Graz University of Technology

PhD Thesis

Probabilistic Model-Based Multiple Pitch Tracking of Speech

conducted at the
Signal Processing and Speech Communications Laboratory
Graz University of Technology, Austria

by
Dipl.-Ing. Michael Wohlmayr, 0130879

Supervisors:
Assoc.Prof. Dipl.-Ing. Dr. Franz Pernkopf
Univ.-Prof. Dipl.-Ing. Dr.techn. Gernot Kubin

Assessors/Examiners:
Assoc.Prof. Dipl.-Ing. Dr. Franz Pernkopf
Adjunct Prof. Dr. Tuomas Virtanen

Graz, May 10, 2012

This work was funded by the Austrian Science Fund (FWF) under projects P22488-N23, S10604-N13 and S10610-N13.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

date	(signature)

Abstract

Multiple pitch tracking of speech is an important task for the segregation of multiple speakers in a single-channel recording. In this thesis, a probabilistic model-based approach for estimation and tracking of multiple pitch trajectories is proposed. A probabilistic model that captures pitch-dependent characteristics of the single-speaker short-time spectrum is obtained a priori from clean speech data. The resulting speaker model, which is based on Gaussian mixture models, can be trained either in a speaker independent (SI) or a speaker dependent (SD) fashion. Speaker models are then combined using an interaction model to obtain a probabilistic description of the observed speech mixture. A factorial hidden Markov model is applied for tracking the pitch trajectories of multiple speakers over time.

The probabilistic model-based approach is capable to explicitly incorporate timbral information and all associated uncertainties of spectral structure into the model. While SI models allow an ad-hoc use in situations where the speakers in a recording are unknown, SD models have the great advantage that pitch trajectories can be assigned to their corresponding speakers. The accuracy of the proposed method is evaluated on two speech databases and compared to a state-of-the-art algorithm for multi-pitch tracking of speech.

Two problems related to the proposed approach are addressed. (i) Exact inference has a high computational demand, mainly due to the fact that the solution is obtained by considering all possible pitch combinations across speakers. A novel method for approximate inference based on likelihood pruning is proposed. The method is based on a computationally efficient upper and lower bound on the likelihood of pitch combinations. The approximate method is experimentally evaluated in terms of accuracy and time requirements, and results for tracking the pitch of three simultaneously talking speakers are demonstrated. (ii) Any mismatch between training and testing conditions (such as different acoustic channel conditions or gain mismatches) deteriorates the accuracy of multi-pitch tracking. It is desirable to adapt speaker models to novel environmental conditions during multi-pitch tracking, i.e. in situations where only a mixture of speakers is available. We propose a modification of the maximum likelihood linear regression (MLLR) technique where the adaptation of model parameters is constrained to modifications of the spectral envelope. This constraint is beneficial for cases where few adaptation data is available. Based on this, we propose a novel expectation-maximization (EM) algorithm for adaptation of speaker models from speech mixtures, and demonstrate tracking results obtained for a distant talking scenario of two speakers which includes room reverberation.

Kurzfassung

Die Ermittlung von mehreren Grundfrequenzverläufen in Sprachsignalen ist ein wichtiger Schritt zur Trennung mehrerer in einer einkanaligen Aufnahme vorhandener Sprecher. In dieser Doktorarbeit wird ein auf statistischen Modellen basierender Ansatz zur Schätzung und zeitlichen Verfolgung mehrerer Grundfrequenzverläufe vorgeschlagen. Dazu wird zunächst ein Wahrscheinlichkeitsmodell auf Sprachdaten trainiert, das die Eigenschaften des Kurzzeitspektrums von Sprache in Abhängigkeit der Grundfrequenz modelliert. Das resultierende Sprechermodell, das auf Gauss'schen Mischmodellen basiert, kann entweder Sprecher-unabhängig oder Sprecher-abhängig trainiert werden. Die Sprechermodelle werden mit Hilfe von Interaktionsmodellen kombiniert, um eine wahrscheinlichkeitsbasierte Beschreibung der beobachteten Mischung mehrerer Sprecher zu erhalten. Ein faktorielles Hidden Markov Modell wird verwendet um die Grundfrequenzverläufe mehrerer Sprecher über die Zeit zu verfolgen.

Der auf statistischen Modellen basierte Ansatz ist in der Lage, Informationen bezüglich Timbre sowie damit verbundene statistische Unsicherheiten der spektralen Struktur in das Modell einzubinden. Während die Verwendung von Sprecher-unabhängigen Modellen den Einsatz in Situationen erlaubt, in denen die Sprecher in einer Aufnahme unbekannt sind, besitzen Sprecher-abhängige Modelle den grossen Vorteil dass die geschätzten Grundfrequenzverläufe deren ursächlichen Sprechern zugeordnet werden können. Die Genauigkeit der vorgeschlagenen Methode wird auf zwei Datenbanken evaluiert und mit einem dem Stand der Technik entsprechenden Verfahren verglichen.

Es werden zwei grundlegende Probleme der vorgeschlagenen Methode behandelt: (i) Exakte Inferenz benötigt einen hohen Rechenaufwand, verursacht durch die Tatsache dass die Lösung durch Betrachten aller möglichen Zustandskombinationen der involvierten Sprecher erhalten wird. Es wird eine neue Methode für approximierte Inferenz vorgeschlagen die auf dem Verwerfen von unwahrscheinlichen Zustandskombinationen basiert. Die Methode basiert auf effizient berechenbaren oberen und unteren Schranken des Likelihoods von Grundfrequenz-Kombinationen. Die approximative Methode wird experimentell evaluiert hinsichtlich Zeitaufwand und Genauigkeit, und Resultate für die Verfolgung der Grundfrequenz von drei gleichzeitigen Sprechern werden gezeigt. (ii) Eine Abweichung der trainierten Modelle von im Testfall vorkommenden Gegebenheiten (z.B. ein anderer akustischer Kanal oder eine Abweichung der Lautstärke) verursacht eine Verschlechterung in der Genauigkeit der Grundfrequenz-Schätzung. Es ist daher nötig, die Sprechermodelle auf die neuen Gegebenheiten der Umgebung hin zu adaptieren, d.h. in Situationen wo lediglich die Mischung mehrerer Sprecher verfügbar ist. Wir schlagen eine Modifikation der Maximum Likelihood Linear Regression Methode vor, bei der die Modell-Parameter lediglich bezüglich der spektralen Einhüllenden modifiziert werden. Diese Einschränkung ist in Situationen von Vorteil, in denen nur wenige Daten zur Modell-Adaptierung vorhanden sind. Basierend darauf wird ein neuer EM-Algorithmus zur Adaptierung der Sprecher-Modelle von Sprachmischungen vorgeschlagen, und Re-

sultate für die daraus resultierende Grundfrequenz-Schätzung werden für ein Szenario gezeigt, in dem zwei Sprecher in einem Raum mit Nachhall aus grösserer Entfernung aufgenommen wurden.

Acknowledgments

I would like to thank all people who supported me and my work during the time of writing this thesis. First of all, I would like to thank my advisor Franz Pernkopf for giving me the opportunity to conduct this thesis, for his tireless help during this time and for always finding the time for helpful discussions. Moreover, I would like to thank my supervisor Gernot Kubin for his scientific guidance as well as his advice in many concerns, and for often helping me to see things from a different perspective.
I would also like to thank my colleagues from the Signal Processing and Speech Communications Laboratory for many interesting discussions, and for often making the time at office fun.
I would like to express my gratitude to Tuomas Virtanen for enabling my research visit at his laboratory. During this time, he shared many interesting ideas and provided valuable input for my work.
Last but not least, my deepest gratitude goes to Birgit, for her love, care and patience, and for helping me to carry on during this sometimes difficult time.

Contents

1 Introduction **1**
1.1 Motivation and Scope . . . 1
1.2 Related Work on Pitch Estimation . . . 3
1.2.1 Single-Pitch Estimation Algorithms . . . 4
1.2.2 Multi-Pitch Estimation Algorithms . . . 5
1.3 Scientific Contributions and Outline . . . 7

2 A Probabilistic Approach to Multi-Pitch Tracking **10**
2.1 Hidden Markov Models . . . 10
2.1.1 Model Training . . . 12
2.2 Factorial Hidden Markov Models . . . 13
2.3 The Speaker Interaction Model . . . 15
2.3.1 The Mixture-Maximization Model . . . 15
2.3.2 The Linear Interaction Model . . . 16
2.4 Tracking . . . 16
2.4.1 The Junction Tree Algorithm . . . 17
2.4.2 The Max-Sum Algorithm . . . 18
2.5 Experimental Setup . . . 20
2.5.1 Data . . . 20
2.5.2 Feature Extraction and Model Training . . . 22
2.5.3 Performance Measures . . . 22
2.6 Experimental Results . . . 24
2.6.1 Influence of Speaker Model and Interaction Model . . . 24
2.6.2 Comparison of Tracking Algorithms . . . 28

3 Methods for Fast Approximate Inference **30**
3.1 Fast Approximate Inference Based on Likelihood Pruning . . . 31
3.1.1 Notation and Definition of the R-best Set of a Function . . . 32
3.1.2 Computationally Efficient Bounds on the MIXMAX Observation Likelihood . . . 32
3.1.3 State Selection Strategies Based on Likelihood Bounds . . . 35
3.1.4 A Modified Junction Tree Algorithm for Sparse Likelihoods . . . 36
3.2 Experiments . . . 38
3.2.1 Mixtures of Two Speakers . . . 38
3.2.2 Mixtures of Three Speakers . . . 40

4 Model Adaptation **48**
4.1 The Basic Principle of MLLR . . . 49

4.2 Cepstrally Smoothed MLLR for Model Adaptation from Speech Mixtures 52
4.2.1 Parameter Transform with Implicit Cepstral Smoothing 52
4.2.2 A General EM Algorithm for MLLR-Based Model Adaptation from Speech Mixtures 53
4.2.3 An EM Algorithm for Cepstrally-Smoothed MLLR-Based Model Adaptation from Speech Mixtures Using the MIXMAX Interaction Model 56
4.2.4 An Approximate Sparse EM Algorithm for Model Adaptation . . 60
4.3 Experiments 60
4.3.1 Model Adaptation from Single-Speaker Data 61
4.3.2 Gain Adaptation 65
4.3.3 Self-Adaptation on Speech Mixtures Recorded in Reverberant Room 66

5 Conclusions 70

A Derivation of MIXMAX Likelihood Bounds 72
A.1 Derivation of the Upper Bound 72
A.2 Derivation of the Lower Bound 73

B Derivative of the Auxiliary Function in Section 4.2.3 74

C Derivation of the MIXMAX Expected Single-Speaker Log-Spectrum 76

Abbreviations

ACF autocorrelation function

AMDF average magnitude difference function

ASR automatic speech recognition

CASA computational auditory scene analysis

csMLLR cepstrally smoothed maximum likelihood linear regression

DCT discrete cosine transform

DDF double difference function

DFT discrete Fourier transform

EM expectation-maximization

FHMM factorial hidden Markov model

GMM Gaussian mixture model

HMM hidden Markov model

KL Kullback-Leibler

MDL minimum description length

MFCC Mel-frequency cepstral coefficient

ML maximum likelihood

MLLR maximum likelihood linear regression

MMSE minimum mean square error

NACF normalized autocorrelation function

NMF nonnegative matrix factorization

SD speaker dependent

SI speaker independent

SIR signal-to-interference ratio

SNR signal-to-noise ratio

1 Introduction

1.1 Motivation and Scope

The analysis of speech and audio signals is a vital area of research which has opened the door for many relevant applications, such as automatic speech recognition, speaker identification, speech enhancement, or the automatic transcription of music. While remarkable results have been achieved within the last few decades, successful operation of such applications heavily depends on the conditions under which the audio signal is gathered. In arbitrary real world conditions, the audio input consists of a multitude of sources, from which mostly a single specific source of interest needs to be extracted for further processing. Humans have the ability to perform this task in many difficult and fast varying conditions, while the same is very hard or even impossible to do on computers with existing state-of-the-art signal processing algorithms.
Methods developed in the field of computational auditory scene analysis (CASA) [1] attempt to mimic the relevant principles of human auditory processing, with the goal to detect, classify, extract and analyse all acoustic objects present in an arbitrary auditory scene. One key task of CASA is therefore the segregation of an audio recording into its individual acoustic sources, where ideally the signal of respectively one acoustic source is extracted while at the same time the contribution of all other sources is fully suppressed. Without simplifying assumptions or prior knowledge on the specific content of a given audio signal, one guiding principle of CASA for segregation of all acoustic sources is to exploit known regularities and patterns of natural sounds (*perceptual grouping cues*). In a time-frequency decomposition of the input signal, it is often the case that for a particular time-frequency atom, the dominant fraction of its energy stems from a single source. Perceptual grouping cues are used to identify those time-frequency atoms that originate from a common source. Some of these known patterns of natural sources are common onset/offset across frequency, joint amplitude modulation, harmonicity, and joint frequency co-modulation [2]. In principle, as long as the signal-to-interference ratio (SIR) of a source is not too low, the time-frequency atoms believed to belong to a single source can then be used to reconstruct the source signal.

Among all structural cues used in CASA, the emphasis of this thesis is on harmonicity. Harmonic sounds exhibit peaks of energy (*harmonic partials*) at frequencies that are an integer multiple of a fundamental frequency f_0. Important classes of harmonic sounds include voiced speech and musical tones,[1] but also acoustic events such as animal vocalizations or the siren of a bypassing police car are examples of harmonic sounds. In time domain, harmonic signals exhibit a periodic or quasi-periodic structure, where the period is the inverse of the fundamental frequency f_0. In contrast to periodic signals (e.g. a sine or a sawtooth wave) where exactly the same signal segment is repeated for every period, the period segments of quasi-periodic signals are 'similar', yet not identical. To illustrate this, Figure 1.1 shows a short segment of voiced speech, together with its frequency domain representation. In time domain, voiced speech exhibits a quasi-periodic structure, where the period corresponds to the inverse of the fundamental frequency f_0. In the frequency domain representation of the shown signal, the first few harmonic partials are clearly visible as prominent peaks at the fundamental frequency and integer multiples of it. While in this example the period remains more or less constant, note that the fundamental frequency of a sound – together with the frequencies of corresponding higher harmonic partials – typically changes over time.

For a human listener, a harmonic sound invokes the sensation of *pitch*, which is defined as the frequency of a pure sinusoid perceived with the same tone as the given signal segment under investigation [4]. Although the fundamental frequency correlates well with the perceived pitch of a sound signal, certain perceptual phenomena such as the *missing fundamental* find no explanatory support by f_0.[2] In that sense, pitch refers to a perceptual quality, while the fundamental frequency refers to a signal property. Despite these differences, we use the term pitch in this thesis as a synonym for f_0, since this is more consistent to previous literature (see Section 1.2).

Because harmonicity is an important characteristic of many natural sounds, one requirement of CASA is the precise estimation of the (time-varying) fundamental frequency of the harmonic sound sources in an auditory scene. Given the f_0 of a sound source, a considerable fraction of its energy can be identified and segregated from the mixture. This is true as well for speech, which consists of a considerable amount of voiced phones. Given the pitch estimate of a target speaker, ideally all voiced portions of the target speech can be recovered from a sound mixture. Some of the methods that use this principle for the segregation of voiced speech are [5–13].

This thesis is concerned with a probabilistic model-based approach for estimation and tracking of multiple pitch trajectories from a mixture of speakers – a problem commonly referred to as *multi-pitch tracking*. The method makes use of a probabilistic description of patterns obtained from real speech that captures the signal characteristics relevant for pitch estimation. This probabilistic description, which is referred to as *speaker model*, is trained on clean speech data prior to multi-pitch tracking. The method can be used either with speaker independent (SI) models, which are obtained from a large corpus of many different speakers, or with speaker dependent (SD) models that capture the dynamics of those speakers for which multi-pitch tracking is to be performed. While SI

[1] Note that stringed musical instruments such as the piano generate tones that are inharmonic, i.e. its partials deviate to some degree from the ideal frequency, which would be at an integer multiple of the fundamental frequency [3].

[2] The missing fundamental phenomenon refers to the observation that human listeners are able to perceive the pitch from a set of harmonic partials even if the fundamental frequency itself is missing, i.e. only some higher partials are present [2].

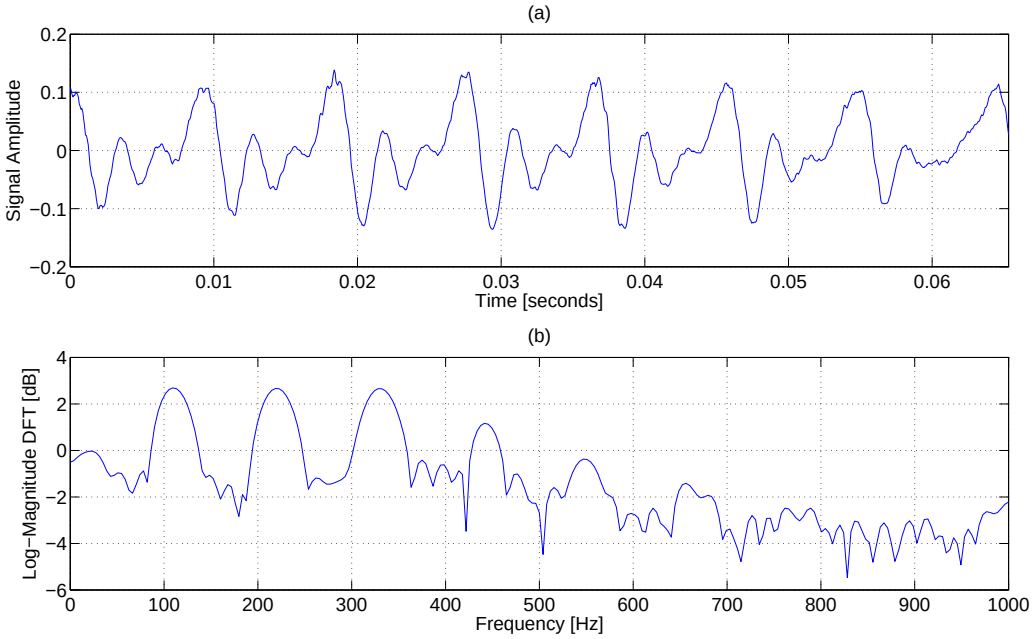

Figure 1.1: A short segment of voiced speech. This example was taken from a male speaker uttering the vowel 'u' from the word 'two'. (a) In time domain, voiced speech exhibits a quasi-periodic structure, where in this example seven period segments are shown. Neighbouring segments have roughly the same shape, and the period of each segment is about 9.1 ms. The inverse of the period corresponds to the fundamental frequency f_0. (b) In the frequency domain representation, the signal exhibits peaks (harmonics) at the fundamental frequency $f_0 \approx 110\,Hz$, as well as at integer multiples of it. Only the band from zero to 1 kHz is shown.

models allow an ad-hoc use in situations where the speakers in a recording are unknown, SD models have the great advantage that pitch trajectories can be assigned to their corresponding speakers. This issue is of great importance for CASA and the segregation of speech from a mixture. Without the knowledge which pitch estimate belongs to which speaker, consistent identification of relevant harmonic partials of a speaker over time is not possible [14]. Before we continue in Section 1.3 with a discussion on the proposed method and the scientific contributions of this thesis, we provide an overview of existing approaches for single and multiple pitch estimation in the sequel.

1.2 Related Work on Pitch Estimation

The emphasis of this overview is on algorithms designed for speech signals, although we give as well some references to algorithms designed for music transcription, where appropriate. In the following, we distinguish between existing approaches for single and multiple pitch estimation.

1.2.1 Single-Pitch Estimation Algorithms

Some of the most cited algorithms in literature for single-pitch estimation are RAPT [15], YIN [16] and the implementation in PRAAT [17, 18]. RAPT extracts a set of candidate peaks from the normalized autocorrelation function (NACF) and tracks the most likely pitch trajectory using the Viterbi algorithm. YIN proposes a series of steps to improve the autocorrelation method used for pitch estimation. Also PRAAT is based on two corrections in the computation of the short-time autocorrelation function (ACF), where then a Viterbi algorithm is used to find a plausible sequence from candidate peaks. Likewise, many other algorithms are based on extracting local maxima from short-time periodicity measures such as the the ACF, the average magnitude difference function (AMDF) [19], the cross-correlation of adjacent variable-length windows [20], the cepstrum [21], or modifications thereof (e.g. [22, 23]). It seems that none of these periodicity measures is well suited for arbitrary conditions, e.g. the cepstrum method is known to be robust to the influence of formants, but sensitive to additive noise, while the opposite is true for the ACF method [4]. The SIFT algorithm [24] performs a low-order inverse filtering on short-time segments to remove the influence of formants, and applies the ACF on the residual. In [25], a maximum likelihood (ML) estimator for an unknown periodic signal embedded in Gaussian noise is developed, which is then further modified for application to speech signals. The resulting likelihood of a fundamental frequency hypothesis is obtained by summing the values of the ACF at time-lags that are integer multiples of the period corresponding to the candidate fundamental frequency.

A different class of methods (see e.g. [26, 27]) is based on auditory models, which try to mimic relevant aspects of auditory processing of humans [28, 29]. A cochlear model decomposes the input speech signal into multiple bandpassed channels, where each channel is then (among other operations) subject to a short-term ACF, which results in a representation sometimes referred to as correlogram. Pitch is then usually estimated from a periodicity measure obtained by combining the ACF across channels. These approaches are believed to model more accurately how humans perceive pitch. Moreover, the robustness to noise or other influences can be increased by rejecting the ACF from channels which are believed to be unreliable. We return to this concept in the discussion of existing multi-pitch estimation algorithms.

Besides the question of how to estimate pitch from a voiced frame, related important aspects regard the detection of voiced frames (voiced/unvoiced detection) and how pitch estimates from neighbouring voiced frames are combined to obtain a consistent pitch trajectory (*tracking*). In principle, detection of voiced/unvoiced frames can be performed either in a prior stage independent of pitch estimation, e.g. [30], or jointly with pitch estimation based on the salience ('strength') of a detected periodic component. For tracking the pitch over consecutive frames, the incorporation of statistical models of pitch variability has become common. As an example, in [31] a graphical model framework for pitch tracking is proposed which allows to incorporate any periodicity measure such as the ACF, and parameters of the graphical model are learned from data.

The above enumeration of single-pitch estimation methods is by no means meant to be complete. For an (early) comparison of different single-pitch estimation methods, we refer the interested reader to [32] and [33]. An in-depth discussion of pitch estimation algorithms can be found in [34].

1.2.2 Multi-Pitch Estimation Algorithms

We broadly categorize existing approaches for multi-pitch estimation into four different groups.[3] Note that the following enumeration of methods is not meant to be complete, but should provide an overview on important directions of research. For a more comprehensive review, we refer the interested reader to [36]. An overview dedicated to methods for automatic music transcription can be found in [37].

Nonparametric Methods Methods of this category make use of short-time periodicity measures that are able to jointly detect periodicity of multiple superimposed sources. In [7,38], for example, the double difference function (DDF) is used as an extension of the AMDF to the case of two speakers. Mixed voiced speech is modelled as the sum of two periodic functions, which is filtered by two cascaded comb filters. If the lag parameter of each comb filter is set to the correct period of either signal, the output of the cascade is zero. Hence, multiple f_0 estimation is performed by searching the joint setting of lag parameters for which the average output magnitude is minimized. A different example is given in the work of Quatieri [39], where the 2-D Fourier transform is applied on small rectangular patches of the magnitude spectrogram. It is shown that the resulting representation contains relevant information not only on pitch, but also on the rate of pitch change for each of two speakers. Although listed here as an example of nonparametric methods, this computational paradigm has an interesting relation to processing principles found in the primary auditory cortex of mammals [40, 41].

Auditory Model-Based Methods Methods of this group work on an intermediate representation of the input signal obtained by computational auditory models, which are believed to replicate relevant aspects of human auditory processing in the inner ear [28,29]. Examples of these methods are given in [42–46] and references therein (see also Section 1.2.1 for related single-pitch estimation algorithms). Common to these methods are the following steps: (i) Decomposition of the input signal into multiple bandpassed channels, (ii) nonlinear processing (dynamic range compression and half-wave rectification followed by lowpass filtering) of each channel, (iii) extraction of periodicity measures in each channel, (iv) combining the periodicity information across all channels. In [43], a computationally efficient variant of this principle is presented which uses only two bandpass filters. In [45], an algorithm is proposed which not only performs multi-pitch estimation, but performs iterative pitch estimation and source segregation using the auditory representation. To illustrate the principles of auditory-based models in more detail, we summarize in the following the algorithm of Wu et al. [44], which is one of the current state-of-the-art algorithms for robust multi-pitch tracking of speech signals and is used in this thesis as a reference algorithm for experimental comparisons:
First, the input signal is decomposed into 128 subbands using a gammatone filterbank [47] with center frequencies uniformly spaced along logarithmic frequency. For high frequency channels (center frequency above 800 Hz), the amplitude envelope is extracted using the Teager energy operator [48] followed by a lowpass filter with cutoff frequency at 800 Hz. The reason for this step is to account for the *beating* phenomenon. The output of high frequency channels possibly contains multiple harmonic partials, such that the channel output is amplitude-modulated with modulation frequency equal to the difference of the

[3] This is a slight extension of the taxonomy presented in [35, Ch. 1, p. 2].

harmonic partial frequencies. As a result, the relevant information on harmonic relations is encoded in the amplitude envelope, which can be recovered using amplitude envelope extraction. See also [42] for an illustration of this principle. In a next step, the NACF is computed on frames for every channel with 16 ms frame-length and 10 ms step-size. Up to this stage, these processing blocks resemble the method for calculation of the correlogram [49]. Summation of the periodicity information across all channels would result in the 'summary autocorrelation'. In contrast, [44] employs a scheme to discard channels whose periodicity information is likely to be unreliable due to noise. For selecting a low frequency channel, the maximum peak at nonzero lag must exceed a certain threshold. For selecting a high frequency channel, the NACF obtained from the 16 ms frame must have a similar shape as the NACF computed on a long time frame of 30 ms. If a high frequency channel is selected, an additional peak selection routine is employed. A peak is only selected if a second peak at double time lag exists. Further, if the peak with smallest nonzero lag exceeds a threshold, all peaks at a multiple lag will not be selected. The final set of peaks selected from various channels serves as a basis to create a probabilistic representation of zero, one or two pitch periodicity values at each time frame. Semi-continuous pitch trajectories of at most two speakers are then obtained by using a hidden Markov model (HMM).

Parametric Methods Methods of this category define a parametric model of harmonic signals, and estimate its parameters from the observed signal by minimizing some cost function that quantifies the difference between model and observation. Examples for parametric methods can be found in [8, 35, 50–52]. The specific form of the parametric model defines the structural constraints imposed on the signal of interest, e.g. harmonicity can be expressed by forcing the frequencies of sinusoidal components to be at integer multiples of a fundamental frequency. Some methods make use of a probabilistic framework, where constraints can as well be expressed by adding prior distributions on model parameters. Parametric models can easily be extended to superpositions of multiple source signals with predefined structure, as well as the case of additional background noise. Parametric models have been proposed for different signal domains, e.g. some models are a parametric description of time-domain signals, while others model the short-time magnitude spectrum.

Christensen [35] provides an extensive study of various estimation methods for parametric time-domain models. In [8], the harmonic signal model of each source allows the pitch period to vary linearly within short time segments. In [51, 52], a parametric model for the joint time-frequency profile of a harmonic sound event observed in a wavelet power spectrum is proposed, where not only a smooth envelope of harmonic partials along frequency but also the power envelope of partials along time is modelled. In [53], a harmonic signal model is embedded in a probabilistic framework, where a prior distribution on the amplitude of harmonic partials enforces smoothness along frequency, and the fundamental frequency parameter evolves in time according to a Markov model. Multiple speakers are then modelled by a factorial hidden Markov model (FHMM), and prior distributions of the model are learned either in a generative or discriminative fashion.

A considerable amount of parametric methods exists as well for polyphonic music transcription, which provide interesting ideas relevant for multi-pitch tracking of speech, see e.g. [54–57].

Data-Driven Methods Methods of this group make use of prior knowledge on how typical spectral patterns of voiced speech or music look like. A representation of such patterns, e.g. in the form of a dictionary or a probability distribution, is obtained prior to pitch estimation from a labeled training corpus consisting of relevant signal examples. Note that in contrast to parametric methods, the shape of such patterns is not described by a signal model. Multi-pitch estimation is performed by matching harmonic patterns to an observed signal and returning the pitch labels associated with the most probable templates. In [58], harmonic templates are learned from voiced speech using nonnegative matrix factorization (NMF), where one template is obtained for each discrete pitch value. The resulting dictionary is then used for multi-pitch estimation using nonnegative deconvolution, where the contribution of each harmonic template to the observed short-time spectrum is evaluated. Only those templates with a contribution above some threshold are selected, from which the number of harmonic sources as well as their respective pitch is retrieved. A similar approach has been proposed in [59] for music signals, where a sparse variant of nonnegative deconvolution is used to detect the minimal number of contributing templates that still explains the observed spectrum. In [60], an adaptive dictionary for NMF-based music transcription is proposed. Because prior training of harmonic patterns is not practical or sometimes impossible, such patterns are learned directly from the polyphonic music signal for which transcription is to be performed. Although NMF is able to identify elementary patterns of a signal, there is no guarantee that these templates are harmonic. In order to ensure that the learned patterns are harmonic and their corresponding pitch is known, each pattern is constrained to be a linear combination of predefined templates that are harmonic within a small band and zero outside. This way, the resulting dictionary entries are indeed harmonic, and their spectral envelopes match the typical characteristics of sounds in the observed music signal.
As we describe further below, the method proposed in this thesis is also part of this category.[4]

1.3 Scientific Contributions and Outline

As already described in the general introduction, we consider a probabilistic model-based approach for multiple pitch tracking of speech. It makes use of the typical patterns of the single-speaker short-time spectrum for a given pitch value, which are learned a priori from clean speech data. In this sense, the proposed method is strongly related to other existing data-driven approaches, as described in 1.2.2. However, because the proposed method uses a probabilistic description of a speaker model, we refer to this approach as a 'probabilistic model-based' method.
Multi-pitch estimation is done by searching for combinations of patterns that best explain the observation – explicit periodicity measurements are absent. This is in strong contrast to methods that are based on short-time periodicity measures to identify possible pitch candidates. Such methods typically suffer from the influence of timbral variations such as formants, and hence try to remove them prior to extraction of periodicity measures. A probabilistic model-based approach, on the other hand, is capable to explicitly incorporate such timbral information and all associated uncertainties of spectral structure into the

[4] In general, we refer to the representation of relevant patterns as *speaker model*.

model. Moreover, whenever speaker-specific information is available in the model (i.e. speaker-specific characteristics of the spectrum), this information can be used to assign a pitch estimate to a speaker. To the best of our knowledge, the speaker assignment problem has so far been considered only by Zissman et al. [14]. In their 'Spectral Envelope Classification' method, two pitch estimates are assumed to be given per time frame which are then used to estimate single-source spectra. These spectra are compared to SD spectra from a pre-trained dictionary, and each pitch estimate is associated with the speaker that best matches the single-speaker estimate.
In Chapter 2, we introduce the general approach proposed for multi-pitch tracking. Moreover, we present experimental results obtained on artificial mixtures of two speakers, and demonstrate the benefit of SD models for the speaker assignment problem. Parts of the results presented in this chapter have been published earlier in [61, 62].

There are two main problems related with the proposed approach described in Chapter 2.

(i) The method has a high computational demand, mainly due to the fact that the exact solution can only be obtained by considering all possible pitch combinations across speakers – a problem that grows exponentially in the number of simultaneous speakers. This can be facilitated by approximate schemes that significantly reduce the computational requirements while maintaining the accuracy of the method.

(ii) Prior training of speaker models requires sufficient amounts of speech material – something that is unpractical if SD models are required. Even in cases where good speaker models have been trained, any mismatch between training and testing conditions such as different acoustic channel conditions or gain mismatches deteriorates the accuracy of multi-pitch tracking. Hence, it is desirable to have mechanisms that allow to adapt speaker models to novel environmental conditions during multi-pitch tracking, i.e. in situations where only a speech mixture is available.

In Chapter 3, we introduce a mechanism for approximate inference in the probabilistic framework, which is based on pruning unlikely combinations of pitch patterns. Therefore, upper and lower bounds for the probability of a combination of patterns are introduced. We demonstrate the accuracy and time requirements of the approximate method compared to exact (i.e. exhaustive) calculations, and present tracking results on mixtures of three simultaneous speakers. Parts of the results presented in this chapter have been published earlier in [63].
In Chapter 4, we deal with the problem of model adaptation. First, we review a method commonly used for model adaptation in speech recognition, called maximum likelihood linear regression (MLLR). Next, we propose a modification of MLLR where the adaptation of model parameters is constrained to modifications of the spectral envelope. This constraint is beneficial for cases where few adaptation data is available. Based on this we propose an expectation-maximization (EM) framework for adaptation of speaker models from speech mixtures. We demonstrate the capabilities of the algorithm on speech mixture recordings from reverberant environments, and discuss limitations of the current approach as well as directions for future research.

Partial results obtained during the work on this thesis have been published in [13, 61–70]. The scientific contributions of this thesis are summarized in the following:

- A probabilistic model-based approach for multi-pitch tracking.[5]
- An experimental comparison of SD and SI models with emphasis on the speaker assignment problem. In case of SD models, the algorithm is able to correctly associate pitch to the corresponding speaker.
- An approach for fast approximate inference in FHMMs in conjunction with a specific speaker interaction model.[6] Computationally efficient upper and lower bounds for likelihood pruning are introduced. This approach is not only relevant for multi-pitch tracking, but as well for other problems where a similar probabilistic framework is used (e.g. [71]).
- An approach for model adaptation using speech mixture data only. First, a modification of MLLR is proposed where the transform is implicitly constrained in cepstral domain. As a result, the proposed method (called cepstrally smoothed maximum likelihood linear regression (csMLLR)) is restricted to the adaptation of the smooth spectral envelope.
- An EM algorithm for model adaptation on mixtures of speech, based on either MLLR or csMLLR. The E-Step of the exact EM algorithm is intractable. It is approximated based on the pruning method proposed in Chapter 3.

[5] The idea for this approach is the result of a joint discussion with Michael Stark and Franz Pernkopf. All work on this approach has been conducted by the author alone.

[6] The MIXMAX interaction model, as described in Section 2.3.1.

2

A Probabilistic Approach to Multi-Pitch Tracking

The aim of this chapter is to introduce the general model-based approach for multi-pitch tracking, as well as to discuss several variants regarding the choice of speaker interaction models and algorithms for inference. We first introduce the modelling of the characteristics of single speakers using hidden Markov models (HMMs), and discuss some details of model training. Next, we describe the general approach for combining single speech HMMs into a factorial hidden Markov model (FHMM), which represents a generative model of speech mixtures.[7] Furthermore, we discuss several strategies for inference in FHMMs. Finally, we present and discuss experimental results.

2.1 Hidden Markov Models

The proposed method relies on a statistical model using standard signal representations such as the spectrogram. We may choose from several variants for the spectral representation of speech, e.g. one may either use the magnitude discrete Fourier transform (DFT) or the log-magnitude DFT, and choose between a spectral representation on the linear frequency axis or the logarithmic frequency axis. Sometimes we use the term *feature vector* to loosely refer to a spectral representation extracted from a short analysis window, which might be the result of any of the aforementioned transforms. Most of the work, however, is based on a log-magnitude spectrogram representation of speech, and in the following we assume this kind of representation, unless stated otherwise.
We model the characteristics of single speech using an HMM with a graphical representation as shown in Figure 2.1. Note that we make use of the factor graph representation of graphical models [72], where the functional dependency of random variables (circles) is made explicit by so called factor nodes (rectangles). The hidden random variables $x^{(t)}$ represent the pitch, where t indicates the time index. Similarly, gray nodes indicate the

[7] Throughout this thesis, the term *single speech* refers to the speech of a single speaker.

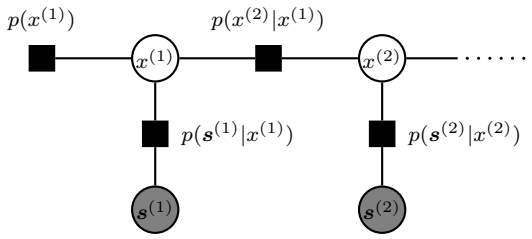

Figure 2.1: HMM represented as factor graph [72]. Factor nodes are depicted as shaded rectangles together with their functional description. Hidden variable nodes are shown as white circles, observed variable nodes as gray circles.

observation variables, where in this case vector $\boldsymbol{s}^{(t)}$ is the short-time log-magnitude DFT at time frame t. Each $x^{(t)}$ represents a discrete random variable with state space X and cardinality $|X|$. The rectangles connecting two nodes indicate a direct conditional dependency between random variables. Specifically, the dependency of hidden variables between two consecutive time instances is defined by the transition probability $p(x^{(t)}|x^{(t-1)})$. The dependency of the observed variable $\boldsymbol{s}^{(t)}$ on hidden variable $x^{(t)}$ is defined by the observation probability $p(\boldsymbol{s}^{(t)}|x^{(t)})$. Finally, the prior distribution of the hidden variables is denoted by $p(x^{(1)})$. Throughout this work, the hidden variable $x^{(t)}$ has $|X| = 170$ states, where state value '1' refers to 'no pitch' (i.e. unvoiced speech or silence), and state values '2'-'170' encode different pitch frequencies ranging from 80 to 500Hz. Specifically, the pitch value corresponding to state $x \in \{2, ..., 170\}$ is $f_0 = \frac{16000}{30+x}$. Similar to autocorrelation-based methods (e.g. [44]), this results in a nonuniform quantization of the pitch interval, where low pitch values have a more fine-grained resolution than high pitch values (see Figure 2.2).

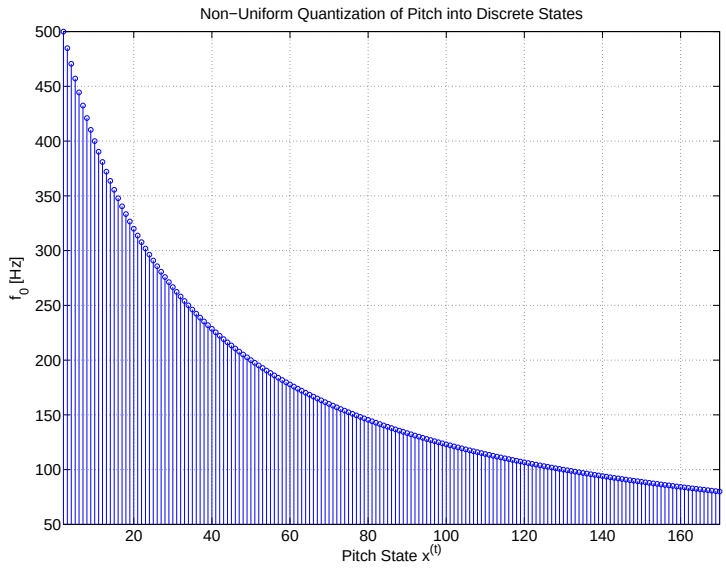

Figure 2.2: States of the discrete random variable $x^{(t)}$ correspond to a non-uniform quantization of the fundamental frequency in the range of $80 - 500$ Hz.

For the sake of brevity, we omit the explicit dependence of random variables on t, where appropriate. We use Gaussian mixture models (GMMs) to model the state-conditional observation according to

$$p(\boldsymbol{s}|x) = p(\boldsymbol{s}|\boldsymbol{\Theta}_x) = \sum_{m=1}^{M_x} \alpha_x^m \mathcal{N}\left(\boldsymbol{s}|\boldsymbol{\theta}_x^m\right), \tag{2.1}$$

where $M_x \geq 1$ is the number of mixture components, and α_x^m corresponds to the weight of each component $m \in \{1, \ldots, M_x\}$. These weights are constrained to be nonnegative, $\alpha_x^m \geq 0$, and $\sum_{m=1}^{M_x} \alpha_x^m = 1$. The corresponding GMM for pitch state x is fully specified by the parameter set $\boldsymbol{\Theta}_x = \{\alpha_x^m, \boldsymbol{\theta}_x^m\}_{m=1}^{M_x}$, where $\boldsymbol{\theta}_x^m = \{\boldsymbol{\mu}_x^m, \boldsymbol{\Sigma}_x^m\}$ is the mean and covariance of the m^{th} component. Furthermore, we assume diagonal covariance matrices.

2.1.1 Model Training

Having defined the overall structure of the probabilistic single-speaker model, one important remaining issue is the training of this model. In general, we perform training using a set of pitch-labeled speech utterances, where the pitch labels have been extracted using some other single-pitch estimation method. Details about the used databases and the single-pitch estimation method used for ground truth extraction are discussed in the experiments in Section 2.5. For now, let us simply assume that a set of labeled speech utterances is available. Training can be performed either in a speaker dependent (SD) or in a speaker independent (SI) fashion. SD models are trained using only speech utterances of a specific speaker, whereas SI models are trained using utterances from a large amount of different speakers. Obviously, the usage of appropriate SD models yields better results when applied to pitch estimation (as shown in Section 2.6). However, collecting a set of speech samples required to build a SD model might be difficult or even impossible in practice. In Chapter 4, we deal with this issue and propose a method for model adaptation.

Computing the log-magnitude spectrogram for each training utterance results in a set of N short-time log-spectra, $\mathcal{S} = \{\boldsymbol{s}^{(1)}, ..., \boldsymbol{s}^{(N)}\}$, together with corresponding reference pitch labels, $\{\bar{x}^{(1)}, ..., \bar{x}^{(N)}\}$, and $\boldsymbol{s}^{(n)} \in \mathbb{R}^D$. For each pitch state x, we can easily learn a GMM $p\left(\boldsymbol{s}|\boldsymbol{\Theta}_x\right)$ using the expectation-maximization (EM) algorithm [73]. Accordingly, we have to determine $|X| = 170$ GMMs. We use the minimum description length (MDL) criterion [74] to determine the number of components of each GMM automatically. Indeed, MDL is a method to find the optimal tradeoff between data-fit and model complexity. We denote the set of training samples for pitch state x as $\mathcal{S}_x = \{\boldsymbol{s}^{(k)} \in \mathcal{S} | \bar{x}^{(k)} = x\}$, and $|\mathcal{S}_x|$ is the size of the set. For each $\mathcal{S}_x$, we train a range of candidate GMMs with different number of components, and select the GMM which minimizes

$$\text{MDL}(\boldsymbol{\Theta}_x) = -\ln p\left(\mathcal{S}_x|\boldsymbol{\Theta}_x\right) + \frac{M_x(2D+1)}{2} \ln |\mathcal{S}_x|, \tag{2.2}$$

where the first term denotes the log-likelihood for the training data, i.e. $\ln p\left(\mathcal{S}_x|\boldsymbol{\Theta}_x\right) = \sum_{\boldsymbol{s} \in \mathcal{S}_x} \ln p(\boldsymbol{s}|\boldsymbol{\Theta}_x)$, and the second term relates to the complexity of the model with respect to the available data.[8]

[8] Throughout this work, the maximal number of components per GMM was restricted to 20.

The transition matrix of the HMM, $p(x^{(t)}|x^{(t-1)})$, is obtained by counting and normalizing the transitions of the reference pitch values from the single-speaker recordings in the training set. Additionally, we apply Laplace smoothing[9] on the transition matrix. The prior distribution $p(x^{(1)})$ is obtained likewise.

2.2 Factorial Hidden Markov Models

HMMs can be used to model the pitch-dependent speech characteristics of single speakers. Indeed, the resulting models can be used for single-pitch tracking, however they fail when processing a mixture of two or more speakers. In this section, we describe a general approach to combine multiple HMMs into an FHMM. While each HMM models the speech of a single speaker, the composed FHMM is capable to model a mixture of several speakers. Related approaches have been proposed for the task of noise robust automatic speech recognition (ASR) [75–77], and more recently for joint speech separation and speech recognition [71]. For simplicity and the ease of illustration, we deal throughout this chapter with the case of two simultaneously talking speakers. We present the general framework for K simultaneous speakers in Chapter 3.

FHMMs enable to track the states of multiple Markov processes evolving in parallel over time, where the available observations are considered as a joint effect of all single Markov processes. First proposed by Varga and Moore in the context of robust ASR [75], FHMMs got their name somewhat later [78], where the framework was introduced in a more general way together with novel mechanisms for inference and learning. The usage of FHMMs was first proposed for the task of multi-pitch tracking in [53] (see discussion of prior art in Section 1.2.2). By combining two single speaker HMMs, we obtain the FHMM shown in Figure 2.3, where each Markov chain models the pitch trajectory of one speaker. Note the additional subscript index used for the random variables $x_k^{(t)}$ and $\boldsymbol{s}_k^{(t)}$, that indicates the assignment of variables to the k^{th} Markov chain. Likewise, we denote by $\boldsymbol{\Theta}_{k,x_k}$ the GMM parameters of the k^{th} Markov chain for pitch state x_k. To keep the notation compact, we use braces to denote a set of variables from all Markov chains, e.g. $\{x_k^{(t)}\} := \{x_k^{(t)}\}_{k=1}^K$ is the set of all hidden pitch states at one time frame. At each time frame, the observation $\boldsymbol{y}^{(t)}$ is considered to be produced jointly by the two single-speaker emissions $\boldsymbol{s}_1^{(t)}$ and $\boldsymbol{s}_2^{(t)}$. This mixing process is modelled by an interaction model $p(\boldsymbol{y}^{(t)}|\boldsymbol{s}_1^{(t)}, \boldsymbol{s}_2^{(t)})$,[10] which is discussed in more detail in Section 2.3.

Given the sequence of observations $\boldsymbol{y}^{(t)}$, our goal is to infer the most likely sequence of hidden states $\{x_k^{(t)}\}$. Whenever possible, a common approach is to first marginalize over the unknown single-speaker features $\boldsymbol{s}_k^{(t)}$ [75]:

$$p(\boldsymbol{y}^{(t)}|x_1^{(t)}, x_2^{(t)}) = \int\int p(\boldsymbol{y}^{(t)}|\boldsymbol{s}_1^{(t)}, \boldsymbol{s}_2^{(t)}) p(\boldsymbol{s}_1^{(t)}|x_1^{(t)}) p(\boldsymbol{s}_2^{(t)}|x_2^{(t)}) d\boldsymbol{s}_1^{(t)} d\boldsymbol{s}_2^{(t)}. \tag{2.3}$$

[9] Laplace smoothing amounts to the initialization of each element of the transition matrix with count one, i.e. adding the prior information that each transition was observed once. This avoids transitions with zero probability.

[10] Although most of the interaction models are deterministic functions, we stick to the common notation and generally denote an interaction model as *probability density* $p(\boldsymbol{y}|\boldsymbol{s}_1, \boldsymbol{s}_2)$. Any deterministic function $\boldsymbol{y} = f(\boldsymbol{s}_1, \boldsymbol{s}_2)$ can still be expressed this way, i.e. $p(\boldsymbol{y}|\boldsymbol{s}_1, \boldsymbol{s}_2) = \delta\left(\boldsymbol{y} - f(\boldsymbol{s}_1, \boldsymbol{s}_2)\right)$, where $\delta(\cdot)$ is the Dirac-delta function.

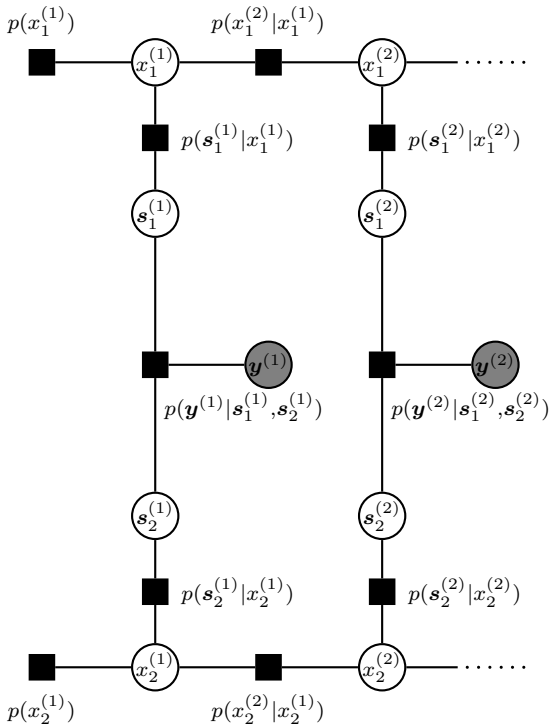

Figure 2.3: Two single-speaker HMMs are combined into an FHMM. Each Markov chain models the pitch trajectory of one speaker. At each time frame, the single-speaker emissions $\boldsymbol{s}_1^{(t)}$ and $\boldsymbol{s}_2^{(t)}$ jointly produce the observation $\boldsymbol{y}^{(t)}$. This process is modelled with the interaction model $p(\boldsymbol{y}^{(t)}|\boldsymbol{s}_1^{(t)}, \boldsymbol{s}_2^{(t)})$.

The resulting *pitch-conditional observation probability*, $p(\boldsymbol{y}^{(t)}|x_1^{(t)}, x_2^{(t)})$, has the advantage that it directly relates observations $\boldsymbol{y}^{(t)}$ and the hidden variables $x_k^{(t)}$ which we want to infer, while still retaining all probabilistic uncertainty induced by $p(\boldsymbol{s}_k^{(t)}|x_k^{(t)})$.

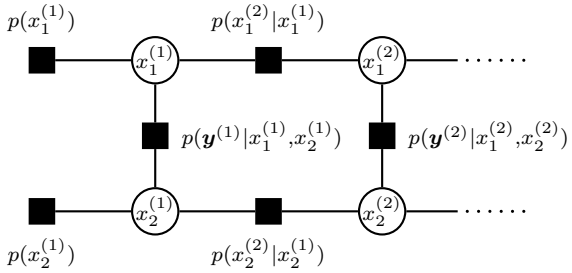

Figure 2.4: Marginalization over the nodes $\boldsymbol{s}_k^{(t)}$ (see Equation (2.3)) results in a more compact FHMM (cf. Figure 2.3), which directly relates observations $\boldsymbol{y}^{(t)}$ and hidden variables $x_k^{(t)}$. For simplicity, observation nodes have been absorbed into factor nodes.

Figure 2.4 shows the FHMM we obtain by marginalizing over all nodes $\boldsymbol{s}_k^{(t)}$. Denoting the whole sequence of variables by $\mathcal{X} = \bigcup_{t=1}^{T} \{x_k^{(t)}\}$ and $\mathcal{Y} = \bigcup_{t=1}^{T} \boldsymbol{y}^{(t)}$, the joint distribution

of all variables is given by

$$\begin{aligned} p(\mathcal{X},\mathcal{Y}) =& p(\mathcal{X})\, p(\mathcal{Y}|\mathcal{X}) \\ =& \prod_{k=1}^{2}\left[p\left(x_k^{(1)}\right)\prod_{t=2}^{T} p\left(x_k^{(t)}\middle|x_k^{(t-1)}\right)\right]\prod_{t=1}^{T} p\left(\boldsymbol{y}^{(t)}\middle|x_1^{(t)},x_2^{(t)}\right). \end{aligned} \tag{2.4}$$

The number of possible hidden states per time frame is $|X|^2$. As pointed out in [78], this could also be accomplished by an ordinary HMM. The main difference of FHMMs, however, is the constraint placed upon the transition structure. While an HMM with $|X|^2$ states would allow any $|X|^2 \times |X|^2$ transition matrix between hidden variables in consecutive time frames, the FHMM is restricted to two $|X| \times |X|$ transition matrices.

2.3 The Speaker Interaction Model

Various types of interaction models have been proposed in literature [71, 75, 76, 79–81]. Their applicability depends on the domain of features being used (e.g. log-magnitude domain, magnitude domain, Mel-frequency cepstral coefficients (MFCCs), etc ...), and they differ by the amount of approximations being applied. In the following, we focus on two types of interaction models, namely the MIXMAX [79] and the linear [81] interaction model. The MIXMAX model is suitable for features in log-magnitude domain, whereas the linear model assumes the superposition of features in magnitude domain. In contrast to other interaction models, both the MIXMAX and the linear interaction model share the beneficial property that the integral in (2.3) has a closed form solution.

2.3.1 The Mixture-Maximization Model

The mixture-maximization (MIXMAX) model was originally proposed in [75,79] for noise robust speech recognition. Since then, it has been used for speech enhancement [82], single-channel source separation [83,84], speaker identification [81] and joint single-channel speech separation and recognition [85]. The MIXMAX model is sometimes also referred to as *log-max approximation* or just *max approximation*. It is based on the insight that the log-magnitude DFT of two speakers can be approximated by the element-wise maximum of their respective single-speech log-magnitude DFTs. Specifically, for each time instant t,

$$\boldsymbol{y}^{(t)} \approx \max(\boldsymbol{s}_1^{(t)}, \boldsymbol{s}_2^{(t)}), \tag{2.5}$$

where $\boldsymbol{s}_k^{(t)}$ is the short-time log-magnitude DFT of speaker k. The underlying assumption of this approximation is based on the sparse nature of speech in time-frequency representations. With high probability, each particular time-frequency bin of a mixed-speech spectrogram is dominated by a single speaker. The same insight leads to the notion of binary masks in computational auditory scene analysis (CASA) [86] and single-channel source separation [83]. In [87], it is shown that (2.5) is a nonlinear MMSE estimator of the mixture log-spectrum assuming that the phase of both sources has uniform distribution. Given single-speaker GMMs and setting $p(\boldsymbol{y}^{(t)}|\boldsymbol{s}_1^{(t)}, \boldsymbol{s}_2^{(t)}) = \delta\left(\boldsymbol{y}^{(t)} - \max(\boldsymbol{s}_1^{(t)}, \boldsymbol{s}_2^{(t)})\right)$, we

obtain the pitch-conditional observation probability by marginalization over $\boldsymbol{s}_k^{(t)}$ (cf. Equation (2.3)) [75, 79]:

$$p(\boldsymbol{y}|x_1,x_2) = \sum_{m_1=1}^{M_{1,x_1}} \sum_{m_2=1}^{M_{2,x_2}} \alpha_{1,x_1}^{m_1} \alpha_{2,x_2}^{m_2} \prod_{d=1}^{D} \left\{ \mathcal{N}(y_d|\theta_{1,x_1}^{m_1,d}) \Phi(y_d|\theta_{2,x_2}^{m_2,d}) + \Phi(y_d|\theta_{1,x_1}^{m_1,d}) \mathcal{N}(y_d|\theta_{2,x_2}^{m_2,d}) \right\}. \tag{2.6}$$

Here, y_d gives the d^{th} element of $\boldsymbol{y}$, $\theta_{k,x_k}^{m_k,d}$ gives the d^{th} element of the corresponding mean and variance of the single-speaker model of speaker k, and $\Phi(y|\theta) := \int_{-\infty}^{y} \mathcal{N}(x|\theta)dx$ denotes the univariate cumulative normal distribution. The scalar M_{k,x_k} denotes the number of GMM components used for speaker k and pitch state x_k.

2.3.2 The Linear Interaction Model

As an alternative to the MIXMAX approach, the linear interaction model works directly in the magnitude domain [81]. In order to distinguish from features in log-magnitude domain, we highlight features defined in magnitude domain by an additional tilde. E.g., we denote the short-time magnitude DFT of speaker $k \in \{1,2\}$ at time t by $\tilde{\boldsymbol{s}}_k^{(t)}$. Under the linear interaction model, we approximate the short-time magnitude DFT of a speech mixture by

$$\tilde{\boldsymbol{y}}^{(t)} \approx \tilde{\boldsymbol{s}}_1^{(t)} + \tilde{\boldsymbol{s}}_2^{(t)}. \tag{2.7}$$

To obtain a pitch-conditional observation probability, we make use of the fact that the sum of two independent random variables is modelled by the convolution of their individual probability densities, i.e. $p(\tilde{\boldsymbol{y}}|x_1,x_2) = p(\tilde{\boldsymbol{s}}_1|x_1) * p(\tilde{\boldsymbol{s}}_2|x_2)$ [88], where $*$ denotes the convolution operator. Although we are now dealing with nonnegative data, we use again GMMs to model the distribution of features. The convolution of two Gaussian densities results again in a Gaussian, with mean and covariance matrix being the sum of the individual means and covariances, respectively. Hence, $\mathcal{N}(\tilde{\boldsymbol{y}}|\boldsymbol{\mu}_1+\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_1+\boldsymbol{\Sigma}_2) = \mathcal{N}(\tilde{\boldsymbol{s}}_1|\boldsymbol{\mu}_1,\boldsymbol{\Sigma}_1) * \mathcal{N}(\tilde{\boldsymbol{s}}_2|\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2)$. This easily extends to GMMs, as the convolution of two GMMs results in a mixture of all pairwise convolved component densities. Similar as with the MIXMAX model, we train single-speaker GMMs to model the magnitude spectrum, $p\left(\tilde{\boldsymbol{s}}_k|\boldsymbol{\Theta}_{k,x_k}\right)$. Then, we obtain the observation model as

$$p(\tilde{\boldsymbol{y}}|x_1,x_2) = \sum_{m_1=1}^{M_{1,x_1}} \sum_{m_2=1}^{M_{2,x_2}} \alpha_{1,x_1}^{m_1} \alpha_{2,x_2}^{m_2} \mathcal{N}\left(\tilde{\boldsymbol{y}}|\boldsymbol{\mu}_{1,x_1}^{m_1} + \boldsymbol{\mu}_{2,x_2}^{m_2}, \boldsymbol{\Sigma}_{1,x_1}^{m_1} + \boldsymbol{\Sigma}_{2,x_2}^{m_2}\right). \tag{2.8}$$

2.4 Tracking

At this point, we assume that we have available an FHMM that jointly models the speakers of a given speech mixture. Given the set of observations $\mathcal{Y}$, the task of tracking involves

searching the sequence of hidden states $\mathcal{X}^*$ that maximizes the conditional distribution:

$$\mathcal{X}^* = \arg\max_{\mathcal{X}} p(\mathcal{X}|\mathcal{Y}). \tag{2.9}$$

For HMMs, the exact solution to this problem is found by the Viterbi algorithm. Although an FHMM could be expressed by an equivalent HMM, more efficient tracking algorithms exploit the explicit factorization into individual Markov chains. Nevertheless, the computational complexity of exact inference in FHMMs increases exponentially with the number of hidden Markov chains.
For the related problem of finding the marginal density in FHMMs, several algorithms are derived in [78] using the framework of variational inference to obtain approximate solutions for the sake of reduced complexity. For a detailed discussion of inference in general graphical models, we refer the interested reader to [72, 89, 90].
In the following, we discuss both the exact junction tree algorithm as well as the loopy max-sum algorithm to solve (2.9). Moreover, we propose a message passing schedule for the max-sum algorithm to enable online tracking. In the experiments, we compare the performance of all presented inference methods in terms of accuracy.

2.4.1 The Junction Tree Algorithm

Exact inference on arbitrary graphical models is usually accomplished by first transforming that graphical model into a junction tree, where then belief propagation or related message passing algorithms are applied [89, 91]. For the problem of finding the marginal distribution

$$p(x_k^{(t)}|\mathcal{Y}) = \sum_{\mathcal{X}\setminus x_k^{(t)}} p(\mathcal{X}|\mathcal{Y}), \tag{2.10}$$

Ghahramani and Jordan [78] provide an exact algorithm for FHMMs, which can be seen as the natural extension of the forward-backward algorithm. Formally, this algorithm can be seen as the application of the sum-product algorithm on the junction tree representation of the FHMM. We present the equivalent formulation of this algorithm on the max-sum semiring[11] [92] in Algorithm 1, which provides an exact solution to (2.9). This algorithm was first proposed in [75] and can be seen as the natural extension of the Viterbi algorithm to FHMMs. The computational complexity (without considering the computation of $p(\boldsymbol{y}^{(t)}|x_1^{(t)}, x_2^{(t)})$) is $O(TK|X|^{K+1})$, where K is the number of Markov chains. For $K = 2$, as is assumed in this chapter, tracking is still feasible. In Chapter 3, we consider an approximate version of this algorithm which can be used if the observation likelihood in (2.3) is sparse, i.e. in cases where only a relative small number of likelihoods is nonzero.

[11] Informally, a semiring is an algebraic structure defined as a set K, together with two binary operations over elements of that set. Among other requirements, the binary operations must satisfy the distributive law. As shown in [92], the sum-product algorithm can be translated to a semiring involving other binary operations, such as the maximum operator. As a result, the algorithmic framework for the problem 'sum of products' can be translated to obtain an algorithm for the problem 'maximum of sums'.

Input: Set of observations $\mathcal{Y}$
Output: Optimal state sequence $\mathcal{X}^*$
Initialization: Compute state likelihoods $p\left(\boldsymbol{y}^{(t)}|x_1^{(t)}, x_2^{(t)}\right) \quad \forall\, t \in \{1, ..., T\}$
$\gamma^{(1)}\left(x_1^{(1)}, x_2^{(1)}\right) \leftarrow \ln p\left(x_1^{(1)}\right) + \ln p\left(x_2^{(1)}\right) + \ln p\left(\boldsymbol{y}^{(1)}|x_1^{(1)}, x_2^{(1)}\right)$
Forward recursion:
foreach $t \in \{2, \ldots, T\}$ **do**
$\gamma_1^{(t)}\left(x_1^{(t)}, x_2^{(t-1)}\right) \leftarrow \max_{x_1^{(t-1)}} \left[\ln p\left(x_1^{(t)}|x_1^{(t-1)}\right) + \gamma^{(t-1)}\left(x_1^{(t-1)}, x_2^{(t-1)}\right)\right]$
$\beta_1^{(t)}\left(x_1^{(t)}, x_2^{(t-1)}\right) \leftarrow \arg\max_{x_1^{(t-1)}} \left[\ln p\left(x_1^{(t)}|x_1^{(t-1)}\right) + \gamma^{(t-1)}\left(x_1^{(t-1)}, x_2^{(t-1)}\right)\right]$
$\gamma_2^{(t)}\left(x_1^{(t)}, x_2^{(t)}\right) \leftarrow \max_{x_2^{(t-1)}} \left[\ln p\left(x_2^{(t)}|x_2^{(t-1)}\right) + \gamma_1^{(t)}\left(x_1^{(t)}, x_2^{(t-1)}\right)\right]$
$\beta_2^{(t)}\left(x_1^{(t)}, x_2^{(t)}\right) \leftarrow \arg\max_{x_2^{(t-1)}} \left[\ln p\left(x_2^{(t)}|x_2^{(t-1)}\right) + \gamma_1^{(t)}\left(x_1^{(t)}, x_2^{(t-1)}\right)\right]$
$\gamma^{(t)}\left(x_1^{(t)}, x_2^{(t)}\right) \leftarrow \gamma_2^{(t)}\left(x_1^{(t)}, x_2^{(t)}\right) + \ln p\left(\boldsymbol{y}^{(t)}|x_1^{(t)}, x_2^{(t)}\right)$
end
$\left(x_1^{(T)^*}, x_2^{(T)^*}\right) \leftarrow \arg\max_{x_1^{(T)}, x_2^{(T)}} \left[\gamma^{(T)}\left(x_1^{(T)}, x_2^{(T)}\right)\right]$
Backtracking:
foreach $t \in \{T, \ldots, 2\}$ **do**
$x_2^{(t-1)^*} \leftarrow \beta_2^{(t)}\left(x_1^{(t)^*}, x_2^{(t)^*}\right)$
$x_1^{(t-1)^*} \leftarrow \beta_1^{(t)}\left(x_1^{(t)^*}, x_2^{(t-1)^*}\right)$
end

Algorithm 1: The junction tree algorithm for a two-chain FHMM on a max-sum semiring. This algorithm gives the exact solution to (2.9). The quantities $\gamma_1^{(t)}$, $\beta_1^{(t)}$, $\gamma_2^{(t)}$, $\beta_2^{(t)}$ and $\gamma^{(t)}$ represent two-dimensional tables, where each entry corresponds to a specific combination of hidden states. During the forward recursion, their entries are computed for each state combination. Note that for the special case of an HMM (i.e. FHMM with a single Markov chain), this algorithm is equivalent to the well known Viterbi algorithm.

2.4.2 The Max-Sum Algorithm

The max-sum algorithm is based on passing messages between connected nodes of a graph.[12] When applied on a graph with loops, as is the case of FHMMs, the solution is in general not guaranteed to converge and can only approximate the optimal solution. Among various types of graphs, factor graphs [72] have become a popular tool to depict the mechanisms of message passing. Consider again Figure 2.4, which shows an FHMM with two Markov chains. In factor graphs, the functional dependency of a variable node, for brevity called x, is made explicit by the rectangular factor nodes connected to it. Let $n(x)$ denote the set of factor nodes that are a neighbour of variable node x. Likewise, let $n(f)$ be the set of variable nodes that are a neighbour of factor node f. Each factor node represents a function $f(\{\hat{x}\})$ of its adjacent (i.e. neighbouring) variable nodes $n(f) = \{\hat{x}\}$. For the max-sum algorithm, each node sends to every neighbour a message $\mu_{a\to b}(x)$, where a denotes the sending node and b is the receiving node. Because each variable node in

[12] In the context of message passing algorithms, a *message* refers to a suitable representation of a probability density function.

Figure 2.4 represents a discrete random variable, a message sent between nodes is a probability mass function, which can be compactly described by a vector. A variable node x sends the following message to an adjacent factor node f:

$$\mu_{x\to f}(x) \;=\; \sum_{g\in n(x)\setminus f} \mu_{g\to x}(x), \tag{2.11}$$

i.e. the message sent to f is the sum of messages that x has received from all neighbouring factor nodes except f. A factor node f sends the following message to an adjacent variable node x:

$$\mu_{f\to x}(x) \;=\; \max_{\{\hat{x}\}\setminus x} \left(\ln f(\{\hat{x}\}) + \sum_{y\in\{\hat{x}\}\setminus x} \mu_{y\to f}(y) \right), \tag{2.12}$$

where $\{\hat{x}\} = n(f)$. We re-normalize each message μ such that its vector elements sum to one in exponential domain: $\sum_{i=1}^{|X|} e^{\mu(i)} = 1$. Although re-normalization does not influence the final results, it ensures the numerical stability of the message passing scheme [93]. We restrict each node to send a maximum of 15 messages per edge. Further, each node only re-sends a message to a neighbour if it is significantly different from the previously sent message in terms of the Kullback-Leibler-divergence (KL-divergence). For initialization, variable nodes send messages with all elements set to zero. After the last iteration, we obtain the maximum a posteriori configuration $p^*(x)$ of each variable node x as a function of its incoming messages:

$$p^*(x) = \max_{\mathcal{X}\setminus x} p(\mathcal{X}|\mathcal{Y}) = \sum_{g\in n(x)} \mu_{g\to x}(x). \tag{2.13}$$

Although the set of maxima, $x^* = \arg\max_x p^*(x)\ \forall x \in \mathcal{X}$, does not necessarily yield the global maximum in (2.9) as multiple global maxima might be present, a backtracking stage may lead to inconsistencies due to the loops in the factor graph. For this reason, we simply set the solution to the set of individual maxima x^*. Neglecting again the computation of $p(\boldsymbol{y}|x_1, x_2)$, the computational complexity of this approach is $O(TK|X|^K)$, i. e. the complexity of the max-sum algorithm is an order of magnitude lower than for the junction tree algorithm.

We consider two different scheduling strategies for max-sum message passing. First, we perform message passing on the FHMM using a complete speech mixture utterance at once. This is suitable for offline processing of recordings, and we refer to this method as *max-sum-batch*. Second, for online processing, we propose to partition the FHMM into overlapping segments of time frames, $\{\mathcal{T}_1, \mathcal{T}_2, ...\}$, and perform message passing on each individual segment exclusively. This concept is illustrated in Figure 2.5. Each segment consists of L time frames, and neighbouring segments overlap by $L - S$ frames, i.e. $\mathcal{T}_\tau = \{(\tau-1)S+1, (\tau-1)S+2, ..., (\tau-1)S+L\}$. In step τ, we restrict message passing to time frames $t \in \mathcal{T}_\tau$. All variable nodes in the set $\{x_1^{(t)}, x_2^{(t)} | t \in \mathcal{T}_{\tau-1} \cap \mathcal{T}_\tau\}$, as well as factor nodes connected to them, have already received messages in the previous step $\tau - 1$. Message passing is continued with those messages, thus enabling information flow from left to right. Similar to the concept of smoothing in e.g. Kalman filters, we wish to incorporate information from future observations at least H time frames ahead. Thus, when message passing has finished in step τ (i.e. each node has sent a maximum of

15 messages per edge), the maximum probability configuration of all variable nodes up to time frame $S(\tau-1)+L-H$ is evaluated, where H is the lower bound on the smoothing lag. Throughout the experiments, we set parameters to $L=8$, $S=3$ and $H=4$. In the experiments, we refer to this method as *max-sum-online*.

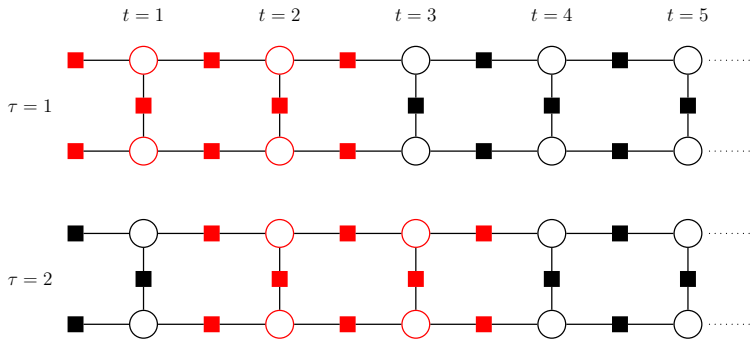

Figure 2.5: For online scheduling, message passing is performed on consecutive segments $\{...,\mathcal{T}_\tau,\mathcal{T}_{\tau+1},...\}$. In this example, each segment has $L=2$ time frames, and consecutive segments are shifted by $S=1$ time frames. Factor and variable nodes involved in message passing on segment $\mathcal{T}_1=\{1,2\}$ and $\mathcal{T}_2=\{2,3\}$ are shown in red, respectively. Nodes shown in black remain inactive. When all nodes have sent a maximum of 15 messages in step $\tau=1$, message passing is continued on segment $\mathcal{T}_2$. All nodes depending on time frames in $\mathcal{T}_1\cap\mathcal{T}_2=\{2\}$ continue with messages received in step $\tau=1$. With a supposed smoothing lag $H=1$ (see text for details), we evaluate after step $\tau=1$ the maximum probability configuration of variables at $t=1$.

2.5 Experimental Setup

2.5.1 Data

For all experimental evaluations throughout this thesis, we used material from the following two speech databases:

1. The GRID corpus [94] is an audio-visual sentence corpus, which consists of 34 speakers (16 female and 18 male) and 1000 sentences per speaker. All sentences consist of 6 words and have a simple predefined structure, such as *"place blue at F 9 now"*. The audio content of the corpus has been used for the 2006 speech separation challenge [95]. We used a subset of 500 utterances per speaker, and selected three female and three male speakers as test speakers (abbreviated as FE1, FE2, FE3 and MA1, MA2, MA3). The corresponding GRID label of test speakers is listed in Table 2.2. For each test speaker, 450 sentences were used to train SD GMMs, 40 sentences were reserved as development data, and 10 sentences were reserved as test data. The assignment of sentences to training, development and test set was done randomly. In addition to SD GMMs, SI GMMs were trained using speakers listed in Table 2.1, where again 450 sentences per speaker were used. The reference pitch trajectories

needed for training and evaluation were obtained using the RAPT method [15].[13]

2. The PTDB-TUG corpus [70] consists of 20 speakers (10 female and 10 male), with 236 utterances per speaker. Similar to other corpora such as Mocha-TIMIT [96] and Keele [97], it also includes laryngograph recordings of the spoken sentences which allows for a more reliable extraction of reference pitch estimates. PTDB-TUG is considerably larger than Mocha-TIMIT and Keele, and the text material consists of 2342 phonetically rich sentences taken from the TIMIT corpus [98].[14] Throughout the experiments, we selected all phonetically-compact (labeled as *sx*) and phonetically-diverse (labeled as *si*) sentences from PTDB-TUG, which results in 234 utterances per speaker. Again, we chose 3 female and 3 male speakers as test speakers (abbreviated as FE1, FE2, FE3 and MA1, MA2, MA3). The corresponding PTDB-TUG label of test speakers is listed in Table 2.4. For each test speaker, 200 sentences were used to train SD GMMs, 24 sentences were reserved as development data, and 10 sentences were reserved as test data. The assignment of sentences to training, development and test set was again random. SI GMMs were trained using speakers listed in Table 2.3, where again 200 sentences per speaker were used. We used the reference pitch trajectories provided with the corpus, which have been obtained from the highpass-filtered laryngograph recordings using the RAPT method.

Note that for both databases, no manual correction of the extracted reference pitch was performed, such that the ground truth may still contain small amounts of errors. However, as stated by Hess [34] and Klapuri [42], inaccuracies in reference pitch are not critical for the evaluation of multi-pitch estimators. Despite occasional errors, the reference pitch still serves to reflect how close a multi-pitch estimator resembles the performance that a good single-pitch estimator achieves on clean speech.
For each of the two databases, 135 test mixtures were created using test sentences of the 6 test speakers. Combining every test speaker with every other speaker results in 15 speaker pairs, and 9 test mixtures were created for each speaker pair. Mixing was performed by linear superposition (instantaneous mixture), and both source signals in a mixture have equal gain (in Chapter 4, we deal with the case of gain differences in speech mixtures). For test sentences from the PTDB-TUG database, long pauses before and after the actual utterance have been removed.

[13] An implementation of the RAPT algorithm is provided by the Entropic speech processing system (ESPS) labeled as "get_f0" method.

[14] The PDTB-TUG corpus and documentation can be downloaded from http://www.spsc.tugraz.at/tools.

	GRID Speaker Label												
FE	4	7	11	15	16	22	23	24	25	29	31	33	34
MA	5	6	9	10	12	13	14	17	19	26	27	28	32

Table 2.1: Labels of female (FE) and male (MA) speakers used for training SI models on the GRID corpus.

TEST SPEAKER	FE1	FE2	FE3	MA1	MA2	MA3
GRID LABEL	1	2	3	18	20	21

Table 2.2: Labels of test speakers from the GRID corpus.

	PTDB-TUG Speaker Label					
FE	F02	F04	F05	F06	F08	F10
MA	M01	M05	M06	M07	M08	M09

Table 2.3: Labels of female (FE) and male (MA) speakers used for training SI models on the PTDB-TUG corpus.

TEST SPEAKER	FE1	FE2	FE3	MA1	MA2	MA3
PTDB-TUG LABEL	F01	F07	F09	M03	M04	M10

Table 2.4: Labels of test speakers from the PTDB-TUG corpus.

2.5.2 Feature Extraction and Model Training

The observed features $\boldsymbol{y}^{(t)}$ or $\tilde{\boldsymbol{y}}^{(t)}$ of the proposed methods are based on the log-spectrogram or magnitude spectrogram of the speech mixture, respectively. Given an input signal at sampling rate $f_s = 16$kHz, we compute the spectrogram via the 1024 point DFT, using a Hamming window of length 32ms and step size of 10ms. Next, we obtain each observation vector $\tilde{\boldsymbol{y}}^{(t)} \in \mathbb{R}^{64}$ by taking the magnitude of spectral bins 2-65, which corresponds to a frequency range up to 1000Hz. This covers the most relevant frequency range, while keeping the number of feature dimensions at a moderate number. Likewise, we obtain $\boldsymbol{y}^{(t)} = \ln \tilde{\boldsymbol{y}}^{(t)}$.
For both types of features, SD and SI GMMs are trained on both databases, as described in Section 2.1.1.

2.5.3 Performance Measures

For every test instance, each method estimates two pitch trajectories, $\tilde{f}_0^{(1)}[t]$ and $\tilde{f}_0^{(2)}[t]$. In [44], an error measure was proposed to quantify the performance of double-pitch tracking algorithms. Let E_{ij} denote the percentage of time frames where i pitch points are misclassified as j pitch points, i.e. E_{12} means the percentage of frames with two pitch values estimated whereas only one pitch point is present. For each of the two reference

pitch trajectories, $f_0^1[t]$ and $f_0^2[t]$, the corresponding pitch frequency deviation is defined as

$$\Delta f^{(k)}[t] = \min_i \frac{\left|\tilde{f}_0^{(i)}[t] - f_0^{(k)}[t]\right|}{f_0^{(k)}[t]}, \tag{2.14}$$

i.e. at each time instance, the closest of the two estimated pitch points is assigned to a reference pitch trajectory. The gross detection error rate E_{Gross} is the percentage of time frames where the frequency deviation $\Delta f^{(k)}[t]$ is larger than 20% for one or both references $f_0^{(k)}$. The fine detection error $E_{Fine}^{(k)}$ is the average frequency deviation in percent at time frames where $\Delta f^{(k)}[t]$ is smaller than 20%. The overall error, E_{Total}, is defined as the sum of all error terms: $E_{Total} = E_{01} + E_{02} + E_{10} + E_{12} + E_{20} + E_{21} + E_{Gross} + E_{Fine}$, where $E_{Fine} = E_{Fine}^{(1)} + E_{Fine}^{(2)}$.
To evaluate the pitch-tracking performance in terms of successful speaker assignment, we propose a slightly modified error measure. First, each of the two estimated pitch trajectories is assigned to a ground truth trajectory, $f_0^{(1)}[t]$ or $f_0^{(2)}[t]$. From the two possible assignments, $(\tilde{f}_0^{(1)} \to f_0^{(1)}, \tilde{f}_0^{(2)} \to f_0^{(2)})$ or $(\tilde{f}_0^{(1)} \to f_0^{(2)}, \tilde{f}_0^{(2)} \to f_0^{(1)})$, the one is chosen for which the overall quadratic error is smallest. Note that this assignment is not done for each individual time frame, but for the global pitch trajectory. Next, we define the *speaker assigned pitch frequency deviation* as

$$\bar{\Delta} f^{(k)}[t] = \frac{\left|\tilde{f}_0^{(k)}[t] - f_0^{(k)}[t]\right|}{f_0^{(k)}[t]}, \tag{2.15}$$

where $f_0^{(k)}[t]$ denotes the reference chosen for $\tilde{f}_0^{(k)}[t]$. For each reference trajectory, we define the corresponding permutation error $\bar{E}_{Perm}^k[t]$ to be one at time frames where the voicing decision for both estimates is correct, but the pitch frequency deviation exceeds 20%, and $\tilde{f}_0^k[t]$ is within the 20% error bound of the other reference pitch. This indicates a permutation of pitch estimates due to incorrect speaker assignment. The overall permutation error rate $\bar{E}_{Perm}$ is the percentage of time frames where either $\bar{E}_{Perm}^1[t]$ or $\bar{E}_{Perm}^2[t]$ is one. Next, we define for each reference trajectory the corresponding gross error $\bar{E}_{Gross}^k[t]$ to be one at time frames where the voicing decision is correct, but the pitch frequency deviation exceeds 20% and no permutation error was detected. This indicates inaccurate pitch measurements independent of permutation errors. Again, the overall gross error rate $\bar{E}_{Gross}$ is the percentage of time frames where either $\bar{E}_{Gross}^{(1)}[t]$ or $\bar{E}_{Gross}^{(2)}[t]$ is one. This slightly different definition of the gross error rate ensures that voicing errors or permutation errors do not account for an additional increase in the gross error rate. The fine detection error $\bar{E}_{Fine}^{(k)}$ is the average speaker assigned frequency deviation in percent at time frames where $\bar{\Delta} f^{(k)}[t]$ is smaller than 20%. Finally, the overall error, $\bar{E}_{Total}$, is the sum of all error terms: $\bar{E}_{Total} = E_{01} + E_{02} + E_{10} + E_{12} + E_{20} + E_{21} + \bar{E}_{Gross} + \bar{E}_{Fine} + \bar{E}_{Perm}$, where $\bar{E}_{Fine} = \bar{E}_{Fine}^{(1)} + \bar{E}_{Fine}^{(2)}$.

2.6 Experimental Results

We evaluate the performance of the proposed algorithm on test mixtures of the GRID and PTDB-TUG database, as described in Section 2.5.1. We compare the performance to the correlogram based method of Wu, Wang and Brown [44], which we call WWB.[15] This method achieves a high accuracy for speech mixtures in difficult signal conditions. However, it does not facilitate a proper assignment of the estimated pitch points to their corresponding speakers. In contrast to this, the proposed algorithm is able to achieve a correct speaker assignment when using SD models. This enables to use the resulting pitch trajectories for single-channel source separation [13]. To allow a proper comparison of the proposed algorithm to WWB, we use an error measure that is invariant to correct speaker assignment, i.e. E_{Total} (see Section 2.5.3). Additionally, we use the slightly modified error measure $\bar{E}_{Total}$ to evaluate the performance of the proposed method in terms of successful speaker assignment.

2.6.1 Influence of Speaker Model and Interaction Model

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	E_{Gross}	E_{Fine}	E_{Total}
SD-MIXMAX	Mean	2.42	0.08	6.78	2.59	1.39	10.94	18.49	3.27	**45.95**
	Std	1.89	0.28	3.79	1.87	1.88	4.98	7.04	1.13	13.98
SI-MIXMAX	Mean	4.13	0.48	5.02	5.02	0.87	11.78	17.40	3.59	**48.29**
	Std	2.49	0.78	2.62	2.87	1.06	6.27	6.63	1.33	12.26
SD-LINEAR	Mean	3.43	0.18	5.50	3.81	0.75	10.33	16.36	3.36	**43.71**
	Std	2.27	0.49	3.67	2.20	1.15	5.39	6.49	1.06	12.35
SI-LINEAR	Mean	6.97	1.29	2.73	12.11	0.33	8.63	14.80	3.69	**50.55**
	Std	2.99	1.33	2.17	4.66	0.69	4.66	6.45	1.03	11.65
WWB	Mean	2.13	0.11	8.46	0.82	2.18	18.55	25.61	2.89	**60.76**
	Std	2.04	0.31	4.78	1.03	2.41	6.90	8.22	1.39	14.68

Table 2.5: Results on GRID database with junction tree algorithm. Performance is measured in terms of E_{Total}.

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	E_{Gross}	E_{Fine}	E_{Total}
SD-MIXMAX	Mean	2.61	0.10	3.35	3.55	0.44	4.31	9.08	3.84	**27.28**
	Std	2.38	0.30	2.81	3.73	0.75	2.78	5.40	1.18	9.81
SI-MIXMAX	Mean	3.07	0.08	2.01	4.02	0.20	4.59	7.51	3.81	**25.29**
	Std	2.21	0.24	1.70	3.23	0.30	3.14	4.21	1.22	8.37
SD-LINEAR	Mean	2.60	0.14	3.29	3.86	0.37	4.11	8.65	3.70	**26.74**
	Std	2.37	0.60	2.47	4.66	0.69	2.61	4.93	1.05	9.33
SI-LINEAR	Mean	5.70	0.16	1.31	5.45	0.12	3.87	6.75	4.01	**27.36**
	Std	3.26	0.33	1.34	4.19	0.25	2.83	3.95	1.18	7.97
WWB	Mean	4.28	0.27	3.42	0.54	0.97	9.69	12.05	3.27	**34.49**
	Std	2.75	0.50	2.81	0.60	1.28	4.44	6.15	1.11	9.52

Table 2.6: Results on PTDB-TUG database with junction tree algorithm. Performance is measured in terms of E_{Total}.

We may either choose the MIXMAX or the linear interaction model (see Section 2.3) to combine single-speaker HMMs. Further, speaker models can be trained either in a SD or SI fashion. In Table 2.5, we show the performance of all 4 variants on GRID test mixtures in terms of E_{Total}, and compare it to the performance achieved by WWB. Table 2.6 shows the same comparison for test mixtures from the PTDB-TUG database. In all cases, the junction tree algorithm was used for tracking. The aim here is to measure the accuracy of

[15] We used the C-implementation provided by the authors of [44] available at http://www.cse.ohio-state.edu/∼dwang/pnl/software.html .

all methods independent of correct speaker assignment. Both tables indicate that for all variants of the proposed algorithm, the main contributors to E_{Total} are E_{Gross}, E_{21}, and sometimes E_{12}. All variants have a comparable performance, and perform slightly better than WWB.

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	$\bar{E}_{Gross}$	$\bar{E}_{Fine}$	$\bar{E}_{Perm}$	$\bar{E}_{Total}$
SD-MIXMAX	Mean	2.42	0.08	6.78	2.59	1.39	10.94	2.13	2.88	0.55	**29.76**
	Std	1.89	0.28	3.79	1.87	1.88	4.98	3.97	0.81	1.20	9.33
SI-MIXMAX	Mean	4.13	0.48	5.02	5.02	0.87	11.78	3.71	4.27	10.00	**45.28**
	Std	2.49	0.78	2.62	2.87	1.06	6.27	2.86	2.48	7.85	11.39
SD-LINEAR	Mean	3.43	0.18	5.50	3.81	0.75	10.33	2.78	3.01	0.63	**30.40**
	Std	2.27	0.49	3.67	2.20	1.15	5.39	3.29	0.64	1.12	8.28
SI-LINEAR	Mean	6.97	1.29	2.73	12.11	0.33	8.63	6.61	4.62	9.39	**52.68**
	Std	2.99	1.33	2.17	4.66	0.69	4.66	3.57	2.49	7.80	11.73
WWB	Mean	2.13	0.11	8.46	0.82	2.18	18.55	1.83	3.95	13.60	**51.63**
	Std	2.04	0.31	4.78	1.03	2.41	6.90	2.42	4.28	9.25	12.34

Table 2.7: Results on GRID database with junction tree algorithm. Performance is measured in terms of $\bar{E}_{Total}$.

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	$\bar{E}_{Gross}$	$\bar{E}_{Fine}$	$\bar{E}_{Perm}$	$\bar{E}_{Total}$
SD-MIXMAX	Mean	2.61	0.10	3.35	3.55	0.44	4.31	2.05	3.78	0.47	**20.65**
	Std	2.38	0.30	2.81	3.73	0.75	2.78	3.13	1.35	0.93	7.56
SI-MIXMAX	Mean	3.07	0.08	2.01	4.02	0.20	4.59	2.29	5.01	5.00	**26.26**
	Std	2.21	0.24	1.70	3.23	0.30	3.14	2.87	2.54	3.64	8.05
SD-LINEAR	Mean	2.60	0.14	3.29	3.86	0.37	4.11	1.67	3.77	0.67	**20.50**
	Std	2.37	0.60	2.47	4.66	0.69	2.61	2.56	1.39	1.13	7.76
SI-LINEAR	Mean	5.70	0.16	1.31	5.45	0.12	3.87	2.41	5.31	5.52	**29.84**
	Std	3.26	0.33	1.34	4.19	0.25	2.83	2.34	2.44	4.11	7.84
WWB	Mean	4.28	0.27	3.42	0.54	0.97	9.69	0.99	5.22	6.70	**32.07**
	Std	2.75	0.50	2.81	0.60	1.28	4.44	1.36	4.52	5.07	7.78

Table 2.8: Results on PTDB-TUG database with junction tree algorithm. Performance is measured in terms of $\bar{E}_{Total}$.

To demonstrate the capability of the proposed methods in correctly assigning pitch trajectories to their corresponding speakers, we compare the performance on both databases using the proposed error measure $\bar{E}_{Total}$ in Tables 2.7 and 2.8. Using SD models, we achieve a significantly better performance in comparison to WWB for both databases. The most stringent advantage of SD models over SI models is their increased accuracy in assigning pitch trajectories to their corresponding speakers. This effect is directly measured by the permutation error $\bar{E}_{Perm}$.
To give an example, Figures 2.6 and 2.7 show tracking results for a test mixture from GRID and PTDB-TUG, respectively. Comparing the pitch estimates with the reference pitch shown on top of the speech mixture spectrogram, one can observe an increased speaker assignment accuracy when using SD models. Tracking results visualized in Figure 2.7 show that estimation and assignment still works well when the pitch trajectories of both speakers are located in the same frequency range crossing each other. In this situation, the assignment of pitch estimates to corresponding speakers based on time-continuity constraints is hard or even impossible – additional consideration of speaker-specific spectral characteristics, as provided by the SD model, is necessary.
Regarding the choice of the interaction model, the results show that the linear interaction model performs on par with the MIXMAX interaction model given SD models, but is slightly inferior when using SI models. This result is in contrast to the fact that the linear interaction model does *less* approximations than the MIXMAX interaction model. Possible reasons for the inferior performance of the linear interaction model might be

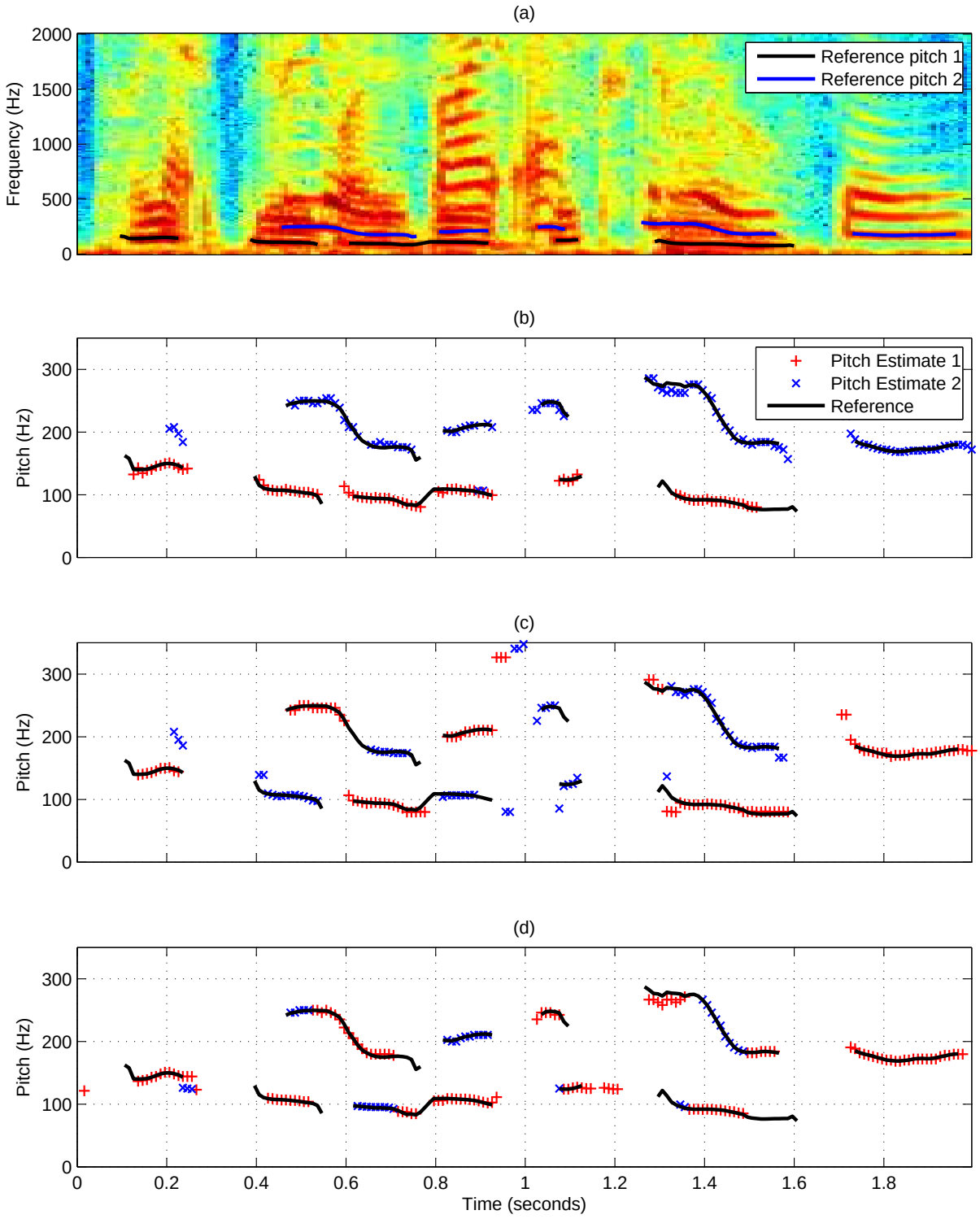

Figure 2.6: Tracking results on GRID test mixture of one male and one female speaker (utterance "pbbv6n" and "sbbf3s"). (a) Spectrogram of speech mixture, together with both reference pitch trajectories. (b)-(d) Estimated pitch trajectories using SD-MIXMAX, SI-MIXMAX and WWB, respectively. Reference pitch trajectories are shown as black lines. The overall error $\bar{E}_{Total}$ achieved by the three methods on this example is 24.57, 51.62 *and* 51.77, *respectively.*

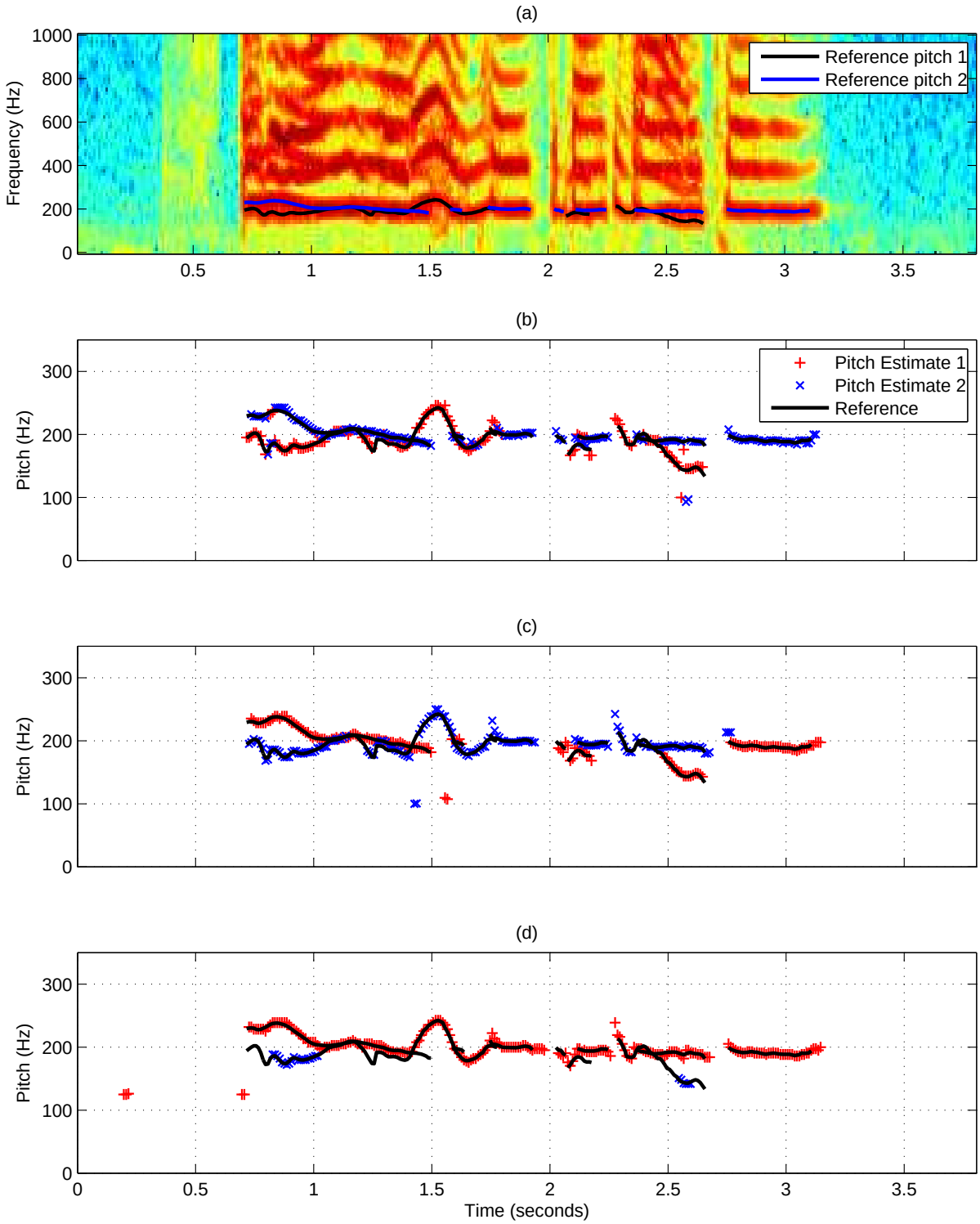

Figure 2.7: Tracking results on PTDB-TUG test mixture of two female speakers (utterance "Although always alone, we survive." and "In the long run, it pays to buy quality clothing."). (a) Spectrogram of speech mixture, together with both reference pitch trajectories. (b)-(d) Estimated pitch trajectories using SD-MIXMAX, SI-MIXMAX and WWB, respectively. Reference pitch trajectories are shown as black lines. The overall error $\bar{E}_{Total}$ achieved by the three methods on this example is 17.35, 20.01 *and* 34.52, *respectively.*

that the magnitude domain or/and the GMM-based speaker model for magnitude domain features is not well suited for modelling speech.
One might argue that the SI models we used so far are still *database dependent*. For this reason, we conducted experiments where we tested the performance of the proposed method on GRID test mixtures using the SI models trained on the PTDB-TUG database, and vice versa. Performance results for this experimental setup in terms of $\bar{E}_{Total}$ are shown in Tables 2.9 and 2.10, respectively. A comparison with Tables 2.7 and 2.8 shows that the performance using the linear interaction model is worse for both databases. The performance of the MIXMAX interaction model is inferior on PTDB-TUG database, but works equally well on the GRID corpus. This result might be explained by the fact that utterances of PTDB-TUG database have more diversity in terms of prosody and pitch variability than utterances from the GRID database. From this perspective, the SI model from PTDB-TUG generalizes better to the GRID database than the other way around.

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	$\bar{E}_{Gross}$	$\bar{E}_{Fine}$	$\bar{E}_{Perm}$	$\bar{E}_{Total}$
MIXMAX	Mean	4.31	0.09	5.02	2.61	1.44	16.79	4.14	4.80	9.46	**48.65**
	Std	3.33	0.28	2.97	2.27	1.78	7.34	3.46	2.77	8.27	12.67
LINEAR	Mean	5.65	1.86	2.26	15.44	0.15	8.40	9.78	7.54	14.42	**65.51**
	Std	2.61	2.31	1.77	8.44	0.39	4.83	4.97	2.80	10.29	14.55

Table 2.9: Results on GRID database with junction tree algorithm using SI models trained on the PTDB-TUG corpus. Performance is measured in terms of $\bar{E}_{Total}$.

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	$\bar{E}_{Gross}$	$\bar{E}_{Fine}$	$\bar{E}_{Perm}$	$\bar{E}_{Total}$
MIXMAX	Mean	7.79	0.35	3.43	7.76	0.68	4.65	3.92	4.46	5.04	**38.07**
	Std	4.20	0.68	2.50	3.51	0.90	2.72	3.11	2.15	4.03	8.81
LINEAR	Mean	8.08	0.69	2.11	7.38	0.36	4.58	6.46	4.76	4.99	**39.41**
	Std	2.93	0.65	1.54	3.70	0.50	2.80	3.84	2.68	3.93	9.68

Table 2.10: Results on PTDB-TUG with junction tree algorithm using SI models trained on the GRID corpus. Performance is measured in terms of $\bar{E}_{Total}$.

2.6.2 Comparison of Tracking Algorithms

For a comparison of the tracking methods discussed in Section 2.4, we evaluated the performance of the *max-sum-batch* and *max-sum-online* algorithm in terms of $\bar{E}_{Total}$ on both databases. Results of *max-sum-batch* are shown in Tables 2.11 and 2.12, results of *max-sum-online* are shown in Tables 2.13 and 2.14. In all cases, the performance is equal to the performance obtained with the junction tree algorithm when using SD models. When using SI models, the situation is different; the parameters of the FHMM are the same in both Markov chains, and the observation likelihood is symmetric, i.e. $p(\boldsymbol{y}|x_1, x_2) = p(\boldsymbol{y}|x_2, x_1)$. In this case, we observe that the performance of *max-sum-batch* and *max-sum-online* is significantly worse compared to results obtained with the (exact) junction tree algorithm. In an attempt to break the symmetry between both Markov chains, we added small amounts of Gaussian noise to the SI transitions, however this approach did not yield any improvements.

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	$\bar{E}_{Gross}$	$\bar{E}_{Fine}$	$\bar{E}_{Perm}$	$\bar{E}_{Total}$
SD-MIXMAX	Mean	2.42	0.08	6.78	2.66	1.40	10.91	2.13	2.89	0.56	**29.82**
	Std	1.89	0.28	3.79	1.90	1.88	4.99	3.97	0.81	1.21	9.35
SI-MIXMAX	Mean	2.86	1.01	6.80	21.45	1.72	7.14	3.93	4.55	15.17	**64.63**
	Std	1.83	1.06	2.99	8.00	1.54	3.64	2.60	2.02	7.68	8.94
SD-LINEAR	Mean	3.41	0.17	5.52	3.79	0.75	10.34	2.77	3.00	0.64	**30.40**
	Std	2.25	0.49	3.68	2.20	1.15	5.32	3.31	0.63	1.13	8.28
SI-LINEAR	Mean	4.82	2.40	4.74	20.40	1.00	6.02	7.73	5.15	14.68	**66.95**
	Std	2.15	1.78	2.68	6.76	1.22	2.89	3.41	2.21	7.34	8.69

Table 2.11: Results on GRID database with max-sum-batch. Performance is measured in terms of $\bar{E}_{Total}$.

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	$\bar{E}_{Gross}$	$\bar{E}_{Fine}$	$\bar{E}_{Perm}$	$\bar{E}_{Total}$
SD-MIXMAX	Mean	2.60	0.10	3.36	3.58	0.45	4.29	2.04	3.79	0.47	**20.67**
	Std	2.38	0.30	2.81	3.78	0.75	2.77	3.11	1.35	0.93	7.58
SI-MIXMAX	Mean	2.41	0.59	2.73	19.08	0.48	2.09	2.51	4.93	7.55	**42.37**
	Std	1.53	0.93	1.78	4.86	0.50	1.22	2.55	1.80	4.33	7.11
SD-LINEAR	Mean	2.60	0.15	3.31	3.88	0.37	4.10	1.67	3.77	0.68	**20.51**
	Std	2.36	0.60	2.47	4.66	0.69	2.61	2.55	1.38	1.13	7.76
SI-LINEAR	Mean	4.75	0.69	2.01	12.79	0.38	2.18	2.64	5.35	7.39	**38.19**
	Std	2.73	0.75	1.48	5.27	0.44	1.51	2.03	1.84	4.46	7.50

Table 2.12: Results on PTDB-TUG database with max-sum-batch. Performance is measured in terms of $\bar{E}_{Total}$.

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	$\bar{E}_{Gross}$	$\bar{E}_{Fine}$	$\bar{E}_{Perm}$	$\bar{E}_{Total}$
SD-MIXMAX	Mean	2.40	0.08	7.33	2.59	1.40	10.96	1.96	2.88	0.52	**30.13**
	Std	1.90	0.28	3.52	1.88	1.89	4.94	3.48	0.81	1.15	9.11
SI-MIXMAX	Mean	0.55	2.02	24.16	20.16	6.90	0.41	2.99	4.63	20.70	**82.52**
	Std	0.69	1.50	6.15	5.74	3.98	1.00	1.74	2.31	10.88	9.02
SD-LINEAR	Mean	3.32	0.18	5.99	3.77	0.75	10.34	2.57	3.00	0.64	**30.56**
	Std	2.27	0.49	3.46	2.17	1.15	5.40	2.88	0.63	1.13	8.17
SI-LINEAR	Mean	1.04	3.87	18.26	25.11	4.44	0.86	5.99	5.03	21.45	**86.03**
	Std	1.02	2.17	6.00	6.55	2.82	2.38	2.42	2.24	11.35	9.29

Table 2.13: Results on GRID database with max-sum-online. Performance is measured in terms of $\bar{E}_{Total}$.

		E_{01}	E_{02}	E_{10}	E_{12}	E_{20}	E_{21}	$\bar{E}_{Gross}$	$\bar{E}_{Fine}$	$\bar{E}_{Perm}$	$\bar{E}_{Total}$
SD-MIXMAX	Mean	2.60	0.11	3.36	3.57	0.45	4.29	2.03	3.79	0.46	**20.66**
	Std	2.37	0.31	2.81	3.75	0.76	2.75	3.10	1.34	0.93	7.57
SI-MIXMAX	Mean	0.17	1.28	18.11	15.34	2.64	0.07	2.08	5.15	10.28	**55.12**
	Std	0.23	0.99	4.10	3.54	1.93	0.15	2.40	2.00	6.20	6.70
SD-LINEAR	Mean	2.59	0.15	3.32	3.86	0.36	4.11	1.67	3.77	0.68	**20.52**
	Std	2.36	0.61	2.47	4.66	0.69	2.62	2.55	1.38	1.13	7.72
SI-LINEAR	Mean	0.42	2.20	16.82	16.49	2.12	0.12	2.06	5.56	10.87	**56.67**
	Std	0.40	1.43	4.32	3.83	1.60	0.41	1.84	2.09	6.57	6.49

Table 2.14: Results on PTDB-TUG database with max-sum-online. Performance is measured in terms of $\bar{E}_{Total}$.

3 Methods for Fast Approximate Inference

One main drawback of the proposed model-based approach for multi-pitch tracking is its high computational demand. This is due to the fact that for exact inference all possible combinations of pitch states across speakers need to be considered. While calculations for the case of two simultaneous speakers are still feasible, exact inference for a mixture of three or more speakers quickly becomes intractable. In general, the computational complexity of inference in FHMMs scales exponentially with the number of Markov chains. Existing methods for approximate inference in FHMMs and related models are usually based on variational approaches, Monte Carlo sampling, or likelihood pruning. In the factorial max vector quantization (MAXVQ) framework of Roweis [83], the most likely combination of hidden states is inferred by using a branch-and-bound technique. It uses an upper bound on the likelihood achieved by fixing the state of speaker k, regardless of the state configuration of all other interfering speakers. This bound is cheap to compute and used to quickly exclude state combinations which cannot be optimal. In general, branch-and-bound methods are guaranteed to find the global optimum of an optimization problem (in this case the best state configuration), but have no theoretical guarantees on the runtime – in worst case it performs brute force search. Nevertheless, applied to MAXVQ the method efficiently finds the optimal configuration in reasonable time [83]. Unfortunately, this MAXVQ likelihood formulation is different to the MIXMAX likelihood used in this work (cf. Section 2.3.1). Therefore, direct application of this method for the proposed multi-pitch tracking framework is not possible. In the work of Rennie et al. [71, 99], an approach for approximate inference in FHMMs using variational inference in conjunction with loopy belief propagation is proposed. The posterior over state combinations is approximated using a set of variational distributions, which factorize across all simultaneous speakers. This way, the messages sent by loopy belief propagation across Markov chains can be approximated without the need to consider combinations of speaker states, such that the complexity of inference is linear in the number of speakers. However, each message passed between Markov chains needs itself be computed using an iterative scheme. Algorithms for approximate inference in FHMMs based on variational inference

as well as Monte Carlo sampling have been introduced as well in [78].
In this chapter, we introduce an approach for fast approximate inference based on pruning of unlikely pitch combinations. This method has been developed specifically for the MIXMAX interaction model, and we do not consider the linear interaction model. Our approach is based on novel upper and lower bounds on the state-conditional observation likelihood, which are then used to efficiently retrieve a set of *probable* state configurations. The cardinality R of the set can be chosen to control the tradeoff between accuracy and computation time. Note that in contrast to the branch-and-bound approach of Roweis [83], we do not attempt to retrieve the single globally best or the set of size R of globally best state configurations, such that the run-time for state pruning remains predictable (no worst case brute force search). As we show in the experiments, the multi-pitch tracking performance of the pruning scheme is comparable to the results obtained with exact inference for relatively small values of R. Moreover, we present tracking results for instantaneous mixtures of three speakers. This is enabled in reasonable time by likelihood pruning.

3.1 Fast Approximate Inference Based on Likelihood Pruning

For the sake of generality, let us consider the problem of tracking the pitch of K simultaneous speakers. In principle, we can extend the approach introduced in Chapter 2 to the case of K speakers in a straightforward manner. Again, the pitch states $x_k^{(t)}$ of each speaker k are modelled by one Markov chain, and at each time frame t, the pitch-conditional observation probability under the MIXMAX interaction model is given as [71]:

$$p(\boldsymbol{y}^{(t)}|\{x_k\}) = \sum_{\{m_k\}} \left(\prod_k \alpha_{k,x_k}^{m_k} \right) \prod_{d=1}^{D} \sum_k \mathcal{N}(y_d^{(t)}|\theta_{k,x_k}^{m_k,d}) \prod_{j \neq k} \Phi(y_d^{(t)}|\theta_{j,x_j}^{m_j,d}). \tag{3.1}$$

As in Chapter 2, we use the shorthand $\{x_k\}$ to denote $\{x_k\}_{k=1}^K$, and $\sum_{\{m_k\}}$ refers to the nested sum $\sum_{m_1=1}^{M_{1,x_1}} \cdots \sum_{m_K=1}^{M_{K,x_K}}$. Exact inference of all K hidden pitch trajectories given a sequence of T observations requires the explicit calculation of $T|X|^K$ likelihood values, and application of the Viterbi algorithm with computational complexity $O(TK|X|^{K+1})$. However, here we make use of the fact that a large fraction of likelihood values is insignificantly small and has more or less no influence on the final tracking result. In the following, we elaborate on this idea and propose a method that is able to detect the state combinations of most of the significant likelihood values in a fast and reliable way. For tracking, we use exact likelihood computation exclusively for the resulting set of promising state combinations, which is a small fraction of all $|X|^K$ likelihoods. Finally, a modified Viterbi algorithm that is able to operate on sparse lists of likelihoods is applied to determine the pitch trajectory of each speaker.

3.1.1 Notation and Definition of the R-best Set of a Function

Consider a function $f(\cdot)$ defined over a finite domain $\mathcal{D}$. For any $\{x_r\}_{r=1}^R = \mathcal{S} \subseteq \mathcal{D}$, we use the short-hand $f(\mathcal{S})$ to denote the set $\{f(x_r)\}_{r=1}^R$.
We define the *R-best set* of $f(\cdot)$, denoted by $\mathcal{S}_R^f$, as the set $\{x_r\}_{r=1}^R, x_r \in \mathcal{D}$, for which it holds that

$$f(x_r) \geq f(y) \qquad \forall x_r \in \mathcal{S}_R^f, \forall y \in \mathcal{D} \setminus \mathcal{S}_R^f. \tag{3.2}$$

3.1.2 Computationally Efficient Bounds on the MIXMAX Observation Likelihood

First of all, let us rewrite Equation (3.1) as

$$p(\boldsymbol{y}|\{x_k\}) = \sum_{\{m_k\}} \left(\prod_k \alpha_{k,x_k}^{m_k} \right) L(\{x_k\}, \{m_k\}), \tag{3.3}$$

where for the sake of brevity we now omit the dependency on time index t and introduced the short-hand symbol L for the likelihood as a function of pitch and component states:

$$\begin{aligned} L(\{x_k\}, \{m_k\}) &= \prod_{d=1}^{D} \sum_k \mathcal{N}(y_d|\theta_{k,x_k}^{m_k,d}) \prod_{j \neq k} \Phi(y_d|\theta_{j,x_j}^{m_j,d}) \\ &= \prod_{d=1}^{D} \sum_k \mathcal{N}_{k,x_k}^{m_k,d} \prod_{j \neq k} \Phi_{j,x_j}^{m_j,d}. \end{aligned} \tag{3.4}$$

Note that we introduced respectively short-hand symbols for the normal density and cumulative normal distribution that omit explicit dependency on observation $\boldsymbol{y}$. As we show in Appendix A, the following upper and lower bound holds for $\ln L(\cdot, \cdot)$:

$$\begin{aligned} \ln L(\{x_k\}, \{m_k\}) &\leq \sum_k \sum_d \ln \left\{ \mathcal{N}_{k,x_k}^{m_k,d} + \Phi_{k,x_k}^{m_k,d} \right\} \\ &= \sum_k u_k(x_k, m_k) \\ &= \mathrm{UB}(\{x_k\}, \{m_k\}), \end{aligned} \tag{3.5}$$

$$\begin{aligned} \ln L(\{x_k\}, \{m_k\}) &\geq \max_k \left\{ \sum_d \ln \mathcal{N}_{k,x_k}^{m_k,d} + \sum_{j \neq k} \sum_d \ln \Phi_{j,x_j}^{m_j,d} \right\} \\ &= \max_k \lambda_k(\{x_k\}, \{m_k\}) \\ &= \mathrm{LB}(\{x_k\}, \{m_k\}). \end{aligned} \tag{3.6}$$

To give an example of the typical behaviour of the proposed bounds, consider Figure 3.1. For a single time frame from a mixture of two speakers, the log-likelihood $\mathrm{LL}(\{x_k\}, \{m_k\}) = \ln L(\{x_k\}, \{m_k\})$ was computed over the domain of all state combi-

nations (i.e. $x_1 \times m_1 \times x_2 \times m_2$), and all resulting values were sorted in decreasing order (blue line). For each state combination, the red and green curve shows the corresponding upper and lower bound, respectively. This example immediately shows that both bounds can be very conservative for some cases, while they can be tight for other cases (especially the lower bound). The gap between the lower or upper bound and the true likelihood seems to vary dramatically from one case to the other. Nevertheless, we show in the following that both bounds are still useful to detect a significant amount of the best state combinations. Specifically, the selection of promising state combinations is based on the R-best set of both bounds.

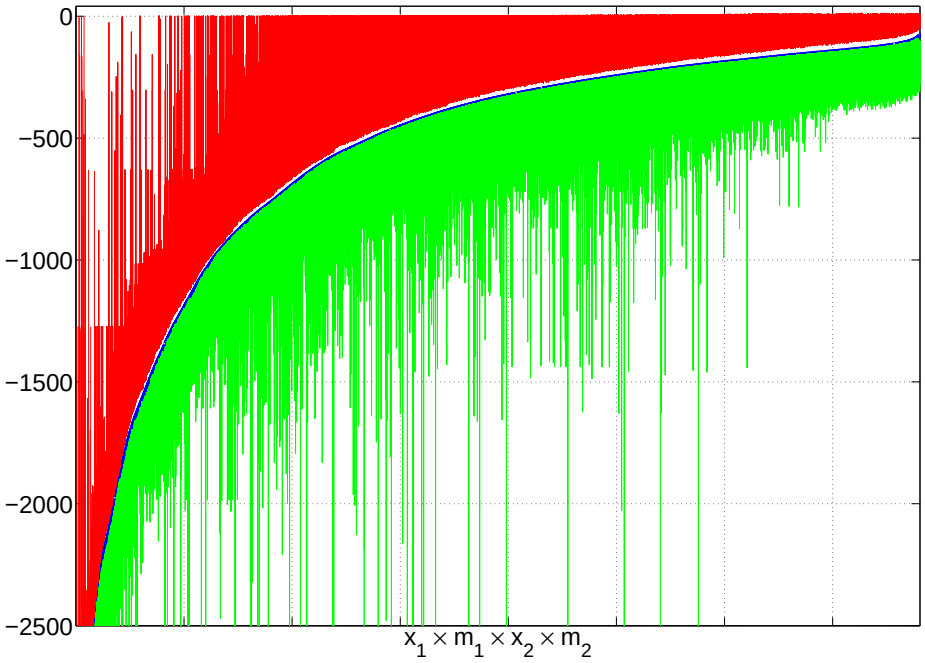

Figure 3.1: Log-likelihood values $\ln L(\{x_k\}, \{m_k\})$ *and their corresponding upper and lower bounds. The log-likelihood was computed for all combinations of pitch states and GMM components for a single time frame from a mixture of two speakers, and values were sorted in descending order (blue line). The red and green line show the corresponding upper and lower bounds, respectively. The number of state and component combinations along the x-axis is in the order of* 10^6.

Let the scalar Λ be the number of all possible combinations of pitch states and GMM components across speakers:

$$\Lambda = \sum_{\{x_k\}} \prod_k M_{k,x_k}, \tag{3.7}$$

where we earlier defined M_{k,x_k} as the number of GMM components used for speaker k and pitch state x_k. Both the upper and the lower bound have the advantage that their R-best set, $\mathcal{S}_R^{\text{UB}}$ and $\mathcal{S}_R^{\text{LB}}$ respectively, can be calculated in a fast and efficient way for $R << \Lambda$. To see this, consider the upper bound (3.5) for the case of two speakers. To simplify the notation, we replace the index pair (x_k, m_k) by a linear index l_k, i.e.

each value of l_k uniquely maps to a value of the pair (x_k, m_k), and vice versa. A brute-force approach to obtain the R-best set of the upper bound would explicitly compute $\mathrm{UB}(l_1, l_2) = u_1(l_1) + u_2(l_2)$ for all combinations of l_1 and l_2, and then retrieve the R-largest elements. However, we can exploit the fact that the upper bound is decomposable (i.e. each term $u_k(x_k, m_k)$ depends on one speaker only), from which it follows that $\mathcal{S}_R^{\mathrm{UB}}$ is guaranteed to be a subset of $\mathcal{S}_R^{u_1} \times \mathcal{S}_R^{u_2}$. Thus, an equivalent yet more efficient way to obtain the R-best set of the upper bound is to determine the R-best set of $u_k(l_k)$, $\mathcal{S}_R^{u_k}$, for $k = 1, 2$, and retrieve the R-best set by explicit computation of all values of $\mathrm{UB}(\mathcal{S}_R^{u_1} \times \mathcal{S}_R^{u_2})$ (see Figure 3.2). We refer to this method as *exact* R-best set retrieval.
We can further modify this method to approximate the R-best set of the upper bound. It is easy to see that with high probability $\mathcal{S}_R^{\mathrm{UB}}$ is in $\mathcal{S}_{\bar{R}}^{u_1} \times \mathcal{S}_{\bar{R}}^{u_2}$ for some sufficiently large $\bar{R} < R$. During preliminary experiments, we found that reducing the R-best set to $\bar{R} = 3\sqrt{R}$ still works well. We argue that limiting the search to a reduced set of most probable indices is a reasonable approximation with the benefit of heavily reduced computational efforts. This concept is also illustrated in Figure 3.2. We refer to this method as *approximate* R-best set retrieval.

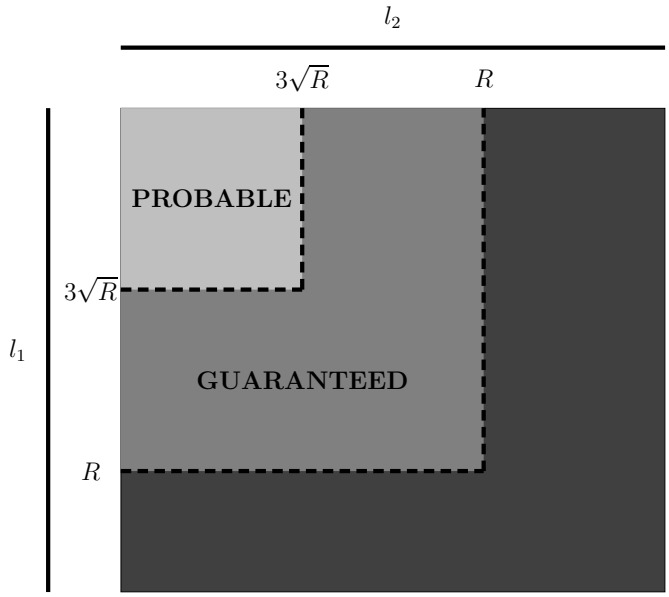

Figure 3.2: Searching the R-best set of $UB(l_1, l_2) = u_1(l_1) + u_2(l_2)$. The list of linear indices l_k is sorted such that the first R elements in the list compose the R-best set of $u_k(l_k)$. Consequently, the R-best set of $UB(l_1, l_2)$ is guaranteed to be in $\mathcal{S}_R^{u_1} \times \mathcal{S}_R^{u_2}$. Empirically, we observed that with sufficient probability the R-best set of the upper bound is in $\mathcal{S}_{3\sqrt{R}}^{u_1} \times \mathcal{S}_{3\sqrt{R}}^{u_2}$. Restricting the search to this set of most probable indices is a reasonable approximation with the benefit of heavily reduced computational efforts.

We use a similar principle to efficiently compute the R-best set of the lower bound (3.6). In a first step, we compute the R-best set of $\lambda_k(\{l_k\})$, $\mathcal{S}_R^{\lambda_k}$, for each k. For this task, we can use again the approximate R-best set retrieval method illustrated in Figure 3.2, because λ_k is decomposable. Next, we obtain $\mathcal{S}_R^{\mathrm{LB}}$ from the R-best set of the union set $\bigcup_k \mathcal{S}_R^{\lambda_k}$.

3.1.3 State Selection Strategies Based on Likelihood Bounds

We empirically studied the following three strategies to select the set of indices where the exact likelihoods are computed:

Upper Bound Selection (UBS): Use $\mathcal{S}_R^{\mathrm{UB}}$.

Lower Bound Selection (LBS): Use $\mathcal{S}_R^{\mathrm{LB}}$.

Union of Bounds Selection (UNBS): Compute the union set $\mathcal{S}_{\breve{R}}^{\mathrm{UB}\cup\mathrm{LB}} := \mathcal{S}_R^{\mathrm{UB}} \cup \mathcal{S}_R^{\mathrm{LB}}$, where $\breve{R}$ is the cardinality of the union. To ensure a proper experimental comparison with the previous two selection strategies, we limit the cardinality of the union set to R elements, i.e. we prune $\mathcal{S}_{\breve{R}}^{\mathrm{UB}\cup\mathrm{LB}}$ to the R-best set of $L(\mathcal{S}_{\breve{R}}^{\mathrm{UB}\cup\mathrm{LB}})$ and use the resulting set denoted by $\mathcal{S}_R^{\mathrm{UB}\cup\mathrm{LB}}$.

To give an impression of how well each of the three strategies performs in retrieving the best 1% likelihoods, we define the *recall* of a selection strategy; specifically, the recall of the UBS method is

$$\mathrm{recall}_R^{\mathrm{UB}} = \frac{|\mathcal{S}_{\tilde{R}}^{LL} \cap \mathcal{S}_R^{\mathrm{UB}}|}{|\mathcal{S}_{\tilde{R}}^{LL}|}, \tag{3.8}$$

where $\tilde{R} = \lfloor \frac{\Lambda}{100} \rfloor$ (i.e. 1% of the total number of likelihood elements), and $\mathcal{S}_{\tilde{R}}^{LL}$ is the $\tilde{R}$-best set of $L(\{x_k\},\{m_k\})$. The recall measures the fraction of 1% best likelihoods contained in $\mathcal{S}_R^{\mathrm{UB}}$. We define the recall of the LBS and UNBS analogously. Figure 3.3 shows the recall of $\mathcal{S}_R^{\mathrm{UB}}$, $\mathcal{S}_R^{\mathrm{LB}}$ and $\mathcal{S}_R^{\mathrm{UB}\cup\mathrm{LB}}$ as a function of R, evaluated on the same analysis frame used in Figure 3.1. We see that UNBS works best, as it always retrieves the highest fraction of the best 1% likelihoods. In this example, the best 22% elements obtained with the UNBS method include all of the best 1% likelihoods. Note that we obtain the same result both for the *exact* as well as the *approximate* R-best set retrieval method.

These preliminary results indicate that the proposed bound selection methods provide a computationally efficient way to retrieve a significant amount of the largest likelihoods $L(\{x_k\},\{m_k\})$. Given the index set $\mathcal{S}$ containing the selected likelihood indices, there are two possibilities to compute the resulting pruned set of observation probabilities (3.3):

- *Exact* likelihood computation for selected pitch combinations:

$$p(\boldsymbol{y}|\{x_k\}) = \begin{cases} \sum_{\{m_k\}} \left(\prod_k \alpha_{k,x_k}^{m_k}\right) L(\{x_k\},\{m_k\}) & \text{if } \{x_k\} \in \mathcal{P}, \\ 0 & \text{otherwise.} \end{cases} \tag{3.9}$$

- Likelihood *accumulation* for selected state combinations:

$$p(\boldsymbol{y}|\{x_k\}) = \begin{cases} 0 & \text{if } \mathcal{M}(\{x_k\}) = \emptyset, \\ \sum_{\{m_k\}\in\mathcal{M}(\{x_k\})} \left(\prod_k \alpha_{k,x_k}^{m_k}\right) L(\{x_k\},\{m_k\}) & \text{otherwise.} \end{cases} \tag{3.10}$$

where $\mathcal{P} = \{\{x_k\}|\exists\{m_k\} : (\{x_k\},\{m_k\}) \in \mathcal{S}\}$ is the set of selected pitch combinations and $\mathcal{M}(\{x_k\}) = \{\{m_k\}|(\{x_k\},\{m_k\}) \in \mathcal{S}\}$ is the set of selected GMM component combinations corresponding to a certain pitch combination. Throughout the experiments, we use

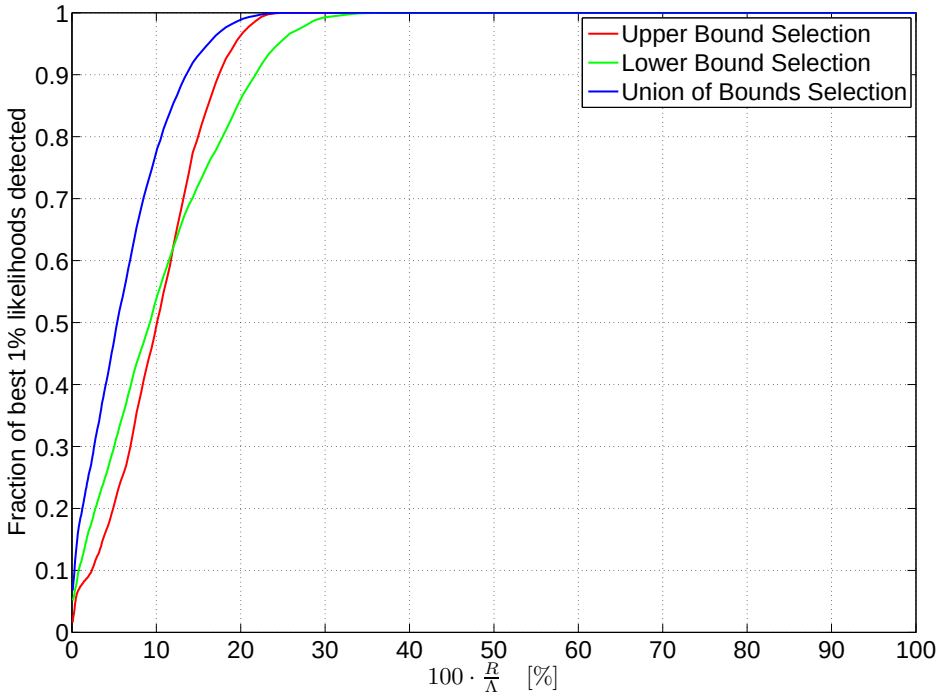

Figure 3.3: Recall of the three selection strategies as a function of R, computed for a single time frame from a mixture of two speakers. On the x-axis, values of R are shown relative to the total number of likelihood elements.

the *exact* computation method for mixtures of two speakers. For three or more speakers, we use the *accumulation* method for computational efficiency.

3.1.4 A Modified Junction Tree Algorithm for Sparse Likelihoods

In principle, it is straightforward to extend the junction tree algorithm presented in Chapter 2 to the case of K Markov chains, as shown in Algorithm 2. To make use of sparse observation likelihood matrices (or tensors, for $K > 2$), we slightly modify this algorithm as described in the following. The approach to make use of sparsity for tracking in a two-chain FHMM has been mentioned before in [100], however no algorithmic details were provided.

Let $\mathcal{P}^{(t)}$ denote the set of selected pitch combinations at time t. Due to the sparsity of $p(\boldsymbol{y}^{(t)}|\{x_k\})$, the variables $\gamma_j^{(t)}$ and $\beta_j^{(t)}$ in Algorithm 2 will be sparse as well, and it is best to represent these objects as lists, where each entry corresponds to a (state combination / value) pair.[16] Consider the maximization step along the j-th chain in the

[16] A sparse matrix is commonly represented in an analogous manner as a list of indices and corresponding values.

forward recursion:

$$\gamma_j^{(t)}\left(\{x_k^{(t)}\}_{k=1}^{j},\{x_k^{(t-1)}\}_{k=j+1}^{K}\right) \leftarrow \max_{x_j^{(t-1)}}\Big[\ln p\left(x_j^{(t)}|x_j^{(t-1)}\right) + \gamma_{j-1}^{(t)}\left(\{x_k^{(t)}\}_{k=1}^{j-1},\{x_k^{(t-1)}\}_{k=j}^{K}\right)\Big]. \tag{3.11}$$

Maximization only needs to consider states $x_j^{(t)}$ that do occur in an element of $\mathcal{P}^{(t)}$. After this maximization step, we can remove all entries from $\gamma_j^{(t)}$ whose first j states, $\{x_k^{(t)}\}_{k=1}^{j}$, do not occur in any element of $\mathcal{P}^{(t)}$.
Unfortunately, the computational complexity of this modified Viterbi algorithm is still exponential in the number of Markov chains. For practical purposes, however, this variant is considerably faster compared to exact inference, and eventually extends practical applicability to a higher number of Markov chains.

Input: Set of observations $\mathcal{Y}$
Output: Optimal state sequence $\mathcal{X}^*$
Initialization: Compute state likelihoods $p\left(\boldsymbol{y}^{(t)}|\{x_k^{(t)}\}\right) \quad \forall\, t \in \{1,...,T\}$
$\gamma_0^{(2)}\left(\{x_k^{(1)}\}\right) \leftarrow \sum_{k=1}^{K}\ln p\left(x_k^{(1)}\right) + \ln p\left(\boldsymbol{y}^{(1)}|\{x_k^{(1)}\}\right)$
Forward recursion:
foreach $t \in \{2,\ldots,T\}$ **do**
  foreach $j \in \{1,\ldots,K\}$ **do**
    $\gamma_j^{(t)}\left(\{x_k^{(t)}\}_{k=1}^{j},\{x_k^{(t-1)}\}_{k=j+1}^{K}\right) \leftarrow$
    $\max_{x_j^{(t-1)}}\left[\ln p\left(x_j^{(t)}|x_j^{(t-1)}\right) + \gamma_{j-1}^{(t)}\left(\{x_k^{(t)}\}_{k=1}^{j-1},\{x_k^{(t-1)}\}_{k=j}^{K}\right)\right]$
    $\beta_j^{(t)}\left(\{x_k^{(t)}\}_{k=1}^{j},\{x_k^{(t-1)}\}_{k=j+1}^{K}\right) \leftarrow$
    $\arg\max_{x_j^{(t-1)}}\left[\ln p\left(x_j^{(t)}|x_j^{(t-1)}\right) + \gamma_{j-1}^{(t)}\left(\{x_k^{(t)}\}_{k=1}^{j-1},\{x_k^{(t-1)}\}_{k=j}^{K}\right)\right]$
  end
  $\gamma_0^{(t+1)}\left(\{x_k^{(t)}\}\right) \leftarrow \gamma_K^{(t)}\left(\{x_k^{(t)}\}\right) + \ln p\left(\boldsymbol{y}^{(t)}|\{x_k^{(t)}\}\right)$
end
$\{x_k^{(T)^*}\} \leftarrow \arg\max_{\{x_k^{(T)}\}}\left[\gamma_0^{(T+1)}\left(\{x_k^{(T)}\}\right)\right]$
Backtracking:
foreach $t \in \{T,\ldots,2\}$ **do**
  foreach $j \in \{K,\ldots,1\}$ **do**
    $x_j^{(t-1)^*} \leftarrow \beta_j^{(t)}\left(\{x_k^{(t)^*}\}_{k=1}^{j},\{x_k^{(t-1)^*}\}_{k=j+1}^{K}\right)$
  end
end

Algorithm 2: The junction tree algorithm for a K-chain FHMM on a max-sum semiring. This algorithm gives the exact solution to (2.9). The quantities $\gamma_j^{(t)}$, $\beta_j^{(t)}$, and $\gamma_0^{(t+1)}$ represent K-dimensional tables, where each entry corresponds to a specific combination of hidden states. During the forward recursion, their entries are computed for each state combination. Note that for $K = 2$, this algorithm is equivalent to Algorithm 1.

3.2 Experiments

3.2.1 Mixtures of Two Speakers

We compare the performance of the three proposed bound selection methods on mixtures of two speakers, using the same experimental setup as in Chapter 2 (see Section 2.5). We varied the pruning parameter R over a range from 20 to 10000. For each setting of R, we evaluated the performance of each of the three bound selection methods on 135 test mixtures. Results are shown in Figures 3.4 and 3.5 for GRID and PTDB-TUG database, respectively. The dashed horizontal line shows the performance achieved by exact inference, i.e. without likelihood pruning, as reported in Section 2.6.1. In every case, each of the three bound selection methods approaches the performance of exact inference with increasing R. UNBS approaches this limit for the smallest R in all cases, while UBS performs worst.

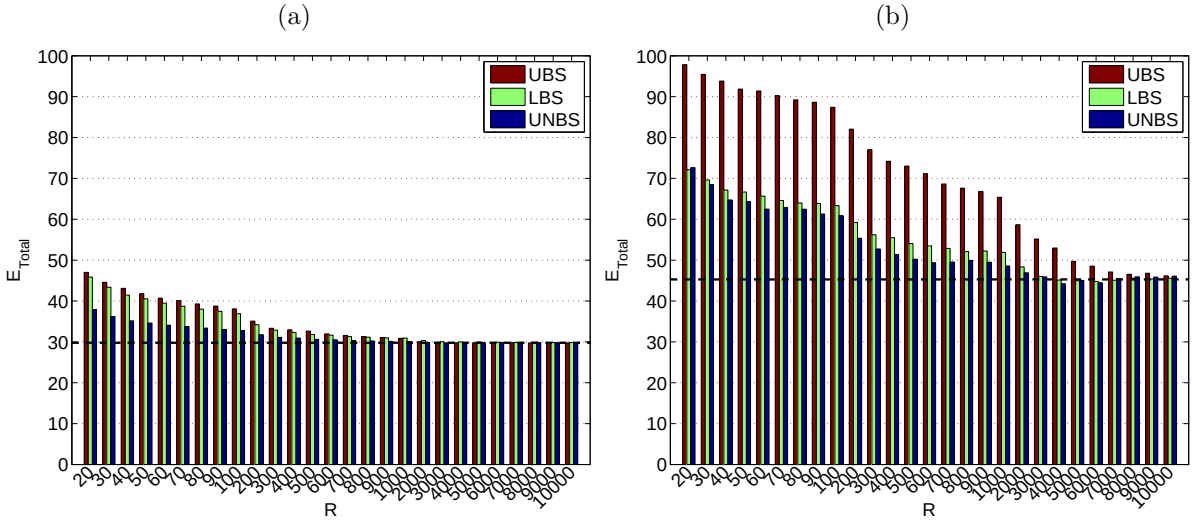

Figure 3.4: Error measure $\bar{E}_{Total}$ evaluated on 135 test mixtures of GRID database for different settings of pruning parameter R: (a) SD models. (b) SI models. For each R, the mean performance is shown for the upper bound selection (UBS), lower bound selection (LBS) and the union of bounds selection (UNBS) method. The dashed horizontal line indicates the exact inference performance (cf. Table 2.7). For both types of speaker models, the UNBS method approaches the exact performance for the lowest number of R.

To indicate the computation time of the methods involved, measurements were performed on a 3.2-GHz six core machine with 12-GB main memory. All algorithms were implemented and tested in Matlab. For the exact computation of the likelihoods for the selected pitch combinations $\mathcal{P}$, a Matlab-MEX implementation was used. Measurements were performed on six test mixtures of the GRID database, where only the time needed for likelihood pruning and subsequent calculation of selected likelihood values was measured. For each test mixture, the measured time was divided by the total number of analysis frames T of the mixture. The averaged results are shown in Figure 3.6 for SD and SI models. In comparison, the average time per analysis frame needed for exact likelihood computation is 0.21 s when using SD models and 2.17 s when using SI models. Note that the SI model requires more computation time than SD models because it contains GMMs

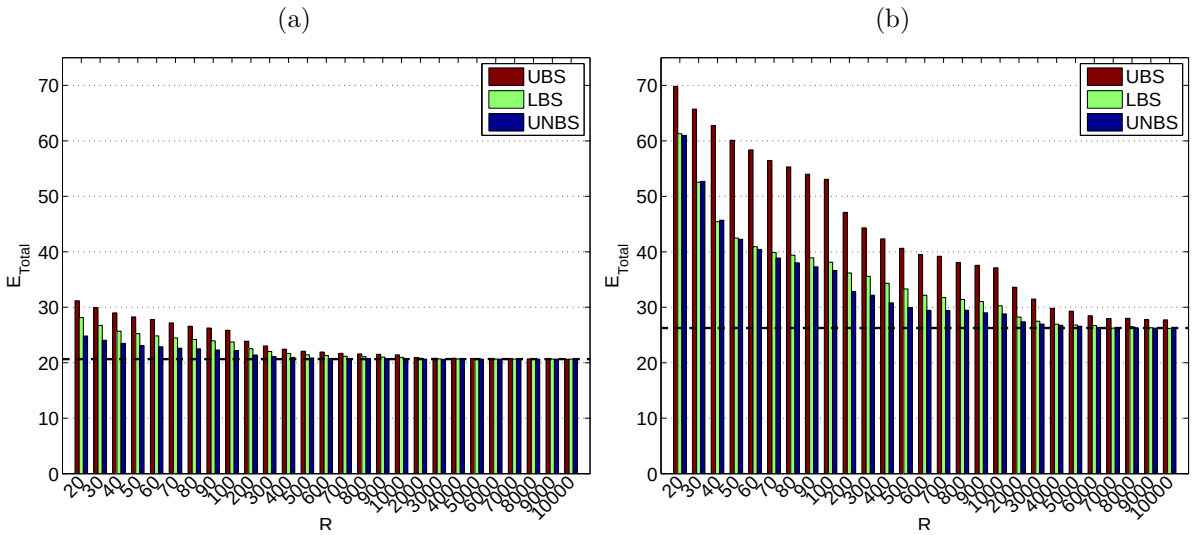

Figure 3.5: Error measure $\bar{E}_{Total}$ evaluated on 135 test mixtures of PTDB-TUG database for different settings of pruning parameter R: (a) SD models. (b) SI models. For each R, the mean performance is shown for the upper bound selection (UBS), lower bound selection (LBS) and the union of bounds selection (UNBS) method. The dashed horizontal line indicates the exact inference performance (cf. Table 2.8). For both types of speaker models, the UNBS method approaches the exact performance for the lowest number of R.

with more components. For small values of R, a constant computational overhead dominates the required time, while for larger increasing values of R the required time scales linearly with R. The UBS method is always fastest, however LBS and UNBS do not need excessively more time compared to UBS.
Considering the time measurements together with the tracking results from Figures 3.4 and 3.5, we conclude that from the three proposed selection methods, UNBS is the best choice for likelihood pruning.

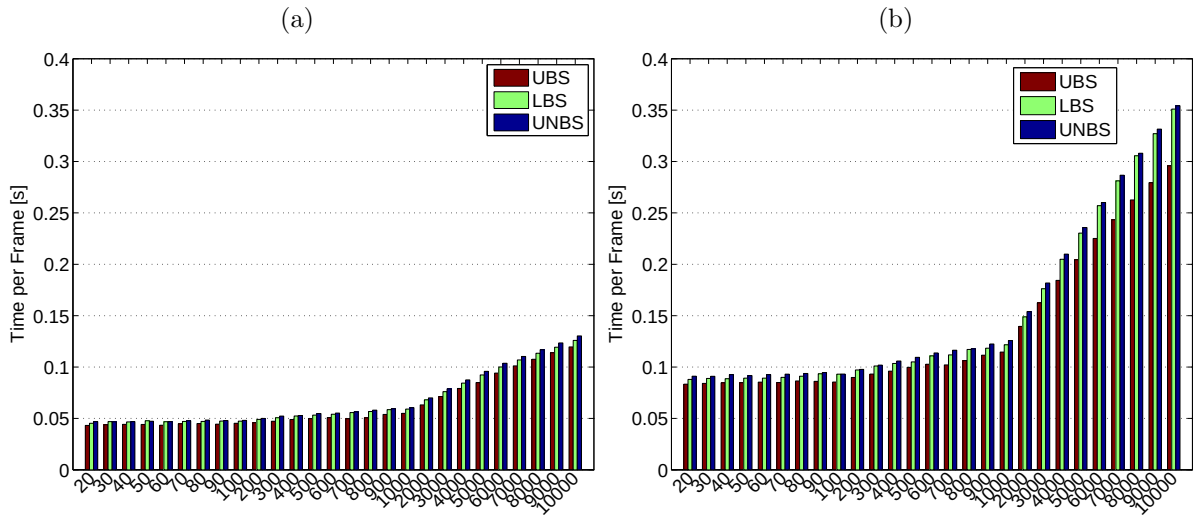

Figure 3.6: Computation time of likelihood calculation per analysis frame, averaged on 6 test mixtures of GRID database for different settings of pruning parameter R: (a) SD models. (b) SI models. For each R, the mean time in seconds is shown for the upper bound selection (UBS), lower bound selection (LBS) and the union of bounds selection (UNBS) method. In comparison, the average time per analysis frame needed for exact likelihood computation is 0.21 *s when using SD models and* 2.17 *s when using SI models.*

3.2.2 Mixtures of Three Speakers

With the proposed approximate inference scheme, we are able to perform tracking of more than two speakers, a case which is practically infeasible when using exact inference. In the experimental evaluation, we restrict ourselves to SD models. Before we present the results, we describe some differences in the experimental setup compared to previous evaluations.

Data

We created mixtures of three speakers from both the GRID and the PTDB-TUG database. For each database, 6 available test speakers result in $\binom{6}{3} = 20$ unique triplets of speakers. We created 5 test mixtures for each triplet, resulting in 100 test mixtures per database. All three utterances within a test mixture have the same gain.

Feature Extraction and Model Training

Due to the increased difficulty of the problem, we extend the bandwidth of the spectral features to 2 kHz, i.e. each observation vector $\boldsymbol{y}^{(t)}$ now has 128 elements. Thus, we increase the chance to observe for each of the three speakers a sufficient number of frequency bins dominated by that speaker. All SD speaker models have been retrained on this extended bandwidth.

Error measure for K speakers

In order to assess the performance of SD multi-pitch tracking with $K \geq 3$ simultaneous speakers, we define a new error measure $\hat{E}_{Total}$. Similar to the error measure proposed in Chapter 2, the goal of this measure is to assess the overall accuracy and how well pitch estimates are assigned to their corresponding speakers. The error measure is calculated as follows: In a first step, the pitch trajectory estimated by the k-th Markov chain, $\tilde{f}_0^{(k)}[t]$, is assigned to the reference trajectory $f_0^{(k)}[t]$ of the corresponding speaker (i.e. to the trajectory it is supposed to model). Next, for each $k = 1, \ldots, K$ we calculate the speaker assigned pitch frequency deviation $\bar{\Delta}f^{(k)}[t]$ (see Equation (2.15)), and define the following error terms:

- The permutation error $\hat{E}_{Perm}^{(k)}$ is the percentage of time frames where $\bar{\Delta}f^{(k)}[t] > 20\%$ and $\tilde{f}_0^{(k)}[t]$ is within the 20% error bound of some other reference pitch.
- The gross error $\hat{E}_{Gross}^{(k)}$ is the percentage of time frames where no permutation error occurred and $\bar{\Delta}f^{(k)}[t] > 20\%$.
- The voicing error $\hat{E}_{01}^{(k)}$ is the percentage of time frames where $f_0^{(k)}[t]$ is unvoiced, $\tilde{f}_0^{(k)}[t]$ is voiced, and no permutation error occurred.
- The voicing error $\hat{E}_{10}^{(k)}$ is the percentage of time frames where $f_0^{(k)}[t]$ is voiced and $\tilde{f}_0^{(k)}[t]$ is unvoiced.
- The fine error $\hat{E}_{Fine}^{(k)}$ is the average of $\bar{\Delta}f^{(k)}[t]$ at those time frames where $\bar{\Delta}f^{(k)}[t] < 20\%$.

Finally, we take the average of each error term over all speakers and compute the total error $\hat{E}_{Total}$:

$$\hat{E}_{Total} = \frac{1}{K} \sum_{k=1}^{K} \left(\hat{E}_{01}^{(k)} + \hat{E}_{10}^{(k)} + \hat{E}_{Gross}^{(k)} + \hat{E}_{Perm}^{(k)} + \hat{E}_{Fine}^{(k)} \right). \tag{3.12}$$

Experimental Results

All results have been obtained with the UNBS method and pruning parameter $R = 50000$. Moreover, the accumulation method was used to compute the sparse set of observation likelihoods.[17]

Performance results in terms of $\hat{E}_{Total}$ obtained on the GRID and PTDB-TUG database are shown respectively in Figures 3.7 and 3.8, where the error measure is shown for each of the 100 test mixtures, grouped by the speaker triplets. To show an example for the performance associated with a certain value of $\hat{E}_{Total}$, Figures 3.9, 3.10, 3.11 and 3.12 show tracking results for four different test mixtures. For each of the two databases, the best performing test mixture is shown, as well as a test mixture where the performance is similar to the average performance of all test mixtures of the database. Despite massive

[17] With this setting, on average only about 23000 out of 170^3 possible pitch combinations (i.e. 0.5%) per time frame were nonzero for one randomly chosen test mixture.

likelihood pruning, the accuracy in terms of pitch estimation as well as speaker assignment is still reasonably well.
To indicate the required computation time, measurements were again performed on a 3.2-GHz six core machine with 12-GB main memory. All algorithms were implemented and tested in Matlab (without MEX acceleration). The required computation time for likelihood selection and calculation averaged on two test mixtures of the GRID corpus was 6.5 s per time frame.

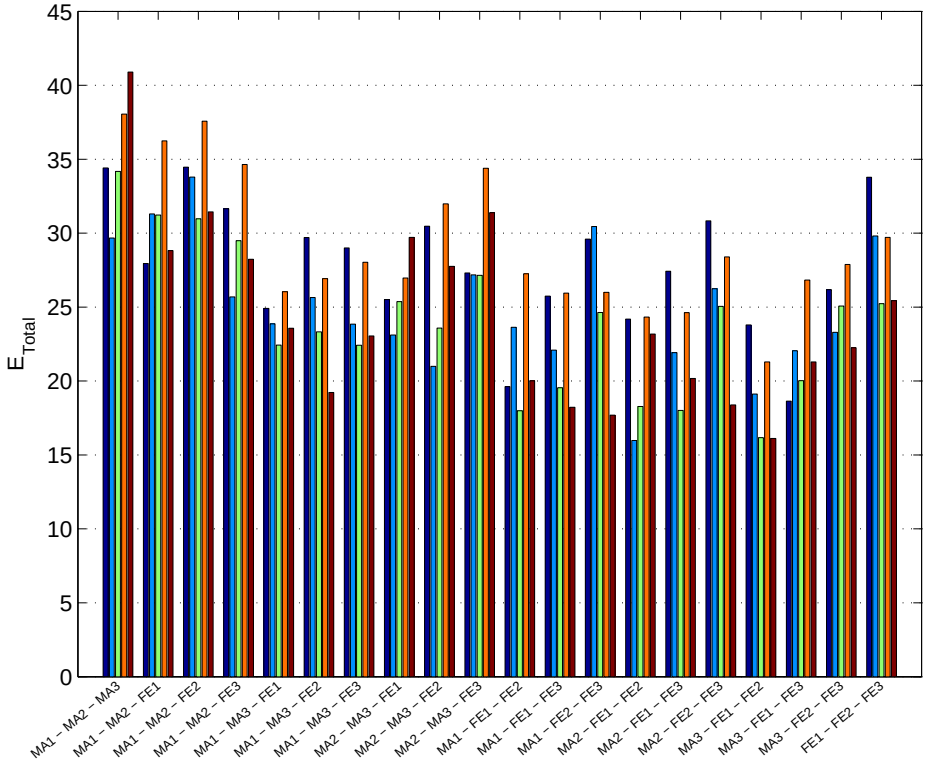

Figure 3.7: Performance of tracking 3 simultaneous speakers from GRID with SD models using the UNBS method. For each of 20 speaker triplets, the error measure $\hat{E}_{Total}$ was evaluated on 5 test mixtures (represented by colored bars). The average of $\hat{E}_{Total}$ on all 100 test mixtures is 26.05.

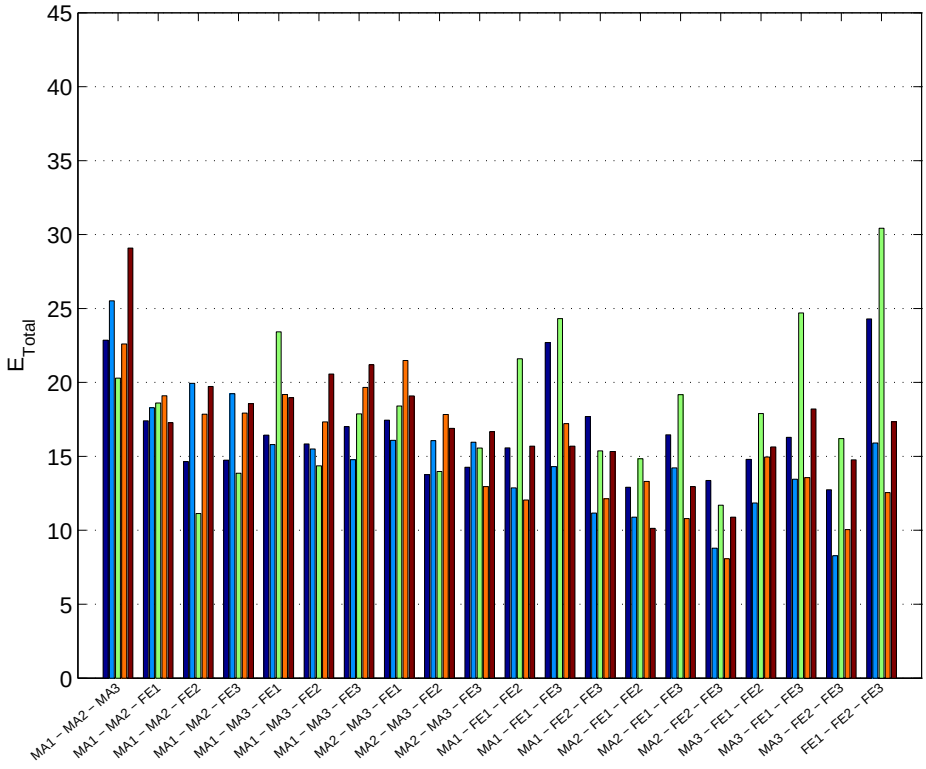

Figure 3.8: Performance of tracking 3 simultaneous speakers from PTDB-TUG with SD models using the UNBS method. For each of 20 speaker triplets, the error measure $\hat{E}_{Total}$ was evaluated on 5 test mixtures (represented by colored bars). The average of $\hat{E}_{Total}$ on all 100 test mixtures is 16.49.

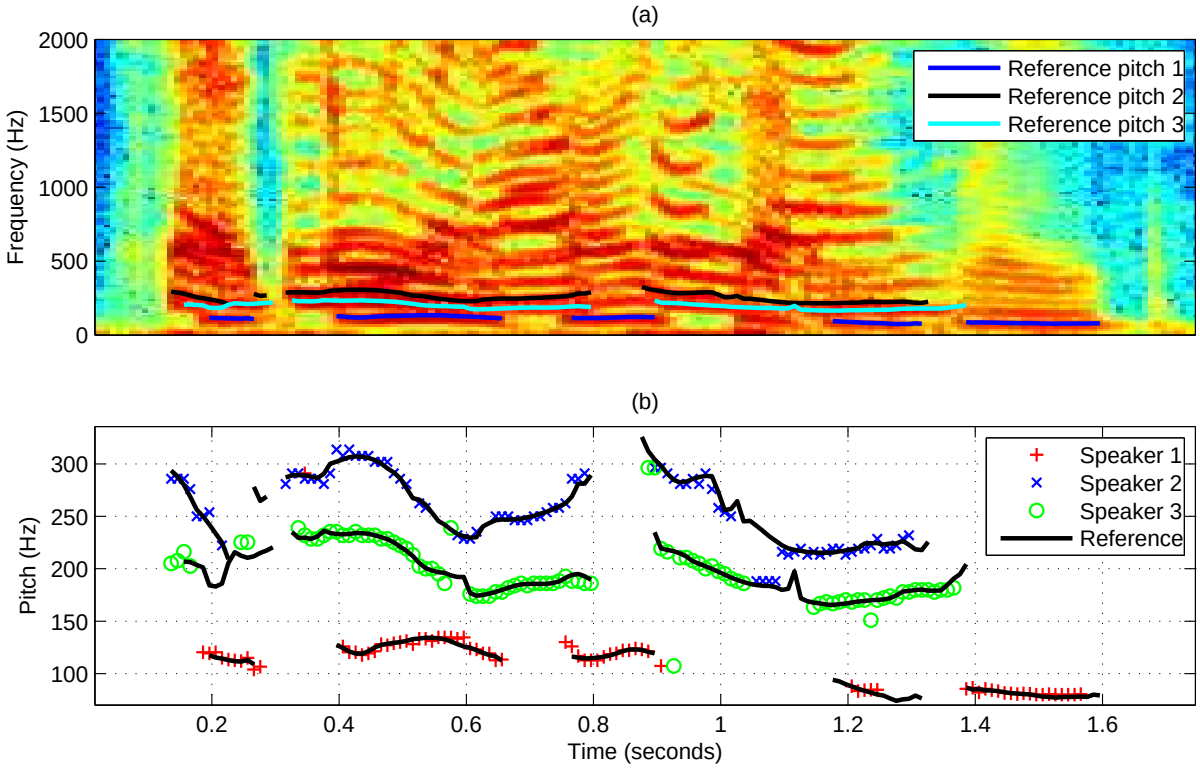

Figure 3.9: Tracking result for mixture of 3 speakers from GRID corpus (MA2: "sgai7p", FE1: "sbil4a", FE2: "sbil2a"). (a) Spectrogram of mixture together with reference pitch trajectories (colored solid lines). (b) Estimated pitch trajectories (colored markers) and reference pitch trajectories (black solid lines). This is the test mixture where the best performance among all GRID test mixtures was achieved ($\hat{E}_{Total} = 15.9811$).

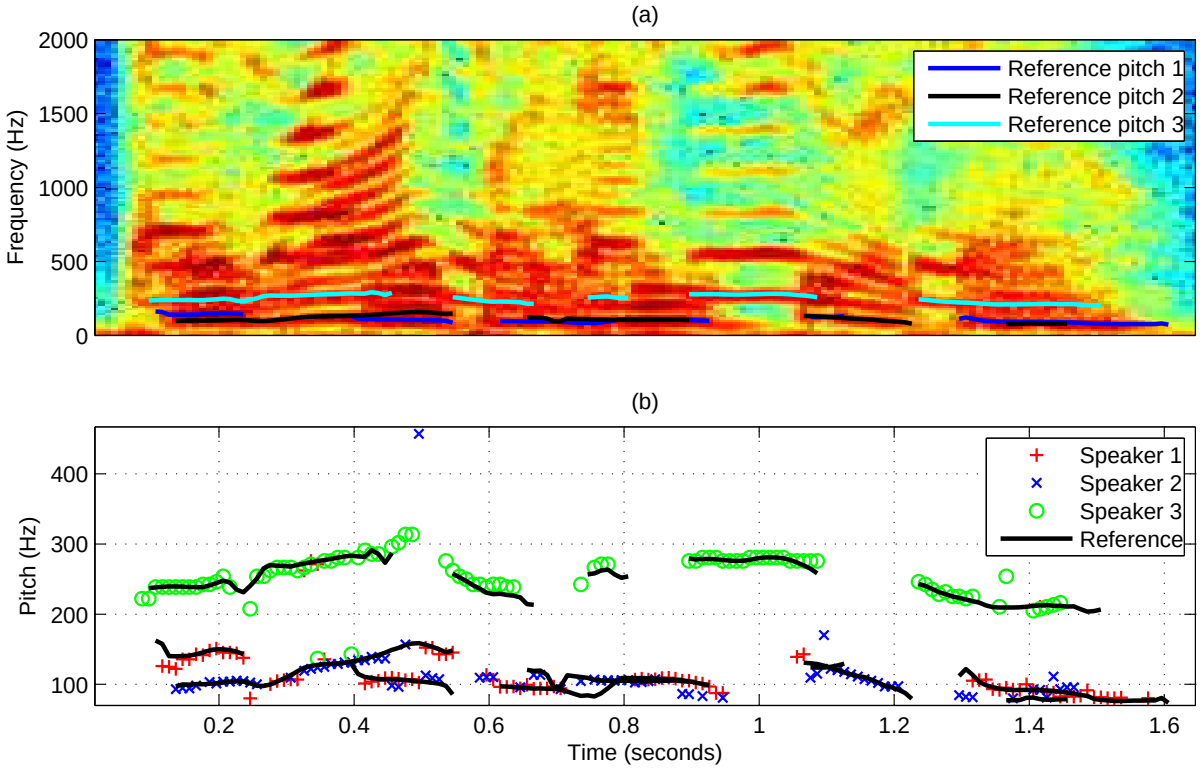

Figure 3.10: Tracking result for mixture of 3 speakers from GRID corpus (MA1: "pbbv6n", MA2: "lwwm2a", FE1: "lwixzs"). (a) Spectrogram of mixture together with reference pitch trajectories (colored solid lines). (b) Estimated pitch trajectories (colored markers) and reference pitch trajectories (black solid lines). For this text mixture, a performance of 27.95 *is achieved, which is approximately the average performance of all test mixtures from the GRID corpus.*

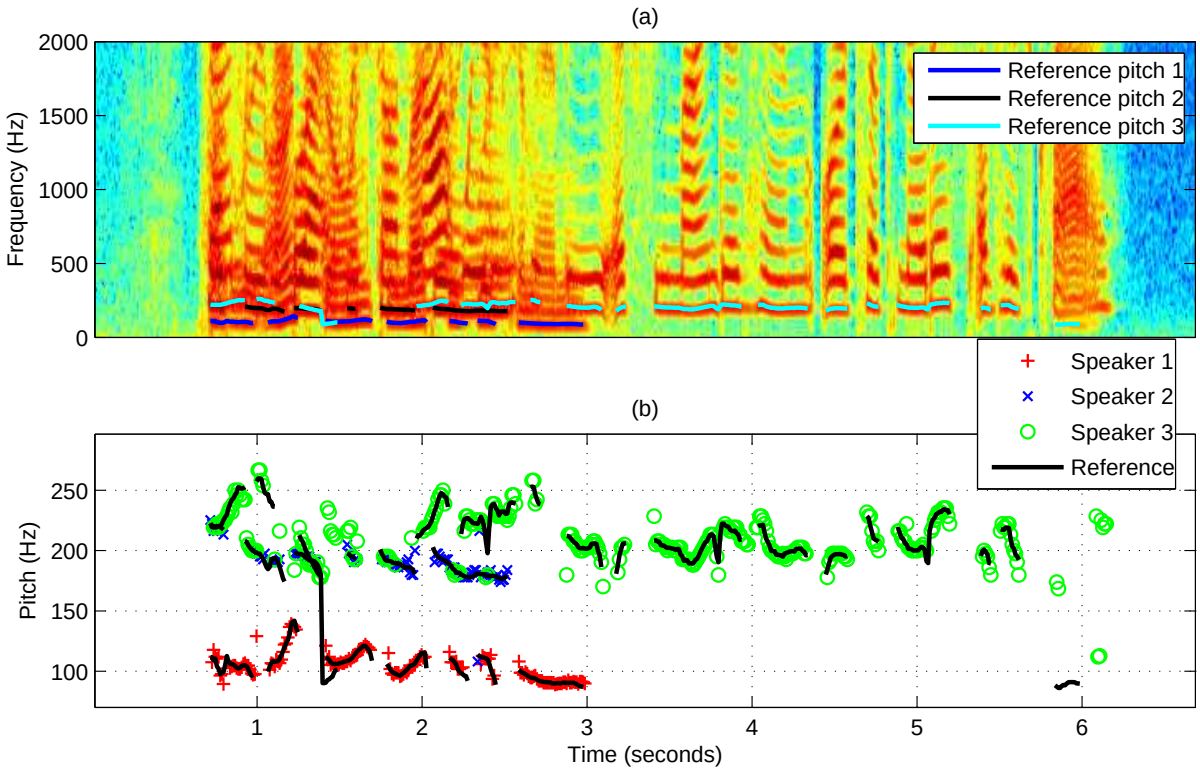

Figure 3.11: Tracking result for mixture of 3 speakers from PTDB-TUG corpus (MA2: "But we aren't going to let you give him any.", FE2: "Hastily the boy switched on a ceiling light.", FE3: "Then he fled, not waiting to see if she minded him or took notice of his cry."). (a) Spectrogram of mixture together with reference pitch trajectories (colored solid lines). (b) Estimated pitch trajectories (colored markers) and reference pitch trajectories (black solid lines). This is the test mixture where the best performance among all PTDB-TUG test mixtures was achieved ($\hat{E}_{Total} = 8.08$). The method correctly recognizes that only one out of three speakers is present in the second half of the file.

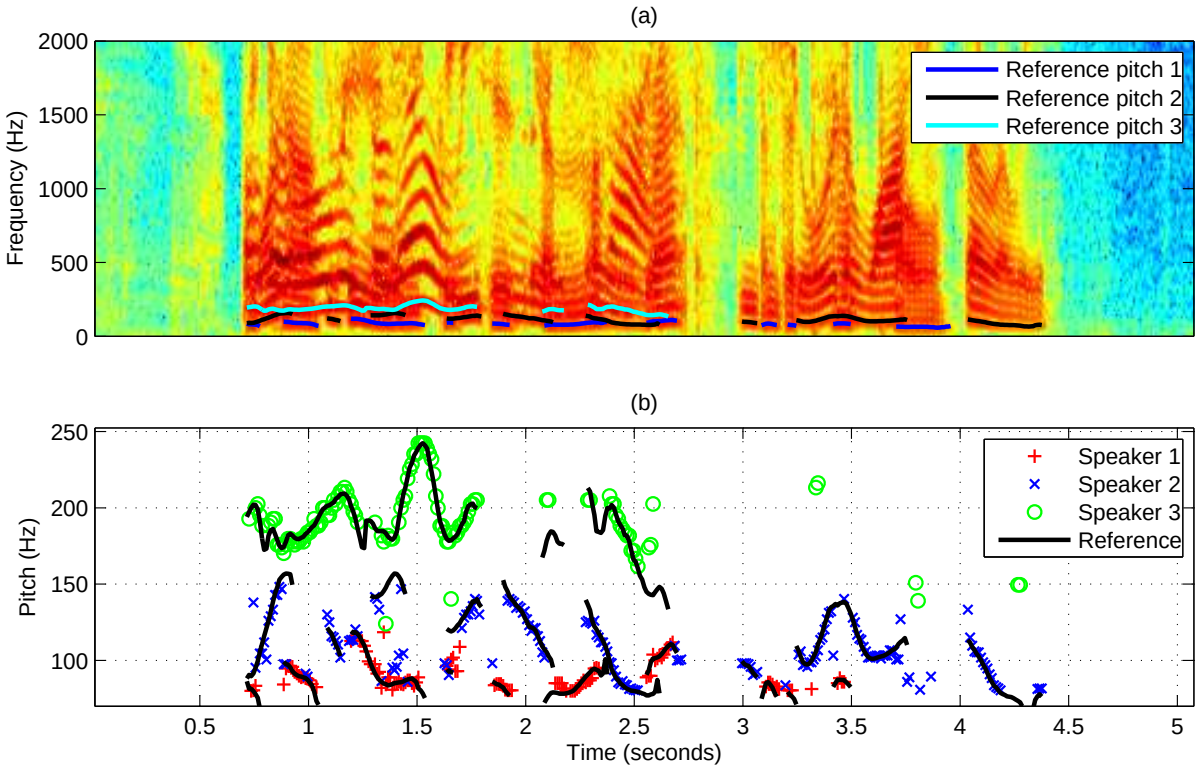

Figure 3.12: Tracking result for mixture of 3 speakers from PTDB-TUG corpus (MA1: "This cartoon features a muskrat and a tadpole.", MA3: "Once you finish greasing your chain, be sure to wash it thoroughly.", FE1: "Although always alone, we survive."). (a) Spectrogram of mixture together with reference pitch trajectories (colored solid lines). (b) Estimated pitch trajectories (colored markers) and reference pitch trajectories (black solid lines). For this test mixture, a performance of 16.44 *is achieved, which is approximately the average performance of all test mixtures from the PTDB-TUG corpus.*

4
Model Adaptation

In Chapter 2, we introduced a model-based approach for multi-pitch tracking and studied its performance given SD and SI models. The availability of SD models is of great advantage, both in terms of accuracy as well as correct speaker assignment. However, the drawback of SD models is that sufficient amounts of speaker-specific speech data must be available for model training. Both data collection and training can be a time consuming task. Even if we have SD models available, we might encounter different channel conditions in the test case, i.e. the spectral characteristics of each source signal might have changed due to multi-path propagation in a room or a different microphone transfer function. The same holds for the problem of gain mismatches. During model training, all training utterances are normalized to unit-variance. This way, the resulting speaker model still captures energy fluctuations due to natural speaking style, but it implicitly assumes that the amplitude of the speech signal has a certain order of magnitude. Any mismatch between the speaker models and the actual condition in a recording results in a degraded tracking accuracy. The goal of model adaptation is to tune the available speaker models to the specific speaker characteristics and channel conditions that are present in a previously unseen recording. Ideally, we want to obtain good SD models from as less speech data as possible. Moreover, we want to be able to adapt these models to the environmental conditions encountered in a given test mixture, i.e. we want to adapt the model of each speaker involved given only the observed mixture of speech.
There exists a large amount of techniques for model adaptation in literature, and many of these approaches have been developed in the context of speech recognition. Some of the most successful approaches are the maximum likelihood linear regression (MLLR) framework [101, 102] (described in Section 4.1), maximum a posteriori (MAP) estimation [103], and rapid adaptation in eigenvoice space [104]. While these approaches assume that adaptation data consists of clean speech, extensions and other methods for learning or adaptation of source models from speech corrupted by a background signal have been developed as well. The general aim is then to learn or adapt an *undistorted* source model from corrupted speech, i.e. the incorporation of the background distortion into the speech model should be avoided. Probably one of the earliest approaches has been proposed by Nadas et al. [79] in the context of noise robust speech recognition. Speech and noise are

represented separately by individual models, which are combined using the MIXMAX approximation to obtain a model of noisy speech. Then, a mechanism is proposed to estimate the GMM-based speaker model from noisy speech assuming the noise model is known. This idea has been extended by Rose et al. [81] in terms of a more general formulation of interaction models and a more flexible background noise model based on GMMs. In [105], the eigenvoice approach is generalized to adapt individual speaker models given a superposition of two speech signals. Other approaches apply related principles to learn a model of the background distortion. In [106], a dynamic noise model is adaptively updated from noisy speech, which is used in conjunction with an a priori learned speech model to obtain an estimate of clean speech. A different approach can be found in [107], where a Gaussian background model as well as a frequency dependent gain of the speech model is adapted from noisy speech using an EM algorithm. In [108], a framework is proposed to learn noise and channel distortions jointly from corrupted speech, assuming that an accurate speech model is available. In [109], the MAP estimation approach is extended to tackle the problem of voice/music separation in popular songs by first adapting a general music model to non-vocal portions of a given song, and then adapting the gain and filters of both the music as well as a general speech model given the whole portion of the song. Recently, an approach for joint learning of source models and source location from multichannel recordings was proposed [110].
For the problem of gain adaptation from speech mixtures, several approaches have been proposed in literature. In a related model-based framework using FHMMs for monaural speech separation, methods for gain adaptation based on iterated tracking and derivative-free optimization have been introduced [77,111]. In [68,112], two different EM algorithms for gain adaptation are proposed. In [99], a variational framework for approximate inference in conjunction with the MIXMAX model was developed, which was further used for gain adaptation.
In this chapter, we introduce a mechanism for model adaptation from speech mixtures, which is specifically tailored to the proposed framework for multi-pitch tracking. Our approach is based on the MLLR framework, and we first give a short overview on the basic principle of MLLR in the following section. Next, we describe our approach in Section 4.2. In 4.3, we demonstrate experimental results, and discuss limitations of the current approach.

4.1 The Basic Principle of MLLR

MLLR [101] was originally developed to adapt the mean parameters of a (SI) continuous density HMM. Assuming for simplicity that the emission density of each HMM state x is a single normal distribution,[18] MLLR modifies the mean parameter $\boldsymbol{\mu}_x$ of state x with an affine transform:

$$\hat{\boldsymbol{\mu}}_x = \boldsymbol{T}_x \boldsymbol{\xi}_x, \tag{4.1}$$

where $\boldsymbol{T}_x$ is a $D \times (D+1)$ transformation matrix and $\boldsymbol{\xi}_x = (1, \boldsymbol{\mu}_x^T)^T$ is the mean vector enhanced by an offset term. MLLR was later extended to a transform of mean and covariance parameters [102]. In this work, we restrict ourselves to the adaptation of mean

[18] The same principle can be applied for GMM-based state emission densities.

parameters only, and the extended transform is not further discussed here.
As stated in [101], a separate transform for each state emission (and in case of GMMs, for every mixture component) would be equivalent to a complete re-estimation of all mean parameters, which would leave the problem of adapting unseen distributions unsolved. For this reason, the same transform is used for multiple emission densities (*parameter tying*), and in the following we assume that the *same* transform is applied to all components:

$$\hat{\boldsymbol{\mu}}_x = \boldsymbol{T}\boldsymbol{\xi}_x \tag{4.2}$$

The transformation matrix is obtained by maximizing the log-likelihood $\mathrm{LL}(\boldsymbol{T})$ of the transformed HMM on adaptation speech data $\mathcal{S} = \{\boldsymbol{s}^{(t)}\}_{t=1}^{T}$:

$$\mathrm{LL}(\boldsymbol{T}) = \ln p(\mathcal{S}|\boldsymbol{T}) = \ln \sum_{\mathcal{X}} p(\mathcal{X}, \mathcal{S}|\boldsymbol{T}), \tag{4.3}$$

where

$$p(\mathcal{X}, \mathcal{S}|\boldsymbol{T}) = p(x^{(1)}) \prod_{t=2}^{T} p(x^{(t)}|x^{(t-1)}) \prod_{t=1}^{T} p(\boldsymbol{s}^{(t)}|x^{(t)}, \boldsymbol{T}) \tag{4.4}$$

is the joint distribution of the adaptation data and a hidden state sequence $\mathcal{X}$. The conditional dependency on transformation parameters is made explicit.
It is difficult to maximize the log-likelihood directly. Instead, Jensen's inequality is applied to construct a lower bound on (4.3), which is in general easier to optimize [113]. For any distribution $q(\cdot)$, and any joint probability $p(\mathcal{X}, \mathcal{Y})$, it follows from Jensen's inequality that

$$\ln \sum_{\mathcal{X}} p(\mathcal{X}, \mathcal{Y}) = \ln \sum_{\mathcal{X}} q(\mathcal{X}) \frac{p(\mathcal{X}, \mathcal{Y})}{q(\mathcal{X})} \geq \sum_{\mathcal{X}} q(\mathcal{X}) \ln \frac{p(\mathcal{X}, \mathcal{Y})}{q(\mathcal{X})}, \tag{4.5}$$

and equality holds if and only if $q(\mathcal{X}) = p(\mathcal{X}|\mathcal{Y})$. As a result, the lower bound Q on the log-likelihood is:

$$\mathrm{LL}(\boldsymbol{T}) \geq Q(\boldsymbol{T}) := \sum_{\mathcal{X}} q(\mathcal{X}) \ln p(\mathcal{X}, \mathcal{S}|\boldsymbol{T}) + \mathrm{const}, \tag{4.6}$$

where const refers to all terms independent of $\boldsymbol{T}$. Starting with an initial guess for the transformation matrix, a local maximum of LL can be found using the EM algorithm. During the E-Step, the variational distribution $q(\mathcal{X})$ is set such that the lower bound (4.6) is tight at the current parameter estimate.[19] In the subsequent M-Step, the lower bound (commonly referred to as *auxiliary function*) is maximized with respect to the parameters. Due to the tightness at the current parameter estimate, any increase in the lower bound guarantees an increase in LL [113, 114]. Both steps are repeated until no further increase in LL is achieved. In the case of MLLR, the two steps of the EM algorithm can be formally written as:

[19] Up to a term that does not depend on adaptation parameters.

E-Step Set the variational distribution according to:

$$q(\mathcal{X}) = p(\mathcal{X}|\mathcal{S}, \boldsymbol{T}^{(old)}). \tag{4.7}$$

M-Step Maximize the lower bound with respect to $\boldsymbol{T}$:

$$\arg\max_{\boldsymbol{T}} Q(\boldsymbol{T}) = \arg\max_{\boldsymbol{T}} \sum_{\mathcal{X}} q(\mathcal{X}) \ln p(\mathcal{X}, \mathcal{S}|\boldsymbol{T}). \tag{4.8}$$

From these formal steps, practically usable update equations are obtained as presented in the sequel.
The lower bound in (4.8) can be further modified by using Equation (4.4):

$$\begin{aligned} Q(\boldsymbol{T}) &= \sum_{\mathcal{X}} q(\mathcal{X}) \ln p(\mathcal{X}, \mathcal{S}|\boldsymbol{T}) && (4.9)\\ &= \sum_{\mathcal{X}} q(\mathcal{X}) \sum_{t=1}^{T} \ln p(\boldsymbol{s}^{(t)}|x^{(t)}, \boldsymbol{T}) + \text{const} && (4.10)\\ &= \sum_{t=1}^{T} \sum_{x^{(t)}=1}^{|X|} \sum_{\mathcal{X}\setminus x^{(t)}} q(\mathcal{X}) \ln p(\boldsymbol{s}^{(t)}|x^{(t)}, \boldsymbol{T}) + \text{const} && (4.11)\\ &= \sum_{t=1}^{T} \sum_{x=1}^{|X|} \gamma_t(x) \ln p(\boldsymbol{s}^{(t)}|x, \boldsymbol{T}) + \text{const}. && (4.12) \end{aligned}$$

The quantity $\gamma_t(x) = \sum_{\mathcal{X}\setminus x^{(t)}} q(\mathcal{X}) = \sum_{\mathcal{X}\setminus x^{(t)}} p(\mathcal{X}|\mathcal{S}, \boldsymbol{T}^{(old)})$ corresponds to the marginal posterior of hidden variable $x^{(t)}$, which is obtained during the E-Step by applying the forward-backward algorithm on the HMM [115].
Further expanding Equation (4.12) by using the definition of the normal distribution, we obtain:

$$Q(\boldsymbol{T}) = -\frac{1}{2} \sum_{t=1}^{T} \sum_{x=1}^{|X|} \gamma_t(x) (\boldsymbol{s}^{(t)} - \boldsymbol{T}\boldsymbol{\xi}_x)^T \boldsymbol{\Sigma}_x^{-1} (\boldsymbol{s}^{(t)} - \boldsymbol{T}\boldsymbol{\xi}_x) + \text{const}. \tag{4.13}$$

During the M-Step, the lower bound is maximized by setting $\frac{\partial Q(\boldsymbol{T})}{\partial \boldsymbol{T}} = 0$, which results in the following matrix equation:

$$\sum_{x=1}^{|X|} \underbrace{\sum_{t=1}^{T} \gamma_t(x) \boldsymbol{\Sigma}_x^{-1}}_{=:\boldsymbol{A}_x} \boldsymbol{T} \underbrace{\boldsymbol{\xi}_x \boldsymbol{\xi}_x^T}_{=:\boldsymbol{B}_x} = \underbrace{\sum_{t=1}^{T} \sum_{x=1}^{|X|} \gamma_t(x) \boldsymbol{\Sigma}_x^{-1} \boldsymbol{s}^{(t)} \boldsymbol{\xi}_x^T}_{\boldsymbol{C}}. \tag{4.14}$$

In general, a matrix equation of this type can be solved in closed form [116]:

$$\text{vec}(\boldsymbol{T}) = \left(\sum_{x=1}^{|X|} (\boldsymbol{B}_x^T \otimes \boldsymbol{A}_x) \right)^{-1} \text{vec}(\boldsymbol{C}), \tag{4.15}$$

where $\otimes$ denotes the Kronecker product and $\text{vec}(\boldsymbol{T})$ is a column vector obtained by

sequentially stacking the columns of $\boldsymbol{T}$.[20] In the case of MLLR, where the covariance matrix (and hence $\boldsymbol{A}_x$) is assumed to be diagonal, the Kronecker product $\boldsymbol{B}_x^T \otimes \boldsymbol{A}_x$ has a special sparsity structure which allows for a more efficient solution than directly solving (4.15). Specifically, the above matrix equation can then be decomposed into a set of D independent linear system equations, where each equation gives the solution for one row of $\boldsymbol{T}$. For more details, we refer the interested reader to [101].

4.2 Cepstrally Smoothed MLLR for Model Adaptation from Speech Mixtures

Similar to the original MLLR approach, the proposed adaptation method applies an affine transform to the mean parameters of a speaker model. For the multi-pitch tracking framework, however, care must be taken regarding the type of transform applied to the mean parameters. Since one individual GMM is used for each pitch state, a parameter transform should not modify or destroy the harmonicity present in the model. In essence, only the spectral envelopes of a speaker model should be subject to adaptation, while all fine-spectral structure modeled by each pitch-conditional GMM should remain unmodified. Hence, changing vocal tract characteristics and channel conditions can be captured, while still ensuring that each GMM represents its associated pitch. For this reason, we propose an affine transform of the log-spectrum mean vectors which is implicitly constrained in cepstral domain, as described in Section 4.2.1.
A different approach to impose constraints on spectral fine structure has been proposed in [60], where the aim was to develop an adaptive dictionary for nonnegative matrix factorization (NMF) based music transcription. A conceptually related approach has been proposed in [117], where a vocal tract length normalization of cepstral features is achieved by estimating a constrained transform in log-domain using MLLR, i.e. cepstral features are transformed to log-domain where then a tridiagonal transformation matrix is applied.

4.2.1 Parameter Transform with Implicit Cepstral Smoothing

For simplicity, we assume that the mean parameters of all GMMs associated with speaker model k are subject to the same transform, i.e. we have full parameter tying across all GMMs and their components. However, the proposed approach can as well be extended to the case where distinct transforms are applied to subsets of GMMs. We propose the following transform for the mean of speaker k, state x_k and component m_k:

$$\hat{\boldsymbol{\mu}}_{k,x_k}^{m_k} = \boldsymbol{W}\left(\tilde{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{\mu}_{k,x_k}^{m_k} + \tilde{\boldsymbol{b}}\right) = \boldsymbol{W}\tilde{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{\mu}_{k,x_k}^{m_k} + \boldsymbol{W}\tilde{\boldsymbol{b}}, \tag{4.16}$$

where matrix $\boldsymbol{W}$ denotes the $D \times D$ discrete cosine transform (DCT) (type I) matrix:

$$W_{i,j} = \begin{cases} \frac{1}{\sqrt{2D-2}} \cos\left(\frac{\pi}{D-1}(i-1)(j-1)\right) & \text{if } j \in \{1, D\}, \\ \frac{2}{\sqrt{2D-2}} \cos\left(\frac{\pi}{D-1}(i-1)(j-1)\right) & \text{otherwise,} \end{cases} \tag{4.17}$$

[20] Using the Kronecker product, we are able to re-express a product of three matrices $\boldsymbol{ATB}$ as $\text{vec}(\boldsymbol{ATB}) = (\boldsymbol{B}^T \otimes \boldsymbol{A})\text{vec}(\boldsymbol{T})$ [116].

which essentially maps a mean vector $\boldsymbol{\mu}$ from log-spectral to cepstral domain.[21] The cepstral representation is then subject to an affine transform, i.e. we multiply matrix $\tilde{\boldsymbol{T}}$ and add a bias vector $\tilde{\boldsymbol{b}}$. Finally, the result is back-transformed to log-spectral domain. Note that matrix $\boldsymbol{W}$ is an involution, i.e. $\boldsymbol{W}\boldsymbol{W} = \boldsymbol{I}$.
We constrain the structure of the cepstral transform matrix to the form

$$\tilde{\boldsymbol{T}} = \left(\begin{array}{c|c} \boldsymbol{T} & \boldsymbol{0} \\ \hline \boldsymbol{0}^T & \boldsymbol{I} \end{array}\right), \tag{4.18}$$

where we denote by $\boldsymbol{T}$ a $C \times C$ submatrix, $\boldsymbol{I}$ is the $(D-C) \times (D-C)$ identity matrix and $\boldsymbol{0}$ the $C \times (D-C)$ zero matrix. Similarly, we constrain the bias vector to

$$\tilde{\boldsymbol{b}} = \left(\begin{array}{c} \boldsymbol{b} \\ \hline \boldsymbol{0} \end{array}\right), \tag{4.19}$$

where $\boldsymbol{b}$ is a vector with C rows and correspondingly $\boldsymbol{0}$ is a zero vector with $D-C$ rows. As a result, only the first C low-order coefficients of the cepstral representation of $\boldsymbol{\mu}$ are subject to the affine transform defined by $\boldsymbol{T}$ and $\boldsymbol{b}$. For an appropriate choice of C, only those coefficients that represent the spectral envelope of $\boldsymbol{\mu}$ are affected by the transform, while the remaining higher-order cepstral coefficients, which carry the information on the fine-spectral structure, are kept unmodified. For $C = 1$, the resulting transform corresponds to a simple scaling and translation of the first cepstral coefficient. The translation operation itself is enough to modify the gain of the mean parameter by a constant factor, i.e. setting $C = 1$ is sufficient to perform gain adaptation.[22] Higher values of C allow to adapt the shape of the spectral envelope as well. For $C = D$, no constraints are imposed on the transform, and the method is conceptually equivalent to MLLR (apart from the fact that adaptation parameters are defined in cepstral domain). Note that through the use of parameter tying, the spectral envelope observed at a time frame can be used to adapt *all* GMMs of a speaker model, and not just the single GMM that corresponds to the pitch of the observed frame. The proposed transform of a speaker model is thus fully specified by parameters $\boldsymbol{T}$ and $\boldsymbol{b}$. Given a mixture of speakers, our goal is now to adapt all involved speaker models towards the observed speech characteristics. The number of parameters to be adapted per speaker model is $C^2 + C$. For small amounts of adaptation data, constraining the size of C and thus the flexibility of the transform can help to avoid overfitting. We refer to this method as *cepstrally smoothed maximum likelihood linear regression* (csMLLR).

4.2.2 A General EM Algorithm for MLLR-Based Model Adaptation from Speech Mixtures

Here, we derive update equations to learn transformation parameters for each individual speaker model, $\boldsymbol{T}_k$ and $\boldsymbol{b}_k$, given a mixture of speech. The value of C, which determines flexibility of the transform, is assumed to be given.

[21] The GMM-based speaker model introduced in Chapter 2 does not include the spectral energy at zero Hz, as it does not hold any pitch-related information. In this chapter, however, we do assume that $\boldsymbol{\mu}$ additionally contains the bias bin at zero Hz, because it simplifies the application of the DCT transform and all related notation.

[22] Note that the gain is *additive* in log-spectrum/cepstrum domain.

We adapt parameters by maximizing the log-likelihood under the observed speech mixture:

$$\mathrm{LL}(\{\boldsymbol{T}_k\},\{\boldsymbol{b}_k\}) = \ln p(\mathcal{Y}|\{\boldsymbol{T}_k\},\{\boldsymbol{b}_k\}) = \ln \sum_{\mathcal{X}} p(\mathcal{X},\mathcal{Y}|\{\boldsymbol{T}_k\},\{\boldsymbol{b}_k\}), \tag{4.20}$$

where

$$p(\mathcal{X},\mathcal{Y}|\{\boldsymbol{T}_k\},\{\boldsymbol{b}_k\}) = \prod_{k=1}^{K}\left[p(x_k^{(1)})\prod_{t=2}^{T} p(x_k^{(t)}|x_k^{(t-1)})\right]\prod_{t=1}^{T} p(\boldsymbol{y}^{(t)}|\{x_k^{(t)}\},\{\boldsymbol{T}_k\},\{\boldsymbol{b}_k\}) \tag{4.21}$$

is the joint distribution of all observed data and hidden variables of an FHMM with K Markov chains. The conditional dependency on the transformation parameters is made explicit. The distribution of the observation at one time instance given the hidden pitch states is

$$\begin{aligned} p(\boldsymbol{y}^{(t)}|\{x_k^{(t)}\},\{\boldsymbol{T}_k\},\{\boldsymbol{b}_k\}) &= \int\!\cdots\!\int p(\boldsymbol{y}^{(t)}|\{\boldsymbol{s}_k^{(t)}\})\prod_{k=1}^{K} p(\boldsymbol{s}_k^{(t)}|x_k^{(t)},\boldsymbol{T}_k,\boldsymbol{b}_k)d\boldsymbol{s}_1^{(t)}\cdots d\boldsymbol{s}_K^{(t)} \\ &= \sum_{\{m_k\}}\prod_{k=1}^{K}\alpha_{k,x_k}^{m_k}\int\!\cdots\!\int p(\boldsymbol{y}^{(t)}|\{\boldsymbol{s}_k^{(t)}\}) \\ &\quad\times\prod_{k=1}^{K} p(\boldsymbol{s}_k^{(t)}|x_k^{(t)},m_k,\boldsymbol{T}_k,\boldsymbol{b}_k)d\boldsymbol{s}_1^{(t)}\cdots d\boldsymbol{s}_K^{(t)}, \end{aligned} \tag{4.22}$$

where no explicit assumption has been made on the interaction model. The k^{th} pitch-conditional single-speaker model is assumed to be a general mixture model of the form

$$p(\boldsymbol{s}_k^{(t)}|x_k^{(t)},\boldsymbol{T}_k,\boldsymbol{b}_k) = \sum_{m_k}\alpha_{k,x_k}^{m_k} p(\boldsymbol{s}_k^{(t)}|x_k^{(t)},m_k,\boldsymbol{T}_k,\boldsymbol{b}_k). \tag{4.23}$$

It is difficult to maximize the log-likelihood in (4.20) directly. Instead, we make use of the EM approach, similar to the principles described in Section 4.1. We systematically apply Jensen's inequality to construct the following sequence of variational lower bounds on the LL in (4.20):

$$\mathrm{LL} \geq \sum_{\mathcal{X}} q(\mathcal{X}) \ln p(\mathcal{X}, \mathcal{Y} | \{\boldsymbol{T}_k\}, \{\boldsymbol{b}_k\}) + \text{const} \tag{4.24}$$

$$= \sum_{\mathcal{X}} q(\mathcal{X}) \sum_{t=1}^{T} \ln p(\boldsymbol{y}^{(t)} | \{x_k^{(t)}\}, \{\boldsymbol{T}_k\}, \{\boldsymbol{b}_k\}) + \text{const} \tag{4.25}$$

$$\geq \sum_{\mathcal{X}} q(\mathcal{X}) \sum_{t=1}^{T} \sum_{\{m_k\}} q(\{m_k\}) \times \ln \int\!\cdots\!\int p(\boldsymbol{y}^{(t)} | \{\boldsymbol{s}_k^{(t)}\}) \prod_{k=1}^{K} p(\boldsymbol{s}_k^{(t)} | x_k^{(t)}, m_k, \boldsymbol{T}_k, \boldsymbol{b}_k) d\boldsymbol{s}_1^{(t)} \cdots d\boldsymbol{s}_K^{(t)} + \text{const} \tag{4.26}$$

$$\geq \sum_{\mathcal{X}} q(\mathcal{X}) \sum_{t=1}^{T} \sum_{\{m_k\}} q(\{m_k\}) \times \int\!\cdots\!\int q(\{\boldsymbol{s}_k^{(t)}\}) \sum_{k=1}^{K} \ln p(\boldsymbol{s}_k^{(t)} | x_k^{(t)}, m_k, \boldsymbol{T}_k, \boldsymbol{b}_k) d\boldsymbol{s}_1^{(t)} \cdots d\boldsymbol{s}_K^{(t)} + \text{const}, \tag{4.27}$$

where const refers to all terms independent of $\{\boldsymbol{T}_k\}$ and $\{\boldsymbol{b}_k\}$. Note that this lower bound is valid for an arbitrary choice of the variational distributions $q(\mathcal{X})$, $q(\{m_k\})$ and $q(\{\boldsymbol{s}_k^{(t)}\})$. Starting with an initial guess for the adaptation parameters, a local maximum of (4.20) can be found using the EM algorithm. During the E-Step, the variational distributions are set such that the lower bound is tight at the current parameter estimate. During the subsequent M-Step, the lower bound is maximized with respect to the parameters - any increase of the lower bound guarantees an increase in LL [113, 114]. Both steps are repeated until no further increase in LL is achieved. In our case, the two steps can be formally written as:

E-Step: Set the variational distributions according to:

$$q(\{x_k^{(t)}\}) = \sum_{\mathcal{X} \setminus \{x_k^{(t)}\}} p(\mathcal{X} | \mathcal{Y}, \{\boldsymbol{T}_k^{(old)}\}, \{\boldsymbol{b}_k^{(old)}\}), \tag{4.28}$$

$$q(\{m_k\}) = p(\{m_k\} | \boldsymbol{y}^{(t)}, \{x_k^{(t)}\}, \{\boldsymbol{T}_k^{(old)}\}, \{\boldsymbol{b}_k^{(old)}\}), \tag{4.29}$$

$$q(\{\boldsymbol{s}_k^{(t)}\}) = p(\{\boldsymbol{s}_k^{(t)}\} | \boldsymbol{y}^{(t)}, \{x_k^{(t)}\}, \{m_k\}, \{\boldsymbol{T}_k^{(old)}\}, \{\boldsymbol{b}_k^{(old)}\}). \tag{4.30}$$

M-Step: For each speaker k, update the parameters according to

$$\{\boldsymbol{T}_k, \boldsymbol{b}_k\} = \arg\max_{\{\boldsymbol{T}, \boldsymbol{b}\}} Q_k(\boldsymbol{T}, \boldsymbol{b}) = \arg\max_{\{\boldsymbol{T}, \boldsymbol{b}\}} \sum_{t, \{x_k^{(t)}\}, \{m_k\}} \gamma_t(\{x_k^{(t)}\}, \{m_k\}) \mathrm{E}_{\{\boldsymbol{s}_k^{(t)}\}} \left\{ \ln p(\boldsymbol{s}_k^{(t)} | x_k^{(t)}, m_k, \boldsymbol{T}, \boldsymbol{b}) \right\}, \tag{4.31}$$

where we introduced the shorthand

$$\gamma_t(\{x_k\},\{m_k\}) = p(\{x_k^{(t)}\},\{m_k\},|\mathcal{Y},\{\boldsymbol{T}_k^{(old)}\},\{\boldsymbol{b}_k^{(old)}\}) = q(\{x_k^{(t)}\})q(\{m_k\}) \tag{4.32}$$

to denote the posterior of states and components obtained in the previous E-Step (Equations (4.28) and (4.29)). The objective for the M-Step (the auxiliary function) was obtained by plugging in Equations (4.28), (4.29) and (4.30) into (4.27). The expectation $\mathrm{E}_{\{\boldsymbol{s}_k^{(t)}\}}\{\cdot\}$ is with respect to the distribution (4.30). Note that the calculation of Equations (4.28) and (4.29) during the E-Step is equivalent to the E-Step for exact parameter learning in FHMMs [78]. Specifically, (4.28) represents the marginal posterior, which can be obtained using the forward-backward algorithm on FHMMs as proposed in [78]. For the special case of one speaker only, this resembles the E-Step applied during HMM parameter learning [115]. Unfortunately, the exact forward-backward algorithm for FHMMs is intractable. We discuss techniques for an approximate E-Step further below. During the M-Step, the adaptation parameters can be optimized independently for each speaker. The occurrence of the hidden single-speaker spectrum $\boldsymbol{s}_k^{(t)}$ has been replaced by its conditional expected value.

So far, we have not made an explicit assumption about the specific choice of the interaction model, nor on how exactly the transformation parameters $\{\boldsymbol{T}_k, \boldsymbol{b}_k\}$ enter the single-speaker model. A related algorithmic framework has been proposed by Rose et al. [81], where the goal was to estimate a GMM-based speech model from noisy speech data, assuming that the noise model is known a priori and that individual time frames are statistically independent. In contrast to their work, we employ the MLLR framework to adapt existing speaker models.

4.2.3 An EM Algorithm for Cepstrally-Smoothed MLLR-Based Model Adaptation from Speech Mixtures Using the MIXMAX Interaction Model

Based on the general EM framework presented in the previous section, here we introduce the MIXMAX interaction model and the transformation for the GMM-based speaker model defined in (4.16), and derive the update equations for the M-Step of the EM algorithm. Under this assumption, each mixture component is of the form

$$p(\boldsymbol{s}_k^{(t)}|x_k^{(t)}, m_k, \boldsymbol{T}_k, \boldsymbol{b}_k) = \mathcal{N}(\boldsymbol{s}_k^{(t)}|\boldsymbol{W}\tilde{\boldsymbol{T}}_k\boldsymbol{W}\boldsymbol{\mu}_{k,x_k^{(t)}}^{m_k} + \boldsymbol{W}\tilde{\boldsymbol{b}}_k, \boldsymbol{\Sigma}_{k,x_k^{(t)}}^{m_k}). \tag{4.33}$$

Consider the auxiliary function $Q_k(\boldsymbol{T}, \boldsymbol{b})$ for speaker k in (4.31). Plugging (4.33) into $Q_k(\boldsymbol{T}, \boldsymbol{b})$ results in

$$Q_k(\boldsymbol{T}, \boldsymbol{b}) = \sum_{t,\{x_k\},\{m_k\}} \gamma_{t,\{x_k\},\{m_k\}} \mathrm{E}_{\{\boldsymbol{s}_k^{(t)}\}}\left\{\ln \mathcal{N}(\boldsymbol{s}_k^{(t)}|\boldsymbol{W}\tilde{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{\mu}_{k,x_k}^{m_k} + \boldsymbol{W}\tilde{\boldsymbol{b}}, \boldsymbol{\Sigma}_{k,x_k}^{m_k})\right\}. \tag{4.34}$$

As $Q_k(\cdot,\cdot)$ is jointly concave in $\boldsymbol{T}$ and $\boldsymbol{b}$, a global optimum can be obtained by setting the derivative to zero [118]. This leads to two conditions:

i) $\frac{\partial Q_k(\boldsymbol{T},\boldsymbol{b})}{\partial \boldsymbol{T}} = 0$, and

ii) $\frac{\partial Q_k(\boldsymbol{T},\boldsymbol{b})}{\partial \boldsymbol{b}} = 0$.

To facilitate the derivation of $Q_k(\cdot,\cdot)$, we define the following submatrices of the cosine transform matrix:

$$\hat{\boldsymbol{W}} = \boldsymbol{W}_{1:D,1:C}, \tag{4.35}$$
$$\check{\boldsymbol{W}} = \boldsymbol{W}_{1:C,1:D}, \text{ and} \tag{4.36}$$
$$\bar{\boldsymbol{W}} = \boldsymbol{W}_{1:D,(C+1):D}\boldsymbol{W}_{(C+1):D,1:D}, \tag{4.37}$$

where $\boldsymbol{W}_{1:D,(C+1):D}$ denotes the matrix containing the first D rows and the $(C+1)^{\text{th}}$ to the D^{th} column of $\boldsymbol{W}$. This way, we can re-express the linear transform as well as the bias vector in terms of the parameters $\boldsymbol{T}$ and $\boldsymbol{b}$ subject to optimization as

$$\boldsymbol{W}\tilde{\boldsymbol{T}}\boldsymbol{W} = \hat{\boldsymbol{W}}\boldsymbol{T}\check{\boldsymbol{W}} + \bar{\boldsymbol{W}}, \text{ and} \tag{4.38}$$
$$\boldsymbol{W}\tilde{\boldsymbol{b}} = \hat{\boldsymbol{W}}\boldsymbol{b}. \tag{4.39}$$

Condition i): Setting $\frac{\partial Q(\boldsymbol{T},\boldsymbol{b})}{\partial \boldsymbol{T}} = 0$, it follows from standard matrix calculus [119] (see Appendix B for a derivation) that

$$\sum_{x_k,m_k} \gamma_{x_k,m_k}\boldsymbol{A}_{x_k,m_k}\boldsymbol{T}\boldsymbol{B}_{x_k,m_k} = \boldsymbol{C}, \tag{4.40}$$

where

$$\gamma_{x_k,m_k} = \sum_{\{x_j\}_{j\neq k}} \sum_{\{m_j\}_{j\neq k}} \sum_t \gamma_{t,\{x_j\},\{m_j\}}, \tag{4.41}$$
$$\boldsymbol{A}_{x_k,m_k} = \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1}\hat{\boldsymbol{W}}, \tag{4.42}$$
$$\boldsymbol{B}_{x_k,m_k} = \check{\boldsymbol{W}}\boldsymbol{\mu}_{k,x_k}^{m_k}{\boldsymbol{\mu}_{k,x_k}^{m_k}}^T\check{\boldsymbol{W}}^T, \tag{4.43}$$
$$\boldsymbol{C} = \sum_{x_k,m_k} \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1}\left(\mathrm{E}\{\boldsymbol{s}_k|x_k,m_k\} - \gamma_{x_k,m_k}\left(\bar{\boldsymbol{W}}\boldsymbol{\mu}_{k,x_k}^{m_k} + \hat{\boldsymbol{W}}\boldsymbol{b}\right)\right){\boldsymbol{\mu}_{k,x_k}^{m_k}}^T\check{\boldsymbol{W}}^T. \tag{4.44}$$

In Equation (4.41), the marginal posterior obtained during the E-Step is accumulated for all time frames and all states of concurrent speakers $j \neq k$. Similarly, the state-conditional expected single-speaker spectrum of speaker k is obtained by weighted accumulation:

$$\mathrm{E}\{\boldsymbol{s}_k|\mathcal{Y},x_k,m_k\} = \sum_{\{x_j\}_{j\neq k}} \sum_{\{m_j\}_{j\neq k}} \sum_t \gamma_{t,\{x_j\},\{m_j\}}\mathrm{E}\left\{\boldsymbol{s}_k|\boldsymbol{y}^{(t)},\{x_k\},\{m_k\}\right\}, \tag{4.45}$$

where $\mathrm{E}\left\{\boldsymbol{s}_k|\boldsymbol{y}^{(t)},\{x_k\},\{m_k\}\right\}$ is the expected single-speaker spectrum conditioned on the observation at time t as well as a pitch and component combination. The d^{th} dimension of this expectation is calculated as

$$\mathrm{E}\left\{s_k^d|\boldsymbol{y}^{(t)},\{x_k\},\{m_k\}\right\} = \frac{y_d^{(t)}\Psi_{k,x_k}^{m_k,d} + \left(\mu_{k,x_k}^{m_k,d} - (\sigma_{k,x_k}^{m_k,d})^2\Psi_{k,x_k}^{m_k,d}\right)\sum_{l\neq k}\Psi_{l,x_l}^{m_l,d}}{\sum_j \Psi_{j,x_j}^{m_j,d}}, \tag{4.46}$$

where

$$\Psi_{k,x_k}^{m_k,d} = \frac{\mathcal{N}(y_d|\theta_{k,x_k}^{m_k,d})}{\Phi(y_d|\theta_{k,x_k}^{m_k,d})} \tag{4.47}$$

is the ratio of the normal density and the cumulative normal distribution of observation y_d. For a derivation of (4.46), we refer the reader to Appendix C. Note that the calculation of the sufficient statistics (4.41) and (4.45) is intractable. We discuss this issue in Section 4.2.4.
Assuming that $\boldsymbol{b}$ is fixed, the matrix equation in (4.40) can be solved in closed form [116] (cf. Equation (4.14) and (4.15) in Section 4.1) as

$$\mathrm{vec}(\boldsymbol{T}) = \left(\sum_{x_k,m_k} \gamma_{x_k,m_k} \boldsymbol{B}_{x_k,m_k}{}^T \otimes \boldsymbol{A}_{x_k,m_k} \right)^{-1} \mathrm{vec}(\boldsymbol{C}), \tag{4.48}$$

where $\otimes$ denotes the Kronecker product and $\mathrm{vec}(\boldsymbol{T})$ is a vector obtained by sequentially stacking the rows of $\boldsymbol{T}$.

Condition ii): Setting $\frac{\partial Q(\boldsymbol{T},\boldsymbol{b})}{\partial \boldsymbol{b}} = 0$, we obtain

$$\begin{aligned} \boldsymbol{b} = &\left(\sum_{x_k,m_k} \gamma_{x_k,m_k} \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \hat{\boldsymbol{W}} \right)^{-1} \\ &\times \sum_{x_k,m_k} \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \left(\mathrm{E}\{\boldsymbol{s}_k|\mathcal{Y}, x_k, m_k\} - \gamma_{x_k,m_k} \boldsymbol{W}\tilde{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{\mu}_{k,x_k}^{m_k} \right), \end{aligned} \tag{4.49}$$

and we can solve for $\boldsymbol{b}$ assuming that $\boldsymbol{T}$ is fixed.
Equations (4.48) and (4.49) are applied iteratively during the M-Step to update the value of one variable while holding the other fixed. This type of block-coordinate ascent method is guaranteed to converge to the global optimum, as the objective (4.34) is jointly concave in $\boldsymbol{T}$ and $\boldsymbol{b}$.

The EM algorithm is summarized in Algorithm 3. During the E-Step, the unknown single-speaker spectrum of every speaker is inferred, based on the currently available speaker models. During the M-Step, the expected single-speaker spectrum is used as a surrogate to the true single-speaker spectrum, and model parameters are updated according to cepstrally smoothed maximum likelihood linear regression (csMLLR). For the special case where only one speaker model is adapted from single speech, the E-Step is not necessary and the true single-speaker spectrum can be used in place of the expected speaker spectrum.
As an alternative to csMLLR, the standard MLLR update equations can be used as well during the M-Step. Although csMLLR is conceptually equivalent to MLLR when setting $C = D$, the standard MLLR update equations allow for a computationally more efficient solution (as discussed at the end of Section 4.1).

Input: Set of observations $\mathcal{Y}$
Output: Adaptation parameters $\{\boldsymbol{T}_k\}$ and $\{\boldsymbol{b}_k\}$ for each speaker k
Initialization: Initialize adaptation parameters
while *not converged* **do**
  E-Step:
  Compute $p(\{x_k^{(t)}\}|\mathcal{Y}, \{\boldsymbol{T}_k^{(old)}\}, \{\boldsymbol{b}_k^{(old)}\})$ using forward-backward algorithm
  $\gamma_{t,\{x_k\},\{m_k\}} \leftarrow p(\{m_k\}|\{x_k^{(t)}\}, \mathcal{Y}, \{\boldsymbol{T}_k^{(old)}\}, \{\boldsymbol{b}_k^{(old)}\}) p(\{x_k^{(t)}\}|\mathcal{Y}, \{\boldsymbol{T}_k^{(old)}\}, \{\boldsymbol{b}_k^{(old)}\})$
  foreach $k \in \{1, \ldots, K\}$ **do**
    $\gamma_{x_k,m_k} \leftarrow \sum_{\{x_j\}_{j\neq k}} \sum_{\{m_j\}_{j\neq k}} \sum_t \gamma_{t,\{x_j\},\{m_j\}}$
    foreach $d \in \{1, \ldots, D\}$ **do**
      $\mathrm{E}\{s_k^d|\boldsymbol{y}^{(t)}, \{x_k\}, \{m_k\}\} \leftarrow \frac{y_d^{(t)} \Psi_{k,x_k}^{m_k,d} + (\mu_{k,x_k}^{m_k,d} - (\sigma_{k,x_k}^{m_k,d})^2 \Psi_{k,x_k}^{m_k,d}) \sum_{l\neq k} \Psi_{l,x_l}^{m_l,d}}{\sum_j \Psi_{j,x_j}^{m_j,d}}$
    end
    $\mathrm{E}\{\boldsymbol{s}_k|\mathcal{Y}, x_k, m_k\} \leftarrow \sum_{\{x_j\}_{j\neq k}} \sum_{\{m_j\}_{j\neq k}} \sum_t \gamma_{t,\{x_j\},\{m_j\}} \mathrm{E}\{\boldsymbol{s}_k|\boldsymbol{y}^{(t)}, \{x_k\}, \{m_k\}\}$
  end
  M-Step:
  foreach $k \in \{1, \ldots, K\}$ **do**
    foreach x_k, m_k **do**
      $\boldsymbol{A}_{x_k,m_k} \leftarrow \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \hat{\boldsymbol{W}}$
      $\boldsymbol{B}_{x_k,m_k} \leftarrow \check{\boldsymbol{W}} \boldsymbol{\mu}_{k,x_k}^{m_k} {\boldsymbol{\mu}_{k,x_k}^{m_k}}^T \check{\boldsymbol{W}}^T$
    end
    $\boldsymbol{C}_1 \leftarrow \sum_{x_k,m_k} \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \left(\mathrm{E}\{\boldsymbol{s}_k|\mathcal{Y}, x_k, m_k\} - \gamma_{x_k,m_k} \bar{\boldsymbol{W}} \boldsymbol{\mu}_{k,x_k}^{m_k}\right) {\boldsymbol{\mu}_{k,x_k}^{m_k}}^T \check{\boldsymbol{W}}^T$
    $\boldsymbol{C}_2 \leftarrow -\sum_{x_k,m_k} \gamma_{x_k,m_k} \left(\check{\boldsymbol{W}} \boldsymbol{\mu}_{k,x_k}^{m_k}\right) \otimes \left(\hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \hat{\boldsymbol{W}}\right)$
    $\boldsymbol{M}_1 \leftarrow \sum_{x_k,m_k} \gamma_{x_k,m_k} {\boldsymbol{B}_{x_k,m_k}}^T \otimes \boldsymbol{A}_{x_k,m_k}$
    $\boldsymbol{M}_2 \leftarrow -\sum_{x_k,m_k} \gamma_{x_k,m_k} \left((\check{\boldsymbol{W}} \boldsymbol{\mu}_{k,x_k}^{m_k}) \otimes \hat{\boldsymbol{W}}^T\right) {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \hat{\boldsymbol{W}}$
    $\boldsymbol{M}_3 \leftarrow \sum_{x_k,m_k} \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \left(\mathrm{E}\{\boldsymbol{s}_k|\mathcal{Y}, x_k, m_k\} - \gamma_{x_k,m_k} \bar{\boldsymbol{W}} \boldsymbol{\mu}_{k,x_k}^{m_k}\right)$
    $\boldsymbol{M}_4 \leftarrow \sum_{x_k,m_k} \gamma_{x_k,m_k} \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \hat{\boldsymbol{W}}$
    Block-Coordinate Ascent:
    $\boldsymbol{b}_k \leftarrow \boldsymbol{0}$
    while *not converged* **do**
      $\mathrm{vec}(\boldsymbol{C}) \leftarrow \mathrm{vec}(\boldsymbol{C}_1) + \boldsymbol{C}_2 \boldsymbol{b}_k$
      $\mathrm{vec}(\boldsymbol{T}_k) \leftarrow \boldsymbol{M}_1^{-1} \mathrm{vec}(\boldsymbol{C})$
      $\boldsymbol{b}_k \leftarrow \boldsymbol{M}_4^{-1} \left(\boldsymbol{M}_3 + \boldsymbol{M}_2^T \mathrm{vec}(\boldsymbol{T}_k)\right)$
    end
  end
end

Algorithm 3: Outline of the exact EM algorithm for model adaptation from speech mixtures derived in Section 4.2.3. The calculation of the sufficient statistics during the E-Step is intractable (see Section 4.2.4 for a discussion on possible approximation strategies). Matrices $\boldsymbol{C}_1$, $\boldsymbol{C}_2$ and $\boldsymbol{M}_1, \cdots, \boldsymbol{M}_4$ have been introduced to avoid redundant calculations during the block-coordinate ascent procedure in the M-Step (which is still equivalent to iterative application of Equations (4.48) and (4.49)).

4.2.4 An Approximate Sparse EM Algorithm for Model Adaptation

The forward-backward algorithm as well as the calculation of sufficient statistics during the E-Step of the exact algorithm outlined above is intractable. To make the proposed algorithm tractable, we make again use of the fast pruning scheme developed for the MIX-MAX interaction model (see Chapter 3). During the E-Step, the observation likelihoods are evaluated only for a plausible subset (the R-best set) of state combinations and then passed to a sparse variant of the forward-backward algorithm, which is based on the same ideas as described in 3.1.4. The resulting state posterior $\gamma_{t,\{x_k\},\{m_k\}}$ is sparse as well, such that the computational complexity of the subsequent calculation of sufficient statistics is directly controlled by R.
This pruning approach can be seen as a combination of the "sparse"- and "winner-take-all" variant of EM proposed by Neal and Hinton [120]. In essence, the state posterior is approximated by a more tractable, i.e. sparse, variant. This principle is strongly related to an alternative approach based on variational approximations [78], where the state posterior is constrained to be from a restricted set of functions (e.g. the set of fully factorized functions). In any case, due to the approximative nature of the posterior, full maximization of the lower bound and hence convergence to a (local) maximum of LL is not guaranteed anymore. Moreover, as the objective of EM is to increase a lower bound on LL, a bad approximation of the posterior might even result in a decreasing LL. The proposed pruning scheme allows for a natural tradeoff between approximation quality and computational complexity, which can be controlled by the value of R. Another beneficial aspect is that the posterior is approximated in a single pass of the forward-backward algorithm. This is in contrast to variational methods, where the posterior is obtained by an iterative scheme.

4.3 Experiments

We evaluate the capabilities of model adaptation for a range of scenarios. First, in Section 4.3.1, we assume that single-speaker recordings are available for SI model adaptation, and we evaluate the performance of adapted models when used for multi-pitch tracking. Next, in Section 4.3.2, we evaluate the performance of gain adaptation given speech mixtures, which can be done using the csMLLR algorithm with the cepstral smoothing parameter set to $C = 1$. In Section 4.3.3, we demonstrate results for model adaptation from speech mixtures. Specifically, we use speech mixtures recorded in a real office environment, where the distance between loudspeakers and the microphone is about 2 m. In this scenario, there is a clear mismatch between SD models trained on clean speech and the test conditions. We show results obtained by adapting the SD models of both speakers on the test mixture itself – a case commonly referred to as *self adaptation*.
Finally, we note that the most general scenario where SI models are adapted on a speech mixture remains a difficult problem. One apparent problem with this scenario is the symmetry problem. When initializing each speaker model to identical SI models, the EM algorithm described in Section 4.2.3 will produce identical transformation parameters for all models, i.e. it cannot converge to individual speaker models. One possible direction to tackle this problem can be based on the identification of speech segments where only

single speech is present.[23] These segments can be used to slightly pre-adapt an SI model slightly into the direction of the speaker. This should provide the necessary incentive to the EM algorithm to produce distinct models, each modelling one unique speaker from the mixture. However, this scenario remains an issue for future research.

4.3.1 Model Adaptation from Single-Speaker Data

We evaluated the multi-pitch tracking performance using SI models adapted on single-speaker utterances. Only the GMMs of the speaker model were subject to adaptation, and the parameters of the Markov chain remained unchanged. Experiments were performed using a subset of the GRID database. We used 4 (out of 6) test speakers from the GRID corpus (two male and two female) with three test utterances each, which results in 6 speaker pairs with 9 test mixtures each (giving a total of 54 test mixtures). For adaptation, we used a set of 40 independent utterances per test speaker from the development set (cf. Section 2.5.1). Note that no additional reference pitch labels are required for adaptation. SI models were adapted using different amounts of development data, and using either MLLR or csMLLR with different settings of the smoothing parameter C. For each instance, the multi-pitch tracking performance was evaluated on all 54 test mixtures in terms of $\bar{E}_{Total}$. During multi-pitch tracking, we used speaker models that were adapted to the correct speakers, i.e. using adaptation utterances from the same speakers that are present in the a test mixture.
Performance results are shown in Figure 4.1, together with the performance achieved using either SI or SD models. For small amounts of adaptation data (about 1.8 s), MLLR cannot improve performance over the SI model, while csMLLR achieves better performance for low values of C (strong smoothing). With larger amounts of adaptation data available, performance of MLLR steadily increases and settles at $\bar{E}_{Total} \approx 34$ for 17.5 s of adaptation data – more adaptation data cannot further increase the performance due to the limited expressive power of a single shared affine transform (parameter tying). While strong cepstral smoothing is beneficial for small amounts of adaptation data, it is detrimental for sufficient amounts of adaptation data. However, the results indicate that performance of csMLLR converges to MLLR performance for increasing yet moderate values of C (around $C = 30$). In this case, csMLLR reaches almost the same performance as MLLR while significantly less parameters need to be learned ($C^2 + C$ instead of $D^2 + D$).
For the case of scarce adaptation data (i.e. 1.8 s), the example in Figure 4.2 shows the effect of parameter adaptation on mean vectors of one pitch-conditional GMM (f_0=100 Hz). For low values of C, csMLLR preserves the periodicity inherent in all mean vectors. Without sufficient cepstral smoothing, on the other hand, periodicity is more or less destroyed. In such situations, the constraints imposed by csMLLR avoid overfitting on scarce data. Still, a meaningful transformation can be learned for low values of C. This is shown in Figure 4.3, where the similarity between adapted speaker GMMs and SD GMMs is evaluated in terms of the KL-divergence for a range of relevant pitch states. In all cases, the csMLLR-adapted GMM is more similar to the corresponding SD GMM than the SI GMM, i.e. the KL-divergence is smaller for these cases. In contrast, the KL-divergence of the MLLR-adapted model GMMs is much larger, i.e. the GMMs are less similar to the corresponding SD GMMs than the SI GMMs.

[23] This is related to the concept of *usable speech* [121].

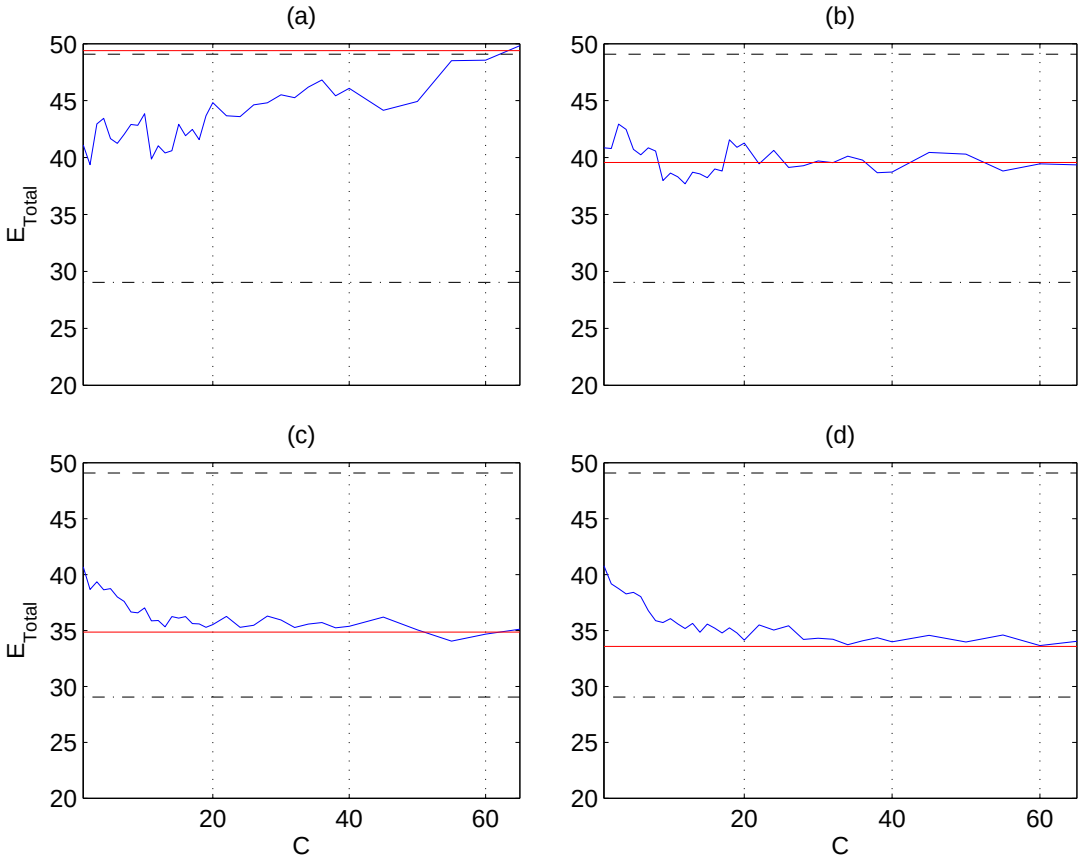

Figure 4.1: Results for single-speaker adaptation. SI models where adapted on single-speaker data, using MLLR or csMLLR for different settings of C. Dashed solid line: SI model performance. Dash-dotted line: SD model performance. Blue solid line: csMLLR performance. Red solid line: MLLR performance. A different amount of adaptation data was used: (a) 1.8 s, (b) 3.5 s, (c) 8.7 s, (d) 17.5 s. For each setting, $\bar{E}_{Total}$ was evaluated on 6 speaker pairs with 9 test mixtures each.

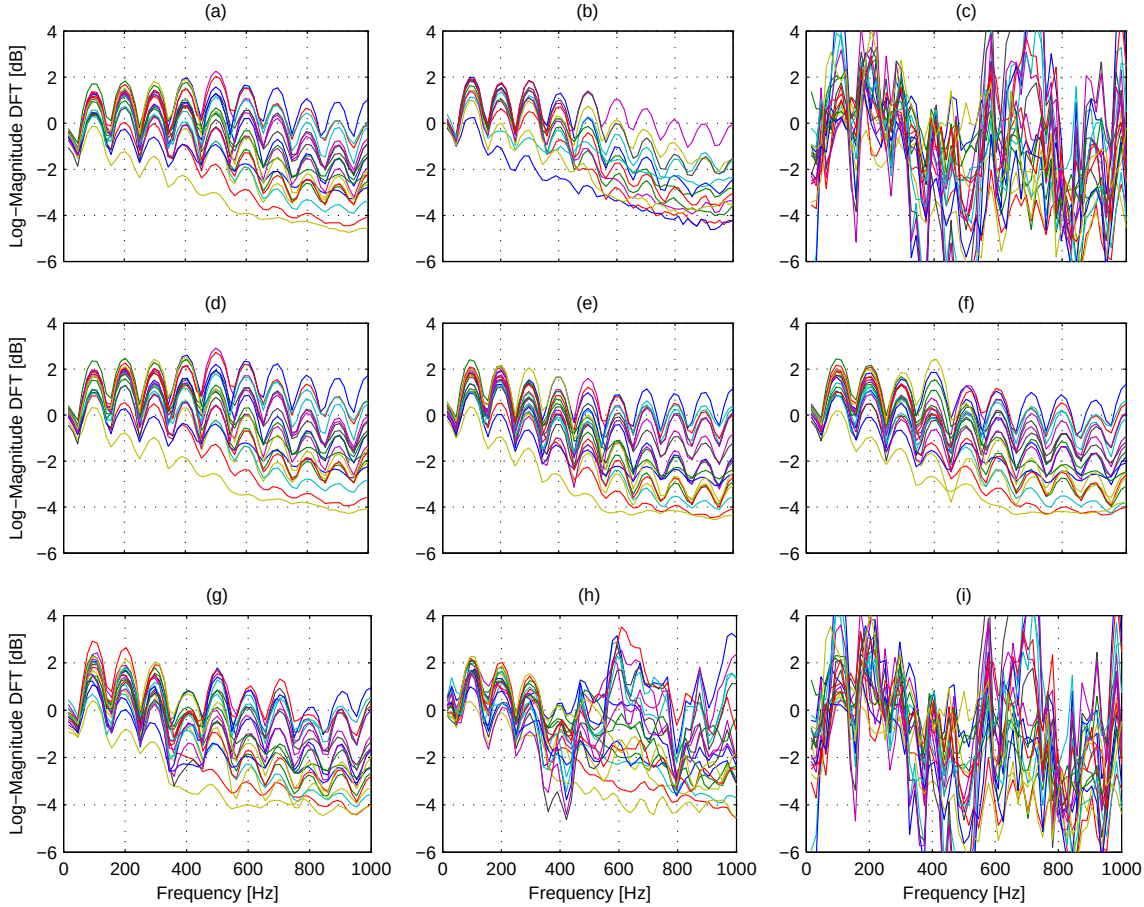

Figure 4.2: Mean vectors of GMM corresponding to a fundamental frequency of 100 *Hz. The mean vector of each mixture component is shown with a separate color. (a) SI model. (b) SD model of a male speaker. (c) Result of standard MLLR applied on SI model. About* 1.8 *s of speech were used for adaptation. (d)-(i) Result of MLLR with cepstral smoothing (csMLLR) applied on SI model; the six plots were obtained by setting* C *to 1,5,10,20,40 and 65, respectively. The same adaptation data was used as for standard MLLR.*

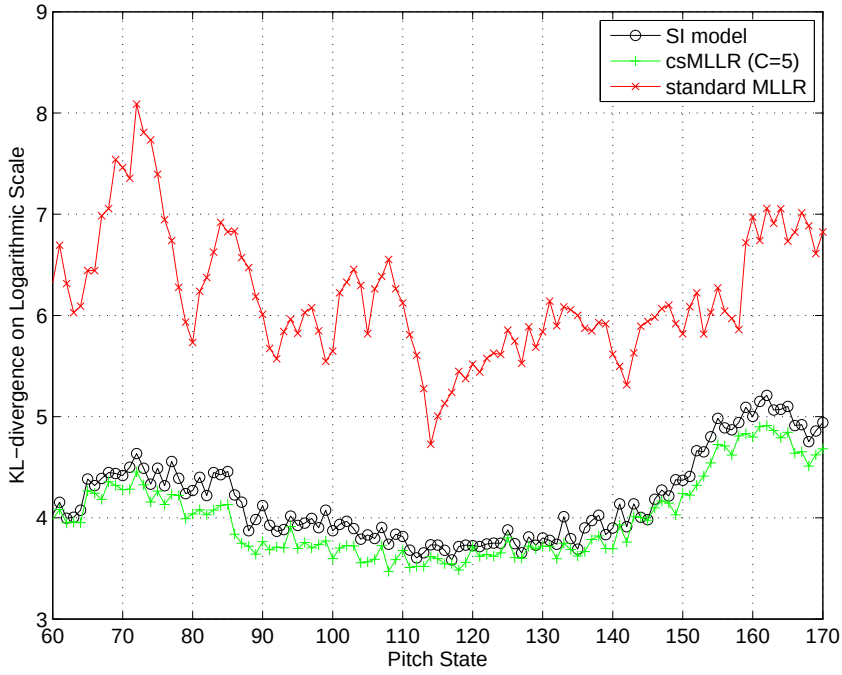

Figure 4.3: For a range of relevant pitch states, the similarity between two pitch-conditional GMMs is shown in terms of their KL-divergence (on logarithmic scale). Black line: KL-divergence between SI model and SD model. Red line: KL-divergence between MLLR-adapted model and SD model. Green line: KL-divergence between csMLLR-adapted model and SD model. About 1.8 *s adaptation data were used (this result is obtained from the same adapted speaker model as results shown in Figure 4.2.) The KL-divergence was approximated using Monte-Carlo sampling with* 10^5 *samples.*

4.3.2 Gain Adaptation

In the empirical evaluation of Chapter 2, we assumed that the gain level (i.e. energy) of each test utterance is the same as for utterances from the training set. In practice, however, this assumption does not hold – and any gain mismatch between training and test conditions results in a degraded tracking performance. In this section, we evaluate the capability of the proposed adaptation framework to compensate gain mismatches between speaker models and the actual conditions in a test mixture composed of two simultaneous speakers. As described in Section 4.2.1, gain adaptation can be performed by setting the spectral smoothness parameter of csMLLR to $C = 1$.
Same as in the experimental setup described in Chapter 2, we used 6 test speakers with 3 test utterances per speaker. Each test mixture was created with a predefined gain difference level Δ from the range $\Delta = \{0, 3, 6, 9, 12, 15, 18\}$ [dB]. For each value of Δ, and each combination of test utterances $s_1[n]$ and $s_2[n]$, a test mixture $y[n]$ was created according to

$$y[n] = 10^{\frac{\Delta}{20}} s_1[n] + s_2[n]. \tag{4.50}$$

This results in a total of 135 test mixtures per gain level (15 speaker pairs with 9 test mixtures each).
For each test mixture, we performed gain adaptation using the approximate csMLLR-based adaptation algorithm, as described in Section 4.2.4. The pruning parameter was set to $R = 1000$, and the EM algorithm was run for 30 iterations (in many cases, no further increase of the lower bound was observed at less iteration numbers). Gain adaptation and subsequent multi-pitch tracking was performed either with SD or SI speaker models. The resulting performance in terms of $\bar{E}_{Total}$ is summarized in Figure 4.4. Without gain adaptation, the error increases significantly with rising gain difference level. With gain adaptation, we reach in all cases almost the same performance as with gain levels known a priori (i.e. perfect knowledge). Using SD models, the tracking performance decreases only moderately with an increasing gain difference level. With SI models, the error first slightly *decreases*, and increases again for high levels of Δ. The reason for the slight decrease is that a moderate gain difference between two speakers serves as a hint for correct speaker assignment, which is otherwise not possible using SI models. Hence, both speakers can be distinguished by their gain level, which results in a lower permutation error $\bar{E}_{Perm}$. If the gain difference is too large, then one speaker is more or less masked by the other, such that the pitch trajectory of the background speaker cannot be estimated reliably.

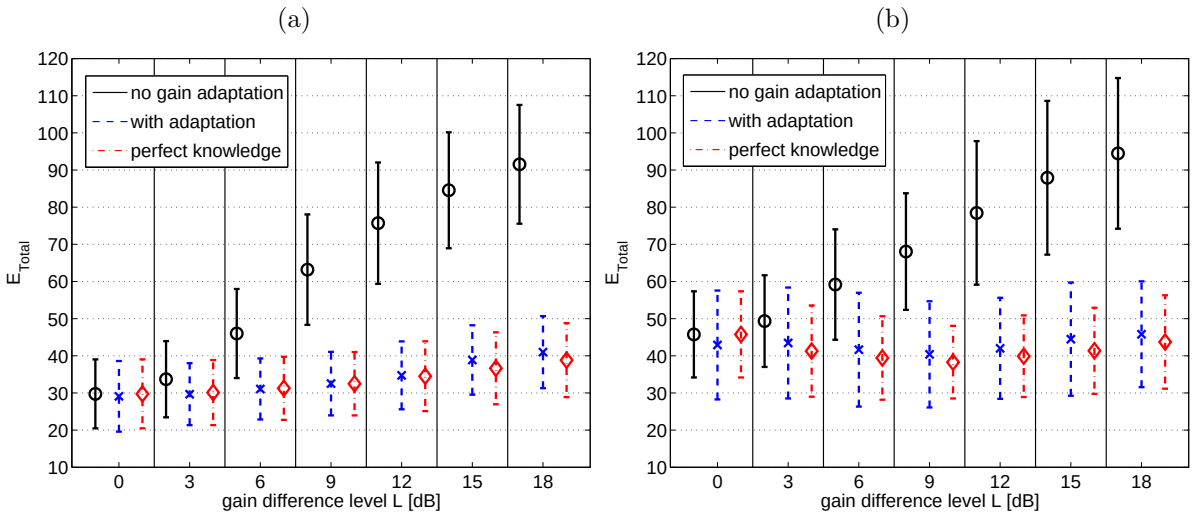

Figure 4.4: Performance results for gain adaptation in terms of $\bar{E}_{Total}$. (a) SD models. (b) SI models. 135 test mixtures were used for each gain difference level Δ, and the corresponding mean and standard deviation of the error is shown for three methods. 'no gain adaptation': Speaker models were used without adaptation. 'with adaptation': gain adaptation was performed prior to multi-pitch tracking using csMLLR. 'perfect knowledge': The speaker models were adapted by the true gain, i.e. the gain factor used to create the test mixture.

4.3.3 Self-Adaptation on Speech Mixtures Recorded in Reverberant Room

Self-adaptation refers to the case were no dedicated adaptation utterances are available, and thus adaptation needs to be performed on the same (short) test mixture on which multi-pitch tracking is to be performed. In the following, we use the proposed framework to perform self-adaptation on mixtures recorded in a real office environment. In such a scenario, the spectral characteristics of each source signal have changed due to multipath propagation or a different microphone transfer function. The mismatch between prior SD models obtained from close-talk microphone recordings and modified spectral characteristics in far-distance recordings results in a deteriorated multi-pitch tracking performance. For this experiment, we used recordings where a set of test utterances from the GRID database was used and played through Yamaha MSP5A loudspeakers.[24] The room where the recordings were performed has the dimensions $6.02 \times 5.32 \times 3\,\mathrm{m}$. One of the walls of the room has a large window, and the floor is covered with a standard carpet. The measured reverberation time (RT_{60}) was $\mathrm{RT}_{60} \approx 500$ ms (no particular effort was made to reduce the reverberation in the room). For each recorded speech mixture, two GRID utterances were played back simultaneously with two loudspeakers positioned at different locations around a circular microphone array (with 0.15 m diameter), from which we selected a single channel. The distance between loudspeakers and the microphone was about 2 m.

We used a total of 27 recorded test mixtures, consisting of 9 test mixtures from three speaker pairs each (female-female, male-female, male-male). For each test mixture, we applied self-adaptation using either csMLLR or MLLR and evaluated the resulting multi-

[24] These recordings have been taken by former members of our lab.

pitch tracking performance. A summary of the results is shown in Figure 4.7, where the performance is additionally compared to the case where (i) no adaptation is performed and (ii) multi-pitch tracking is performed on the equivalent synthetic test mixture.[25] Application of SD models without adaptation works very well when applied to a synthetic mixture, but results in heavily degraded performance when applied to the recorded mixture. Self-adaptation is able to improve the performance, however the optimal choice of smoothing parameter C varies among speaker pairs. Generally, a low value of C works better, as only few data for adaptation is available. Self-adaptation clearly works best for mixtures of a male and female speaker, using csMLLR with $C = 3$. Figures 4.5 and 4.6 show a detailed example for one male-female test mixture. In Figure 4.5, the tracking result on the synthetic mixture of both test utterances is shown. As there is no significant mismatch between SD models and the test condition, multi-pitch estimation works reasonably well. Tracking results with and without self-adaptation on the real recording of the same corresponding test mixture are shown in Figure 4.5.

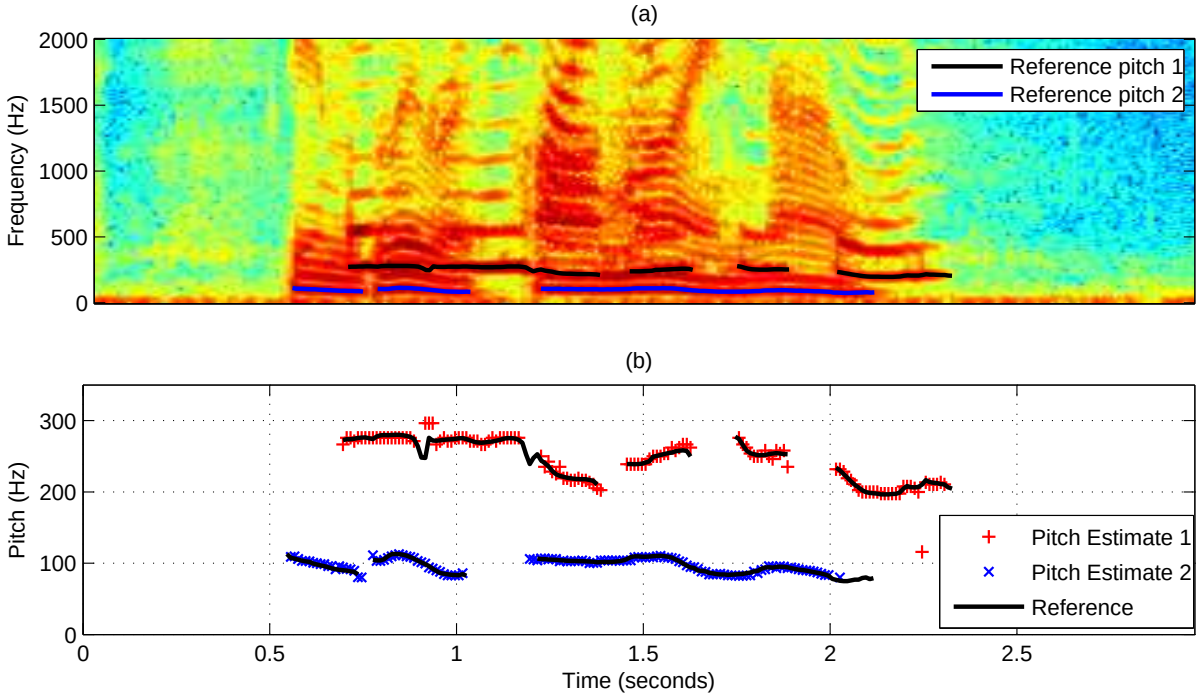

Figure 4.5: Tracking result on synthetic mixture of one male and one female speaker from the GRID corpus (utterance "bbar9n" and "bgbm2s"). (a) Spectrogram of synthetic speech mixture, together with both reference pitch trajectories. (b) Estimated pitch trajectories using SD speaker models. The overall error $\bar{E}_{Total}$ is 12.2.

[25] For each recorded test mixture, a corresponding synthetic test mixture was created by linear superposition of the time-aligned original GRID utterances.

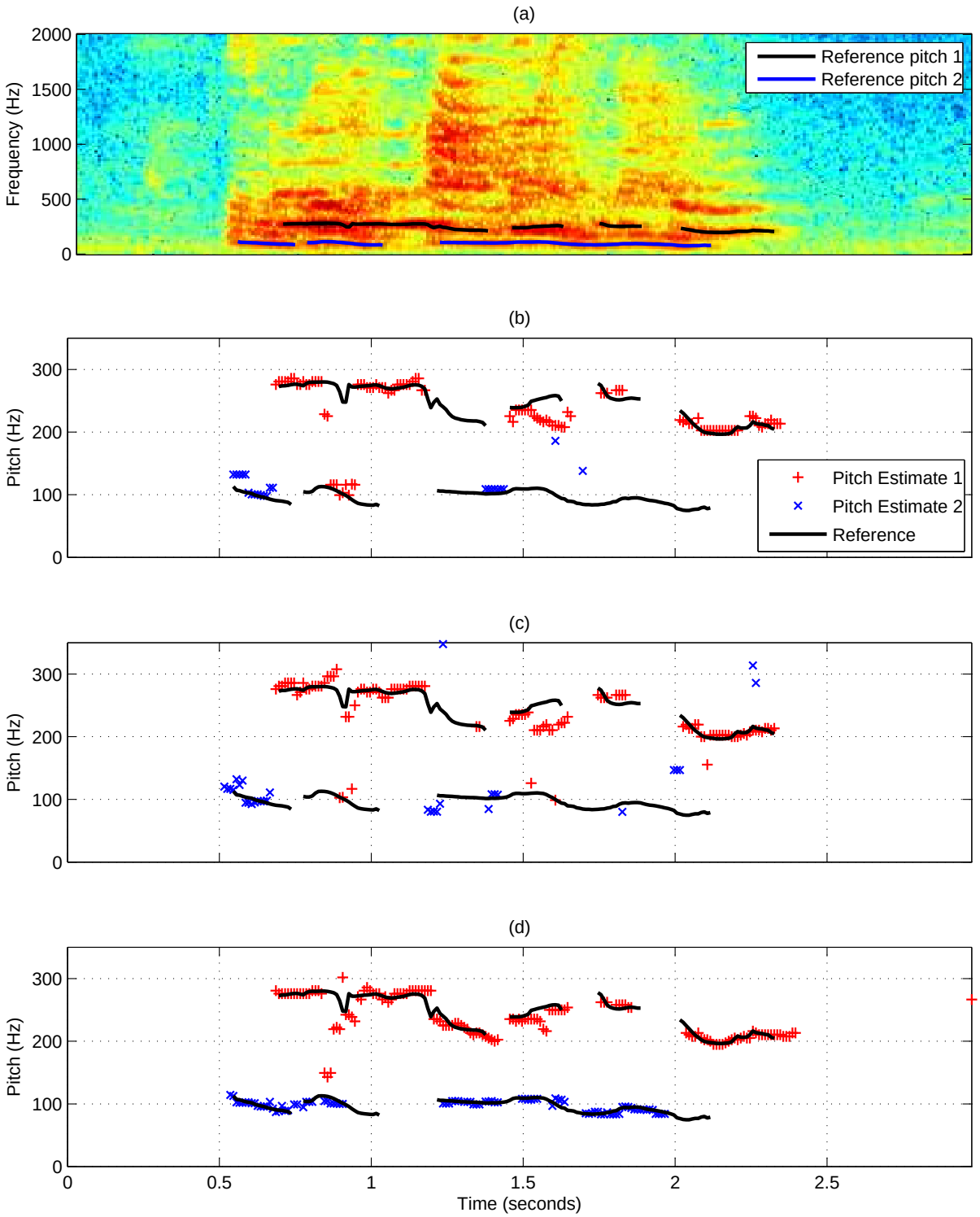

Figure 4.6: Tracking results on real recording with and without self-adaptation. The same test utterances were used as for the synthetic mixture shown in Figure 4.5. (a) Spectrogram of recorded speech mixture. (b) Estimated pitch trajectories using SD speaker models without self-adaptation. (c) Estimated pitch trajectories after self-adaptation with MLLR. (d) Estimated pitch trajectories after self-adaptation with csMLLR ($C = 3$). The overall error $\bar{E}_{Total}$ achieved by the three methods is 65.2, 57.0 and 32.8, respectively.

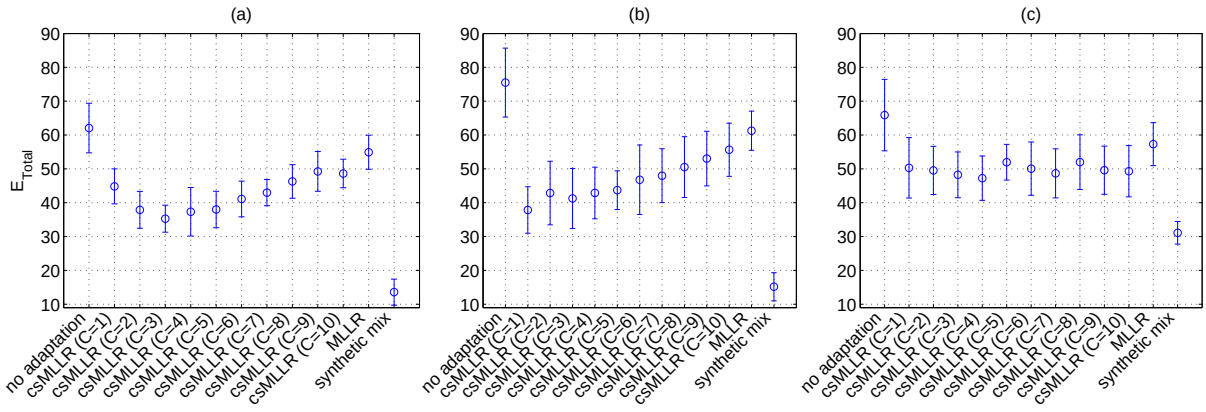

Figure 4.7: Multi-pitch tracking performance in terms of $\bar{E}_{Total}$ after self-adaptation of SD models on real recordings. (a) male-female mixtures. (b) female-female mixtures. (c) male-male mixtures. Self-adaptation and subsequent multi-pitch tracking was performed for each test mixture separately. 9 test mixtures were used per speaker pair, and error bars indicate the corresponding mean and standard deviation of $\bar{E}_{Total}$ for various methods. 'no adaptation': SD models of both speakers were used without adaptation. 'csMLLR': SD models of both speakers were adapted on the test mixture using csMLLR. 'MLLR': SD models of both speakers were adapted on the test mixture using MLLR. 'synthetic mix': SD models of both speakers were used without adaptation and applied for multi-pitch tracking on the synthetic mixture (i.e. no recording in room environment).

5 Conclusions

In this thesis, we developed a probabilistic model-based approach for multi-pitch tracking of speech. The continuous range of relevant pitch values is quantized into a set of discrete pitch states, and the single-speaker spectrum corresponding to each of the pitch states is modelled by a set of GMMs. The dynamic change of pitch states over time is modelled by means of a Markov model. Single-speaker models are combined using the MIXMAX or a linear interaction model. This allows to directly relate the observed speech mixture to the hidden pitch states, while still retaining all probabilistic uncertainty induced by the single-speaker models. The resulting FHMM provides a probabilistic description of the spectrogram of simultaneous speakers, and the sequence of most likely pitch trajectories is obtained by tracking within the model.

The advantage of this model-based approach is the possibility to integrate a priori knowledge about speaker characteristics into the statistical model. Speaker models are trained from speech data prior to multi-pitch tracking. SI models are obtained from a set of independent speakers, while SD models are trained on speech material of the relevant test speakers. In Chapter 2, we tested the performance of both model types and compared the proposed method to a well-known state-of-the-art algorithm. It was shown that SI models perform slightly better than the reference method in terms of estimation accuracy, and that the use of SD models achieves a good assignment of estimated pitch values to their corresponding speakers. The ability for correct speaker assignment, which has been largely ignored in previous literature, is a beneficial property for pitch-based speech segregation and CASA, and can be attributed to the fact that SD models explicitly incorporate the timbral characteristics of a speaker.

Despite the promising results, the approach proposed in Chapter 2 suffers from two practical limitations. First, the computational complexity of exact inference scales exponentially with the number of simultaneous speakers. Second, any mismatch between a priori trained speaker models and testing conditions results in a degraded tracking performance. In Chapter 3, we proposed a method for approximate inference based on computationally efficient upper and lower bounds on the probability of state-combinations, which are then used to select a subset of likely combinations. The number of selected state-combinations can be controlled, which allows for a tradeoff between approximation quality and time

requirements. Experiments performed for mixtures of two and three speakers demonstrated that good tracking results can still be obtained from a small fraction of selected likelihoods.
In Chapter 4, we proposed a modification of the MLLR technique where the adaptation of model parameters is constrained to modifications of the spectral envelope. We showed that this constraint maintains the periodicity information contained in GMMs, and is beneficial for cases where few adaptation data is available. Based on the modified MLLR framework, we proposed an EM algorithm for adaptation of speaker models from speech mixtures. While the exact EM algorithm is intractable, an approximate scheme was proposed using the likelihood pruning framework proposed in Chapter 3, which significantly reduces the computational requirements of the algorithm. We demonstrated tracking results obtained for a distant talking scenario of two speakers which includes room reverberation, and showed significant improvements obtained by self-adaptation of SD models.

The results provided in this thesis indicate some interesting directions for future research. Although the focus of this thesis is restricted to mixtures of speakers, we note that the modularity of the proposed approach allows to use it as well for other challenging scenarios with different types of acoustic (background) sources. E.g., one might think of tracking the pitch of passengers in a car, where the noise of the engine and other acoustic clutter is modelled by one or several separate source-models. Moreover, we note that the spectrogram features used in this thesis can be replaced by other types of features, providing that the chosen interaction model remains valid (e.g., the log-magnitude DFT coefficients can be replaced by the log-magnitude coefficients of a constant-Q transform). In general, whenever good source-models are available for the given acoustic condition, we expect that the proposed framework is likely to produce reasonable tracking results, and the characteristics of specific source-models can be used to link pitch estimates to their sources. From this perspective, the problem of pitch estimation is somewhat replaced by the problem of finding good source models. In certain application scenarios, obtaining SD models of some speakers might be easy. E.g., think about a mobile device such as a cell phone that trains a source model of its primary user to facilitate processing of far distant speech commands. For arbitrary acoustic scenes, however, the process of collecting relevant data to build SD models is much more difficult. Hence, one important question for future research is how to obtain SD models from arbitrary speech mixtures, given only some general SI models.

A Derivation of MIXMAX Likelihood Bounds

A.1 Derivation of the Upper Bound

Lemma 1. *Let A_i and B_i be nonnegative real numbers and $i = 1, \ldots, K$, where $K \in \mathbb{N}$. Then*

$$\prod_{i=1}^{K} (A_i + B_i) \geq \sum_{i=1}^{K} A_i \prod_{j \neq i} B_j. \tag{A.1}$$

Proof. By the multi-binomial theorem, which is a straightforward extension of the binomial theorem to the product of independent binomials, it holds that

$$\prod_{i=1}^{K} (A_i + B_i) = \sum_{n_1=0}^{1} \cdots \sum_{n_K=0}^{1} \prod_{i=1}^{K} A_i^{n_i} B_i^{1-n_i}, \tag{A.2}$$

where the right hand side consists of 2^K terms. Selecting only those K terms where exactly one index n_k is 1 and all other indices are zero, and exploiting the nonnegativity of A_i and B_i, we obtain the lower bound

$$\sum_{n_1=0}^{1} \cdots \sum_{n_K=0}^{1} \prod_{i=1}^{K} A_i^{n_i} B_i^{1-n_i} \geq \sum_{i=1}^{K} A_i \prod_{j \neq i} B_j. \tag{A.3}$$

□

Corollary 1. *The likelihood $L(\cdot)$ obeys*

$$\begin{aligned} L(\{x_k\},\{m_k\}) &= \prod_{d=1}^{D} \sum_{k} \mathcal{N}_{k,x_k}^{m_k,d} \prod_{j\neq k} \Phi_{j,x_j}^{m_j,d} \\ &\leq \prod_{d=1}^{D} \prod_{k} \left\{ \mathcal{N}_{k,x_k}^{m_k,d} + \Phi_{k,x_k}^{m_k,d} \right\}. \end{aligned} \tag{A.4}$$

Proof. Follows directly from Lemma 1. □

By applying the logarithm to (A.4), we finally obtain (3.5).

A.2 Derivation of the Lower Bound

We make use of the following two well known results (proofs can be found in [118]):

Lemma 2. *For $i = 1, \ldots, K$ with $K \in \mathbb{N}$, let $A_i \in \mathbb{R}$. Then*

$$\ln \sum_{i=1}^{K} e^{A_i} \geq \max_i A_i. \tag{A.5}$$

Lemma 3. *Let $A_{i,j} \in \mathbb{R}$ and $i = 1, \ldots, K$, $j = 1, \ldots, N$ with $K \in \mathbb{N}$, $N \in \mathbb{N}$. Then*

$$\max_i \left\{ \sum_j A_{i,j} \right\} \leq \sum_j \max_i A_{i,j}. \tag{A.6}$$

We obtain (3.6) by rewriting the log-likelihood and applying Lemma 2 and 3:

$$\begin{aligned} \ln L(\{x_k\},\{m_k\}) &= \ln \prod_{d=1}^{D} \sum_{k} \mathcal{N}_{k,x_k}^{m_k,d} \prod_{j\neq k} \Phi_{j,x_j}^{m_j,d} \\ &= \sum_{d=1}^{D} \ln \left\{ \left(\sum_k \frac{\mathcal{N}_{k,x_k}^{m_k,d}}{\Phi_{k,x_k}^{m_k,d}} \right) \prod_k \Phi_{k,x_k}^{m_k,d} \right\} \\ &= \sum_d \left(\ln \sum_k \mathrm{e}^{\ln \mathcal{N}_{k,x_k}^{m_k,d} - \ln \Phi_{k,x_k}^{m_k,d}} + \sum_k \ln \Phi_{k,x_k}^{m_k,d} \right) \\ &\geq \sum_d \left(\max_k \left\{ \ln \mathcal{N}_{k,x_k}^{m_k,d} - \ln \Phi_{k,x_k}^{m_k,d} \right\} + \sum_k \ln \Phi_{k,x_k}^{m_k,d} \right) \\ &= \sum_d \max_k \left\{ \ln \mathcal{N}_{k,x_k}^{m_k,d} + \sum_{j\neq k} \ln \Phi_{j,x_j}^{m_j,d} \right\} \\ &\geq \max_k \left\{ \sum_d \ln \mathcal{N}_{k,x_k}^{m_k,d} + \sum_{j\neq k} \sum_d \ln \Phi_{j,x_j}^{m_j,d} \right\}. \end{aligned} \tag{A.7}$$

B

Derivative of the Auxiliary Function in Section 4.2.3

In the following, we derive the partial derivative $\frac{\partial Q_k(\boldsymbol{T},\boldsymbol{b})}{\partial \boldsymbol{T}}$ of the auxiliary function Q_k defined in (4.34), which is repeated here for convenience:

$$Q_k(\boldsymbol{T},\boldsymbol{b}) = \sum_{t,\{x_k\},\{m_k\}} \gamma_{t,\{x_k\},\{m_k\}} \mathrm{E}_{\{\boldsymbol{s}_k^{(t)}\}} \left\{ \ln \mathcal{N}(\boldsymbol{s}_k^{(t)} | \boldsymbol{W}\tilde{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{\mu}_{k,x_k}^{m_k} + \boldsymbol{W}\tilde{\boldsymbol{b}}, \boldsymbol{\Sigma}_{k,x_k}^{m_k}) \right\}. \quad \text{(B.1)}$$

The differential operator can be pulled into the expectation:

$$\frac{\partial Q_k(\boldsymbol{T},\boldsymbol{b})}{\partial \boldsymbol{T}} = \sum_{t,\{x_k\},\{m_k\}} \gamma_{t,\{x_k\},\{m_k\}} \mathrm{E}_{\{\boldsymbol{s}_k^{(t)}\}} \left\{ \frac{\partial}{\partial \boldsymbol{T}} \ln \mathcal{N}(\boldsymbol{s}_k^{(t)} | \boldsymbol{W}\tilde{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{\mu}_{k,x_k}^{m_k} + \boldsymbol{W}\tilde{\boldsymbol{b}}, \boldsymbol{\Sigma}_{k,x_k}^{m_k}) \right\}. \quad \text{(B.2)}$$

The logarithm of the normal distribution has quadratic form:

$$\begin{aligned} \ln \mathcal{N}(\boldsymbol{s}_k^{(t)} | \cdot) = &- \frac{1}{2} \left(\boldsymbol{s}_k^{(t)} - \boldsymbol{W}\tilde{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{\mu}_{k,x_k}^{m_k} - \boldsymbol{W}\tilde{\boldsymbol{b}} \right)^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \\ &\times \left(\boldsymbol{s}_k^{(t)} - \boldsymbol{W}\tilde{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{\mu}_{k,x_k}^{m_k} - \boldsymbol{W}\tilde{\boldsymbol{b}} \right) + \text{const}, \end{aligned} \quad \text{(B.3)}$$

where const denotes all terms independent of $\boldsymbol{T}$. Next, we expand this quadratic form:

$$\begin{aligned} \ln \mathcal{N}(\boldsymbol{s}_k^{(t)} | \cdot) = &- \frac{1}{2} {\boldsymbol{\mu}_{k,x_k}^{m_k}}^T \check{\boldsymbol{W}}^T \boldsymbol{T}^T \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \hat{\boldsymbol{W}} \boldsymbol{T} \check{\boldsymbol{W}} \boldsymbol{\mu}_{k,x_k}^{m_k} \\ &+ {\boldsymbol{\mu}_{k,x_k}^{m_k}}^T \check{\boldsymbol{W}}^T \boldsymbol{T}^T \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \left(\boldsymbol{s}_k^{(t)} - \bar{\boldsymbol{W}} \boldsymbol{\mu}_{k,x_k}^{m_k} - \hat{\boldsymbol{W}} \boldsymbol{b} \right) + \text{const}, \end{aligned} \quad \text{(B.4)}$$

where we made use of Equations (4.38) and (4.39). The derivative of (B.4) with respect to $\boldsymbol{T}$ is [119]:

$$\begin{aligned}\frac{\partial \ln \mathcal{N}(\boldsymbol{s}_k^{(t)}|\cdot)}{\partial \boldsymbol{T}} = &- \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \hat{\boldsymbol{W}} \boldsymbol{T} \check{\boldsymbol{W}} {\boldsymbol{\mu}_{k,x_k}^{m_k}} {\boldsymbol{\mu}_{k,x_k}^{m_k}}^T \check{\boldsymbol{W}}^T \\ &+ \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \left(\boldsymbol{s}_k^{(t)} - \bar{\boldsymbol{W}} \boldsymbol{\mu}_{k,x_k}^{m_k} - \hat{\boldsymbol{W}} \boldsymbol{b} \right) {\boldsymbol{\mu}_{k,x_k}^{m_k}}^T \check{\boldsymbol{W}}^T. \end{aligned} \tag{B.5}$$

We may now combine (B.5) with (B.2), and make use of the fact that the expectation operator only depends on $\boldsymbol{s}_k^{(t)}$:

$$\begin{aligned}\frac{\partial Q_k(\boldsymbol{T}, \boldsymbol{b})}{\partial \boldsymbol{T}} = &\sum_{t,\{x_k\},\{m_k\}} \gamma_{t,\{x_k\},\{m_k\}} \\ &\times \Big(-\hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \hat{\boldsymbol{W}} \boldsymbol{T} \check{\boldsymbol{W}} {\boldsymbol{\mu}_{k,x_k}^{m_k}} {\boldsymbol{\mu}_{k,x_k}^{m_k}}^T \check{\boldsymbol{W}}^T \\ &+ \hat{\boldsymbol{W}}^T {\boldsymbol{\Sigma}_{k,x_k}^{m_k}}^{-1} \left(\mathrm{E}_{\{\boldsymbol{s}_k^{(t)}\}} \left\{ \boldsymbol{s}_k^{(t)} \right\} - \bar{\boldsymbol{W}} \boldsymbol{\mu}_{k,x_k}^{m_k} - \hat{\boldsymbol{W}} \boldsymbol{b} \right) {\boldsymbol{\mu}_{k,x_k}^{m_k}}^T \check{\boldsymbol{W}}^T \Big). \end{aligned} \tag{B.6}$$

C

Derivation of the MIXMAX Expected Single-Speaker Log-Spectrum

The EM algorithm described in Section 4.2.3 replaces the occurrence of the hidden single-speaker spectrum of speaker k, $\boldsymbol{s}_k$, by its expectation with respect to the posterior defined in (4.30):

$$\mathrm{E}\{\boldsymbol{s}_k|\boldsymbol{y},\{x_k\},\{m_k\}\} = \begin{pmatrix} \mathrm{E}\{s_k^1|\boldsymbol{y},\{x_k\},\{m_k\}\} \\ \mathrm{E}\{s_k^2|\boldsymbol{y},\{x_k\},\{m_k\}\} \\ \vdots \\ \mathrm{E}\big\{s_k^D|\boldsymbol{y},\{x_k\},\{m_k\}\big\} \end{pmatrix}, \tag{C.1}$$

where the d^{th}dimension of the expected spectrum is formally defined as

$$\mathrm{E}\big\{s_k^d|\boldsymbol{y},\{x_k\},\{m_k\}\big\} = \int\!\!\cdots\!\!\int s_k^d p(\{\boldsymbol{s}_k\}|\boldsymbol{y},\{x_k\},\{m_k\}) d\boldsymbol{s}_1 \cdots d\boldsymbol{s}_K. \tag{C.2}$$

Note that for the sake of brevity, we have omitted the explicit dependency on time index t and adaptation parameters $\{\boldsymbol{T}_k^{(old)}\}$ and $\{\boldsymbol{b}_k^{(old)}\}$. Assuming that two sources interact according to the MIXMAX model, a closed form solution for the expectation (C.2) was first derived in [79], and a similar derivation can be found in [81, 82, 84, 122]. In the following, we provide a derivation for the slightly more general case of K simultaneous sources. The posterior (4.30) can be rewritten according to Bayes rule:

$$\begin{aligned} p(\{\boldsymbol{s}_k\}|\boldsymbol{y},\{x_k\},\{m_k\}) &= \frac{p(\boldsymbol{y}|\{\boldsymbol{s}_k\})\prod_k p(\boldsymbol{s}_k|x_k,m_k)}{p(\boldsymbol{y}|\{x_k\},\{m_k\})} \\ &= \prod_d \frac{p(y_d|\{s_k^d\})\prod_k p(s_k^d|x_k,m_k)}{p(y^d|\{x_k\},\{m_k\})}. \end{aligned} \tag{C.3}$$

We plug (C.3) into (C.2) and observe that for all $j \neq d$, the integral of the numerator along $\{s_k^j\}$ is identical to the corresponding denominator, i.e. all factors independent of

dimension d cancel:

$$\begin{aligned}\mathrm{E}\{s_k^d|\boldsymbol{y},\{x_k\},\{m_k\}\} &= \prod_{j\neq d}\left(\int\!\!\cdots\!\!\int \frac{p(y_j|\{s_k^j\})\prod_k p(s_k^j|x_k,m_k)}{p(y^j|\{x_k\},\{m_k\})}ds_1^j\cdots ds_K^j\right)\\ &\quad\times\int\!\!\cdots\!\!\int\frac{s_k^d p(y_d|\{s_k^d\})\prod_k p(s_k^d|x_k,m_k)}{p(y^d|\{x_k\},\{m_k\})}ds_1^d\cdots ds_K^d\\ &= \int\!\!\cdots\!\!\int\frac{s_k^d p(y_d|\{s_k^d\})\prod_k p(s_k^d|x_k,m_k)}{p(y^d|\{x_k\},\{m_k\})}ds_1^d\cdots ds_K^d. \end{aligned}\tag{C.4}$$

Next, we rewrite the MIXMAX interaction model:

$$\begin{aligned}p(y_d|\{s_k^d\}) &= \delta(y_d-\max(s_1^d,\ldots,s_K^d))\\ &= \sum_k \delta(y_d-s_k^d)\prod_{j\neq k}u(y_d-s_j^d),\end{aligned}\tag{C.5}$$

where $\delta(\cdot)$ and $u(\cdot)$ denote the Dirac delta and the unit step function, respectively. Combining (C.5) with (C.4), and assuming that $p(s_k^d|x_k,m_k)$ is a normal distribution, we see that basically four types of standard integrals occur which all have a closed form solution:

$$\int\mathcal{N}(s|\mu,\sigma^2)\delta(y-s)ds = \mathcal{N}(y|\mu,\sigma^2),\tag{C.6}$$

$$\int\mathcal{N}(s|\mu,\sigma^2)u(y-s)ds = \Phi(y|\mu,\sigma^2),\tag{C.7}$$

$$\int s\mathcal{N}(s|\mu,\sigma^2)\delta(y-s)ds = y\mathcal{N}(y|\mu,\sigma^2),\text{ and}\tag{C.8}$$

$$\int s\mathcal{N}(s|\mu,\sigma^2)u(y-s)ds = \mu\Phi(y|\mu,\sigma^2)-\sigma^2\mathcal{N}(y|\mu,\sigma^2).\tag{C.9}$$

For brevity, we introduce the following short-hand symbols for the normal density and the cumulative normal distribution:

$$\mathcal{N}_{k,x_k}^{m_k,d} := \mathcal{N}(y_d|\theta_{k,x_k}^{m_k,d}),\tag{C.10}$$

$$\Phi_{k,x_k}^{m_k,d} := \Phi(y_d|\theta_{k,x_k}^{m_k,d}).\tag{C.11}$$

As described in Chapter 3, the denominator in (C.4) is given by

$$p(y^d|\{x_k\},\{m_k\}) = \sum_k \mathcal{N}_{k,x_k}^{m_k,d}\prod_{j\neq k}\Phi_{j,x_j}^{m_j,d}.\tag{C.12}$$

Putting all together, we finally obtain:

$$\begin{aligned}\mathrm{E}\{s_k^d|\boldsymbol{y},\{x_k\},\{m_k\}\} &= \frac{y_d \mathcal{N}_{k,x_k}^{m_k,d} \prod_{j\neq k} \Phi_{j,x_j}^{m_j,d}}{p(y^d|\{x_k\},\{m_k\})} \\ &\quad + \frac{\sum_{l\neq k} \mathcal{N}_{l,x_l}^{m_l,d} \left(\mu_{k,x_k}^{m_k,d}\Phi_{k,x_k}^{m_k,d} - \left(\sigma_{k,x_k}^{m_k,d}\right)^2 \mathcal{N}_{k,x_k}^{m_k,d}\right) \prod_{j\notin\{l,k\}} \Phi_{j,x_j}^{m_j,d}}{p(y^d|\{x_k\},\{m_k\})} \qquad \text{(C.13)} \\ &= \frac{y_d \Psi_{k,x_k}^{m_k,d} + \left(\mu_{k,x_k}^{m_k,d} - \left(\sigma_{k,x_k}^{m_k,d}\right)^2 \Psi_{k,x_k}^{m_k,d}\right) \sum_{l\neq k} \Psi_{l,x_l}^{m_l,d}}{\sum_j \Psi_{j,x_j}^{m_j,d}}, \qquad \text{(C.14)}\end{aligned}$$

where we defined the ratio

$$\Psi_{k,x_k}^{m_k,d} = \frac{\mathcal{N}(y_d|\theta_{k,x_k}^{m_k,d})}{\Phi(y_d|\theta_{k,x_k}^{m_k,d})}. \qquad \text{(C.15)}$$

To interpret Equation (C.14), note that $\Psi_{k,x_k}^{m_k,d}$ can be crudely approximated as

$$\Psi_{k,x_k}^{m_k,d} \approx \max\left(\frac{\mu_{k,x_k}^{m_k,d} - y_d}{(\sigma_{k,x_k}^{m_k,d})^2}, 0\right). \qquad \text{(C.16)}$$

From this we see that $\Psi_{k,x_k}^{m_k,d}$ is large if $y_d \ll \mu_{k,x_k}^{m_k,d}$ and zero if $y_d \gg \mu_{k,x_k}^{m_k,d}$. In the former case, the expected value (C.14) is more or less set to y_d. In the latter case, source k is 'masked', and the expectation is dominated by the mean of the model. For any intermediate case, the expectation is an interpolation between these two extreme cases.

Bibliography

[1] D. Wang and G. Brown, Eds., *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*. John Wiley and Sons, 2006.

[2] A. S. Bregman, *Auditory Scene Analysis*. MIT Press, 1990.

[3] H. Järveläinen, V. Välimäki, and M. Karjalainen, "Audibility of the timbral effects of inharmonicity in stringed instrument tones," *Acoustics Research Letters Online*, vol. 2, 2001.

[4] P. Vary, U. Heute, and W. Hess, *Digitale Sprachsignalverarbeitung*. B.G. Teubner Stuttgart, 1998.

[5] T. W. Parsons, "Separation of speech from interfering speech by means of harmonic selection," *The Journal of the Acoustical Society of America*, vol. 60, no. 4, pp. 911–918, 1976.

[6] M. Weintraub, "A computational model for separating two simultaneous talkers," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 11, 1986, pp. 81 – 84.

[7] A. de Cheveigné, "Separation of concurrent harmonic sounds: Fundamental frequency estimation and a time-domain cancellation model of auditory processing," *The Journal of the Acoustical Society of America*, vol. 93, no. 6, pp. 3271–3290, 1993.

[8] D. Chazan, Y. Stettiner, and D. Malah, "Optimal multi-pitch estimation using the EM algorithm for co-channel speech separation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, 1993, pp. 728 –731 vol.2.

[9] D. Morgan, E. George, L. Lee, and S. Kay, "Cochannel speaker separation by harmonic enhancement and suppression," *IEEE Trans. Speech and Audio Processing*, vol. 5, no. 5, pp. 407–424, 1997.

[10] G. Hu and D. Wang, "Monaural speech segregation based on pitch tracking and amplitude modulation," *IEEE Transactions on Neural Networks*, vol. 15, no. 5, pp. 1135 – 1150, 2004.

[11] M. H. Radfar, R. M. Dansereau, and A. Sayadiyan, "A maximum likelihood estimation of vocal-tract-related filter characteristics for single channel speech separation," *EURASIP J. Audio Speech Music Process.*, vol. 2007, pp. 1–15, 2007.

[12] S. Vishnubhotla and C. Espy-Wilson, "An algorithm for speech segregation of co-channel speech," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 109 –112.

[13] M. Stark, M. Wohlmayr, and F. Pernkopf, "Source-filter-based single-channel speech separation using pitch information," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 2, pp. 242 –255, 2011.

[14] M. Zissman and D. Seward IV, "Two-talker pitch tracking for co-channel talker interference suppression," Massachusetts Institute of Technology, Tech. Rep. AD-A253 418, 1992.

[15] D. Talkin, "A robust algorithm for pitch tracking (RAPT)," *Kleijn W.B. and Paliwal K.K. [Ed], Speech Coding and Synthesis, Elsevier Science*, pp. 495–518, 1995.

[16] A. de Cheveigné and H. Kawahara, "YIN, a fundamental frequency estimator for speech and music," *J. Acoust. Soc. Am.*, vol. 111, no. 4, pp. 1917–1930, 2002.

[17] P. Boersma, "Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound," *Proceedings of the Institute of Phonetic Sciences, Amsterdam*, vol. 17, pp. 97–110, 1993.

[18] ——, "Praat, a system for doing phonetics by computer," *Glot International*, vol. 5, pp. 341–345, 2002.

[19] M. Ross, H. Shaffer, A. Cohen, R. Freudberg, and H. Manley, "Average magnitude difference function pitch extractor," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 22, no. 5, pp. 353 – 362, 1974.

[20] Y. Medan, E. Yair, and D. Chazan, "Super resolution pitch determination of speech signals," *IEEE Transactions on Signal Processing*, vol. 39, no. 1, pp. 40 –48, 1991.

[21] A. M. Noll, "Cepstrum pitch determination," *Journal of the Acoustical Society of America*, vol. 41, pp. 293–309, 1967.

[22] L. Rabiner, "On the use of autocorrelation analysis for pitch detection," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 25, no. 1, pp. 24 – 33, 1977.

[23] S. Ahmadi and A. Spanias, "Cepstrum-based pitch detection using a new statistical v/uv classification algorithm," *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 3, pp. 333 –338, 1999.

[24] J. Markel, "The SIFT algorithm for fundamental frequency estimation," *IEEE Transactions on Audio and Electroacoustics*, vol. 20, no. 5, pp. 367 – 377, 1972.

[25] J. Wise, J. Caprio, and T. Parks, "Maximum likelihood pitch estimation," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 24, no. 5, pp. 418 – 423, 1976.

[26] M. Slaney and R. Lyon, "A perceptual pitch detector," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 1990, pp. 357 –360 vol.1.

[27] J. Rouat, Y. C. Liu, and D. Morissette, "A pitch determination and voiced/unvoiced decision algorithm for noisy speech," *Speech Communication Journal*, vol. 21, no. 3, pp. 191 – 207, 1995.

[28] R. Meddis and M. J. Hewitt, "Virtual pitch and phase sensitivity of a computer model of the auditory periphery. I: Pitch identification," *The Journal of the Acoustical Society of America*, vol. 89, no. 6, pp. 2866–2882, 1991.

[29] R. Meddis and L. O'Mard, "A unitary model of pitch perception," *The Journal of the Acoustical Society of America*, vol. 102, no. 3, pp. 1811–1820, 1997.

[30] B. Atal and L. Rabiner, "A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 24, no. 3, pp. 201 – 212, 1976.

[31] X. Li, J. Malkin, and J. Bilmes, "Graphical model approach to pitch tracking," in *International Conference on Spoken Language Processing*, 2004, pp. 1101–1104.

[32] L. Rabiner, M. Cheng, A. Rosenberg, and C. McGonegal, "A comparative performance study of several pitch detection algorithms," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 24, no. 5, pp. 399–418, 1976.

[33] P. Bagshaw, S. Hiller, and M. Jack, "Enhanced pitch tracking and the processing of F0 contours for computer aided intonation teaching," in *Proc. Eurospeech*, 1993, pp. 1003–1006.

[34] W. Hess, *Pitch determination of speech signals: Algorithms and devices*. Springer Verlag, 1983.

[35] M. Christensen and A. Jakobsson, *Multi-Pitch Estimation*, ser. Synthesis Lectures on Speech and Audio Processing, B.H.Juang, Ed. Morgan & Claypool, 2008.

[36] A. de Cheveigné, "Multiple f0 estimation," in *Computational Auditory Scene Analysis*, D. Wang and G. Brown, Eds. John Wiley and Sons, 2006, pp. 45–80.

[37] A. Klapuri and M. Davy, Eds., *Signal processing methods for music transcription*. Springer, 2006.

[38] S. Vishnubhotla and C. Espy-Wilson, "An algorithm for multi-pitch tracking in co-channel speech," in *International Conference on Spoken Language Processing (Interspeech)*, 2008, pp. 143–146.

[39] T. Quatieri, "2-D processing of speech with application to pitch estimation," in *International Conference on Spoken Language Processing (Interspeech)*, 2002.

[40] T. Ezzat, J. Bouvrie, and T. Poggio, "Spectro-temporal analysis of speech using 2-D gabor filters," in *International Conference on Spoken Language Processing (Interspeech)*, 2007, pp. 506–509.

[41] T. Chi, P. Ru, and S. A. Shamma, "Multiresolution spectrotemporal analysis of complex sounds," *The Journal of the Acoustical Society of America*, vol. 118, no. 2, pp. 887–906, 2005.

[42] A. Klapuri, "Multipitch analysis of polyphonic music and speech signals using an auditory model," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 255 –266, 2008.

[43] T. Tolonen and M. Karjalainen, "A computationally efficient multipitch analysis model," *IEEE Transactions on Speech and Audio Processing*, vol. 8, pp. 708–716, 2000.

[44] M. Wu, D. Wang, and G. Brown, "A multipitch tracking algorithm for noisy speech," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 3, pp. 229–241, 2003.

[45] G. Hu and D. Wang, "A tandem algorithm for pitch estimation and voiced speech segregation," *IEEE Transactions on Audio, Speech, and Language Processing*, 2010.

[46] Z. Jin and D. Wang, "HMM-based multipitch tracking for noisy and reverberant speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 5, pp. 1091 –1102, 2011.

[47] R. Patterson, I. Nimmo-Smith, J. Holdsworth, and P. Rice, "An efficient auditory filterbank based on the gammatone function," Applied Psychology Unit, Cambridge, Tech. Rep., 1988.

[48] J. Kaiser, "On a simple algorithm to calculate the 'energy' of a signal," in *International Conference on Acoustics, Speech, and Signal Processing, 1990. ICASSP-90.* IEEE, 1990, pp. 381–384 vol.1.

[49] R. Duda, R. Lyon, and M. Slaney, "Correlograms and the separation of sounds," *Signals, Systems and Computers*, vol. 1, pp. 457–461, 1990.

[50] R. McAulay and T. Quatieri, "Pitch estimation and voicing detection based on a sinusoidal speech model," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 1990, pp. 249–252.

[51] H. Kameoka, "Statistical approach to multipitch analysis," Ph.D. dissertation, University of Tokyo, 2007.

[52] J. Le Roux, H. Kameoka, N. Ono, A. de Cheveigné, and S. Sagayama, "Single and multiple F0 contour estimation through parametric spectrogram modeling of speech in noisy environments," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1135 –1145, 2007.

[53] F. Bach and M. Jordan, "Discriminative training of hidden Markov models for multiple pitch tracking," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2005, pp. 489–492.

[54] A. Klapuri, T. Virtanen, and T. Heittola, "Sound source separation in monaural music signals using excitation-filter model and EM algorithm," in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2010, pp. 5510 –5513.

[55] T. Virtanen and A. Klapuri, "Separation of harmonic sounds using multipitch analysis and iterative parameter estimation," in *IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics*, 2001, pp. 83 –86.

[56] S. Godsill and M. Davy, "Bayesian harmonic models for musical pitch estimation and analysis," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2002, pp. 1769–1772.

[57] M. Davy, S. Godsill, and J. Idier, "Bayesian analysis of polyphonic western tonal music," *The Journal of the Acoustical Society of America*, vol. 119, no. 4, pp. 2498–2517, 2006.

[58] F. Sha and L. K. Saul, "Real-time pitch determination of one or more voices by nonnegative matrix factorization," *Advances in Neural Information Processing Systems (NIPS)*, 2005.

[59] A. Cont, "Realtime multiple pitch observation using sparse non-negative constraints," in *International Conference on Music Information Retrieval*, 2006.

[60] E. Vincent, N. Bertin, and R. Badeau, "Adaptive harmonic spectral decomposition for multiple pitch estimation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 528 –537, 2010.

[61] M. Wohlmayr, M. Stark, and F. Pernkopf, "A mixture maximization approach to multipitch tracking with factorial hidden Markov models," in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2010, pp. 5070 –5073.

[62] ——, "A probabilistic interaction model for multipitch tracking with factorial hidden Markov models," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 799 –810, 2011.

[63] M. Wohlmayr, R. Peharz, and F. Pernkopf, "Efficient implementation of probabilistic multi-pitch tracking," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 5412 –5415.

[64] M. Wohlmayr and F. Pernkopf, "Multipich tracking using a factorial hidden Markov model," in *International Conference on Spoken Language Processing (Interspeech)*, 2008, pp. 147–150.

[65] ——, "Finite mixture spectrogram modeling for multipitch tracking using a factorial hidden Markov model," in *International Conference on Spoken Language Processing (Interspeech)*, 2009, pp. 1079–1082.

[66] F. Pernkopf and M. Wohlmayr, "Maximum margin training for Gaussian mixture models with application to multipitch tracking," in *COMPSTAT 2010*, Paris, 2010.

[67] M. Stark, M. Wohlmayr, and F. Pernkopf, "Single channel speech separation using source-filter representation," in *ICPR 2010*, Turkey, 2010, pp. 826 – 829.

[68] M. Wohlmayr and F. Pernkopf, "EM-based gain adaptation for probabilistic multipitch tracking," in *International Conference on Spoken Language Processing (Interspeech)*, 2011, pp. 1969–1972.

[69] R. Peharz, M. Wohlmayr, and F. Pernkopf, "Gain-robust multi-pitch tracking using sparse nonnegative matrix factorization," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 5416 –5419.

[70] G. Pirker, M. Wohlmayr, S. Petrik, and F. Pernkopf, "A pitch tracking corpus with evaluation on multipitch tracking scenario," in *International Conference on Spoken Language Processing (Interspeech)*, 2011, pp. 1509–1512.

[71] S. Rennie, J. Hershey, and P. Olsen, "Single-channel multitalker speech recognition," *IEEE Signal Processing Magazine*, vol. 27, no. 6, pp. 66 –80, 2010.

[72] F. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[73] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc.*, vol. B30, pp. 1–38, 1977.

[74] G. McLachlan and K. Basford, *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, 1988.

[75] A. Varga and R. Moore, "Hidden Markov model decomposition of speech and noise," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 845–848 vol.2, 1990.

[76] T. T. Kristjansson, "Speech recognition in adverse environments: A probabilistic approach," Ph.D. dissertation, University of Waterloo, 2002.

[77] T. Virtanen, "Speech recognition using factorial hidden Markov models for separation in the feature space," in *International Conference on Spoken Language Processing (Interspeech)*, 2006, pp. 89–92.

[78] Z. Ghahramani and M. Jordan, "Factorial hidden Markov models," *Machine Learning*, vol. 29, no. 2-3, pp. 245–273, 1997.

[79] A. Nadas, D. Nahamoo, and M. Picheny, "Speech recognition using noise-adaptive prototypes," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 10, pp. 1495–1503, 1989.

[80] M. Gales and S. Young, "An improved approach to the hidden Markov model decomposition of speech and noise," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, 1992, pp. 233 –236 vol.1.

[81] R. Rose, E. Hofstetter, and D. Reynolds, "Integrated models of signal and background with application to speaker identification in noise," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 245–257, 1994.

[82] D. Burshtein and S. Gannot, "Speech enhancement using a mixture-maximum model," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 6, pp. 341–351, 2002.

[83] S. T. Roweis, "Factorial models and refiltering for speech separation and denoising," in *Proceedings of the 8th European Conference on Speech Communication and Technology (Eurospeech)*, 2003, pp. 1009–1012.

[84] A. M. Reddy and B. Raj, "Soft mask estimation for single channel speaker separation," in *ISCA Tutorial and Research Workshop on Statistical and Perceptual Audio Processing*, 2004.

[85] S. Rennie, J. Hershey, and P. Olsen, "Single-channel speech separation and recognition using loopy belief propagation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2009, pp. 3845–3848.

[86] D. Wang, "On ideal binary mask as the computational goal of auditory scene analysis," in *Speech Separation by Humans and Machines*, P. Divenyi, Ed. Springer, 2004, pp. 181–197.

[87] M. H. Radfar, R. M. Dansereau, and A. Sayadiyan, "Nonlinear minimum mean square error estimator for mixture-maximisation approximation," *Electronics Letters*, vol. 42, no. 12, pp. 724–725, 2006.

[88] A. Papoulis, *Probability, random variables, and stochastic processes.* McGraw-Hill, 1991.

[89] M. Jordan, *Learning in graphical models.* MIT Press, 1999.

[90] T. Minka, "Divergence measures and message passing," Microsoft Research Cambridge, Tech. Rep. MSR-TR-2005-173, 2005.

[91] C. Huang and A. Darwiche, "Inference in belief networks: A procedural guide," *Int. J. Approximate Reason.*, vol. 15, no. 3, pp. 225–263, 1996.

[92] S. Aji and R. McEliece, "The generalized distributive law," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 325–343, 2000.

[93] Y. Weiss, "Belief propagation and revision in networks with loops," Tech. Rep. AIM-1616, CBCL-155, 1997.

[94] M. Cooke, J. Barker, S. Cunningham, and X. Shao, "An audio-visual corpus for speech perception and automatic speech recognition," *Journal of the Acoustical Society of America*, vol. 120, pp. 2421–2424, 2005.

[95] M. Cooke, J. R. Hershey, and S. J. Rennie, "Monaural speech separation and recognition challenge," *Computer Speech and Language*, vol. 24, pp. 1–15, 2010.

[96] A. Wrench, "A multichannel/multispeaker articulatory database for continuous speech recognition research," *Phonus*, vol. 5, pp. 3–17, 2000.

[97] F. Plante, G. Meyer, and A. Ainsworth, "A pitch extraction reference database," in *European Conference on Speech Communication and Technology (Eurospeech)*, 1995, pp. 837–840.

[98] L. Lamel, R. Kassel, and S. Seneff, "Speech database development: Design and analysis of the acoustic-phonetic corpus," in *DARPA Speech Recognition Workshop, Report No. SAIC-86/1546*, 1986.

[99] S. Rennie, J. Hershey, and P. Olsen, "Variational loopy belief propagation for multi-talker speech recognition," in *International Conference on Spoken Language Processing (Interspeech)*, 2009, pp. 1331–1334.

[100] T. Kristjansson, J. Hershey, P. Olsen, S. Rennie, and R. Gopinath, "Super-human multi-talker speech recognition: the IBM 2006 speech separation challenge system," in *International Conference on Spoken Language Processing (Interspeech)*, 2006, pp. 97–100.

[101] C. Leggetter and P. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer Speech and Language*, vol. 9, no. 2, pp. 171 – 185, 1995.

[102] M. Gales and P. Woodland, "Mean and variance adaptation within the MLLR framework," *Computer Speech & Language*, vol. 10, no. 4, pp. 249 – 264, 1996.

[103] J.-L. Gauvain and C.-H. Lee, "Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, pp. 291–298, 1994.

[104] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski, "Rapid speaker adaptation in eigenvoice space," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, pp. 695 –707, 2000.

[105] R. Weiss and D. Ellis, "A variational EM algorithm for learning eigenvoice parameters in mixed signals," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 113 –116.

[106] S. Rennie, T. Kristjansson, P. Olsen, and R. Gopinath, "Dynamic noise adaptation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1, 2006, pp. 1197–1200.

[107] J. Hao, H. Attias, S. Nagarajan, T.-W. Lee, and T. Sejnowski, "Speech enhancement, gain, and noise spectrum adaptation using approximate Bayesian estimation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 1, pp. 24 –37, 2009.

[108] T. Krisjansson, B. Frey, L. Deng, and A. Acero, "Joint estimation of noise and channel distortion in a generalized EM framework," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2001, pp. 155 – 158.

[109] A. Ozerov, P. Philippe, F. Bimbot, and R. Gribonval, "Adaptation of Bayesian models for single-channel source separation and its application to voice/music separation in popular songs," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 5, pp. 1564 –1578, 2007.

[110] T. Nakatani, S. Araki, T. Yoshioka, and M. Fujimoto, "Joint unsupervised learning of hidden Markov source models and source location models for multichannel source separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 237 –240.

[111] M. Radfar, W. Wong, R. Dansereau, and W.-Y. Chan, "Scaled factorial hidden Markov models: A new technique for compensating gain differences in model-based single channel speech separation," in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2010, pp. 1918 –1921.

[112] S. Rennie, P. Olsen, J. Hershey, and T. Kristjansson, "The Iroquois model: Using temporal dynamics to separate speakers," in *Workshop on Statistical And Perceptual Audition*, 2006, pp. 24–30.

[113] C. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[114] M. T.P., "Expectation-maximization as lower bound maximization," Tech. Rep., 1998.

[115] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *IEEE ASSP Magazine*, pp. 4–16, 1986.

[116] T. Moon and W. Stirling, *Mathematical Methods and Algorithms for Signal Processing*. Prentice Hall, 2000.

[117] G.-H. Ding, Y.-F. Zhu, C. Li, and B. Xu, "Implementing vocal tract length normalization in the MLLR framework," in *Proceedings of the International Conference on Spoken Language Processing (Interspeech)*, 2002, pp. 1389–1392.

[118] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[119] K. Petersen and M. Pedersen, *The Matrix Cookbook*, 2008.

[120] R. Neal and G. Hinton, "A view of the EM algorithm that justifies incremental, sparse, and other variants," in *Learning in Graphical Models*, M. Jordan, Ed. Kluwer Academic Publisher, 1998, pp. 355–368.

[121] Y. Shao and D. Wang, "Model-based sequential organization in cochannel speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 289 – 298, 2006.

[122] A. Erell and D. Burshtein, "Noise adaptation of HMM speech recognition systems using tied-mixtures in the spectral domain," *IEEE Transactions on Speech and Audio Processing*, vol. 5, no. 1, pp. 72 –74, 1997.