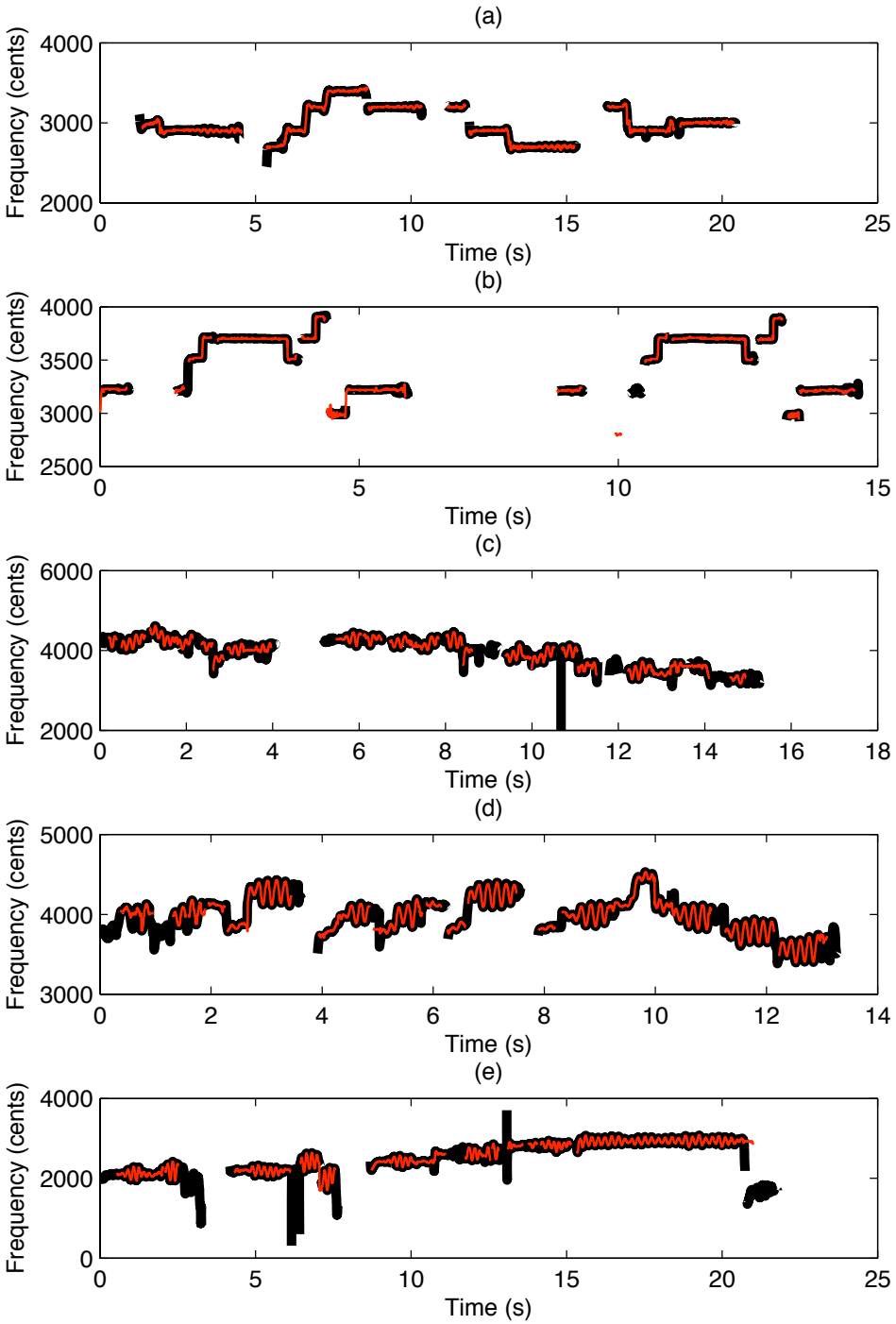

Figure 4.6: Examples of melodies extracted using the proposed model-free algorithm. The ground truth is indicated by the thick black line, and the extracted melody by the thin red line.

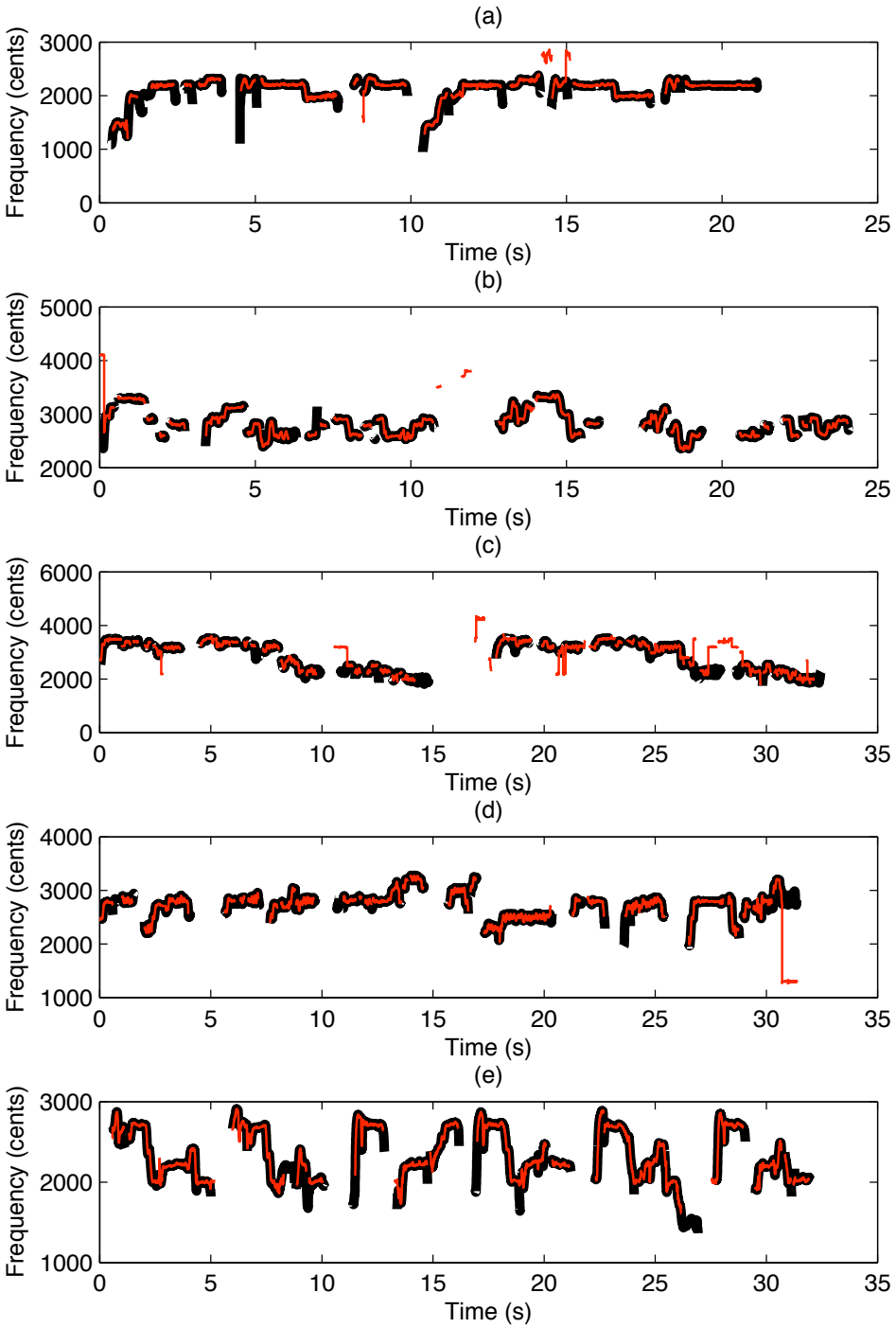

Figure 4.7: More examples of melodies extracted using the proposed model-free algorithm. The ground truth is indicated by the thick black line, and the extracted melody by the thin red line.

4.3 Statistical modelling of melodic pitch contours

4.3.1 Introduction

Similar to other extraction algorithms such as the ones proposed by Goto (2004) and Dressler (2011a), one characteristic of our approach is that it relies on heuristics for the melody selection stage. Whilst this in itself is not a problem (some of the most successful algorithms also rely on heuristics, e.g. Dressler, 2011a, and Salamon & Gómez, 2012), it has the disadvantage that new rules must be devised whenever we want to incorporate new musical information into the algorithm. This motivates us to explore the possibility of exploiting contour features in an automated manner, that is, creating a model based on contour features that can be easily updated whenever we want to incorporate new features.

In this section we present a method for the statistical characterisation of pitch contours using the contour feature distributions presented in the previous section. We do this by combining the distributions of the different contour features into a single multivariate Gaussian distribution which embodies most of the features currently used by the algorithm. By learning separate feature distributions for melodic and non-melodic contours, we are able to create two different multivariate distributions, one for computing the likelihood that a contour is melodic, and the other for computing the likelihood that a contour is not melodic (i.e. accompaniment). The likelihoods are used to compute a single "melodiness" index, which is then used to select the final melody sequence. This selection process replaces the melody selection block described in Section 4.2.3. As can be inferred from the above description, the proposed method is flexible in that new features can be easily incorporated into the model without the need to manually devise rules to exploit them.

4.3.2 Statistical modelling

Our goal is to define a statistical model that encompasses all of the contour feature distributions provided in Figure 4.3. To do so, we represent all feature distributions as two multivariate normal distributions, one for melodic contour features and one for non-melodic contour features. Peeters (2003) showed that using a multivariate Gaussian results in comparable classification performance to using a Gaussian Mixture Model (GMM) when the amount of training data is relatively small.

As seen in the plots of Figure 4.3, though some distributions (in particular the distribution of pitch height for melodic contours) appear to be normally distributed, this is not the case for all distributions. Thus, in the first step of the modelling we apply a power transform to obtain a normal-like distribution for each contour feature. Specifically, we apply the Box-Cox transform (Box & Cox, 1964), which for a variable Z with data samples $z_i > 0$ is defined as:

$$z_i^{(\lambda)} = \begin{cases} \frac{(z_i^{\lambda}-1)}{\lambda}, & \text{if } \lambda \neq 0, \\ \log(z_i), & \text{if } \lambda = 0, \end{cases} \tag{4.2}$$

where the power parameter λ is selected such that it maximises the log-likelihood of λ given the transformed data, which is assumed to be normally distributed. An example of the distributions of the contour total salience feature $C_{\Sigma s}$ before an after transformation is provided in Figure 4.8. In plots (a) and (b) we show the feature distribution for melodic contours before and after transformation respectively, and in plots (c) and (d) we plot the corresponding distributions for non-melodic contours. In plots (b) and (d) we also display the normal distribution that best fits the transformed data.

The mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ (of size $N_F \times N_F$ where N_F is the number of features used) of the transformed data are obtained using the standard maximum likelihood estimators, allowing us to construct a multivariate normal distribution with parameters $\theta = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ of the form:

$$\mathcal{N}_\theta(\mathbf{x}) = \frac{1}{(2\pi)^{N_F/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right). \tag{4.3}$$

This procedure is repeated twice, once for the melodic contour feature distributions and once for the non-melodic (i.e. background) contour feature distributions, resulting in two multivariate normal distributions which we denote $\mathcal{N}_{\theta m}$ and $\mathcal{N}_{\theta bg}$ respectively. Given the feature vector $\mathbf{x}$ of a pitch contour, we can now use $\mathcal{N}_{\theta m}$ and $\mathcal{N}_{\theta bg}$ to compute the likelihood of the contour being a melodic contour and the likelihood of it being a non-melodic contour (equations 4.4 and 4.5 respectively):

$$\mathcal{L}(\theta_m|\mathbf{x}) = \mathcal{N}_{\theta m}(\mathbf{x}) \tag{4.4}$$

$$\mathcal{L}(\theta_{bg}|\mathbf{x}) = \mathcal{N}_{\theta bg}(\mathbf{x}) \tag{4.5}$$

Given the two likelihoods, we define the "melodiness" index $\mathcal{M}(\mathbf{x})$ of a pitch contour with feature vector $\mathbf{x}$ as the likelihood ratio of the melodic

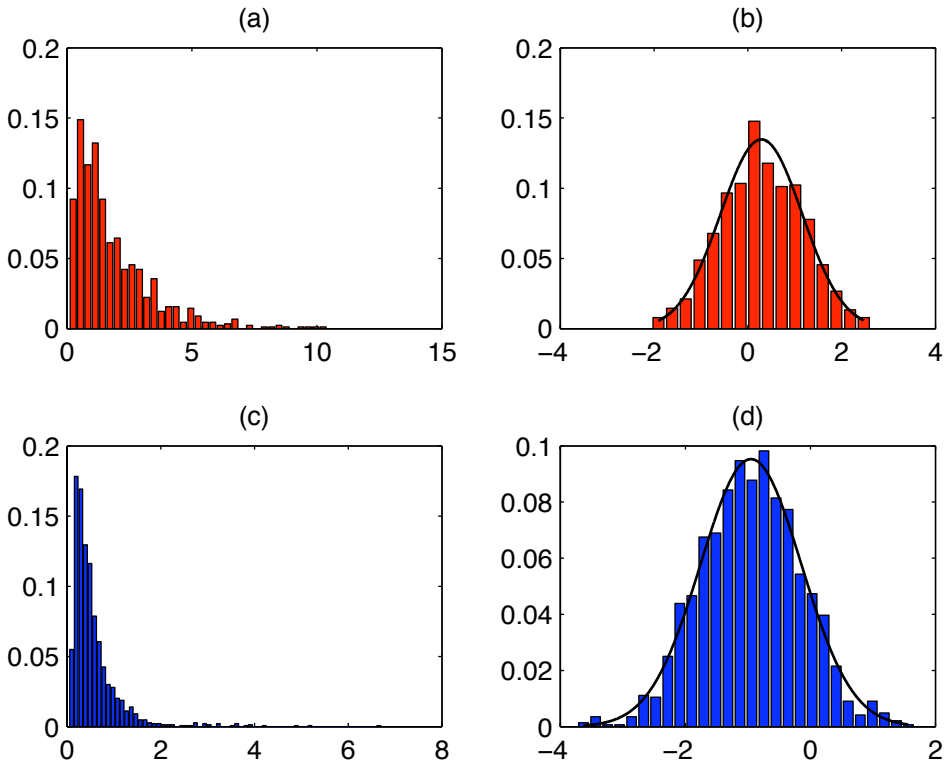

Figure 4.8: Contour total salience distributions. For melodic contours: (a) raw data, (b) after applying the Box-Cox transform. For non-melodic contours: (c) raw data, (d) after the Box-Cox transform.

and non-melodic likelihoods:

$$\mathcal{M}(\mathbf{x}) = \frac{\mathcal{L}(\theta_m|\mathbf{x})}{\mathcal{L}(\theta_{bg}|\mathbf{x})} \tag{4.6}$$

As a final step, we need to define how to exploit $\mathcal{M}(\mathbf{x})$ in order to select the melody out of all the pitch contours. We define a straight-forward rule for melody selection based on our proposed melodiness index $\mathcal{M}(\mathbf{x})$: given all the contours generated for a musical excerpt, at each frame we check to see which contours are present in the frame and select as the melody the pitch value belonging to the contour whose features $\mathbf{x}$ result in the highest melodiness index $\mathcal{M}(\mathbf{x})$.

4.4 Evaluation and results

4.4.1 Introduction

The model-free algorithm described in Section 4.2 was submitted to the 2010 and 2011 Music Information Retrieval Evaluation eXchange (MIREX; cf. Chapter 2; Downie, 2008). This allowed us not only to evaluate our proposed method on a varied set of testing material, but also to compare it with other state-of-the-art approaches. The difference between the 2010 and 2011 submissions is that the former was submitted prior to the optimisation of the salience function parameters described in Chapter 3. By comparing our results before optimisation (2010) and after (2011) we can evaluate the effect of the optimisation on the overall performance of the algorithm. Furthermore, in 2011 we submitted two versions of the algorithm, one using the STFT and the other using the MRFFT (cf. Section 3.2.2), in this way complementing the comparative evaluation carried out in Chapter 3.

In addition to the MIREX results, we carried out several complementary evaluation experiments, providing further insight into the performance of the algorithm. These include: a qualitative error analysis focusing on octave errors, a study of the effect of the key parameter in our voicing detection method ν on performance, an evaluation of the influence of each algorithmic component of the system on overall performance, and a glass ceiling study in which we examine the current limitations of the algorithm, including the quality of contour formation (peak streaming). The results of these experiments, in particular the glass ceiling study, allow us to identify which parts of the algorithm could be further improved, and provide future directions for our research. Finally, we evaluate the model-based version of the algorithm proposed in Section 4.3 and compare the results to those obtained by the model-free algorithm.

For a detailed description of the music collections and evaluation measures used in MIREX (both 2010 and 2011) the reader is referred to Section 2.4 of this dissertation. For the additional experiments on voicing detection (Section 4.4.4), component evaluation (Section 4.4.5), glass ceiling analysis (Section 4.4.6) and for comparing the model-free and model-based versions of the algorithm (Section 4.4.7), we have compiled a representative test set comprised of excerpts from different annotated collections that are freely available to researchers. The test set includes 65 audio excerpts from a variety of musical genres including rock, pop, R&B, jazz and opera singing. More specifically, it consists of 16 of the ADC2004 excerpts (all excerpts

except for those synthesised from MIDI files), 9 excerpts similar to those used in the MIREX05 collection, and 40 excerpts similar to those used in the MIREX09 collection. The durations of the excerpts range from 5 to 35 seconds.

The evaluation results are presented in chronological order: we start with the results of MIREX 2010, followed by a qualitative error analysis of our submission that year. Next, we present the results obtained by the algorithm in MIREX 2011, now using the optimised salience function parameter values. Then we discuss the additional evaluation experiments mentioned above, and finally we evaluate and compare the model-free and model-based versions of the algorithm.

4.4.2 Comparative evaluation: MIREX 2010

A preliminary version of the algorithm was submitted to MIREX 2010. The submission was made before the detailed study of salience function parameters was carried out (cf. Chapter 3). Recall that the key parameters affecting the salience function are the weighting parameters α and β, the magnitude threshold γ and the number of harmonics N_h. Following the analysis described in chapter 3 the optimal parameter values were determined as $\alpha = 0.8$, $\beta = 1$, $\gamma = 40$ and $N_h = 20$. The values used in the 2010 submission, which were empirically assigned based on initial experiments, were 0.8, 2, 40 and 8 respectively.

Five algorithms participated in the 2010 Audio Melody Extraction (AME) task of the MIREX campaign. In Table 4.1 we present the overall accuracy results obtained by each algorithm for each of the test collections. Each submission is denoted by the initials of its authors – HJ (Hsu & Jang, 2010a), TOOS (Tachibana et al., 2010a), JJY (who submitted two variants; Joo et al., 2010) and SG (our submission). For completeness, we also include the results obtained by the best performing algorithm from the previous year's campaign (Dressler, 2009), denoted KD. In the last column we provide the mean overall accuracy computed over all six collections[2]. The best result obtained in 2010 for each collection is highlighted in bold.

We see that of the algorithms participating in 2010, our algorithm achieved the highest mean overall accuracy, surpassed only by the best performing

[2]The mean is not weighted by the size of the datasets due to the order of magnitude difference in size between the 2009 datasets and the other collections which, though smaller, are more representative of the type of material one would encounter in a real-world scenario.

Algorithm	2004	2005	2008	2009 -5dB	2009 0dB	2009 +5dB	Mean
HJ	0.61	0.54	0.77	**0.63**	**0.76**	**0.83**	0.69
TOOS	0.54	0.61	0.72	**0.63**	0.72	0.79	0.67
JJY2	**0.72**	0.61	**0.80**	0.47	0.63	0.79	0.67
JJY1	0.70	**0.62**	**0.80**	0.47	0.63	0.79	0.67
SG	0.70	**0.62**	0.78	0.58	0.74	0.81	**0.70**
KD	0.86	0.75	0.81	0.52	0.68	0.78	0.73

Table 4.1: Overall accuracy results: MIREX 2010.

algorithm from the previous year. Nonetheless, the performance of all algorithms is quite similar (with the exception of KD for the 2004 and 2005 datasets[3]). To assess the significance of the differences in performance, we performed a two-way analysis of variance (ANOVA) of the results obtained by the algorithms participating in 2010, followed by a Tukey range test with $\alpha = 0.05$. Indeed, for MIREX05 there is no statistically significant difference between the algorithms, and for ADC2004 and INDIAN08 the difference is significant only between the top 2-3 algorithms the worst performing algorithm. This is probably in part due to the small size of these collections. For the three 2009 collections, a statistically significant difference was found between most algorithms, though the artificial nature of these collections (karaoke accompaniment, amateur singing and no studio mixing or post production) makes them less representative of a real-world scenario.

The comparable performance of most algorithms suggests that further error analysis would be necessary to gain insight into the advantages and caveats of each algorithm. To this end, we performed a qualitative error analysis of our submission, focusing on octave errors. We noted that for the MIREX05 collection there was a significant difference between the raw pitch accuracy and the raw chroma accuracy of our approach, indicating a larger number of octave errors compared to the other collections. In plot (a) of Figure 4.9 we display the raw pitch accuracy and the raw chroma accuracy obtained by our 2010 submission for each of the MIREX collections, and in plot (b) we provide the per-song results for the MIREX05 collection. Examining the per-song results we discovered that the largest differences between pitch and

[3] A possible explanation for this is KD's better ability at extracting non-vocal melodies, which constitute a larger proportion of these collections.

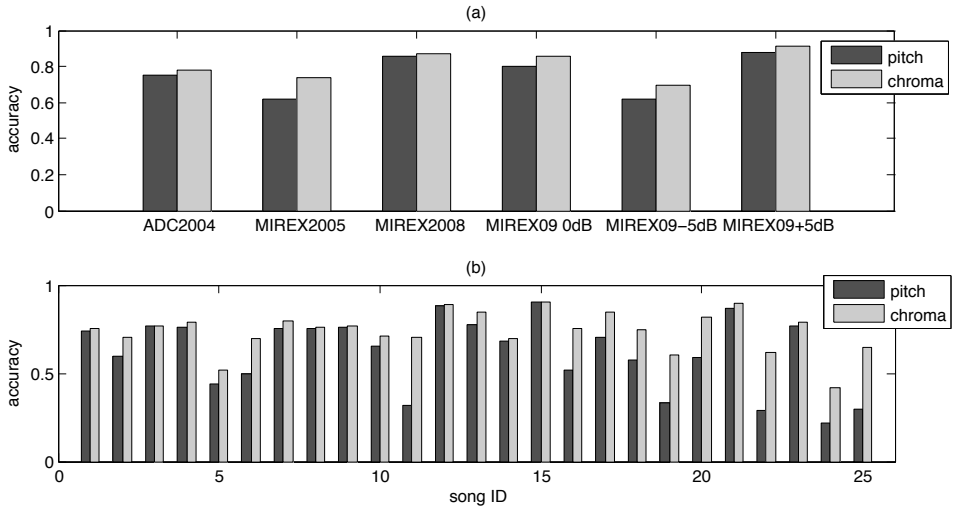

Figure 4.9: Pitch and chroma results for our 2010 submission: (a) mean raw pitch and raw chroma accuracies for each test collection, and (b) Per-song raw pitch and raw chroma accuracies for the MIREX05 collection.

chroma accuracy occur mainly in non-vocal excerpts, especially solo piano pieces. This suggests that whilst our octave selection method works well for vocal music, further work would be required to adapt it for instrumental music, especially that performed by a single (polyphonic) instrument such as the piano.

4.4.3 Comparative evaluation: MIREX 2011

In the following year, we submitted two updated versions of our algorithm: in the first, denoted SG1, we apply the STFT, and in the second, denoted SG2, we use the MRFFT instead. For both versions, we now use the optimised parameter values for the salience function. Eight participants took part in the MIREX 2011 melody extraction task, submitting a total of 10 algorithms: TY1 and TY2 (unpublished), TOS (Tachibana et al., 2010b), LYRS (Liao et al., 2011), HCCPH (unpublished), CWJ (Chien et al., 2011), YSLP (Yoon et al., 2011), PJY (Park et al., 2011), and our two submissions (SG1 and SG2). The overall accuracy results are provided in Table 4.2. For easy comparison, our result from 2010 is repeated at the end of the table.

We see that regardless of the spectral transform used, our optimised algorithm achieves the highest overall accuracy in four of the six test sets.

Algorithm	2004	2005	2008	2009 -5dB	2009 0dB	2009 +5dB	Mean
TY1	0.47	0.51	0.70	0.41	0.52	0.56	0.53
TY2	0.47	0.51	0.70	0.41	0.52	0.56	0.53
TOS	0.59	0.57	0.72	**0.62**	0.74	0.82	0.68
LYRS	0.73	0.59	0.72	0.36	0.47	0.54	0.57
HCCPH	0.44	0.45	0.64	0.39	0.50	0.59	0.50
CWJ	0.73	0.57	0.69	0.40	0.53	0.62	0.59
YSLP	**0.85**	0.65	0.73	0.39	0.52	0.66	0.63
PJY	0.81	0.65	0.71	0.54	0.74	0.83	0.71
SG1	0.74	**0.66**	**0.83**	0.61	**0.78**	**0.85**	**0.75**
SG2	0.74	**0.68**	**0.84**	0.61	**0.78**	**0.85**	**0.75**
SG (2010)	0.70	0.62	0.78	0.58	0.74	0.81	0.70

Table 4.2: Overall accuracy results: MIREX 2011.

Consequently, our method also achieves the highest mean overall accuracy (surpassing KD), making it the best performing melody extraction algorithm to be evaluated on the current MIREX test sets (2009 to date). Since more algorithms participated this year, and following the improvement in the performance of our algorithm, the differences between the algorithms are more significant now. Once again we conducted a two-way ANOVA followed by a Tukey range test with $\alpha = 0.05$: for ADC04 the top 6 algorithms are significantly better than the bottom 3 algorithms, for MIREX05 the top 4 are significantly better than the bottom 3, for INDIAN08 the top 2 are significantly better than the worst algorithm, for MIREX09 (-5dB) the top 3 algorithms (including SG1 and SG2) are significantly better than the rest, and for MIREX09 (0dB) and MIREX09 (+5dB) the top 2 algorithms, namely our two submissions, are significantly better than all the rest. We see that in addition to being the top performing algorithm in four out of the six collections, our algorithm is always amongst the group of algorithms which perform significantly better than the rest. When comparing our results before optimisation (2010) and after (2011), we see that for all collections there is a notable improvement in accuracy. For all collections but ADC2004 the improvement is also statistically significant. The increase can be attributed to better voicing detection (resulting in lower voicing false alarm rates), better contour generation (higher pitch and chroma accuracies) and less octave errors (a smaller difference between pitch and chroma

accuracies). When comparing the performance of SG1 (STFT) versus SG2 (MRFFT), we see that for most collections the performance is practically identical. For two of the six collections using the MRFFT actually yields slightly better results. However, in concurrence with the analysis conducted in Chapter 3, the difference between SG1 and SG2 is not statistically significant for any of the collections. Following this result, for the remainder of this chapter the experiments will be carried out using the STFT version of the algorithm (Salamon & Gómez, 2012). Finally, we note that whilst the algorithm's parameters have been optimised, it could still be improved through the introduction of new contour characteristics or additional signal processing steps. These options are discussed further in Section 4.4.6.

4.4.4 Voicing

In Section 4.2.3 we proposed a new voicing detection method in which the determination of voiced sections is based on the study of contour feature distributions. The method was in part responsible for the successful results in MIREX, where our algorithm achieved the best trade-off between voicing recall and voicing false alarm rates. In this section we study the sensitivity of our algorithm to the method's key parameter ν (Equation 4.1). Recall that ν determines the lenience of the filtering: increasing ν makes it more lenient (less contours are filtered out), whilst decreasing ν makes it stricter (more contours are filtered out). In Figure 4.10 we plot the overall accuracy, voicing recall and voicing false alarm rates for our representative test set as a function of ν. To see how the optimal value of ν changes with respect to the characteristics of the music collection being analysed, we split the test set into three subsets: excerpts taken from ADC04, excerpts similar to those in MIREX05 and excerpts similar to those in MIREX09.

As expected, the trade-off between the voicing recall and voicing false alarm rates is clearly visible. As ν is increased (reducing the filtering threshold τ_v) the recall rate goes up for all subsets, but so does the false alarm rate. The optimal value for ν is the one which gives the best balance between the two, and can be inferred from the overall accuracy. As expected, we see that this optimal value is slightly different for each of the three subsets. This is because the relationship between the salience distribution of melody contours and the salience distribution of non-melody contours (cf. plot (c) of Figure 4.3) is determined by the type of musical accompaniment used, which varies between the collections. Nonetheless, the optimal value of ν for the three subsets lies within a sufficiently limited range (0.0-0.4) such

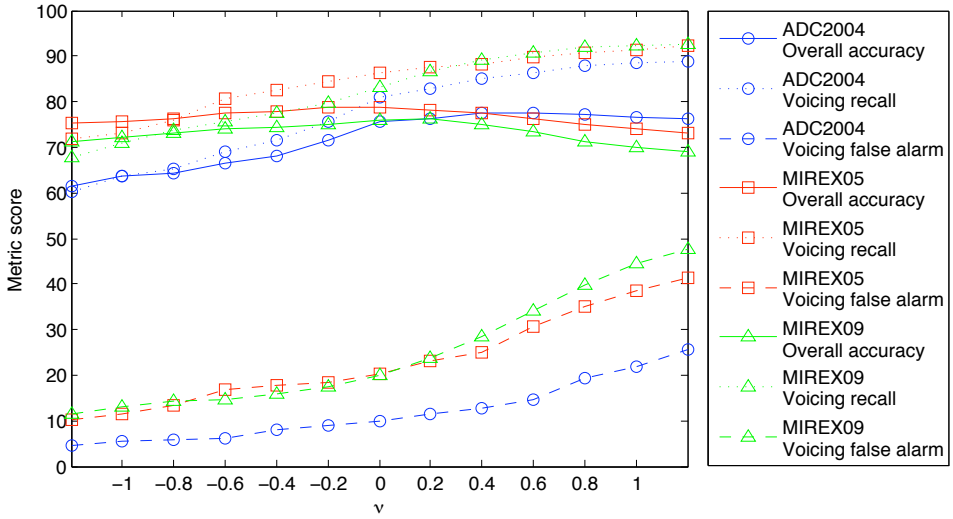

Figure 4.10: Overall accuracy, voicing recall and voicing false alarm rates versus the voicing parameter ν.

that a satisfactory compromise can be made (e.g. for the collections under investigation, $\nu = 0.2$). Finally, this (albeit small) difference between the optimal value of ν for each collection suggests that whilst the proposed approach already provides good results, further contour characteristics would have to be considered in order to improve voicing detection rates across a wide range of musical styles and genres. As future work we propose the development of a voiced contour classifier trained using a wider set of contour features.

4.4.5 Component evaluation

As with the voicing filter, each algorithmic component of the system influences its overall performance. In Table 4.3 we evaluate the complete system on the representative test set (cf. Section 4.4.1) each time removing one component, in this way assessing its effect on overall performance. The components removed are: equal loudness filter (EQ), peak frequency correction (FC), voicing filter (VF), octave duplicate and outlier removal (OO). We also test replacing the optimised salience function parameter values with ones used in MIREX 2010 (SF), as well as removing different combinations of components.

Component Removed	Voicing Recall	Voicing False Alarm	Raw Pitch	Raw Chroma	Overall Accuracy
None	0.86	0.19	0.81	0.83	0.77
EQ	0.83	0.19	0.79	0.81	0.75
FC	0.85	0.19	0.79	0.82	0.76
EQ & FC	0.83	0.18	0.77	0.80	0.75
SF	0.85	0.24	0.77	0.81	0.74
VF	0.92	0.42	0.81	0.83	0.72
OO	0.87	0.24	0.79	0.83	0.75
VF & OO	0.95	0.56	0.79	0.83	0.67
All	0.95	0.64	0.71	0.78	0.60

Table 4.3: System performance with different components removed.

We see that each component has a direct effect on the overall accuracy. Importantly, we note that there is a strong interaction between components. For example, without the voicing filter (VF) the accuracy goes down by 5 percentage points and without the octave duplicate and outlier removal (OO) it goes down by 2 points, but if both were removed the accuracy would drop by 10 points. This reveals that the latter step (OO), in addition to its primary role, also improves voicing detection by removing contours at unvoiced sections of the piece that were missed by the voicing filter. If all components were removed the combined effect would cause a drop of 17 points in overall accuracy, which is 4 points more than the sum of all individual accuracy decreases combined.

4.4.6 Glass ceiling analysis

As a final step in the evaluation of the model-free algorithm, we test to see what would be the best result our algorithm could possibly achieve assuming we had a perfect contour filtering approach. To do this, we compare all contours generated for an excerpt with its ground truth, and keep only those which overlap with the annotated melody. These contours are then passed to the final melody selection stage as before, and the resulting melody is evaluated against the ground truth. In Table 4.4 we present for each subset of the representative test set the best result obtained by our algorithm (SG), followed by the result obtained using the perfect filtering simulation (SIM).

Comparing the results obtained by our algorithm to the results using the

Subset	Alg.	Voicing Recall	Voicing False Alarm	Raw Pitch	Raw Chroma	Overall Accuracy
ADC2004	SG	0.83	0.11	0.79	0.81	0.76
	SIM	0.84	0.05	0.84	0.84	0.84
MIREX05	SG	0.88	0.23	0.83	0.84	0.78
	SIM	0.86	0.07	0.84	0.85	0.86
MIREX09	SG	0.87	0.24	0.81	0.84	0.76
	SIM	0.86	0.14	0.85	0.85	0.83

Table 4.4: Results obtained by the algorithm (SG) compared to the glass ceiling results obtained using perfect contour filtering simulation (SIM).

perfect filtering simulation, we can make several important observations. First of all, we see that the overall accuracy using the perfect contour filtering simulation is still below 100%. As suggested by the title of this section, this reveals a glass ceiling, i.e. a top limit on the overall accuracy that could be obtained by the algorithm in its current configuration. We begin by discussing the differences between our algorithm's results and the glass ceiling results, and then analyse the limitations of the algorithm that result in this glass ceiling.

We start by drawing the reader's attention to the raw chroma measure. We see that the chroma accuracy of the algorithm is practically equal to the glass ceiling result. This suggests that the algorithm can select the correct contour when faced with two or more simultaneous contours (that are not octave duplicates) almost perfectly. Turning to the raw pitch accuracy, the results obtained by the algorithm are on average only 3.5 percentage points below the glass ceiling result. Again, this implies that whilst there is still room for improvement, the octave error minimisation method proposed in Section 4.2 is certainly promising. The main difference between our algorithm and the glass ceiling results is the voicing false alarm rate. Though the voicing detection method we have proposed here is already one of the best according to MIREX, we see that further improvements to the method would provide the most significant increase in the overall accuracy of our approach.

Finally, we consider the possible cause of the identified glass ceiling limit. Assuming the algorithm can perform perfect contour filtering, the overall accuracy is determined entirely by the accuracy of the contour creation block.

If all melody contours were perfectly tracked, the raw pitch and chroma scores of the glass ceiling should reach 100%. This implies that to increase the potential performance of the algorithm, we would have to improve the accuracy of the contour formation process. Currently, our tracking procedure takes advantage of temporal, pitch and salience information. We believe that an important part of the puzzle that is still missing is timbre information. Timbre attributes have been shown to provide important cues for auditory stream segregation (Iverson, 1995), suggesting they could similarly be of use for pitch contour tracking. Furthermore, the extraction of pitch specific timbre attributes could lead to the development of a contour timbre feature C_{timbre}, that could be used in the melody selection process by introducing rules based on timbre similarity between contours. Another possibility for improving contour formation would be the suppression of noise elements in the signal before the salience function is computed. For instance, we could apply harmonic/percussive source separation as performed by Ono et al. (2010); Tachibana et al. (2010a) to minimise the disruptions in the salience function caused by percussive instruments.

4.4.7 Statistical modelling results

We now turn to evaluate the modified version of the algorithm proposed in Section 4.3. Recall that here, instead of applying the heuristic-based contour filtering, we compute a "melodiness" index $\mathcal{M}(\mathbf{x})$ for every contour, and then at each frame we check which contours are present and select the pitch value belonging to the contour with the highest $\mathcal{M}(\mathbf{x})$ as the melody. For the evaluation we use the representative test set described previously (cf. Section 4.4.1). Since this approach is based on training (learning the parameters of the Gaussian distributions based on observed data), to avoid any bias in the results we separate the training and evaluation material by conducting a 3-fold cross validation: the training data in each fold (two thirds of the data) is used to obtain the model, consisting of the melodic and non-melodic multivariate normal distributions, and the remaining data is used to evaluate the model-based algorithm. We then report the average result over the three folds. In Table 4.5 we present the results obtained by the model-based approach. Note that we do not apply any voicing detection step, that is, the algorithm estimates all frames as voiced, with the exception of frames in which no contours were created. For completeness we calculate all the melody extraction evaluation measures described in Section 2.4.1 though, since we do not attempt to perform any voicing detection, only the raw pitch and raw chroma measures (highlighted in bold) should be taken

Algorithm	Voicing Recall	Voicing False Alarm	**Raw Pitch**	**Raw Chroma**	Overall Accuracy
Model-based	0.95	0.60	**0.77**	**0.83**	0.65
Model-free	0.86	0.19	**0.81**	**0.83**	0.77

Table 4.5: Results obtained using the model-based algorithm without voicing detection, compared to those obtained by the model-free algorithm.

into consideration at this point. For comparison, we include the results obtained by the model-free version of the algorithm (note that this version includes voicing detection). In Section 4.4.3 we saw that the model-free algorithm obtained the highest mean overall accuracy results in MIREX 2011 (Salamon & Gómez, 2011). As such, it is worth noting that we are not comparing the model-based algorithm to a baseline approach, but rather to a state-of-the-art method. We see that the model-based approach obtains the same chroma accuracy as the state-of-the-art algorithm. The lower raw pitch accuracy indicates that the model-based approach makes slightly more octave errors. Nonetheless, the results are definitely promising, and their comparability to the model-free approach suggests that the performance of the model-based algorithm is also comparable with other state-of-the-art melody extraction algorithms evaluated in MIREX.

Next, we combine the model-based approach with the voicing detection method proposed in Section 4.2.3. Recall that the method is based on filtering out contours by setting a salience threshold determined from the distribution of the contour mean salience $C_{\bar{s}}$ in a given excerpt. Thus, the combined approach consists of first filtering out contours from unvoiced sections of the excerpt using the voicing filter, and then selecting the melody out of the remaining contours based on their "melodiness" index $\mathcal{M}(x)$ as before. The f_0 estimate for unvoiced frames (recall that algorithms can return f_0 estimates for unvoiced frames so that pitch and voicing accuracies can be estimated independently) is obtained by considering all the contours that were present in these frames before applying the voicing detection step and selecting the pitch value belonging to the contour with the highest $\mathcal{M}(x)$. The results are presented in Table 4.6, once again alongside the results obtained by the model-free algorithm for comparison. This time we focus on the voicing evaluation measures and the overall accuracy.

As expected, by combining the model-based approach with a voicing detection method we are able to considerably reduce the voicing false-alarm

Algorithm	**Voicing Recall**	**Voicing False Alarm**	Raw Pitch	Raw Chroma	**Overall Accuracy**
Model-based	**0.87**	**0.25**	0.78	0.83	**0.74**
Model-free	**0.86**	**0.19**	0.81	0.83	**0.77**

Table 4.6: Results obtained using the model-based algorithm with voicing detection, compared to those obtained by the model-free algorithm.

rate (from 60% to 25%) whilst maintaining a relatively high voicing recall rate (87%). As a result, the overall accuracy of the model-based approach goes up from 65% without voicing detection to 74% with voicing detection. Even though the same voicing detection approach is applied in both cases, we note that the voicing false-alarm rate is not the same. This is because, as seen in Section 4.4, some steps in the heuristic-based contour filtering have a positive effect on voicing detection even though they were not designed to address this problem (e.g. removing pitch outliers). Whilst the combined approach still does not outperform the model-free algorithm, the results serve as a promising proof-of-concept, with an overall accuracy which is comparable to other state-of-the-art melody extraction algorithms.

As a final step, we inspect the values of our melodiness index $\mathcal{M}(x)$ for melody and non-melody contours. In Figure 4.11, we plot the values of $\mathcal{M}(x)$ for all pitch contours of all excerpts in the representative test set (on a log scale). For each excerpt we first normalise all $\mathcal{M}(x)$ values by the highest value in the excerpt, so that we can plot all values from all excerpts in a single graph. Values for melody contours are represented by a red circle, and values for non-melody contours by a blue x. We see that the $\mathcal{M}(x)$ values for the two classes are fairly distinguishable, with the vast majority of melody contours having higher $\mathcal{M}(x)$ values than non-melody contours. This, apart from suggesting that $\mathcal{M}(x)$ is indeed a good indicator for melody contours, means that in future work we might be able to refine our melody selection algorithm by studying the distributions of $\mathcal{M}(x)$ for melody and non-melody contours.

4.5 Conclusion

In this chapter we presented a salience-based melody extraction algorithm which exploits the creation and characterisation of pitch contours. We showed that by studying the distributions of different pitch contour char-

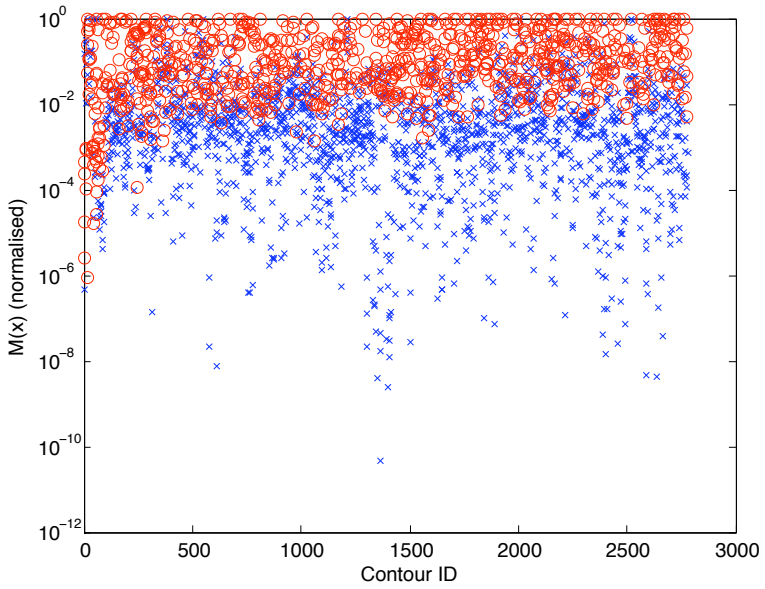

Figure 4.11: Normalised $\mathcal{M}(x)$ values for melody contours (red circle) and non-melody contours (blue x).

acteristics we can identify features that distinguish melody contours from non-melody contours. It was then explained how these features are used for filtering out non-melody contours, resulting in novel voicing detection and octave error minimisation methods. We also proposed an alternative version for the algorithm's melody selection block which is based on a statistical model of the contour features. We showed how the features can be used to build a model to describe melodic and non-melodic contours, leading to the computation of a "melodiness" index $\mathcal{M}(\mathbf{x})$.

The model-free algorithm was evaluated in two MIREX campaigns, where the latest version of the algorithm (2011) was shown to outperform all other participating state-of-the-art melody extraction algorithms. The results were complemented with a qualitative error analysis, revealing that the different characteristics of instrumental music complicate the task of octave error minimisation, requiring further adjustments to the proposed method for this type of musical content. The MIREX 2011 results also confirmed the expected increase in performance following the optimisation of the salience function parameter values. We evaluated the influence of individual algorithmic components on overall performance, and noted that the interaction between different components can be important for maintaining high accuracies. A glass ceiling analysis confirmed that in most cases the proposed

contour filtering approach is successful at filtering out non-melody contours, though a further increase in accuracy could still be achieved by reducing the voicing false alarm rate of the system. In addition, it was determined that to increase the potential performance of the algorithm we would have to improve the contour formation stage, and possible methods for achieving this were proposed.

Finally, we evaluated the proposed model-based version of the algorithm. The results of the evaluation showed that the approach achieves pitch and chroma accuracies comparable to the model-free algorithm, which is state-of-the-art. By combining the model-based approach with a voicing detection method, we were able to obtained satisfying overall accuracy results as well. When considering the caveats of the model-based approach compared to the model-free algorithm, one clear difference is that whilst the contour filtering steps (applied in the latter) take into account temporal information, in the model-based approach the melody selection at each frame is performed using the "melodiness" index $\mathcal{M}(\mathbf{x})$ only, and no temporal continuity is taken into account. This means that the pitch trajectory of the melody is allowed to contain large jumps which are not common in melodies, which tend to have a relatively smooth pitch trajectory in accordance with voice leading principles (Huron, 2001). Thus, a possible direction for improving the performance of the model-based approach is to combine the melodiness index with some type of temporal constraint. For instance, we could use the "melodiness" index in combination with one of the tracking techniques mentioned Chapter 2, such as HMMs (Ryynänen & Klapuri, 2008a) or tracking agents (Dressler, 2011a; Goto, 2004). Another possibility for improvement is to consider more contour features. For instance, earlier in this chapter it was explained that in 95% of the cases where the algorithm detected vibrato in a contour, that contour belonged to the melody. This information is not exploited in the current model (with the exception of the voicing detection method). An additional important research direction would be the gathering of more data to enable the use of more sophisticated statistical models (e.g. GMMs). Finally, another interesting research direction would be to learn genre specific feature distributions, and depending on the genre of the excerpt use a different model to compute $\mathcal{M}(\mathbf{x})$. This could be done by creating a two stage system, where in the first stage a classification algorithm is used to identify the genre of the piece and in the second stage the model-based melody extraction algorithm is used with genre-specific distributions. The classification stage could even be performed by exploiting the contour features presented in this chapter, as proposed by Salamon et al. (2012c) (cf. Chapter 6).

Chapter 5

Evaluation Methodology: Challenges and Prospects

5.1 Introduction

As seen in Chapter 2, the task of melody extraction has received growing attention from the research community in recent years. As the number of researchers working on the task grew, so did the need for proper means of evaluating and comparing the performance of different algorithms. In 2004, the first Audio Description Contest (ADC) was hosted by the Music Technology Group at Universitat Pompeu Fabra in Barcelona, Spain. Amongst the different MIR tasks included in this contest, there was a specific track for melody extraction algorithms. This initiative was followed by the first Music Information Retrieval Evaluation eXchange (MIREX) in 2005 (Downie, 2008), which has been held annually since in conjunction with the International Society for Music Information Retrieval Conference (ISMIR).

MIREX has become the de-facto benchmark for evaluating and comparing the performance of melody extraction algorithms, with over 50 algorithms evaluated since the first run in ADC 2004. Whilst this is without doubt an indication of the formalisation of the topic as an established research area, it has recently been argued that some of the evaluation procedures employed by the Music Information Retrieval (MIR) research community in general still lack the rigour found in other disciplines such as Text IR (Urbano, 2011; Urbano et al., 2013b). In this chapter, based on Salamon & Urbano (2012), we examine the evaluation of melody extraction algorithms, as currently

carried out in the MIREX Audio Melody Extraction (AME) task. We focus on three aspects of the evaluation: first, we examine the annotation procedure used for generating a ground truth for evaluation. Specifically, we study the influence of a systematic error in the annotations, in the form of a fixed time offset between the ground truth annotation and the output of the algorithms. This issue is particularly relevant, as such an error was actually identified (by the author of this thesis) in past MIREX AME evaluations. Next, we consider the duration of the audio excerpts (clips) used for evaluation. Currently all collections used for evaluation are comprised of short excerpts taken from full songs. The underlying assumption is that the performance of an algorithm on a short clip is a good predictor for its performance on the full song. However, to date this assumption has neither been confirmed nor confuted. Finally, we consider the aspect of collection size. Currently, the size of most collections used for AME evaluation is relatively small compared to collections used in other IR tasks, and so we assess whether this presents any problems or not. Through these factors, we aim to assess the reliability of the evaluation procedure, as well as the meaningfulness of the results and the conclusions that are drawn from them.

To correctly follow this chapter it is important that the reader be well aware of the evaluation procedure followed in MIREX, including the test collections used, how they were annotated, and the evaluation measures employed to assess algorithmic performance. These were all described in detail in Section 2.4 of this thesis, to which the reader is referred. The remainder of this chapter is organised as follows: in Section 5.2 we examine the annotation procedure, and assess the potential influence of a systematic error in the annotation on the results. In Section 5.3 we study the relationship between algorithm performance and clip duration. In Section 5.4 we consider the influence of the size of the music collection used for evaluation on the stability of the results. Then, in Section 5.5 we provide some further discussion and analysis of the results obtained in the previous sections, and finally we present our conclusions in Section 5.6.

5.2 Ground truth annotation offset

We start by studying the influence of a specific type of systematic error in the annotation on the results. Whilst there are other aspects of the annotation process that are also worth consideration, we find this issue to be of particular interest, since it was actually identified recently in one of the music collections used for AME evaluation in MIREX. As explained in

the Section 2.4 of this thesis, all AME evaluation measures are based on a frame-by-frame comparison of the algorithm's output to the ground truth annotation. Hence, if there is a time offset between the algorithm's output and the ground truth annotation, this will cause a mismatch in all frames. Since melody pitch tends to be continuous, a very small time offset may not be noticed. However, as we increase the offset between the two sequences, we can expect it to have an increasingly detrimental effect on the results.

To evaluate the effect of such an offset, we compiled a collection of 30 music clips from publicly available MIREX training sets: 10 from ADC2004, 9 similar to MIREX05 and 11 similar to MIREX09. We used the ground truth annotations generated by the original authors of each collection, and ensured that the first frame of each annotation was centred on time 0. For evaluation, we use the output of six different melody extraction algorithms that were kindly provided by their authors: KD (Dressler, 2009), DR (Durrieu et al., 2010)[1], FL (Fuentes et al., 2012), HJ (Hsu et al., 2011), RP (Paiva, 2007) and finally the model-free algorithm presented in this dissertation (Salamon & Gómez, 2012; cf. Section 4.2), denoted SG. For each algorithm, we computed the mean raw pitch accuracy and the overall accuracy for the entire collection, as a function of a fixed time offset introduced in the ground truth annotation, from -50 ms to 50 ms using 1 ms steps. To emulate offsets smaller than the hop size of the annotation (10 ms), the ground truth was upsampled using linear interpolation.

5.2.1 Results

In Figure 5.1 we display the results of the evaluation, where we have subtracted from the values obtained by each algorithm their score at offset 0. In this way, the graph reflects the absolute difference between the score for a given offset and the optimal score of the algorithm (assuming its first frame is centred on time 0). Plot (a) displays the results for the raw pitch accuracy, and plot (b) for the overall accuracy. As can be seen, the effect of the offset is quite dramatic, causing an absolute drop of up to 25 percentage points in the raw pitch accuracy and 20 points in the overall accuracy for the most extreme offset evaluated (50 ms). Though a 50 ms offset is perhaps an exaggerated case, in 2011 (prior to the MIREX 2011 evaluations) it was discovered that one of the MIREX collections had a 20 ms offset. In our evaluation, a 20 ms offset would cause the most affected

[1]The output was computed using a different implementation than that of the paper, which is available at: https://github.com/wslihgt/separateLeadStereo

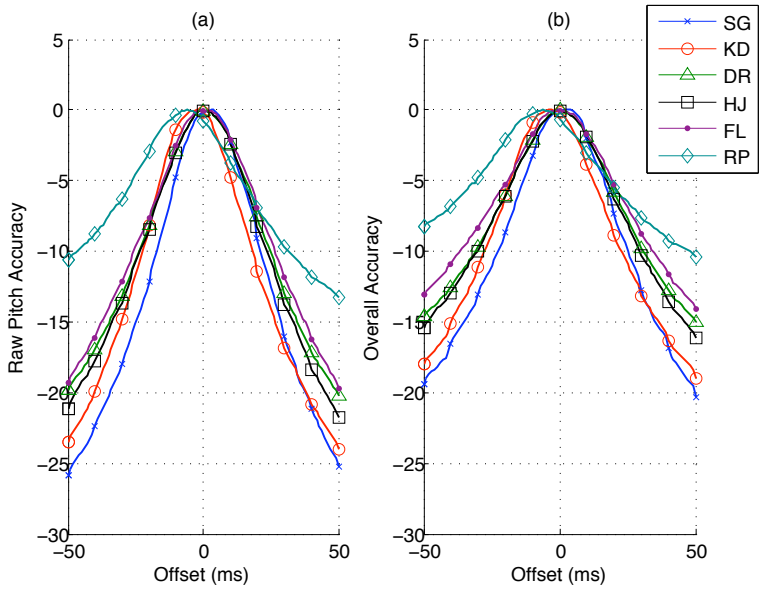

Figure 5.1: Absolute performance drop versus annotation offset: (a) raw pitch accuracy, (b) overall accuracy.

algorithms to loose 17 points in raw pitch accuracy and 13 points in overall accuracy. Another interesting observation is that some algorithms do not perform best at offset 0 (most visibly RP, whose peak performance is at -6 ms). This emphasises the fact that it does not suffice for the annotation to be centred on time 0, but rather, that there must be a strict convention to which both the annotations and algorithms adhere. Finally, we found that there is a correlation between the absolute performance of an algorithm and the effect of the annotation offset: the higher the absolute performance of the algorithm, the more sensitive it is to an offset in the annotation. This is particularly important, since it suggests that the best performing algorithms (in this evaluation SG and KD) are those that will be most affected by this type of systematic error.

5.3 Clip duration

A common criticism of evaluation in MIR, and particularly in MIREX, is the use of clips (short excerpts) instead of full songs. One might argue that the use of clips is unrealistic and that the observed performance on those

clips may be very different from the performance of the same algorithm on full songs (Urbano, 2011). The collections used in the AME evaluation contain some very short excerpts, some only 10 seconds long. The use of such small clips is especially striking in AME: these clips contain mostly voiced frames, and so the generalisation of the results to full songs could be questioned. We designed an experiment to assess the effect of clip duration on the reliability of the AME evaluations.

For each of the 30 clips used in the previous experiment (referred to as the $\times 1$ clips), we created a series of subclips: 2 subclips of half the duration, 3 subclips of one third of the duration, and 4 subclips of one forth of the duration (referred to as the $\times 1/2$, $\times 1/3$ and $\times 1/4$ subclips). Note that the $\times 1/4$ subclips can also be considered as $\times 1/2$ versions of the $\times 1/2$ subclips. This gives us 180 $\times 1/2$ subclips, 90 $\times 1/3$ subclips and 120 $\times 1/4$ subclips, all of which were used to evaluate the six algorithms mentioned in the previous section. We computed the performance difference between all subclips and their corresponding $\times 1$ versions, leading to a grand total of 2340 data-points.

5.3.1 Results

In Figure 5.2 we show the log-scaled distribution of relative performance differences. We see that the mean differences vary between 13% and 21% for the overall accuracy and raw pitch accuracy, whilst for the voicing false alarm rate the means are around 50%. We also note that there is a large amount of outliers in the distributions. However, these outliers were not found to correspond to particular songs or algorithms (they are fairly randomly distributed). There seems to be a clear correlation: the shorter the subclips, the larger the performance differences (all significant by a 1-tailed Wilcoxon test, α=0.01). In principle, therefore, one would want the clips used for evaluation to be as long as possible; ideally, the full songs.

In Figure 5.3 we plot the log-scaled relative performance differences in overall accuracy, this time as a function of the log-scaled actual subclip duration (other measures produce very similar plots). We see that the negative correlation between subclip duration and performance difference appears to be independent of the duration of the $\times 1$ clip. We fitted a non-linear model of the form $\mathit{diff} = a \cdot \mathit{duration}^b$, where a and b are the parameters to fit, to the results of each of the relative durations ($\times 1/2$, $\times 1/3$, $\times 1/4$), and as the plot shows, they are very similar. In fact, an analysis of covariance revealed no significant difference between them. This suggests that the error decreases

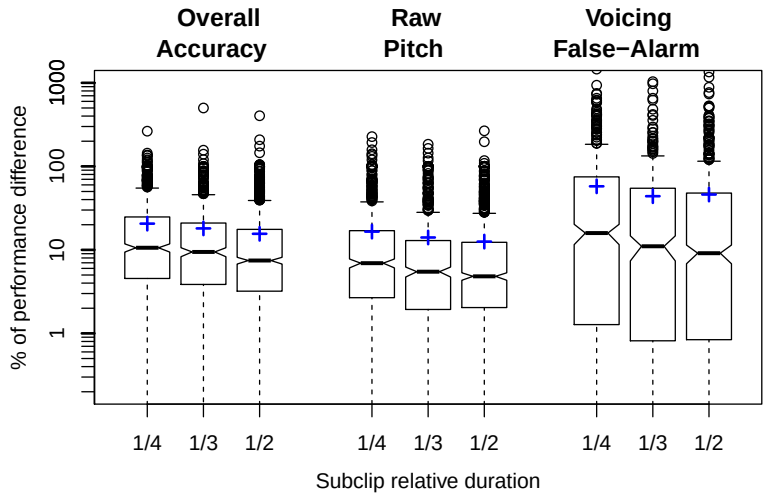

Figure 5.2: Relative performance differences between subclips and their corresponding $\times 1$ clips. The blue crosses mark the means of the distributions.

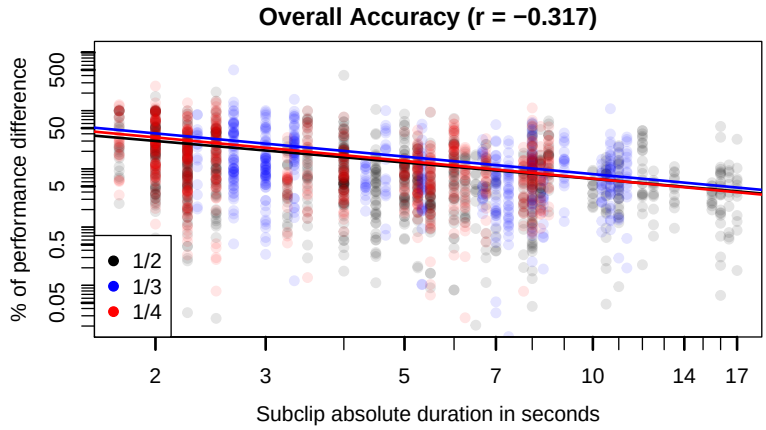

Figure 5.3: Relative performance differences with subclips as a function of subclip absolute duration.

as the clip duration increases, regardless of the duration of the full song. Recall that by error we mean the difference in the result obtained for an excerpt compared to the result obtained for the full song from which it was taken. This observation is discussed further in Section 5.5.

5.4 Collection size

Regardless of the effectiveness (evaluation) measure used, an AME experiment consists of evaluating a set of algorithms $\mathcal{A}$ using a set of songs $\mathcal{Q}$. Such an evaluation experiment can be viewed as fitting the following model:

$$\varphi_{as} = \overline{\varphi} + \overline{\varphi}_a + \overline{\varphi}_s + \varepsilon_{as} \tag{5.1}$$

where φ_{as} is the score of algorithm a for song s, $\overline{\varphi}$ is the grand average score of all possible algorithms over all possible songs, $\overline{\varphi}_a$ is the algorithm effect (the average deviation of algorithm a from the grand average $\overline{\varphi}$), $\overline{\varphi}_s$ is the song effect and ε_{as} is a residual modelling the particular deviation of algorithm a for song s. In our case, where we do not consider other effects such as annotators, this ε_{as} residual actually models the algorithm-song interaction effect: some algorithms are particularly better (or worse) for particular songs.

When a researcher carries out an AME evaluation experiment, they evaluate how well an algorithm performs for the set $\mathcal{Q}$ of songs, but ideally they want to *generalise* from the performance of that specific experiment to the average score the algorithm would obtain for the universe of all songs represented by the sample $\mathcal{Q}$, not just the sample itself. The reliability when drawing such general conclusions based on the observations on samples (test collections) can be measured with Generalisability Theory (GT; Bodoff, 2008; Brennan, 2001).

From the model in Equation 5.1 we can identify two sources of variability in the observed scores: actual performance differences among algorithms and difficulty differences among songs. Ideally, we want most of the variability in y_{as} to be due to the algorithm effect, that is, the observed effectiveness differences to be due to actual differences between algorithms and not due to other sources of variability such as songs, annotators, or specific algorithm-song interactions. Note that this does not mean a collection should not contain varied musical content. Ideally, we want an algorithm to work well for all types of musical material, and hence a varied collection in terms of content does not necessarily imply large performance variability due to the song effect. However, a small collection that contains songs with a great degree of variability (in terms of difficulty) is likely to result in performance variability that is dominated by the song effect and possibly by algorithm-song interactions (e.g. a certain algorithm is especially good for jazz but poor for rock), thus reducing our ability to claim that the observed differences between the algorithms can be generalised to the universe of all songs.

Using GT (Bodoff, 2008; Brennan, 2001), we can measure the proportion of observed variability that is due to actual differences between the algorithms. This proportion reflects the stability of the evaluation, and as such it is also a measure of efficiency: the higher the stability, the fewer the songs necessary to reliably evaluate the algorithms (Bodoff, 2008; Kanoulas & Aslam, 2009). GT does not only help evaluate the stability of past collections, but also estimate the reliability of yet-to-be created collections as a function of their size. However, the results of GT only hold if the original data used for the analysis is *representative* of the wider population of songs to which we want to generalise in the future.

5.4.1 Variance analysis and collection stability

In the model in Equation 5.1, the grand mean $\overline{\varphi}$ is a constant, and the other effects can be modelled as random variables with their own expectation and variance. As such, the variance of the observed scores is modelled as the sum of these variance components:

$$\sigma^2 = \sigma_a^2 + \sigma_s^2 + \sigma_{as}^2 \tag{5.2}$$

where σ_a^2 is the variance due to the algorithm effect, σ_s^2 is the variance due to the song effect, and σ_{as}^2 is the variance due to the algorithm-song interaction effect (the residual). This variance decomposition can be estimated by fitting a fully-crossed ANOVA model for Equation 5.1:

$$\widehat{\sigma}_{as}^2 = \mathrm{EMS}_{as} = \mathrm{EMS}_{residual} \ , \tag{5.3}$$

$$\widehat{\sigma}_a^2 = \frac{\mathrm{EMS}_a - \widehat{\sigma}_{as}^2}{|\mathcal{Q}|} \ , \tag{5.4}$$

$$\widehat{\sigma}_s^2 = \frac{\mathrm{EMS}_s - \widehat{\sigma}_{as}^2}{|\mathcal{A}|} \ , \tag{5.5}$$

where EMS_x is the expected Mean Square of component x. In practice, EMS_x is approximated by the Mean Square of component x as computed with the ANOVA model (Bodoff, 2008; Brennan, 2001). This step, of estimating the variance components based on an existing collection and set of algorithms, is referred to in GT terminology as a G-study. Using the estimates provided by the G-study (Equations 5.3–5.5) we can now estimate the proportion of variability due to the algorithm effect as per Equation 5.2. The stability of the evaluation can then be quantified with the dependability index Φ:

$$\Phi = \frac{\sigma_a^2}{\sigma_a^2 + \frac{\sigma_s^2 + \sigma_{as}^2}{|\mathcal{Q}'|}} \tag{5.6}$$

which measures the ratio between algorithm variance and the variance in absolute effectiveness scores (total variance; Bodoff, 2008; Brennan, 2001). This measure (which goes from 0 to 1) increases with the collection size (i.e. with an infinite number of songs all the observed variability would be due to algorithm differences; Kanoulas & Aslam, 2009). This second step is referred to as a D-study. Note that in the above equation the number of songs in the collection $|\mathcal{Q}|$ has been replaced by $|\mathcal{Q}'|$, representing an arbitrary collection size of our choice. This is where the power of GT lies: we can measure the current stability of the collection by setting $|\mathcal{Q}'| = |\mathcal{Q}|$, but we can also estimate how large the collection *should* be in order to reach a certain degree of stability by plugging in different values for $|\mathcal{Q}'|$ into the D-study (Equation 5.6) and computing the corresponding Φ value. An important question is then – what is an acceptable value for Φ? Urbano et al. (2013a) answer this question by mapping GT measures (including Φ) to well established data-based reliability indicators for a large set of TREC collections (Voorhees & Harman, 2005). Based on their findings, we take $\Phi = 0.9$ as an acceptable degree of stability. Another important aspect of GT that we must consider, is that the values of σ_a^2, σ_s^2 and σ_{as}^2 used to derive Φ are only estimates, obtained from the specific set of algorithms/songs used for the GT analysis. As such, there will be some variability in the GT indicators themselves. For this reason, for a more reliable analysis we should also compute a confidence interval for Φ. Then, we can use the lower bound of the interval (for a desired confidence, e.g 95%) to compute how many songs the collection should contain to give $\Phi \geq 0.9$ with 95% confidence (Urbano et al., 2013a). Further details on how to compute this confidence interval are provided in Brennan (2001). In the following section we use the GT analysis described here to evaluate the stability of the current MIREX collections.

5.4.2 Results

In Table 5.1 we show the estimated proportion of variability due to the algorithm, song and algorithm-song interaction effects. For these calculations we used the results of the MIREX campaign directly, combining the results of the five algorithms from MIREX 2010 and ten algorithms from MIREX 2011. In both years the same six test-collections were used for evaluation, so we can consider the grouping of algorithms from both years as a single larger evaluation round leading to a fully crossed experimental design. We also joined the three smaller collections into a single larger collection referred to as "04+05+08", which shall be discussed in Section 5.5. For each

evaluation measure and test collection, in addition to the estimated variance components $\widehat{\sigma}_a^2$, $\widehat{\sigma}_s^2$ and $\widehat{\sigma}_{as}^2$, we also provide the Φ score with a 95% confidence interval. In general, it can be seen that the estimated variance due to the algorithm effect $\widehat{\sigma}_a^2$ is much larger in the MIREX09 collections compared to the earlier collections. For the overall accuracy measure, the average variance due to the algorithm effect in the MIREX09 collections is 50%, whilst for the earlier collections it is just 18%, and as low as 11% for MIREX05. These differences indicate that generalisations of results based on the earlier collections are not very reliable, especially in the case of the MIREX05 and INDIAN08 collections, because a large part of the variability in the scores is due to the song characteristics rather than differences between the algorithms.

In Figure 5.4 we plot the estimated dependability index $\widehat{\Phi}$ as a function of the (hypothetical) number of songs used in each collection (log scaled). The point on each curve marks the value of $\widehat{\Phi}$ for the actual number of songs in each collection (cf. Table 5.1). Again we observe that the MIREX09 collections are considerably more stable than the earlier collections, especially MIREX05 and INDIAN08, where $\widehat{\Phi}$ is as low as 0.6 and the confidence interval is quite large. More interesting is the fact that the dependability index in the MIREX09 collections rapidly converges to 1, and there is virtually no appreciable difference between using all 374 songs in the collection or just 100: $\widehat{\Phi}$ would only drop from an average of 0.997 to 0.990, showing that most of the variability in performance scores would still be attributable to the algorithm effect. However, we must also consider the *content validity* of this collection, i.e. whether it is representative or not (Urbano, 2011). This will be discussed this in Section 5.5.

Finally, we use the GT analysis to estimate how many songs each collection should contain to give more stable results. In Table 5.2 we provide the estimated dependability index $\widehat{\Phi}$ and its 95% confidence interval for the overall accuracy measure (as in Table 5.1). But now, for each collection we also display the collection size $|\mathcal{Q}|$ and the estimated minimum collection size $|\widehat{\mathcal{Q}}|_{\Phi=0.9}$ for which $\Phi = 0.9$, and a 95% confidence interval on that estimate. As could be expected, the early collections are all smaller than ideally required (assuming our criterion for stability is $\Phi = 0.9$), an extreme case being INDIAN08. Somewhat surprisingly, the MIREX09 collections are an order-of-magnitude larger than what would be required to give stable results. This is most likely due to the high homogeneity of these collections, meaning that even with a relatively small number of songs the majority of the variance in the results would still be due to the algorithm effect.

Collection	Overall Accuracy				Raw Pitch				Voicing False Alarm			
	$\widehat{\sigma}_a^2$	$\widehat{\sigma}_s^2$	$\widehat{\sigma}_{as}^2$	$\widehat{\Phi}$	$\widehat{\sigma}_a^2$	$\widehat{\sigma}_s^2$	$\widehat{\sigma}_{as}^2$	$\widehat{\Phi}$	$\widehat{\sigma}_a^2$	$\widehat{\sigma}_s^2$	$\widehat{\sigma}_{as}^2$	$\widehat{\Phi}$
ADC2004	27%	27%	46%	.879 (.755–.953)	23%	28%	49%	.859 (.717–.944)	55%	21%	23%	.961 (.914–.985)
MIREX05	11%	47%	42%	.758 (.553–.901)	15%	54%	31%	.817 (.648–.926)	57%	20%	23%	.971 (.938–.989)
INDIAN08	16%	50%	34%	.600 (.240–.838)	24%	57%	19%	.721 (.374–.895)	70%	13%	16%	.950 (.852–.982)
04 + 05 + 08	16%	39%	45%	.909 (.823–.965)	16%	43%	41%	.912 (.829–.966)	56%	21%	23%	.986 (.971–.995)
MIREX09 -5dB	40%	23%	37%	.996 (.993–.998)	40%	24%	35%	.996 (.993–.998)	82%	5%	13%	.999 (.999–1.00)
MIREX09 0dB	52%	20%	28%	.998 (.995–.999)	50%	20%	31%	.997 (.995–.999)	81%	5%	14%	.999 (.999–1.00)
MIREX09 +5dB	58%	17%	26%	.998 (.996–.999)	48%	18%	34%	.997 (.995–.999)	83%	4%	14%	.999 (.999–1.00)

Table 5.1: Variance components and $\widehat{\Phi}$ score (with 95% confidence interval) for the overall accuracy, raw pitch accuracy and voicing false alarm rate evaluation measures and all six MIREX collections plus the joint 04+05+08 collection.

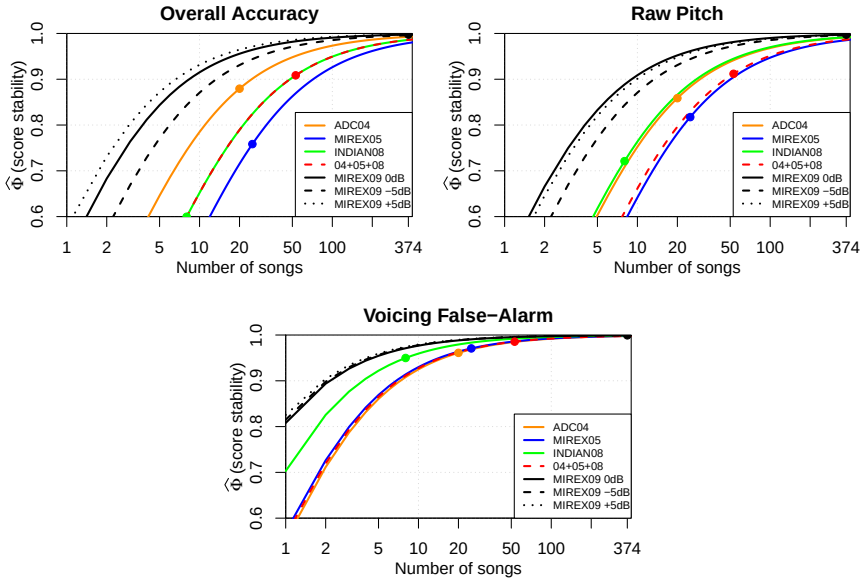

Figure 5.4: Dependability index $\widehat{\Phi}$ as a function of the number of songs for the following evaluation measures: overall accuracy (top left), raw pitch accuracy (top right) and voicing false alarm rate (bottom). The points mark the actual number of songs per collection.

Collection	$\widehat{\Phi}$	$\lvert\mathcal{Q}\rvert$	$\lvert\widehat{\mathcal{Q}}\rvert_{\Phi=0.9}$
ADC2004	.879 (.755–.953)	20	25 (9–59)
MIREX05	.758 (.553–.901)	25	72 (25–182)
INDIAN08	.600 (.240–.838)	8	49 (14–229)
04 + 05 + 08	.909 (.823–.965)	53	49 (18–103)
MIREX09 -5dB	.996 (.993-.998)	374	14 (6–26)
MIREX09 0dB	.998 (.995-.999)	374	9 (4–16)
MIREX09 +5dB	.998 (.996-.999)	374	7 (3–13)

Table 5.2: $\widehat{\Phi}$ score (with 95% confidence interval), collection size $\lvert\mathcal{Q}\rvert$ and estimated minimum collection size $\lvert\widehat{\mathcal{Q}}\rvert_{\Phi=0.9}$ (with 95% confidence interval) to get $\Phi = 0.9$ based on the overall accuracy measure, for all six MIREX collections plus the joint 04+05+08 collection.

5.5 Discussion

Starting with the annotation offset issue, we note that there are two crucial parameters that must be fixed in order to prevent this problem: the precise time of the first frame, and the hop size. Since 2005, all the annotations use a hop size of 10 ms, and all algorithms are required to use this hop size for their output. However, the exact time of the first frame has not been explicitly agreed upon by the community. When the short-time Fourier transform (or any other transform which segments the audio signal into short frames) is used, it is common practice to consider the time-stamp of each frame to be the time exactly at the middle of the frame. Thus, for the first frame to start exactly at time zero, it must be centred on the first sample of the audio (filling the first half of the frame with zeros). Nonetheless, whilst this is common practice, it is not strictly imposed, meaning algorithms and annotators might, rather than centre the first frame on the first sample, start the frame at this sample. In this case, the frame will not be centred on time zero, but rather on an arbitrary time which depends on the length of the frame. Since different algorithms and annotations use different frame sizes, this scenario could lead to a different fixed offset between every algorithm and every annotation, leading to a systematic error in the evaluation.

In terms of clip duration, we saw that there is a clear correlation between the relative duration of the clip (compared to the full song) and evaluation error, suggesting that performance based on clips might not really predict performance on full songs. However, Figure 5.3 suggests that this correlation is independent of the actual duration of the full song. That is, there might be a duration threshold of x seconds for which the observed performance on clips does predict the performance on full songs (within some error rate), no matter how long they are. Whilst counter-intuitive at first, this result does somehow agree with general statistical theory. How large a sample needs to be in order to reliably estimate unknown parameters of the underlying population, is independent of how large the population actually is, as long as the sample is *representative* of the population. This usually requires to sample randomly or follow other techniques such as systematic or stratified sampling. For AME evaluation it does not make sense to randomly sample frames of a song, but the results suggest that there might be a sampling technique such that audio clips, if selected appropriately, can be representative of the full songs.

Regarding the collection size, we observed that the earlier ADC04, MIREX05 and INDIAN08 collections are unstable because a large proportion of the

variability in the observed performance scores is due to differences in song difficulty rather than differences between the algorithms. As such, results from these collections alone are expected to be unstable, and therefore evaluations that rely solely on *one* of these collections are not very reliable. In Tables 5.1 and 5.2 (and Figure 5.4) we see that by joining these collections into a single larger one ("04+05+08") the evaluation results are considerably more stable ($\widehat{\Phi} > 0.9$ for all three measures). Thus, for the sake of evaluation stability it would be beneficial to fuse the three into a single collection in future evaluations. On the other hand, we saw that the MIREX09 collections are in fact much larger than necessary: about 25% of the current songs would suffice for results to be highly stable and therefore generalise to a wider population of songs. However, all the MIREX09 music material consists of Chinese karaoke songs with non-professional singers, and therefore we should expect the results to generalise to *this* population of songs, but not to the general universe of *all* songs (i.e. everything other than Chinese karaoke music).

5.6 Conclusion

In this chapter we analysed the reliability of the evaluation of melody extraction algorithms, as carried out in the MIREX AME task. Three main factors were studied: ground truth annotations, clip duration and collection size. We demonstrated how an offset between the ground truth and an algorithm's output can significantly degrade the results, the solution to which is the definition and adherence to a strict protocol for annotation. Next, it was shown that the clips currently used are too short to predict performance on full songs, stressing the need to use complete musical pieces. It was also shown that results based on only one of the ADC04, MIREX05 or INDIAN08 collections are not reliable due to their small size, whilst the MIREX09 collections, though more reliable, do not represent all the real-world musical content to which we wish to generalise. The above demonstrates that whilst the MIREX AME evaluation task is an important initiative, it currently suffers from problems which require our attention. As a solution, we propose the creation of a new and open test collection through a joint effort of the research community. If the collection is carefully compiled and annotated, keeping in mind the issues mentioned here, it should, in theory, solve all of the aforementioned problems that current AME evaluation suffers from. Furthermore, we could consider the application of low-cost evaluation methodologies that dramatically reduce the

annotation effort required (Urbano & Schedl, 2013). To this end, the author of this thesis has launched the Audio Melody Extraction Annotation Initiative[2]. At the time of writing this dissertation the initiative is still at its early stages, but we hope that it will be successful in producing a new, high quality dataset for algorithm development and evaluation. In the future it would also be worth studying the appropriateness of the evaluation measures themselves, as well as the accuracy of the manual ground truth annotations.

Finally, it is important to note that the conclusions of this chapter do not imply that the MIREX results are not valid. What they tell us is that we must be cautious if we wish to generalise from the MIREX results to the universe of all songs we would like to apply our algorithms to. As suggested above, one way to obtain more generalisable results would be to increase the collection size. Whilst this would allow us to make stronger statements regarding how well an algorithm performs, there is still an important question that would remain unanswered – how good is *good enough*? As noted in Chapter 1, the end-goal of using a melody extraction algorithm in most cases will be the development of a system which exploits the extracted melody sequences. Whether the accuracy of an algorithm is sufficient for a certain type of application (for example music retrieval based on melodic similarity) or not is something that the MIREX evaluations on their own cannot tell us. To answer this question, we have to actually build and evaluate such systems. This is the topic of the next chapter of this thesis.

[2] http://ameannotationinitiative.wikispaces.com

Chapter 6

Applications

6.1 Introduction

In Chapter 4 we saw that the melody extraction algorithm proposed in this thesis obtains the highest mean overall accuracy for the MIREX collections to date, 75%. In Chapter 5 we noted that despite the good results, due to the size of the collections we cannot be certain that this performance will translate to larger, untested collections. Importantly, we noted that even if we get the same accuracy for real-world collections, the question of whether it is sufficient for developing applications which exploit the output of our algorithm remains unanswered.

In this chapter we aim to answer this question by developing and evaluating a set of prototype systems which are based on the output of our model-free melody extraction algorithm. We consider a variety of application domains for the algorithm, including music similarity and retrieval, music classification and computational music analysis, particularly of non-Western music traditions. The collections used for evaluating the applications discussed in this chapter are all considerably larger than the collections used for evaluation thus far in the thesis, and they all contain real-world music from a variety of genres and music traditions. A caveat of these collections is that their melodies have not been annotated, meaning we cannot assess the effect of extraction accuracy on the overall performance of the system. Nonetheless, by demonstrating the plausibility of systems that are based on the output of our algorithm, we are able to show that even though there is still room for improving the algorithm itself, it is already sufficiently accurate to be exploited for both research and end-user applications.

In Section 6.2 we study the use of our algorithm for music similarity and retrieval, specifically for two popular end-user applications: version identification (cover song detection) and query-by-humming (Salamon et al., 2013b). In Section 6.3 we use the algorithm to compute a set of melodic features which are then exploited for automatic musical genre classification (Salamon et al., 2012c). In Section 6.4 we demonstrate how the algorithm can be used for computational music analysis, in this case for automatic tonic identification in Indian classical music (Salamon et al., 2012a). Unlike the previous applications, here we exploit the salience function rather than the final melody output of the algorithm. At the end of Section 6.4 we also briefly discuss an extension of the algorithm which incorporates the final melody output as well to improve identification accuracy. Finally, in Section 6.5 we demonstrate how the complete algorithm can be combined with a method for note segmentation and labelling to produce a fully automatic melodic transcription system, where we focus on flamenco music for evaluation. Final conclusions for the chapter are provided in Section 6.6.

6.2 Music similarity and retrieval

6.2.1 Introduction

Music similarity is a key aspect in the development of music retrieval systems (Pachet, 2005), and developing automatic methods for quantifying music similarity addresses part of a more general problem: making sense of digital information (Ratzan, 2004). Music similarity is, however, an ambiguous term. Apart from involving different musical facets such as instrumentation, tonality or rhythm, it also depends on cultural (or contextual) and personal (or subjective) aspects (Harwood, 1976; Lynch et al., 1990). There are many factors involved in music similarity judgements, and some of them are difficult to measure (Berenzweig et al., 2004).

To assess the similarity between music documents, some MIR researchers have devoted their efforts to the related task of *version identification*. Importantly, in contrast to music similarity, the relation between versions is context-independent and can be objectively measured (Serrà, 2011). In addition, research on version identification can yield valuable clues for modelling music similarity. This task, of automatically detecting versions of the same musical piece, has received much attention from the research community over recent years (see Serrà et al. (2010) for a survey). Potential applications range from the detection of copyright violations on websites

such as YouTube, to the automation of computational analyses of musical influence networks (Bryan & Wang, 2011). Version identification on its own also represents an attractive retrieval task for end users.

Systems for the automatic detection of versions exploit musical facets which remain mostly unchanged across different renditions, primarily tonal information (Serrà et al., 2010). In this context, the term tonality refers, in a broad sense, to "the systematic arrangement of pitch phenomena and relations between them" (Hyer, 2013). Perhaps the most common tonal representation for version identification is chroma. Chroma features (also called pitch class profiles) represent, in polyphonic music signals, the relative intensity of each of the 12 semitones in an equal-tempered chromatic scale, discarding octave information. As such, chroma features are related to harmony, understood as "the combining of notes simultaneously, to produce chords, and successively, to produce chord progressions" (Dahlhaus, 2013). Common techniques for comparing sequences of chroma features include dynamic time warping and simple cross-correlation (Serrà et al., 2010). Another tonal representation that has been considered for version identification is the predominant melody, either by attempting to fully transcribe it (Tsai et al., 2008), or by using it as a mid-level representation for computing similarity (Marolt, 2008). Melodic representations have also been widely used for related tasks such as query-by-humming (Dannenberg et al., 2007) or music retrieval using symbolic data (Typke, 2007).

Whilst good results have been achieved using single music representations (in particular harmony as represented by chroma features; Serrà et al., 2010), some recent studies suggest that version identification could be improved through the combination of different musical cues (Foucard et al., 2010; Liem & Hanjalic, 2009; Serrà, 2011). However, not much research has been carried out in this direction. One of the first studies to automatically extract features derived from different music representations for version identification was conducted by Foucard et al. (2010), in which a source separation algorithm was used to separate the melody from the accompaniment. The authors then compared the performance of a version identification system using the melody, the accompaniment, the original mix, and their combination, by employing different fusion schemes. The study showed that considering different information modalities (i.e. main melody and accompaniment) is a promising research direction, but also noted the intrinsic limitation of simple fusion schemes whose capabilities seemed to be limited to merging modalities that carry more or less the same type of information. In the work of Ravuri & Ellis (2010), the task of

detecting musical versions was posed as a classification problem, and different similarity measures were combined to train a classifier for determining whether two musical pieces were versions or not. However, only chroma features were used to derive these similarity measures. Therefore, they were all in fact accounting for the same musical facet: the harmony.

In Sections 6.2.2–6.2.3 we study the application of our melody extraction algorithm for the task of version identification. To do so, we explore and compare the use of three related yet different tonal representations of music: melody, bass line and harmony. To extract the melody we use the algorithm described in Section 4.2 of this thesis (Salamon & Gómez, 2012). The bass line is extracted using a modified version of the algorithm, described in Section 6.2.2. Harmony is represented by means of chroma features (Gómez, 2006b), which have already been used successfully in state-of-the-art version identification systems (Serrà et al., 2010). Beyond comparing identification performance for each of the three tonal representations separately, we also study their combination. For this we use the power of a standard classification approach, similar to Ravuri & Ellis (2010). In addition, we compare a number of classification algorithms and assess their ability to fuse the information coming from the three different representations.

As mentioned earlier, a task very much related to version identification is that of query-by-humming (QBH). A QBH system allows users to search for songs by singing or humming part of the melody. As such, QBH can be considered a special case of version identification in which the version used as a query is produced by the user (rather than an artist) and contains only melody information. The task has received much attention from the research community, both because of the challenges it presents in computational music similarity and because of its attractive potential application for end users (see Dannenberg et al. (2007) for a comparative evaluation and Kotsifakos et al. (2012) for a more recent survey). An important problem in the creation of QBH systems is the generation of a melody database (song index) against which the sung queries are to be compared. Until recently, the lack of reliable melody extraction algorithms meant that most research effort was focused on matching queries against symbolic databases (e.g. MIDI files; Dannenberg et al., 2007). Whilst it is possible to find MIDI versions of many songs on the Internet, such an approach will always be limited since it is not feasible to generate (i.e. transcribe) MIDI files manually for very large music collections, not to mention all the new music that will be composed in the future. An alternative solution that has been proposed is to match queries against other queries, as performed by ser-

vices such as SoundHound[1] and Tunebot (Pardo et al., 2008). Whilst this avoids the need for manual transcription, the approach still suffers from the same "cold start" problem – a song "does not exist" until someone records it, and so the need for manual labour remains. In light of this, a fully automated solution in which the melody database can be generated automatically is highly attractive. Audio-to-audio QBH algorithms have been proposed by, e.g., Duda et al. (2007); Ryynänen & Klapuri (2008b); Song et al. (2002). However, results indicate there is still much work to be done before these systems perform as well as audio-to-symbolic QBH. For example, the approach proposed by Ryynänen & Klapuri (2008b) obtains a Mean Reciprocal Rank (MRR; cf. Section 6.2.4) of 0.57 for a database of 427 audio songs compared to 0.91 for a database of 2048 symbolic songs.

In Section 6.2.4 we explain how our proposed version identification approach can be adapted into a fully automated audio-to-audio QBH system with very few modifications. We evaluate the system using a relatively large collection of full-length commercial music, and show that it can successfully retrieve target songs from a collection of polyphonic audio given sung queries, where both the query and target-melody representations are obtained using our melody extraction algorithm. In this way we show that even though the melody sequences extracted by our algorithm are likely to contain some errors, they are sufficiently accurate for developing similarity-based retrieval applications.

The structure of the remainder of this section (6.2) is as follows: in Section 6.2.2 we describe the music representations compared for version identification, how we compute descriptors to represent them, the computation of descriptor-sequence similarity, and our approach for descriptor fusion. In Section 6.2.3 we start by describing our evaluation strategy for version identification, including the music collection and evaluation measures used to assess the retrieval accuracy. This is followed by the results of the evaluation for both individual descriptors and descriptor fusion, including a detailed discussion of the results. In Section 6.2.4 we explain how the proposed approach can be applied to the related task of query-by-humming. We describe the test collections, evaluation measures and queries used to evaluate the approach and discuss the retrieval results. The contributions of Section 6.2 are summarised in (sub)Section 6.2.5.

[1] http://www.soundhound.com/

6.2.2 Methodology

In the following subsections we present our approach for (dis)similarity-based music retrieval. We start by describing the different tonal representations extracted from the audio signal, which are based on the melody, bass line and harmonic progression of a musical piece. For melody and bass line, we define a representation abstraction process for removing performance-specific information that may hinder the matching procedure. Next, we explain how song matching is performed using a local alignment algorithm based on nonlinear time-series analysis. Finally, we describe our approach for integrating the different music representations for version identification using a classification technique.

Melody and bass line representation

To extract a representation of the melody from the audio signal we use the model-free melody extraction algorithm presented in Section 4.2 (Salamon & Gómez, 2012). To extract the bass line, we adapt the algorithm so that it extracts the bass line instead of the melody. Using the same algorithm for melody and bass line extraction (with different parametrisation) was first proposed by Goto (2004), the underlying assumption being that the bass line can be treated as a low-frequency melody: the pitch of the bass line has a harmonic structure, it is (often) the most predominant instrument in the low-frequency region, and tends to have temporally continuous trajectory. Thus, in order to adapt our algorithm for bass line extraction, we make the following adjustments: instead of applying an equal loudness filter (which attenuates low frequency content), we apply a low-pass filter with a cutoff frequency of 261.6 Hz, as proposed by Goto (2004). The window size is increased to 185 ms, since for the bass we require more frequency resolution. The salience function is adjusted to cover a range of two octaves from 27.5 to 110 Hz. As before, the salience peaks are grouped into pitch contours. However, since we do not expect other instruments to compete for predominance in the bass frequency range, the detailed contour characterisation and filtering used for melody extraction is less important in the case of bass line extraction. Therefore, the bass line is selected directly from all the generated contours by choosing at each frame the pitch value belonging to the contour with the highest total salience $C_{\Sigma s}$

Representation abstraction

Once the melody and bass line sequences are extracted, we must choose an adequate representation for computing music similarity or, in the case of this study, a representation for retrieving versions of the same musical piece. Since the matching algorithm can handle transposition, a first guess might be to use the extracted representation as is, i.e. to compare the f_0 sequences directly. However, our initial experiments showed that this (somewhat naïve) approach is unsuccessful.

When considering the task of version identification, we must take into consideration what kind of musical information is maintained between versions, and what information is subject to change. In the case of the melody, we can expect the general melodic contour to be maintained. However, more detailed performance information is likely to change between versions (Serrà et al., 2010). Besides changing the key and tempo in which the melody is sung (or played), performers might change the octave in which some segments of the melody are sung to adjust them to their vocal range. More importantly, the use of expressive effects (such as ornaments, glissando and vibrato) will obviously vary across versions. Overall, this means we should aim for a representation which abstracts away performance-specific information and details, whilst maintaining the basic melodic tonal progression. To this effect, we defined the following types of information abstraction:

- Semitone abstraction: quantise pitch information into semitones. This will help in removing some local expressive effects.
- Octave abstraction: map all pitch information onto a single octave. This will help in removing potential octave changes of the melody within the piece.
- Interval abstraction: replace absolute pitch information with the difference between consecutive pitch values (delta values). This may provide robustness against key changes.

Before applying any of the aforementioned abstraction steps, the frequency values of the f_0 sequences were converted from Hz to cents (cf. Section 2.4.1), so that pitch is measured in a perceptually meaningful way. We then ran initial matching experiments comparing the different degrees of abstraction applied to melody sequences: none, semitone, interval, interval+semitone, and semitone+octave (by definition, the interval and octave abstractions

are not compatible). For these experiments we used a collection of 76 songs (which is described in Section 6.2.3), and evaluated the results as detailed in the same section (6.2.3). We found that the results obtained when applying the semitone+octave abstraction were considerably better than the other types of abstraction, obtaining a mean average precision (MAP; cf. Section 6.2.3) of 0.73, compared to 0.26–0.28 for all other abstractions considered. Perhaps not surprisingly, we note that this abstraction process is quite similar to the one applied for computing chroma features (described below). In particular, the observations above suggest that octave information can be quite detrimental for the task of version identification. These findings are in concurrence with Müller et al. (2009), who use a similar abstraction process to robustly compare f_0 sequences of monophonic folk song recordings to MIDI references in order to perform automatic segmentation. For the remainder of the study we use the semitone+octave abstraction for both the melody and bass line descriptors.

The exact abstraction process is as follows: first, all frequency values are converted into cents. Then, pitch values are quantised into semitones, and mapped onto a single octave. Next, we reduce the length of the sequence (whose original hop size is 2.9 ms), by summarising every 150 frames as a pitch class histogram. The contribution of each frame to the histogram is weighted by the salience of the melody at that frame, as determined by the melody extraction algorithm (the contribution is weighted by the total salience $C_{\Sigma s}$ of the contour from which the melody pitch was taken). This produces a shortened sequence where each frame is a 12-bin vector representing the distribution of the pitch classes of the melody over roughly half a second. This window length has been reported to be suitable for version identification by several authors, see for example Liem & Hanjalic (2009); Müller & Ewert (2010); Serrà et al. (2010). The motivation for the summary step is twofold: first, it reduces the sequence length and therefore reduces the computation time of the matching algorithm. Second, it reduces the influence of very short pitch changes which are more likely to be performance specific (e.g. ornamentations). Further evidence of the benefits of coarsening pitch-related time series for similarity-based audio retrieval is provided by Müller et al. (2005). Finally, the vector of each frame is normalised by the value of its highest bin. The steps of the representation abstraction are depicted in Figure 6.1 for a melody and in Figure 6.2 for a bass line.

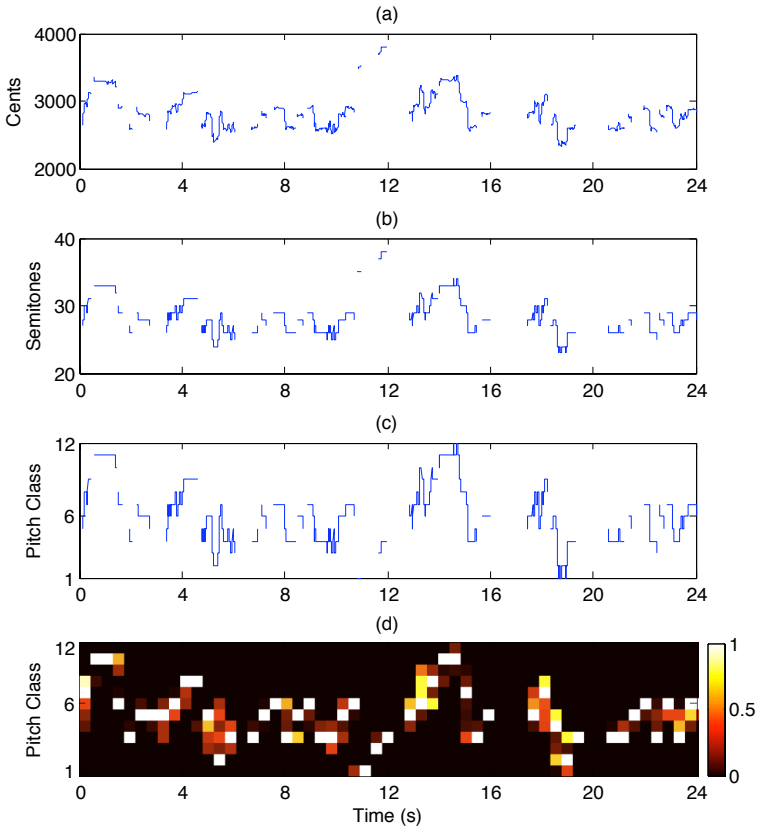

Figure 6.1: Melody representation abstraction process: (a) melody pitch in cents, (b) quantised into semitones, (c) mapped onto a single octave, (d) summarised as a pitch class histogram and normalised.

Harmony representation

To represent harmony, we compute the sequence of harmonic pitch class profiles (HPCP; Gómez, 2006a,b), a specific chroma feature implementation. The HPCP is derived from the frequency-dependent energy in a given range (typically from 50 to 5000 Hz) in short-time spectral representations of the audio signal (e.g. 100 ms; frame-by-frame extraction). The energy is mapped into an octave-independent histogram representing the relative intensity of each of the 12 semitones of the equal-tempered chromatic scale (12 pitch classes). To normalise with respect to loudness, the histogram is divided by its maximum value, leading to values between 0 and 1. Three important preprocessing steps are applied during the computation

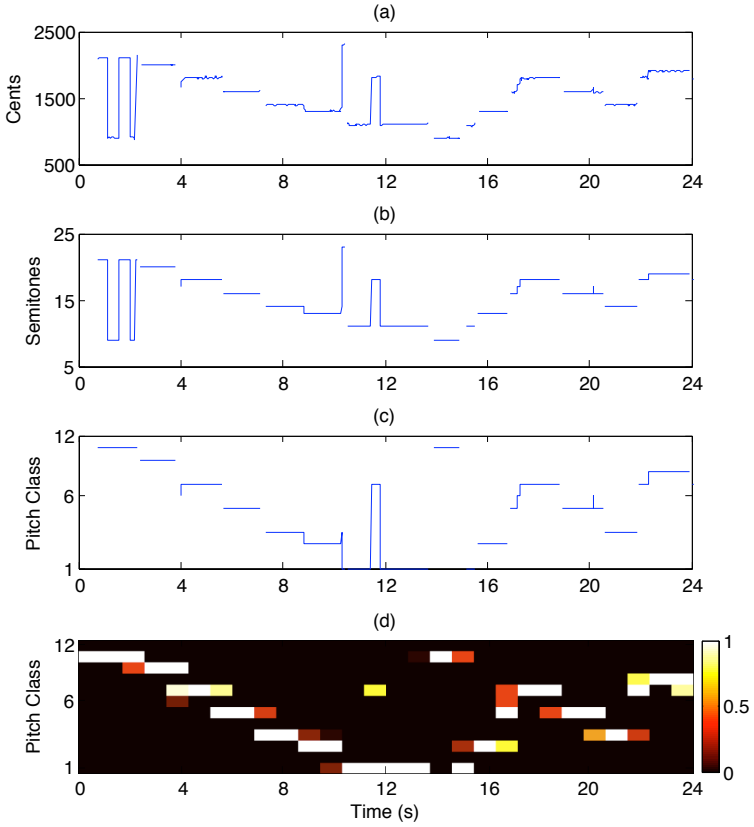

Figure 6.2: Bass line representation abstraction process: (a) bass line pitch in cents, (b) quantised into semitones, (c) mapped onto a single octave, (d) summarised as a pitch class histogram and normalised.

of the HPCP: tuning estimation, sinusoid extraction and spectral whitening (Gómez, 2006a). This means the HPCP is tuning-frequency independent and robust to noise and changes in timbre, which makes it especially attractive for version identification.

Chroma features are a standard tool in music information research, and the HPCP in particular has been shown to be a robust and informative chroma feature implementation (Gómez, 2006a; Ong et al., 2006; Serrà et al., 2008). For more details we refer the interested reader to Gómez (2006a) and references therein. For the purpose of this study, and in order to facilitate the comparison with previous work on version identification, the HPCP is computed using the same settings and parameters used by Serrà et al. (2009).

Matching

For deriving a similarity measure of how well two sequences match we employ the $Q_{\max}$ method (Serrà et al., 2009). This is a dynamic programming algorithm which computes a similarity measure based on the best subsequence partial match between two time series. Therefore, it can be framed under the category of local alignment algorithms. Dynamic programming approaches using local alignment are among the best-performing state-of-the-art systems for version identification (Serrà et al., 2010), and have also been used extensively for melody-based retrieval (Dannenberg et al., 2007).

The $Q_{\max}$ algorithm is based on general tools and concepts of nonlinear time-series analysis (Kantz & Schreiber, 2004). Therefore, since the algorithm is not particularly tied to a specific type of time series, it can be easily used for the comparison of different (potentially multivariate) signals. Furthermore, the $Q_{\max}$ method has provided the highest MIREX accuracies in the version identification task to date, using only HPCPs (Serrà et al., 2009). Therefore, it is a very good candidate for testing how melody and bass line compare to HPCPs, and for deriving competitive version similarity measures to be used in our fusion scheme.

Given a music collection containing various sets of covers, we use the $Q_{\max}$ algorithm to compute the similarity, or in the case of our method, the dissimilarity, between every pair of songs in the collection. The resulting pairwise dissimilarities are stored in a dissimilarity matrix which can then be used either to evaluate the performance of a single descriptor (as explained in Section 6.2.3), or for descriptor fusion as described next.

Fusing descriptors

In addition to evaluating each tonal representation separately for version identification and comparing the performance of melody-based version retrieval to alternative approaches (namely bass line or harmony-based retrieval), another goal of this study is to see whether there is any information overlap between these representations, and whether the results for this task can be improved by combining them. To this end, we propose a classification approach similar to that of Ravuri & Ellis (2010) – each descriptor is used to calculate a dissimilarity matrix between all query-target pairs as described above (4,515,625 pairs in total for the collection used in this study). Every query-target pair is annotated to indicate whether the query and target are versions or not. We then use five different balanced subsets of 10,000 randomly selected query-target pairs to train a classifier

for determining whether two songs are versions of the same piece. The feature vector for each query-target pair is three-dimensional, and contains the dissimilarities produced by the matching algorithm using each of the three representations: melody, bass line and harmony (feature columns are linearly normalised between 0 and 1 prior to classification). In this way we can study different combinations of these descriptors, and most importantly, rather than imposing a simple fusion scheme, the combination of different descriptors is determined in an optimal way by the classifier itself. The only potential limitation of the proposed approach is our employment of a late-fusion strategy (as opposed to early-fusion). Nonetheless, in addition to being straightforward, previous evidence suggests that late-fusion provides better results for version identification (Foucard et al., 2010). Since the modalities employed in this study are different from the ones employed by Foucard et al. (2010), the preferability of a late-fusion strategy over early-fusion should still be validated, a matter which we leave for future work.

The classification is performed using the Weka data mining software (Hall et al., 2009). We compare five different classification algorithms: random forest, support vector machines (sequential minimal optimisation (SMO) with a polynomial kernel), simple logistic regression, K* (instance-based), and Bayesian network (Witten & Frank, 2005). For all classifiers we use the default parameter values provided in Weka. By comparing different classifiers we are able to assess which classification approach is the most suitable for our task. Furthermore, by verifying that any increase (or decrease) in performance is consistent between classifiers, we ensure that the improvement is indeed due to the descriptor fusion and not merely an artefact of a specific classification technique.

6.2.3 Version identification

In this section we describe how the tonal representations, matching technique and fusion approach are evaluated for version identification. We start by describing the music collection and measures we use for the evaluation, followed by the evaluation results and an in-depth discussion of the results. The complete matching process for version identification, using either a single tonal representation or descriptor fusion, is depicted by the block diagram in Figure 6.3.

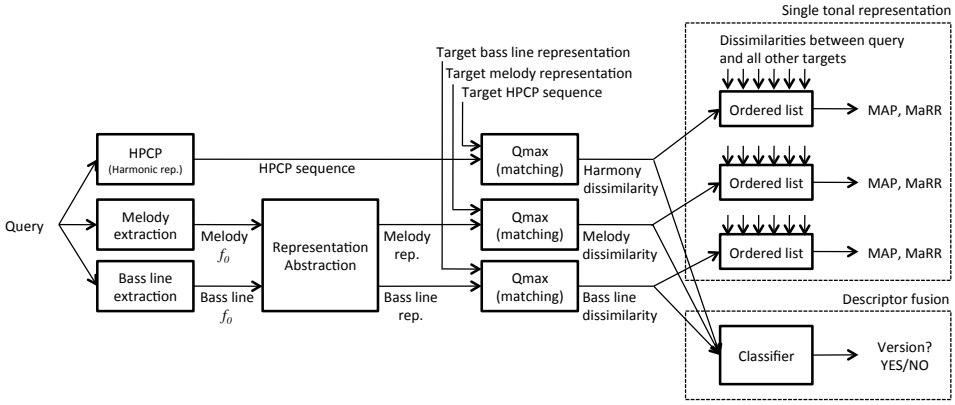

Figure 6.3: Matching process for version identification using either a single tonal representation (top right corner) or descriptor fusion (bottom right corner).

Music collection

To evaluate the performance of our method (using either a single music representation or the descriptor fusion strategy), we use a music collection of 2125 songs (Serrà et al., 2012). The collection includes 523 version sets (i.e. groups of versions of the same musical piece) with an average set cardinality of 4.06. The collection spans a variety of genres including pop, rock, electronic, jazz, blues, world, and classical music. We note that this collection is larger than the collection used in the MIREX version identification task[2], and as such contains a greater variety of artists and styles.

For training the parameters of the $Q_{\max}$ matching algorithm, a small subset of 76 songs from the full collection was used. This 76-song collection was also used for the preliminary experiments on information abstraction outlined in Section 6.2.2. Importantly, we made sure that all songs in this subset have a main melody (and all but 3 have a clear bass line). The full collection, on the other hand, includes versions where there is no main melody (e.g. minus-one versions of jazz standards) or no bass line (e.g. singing voice with acoustic guitar accompaniment only). In a manually annotated random sample of 300 songs from the full collection, 88.7% had a melody, 89.7% a bass line and 95.3% included harmony (the confidence intervals for the statistics as representative of the full collection with 95% confidence are 3.3, 3.2 and 2.2 respectively). Whilst we can expect this difference to affect the relative performance of the system when using the melody or bass line representa-

[2] http://www.music-ir.org/mirex/wiki/Audio_Cover_Song_Identification

tions, the statistics are representative of a real-world music collection and, as such, the results for this collection will reflect those we would expect to obtain in a real-world scenario.

Evaluation measures

The dissimilarity matrix produced by each descriptor can be used to generate an ordered list of results for each query. The relevance of the results (ideally versions of a query should all appear at the top of the list) can then be evaluated using standard information retrieval metrics, namely the mean average precision (MAP) and the mean reciprocal rank (MRR; Manning et al., 2008).

Given a query song u, we use the values of the dissimilarity matrix to order the remaining songs in the collection by ascending dissimilarity (compared to u), resulting in an ordered list Λ_u of size $U-1$ (U being the size of the evaluation collection including u). Suppose that u belongs to a version set of cardinality $\mathcal{V}_u+1$, i.e. u is one of $\mathcal{V}_u+1$ versions of the same piece included in the collection. Then, the average precision $\overline{\psi}_u$ is calculated as:

$$\overline{\psi}_u = \frac{1}{\mathcal{V}_u} \sum_{k=1}^{U-1} \psi_u(k)\Gamma_u(k), \tag{6.1}$$

where $\psi_u(k)$ is the precision of the sorted list Λ_u at rank k,

$$\psi_u(k) = \frac{1}{k} \sum_{i=1}^{k} \Gamma_u(i), \tag{6.2}$$

and $\Gamma_u(j)$ is a relevance function such that $\Gamma_u(j)=1$ if the song at rank j is a version of the query song u, and $\Gamma_u(j)=0$ otherwise. Note that $\overline{\psi}_u$ ranges from 0 to 1, where $\overline{\psi}_u=1$ if all the versions of u in the collection are positioned at the top of the list Λ_u (best case scenario), and $\overline{\psi}_u \approx 0$ if they are all positioned towards the end of the list (worst case scenario). Finally, the mean average precision (MAP) is given by computing the mean value of $\overline{\psi}_u$ across all queries $u=1\ldots U$:

$$\text{MAP} = \frac{1}{U} \sum_{u=1}^{U} \overline{\psi}_u. \tag{6.3}$$

The mean reciprocal rank (MRR) is typically used to evaluate retrieval systems in which there is only one target in the database which matches

the query (i.e. $\mathcal{V}_u = 1$). Given a set of U queries, the MRR (which also ranges from 0 in the worst case to 1 in the best case) is defined as:

$$\text{MRR} = \frac{1}{U}\sum_{u=1}^{U}\frac{1}{r_u} \tag{6.4}$$

where r_u is the rank of the correct target song for query u in the ordered list Λ_u. Since in the case of version identification it is often the case that $\mathcal{V}_u > 1$, we cannot compute the MRR (which assumes there is only one correct answer) directly. Hence, we define a measure we refer to as the mean *averaged* reciprocal rank (MaRR). Given a query u, we defined the averaged reciprocal rank aRR_u as the mean of the reciprocal ranks of all the targets in the result list Λ_u that are versions of u:

$$\text{aRR}_u = \frac{1}{\mathcal{V}_u}\sum_{i=1}^{\mathcal{V}_u}\frac{1}{r_{i\sim u}} \tag{6.5}$$

where $r_{i\sim u}$ is the rank of the i^{th} target which is a version of u. The MaRR is then the mean aRR over all queries:

$$\text{MaRR} = \frac{1}{U}\sum_{u=1}^{U}\text{aRR}_u. \tag{6.6}$$

Note that unlike the MAP and MRR which range from 0 to 1, the range of the MaRR depends on the number of versions each query has in the collection. For example, if a query has 3 target versions in the collection ($\mathcal{V}_u = 3$), the highest possible aRR will be $(\frac{1}{1} + \frac{1}{2} + \frac{1}{3})/3 = 0.61$. Since the average version-set cardinality in our collection is approximately 4, we can consider this value (0.61) as a rough upper-bound for the MaRR. Both the MAP and MaRR[3] measures are a common choice for assessing the accuracy of version identification systems based on a single information source (Serrà et al., 2010).

Since we use classification to fuse different information sources (dissimilarities based on different descriptors), an alternative evaluation approach is required to evaluate the results obtained using descriptor fusion. Here, the

[3]In the version identification literature the MaRR is often referred to simply as MRR, which could potentially lead to confusion if the authors do not explicitly explain how they compute the measure. For this reason in this dissertation we have decided to clearly distinguish between the two measures by giving the MaRR a unique abbreviation.

Feature	MAP	MaRR
Melody	0.732	0.422
Bass line	0.667	0.387
Harmony	0.829	0.458

Table 6.1: Results for single tonal representation (76 songs).

results produced by each classifier are evaluated in terms of classification accuracy (%) using 10-fold cross validation, averaged over 10 runs per classifier. The classification is carried out using the balanced subsets of 10,000 randomly selected query-target pairs mentioned in Section 6.2.2. We repeat the evaluation process for 5 such subsets (non-overlapping), and average the results over all subsets. Note that we ensure each training subset contains an equal amount of pairs that are versions and pairs that are not. In this way we ensure the subsets are not biased and, therefore, the baseline accuracy (corresponding to making a random guess) is 50%. The statistical significance of the results is assessed using the paired t-test (Wasserman, 2003) with a significance threshold of $p < 0.001$.

Results using a single tonal representation

We start by comparing the results obtained when using a single descriptor, either the melody, the bass line or the harmony. In Table 6.1 we present the MAP and MaRR results for the 76-song subset which was used for training the parameters of the matching algorithm. At first glance we see that the harmonic representation yields better results compared to the melody and bass line descriptions. Nonetheless, the results also suggest that the latter two representations do indeed carry useful information for version identification. Evidence for the suitability of melody as a descriptor for version identification has been reported elsewhere (Marolt, 2008; Serrà et al., 2010; Tsai et al., 2008). However, no evidence for the suitability of bass lines has been acknowledged prior to this study. Moreover, to the best of our knowledge, no direct comparison between these three music representations has been performed previously in the literature.

To properly assess the performance of each descriptor, however, a more realistic collection size is required. Thus, we now turn to the results obtained using the full 2125 song collection, presented in Table 6.2. As expected, there is a drop in performance for all three representations (cf. Serrà et al.,

Feature	MAP	MaRR
Melody	0.483	0.332
Bass line	0.528	0.355
Harmony	0.698	0.444

Table 6.2: Results for single tonal representation (full collection, 2125 songs).

2010). The harmonic representation still outperforms the melody and bass line descriptors, for which the drop in performance is more considerable. Nevertheless, the MAP results we obtain using the melody (or the bass line), though lower than those obtained using harmony, are still considerably higher than those obtained by other version identification systems using similar (and different) types of descriptors (Serrà et al., 2010).

As suggested earlier, one probable reason for the superiority of the harmonic representation is that some versions simply do not contain a main melody, and (though less often) some songs do not contain a bass line (e.g. a singer accompanied by a guitar only). Still, as seen in the results for the 76-song subset, even when the melody and bass line are present, the harmonic representation produces better matching results in most cases. This can be attributed to the different degree of modification applied to each tonal representation across versions: whilst some versions may apply reharmonisation, in most cases the harmony remains the least changed out of the three music representations. Differences in the melody and bass line may also be increased due to transcription errors, an additional step which is not necessary for computing the HPCP.

Since the HPCP is computed using the complete audio mix, we know that the melody and bass line may also be, to some degree, represented in the HPCP. Thus, even though the HPCP descriptor is considered to be related to harmony, it is interesting to ask to what degree is the information it encapsulates different from the melody and bass line descriptors. This aspect, albeit very simple, has not been formally assessed before. To answer this question we turn to the second part of the evaluation, in which we examine whether fusing the different representations results in improved matching or not.

Feature	Random Forest	SMO (PolyKernel)	Simple Logistic	K*	Bayes Net
M	69.84	76.73	75.29	77.98	77.90
B	73.34	81.03	78.98	81.31	81.03
H	82.04	87.69	86.42	87.74	87.58
M+B	79.80	82.05	80.91	84.62	84.46
H+M	84.29	87.73	86.51	88.01	87.81
H+B	84.72	87.80	86.77	88.32	88.14
H+M+B	86.15	87.80	86.83	88.46	88.24

Table 6.3: Fusion results for the different classifiers considered.

Results using representation fusion

In Table 6.3 we present the results of our descriptor fusion approach, where we use "M" for melody, "B" for bass line and "H" for harmony (HPCP). Since we are now using a different evaluation measure (classification accuracy), in order to compare the results to those obtained when using a single tonal representation we also evaluated the classifiers using each representation separately (by including just one of the three dissimilarity measures in the feature vector), as well as all possible combinations of two out of the three representations. Several observations can be made from the results. First, we note that for all descriptors and all classifiers the results are significantly above the baseline of 50%. We see that most classifiers perform relatively similarly, though there are some notable differences. In particular, the random forest classifier consistently provides the lowest results, whilst the K* classifier consistently provides the highest (the difference between the two is for all cases statistically significant). As before, we note that when using only a single representation, the harmony provides the best performance, followed by the bass line and, finally, the melody.

Perhaps the most interesting results are those obtained by descriptor fusion. For all classifiers, combining the melody and bass line provides increased classification accuracy compared to using either of the two descriptors separately (the increase is statistically significant). Not surprisingly, this confirms that the two music representations carry complementary information and hence their combination results in increased performance. Still, using melody and bass line together does not outperform using the harmony on its own. The remaining question is thus whether combining the harmony with the other descriptors improves classification accuracy.

The results (as indicated by H+M+B) are less straightforward this time. In the case of the random forest classifier, the improvement is clear and statistically significant. However, for the remainder of classifiers the increase is not as considerable. This suggests that the benefits of considering different music representations are particularly important when the classifier has (relatively) low performance. Nonetheless, if we consider the results of the best performing classifier (K*), it turns out that the increase in accuracy when combining harmony, melody, and bass line compared to harmony alone is in fact statistically significant. Still, the small increase in accuracy (less than 1%) indicates that our harmonic representation (HPCP), to a great extent, carries overlapping information with the melody and bass line.

Discussion

To better understand how these different tonal representations can complement each other, we manually examined cases where the melody or bass line descriptors produced better matching results than the HPCP. In Figure 6.4 we present three dissimilarity matrices of 10 queries compared to 10 targets, where the same 10 songs are used both as the queries and the targets. The three dissimilarity matrices are computed using (a) HPCP, (b) melody, and (c) bass line. The dissimilarities in each matrix are normalised by the greatest value in the matrix so that they are visually comparable. Cells for which the query and target are versions of the same musical piece are marked with a black box.

An example where the melody works better than the HPCP can be seen for the version group with IDs 3, 4, and 5. We see that when using the HPCP, song 4 is considered relatively different from songs 3 and 5 (light colour), whilst the dissimilarity is much smaller (darker colour) when using the melody. The three songs are different versions of the song "Strangers in the Night", popularised by Frank Sinatra. Listening to the songs we found that whilst versions 3 and 5 have relatively similar orchestral arrangements, version 4 includes several reharmonisations and entire sections where the melody is played without any accompaniment. It is clear that in such a case using the melody on its own will produce smaller dissimilarities between the versions. The bass line descriptor on the other hand does not work well in this example, for the very same reasons.

Another interesting example is provided by the version group with IDs 8, 9 and 10. The three songs are different versions of the song "White Christmas" by Irving Berlin, made famous by Bing Crosby back in 1941.

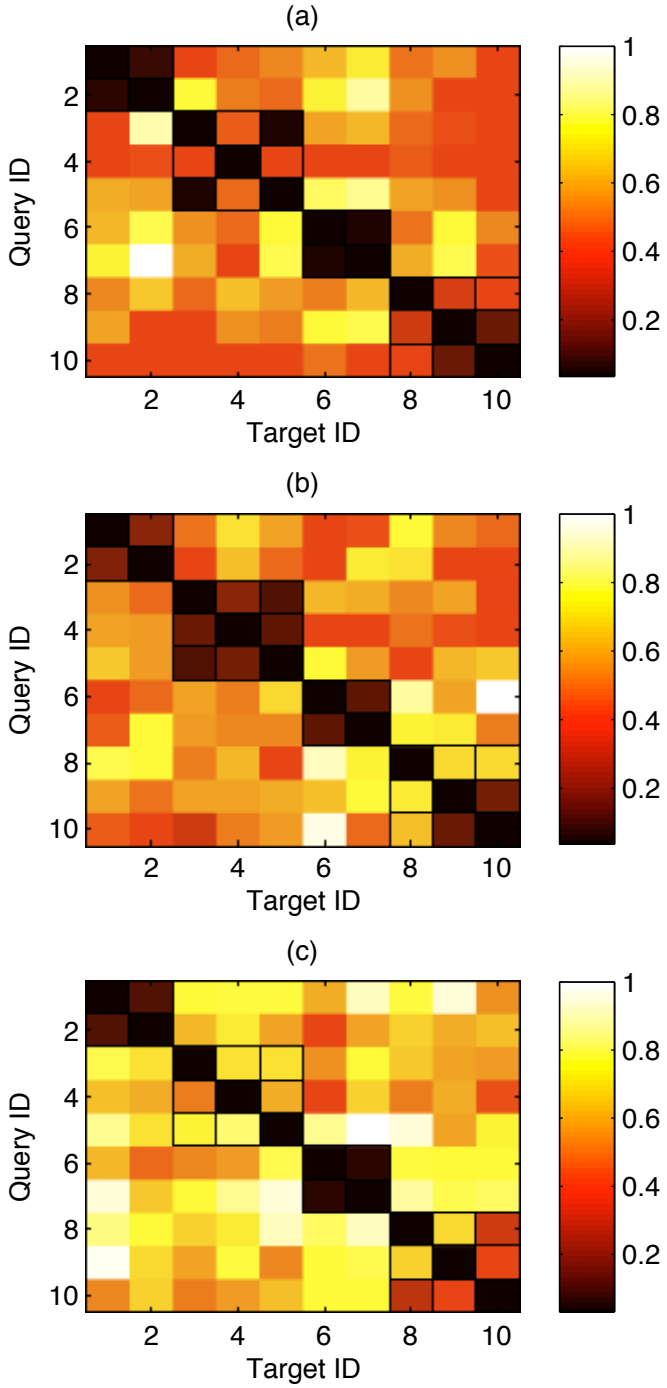

Figure 6.4: Dissimilarity matrices for 10 queries and 10 targets, produced using: (a) HPCP, (b) melody, (c) bass line. Cells for which the query and target are versions of the same musical piece are marked with a black box.

Here we see that whilst song 8 is poorly matched to songs 9 and 10 using either HPCP or melody, it is well matched to song 10 when we use the bass line. When listening to the songs we found that unlike versions 9 and 10, in version 8 there are sections where the melody is solely accompanied by the bass line. In other parts of the song the accompaniment, played by a string section, consists of melodic motifs which interleave with the singing. Furthermore, unlike the more traditional vocal renditions in 9 and 10, the melody in 8 is sung in a more "talk-like" fashion, which combined with the predominant melodic motifs of the string section causes greater confusion in the melody extraction. The various aforementioned differences explain why in this case the bass line succeeds whilst the melody and HPCP do not perform as well. Curiously, whilst song pairs 8-10 and 9-10 are well matched using the bass line, the pair 8-9 is not. Though investigating the exact cause for this inequality is beyond the scope of this study, a possible explanation could be the greater degree of transcription errors in the extracted bass line of song 9. Since the dissimilarity computation is not metric, it is possible for transcription errors to have a greater effect on the matching of some songs compared to others.

The results above show that, whilst in most cases the HPCP (most closely related to the harmony) is the most reliable music representation for version matching, the melody and bass line can provide useful information in cases where the harmony undergoes considerable changes or is otherwise completely removed (e.g. a cappella singing in unison). Although this observation may seem somewhat obvious, approaches for version matching using descriptor fusion such as the one proposed by Foucard et al. (2010) and the one proposed in the current study do not take this into account since they always use all descriptors even when one of them may not be appropriate. Thus, a potential approach for improving matching accuracy would be, rather than always using all descriptors, to first attempt to determine which descriptors will provide the most reliable matching results and then use only those. For example, if we detect that one version has accompaniment and the other does not, we might decide to use just the melody rather than the melody, bass line and harmony. Another possibility would be to train a classifier for each representation using a classification algorithm that returns a confidence measure along with its prediction. Then, the prediction of the classifier with the highest confidence could be selected, unless there is no clear winner in which case we could use the prediction obtained by descriptor fusion as described in this study.

Finally, whilst the generality of the matching algorithm employed in this

study (cf. Section 6.2.2) means it can be easily adapted to different types of time series, it is still relevant to ask whether it is the most appropriate matching approach for the melody and bass line sequences. Since the algorithm was originally designed to work with chroma features (HPCPs), it is possible that it introduces a slight bias towards this type of time series. Another conjecture is that the intrinsic lower dimensionality of the melody and bass line features may in part be the cause for the reduced performance of these features. One of our goals for future work will be to address these questions by evaluating and comparing different matching algorithms with the melody and bass line representations proposed in this study.

6.2.4 Query-by-humming

As mentioned earlier, query-by-humming can be considered a special case of the more general task of version identification. We are still interested in matching different renditions of the same musical piece, only this time one of the renditions is monophonic and produced by the user themselves. The QBH method proposed here is an almost direct application of the version identification approach proposed above, with very little modification. One important difference however is that unlike the version versus version scenario, in this case the queries will only contain melody information (end-users cannot/will rarely sing the harmony or bass line and will almost always focus on the melody). This means that for QBH, of the three tonal-representations considered earlier, the melody-based representation is the most (and possibly only) suitable representation both for the queries and for the target database. As such, if we want to successfully match the queries against a dataset of *polyphonic audio*, using an automatic melody extraction algorithm becomes essential.

The QBH system presented here relies solely on the matching of melody-based representations. The melody descriptor database is created using the same process described earlier for representing a song for version identification: given a polyphonic recording, we first extract the melody using the algorithm proposed in Section 4.2, and then perform the representation abstraction described in Section 6.2.2. This step only needs to be performed once for every song in the database and, as noted earlier, is fully automatic.

To query for a song, the user records a (relatively) short segment of the melody by either singing or humming into a microphone. The query may contain any section of the melody (i.e. it does not necessarily start at the beginning of the song), and may be sung in any key, with or without lyrics.

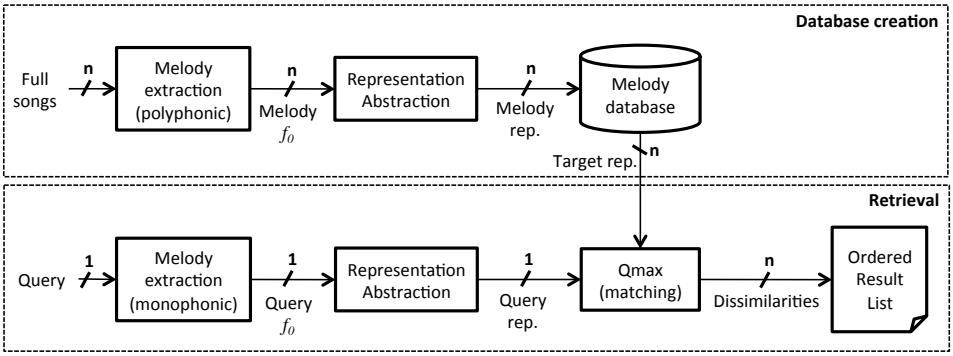

Figure 6.5: Block diagram of the proposed query-by-humming system.

The recording must then be converted into the same representation used for complete songs in the database. Conveniently, our melody extraction algorithm, even though it was designed for processing polyphonic material, also works very well for monophonic f_0 transcription. This is because a query can be viewed as a simplified song where only the melody is present without any accompaniment. The only significant change that needs to be made to the algorithm is the removal of the voicing detection step, since all detected pitch contours will belong to the melody. Additionally, a simple energy threshold is applied to filter any microphone noise detected during silent segments of the query. Once the query pitch is extracted we apply the same representation abstraction applied to the f_0 sequences of the full-length songs.

The matching is performed as before using the $Q_{\max}$ algorithm (Serrà et al., 2009). The query representation is compared against every song in the database and songs are then returned in order of increasing dissimilarity (i.e. the song most similar to the query is returned first, then the second closest song, etc.). A block diagram of the proposed approach is presented in Figure 6.5.

It is important to note that the proposed approach is a proof-of-concept prototype. Most importantly, in a large-scale QBH system it might not be feasible to compare the query against every song in the database in a reasonable amount of time, and some type of indexing technique would be required to speed up the search. Although the dissimilarity measure returned by $Q_{\max}$ is not metric, indexing could be achieved using techniques based on hashing (Casey et al., 2008b; Grosche & Müller, 2012; Salamon &

Rohrmeier, 2009), inverted file indexing (Kurth & Müller, 2008) or vantage indexing (Skalak et al., 2008; Typke & Walczak-Typke, 2008). Alternatively, methods exist for converting non-metric distance measures into metric distances (Liu & Hua, 2009), which would allow the use of more standard indexing methods (Bozkaya & Ozsoyoglu, 1999).

Music collections

The proposed QBH approach is evaluated using two collections. The first collection contains only the canonical version of every song from the full 2125 song collection described in Section 6.2.3. Version sets which do not contain the original/canonical version were discarded due to potential confusion of the user when recording a query (33 out of the 523 version sets described in Section 6.2.3). This resulted in a total of 481 songs for this "canonical" collection. Note that for this collection there is only one correct answer for every query.

The second collection is the full set of 2125 songs (Serrà et al., 2012). Note that this collection presents both an advantage and a disadvantage compared to the canonical collection. On the one hand, it is more than four times larger, and an increase in database size often results in a decrease in retrieval performance (Salamon & Rohrmeier, 2009; Serrà et al., 2010). On the other hand, the additional songs are (almost) all versions of the songs in the canonical collection, which might increase the probability of successful retrieval since a query that is poorly matched against the canonical version of a song might be well matched against one of its alternative versions. A list of the songs included in the canonical and full collections is provided online[4].

Queries

For the purpose of this evaluation, we recorded a set of sung queries corresponding to songs in the canonical collection described above. Subjects were presented with a list of all 481 songs in the collection out of which they were asked to select songs and record queries. A total of 118 queries were recorded by 17 subjects (9 female and 8 male). The smallest amount of queries recorded by a subject was 1 and the largest was 11, with a mean of 6.8 queries per subject. The musical experience of the subjects ranged from none at all to amateur musicians. To simulate a realistic scenario all queries

[4] http://mtg.upf.edu/download/datasets/MTG-QBH

were recorded using a basic laptop microphone and no post-processing was applied. Query duration ranged from 11 seconds to 98 seconds and the average query duration was 26.8 seconds. The complete set of 118 queries is available online (cf. footnote 4).

One factor that we expected to have a notable influence on the performance of the system was the self-tuning of the sung queries. By this we refer not to the difference in key between the query and the target song, but to whether the singer maintains the same reference tuning throughout the query. If they do not, the same (theoretical) note might be represented by very different frequencies within a single query, thus dramatically changing the contour of the melody. Such de-tuning was observed for roughly one third of the subjects, where the reference tuning was abruptly changed several times during a single query. To observe the effect of this de-tuning on performance, we manually divided the subjects into two groups: "good tuning" and "bad tuning". It is important to note that this division was only based on the tuning of the singer with respect to themselves, not on the resemblance of the sung query to the original melody nor on any other singing quality criterion.

Evaluation measures

Two evaluation measures are used to asses the performance of the proposed approach. The first is the mean reciprocal rank (MRR) defined previously (cf. Equation 6.4), which is the standard measure for evaluating QBH algorithms (Dannenberg et al., 2007). Recall that this measure is different from the MaRR used to evaluate version identification in the previous sections – for version identification we were interested in recovering all versions of the query, and hence for every query we computed the reciprocal ranks of all target versions and averaged them, and then averaged this value over all queries. For QBH, we are only interested in the rank of the highest correct match. Thus, when the collection contains more than one correct match for query u, we set r_u in Equation 6.4 to the highest rank obtained by any target in the result list Λ_u which is a version of u. Recall that the MRR goes from 0 (worst performance) to 1 (best performance).

The second measure is the top-X hit rate, which reports the proportion of queries for which $r_u \leq X$. If the system had an interface which returned X results for every query, the top-X hit rate would describe the percentage of queries for which at least one of the displayed results corresponds to the correct target song.

Singers	MRR	Top-X hit rate (%) 1	3	5	10
Good tuning	0.56	50.00	60.00	61.25	63.75
Bad tuning	0.23	21.05	21.05	23.68	26.32
All	0.45	40.68	47.46	49.15	51.69

Table 6.4: QBH results for the canonical collection (481 songs).

Singers	MRR	Top-X hit rate (%) 1	3	5	10
Good tuning	0.67	61.25	70.00	73.75	78.75
Bad tuning	0.33	28.95	34.21	34.21	39.47
All	0.56	50.85	58.47	61.02	66.10

Table 6.5: QBH results for the full collection (2125 songs).

Results and discussion

The retrieval results for the canonical collection are presented in Table 6.4. The first thing we note is that there is a significant difference in performance between subjects with good tuning and subjects with bad tuning. For the former group, the correct song is ranked first in the results list 50% of the time, whilst for the latter group only 20% of the time. Still, it is worth noting that for all singers the results are well above the random baseline (returning a song at random from the database) which would obtain an MRR of 0.01, top-1 hit rate of 0.15% and top-10 hit rate of approximately 2%. It is also interesting to note that the top-1 hit rate for singers with good tuning is not too far below the top-1 hit rate reported by Pardo & Birmingham (2003) for humans attempting to identify queries manually (66%). When comparing the two groups we note that whilst for subjects with good tuning increasing the size of the result list increases the chance of finding the correct answer (we observe a 14% increase between the top-1 and top-10 hit rates), for subjects with poor tuning the increase is a lot smaller (5%).

Another interesting observation is that even for subjects with good tuning, the correct song appears in the top-10 results just 64% of the time, suggesting there is still much room for improvement. Nonetheless, the results are definitely encouraging, obtaining comparable performance to the

that reported by Ryynänen & Klapuri (2008b) (and outperforming Duda et al., 2007) on a collection of similar size even though only few changes were made to adapt the approach from version identification to QBH. For singers with poor tuning on the other hand it is clear that there is much work to be done. It is interesting to note that despite its significant influence on performance, the issue of query self-tuning has not been addressed in the previous studies on audio-to-audio QBH mentioned here. In the future we could further investigate the influence of query de-tuning on performance and research techniques for overcoming such de-tuning, for example query reformulation (Zhang et al., 2012). Another possibility would be to try and avoid the de-tuning problem altogether by helping subjects maintain a fixed reference tuning (e.g. providing a fixed reference tone during query recording). Finally, it is probably valid to ask whether we should expect a singing-based retrieval system to work for subjects with poor tuning, who might be directly better off trying search methods that do not involve singing such as query-by-example (Salamon & Rohrmeier, 2009) or query-by-tapping (Hanna & Robine, 2009).

Next, we turn to the results obtained for the full 2125 song collection (Table 6.5). We see that there is a significant improvement in performance for both subject groups. As proposed earlier, the cause for this improvement (despite the increased database size) is the addition of cover versions to the collection, meaning we increase the possibility of finding a correct match for queries that do not match the canonical version of a song. Another potential cause for this improvement is that using several versions of each song increases the probability of extracting at least one version of the melody with high accuracy, thus improving retrieval performance for songs where the melody extraction step did not work well for the canonical version.

The improved performance for the full collection is encouraging, with an MRR of 0.67 and a top-10 hit rate of almost 80% for subjects with stable reference tuning. It also highlights an important fact – the more versions we have of the same song in the collection, the better the chances of retrieving it will be. This fact is exploited by approaches such as the one proposed by Pardo et al. (2008), where the target database is directly comprised of user queries. By combining the two approaches, we can obtain a system that does not suffer from the cold start problem (the initial descriptor database is created using the melody extraction algorithm) and whose performance improves the more people use it (by adding successful queries into the database).

6.2.5 Conclusion

In this section (6.2) we demonstrated how the output of our melody extraction algorithm can be exploited for automatic similarity-based music retrieval. We studied two related retrieval tasks: version identification and query-by-humming. For the former, we used the algorithm presented in Section 4.2 to extract two different tonal representations of a song: the melody and the bass line (using a modified version of the algorithm adapted for bass line extraction). We studied different degrees of abstraction for representing the melody and bass line, and found that abstracting away octave information and quantising pitch information to a semitone level are both necessary steps for obtaining useful descriptors for version identification. The new melody and bass line descriptors were evaluated on a relatively large test collection, and shown to carry useful (and complementary) information for version identification. Combined with the proposed matching algorithm, our melody and bass line descriptors obtain MAP results comparable to (and in some cases higher than) other state-of-the-art version identification systems. Still, it was determined that in most cases the harmony-related HPCP descriptor produces better matching accuracy for this task. We have also shown that by using a classification approach for descriptor fusion we can improve accuracy, though the increase over using HPCPs alone is (albeit significant) small. One could argue that, to date, the use of different music representations for computing version similarity has not received the attention it deserves. In this study we have taken a necessary step in this research direction, which not only holds the promise of improving identification accuracy, but also improving our understanding of the relationship between different musical cues in the context of music similarity.

Next, we demonstrated how the proposed version identification method can be adapted to perform fully automatic query-by-humming of polyphonic audio collections. A prototype system was presented and evaluated against two collections, one containing only the canonical version of each song and the other containing both the canonical and cover versions of each song. The approach was shown to obtain results comparable to those presented in previous studies, and current limitations were identified for future improvement. It was then shown how performance can be increased significantly by including more than one version of each song in the target database. In the future we intend to investigate the influence of different factors on retrieval performance such as query length and melody extraction accuracy. Also, as with the proposed version identification method, we intend to evaluate the proposed QBH approach using different distance measures and compare

the results to those obtained using $Q_{\max}$. Whilst there is still much work to be done in this area, the results presented here demonstrate that it is definitely feasible to develop a fully automated QBH system for polyphonic audio collections using the output of our melody extraction algorithm.

6.3 Genre classification

6.3.1 Introduction

Genre classification involves the assigning of categorical labels to pieces of music in order to group them by common characteristics (Tzanetakis & Cook, 2002). Genres are commonly used to organise large music collections both private and commercial, and the benefits of automating the classification process have led to the topic receiving much attention from the MIR community in recent years. Various approaches have been proposed, utilising features that describe different aspects of music such as pitch, timbre, rhythm and their combination (Guaus, 2009; Scaringella et al., 2006). Approaches using source separation (Rump et al., 2010) or models of auditory human perception (Panagakis et al., 2009; Sturm, 2012) have also been proposed. However, one key aspect in music that has received little attention in the context of genre classification is the melody. Melodies in different genres can be expected to have different characteristics, especially in the case of sung melodies where it has been shown that the human voice is used in different ways depending on the musical genre (Sundberg & Thalén, 2001). Whilst there have been studies on genre classification using melody characteristics computed from symbolic data (MIDI files; McKay & Fujinaga, 2004), to the best of our knowledge there is no study on genre classification using high-level melodic characteristics extracted directly from the audio signal of polyphonic music.

In this section, based on Salamon et al. (2012c), we propose a method for genre classification based on melodic characteristics extracted from polyphonic music excerpts. We use the melody extraction algorithm proposed in Section 4.2 to identify and characterise the contours that belong to the predominant melodic line, resulting in a set of melody-related features. The initial set of melody features computed by the algorithm is extended by computing characteristics derived from a musicological study of melody pitch contour. We use standard machine learning algorithms for the classification and compare our results to a baseline approach based on timbral features. An important aspect of our system is that the melodic features we use are

what we consider high-level features. That is, unlike some of the features commonly used for genre classification, the features we use can be easily understood by humans. This means the classification results can be (in some cases) directly linked to aspects of the melody, for example that the vibrato rate applied in flamenco singing is on average lower than that applied in opera singing, or that the average pitch range used in vocal jazz is greater than that used in pop music.

In Section 6.3.2 we describe the melodic features extracted from the audio signal. In Section 6.3.3 we describe the classification algorithms employed, and in Section 6.3.4 we describe our evaluation methodology. The results are presented in Section 6.3.5, followed by some conclusions in Section 6.3.6.

6.3.2 Melody features

For the purpose of genre classification, instead of using the final sequence of f_0 values returned by our melody extraction algorithm, we use the actual set of pitch contours from which the sequence was selected. Recall that every pitch contour is represented by two discrete series $p(n)$ and $s(n)$ ($n = 1 \dots N_c$), where the former contains the contour's pitch values (in cents) and the latter its salience values (cf. Section 4.2.1). For this study we define features based on $p(n)$ only. Since $p(n)$ is not quantised into semitones, it allows us to capture aspects of the pitch evolution that are important for genre characterisation such as vibrato. An example of contours extracted from excerpts of different genres was provided in Figure 4.2 of Chapter 4, in which melody contours are highlighted in bold.

We divide the features computed for each contour into three categories: pitch and duration features, vibrato features, and contour typology. Once the per-contour features are computed they are used to compute a set of global per-excerpt features for training the classifier.

Pitch and duration features

These features are related directly to contour pitch or length (duration), and are (with the exception of the new pitch range feature) a subset of the features computed by the algorithm as described in Section 4.2.2:

- **Length** $C_l = N_c \cdot \frac{H}{f_S}$ (in seconds).
- **Pitch mean** $C_{\bar{p}} = \frac{1}{N_c} \sum_{n=1}^{N_c} p(n)$.

- **Pitch deviation** $C_{\sigma_p} = \sqrt{\frac{1}{N_c}\sum_{n=1}^{N_c}(p(n) - C_{\bar{p}})^2}$.
- **Pitch range** $C_r = \max(p(n)) - \min(p(n))$.

Vibrato features

Vibrato is a periodic variation of pitch that is characterised by its rate and extent (depth) (Sundberg, 1995). Apart from being a distinctive element of the singing voice, the way in which it is applied varies between different singing styles (Sundberg & Thalén, 2001), and thus we expect features related to vibrato to be important for genre classification. As a first step the system detects whether a contour has vibrato or not. This is done by applying the STFT to the pitch contour $p(n)$ after subtracting the mean as proposed by Herrera & Bonada (1998) and checking for a prominent peak in the magnitude spectrum $|P(k)|$ at the expected range for vibrato in human voice (5–8 Hz). If vibrato is detected, the rate and extent can be computed from the peak's frequency and magnitude respectively. We use a frame size of 120 samples (350 ms) to ensure we capture at least 2 cycles of the lowest period expected for vibrato, and a hop size of 1 sample.

In addition to these features, we wanted to capture the amount of vibrato applied throughout a contour. That is, the proportion of the contour in which vibrato is applied. We refer to this as vibrato coverage, and we expect it to vary between genres where vibrato is used a lot (e.g. opera) and genres where it might be applied just at the end of a phrase (e.g. vocal jazz). A summary of the vibrato features is:

- **Vibrato rate** V_r: the frequency of pitch modulation, indicated by the location of the prominent peak of $|P(k)|$ within the expected vibrato range (in Hz).
- **Vibrato extent** V_e: the magnitude of said peak (in cents).
- **Vibrato coverage** V_c: the ratio of samples with vibrato to the total number of samples in the contour (ranges between 0–1).

Contour typology

In his study of melodic contour typology, Adams (1976) proposes to categorise melodic segments based on the "distinctive relationship among their minimal boundaries". By categorising the possible relationships between a

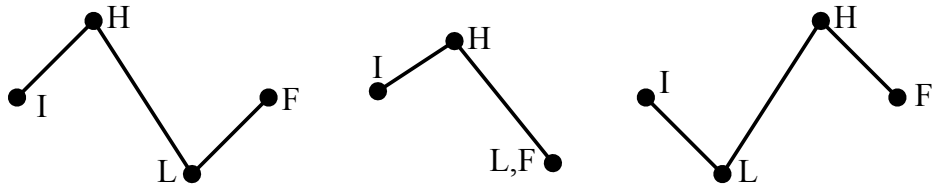

Figure 6.6: Different types of melodic contour.

segment's initial (I), final (F), highest (H) and lowest (L) pitch, 15 "contour types" are defined. An example of three different contour types is provided in Figure 6.6.

We adopt Adam's melodic contour typology and compute the type of each contour. Before the type is computed the contour pitch is quantised into a quarter-tone resolution, to avoid smaller pitch variations affecting the contour type. To summarise:

- **Contour type** ζ_i: one of 15 melodic contour types ($i = 1 \ldots 15$).

Global features

The contour features are used to compute global excerpt features which are used to train the classifier. For the pitch, duration and vibrato features we compute the mean, standard deviation, skewness and kurtosis of each feature over all the melodic contours in the excerpt. The contour typology is used to compute a type distribution describing the proportion of each contour type out of all the pitch contours forming the melody. In addition to these features several additional global features are added:

- **Global highest pitch** G_{pmax}: The highest pitch in the melody.
- **Global lowest pitch** G_{pmin}: The lowest pitch in the melody.
- **Global pitch range** $G_r = G_{\text{pmax}} - G_{\text{pmin}}$.
- **Global vibrato presence** G_v: the ratio between the number of contours with vibrato to the total number of contours in the melody (ranges from 0–1).
- **Interval features**: we compute the interval between each pair of consecutive contours as the difference between their mean pitch height.

We then compute the mean, standard deviation, skewness and kurtosis of all the intervals in the melody.

This gives us a total of 51 features. Initial experiments revealed that some features resulted in better classification if they were computed using only the longer contours in the melody. This is probably because long contours are less likely to be an error of the melody extraction algorithm, and also there is a greater chance of detecting vibrato features in longer contours. For this reason we computed for each feature (except for the interval features) a second value using only the top third of the melody contours when ordered by duration. This gives us a total of 98 features for the next stage.

6.3.3 Classification

To classify the audio excerpts we compare several classification algorithms from the Weka data mining software (Hall et al., 2009). We start by performing attribute selection using the *CfsSubsetEval* attribute evaluator and *BestFirst* search method (Hall, 1999) with 10-fold cross validation, only keeping features that were used in all folds. Each attribute is normalised feature-wise between 0 and 1. For each classification algorithm we perform 10-fold cross validation and repeat the experiment 10 times, reporting the average accuracy. The algorithms compared are support vector machines (SMO; radial basis function kernel), random forest (RF), K-nearest neighbours (K*) and Bayesian network (BNet).

6.3.4 Evaluation methodology

Datasets

For evaluation we constructed a dataset of five musical genres in which the melody plays an important role: opera, pop, flamenco, vocal jazz and instrumental jazz (where the melody is played by a saxophone or trumpet rather than sung). For initial experiments the dataset consisted of fifty 30-second excerpts per genre (250 excerpts in total). The set was later expanded to include 100 excerpts per genre (500 excerpts in total). To cover variations within a genre the excerpts for each genre were selected from a wide set of artists. All excerpts were taken from a section of the song where the melodic line is clearly present. As a final experiment we evaluate our method on the GTZAN collection (Tzanetakis & Cook, 2002), which consists 10 genres with one hundred 30-second excerpts per genre (1000 excerpts in total). Note that in this collection some excerpts might

not have a melody at all, and for some genres (e.g. heavy metal) the melody extraction may not perform very well. Still, we wanted to see what could be achieved for this collection without any modification to the proposed method or excerpts.

Baseline and combined feature sets

To compare our results we also compute a baseline set of low-level timbral features which are commonly used in genre classification. For each excerpt we compute the first 20 Mel-frequency cepstral coefficients (MFCCs) as done by Pampalk et al. (2005), using a 23 ms window size with 50% overlap, taking 40 mel-frequency bands up to 16 kHz. We calculate the mean and variance of each coefficient, resulting in a total of 40 descriptors. We also wanted to see whether results could be improved by combining low-level and high-level information. To do this we created a third feature set which combines our melodic features with the MFCC features, giving a total of 138 descriptors.

6.3.5 Results

We start by presenting the results for the initial 250 excerpt dataset. A total of 10 melodic attributes were selected out of the initial 98 (a * indicates the feature was computed from long contours only): C_r:mean, $C_{\bar{p}}$:mean, V_r:mean*, V_r:skewness*, V_e:mean*, V_c:mean*, V_c:stddev*, ζ_9*, ζ_{10}*, ζ_{14}*. We see that most of the selected features are computed from the longer contours of the melody only. We also note a strong presence of vibrato related features. In Figure 6.7 we present the classification results comparing the melodic, MFCC and combined feature sets. The number of features selected for each set is indicated in brackets.

We see that with all classifiers we obtain a classification accuracy of over 90% using the melodic features. In all cases the melodic feature set outperforms the baseline approach. Next, we note that for most classifiers we can increase the classification accuracy by combining the MFCC features with our high-level melodic features. To see whether any descriptors were especially discriminative we also classified the data using a decision tree. It turned out that two important features are the mean vibrato coverage and mean vibrato rate. In Figure 6.8 we see that the genres can be fairly well separated using just these two descriptors. Furthermore, both descriptors are musically meaningful (the former expressing the degree to which vibrato is applied and the latter the average rate of the vibrato).

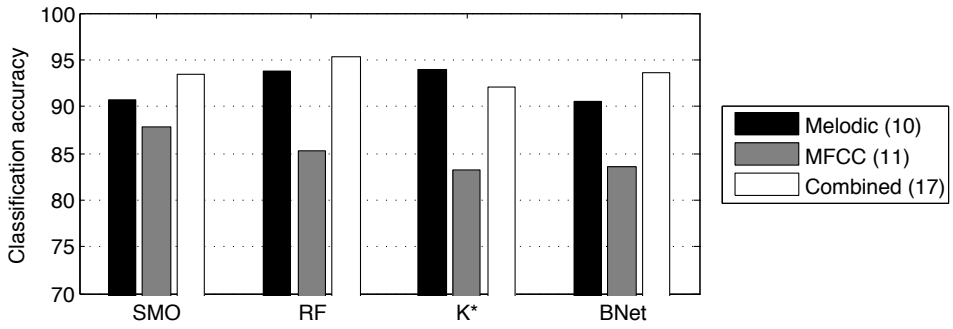

Figure 6.7: Classification results for the initial 250 excerpt dataset.

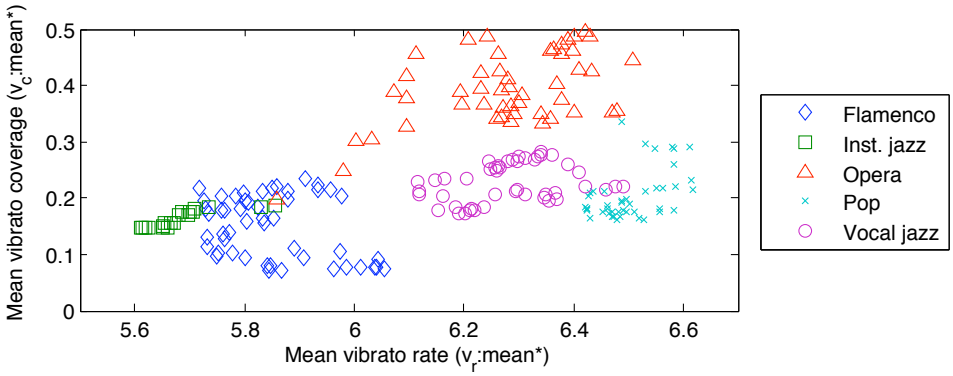

Figure 6.8: Mean vibrato coverage versus mean vibrato rate.

Next we examine the results for the extended dataset (500 excerpts), provided in Figure 6.9. Note that this time only 7 descriptors were selected for the melodic feature set. We see that for all classifiers the melodic feature set maintains classification accuracies above 90%. We also note that for RF and BNet the melodic set still outperforms the baseline approach even though it uses less than half the amount of descriptors. This time results for all classifiers are improved when combining the two different sets of descriptors. To ensure the results were not biased by the different size of each feature set, we ran two further experiments imposing a fixed number of descriptors for all three sets (21 and 10). In both cases the results were consistent with those of Figure 6.9, with the combined set outperforming the other two. Examining the confusion matrices of the classification results, we found that for the melodic feature set the confusion occurs primarily between pop and vocal jazz. This is understandable as these singing styles

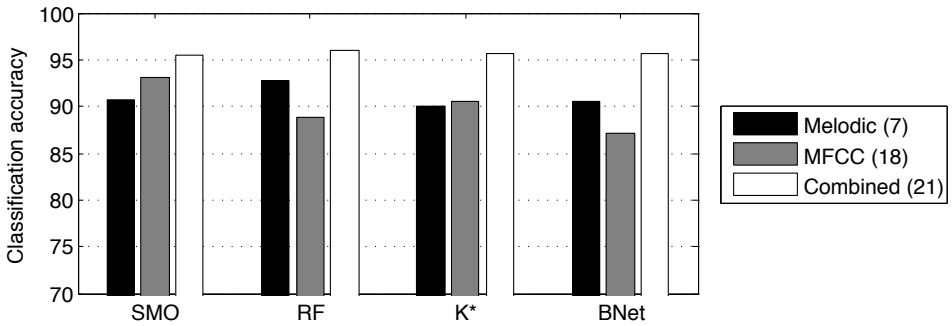

Figure 6.9: Classification results for the extended 500 excerpt dataset.

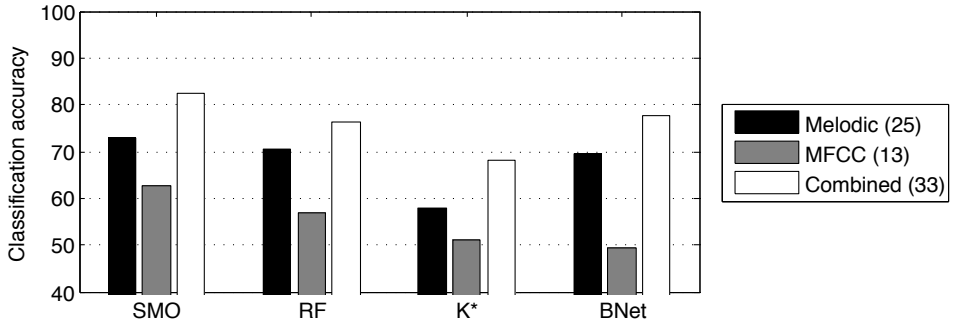

Figure 6.10: Classification results for the GTZAN dataset.

have common characteristics, making them hard to distinguish even for humans (Sundberg & Thalén, 2001). Combining the melodic features with the MFCC features reduces this confusion, leading to an overall increase in accuracy.

Finally, we examine the results obtained for the GTZAN collection, presented in Figure 6.10. As expected, the classification results are not as high as those obtained for the collections where we ensured that there is a melody in each excerpt. Still, with the SMO classifier and the combined feature set we obtain an accuracy of 82%, improving significantly on both the melodic and MFCC feature sets. Whilst this does not surpass the highest accuracies reported for this collection to date (e.g. Guaus, 2009), the results illustrate an important point – that combining low-level features with high-level melodic features is a promising approach for improving genre classification.

6.3.6 Conclusion

We presented a genre classification method based on a novel set of melodic features. By using our proposed melody extraction algorithm we were able to compute these features directly from the audio signal of polyphonic music, without the need to obtain the monophonic melody track beforehand. A set of melodic features was proposed, based on pitch, duration and vibrato characteristics, and on contour typology. The melodic feature set was evaluated on three different datasets and was shown to outperform a baseline low-level timbral feature set based on MFCCs. Most importantly, we demonstrated that the classification accuracy can be improved by combining the two feature sets. This suggests that adding high-level melodic features to traditional low-level features is a promising approach for genre classification. It is worth recalling that the mean overall accuracy of our algorithm, which was shown to be state-of-the-art (cf. Sections 2.5 and 4.4.3), is 75%. The positive results obtained in this study demonstrate once more that the mid-level representation of the melody extracted by our algorithm, even though it is not 100% accurate, can still be used successfully for addressing MIR challenges. Finally, another important aspect of the approach presented in this study is the fact that most of the melodic features proposed can be easily understood by humans. This means that the classification results can be interpreted more easily, allowing us to make straight forward links between musical genres and melodic characteristics.

6.4 Tonic identification in Indian classical music

In Section 6.2 we presented applications which exploit the final output of our melody extraction algorithm (i.e. the f_0 sequence of the melody). In Section 6.3 we went one step back in the architecture of the algorithm, exploiting the contour features computed in the penultimate block of the algorithm. In this section, based on Salamon et al. (2012a), we go back a step further, and present an application that directly exploits the salience function introduced in Chapter 3: a method for tonic identification in Indian classical music based on a multipitch analysis of the audio signal. At the end of this section we will briefly discuss an extension of the method presented here which also exploits the final f_0 sequence extracted by the complete algorithm.

6.4.1 Introduction

One of the fundamental concepts in Indian classical music is the tonic. The tonic is a base pitch chosen by the performer, and serves as the foundation for the melodic tonal relationships throughout the performance. Every performer chooses a tonic pitch which best allows them to fully explore their vocal (or instrumental) pitch range for a given raga exposition (Deva, 1980). Consequently, all accompanying instruments are tuned in relation to the tonic pitch chosen by the lead performer.

In Indian classical music, the notes of the melodic mode (*raga*) are called *swaras*. Since the entire performance is relative to the tonic, which corresponds to the *Shadja* swara (abbreviated to *Sa*) of the raga, when the lead performer is a singer they need to be able to continuously hear the tonic pitch throughout the concert. This is provided by a constantly sounding drone which plays in the background and reinforces the tonic. The drone may be produced by a variety of instruments such as the *Tanpura* (Figure 6.11), the electronic *Shruti box* or by the *sympathetic strings* of an instrument such as the *Sitar* or *Veena*. For definitions and details regarding all of the Indian classical music terminology used above (in italics) the reader is referred to Viswanathan & Allen (2004)[5]. Along with the tonic, the drone typically produces other important notes in the raga such as the *Pa* (fifth) or the *Ma* (fourth), and slightly less often the *Ni* (seventh), depending on the choice of raga. When the two prominent notes of the drone are *Sa* and *Pa* we say that it is tuned using *Pa tuning*, and when they are *Sa* and *Ma* we say it is tuned using *Ma tuning*. The drone serves as a reference for establishing all the harmonic and melodic relationships during a given performance. Other notes used in the performance derive their meaning and purpose in relation to the *Sa* swara and the tonal context established by the particular raga (Danielou, 2010).

When considering the computational analysis of Indian classical music, it becomes evident that identifying the tonic is a crucial first step for more detailed tonal studies such as intonation analysis (Koduri et al., 2012b), motif analysis (Ross et al., 2012) and raga recognition (Chordia et al., 2009). This makes automatic tonic identification a fundamental research problem in this context. However, despite its importance in Indian classical music, the problem of automatic tonic identification has received very little attention from the research community to date. To the best of our knowledge, all previous

[5]At the time of writing this dissertation, Wikipedia (`http://www.wikipedia.org/`) is also a good source of information for all of the terms mentioned here.

Figure 6.11: Tanpura, a string instrument used to produce the drone sound in Indian classical music. Original photo courtesy of S. Gulati, edited by J. Salamon.

approaches for automatic tonic identification are based on applying monophonic pitch trackers to the audio recording, meaning they solely use the information proportioned by the predominant pitch (Sengupta et al., 2005). In some cases a monophonic pitch tracker is used even though the audio recording contains several instruments playing simultaneously (e.g. Ranjani et al., 2011), which is bound to produce errors in the analysis. Furthermore, these approaches are fairly restricted in terms of the musical content studied: Sengupta et al. (2005) only use the *Alap* (opening) sections of 118 solo vocal recordings for evaluation, and Ranjani et al. (2011) restrict the evaluation material to a specific class of ragas (*Sampurna raga*). Both approaches also restrict the allowed frequency range for the tonic to a single octave, a restriction which can not be imposed if we wish to devise a single method for tonic identification in both male and female vocal performances.

In this study we present a method for tonic identification in vocal Indian classical music based on a multipitch analysis of the audio signal (Salamon et al., 2012a). The motivation for a multipitch approach is twofold: first, the music material under investigation often includes several instruments playing simultaneously (i.e. it is polyphonic). Apart from the lead performer, recordings contain the drone instrument, and may also include other pre-

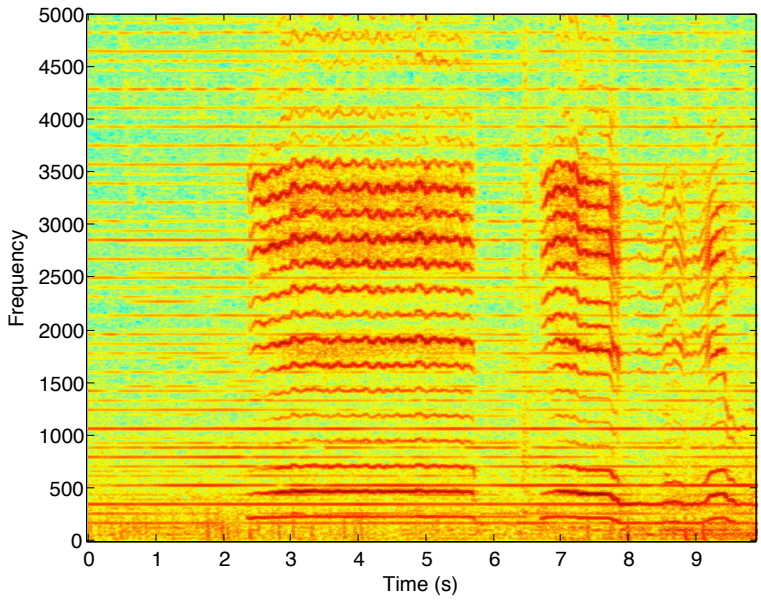

Figure 6.12: Spectrogram of an excerpt of Hindustani music with two clearly visible types of harmonic series, one belonging to the drone and the other to the lead voice.

dominant instruments such as the violin, as well as percussive instruments. Second, we know that the tonic is continually reinforced by the drone instrument, an important fact that is not exploited if we only extract a single pitch estimate for each frame of the recording. To illustrate this point, in Figure 6.12 we display the spectrogram of an excerpt of Hindustani[6] music (Danielou, 2010). Two types of harmonic series are clearly visible in the spectrogram: the first type of harmonic series, which consist of almost perfectly flat lines, belongs to the notes of the drone instrument (playing *Sa* and *Pa*). The second type of harmonic series (which starts roughly at time 2 s) belongs to the voice of the lead performer. Evidently, if we only consider the pitch of the lead performer, we loose the pitch information proportioned by the drone instrument which in this case is a better indicator of the tonic pitch.

At the outset of this study, we defined three goals for the method to be

[6]Indian classical music can generally be divided into two sub-genres: Hindustani music, found throughout the northern Indian subcontinent, and Carnatic music, the classical tradition of south India.

developed: first, it should be applicable to a wide range of performances, including both the Carnatic (Viswanathan & Allen, 2004) and Hindustani musical styles, male and female singers, and different recording conditions. Second, the approach should identify the tonic pitch in the correct octave, without restricting the allowed frequency range to a single octave. Finally, the approach should be able to identify the tonic using a limited segment of the full recording, and this segment can be taken from any part of the piece.

The structure of the remainder of this section is as follows. In Section 6.4.2 we present our proposed tonic identification method. In Section 6.4.3 we describe the evaluation methodology employed in this study, including the music collection used for evaluation and the annotation procedure used to generate the ground truth. Then, in Section 6.4.4 we present and discuss the results of the evaluation, and finally in Section 6.4.5 we provide some conclusions and proposals for future work.

6.4.2 Proposed method

Our method is comprised of four main blocks: sinusoid extraction, salience function, candidate generation and tonic selection. The first two blocks of the method are based on the processing blocks presented in Chapter 3 (cf. Figure 3.1) with just one slight modification to adapt them for the task at hand (detailed below). A block diagram of the proposed method is provided in Figure 6.13.

Sinusoid extraction and salience function

For a detailed description of the first two blocks, the reader is referred to Chapter 3 of this thesis. The only difference with respect to the processing steps presented in Chapter 3 is that we remove the initial filtering step (i.e. we do not apply the equal loudness filter). This is because we intend to use the salience function to detect the notes of the drone instrument, which can include notes at fairly low frequencies, and hence the bias against low-frequency content introduced by the equal loudness filter can have a negative effect on the analysis. In Figure 6.14 we plot the peaks of the salience function for the same excerpt from Figure 6.12. The tonic (Sa) pitch which is played by the drone instrument is clearly visible, as well as the upper and lower fifth (Pa), and the pitch trajectory of the lead voice.

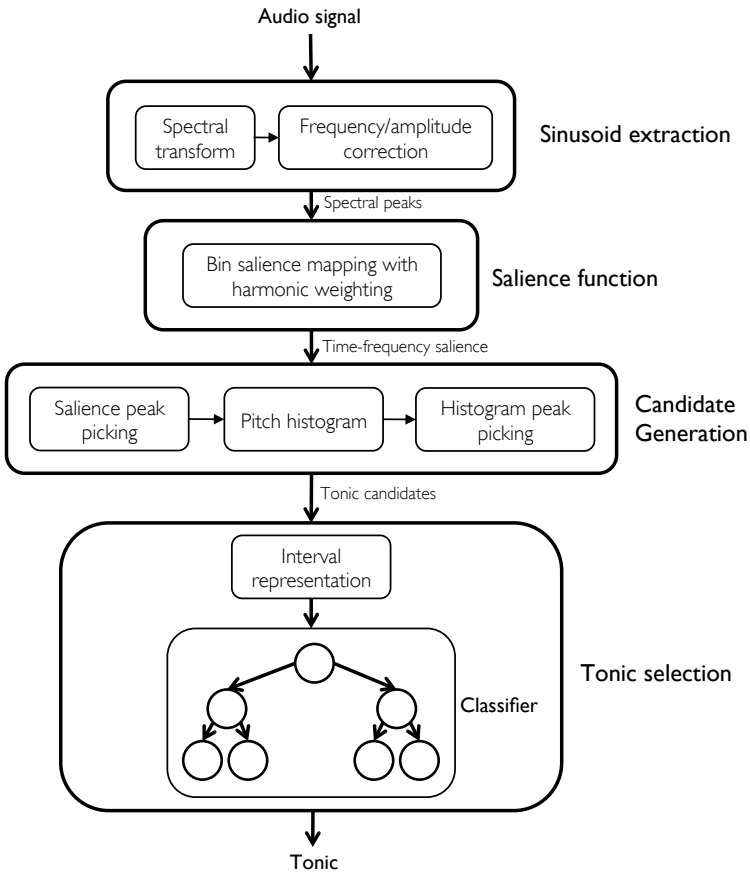

Figure 6.13: Block diagram of the proposed tonic identification method.

Tonic candidate generation

The peaks of the salience function represent the pitches of the lead performer and other predominant instruments present in the recording at every point in time, including the drone instrument. Thus, by computing a histogram of the pitch values of these peaks for the entire excerpt, we obtain an estimate of which pitches are repeated most often throughout the excerpt. Though pitch histograms have been used previously for tonic identification (e.g. Ranjani et al., 2011), they were constructed using only the most predominant pitch at each frame, which means that in many cases the tonal information provided by the drone instrument is not taken into consideration.

We start by taking the peaks of the salience function at each frame. Since

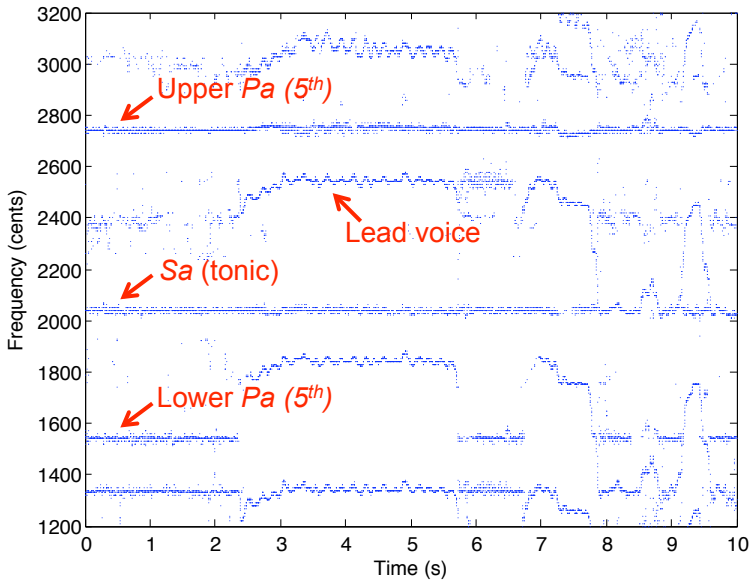

Figure 6.14: Peaks of the salience function for an excerpt of Hindustani music.

the possible frequency range for the tonic pitch used by singers in Indian classical music is relatively limited, we can reduce the range from which salient pitches are selected. To ensure we cover the complete range for both male and female singers, we consider salient pitches with a fundamental frequency ranging from 110 Hz to 370 Hz. Importantly, note that this range spans almost 2 octaves, meaning the system must be able to identify not only the correct tonic pitch class, but also the octave in which it is played. Within this range, at each frame we take the top five peaks (pitches) of the salience function. The selected pitches are used to construct a pitch histogram. As the drone is usually weaker than the lead voice, we avoid weighting each peak by its magnitude. The resulting pitch histogram goes from 110 Hz to 370 Hz and has a resolution of 10 cents. The peaks of the histogram represent the most frequent pitches in the excerpt, one of which will be the tonic. In Figure 6.15 we present the histogram computed from the complete 3 minute excerpt used in the previous examples (Figures 6.12 and 6.14). The pitch axis is plotted in cents, and the histogram is normalised by the magnitude of its highest peak. For the excerpt under consideration, we can see three clear peaks: the tonic *Sa* (2040 cents), the upper *Pa* (2740 cents) and the tonic again, one octave up (3240 cents). This illustrates one of the challenges the system will have to deal with – selecting

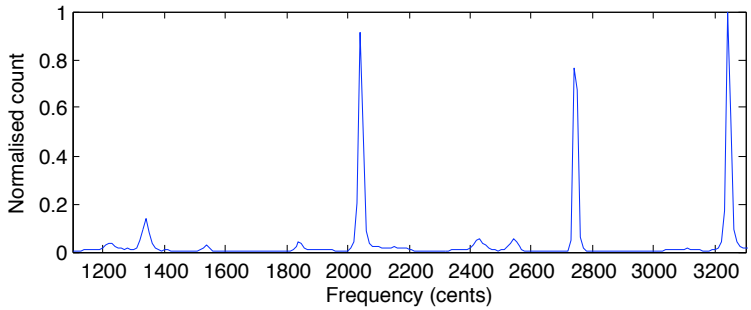

Figure 6.15: Pitch histogram for an excerpt of Hindustani music.

the tonic at the correct octave. It also highlights another important issue – the peak corresponding to the tonic will not always be the highest peak in the histogram, meaning the (perhaps naïve) approach of selecting the highest peak of the histogram would not provide satisfactory results.

Tonic selection

Since the tonic will not always be the highest peak of the pitch histogram, we take the top 10 peaks of the histogram $\hbar_i$ ($i = 1 \ldots 10$), one of which represents the pitch of the tonic. As mentioned in the introduction, all other notes present in the musical piece are tuned in relation to the tonic. Bearing this in mind, we hypothesise that the tonic can be identified based on the pitch intervals between the most frequent notes in the recording and their rate of occurrence. For example, in the excerpt in Figure 6.14, the drone plays the tonic alongside the lower and upper fifth. Thus, a fifth relationship between two frequent notes might serve as a good indicator for the tonic.

In the study of Western music, templates learnt from music cognition experiments have been used for the related task of key detection, where a pitch histogram (derived from a symbolic representation of the musical piece) is matched against templates representing the probability of different pitch classes given a certain tonal context (Krumhansl, 2001). Approaches based on training a classifier to determine the key of a musical piece using chroma features automatically extracted from the audio signal have also been proposed (Gómez & Herrera, 2004). In this study, we propose a classification approach to automatically learn the best set of rules for selecting the tonic, based on the pitch intervals between the most frequent notes in the piece

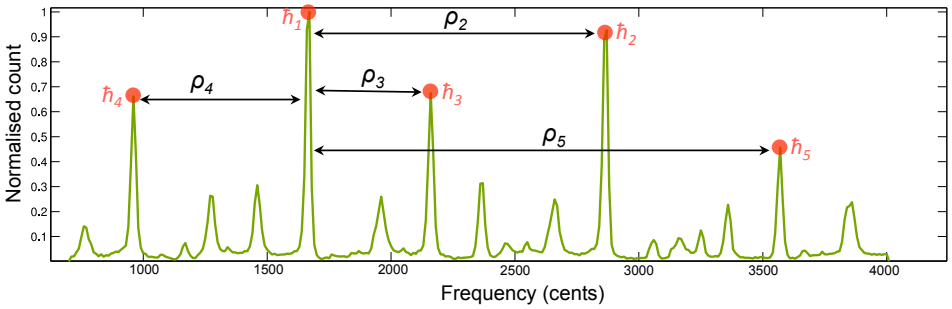

Figure 6.16: Illustration of computing ρ_2–ρ_5 from a pitch histogram.

and their relative rate of occurrence (as indicated by the magnitude of the peaks of the pitch histogram).

We start by annotating for each piece the rank r_T of the tonic (in terms of peak magnitude) out of the top 10 peaks $\hbar_i$ of the pitch histogram. Then, we encode the 10 tonic candidates as the distance in semitones between every candidate $\hbar_i$ and the highest candidate in the histogram $\hbar_1$. This gives us a set of features ρ_i ($i = 1 \ldots 10$), where ρ_i represents the distance (in semitones) between $\hbar_i$ and $\hbar_1$. The features ρ_i and the annotated rank of the tonic r_T are used to train a classifier for selecting the tonic. That is, we pose the task of tonic identification as a classification problem where we have 10 classes (10 candidates) and the classifier must choose the rank of the candidate corresponding to the tonic. Note that for all files in our collection the tonic was always amongst the top 10 peaks $\hbar_i$ of the pitch histogram. An illustration of computing ρ_2–ρ_5 (by definition $\rho_1 = 0$) from a pitch histogram is provided in Figure 6.16.

For classification we use the Weka data-mining software (Hall et al., 2009). We start by performing attribute selection using the *CfsSubsetEval* attribute evaluator and *BestFirst* search method (Hall, 1999) with 10-fold cross validation, only keeping features that were used in at least 80% of the folds. The selected features are: ρ_2, ρ_3, ρ_5, ρ_6, ρ_8 and ρ_9. Then, we train a C4.5 decision tree (Quinlan, 1993) in order to learn the optimal set of rules for selecting the tonic based on the pitch intervals between the tonic candidates. Note that we also evaluated other classification algorithms, namely support vector machines (SMO with polynomial kernel) and an instance-based classifier (K*) (Witten & Frank, 2005). However, the accuracy obtained using the decision tree was significantly higher (6% better than the SVM and 5% better than K*), and so for the rest of the study we will focus on the results

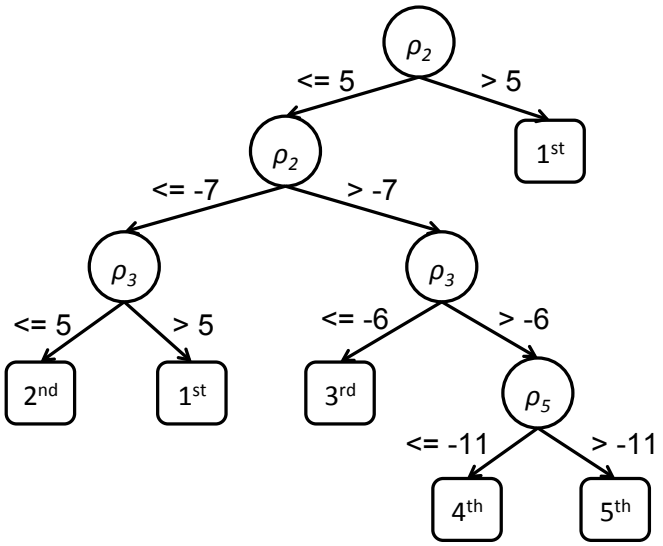

Figure 6.17: Obtained decision tree for tonic identification.

obtained using this classifier. Additionally, using a decision tree has the advantage that the resulting classification rules can be easily interpreted and, as shall be seen, are musically meaningful.

The resulting tree is presented in Figure 6.17. As it turns out, only 3 features are finally used: ρ_2, ρ_3 and ρ_5. Another interesting observation is that the pitch intervals used by the tree for making decisions correspond quite well to the intervals between the notes commonly played by the drone instrument: 5 (i.e. 500 cents) corresponds to the interval between the lower *Pa* and the tonic *Sa*, and 7 (700 cents) to the interval between the *Sa* and upper *Pa*. Note that a distance of 500 cents may also correspond to the distance between the *Sa* and upper *Ma*, which might be a cause for confusion in our system, and we will assess this when we analyse the results.

Examining the rules of the tree, we see that the most important relationship is between the top two peaks of the histogram (ρ_2). When the second highest peak is more than 500 cents above the highest peak, the latter is chosen as the tonic. Examining the data we found that this almost always corresponds to one of two cases – either the second peak is found at *Pa* (i.e. *Pa* tuning) or it is found at *Sa* one octave above the tonic. Branching left, the tree checks whether the highest peak is actually *Pa* (700 cents above the tonic). To confirm this it checks if the third peak is found 500 cents above the

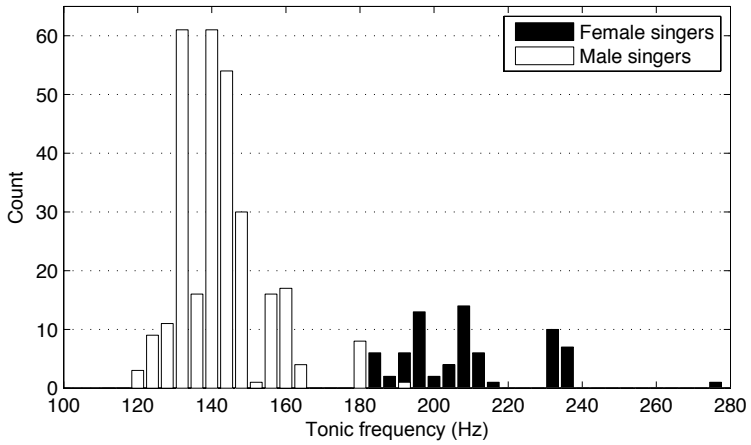

Figure 6.18: Distribution of tonic frequency for male and female vocal performances in our music collection.

highest peak (thus corresponding to *Sa* one octave above the tonic). In this case the highest peak is indeed *Pa*, and the second highest peak is the tonic. Otherwise, we have a case of *Ma* tuning (the second peak is tuned to *Ma*), and the highest peak is the tonic. Similar interpretations can be made for the remaining rules of the tree.

6.4.3 Evaluation methodology

Music collection

The music collection used to evaluate the proposed approach was compiled as part of the CompMusic project (Serra, 2011). It consists of 364 excerpts of Indian classical music including both Hindustani (38%) and Carnatic (62%) music. The excerpts were extracted from 231 unique performances by 36 different artists, including both male (80%) and female (20%) singers. Every excerpt is 3 minutes long, and extracted from either the beginning, middle or end of the full recording. For recordings longer than 12 minutes we extract 3 excerpts (1 from the beginning, 1 from the middle and 1 from the end), for shorter recordings a single excerpt from the beginning of the piece is used. Including excerpts from sections other than the beginning of the piece is important, since in both the Hindustani and Carnatic music traditions different sections of a performance can have very different acoustic characteristics. In Figure 6.18 we display the distribution of tonic frequencies in our collection for male and female singers.

Annotation procedure

The tonic frequency for each excerpt was manually annotated. To assist the annotation process, we used the candidate generation part of our proposed method to extract 10 candidate frequencies for the tonic in the range of 110 Hz to 300 Hz. The annotator could then listen to the candidate frequencies one by one together with the original recording in order to identify the tonic frequency. Note that for all excerpts in our collection the true tonic frequency was present amongst the 10 candidates provided by the system.

It is worth noting that as part of the annotation process, the listener must determine the octave in which the tonic is played. Since the drone instrument may play the tonic pitch in two octaves simultaneously, the octave of the tonic is determined by the vocal range of the singer rather than the drone instrument directly. Whilst in most cases the correct octave is fairly unambiguous for vocal performances, we encountered a small number of cases in which determining the octave of the tonic was more difficult. In future work, we intend to study the relation between performer and drone instrument in greater depth, as well as conduct listening tests to assess the degree of agreement between listeners when asked to determine the octave of the tonic.

6.4.4 Results

We evaluate the proposed classification-based approach using 10-fold cross validation. The experiment is repeated 10 times, and the average results for all 10 repetitions are reported. In Figure 6.19 we present the classification accuracy obtained for our collection of 364 excerpts, as well as a breakdown of the results based on the musical style (Hindustani/Carnatic) or the gender of the lead performer (male/female).

We see that the proposed approach obtains a high classification accuracy (hence tonic identification accuracy) of 93% for our complete collection. Importantly, since the allowed tonic frequency range spans more than one octave, it means we are correctly identifying not only the pitch-class of the tonic, but also the octave in which it is played. Next, we examine the results depending on the musical style. We see that we have almost perfect classification for Hindustani music (98%), whilst for Carnatic music the performance is somewhat lower (90%). When examining the data, we noted that in the Carnatic excerpts there are more cases where the Tanpura is quite soft (in terms of loudness). Consequently, this results in frames where the pitch corresponding to the tonic does not have a prominent peak

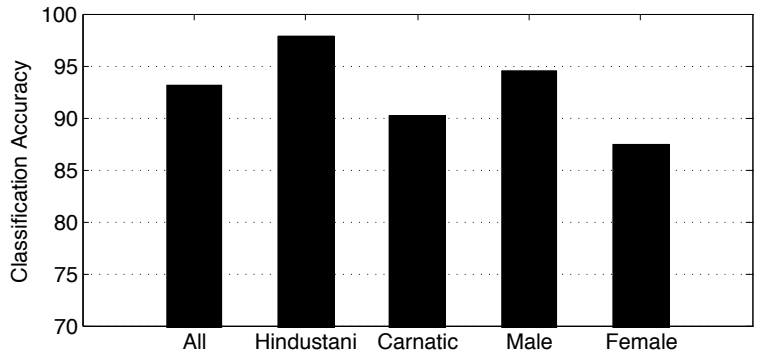

Figure 6.19: Classification accuracy of the proposed approach. All excerpts: 93%, Hindustani: 98%, Carnatic: 90%, male: 95% and female: 88%.

in the salience function. This in turn means the peak of the pitch histogram which corresponds to the tonic has a fairly low rank, leading to incorrect identification of the tonic.

When considering identification accuracy as a function of the gender of the lead performer, we see that the system performs better for pieces performed by male singers compared to those performed by female singers. A possible cause for this is the different amount of male and female performances in our collection. Since there are considerably more male performances, the rules learnt by the system are better suited for identifying the tonic in this type of musical material. Another factor that was identified as influential was the frequency range used to compute the pitch histogram. Whilst our frequency range covers the entire range in which we expect to find the tonic for both male and female singers, for high frequency tonics this range will not include the higher *Sa* one octave above the tonic. As it turns out, the presence of a higher *Sa* is one of the cues used by the system, and for many female excerpts it is outside the range of the pitch histogram. In the future, we intend to experiment with different frequency ranges for the pitch histogram, as well as consider separate ranges for male and female performances to see whether tonic identification accuracy can be improved by including this extra piece of information prior to classification.

As a final step in our analysis of the results, we checked what types of errors were the most common in our evaluation. We found that for male singers the most common error was selecting the higher *Pa* or *Ma* as the tonic, whilst for female singers it was selecting the lower *Pa* or *Ma*. This is understandable,

since these are two important notes that are often played by the drone instrument in addition to the tonic. The difference in tonic frequency for male and female singers, together with the frequency range used for the pitch histogram, explains why for male singers we erroneously select a higher note, whilst for female singers we select a lower one. Additionally, for female singers we found that the confusion was often caused due to the use of *Ma* tuning (*Sa* - *Ma* - *Sa*) of the drone instrument. If the higher *Sa* is not present, the *Ma* tuning is equivalent to a rotated version of *Pa* tuning, resulting in the wrong rule being applied.

6.4.5 Conclusion

In this section we presented a novel approach for tonic identification in Indian classical music that exploits the salience function presented in Chapter 3 of this thesis. The salience function is used to perform a multipitch analysis of the audio signal, in which the predominant pitches in the mixture are used to construct a pitch histogram representing the most frequently played notes in the piece. In this way, our representation also captures the notes played by the drone instrument, and not only the pitch of the lead performer. Using a classification approach, we were able to automatically learn the best set of rules for tonic identification given our pitch histogram representation. The resulting decision tree was evaluated on a large collection of excerpts consisting of a wide selection of pieces, artists and recording conditions, and was shown to obtain high tonic identification accuracy. Importantly, the approach is suitable for both Hindustani and Carnatic music, male and female performances, and only requires a short excerpt of the full performance. In addition, the rules learnt by the system are easy to interpret and musically coherent. Following the presentation of the results, we examined the types of errors most commonly made by the proposed tonic identification method, and the main causes for these errors where identified. We proposed some directions for future work, including a study of tonic octave perception, considering different frequency ranges for the pitch histogram in our proposed method, and devising gender-specific tonic identification approaches.

A continuation of this study was carried out by Gulati (2012), who showed that the amount of octave errors committed by the proposed method can be reduced by incorporating information about the pitch range of the lead singer. As noted earlier, the drone instrument often plays the tonic *Sa* in at least two octaves, and so the actual octave of the tonic depends on the

pitch range used by the singer. Gulati uses a classification approach based on the one proposed in this study, except that he only uses the classifier to determine the pitch class of the tonic (i.e. without octave information). Then, in a second stage, he uses the melody extraction algorithm presented in this dissertation (Section 4.2) to extract the f_0 sequence of the lead singer, and uses this sequence to determine the actual octave of the tonic based on the pitch range employed by the singer. In this way he shows how the signal processing blocks presented in this thesis can be exploited twice for tonic identification in Indian classical music: the salience function presented in Chapter 3 can be used to identify the tonic pitch class, and the complete melody extraction algorithm presented in Chapter 4 can be used to determine the exact octave of the tonic by analysing the pitch sequence of the lead singer.

6.5 Transcription of accompanied flamenco singing

In this final application scenario we describe how the melody extraction algorithm proposed in this thesis can be combined with a note segmentation and labelling method to produce a symbolic transcription of the melody directly from the audio signal. In the study presented here we focus on flamenco music: by working towards the automatic transcription of music traditions that do not have standardised symbolic notation and large corpora of digitised scores, we can open the door to a variety of large scale musicological studies that were previously unfeasible for these music traditions. Still, it should be noted that the transcription solution proposed here could also be applied to other music traditions (including Western classical and popular music), and we believe it is an important step towards the facilitation of large-scale computational music analysis in general.

6.5.1 Introduction

Flamenco is a music tradition which originates mostly from Andalusia in southern Spain. The origin and evolution of the different flamenco styles (*palos*) have been studied by a variety of disciplines including ethnomusicology, literature and anthropology. Since flamenco music germinated and nourished primarily from a singing tradition, the singer's role in many flamenco styles is dominant and fundamental. Often, the singer is accompanied by the flamenco guitar; other common types of accompaniment include

percussive instruments (e.g. the *cajón*), the clapping of hands and stamping of feet. Due to its oral transmission, there are no written scores in flamenco music. Flamenco experts have invested considerable amounts of time and effort into producing manual transcriptions after listening to live performances or field recordings, as a means to catalogue, classify and imitate the most relevant performers and their stylistic traits (Fernandez, 2004; Hoces, 2011; Hutardo & Hutardo, 2002, 1998). As pointed out by Toiviainen & Eerola (2006) and Lesaffre et al. (2004) in other contexts, whilst manual analyses can provide very accurate information and incorporate expert knowledge, they are also very time consuming and could contain a degree of subjectivity or even errors. The same is true in the case of flamenco transcription: first, there is no consensus regarding the most adequate transcription methodology. For instance, Donnier (1997) proposes the adaptation of plainchant neumes, whilst Hutardo & Hutardo (2002, 1998), on the contrary, forcefully argue for the use of Western notation. Second, even if we agree on the use of a certain format, there will still be a degree of subjectivity in the transcription process, given the high degree of ornamentation in flamenco music. By applying MIR technologies to flamenco music we can work towards the solution of the aforementioned issues. First, it could help bring about a standard methodology for flamenco description, transcription and comparative analysis, and support the formalisation of expert knowledge. Second, it could facilitate the study of large music collections, which (unlike Western classical music for which there are large corpora of digitised scores) has not been possible before.

In this section we present a system for the melodic transcription of accompanied flamenco singing (Gómez et al., 2012). The system is based on combining the melody extraction algorithm proposed in this dissertation (Section 4.2) with a note segmentation and labelling method (Gómez & Bonada, 2013; Janer et al., 2008) to produce a symbolic transcription of the melody directly from the audio signal. For evaluation we focus on the *fandango* style (Fernandez, 2011), one of the principal palos in flamenco music typically consisting of a lead singer and guitar accompaniment.

Automatic transcription is a key challenge in MIR (cf. Section 1.4.3). Within the large topic of automatic music transcription, a specifically challenging task is that of transcribing the singing voice. Even though some successful approaches for singing transcription have been proposed (De Mulder et al., 2003; Ryynänen, 2006), the human voice is still one of the most complex instruments to transcribe, given its continuous character and the variety of pitch ranges and timbres it encompasses. Additional challenges in fla-

menco arise from the (sometimes low) quality of recordings, the acoustic and expressive particularities of flamenco singing, its ornamental and improvisational character and its yet to be formalised musical structures (Mora et al., 2010).

6.5.2 Transcription method

Predominant f_0 estimation

In the first stage of the method, the f_0 sequence of the voice is extracted from the audio signal. This is accomplished using the melody extraction algorithm proposed in Section 4.2 of this dissertation. In order to assess the influence of our algorithm on the final transcription, we also consider two alternatives for this stage. In the first alternative, we use the same algorithm only now we manually adjust a small number of the algorithm's parameters for each excerpt in our music collection, in order to maximise the overall accuracy of the extraction. The parameters adjusted are the minimum and maximum allowed frequencies for the melody (thus reducing potential octave errors), and the voicing parameter ν (cf. Section 4.2.3). The second alternative is a source separation based melody extraction method (Gómez et al., 2012). The method is comprised of two steps: first, the melody source is separated from the audio mixture using a technique similar to the HPSS approach by Ono et al. (2010) (cf. Section 2.3.2). The technique is based on factorising the spectrogram of the audio mixture into three components: a percussive spectrogram, a harmonic spectrogram and a vocal spectrogram. The underlying assumption is that harmonic sounds (e.g. chords) are sparse in frequency and smooth in time, percussive sounds are smooth in frequency and sparse in time, and the vocal source is sparse both in time and frequency. It is important to note that, currently, this stage requires the voiced sections of the piece (i.e. the time intervals where the voice is present) to be manually labelled for training. Once the vocal signal is separated from the mixture, the f_0 sequence of the voice is extracted from the separated signal by applying a modified version of the YIN pitch tracker (de Cheveigné & Kawahara, 2002; cf. Section 2.2): instead of taking the minimum of the mean normalised difference function at each frame, dynamic programming is used to obtain a smooth f_0 sequence by treating the difference function as a cost function and searching for the path (i.e. f_0 sequence) which minimises the total cost, subject to a pitch continuity constraint. Further details can be found in Gómez et al. (2012).

Note segmentation and labelling

To convert the continuous f_0 sequence of the melody into a discrete set of notes, we use the note segmentation and labelling method proposed by Janer et al. (2008) which was later adapted for transcription of flamenco a capella singing by Gómez & Bonada (2013). By segmentation we mean splitting the continuous f_0 sequence into discrete notes with onset and offset times. By labelling we mean assigning a pitch (semitone) label to each note. The method is comprised of three main steps: tuning frequency estimation, short note transcription and iterative note consolidation and tuning frequency refinement. A brief summary of each step is provided below. For further details the reader is referred to Gómez & Bonada (2013); Gómez et al. (2012); Janer et al. (2008).

As a first step, the tuning frequency of the melody must be estimated, as it will not necessarily be tuned to 440 Hz. The melody is assumed to follow an equal-tempered scale, and it is assumed that the tuning frequency remains constant for a given excerpt. The tuning frequency is estimated by measuring the frequency deviation in cents of each f_0 value from an equal-tempered scale tuned to 440 Hz. A histogram of the deviations is then computed, giving greater weight to frames where the f_0 is relatively stable (indicated by a low value of the f_0 derivative). The maximum of the histogram represents the deviation (in cents) c_{tuning} of the tuning frequency from 440 Hz, and so the tuning frequency can be estimated as

$$f_{\text{tuning}} = 440 \cdot 2^{\frac{c_{\text{tuning}}}{1200}}. \tag{6.7}$$

Once the tuning frequency is estimated, the f_0 sequence is segmented into short notes. Using dynamic programming, the note segmentation that maximises a set of probability functions is found. This translates into finding the optimal path through a two-dimensional matrix whose dimensions represent analysis frames (i.e. time) and pitch (in semitones). In Figure 6.20, we illustrate how the best path through a node with frame index k and note index j is determined. All possible note durations between d_{min} and d_{max} are considered, as well as all possible transitions between notes. The chosen segmentation (path) is coloured in dark grey. The likelihood of a certain path is given by the product of the likelihoods of its constituent notes multiplied by the likelihood of every note transition. For flamenco transcription no a priori knowledge of the melodic contour is assumed, meaning all note transitions are equally likely. The likelihood of each note is determined based on a set of criteria related to note duration, pitch value, voicing, and

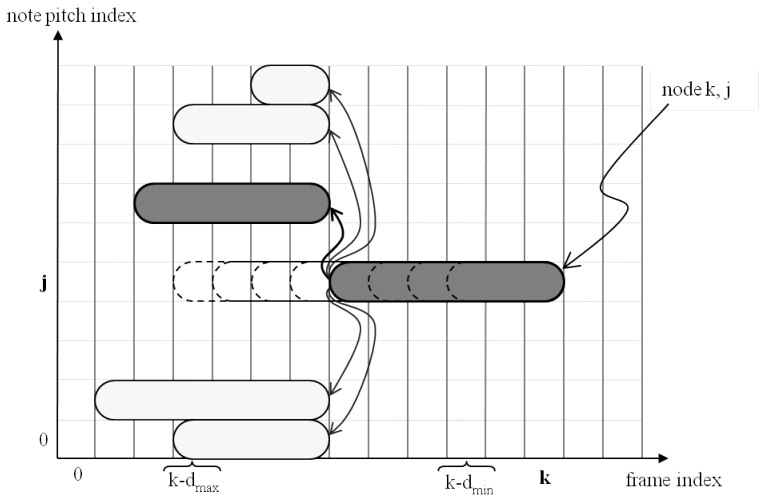

Figure 6.20: A two dimensional matrix used to segment the voice f_0 sequence into short notes.

note stability as indicated by low-level timbre and energy features computed directly from the audio signal (cf. Gómez & Bonada (2013); Gómez et al. (2012) for further details).

In the third step, short notes with the same pitch are consolidated if their combination results in a longer, stable note. Notes with significant energy/timbre changes between them, which could be indicative of phonetic changes, are not consolidated. Then, the deviation of each note from the original f_0 sequence is computed and the deviations (this time per-note rather than per-frame) are used to re-estimate the tuning frequency as explained earlier (the deviation of each note is now weighted by the estimated note energy and duration). The note labels are updated according to the new tuning frequency, and this process of consolidation/tuning estimation is repeated iteratively until no further consolidations take place. An example of the output of the complete transcription system, which has been incorporated into a visualisation tool, is provided in Figure 6.21.

6.5.3 Evaluation methodology

Music collection

We compiled a collection of 40 excerpts of fandango performances with a duration of approximately 1 minute per excerpt (the collection was com-

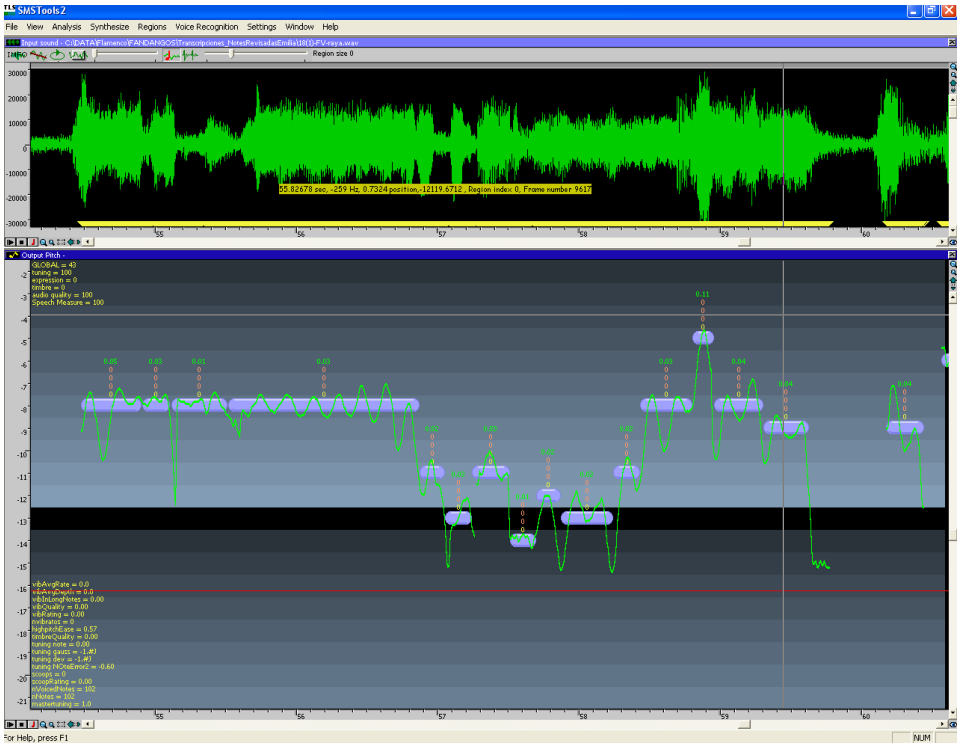

Figure 6.21: Visualisation tool for melodic transcription. Top pane: audio waveform. Bottom pane: extracted f_0 sequence (in green) and the segmented note sequence (in a piano roll representation).

piled as part of the COmputational analysis of FLAmenco music (COFLA) project[7]). All excerpts consist of a lead singer with guitar accompaniment, and the collection includes a variety of male and female singers and recording conditions. In total, the excerpts contain 3285 notes.

Ground truth annotation

The notes in each excerpt were manually annotated by a musician with limited knowledge of flamenco music, in order to avoid implicit knowledge being applied during the transcription process. The annotation was performed using the interface shown in Figure 6.21. Since transcribing every note manually is very time consuming, the annotator was provided with the output of the proposed transcription method (using our melody extraction

[7] http://mtg.upf.edu/research/projects/cofla

algorithm with manually adjusted parameters for the f_0 estimation stage) as a starting point. The annotator could then listen to the original recording along with a synthesised version of the transcription, and edit the pitch and duration of each note until they were satisfied with the transcription. The criteria used to differentiate ornaments and pitch glides from actual notes were first discussed with two flamenco experts, so that the annotator could follow a well-defined and consistent strategy.

Frame-level evaluation measures

Given the ground truth note sequence and the system's estimated note sequence, we start our evaluation by comparing the two sequences on a per-frame basis. That is, we compare the sequences in the same way we compare two f_0 sequences for melody extraction evaluation (cf. Section 2.4), except that now the pitch value of each frame is determined by the nominal pitch of the note in that frame. As such, we can use the same evaluation measures used to evaluated melody extraction, which were described in detail in Section 2.4.1 of this dissertation. Since the hop size used for analysis in this study is 5.8 ms, this translates into a total of 375,575 analysis frames for the complete collection. Note that the melody extraction algorithm presented in this dissertation uses a hop size of exactly half the aforementioned value (2.9 ms), and so the output could be easily downsampled to fit the hop size used in this study.

Before considering the results, there are a couple of aspects of the evaluation we must be aware of. First, due to the music material being studied (which includes old and field recordings), there are no multitrack versions of the recordings and so we cannot obtain the separated voice track. This in turn means that whilst we are able to manually annotate the note sequence of the singer, we cannot annotate the continuous f_0 sequence of the voice as done for melody extraction evaluation. Consequently, we cannot evaluate the performance of the melody extraction algorithms separately from the performance of the complete transcription system. Thus, any observed errors in the transcriptions will represent the combined errors introduced by the two stages of the system (f_0 estimation and note segmentation/labelling). Second, since the transcription system does not provide pitch estimates for frames determined as unvoiced, incorrect voicing detection will also influence the pitch accuracy measure (but not the overall accuracy measure).

Note-level evaluation measures

In the second part of the evaluation, we compare the ground truth and transcribed note sequence on a note level. To do this, we use a standard set of measures for note-level transcription evaluation (Bello, 2003; Ryynänen & Klapuri, 2008a):

$$\text{Precision} = \frac{\#(\text{correctly transcribed notes})}{\#(\text{transcribed notes})} \tag{6.8}$$

$$\text{Recall} = \frac{\#(\text{correctly transcribed notes})}{\#(\text{reference notes})} \tag{6.9}$$

$$\text{F-measure} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{6.10}$$

To determine whether a transcribed note matches a ground truth note, we use the same criteria used in the note tracking sub-task of the MIREX multiple fundamental frequency estimation and tracking task[8]: a transcribed note is considered to match a ground truth (reference) note if its pitch is within half a semitone (±50 cents) of the reference pitch, its onset is within ±50 ms of the reference note's onset and its offset is either within ±50 ms of the reference note's offset or inside a time interval whose duration is 20% of the reference note's duration centred on the reference note's offset, whichever is greater. Following the methodology proposed in MIREX, we also compute the evaluation measures using only the pitch and onset criteria for finding note matches, ignoring note offsets. The motivation for this is twofold: first, note offsets are considerably harder to detect automatically. Furthermore, our ability to perceive offsets can be strongly affected by the duration of note decay, reverberation and masking (Bregman et al., 1994; Mason & Harrington, 2007). Second, consider the case where the system consolidates two short notes which are separate in the ground truth (or vice versa) – even though we have detected a note with the correct pitch and onset, if we use the offset criterion it will not match neither the first note reference nor the second reference note, resulting in three transcription errors (missing two reference notes and transcribing a wrong note) when we might want to consider it as just one error.

[8] http://www.music-ir.org/mirex/wiki/2012:Multiple_Fundamental_Frequency_Estimation_&_Tracking

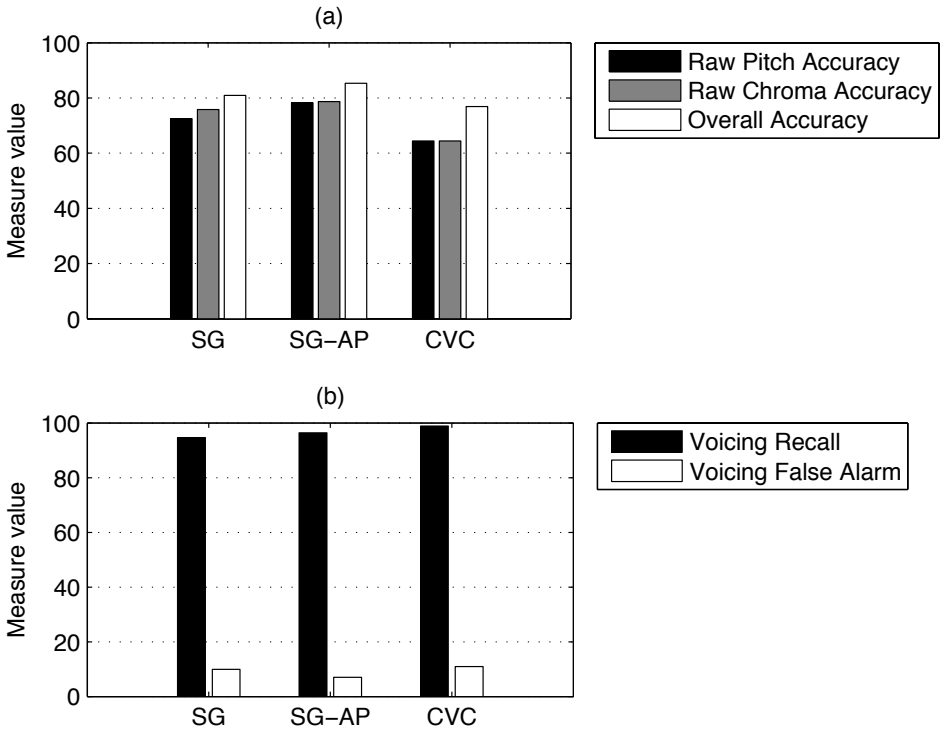

Figure 6.22: Frame-based evaluation results when using our melody extraction algorithm (SG), our algorithm with manually adjusted parameters (SG-AP) or a source separation based algorithm (CVC) for the first stage of the transcription system. Plot (a): raw pitch accuracy, raw chroma accuracy and overall accuracy. Plot (b): voicing recall and voicing false alarm rates.

6.5.4 Results

Frame-based transcription accuracy

The frame-based transcription accuracy of the three alternative approaches (as indicated by the five evaluation measures described in Section 2.4.1) is presented in Figure 6.22. We use SG to denote our algorithm (as before), SG-AP to denote our algorithm with manually adjusted parameters and CVC for the source separation based algorithm. We see that all three approaches obtain an overall accuracy above 75%. Still, using the melody extraction algorithm proposed in this thesis (SG) for the first stage of the transcription system results in a higher overall accuracy (81%) compared to using the separation based approach (76%, the difference is statistically

significant according to a two-sample t-test with $p < 0.05$). Furthermore, our algorithm is fully automatic whereas the separation based approach (currently) requires manual labelling of the voiced sections in each excerpt. If we allow some manual adjustment of the parameters of our algorithm for each excerpt (SG-AP), we can bring the overall accuracy of the transcription up to 85%. Whilst the separation-based approach also provides good performance, the results of the evaluation are good evidence in support of the *understanding without separation* strategy adopted in this thesis (cf. Section 1.4.5).

For all three approaches we note that the overall accuracy is greater than the raw pitch accuracy. In the case of CVC and SG-AP, since the voicing detection is done manually or just semi-automatically, we can expect to have near-perfect voicing detection (as confirmed in plot (b) of Figure 6.22), and so this is to be expected. However, we see that our fully automatic approach (SG) also obtains very good voicing detection results and consequently a higher overall accuracy. This suggests that the automatic voicing detection method of our algorithm works very well for this type of music material. We see that regardless of the f_0 estimation approach used, the transcription system makes very few octave errors (when using SG, which makes more octave errors compared to the alternative approaches, the amount of frames with octave errors in still just 3% of the total number voiced frames). It is likely that both stages of the system contribute to this good result – the f_0 estimation algorithms make few octave errors to begin with, and these are further reduced by the dynamic-programming-based note segmentation and consolidation stage.

Finally, it is interesting to note that the results obtained are slightly better than those obtained for a dataset of a capella singing using the same note segmentation/labelling algorithm with a monophonic pitch tracker for the first stage of the system (Gómez & Bonada, 2013). This is most likely due to two factors: first, accompanied flamenco singing tends to be less ornamented compared to a capella singing styles, meaning melodies extracted from accompanied recordings should be easier to segment correctly. Second, in the a capella case the singer does not have any accompaniment to help them maintain the tuning frequency constant, and this can lead to a gradual change of tuning throughout the recording which in turn degrades the performance of the note labelling stage. It also suggests that our melody extraction algorithm works very well for this type of music material.

Algorithm	Precision	Recall	F-measure
SG	0.41	0.46	0.43
SG-AP	0.47	0.54	0.50
CVC	0.16	0.22	0.19

Table 6.6: Note transcription accuracy with the offset criterion for matching. Results obtained using our melody extraction algorithm (SG), our algorithm with adjusted parameters (SG-AP) and the source separation based algorithm (CVC).

Algorithm	Precision	Recall	F-measure
SG	0.53	0.60	0.57
SG-AP	0.60	0.69	0.64
CVC	0.30	0.40	0.35

Table 6.7: Note transcription accuracy without the offset criterion for matching. Results obtained using our melody extraction algorithm (SG), our algorithm with adjusted parameters (SG-AP) and the source separation based algorithm (CVC).

Note-level transcription accuracy

We now turn to the note-level evaluation results. We start by examining the results when considering all three criteria (pitch, onset and offset) for note matching, presented in Table 6.6. We see that when we include the offset criterion for note matching the results are not very high in general. As mentioned earlier, the difficulty in determining note offsets together with fragmentation/consolidation errors are likely to be (in part) responsible. Also, we note that there is a considerable difference in performance between using our melody extraction algorithm and using the source separation based approach. We see that this difference is much greater compared to the difference between the approaches according to the frame-based measures. This is likely due to two reasons: first, the better pitch accuracy of our algorithm (cf. Figure 6.22) results in better note labelling. Second, as mentioned earlier, the voiced segments have to be annotated manually in the separation based approach. For these manual annotations only relatively long voiced/unvoiced segments were labelled, meaning many short unvoiced segments (which are important for correct note offset detection and for identifying note transitions) were not labelled correctly. As evidenced by the results, accurate voicing detection (or annotation if done manually) seems to be crucial for correct note transcription.

Next, we examine the results obtained when ignoring the offset criterion for note matching, presented in Table 6.7. As expected, the results go up for all approaches. Due to the voicing issue mentioned above, the results for CVC are still quite low. Using our melody extraction algorithm for the first stage of the system we obtain an F-measure of 0.57, which is increased to 0.64 if we manually adjust some of the algorithm's parameters. Whilst there is clearly room for improvement, the results are encouraging: the highest F-measure obtained by current onset detection algorithms for solo singing voice material (as indicated by the MIREX 2012 onset detection results[9]) is 0.56, and Ryynänen & Klapuri (2008a) report an F-measure of 0.54 for singing voice transcription (from polyphonic music) when ignoring offsets and using a more relaxed onset threshold of ±150 ms. Note that since different music collections were used for evaluation in those experiments the results are not directly comparable to our study, and are only provided to give an idea of the current performance achieved for singing voice transcription.

A second example of the output produced by our proposed transcription system (using SG for the first stage) is provided in Figure 6.23, where the ground truth note sequence is displayed in blue, the transcribed note sequence in red and the extracted f_0 sequence (before segmentation) is displayed in green. We see that the estimated tuning frequency does not match the ground truth perfectly, but it is well within the allowed threshold. Whilst the overall the transcription appears to match the ground truth quite well, we can still observe different types of errors: at times 22 s and 29 s we see very short outlier notes which are the result of incorrect f_0 estimation. Between seconds 25–26 we see four ground truth notes that have been mistakenly consolidated into a single note by the automatic segmentation method, and at the very end of the sequence we see a single note in the ground truth that has been separated into two notes by the segmentation method. Between seconds 17–18 we observe an error which exemplifies the added difficulty in transcribing flamenco singing: a fast pitch modulation in the f_0 sequence is annotated in the ground truth as four short notes forming a trill (MIDI notes 67 68 67 68), but erroneously transcribed as a single long note with MIDI pitch 67. These observations help us understand why a transcription which looks overall quite good can still obtain relatively low evaluation results – the transcription in Figure 6.23 obtains a Precision of 0.54, Recall of 0.68 and F-measure of 0.60 (when ignoring the offset criterion).

[9] http://nema.lis.illinois.edu/nema_out/mirex2012/results/aod/resultsperclass.html

Figure 6.23: Example of a transcription produced by the proposed system using SG for the first stage (f_0 estimation).

Error analysis

Whilst it is not possible to quantitatively measure the amount of errors introduced by each stage of the system (f_0 estimation and note segmentation/labelling) separately, we can still gain some insight into the influence of each stage on the final result by manually inspecting the output of each stage. When inspecting the output of the first stage (singing voice f_0 estimation), it turns out that even though we already obtain good voicing detection performance, voicing seems to be the main aspect to improve. The main type of error is voicing false positives, which sometimes appear during melodic guitar segments or in the presence of short unvoiced phonemes (e.g. fricatives). Voicing false negatives are less common but can occur if the dynamic range of the singing is very high or if strong instrumental accompaniment disrupts the correct tracking of the melodic pitch contour.

When inspecting the output of the second stage (note segmentation), we see that most of the errors occur for short notes: either they are mistakenly consolidated into a single longer note (whilst the annotation consists of several consecutive short notes), or the other way around. This was found to occur particularly when the guitar accompaniment is relatively strong, in which case the energy envelope of the signal captures the guitar, making

onset estimation more difficult. Finally, we note that some of the errors occur due to wrong pitch labelling of very short notes, which is caused by the rapidly changing f_0 contour. This highlights the added difficulty in obtaining accurate note transcriptions for flamenco singing, due to its ornamental character and continually varying f_0, which could easily be confused with deep vibrato or pitch glides. The instability of the f_0 contour in flamenco singing can be observed in Figures 6.21 and 6.23 (the green curve in both figures).

6.5.5 Conclusion

In this section we demonstrated how the melody extraction algorithm proposed in this dissertation can be combined with a note segmentation and labelling method to produce a completely automatic melody transcription system. We evaluated the approach for transcribing accompanied flamenco singing, a particularly difficult genre for vocal transcription due to the highly unstable pitch contour which characterises the singing in this musical style. The proposed approach was shown to obtain promising results (a per-frame overall accuracy of 81% and note-level F-measure of 0.57), and we saw that we get better transcription results when using our melody extraction algorithm for the first stage compared to using a source separation based approach. We also noted that the results are comparable to (and even better than) previous results obtained for monophonic flamenco singing transcription. The main sources of transcription errors were identified: in the first stage (f_0 estimation) the main issue is voicing detection (e.g. missclassification of the guitar as voice). In the second stage (note segmentation), we observed that most of the errors occur when segmenting short notes and labelling notes with an unstable f_0 contour.

One important limitation of this study is that we only have manual annotations on a note level (quantised into 12 semitones), and not the continuous f_0 ground truth. This meant that we were not able to evaluate the accuracy of each stage of the transcription system separately. In the future we plan to evaluate the proposed system on data for which both stages can be evaluated separately. Furthermore, it would be interesting to evaluate the approach on other musical genres as well. For genres which typically have simpler melodies (e.g. pop music) we can expect to obtained even better note transcription results (Gómez & Bonada, 2013). Still, considering that we already obtain results comparable to (and in some cases better than) those obtained for monophonic flamenco singing transcription (Gómez &

Bonada, 2013), we can already say that the method presented here immediately opens the door to a variety of computational analyses that so far have only been conducted on monophonic material, such as a comparative analysis of different flamenco singing styles (Cabrera et al., 2008) or the automatic detection of frequent and representative ornamentations (Gómez et al., 2011).

6.6 Chapter conclusion

In this chapter we presented a number of systems that are all based on the output of the melody extraction algorithm presented in this dissertation or one of its intermediate blocks. Our goal was to show that even though there is still room for improving the algorithm itself, it is already sufficiently accurate to be exploited in a variety of application domains.

In Section 6.2 we started by demonstrating how the algorithm can be exploited for automatic similarity-based music retrieval. We studied two related retrieval tasks: version identification and query-by-humming. For version identification we used the algorithm to extract two different tonal representations of a song: the melody and the bass line. Using a state-of-the-art sequence matching algorithm, we evaluated the utility of these representations for version retrieval on a relatively large music collection. We showed that they indeed carry useful (and complementary) information, obtaining MAP results comparable to, and in some cases higher than, other state-of-the-art version identification systems. We saw that the results are still outperformed by those obtained when using the harmony-related HPCP descriptor, but also noted that to some extent the melody and bass line are already represented by the HPCP. We also presented a classification approach for descriptor fusion which improved retrieval results compared to using just the melody or bass line representation. Next, we demonstrated how the proposed version identification method can be adapted to perform fully automatic query-by-humming of polyphonic audio collections. The approach was shown to obtain results comparable to those presented in previous studies, and current limitations were identified for future improvement. In particular, we noted that one important source of errors is not the system itself but rather the quality of the sung queries, and we saw that for well-tuned queries the system obtains very promising retrieval results. We also showed how performance can be further increased by including more than one version of each song in the target database. Finally, we noted that whilst there is still much work to be done in this area, the results demon-

strate that it is definitely feasible to develop a fully automated QBH system for polyphonic audio collections using the output of our melody extraction algorithm.

In Section 6.3 we demonstrated how the melody extraction algorithm can be used for genre classification. We used the algorithm to compute a set of melodic features (directly from the audio signal) based on pitch, duration, vibrato and contour typology. The melodic feature set was evaluated on three different datasets and was shown to outperform a baseline low-level timbral feature set based on MFCCs. Most importantly, we demonstrated that the classification accuracy can be improved by combining the two feature sets. This suggested that adding high-level melodic features to traditional low-level feature sets is a promising approach for genre classification. Another important aspect of the study is the fact that most of the melodic features proposed can be easily understood by humans. This meant that the classification results could be interpreted more easily, allowing us to make straight forward links between musical genres and melodic characteristics.

Next, in Section 6.4 we presented an application which exploits the salience function presented in Chapter 3 of this dissertation: a novel approach for tonic identification in Indian classical music based on a multipitch analysis of the audio signal. We used the peaks of the salience function to construct a pitch histogram representing the most frequently played notes in the piece, in this way capturing the notes played by the drone instrument. Using a classification approach, we were able to automatically learn the best set of rules for tonic identification given our pitch histogram representation. The resulting decision tree was evaluated on a varied collection of excerpts and was shown to obtain high tonic identification accuracy. Importantly, the approach was shown to be suitable for a variety of performance types and the rules learnt by the system were easy to interpret and musically coherent.

Finally, in Section 6.5 we demonstrated how the melody extraction algorithm can be combined with a note segmentation and labelling method to produce a completely automatic melody transcription system. We evaluated the approach for transcribing accompanied flamenco singing and obtained promising results (a per-frame overall accuracy of 81% and a note-level F-measure of 0.57), and we showed that using our algorithm provides higher transcription accuracy compared to a source separation based approach. We noted the limitation of the study due to the lack of ground truth annotations for the f_0 sequences of the singing voice, but were still able to identify the main sources of transcription errors through a manual inspection of the data.

Since we published our melody extraction algorithm, a number of applications which exploit it have also been developed by other researchers, in particular for computational music analysis. These include studies on computational analysis of Indian classical music, such as applications for tonic identification (Gulati, 2012), raga recognition (Koduri et al., 2012a) and intonation characterisation (Koduri et al., 2012b); flamenco music, such as melodic pattern detection (Pikrakis et al., 2012) and singing characterisation (Kroher, 2013); intonation characterisation of Beijing opera singing (Chen, 2013) and tonic identification in Turkish makam music (Şentürk et al., 2013).

Now that we have reached the end of this chapter, it makes sense to ask whether we have answered the question raised at the introduction of the chapter: is the melody extraction algorithm presented in this dissertation sufficiently accurate for developing applications which exploit its output? After presenting several functioning systems which obtain promising (and in some cases state-of-the-art) results in a variety of application domains, we think it is fair to say that, even though there is still room for improvement, the answer is yes.

Chapter 7

Conclusion

7.1 Introduction

At the outset of this dissertation we defined two hypotheses: first, that the characteristics of the pitch contour of the melody can be exploited for the identification of the melodic line within a polyphonic mixture, and consequently for the development of a melody extraction method based on an understanding-without-separation strategy. Second, that the mid-level representation obtained using melody extraction can be highly useful for developing applications for music description, retrieval and organisation. Based on these hypotheses we set ourselves two principal goals: to design a melody extraction algorithm which exploits melodic pitch contour characteristics to extract the pitch sequence of the melody as accurately as possible, and to develop a set of applications that demonstrate the utility of the algorithm for research and end-user applications.

Now we can say that these goals have been successfully met: in Chapters 3 and 4 we presented a novel melody extraction algorithm which exploits pitch contour characteristics to extract the f_0 sequence of the melody. The algorithm was evaluated in the MIREX international evaluation campaign and obtained the highest mean overall accuracy obtained by any algorithm to date for the current evaluation collections. Furthermore, we believe we have been successful in keeping each step of the algorithm conceptually simple and musically meaningful. Then, in Chapter 6 we described how to accomplish our second goal by presenting a number of applications which exploit the output of the proposed melody extraction algorithm. We showed how the algorithm can be used successfully for music retrieval, classification,

transcription and computational music analysis.

In addition to advancing the state of the art in melody extraction, we have made a special effort to ensure that the research community will be able to take advantage of the results of this thesis. We created an easy-to-use plug-in implementation of our algorithm, MELODIA, and made it freely available online. Within months of its announcement the plug-in has been downloaded over 4,000 times by people from all over the world (see Appendix A for a world map). In this way, not only have we provided researchers working on melody extraction with a state-of-the-art algorithm against which they can benchmark their own systems, we have also provided the wider MIR research community with a much needed tool that opens the door to a wide range of computational analyses of polyphonic music. As noted in Sections 1.5 and 6.6, several studies in which the algorithm is used (other than our own) have already been published, and we expect this number to grow as the plug-in becomes more widespread. Furthermore, it turns out that MELODIA is also quite popular outside the research community – more on this in final section of this chapter.

7.2 Summary of contributions and key results

We now present a summary of the contributions of this thesis. In addition to the brief summary already provided in Section 1.5 of Chapter 1, here we also provide a summary of the key results obtained in each chapter.

Discussion of "melody" and definition of "melody extraction". In Chapter 1 we saw that even after centuries of musicological research there is still no consensus regarding the definition of the term melody. We explained how this problem has been addressed by the MIR research community, and provided a clear and pragmatic definition for melody extraction: *fundamental frequency estimation of a single predominant pitched source from polyphonic music signals with a lead voice or instrument.* We explained the motivation for working on melody extraction and the challenges it entails, followed by an introduction to the key research areas related to melody extraction: music information retrieval, computational auditory scene analysis, automatic transcription and source separation. Finally we discussed the notion of *understanding without separation*, which has been the underlying strategy for the work presented in this dissertation.

Review of the state of the art in melody extraction. In Chapter 2 we provided a detailed review of the current state of the art in melody extraction. We noted that monophonic pitch tracking can be considered a precursor to melody extraction, but melody extraction requires novel methods to robustly process polyphonic signals and track the pitch of the melodic line. We identified two main strategies for melody extraction: salience based approaches and source separation based approaches, and described the different techniques applied in the main processing blocks of 16 melody extraction algorithms submitted to MIREX between 2005 and 2012. We saw that performance has been improving gradually over the past decade, and that the best performing approaches obtain a mean pitch accuracy of approximately 80% and mean overall accuracy of 75%. Finally we provided a case study where we highlighted the main challenges and sources of confusion for melody extraction algorithms.

Comparative evaluation of processing techniques for sinusoid extraction and salience computation. In Chapter 3 we described the first two blocks of our proposed melody extraction method. We compared different analysis techniques for sinusoid extraction and showed that we can improve spectral peak estimation accuracy by applying phase-based frequency and amplitude correction. We compared two spectral transforms (STFT and MRFFT) and showed that they perform similarly in terms of melody energy recall and frequency accuracy. For the second block of the method we proposed a salience function based on harmonic summation. We showed that the salience of the melody is enhanced when we apply an equal loudness filter, and our estimate of the exact melody frequency is improved when applying frequency correction. On the other hand, we noted that the MRFFT does not seem to provide a significant advantage over the STFT for our approach. Finally we conducted a grid search to optimise the parameters of the proposed salience function which was shown to provide a significant improvement in the results in the following chapter.

Novel melody extraction algorithm based on pitch contour characteristics. In Chapter 4 we presented the final two blocks of the proposed melody extraction method. We showed that by studying the distributions of different pitch contour characteristics we can identify features that distinguish melody contours from non-melody contours. We then explained how these features can be used to filter out non-melody contours, resulting in novel voicing detection and octave error minimisation methods. We also

proposed an alternative version for the algorithm's melody selection block based on a statistical model of contour features, and explained how to use the model to compute a "melodiness" index $\mathcal{M}(\mathbf{x})$. The model-free version of the algorithm with optimised salience function parameters was shown to outperform all other melody extraction algorithms participating in MIREX 2011. We complemented the results with a qualitative error analysis, revealing that the different characteristics of instrumental music make it harder to avoid octave errors. We evaluated the influence of individual algorithmic components on system performance, and noted that the interaction between different components can be important for maintaining high accuracies. A glass ceiling analysis confirmed that in most cases the proposed contour filtering process is successful at filtering out non-melody contours, though a further increase in accuracy could still be achieved by reducing the voicing false alarm rate of the approach. In addition, it was determined that to increase the potential performance of the method we would have to improve the contour formation stage, and possible methods for achieving this were proposed. Finally, we evaluated the proposed model-based version of the algorithm, and showed that the approach achieves pitch and chroma accuracies comparable to the model-free method, which is state-of-the-art. By combining the model-based approach with a voicing detection method, we were able to obtained satisfying overall accuracy values as well. Finally, we identified several research directions for improving the performance of the model-based approach including the addition of temporal constraints, incorporating more contour features and using genre specific feature distributions.

Study of the current challenges in melody extraction evaluation. In Chapter 5 we turned our focus to an important aspect of melody extraction that until now had not been studied properly – the reliability of the evaluation of melody extraction algorithms, as carried out in the MIREX AME task. We demonstrated how an offset between the ground truth and an algorithm's output can significantly degrade the results, the solution to which is the definition and adherence to a strict protocol for annotation. Next, we showed that the clips currently used are too short to predict performance on full songs, stressing the need to use complete musical pieces. Using generalisability theory, we showed that evaluation results based on just one of the early MIREX collection (ADC04, MIREX05 or INDIAN08) are not reliable due to their small size, and noted that the MIREX09 collections, though more reliable, do not represent all the real-world musical

content to which we wish to generalise. As a solution, we proposed the creation of a new and open test collection through a joint effort of the research community, and launched the Audio Melody Extraction Annotation Initiative[1]. Finally, we noted that the outcomes of the analysis in Chapter 5 do not invalidate the MIREX results – what they tell us is that we must be cautious if we wish to generalise from the MIREX results to the universe of all songs we would like to apply our algorithms to. We also noted that even with a larger collection we would still not be able to answer the following question: how good is *good enough*? To answer this question, we developed and evaluated the applications described in the following paragraph.

Melody-extraction-based applications for music retrieval, classification, transcription and computational music analysis. With the goal of demonstrating that the melody extraction algorithm presented in this dissertation is sufficiently accurate to be exploited both for research and for end-user applications, in Chapter 6 we presented a number of systems that are all based on the output of the algorithm or one of its intermediate blocks. We presented novel techniques in several domains, namely: version identification, query-by-humming, genre classification, tonic identification in Indian classical music and melodic transcription with a focus on accompanied flamenco singing.

In Section 6.2 we started by demonstrating how the algorithm can be exploited for automatic similarity-based music retrieval, with applications for version identification and query-by-humming. For version identification we used the algorithm to extract two different tonal representations of a song: the melody and the bass line. Using a state-of-the-art matching algorithm we showed that both representations carry useful (and complementary) information, obtaining MAP results comparable to (and in some cases higher than) other state-of-the-art version identification systems. We also presented a classification approach for descriptor fusion which improved retrieval results compared to using just the melody or bass line representation. Next, we demonstrated how the proposed version identification method can be adapted to perform fully automatic query-by-humming of polyphonic audio collections. The approach was shown to obtain results comparable to those presented in previous studies, and we noted that one important source of errors was not the system itself but rather the quality of the sung queries. We saw that for well-tuned queries the system obtains very promis-

[1] http://ameannotationinitiative.wikispaces.com

ing retrieval results, and showed that performance can be further increased by including more than one version of each song in the target database.

In Section 6.3 we demonstrated how the melody extraction algorithm can be used for genre classification by computing a set of melodic features (directly from the audio signal) based on pitch, duration, vibrato and contour typology. The melodic feature set was evaluated on three different datasets and was shown to outperform a baseline low-level timbral feature set based on MFCCs. Most importantly, we demonstrated that the classification accuracy could be improved by combining the two feature sets, suggesting that adding high-level melodic features to traditional low-level features is a promising approach for genre classification. An important aspect of the study is the fact that most of the melodic features proposed can be easily understood by humans, meaning that classification results can be interpreted in a musically meaningful way.

Next, in Section 6.4 we presented an application which exploits the salience function presented in Chapter 3: a novel approach for tonic identification in Indian classical music. We used the peaks of the salience function to construct a pitch histogram representing the most frequently played notes in the piece, in this way capturing the notes played by the drone instrument. Using a classification approach, we were able to automatically learn an optimal set of rules for tonic identification. The resulting decision tree was evaluated on a varied collection of excerpts and was shown to obtain high tonic identification accuracy. Importantly, the approach was shown to be suitable for a variety of performance types and the rules learnt by the system were easy to interpret and musically coherent.

Finally, in Section 6.5 we demonstrated how the melody extraction algorithm presented in this dissertation can be combined with a note segmentation and labelling method to produce a completely automatic melody transcription system. We evaluated the approach for transcribing accompanied flamenco singing and obtained promising results (a per-frame overall accuracy of 81% and a note-level F-measure of 0.57), and showed that using our algorithm provides higher transcription accuracy compared to a source separation based approach. We noted that good voicing detection is crucial for correct note transcription, and identified the main types of transcription errors: erroneous segmentation/consolidation of short notes and the mislabelling of notes due to the highly unstable and rapidly changing f_0 contour of flamenco singing.

MTG-QBH. For evaluating the QBH system presented in Chapter 6 we compiled a dataset of 118 sung melodies (queries). We have made the dataset freely available online[2], including detailed metadata files describing the queries and the songs in the target collections used for evaluation in our experiments.

MELODIA - Melody Extraction vamp plug-in. An implementation of the melody extraction algorithm proposed in this dissertation, in the form of a vamp[3] plug-in. The plug-in is available for all three major operating systems (Windows, Linux and OSX) and can be freely downloaded for non-commercial purposes[4]. The plug-in can be used for detailed analysis and visualisation of the predominant melody of an audio recording using Sonic Visualiser[5], including the visualisation of intermediate steps of the algorithm, or for batch processing of large music collections using Sonic Annotator[6]. Since its announcement in October of 2012, MELODIA has been downloaded over 4,000 times (4,400 as of June 15$^{\text{th}}$ 2013) by researchers, educators, artists and hobbyists from all over the world (cf. Appendix A).

Impact. As far as the author is aware, at the time of writing this dissertation the algorithm has been (or is being) used in at least 7 Master's theses, 5 ongoing doctoral theses (not including this one) and 2 European funded research projects (cf. Appendix A). The outcomes of the research carried out in this thesis have been published in a number of peer-reviewed journals and international conference proceedings. A full list of the author's publications is provided in Appendix B.

7.3 Future perspectives

"There's no problem, only solutions" (John Lennon)

There are many future directions in which the research presented in this dissertation could be explored. Alongside every method and solution we have presented, we have also discussed possible avenues for further work

[2] http://mtg.upf.edu/download/datasets/mtg-qbh
[3] http://vamp-plugins.org/
[4] http://mtg.upf.edu/technologies/melodia
[5] http://www.sonicvisualiser.org/
[6] http://www.omras2.org/sonicannotator

and improvement. In the following paragraphs we discuss some of the research avenues we consider particularly interesting. Also, as seen in Chapter 6, whilst there is still room for improving the performance of the melody extraction algorithm itself, it already opens the door to a large number of MIR applications and computational music analyses. In the final paragraphs of this section we discuss some of these applications, and also some of the less-expected uses people have found for the MELODIA plug-in.

Improving melody extraction accuracy. Perhaps the most obvious continuation of this thesis is to investigate how to further increase the extraction accuracy of our algorithm. Even though we obtain state-of-the-art results, it is clear that there is still much that could be done to improve the performance of our method. In the glass ceiling analysis performed in Chapter 4 we noted that the main difference between the performance of our algorithm and the glass ceiling results was the voicing false alarm rate. If we intend to use the algorithm for an application which is sensitive to voicing errors (e.g. transcription, cf. Section 6.5), then this is an issue worth addressing. Currently, our voicing detection method is based on the distribution of one feature – the contour mean salience. One possible way to improve the method would be to incorporate a wider set of features. Alternatively, if we plan to work with a music collection where the source of the melody is known (e.g. the melody is always sung or played by a specific instrument) we could take advantage of methods for detecting the timbre of that instrument in the polyphonic mixture. For example, for singing voice we could implement one of the approaches proposed by Hsu & Jang (2010a); Régnier & Peeters (2009, 2010); Rocamora (2011); Rocamora & Herrera (2007). For detecting the presence of a specific instrument we could for example consider the methods proposed by Burred et al. (2010); Fuhrmann (2012). In the glass ceiling study we also noted that the contour creation process itself could be improved, since not all of the melody contours were always tracked correctly. One way to address this would be to include a source separation preprocessing step as proposed by Hsu & Jang (2010a); Yeh et al. (2012) to reduce the interference of other sources in the polyphonic mixture. An alternative (yet related) strategy would be to characterise the timbre of individual contours: as commented in section 4.2, timbre attributes have been shown to provide important cues for auditory stream segregation (Iverson, 1995), suggesting they could similarly be of use for pitch contour tracking. Furthermore, characterising the timbre of individual pitch contours could also improve voicing detection and

melody selection. Promising approaches for timbre description of individual sources in polyphonic mixtures have been recently proposed by Marxer (2013); Rocamora & Pardo (2012).

Evaluation methodology. In Chapter 5 we presented what is to the best of our knowledge the first comprehensive study of evaluation methodology for melody extraction algorithms. Our most important conclusion was that the current collections used for evaluation in MIREX are either too small or not sufficiently heterogeneous. We showed that in order to obtain statistically stable and generalisable results we need to compile a new large collection of full-length songs containing music material that is representative of the universe of songs to which we would like to generalise the performance of our methods. To this end, we launched the Audio Melody Extraction Annotation Initiative (cf. footnote 1). At the time of writing this dissertation the initiative is still in its early days, but we hope it will be successful in accomplishing this goal. Whilst carefully compiling and annotating a dataset for evaluation can be time consuming and even monotonous, we believe it is an important endeavour that is necessary to ensure the future advancement of the field.

Another aspect of evaluation that we would be very interested in exploring in the future is that of perceived accuracy and qualitative evaluation. That is, we know that current extraction algorithms can extract the melody with an overall accuracy between of 65–75%. But does this mean we can expect between 65–75% user satisfaction with the extracted f_0 sequences? How do different types of extraction errors affect our perception of the quality of the extracted sequences? A simple experiment that could provide some preliminary answers to these questions would be to perform listening tests using synthesised versions of the extracted f_0 sequences and have subjects rate them in some way. We could then study the relationship between the quantitative evaluation measures currently used in MIREX and the qualitative judgements of human listeners.

Melody-extraction-based applications. In Chapter 6 we presented novel methods that exploit the output of our melody extraction algorithm for music retrieval, classification, transcription and computational music analysis. As can be expected, there are numerous research avenues that can be explored for each application. As these have been discussed at length in the chapter itself, we will only provide a very brief summary here. For music retrieval we noted the need to combine our proposed approach with

an efficient indexing method so that we can apply it to large-scale music collections. In the specific case of query-by-humming we observed the negative effect that mistuned queries can have on retrieval accuracy and noted it would be interesting to explore query correction and reformulation techniques. In the case of music classification, we think it would be interesting to explore the use of the proposed melodic features for other classification tasks, for example singer characterisation or even mood classification. For our tonic identification approach for Indian classical music we explained how incorporating information about the pitch sequence of the lead voice (which can be automatically extracted using our algorithm) can help reduce octave errors. For the proposed melodic transcription approach we noted the importance of correct voicing detection and expressed our interest in evaluating it on other musical genres (ideas for improving voicing detection were proposed earlier in this section). Finally, in the case of all the aforementioned applications we could gain much insight by measuring the extraction accuracy of our algorithm for the different collections used for evaluating these applications. As noted in the case of melodic transcription, this would allow us to assess what proportion of the errors is due to the melody extraction algorithm itself as opposed to other parts of the system. It would also allow us to measure the correlation between melody extraction accuracy and overall system performance (e.g. transcription accuracy, retrieval precision, etc.).

Apart from improving the applications discussed in this thesis, there is a wide range of MIR applications that could be built on top of the melody extraction algorithm presented here. At the end of Chapter 6 we mentioned some of the studies in computational music analysis which make use of our algorithm that have already been published (Chen, 2013; Şentürk et al., 2013; Gulati, 2012; Koduri et al., 2012a,b; Kroher, 2013; Pikrakis et al., 2012). Further examples of possible uses for the algorithm are provided below.

Unexpected avenues. When people download MELODIA (the plug-in implementation of the melody extraction method proposed in this thesis), we ask them what is their intended use for the algorithm. As could be expected, a considerable amount of responses are research-related. Of the thousands of responses we have received, here is a small selection of research oriented answers:

- “Musicological analysis of the voice; French-Canadian popular music

(1920's-1940's). Postdoctoral research."

- "Research, jazz solo transcription."
- "Research on ethnomusicology, traditional music from west Africa."
- "Test the usage of the output data as input for source separation."
- "Research in musicology (mainly second half of 20th century and 21st century)."
- "Research on speech melody and prosodic disorders."
- "Learning Indian raga, pitch tracking."
- "Use for thesis: feature extraction for music genre classification."
- "Research on query-by-humming."
- "Animal sound classification."
- "Research on music similarity."
- "Analysis of recordings for PhD in piano performance."
- "Folk music analysis."
- "Master thesis. Research on a remote music culture in south China/north Burma."
- "I'm doing a PhD in music perception research. In order to find neural correlates of various musical features I need a multi-dimensional description of the music signal used as a stimulus."

But research is not the only reason people are downloading MELODIA. We were happy to discover that many people are downloading it for educational purposes too. Here is a small selection of education-related use cases people have reported:

- "Use for teaching music in school."
- "Demonstration and education purposes."
- "To aid in self-directed study of music theory and composition."

- "Helping my wife learning how to sing some songs for which we can't find scores."
- "Teaching computer-assisted music, acoustics and electroacoustic music in general."
- "Experimenting with different ways to use visual representations of sound to help teach music / music theory."
- "Teaching musical audio analysis at the ESMUC."
- "Teaching transcription and analysis."
- "This will be used for educational purposes, more specifically for teaching sound design."
- "Visual/ear practice with my synthesizer. Self educational reasons."
- "Educational purposes – discussing various methods of analysing musical texts."

People have also downloaded MELODIA for creative purposes, for example:

- "Music composition."
- "Audio reactive visualizers for electronic dance music"
- "Analyze pitch to compare and improve my Vocaloid[7] results."
- "Non-commercial use for creative resources as a contemporary music composer and teacher."
- "I intend to use the plug-in in conjunction with Vocaloid to achieve more realistic vocals."
- "To design soundscapes for live theatre."
- "Performance art."

Some people noted that they wanted to try it out "for fun and experimentation". Finally, some people provided slightly less serious answers – here is a small selection of humorous responses we have received that all share a common theme:

[7] Vocaloid is a popular singing voice synthesiser by YAMAHA: http://www.vocaloid.com/en/

- "Research and conquer the world with music."
- "I will destroy the world with this! HAHAHAHAHAAAA!!!"
- "The same thing we do every night Pinky. Take over the world."
- "Take over the universe!"

Melody extraction remains an intriguing topic of research, heavily intertwined with a varied set of disciplines including signal processing, music cognition, musicology and computer science. We believe this thesis is a step forward toward the extraction and exploitation of musically meaningful information from complex polyphonic music signals. Melody is without doubt a very important and distinct aspect of music information, and systems for automatically extracting it from music audio are sure to be central to future music information technologies.

Justin J. Salamon, Barcelona, July 11, 2013.

Bibliography

The numbers at the end of each bibliographic entry indicate the pages in which it is cited.

Adams, C. R. (1976). Melodic contour typology. *Ethnomusicology*, *20*(2), 179–215. 159

Arora, V. & Behera, L. (2013). On-line melody extraction from polyphonic audio using harmonic cluster tracking. *IEEE Trans. on Audio, Speech, and Language Processing*, *21*(3), 520–530. 31, 33, 36

Baines, A. & Temperley, N. (2013). Pitch. The Oxford Companion to Music, Oxford Music Online (last checked Feb. 2013). Online: `http://www.oxfordmusiconline.com/subscriber/article/opr/t114/e5199`. 8

Bello, J. P. (2003). *Towards the automated analysis of simple polyphonic music: A knowledge-based approach*. Ph.D. thesis, University of London, London, UK. 14, 15, 16, 186

Benaroya, L., Bimbot, F., & Gribonval, R. (2006). Audio source separation with a single sensor. *IEEE Trans. on Audio, Speech, and Language Processing*, *14*(1), 191–199. 38

Berenzweig, A., Logan, B., Ellis, D. P. W., & Whitman, B. (2004). A large scale evaluation of acoustic and subjective music similarity measures. *Computer Music Journal*, *28*(2), 63–76. 130

Bodoff, D. (2008). Test theory for evaluating reliability of IR test collections. *Inf. Process. Manage.*, *44*(3), 1117–1145. 119, 120, 121

Bogdanov, D., Haro, M., Fuhrmann, F., Xambó, A., Gómez, E., & Herrera, P. (2013). Semantic audio content-based music recommendation and visualization based on user preference examples. *Information Processing & Management*, *49*(1), 13–33. 14

Bonada, J. (2008). Wide-band harmonic sinusoidal modeling. In *11th Int. Conf. on Digital Audio Effects (DAFx-08)*, pp. 265–272. Espoo, Finland. 66

Box, G. E. P. & Cox, D. R. (1964). An analysis of transformations. *J. of the Royal Statistical Society. Series B (Methodological)*, *26*(2), 211–252. 97

Bozkaya, T. & Ozsoyoglu, M. (1999). Indexing large metric spaces for similarity search queries. *ACM Trans. on Database Systems*, *24*(3), 361–404. 152

Bregman, A. S. (1990). *Auditory scene analysis*. Cambridge, Massachussetts: MIT Press. 15, 36, 62, 79, 82

Bregman, A. S. (1995). Constraints on computational models of auditory scene analysis, as derived from human perception. *J. of the Acoustical Soc. of Japan (E)*, *16*(3), 133–136. 18

Bregman, A. S., Ahad, P. A., & Kim, J. (1994). Resetting the pitch-analysis system. 2. role of sudden onsets and offsets in the perception of individual components in a cluster of overlapping tones. *J. Acoust. Soc. Am.*, *96*(5), 2694–2703. 186

Brennan, R. L. (2001). *Generalizability Theory*. New York: Springer-Verlag. 119, 120, 121

Brown, J. C. (1991). Calculation of a constant-Q spectral transform. *J. Acoust. Soc. Am.*, *81*(1), 425–434. 33

Bryan, N. J. & Wang, G. (2011). Musical influence network analysis and rank of sampled-based music. In *12th Int. Soc. for Music Info. Retrieval Conf.*, pp. 329–334. Miami, Florida. 131

Burred, J. J., Röbel, A., & Sikora, T. (2010). Dynamic spectral envelope modeling for the analysis of musical instrument sounds. *IEEE Trans. on Audio, Speech, and Language Processing*, *18*(3), 663–674. 204

Cabrera, J. J., Díaz-Báñez, J. M., Escobar, F., Gómez, E., & Mora, J. (2008). Comparative melodic analysis of a cappella flamenco cantes. In *4th Conf. on Interdisciplinary Musicology*. Thessaloniki, Greece. 193

Cancela, P. (2008). Tracking melody in polyphonic audio. In *4th Music Inform. Retrieval Evaluation eXchange (MIREX)*. 31, 33, 34, 36, 62, 79, 81, 88, 91

Cancela, P., Rocamora, M., & López, E. (2009). An efficient multi-resolution spectral transform for music analysis. In *10th Int. Soc. for Music Info. Retrieval Conf.*, pp. 309–314. Kobe, Japan. 56

Candès, E. J., Li, X., Ma, Y., & Wright, J. (2009). Robust principal component analysis? *J. of the ACM*, *58*(1), 1–37. 39

Cano, P. (2007). *Content-Based Audio Search from Fingerprinting to Semantic Audio Retrieval*. Ph.D. thesis, UPF, Barcelona. 14

Cano, P., Gómez, E., Gouyon, F., Herrera, P., Koppenberger, M., Ong, B., Serra, X., Streich, S., & Wack, N. (2006). ISMIR 2004 audio description contest. Tech. rep., Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain. 44

Casey, M., Veltkamp, R., Goto, M., Leman, M., Rhodes, C., & Slaney, M. (2008a). Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, *96*(4), 668–696. 13

Casey, M. A., Rhodes, C., & Slaney, M. (2008b). Analysis of minimum distances in high-dimensional musical spaces. *IEEE Trans. on Audio, Speech, and Language Processing*, *16*(5), 1015–1028. 151

Chen, K. (2013). *Characterization of Pitch Intonation in Beijing Opera*. Master's thesis, Universitat Pompeu Fabra, Barcelona, Spain. 195, 206

Chien, Y.-R., Wang, H.-M., & Jeng, S.-K. (2011). An acoustic-phonetic approach to vocal melody extraction. In *12th Int. Soc. for Music Info. Retrieval Conf.*, pp. 25–30. Miami, USA. 102

Chordia, P., Jagadeeswaran, J., & Rae, A. (2009). Automatic carnatic raag classification. *J. of the Sangeet Research Academy (Ninaad)*. 166

Copeland, A. (1957). Musical texture. In *What To Listen For In Music*, chap. 5. New York: McGraw Hill. 10

Şentürk, S., Gulati, S., & Serra, X. (2013). Score informed tonic identification for makam music of Turkey. In *14th Int. Soc. for Music Info. Retrieval Conf.* Curitiba, Brazil. (to appear). 195, 206

Dahlhaus, C. (2013). Harmony. Grove Music Online. Oxford Music Online (last checked May 2013). Online http://www.oxfordmusiconline.com/subscriber/article/grove/music/50818. 131

Danielou, A. (2010). *The Ragas of Northern Indian Music*. New Delhi: Munshiram Manoharlal Publishers. 166, 168

Dannenberg, R. B., Birmingham, W. P., Pardo, B., Hu, N., Meek, C., & Tzanetakis, G. (2007). A comparative evaluation of search techniques for query-by-humming using the MUSART testbed. *J. of the American Soc. for Inform. Science and Technology*, *58*(5), 687–701. 131, 132, 139, 153

de Cheveigné, A. (2005). Pitch perception models. In R. R. Fay, A. N. Popper, C. Plack, R. Fay, A. Oxenham, & A. Popper (Eds.) *Pitch*, *Springer Handbook of Auditory Research*, vol. 24, pp. 169–233. Springer New York. 9

de Cheveigné, A. & Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and music. *J. Acoust. Soc. Am.*, *111*(4), 1917–1930. 26, 27, 181

De Mulder, T., Martens, J.-P., Lesaffre, M., Leman, M., De Baets, B., & De Meyer, H. (2003). An auditory model based transcriber of vocal queries. In *4th Int. Conf. on Music Info. Retrieval*, pp. 26–30. Baltimore, MA, USA. 180

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. of the Roy. Statist. Soc. Series B (Methodological)*, *39*(1), 1–38. 34

Deva, B. C. (1980). *The Music of India: A Scientific Study*. New Delhi: Munshiram Manoharlal Publishers. 166

Donnier, P. (1997). Flamenco: elementos para la transcripción del cante y la guitarra. In *3rd Congress of the Iberian Ethnomusicology Society*, pp. 103–120. Benicàssim, Spain. 180

Downie, J. S. (2008). The music information retrieval evaluation exchange (2005–2007): A window into music information retrieval research. *Acoustical Science and Technology*, *29*(4), 247–255. 14, 30, 99, 113

Dressler, K. (2006). Sinusoidal extraction using an efficient implementation of a multi-resolution FFT. In *Proc. 9th Int. Conf. on Digital Audio Effects (DAFx-06)*, pp. 247–252. Montreal, Canada. 33, 57, 58, 59

Dressler, K. (2009). Audio melody extraction for mirex 2009. In *5th Music Inform. Retrieval Evaluation eXchange (MIREX)*. 79, 100, 115

Dressler, K. (2011a). An auditory streaming approach for melody extraction from polyphonic music. In *12th International Society for Music Information Retrieval Conference*, pp. 19–24. Miami, USA. 31, 34, 36, 91, 96, 112

Dressler, K. (2011b). Pitch estimation by the pair-wise evaluation of spectral peaks. In *AES 42nd Int. Conf.*, pp. 278–290. Ilmenau, Germany. 34

Duda, A., Nürnberger, A., & Stober, S. (2007). Towards query by singing/humming on audio databases. In *8th Int. Conf. on Music Info. Retrieval*, pp. 331–334. Vienna, Austria. 133, 155

Durrieu, J.-L. (2010). *Automatic Transcription and Separation of the Main Melody in Polyphonic Music Signals*. Ph.D. thesis, Télécom ParisTech. 5, 6, 7, 17

Durrieu, J.-L., David, B., & Richard, G. (2011). A musically motivated mid-level representation for pitch estimation and musical audio source separation. *IEEE Journal on Selected Topics on Signal Processing*, *5*(6), 1180–1191. 37

Durrieu, J.-L., Richard, G., & David, B. (2009). An iterative approach to monaural musical mixture de-soloing. In *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, pp. 105–108. 3

Durrieu, J.-L., Richard, G., David, B., & Févotte, C. (2010). Source/filter model for unsupervised main melody extraction from polyphonic audio signals. *IEEE Trans. on Audio, Speech, and Language Processing*, *18*(3), 564–575. 30, 31, 37, 38, 39, 115

Ellis, D. P. W. (1996). *Prediction-driven computational auditory scene analysis*. Ph.D. thesis, Massachusetts Institute of Technology, Media Laboratory. 15, 18

Fernandez, L. (2004). *Flamenco Music Theory*. Acordes Concert. 180

Fernandez, L. (2011). La bimodalidad en las formas del fandango y en los cantes de levante: origen y evolución. *Revista de Investigación sobre Flamenco*, *5*, 37–53. 180

Flanagan, J. L. & Golden, R. M. (1966). Phase vocoder. *Bell Systems Technical Journal*, *45*, 1493–1509. 60

Foucard, R., Durrieu, J.-L., Lagrange, M., & Richard, G. (2010). Multimodal similarity between musical streams for cover version detection. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 5514–5517. Dallas, Texas, USA. 3, 131, 140, 149

Fuentes, B., Liutkus, A., Badeau, R., & Richard, G. (2012). Probabilistic model for main melody extraction using constant-Q transform. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 5357–5360. Kyoto, Japan. 115

Fuhrmann, F. (2012). *Automatic musical instrument recognition from polyphonic music audio signals*. Ph.D. thesis, Universitat Pompeu Fabra, Barcelona, Spain. 204

Gold, B. & Rabiner, L. R. (1969). Parallel processing techniques for estimating pitch periods of speech in the time domain. *J. Acoust. Soc. Am.*, *46*(2B), 442–448. 26

Gómez, E. (2006a). *Tonal Description of Music Audio Signals*. Ph.D. thesis, Universitat Pompeu Fabra, Barcelona. 14, 137, 138

Gómez, E. (2006b). Tonal description of polyphonic audio for music content processing. *INFORMS J. on Computing, Special Cluster on Computation in Music*, *18*(3), 294–304. 62, 63, 132, 137

Gómez, E. & Bonada, J. (2013). Towards computer-assisted flamenco transcription: An experimental comparison of automatic transcription algorithms as applied to a cappella singing. *Computer Music J.*, *37*(2), 73–90. 180, 182, 183, 188, 192

Gómez, E., Cañadas, F., Salamon, J., Bonada, J., Vera, P., & Cabañas, P. (2012). Predominant fundamental frequency estimation vs singing voice separation for the automatic transcription of accompanied flamenco singing. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 601–606. Porto, Portugal. 3, 180, 181, 182, 183

Gómez, E. & Herrera, P. (2004). Estimating the tonality of of polyphonic audio files: Cognitive versus machine learning modelling strategies. In *5th Int. Conf. on Music Info. Retrieval*, pp. 92–95. Barcelona, Spain. 172

Gómez, E., Klapuri, A., & Meudic, B. (2003). Melody description and extraction in the context of music content processing. *J. of New Music Research*, *32*(1), 23–40. 5, 16, 26

Gómez, E., Streich, S., Ong, B., Paiva, R., Tappert, S., Batke, J., Poliner, G., Ellis, D., & Bello, J. (2006). A quantitative comparison of different approaches for melody extraction from polyphonic audio recordings. Tech. Rep. MTG-TR-2006-01, Music Technology Group, Universitat Pompeu Fabra. 30

Gómez, F., Pikrakis, A., Mora, J., Díaz-Bañez, J. M., Gómez, E., & Escobar, F. (2011). Automatic detection of ornamentation in flamenco. In *4th Int. Workshop on Machine Learning and Music, 25th Annual Conf. on Neural Info. Processing Systems*. Granada, Spain. 193

Goto, M. (2000). A robust predominant-f0 estimation method for real-time detection of melody and bass lines in cd recordings. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, pp. II757–II760 vol.2. 6, 30

Goto, M. (2004). A real-time music-scene-description system: predominant-f0 estimation for detecting melody and bass lines in real-world audio signals. *Speech Communication*, *43*, 311–329. 7, 18, 30, 31, 32, 33, 34, 36, 38, 47, 56, 81, 91, 96, 112, 134

Goto, M. (2006). Analysis of musical audio signals. In D. Wang & G. J. Brown (Eds.) *Computational Auditory Scene Analysis: Principles, Algorithms and Applications*, chap. 8. New York: Wiley-IEEE. 15

Goto, M. & Hayamizu, S. (1999). A real-time music scene description system: Detecting melody and bass lines in audio signals. In *Working Notes of the IJCAI-99 Workshop on Computational Auditory Scene Analysis*, pp. 31–40. 5, 30

Gouyon, F. (2008). *Computational Rhythm Description*. VDM Verlag. 14

Gouyon, F., Herrera, P., Gómez, E., Cano, P., Bonada, J., Loscos, A., Amatriain, X., & Serra, X. (2008). Content processing of music audio signals. In P. Polotti & D. Rocchesso (Eds.) *Sound to Sense, Sense to Sound: A State of the Art in Sound and Music Computing*, chap. 3, pp. 83–160. Berlin: Logos Verlag Berlin GmbH. 13, 14

Grosche, P. & Müller, M. (2012). Towards characteristic audio shingles for efficient cross-version music retrieval. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 473–476. Kyoto, Japan. 151

Guaus, E. (2009). *Audio content processing for automatic music genre classification: descriptors, databases, and classifiers.* Ph.D. thesis, Universitat Pompeu Fabra, Barcelona, Spain. 157, 164

Gulati, S. (2012). *A Tonic Identification Approach for Indian Art Music.* Master's thesis, Universitat Pompeu Fabra, Barcelona, Spain. 178, 195, 206

Hall, M. (1999). *Correlation-based Feature Selection for Machine Learning.* Ph.D. thesis, University of Waikato, Hamilton, New Zealand. 161, 173

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA data mining software: an update. *SIGKDD Explorations Newsletter*, *11*(1), 10–18. 140, 161, 173

Hanna, P. & Robine, M. (2009). Query by tapping system based on alignment algorithm. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 1881–1884. Taipei, Taiwan. 155

Hartmann, W. M. (1996). Pitch, periodicity, and auditory organization. *J. Acoust. Soc. Am.*, *100*(6), 3491–3502. 9

Harwood, D. L. (1976). Universals in music: a perspective from cognitive psychology. *Ethomusicology*, *20*(3), 521–533. 130

Hermes, D. J. (1988). Measurement of pitch by subharmonic summation. *J. Acoust. Soc. Am.*, *83*(1), 257–264. 63

Herrera, P., Amatriain, X., Batlle, E., & Serra, X. (2000). Towards instrument segmentation for music content description: a critical review of instrument classification techniques. In *Int. Symposium for Music Inform. Retrieval*, pp. 23–25. Plymouth, Mass, USA. 18

Herrera, P. & Bonada, J. (1998). Vibrato extraction and parameterization in the spectral modeling synthesis framework. In *Proc. Workshop on Digital Audio Effects (DAFx-98)*, pp. 107–110. 84, 159

Hess, W. (1983). *Pitch determination of speech signals: algorithms and devices*. Berlin: Springer-Verlag. 16, 26

Hoces, R. (2011). *La Transcripción Musical Para Guitarra Flamenca: Análisis e Implementación Metodológica*. Ph.D. thesis, University of Seville, Seville, Spain. 180

Hsu, C. & Jang, J. R. (2010a). Singing pitch extraction by voice vibrato/tremolo estimation and instrument partial deletion. In *11th Int. Soc. for Music Info. Retrieval Conf.*, pp. 525–530. Utrecht, The Netherlands. 31, 32, 33, 34, 36, 39, 47, 100, 204

Hsu, C.-L. & Jang, J.-S. (2010b). On the improvement of singing voice separation for monaural recordings using the MIR-1K dataset. *IEEE Trans. on Audio, Speech, and Language Processing*, *18*(2), 310–319. 37

Hsu, C.-L., Wang, D., & Jang, J.-S. (2011). A trend estimation algorithm for singing pitch detection in musical recordings. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 393–396. 115

Huang, P.-S., Chen, S. D., Smaragdis, P., & Hasegawa-Johnson, M. (2012). Singing voice separation from monaural recordings using robust principal component analysis. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 57–60. Kyoto, Japan. 37, 39

Huron, D. (2001). Tone and voice: A derivation of the rules of voice-leading from perceptual principles. *Music Perception*, *19*(1), 1–64. 6, 15, 79, 84, 89, 112

Hutardo, A. & Hutardo, D. (2002). *La Voz de la Tierra, Estudio y Transcripción de los Cantes Campesinos en las Provincias de Jaén y Córdoba*. Jerez: Centro Andaluz de Flamenco. 180

Hutardo, D. & Hutardo, A. (1998). *El Arte de la Escritura Musical Flamenca*. Seville: Bienal de Arte Flamenco. 180

Hyer, B. (2013). Tonality. Grove Music Online. Oxford Music Online (last checked May 2013). Online: `http://www.oxfordmusiconline.com/subscriber/article/grove/music/28102`. 131

Iverson, P. (1995). Auditory stream segregation by musical timbre: Effects of static and dynamic acoustic attributes. *J. of Experimental Psychology: Human Perception and Performance*, *21*(4), 751–763. 108, 204

Janer, J., Bonada, J., de Boer, M., & Loscos, A. (2008). Audio recording analysis and rating. Patent pending US20080026977, Universitat Pompeu Fabra, 06/02/2008. 180, 182

Jo, S., Joo, S., & Yoo, C. D. (2010). Melody pitch estimation based on range estimation and candidate extraction using harmonic structure model. In *Interspeech*, pp. 2902–2905. Makuhari, Japan. 31, 34

Joo, S., Jo, S., & Yoo, C. D. (2010). Melody extraction from polyphonic audio signal MIREX 2010. In *6th Music Info. Retrieval Evaluation eXchange (MIREX)*, extended abstract. Utrecht, The Netherlands. 100

Kanoulas, E. & Aslam, J. A. (2009). Empirical justification of the gain and discount function for nDCG. In *18th ACM Conf. on Info. and Knowledge Manage. (CIKM)*, pp. 611–620. Hong Kong, China. 120, 121

Kantz, H. & Schreiber, T. (2004). *Nonlinear time series analysis*. Cambridge, UK: Cambridge University Press, 2nd edn. 139

Kassler, M. (1966). Toward musical information retrieval. *Perspectives of New Music*, *4*(2), 59–67. 13

Keiler, F. & Marchand, S. (2002). Survey on extraction of sinusoids in stationary sounds. In *5th Int. Conf. on Digital Audio Effects (DAFx-02)*, pp. 51–58. Hamburg, Germany. 59, 60, 65, 67

Klapuri, A. (2000). Qualitative and quantitative aspects in the design of periodicity estimation algorithms. In *European Signal Processing Conference*. Tampere, Finland. 26, 27

Klapuri, A. (2003). Multiple fundamental frequency estimation based on harmonicity and spectral smoothness. *IEEE Trans. on Speech and Audio Processing*, *11*(6), 804–816. 16, 34

Klapuri, A. (2004). *Signal processing methods for the automatic transcription of music*. Ph.D. thesis, Tampere University of Technology, Finland. 16, 30, 34

Klapuri, A. (2006). Multiple fundamental frequency estimation by summing harmonic amplitudes. In *Proc. 7th Int. Conf. on Music Inform. Retrieval*, pp. 216–221. Victoria, Canada. 62, 63

Klapuri, A. (2009). A method for visualizing the pitch content of polyphonic music signals. In *10th Int. Soc. for Music Info. Retrieval Conf.*, pp. 615–620. Kobe, Japan. 88

Klapuri, A. & Davy, M. (Eds.) (2006). *Signal Processing Methods for Music Transcription*. New York: Springer. 9, 14, 16, 30

Kob, M., Henrich, N., Herzel, H., Howard, D., Tokuda, I., & Wolfe, J. (2011). Analysing and understanding the singing voice: recent progress and open questions. *Current Bioinformatics*, *6*(3), 362–374. 62

Koduri, G. K., Gulati, S., Rao, P., & Serra, X. (2012a). Rāga recognition based on pitch distribution methods. *J. of New Music Research*, *41*(4), 337–350. 195, 206

Koduri, G. K., Serrà, J., & Serra, X. (2012b). Characterization of intonation in carnatic music by parametrizing pitch histograms. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 199–204. Porto, Portugal. 3, 166, 195, 206

Kotsifakos, A., Papapetrou, P., Hollmén, J., Gunopulos, D., & Athitsos., V. (2012). A survey of query-by-humming similarity methods. In *Int. Conf. on Pervasive Technologies Related to Assistive Environments (PETRA)*, pp. 5:1–5:4. Heraklion, Crete, Greece. 132

Kreiman, J. & Sidtis, D. V. L. (2011). Miscellany: Voice in law enforcement, media and singing. In *Foundations of voice studies: an interdisciplinary approach to voice production and perception*, chap. 10, pp. 361–397. Hoboken, NJ: Wiley-Blackwell. 52

Kroher, N. (2013). *The Flamenco Cante: Automatic Characterization of Flamenco Singing by Analyzing Audio Recordings*. Master's thesis, Universitat Pompeu Fabra, Barcelona, Spain. 86, 195, 206

Krumhansl, C. L. (2001). *Cognitive Foundations of Musical Pitch*. New York: Oxford University Press. 172

Kurth, F. & Müller, M. (2008). Efficient index-based audio matching. *IEEE Trans. on Audio, Speech, and Language Processing*, *16*(2), 382–395. 152

Lagrange, M., Martins, L. G., Murdoch, J., & Tzanetakis, G. (2008). Normalized cuts for predominant melodic source separation processing. *IEEE Trans. on Audio, Speech, and Language Processing*, *16*(2), 278–290. 79

Lahat, M., Niederjohn, R., & Krubsack, D. (1987). A spectral autocorrelation method for measurement of the fundamental frequency of noise-corrupted speech. *IEEE Trans. on Acoustics, Speech and Signal Processing*, *35*(6), 741–750. 27

Laurier, C. (2011). *Automatic Classification of Musical Mood by Content-Based Analysis*. Ph.D. thesis, Universitat Pompeu Fabra. 14

Lesaffre, M., Leman, M., De Baets, B., & Martens, J. (2004). Methodological considerations concerning manual annotation of musical audio in function of algorithm development. In *5th Int. Conf. on Music Info. Retrieval*, pp. 64–71. Barcelona, Spain. 180

Li, Y. & Wang, D. (2007). Separation of singing voice from music accompaniment for monaural recordings. *IEEE Trans. on Audio, Speech, and Language Processing*, *15*(4), 1475–1487. 37

Liao, W.-H., Su, A. W. Y., Yeh, C., & Röbel, A. (2011). Melody estimation for MIREX 2011. In *7th Music Info. Retrieval Evaluation eXchange (MIREX)*, extended abstract. Miami, USA. 102

Liem, C. C. S. & Hanjalic, A. (2009). Cover song retrieval: a comparative study of system component choices. In *Int. Soc. for Music Inform. Retrieval Conf.*, pp. 573–578. Kobe, Japan. 131, 136

Liu, D. & Hua, K. A. (2009). Transfer non-metric measures into metric for similarity search. In *17th ACM Int. Conf. on Multimedia*, pp. 693–696. Beijing, China. 152

Liutkus, A., Rafii, Z., Badeau, R., Pardo, B., & Richard, G. (2012). Adaptive filtering for music/voice separation exploiting the repeating musical structure. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 53–56. Kyoto, Japan. 18, 37, 40

Lynch, M. P., Eilers, R. E., Oller, D. K., & Urbano, R. C. (1990). Innateness, experience and music perception. *Psychological Science*, *1*(4), 272–276. 130

Maher, R. C. & Beauchamp, J. W. (1994). Fundamental frequency estimation of musical signals using a Two-Way Mismatch procedure. *J. Acoust. Soc. Am.*, *95*(4), 2254–2263. 27, 34

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge, UK: Cambridge University Press. 142

Marolt, M. (2004). On finding melodic lines in audio recordings. In *7th Int. Conf. on Digial Audio Effects (DAFx'04)*, pp. 217–221. Naples, Italy. 30, 31, 33, 34, 36

Marolt, M. (2008). A mid-level representation for melody-based retrieval in audio collections. *IEEE Trans. on Multimedia*, *10*(8), 1617–1625. 131, 144

Marxer, R. (2013). *Audio Source Separation for Music in Low-latency and High-latency Scenarios*. Ph.D. thesis, Universitat Pompeu Fabra, Barcelona, Spain. 205

Mason, R. & Harrington, S. (2007). Perception and detection of auditory offsets with single simple musical stimuli in a reverberant environment. In *30th Audio Engineering Soc. Int. Conf.*, pp. 331–342. Saariselkä, Finland. 186

McKay, C. & Fujinaga, I. (2004). Automatic genre classification using large high-level musical feature sets. In *5th Int. Conf. on Music Info. Retrieval*, pp. 525–530. Barcelona, Spain. 157

Medan, Y., Yair, E., & Chazan, D. (1991). Super resolution pitch determination of speech signals. *IEEE Trans. on Signal Processing*, *39*(1), 40–48. 27

Mesaros, A., Virtanen, T., & Klapuri, A. (2007). Singer identification in polyphonic music using vocal separation and pattern recognition methods. In *Proc. 8th Int. Conf. on Music Inform. Retrieval*, pp. 375–378. 3

Moore, B. C. J. (2003). *An Introduction to the Psychology of Hearing*. San Diego: Academic Press, fifth edn. 32, 33

Mora, J., Gómez, F., Gómez, E., Escobar, F., & Díaz-Báñez, J. M. (2010). Characterization and melodic similarity of a cappella flamenco cantes. In *11th Int. Soc. for Music Info. Retrieval Conf.*, pp. 351–356. Utrecht, The Netherlands. 181

Müller, M. & Ewert, S. (2010). Towards timbre-invariant audio features for harmony-based music. *IEEE Trans. on Audio, Speech, and Language Processing*, *18*(3), 649–662. 136

Müller, M., Grosche, P., & Wiering, F. (2009). Robust segmentation and annotation of folk song recordings. In *10th Int. Soc. for Music Info. Retrieval Conf.*, pp. 735–740. Kobe, Japan. 136

Müller, M., Kurth, F., & Clausen, M. (2005). Audio matching via chroma-based statistical features. In *6th Int. Conf. on Music Info. Retrieval*, pp. 288–295. London, UK. 136

Noll, A. M. (1967). Cepstrum pitch determination. *J. Acoust. Soc. Am.*, *41*(2), 293–309. 27

Ong, B. S., Gómez, E., & Streich, S. (2006). Automatic extraction of musical structure using pitch class distribution features. In *Workshop on Learning the Semantics of Audio Signals (LSAS)*, pp. 53–65. Athens, Greece. 138

Ono, N., Miyamoto, K., Kameoka, H., Le Roux, J., Uchiyama, Y., Tsunoo, E., Nishimoto, T., & Sagayama, S. (2010). Harmonic and percussive sound separation and its application to mir-related tasks. In Z. Ras & A. Wieczorkowska (Eds.) *Advances in Music Information Retrieval*, *Studies in Computational Intelligence*, vol. 274, pp. 213–236. Springer Berlin / Heidelberg. 38, 108, 181

Orio, N. (2006). Music retrieval: A tutorial and review. *Foundations and Trends in Information Retrieval*, *1*(1), 1–90. 13

Ozerov, A., Philippe, P., Bimbot, F., & Gribonval, R. (2007). Adaptation of bayesian models for single-channel source separation and its application to voice/music separation in popular songs. *IEEE Trans. on Audio, Speech, and Language Process.*, *15*(5), 1564–1578. 37

Pachet, F. (2005). Knowledge management and musical metadata. In D. Schwartz (Ed.) *Encyclopedia of Knowledge Management*. Harpenden, UK: Idea Group. 130

Paiva, R. P. (2007). *Melody Detection in Polyphonic Audio*. Ph.D. thesis, Department of Informatics Engineering, University of Coimbra, Portugal. 115

Paiva, R. P., Mendes, T., & Cardoso, A. (2004). A methodology for detection of melody in polyphonic music signals. In *116th AES Convention*, 6029. 30

Paiva, R. P., Mendes, T., & Cardoso, A. (2006). Melody detection in polyphonic musical signals: Exploiting perceptual rules, note salience, and melodic smoothness. *Comput. Music J.*, *30*(4), 80–98. 5, 6, 8, 30, 31, 33, 34, 35, 36, 79, 80, 81, 83

Pampalk, E., Flexer, A., & Widmer, G. (2005). Improvements of audio-based music similarity and genre classification. In *6th Int. Conf. on Music Info. Retrieval*, pp. 628–633. London, UK. 162

Panagakis, Y., Kotropoulos, C., & Arce, G. R. (2009). Music genre classification using locality preserving non-negative tensor factorization and sparse representations. In *10th Int. Soc. for Music Info. Retrieval Conf.*, pp. 249–254. Kobe, Japan. 157

Pardo, B. & Birmingham, W. (2003). Query by humming: How good can it get? In *Workshop on the Evaluation of Music Info. Retrieval Systems, 26th Annual Int. ACM SIGIR Conf.*, pp. 107–109. Toronto, Canada. 154

Pardo, B., Little, D., Jiang, R., Livni, H., & Han, J. (2008). The VocalSearch music search engine. In *ACM/IEEE-CS Joint Conf. on Digital Libraries (JCDL)*, pp. 430–430. Pittsburgh PA, USA. 133, 155

Park, S., Jo, S., & Yoo, C. D. (2011). Melody extraction from polyphonic audio signal MIREX 2011. In *7th Music Info. Retrieval Evaluation eXchange (MIREX)*, extended abstract. Miami, USA. 102

Peeters, G. (2003). Automatic classification of large musical instrument databases using hierarchical classifiers with inertia ratio maximization. In *115th Audio Engineering Society Convention*, 5959. New York, NY, USA. 96

Peeters, G. (2004). A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Tech. rep., Institut de Recherche et Coordination Acoustique/Musique (IRCAM). 14, 58

Pikrakis, A., Gómez, F., Oramas, S., Díaz-Báñez, J. M., Mora, J., Escobar, F., Gómez, E., & Salamon, J. (2012). Tracking melodic patterns in flamenco singing by analyzing polyphonic music recordings. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 421–426. Porto, Portugal. 3, 195, 206

Piszczalski, M. & Galler, B. A. (1979). Predicting musical pitch from component frequency ratios. *J. Acoust. Soc. Am.*, *66*(3), 710–720. 27

Plumbley, M. D., Abdallah, S. A., Bello, J. P., Davies, M. E., Monti, G., & Sandler, M. B. (2002). Automatic music transcription and audio source separation. *Cybernetics and Systems*, *33*(6), 603–627. 17

Poliner, G. & Ellis, D. (2005). A classification approach to melody transcription. In *Proc. 6th Int. Conf. on Music Inform. Retrieval*, pp. 161–166. London. 30, 31, 40

Poliner, G. E., Ellis, D. P. W., Ehmann, A. F., Gómez, E., Steich, S., & Ong, B. (2007). Melody transcription from music audio: Approaches and evaluation. *IEEE Trans. on Audio, Speech, and Language Process.*, *15*(4), 1247–1256. 4, 5, 42

Pope, S. T. (1986). The development of an intelligent composer's assistant: Interactive graphics tools and knowledge representation for music. In *Int. Computer Music Conf.*, pp. 131–144. The Hague, The Netherlands. 13

Quinlan, R. (1993). *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann Publishers. 173

Rabiner, L. R. & Schafer, R. W. (1978). *Digital Processing of Speech Signals*. NJ: Prentice-Hall. 26

Rafii, Z. & Pardo, B. (2012). Music/voice separation using the similarity matrix. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 583–588. Porto, Portugal. 40

Rafii, Z. & Pardo, B. (2013). Repeating pattern extraction technique (REPET): A simple method for music/voice separation. *IEEE Trans. on Audio, Speech, and Language Processing*, *21*(1), 71–82. 18, 37, 39, 40

Ranjani, H. G., Arthi, S., & Sreenivas, T. V. (2011). Carnatic music analysis: Shadja, swara identification and rAga verification in AlApana using stochastic models. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 29–32. New Paltz, NY, USA. 167, 170

Rao, V. & Rao, P. (2010). Vocal melody extraction in the presence of pitched accompaniment in polyphonic music. *IEEE Trans. on Audio, Speech, and Language Processing*, *18*(8), 2145–2154. 31, 33, 34, 36, 81

Rasch, R. & Plomp, R. (1999). The perception of musical tones. In D. Deutsch (Ed.) *The Psychology of Music*, 2nd edition, chap. 4, pp. 89–112. San Diego: Academic Press. 9

Ratzan, L. (2004). *Understanding information systems: what they do and why we need them*. Chicago, USA: American Library Association. 130

Ravuri, S. & Ellis, D. P. W. (2010). Cover song detection: From high scores to general classification. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 65–68. Dallas, Texas, USA. 131, 132, 139

Régnier, L. & Peeters, G. (2009). Singing voice detection in music tracks using direct voice vibrato detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 204

Régnier, L. & Peeters, G. (2010). Partial clustering using a time-varying frequency model for singing voice detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 204

Ringer, A. L. (2013). Melody. Grove Music Online, Oxford Music Online (last checked Jan. 2013). Online: `http://www.oxfordmusiconline.com/subscriber/article/grove/music/18357`. 1, 3, 8

Robinson, D. (2013). Equal loudness filter. Hydrogenaudio Knowledgebase (last checked May 2013). Online: `http://replaygain.hydrogenaudio.org/proposal/equal_loudness.html`. 57

Robinson, D. W. & Dadson, R. S. (1956). A re-determination of the equal-loudness relations for pure tones. *British J. of Applied Physics*, *7*, 166–181. 32, 57

Rocamora, M. (2011). *Singing voice detection in polyphonic music*. Master's thesis, Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay. 204

Rocamora, M. & Herrera, P. (2007). Comparing audio descriptors for singing voice detection in music audio files. In *11th Brazilian Symposium on Computer Music*, pp. 187–196. São Paulo, Brazil. 204

Rocamora, M. & Pardo, A. (2012). Separation and classification of harmonic sounds for singing voice detection. In L. Alvarez, M. Mejail, L. Gomez, & J. Jacobo (Eds.) *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, *Lecture Notes in Computer Science*, vol. 7441, pp. 707–714. Springer Berlin Heidelberg. 205

Ross, J. C., Vinutha, T. P., & Rao, P. (2012). Detecting melodic motifs from audio for Hindustani classical music. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 193–198. Porto, Portugal. 166

Rump, H., Miyabe, S., Tsunoo, E., Ono, N., & Sagayama, S. (2010). Autoregressive MFCC models for genre classification improved by harmonic-percussion separation. In *11th Int. Soc. for Music Info. Retrieval Conf.*, pp. 87–92. Utrecht, The Netherlands. 157

Ryynänen, M. (2006). Singing transcription. In A. Klapuri & M. Davy (Eds.) *Signal Processing Methods for Music Transcription*, chap. 12. New York: Springer. 180

Ryynänen, M. (2008). *Automatic Transcription of Pitch Content in Music and Selected Applications*. Ph.D. thesis, Tampere University of Technology, Finland. 16

Ryynänen, M. & Klapuri, A. (2008a). Automatic transcription of melody, bass line, and chords in polyphonic music. *Computer Music J.*, *32*(3), 72–86. 15, 29, 31, 33, 34, 36, 62, 112, 186, 190

Ryynänen, M. & Klapuri, A. (2008b). Query by humming of MIDI and audio using locality sensitive hashing. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 2249–2252. Las Vegas, USA. 133, 155

Salamon, J. & Gómez, E. (2011). Melody extraction from polyphonic music: MIREX 2011. In *7th Music Info. Retrieval Evaluation eXchange (MIREX)*, extended abstract. Miami, USA. 109

Salamon, J. & Gómez, E. (2012). Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Trans. on Audio, Speech, and Language Processing*, *20*(6), 1759–1770. 31, 32, 33, 34, 35, 36, 47, 50, 51, 79, 96, 104, 115, 132, 134

Salamon, J., Gómez, E., & Bonada, J. (2011). Sinusoid extraction and salience function design for predominant melody estimation. In *Proc. 14th Int. Conf. on Digital Audio Effects (DAFx-11)*, pp. 73–80. Paris, France. 33, 55

Salamon, J., Gómez, E., Ellis, D. P. W., & Richard, G. (2013a). Melody extraction from polyphonic music signals: Approaches, applications and challenges. *IEEE Signal Processing Magazine*. (to appear). 25

Salamon, J., Gulati, S., & Serra, X. (2012a). A multipitch approach to tonic identification in indian classical music. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 499–504. Porto, Portugal. 130, 165, 167

Salamon, J., Peeters, G., & Röbel, A. (2012b). Statistical characterisation of melodic pitch contours and its application for melody extraction. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 187–192. Porto, Portugal. 81

Salamon, J., Rocha, B., & Gómez, E. (2012c). Musical genre classification using melody features extracted from polyphonic music signals. In *Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, pp. 81–84. Kyoto, Japan. 3, 86, 112, 130, 157

Salamon, J. & Rohrmeier, M. (2009). A quantitative evaluation of a two stage retrieval approach for a melodic query by example system. In *10th Int. Soc. for Music Info. Retrieval Conf.*, pp. 255–260. Kobe, Japan. 151, 152, 155

Salamon, J., Serrà, J., & Gómez, E. (2013b). Tonal representations for music retrieval: from version identification to query-by-humming. *Int. J. of Multimedia Info. Retrieval, special issue on Hybrid Music Info. Retrieval*, *2*(1), 45–58. 3, 130

Salamon, J. & Urbano, J. (2012). Current challenges in the evaluation of predominant melody extraction algorithms. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 289–294. Porto, Portugal. 113

Scaringella, N., Zoia, G., & Mlynek, D. (2006). Automatic genre classification of music content: a survey. *IEEE Signal Processing Magazine*, *23*(2), 133–141. 14, 157

Scheirer, E. D. (1996). Bregman's chimerae: Music perception as auditory scene analysis. In *Int. Conf. on Music Perception and Cognition*, pp. 317–322. Montreal, Quebec, Canada. 18

Scheirer, E. D. (1999). Towards music understanding without separation: Segmenting music with correlogram comodulation. In *IEEE Worksh. on Apps. of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 99–102. New Paltz, New York, USA. 18

Scheirer, E. D. (2000). *Music-Listening Systems*. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, USA. 18, 19

Selfridge-Field, E. (1998). Conceptual and representational issues in melodic comparison. In W. B. Hewlett & E. Selfridge-Field (Eds.) *Melodic Similarity – Concepts, Procedures, and Applications*, chap. 1, pp. 3–64. Cambridge, Massachusetts: MIT Press. 1

Sengupta, R., Dey, N., Nag, D., Datta, A., & Mukerjee, A. (2005). Automatic tonic (sa) detection algorithm in Indian classical vocal music. In *National Symposium on Acoustics*, pp. 1–5. Bangalore, India. 167

Serrà, J. (2011). *Identification of versions of the same musical composition by processing audio descriptions*. Ph.D. thesis, Universitat Pompeu Fabra, Barcelona, Spain. 14, 130, 131

Serrà, J., Gómez, E., & Herrera, P. (2010). Audio cover song identification and similarity: background, approaches, evaluation, and beyond. In Z. W. Raś & A. A. Wieczorkowska (Eds.) *Advances in Music Information Retrieval*, *Studies in Computational Intelligence*, vol. 274, chap. 14, pp. 307–332. Berlin, Germany: Springer. 130, 131, 132, 135, 136, 139, 143, 144, 145, 152

Serrà, J., Gómez, E., Herrera, P., & Serra, X. (2008). Statistical analysis of chroma features in western music predicts human judgments of tonality. *J. of New Music Research*, *37*(4), 299–309. 138

Serrà, J., Kantz, H., Serra, X., & Andrzejak, R. G. (2012). Predictability of music descriptor time series and its application to cover song detection. *IEEE Trans. on Audio, Speech, and Language Processing*, *20*(2), 514–525. 141, 152

Serrà, J., Serra, X., & Andrzejak, R. G. (2009). Cross recurrence quantification for cover song identification. *New J. of Physics*, *11*(9), 093017. 138, 139, 151

Serra, X. (2011). A multicultural approach in music information research. In *12th Int. Soc. for Music Info. Retrieval Conf.*, pp. 151–156. Miami, USA. 175

Serra, X., Bresin, R., & Camurri, A. (2007). Sound and music computing: Challenges and strategies. *J. of New Music Research*, *36*(3), 185–190. 3

Skalak, M., Han, J., & Pardo, B. (2008). Speeding melody search with vantage point trees. In *9th Int. Conf. on Music Info. Retrieval*, pp. 95–100. Philadelphia, USA. 152

Smaragdis, P. & Brown, J. (2003). Non-negative matrix factorization for polyphonic music transcription. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pp. 177–180. New Paltz, NY. 17

Smith III, J. O. (2013). Spectral audio signal processing. Center for Computer Research in Music and Acoustics (CCRMA), Stanford University, USA. Online: https://ccrma.stanford.edu/~jos/sasp/. 26

Solomon, L. (1997). Melody. Glossary of Technical Musical Terms (last checked Jan. 2013). Online: http://solomonsmusic.net/glossary.htm. 4

Song, J., Bae, S. Y., & Yoon, K. (2002). Mid-level music melody representation of polyphonic audio for query-by-humming system. In *3rd Int. Conf. on Music Info. Retrieval*, pp. 133–139. Paris, France. 133

Sturm, B. L. (2012). On automatic music genre recognition by sparse representation classification using auditory temporal modulations. In *9th Int. Symp. on Computer Music Modeling and Retrieval*, pp. 379–394. London, UK. 157

Sun, R. (2008). Introduction to computational cognitive modeling. In R. Sun (Ed.) *The Cambridge Handbook of Computational Psychology*, chap. 1, pp. 3–19. New York: Cambridge University Press. 2

Sundberg, J. (1987). *The Science of the Singing Voice*. DeKalb, IL: Northern Illinois University Press. 32, 62

Sundberg, J. (1995). Acoustic and psychoacoustic aspects of vocal vibrato. In P. Dejonckere, M. Hirano, & J. Sundberg (Eds.) *Vibrato*, pp. 35–62. San Diego: Singular Publishing Group. 84, 159

Sundberg, J. & Thalén, M. (2001). Describing different styles of singing: A comparison of a female singer's voice source in 'classical', 'pop', 'jazz' and 'blues'. *Logopedics, Phoniatrics Vocology*, *26*(2), 82–93. 157, 159, 164

Sutton, C. (2006). *Transcription of vocal melodies in popular music*. Master's thesis, Queen Mary, University of London, London, UK. 31, 40

Tachibana, H., Ono, T., Ono, N., & Sagayama, S. (2010a). Extended abstract for audio melody extraction in MIREX 2010. In *6th Music Info. Retrieval Evaluation eXchange (MIREX)*, extended abstract. Utrecht, The Netherlands. 100, 108

Tachibana, H., Ono, T., Ono, N., & Sagayama, S. (2010b). Melody line estimation in homophonic music audio signals based on temporal-variability of melodic source. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 425–428. 30, 31, 37, 38, 39, 47, 102

Talkin, D. (1995). A robust algorithm for pitch tracking (RAPT). In W. B. Kleijn & K. K. Paliwal (Eds.) *Speech Coding and Synthesis*, chap. 14, pp. 495–518. Elsevier Science B.V. 27

Terhardt, E. (1974). Pitch, consonance, and harmony. *J. Acoust. Soc. Am.*, *55*(5), 1061–1069. 9

Terhardt, E. (1979). Calculating virtual pitch. *Hearing Research*, *1*(2), 155–182. 26

Terhardt, E., Stoll, G., & Seewann, M. (1982). Algorithm for extraction of pitch and pitch salience from complex tonal signals. *J. Acoust. Soc. Am.*, *71*(3), 679–688. 26

Toiviainen, P. & Eerola, T. (2006). Visualization in comparative music research. In A. Rizzi & M. Vichi (Eds.) *COMPSTAT 2006 – Proceedings in Computational Statistics*, pp. 209–221. Heidelberg: Physica-Verlag. 180

Tsai, W.-H., Yu, H.-M., & Wang, H.-M. (2008). Using the similarity of main melodies to identify cover versions of popular songs for music document retrieval. *J. of Info. Science and Engineering*, *24*(6), 1669–1687. 131, 144

Typke, R. (2007). *Music Retrieval based on Melodic Similarity*. Ph.D. thesis, Utrecht University, Netherlands. 131

Typke, R. & Walczak-Typke, A. (2008). A tunneling-vantage indexing method for non-metrics. In *9th Int. Conf. on Music Info. Retrieval*, pp. 683–688. Philadelphia, USA. 152

Tzanetakis, G. & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Trans. on Speech and Audio Processing*, *10*(5), 293–302. 157, 161

Urbano, J. (2011). Information retrieval meta-evaluation: Challenges and opportunities in the music domain. In *Int. Soc. for Music Inform. Retrieval Conf.*, pp. 609–614. Miami, USA. 113, 117, 122

Urbano, J., Marrero, M., & Martín, D. (2013a). On the measurement of test collection reliability. In *36th Annual Int. ACM SIGIR Conf.* Dublin, Ireland. (to appear). 121

Urbano, J. & Schedl, M. (2013). Minimal test collections for low-cost evaluation of audio music similarity and retrieval systems. *Int. J. of Multimedia Info. Retrieval, special issue on Hybrid Music Info. Retrieval*, *2*(1), 59–70. 127

Urbano, J., Schedl, M., & Serra, X. (2013b). Evaluation in music information retrieval. *J. of Intelligent Info. Systems*. (to appear). 113

Vembu, S. & Baumann, S. (2005). Separation of vocals from polyphonic audio recordings. In *6th Int. Conf. on Music Info. Retrieval*, pp. 337–344. London, UK. 37

Verfaille, V., Zölzer, U., & Arfib, D. (2006). Adaptive digital audio effects (a-DAFx): a new class of sound transformations. *IEEE Trans. on Audio, Speech, and Language Processing*, *14*(5), 1817–1831. 14

Vickers, E. (2001). Automatic long-term loudness and dynamics matching. In *111th Audio Engineering Society Convention*, 5495. New York, USA. 57

Vincent, E. (2004). *Instrument Models for Source Separation and Transcription of Music Recordings*. Ph.D. thesis, University of Paris 6, France. 17

Vincent, E., Araki, S., Theis, F., Nolte, G., Bofill, P., Sawada, H., Ozerov, A., Gowreesunker, V., Lutter, D., & Duong, N. Q. K. (2012). The signal separation evaluation campaign (2007–2010): Achievements and remaining challenges. *Signal Processing*, *92*(8), 1928–1936. 17

Vincent, E. & Deville, Y. (2010). Audio applications. In P. Comon & C. Jutten (Eds.) *Handbook of Blind Source Separation, Independent Component Analysis and Applications*, chap. 19, pp. 779–819. Academic Press. 17

Virtanen, T. (2006). *Sound Source Separation in Monaural Music Signals*. Ph.D. thesis, Tampere University of Technology, Finland. 17

Viswanathan, T. & Allen, M. H. (2004). *Music in South India*. Oxford University Press. 166, 169

Voorhees, E. M. & Harman, D. K. (2005). *TREC: Experiment and Evaluation in Information Retrieval*. Cambridge MA: MIT Press. 121

Wang, D. & Brown, G. J. (Eds.) (2006). *Computational Auditory Scene Analysis: Principles, Algorithms and Applications*. New York: Wiley-IEEE. 15

Wasserman, L. (2003). *All of statistics: a concise course in statistical inference*. Berlin, Germany: Springer. 144

Witten, I. H. & Frank, E. (2005). *Data mining: practical machine learning tools and techniques*. Waltham, USA: Morgan Kaufmann, 2nd edn. 140, 173

Wordsmyth (2013). Melody. Educational Dictionary-Thesaurus (last checked Jan. 2013). Online: `http://www.wordsmyth.net/?level=3&ent=melody`. 4

Yeh, T.-C., Wu, M.-J., Jang, J.-S., Chang, W.-L., & Liao, I.-B. (2012). A hybrid approach to singing pitch extraction based on trend estimation and hidden markov models. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 457–460. Kyoto, Japan. 31, 32, 33, 34, 36, 39, 47, 204

Yoon, J.-Y., Song, C.-J., Lee, S.-P., & Park, H. (2011). Extracting predominant melody of polyphonic music based on harmonic structure. In *7th Music Info. Retrieval Evaluation eXchange (MIREX)*, extended abstract. Miami, USA. 102

Zhang, G., Lu, K., & Wang, B. (2012). Query reformulation based on user habits for query-by-humming systems. In Y. Hou, J.-Y. Nie, L. Sun, B. Wang, & P. Zhang (Eds.) *Information Retrieval Technology*, *Lecture Notes in Computer Science*, vol. 7675, pp. 386–395. Springer Berlin Heidelberg. 155

Zwicker, E. (1961). Subdivision of the audible frequency range into critical bands (frequenzgruppen). *J. Acoust. Soc. Am.*, *33*(2), 248–248. 59

Appendix A

MELODIA

MELODIA is an implementation of the melody extraction algorithm presented in this thesis, in the form of a vamp[1] plug-in. Vamp plug-ins are similar to VST plug-ins in that they are implemented as multi-platform libraries which can be loaded by any host which implements the vamp architecture. Unlike VST plug-ins, vamp plug-ins do not output audio. Rather, they take an audio signal as input and output *information*. This makes the vamp architecture and ideal solution for distributing MIR algorithms.

MELODIA was announced on October 5th 2012 and presented at the ISMIR 2012 International Society for Music Information Retrieval Conference. It is available as a compiled library for all three major operating systems (Windows, Linux and OSX), and can be downloaded for free for non-commercial purposes[2]. The logo of the plug-in is presented in Figure A.1:

Figure A.1: Logo of the MELODIA - Melody extraction vamp plug-in.

[1] http://vamp-plugins.org/
[2] http://mtg.upf.edu/technologies/melodia

There are two main way in which the plug-in can be used. First, it can be used to visualise the predominant melody of an audio recording, using Sonic Visualiser[3] vamp host. It allows the user to load an audio file, extract the melody and visualise its f_0 sequence. It also allows to visualise the intermediate steps of the algorithm: the salience function, the pitch contours tracked from the salience function, and the pitch contours after filtering non-melodic contours. The user can also listen to the audio file whilst viewing the different visualisations of the data, which are synchronised with the audio in time. An example of using MELODIA in Sonic Visualiser is provided in Figure A.2. In the top pane we see the waveform of the recording being analysed. In the second pane, the salience function. In the third pane, all the pitch contours tracked from the salience function (before contour filtering). Finally in the bottom pane we display spectrogram of the signal with the final melody f_0 sequence extracted by the algorithm overlaid in red.

In addition to running the algorithm default parameter values (those reported in the thesis), the user can also manually set the value of three key parameters: the maximum and minimum allowed frequency (in Hz) of the melody, and the voicing parameter ν (cf. Section 4.2.3). We have also implemented a basic energy-based threshold parameter for filtering out background noise (e.g. laptop fan) when analysing monophonic recordings. In Figure A.3 we present the interface for controlling the parameters of the algorithm in Sonic Visualiser.

The second way in which MELODIA can be used is for batch processing of an entire music collection. For this, MELODIA can be run using the Sonic Annotator[4] command-line tool. In this way we facilitate the use of our algorithm for large-scale computational music analysis. Detailed instructions for using MELODIA both with Sonic Visualiser and Sonic Annotator are provided in the README file which is included with the library.

Since its announcement in October of 2012, MELODIA has been downloaded over 4,000 times (4,400 as of June 15$^{\text{th}}$ 2013) by researchers, educators, artists and hobbyists from all over the world. The estimated location of the first 2,000 downloads is visualised in Figure A.4.

[3] http://www.sonicvisualiser.org/
[4] http://www.omras2.org/sonicannotator

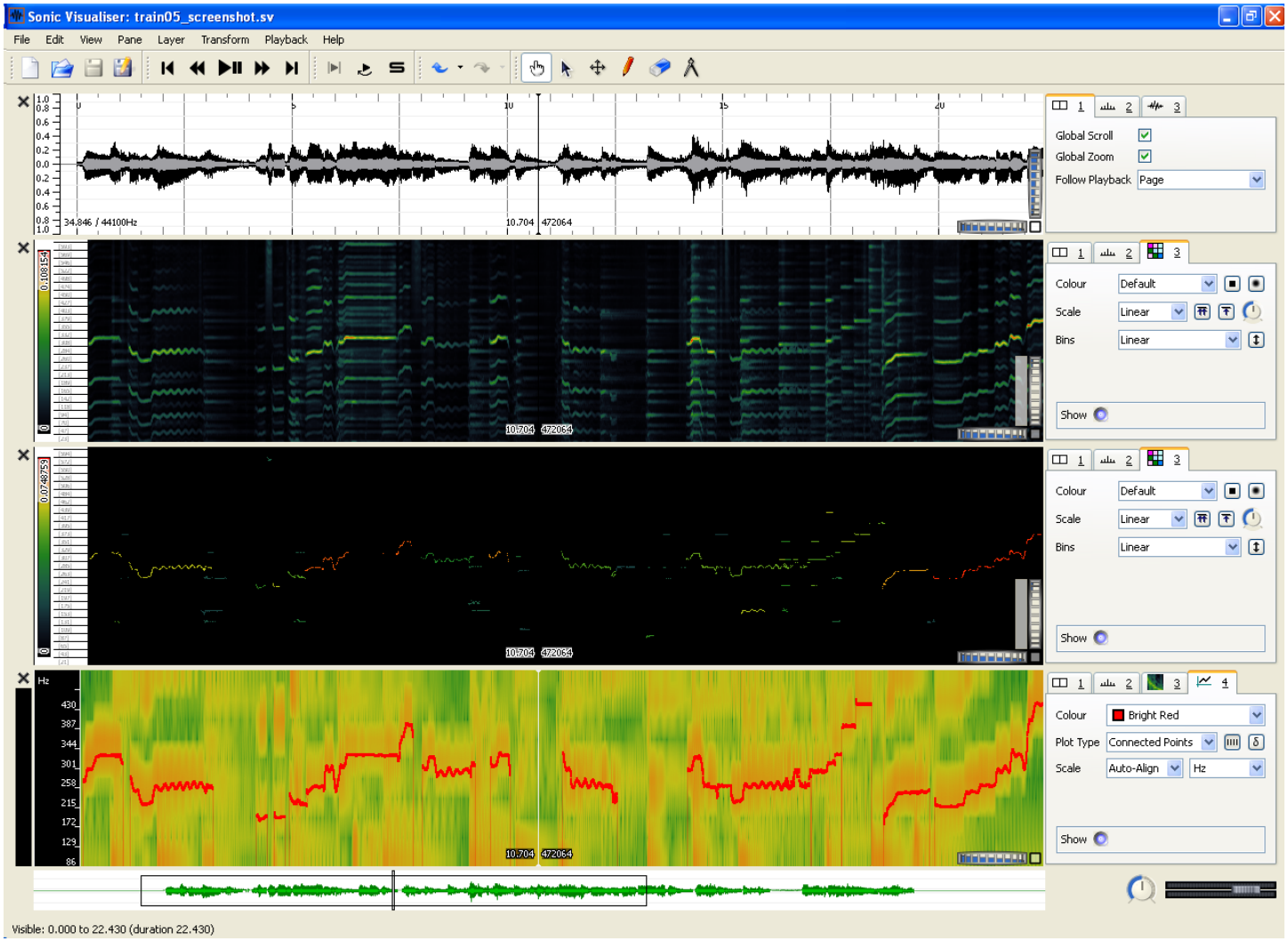

Figure A.2: Using MELODIA with Sonic Visualiser. Top pane: waveform. Second pane: salience function. Third pane: pitch contours (before contour filtering). Bottom pane: spectrogram with the final melody f_0 sequence extracted by the algorithm overlaid in red.

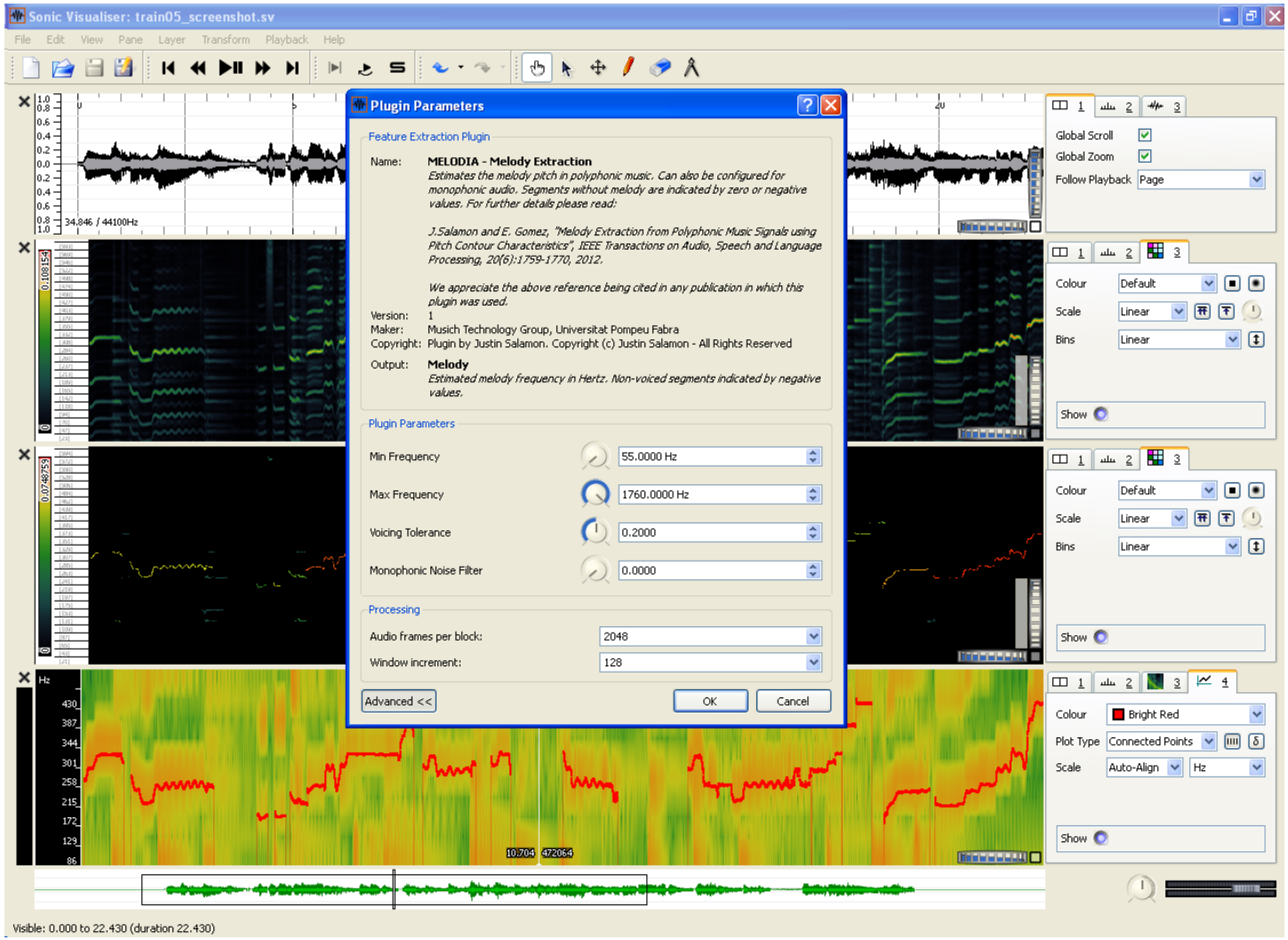

Figure A.3: Controlling the parameters of the melody extraction algorithm in MELODIA using the Sonic Visualiser interface.

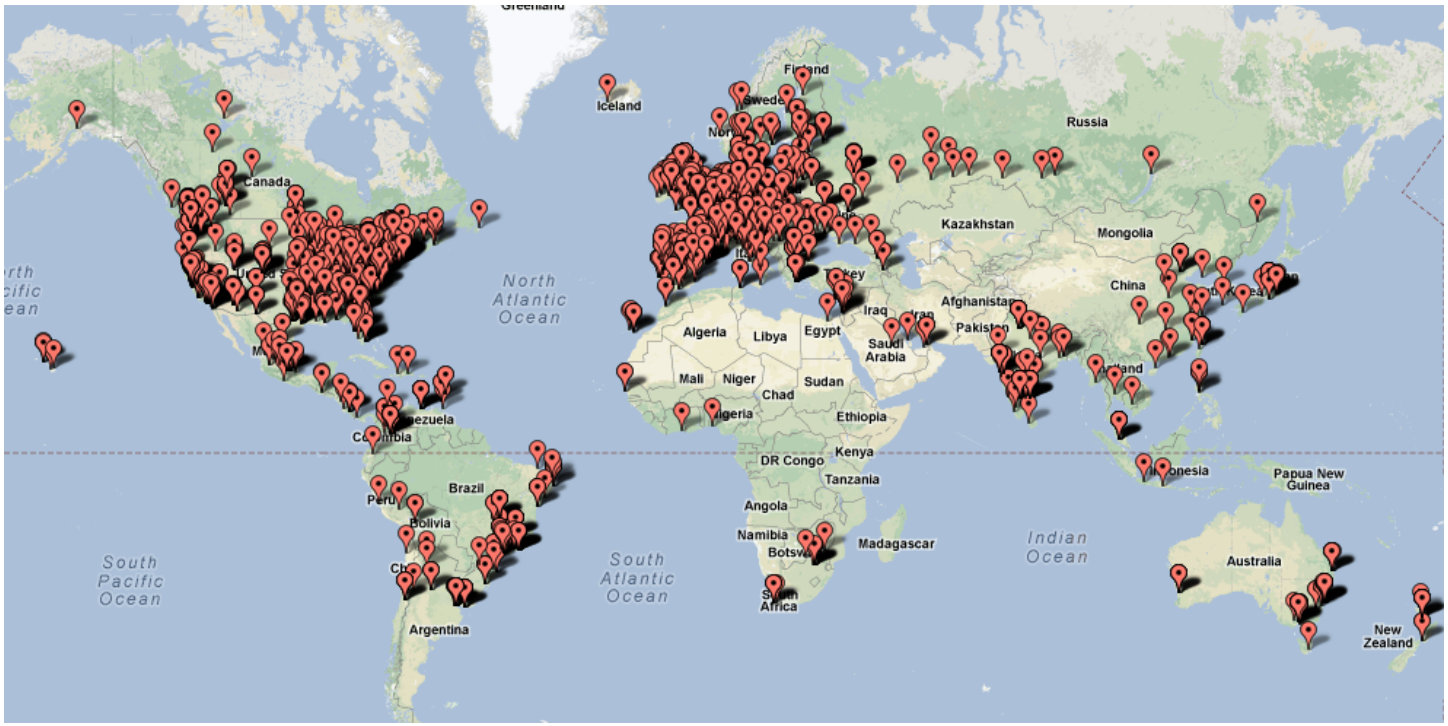

Figure A.4: MELODIA: estimated location of the first 2,000 downloads.

As far as the author is aware, at the time of writing this dissertation the algorithm has been (or is being) used in at least 2 European funded research projects:

- PHENICX EC project (contract no. 601166)
  http://phenicx.upf.edu
- CompMusic EC project (ERC-2010-AdG-267583)
  http://compmusic.upf.edu

5 ongoing doctoral theses (not including this one) at Universitat Pompeu Fabra, Barcelona, Spain (provisional titles):

- Gulati, S. Discovery and characterization of melodic motives in large audio music collections.
- Giraldo, S. Computational modeling of emotion, expression and interaction in music performance.
- Bosch, J. J. Melodic and structural analysis of musical audio.
- Şentürk, S. Linking fragments of score and audio recordings.
- Koduri, G. K. Knowledge-based similarity measures for music.

and 7 Master's theses at Universitat Pompeu Fabra, Barcelona, Spain:

- Kroher, N. (2013). The flamenco cante: Automatic characterization of flamenco singing by analyzing audio recordings.
- Parra, H. (2013). Study of robust pitch estimation with de-reverberation techniques.
- Valero, J. (2013). Measuring similarity of automatically extracted melodic pitch contours for audio-based query by humming of polyphonic music collections.
- Morelli, F. (2013). The bad and the good singer: Query analysis and reformulation for audio to audio query by humming.
- Chen, K. (2013). Characterization of pitch intonation in Beijing opera singing.
- Gulati, S. (2012). A tonic identification approach for Indian art music.
- Rocha, B. (2011). Genre classification based on predominant melodic pitch contours.

Appendix B

Publications by the author

Peer-reviewed journals

Salamon, J., Gómez, E., Ellis, D. P. W. & Richard, G. (2013). Melody extraction from polyphonic music signals: Approaches, applications and challenges. *IEEE Signal Processing Magazine*. (to appear).

Salamon, J., Serrà, J., & Gómez, E. (2013). Tonal representations for music retrieval: from version identification to query-by-humming. *Int. J. of Multimedia Info. Retrieval, special issue on Hybrid Music Info. Retrieval*, *2*(1):45–58.

Salamon, J. & Gómez, E (2012). Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Trans. on Audio, Speech, and Language Processing*, *20*(6):1759-1770.

Full-article contributions to peer-reviewed conferences

Bogdanov, D., Wack, N., Gómez, E., Gulati, S., Herrera, P., Mayor, O., Roma, G., Salamon, J., Zapata, J. & Serra, X. (2013). ESSENTIA: an audio analysis library for music information retrieval. In *14th Int. Soc. for Music Info. Retrieval Conf.*, Curitiba, Brazil. (to appear).

Salamon, J., Peeters, G., & Röbel, A. (2012). Statistical characterisation of melodic pitch contours and its application for melody extraction. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 178–192. Porto, Portugal.

Salamon, J. & Urbano, J. (2012). Current challenges in the evaluation of predominant melody extraction algorithms. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 289–294. Porto, Portugal.

Salamon, J., Gulati, S. & Serra, X. (2012). A multipitch approach to tonic identification in Indian classical music. In *13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 499–504. Porto, Portugal.

Gómez, E., Cañadas, F., Salamon, J., Bonada, J., Vera, P., & Cabañas, P. (2012). Predominant fundamental frequency estimation vs singing voice separation for the automatic transcription of accompanied flamenco singing. *In 13th Int. Soc. for Music Info. Retrieval Conf.*, pp. 601–606. Porto, Portugal.

Pikrakis, A., Gómez, F., Oramas, S., Báñez, J. M. D., Mora, J., Escobar, F., Gómez, E., & Salamon, J. (2012). Tracking melodic patterns in flamenco singing by analyzing polyphonic music recordings. In 13th Int. Soc. for Music Info. Retrieval Conf., pp. 421–426. Porto, Portugal.

Salamon, J., Serrà, J. & Gómez, E. (2012). Melody, bass line, and harmony representations for music version identification. In *21st Int. World Wide Web Conf. (WWW 2012): 4th Int. Workshop on Advances in Music Information Research (AdMIRe 2012)*, pp. 887–894. Lyon, France.

Gómez, E., Bonada, J., & Salamon, J. (2012). Automatic transcription of flamenco singing from monophonic and polyphonic music recordings, In *2nd Int. Workshop on Folk Music Analysis (FMA), 3rd Interdisciplinary Conf. on Flamenco Research (INFLA)*, pp. 199–210. Seville, Spain.

Gómez, F., Pikrakis, A., Mora, J., Bañez, J. M. D., Gómez, E., Escobar, F., Oramas, S., & Salamon, J. (2012). Automatic detection of melodic patterns in flamenco singing by analyzing polyphonic music recordings", In *2nd Int. Workshop on Folk Music Analysis (FMA), 3rd Interdisciplinary Conf. on Flamenco Research (INFLA)*, pp. 225–234. Seville, Spain.

Salamon, J., Rocha, B., & Gómez, E. (2012). Musical genre classification using melody features extracted from polyphonic music signals. In *IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, pp. 81–84. Kyoto, Japan.

Salamon, J., Gómez, E., & Bonada, J. (2011). Sinusoid extraction and salience function design for predominant melody estimation. In *14th Int. Conf. on Digital Audio Effects (DAFx-11)*, pp. 73–80. Paris, France.

Salamon, J. & Rohrmeier, M. (2009). A quantitative evaluation of a two stage retrieval approach for a melodic query by example system. In *10th Int. Soc. for Music Info. Retrieval Conf.*, pp. 255–260. Kobe, Japan.

Salamon, J. & Gómez, E. (2009). A chroma-based salience function for melody and bass line estimation from music audio signals. In *Sound and Music Computing Conf.*, pp. 331–336. Porto, Portugal.

Other contributions to conferences

Salamon, J. & Gómez, E. (2011). Melody extraction from polyphonic music: MIREX 2011. In *5th Music Info. Retrieval Evaluation eXchange (MIREX)*, extended abstract. Miami, USA.

Salamon, J. & Gómez, E. (2010). Melody extraction from polyphonic music audio. In *6th Music Info. Retrieval Evaluation eXchange (MIREX)*, extended abstract. Utrecht, The Netherlands.

Theses

Salamon, J. (2008). Chroma-based predominant melody and bass line extraction from music audio signals. Master's thesis, Universitat Pompeu Fabra, Barcelona, Spain.

Additional and up-to-date information about the author may be found at the author's web page[1].

[1] http://www.justinsalamon.com

Appendix C

Salience function parameter configurations

This appendix contains the graphs referred to in Section 3.5.2 of the thesis. Each graph presents the results obtained for one of the four measures: $\overline{\Delta f_m}$, $\overline{RR_m}$, $\overline{S_1}$ and $\overline{S_3}$, and includes the results obtained using each of the 12 analysis configurations summarised in Table 3.1 of Chapter 3, reproduced here as a legend in for the plots that follow:

Config.	Filtering	Spectral Transform	Frequency/Amplitude Correction	Marker
1	none	STFT	none	×
2			Parabolic	○
3			IF (phase)	△
4		MRFFT	none	×
5			Parabolic	○
6			IF (phase)	△
7	Eq. loudness	STFT	none	×
8			Parabolic	○
9			IF (phase)	△
10		MRFFT	none	×
11			Parabolic	○
12			IF (phase)	△

Table C.1: Analysis configurations for sinusoid extraction.

For each measure we plot two graphs: in the first the parameter $\beta = 1$ and in the second $\beta = 2$. Each datapoint in the graph represents one of the 64 possible combinations of the other salience function parameters:

- $\alpha = 1.0$, 0.9, 0.8, 0.6
- $N_h = 4$, 8, 12, 20
- $\gamma = \infty$, 60 dB, 40 dB, 20 dB

The first 16 datapoints represent configurations where $\alpha = 1$, the next 16 where $\alpha = 0.9$ and so on. Within each group of 16, the first 4 have $N_h = 4$, the next 4 have $N_h = 8$ etc. Finally within each group of 4, each datapoint has a different γ value from ∞ down to 20 dB.

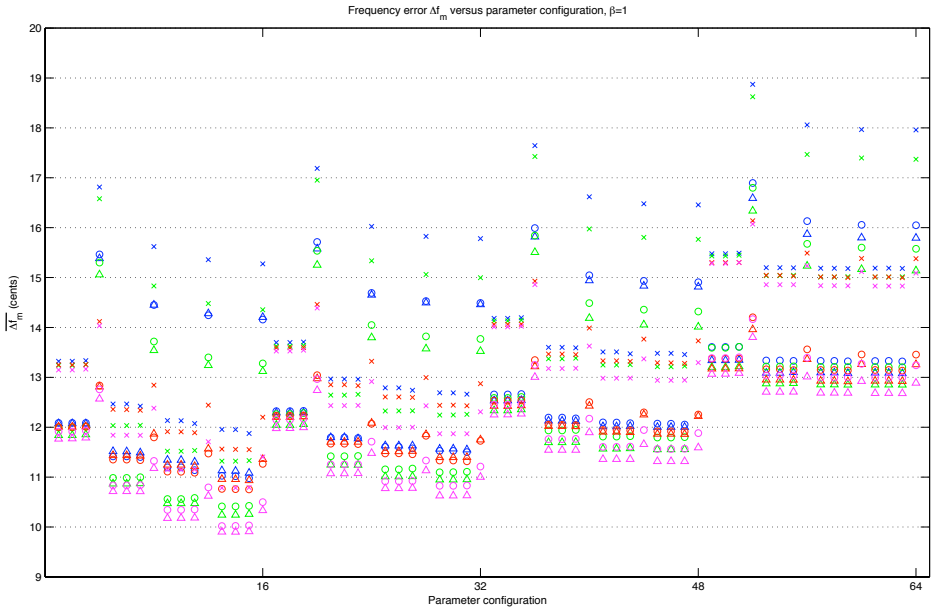

Figure C.1: Results for $\overline{\Delta f_m}$ by parameter configuration when $\beta = 1$.

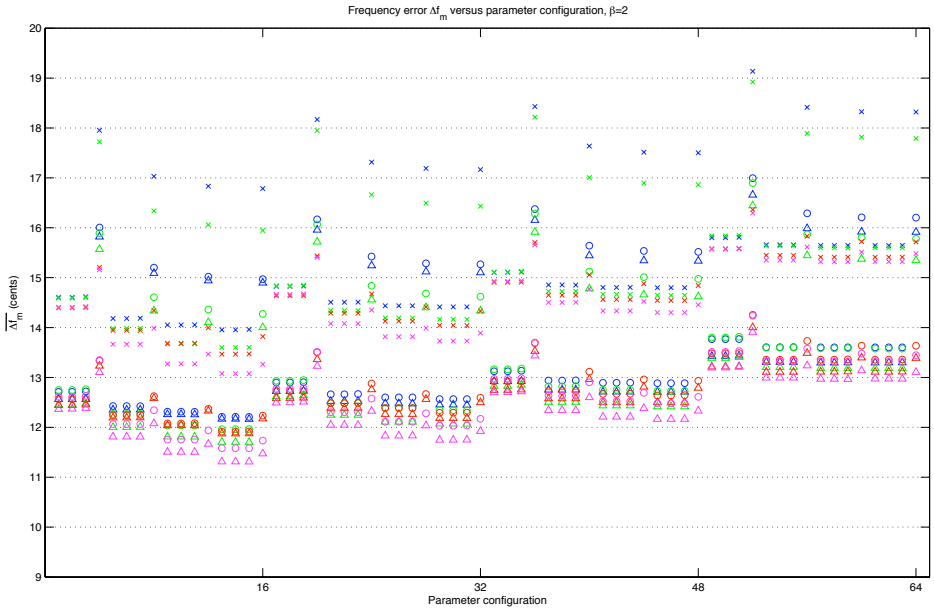

Figure C.2: Results for $\overline{\Delta f_m}$ by parameter configuration when $\beta = 2$.

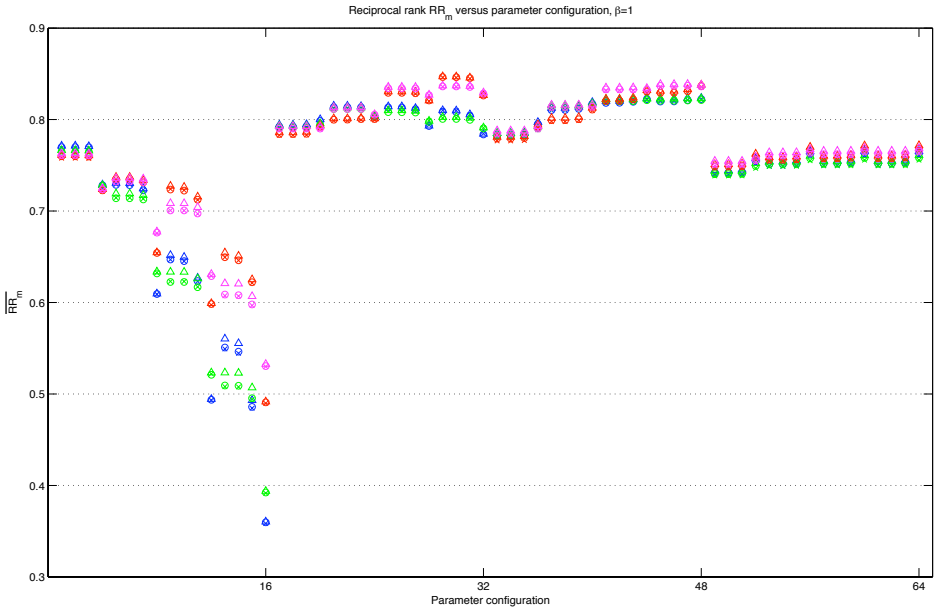

Figure C.3: Results for $\overline{RR_m}$ by parameter configuration when $\beta = 1$.

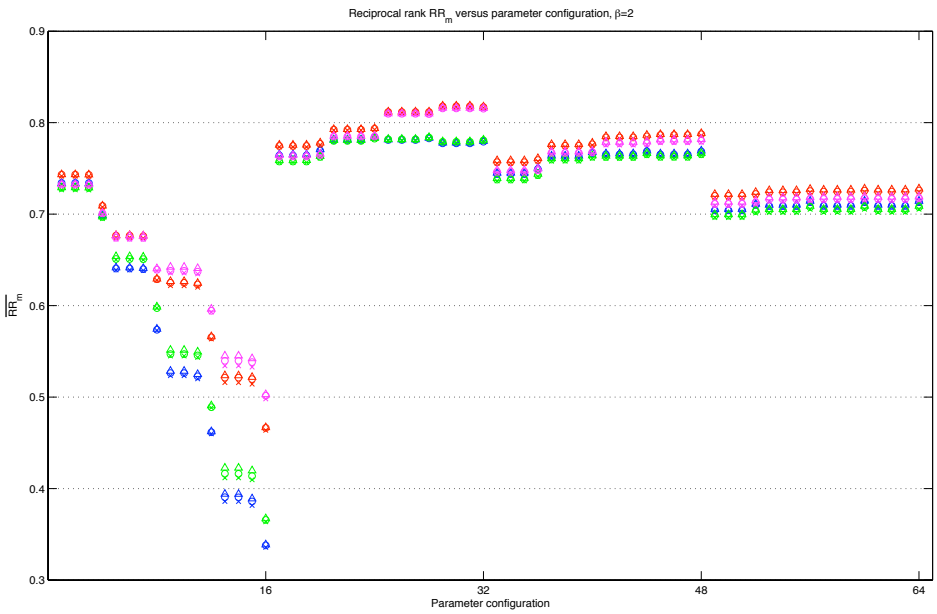

Figure C.4: Results for $\overline{RR_m}$ by parameter configuration when $\beta = 2$.

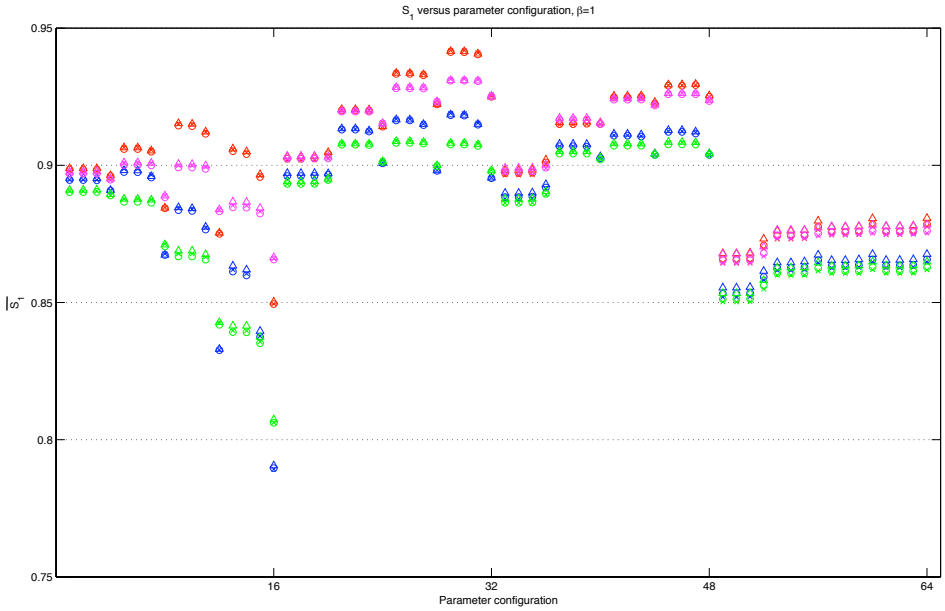

Figure C.5: Results for $\overline{S_1}$ by parameter configuration when $\beta = 1$.

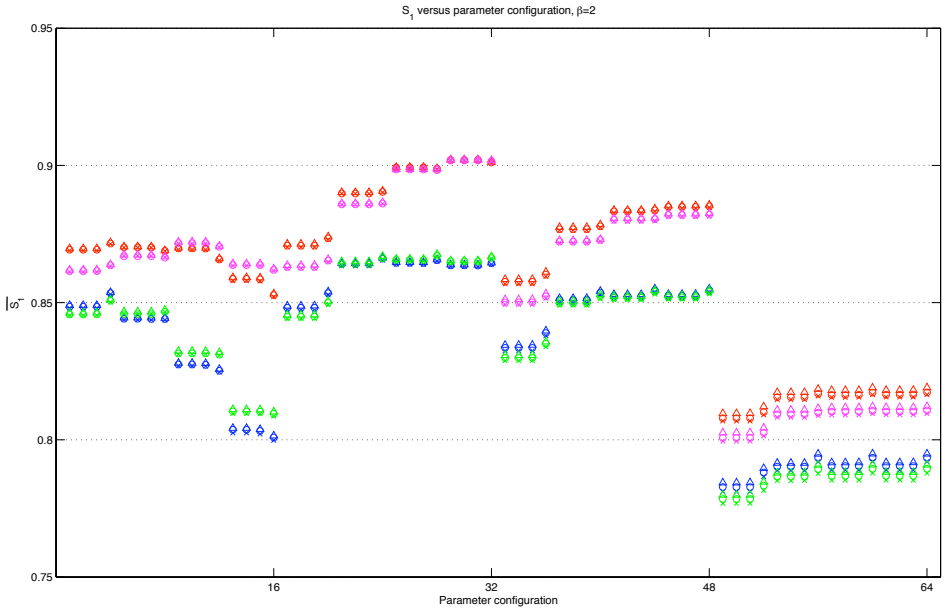

Figure C.6: Results for $\overline{S_1}$ by parameter configuration when $\beta = 2$.

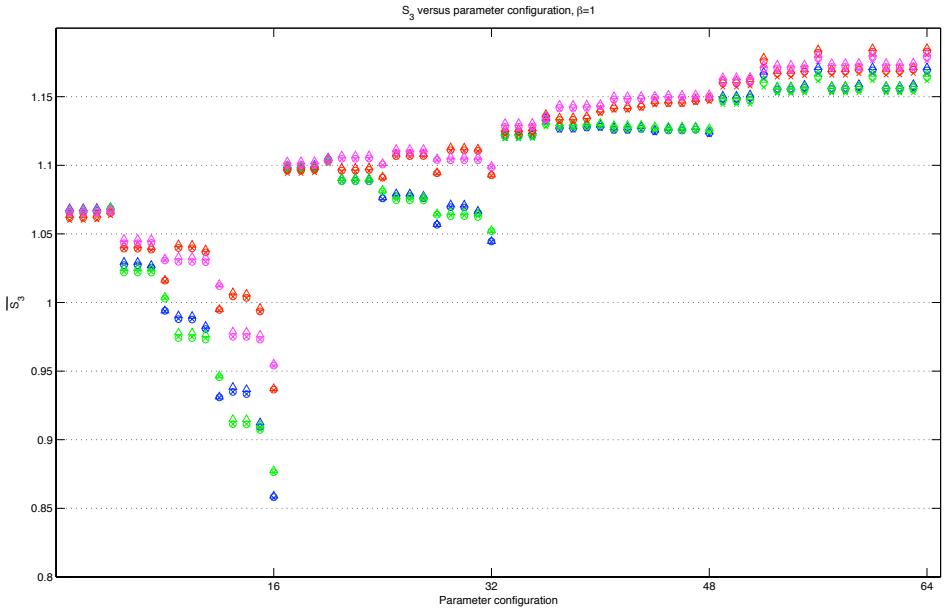

Figure C.7: Results for $\overline{S_3}$ by parameter configuration when $\beta = 1$.

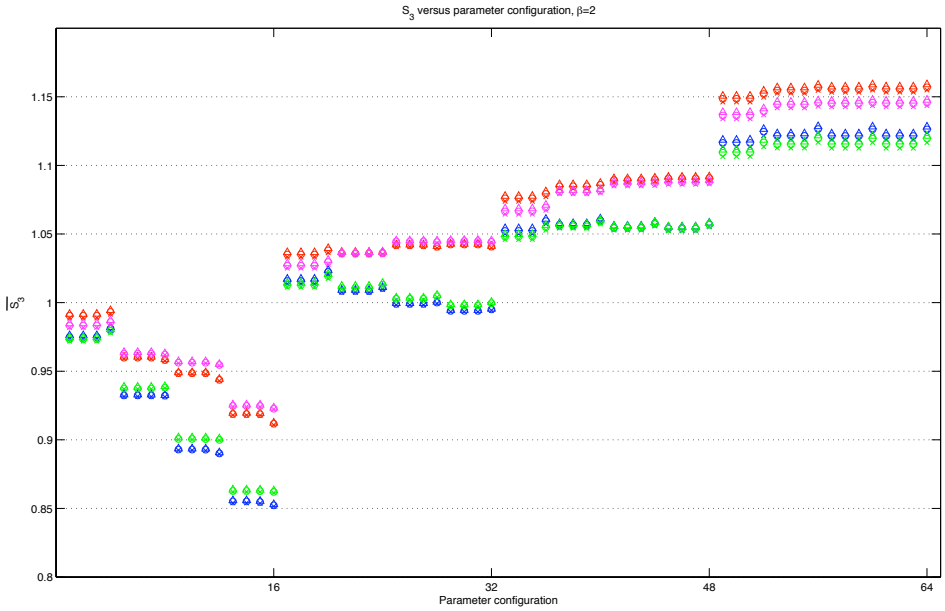

Figure C.8: Results for $\overline{S_3}$ by parameter configuration when $\beta = 2$.