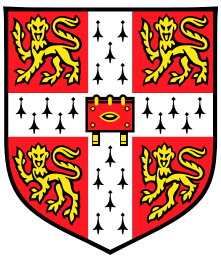
Model-based Approaches to Robust Speech Recognition in Diverse Environments

Yongqiang Wang

Darwin College
Engineering Department Cambridge University



October 2015

This dissertation is submitted to the University of Cambridge
for the degree of Doctor of Philosophy.

Declaration

This dissertation is the result of my own work carried out at the Cambridge University and includes nothing which is the outcome of any work done in collaboration except where explicitly stated. It has not been submitted in whole or in part for a degree at any other university. Some of the work has been previously presented in international conferences and workshops [71, 241, 242, 244–246], or published as a journal article [243]. The length of this thesis including footnotes, appendices and references is approximately 57,000 words. This thesis contains 29 figures and 22 tables.

Summary

Model-based approaches are a powerful and flexible framework for robust speech recognition. This framework has been extensively investigated during the past decades and has been extended in a number of ways to handle distortions caused by various acoustic factors, including speaker differences, channel distortions and environment noise. This thesis investigated model-based approaches to robust speech recognition in diverse conditions and proposed two extensions to this framework.

Many speech recognition applications will benefit from distant-talking speech capture. This avoids problems caused by using hand-held or body-worn equipment. However, due to the large speaker-to-microphone distance, both background noise and reverberant noise will significantly corrupt speech signals and negatively impact speech recognition accuracies. This work will propose a new model-based scheme for those applications in which only a single distant microphone is available. To compensate for the influence of previous speech frames on the current speech frame in reverberant environments, extended statistics are appended to the standard acoustic model to represent the distribution of a window of contextual clean feature vectors at the Gaussian component level. Given these statistics and the reverberant noise model parameters, the standard Vector Taylor series (VTS) expansion is extended to compensate the acoustic model parameters for the effect of reverberation and background noise. A maximum likelihood (ML) estimation algorithm is also developed to estimate the reverberant noise model parameters. Adaptive training of acoustic model parameters on data recorded in multiple reverberant environments is also proposed. This allows a consistent ML framework to estimate both the reverberant noise parameters and the acoustic model parameters. Experiments are performed on an artificially corrupted corpus and a corpus recorded in real reverberant environments. It is observed that the proposed model-based schemes significantly improve the model robustness to reverberation for both clean-trained and adaptively-trained acoustic models.

As the speech signals are usually affected by multiple acoustic factors simultaneously, another important aspect in the model-based framework is the ability to adapt canonical models to the target acoustic condition with multiple acoustic factors in a flexible manner. An acoustic factorisation framework has been proposed to factorise the variability caused by different acoustic factors. This is achieved by associating each acoustic factor with a distinct factor transform. In this way, it enables factorised adaptation, which gives extra flexibility for model-based approaches. The second part of this thesis proposes several extensions to acoustic factorisation. It is first established that the key to acoustic factorisation is to keep the factor transforms independent of each other. Several approaches are discussed to construct such

independent factor transforms. The first one is the widely used data constrained approach, which solely relies on the adaptation data to achieve the independence attribute. The second, transform constrained approach utilises partial knowledge of how acoustic factors affect the speech signals and relies on different forms of transforms to achieve factorisation. Based on a mathematical analysis of the dependence between ML estimated factor transforms, the third approach explicitly enforces the independence constraint, thus it is not relying on balanced data or particular forms of transforms. The transform constrained and the explicit independence constrained factorisation approaches are applied to the speaker and noise factorisation for speech recognition, yielding two flexible model-based schemes which can use the speaker transforms estimated in one noise condition in other unseen noise conditions. Experimental results on artificially corrupted corpora demonstrate the flexibility of these schemes and also illustrate the importance of the independence attribute to factorisation.

Keywords: automatic speech recognition, hidden Markov models, model adaptation, adaptive training, environment robustness, reverberant noise robustness, acoustic factorisation

Acknowledgement

First of all, I would like to express my utmost gratitude to my supervisor, Professor Mark Gales, for his guidance and support throughout my study at Cambridge University. I learned a lot from his precise logical flow in thinking, rigorousness in scientific principles, and his attitude towards excellent research. This thesis would not become possible without his well-motivated questions, experienced suggestions, and constructive advice. I am deeply indebted to his patient and constant help on every aspect during my PhD study. It is truly a privilege to have such precious chance to work with him fruitfully in the past four years.

I would also like to thank my advisor, Professor Phil. Woodland for both his constructive suggestions and funding support. My thanks also go to every member in the Machine Intelligence Lab for their help and encouragements, particularly Xunying Liu, Kai Yu, Kate Knill, Federico Flego, Frank Diehl, Rogier van Dalen, Anton Ragni, Zoi Roupakia, Matthew Seigel, Matt Shannon, Yanhua Long, Shixiong Zhang, Xie Chen, Chao Zhang and Jingzhou Yang, who make the study in this lab enjoyable and fruitful. Special thanks to Patrick Gosling and Anna Langley for their excellent work in creating and maintaining the outstanding computing facilities.

This work is partially funded by Google research award and Defense Advanced Research Projects Agency (DARPA) under the GALE and RATS programs. I am very grateful to all of them for providing the financial support over the years. I must also thank the speech technology group at Microsoft Research Redmond center, in particular, my mentor Dr. Michael Seltzer, and Dr. Dong Yu, Dr. Li Deng for giving me a precious opportunity to intern there and making the internship so enjoyable.

Finally, I am also much thankful to my parents to whom I owe everything. Special thanks go to my my wife, for her sacrifice, love and patient support.

Notation

General Notation

$\approx$	approximately equal to
$\propto$	proportional to
s	scalar quantity (lowercase plain letter)
$\mathbf{v}$	vector quantity (lowercase bold letter)
$\mathbf{M}$	matrix (uppercase bold letter)
$\mathbf{M}^{\mathrm{T}}$	transpose of matrix $\mathbf{M}$
$\lvert \cdot \rvert$	determinant of a square matrix
$(\cdot)^{-1}$	inverse of a square matrix
$\mathrm{diag}(\mathbf{v})$	diagonal matrix with vector $\mathbf{v}$ as its diagonal elements
$\mathrm{diag}(\mathbf{M})$	diagonal matrix derived from a squared matrix $\mathbf{M}$
$\mathrm{tr}(\cdot)$	trace of a square matrix
$\mathrm{vec}(\cdot)$	vectorised form of a matrix

Functions

$\mathcal{F}(\cdot)$	objective function or a mapping function
$\mathcal{Q}(\cdot;\cdot)$	auxiliary function at the current estimates of parameters
$\frac{\partial}{\partial \boldsymbol{x}} f(\boldsymbol{x})$	derivate of a function
$\frac{\partial^2}{\partial \boldsymbol{x} \partial \boldsymbol{x}} f(\boldsymbol{x})$	Hessian of a function
$f(x)\big\rvert_{x=\hat{x}}$	value of function $f(x)$ at $x = \hat{x}$
$\arg\max_x f(x)$	value of x that maximises $f(x)$
$\arg\min_x f(x)$	value of x that minimises $f(x)$

Probability Distributions

$p(\cdot)$	probability density function
$p(\cdot\lvert\cdot)$	conditional probability density
$P(\cdot)$	probability mass distribution
$P(\cdot\lvert\cdot)$	conditional probability mass distribution
$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian multivariate distributions of $\mathbf{x}$
$\mathcal{L}(\cdot)$	(log-) likelihood function
$\mathbf{X} \perp \mathbf{Y} \lvert \mathbf{Z}$	$\mathbf{X}$ is conditionally independent of $\mathbf{Y}$ given $\mathbf{Z}$

HMM Parameters

$\mathcal{M}$	HMM parameters set
$\mathcal{H}$	hypothesis, or word sequence $\{\mathcal{W}_1, \ldots, \mathcal{W}_K\}$
$\mathbf{o}_t$	observation vector at time t
D	dimension of feature vector $\mathbf{o}_t$
$\mathbf{O}$	observation sequence $\mathbf{O} = \{\mathbf{o}_1, \ldots, \mathbf{o}_T\}$
a_{ij}	discrete state transition probability from state i to state j
$b_j(\mathbf{o})$	output probability distribution at state j
ω_t	state at time t
$\boldsymbol{\omega}$	state sequence $\boldsymbol{\omega} = \{\omega_1, \ldots, \omega_T\}$
θ_t	Gaussian component at time t
$\boldsymbol{\theta}$	Gaussian component sequence $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_T\}$
m	Gaussian component index
$\boldsymbol{\mu}^{(m)}$	mean vector of the mth Gaussian component
$\boldsymbol{\Sigma}^{(m)}$	covariance matrix of the mth Gaussian component
$\gamma_t^{(m)}$	posterior probability of component m at time t

Adaptation and Environment Robustness

$\mathtt{s}$	acoustic factor $\mathtt{s}$
$\mathtt{s}_i$	the i-th condition within the acoustic factor $\mathtt{s}$
r	utterance or homogeneous block index
$\mathbf{O}^{(r)}$	observation sequence for the r-th homogeneous data block
$\mathcal{H}^{(r)}$	hypothesis for the r-th homogeneous data block
$\mathbf{W}^{(s)}$	transform for homogeneous data block s
$\boldsymbol{x}_t^{\mathtt{s}}$	clean speech static vector at time t
$\boldsymbol{y}_t^{\mathtt{s}}$	noise corrupted speech static vector at time t
$\boldsymbol{z}_t^{\mathtt{s}}$	reverberant noise corrupted speech static vector at time t
$\Delta\boldsymbol{x}_t$	clean speech delta vector at time t
$\Delta\boldsymbol{y}_t$	noise corrupted speech delta vector at time t
$\Delta\boldsymbol{z}_t$	reverberant noise corrupted speech delta vector at time t
$\mathbf{A}$	linear transform matrix
$\mathbf{b}$	bias vector
$\mathbf{W}$	affine transform, $\mathbf{W} = [\mathbf{A}\ \mathbf{b}]$
r_m	regression base class to which component m belongs
$\boldsymbol{\xi}_m$	extended mean vector of component m, $\boldsymbol{\xi}_m = [\boldsymbol{\mu}_m^{\mathsf{T}}\ \ 1]^{\mathsf{T}}$
$\boldsymbol{\zeta}_t$	extended observation vector at time t, $\boldsymbol{\zeta}_t = [\mathbf{o}_t^{\mathsf{T}}\ \ 1]^{\mathsf{T}}$
$\boldsymbol{\Phi}$	noise model parameters

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

Automatic speech recognition (ASR) has been extensively studied in the past several decades. Driven by both commercial and military interests, ASR technology has been developed and investigated on a great variety of tasks with increasing scales and difficulty. Since the middle of the 1970s, when hidden Markov models (HMMs) were introduced for speech recognition [14, 114], they have gradually become the dominant techniques for speech recognition and the recognition accuracies of ASR systems has been steadily improved [15, 76, 144]. As a result, the development of speech recognition techniques first focused on isolated speech recognition and was shifted to continuous speech recognition around the late 1980s. Since 1988, when two U.S. government agencies, the National Institute of Standards and Technology (NIST) and the Defense Advanced Research Project Agency (DARPA), started to organise world-wide evaluations on continuous speech recognition, many milestones in speech recognition have been set [15]. For example, the size of recognition vocabulary had increased from about 900 words in the Resource Management (RM) task (1988-1992)[195] to 20,000 words in the Wall Street Journal (WSJ) task (1993-1995) [52]. In more recent tasks, e.g., voice search [13, 247], virtually unlimited vocabulary size is a requirement. Besides vocabulary, the difficulty of these tasks has steadily increased: the ASR systems are required to operate in more realistic and practical conditions. For example, evaluations were primarily focused on read speech recorded in a clean condition before 1995 [144]. Since that time, research interests have gradually

switched to recognition of spontaneous and conversational speech in noisy environments. Broadcast news transcription (BN) [33, 77] and conversational telephone speech (CTS) [88] dictation are two examples. Driven by the commercial need, several more challenging tasks have emerged recently. These tasks include voice search [209], short message dictation [186] and YouTube transcription [8] and so on. Compared with previous speech recognition systems which were usually built on well-controlled and specifically collected data, the main data source to build these systems is *found data*. Found data refers to data recorded from natural conversation without a careful control or protocol of the collection procedure. Furthermore, as these systems are designed to be used in everyday-life scenarios, they need to operate in diverse acoustic environments and domains, serving the needs of a large number of users.

As speech recognition technology moves out of laboratories and is widely applied in more and more practical scenarios, many challenging technical problems emerge. One of these challenges is the acoustic diversity of speech data. For example, providing speech services for millions of users requires the system to have the ability to understand a broad range of voices spoken by users coming from different backgrounds, with accents and varying styles. Environmental noise is another major factor which contributes to the diversity of speech. As speech services are provided on various devices, ranging from telephones, desktop computers to tablets and game consoles, speech signals also exhibit large variations caused by channel characteristic differences. These non-speech variabilities are not related to what the users say, which introduces confusions to ASR systems and significantly degrades the recognition accuracy. This phenomenon severely restricts the usability of speech technology in everyday life. It happens when statistical models are built on training data which is largely mismatched with actual test data. *Adaptation* techniques have been used to quickly adapt ASR systems towards the test domain. This can be achieved by either normalising the feature vectors of speech or tuning acoustic model parameters to better reflect the nature of the test acoustic conditions. Adaptation has been intensively studied in the past decades. Schemes developed to adapt speech recognisers to specific speakers are often known as *speaker adaptation*, while schemes designed to handle impacts of environment are referred to as *environmental robustness*. Moreover, to build compact acoustic models which only model the generic speech variability on found data, *adaptive training* techniques are widely used. The idea is to separate the speech variability from the non-speech variabilities seen in found data, and build the canonical models accounting only for the speech variability, while the non-speech variabilities are "absorbed" by a set of model transforms.

This thesis will focus on the acoustic adaptation techniques for speech recognition. In particular, attempts have been made to address the following problems:

- **Robustness to reverberation:**
  Speech signals captured in enclosed environments by distant microphones are usually subject to reverberation. Compared with background noise and channel distortions, reverberant noise is highly dynamic and strongly correlated with the original clean speech signals. Speech recognition in reverberant environments thus remains a challenging problem and the recognition performance is still far from usable. As such, most of today's speech applications still require a microphone located near the speaker. This is inconvenient and troublesome in many applications, e.g., meeting speech recognition, lecture transcription and hands-free interfaces for consumer products. In some application scenarios, it is possible to deploy *microphone arrays* to capture speech signals. In this case, the microphone array processing techniques can be applied for speech recognition in reverberant environments. In other application scenarios, only *single distant microphones* are available. The first part of this thesis focuses on the second scenario, in which new reverberant noise robustness schemes are proposed to adapt acoustic models to the target reverberant environment.

- **Factorised acoustic model adaptation:**
  Due to the nature of found data, there are a large number of acoustic factors that can simultaneously affect the speech signals. These factors include speaker differences, channel distortions, background and reverberant noise and so on. It is clearly not possible to model the combined effect of all possible acoustic factors. Instead, the concept of acoustic factorisation [69] has been proposed to factorise the variabilities of acoustic factors. Factorisation offers additional flexibility to adapt acoustic models to diverse acoustic conditions. For example, the speaker information extracted from one environment can be efficiently used for the same speaker in another, possibly unseen, environment. This is referred to as *factorised adaptation*. This factorisation attribute relies on the independence between factor transforms. The second part of this thesis is thus to investigate how the independence attribute can be enforced to enable factorised adaptation.

As mentioned before, adaptation can be performed by normalising the feature vectors of test data or modifying the acoustic model parameters to better model the test conditions. The first is commonly referred to as the *feature-based approach* while the latter is usually referred to as the *model-based approach*. This thesis will focus on the model-based approach to addressing the above problems.

1.1 Thesis Organisation

This thesis is organised as follows. Chapter 2 describes standard acoustic modelling techniques for HMM-based ASR systems. HMM training and recognition algorithms are also discussed. Chapter 3 reviews acoustic model adaptation and robustness techniques. Speaker adaptation techniques, including maximum a posteriori (MAP) and linear transformed based schemes are first discussed. This is followed by a presentation of environmental robustness techniques in section 3.3. The impact of acoustic environments on speech recognition systems is first analysed in section 3.3.1. Previous work on improving the ASR robustness against background noise and channel distortion is discussed in section 3.3.2. Relevant work on robustness in reverberant environments is presented in section 3.3.3. Adaptive training techniques are also discussed in section 3.4.

Inspired by previous work, a new model-based approach to robust speech recognition in reverberant environments is proposed in Chapter 4. This includes: a novel model compensation scheme; an algorithm that performs maximum likelihood estimation of reverberant noise parameters; and an reverberant adaptive training algorithm. This forms the first contribution of this thesis. The second contribution of this thesis is presented in Chapter 5 and Chapter 6. In Chapter 5, the general concept of acoustic factorisation is first introduced and the independence between transforms for each factor is discussed as the key to factorisation. Two new approaches to enforcing the independence constraint are proposed in Chapter 6. The first one is a transform-constrained approach, which is based on the knowledge of how acoustic factors impact the speech signals. The second approach is based on a proposed explicit independence constraint and does not rely on prior knowledge of acoustic factors or the attributes of adaptation data. These two approaches are applied to speaker and noise factorisation problem in Chapter 6. Experimental validations of proposed schemes on both reverberant environment robustness and acoustic factorisation are presented in chapter 7 and 8. Finally, this thesis is concluded in Chapter 9, with a summary of contributions and a discussion of future work.

CHAPTER 2
Speech Recognition Systems

This chapter will introduce the speech recognition systems using hidden Markov models (HMMs) as acoustic models. Various acoustic modelling techniques are discussed, including front-end processing, choice of acoustic unit, state output distributions, and model parameter estimation algorithms. Using the acoustic models along with language models in decoding to find the best hypothesis is also discussed in this chapter.

2.1 Overview of ASR

The aim of an automatic speech recognition (ASR) system is to produce the most likely word sequence given an incoming speech signal. Figure 2.1 shows the architecture of an ASR system and its main components. In the first stage of speech recognition, input speech signals are processed by a front-end to provide a stream of acoustic feature vectors, or *observations*. These observations should be compact and carry sufficient information for recognition in the later stage. This process is usually known as *front-end processing* or *feature extraction*. In the second stage, the extracted observation sequence is fed into a decoder to recognise the mostly likely word sequence. Three main knowledge sources, i.e., *lexicon*, *language models*

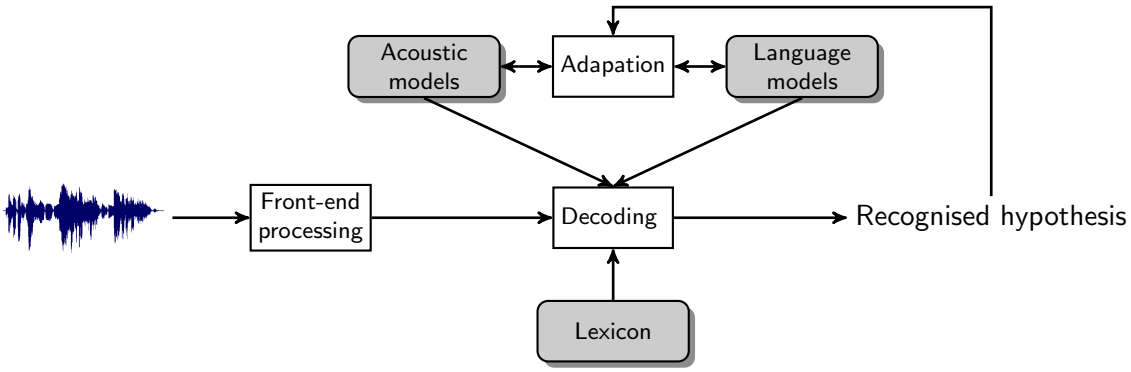


Figure 2.1: Architecture of an ASR system.

and *acoustic models*, are used in this stage. The lexicon, also known as the *dictionary*, is usually used in large vocabulary continuous speech recognition (LVCSR) systems to map sub-word units to words used in the language model. The language model represents the prior knowledge about the syntactic and semantic information of word sequences. The acoustic model represents the acoustic knowledge of how an observation sequence can be mapped to a sequence of sub-word units. In the mean time, additional knowledge like indications of speaker, environment or domains, can be incorporated into an *adaptation* module, so that both the acoustic and language models can better fit the current operating conditions.

Statistical approaches are widely used in modern ASR systems. In the statistical framework, the Bayesian decision rule is employed to find the most probable word sequence, $\hat{\mathcal{H}}$, given the observation sequence $\mathbf{O} = (\mathbf{o}_1, \ldots, \mathbf{o}_T)$:

$$\hat{\mathcal{H}} = \arg\max_{\mathcal{H}} P(\mathcal{H}|\mathbf{O}) \tag{2.1}$$

Following Bayes' rule, the posterior probability in the above equation can be expressed as a conditional probability of the word sequence given the acoustic observations, $p(\mathbf{O}|\mathcal{H})$, multiplied by a prior probability of the word sequence, $P(\mathcal{H})$, and normalised by the marginal likelihood of observation sequences, $p(\mathbf{O})$:

$$\begin{aligned} \hat{\mathcal{H}} &= \arg\max_{\mathcal{H}} \frac{p(\mathbf{O}|\mathcal{H})P(\mathcal{H})}{p(\mathbf{O})} \\ &= \arg\max_{\mathcal{H}} p(\mathbf{O}|\mathcal{H})P(\mathcal{H}) \end{aligned} \tag{2.2}$$

Note that the marginal probability, $p(\mathbf{O})$, has been discarded in the second equation since it is constant with respect to the ranking of hypotheses, and therefore does not alter the search for the best hypothesis. $p(\mathbf{O}|\mathcal{H})$ is calculated by the acoustic model and $P(\mathcal{H})$ is modelled by the language model.

This thesis will focus on adaptation and adaptive training of acoustic models. The rest of this chapter describes each module of speech recognition systems in Figure 2.1 in detail. Adaptation and environment robustness techniques are discussed in the next chapter.

2.2 Front-end Processing

Front-end processing is the first stage in a speech recognition system. The aim of front-end processing is to extract features that are optimal for the recognition task and (ideally) invariant to irrelevant factors, such as speaker differences and environment distortions. There are two main steps in front-end processing: *segmentation* and *feature extraction*. The segmentation step is used to isolate relevant speech segments from the whole audio stream. For example, in a telephone speech recognition system, a speech detector is normally used to detect the beginning and the end of speech events; in a broadcast news transcription system, a segmenter is needed to remove unwanted music and commercials. The identified speech segments are used in the second step to yield salient features for recognition.

Mel-frequency cepstral coefficients (MFCC)[39] and perceptual linear prediction (PLP) coefficients [95] are two commonly used speech feature representations in state-of-the-art speech recognition systems. Both forms are based on short-time spectral analysis and use a perceptionally motivated filter bank. In both types of feature extraction, a window function with a typical 10ms frame rate, is applied to the input audio stream and divides the audio into a series of overlapping *frames*. Each frame is usually 25-30ms long, in which the shape of vocal tract is assumed to be relatively constant. A first-order high pass filter is applied to accentuate the high frequencies in the formant structure. The windowed signal is analysed by a short time Fourier transform to obtain the power or magnitude spectrum. Since there is perceptional evidence which suggests that larger intervals are judged by human listeners to produce equal pitch increments [39] at a higher frequency, it is necessary to warp the normal frequency to a perceptually motivated frequency scale. The Mel-frequency scale is such a measurement which is widely used in speech recognition systems. It warps the normal frequency scale f_{Hz} by logarithmically compressing it to a linear scale to a perceived pitch f_{mel}:

$$f_{\text{mel}} = 1127 \log \left(1 + \frac{f_{\text{Hz}}}{700} \right) \tag{2.3}$$

When extracting MFCC features, a series of triangular band-passed filters, linearly spaced in the Mel scale, are convolved with the linear spectrum to produce a set of filter bank coefficients. A logarithmic transform is used to compress the dynamic range of these linear

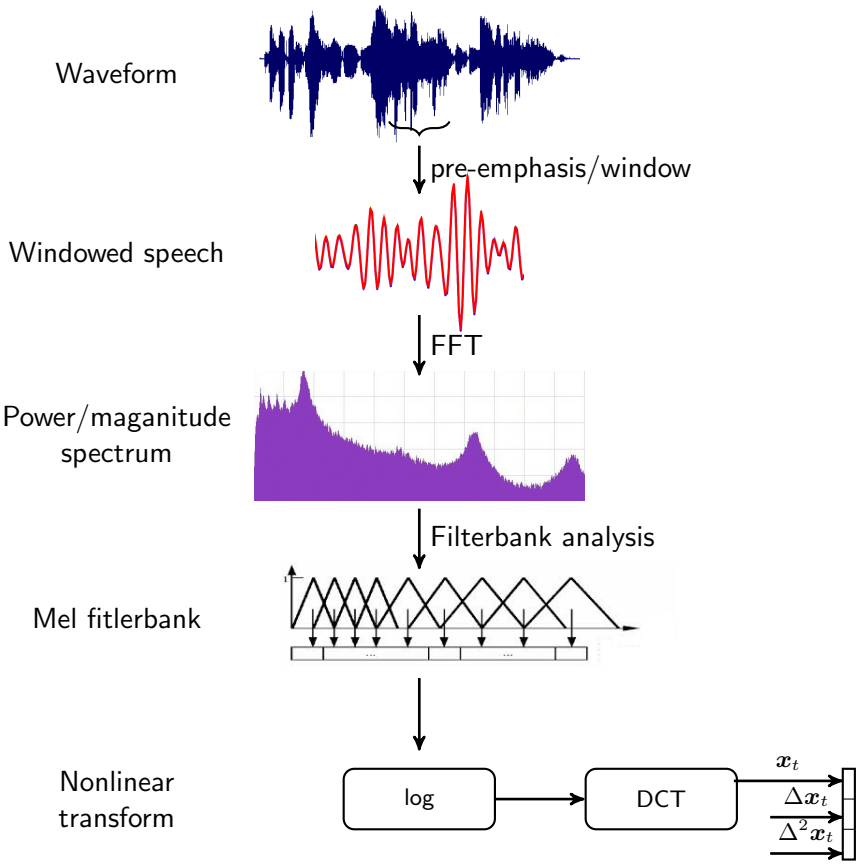


Figure 2.2: The MFCC feature extraction procedure.

filter bank coefficients, and this yields the so called log Mel-spectral filterbank coefficients, or filterbank features. It is found that the filterbank coefficients are highly correlated. This means the filterbank coefficient is a poor representation for Gaussian mixture models (GMM) with diagonal covariance matrices, which are widely used in the standard GMM-HMM based speech recognition systems. To decorrelate the feature, the discrete cosine transform (DCT) is applied to yield the cepstral coefficients, or MFCCs. Let $x_f^{(1)}$ be the f-th filterbank feature, x_d be the d-th cepstral feature. The d-th cepstral feature is calculated by:

$$x_d = \sqrt{\frac{2}{L_\mathrm{f}}} \sum_{f=0}^{L_\mathrm{f}-1} x_f^{(1)} \cos\left(\frac{\pi d}{L_\mathrm{f}}(f+0.5)\right) \quad d = 0, \cdots, L_\mathrm{c}-1 \tag{2.4}$$

where L_f is the number of filterbank features, L_c is the number of cepstral features. Normally, a truncated DCT transform is used, i.e., $L_\mathrm{c}(\sim 13)$ is smaller than $L_\mathrm{f}(\sim 24)$ and the high order cepstral coefficients are discarded. The 0-th cepstral coefficient is sometimes replaced by a normalised log energy. As a summary, Figure 2.2 illustrates the MFCC feature extraction procedure. Further details about MFCC feature extraction can be found in Appendix A.1.

PLP is another popular feature representation based on the short-time spectrum analysis. In PLP, the linear frequency of the power or magnitude spectrum is wrapped into another perceptually motivated scale, the Bark frequency scale, via:

$$f_{\text{bark}} = 6 \log \left(\left[\frac{f_{\text{Hz}}}{600} + 1 \right]^{0.5} + \frac{f_{\text{Hz}}}{600} \right) \tag{2.5}$$

Critical band filters spaced equally in the Bark frequency scale are used to filter the power or magnitude spectrum. The output of these filters are nonlinearly transformed, based on an equal-loudness and intensity-loudness power law. Linear prediction (LP) analysis is applied and the resulting LP coefficients are converted to cepstral coefficients. A modified form of PLP features is used in [257], where the Mel filterbank outputs are scaled by an equal-loudness curve and compressed by taking a cubic root. The resulting spectrum is used for LP analysis, and the LP coefficients are converted to cepstral coefficients, yielding the MF-PLP feature. It was found in [257] MF-PLP features to be more effective than the standard PLP feature.

2.2.1 Feature Post-processing

The MFCC or PLP features discussed in the previous section are often referred to as *static features*. Due to the conditional independence assumption (see section 2.3.1) in HMMs, if only static features are used, the speech dynamic information will not be incorporated into the recognition systems. A simple way to address this limitation is to append dynamic features, such as delta and delta-delta features, to the base, static, features[59]. Delta features $\Delta \boldsymbol{x}_t$ can be computed using simple differences, e.g., $\Delta \boldsymbol{x}_t = \boldsymbol{x}_t - \boldsymbol{x}_{t-1}$, or using a linear regression:

$$\Delta \boldsymbol{x}_t = \frac{\sum_{i=1}^{w} i(\boldsymbol{x}_{t+i} - \boldsymbol{x}_{t-i})}{2i^2} \tag{2.6}$$

where w is the window length. It is clear that this delta feature is a linear combination of $2w+1$ static features. Higher order coefficients, such as delta-delta features $\Delta^2 \boldsymbol{x}_t$, can be calculated using the delta features in a similar way. In many state-of-the-art ASR systems, a 13-dimensional MFCC static feature vector is augmented with the first and the second order derivatives, forming a feature vector $\mathbf{o}_t$ of 39 elements as follows:

$$\mathbf{o}_t = \begin{bmatrix} \boldsymbol{x}_t \\ \Delta \boldsymbol{x}_t \\ \Delta^2 \boldsymbol{x}_t \end{bmatrix} \tag{2.7}$$

The use of dynamic features introduces correlations between elements in the observation vector $\mathbf{o}_t$. The correlations reduce the discrimination ability of features. This is especially true when the features are modelled with GMMs using diagonal covariance matrices. *Linear projection* schemes are used to tackle this problem. In these schemes, feature vectors in

the original space are projected to a subspace in which features are assumed to be uncorrelated. These schemes include discriminant analysis (LDA)[32] and Heteroscedastic linear discriminant analysis (HLDA)[139].

Ideally, the extracted feature should be invariant to irrelevant acoustic factors, such as speaker / transducer differences and noise distortions. However, such attributes do not hold for widely used MFCC or PLP features. Feature normalisation techniques are commonly used to minimise the impact caused by irrelevant factors. For example, different transducers may have different impulse responses, which can be translated to bias vectors added on the observation vectors. Cepstral mean normalisation (CMN) [11] can be applied to remove this bias vector. Similarly, cepstral mean variance normalisation (CVN) can be applied to normalise the variances of input feature vectors. CMN and CVN normalise the first and the second order moments of the observation vectors, while higher order moments can be also normalised via Gaussianisation using a histogram matching[208] or a GMM-based approach[165]. Vocal Tract Length Normalisation (VTLN) [146] can be also used to compensate for the differences in vocal tract length and shape between speakers.

It is also possible to use a neural network as a feature extractor. For example, in the Tandem approach [98], MFCCs or PLPs are fed into neural network to extract more discriminant features, which are appended to the original MFCCs or PLPs to generate a Tandem feature. This will be briefly discussed in Section 2.3.1.3.

2.3 Acoustic Modelling

The previous section has discussed feature representations of speech signals. This section will discuss acoustic modelling for speech recognition, in which statistical models are used to calculate the probability of a sequence of observed feature vectors. Hidden Markov models (HMMs) are the most widely used statistical model in the speech processing area [196]. This section will discuss the basic concept of HMMs and their application to speech recognition.

2.3.1 Hidden Markov Models (HMMs)

In the HMM-based speech recognition, the observation vectors of a particular acoustic unit (e.g., a word or a phone) are assumed to be generated by a finite state machine. At each time instance, there is a hidden state. The hidden state can jump from the current state to other states according to certain probabilities. An observation vector is also generated at each time instance, according to a state-dependent output distribution. There are two fundamental assumptions when using this finite state machine to represent speech signals:

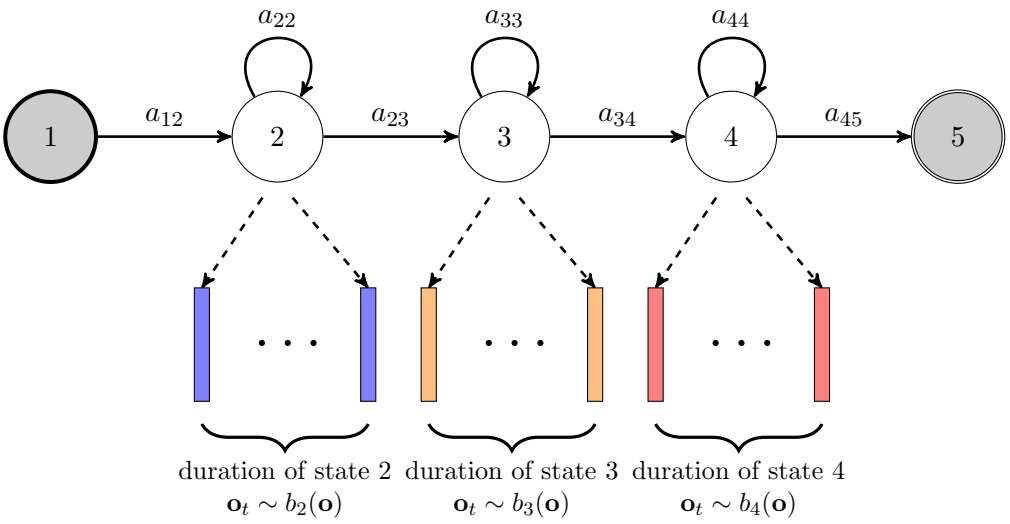


Figure 2.3: A left-to-right HMM with 3 emitting states.

- *Quasi-stationary.* This assumes that observation vectors within certain acoustic units can be segmented into several phases, in which the observation vectors are considered to be stationary.

- *Conditional independence.* Given the hidden states, the feature vectors follow the state-dependent output distributions. As a result, the feature vector at the current time is conditionally independent of the previous and following feature vectors, given the current hidden state.

Although neither of these two assumptions is true for real speech signals, HMM-based acoustic models are still very successful and they have dominated the speech recognition area since the 1980s [114].

2.3.1.1 Left-to-right HMMs

As speech signals are sequences in time, *left-to-right* HMMs are often used to model speech signals. Figure 2.3 shows an example of such an HMM with 3 emitting states. Let $\mathbf{O} = [\mathbf{o}_1, \cdots, \mathbf{o}_T]$ be a sequence of observation vectors that is generated by this 3-state left-to-right HMM, in which $\mathbf{o}_t$ is the observation vector at time t and T is the length of the speech sequence. The generation process starts from the first, non-emitting state, i.e., state 1. At each time, the state can jump to the next state or stay at the current state according to transition probabilities, a_{ij}. Here, a_{ij} denotes the probability of switching from state i to j. Once an emitting state j (e.g., states 2 to 4 in Figure 2.3) is reached, an observation vector is generated at the current time instant with a probability density $b_j(\mathbf{o})$. Note that the entry

and exit states in Figure 2.3 are non-emitting. The use of non-emitting states is to facilitate the construction of composite HMMs. It is clear that for the observation vector sequence $\mathbf{O}$ with the length T, there is a state sequence $\boldsymbol{\omega} = [\omega_1, \cdots, \omega_T]$ with the same length, where ω_t is the state at time t. However, only $\mathbf{O}$ can be observed, while $\boldsymbol{\omega}$ is hidden and needs to be inferred from the observations. The sequence of observation vectors and the sequence of hidden states can be written together as $\{\mathbf{O}, \boldsymbol{\omega}\}$ and will be referred to as the *complete data set*. The parameters of an N-state HMM include the following parts:

- ***A* – State transition probability matrix**
  The transition probability a_{ij} can be arranged into a state transition probability matrix $\boldsymbol{A}$, with its element at the j-th row and the i-th column defined as:

$$(\boldsymbol{A})_{ji} = a_{ij} = P(\omega_{t+1} = j | \omega_t = i) \tag{2.8}$$

  Note that a_{ij} does not dependent on the time index t. To be a valid probability distribution, each column of this matrix must satisfy

$$\sum_{j=1}^{N} P(\omega_{t+1} = j | \omega_t = i) = \sum_{j=1}^{N} a_{ij} = 1; \quad \forall i = 1, \cdots, N \tag{2.9}$$

  Note that because of the use of the entry state, state 1, a_{1j} specifies the initial state distributions of emitting states j, $j = 2, \cdots, N-1$.

- ***B* – State output probability distribution**
  At each emitting state j, a state-dependent output probability,

$$b_j(\mathbf{o}) = p(\mathbf{o} | \omega = j) \tag{2.10}$$

  is used to govern the observation generation process, in which ω is the current state. $b_j(\mathbf{o})$ can be a discrete distribution, which yields the so called discrete HMM (DHMM). Alternatively, $b_j(\mathbf{o})$ can a probability density function. This yields the so called continuous density HMM (CDHMM).

To use HMMs as acoustic models for speech recognition, there are a few practical considerations, e.g., the choice of acoustic units and state output probability distributions. These acoustic modelling techniques are briefly reviewed in the following sections.

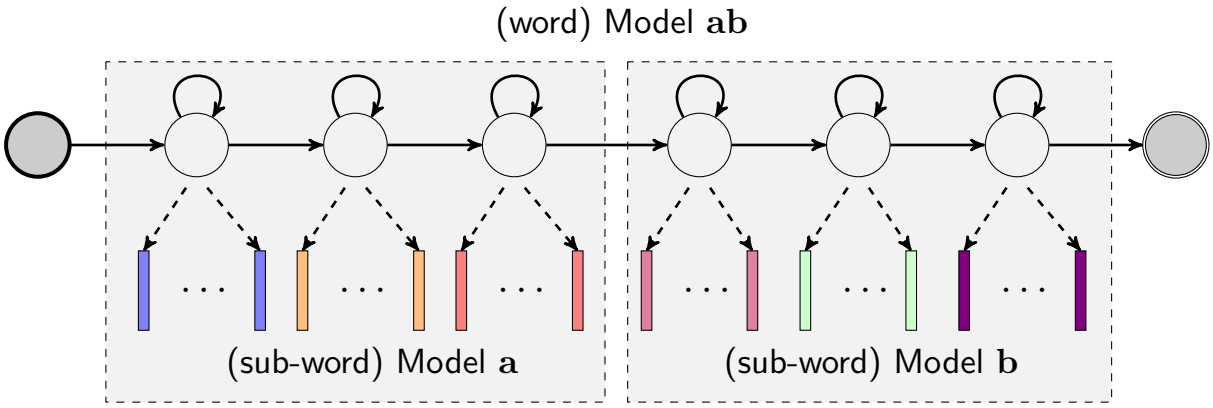


Figure 2.4: A composite HMM constructed from two individual HMMs.

2.3.1.2 Acoustic Units

For speech recognition tasks with a small recognition vocabulary (less than 1000 words), e.g., in a digit recognition task, HMMs are often used to model individual words. However, for speech recognition tasks with a medium (1K-10K words) vocabulary to a large vocabulary (> 10K words), it is not possible to collect sufficient training data for each word in the vocabulary. To solve this problem, HMMs are normally used to model *sub-word* units. Sub-word units are then composed to form word HMMs according to rules specified by a *dictionary*. As an example, the composition of two sub-word unit (**a** and **b**) HMMs to form a word (**ab**) HMM is illustrated in Figure 2.4. Note that in the figure, the non-emitting exit state of model **a** and the entry state of model **b** have been removed while the last emitting state of model **a** is connected to the first emitting state of model **b**. The entry state of model **a** and the exit state of model **b** become the new entry and exit state of the newly formed word model **ab**.

The phone, which is a small speech segment that has distinct perceptual properties, is often chosen as the sub-word unit. The number of phones is normally significantly smaller than the number of words in a vocabulary. For example, the number of phones in English is around 40 to 60, while typical state-of-the-art speech recognition systems for English use a vocabulary which ranges from 20K to 64K words. Given a phone set, it is possible to build one HMM for each phone, regardless of its contexts. This is referred to as a *monophone* system. However, the factorisation from a word HMM to context independent phone HMMs discards the contextual information. Due to co-articulation, the pronunciation of the current phone is influenced by the preceding and following phones and thus varies with respect to its context [145]. To model the variations caused by context, *context-dependent* phone sets are normally used in large vocabulary speech recognition systems. For instance, the *triphone* [145] is the

most widely used context-dependent phone set. A triphone is a context-dependent phone which uses the immediate left and right phone as the context for the phone in the centre position. For example, a phone **b** in the context **ey b l** is usually denoted as a triphone **ey-b+l**, where "**-**" denotes the left context and "**+**" denotes the right context. In this way, an isolated word "*able*" with silence "*sil*" at the start and the end of the word, can be mapped to a sequence of triphones as:

$$\textit{sil able sil} \rightarrow \textbf{sil ey b l sil} \rightarrow \textbf{sil-ey+b ey-b+l b-l+sil}$$

The triphone set can be further classified as a *word-internal triphone* set and a *cross-word triphone* set. In a word-internal triphone set, only the context within words is considered, while in a cross-word triphone set, the cross-word context is also considered. It has been demonstrated that cross-word triphone systems generally perform better than the word-internal triphone systems [256].

Using context-dependent triphone models significantly increases the number of acoustic units and thus requires a large amount of training data. For example, for a monophone set with 46 phones, the number possible triphones is around 100K. Moreover, some triphones may not exist in the training set. To solve this problem, *parameter tying* techniques are usually used. The most widely used parameter tying technique is *state clustering* [259]. The basic idea of state clustering is to share the state output distributions between similar acoustic units. This is illustrated in Figure 2.5 using a system with three acoustic units or HMMs. Initially, in the un-tied system, each state of each HMM has a unique output distribution. This gives 9 output distributions to be estimated. Clustering algorithms (e.g., [112, 259]) can be used to cluster these 9 distributions into several groups. In this example, the first and last states of 3 HMMs are clustered into groups 1 and 2 respectively, while the second states of the first and third HMMs fall into the third group, and the second state of the second HMM forms the fourth group. In this way, only 4 state distributions need to be estimated. Observation vectors belonging to the same group can be pooled together to estimate the parameters of one distribution. This ensures there are sufficient training data for each of the clustered state distributions.

There are generally two approaches can be used for state clustering. One is a *bottom-up* approach, in which clusters are built in a bottom-up fashion, e.g., [112]. Initially, every un-tied distribution is a class. The most similar distributions are then merged and a new distribution is generated. This process can be repeated until a pre-defined number of classes is reached. The main problem with this approach approach is that it can not appropriately handle unseen contexts which do not appear in the training data. This problem can be

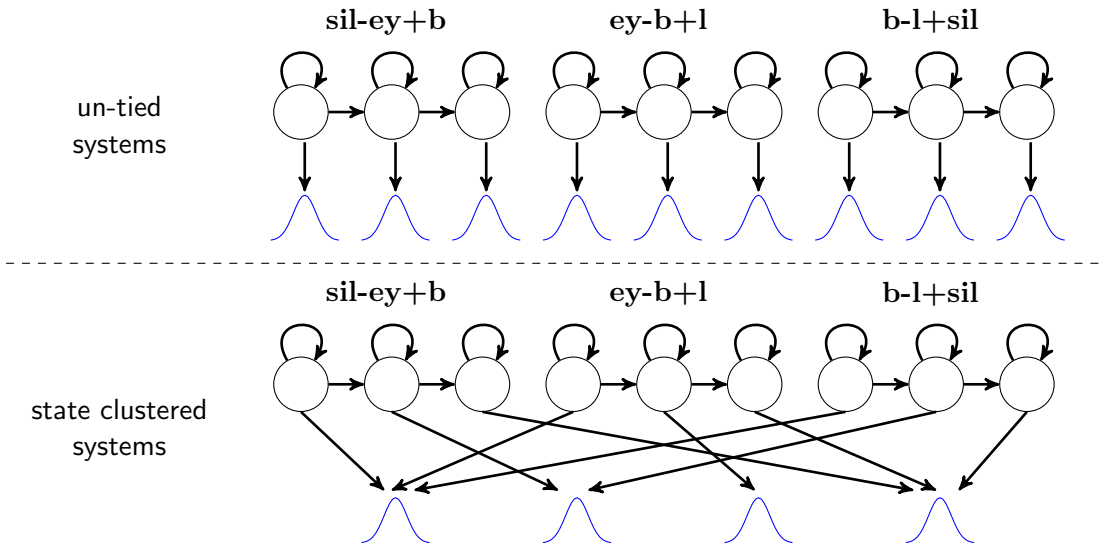


Figure 2.5: Illustration of state clustering for a three-HMM system. In the un-tied system, each state in each HMM has a unique state output distribution. After state clustering, some of the states share the same distribution.

addressed using the second, *top-down*, approach, e.g., the *phonetic decision tree clustering* method in [259]. Initially, all the states are grouped into a single root node. At each node, a phonetic question about the context of states is asked to split the states within the current node into left and right children nodes. For example, a question may ask whether the left context of the states in the current node is a nasal, i.e., in the set {**ng-***, **n-***, **m-***}; the states with a positive answer will be grouped into the left child's node while the states with a negative answer will be grouped into the right child's node. Many phonetic questions can be asked at each node. The best question which maximises the likelihood after splitting is selected. This process is repeated until the amount of training data associated with the current node falls below a threshold. Besides efficiency in clustering acoustic units, another advantage of using this top-down clustering is that it can easily handle unseen contexts. For example, when a new triphone is observed in test data, a series of phonetic questions can be asked to classify each state of this triphone into a leaf node and synthesis a new HMM for this triphone. The phonetic decision tree clustering method has been widely adopted in the most state-of-the-art ASR systems [109] and is also used in this thesis.

2.3.1.3 State Output Probability Distributions

A variety of probability distributions $b_j(\mathbf{o})$ can be used to calculated the j-th state emission probability. As most speech recognition systems use continuous acoustic feature vectors, a probability density function (PDF) is used to represent $b_j(\mathbf{o})$. The *multivariate Gaussian*

distribution is one commonly used form:

$$\begin{aligned} b_j(\mathbf{o}) &= \mathcal{N}(\mathbf{o}; \boldsymbol{\mu}^{(j)}, \boldsymbol{\Sigma}^{(j)}) \\ &= \frac{1}{\sqrt{(2\pi)^D |\boldsymbol{\Sigma}^{(j)}|}} \exp\left\{ -\frac{1}{2} (\mathbf{o} - \boldsymbol{\mu}^{(j)})^{\mathsf{T}} \boldsymbol{\Sigma}^{(j)-1} (\mathbf{o} - \boldsymbol{\mu}^{(j)}) \right\} \end{aligned} \tag{2.11}$$

where D is the dimension of $\mathbf{o}$, and $(\boldsymbol{\mu}^{(j)}, \boldsymbol{\Sigma}^{(j)})$ are the parameters of the j-th state output distribution. Usually the covariance matrix $\boldsymbol{\Sigma}^{(j)}$ is assumed to be diagonal. This is because using full covariance matrices in a large system is computationally expensive. Another reason is that a full covariance matrix has $\mathcal{O}(D^2)$ number of parameters, which requires a considerable amount of training data for robust covariance matrix estimation.

Using a single Gaussian distribution as the emitting PDF assumes that the output distribution only has one mode, which is a poor assumption in practice. Hence, Gaussian mixture models (GMMs) are widely used as the state emission PDFs [163]. A GMM-based state emission PDF can be expressed by:

$$b_j(\mathbf{o}) = p(\mathbf{o}|\omega = j) = \sum_{m=1}^{M_j} c_{jm} \mathcal{N}(\mathbf{o}; \boldsymbol{\mu}^{(jm)}, \boldsymbol{\Sigma}^{(jm)}) \tag{2.12}$$

where M_j is the number of Gaussian components in state j, and c_{jm} is the weight of component m in state j. To make $b_j(\mathbf{o})$ a valid probability distribution, the component weights must satisfy the following constraint:

$$\sum_{m=1}^{M_j} c_{jm} = 1 \quad \text{and} \quad c_{jm} \geq 0 \tag{2.13}$$

A HMM-based system which uses GMMs to represent the state distribution will be referred to as a *GMM-HMM system*.

In parallel with the development of using GMMs as the emission PDFs, using neural networks for density estimation became a popular alternative in the 1980s and 1990s. Different from the generative GMMs which model the conditional probability density of observation given the current state, $p(\mathbf{o}|j)$, various types of neural networks (e.g., [199, 202]) can be used to model the posterior probability of states given the observations, $p(j|\mathbf{o})$, which yields a discriminative model [23]. This approach is normally referred to as a *hybrid connectionist-HMM*, or a *hybrid* approach [27, 28]. Among a variety of neural networks with different architectures, the *multilayer perception* (MLP) is a popular choice. Figure 2.6 illustrates a hybrid MLP-HMM system with 3 context-dependent triphones. In this figure, a 3-layer MLP is used to estimate the posterior probability of context-dependent states from a window of

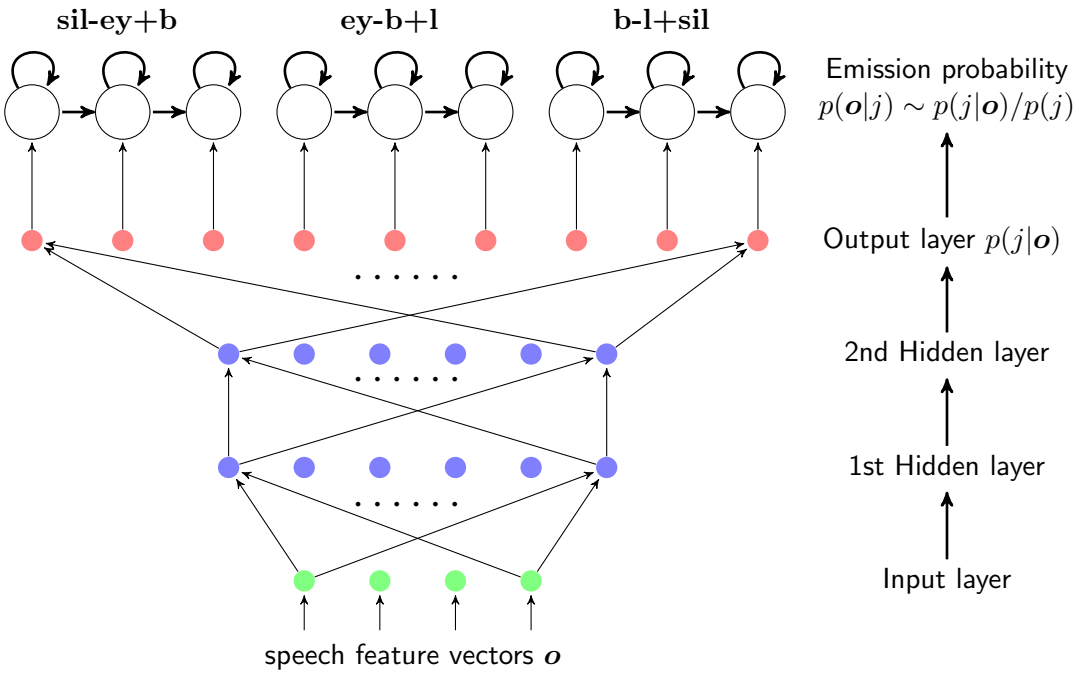


Figure 2.6: Illustration of a hybrid MLP/HMM system with 3 context-dependent triphones.

speech feature vectors. Once the state posterior probability is calculated, it can be converted to the output distribution using the Bayesian rule:

$$p(\mathbf{o}|j) = \frac{P(j|\mathbf{o})p(\mathbf{o})}{P(j)} \sim P(j|\mathbf{o})/P(j) \tag{2.14}$$

where $P(j)$ is the prior probability of the state j. Note the marginal distribution $p(\mathbf{o})$ can be ignored as it does not depend on a particular state.

The hybrid approach is a discriminative model which directly represents the posterior probability of hidden states. This can be compared with generative models which usually model the joint distribution of both hidden states and the observation vectors. It is argued that the discriminative model may provide a better use of model parameters [199]. The universal approximation attribute [36, 106] which states that a feed-forward network is able to approximate any continuous function, is another attractive advantage of the hybrid approach. However, there are a series of limitations which restricted the development of hybrid approach in the 1990s. As the use of context-dependent acoustic units become widely used in large vocabulary systems, the MLP needs to classify a large number of targets, which results in neural networks not easy to be trained at the time in the 1990s. To solve this issue, a context factorisation approach (e.g., [175]) was proposed, which nevertheless limits the potential improvement by the hybrid approach. Moreover, due to the discriminative nature

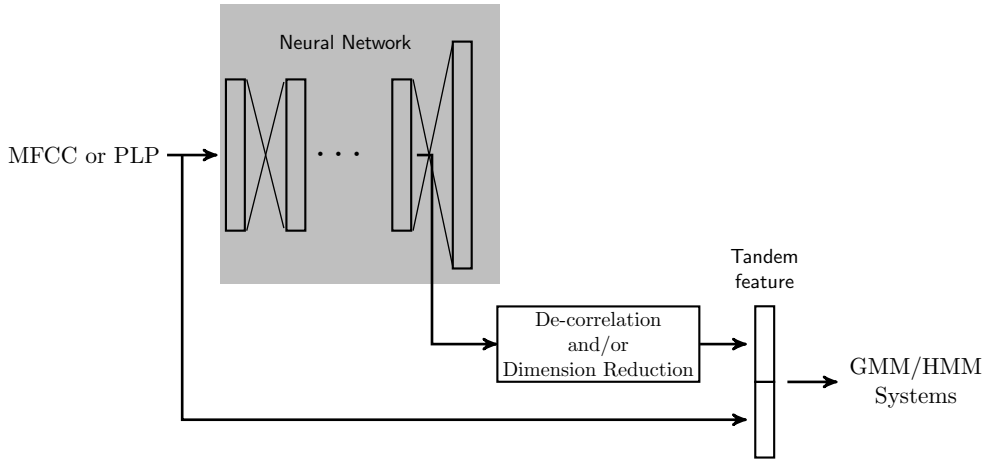


Figure 2.7: Generation of Tandem feature for GMM/HMM systems.

of neural network modelling, parameter training needs to consider not only the correct class but also all the competing classes. This is much more expensive than the maximum likelihood training for generative models. In addition, stochastic gradient descent methods, like back propagation [206], are widely used to train neural networks. The stochastic gradient descent procedure is very difficult to parallelize. Due to these reasons, early development of the hybrid approach normally focused on a shallow neural network (i.e., less than 2 hidden layers) and used context independent states as the output targets. The lack of effective and efficient adaptation techniques for the hybrid approach also contributes to its decline in the middle of the 1990s. Very recently, it has been demonstrated that the hybrid approach is able to deliver comparable, sometimes better, recognition performance compared with the well-developed GMM-HMM systems on a large vocabulary speech recognition task [215, 216]. Several factors, including increased computation power and a better network initialisation scheme [100], contribute to this recent breakthrough. It is now possible to train a deep neural network (with more than 5 hidden layers) to directly classify the (decision-tree clustered) context-dependent states [37]. This is referred to as a *deep neural network* (DNN) model.

Though it has been demonstrated that systems based on DNN models can provide better performance than the GMM-HMM systems on a number of tasks [99], it is difficult to adapt DNN models to the target operating conditions and many of the adaptation techniques developed for the GMM-HMM based systems can not be applied to the DNN model in a straightforward way [2, 157, 262]. As a result, an alternative way to use neural networks for speech recognition, the Tandem approach [98], is attractive. In this approach, neural networks are used as feature extractors, as shown in Figure 2.7. Features extracted by the neural network are appended to the MFCC or PLP features to form the *Tandem features*,

which are subsequently modelled by a GMM-HMM system. In this way, most of the adaptation techniques developed for the GMM-HMM systems can be also applied to the Tandem system as well, e.g., [245]. In the initial development of the Tandem approach, neural networks were used to classify monophones or monophone state targets [98] and the posteriors obtained by neural networks are combined with the MFCC or PLP feature. An alternative form was proposed in [58], where a bottleneck layer is introduced in which neural networks are constrained to have a narrow hidden layer, the bottleneck layer, in the middle and the linear output of that layer is taken as output instead of posteriors. It has been demonstrated that using Tandem feature provides considerable performance gains over MFCC or PLP features on a number of tasks [58, 189]. On the other hand, many of the techniques recently developed for DNN acoustic models can be also applied for the Tandem approach. For example, [261] used a deep neural network to classify context-dependent targets rather than the monophone targets to improve the bottleneck feature extraction.

As the development of using neural network for speech recognition has just (re-)started, the remaining part of this thesis will focus on the GMM-HMM systems for speech recognition.

2.3.2 Likelihood Calculation

Likelihood calculation is a basic function when using HMMs. Its aim is to calculate the likelihood of a sequence of observation vectors $\mathbf{O} = [\mathbf{o}_1, \cdots, \mathbf{o}_T]$ being generated according to the hypothesis $\mathcal{H}$, given the model parameters of HMMs $\mathcal{M}$. If the state sequence $\boldsymbol{\omega} = [\omega_1, \cdots, \omega_T]$ is known, the likelihood of the complete data $(\mathbf{O}, \boldsymbol{\omega})$ can be easily calculated. However, as the state sequence is hidden, the required likelihood must be computed as an expectation over all possible state sequences $\boldsymbol{\omega}$ associated with the hypothesis $\mathcal{H}$:

$$\begin{aligned} p(\mathbf{O}|\mathcal{H}, \mathcal{M}) &= \sum_{\boldsymbol{\omega} \in \mathcal{H}} p(\mathbf{O}, \boldsymbol{\omega}|\mathcal{M}) \\ &= \sum_{\boldsymbol{\omega} \in \mathcal{H}} a_{\omega_0 \omega_1} \prod_t a_{\omega_{t-1}\omega_t} b_{\omega_t}(\mathbf{o}_t) \end{aligned} \tag{2.15}$$

where $\boldsymbol{\omega} \in \mathcal{H}$ denotes a possible hidden state sequence according to the hypothesis $\mathcal{H}$, model parameters $\mathcal{M}$ includes the initial state distribution, the transition probability a_{ij} and the parameters for the state output distribution. Simply summing over all possible state sequences is not practical, as the number of paths is an exponential function $\mathcal{O}(N^T)$, where N is the number of states. The *forward-backward algorithm* [196] is used instead to compute the likelihood in an efficient way.

The forward-backward algorithm is based on a factorisation of the likelihood function in Eq. (2.15) into the forward probability and the backward probability:

$$\begin{aligned} p(\mathbf{O}|\mathcal{H},\mathcal{M}) &= \sum_{\boldsymbol{\omega}_1^T \in \mathcal{H}} p(\mathbf{O}_1^t|\mathbf{O}_{t+1}^T, \boldsymbol{\omega}_1^T) p(\mathbf{O}_{t+1}^T, \boldsymbol{\omega}_1^T) \\ &= \sum_{\boldsymbol{\omega}_1^T \in \mathcal{H}} p(\mathbf{O}_1^t|\boldsymbol{\omega}_1^t) p(\mathbf{O}_{t+1}^T, \boldsymbol{\omega}_{t+1}^T|\boldsymbol{\omega}_1^t) p(\boldsymbol{\omega}_1^t) \\ &= \sum_{\boldsymbol{\omega}_1^T \in \mathcal{H}} p(\mathbf{O}_1^t, \boldsymbol{\omega}_1^t) p(\mathbf{O}_{t+1}^T, \boldsymbol{\omega}_{t+1}^T|\omega_t) \\ &= \sum_{i:\omega_t=i} \left(\sum_{\boldsymbol{\omega}_1^{t-1}} p(\mathbf{O}_1^t, \boldsymbol{\omega}_1^t) \right) \left(\sum_{\boldsymbol{\omega}_{t+1}^T} p(\mathbf{O}_{t+1}^T, \boldsymbol{\omega}_{t+1}^T|\omega_t) \right) \\ &= \sum_{i:\omega_t=i} p(\mathbf{O}_1^t, \omega_t = i) p(\mathbf{O}_{t+1}^T|\omega_t = i) \end{aligned} \tag{2.16}$$

Here $\mathbf{O}_1^t$ and $\boldsymbol{\omega}_1^t$ denote the sequence of observation vectors and hidden states from 1 to time t, respectively. Note the second equation uses the following conditional independence:

$$\mathbf{O}_1^t \perp (\mathbf{O}_{t+1}^T, \boldsymbol{\omega}_{t+1}^T) \mid \omega_t \tag{2.17}$$

and the third equation is based on another conditional independence:

$$(\mathbf{O}_{t+1}^T, \boldsymbol{\omega}_{t+1}^T) \perp \boldsymbol{\omega}_1^{t-1} \mid \omega_t \tag{2.18}$$

The forward probability, $\alpha_t(i) = p(\mathbf{O}_1^t, \omega_t = i)$, can be calculated recursively using

$$\alpha_t(i) = \sum_{j=2}^{N-1} \alpha_{t-1}(j) a_{ji} b_i(\mathbf{o}_t) \qquad 1 < i < N \text{ and } 1 \leq t \leq T \tag{2.19}$$

and the initial and final condition:

$$\alpha_t(i) = \begin{cases} 1 & i = 1 \quad \text{and} \quad t = 0 \\ a_{1i} b_i(\mathbf{o}_t) & 1 < i < N \quad \text{and} \quad t = 1 \\ \sum_{j=2}^{N-1} \alpha_T(j) a_{jN} & i = N \quad \text{and} \quad t = T \end{cases} \tag{2.20}$$

In a similar way, the backward probability, $\beta_t(i) = p(\mathbf{O}_{t+1}^T|\omega_t = i)$, can also be computed using a recursion:

$$\beta_t(i) = \sum_{j=2}^{N-1} \beta_{t+1}(j) a_{ij} b_j(\mathbf{o}_t) \qquad 1 < i < N \text{ and } 1 < t \leq T-1 \tag{2.21}$$

and the initial and final condition:

$$\beta_t(i) = \begin{cases} a_{iN} & 1 < i < N \quad \text{and} \quad t = T \\ \sum_{j=2}^{N-1} a_{1j} b_j(\mathbf{o}_t) \beta_{t+1}(j) & i = 1 \quad \text{and} \quad t = 1 \end{cases} \tag{2.22}$$

The likelihood of the observation sequence can be obtained either from the forward or the backward probability as:

$$p(\mathbf{O}|\mathcal{H},\mathcal{M}) = \alpha_T(N) = \beta_1(1) \tag{2.23}$$

The posterior probability of being at state i at time t, $\gamma_t(i)$, can be also calculated by:

$$\gamma_t(i) = p(\omega_t = i|\mathbf{O},\mathcal{M},\mathcal{H}) = \frac{\alpha_t(i)\beta_t(i)}{\alpha_T(N)} \tag{2.24}$$

Similarly, the transition posterior, $\xi_t(i,j)$, i.e., the probability of transiting from state i to state j at time t, $\chi_t(i,j)$, can be expressed as:

$$\chi_t(i,j) = p(\omega_{t-1} = i, \omega_t = j|\mathbf{O},\mathcal{H},\mathcal{M}) = \frac{\alpha_{t-1}(i)a_{ij}b_j(\mathbf{o}_t)\beta_t(j)}{\alpha_T(N)} \tag{2.25}$$

These posterior probabilities are useful when estimating the model parameters, which will be discussed in the following section.

2.3.3 Parameter Estimation

In the previous section, likelihood calculation using a forward-backward algorithm is presented while the model parameters are assumed to be known. The model parameters need to be estimated from a set of training data $\{\mathbf{O}^{(1)},\cdots,\mathbf{O}^{(R)}\}$ with their associated hypotheses $\{\mathcal{H}^{(1)},\cdots,\mathcal{H}^{(R)}\}$. The model parameters can be estimated by maximising the likelihood of training data given their hypotheses. This is referred to as *maximum likelihood* (ML) training, which will be presented in section 2.3.3.1. A major limitation of ML training is that ML training does not consider the competing hypothesis and is not directly related to minimising the Bayes risk [30, 197]. A number of schemes which estimate model parameters by minimising different kinds of risk have been investigated in the literature [30, 191, 210, 236]. This is referred to as *discriminative training*, which will be discussed in section 2.3.3.2.

2.3.3.1 Maximum Likelihood Training

In maximum-likelihood training, the model parameters $\mathcal{M}$ are estimated by maximising the likelihood function given by

$$\hat{\mathcal{M}} = \arg\max_{\mathcal{M}} \sum_r \log p(\mathcal{O}^{(r)}|\mathcal{H}^{(r)},\mathcal{M}) \tag{2.26}$$

The above equation assumes that there are multiple utterances in the training data. Without loss of generality, in the following discussions, the summation over utterances and the superscript (r) are dropped for the sake of notational convenience.

Due to the existence of hidden states, directly optimising the above likelihood function using standard optimisation schemes is difficult. The Expectation Maximisation (EM) algorithm [42] is more efficient for this problem and thus widely used. An implementation of the EM algorithm for HMM models is also called the Baum-Welch algorithm [16]. In the EM algorithm, an auxiliary function $\mathcal{Q}(\mathcal{M};\mathcal{M}_k)$, is defined at the current model parameters $\mathcal{M}_k$ at the k-th iteration. This auxiliary function is a lower bound to the log-likelihood function. By maximising this lower bound with respect to $\mathcal{M}$, the $(k+1)$-th model parameters, $\mathcal{M}_{k+1}$, can be obtained. The new estimate of model parameters is guaranteed not to decrease the log-likelihood function. The auxiliary function is usually derived by introducing a variational distribution of the hidden variables $q(\boldsymbol{\omega})$ and using the Jensen's inequality:

$$\begin{aligned}\log p(\mathbf{O}|\mathcal{H},\mathcal{M}) &= \log\left(\sum_{\boldsymbol{\omega}\in\mathcal{H}} q(\boldsymbol{\omega})\frac{p(\mathbf{O},\boldsymbol{\omega}|\mathcal{H},\mathcal{M})}{q(\boldsymbol{\omega})}\right)\\ &\geq \sum_{\boldsymbol{\omega}\in\mathcal{H}} q(\boldsymbol{\omega})\log p(\mathbf{O},\boldsymbol{\omega}|\mathcal{H},\mathcal{M}) - \sum_{\boldsymbol{\omega}\in\mathcal{H}} q(\boldsymbol{\omega})\log q(\boldsymbol{\omega})\end{aligned} \tag{2.27}$$

The above inequality becomes an equality when the variational distribution $q(\boldsymbol{\omega})$ is the posterior distribution of hidden variables $p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M},\mathbf{O})$. However, it is often difficult to optimise $\mathcal{M}$ when the posterior distribution $p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M},\mathbf{O})$ is directly used. Instead, in the standard EM algorithm, an iterative procedure is used, where in the k-th iteration, the current model parameters $\mathcal{M}_k$ are used to calculate the posterior of hidden variables, i.e., $q(\boldsymbol{\omega}) = p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M}_k,\mathbf{O})$. This yields

$$\begin{aligned}\log p(\mathbf{O}|\mathcal{H},\mathcal{M}) \geq &\sum_{\boldsymbol{\omega}\in\mathcal{H}} p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M}_k,\mathbf{O})\log p(\mathbf{O},\boldsymbol{\omega}|\mathcal{H},\mathcal{M})\\ &- \sum_{\boldsymbol{\omega}\in\mathcal{H}} p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M}_k,\mathbf{O})\log p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M}_k,\mathbf{O})\end{aligned} \tag{2.28}$$

and

$$\begin{aligned}\log p(\mathbf{O}|\mathcal{H},\mathcal{M}_k) = &\sum_{\boldsymbol{\omega}\in\mathcal{H}} p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M}_k,\mathbf{O})\log p(\mathbf{O},\boldsymbol{\omega}|\mathcal{H},\mathcal{M}_k)\\ &- \sum_{\boldsymbol{\omega}\in\mathcal{H}} p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M}_k,\mathbf{O})\log p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M}_k,\mathbf{O})\end{aligned} \tag{2.29}$$

Therefore, the auxiliary function is defined as the first term in the right side of Eq. (2.28):

$$\mathcal{Q}(\mathcal{M};\mathcal{M}_k) = \sum_{\boldsymbol{\omega}\in\mathcal{H}} p(\boldsymbol{\omega}|\mathcal{H},\mathcal{M}_k,\mathbf{O})\log p(\mathbf{O},\boldsymbol{\omega}|\mathcal{H},\mathcal{M}) \tag{2.30}$$

Comparing the equations (2.28-2.29), it is obvious that the auxiliary function satisfies the following attribute:

$$\log p(\mathcal{O}|\mathcal{H},\mathcal{M}) - \log p(\mathcal{O}|\mathcal{H},\mathcal{M}_k) \geq \mathcal{Q}(\mathcal{M};\mathcal{M}_k) - \mathcal{Q}(\mathcal{M}_k;\mathcal{M}_k) \tag{2.31}$$

Algorithm 2.1: The EM algorithm.

Initialisation: $\mathcal{M} \leftarrow \mathcal{M}_0$ and $k = 0$

repeat

- $k \leftarrow k + 1$
- E-step:
  Compute $p(\boldsymbol{\omega}|\mathcal{H}, \mathcal{M}_{k-1}, \mathbf{O})$ and form the auxiliary function $\mathcal{Q}(\mathcal{M}; \mathcal{M}_{k-1})$;
- M-step:
  Update the model parameters by maximising the auxiliary function:

$$\mathcal{M}_k = \arg\max_{\mathcal{M}} \mathcal{Q}(\mathcal{M}; \mathcal{M}_{k-1})$$

until $\mathcal{Q}(\mathcal{M}_k; \mathcal{M}_{k-1}) - Q(\mathcal{M}_{k-1}; \mathcal{M}_{k-1}) \geq \varepsilon$;

Therefore maximising the auxiliary function $\mathcal{Q}(\mathcal{M}; \mathcal{M}_k)$ with respect to $\mathcal{M}$ is guaranteed not to decrease the log-likelihood.

In summary, the EM algorithm runs iteratively, and in each iteration two steps are performed. In the E-step, the posterior distribution of hidden variables is calculated using the current model parameters and an auxiliary function is formed in the E-step. In the M-step, this auxiliary function is maximised and the model parameters are updated. A summary of the general EM algorithm is illustrated in Algorithm 2.1.

The exact update formulae of model parameters using EM depend on the form of the model used. For the HMM , the auxiliary function can be expressed as [22]:

$$\mathcal{Q}(\mathcal{M}; \mathcal{M}_k) = \sum_{i,t} \gamma_t(i) \log b_i(\mathbf{o}_t) + \sum_{i,j,t} \chi_t(i,j) \log a_{ij} \tag{2.32}$$

where $\gamma_t(i)$ and $\chi_t(i,j)$ are the state occupancy posterior probability and the state pairwise posterior occupancy defined in Eq. (2.24) and Eq. (2.25) respectively. The optimal estimate of the state transition probability can be obtained by:

$$\hat{a}_{ij} = \frac{\sum_{t=2}^{T} \chi_t(i,j)}{\sum_{t=1}^{T} \gamma_t(i)} \tag{2.33}$$

and the estimate of state transition probability of non-emitting states is given by:

$$\hat{a}_{ij} = \begin{cases} \gamma_1(j) & i = 1 \quad 1 < j < N \\ \frac{\gamma_T(i)}{\sum_{t=1}^{T} \gamma_t(i)} & 1 < i < N \quad j = N \end{cases} \tag{2.34}$$

When a GMM is used as the state output distribution, Gaussian mixture components can be regarded as another level of hidden variables. The posterior probability that $\mathbf{o}_t$ belongs to the state j and component m, $\gamma_t(j,m)$, is given by:

$$\gamma_t(j,m) = \gamma_t(j) \frac{c_{jm} \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}^{(jm)}, \boldsymbol{\Sigma}^{(jm)})}{b_j(\mathbf{o}_t)} \tag{2.35}$$

where $b_j(\mathbf{o}_t)$ is the state output distribution defined in Eq. (2.12). The re-estimation formulae for the parameters of the GMM at state j and component m are thus given by:

$$\hat{c}_{jm} = \frac{\sum_t \gamma_t(j,m)}{\sum_{m,t} \gamma_t(j,m)} \tag{2.36}$$

$$\hat{\boldsymbol{\mu}}^{(jm)} = \frac{\sum_t \gamma_t(j,m)\mathbf{o}_t}{\sum_t \gamma_t(j,m)} \tag{2.37}$$

$$\hat{\boldsymbol{\Sigma}}^{(jm)} = \text{diag}\left(\frac{\sum_t \gamma_t(j,m)(\mathbf{o}_t - \hat{\boldsymbol{\mu}}^{(jm)})(\mathbf{o}_t - \hat{\boldsymbol{\mu}}^{(jm)})^{\mathsf{T}}}{\sum_t \gamma_t(j,m)}\right) \tag{2.38}$$

As only the mean and the diagonal covariance matrix of each distinct Gaussian component are of interest in this thesis, a new notation, $\boldsymbol{\theta} = [\theta_1, \cdots, \theta_T]$ is introduced to denote the hidden Gaussian component sequence. Now, θ_t denotes the distinct Gaussian component at time t, and m will be used to denote the index of each distinct Gaussian component. Using this notation, the likelihood calculation can be re-expressed as:

$$\begin{aligned} p(\mathbf{O}|\mathcal{H},\mathcal{M}) &= \sum_{\boldsymbol{\theta}} p(\mathcal{O},\boldsymbol{\theta}|\mathcal{H},\mathcal{M}) \\ &= \sum_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{H},\mathcal{M}) \prod_t p(\mathbf{o}_t|\mathcal{M},\theta_t) \end{aligned} \tag{2.39}$$

and a component level auxiliary function can be also written as:

$$\mathcal{Q}(\mathcal{M};\mathcal{M}_k) = -\frac{1}{2}\sum_{t,m} \gamma_t^{(m)} \left\{\log|\boldsymbol{\Sigma}^{(m)}| + (\mathbf{o}_t - \boldsymbol{\mu}^{(m)})^{\mathsf{T}}\boldsymbol{\Sigma}^{(m)-1}(\mathbf{o}_t - \boldsymbol{\mu}^{(m)})\right\} \tag{2.40}$$

where $\gamma_t^{(m)}$ is the m-th component occupancy at time t, calculated using the current model parameters $\mathcal{M}_k$.

2.3.3.2 Discriminative Training

ML training of HMMs maximises the likelihood of the data for the given reference hypothesis. However, this does not guarantee that the likelihood of data for the competing hypotheses is also minimised. Rather than maximising the likelihood of training data for the reference hypothesis, discriminative training will consider the posterior probability of the reference hypothesis and explicitly consider the recognition accuracy metric. A number of discriminative criteria have been proposed and investigated in the past, e.g., [170, 184, 191, 210]. It has been demonstrated that discriminative training provides consistent performance gains over ML training and is widely used in the state-of-the-art speech recognition systems. The following will briefly present some of the most widely used criteria.

- **Maximum Mutual Information (MMI):**
  In MMI training [30, 184, 236], the posterior probability of the correct hypothesis for the given training data is maximised, i.e.,

$$\begin{aligned}\mathcal{F}_{\text{mmi}}(\mathcal{M}) &= \sum_r \log P(\mathcal{H}^{(r)}|\mathbf{O}^{(r)}, \mathcal{M}) \\ &\approx \sum_r \log \frac{p(\mathbf{O}^{(r)}|\mathcal{H}^{(r)}, \mathcal{M})^\kappa p(\mathcal{H}^{(r)})}{\sum_{\check{\mathcal{H}}\in\mathbb{H}} p(\mathbf{O}^{(r)}|\check{\mathcal{H}}, \mathcal{M})^\kappa p(\check{\mathcal{H}})}\end{aligned} \tag{2.41}$$

  where $\mathbf{O}^{(r)}$ and $\mathcal{H}^{(r)}$ are the r-th observation sequence and associated hypothesis respectively, and $\mathbb{H}$ represents all the possible hypothesis or the denominator hypothesis. The scaling factor κ is used to smooth the posterior probability of correct hypothesis to improve discriminative training's generalization [255]. It is usually set as the inverse of the language model scaling factor normally used in decoding. The denominator hypothesis $\mathbb{H}$ is usually approximated by N-Best list [211] or lattices [255]. Since a lattice is a more compact representation of competing hypothesis, it is widely used in discriminative training.

  An interesting variant of MMI is the boosted MMI criterion proposed in [193]. Instead of maximising the posterior probability of the correct hypothesis in MMI training, the boosted MMI training is designed to maximise the following criterion:

$$\mathcal{F}_{\text{bmmi}}(\mathcal{M}) = \sum_r \log \frac{p(\mathbf{O}^{(r)}|\mathcal{H}^{(r)}, \mathcal{M})^\kappa p(\mathcal{H}^{(r)})}{\sum_{\check{\mathcal{H}}} p(\mathbf{O}^{(r)}|\check{\mathcal{H}}, \mathcal{M})^\kappa p(\check{\mathcal{H}}) \exp(-b\mathcal{R}(\check{\mathcal{H}}, \mathcal{H}^{(r)}))} \tag{2.42}$$

  where $\mathcal{R}(\check{\mathcal{H}}, \mathcal{H}^{(r)})$ is the risk or cost associated with assigning recognition output $\check{\mathcal{H}}$ while the ground truth is $\mathcal{H}^{(r)}$, b is a control parameter. $\exp(-b\mathcal{R}(\check{\mathcal{H}}, \mathcal{H}^{(r)}))$ is a boosting factor which increases the likelihood of hypotheses with more errors; therefore it improves the generalization of MMI training. The boosted MMI criterion can be also viewed as imposing a soft margin which is proportional to the number of errors in a hypothesis [93].

- **Minimum Classification Error (MCE):**
  In MCE training [116], the classification error is minimised. This gives the following criterion:

$$\mathcal{F}_{\text{mce}}(\mathcal{M}) = \sum_r f\left(\log \frac{p(\mathbf{O}^{(r)}|\mathcal{H}^{(r)}, \mathcal{M})^\kappa p(\mathcal{H}^{(r)})}{p(\mathbf{O}^{(r)}|\check{\mathcal{H}}^{(r)}, \mathcal{M})^\kappa p(\check{\mathcal{H}}^{(r)})}\right) \tag{2.43}$$

  where $f(\cdot)$ is the sigmoid function:

$$f(z) = \frac{1}{1+\exp(-\gamma z)} \tag{2.44}$$

with γ being a tuning parameter, and $\check{\mathcal{H}}^{(r)}$ is the best competing hypothesis for the r-th utterance. The denominator is usually approximated by a soft-max over N best hypotheses:

$$p(\mathbf{O}^{(r)}|\check{\mathcal{H}}^{(r)},\mathcal{M})^{\kappa}p(\check{\mathcal{H}}^{(r)}) \approx \frac{1}{\eta}\log\left(\frac{1}{N}\sum_{\mathcal{H}\neq\mathcal{H}^{(r)}}\Big(p(\mathbf{O}^{(r)}|\mathcal{H},\mathcal{M})^{\kappa}p(\mathcal{H})\Big)^{\eta}\right) \tag{2.45}$$

where the summation is over the top N best hypotheses, excluding the correct hypothesis. The scaling factor η is used to consider more confusing patterns during training. When the competing hypotheses are represented in a lattice form, it is difficult to exclude the correct hypothesis from the denominator lattices. Some heuristics have been proposed to tackle this problem, e.g., [210].

- **Minimum Bayes Risk (MBR):**
  In MMI and MCE training, the risk related to the sentence error rate is minimised. This can be illustrated that the same risk is associated with one incorrect hypothesis, regardless how many errors are made in the incorrect hypothesis. However, in speech recognition, the ultimate goal is to minimise the word error rate (WER). The MBR criterion considers more general error metrics in the objective function. The Bayesian risk or expected loss can be expressed by:

$$\mathcal{L}_{\text{mbr}}(\mathcal{M}) = \sum_{r}\sum_{\mathcal{H}} P(\mathcal{H}|\mathcal{M},\mathbf{O}^{(r)})\mathcal{R}(\mathcal{H},\mathcal{H}^{(r)}) \tag{2.46}$$

  where $\mathcal{R}(\mathcal{H},\mathcal{H}^{(r)})$ computes the risk or cost between the hypothesis $\mathcal{H}$ and the reference hypothesis $\mathcal{H}^{(r)}$. When this loss function computes the WER between $\mathcal{H}$ and $\mathcal{H}^{(r)}$, this becomes the *minimum word error* (MWE) criterion [192]. When the loss function is defined as phone error rate between two hypothesis, this becomes the popular *minimum phone error* (MPE) criterion [191]. Calculation of either word or phone error rate involves aligning the hypothesis with the reference, which is time consuming when the alternative hypotheses are represented by lattices. A heuristic measure has been used to approximate the loss function using a local error measure, which removes the need for alignment [191].

Parameter estimation under these discriminative criteria have been extensively investigated in the past [94, 115]. As the discriminative criteria involves the competing hypotheses, the estimation algorithm is much more complicated that the ML training algorithm. The extended Baum-Welch (EBW) algorithm and its extensions [85, 184] are widely used. For complex criteria, such as MPE, the estimated model parameters are usually interpolated with

the ML or MMI-trained models to avoid over-training. There are several overview papers summarizing the status and progress in discriminative training, e.g., [94, 115].

2.4 Recognition of Speech Using HMMs

The above sections briefly reviewed the HMMs and the associated parameter estimation algorithms. This section will discuss techniques used to find the most likely hypothesis using HMMs. This is also known as the *decoding* problem. The aim of decoding is to find the best hypothesis given the observed acoustic data. The search follows the Bayesian decision rule: the recognised word sequence $\hat{\mathcal{H}}$ is the one gives the highest posterior probability after observing the acoustic vectors $\mathbf{O}$, i.e.,

$$\hat{\mathcal{H}} = \arg\max_{\mathcal{H}} P(\mathcal{H}|\mathbf{O}, \mathcal{M}) \,. \tag{2.47}$$

Using Bayes' rule, this is achieved by

$$\begin{aligned} \hat{\mathcal{H}} &= \arg\max_{\mathcal{H}} p(\mathbf{O}|\mathcal{H}, \mathcal{M}) P(\mathcal{H}) \\ &= \arg\max_{\mathcal{H}} \{\log p(\mathbf{O}|\mathcal{H}, \mathcal{M}) + \log p(\mathcal{H})\} \end{aligned} \tag{2.48}$$

where $\log p(\mathbf{O}|\mathcal{H}, \mathcal{M})$ is the log-likelihood of the observation sequence $\mathbf{O}$ given the acoustic model $\mathcal{M}$ and word hypothesis $\mathcal{H}$, which is also referred to as the *acoustic score*; $\log P(\mathcal{H})$ is the prior probability of the hypothesis, or the *language score*, calculated using a *language model*. The combination of the acoustic score with the language score gives the overall score. Recognition of speech is done by searching the best hypothesis with the maximum overall score.

2.4.1 Language Modelling

The language model is a discrete probability model, defined on strings of all allowable symbols. Let $\mathcal{H} = (\mathcal{W}_1, \cdots, \mathcal{W}_K)$. The probability of $\mathcal{H}$ can be factorised into a product of conditional probabilities:

$$P(\mathcal{H}) = \prod_{k=1}^{K} P(\mathcal{W}_k|\mathcal{W}_{k-1}, \cdots, \mathcal{W}_1) \tag{2.49}$$

Assuming the word sequences have the n-order Markovian properties [167], the probability of each conditional probability is approximated by

$$P(\mathcal{W}_k|\mathcal{W}_{k-1}, \cdots, \mathcal{W}_1) \approx P(\mathcal{W}_k|\mathcal{W}_{k-1}, \cdots, \mathcal{W}_{k-n+1}) \,. \tag{2.50}$$

and the probability of the given hypothesis can be expressed by

$$P(\mathcal{H}) = \prod_{k=1}^{K} P(\mathcal{W}_k|\mathcal{W}_{k-1}, \cdots, \mathcal{W}_{k-n+1}) \tag{2.51}$$

This is referred to as the n-gram language model.

The ML estimation of probability of word $\mathcal{W}_k$ given its history up to $n-1$ previous words is given by

$$P(\mathcal{W}_k|\mathcal{W}_{k-1}, \cdots, \mathcal{W}_{k-n+1}) = \frac{f(\mathcal{W}_k, \mathcal{W}_{k-1}, \cdots, \mathcal{W}_{k-n+1})}{\sum_{\mathcal{W}} f(\mathcal{W}, \mathcal{W}_{k-1}, \cdots, \mathcal{W}_{k-n+1})}, \tag{2.52}$$

where $f(\mathcal{W}_k, \mathcal{W}_{k-1}, \cdots, \mathcal{W}_{k-n+1})$ denotes the frequency count of the n-gram word sequence in the training text data. However, to get a robust estimate of these conditional probabilities, a good coverage of all possible n-grams is needed. For a large vocabulary system, some n-grams are rarely seen, even for a large text corpus. As a consequence, smoothing of the conditional probabilities is necessary. These smoothing schemes can be divided into the following three categories:

- **Discounting**

  When ML estimation is used, if some n-grams are unobserved in the training data, then the conditional probabilities of those n-grams will become 0. To solve this problem, a certain amount of overall probability mass is allocated to the unseen n-grams. The ratio of re-allocated probabilities to the overall probabilities is controlled by a discounting factor. Popularly used discounting approaches include Good-Turing discounting[84, 124], Kneser-Ney smoothing [131], and absolute discounting[182].

- **Back-off**

  Back-off is to make use of shorter histories instead of assigning probabilities mass to the unseen (thus less likely) n-grams in the above discounting approaches. The back-off strategy can be applied recursively. For example, a 4-gram distribution can be backed-off to tri-gram, bi-gram or uni-gram distributions.

- **Interpolation**

  High-order n-gram language models can be interpolated with low-order n-gram language models to construct a more reliable estimation. Similarly, several language models estimated using different sources of text data can be interpolated. The interpolation weights are often tuned on a separate held-out dataset.

Though generative language models discussed above have been widely used, discriminative language models, such as neural network language models (NNLMs) [19] and recurrent neural network language models (RNNLMs) [171] have recently gained more and more attention. In the neural network language model (NNLM), each word $\{\mathcal{W}_{k-i} : i = 1, \cdots, n-1\}$ is mapped to a vector $\boldsymbol{w}_i$ in a continuous space. Give these word embedding vectors $[\boldsymbol{w}_1, \cdots, \boldsymbol{w}_{n-1}]$, a feed-forward neural network is used to predicate the posterior probability of $\mathcal{W}_k$. In the recurrent neural network language model, a recurrent connection is added between the input layer and the hidden layers [171]. Due to this recurrent connection, RNNLMs are able to model long-term dependency. Since neural network based language models are still being developed, this thesis will use the well-developed n-gram language models in all the experiments.

2.4.2 Decoding

Decoding is to search for the best path that gives the highest likelihood. This is achieved by the decoding algorithm [240]. As demonstrated in Figure 2.1, there are three main components used in the decoding algorithm: the acoustic model based on HMMs which gives the acoustic score of the observation sequence given a phone sequence, the lexicon which specifies how a word sequence is composed by individual phones and the language model which gives the probability of a word sequence. Moreover, a word $\mathcal{W}_k$ may have more than one phone representation because of the multiple pronunciation variants; meanwhile, a sequence of HMM phone models may have more than one possible state sequences. Hence, the Eq. (2.48) can be expanded as a multiple level marginalisation expressed in the following equation:

$$\hat{\mathcal{H}} = \arg\max_{\mathcal{H}} \left\{ P(\mathcal{H}) \sum_{\boldsymbol{\phi} \in \mathcal{H}} P(\boldsymbol{\phi}|\mathcal{H}) \sum_{\boldsymbol{\omega} \in \boldsymbol{\phi}} p(\mathbf{O}, \boldsymbol{\omega}|\boldsymbol{\phi}) \right\} \tag{2.53}$$

where $\boldsymbol{\phi}$ is a possible phone sequence of the hypothesis $\mathcal{H}$ and $\boldsymbol{\omega}$ is an allowable state sequence of the phone sequence $\boldsymbol{\phi}$; $P(\boldsymbol{\phi}|\mathcal{H})$ is a pronunciation model [88]; $P(\mathcal{H})$ and $p(\mathbf{O}, \boldsymbol{\omega}|\boldsymbol{\phi})$ are the language and the acoustic scores respectively. To avoid an infeasible computational cost for direct evaluation of all possible phone and state sequences in Eq. (2.53), the *Viterbi* approximation [240] is used, in which the summation over possible paths is approximated by a maximum:

$$\hat{\mathcal{H}} \approx \arg\max_{\mathcal{H}} \left\{ P(\mathcal{H}) \max_{\boldsymbol{\phi} \in \mathcal{H}} P(\boldsymbol{\phi}|\mathcal{H}) \max_{\boldsymbol{\omega} \in \boldsymbol{\phi}} p(\mathbf{O}, \boldsymbol{\omega}|\boldsymbol{\phi}) \right\} \tag{2.54}$$

The Viterbi algorithm [240] can be used to calculate the likelihood of the best state sequence within an HMM or a given HMMs sequence. For example, to find the best state

sequence $\hat{\boldsymbol{\omega}}$ for an observation sequence $\mathbf{O} = [\boldsymbol{o}_1, \cdots, \boldsymbol{o}_T]$, a partial best-path score $\phi_j(t)$ is defined as:

$$\phi_j(t) = \max_{\omega_1, \cdots, \omega_{t-1}} p(\mathbf{o}_1, \cdots, \mathbf{o}_t, \omega_1, \cdots, \omega_t = j) \tag{2.55}$$

This partial best-path score can be recursively calculated by

$$\phi_j(t) = \max_i \{\phi_i(t-1)a_{ij}\} b_j(\mathbf{o}_t) \qquad 1 < j < N, 1 \leq t < T \tag{2.56}$$

$$\varphi_j(t) = \arg\max_i \{\phi_i(t-1)a_{ij}\} \tag{2.57}$$

with an initialisation

$$\phi_1(0) = 1 \tag{2.58}$$

$$\phi_j(1) = a_{1j} b_j(\mathbf{o}_t) \tag{2.59}$$

where N is the number of states in the given HMM sequence; $\varphi_j(t)$ stores the best previous state for a partial path ending at state j at time t. The likelihood of the best path is given by $\phi_N(T)$, while the best state sequence $(s_1, \cdots, s_T)$ can be retrieved by the following recursion:

$$s_T = N \tag{2.60}$$

$$s_t = \varphi_{s_{t+1}}(t+1), \quad t = T-1, \cdots, 1 \tag{2.61}$$

The Viterbi algorithm can be also extended to continuous speech recognition. An implementation of this algorithm is known as the *token passing* algorithm [181, 260]. In this algorithm, each state has one or more tokens at each time instance. Each token carries a word-end link and the value of the partial path it represents. When multiple tokens are merged at the same node, only the most likely token is propagated. When a token is propagated through an arc linking two words, the word-end link is updated to record the last word. At the end of an utterance, the most likely sequence of words can be traced back using the word-end link of the token with the highest score. By propagating not only the best token but also a few tokens with top scores at every node, it is also possible to generate N-best hypotheses or word lattices [185]. The multiple hypotheses can be recovered by retrieving the history of tokens which arrive at the end of an utterance. Though it will not generate the exact n-best search paths, as it implicitly assumes the start time of each word is independent of all words before [228], it is a reasonable approximation and can be implemented very efficiently.

The complexity of the decoding network can be considerably increased when a high order n-gram language model is used. This is because tokens can only be merged when its n-1 previous words are identical as the word probability depends on the word history in the

language model. To reduce the decoding complexity, unpromising tokens, whose scores are below a certain threshold or *beam-width* below the current best score, can be removed. This is referred to as *pruning* [181]. However, if the beam-width is too tight, the most likely path may be pruned at an early stage, resulting in *search errors*. Therefore, the beam-width is often set to balance the computational cost and the search errors. A review of decoding algorithms can be found in [181].

There are also a number of practical issues in decoding. For instance, the acoustic score and the language score have different dynamic ranges: the language score is often much smaller than the acoustic score. To handle this problem, the language scores are often scaled up. The scaling factor is often empirically set for a particular task. Similarly the pronunciation probability can be also scaled using a separate scaling factor. Another issue is the use of a word insertion penalty to avoid recognition errors from a number of short words. With these techniques, the most probable hypothesis is searched by

$$\hat{\mathcal{H}} = \arg\max_{\mathcal{H}} \left\{ \alpha \log P(\mathcal{H}) + \beta \log \max_{\boldsymbol{\phi} \in \mathcal{H}} P(\boldsymbol{\phi}|\mathcal{H}) + \log \max_{\boldsymbol{\omega} \in \boldsymbol{\phi}} p(\mathbf{O}, \boldsymbol{\omega}|\boldsymbol{\phi}) + \gamma L_{\mathcal{H}} \right\} \tag{2.62}$$

where α is the language scaling factor, β is the pronunciation probability scaling factor, γ is the insertion penalty, and $L_{\mathcal{H}}$ is the length of the word sequence $\mathcal{H}$.

2.4.3 Evaluation

The quality of an ASR system on a particular task is often measured by comparing the hypothesised transcriptions and reference transcriptions. Word error rate (WER) is the most widely used metric. The two word sequences (hypothesis and reference) are aligned using a dynamic programming-based string alignment algorithm [168]. After the alignment, the number of deletions (D), substitutions (S), and insertions (I) can be determined. The deletions, substitutions and insertions are all considered as errors, and the WER is calculated by the rate of the number of errors to the number of words (N) in the reference, i.e.,

$$\text{WER} = \frac{\text{S} + \text{I} + \text{D}}{N} \times 100\% \tag{2.63}$$

WERs are used to measure the quality of ASR systems throughout this thesis.

In some scenarios, it is interesting to compare the quality of two ASR systems. Apart from comparing the WERs of each ASR system, it is also possible to perform a significance test to determine whether the system with a lower WER is *statistically* better than the other system. The *matched-pairs* significant test [82] is widely used in the speech processing community. It will be also used in this thesis.

2.5 Summary

This chapter has reviewed the basic concepts and fundamental of speech recognition using hidden Markov models (HMMs). An overview of speech recognition systems is first presented. Front-end processing of speech signals are discussed. This is followed by an introduction of HMM for acoustic modelling, including the basic concept of HMMs, the choice of acoustic units and the state output distributions, and the likelihood calculation. Maximum likelihood (ML) estimation of model parameters using the EM algorithm is presented. To overcome the limitations of the ML criterion, discriminative training has been discussed. A review of using HMMs in recognition is also given. The concept of n-gram language models is presented. With the acoustic model and the language models, the Viterbi algorithm is used to search for the best hypothesis. Some practical issues in decoding are also discussed.

CHAPTER 3
Acoustic Model Adaptation and Robustness

Statistical algorithms which are used to model speech variability for speech recognition have been discussed in Chapter 2. Speech variability will be referred to as *intrinsic* or desired variability in this thesis. A fundamental assumption underlying the statistical approaches discussed in the previous chapter is that the training and testing data are sampled from the same distribution. However, this assumption is poor in real-life scenarios where speech signals can exhibit not only intrinsic variability but also *extrinsic* variabilities. Here the extrinsic variabilities stand for the variabilities caused by various *acoustic factors*, for example, speaker differences, transmission channels and background noise. These acoustic factors may change during test, which results in mismatches between training and testing conditions. To combat this problem, there has been a great amount of research aiming to improve model robustness against distortions caused by various acoustic factors, e.g., [71, 83, 143, 254]. In particular, approaches developed to adapt speech recognition systems to specific speakers are often known as *speaker adaptation* [254], while approaches designed to compensate for environmental effects

are referred to as *environmental robustness* [4]. These approaches seek to either normalise features (referred to as feature-based approaches) or transform acoustic models (referred to as model-based approaches) towards target acoustic conditions. This chapter will review both speaker adaptation (in section 3.2) and environmental robustness (in section 3.3) techniques with a focus on model-based approaches. Note that speaker adaptation techniques are usually referred to as *adaptation* schemes while environment robustness techniques are commonly called as *compensation* schemes in the literature. However these approaches have become similar in recent years and share many of the same attributes. Therefore, they will be used in this thesis interchangeably. Besides adaptation schemes which are designed to adapt acoustic models to the target conditions, an adaptive training framework [9] can be also used to build acoustic models on found data. The challenge of building acoustic models on found data is that these data exhibit a broad range of variabilities. With this adaptive training framework, generic acoustic models can be effectively built to accommodate the wide range of variabilities in found data. This framework will be presented and discussed in section 3.4.

3.1 Adaptation in Speech Recognition

As discussed at the beginning of this chapter, speech signals can be distorted by a variety of acoustic factors. As a result, it is possible to associate a set of acoustic factors, such as speaker or environment tags [1] with a spoken utterance. Utterances with the same acoustic factors can then be grouped together as they are in the same *acoustic condition*. An important assumption in adaptation is that the extrinsic variabilities for a particular acoustic condition have the same statistical properties, and adaptation must be performed for each homogeneous data block separately. In practice, incoming test data are often non-homogeneous and it is necessary to first cluster the test data into homogeneous blocks. For some applications, such as broadcast news transcription, there is no information that can be directly used to partition the input data. For these applications, automatic segmentation and clustering methods have been developed to partition test data into homogeneous block [226, 234, 235]. For some other applications, there is partial information associated with the input data which can be used for segmentation and clustering. For instance, in voice search applications, it is feasible to identify which user makes the query; in conversational telephone speech (CTS) tasks, the speech signals are often split into speaker sides. Automatic clustering methods, e.g.[219], are still needed in these applications to determine the partition according to additional acoustic factors, such as the environment. In this thesis, as the focus is on adaptation schemes, it

[1] In this thesis, it is assumed that acoustic factors are stationary within the same utterance.

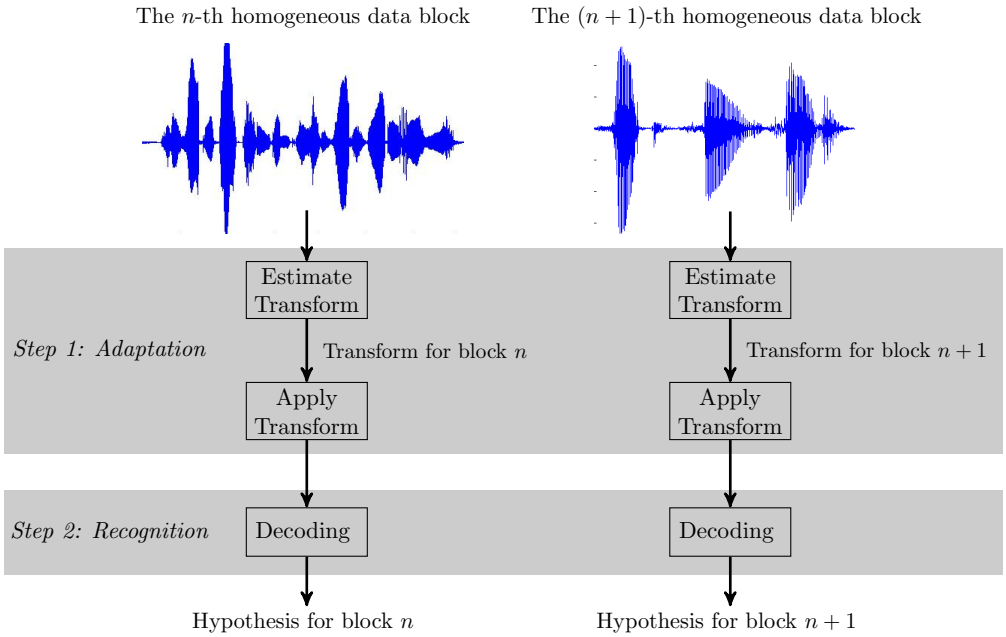


Figure 3.1: A two-step adaptation and recognition process applied on two homogeneous data blocks within the adaptation framework.

is assumed that input data have already been segmented into homogeneous blocks and the underlying acoustic factors associated with these blocks have already been labelled.

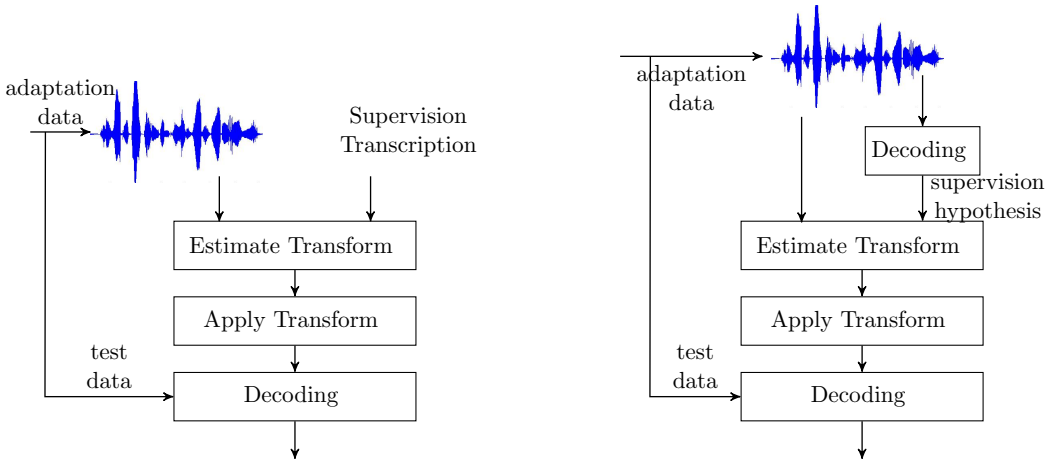
Once the test data are split into homogeneous blocks, a two-step procedure, illustrated in Figure 3.1, is used in recognition within the adaptation framework. In the first step, features or acoustic models are transformed for each block. In most adaptation schemes, the transform parameters need to be estimated during the test stage. These parameters are often estimated on the adaptation data sampled from the same test acoustic condition. The transformed features or acoustic models are subsequently used in the second step to recognise the data from the same acoustic condition.

Depending on how the transform is estimated and applied, there are several adaptation modes:

- *Off-line and online adaptation*: in *off-line adaptation*, all the adaptation data is assumed to be available at once. Transforms are estimated and applied before recognition on test data start coming in. As adaptation data comes in bulk, this is also referred to as *static or batch adaptation*. In *online adaptation* mode, recognition systems must start to produce transcription before all the adaptation data is available and adaptation data comes in stages. In this scenario, adaptation is often performed in an *incremental* fashion, in which information about acoustic factors are estimated in one stage and

propagated into the next stage [264]. Compared with batch-mode adaptation, this incremental adaptation is able to track slowly changing acoustic conditions.

- *supervised and unsupervised adaptation*: depending on whether the the correct transcription corresponding to the adaptation data is available, adaptation can be performed in a *supervised* or *unsupervised* mode. In supervised adaptation, the correct transcription is available, and the quality of adaptation depends on the amount of supervision data. A typical scenario where supervised adaptation can be performed is when a user enrolls himself or herself (e.g., read some sentences according to the prompts given by the system) before using speech recognition systems. As supervised adaptation often requires this additional step, unsupervised adaptation is more widely used in practice. In this scenario, the supervision transcription must be derived from the recogniser output. In this case, the quality of adaptation depends not only on the amount of adaptation data, but also the quality of recognised hypothesis. When the recognised hypothesis contains errors, the adapted systems will be biased towards the erroneous hypothesis, which degrades recognition performance. A comparison between supervised and unsupervised adaptation is illustrated in Figure 3.2.



(a) Supervised adaptation

(b) Unsupervised adaptation

Figure 3.2: Supervised and unsupervised adaptation

Adaption can be performed by model-based approaches or feature-based approaches or a combination of both [207]. Model-based approaches modify the parameters of acoustic models so that the adapted models better reflect the target, test, condition. Feature-based approaches are used to normalise acoustic features such that the mismatch between training and the target testing conditions can be reduced. Model-based approaches are generally more powerful than

feature-based approaches as they have more modelling power to represent acoustic variations and can take uncertainty caused by those variations into account [71]. On the other hand, feature-based approaches are usually computationally cheap and require few changes to the decoder. This thesis will focus on the model-based adaptation approaches.

3.2 Speaker Adaptation

The difference in speaker characteristics is one of the major factors which contributes to the variabilities of speech signals. Even for the same word uttered by different speakers, significant variations can be perceived. These variations are often related to the speakers' voice, age, gender, dialect, emotion speaking rate and style [109].

It is observed that there are large variations in recognition performance among different speakers. For instance, recognition on speech uttered by non-native speakers[109] often yields a high error rate, due to the mismatch between training and test conditions. Recognition accuracies can be improved using various schemes. For example, it is possible to train a gender dependent (GD) system [256], where male and female speakers are recognised by the acoustic models according to their gender; it is also found that a speaker dependent (SD) system in which each speaker has his or her own acoustic model performs better than a speaker-independent (SI) system, provided that there is enough data to train the acoustic models for each speaker [254]. However, in practice, it is not feasible to collect enough data for every speaker to train such a SD system. Instead, a speaker adapted (SA) system is usually used [110], where a well-trained acoustic model is adapted to each speaker using a small amount of speaker specific data. This will be referred to as speaker adaptation.

A number of speaker adaptation techniques have been developed, e.g., [67, 70, 74, 79, 148]. These techniques can be grouped into three families: maximum a posterior (MAP) adaptation [79], linear transform-based adaptation [67, 74, 148] and speaker cluster-based adaptation [70] or eigenvoices [183]. They will be briefly reviewed in the following.

3.2.1 Maximum a Posterior Adaptation

As adaptation data can be also viewed as additional training data, the simplest adaptation method is to perform additional ML re-estimation iterations. However, this is problematic as the amount of adaptation data is often small, compared with the number of re-estimated parameters. Bayesian methods [21] can be used to tackle the data sparsity problem. In the Bayesian schemes such as [111, 248], the model parameters $\mathcal{M}$ are treated as random parameters with a prior probability $p(\mathcal{M}|\boldsymbol{\Theta})$, where $\boldsymbol{\Theta}$ is a set of hyper-parameters. Upon

the observation of adaptation data $\mathbf{O}$ and the associated supervision transcription $\mathcal{H}$, the prior probability $p(\mathcal{M}|\boldsymbol{\Theta})$ is updated to a posterior probability $p(\mathcal{M}|\mathbf{O},\mathcal{H})$. The likelihood of test data $\mathbf{O}'$ for a hypothesis $\mathcal{H}'$ is given by marginalising over $\mathcal{M}$, i.e.,

$$p(\mathbf{O}'|\mathbf{O},\mathcal{H}') = \int p(\mathcal{M}|\mathbf{O},\mathcal{H})p(\mathbf{O}'|\mathcal{M},\mathcal{H}')\,d\,\mathcal{M} \tag{3.1}$$

A limitation of these Bayesian methods is that the marginalisation in Eq. (3.1) is normally computationally expensive. Instead of considering all possible model parameters, *maximum a posterior* (MAP) estimation has been proposed in [79], in which a point estimate of model parameters, the MAP estimator, is used. The MAP estimation of model parameters is given by:

$$\mathcal{M}_{\text{map}} = \arg\max_{\mathcal{M}} \{p(\mathcal{M}|\mathbf{O},\mathcal{H}\} = \arg\max_{\mathcal{M}} \{p(\mathbf{O}|\mathcal{M},\mathcal{H})p(\mathcal{M}|\boldsymbol{\Theta})\} \tag{3.2}$$

It is important to choose an appropriate prior distribution in MAP. For mathematical convenience, a conjugate prior is often selected as the posterior probability is in the same form as the prior probability. Due to the hidden variables in HMMs, a finite dimensional conjugate prior to the likelihood of observations does not exist [132]. Instead, independence between the Gaussian components' parameters is assumed. In this case, conjugate priors to the likelihood function of the complete dataset can be obtained. In continuous density GMM-HMMs, the conjugate prior for each individual Gaussian component $\mathcal{N}(\boldsymbol{\mu}^{(m)},\boldsymbol{\Sigma}^{(m)})$, is given by the Normal-Wishart distribution [251]:

$$\begin{aligned} p(\boldsymbol{\mu}^{(m)},\boldsymbol{\Sigma}^{(m)}|\boldsymbol{\Theta}) \propto |\boldsymbol{\Sigma}^{(m)}|^{-0.5(\alpha^{(m)}-D)} \exp\left(-\frac{1}{2}\text{tr}(\mathbf{R}^{(m)}\boldsymbol{\Sigma}^{(m)-1})\right) \\ \times \exp\left(-\frac{\tau^{(m)}}{2}(\boldsymbol{\mu}^{(m)}-\boldsymbol{m}^{(m)})^{\mathsf{T}}\boldsymbol{\Sigma}^{(m)-1}(\boldsymbol{\mu}^{(m)}-\boldsymbol{m}^{(m)})\right) \end{aligned} \tag{3.3}$$

where D is the dimensional of observation vector, $\boldsymbol{\Theta} = \{\alpha^{(m)},\tau^{(m)},\mathbf{R}^{(m)},\boldsymbol{m}^{(m)}\}$ is the set of hyper-parameters and $\alpha^{(m)} > D-1, \tau^{(e} > 0$. The hyper-parameters can be obtained from subjective knowledge of the underlying stochastic process, following the subjective Bayesian principle [194]. However in many cases this knowledge is not available. The empirical Bayesian approach [200, 201] can be used to estimate the hyper-parameters by maximising the marginal likelihood of training data, i.e.

$$\boldsymbol{\Theta}^* = \arg\max_{\boldsymbol{\Theta}} p(\mathbf{O}|\mathcal{H}) = \arg\max_{\boldsymbol{\Theta}} \int p(\mathbf{O}|\mathcal{H},\mathcal{M})p(\mathcal{M}|\boldsymbol{\Theta})\,d\mathcal{M} \tag{3.4}$$

As the maximising the marginal likelihood of observations is hard due to hidden variables in HMMs, various approximation methods (e.g., variational Bayes [17, 179, 248]) can be used.

After the prior probability is fully specified, MAP estimation can be obtained in a similar way as the ML estimation, using the EM algorithm. The auxiliary function of MAP estimation is obtained by adding the prior term to the auxiliary function of ML estimation, i.e.:

$$\mathcal{Q}(\mathcal{M};\hat{\mathcal{M}}) = \log p(\mathcal{M}|\boldsymbol{\Theta}) - \frac{1}{2}\sum_{t,m}\gamma_t^{(m)}\left\{\log|\boldsymbol{\Sigma}^{(m)}| + (\boldsymbol{o}_t - \boldsymbol{\mu}^{(m)})^{\mathsf{T}}\boldsymbol{\Sigma}^{(m)-1}(\boldsymbol{o}_t - \boldsymbol{\mu}^{(m)})\right\} \quad (3.5)$$

where $\hat{\mathcal{M}}$ are the current model parameters, $\mathcal{M}$ are the new estimate of model parameters; $\gamma_t^{(m)}$ is the ML posterior occupancy of t-th observation vector, $\boldsymbol{o}_t$, in component m, which is obtained using $\hat{\mathcal{M}}$. The MAP estimation of a mean vector is thus given by:

$$\boldsymbol{\mu}^{(m)} = \frac{\sum_t \gamma_t^{(m)}\boldsymbol{o}_t}{\sum_t \gamma_t^{(m)} + \tau^{(m)}} + \frac{\tau^{(m)}\boldsymbol{m}^{(m)}}{\sum_t \gamma_t^{(m)} + \tau^{(m)}} \quad (3.6)$$

where $\tau^{(m)}$ and $\boldsymbol{m}^{(m)}$ are the hyper-parameters. It can be observed from Eq. (3.6) that the adapted model parameters are obtained by interpolating the mean of the prior probability and the sufficient statistics of observed adaptation data. When there is only a small amount of adaptation data available for component m, the adapted mean vector is close to the mean of the prior distribution. As more data becomes available, the MAP estimation of a mean vector tends towards its ML estimator. The MAP estimator will asymptotically converge to the ML estimator when the quantity of data approaches infinity.

One limitation of MAP adaptation can also be seen from Eq. (3.6). MAP adaptation only updates the Gaussian components which are observed in the adaptation data. As many Gaussian components are used in a typical large vocabulary speech recognition, most of the Gaussian components will be unobserved when only a small amount of adaptation data is used. Standard MAP adaptation requires a considerable amount of adaptation data to update all the parameters; therefore the adaptation is usually slow and not suitable for rapid adaptation. Extensions of MAP have been proposed to tackle this issue using different methods. For example, regression based model prediction (RMP)[7] first finds the linear regression relationship between HMM parameters and then uses this relationship to update rarely observed, or unobserved, Gaussian components. Alternatively, structured MAP (SMAP) [223] arranges all the Gaussian components in a tree structure. A mean offset and a diagonal covariance scaling vector are recursively computed for each layer of the tree, starting from the root node down to a leaf node. At each node, the distribution from its parent node is used as a prior. It is shown that in practice SMAP converges to the standard MAP performance when a sufficient amount of adaptation data is available and it gives better performance than MAP when only a limited amount of data is used [223].

3.2.2 Linear Transform-based Speaker Adaptation

The standard MAP scheme may be viewed as a "local" approach to adaptation, where model parameters are usually separately adapted. This approach may give good recognition performance when there is a sufficient amount of data, but it leads to slow adaptation speed. In contrast to this local approach, linear transform-based adaptation schemes use linear transforms to adapt a large amount, sometimes all the, Gaussian components. By sharing linear transforms between many Gaussian components, these schemes are able to adapt both observed and unobserved Gaussian components, which improves adaptation speed. Additionally, using the same linear transforms for a large number of Gaussian components is relatively robust to the errors in supervision hypothesis. This is suitable for unsupervised adaptation.

There is some flexibility in the form of linear transforms. When adaptation data is limited, linear transform can be performed in a translation form (or scaling for variances) using bias vectors (or scaling vectors), e.g., [207]. When a small amount of adaptation data is available, a diagonal transform with a bias term(e.g., [49, 125]) can be used. When there is a sufficient amount of data, a more powerful transform with a full matrix can be used. The linear transform with a full matrix can model the change of correlations. A block diagonal transform is a trade-off between modelling power and data requirement, in which the correlations between certain parts of a vector is assumed to be unchanged. For example, in a system that uses static, delta and delta-delta cesptral coefficients, separate linear transforms can be performed for these three parts of features. This is expressed in the following form of a block diagonal matrix:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A_s} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A_\Delta} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A_{\Delta^2}} \end{bmatrix} \tag{3.7}$$

where $\mathbf{A_s}$, $\mathbf{A_\Delta}$ and $\mathbf{A_{\Delta^2}}$ are the matrices for static, delta and delta-delta coefficients respectively. It is also possible to control the complexity of linear transforms by sharing the linear transforms among different Gaussian components. This is usually done by using a regression tree, which will be presented in section 3.2.4.

Depending on how the linear transforms modify the model parameters (mean and/or variance) or the observation vectors, various schemes have been developed. Some of the most popular forms are described in the following. For convenience, it is assumed that a full matrix transform is used and all the Gaussian components share the same linear transform.

3.2.2.1 Maximum Likelihood Linear Regression (MLLR)

MLLR was originally designed to adapt the mean vectors [148] and was extended to variance adaptation [67, 74] later. In this thesis, the term "MLLR" is used to refer to only mean

MLLR. In MLLR, the mean vector of m-th Gaussian component, $\boldsymbol{\mu}^{(m)}$, is adapted in the following form:

$$\hat{\boldsymbol{\mu}}^{(m)} = \mathbf{A}\boldsymbol{\mu}^{(m)} + \boldsymbol{b}^{(m)} = \mathbf{W}\boldsymbol{\xi}^{(m)} \tag{3.8}$$

where $\hat{\boldsymbol{\mu}}^{(m)}$ is adapted mean vector, $\mathbf{W} = [\mathbf{A}, \boldsymbol{b}]$ is the linear transform, and $\boldsymbol{\xi}^{(m)} = [\boldsymbol{\mu}^{(m)\mathsf{T}}, 1]^{\mathsf{T}}$ is the extended mean vector.

MLLR transforms are estimated by maximising the likelihood of adaptation data as

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} \{\log p(\mathbf{O}|\mathcal{H}; \mathcal{M}, \mathbf{W}\} \tag{3.9}$$

In EM, this optimisation is achieved by maximising the following auxiliary function [148]:

$$\mathcal{Q}(\hat{\mathbf{W}}; \mathbf{W}, \mathcal{M}) = K + \sum_{m,t} \gamma_t^{(m)} \log \mathcal{N}(\boldsymbol{o}_t; \hat{\mathbf{W}}\boldsymbol{\xi}^{(m)}, \boldsymbol{\Sigma}^{(m)}) \tag{3.10}$$

where $\mathbf{W}$ and $\hat{\mathbf{W}}$ are the current and new estimate of the linear transform respectively; $\gamma_t^{(m)}$ is the occupancy probability of component m at time t, calculated using the acoustic model $\mathcal{M}$ and $\mathbf{W}$; K is a constant term independent of $\hat{\mathbf{W}}$. By rearranging terms relevant with $\hat{\mathbf{W}}$, Eq. (3.10) can be written in the following form:

$$\mathcal{Q}(\hat{\mathbf{W}}; \mathbf{W}, \mathcal{M}) = -\frac{1}{2}\mathrm{tr}\left(\sum_{t,m} \gamma_t^{(m)} \hat{\mathbf{W}}^{\mathsf{T}} \boldsymbol{\Sigma}^{(m)-1} \hat{\mathbf{W}} \boldsymbol{\xi}^{(m)} \boldsymbol{\xi}^{(m)\mathsf{T}}\right) + \mathrm{tr}\left(\hat{\mathbf{W}} \sum_{t,m} \boldsymbol{\xi}^{(m)} \boldsymbol{o}_t^{\mathsf{T}} \boldsymbol{\Sigma}^{(m)-1}\right) \tag{3.11}$$

Differentiating the above auxiliary function with respect to $\hat{\mathbf{W}}$ and equating to zero yields:

$$\sum_{t,m} \gamma_t^{(m)} \boldsymbol{\Sigma}^{(m)-1} \hat{\mathbf{W}} \boldsymbol{\xi}^{(m)} \boldsymbol{\xi}^{(m)\mathsf{T}} = \sum_{t,m} \gamma_t^{(m)} \boldsymbol{\Sigma}^{(m)-1} \boldsymbol{o}_t \boldsymbol{\xi}^{(m)\mathsf{T}} \tag{3.12}$$

For a non-diagonal covariance matrix, the linear transform which maximises the above auxiliary function is given by [67]:

$$\mathrm{vec}(\hat{\mathbf{W}}) = \mathbf{G}^{-1} \cdot \mathrm{vec}(\mathbf{K}) \tag{3.13}$$

where $\mathrm{vec}(\mathbf{W})$ denotes the vectorisation of the matrix $\mathbf{W}$ formed by stacking the columns vector into a single column vector, and

$$\mathbf{G} = \sum_{t,m} \gamma_t^{(m)} \left((\boldsymbol{\xi}^{(m)} \boldsymbol{\xi}^{(m)\mathsf{T}}) \otimes \boldsymbol{\Sigma}^{(m)-1}\right) \tag{3.14}$$

$$\mathbf{K} = \sum_{t,m} \gamma_t^{(m)} \boldsymbol{\Sigma}^{(m)-1} \boldsymbol{o}_t \, \boldsymbol{\xi}^{(m)\mathsf{T}} \tag{3.15}$$

$\otimes$ is the Kronecker product. Solving this linear equation requires inverting a $D(D+1)$-by-$D(D+1)$ matrix where D is the dimension of feature vectors. This is computationally expensive. Instead, a row-by-row method can be used to iteratively optimise each row vector of $\hat{\mathbf{W}}$ in an iterative manner [225].

In standard GMM-HMM models, the Gaussian covariance matrices $\mathbf{\Sigma}^{(m)}$ are usually assumed to be diagonal, i.e., $\mathbf{\Sigma}^{(m)} = \text{diag}(\cdots, \sigma_d^{(m)2}, \cdots)$. This diagonal covariance matrix structure will largely simplify the estimation. Let $\hat{\mathbf{w}}_d^{\mathsf{T}}$ be the d-th row vector of $\hat{\mathbf{W}}$. The ML estimate of $\hat{\mathbf{w}}_d^{\mathsf{T}}$ is given by:

$$\hat{\mathbf{w}}_d^{\mathsf{T}} = \mathbf{k}_d^{\mathsf{T}} \mathbf{G}_d^{-1} \tag{3.16}$$

where

$$\mathbf{G}_d = \sum_{t,m} \frac{\gamma_t^{(m)}}{\sigma_d^{(m)2}} \boldsymbol{\xi}^{(m)} \boldsymbol{\xi}^{(m)\mathsf{T}} \tag{3.17}$$

$$\mathbf{k}_d = \sum_{t,m} \frac{\gamma_t^{(m)} o_{t,d}}{\sigma_d^{(m)2}} \boldsymbol{\xi}^{(m)\mathsf{T}} \tag{3.18}$$

$\sigma_d^{(m)2}$ is the d-th diagonal element of $\mathbf{\Sigma}^{(m)}$ and $o_{t,d}$ is the d-th element of $\boldsymbol{o}_t$.

3.2.2.2 Variance MLLR

To further improve performance, the covariance matrices of Gaussian components can be also adapted using linear transforms. This will be referred to as *variance MLLR*. One form of variance MLLR is given in [74], where the covariance matrix $\mathbf{\Sigma}^{(m)}$ is adapted by:

$$\hat{\mathbf{\Sigma}}^{(m)} = \mathbf{L}^{(m)\mathsf{T}} \mathbf{H} \mathbf{L}^{(m)} \tag{3.19}$$

Here, $\mathbf{L}^{(m)}$ is obtained by Cholesky factorisation of $\mathbf{\Sigma}^{(m)}$, i.e.,

$$\mathbf{L}^{(m)} \mathbf{L}^{(m)\mathsf{T}} = \mathbf{\Sigma}^{(m)} \tag{3.20}$$

and $\mathbf{H}$ is the linear transform used to adapt the covariance matrices. An EM algorithm is developed to estimate this linear transform [74]. One limitation of this adaptation form is that the adapted covariance matrix is a full matrix, which increases the computational complexity of likelihood calculation after adaptation. An alternative form is proposed in [67] to solve this problem. The covariance matrix is adapted using the following form in [67]:

$$\hat{\mathbf{\Sigma}}^{(m)} = \mathbf{H} \mathbf{\Sigma}^{(m)} \mathbf{H}^{\mathsf{T}} \tag{3.21}$$

where $\mathbf{H}$ is the linear transform. The advantage of this form is that the Gaussian component likelihood can be calculated in a very efficient way as

$$\log p(o_t|m) = \log \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}^{(m)}, \hat{\boldsymbol{\Sigma}}^{(m)}) = \log \mathcal{N}(\mathbf{H}^{-1}\boldsymbol{o}_t; \mathbf{H}^{-1}\boldsymbol{\mu}^{(m)}, \boldsymbol{\Sigma}^{(m)}) - \frac{1}{2}\log(|\mathbf{H}|^2) \quad (3.22)$$

where $|\mathbf{H}|$ denotes the absolute value of determinant of $\mathbf{H}$. This adaptation form only involves transforming the mean and observation vector and the likelihood calculation can de performed using the original diagonal covariance matrix. An EM algorithm is also developed in [67] to ML estimate $\mathbf{H}$.

3.2.2.3 Constrained MLLR (CMLLR)

The above mean and covariance adaptation can be combined, yielding the following form for likelihood calculation for the m-th Gaussian component:

$$\begin{aligned} \log p(\boldsymbol{o}_t|m) &= \log \mathcal{N}(\boldsymbol{o}_t; \tilde{\mathbf{A}}\boldsymbol{\mu}^{(m)} + \tilde{\boldsymbol{b}}, \mathbf{H}\boldsymbol{\Sigma}^{(m)}\mathbf{H}^{\mathsf{T}}) \\ &= \log \mathcal{N}(\mathbf{H}^{-1}\boldsymbol{o}_t - \mathbf{H}^{-1}\tilde{\boldsymbol{b}}; \mathbf{H}^{-1}\tilde{\mathbf{A}}\boldsymbol{\mu}^{(m)}, \boldsymbol{\Sigma}^{(m)}) - \frac{1}{2}\log(|\mathbf{H}|^2) \end{aligned} \quad (3.23)$$

where $[\tilde{\mathbf{A}}, \tilde{\boldsymbol{b}}]$ is the mean MLLR transform, and $\mathbf{H}$ is the variance MLLR transform. Constrained MLLR (CMLLR)[50, 67] is a special form of the above mean and variance MLLR adaptation, where the same linear transform is used for mean and variance transform, i.e., $\tilde{\mathbf{A}} = \mathbf{H}$. This results in a computationally more efficient form of likelihood calculation:

$$\log p(\boldsymbol{o}_t|m) = \log \mathcal{N}(\hat{\boldsymbol{o}}_t; \boldsymbol{\mu}^{(m)}, \boldsymbol{\Sigma}^{(m)}) + \frac{1}{2}\log(|\mathbf{A}|^2) \quad (3.24)$$

$$\hat{\boldsymbol{o}}_t = \mathbf{A}\boldsymbol{o}_t + \boldsymbol{b} = \mathbf{W}\boldsymbol{\zeta}_t \quad (3.25)$$

where $\mathbf{A} = \tilde{\mathbf{A}}^{-1}$ and $\boldsymbol{b} = -\mathbf{H}^{-1}\tilde{\boldsymbol{b}}$, $\boldsymbol{\zeta}_t = [\boldsymbol{o}_t^{\mathsf{T}}, 1]^{\mathsf{T}}$ and $\mathbf{W} = [\mathbf{A}, \boldsymbol{b}]$ is a linear transform of the feature vectors. It is interesting to note that this constrained MLLR adaptation of mean and covariance can be implemented as a transformation acting on the feature space while the model parameters are not altered. This saves the computation to modify the model parameters during runtime. It is worth pointing out that when multiple base classes are used, CMLLR is still a model-based approach, as features are separately transformed for Gaussian components in different base classes.

The EM algorithm can be also used to estimate the CMLLR transform. The estimation algorithm, presented in [67], is briefly described. In the EM algorithm, an auxiliary function for CMLLR transform is defined as:

$$\mathcal{Q}(\hat{\mathbf{W}}; \mathbf{W}, \mathcal{M}) = -\frac{1}{2}\sum_{t,m} \gamma_t^{(m)} \left\{ (\mathbf{W}\boldsymbol{\zeta}_t - \boldsymbol{\mu}^{(m)})^{\mathsf{T}} \boldsymbol{\Sigma}^{(m)-1} (\mathbf{W}\boldsymbol{\zeta}_t - \boldsymbol{\mu}^{(m)}) + \log|\mathbf{W}| \right\} \quad (3.26)$$

Given the sufficient statistics

$$\mathbf{G}_d = \sum_m \frac{1}{\sigma_d^{(m)2}} \sum_t \gamma_t^{(m)} \boldsymbol{\zeta}_t \boldsymbol{\zeta}_t^{\mathsf{T}} \tag{3.27}$$

$$\mathbf{k}_d = \sum_m \frac{\mu_d^{(m)}}{\sigma_d^{(m)2}} \sum_t \gamma_t^{(m)} \boldsymbol{\zeta}_t \tag{3.28}$$

where $\sigma_d^{(m)2}$ and $\mu_d^{(m)}$ are the d-th element of $\boldsymbol{\Sigma}^{(m)}$ and $\boldsymbol{\mu}^{(m)}$ respectively, the d-th row vector of $\hat{\mathbf{W}}$, $\hat{\mathbf{w}}_d^{\mathsf{T}}$, can be solved by

$$\hat{\mathbf{w}}_d^{\mathsf{T}} = (\alpha \mathbf{p}_d + \mathbf{k}_d)^{\mathsf{T}} \mathbf{G}_d^{-1} \tag{3.29}$$

where $\mathbf{p}_d = [c_{d,1}, \ldots, c_{d,D}, 0]$ is extended cofactor vector with $c_{i,j}$ being the (i,j)-th cofactor of $\mathbf{A}$; the coefficient α is given in a quadratic equation:

$$\alpha^2 \mathbf{p}_d^{\mathsf{T}} \mathbf{G}^{-1} \mathbf{p}_d + \alpha \mathbf{p}_d^{\mathsf{T}} \mathbf{G}^{-1} \mathbf{k}_d - \beta = 0 \tag{3.30}$$

and $\beta = \sum_{t,m} \gamma_t^{(m)}$. Apparently, the estimated d-th row vector depends on all the other row vectors through the extended cofactor vector. Therefore, the estimate is an iterative procedure where $\hat{\mathbf{W}}$ is updated in a row-by-row manner.

3.2.3 Cluster-based Speaker Adaptation

The adaptation schemes discussed so far are based on a single set of acoustic models. An alternative approach to adaptation is to use a number of HMM sets, each representing different acoustic conditions. A simple example is gender dependent acoustic models (GD). Traditional cluster-based approaches [60, 133] have been developed along this line, where a set of acoustic models, usually referred to as *reference models*, are built. Adaptation is performed by choosing the most representative cluster. The limitation of this approach is that the cluster selection is a "hard" choice, and only a finite number of acoustic models can be derived after adaptation. As an extension to this approach, schemes that use linear combination of a set of cluster-dependent models have been developed, e.g., [91, 92]. In these schemes, a "soft" choice is made where the adapted acoustic models are not necessarily one set of the reference models but a new, interpolated, set of acoustic models. By interpolating the reference models, it is possible to generate an infinite number of acoustic models. Transform smoothing [64], cluster adaptation[70] and eigenvoices [136] are all developed based on this idea. The cluster-dependent model parameters are normally adaptively trained, thus it is related to the adaptive training framework. This will be described in detail in section 3.4.1.2.

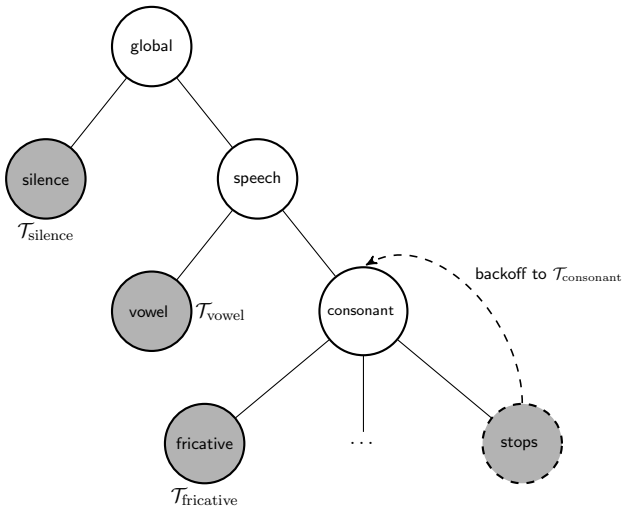


Figure 3.3: Illustration of regression tree and transform parameter tying.

3.2.4 Regression Classes and Transform Parameter Tying

When discussing the linear transform-based and the cluster based adaptation schemes in the previous sections, it was assumed that all Gaussian components are tied to a *global* transform. This may be necessary for robust estimation of transform parameters when only a small amount of adaptation data is available. As more adaptation data becomes available, it is possible to untie the global transform to further improve the adaptation performance. Usually, Gaussian components can be clustered into several groups and each group is associated with a separate transform. These groups are referred to as *base classes*.

One approach to group the Gaussian components into base classes is based on phonetic knowledge [147]. Based on their associated states, Gaussian components can be grouped into the broad phone classes, such as silence, vowels, stops, glides, nasals, fricatives, etc. Alternatively, Gaussian components can be automatically clustered [63] This can be done via k-means clustering[162, 222] using the Kullback-Leibler divergence [222, 249] or a centroid-splitting algorithm with the Euclidean distance measure [63, 260].

Base classes of Gaussian components are usually organised in a regression tree [63, 148] structure, which is used to dynamically assign estimated transforms to base classes, based on the amount of available data. This is illustrated in Figure 3.3. The dark node denotes the base classes in the regression tree. When there is sufficient data for a base class node, a transform will be generated for that node. This is determined by comparing the occupancy counts in the node with a pre-defined threshold. In case there is insufficient data associated with a base class node, such as the "stops" node in Figure 3.3, a back-off scheme is used:

all the data associated with its parent node, "consonant", are pooled together to estimate a "consonant" transform, and this transform will be used for the "stops" node. This back-off scheme can be done in a recursive manner, i.e., when the occupancy count of the parent node is also less than the threshold, the transform estimation is further backed off to the upper level until either the root node or a node with sufficient occupancy count is reached.

When a regression class tree is used, the transform estimation procedure in section 3.2.2 needs to be slightly modified. As each base class of Gaussian components is transformed separately, the sufficient statistics are collected at the base class level. Given the sufficient statistics, one transform is estimated for each base class, provided that there is a sufficient occupancy count associated with that base class. The transform re-estimation formula are unchanged given that the appropriate sufficient statistics are available.

3.3 Environmental Robustness

In addition to speaker differences, the impact of acoustic environments, also known as the environment noise, is another major factor which makes modelling speech signals challenging: environments can impact the speech signals in many different ways and the noises from environments are largely unpredictable. When there is is a large mismatch between the training and testing conditions due to environmental differences, recognition performance will be substantially impacted. This is one of the major obstacles to speech recognition technology being used in realistic scenarios. Over the past decades, many research efforts have been made to improve the environmental robustness of speech recognition systems, and performance of state-of-the-art ASR systems in adverse environments have steadily improved. In this section, some of the most widely used techniques are briefly reviewed.

3.3.1 Impact of the Environment

The first step in most environmental robustness techniques is to specify how speech signals are altered by the acoustic environment. Though the relationship between the environment corrupted signals and the original, noise free, signals is often complicated, it is possible to relate them *approximately* using a model of the acoustic environment [4, 89]. Figure 3.4 demonstrates one such model, in which several major types of environment noise are depicted.

First, the production of speech can be influenced by the ambient background noise. This is the *Lombard effect* [119]: as the level of ambient noise increases, speakers tend to hyper-articulate: vowels are emphasised while consonants become distorted. It is reported that

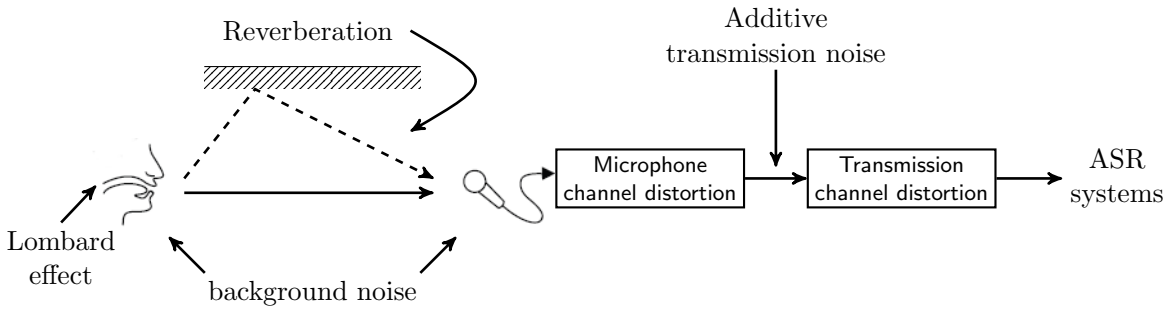


Figure 3.4: Types of environmental noise which can affect speech signals.

recognition performance with such stressed speech can degrade substantially [26, 118]. There have been a few attempts to tackle the Lombard effect as well as stressed, emotional speech, e.g., [25, 227]. However, in this thesis, these effects are not addressed.

Second, as shown in Figure 3.4, a major source of environmental distortions is the ambient noise presented in the background. The background noise can be emitted near the speaker or the microphone. It is considered as additive to the original speech signals in the time domain and can be viewed as statistically independent of the original speech. Additive background noise can be stationary (e.g., noise generated by a ventilation fan), semi-stationary (e.g., noise caused by interfering speakers or music), or abrupt (e.g., noise caused by incoming vehicles). Additive background noise occurs in all speech recognition applications. Therefore, it has been intensively studied in the past several decades [4, 62, 71, 83].

If the speaker uses a distant microphone in an enclosed space, the speech signals are subject to another type of environment distortion – reverberation. As shown in Figure 3.4, reverberation is usually caused by the reflection of speech wavefronts on flat surfaces, e.g., a wall or other objects in a room. These reflections result in several delayed and attenuated copies of the original signals, and these copies are also captured by the recording microphone. In theory, the reverberant speech signals can be described as convolving the speech signals with the room impulse response (RIR), resulting in a multiplicative distortion in the spectral domain. However, as the RIR in a reverberant environment is usually much longer ($\geq$ 200ms) than the length of analysis window (typically 25-30ms) used in the front-end processing, the feature vector extracted from reverberant speech signals are influenced by several previous frames. As a result, reverberant speech signals are highly dynamic. Compared with the additive background noises, which are usually statistically independent of the original clean signals, the reverberant noises are correlated with the original signals and the reverberant distortions are different from the distortions caused by additive background noise. Reverberation has a strong detrimental effect on the recognition performance of an ASR systems: for

instance, without any reverberant robustness technique, the recognition accuracy of a ASR system can easily drop to 60% or even more in a moderate reverberant environment (length of RIR $\sim$ 200ms)[258]. Therefore, reverberant robustness techniques are an essential component in many applications which use distant talking microphones as the receiver. These applications include meeting transcription, hands-free interfaces for controlling consumer-products. In some of these applications, it is possible to use a microphone array to record the speech signals, while in other applications, only a single distant microphone is allowed. Partly driven by commercial interest in deploying speech recognition systems for these applications, research on both microphone array signal processing and reverberant speech processing has achieved significant progress in recent years [18, 252, 258]. Some of these techniques will be reviewed in section 3.3.3, and a new model-based approach to reverberant speech recognition will be proposed and discussed in Chapter 4.

Figure 3.4 also shows that speech signals can be transmitted by a series of transducers, including microphones, and some communication channels. Differences in characteristics of these transducers add another level of distortions, i.e., the channel distortion. Compared with reverberant distortions, the channel distortions are usually caused by linearly filtering the incoming signals with the microphone impulse response, which is shorter than the length of analysis window. As a result, the channel distortions mainly result in spectral tilt. Therefore, the channel distortion is often called short-term convolutional distortion, or convolutional distortion[1] .

3.3.1.1 Mismatch Functions

Given the environment model described in the previous sections, it is possible to relate the corrupted speech signals with the original speech signals in a functional form. This is normally referred to as the *mismatch function* [4, 62]. Depending on the application scenario, there are usually one or two main environment distortions, while other types of distortions can be neglected. Two typical scenarios are considered in the following:

- *Background noise and channel distortion dominate :*

  If the Lombard effect and the additive transmission noise in the environment model depicted in Figure 3.4 are ignored, and the speaker does not talk in an enclosed acoustic space, the background noise and the channel (both the microphone and transmission channel) distortion become the main distortions. Figure 3.5 depicts this simplified environment

[1] Although the nature of reverberation is also convolutional, to keep it consistent with the terminology widely used in the literature, "convolutional distortion" is used in this thesis to stand for the short-term convolutional distortion only.

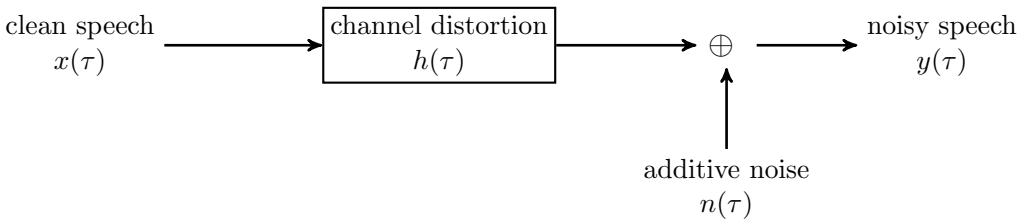


Figure 3.5: A simplified environment model where background noise and channel distortion dominate.

model, where $x(\tau)$ and $y(\tau)$ are the time-domain clean speech signal and noisy speech signal respectively; $h(\tau)$ is the impulse response of the microphone and transmission network, and $n(\tau)$ is the channel filtered version of the ambient background noise. Using this simplified environment model, the noisy speech signal is related to the original clean signal by the following mismatch function in the time domain:

$$y(\tau) = x(\tau) * h(\tau) + n(\tau) \tag{3.31}$$

where $*$ denotes convolution in the time domain. The time-domain mismatch function can be approximated by a mismatch function in the MFCC domain [72, 155]:

$$\boldsymbol{y}_t^{\mathtt{s}} = \mathbf{C} \log \left(\exp(\mathbf{C}^{-1}(\boldsymbol{x}_t^{\mathtt{s}} + \boldsymbol{\mu}_{\mathtt{h}})) + \exp(\mathbf{C}^{-1}\boldsymbol{n}_t^{\mathtt{s}}) + 2\boldsymbol{\alpha} \exp\left(\frac{\mathbf{C}^{-1}(\boldsymbol{x}_t^{\mathtt{s}} + \boldsymbol{\mu}_{\mathtt{h}} + \boldsymbol{n}_t^{\mathtt{s}})}{2} \right) \right) \tag{3.32}$$

where $\boldsymbol{x}_t^{\mathtt{s}}$, $\boldsymbol{y}_t^{\mathtt{s}}$ and $\boldsymbol{n}_t^{\mathtt{s}}$ are the t-th static MFCC feature vector derived from signals $x(\tau)$, $y(\tau)$ and $n(\tau)$ respectively, and $\boldsymbol{\mu}_{\mathtt{h}}$ is an unknown constant derived from the channel impulse response $h(\tau)$; $\boldsymbol{\alpha}$ is a phase factor; $\mathbf{C}$ is the DCT matrix and $\mathbf{C}^{-1}$ is its inverse matrix. Note that most implementations of MFCC feature extraction only use the first several cepstral coefficients. In this case, the $\mathbf{C}$ is the truncated DCT matrix, and $\mathbf{C}^{-1}$ becomes the Moore-Penrose pseudo-inverse matrix of $\mathbf{C}$ [105]. Appendix A.2 gives a detailed derivation of this mismatch function.

The phase factor $\boldsymbol{\alpha}$ is related to the phase differences between the speech and noise signal at each frequency bin. In theory, it is a random variable with all its elements are constrained in the range $[-1, 1]$ [47]. Further approximations are often made to ignore the phase differences, resulting an phase-insensitive mismatch function:

$$\boldsymbol{y}_t^{\mathtt{s}} = \mathbf{C} \log \left(\exp(\mathbf{C}^{-1}(\boldsymbol{x}_t^{\mathtt{s}} + \boldsymbol{\mu}_{\mathtt{h}})) + \exp(\mathbf{C}^{-1}\boldsymbol{n}_t^{\mathtt{s}}) \right) \tag{3.33}$$

Alternatively, in [155] the phase factor is treated as a fixed variable and it is optimised for specific tasks. The use of a phase-sensitive mismatch function in Eq. (3.32) has also been

explored in [149, 237]. This thesis will focus on the standard, phase-insensitive, mismatch function specified in Eq. (3.33).

The above mismatch function can be used for static parameters. As discussed in Section 2.2, dynamic parameters are also used in speech recognition systems. The dynamic parameters are linear transforms of several neighbouring static parameters. For example, the delta parameters are calculated by a linear regression of a few static parameters shown in the following equation:

$$\Delta \boldsymbol{y}_t = \frac{\sum_{\tau=1}^{w} \tau(\boldsymbol{y}^{\mathsf{s}}_{t+\tau} - \boldsymbol{y}^{\mathsf{s}}_{t-\tau})}{2\sum_{\tau} \tau^2} \tag{3.34}$$

$$= \mathbf{D} \begin{bmatrix} \boldsymbol{y}^{\mathsf{s}}_{t+w} \\ \vdots \\ \boldsymbol{y}^{\mathsf{s}}_{t-w} \end{bmatrix} \tag{3.35}$$

where $\mathbf{D}$ is the projection matrix derived from Eq. (3.34). Along with the static mismatch function in Eq. (3.33), Eq. (3.35) yields an accurate form of the mismatch function for delta parameters. This method has been explored in [38, 233]. The limitation of this form of mismatch function is that it requires extended model statistics to represent the distribution of clean speech vectors within a context window $\overline{\boldsymbol{x}}_t = [\boldsymbol{x}_{t+w}, \cdots, \boldsymbol{x}_{t-w}]^{\mathsf{T}}$. Using this form of mismatch functions is also computationally expensive. Though this form of dynamic mismatch function is not used in this thesis, the method to use extended model statistics to represent feature vectors within a context window is useful and it inspires a new model based approach to robust speech recognition in reverberant environments, which will be discussed in Chapter 4.

The most common form of dynamic parameter mismatch function is the continuous time approximation [86]:

$$\Delta \boldsymbol{y}_t \approx \frac{\partial \boldsymbol{y}^{\mathsf{s}}_t}{\partial t} = \frac{\partial \boldsymbol{y}^{\mathsf{s}}_t}{\partial \boldsymbol{x}^{\mathsf{s}}_t} \frac{\partial \boldsymbol{x}^{\mathsf{s}}_t}{\partial t} \approx \frac{\partial \boldsymbol{y}^{\mathsf{s}}_t}{\partial \boldsymbol{x}^{\mathsf{s}}_t} \Delta \boldsymbol{x}_t \tag{3.36}$$

A similar mismatch function can be derived for the delta-delta coefficients. This form has been widely used in the literature and will be also used in this thesis.

- *Reverberant and background noise dominate:*
  When a distant microphone is used in an enclosed acoustic environment, reverberation and background noise will become the main distortions in speech signals. Depending on the attributes of the source of background noises, they can interact with reverberation in a few ways, thus creating different distortions of the speech signal. The background noises can be emitted from just one source, which will be referred to as a *point source*. On the

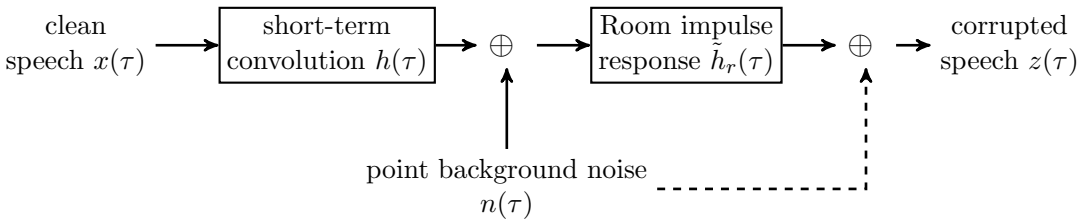


Figure 3.6: Point background noise source in a reverberant environment. The dashed line shows the scenario where the point noise source is sufficiently close to the microphone.

other hand, the sources of background noise could be dispersed over the room space (e.g., restaurant noise). This noise source will be referred to as a *diffuse source*.

For the background point noise source located near the speaker, the noise signals will be convolved with approximately the same RIR as the one which is convolved with the speaker's voice. Let $x(\tau)$, $n(\tau)$ and $h_r(\tau)$ be the clean speech signal, the background noise signal and the RIR measured at the speaker's position. The signal received by the microphone, $z(\tau)$, can be expressed as:

$$z(\tau) = h_r(\tau) * (x(\tau) + n(\tau)) \tag{3.37}$$

Note that because of the $h_r(\tau)$, the effect of noise $n(\tau)$ will spread over several previous frames. This will be referred to as *frame correlated background noise*. On the other hand, if the point noise source is near the microphone, this time-domain mismatch function becomes

$$z(\tau) = h_r(\tau) * x(\tau) + n(\tau) \tag{3.38}$$

In this case, the background noise $n(\tau)$ will only have an impact on the current frame. Therefore, it will be referred to as *frame un-correlated background noise*. For the point noise source positioned somewhere in the middle, $n(\tau)$ will be filtered by another RIR $\tilde{h}_r(\tau)$, and eventually picked up by the receiver. This yields another time domain mismatch function:

$$\begin{aligned} z(\tau) &= h_r(\tau) * x(\tau) + \tilde{h}_r(\tau) * n(\tau) \\ &\approx \tilde{h}_r(\tau) * (h(\tau) * x(\tau) + n(\tau)) \end{aligned} \tag{3.39}$$

Here, it is assumed that a RIR $h_r(\tau)$ can be approximated by filtering another RIR $\tilde{h}_r(\tau)$ with a short-term impulse response $h(\tau)$, i.e., $h_r(\tau) \approx \tilde{h}_r(\tau) * h(\tau)$. Note that the mismatch function in Eq. (3.37) is a special case of Eq. (3.39) with $h(\tau) = \delta(\tau)$; therefore only the mismatch function in Eq. (3.39) will be considered in this thesis. A point background noise source in a reverberant environment is illustrated in Figure 3.6.

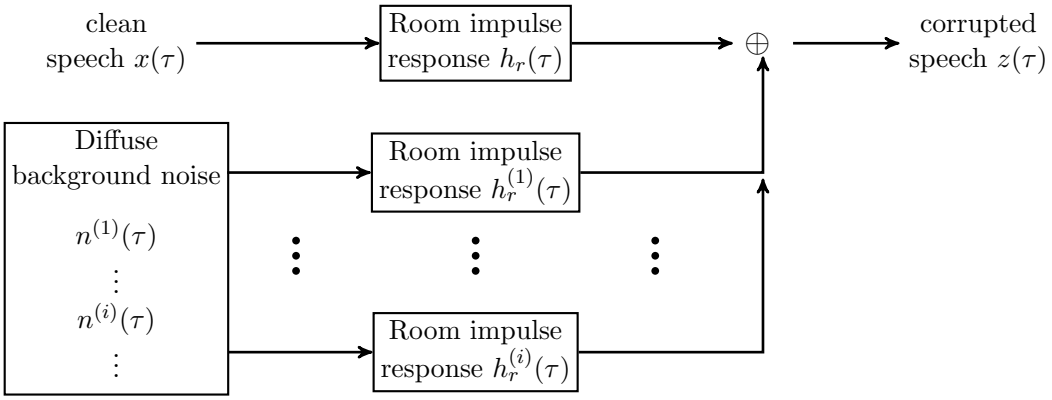


Figure 3.7: Diffuse background noise in a reverberant environment.

The diffuse background noise source can be viewed as a collection of many point background noise sources dispersed over the room space. The time domain mismatch function can be expressed as

$$z(\tau) = h_r(\tau) * x(\tau) + \sum_i n^{(i)}(\tau) * h_r^{(i)}(\tau) \tag{3.40}$$

where $n^{(i)}(\tau)$ is the signal emitted by the i-th source and $h_r^{(i)}(\tau)$ is the RIR measured at the position of i-th noise source. This is illustrated in Figure 3.7. Although there may be some correlation between current and previous frames in each individual RIR filtered noise signal $n^{(i)}(\tau) * h_r^{(i)}(\tau)$ due to the room reverberation, the overall noise signal $\sum_i n^{(i)}(\tau) * h_r^{(i)}(\tau)$ is approximately frame un-correlated. This is based on the assumption that the RIR $h_r^{(i)}(\tau)$ is statistically independent of each other. Hence the signal $\sum_i n^{(i)}(\tau) * h_r^{(i)}(\tau)$ can be approximated by a single point source $n(\tau)$ which is directly received by the microphone. This yields the same time domain mismatch function as in Eq. (3.38).

Given the time domain mismatch functions in Eqs. (3.38 - 3.39), it is also possible to derive the corresponding mismatch functions in the log-spectral or the MFCC domain. Assume the RIR spreads over $n+1$ frames, from the current t-th frame to $t-n$ frame. For the frame-correlated background noise, the corrupted speech static MFCCs $\boldsymbol{z}_t^{\mathsf{s}}$ is related to the clean speech static MFCC $\boldsymbol{x}_t^{\mathsf{s}}$ by:

$$\begin{aligned} \boldsymbol{z}_t^{\mathsf{s}} &= \mathbf{C} \log \left(\sum_{\delta=0}^{n} \exp(\mathbf{C}^{-1}(\boldsymbol{y}_{t-\delta}^{\mathsf{s}} + \tilde{\boldsymbol{\mu}}_{\mathtt{1}\delta})) \right) \\ &= \tilde{\mathbf{g}} \left(\boldsymbol{y}_t^{\mathsf{s}}, \ldots, \boldsymbol{y}_{t-n}^{\mathsf{s}}, \tilde{\boldsymbol{\mu}}_{\mathtt{1}} \right) \end{aligned} \tag{3.41}$$

where $\tilde{\boldsymbol{\mu}}_{\mathtt{1}} = [\tilde{\boldsymbol{\mu}}_{\mathtt{10}}^{\mathsf{T}}, \cdots, \tilde{\boldsymbol{\mu}}_{\mathtt{1n}}^{\mathsf{T}}]$ are the MFCC coefficients of RIR and $\boldsymbol{y}_t^{\mathsf{s}}$ is given in Eq. (3.33). For the frame-uncorrelated background noise case, the following mismatch function can be

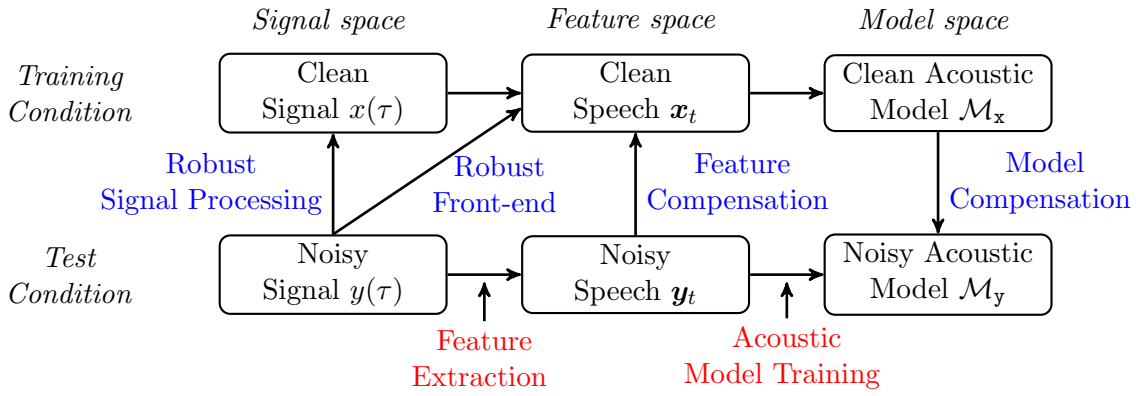


Figure 3.8: Strategies to reduce mismatch between test and training conditions: robust front-end and signal processing, feature-based approach and model-based approach.

derived (detailed in Appendix A.3) :

$$\begin{aligned}\boldsymbol{z}_t^{\mathbf{s}} &= \mathbf{C}\log\left(\sum_{\delta=0}^{n}\exp\left(\mathbf{C}^{-1}(\boldsymbol{x}_{t-\delta}^{\mathbf{s}}+\boldsymbol{\mu}_{1\delta})\right)+\exp\left(\mathbf{C}^{-1}\boldsymbol{n}_t^{\mathbf{s}}\right)\right)\\ &= \mathbf{g}(\boldsymbol{x}_t^{\mathbf{s}},\cdots,\boldsymbol{x}_{t-n}^{\mathbf{s}},\boldsymbol{\mu}_1,\boldsymbol{n}_t^{\mathbf{s}})\end{aligned} \tag{3.42}$$

Similarly, mismatch functions for the dynamic parameters can be derived using the continuous time approximation, i.e.,

$$\Delta\boldsymbol{z}_t \approx \sum_{\delta=0}^{n}\frac{\partial\boldsymbol{z}_t^{\mathbf{s}}}{\partial\boldsymbol{x}_{t-\delta}^{\mathbf{s}}}\frac{\partial\boldsymbol{x}_{t-\delta}^{\mathbf{s}}}{\partial t} = \sum_{\delta=0}^{n}\frac{\partial\boldsymbol{z}_t^{\mathbf{s}}}{\partial\boldsymbol{x}_{t-\delta}^{\mathbf{s}}}\Delta\boldsymbol{x}_{t-\delta} \tag{3.43}$$

3.3.1.2 Strategies to Handle Environment Distortions

The impact of environment distortions on clean speech signals have been described in the previous section. The following sections will review some of the most-widely used environmental robustness techniques. These techniques can be grouped into three broad categories:

- *Inherently robust front-end or robust signal processing*: in which the front-end is designed to extract feature vectors $\boldsymbol{x}_t$ which are insensitive to the differences between clean speech signals $x(\tau)$ and noise corrupted signals $y(\tau)$, or the clean speech signals $x(\tau)$ can be re-constructed given corrupted signals $y(\tau)$;
- *Feature compensation*, in which the corrupted feature vector $\boldsymbol{y}_t$ is compensated such that it closely resembles the clean speech vector $\boldsymbol{x}_t$. This will be referred to as the *feature-based approach*;

- *Model compensation*, in which the clean-trained acoustic model $\mathcal{M}_{\mathbf{x}}$ is adapted to $\mathcal{M}_{\mathbf{y}}$ such that it better matches the environment corrupted feature vectors in the target condition. This will be referred to as the *model-based approach*.

Figure 3.8 illustrates the relationship between these approaches: they can be viewed as operating in different spaces[207]. The inherently robust font-end can be viewed as an approach to reducing the mismatch in the signal or feature space, as it is used to produce signals or feature vectors that are immune to environment distortions; the feature-based approach operates in the feature space, in which the corrupted feature vectors are mapped to the most likely clean feature vectors; the model-based approach adapts the (possibly clean-trained) acoustic model to the target condition, and works in the model space.

3.3.2 Handling Additive Noise and Convolutional Distortions

This section discusses some widely used techniques to combat distortions caused by the additive background noise and the convolutional noise.

3.3.2.1 Inherently Robust Front-end and Robust Signal Processing

A straightforward approach to combat the environment noise is to extract feature vectors which are robust to noises or distortions. This can be achieved by robust signal processing such that the clean speech signals can be reconstructed, or by designing an inherently robust front-end such that a clean speech vector can be directly extracted from the noise corrupted signals.

Spectral subtraction (SS) [24] is a robust signal processing scheme. In this scheme, an estimate of the noise power spectrum $\hat{n}(t,b)$ is first obtained using frames that are classified as non-speech frames, e.g., using voice activity detection (VAD) [101]. Here $\hat{n}(t,b)$ is an estimate of the energy in the b-th frequency bin of the noise signal at time t. It is assumed that the noise power spectrum $n(t,b)$ and the clean speech power spectrum $x(t,b)$ are statistically independent, so the power spectrum of noisy speech $y(t,b)$ can be approximated by

$$y(t,b) \approx x(t,b) + n(t,b) \tag{3.44}$$

Further assuming that the noise power spectrum changes slowly, an estimate of clean speech spectrum, $\hat{x}(t,b)$, can be obtained by:

$$\hat{x}(t,b) = y(t,b) - \hat{n}(t,b) \tag{3.45}$$

This removes the noise energy from the power spectrum domain. In practice, the noise spectrum estimate $\hat{n}(t,b)$ is not accurate, which may result in a negative power spectrum

when $\hat{n}(t,b) > y(t,b)$. To address this issue, the clean speech spectrum estimate $\hat{x}(t,b)$ is normally floored to a small value ε, i.e.,

$$\hat{x}(t,b) = \max\{y(t,b) - \hat{n}(t,b), \varepsilon\} \tag{3.46}$$

Another method to combat additive noise distortion is based on a classical noise reduction algorithm, the *Wiener filter*. Assuming both the clean speech signal $x(t)$ and the noise signal $n(t)$ are stationary stochastic processes, an estimate of $x(t)$, $\hat{x}(t)$, which minimises the squared error can be obtained by passing the signal $y(t)$ through the Wiener filter whose frequency response $H(f)$ is determined by

$$H(f) = \frac{x(f)}{x(f) + n(f)} \tag{3.47}$$

where $x(f)$, $n(f)$ and $y(f)$ is the power spectrum of $x(t)$, $n(t)$ and $y(t)$ respectively. The difficulty in using the Wiener filter is that it requires that the noise and speech spectrum are known, and both the noise and clean speech signals are stationary. It is well know that speech signals are quasi-stationary. As a consequence, it is necessary to segment the speech signals into several stationary states to apply the Wiener filter. This is explored in [56, 239], where a front-end HMM is used to align the corrupted speech into quasi-stationary states and the state statistics in the front-end HMM are used to construct the clean speech and the noise spectrum.

Wiener filtering is also used in European Telecommunications Standards Institute (ETSI) advanced front-end (AFE) standard [230]. In AFE, a two stage Wiener filter (e.g., [6]) is applied to the speech signals as a pre-processing step to reduce background noise. The filter estimation is individually done for short segments of signal – the neighbouring two frames, rather than using front-end HMMs to segment signals into qusi-stationary segments as in [56]. After noise reduction using the two stage Wiener filtering, the cepstral coefficients are extracted. These cepstral coefficients are processed by blind equalization to compensate for possible convolutional distortions. The AFE has been shown to be effective for additive noise and convolutional distortions and is regarded as one of the state-of-the-art front-end based robustness techniques.

Besides robust signal processing, attempts have been made to design inherently robust front-ends. For example, RASTA-PLP [96, 97] is a popularly cited robustness technique in this category. Relative spectral (RASTA) processing is applied to PLP coefficients to give RASTA-PLP coefficients. Inspired by the auditory evidence that modulations in the spectrum below 1 Hz and above 12 Hz are usually noise, a band pass filtering is applied in RASTA. The smoothing over speech segments whose length is about 150ms simulates the human ears'

ability to incorporate information over time. This also overcomes the limitation of widely used short-time spectrum analysis-based front-end, such as MFCC and PLP. RASTA-PLP achieves only moderate success against additive noise and convolutional distortions [98].

3.3.2.2 Feature-based Approaches

Feature-based approaches aim to produce an estimate of a clean speech vector $\boldsymbol{x}_t$ given the observed speech vectors [1]. This is usually achieved by using *minimum mean squared error estimation (MMSE)*, i.e., (e.g., [45, 55, 180]):,

$$\begin{aligned}\hat{\boldsymbol{x}}_t &= \arg\min_{\hat{\boldsymbol{x}}} \mathcal{E}\{|\boldsymbol{x}_t - \hat{\boldsymbol{x}}|^2|\boldsymbol{O}\} \\ &= \mathcal{E}\{\boldsymbol{x}_t|\boldsymbol{O}\}\end{aligned} \tag{3.48}$$

where $\hat{\boldsymbol{x}}_t$ is the estimate of clean speech $\boldsymbol{x}$ at time t, and $\boldsymbol{O}$ is the sequence of observed feature vectors. This requires calculating the posterior probability of $\boldsymbol{x}_t$ given $\boldsymbol{O}$, which is often computationally expensive, though it is still feasible (e.g., [113]). It is therefore commonly assumed that additive noise corrupts speech independently for each frame. In this case, the MMSE becomes

$$\hat{\boldsymbol{x}}_t = \mathcal{E}\{\boldsymbol{x}_t|\boldsymbol{y}_t\} \tag{3.49}$$

The estimated clean speech vector, $\hat{\boldsymbol{x}}_t$, is then passed to a back-end acoustic model for likelihood calculation. For the m-th Gaussian component, the likelihood calculation is approximated by

$$p(\boldsymbol{y}_t|m) \approx \mathcal{N}(\hat{\boldsymbol{x}}_t; \boldsymbol{\mu}_{\mathbf{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathbf{x}}^{(m)}) \tag{3.50}$$

where $\boldsymbol{\mu}_{\mathbf{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathbf{x}}^{(m)}$ are the mean vector and covariance matrix in the clean-trained acoustic model for the component m. This process is demonstrated by the solid lines in Figure 3.9. Using the MMSE estimate as the clean speech directly in the likelihood calculation assumes that the feature compensation is done perfectly. However, for low-SNR conditions, where $\boldsymbol{x}_t$ is almost masked by $\boldsymbol{n}_t$, it is hard to accurately estimate the clean speech vector. Instead, the uncertainty in feature enhancement can be propagated to the back-end acoustic model. This approach has been explored in many works in the literature, e.g., [10, 48, 53, 71]. The idea of using uncertainty in decoding is illustrated by the dashed line in Figure 3.9.

An intuitive approach to use uncertainty in decoding is to add the variance of the clean speech estimate, $\boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}_t} = \text{var}(\boldsymbol{x}_t|\boldsymbol{y}_t)$, to the back-end model variance $\boldsymbol{\Sigma}_{\mathbf{x}}^{(m)}$. In [48], it is shown

[1] For convenience, only static parameter compensation is discussed, and the symbol $\mathbf{s}$ in subscripts or superscripts is omitted in this section (section 3.3.2.2). It is possible to compensate dynamic parameters as well, but this is not discussed in this section.

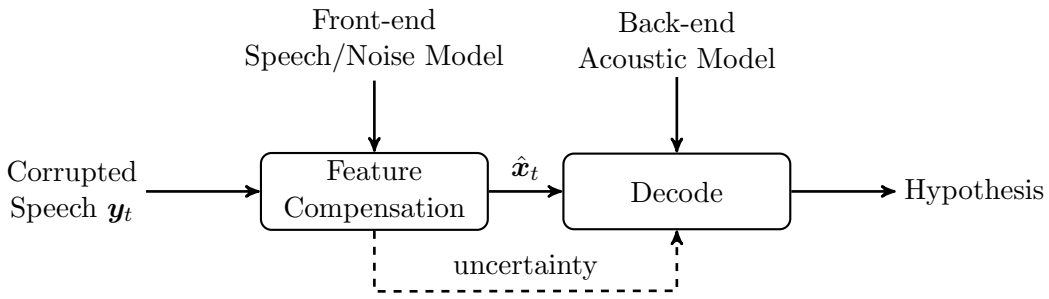


Figure 3.9: Feature compensation with uncertainty decoding for robust speech recognition.

that this amounts to propagating the conditional probability $p(\boldsymbol{x}_t|\boldsymbol{y}_t)$ to the back-end, where the likelihood is calculated by:

$$\begin{aligned} p(\boldsymbol{y}_t|m) &\approx \int p(\boldsymbol{x}_t|\boldsymbol{y}_t)p(\boldsymbol{x}_t|m)\,d\boldsymbol{x}_t \\ &\approx \int \mathcal{N}(\boldsymbol{x};\hat{\boldsymbol{x}}_t,\boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}_t})\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu}_{\mathrm{x}}^{(m)},\boldsymbol{\Sigma}_{\mathrm{x}}^{(m)})\,d\boldsymbol{x} \\ &= \mathcal{N}(\hat{\boldsymbol{x}}_t;\boldsymbol{\mu}_{\mathrm{x}}^{(m)},\boldsymbol{\Sigma}_{\mathrm{x}}^{(m)}+\boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}_t}) \end{aligned} \tag{3.51}$$

and $p(\boldsymbol{x}_t|\boldsymbol{y}_t)$ is approximated by a Gaussian with $\hat{\boldsymbol{x}}_t$ as the mean vector and $\boldsymbol{\Sigma}_{\hat{\boldsymbol{x}}_t}$ as the covariance matrix. This approach is usually referred to as *observation uncertainty*.

Although the observation uncertainty approach is intuitive and has been widely used in the literature (e.g. [10, 20, 46, 48, 80, 253]), it is not mathematically well defined. This can be shown in Eq. (3.51) that the first approximation does not corresponding to a valid probability rule. Another approach to uncertainty decoding is to propagate the conditional distribution $p(\boldsymbol{y}_t|\boldsymbol{x}_t)$ to the back-end acoustic models [53, 158, 159]. This yields a mathematically consistent formulation to compute the likelihood function:

$$p(\boldsymbol{y}_t|m) = \int p(\boldsymbol{y}_t|\boldsymbol{x}_t)p(\boldsymbol{x}_t|m)\,d\boldsymbol{x}_t \tag{3.52}$$

This is usually referred to as *uncertainty decoding*.

A number of feature compensation schemes can be derived based on the relationship between clean speech $\boldsymbol{x}$ and the corrupted speech $\boldsymbol{y}$. Due to its mathematical convenience, the Gaussian mixture model (GMM) is widely used to model the joint distribution of $\boldsymbol{x}$ and $\boldsymbol{y}$:

$$\left[\begin{array}{c}\boldsymbol{y}_t\\ \boldsymbol{x}_t\end{array}\right]\Big|n = \mathcal{N}\left(\left[\begin{array}{c}\boldsymbol{\mu}_{\mathrm{y}}^{(n)}\\ \boldsymbol{\mu}_{\mathrm{x}}^{(n)}\end{array}\right],\left[\begin{array}{cc}\boldsymbol{\Sigma}_{\mathrm{y}}^{(n)} & \boldsymbol{\Sigma}_{\mathrm{yx}}^{(n)}\\ \boldsymbol{\Sigma}_{\mathrm{xy}}^{(n)} & \boldsymbol{\Sigma}_{\mathrm{x}}^{(n)}\end{array}\right]\right) \tag{3.53}$$

where n is the front-end component, $\boldsymbol{\mu}_{\mathrm{y}}^{(n)}$ and $\boldsymbol{\mu}_{\mathrm{x}}^{(n)}$ are the mean vectors of $\boldsymbol{y}_t$ and $\boldsymbol{x}_t$ given the front-end component n, respectively; $\boldsymbol{\Sigma}_{\mathrm{y}}^{(n)}$ and $\boldsymbol{\Sigma}_{\mathrm{y}}^{(n)}$ are the corresponding covariance

matrices; $\mathbf{\Sigma}_{\mathsf{yx}}^{(n)} = \mathbf{\Sigma}_{\mathsf{xy}}^{(n)\mathsf{T}}$ is the correlation matrix. Under this joint distribution, it is easy to show that the conditional distribution of $\boldsymbol{x}_t$ given $\boldsymbol{y}_t$ and n is again Gaussian [23] and

$$p(\boldsymbol{x}_t|\boldsymbol{y}_t, n) = \mathcal{N}(\boldsymbol{\mu}_{\mathsf{x|y}}^{(n)}; \mathbf{\Sigma}_{\mathsf{x|y}}^{(n)}) \tag{3.54}$$

with

$$\boldsymbol{\mu}_{\mathsf{x|y}}^{(n)} = \boldsymbol{\mu}_{\mathsf{x}}^{(n)} + \mathbf{\Sigma}_{\mathsf{xy}}^{(n)}\mathbf{\Sigma}_{\mathsf{y}}^{(n)-1}(\boldsymbol{y}_t - \boldsymbol{\mu}_{\mathsf{y}}^{(n)}) \tag{3.55}$$

$$\mathbf{\Sigma}_{\mathsf{x|y}}^{(n)} = \mathbf{\Sigma}_{\mathsf{x}}^{(n)} - \mathbf{\Sigma}_{\mathsf{xy}}^{(n)}\mathbf{\Sigma}_{\mathsf{y}}^{(n)-1}\mathbf{\Sigma}_{\mathsf{yx}}^{(n)} \tag{3.56}$$

Similar formulae can be derived for the conditional distribution $p(\boldsymbol{y}_t|\boldsymbol{x}_t, n)$. It is then possible to calculate the MMSE estimtor of clean speech by

$$\begin{aligned}\mathcal{E}\{\boldsymbol{x}_t|\boldsymbol{y}_t\} &= \sum_n p(n|\boldsymbol{y}_t)\mathcal{E}\{\boldsymbol{x}_t|\boldsymbol{y}_t, n\} \\ &= \sum_n p(n|\boldsymbol{y}_t)\left(\boldsymbol{\mu}_{\mathsf{x}}^{(n)} + \mathbf{\Sigma}_{\mathsf{xy}}^{(n)}\mathbf{\Sigma}_{\mathsf{y}}^{(n)-1}(\boldsymbol{y} - \boldsymbol{\mu}_{\mathsf{y}}^{(n)})\right)\end{aligned} \tag{3.57}$$

The conditional distribution $p(\boldsymbol{y}_t|\boldsymbol{x}_t, n)$ or $p(\boldsymbol{x}_t|\boldsymbol{y}_t, n)$ can be also used within the framework of uncertainty decoding or observation uncertainty.

Depending on how the joint distribution is modelled, many feature-based schemes have been developed [45, 71, 173, 231] . Stereo-based piecewise linear compensation for environments (SPLICE) [45] and feature space vector Taylor series (VTS)[173, 174] are two examples.

- *SPLICE*

  SPLICE was first proposed in [45]. SPLICE is based on stereo data, where it is assumed that corrupted speech $\boldsymbol{y}$ can be piecewise linearly transformed to clean speech $\boldsymbol{x}$ using a simple bias, though it is possible to use an affine transform as suggested in [45]. In SPLICE, the conditional distribution $p(\boldsymbol{x}|\boldsymbol{y}, n)$ is directly modelled, again as a Gaussian distribution:

$$p(\boldsymbol{x}|\boldsymbol{y}, n) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{y} + \boldsymbol{b}^{(n)}, \mathbf{\Gamma}^{(n)}) \tag{3.58}$$

  The parameter $\boldsymbol{b}^{(n)}$ and $\mathbf{\Gamma}^{(n)}$ can be estimated from stereo data. Given a corrupted observation $\boldsymbol{y}_t$ in testing, its MMSE estimate is thus given by

$$\hat{\boldsymbol{x}}_t = \sum_n p(n|\boldsymbol{y}_t)(\boldsymbol{y}_t + \boldsymbol{b}^{(n)}) \tag{3.59}$$

  Alternatively, the front-end component $n* = \arg\max_n p(n|\boldsymbol{y}_t)$ with the highest posterior can be selected and a hard decision is made to yield an efficient implementation of SPLICE:

$$\hat{\boldsymbol{x}}_t = \boldsymbol{y}_t + \boldsymbol{b}^{(n*)} \tag{3.60}$$

- *Feature-space VTS*

  In SPLICE, the relationship between clean speech and corrupted speech is learned from stereo data. As a consequence, in the unseen noise condition, the performance gains obtained by SPLICE are often limited. Instead of learning the relationship between clean and corrupted data from the stereo data, the mismatch function in Eq. (3.33) can be used. Because of the nonlinearity of the mismatch function, it is not possible to derive the posterior probability $p(\boldsymbol{x}|\boldsymbol{y})$ analytically. The vector Taylor series (VTS) technique, first proposed in [174], approximates the mismatch function using a vector Taylor series expansion.

  In feature space VTS, it is assumed that clean speech $\boldsymbol{x}_t$ can be modelled by a GMM, and the noise vector follows a Gaussian distribution and $\boldsymbol{\mu}_{\mathtt{h}}$ is an unknown constant:

$$\boldsymbol{x}_t \sim \sum_n p(n)\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_{\mathtt{x}}^{(n)}, \boldsymbol{\Sigma}_{\mathtt{x}}^{(n)}) \tag{3.61}$$

$$\boldsymbol{n}_t \sim \mathcal{N}(\boldsymbol{n}; \boldsymbol{\mu}_{\mathtt{n}}, \boldsymbol{\Sigma}_{\mathtt{n}}) \tag{3.62}$$

  where n can be viewed as the front-end component in Eq. (3.53) [1] . VTS uses vector Taylor series to approximate the mismatch function in Eq. (3.33). [2] The expansion is performed around the point $(\boldsymbol{\mu}_{\mathtt{x}}^{(n)}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\mu}_{\mathtt{n}})$, which yields

$$\begin{aligned}\boldsymbol{y}_t|\boldsymbol{x}_t, n &= \mathbf{C}\log\left(\exp(\mathbf{C}^{-1}(\boldsymbol{x}_t + \boldsymbol{\mu}_{\mathtt{h}})) + \exp(\mathbf{C}^{-1}\boldsymbol{n}_t)\right) = \mathbf{f}(\boldsymbol{x}_t, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{n}_t) \\ &\approx \mathbf{f}(\boldsymbol{\mu}_{\mathtt{x}}^{(n)}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\mu}_{\mathtt{n}}) + \mathbf{J}_{\mathtt{x}}^{(n)}(\boldsymbol{x}_t - \boldsymbol{\mu}_{\mathtt{x}}^{(n)}) + \mathbf{J}_{\mathtt{n}}^{(n)}(\boldsymbol{n}_t - \boldsymbol{\mu}_{\mathtt{n}})\end{aligned} \tag{3.63}$$

  where

$$\mathbf{J}_{\mathtt{x}}^{(n)} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{x}_t}\Big|_{(\boldsymbol{\mu}_{\mathtt{x}}^{(n)}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\mu}_{\mathtt{n}})}, \qquad \mathbf{J}_{\mathtt{n}}^{(n)} = \mathbf{I} - \mathbf{J}_{\mathtt{x}}^{(n)} \tag{3.64}$$

  Due to the VTS approximation, $\boldsymbol{y}_t$ is now a linear function of $\boldsymbol{x}_t$. It is easy to derive the corresponding parameters:

$$\boldsymbol{\mu}_{\mathtt{y}}^{(n)} = \mathcal{E}\{\boldsymbol{y}_t|n\} = \mathbf{f}(\boldsymbol{x}_t, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{n}_t) \tag{3.65}$$

  and

$$\begin{aligned}\boldsymbol{\Sigma}_{\mathtt{y}}^{(n)} &= \mathcal{E}\{\boldsymbol{y}_t\boldsymbol{y}_t^{\mathsf{T}}\} - \boldsymbol{\mu}_{\mathtt{y}}^{(n)}\boldsymbol{\mu}_{\mathtt{y}}^{(n)\mathsf{T}} \\ &= \mathbf{J}_{\mathtt{x}}^{(n)}\boldsymbol{\Sigma}_{\mathtt{x}}^{(n)}\mathbf{J}_{\mathtt{x}}^{(n)\mathsf{T}} + \mathbf{J}_{\mathtt{n}}^{(n)}\boldsymbol{\Sigma}_{\mathtt{n}}\mathbf{J}_{\mathtt{n}}^{(n)\mathsf{T}}\end{aligned} \tag{3.66}$$

$$\begin{aligned}\boldsymbol{\Sigma}_{\mathtt{xy}}^{(n)} &= \mathcal{E}\{\boldsymbol{x}_t\boldsymbol{y}_t^{\mathsf{T}}|n\} - \boldsymbol{\mu}_{\mathtt{x}}^{(n)}\boldsymbol{\mu}_{\mathtt{y}}^{(n)\mathsf{T}} \\ &= \boldsymbol{\Sigma}_{\mathtt{x}}^{(n)}\mathbf{J}_{\mathtt{x}}^{(n)\mathsf{T}}\end{aligned} \tag{3.67}$$

[1] Note that $\mathtt{n}$ in teletype font denotes "noise" and it is distinguished from the front-end component n using different font families.

[2] In the initial VTS paper [174], a mismatch function in the log-spectral domain was used. This is extended to the mismatch function in the cepstral coefficient domain in [129].

Correspondingly the conditional distributions $p(\boldsymbol{x}_t|\boldsymbol{y}_t, n) = \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_{\mathtt{x}|\mathtt{y}}^{(n)}, \boldsymbol{\Sigma}_{\mathtt{x}|\mathtt{y}}^{(n)})$ with :

$$\boldsymbol{\mu}_{\mathtt{x}|\mathtt{y}}^{(n)} = \boldsymbol{\mu}_{\mathtt{x}} + \boldsymbol{\Sigma}_{\mathtt{x}}^{(n)}\mathbf{J}_{\mathtt{x}}^{(n)\mathsf{T}}\boldsymbol{\Sigma}_{\mathtt{y}}^{(n)-1}(\boldsymbol{y}_t - \boldsymbol{\mu}_{\mathtt{y}}^{(n)}) \tag{3.68}$$

$$\boldsymbol{\Sigma}_{\mathtt{x}|\mathtt{y}}^{(n)} = \boldsymbol{\Sigma}_{\mathtt{x}}^{(n)} - \boldsymbol{\Sigma}_{\mathtt{x}}^{(n)}\mathbf{J}_{\mathtt{x}}^{(n)\mathsf{T}}\boldsymbol{\Sigma}_{\mathtt{y}}^{(n)-1}\mathbf{J}_{\mathtt{x}}^{(n)}\boldsymbol{\Sigma}_{\mathtt{x}}^{(n)} \tag{3.69}$$

The MMSE estimation of clean speech $\boldsymbol{x}_t$ is thus given by:

$$\hat{\boldsymbol{x}}_t = \sum_n p(n|\boldsymbol{y}_t)\boldsymbol{\mu}_{\mathtt{x}|\mathtt{y}}^{(n)} \tag{3.70}$$

The above formula assumes that the noise model parameters $(\boldsymbol{\mu}_{\mathtt{n}}, \boldsymbol{\Sigma}_{\mathtt{n}})$ are known. They can be estimated using the first and last few (usually 20) speech frames of each utterance which are assumed to contain only noise. A better way is to employ a EM algorithm to ML estimate the noise parameters as suggested in [5, 174], which is first explored in [129]. As the noise estimation algorithm for feature space VTS is very similar to the one for model space VTS, this algorithm will be discussed in detail in section 3.3.2.3.

A fundamental issue in the feature based approach to handle uncertainty is pointed out in [160]. At very low SNR, the uncertainty associated with the enhanced feature vectors $p(\boldsymbol{y}_t|\boldsymbol{x})$ is very large, which effectively assumes that $p(\boldsymbol{y}_t|\boldsymbol{x}) \approx p(\boldsymbol{n}_t)$. This means that every back-end recognition component m will give the same likelihood score, as $p(\boldsymbol{y}_t|m) = \int p(\boldsymbol{y}_t|\boldsymbol{x})p(\boldsymbol{x}|m)\,d\boldsymbol{x} \approx p(\boldsymbol{n}_t)$. Therefore, the frame $\boldsymbol{y}_t$ is effectively ignored in terms of acoustic discrimination. This issue is raised due to that feature enhancement is decoupled from the backend acoustic models. As long as the feature enhancement is coupled with the backend acoustic models, e.g., acoustic features can be compensated (or enhanced) differently for Gaussian components in different sets, the scheme can be viewed as an instance of model-based approach, which will be presented in the next section.

3.3.2.3 Model-based Approaches

Rather than compensating feature vectors for the environmental distortions as in the feature-based approaches, model-based approaches adapt the back-end acoustic model to the target condition. The simplest method within model-based approaches is to re-train the acoustic model using data from the same test environment. This is normally referred to as *matched* training. Single-pass re-training [62] can be used to speed up the training process. However, large vocabulary ASR systems are usually trained on hundreds or thousands of hours of data. It is not practical to collect a large amount of noisy data for a particular environment. Instead, *multi-style training* or *multi-condition training* [45, 164] has been proposed, in which a variety of noise samples are used to corrupt the clean data, or data from a number of environment

conditions are pooled together in training. Acoustic models trained in this way are expected to model not only the intrinsic variabilities but also the environment variabilities. This has been shown to improve the noise robustness, e.g.,[190]. Note that the concept of multi-style training is not limited to the environment acoustic factors; it can be applied to various acoustic factors. It is a simple and widely used technique to improve the model robustness.

As the environment usually varies from utterance to utterance, transforming the acoustic models to the target condition, rather than re-training the whole model is attractive. Here, a model transform $\mathcal{T}_{\mathtt{n}}$ for a particular noise condition is used to adapt the clean-trained acoustic $\mathcal{M}$ to $\hat{\mathcal{M}}$ to better reflect the target condition:

$$\hat{\mathcal{M}} = \mathcal{F}(\mathcal{M}, \mathcal{T}_{\mathtt{n}}) \tag{3.71}$$

where $\mathcal{F}$ is a mapping function which transforms the clean acoustic model to the adapted model.

It has been demonstrated in [62] that even if the clean speech $\boldsymbol{x}_t$ and the noise $\boldsymbol{n}_t$ is jointly Gaussian distributed, the resultant corrupted speech $\boldsymbol{y}_t$, generated according to the mismatch function in Eq. (3.33), is non-Gaussian. When the clean speech $\boldsymbol{x}_t$ is considered to be emitted from a state, which is usually modelled by a GMM, the distribution of resultant $\boldsymbol{y}_t$ is complicated with multiple modes. It is possible to model this distribution by GMMs (e.g., [62]) or using non-parametric schemes (e.g., [237]). However, these methods are computationally expensive. Approximation is needed to reduce the computational cost. The most widely used approximation is to assume the resultant corrupted speech $\boldsymbol{y}_t$ is Gaussian nevertheless, provided the clean speech and the noise is joint Gaussian. It is also assumed that the noise does not alter the hidden states and Gaussian components' alignment. Therefore, for a particular Gaussian component m in $\mathcal{M}$, $p(\boldsymbol{x}_t|m)$, it is adapted to another Gaussian distribution, $p(\boldsymbol{y}_t|m)$, where

$$\begin{aligned} p(\boldsymbol{x}_t|m) &= \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_{\mathtt{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{x}}^{(m)}) \,, \\ p(\boldsymbol{y}_t|m) &\approx \mathcal{N}(\boldsymbol{y}_t; \boldsymbol{\mu}_{\mathtt{y}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{y}}^{(m)}) \,, \end{aligned}$$

and $(\boldsymbol{\mu}_{\mathtt{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{x}}^{(m)})$ are the parameters of the clean acoustic model $\mathcal{M}$, $(\boldsymbol{\mu}_{\mathtt{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{x}}^{(m)})$ the parameters of the compensated acoustic model $\hat{\mathcal{M}}$. The parameters of the compensated acoustic model can be obtained by ML-estimation:

$$\boldsymbol{\mu}_{\mathtt{y}}^{(m)} = \mathcal{E}\{\boldsymbol{y}_t|m\} \,, \quad \boldsymbol{\Sigma}_{\mathtt{y}}^{(m)} = \mathcal{E}\{\boldsymbol{y}_t\boldsymbol{y}_t^{\mathsf{T}}|m\} - \boldsymbol{\mu}_{\mathtt{y}}^{(m)}\boldsymbol{\mu}_{\mathtt{y}}^{(m)\mathsf{T}} \tag{3.72}$$

This is the *Parallel Model Combination*(PMC) framework [62]. There are a number of approximation schemes that can be used to derive the compensated parameters. Some approximation

schemes are discussed in the following. Note for all the schemes discussed in the following, the noise model is assumed to be a single Gaussian distribution, i.e.,

$$\boldsymbol{n}_t \sim \mathcal{N}(\boldsymbol{\mu}_{\mathtt{n}}, \boldsymbol{\Sigma}_{\mathtt{n}}) \tag{3.73}$$

It is also assumed that the background noise is a stationary process, therefore the mean vectors of dynamic noise parameters are zeros, i.e., $\boldsymbol{\mu}_{\Delta\mathtt{n}} = 0$ and $\boldsymbol{\mu}_{\Delta^2\mathtt{n}} = 0$. Further, the convolutional noise $\boldsymbol{\mu}_{\mathtt{h}}$ is assumed to be constant but unknown.

- *Log-normal Approximation*

  The log-normal approximation assumes that the sum of log-normal distributions are also log-normal. This allows the clean speech distribution and the noise speech distribution to be combined in the linear power spectrum domain. Assuming that a static noise MFCC vector $\boldsymbol{n}_t^{\mathtt{s}}$ follows a Gaussian distribution $\mathcal{N}(\boldsymbol{n}_t^{\mathtt{s}}; \boldsymbol{\mu}_{\mathtt{sn}}, \boldsymbol{\Sigma}_{\mathtt{sn}})$ in the cepstral domain, the corresponding power spectrum domain feature vector $\boldsymbol{N} = \exp\left(\mathbf{C}^{-1}\boldsymbol{n}^{\mathtt{s}}\right)$ follows a log-normal distribution with the mean $\boldsymbol{\mu}_{\boldsymbol{N}}$ and covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{N}}$ given by [75]:

$$\mu_{\boldsymbol{N},i} = \exp(\mu^{\mathtt{l}}_{\mathtt{sn},i} + \sigma^{\mathtt{l}}_{\mathtt{sn},ii}/2) \tag{3.74}$$

$$\sigma_{\boldsymbol{N},ij} = \mu^{\mathtt{l}}_{\mathtt{sn},i}\mu^{\mathtt{l}}_{\mathtt{sn},j}\left(\exp(\sigma^{\mathtt{l}}_{\mathtt{sn},ij}) - 1\right) \tag{3.75}$$

  Here $\mu_{\boldsymbol{N},i}$ and $\sigma_{\boldsymbol{N},ij}$ are the i-th and (i,j)-th element of $\boldsymbol{\mu}_{\boldsymbol{N}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{N}}$, respectively; $\mu^{\mathtt{l}}_{\mathtt{sn},i}$ and $\sigma^{\mathtt{l}}_{\mathtt{sn},ij}$ are the i-th and (i,j)-th element of the log spectral domain mean $\boldsymbol{\mu}^{\mathtt{l}}_{\mathtt{sn}}$ and covariance matrix $\boldsymbol{\Sigma}^{\mathtt{l}}_{\mathtt{sn}}$, respectively, and

$$\boldsymbol{\mu}^{\mathtt{l}}_{\mathtt{sn}} = \mathbf{C}^{-1}\boldsymbol{\mu}_{\mathtt{sn}} \tag{3.76}$$

$$\boldsymbol{\Sigma}^{\mathtt{l}}_{\mathtt{sn}} = \mathbf{C}^{-1}\boldsymbol{\Sigma}_{\mathtt{sn}}\mathbf{C}^{-\mathsf{T}} \tag{3.77}$$

  Similar formulae are used to derive the mean $\boldsymbol{\mu}_{\boldsymbol{X}}^{(m)}$ and covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{X}}^{(m)}$ of the log-normal distribution of $\boldsymbol{X} = \exp(\mathbf{C}^{-1}\boldsymbol{x}_t^{\mathtt{s}})$ for a Gaussian component m. As $\boldsymbol{Y} = \exp(\mathbf{C}^{-1}\boldsymbol{y}^{\mathtt{s}}) = \boldsymbol{X} + \boldsymbol{N}$ is assumed to be log-normal distributed, its mean and covariance matrix can be obtained by:

$$\boldsymbol{\mu}_{\boldsymbol{Y}}^{(m)} = \boldsymbol{\mu}_{\boldsymbol{X}}^{(m)} + \boldsymbol{\mu}_{\boldsymbol{N}} \tag{3.78}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{Y}}^{(m)} = \boldsymbol{\Sigma}_{\boldsymbol{X}}^{(m)} + \boldsymbol{\Sigma}_{\boldsymbol{N}} \tag{3.79}$$

This combined log-normal distribution can be then converted back to the log-spectral domain by

$$\mu_{\mathtt{y},i}^{\mathtt{l},(m)} = \log \mu_{\boldsymbol{Y},i}^{(m)} - \frac{1}{2} \log \left(1 + \frac{\sigma_{\boldsymbol{Y},ij}^{(m)}}{\mu_{\boldsymbol{Y},i}^{(m)} \mu_{\boldsymbol{Y},j}^{(m)}} \right) \tag{3.80}$$

$$\sigma_{\mathtt{y},ij}^{\mathtt{l},(m)} = \log \left(1 + \frac{\sigma_{\boldsymbol{Y},ij}^{(m)}}{\mu_{\boldsymbol{Y},i}^{(m)} \mu_{\boldsymbol{Y},j}^{(m)}} \right) \tag{3.81}$$

where $\mu_{\mathtt{y},i}^{\mathtt{l},(m)}$ and $\sigma_{\mathtt{y},ij}^{\mathtt{l},(m)}$ are the i-th and (i,j)-th element of the log-spectral domain parameters $\boldsymbol{\mu}_{\mathtt{y}}^{\mathtt{l},(m)}$ and $\boldsymbol{\Sigma}_{\mathtt{y}}^{\mathtt{l},(m)}$ respectively. These parameters can be also converted back to the cepstral domain, yielding $(\boldsymbol{\mu}_{\mathtt{sy}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{sy}}^{(m)})$. The general procedure of using the log-normal approximation in the PMC framework is illustrated in Figure 3.10.

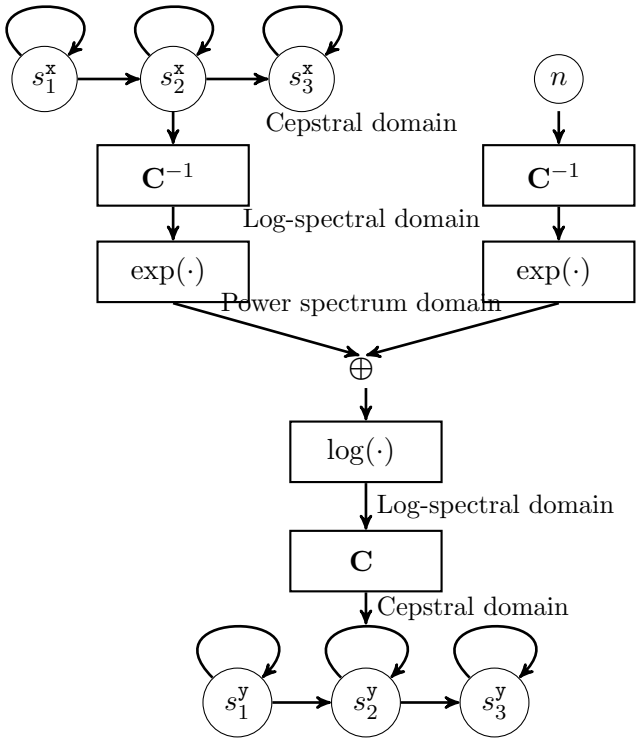


Figure 3.10: Parallel model combination of a three-state clean speech HMM and with a Gaussian noise model using the log-normal approximation.

The Log-normal approximation cannot be used for delta and delta-delta parameters unless the dynamic parameters are computed using simple differences. Data-driven parallel model combination (DPMC)[62] can be used for both static and dynamic parameter compensation. In DPMC Monte Carlo simulation is used to draw random cepstrum vectors from both the clean-speech HMM and noise distribution to create the cepstrum of the noisy speech by

applying the mismatch function. DPMC gave similar results as matched training systems, but it is computational expensive. PMC is also extended in [61] to compensate the effect of convolutional distortion.

- *Model-space VTS*

  The original VTS technique was proposed in [174], where vector Taylor series expansion was applied to compensating the corrupted features, though it was also suggested in [174] that this technique can be used for model compensation. This is first explored in [5, 129]. For a clean speech vector $\boldsymbol{x}_t$ generated from the component m, a first-order Taylor series expansion is used to approximate the mismatch function of the static parameters by

$$\begin{aligned}\boldsymbol{y}_t^{\mathtt{s}}|m &= \mathbf{f}(\boldsymbol{x}_t^{\mathtt{s}}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{n}_t^{\mathtt{s}}) \\ &\approx \mathbf{f}(\boldsymbol{\mu}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\mu}_{\mathtt{sn}}) + \mathbf{J}_{\mathtt{x}}^{(m)}(\boldsymbol{x}_t^{\mathtt{s}} - \boldsymbol{\mu}_{\mathtt{sx}}^{(m)}) + \mathbf{J}_{\mathtt{n}}^{(m)}(\boldsymbol{n}_t^{\mathtt{s}} - \boldsymbol{\mu}_{\mathtt{sn}})\end{aligned} \tag{3.82}$$

  Here $\mathtt{s}$ denotes the static parameter; $\mathbf{f}$ is the mismatch function specified in Eq. (3.33); $\mathbf{J}_{\mathtt{x}}^{(m)}$ and $\mathbf{J}_{\mathtt{n}}^{(m)}$ are the Jacobian matrices evaluated at $(\boldsymbol{\mu}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\mu}_{\mathtt{sn}})$. Based on this approximation, the static parameter is compensated by:

$$\boldsymbol{\mu}_{\mathtt{sy}}^{(m)} = \mathbf{f}(\boldsymbol{\mu}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{sn}}, \boldsymbol{\mu}_{\mathtt{h}}) \tag{3.83}$$

$$\boldsymbol{\Sigma}_{\mathtt{sy}}^{(m)} = \mathrm{diag}\left(\mathbf{J}_{\mathtt{x}}^{(m)}\boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}\mathbf{J}_{\mathtt{x}}^{(m)\mathsf{T}} + \mathbf{J}_{\mathtt{n}}^{(m)}\boldsymbol{\Sigma}_{\mathtt{sn}}\mathbf{J}_{\mathtt{n}}^{(m)\mathsf{T}}\right) \tag{3.84}$$

  Note that the diagonalisation operation is used to maintain an efficient likelihood calculation. The consequence is that the change of feature correlation due to the noise is not modelled in this compensation scheme. The impact of diagonalisation has been discussed in detail in [38]. For the dynamic parameters, the continuous time approximation [86] in Eq. (3.36) is widely used. For example, the delta model parameter is compensated by

$$\boldsymbol{\mu}_{\Delta\mathtt{y}}^{(m)} = \mathbf{J}_{\mathtt{x}}^{(m)}\boldsymbol{\mu}_{\Delta\mathtt{x}}^{(m)} \tag{3.85}$$

$$\boldsymbol{\Sigma}_{\Delta\mathtt{y}}^{(m)} = \mathrm{diag}\left(\mathbf{J}_{\mathtt{x}}^{(m)}\boldsymbol{\Sigma}_{\Delta\mathtt{x}}^{(m)}\mathbf{J}_{\mathtt{x}}^{(m)\mathsf{T}} + \mathbf{J}_{\mathtt{n}}^{(m)}\boldsymbol{\Sigma}_{\Delta\mathtt{n}}\mathbf{J}_{\mathtt{n}}^{(m)\mathsf{T}}\right) \tag{3.86}$$

  Note that the diagonalisation is used again to keep a low computation cost. The similar form can be used for delta-delta parameter compensation as well.

  Because of the linearisation used in VTS, it is possible to devise simple methods for the ML estimation of noise model parameters and clean speech model parameters as well. The ML noise model parameters estimation for the model space VTS is briefly discussed in the following, while the estimation of clean speech model parameters is discussed in section 3.4.2.

As discussed in section 3.3.1.1, the additive noise $\boldsymbol{n}_t$ is commonly assumed to be stationary; therefore the noise model parameters $\boldsymbol{\Phi}$ can be denoted as $\boldsymbol{\Phi} = (\boldsymbol{\mu}_{\mathtt{sn}}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\Sigma}_{\mathtt{n}})$. In the early development (e.g., [5]), the additive noise parameters $(\boldsymbol{\mu}_{\mathtt{sn}}, \boldsymbol{\Sigma}_{\mathtt{n}})$ were estimated from the "silence" portion in each noise homogeneous block. This requires the use of a voice activity detection (VAD) scheme. Furthermore, the convolutional noise $\boldsymbol{\mu}_{\mathtt{h}}$ cannot be estimated. An alternative way is to to optimise $\boldsymbol{\Phi}$ such that the likelihood of test data is maximised. This is discussed in [5] and fully explored in [153, 158]. An EM algorithm is used to maximise the log-likelihood of test data, and the following auxiliary function is used:

$$\mathcal{Q}(\hat{\boldsymbol{\Phi}}; \boldsymbol{\Phi}) = \sum_{m,t} \gamma_t^{(m)} \log p(\boldsymbol{y}_t; \hat{\boldsymbol{\mu}}_{\mathtt{y}}^{(m)}, \hat{\boldsymbol{\Sigma}}_{\mathtt{y}}^{(m)}) \tag{3.87}$$

where $\hat{\boldsymbol{\Phi}}$ is the new set of noise model parameters, $\hat{\boldsymbol{\mu}}_{\mathtt{y}}^{(m)}, \hat{\boldsymbol{\Sigma}}_{\mathtt{y}}^{(m)}$ are the compensated model parameters; $\gamma_t^{(m)}$ is the posterior probability of the observation in component m at time t. This posterior probability is determined by the current noise model parameters $\boldsymbol{\Phi}$ and a supervision hypothesis.

There are two main approaches used in the literature for this estimation problem. The first one is to introduce a second level of EM, where the noise vector $\boldsymbol{n}_t$ is treated as a continuous latent variable (e.g., [108, 129]). This approach will be referred to as the EM-based approach. The advantage of this approach is that it results in mathematically elegant and less complex estimation formula but the limitation is that the Jacobian matrices and the bias term are assumed to be irrelevant of $\boldsymbol{\Phi}$. The second approach is to optimise the auxiliary function in Eq. (3.87) directly using standard gradient descent or second-order schemes (e.g., [122, 155, 158, 266]). This approach is adopted in this thesis and will be presented in the following. To estimate noise mean $\boldsymbol{\mu}_{\mathtt{sn}}$ and convolutional noise $\boldsymbol{\mu}_{\mathtt{h}}$, VTS is used again to approximate the compensated static mean, i.e.,

$$\hat{\boldsymbol{\mu}}_{\mathtt{sy}}^{(m)} \approx \boldsymbol{\mu}_{\mathtt{sy}}^{(m)} + \mathbf{J}_{\mathtt{n}}^{(m)}(\hat{\boldsymbol{\mu}}_{\mathtt{sn}} - \boldsymbol{\mu}_{\mathtt{sn}}) + \mathbf{J}_{\mathtt{h}}^{(m)}(\hat{\boldsymbol{\mu}}_{\mathtt{h}} - \boldsymbol{\mu}_{\mathtt{h}}) \tag{3.88}$$

Here $\mathbf{J}_{\mathtt{h}}^{(m)}$ is the Jacobian matrix of $\boldsymbol{y}_t^{\mathtt{s}}$ with respect to $\boldsymbol{\mu}_{\mathtt{h}}$. Both the Jacobian matrices are evaluated using the current noise model parameter. Using this approximation, it can be shown that the auxiliary is now a quadratic function of $[\hat{\boldsymbol{\mu}}_{\mathtt{sn}}^{\mathsf{T}}, \hat{\boldsymbol{\mu}}_{\mathtt{h}}^{\mathsf{T}}]^{\mathsf{T}}$, therefore yielding the following update formulation:

$$\begin{bmatrix} \hat{\boldsymbol{\mu}}_{\mathtt{sn}} \\ \hat{\boldsymbol{\mu}}_{\mathtt{h}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_{\mathtt{sn}} \\ \boldsymbol{\mu}_{\mathtt{h}} \end{bmatrix} + \left(\sum_{t,m} \gamma_t^{(m)} \begin{bmatrix} \mathbf{J}_{\mathtt{n}}^{(m)\mathsf{T}} \\ \mathbf{J}_{\mathtt{x}}^{(m)\mathsf{T}} \end{bmatrix} \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)-1} [\mathbf{J}_{\mathtt{n}}^{(m)}, \mathbf{J}_{\mathtt{x}}^{(m)}] \right)^{-1} \cdot \left(\sum_{t,m} \gamma_t^{(m)} \begin{bmatrix} \mathbf{J}_{\mathtt{n}}^{(m)\mathsf{T}} \\ \mathbf{J}_{\mathtt{x}}^{(m)\mathsf{T}} \end{bmatrix} (\boldsymbol{y}_t^{\mathtt{s}} - \boldsymbol{\mu}_{\mathtt{sy}}^{(m)}) \right) \tag{3.89}$$

This has also been explained as a variant of the Gaussian-Newton method in [266]. The noise variance estimation is usually done via an iterative second-order method, e.g., in [155]:

$$\hat{\boldsymbol{\Sigma}}_{\mathtt{n}} = \boldsymbol{\Sigma}_{\mathtt{n}} - \zeta \left(\frac{\partial^2 \mathcal{Q}}{\partial^2 \boldsymbol{\Sigma}_{\mathtt{n}}} \right)^{-1} \frac{\partial \mathcal{Q}}{\partial \boldsymbol{\Sigma}_{\mathtt{n}}} \tag{3.90}$$

where ζ, usually initialised as 1, is the line search step size. Note that the Hessian matrix is not guaranteed to be positive definite. If it is not positive definite, the Hessian matrix must be regularised. To make sure variances are positive after update, a change of variable scheme is usually used, i.e., $\boldsymbol{\Sigma}_{\mathtt{n}}$ is transformed to the log-domain $\tilde{\boldsymbol{\Sigma}}_{\mathtt{n}} = \log \boldsymbol{\Sigma}_{\mathtt{n}}$ and the Hessian and gradient is evaluated with respect to $\tilde{\boldsymbol{\Sigma}}_{\mathtt{n}}$. After update, it is converted back via $\boldsymbol{\Sigma}_{\mathtt{n}} = \exp(\tilde{\boldsymbol{\Sigma}}_{\mathtt{n}})$. It is noted that due to the linear approximation in Eq. (3.88), the mean update equation no logner guarantees the auxiliary function does not decrease. It is important to explicitly evaluate the auxiliary function after every update to make sure the auxiliary function does not decrease. If it is not the case, a back-off scheme must be used to reduce the step size until the auxiliary function stops decreasing. After a new estimate of noise model parameters is obtained, it is also possible to update the expansion point and re-estimate the noise model parameters. This process can be done for multiple iterations until the auxiliary function stops increasing.

- *Other approximation-based schemes*

  Besides using the log-normal assumption and VTS to approximate the mismatch function, there are a number of works attempting to approximate the mismatch function using various other schemes. These schemes can be categorised into two broad approaches: a sampling-based approach, and a more accurate functional approximation approach.

  In the sampling-based approaches, speech and noise samples are drawn from the known clean speech and noise models and then used to generate the corrupted speech according to the mismatch function. For example, for the static mean compensation,

$$\begin{aligned} \boldsymbol{\mu}_{\mathtt{sy}}^{(m)} &= \mathcal{E}\{\boldsymbol{y}_t | m\} \\ &= \int\int \mathbf{f}(\boldsymbol{x}^{\mathtt{s}}, \boldsymbol{n}^{\mathtt{s}}, \boldsymbol{\mu}_{\mathtt{h}}) \mathcal{N}(\boldsymbol{x}^{\mathtt{s}}; \boldsymbol{\mu}_{\mathtt{sx}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}) \mathcal{N}(\boldsymbol{n}^{\mathtt{s}}; \boldsymbol{\mu}_{\mathtt{sn}})\, d\boldsymbol{x}^{\mathtt{s}}\, d\boldsymbol{n}^{\mathtt{s}} \\ &\approx \sum_{k=1}^{K} \mathbf{f}(\boldsymbol{x}_k^{\mathtt{s}}, \boldsymbol{n}_k^{\mathtt{s}}, \boldsymbol{\mu}_{\mathtt{h}}) \end{aligned} \tag{3.91}$$

  where $(\boldsymbol{x}_k^{\mathtt{s}}, \boldsymbol{n}_k^{\mathtt{s}})$ are the k-th sample draw from the joint distribution of clean speech and noise. Note that here it is assumed that the speech and noise distributions are independent of each other. In Data driven PMC (DPMC) [62], Monte-Carlo sampling is used. The advantage of DPMC is that when the number of samples, K, approaches to infinity, the

compensation will be exact. However, DPMC usually requires a large number of samples for each Gaussian component for a robust estimation, which is computationally expensive. Another sampling approach is to use the unscented transform [117], in which samples are drawn from the joint distribution in a deterministic way, given the means and variances of the joint distributions. Let $\boldsymbol{u} = [\boldsymbol{x}^{\mathrm{s}}, \boldsymbol{n}^{\mathrm{s}}]$. If the feature vector is a D-dimensional vector, $\boldsymbol{u}$ is a $2D$ dimensional vector, and $4D+1$ samples are drawn based on:

$$\boldsymbol{u}_k = \begin{cases} \boldsymbol{\mu}_{\mathrm{u}}^{(m)} & \text{if } k = 0 \\ \boldsymbol{\mu}_{\mathrm{u}}^{(m)} + \left(\sqrt{(2D+\kappa)\boldsymbol{\Sigma}_{\mathrm{u}}^{(m)}}\right)_k & \text{if } k = 1, \cdots, 2D \\ \boldsymbol{\mu}_{\mathrm{u}}^{(m)} - \left(\sqrt{(2D+\kappa)\boldsymbol{\Sigma}_{\mathrm{u}}^{(m)}}\right)_k & \text{if } k = 2D, \cdots, 4D+1 \end{cases} \tag{3.92}$$

where $\boldsymbol{\mu}_{\mathrm{u}}^{(m)}$ and $\boldsymbol{\Sigma}_{\mathrm{u}}^{(m)}$ are the mean and covariance of joint distribution $p(\boldsymbol{u}|m)$; $\boldsymbol{u}_k$ is the k-th sample of $\boldsymbol{u}$; $(\boldsymbol{\Sigma})_k$ denotes the k-th column of the transposed Cholesky factorisation of $\boldsymbol{\Sigma}$; κ is a tunable parameter. There are also different weights associated with different samples: for $k = 1$, the weight is $\frac{\kappa}{2D+\kappa}$ and $\frac{1}{2(2D+\kappa)}$ for the other samples. The advantage of using the unscented transform is that the number of samples growths linearly as the dimension increases. The unscented transform was first explored in [107] for HMM compensation and further explored in [156, 166, 224, 266].

Besides the sampling-based approaches, it is also possible to use the functional approximation methods to approximate the mismatch function. VTS itself is a linear approximation of the non-linear mismatch function. A piecewise linear approximation is proposed in [54]; this is extended in [121, 221] where a linear spline interpolation is used to approximate the mismatch function. It is expected that by using piecewise linear approximation, a better modelling accuracy can be obtained. However, for efficiency reasons, these methods usually operate in the log-spectral domain and the model parameters are converted to the cepstral domain after compensation.

The Model-based schemes discussed in this section are based on some mismatch function (e.g., Eq. (3.33)) to derive the model transform. Model transformation is performed by combining the clean-speech models with noise models using the mismatch function. As a result, they make strong assumptions about how the environment influences the speech vectors and only a small number of noise model parameters need to be estimated. Model transforms constructed in this way are normally referred to as *predictive transforms*. The predictive transforms can be constrasted with the *adaptive transforms* discussed in Section 3.2, e.g., the MLLR and CMLLR transforms. The adaptive transforms modify the acoustic model such that speech data observed in the new acoustic condition can be better represented. They

generally make few assumptions on how the acoustic conditions affect the speech data. As a consequence, the adaptive transforms can be extended to environment adaptation as well. On the other hand, as adaptive transforms do not rely on the prior knowledge of the environment model, they usually require thousands of frames of adaptation data for a robust transform estimation. It is possible to combine the predictive techniques and the adaptive technqiues to adapt the acoustic model to the target environment and speaker. This is discussed in [66]. Moreover, as the predictive transforms are complementary to the adaptive transforms, these two types of transforms can be combined in a way that there is an "orthogonality" between them. This attribute will be discussed in Section 5.2.2.

3.3.3 Handling Reverberant Environments

In this section, schemes that are designed to improve the robustness of ASR systems in reverberant environments will be discussed. These schemes can be roughly categorised into three groups: the linear filtering approach, the feature enhancement approach and the model compensation approach.

3.3.3.1 Linear Filtering

As reverberation can be described as an convolution of the clean speech signal $x(\tau)$ with a RIR $h_r(\tau)$, de-reverberation can be achieved by linearly filtering the observed signal $z(\tau)$. In a single-channel setup, this can be expressed as

$$\hat{x}(\tau) = \sum_{\delta=0}^{T} g(\delta) z(\tau - \delta) \tag{3.93}$$

where $\mathbf{G} = \{g(\delta)\}$ is the set of coefficients of the linear filter, and $\hat{x}(\tau)$ is the de-reverberated signal. In a multi-channel setup (for example, when signals are recorded by a microphone array), this can be expressed as

$$\hat{x}(\tau) = \sum_{l=1}^{L} \sum_{\delta=0}^{T} g^{(l)}(\delta) z^{(l)}(\tau - \delta) \tag{3.94}$$

where $z^{(l)}(\tau)$ is the signal recorded by the l-th microphone, and $g^{(l)}(\delta)$'s are its associated linear filter coefficients. In this setup, $\mathbf{G} = \{g^{(l)}(\delta) | l = 1, \cdots, L\}$ is the set of linear filter coefficients to be estimated. This is usually referred to as the *blind deconvolution* problem, which has been extensively studied for the digital communication signal processing (e.g., [3]) and has also been applied to de-reverberation in speech recognition (e.g., [12, 81, 104, 130]). For convenience of presentation, the above equations assume the filtering is done in the time

domain. It is also possible to perform the filtering in the short time Fourier transform (STFT) domain.

When a microphone array is used to record signals, *inverse filtering* methods can be used to linearly filter the received signals. The multiple input/output inverse theorem presented in [172] guarantees that when the RIRs of different channels are known and do not share common zeros, an exact realization of an inverse filter exists. The subspace methods (e.g., [78, 87]) are developed based on this theorem. These methods estimate the inverse filter independent of the source (i.e., speech signals) characteristics. The smallest eigenvector of the covariance matrix of the signals recorded by the microphone array is calculated as the RIR estimate. Regularisation is needed to make the inverse filter robust against the background noise and the fluctuation of the RIR. The limitations of subspace methods are obvious: due to the nature of smallest eigenvector, subspace methods are numerically unstable and sensitive to the background noise.

Beamforming [90] is also a candidate solution for signal de-reverberation when a microphone array is used. In beamforming, signals coming from the selected direction are maintained while signals from all the other directions are attenuated. This spatial selectivity is useful to depress the reverberation and background noise as well. The simplest method is the delay-and-sum beamforming [90, 187, 252], in which the signals received by the microphone array are time-aligned. The time-aligned signals are then weighted and added together. Interfering signals, including background noise and reflected signals that are not coincident with the speech are thus attenuated. The delay-and-sum beamforming can be extended to filter-and-sum beamforming [35, 252], in which each microphone signal is separately filtered before combination. The filter coefficients can also made to be adaptive and are updated on a sample-by-sample basis. This yields adaptive beamforming [152]. However, the adaptive filtering methods generally assume that the target and observed signals are uncorrelated. As the clean signal is highly correlated with its reverberation, the methods suffer from the "signal cancellation" effect [250] because of the reflected copies of the target signal. Conventional beamforming focuses on improving the SNR of the filtered signal, which is not directly related to the performance of speech recognition. In [220], an integrated beamforming and speech recognition method is proposed. The linear filter coefficients in the filter-and-sum beamformer, $\mathbf{G}$, are optimised to maximise the likelihood of the feature vector sequence , $\{\boldsymbol{z}_t\}$, extracted from the filtered signals, given the correct transcription $\mathcal{H}$. This is expressed as:

$$\hat{\mathbf{G}} = \arg\max_{\mathbf{G}} \sum_{t,m} \gamma_t^{(m)} \log p(z_t(\mathbf{G})|m) \tag{3.95}$$

where $\gamma_t^{(m)}$ is the posterior probability of component m at time t, aligned against transcription $\mathcal{H}$, and $\boldsymbol{z}_t$ is the extracted feature vector which dependents on beamforming coefficients $\mathbf{G}$. A gradient descent method is used to optimise $\mathbf{G}$.

Though multi-channel data is helpful to combat reverberation, it is not practical to record signals using multiple microphones in many applications. Therefore, the single distant microphone scenario will be the focus in this thesis.

3.3.3.2 Feature Enhancement-based Schemes

The previous linear filtering approaches aim to restore the clean speech signals in the time domain. Most of the linear filtering schemes are designed to minimise the squared differences between enhanced signals and the original clean signals. However, this objective function is not consistent with the feature vectors used in speech recognition systems. ASR systems usually use short-time spectrum derived features, which discard lots of information in the waveform. Instead, feature enhancement-based schemes aim to restore the clean *features* rather than the clean *signals*.

Feature enhancement can be carried out in various stages in feature extraction. For example, in the Mel-spectrum power domain, the relationship between corrupted speech features and the clean speech features can be approximated as:

$$|z(t,k)|^2 \approx \sum_{\delta=0}^{n} |h(\delta,k)|^2 |x(t-\delta,k)|^2 \tag{3.96}$$

where $|z(t,k)|^2$ and $|x(t,k)|^2$ are the power of the k-th Mel filter at the t-th frame, calculated for the signals $z(\tau)$ and $x(\tau)$ respectively, and $|h(\delta,k)|^2$ is the power spectrum-domain representation of the RIR. Note that it is assumed that the background noise can be ignored and the cross-term difference between frames can be discarded. Given the power spectrum representation of corrupted signals, techniques such as correlation analysis [57], iterative least square methods [138] and non-negative matrix factorisation [123] can be used to estimate the underlying clean speech power spectrum. Alternatively, if the reverberation time T_{60} is known in advance, a very simple model of the power spectrum representation of RIR can be utilised [142]. This model assumes that $h(\delta,k)$ exponentially decays with respect to δ and the decay rate can be determined by T_{60}.

Feature enhancement can be also carried out in the log Mel-spectrum domain or the cepstral coefficient domain. In this case, the objective of feature enhancement is to infer the most likely clean feature vectors $\{\hat{\boldsymbol{x}}_t | t = 1, \cdots, T\}$, given a sequence of reverberant feature vectors $\{\boldsymbol{z}_t | t = 1, \cdots, T\}$, where $\boldsymbol{z}_t$ and $\hat{\boldsymbol{x}}_t$ are the features in either the log Mel-spectrum domain or the cepstral coefficient domain. In [134], the underlying speech $\boldsymbol{x}_t$ is modelled by

a switching linear dynamic system [176], and the observation feature vector $\boldsymbol{z}_t$ is generated according to a mismatch function (e.g., if enhancement is performed in the cepstral coefficient domain, Eq. (3.42) will be the mismatch function). An extended Kalman filter algorithm [229] is used to recursively calculate a posterior distribution of $\boldsymbol{x}_t$, $p(\boldsymbol{x}_t|\boldsymbol{z}_t, \cdots, \boldsymbol{z}_1)$. To make the inference tractable, linearisation similar as the vector Taylor series (VTS) expansion is used to approximate the mismatch function. Given the posterior distribution, the enhanced feature vector is given by

$$\begin{aligned}\hat{\boldsymbol{x}}_t &= \mathcal{E}\{\boldsymbol{x}|\boldsymbol{z}_t, \cdots, \boldsymbol{z}_1\} \\ &= \int \boldsymbol{x} p(\boldsymbol{x}|\boldsymbol{z}_t, \cdots, \boldsymbol{z}_1) d\,\boldsymbol{x}\end{aligned} \tag{3.97}$$

Note that since the posterior probability $p(\boldsymbol{x}_t|\boldsymbol{z}_t, \cdots, \boldsymbol{z}_1)$ is available as a by-product of feature enhancement, the posterior probability can be also propagated to the back-end acoustic model in the uncertainty decoding framework. This model-based feature enhancement scheme was originally presented in [138], which only handles the background noise free scenario, but is extended in [151] to handle the background noise as well.

3.3.3.3 Reverberant Model Adaptation

The third approach to handling the effects of reverberation is to modify the model parameters such that the compensated model better reflects the nature of reverberant speech. This will be referred to as reverberant model adaptation or model-based approaches to reverberant noise robustness.

General adaptation methods, such as MAP adaptation and linear transform-based adaptation, which were discussed in section 3.2, can be used to reduce the mismatch between a clean acoustic model and the reverberant data. However, these techniques are not designed to handle the effect of long reverberation. For example, the standard CMLLR adaptation only transforms the current frame for a group of components, which is clearly a limitation when the reverberant speech frame is usually affected by a few preceding frames. In [71], the standard CMLLR transform is extended to transform a set of neighbouring frames in a way that the reverberant feature vector can be normalised. In this scheme, the likelihood of the t-th reverberant feature vector $\boldsymbol{z}_t$ in the m-th component is evaluated as

$$p\left(\left[\begin{array}{c}\boldsymbol{z}_t^{\mathsf{s}} \\ \Delta\boldsymbol{z}_t \\ \Delta^2\boldsymbol{z}_t\end{array}\right]\middle| m\right) = \mathcal{N}\left(\mathbf{A}\left[\begin{array}{c}\boldsymbol{z}_{t+w}^{\mathsf{s}} \\ \vdots \\ \boldsymbol{z}_{t-w}^{\mathsf{s}}\end{array}\right] + \boldsymbol{b}; \left[\begin{array}{c}\boldsymbol{\mu}_{\mathsf{x}}^{(m)} \\ \overline{\boldsymbol{\mu}}_{\mathsf{x}}\end{array}\right], \left[\begin{array}{cc}\boldsymbol{\Sigma}_{\mathsf{x}}^{(m)} & \mathbf{0} \\ \mathbf{0} & \overline{\boldsymbol{\Sigma}}_{\mathsf{x}}\end{array}\right]\right) \tag{3.98}$$

where $\boldsymbol{z}_t^{\mathsf{s}}$, $\Delta\boldsymbol{z}_t$ and $\Delta^2\boldsymbol{z}_t$ are the reverberant static, delta and delta-delta feature at time t, w is the window size of neighbouring frames, $\mathbf{A}$ is a transformation matrix with a size

$(2w+1)D \times (2w+1)D$; D is the dimension of static features, and $(\overline{\boldsymbol{\mu}}_{\mathbf{x}}, \overline{\boldsymbol{\Sigma}}_{\mathbf{x}})$ are the parameters of the nuisance space, which are shared by all the components. As the nuisance space is shared by all the components, it can be ignored during decoding. Thus only the first $3D$ rows of the matrix $\mathbf{A}$ need to be estimated. This is referred to as the *direct CMLLR* transform. A number of structures, reflecting the reverberation nature, were also imposed on the matrix $\mathbf{A}$ in [71], which yields a number of linear transform schemes tailored to reverberation.

One limitation of the linear transform based schemes for reverberation model compensation is that these schemes often have a large number of parameters to be estimated. For example, the above direct CMLLR transform includes $3D \times (2w+1)D$ parameters. This number is significantly larger than the number of parameters in the standard CMLLR ($3D \times 3D$) when a large window size w is used. By using the reverberation model, e.g., a reverberant mismatch function, it is possible to reduce the number of transform parameters. One approach to do this is to compensate the model parameters according to the mismatch function prior to decoding. For instance, in [198] where a single Gaussian per state HMM system was used and the effect of background noise is ignored, it is proposed that the clean static mean of state j, $\boldsymbol{\mu}_{\mathtt{sx}}^{(j)}$, can be adapted to $\boldsymbol{\mu}_{\mathtt{sz}}^{(j)}$ by

$$\boldsymbol{\mu}_{\mathtt{sz}}^{(j)} = \mathbf{C} \log \left(\sum_{\delta=0}^{n} \exp(\mathbf{C}^{-1} \boldsymbol{\mu}_{\mathtt{sx}}^{(j-\delta)} + \boldsymbol{\mu}_{\delta}^{(1)}) \right) \tag{3.99}$$

where $j-\delta$ denotes the δ-th state before the current state j, $\boldsymbol{\mu}_{\mathtt{sx}}^{(j-\delta)}$ is the clean static mean vector of that preceding state, $\boldsymbol{\mu}_{\delta}^{(1)}$ is the state-level filter coefficient, describing the energy dispersion of state $j-\delta$ to state j in the log Mel-spectrum domain. This compensation can be extended to the GMM-HMM system as suggested in [103, 198]: for the m-th Gaussian component in j-th state, the compensation formula is expressed as:

$$\boldsymbol{\mu}_{\mathtt{sz}}^{(j,m)} = \mathbf{C} \log \left(\exp(\mathbf{C}^{-1} \boldsymbol{\mu}_{\mathtt{sx}}^{(j,m)} + \boldsymbol{\mu}_{0}^{(1)}) + \sum_{\delta=1}^{n} \exp(\mathbf{C}^{-1} \overline{\boldsymbol{\mu}}_{\mathtt{sx}}^{(j-\delta)} + \boldsymbol{\mu}_{\delta}^{(1)}) \right) \tag{3.100}$$

where $\overline{\boldsymbol{\mu}}_{\mathtt{sx}}^{(j)}$ is the composite mean of state j:

$$\overline{\boldsymbol{\mu}}_{\mathtt{sx}}^{(j)} = \sum_{m \in j} w_m \boldsymbol{\mu}_{\mathtt{sx}}^{(j,m)} \tag{3.101}$$

Based on the compensation form in Eq. (3.99) (or Eq. (3.100) for GMM-HMM systems), parameters can be compensated using the log-normal approximation [198]. Dynamic parameters and variances were not compensated in [198]. The state-level filter coefficient $\{\boldsymbol{\mu}_{\delta}^{(1)} | \delta = 0, \cdots, n\}$ is optimised to maximise the likelihood of reverberant speech. Compared with the mismatch function in Eq. (3.42) (ignoring the background noise), the key idea in

[198] is to use the mean vectors of preceding states to approximate the preceding frames. However, there are several issues with this approach. Due to the use of mean vectors instead of the actual preceding frames, it introduces unwanted smoothness over the speech trajectories, which reduces the discrimination of compensated models. The use of the composite mean in Eq. (3.101) also introduces another level of smoothness. Another issue is that it is difficult to determine the identities of the previous n states prior to decoding. In [198], to determine the previous n states, it is assumed that each state is usually occupied only once. However, it is still difficult to determine the previous states for the start state of each HMM. In practice, a rather small n (=4) was used in [198], which limits its advantage by taking the previous states into account. Due to this limitation, only a connected digit recognition task, which uses 16-state whole word HMM models, was performed in [198].

The second issue is partially addressed in [103], where a duration model of each state is used. Spline interpolation is applied to the contour of energy of each Mel filter bank for each HMM, therefore the spectrogram of each HMM model can be recreated, and the previous δ-th clean speech can be approximated by the interpolated spectrogram. This removes the assumption that each state is only occupied once. Further, [103] argues that when the acoustic unit is context dependent, e.g., a tri-phone HMM, its left context is often restricted, which can be used to determine the previous states. This is illustrated in Figure 3.11. In the figure, state j is considered as the last state of the $\mathtt{u}-\mathtt{v}+\mathtt{w}$ model. Its immediate left two states are already known, while further left states must be in the HMMs with the centre phone $\mathtt{u}$ and right context $\mathtt{v}$. This information can be used to infer the identifies of n preceding states. For example, in [103], a left HMM model with the smallest dissimilarity between its exit state and the entry state of the current HMM model is selected. This left HMM is joined to the current HMM and the spectrogram of the composed HMM is interpolated, which can provide a longer context. The spline interpolation is also used in [103] for dynamic parameter compensation. The noise model parameter estimation is done by a simple model which assumes that $\boldsymbol{\mu}_{\delta}^{(1)}$ exponentially decays and the decay rate is determined by T_{60}. Therefore T_{60} is the only noise parameter to be estimated, which can be done via a grid search. This scheme was applied to both a connected digit recognition task and a medium vocabulary continuous speech recognition task (the reverberant WSJ0 task). Large performance improvements are observed on both tasks.

As argued in [258], the above model adaptation schemes may not be optimal for reverberant speech recognition, as the acoustic model is adapted prior to decoding. The adapted acoustic model is still a standard HMM, which uses the conditional independence assumption

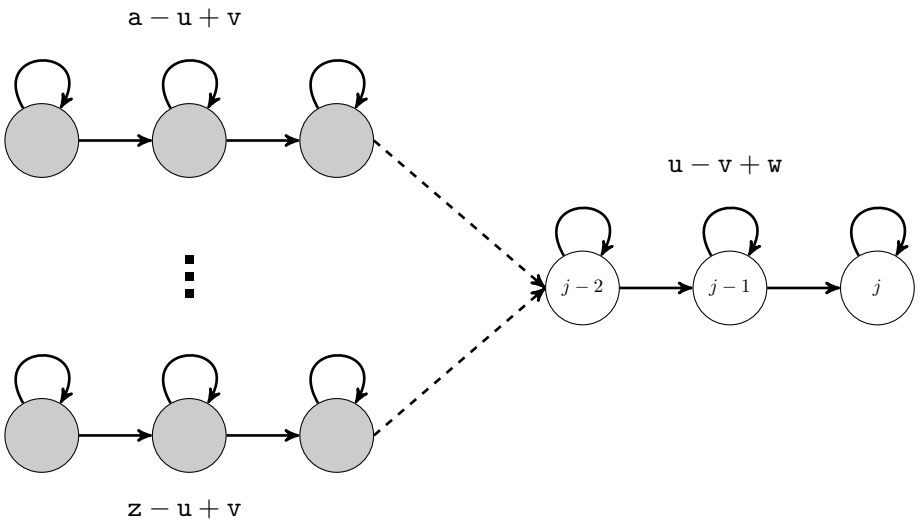


Figure 3.11: Illustration of restricted left contexts when a triphone HMM is used. State j is the last state in the HMM representing the context $\mathtt{u}-\mathtt{v}+\mathtt{w}$. Its immediate left two states are known: the state $j-1$ and the state $j-2$, while further left states are restricted in the HMMs with the "$-\mathtt{u}+\mathtt{v}$" context.

between the neighbouring frames given the hidden state. To model the long term reverberation effect more accurately, dynamic model adaptation is needed. For example, in [232] a first-order linear prediction is used where the adapted static mean of component m at time t is expressed as:

$$\boldsymbol{\mu}_{\mathtt{sz}}^{(m,t)} = \mathbf{C}\log\left(\exp(\mathbf{C}^{-1}(\boldsymbol{\mu}_{\mathtt{sx}}^{(m)} + \boldsymbol{\mu}_{10})) + \alpha\exp(\mathbf{C}^{-1}\boldsymbol{z}_{t-1}^{\mathtt{s}})\right) \tag{3.102}$$

where $\boldsymbol{\mu}_{10}$ is the spectral distortion term in the cesptral coefficient domain. Note that due to the dependency on the previous observation vector $\boldsymbol{z}_{t-1}^{\mathtt{s}}$, the adapted mean vector varies with respect to time t. The $\boldsymbol{\mu}_{10}$ and the linear prediction coefficient α can be estimated from adaptation data using an EM algorithm. During decoding, the acoustic model is adapted at every frame.

Dynamic model adaptation is also explored in [41], where a de-reverberation module is used to produce a sequence of clean speech estimates $(\hat{\boldsymbol{x}}_t)_{t=1}^{T}$. To compensate for the dynamic uncertainty caused by the de-reverberation module, a combination of static and dynamic model adaptation was used. For the m-th Gaussian component, the likelihood of $\boldsymbol{z}_t$ is evaluated by:

$$p(\boldsymbol{z}_t|m,t) = \mathcal{N}(\hat{\boldsymbol{x}}_t; \boldsymbol{\mu}_{\mathtt{x}}^{(m)}, \mathbf{L}\boldsymbol{\Sigma}_{\mathtt{x}}^{(m)}\mathbf{L}^{\mathsf{T}} + \mathbf{A}\boldsymbol{\Sigma}_t\mathbf{A}^{\mathsf{T}}) \tag{3.103}$$

where $\mathbf{L}$ is used for static variance compensation, while $\mathbf{A}$ is the matrix for dynamic variance compensation with the time varying uncertainty $\boldsymbol{\Sigma}_t$. In [41], a simple form is used for the

Algorithm 3.1: Recursion in the Modified Viterbi decoding algorithm presented in [212].

for *t=1, . . . , T* **do**
  for *each state j* **do**
    • $\gamma_t^{(j)} = \max_i\{\gamma_{t-1}^{(i)} \cdot a_{ij} \cdot p(\boldsymbol{z}_t^{\mathsf{s}}|q_t = j, q_{t-1} = i)\}$
    where

$$p(\boldsymbol{z}_t^{\mathsf{s}}|q_t = j, q_{t-1} = i) \approx \int p(\boldsymbol{z}_t^{\mathsf{s}}|\boldsymbol{x}_t^{\mathsf{s}}, (\hat{\boldsymbol{x}}_{t-\tau}^{\mathsf{s}}|i)_{\tau=1}^{n}, \overline{\boldsymbol{h}}_t) p(\overline{\boldsymbol{h}}_t|\boldsymbol{\eta}) p(\boldsymbol{x}_t^{\mathsf{s}}|q_t = j)\, d\boldsymbol{x}_t^{\mathsf{s}}\, d\overline{\boldsymbol{h}}_t \tag{3.106}$$

    • Update state dependent enhanced feature $(\hat{\boldsymbol{x}}_{t-\tau}^{\mathsf{s}}|j)_{\tau=0}^{n}$
  end
end

uncertainty term: $\boldsymbol{\Sigma}_t$ is assumed to be diagonal and is calculated as the differences between observed and de-reverberated speech vectors.

Dynamic model compensation is further extended in [212], in which conventional Viterbi decoding is modified to perform a frame-by-frame model compensation. In contrast to most reverberant robustness work, a time varying reverberant observation model is assumed in [212], i.e., the observation vector $\boldsymbol{z}_t^{\mathsf{s}}$ is generated according to the following mismatch function [1] :

$$\boldsymbol{z}_t^{\mathsf{s}} = \mathbf{C} \log \left(\sum_{\delta=0}^{n} \exp(\mathbf{C}^{-1}(\boldsymbol{x}_{t-\delta}^{\mathsf{s}} + \boldsymbol{h}_{t-\delta})) \right) \tag{3.104}$$

where $\overline{\boldsymbol{h}}_t = [\boldsymbol{h}_t^{\mathsf{T}}, \cdots, \boldsymbol{h}_{t-n}^{\mathsf{T}}]^{\mathsf{T}}$ follows a reverberant observation model,

$$\overline{\boldsymbol{h}}_t \sim \mathcal{N}(\boldsymbol{\mu}_{\overline{\mathtt{h}}}, \boldsymbol{\Sigma}_{\overline{\mathtt{h}}}) \tag{3.105}$$

The parameter of the reverberant observation model, $\boldsymbol{\eta} = (\boldsymbol{\mu}_{\overline{\mathtt{h}}}, \boldsymbol{\Sigma}_{\overline{\mathtt{h}}})$, is independent of time t. Given this reverberant observation model $\boldsymbol{\eta}$, the core recursion part in the Viterbi decoding algorithm is modified as shown in Algorithm 3.1 [2], in which, $\gamma_t^{(j)}$ is the Viterbi score of the partial path ending at state j at time t, a_{ij} is the transition probability from state i to state j, q_t is the state at time t. The integral in Eq. (3.106) is approximated by its maximal value

[1] The cepstral coefficient domain mismatch function is used for the illustration purpose while work in [212] used a log Mel-spectrum domain mismatch function.

[2] Note that only static parameters are used, as the algorithm in [212] only handles static parameter compensation.

of the integrand, therefore it becomes the following constrained optimisation:

$$
\begin{aligned}
(\hat{\boldsymbol{x}}_t^{\mathbf{s}}, \hat{\overline{\boldsymbol{h}}}_t) &= \max_{\boldsymbol{x}_t^{\mathbf{s}}, \overline{\boldsymbol{h}}_t} p(\overline{\boldsymbol{h}}_t|\boldsymbol{\eta}) p(\boldsymbol{x}_t^{\mathbf{s}}|j) \\
\text{subject to} \quad \boldsymbol{z}_t^{\mathbf{s}} &= \mathbf{C} \log \left(\sum_{\delta=0}^{n} \exp(\mathbf{C}^{-1}(\boldsymbol{x}_{t-\delta}^{\mathbf{s}} + \boldsymbol{h}_{t-\delta})) \right)
\end{aligned} \tag{3.107}
$$

To solve this constrained optimisation problem, the clean speech vector, $(\boldsymbol{x}_{t-\tau}^{\mathbf{s}})_{\tau=1}^{n}$, needs to be known in advance. This is replaced by $(\hat{\boldsymbol{x}}_{t-\tau}^{\mathbf{s}}|i)_{\tau=1}^{n}$, which is recursively computed and updated as the by-product of the previous optimisation for state i at time $t-1$. This is referred to as the REMOS model. Compared with the conventional Viterbi algorithm, the state emitting distribution $p(\boldsymbol{z}_t^{\mathbf{s}}|q_t, q_{t-1})$ not only depends on the current state $q_t = j$, but also the previous state $q_{t-1} = i$. Furthermore, a clean speech estimate $(\hat{\boldsymbol{x}}_t^{\mathbf{s}}|j)$ is computed for every state and is cached to compute enhanced features for the future frames. It is demonstrated that REMOS maintains high word accuracies even in severely reverberant environments and it exhibits a high flexibility to the changes of speaker positions and even changes of the room due to its time varying reverberation model assumption. However, this accuracy comes at a high cost. It is clear from Algorithm 3.1 that the optimisation problem needs to be solved for every frame and every pair of current state and preceding state. The computational cost is prohibitive even for small vocabulary tasks. Moreover, REMOS only compensates the static parameters and uses single Gaussian HMM models.

This section has discussed two main approach to reverberant model adaptation: the static and dynamic reverberant model adaptation approaches. It is also interesting to compare these two approaches. The static model adaptation nevertheless introduces smoothness over speech trajectories and thus sacrifices modelling accuracy. However, the dynamic model adaptation approach, although it models the reverberation more accurately, is computationally expensive, as a large number of Gaussian components need to be adapted for every frame. This is not practical for medium to large vocabulary speech recognition. Inspired by the static model adaptation approach, Chapter 4 will present a new model-based approach to robust speech recognition in reverberant environments. In contrast to existing static model adaptation schemes such as [103] and [198], the preceding clean speech frames are not inferred from the preceding states. Instead, extended statistics are explicitly extracted at the Gaussian component level. These statistics are then used to represent the preceding clean speech frames. This gives two advantages over the existing schemes. One is that there is no need to determine the preceding states. The other advantage is that the distribution of preceding frames is now conditioned on Gaussian components, which gives a more detailed representation of

preceding frames than the distribution conditioned on the state. This also enables an efficient reverberant noise model estimation procedure using the ML criterion.

3.4 Adaptive Training

In the previous sections, a number of adaptation schemes were discussed. These schemes are designed to adapt well-trained acoustic models. Specially collected and well controlled data are used to build the acoustic models, which often exhibit minimal unwanted variations. In this way, it is expected that the statistical algorithms presented in Chapter 2 can build acoustic models that reflect only the desired, phone, variability. However, it is often impractical to collect a large amount of well-controlled data from the same source. This has led to a growing trend towards building speech recognition systems on *found* data. For example, when building speech recognition systems for a broadcast news transcription task, it is possible to also include data collected for other tasks, e.g., conversational telephone speech data. As found data are collected from various acoustic conditions and exhibit a broad range of variability, they are *non-homogeneous* in nature.

The simplest approach to build acoustic models on found data is to treat them as a single homogeneous block regardless of the differences in acoustic conditions. This is normally referred to as *multi-style* or *multi-condition* training. The problem with this approach is that acoustic models trained in this way represent not only the desired variability but also many other unwanted, non-speech, variabilities as well. All the variabilities are represented using a single acoustic model, which sacrifices the ability to discriminate desired variabilities.

The concept of adaptive training, first proposed in [9], is an alternative approach to train acoustic models on non-homogeneous data. The basic idea of adaptive training is to separate the desired or intrinsic variabilities from the unwanted or extrinsic variabilities: a canonical model is built to represent the intrinsic variability while the extrinsic variabilities are modelled by a set of transforms. This is a natural concept for speech recognition. Many related schemes can be viewed as instances of adaptive training. For example, a number of schemes, like CMN/CVN [11] and Gaussianisation [208], are designed to normalise feature vectors such that extrinsic variabilities can be reduced in the feature space. This was then extended to model-dependent feature normalisation, for instance, vocal tract length normalisation (VTLN) [146], where feature normalisation is done via maximising the log-likelihood of normalised data with respect to the back-end acoustic model. Model-based adaptive training was first proposed in [9] under the name of speaker adaptive training, where a set of model transforms (e.g., MLLR in [9]) are used to represent the speaker variability, and the canonical model is estimated given

all the speaker transforms. This was subsequently extended to adaptively training of multi-cluster HMMs [70] and also noise adaptive training [45, 108, 122, 159]. Since the focus of this thesis is model-based approaches to robust speech recognition, the notion of adaptive training will be used to stand for the model-based adaptive training schemes in the rest of this thesis.

In adaptive training, rather than using all the multi-condition data as a single block, the training data $\mathcal{O}$ is split into a set of homogeneous blocks, $\{\mathcal{O}^{(1)}, \cdots, \mathcal{O}^{(R)}\}$, each of which has the same acoustic condition. Two sets of parameters are then used to separately model the intrinsic and extrinsic variabilities:

- **A set of model transforms** $\{\mathcal{T}^{(1)}, \cdots, \mathcal{T}^{(R)}\}$.
  A set of R model transforms are used to represent the extrinsic variabilities in each of the R homogeneous data blocks. The transforms are used to adapt the acoustic model $\mathcal{M}_{\mathsf{c}}$ to each acoustic condition:
  $$\mathcal{M}^{(r)} = \mathcal{F}(\mathcal{M}_c; \mathcal{T}^{(r)}) \qquad \forall r \tag{3.108}$$
  where $\mathcal{F}$ is the mapping function; $\hat{\mathcal{O}}^{(r)}$ is the normalised features for the r-th block, and $\mathcal{M}^{(r)}$ is the r-th adapted model.

- **A canonical model** $\mathcal{M}_{\mathsf{c}}$.
  Given the model transforms for each homogeneous block, an acoustic model $\mathcal{M}_{\mathsf{c}}$ can be built on all the training data. As the extrinsic variabilities are represented by the transforms, the acoustic model $\mathcal{M}_{\mathsf{c}}$ represents the intrinsic variabilities and will be referred to as the *canonical model*. The canonical model is estimated by maximising the following objective function:
  $$\begin{aligned} \mathcal{M}_{\mathsf{c}} &= \arg\max_{\mathcal{M}} \sum_{r=1}^{R} \mathcal{L}(\mathcal{O}^{(r)}; \mathcal{M}^{(r)}) \\ &= \arg\max_{\mathcal{M}} \sum_{r=1}^{R} \mathcal{L}(\mathcal{O}^{(r)}; \mathcal{F}(\mathcal{M}_{\mathsf{c}}; \mathcal{T}^{(r)})) \end{aligned} \tag{3.109}$$
  where $\mathcal{L}(\mathcal{O}; \mathcal{M})$ is the criterion function to be maximised. The ML objective function is normally used. Since the extrinsic variabilities are "absorbed" by the model transforms, the canonical model represents the intrinsic attributes of speech. Hence, it is expected to be more amenable to adaptation than a multi-condition trained acoustic model.

It is worthwhile emphasising that the canonical model is estimated given the model transform. Therefore, the canonical model cannot be used directly; it must be adapted by an appropriate transform. This often requires multi-pass decoding where the first past can be performed using multi-condition trained acoustic models.

Algorithm 3.2: The ML-SAT algorithm.

Given: The observation sequences $\{\mathcal{O}^{(r)}\}$ for R homogeneous blocks and their associated transcription $\{\mathcal{H}^{(r)}\}$;

Output: A canonical model $\mathcal{M}_{\mathtt{c}}$ which maximises the likelihood function in Eq. (3.110) ;

1. Initialisation:

Initialise the canonical model $\mathcal{M}_{\mathtt{c}}$ as $\mathcal{M}_{\mathtt{c},0}$ using the SI model ;

Initialise the transform $\mathcal{T}^{(r)}$ as $\mathcal{T}_0^{(r)}$ using the identity transform ;

2. Iterating:

for *k=1, . . . , K* **do**

• *update speaker transforms by:*

for *r=1, . . . , R* **do**

$$\mathcal{T}_k^{(r)} = \arg\max_{\mathcal{T}} \sum_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{H}^{(r)}, \mathcal{M}_{\mathtt{c},k-1}, \mathcal{T}_{k-1}^{(r)}, \mathcal{O}^{(r)}) \sum_t \log p(\boldsymbol{o}_t^{(r)}|\theta_t, \mathcal{M}_{\mathtt{c},k-1}, \mathcal{T})$$

end

• *update the canonical model by:*

$$\mathcal{M}_{\mathtt{c},k} = \arg\max_{\mathcal{M}} \sum_r \sum_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{H}^{(r)}, \mathcal{M}_{\mathtt{c},k-1}, \mathcal{T}_k^{(r)}, \mathcal{O}^{(r)}) \sum_t \log p(\boldsymbol{o}_t^{(r)}|\theta_t, \mathcal{M}, \mathcal{T}_k^{(r)})$$

end

3. Finish:

The optimal canonical model $\mathcal{M}_{\mathtt{c}}$ is given by $\mathcal{M}_{\mathtt{c},k}$

3.4.1 Speaker Adaptive Training

In speaker adaptive training, a set of speaker transforms $\{\mathcal{T}^{(r)}\}$ are estimated for each speaker in the training data. The canonical model $\mathcal{M}_{\mathtt{c}}$ is built on all the data given the speaker transforms. Assuming that there are R speakers in the training data, the ML speaker adaptive training (ML-SAT) is to maximise the log-likelihood function of two sets of parameters on R speakers' data:

$$\mathcal{L}(\mathcal{M}_{\mathtt{c}}, \{\mathcal{T}^{(r)}\}) = \sum_{r=1}^{R} \log p(\mathcal{O}^{(r)}; \mathcal{H}^{(r)}, \mathcal{M}_{\mathtt{c}}, \mathcal{T}^{(r)}) \tag{3.110}$$

where $\{\mathcal{H}^{(r)}\}$ is the transcription of the r-th speaker's data.

Due to the hidden states and Gaussian components in GMM-HMM models, the above log-likelihood function is normally maximised using EM with an auxiliary function defined by

$$\mathcal{Q}(\hat{\mathcal{M}}_{\mathtt{c}}, \{\hat{\mathcal{T}}^{(r)}\}) = \sum_r \sum_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathcal{H}^{(r)}, \mathcal{M}_{\mathtt{c}}, \mathcal{T}^{(r)}, \mathcal{O}^{(r)}) \sum_t \log p(\boldsymbol{o}_t^{(r)}|\theta_t, \hat{\mathcal{M}}_{\mathtt{c}}, \hat{\mathcal{T}}^{(r)}) \tag{3.111}$$

where $\boldsymbol{\theta}$ is the hidden component sequence [1], $p(\boldsymbol{\theta}|\mathcal{H}^{(r)}, \mathcal{M}_{\mathtt{c}}, \mathcal{T}^{(r)}, \mathcal{O}^{(r)})$ is the posterior prob-

[1] As in the standard HMMs case, the symbol $\boldsymbol{\theta}$ is used to indicate a sequence of both hidden state and hidden Gaussian component. This allows a convenient notation to derive formula for updating the Gaussian component parameters.

ability of a particular hidden component sequence for the r-th speaker's data; θ_t is the t-th hidden component in $\boldsymbol{\theta}$, and $\boldsymbol{o}_t^{(r)}$ is the t-th observation in the r-th block; $\hat{\mathcal{M}}_{\mathsf{c}}, \{\hat{\mathcal{T}}^{(r)}\}$ are the parameters to be updated in M-step. Jointly maximising the likelihood function with respect to the canonical model and the speaker transforms is difficult, and an iterative procedure illustrated in Algorithm 3.2, is used.

Depending on the form of speaker model adaptation, a number of speaker adaptive training schemes can be derived. Among them, linear transform-based speaker adaptive training, e.g., [9, 67] and cluster-based adaptive training [68] are two widely used schemes.

3.4.1.1 Linear Transform-based Speaker Adaptive Training

Linear transform-based adaptation schemes are widely used for speaker adaptation. Hence, it is natural to extend linear transform-based adaptation schemes to speaker adaptive training (SAT). The concept of SAT was first proposed in [9], in which the MLLR transform was used for speaker adaptation. This is normally referred to as MLLR-based SAT. The m-th mean vector $\boldsymbol{\mu}^{(m)}$ is adapted to $\boldsymbol{\mu}^{(mr)}$ for the r-th speaker:

$$\boldsymbol{\mu}^{(mr)} = \mathbf{A}^{(r)}\boldsymbol{\mu}^{(m)} + \boldsymbol{b}^{(r)} = \mathbf{W}^{(r)}\boldsymbol{\xi}^{(m)} \tag{3.112}$$

where $\mathbf{W}^{(r)} = [\mathbf{A}^{(r)}, \boldsymbol{b}^{(r)}]$ is the r-th speaker transform and $\boldsymbol{\xi}^{(m)} = [\boldsymbol{\mu}^{(m)\mathsf{T}}, 1]^{\mathsf{T}}$. Note that Eq. (3.112) is similar to the Eq. (3.8), except the index of an homogeneous block (in this case, the speaker index) is used to emphasise that the canonical model estimation depends on all the speaker transforms [1]. With this adaptation form, the general auxiliary function for adaptive training in Eq. (3.111) can be written more explicitly as follows:

$$\mathcal{Q}(\hat{\mathcal{M}}_{\mathsf{c}}; \mathcal{M}_{\mathsf{c}}) = -\frac{1}{2}\sum_{r,m,t}\gamma_t^{(mr)}\left\{\log|\hat{\boldsymbol{\Sigma}}^{(m)}| + (\boldsymbol{o}_t^{(r)} - \mathbf{W}^{(r)}\hat{\boldsymbol{\xi}}^{(m)})^{\mathsf{T}}\hat{\boldsymbol{\Sigma}}^{(m)-1}(\boldsymbol{o}_t^{(r)} - \mathbf{W}^{(r)}\hat{\boldsymbol{\xi}}^{(m)})\right\} \tag{3.113}$$

where the canonical model $\mathcal{M}_{\mathsf{c}} = \{(\boldsymbol{\mu}^{(m)}, \boldsymbol{\Sigma}^{(m)})\}$ and $\gamma_t^{(mr)}$ is the posterior probability of $\boldsymbol{o}_t^{(r)}$ in component m at time t. It is shown in [9] that the ML update of the canonical mean vectors can be performed by

$$\hat{\boldsymbol{\mu}}^{(m)} = \mathbf{G}^{(m)-1}\mathbf{k}^{(m)} \tag{3.114}$$

where $\mathbf{G}^{(m)}$ and $\mathbf{k}^{(m)}$ are the statistics collected at the component level:

[1] It is also possible to use regression trees instead of a global transform. However, for the sake of notation convenience, only the global transform is used. Extending to the use of a regression tree in adaptive training is straightforward.

$$\begin{aligned}\mathbf{G}^{(m)} &= \sum_{r,t} \gamma_t^{(mr)} \mathbf{A}^{(r)\mathsf{T}} \boldsymbol{\Sigma}^{(m)-1} \mathbf{A}^{(r)} \\ \mathbf{k}^{(m)} &= \sum_{r,t} \mathbf{A}^{(r)\mathsf{T}} \boldsymbol{\Sigma}^{(m)-1} \left(\boldsymbol{o}_t^{(r)} - \boldsymbol{b}^{(r)} \right)\end{aligned} \tag{3.115}$$

It is difficult to update the mean and variance simultaneously in the MLLR-based SAT. Variance update is normally performed after mean update, and is very similar to the standard variance update formulation:

$$\hat{\boldsymbol{\Sigma}}^{(m)} = \text{diag}\left(\frac{\sum_{r,t} \gamma_t^{(mr)} (\boldsymbol{o}_t^{(r)} - \mathbf{W}^{(r)} \hat{\boldsymbol{\xi}}^{(m)}) (\boldsymbol{o}_t^{(r)} - \mathbf{W}^{(r)} \hat{\boldsymbol{\xi}}^{(m)})^{\mathsf{T}}}{\sum_{r,t} \gamma_t^{(mr)}} . \right) \tag{3.116}$$

A practical limitation of the MLLR-based SAT scheme can be seen from the required statistics in Eqs. (3.115): it needs to store a full matrix, $\mathbf{G}^{(m)}$, for every Gaussian component. For a large vocabulary ASR system which often has over 10,000 components, this requires a considerable amount of memory and computational load. Furthermore, the variance update needs to be performed in a separate iteration. Due to this limitation, MLLR-based SAT is less frequently used in large vocabulary ASR systems than the following CMLLR-based SAT scheme [67].

In CMLLR-based SAT, the speaker adaptation on training data is performed using the CMLLR scheme. As the CMLLR transform can be performed to transform the features rather than the model parameters, the adaptive training of the canonical model parameters can be more easily implemented after a feature transform. To illustrate this, the formulation of feature transform on the observation vector in Eq. (3.25) is rewritten in the following with the explicit notations of the speaker index r and the regression base class p_m:

$$\hat{\boldsymbol{o}}_t^{(mr)} = \mathbf{A}^{(r,p_m)} \mathbf{o}_t^{(r)} + \boldsymbol{b}^{(r,p_m)} = \mathbf{W}^{(r,p_m)} \boldsymbol{\zeta}^{(r,p_m)} \tag{3.117}$$

where p_m is the regression base class for component m, $\hat{\boldsymbol{o}}_t^{(mr)}$ is the transformed observation vector for component m, $\mathbf{W}^{(r,p_m)}$ is the linear transform for the base class p_m and r-th homogeneous block. Note that due to the use of regression tree here, different linear transforms are applied according to the back-end components, and the CMLLR transform here is a model-based transform. The auxiliary function for canonical model ML update is expressed as:

$$\mathcal{Q}(\hat{\mathcal{M}}_{\mathsf{c}}, \mathcal{M}_{\mathsf{c}}) = -\frac{1}{2} \sum_{r,m,t} \gamma_t^{(mr)} \left\{ \log |\hat{\boldsymbol{\Sigma}}^{(m)-1}| + (\boldsymbol{o}_t^{(mr)} - \hat{\boldsymbol{\mu}}^{(m)})^{\mathsf{T}} \hat{\boldsymbol{\Sigma}}^{(m)-1} (\boldsymbol{o}_t^{(mr)} - \hat{\boldsymbol{\mu}}^{(m)}) \right\} \tag{3.118}$$

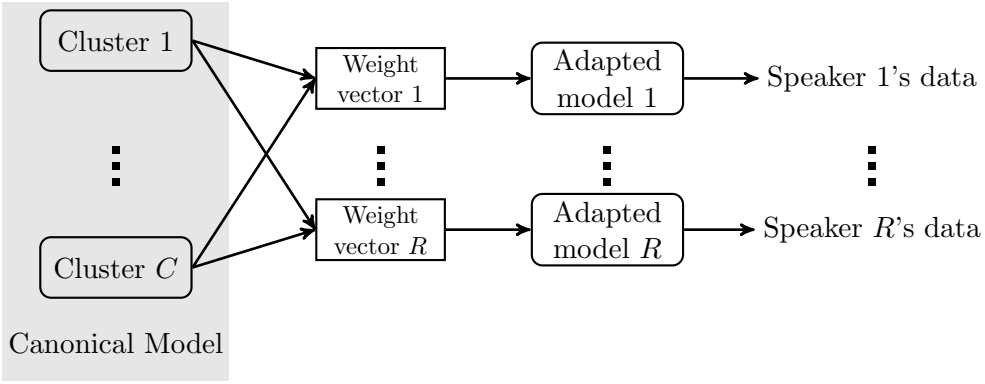


Figure 3.12: Illustration of cluster based adaptive training.

Comparing this with the standard auxiliary function, the main difference is that the transformed observation $\boldsymbol{o}_t^{(mr)}$ is used instead of the original observation $\boldsymbol{o}_t^{(r)}$. The update formulae are given by:

$$\hat{\boldsymbol{\mu}}^{(m)} = \frac{\sum_{r,t} \gamma_t^{(mr)} \boldsymbol{o}_t^{(mr)}}{\sum_{r,t} \gamma_t^{(mr)}} \tag{3.119}$$

$$\hat{\boldsymbol{\Sigma}}^{(m)} = \frac{\sum_{r,t} \gamma_t^{(mr)} (\boldsymbol{o}_t^{(mr)} - \boldsymbol{\mu}^{(m)})(\boldsymbol{o}_t^{(mr)} - \boldsymbol{\mu}^{(m)})^{\mathsf{T}}}{\sum_{r,t} \gamma_t^{(mr)}} \tag{3.120}$$

As these update formula are very similar to those in the standard Baum-Welsh algorithm, the adaptive training can be performed with the similar computational cost and memory load. As a result, this CMLLR-based SAT scheme is widely used in the state-of-the-art large vocabulary ASR systems.

3.4.1.2 Cluster-based Adaptive Training

Cluster-based adaptive training (CAT)[68] is an alternative approach to adaptive training. In CAT, rather than transforming a single acoustic model as in the linear transform-based adaptive training, multiple clusters are interpolated where the interpolation weights are speaker dependent. This is illustrated in Figure 3.12. When the weights are binary vectors or *hard weights*, this scheme reduces to the traditional cluster dependent modelling, where a particular homogeneous block is associated with a specific acoustic model. The CAT or eigenvoices [137] schemes generalise the cluster dependent modelling where speaker-dependent *soft weights* are used.

In a standard CAT model, different clusters share the same covariance matrices, transition matrices and mixture weights. Only the mean vectors differ between clusters. In this way,

the model adaptation using a C-cluster CAT model can be expressed as:

$$\hat{\boldsymbol{\mu}}^{(mr)} = \sum_{c=1}^{C} \lambda_c^{(r,p_m)} \boldsymbol{\mu}_c^{(m)} = \mathbf{M}^{(m)} \boldsymbol{\lambda}^{(r,p_m)} \tag{3.121}$$

where p_m is the regression base class of the component m, $\hat{\boldsymbol{\mu}}^{(mr)}$ is the adapted mean for the r-th speaker, $\boldsymbol{\lambda}^{(r,p_m)} = [\lambda_1^{(r,p_m)}, \cdots, \lambda_C^{(r,p_m)}]^{\mathsf{T}}$ is the weight vector for the base class p_m and the r-th speaker, and

$$\mathbf{M}^{(m)} = [\boldsymbol{\mu}_1^{(m)}, \cdots, \boldsymbol{\mu}_C^{(m)}] \tag{3.122}$$

$\boldsymbol{\mu}_c^{(m)}$ is the c-th cluster mean for the component m. A bias cluster is often used, whose weight is fixed as 1. In this case, the adapted mean is also expressed as:

$$\hat{\boldsymbol{\mu}}^{(mr)} = \boldsymbol{\mu}^{(m)} + \sum_{c=1}^{C} \lambda_c^{(r,p_m)} \boldsymbol{\mu}_c^{(m)} = \boldsymbol{\mu}^{(m)} + \mathbf{M}^{(m)} \boldsymbol{\lambda}^{(r,p_m)} \tag{3.123}$$

where $\boldsymbol{\mu}^{(m)}$ is the bias, and $\mathbf{M}^{(m)}$ is the non-bias cluster parameters.

The ML-CAT training algorithm is used to estimate two sets of parameters: the canonical model parameters $\mathcal{M}_{\mathsf{c}} = \{(\mathbf{M}^{(m)}, \boldsymbol{\Sigma}^{(m)})\}$ and the speaker transforms $\{\mathcal{T}^{(r)} | r = 1, \cdots, R\}$ in which $\mathcal{T}^{(r)} = \{\boldsymbol{\lambda}^{(r,p)} | p = 1, \cdots, P\}$. Again, interleaved updates of two sets of parameters are used. The auxiliary function of the canonical model parameters for CAT can be expressed as:

$$\mathcal{Q}(\hat{\mathcal{M}}_{\mathsf{c}}; \mathcal{M}_{\mathsf{c}}) = -\frac{1}{2} \sum_{r,m,t} \gamma_t^{(mr)} \left\{ \log |\hat{\boldsymbol{\Sigma}}^{(m)}| + (\boldsymbol{o}_t^{(r)} - \hat{\mathbf{M}}^{(m)} \boldsymbol{\lambda}^{(r,p_m)})^{\mathsf{T}} \hat{\boldsymbol{\Sigma}}^{(m)-1} (\boldsymbol{o}_t^{(r)} - \hat{\mathbf{M}}^{(m)} \boldsymbol{\lambda}^{(r,p_m)}) \right\} \tag{3.124}$$

By differentiating the above auxiliary function with respect to the canonical model parameters and equating it to zero, the ML update of canonical model parameters can be obtained by:

$$\hat{\mathbf{M}}^{(m)} = \mathbf{G}^{(m)} \mathbf{K}^{(m)-1} \tag{3.125}$$

$$\hat{\boldsymbol{\Sigma}}^{(m)} = \text{diag} \left\{ \frac{\mathbf{L}^{(m)} - \hat{\mathbf{M}}^{(m)} \mathbf{K}^{(m)}}{\sum_{r,m,t} \gamma_t^{(mr)}} \right\} \tag{3.126}$$

where

$$\mathbf{G}^{(m)} = \sum_r (\sum_t \gamma_t^{(mr)}) \boldsymbol{\lambda}^{(r,p_m)} \boldsymbol{\lambda}^{(r,p_m)\mathsf{T}} \tag{3.127}$$

$$\mathbf{K}^{(m)} = \sum_{r,t} \gamma_t^{(mr)} \boldsymbol{\lambda}^{(r,p_m)} \boldsymbol{o}_t^{(r,p_m)\mathsf{T}} \tag{3.128}$$

$$\mathbf{L}^{(m)} = \sum_{r,m,t} \gamma_t^{(mr)} \boldsymbol{o}_t^{(r)} \boldsymbol{o}_t^{(r)\mathsf{T}} \tag{3.129}$$

The auxiliary function for the r-th weight vector update is:

$$\mathcal{Q}(\hat{\mathcal{T}}^{(r)};\mathcal{T}^{(r)}) = -\frac{1}{2}\sum_{t,m}(\boldsymbol{o}_t^{(r)} - \mathbf{M}^{(m)}\hat{\boldsymbol{\lambda}}^{(r,p_m)})^{\mathsf{T}}\boldsymbol{\Sigma}^{(m)-1}(\boldsymbol{o}_t^{(r)} - \mathbf{M}^{(m)}\hat{\boldsymbol{\lambda}}^{(r,p_m)}) \tag{3.130}$$

Therefore, the ML update of the r-th transform can be performed by:

$$\hat{\boldsymbol{\lambda}}^{(r,p)} = \mathbf{G}_{\mathbf{w}}^{(r,p)-1}\boldsymbol{k}_{\mathbf{w}}^{(r,p)} \tag{3.131}$$

where

$$\mathbf{G}_{\mathbf{w}}^{(r,p)} = \sum_{m\in p}\sum_{t}\gamma_t^{(mr)}\mathbf{M}^{(m)\mathsf{T}}\boldsymbol{\Sigma}^{(m)-1}\mathbf{M}^{(m)} \tag{3.132}$$

$$\mathbf{k}_{\mathbf{w}}^{(r,p)} = \sum_{m\in p}\sum_{t}\gamma_t^{(mr)}\mathbf{M}^{(m)\mathsf{T}}\boldsymbol{\Sigma}^{(m)-1}\boldsymbol{o}_t^{(r)} \tag{3.133}$$

and $m \in p$ means the summation is over the components which belongs to the base class p. Compared with linear transform-based adaptive training, only a small number of parameters, C parameters for each base class and each speaker, need to be estimated in CAT. As a consequence, the CAT scheme is suitable for very rapid speaker adaptation.

3.4.2 Noise Adaptive Training

The concept of adaptive training has also been extended to noise adaptive training. Most of the environmental robustness techniques described in section 3.3 assume the back-end acoustic model is built on the clean data. There are two major limitations to use the clean-trained acoustic models. The first one is that this requires training on carefully controlled clean data and the amount of clean data is often limited in practice. The second limitation is that most model compensation schemes are based on some mismatch function, which makes many approximations about how clean speech is distorted by the environment. Adaptive trained canonical acoustic models will take the approximations into account. This is helpful to address the approximation errors in deriving the mismatch functions.

Compared with feature-based noise adaptive training schemes, e.g., [45], model-based noise adaptive training schemes can propagate the uncertainty caused by environment noise to the back-end acoustic models [160]. This approach has been extensively studied in recent years. Two notable pieces of work in this category are VTS-based adaptive training (VAT) in [122] and Joint adaptive training (JAT) [159]. In the following, the VAT algorithm will be presented to illustrate noise adaptive training.

In line with the adaptive training framework, there are two sets of parameters to be estimated on R homogeneous data blocks $\{\mathcal{Y}^{(r)}\}$ in a noise adaptive training scheme [1]: the

[1] In order to distinguish different domains, rather than using $\mathcal{O}$ or $\boldsymbol{o}_t$ to denote the observation vector as in the previous section, $\boldsymbol{y}$ is used to denote the observed the noisy speech, while $\boldsymbol{x}$ is used to denote the clean speech.

canonical model $\mathcal{M}_{\mathsf{c}} = \{(\boldsymbol{\mu}_{\mathsf{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathsf{x}}^{(m)})\}$ and a set of noise transform parameters $\boldsymbol{\Phi} = \{\boldsymbol{\Phi}^{(r)}\}$. Note a homogeneous data block is now a block of data which shares the same noise condition. In the case of VTS-based model compensation $\boldsymbol{\Phi}^{(r)} = (\boldsymbol{\mu}_{\mathsf{sn}}^{(r)}, \boldsymbol{\mu}_{\mathsf{h}}^{(r)}, \boldsymbol{\Sigma}_{\mathsf{n}}^{(r)})$. The canonical model and the noise transform parameters are estimated to maximise the likelihood of training data. This is done by optimising the following auxiliary function in the EM framework:

$$\mathcal{Q}(\hat{\mathcal{M}}_{\mathsf{c}}, \{\hat{\boldsymbol{\Phi}}^{(r)}\}) = \sum_{t,m,r} \gamma_t^{(mr)} \log p(\boldsymbol{y}_t^{(r)}; \hat{\boldsymbol{\mu}}_{\mathsf{y}}^{(mr)}, \hat{\boldsymbol{\Sigma}}_{\mathsf{y}}^{(mr)}) \tag{3.134}$$

where $\boldsymbol{y}_t^{(r)}$ is the t-th observation vector in the r-th block; $\gamma_t^{(mr)}$ is the posterior probability of $\boldsymbol{y}_t^{(r)}$ in component m at time t, which is calculated using the current canonical model $\mathcal{M}_{\mathsf{c}}$ and current noise transform parameters $\boldsymbol{\Phi}^{(r)}$; $\hat{\boldsymbol{\mu}}_{\mathsf{y}}^{(mr)}$ and $\hat{\boldsymbol{\Sigma}}_{\mathsf{y}}^{(mr)}$ are the compensated model parameters, expressed by the following equations:

$$\hat{\boldsymbol{\mu}}_{\mathsf{sy}}^{(mr)} = \mathbf{f}(\hat{\boldsymbol{\mu}}_{\mathsf{sx}}^{(m)}, \hat{\boldsymbol{\mu}}_{\mathsf{h}}^{(r)}, \hat{\boldsymbol{\mu}}_{\mathsf{sn}}^{(r)}) \tag{3.135}$$

$$\hat{\boldsymbol{\Sigma}}_{\mathsf{sy}}^{(mr)} = \mathrm{diag}\left(\mathbf{J}_{\mathsf{x}}^{(mr)} \hat{\boldsymbol{\Sigma}}_{\mathsf{sx}}^{(m)} \mathbf{J}_{\mathsf{x}}^{(mr)\mathsf{T}} + \mathbf{J}_{\mathsf{n}}^{(mr)} \hat{\boldsymbol{\Sigma}}_{\mathsf{sn}}^{(r)} \mathbf{J}_{\mathsf{n}}^{(mr)\mathsf{T}}\right) \tag{3.136}$$

for the static parameter part and

$$\hat{\boldsymbol{\mu}}_{\Delta\mathsf{y}}^{(mr)} = \mathbf{J}_{\mathsf{x}}^{(m)} \hat{\boldsymbol{\mu}}_{\Delta\mathsf{x}}^{(m)} \tag{3.137}$$

$$\hat{\boldsymbol{\Sigma}}_{\Delta\mathsf{y}}^{(mr)} = \mathrm{diag}\left(\mathbf{J}_{\mathsf{x}}^{(mr)} \hat{\boldsymbol{\Sigma}}_{\Delta\mathsf{x}}^{(m)} \mathbf{J}_{\mathsf{x}}^{(mr)\mathsf{T}} + \mathbf{J}_{\mathsf{n}}^{(mr)} \hat{\boldsymbol{\Sigma}}_{\Delta\mathsf{n}}^{(r)} \mathbf{J}_{\mathsf{n}}^{(mr)\mathsf{T}}\right) \tag{3.138}$$

for the delta parameter part. Delta-delta parameters are compensated in a similar way to the delta parameters. Note $\mathbf{f}$ is the mismatch function defined in Eq. (3.33), and $\mathbf{J}_{\mathsf{x}}^{(mr)}$ and $\mathbf{J}_{\mathsf{n}}^{(mr)}$ are the Jacobian matrices of the mismatch function with respect to the clean speech and noise respectively, defined for the m-th component and the r-th data block. These Jacobian matrices are evaluated using the current model parameters $\mathcal{M}_{\mathsf{c}}$ and the current noise transform $\boldsymbol{\Phi}^{(r)}$.

The two sets of parameters are optimised in an interleaved fashion. Given the canonical model parameters, estimation of noise model parameters was described in Section 3.3.2.3. In parallel with the noise model parameter estimation, there are two main approaches to estimating the canonical model parameters. One is the EM-based approach, e.g., [108], where the clean speech at time t is treated as another latent variable and another level of EM is introduced. The auxiliary function becomes

$$\mathcal{Q}(\hat{\mathcal{M}}_{\mathsf{c}}; \mathcal{M}_{\mathsf{c}}) = \sum_{r,t,m} \gamma_t^{(mr)} \int p(\boldsymbol{x}_t | \boldsymbol{y}_t^{(r)}, m, \mathcal{M}_{\mathsf{c}}, \boldsymbol{\Phi}^{(r)}) \log \mathcal{N}(\boldsymbol{x}_t; \hat{\boldsymbol{\mu}}_{\mathsf{x}}^{(m)}, \hat{\boldsymbol{\Sigma}}_{\mathsf{x}}^{(m)})\, d\boldsymbol{x}_t \tag{3.139}$$

With this auxiliary function, the canonical model update formula are thus

$$\hat{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)} = \frac{\sum_{r,m,t} \gamma_t^{(mr)} \mathcal{E}\{\boldsymbol{x}_t | \boldsymbol{y}_t^{(r)}, m, \mathcal{M}_{\mathtt{c}}, \boldsymbol{\Phi}^{(r)}\}}{\sum_{r,m,t} \gamma_t^{(mr)}} \tag{3.140}$$

$$\hat{\boldsymbol{\Sigma}}_{\mathtt{x}}^{(m)} = \frac{\sum_{r,m,t} \gamma_t^{(mr)} \mathcal{E}\{\boldsymbol{x}_t \boldsymbol{x}_t^{\mathsf{T}} | \boldsymbol{y}_t^{(r)}, m, \mathcal{M}_{\mathtt{c}}, \boldsymbol{\Phi}^{(r)}\}}{\sum_{r,m,t} \gamma_t^{(mr)}} - \hat{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)} \hat{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)\mathsf{T}} \tag{3.141}$$

An approximation is needed to calculate the posterior probability $p(\boldsymbol{x}_t | \boldsymbol{y}_t^{(r)}, m, \mathcal{M}_{\mathtt{c}}, \boldsymbol{\Phi}^{(r)})$, e.g., using VTS to approximate the static mismatch function for each Gaussian component m:

$$\boldsymbol{y}_t^{\mathtt{s}} | m \approx \mathbf{f}(\boldsymbol{\mu}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{h}}^{(r)}, \boldsymbol{\mu}_{\mathtt{sn}}^{(r)}) + \mathbf{J}_{\mathtt{x}}^{(mr)} (\boldsymbol{x}_t^{\mathtt{s}} - \boldsymbol{\mu}_{\mathtt{sx}}^{(m)}) + \mathbf{J}_{\mathtt{n}}^{(mr)} (\boldsymbol{n}_t^{\mathtt{s}} - \boldsymbol{\mu}_{\mathtt{sn}}^{(m)}) \tag{3.142}$$

Note that the expansion point is determined by the current canonical model parameters and the current noise model parameters. It does not depend on the parameters to be estimated during the M-step. This is similar to the factor analysis model [205] and the posterior probability $p(\boldsymbol{x}_t^{\mathtt{s}} | \boldsymbol{y}_t^{\mathtt{s}}, m, \mathcal{M}_{\mathtt{c}}, \boldsymbol{\Phi}^{(r)})$ is again Gaussian. However, due to the VTS approximation of the mismatch function, the Jacobian matrices are fixed while they should change with respect to the model parameters. Therefore the resultant algorithm cannot guarantee that the likelihood increases in every step.

The other approach to canonical model parameter estimation is to use standard gradient descent schemes to directly optimise the canonical model. This approach is explored in [122, 158], where slightly different optimisation schemes are used. As algorithms developed in [122] are used in this thesis, the adaptive training scheme presented therein will be briefly reviewed. In the scheme presented in [122], a second-order method is used to directly optimise the auxiliary function in Eq. (3.134) with respect to the canonical model parameters. For example, the static mean update has the form:

$$\hat{\boldsymbol{\mu}}_{\mathtt{sx}}^{(m)} = \boldsymbol{\mu}_{\mathtt{sx}}^{(m)} - \eta_{\mu} \left(\frac{\partial^2 \mathcal{Q}}{\partial \boldsymbol{\mu}_{\mathtt{sx}}^{(m)} \partial \boldsymbol{\mu}_{\mathtt{sx}}^{(m)}} \right)^{-1} \frac{\partial \mathcal{Q}}{\partial \boldsymbol{\mu}_{\mathtt{sx}}^{(m)}} \tag{3.143}$$

where η_{μ} is a step size; the gradient and the Hessian matrix are evaluated at the current model parameters given by:

$$\frac{\partial \mathcal{Q}}{\partial \boldsymbol{\mu}_{\mathtt{sx}}^{(m)}} \approx \frac{1}{2} \sum_{t,r} \gamma_t^{(mr)} \mathbf{J}_{\mathtt{x}}^{(mr)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{sy}}^{(mr)-1} (\boldsymbol{y}_t^{\mathtt{s}(r)} - \boldsymbol{\mu}_{\mathtt{sy}}^{(mr)}) \tag{3.144}$$

$$\frac{\partial^2 \mathcal{Q}}{\partial \boldsymbol{\mu}_{\mathtt{sx}}^{(m)} \partial \boldsymbol{\mu}_{\mathtt{sx}}^{(m)}} \approx -\frac{1}{2} \sum_{t,r} \gamma_t^{(mr)} \mathbf{J}_{\mathtt{x}}^{(mr)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{sy}}^{(mr)-1} \mathbf{J}_{\mathtt{x}}^{(mr)} \tag{3.145}$$

Note the approximation sign is used because the Jacobian matrix $\mathbf{J}_{\mathtt{x}}^{(mr)}$ is assumed to be independent of the model parameters which are to be estimated. Similarly, the static variance is updated by:

$$\hat{\boldsymbol{\Sigma}}_{\mathtt{sx}}^{(m)} = \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)} - \eta_{\sigma} \left(\frac{\partial^2 \mathcal{Q}}{\partial \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)} \partial \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}} \right)^{-1} \frac{\partial \mathcal{Q}}{\partial \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}} \tag{3.146}$$

where η_{σ} is another step size, and

$$\frac{\partial \mathcal{Q}}{\partial \sigma_{\mathtt{sx},i}^{(m)2}} = \frac{1}{2} \sum_{t,r} \gamma_t^{(mr)} \sum_{d=1}^{D} \frac{(\mathbf{J}_{\mathtt{x}}^{(mr)})_{di}^2 \sigma_{\mathtt{sx},i}^{(m)2}}{\sigma_{\mathtt{sy},d}^{(mr)2}} \left(1 - \frac{(y_{t,d}^{\mathtt{s}(r)} - \mu_{\mathtt{y},d}^{(mr)})^2}{\sigma_{\mathtt{sy},d}^{(mr)2}} \right) \tag{3.147}$$

$$\begin{aligned} \frac{\partial^2 \mathcal{Q}}{\partial \sigma_{\mathtt{sx},i}^{(m)2} \partial \sigma_{\mathtt{sx},j}^{(m)2}} \approx -\frac{1}{2} \sum_{t,r} \gamma_t^{(mr)} \Bigg\{ & \sum_{d=1}^{D} \frac{(\mathbf{J}_{\mathtt{x}}^{(mr)})_{di}^2 \sigma_{\mathtt{sx},i}^{(m)2} (\mathbf{J}_{\mathtt{x}}^{(mr)})_{dj}^2 \sigma_{\mathtt{sx},j}^{(m)2}}{\sigma_{\mathtt{sy},d}^{(mr)4}} \cdot \left(1 - 2 \frac{(y_{t,d}^{\mathtt{s}(r)} - \mu_{\mathtt{y},d}^{(mr)})^2}{\sigma_{\mathtt{sy},d}^{(mr)2}} \right) \\ & - \mathbf{1}[i=j] \sum_{d=1}^{D} \frac{(\mathbf{J}_{\mathtt{x}}^{(mr)})_{di}^2 \sigma_{\mathtt{sx},i}^{(m)2}}{\sigma_{\mathtt{sy},d}^{(mr)2}} \left(1 - \frac{(y_{t,d}^{\mathtt{s}(r)} - \mu_{\mathtt{y},d}^{(mr)})^2}{\sigma_{\mathtt{sy},d}^{(mr)2}} \right) \Bigg\} \end{aligned} \tag{3.148}$$

$(\mathbf{J})_{ij}$ is the (i,j)-th element of matrix $\mathbf{J}$; $\boldsymbol{y}_{t,d}^{\mathtt{s}(r)}$ the i-th static parameter of $\boldsymbol{y}_t^{(r)}$, $\sigma_{\mathtt{sx},i}^{(m)2}$ and $\sigma_{\mathtt{sy},i}^{(mr)2}$ are the i-th variance of $\boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}$ and $\boldsymbol{\Sigma}_{\mathtt{sy}}^{(mr)}$ respectively. Again, when calculating the Hessian matrix, the Jacobian matrix is assumed to be fixed. Similar equations can be derived to update canonical model parameters for the dynamic features. Since the Hessian matrix is not guaranteed to be negative definite and is sometimes near to singular, extra care must be taken to make sure the update direction is correct. This is usually done by regularising the Hessian matrix such that it is sufficiently negative definite, i.e.,

$$\hat{\boldsymbol{\Sigma}}_{\mathtt{sx}}^{(m)} = \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)} - \eta_{\sigma} \left(\frac{\partial^2 \mathcal{Q}}{\partial \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)} \partial \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}} - \alpha \mathbf{I} \right)^{-1} \frac{\partial \mathcal{Q}}{\partial \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}} \tag{3.149}$$

To avoid negative variance, a change of variable, $\tilde{\sigma}_{\mathtt{x},i}^{(m)2} = \log \sigma_{\mathtt{x},i}^{(m)2}$, can be used. A heuristic, e.g., limiting the step size of variance update at each iteration in [159], can be helpful to stabilise the estimation process.

3.5 Summary

This chapter has reviewed adaptation schemes to improve the robustness of speech recognition systems. These schemes are designed to compensate the mismatch between training and test data caused by various acoustic factors, including speaker and environment. This is usually done by either normalising the feature vectors (feature-based approach) or compensating acoustic models (model-based approach). Adaptation schemes developed to compensate the

speaker factor are often known as speaker adaptation, while schemes designed to compensate the environment effects are commonly known as environmental robustness. Widely used speaker adaptation schemes, such as MAP, MLLR and CMLLR, were first reviewed in section 3.2. The first step in designing an environment robustness scheme is often to specify how the concerned speech signals are altered by the acoustic environments. Therefore, the impact of acoustic environments were reviewed in section 3.3.1, where a number of mismatch functions were derived to describe how the speech signals are corrupted by additive noise, channel distortion and reverberation in different scenarios. Based on these mismatch functions, a wide range of robustness schemes have been proposed in the literature. Some of them were reviewed in Section 3.3.2 for additive noise and channel distortions, and Section 3.3.3 for reverberant distortions. Adaptation schemes can also be used to "absorb" the non-speech variabilities on inhomogeneous training data. In this way, a canonical model can be trained to model only the speech variability. This is the adaptive training framework discussed in Section 3.4. Speaker adaptive training and noise adaptive training schemes were reviewed in section 3.4.1 and 3.4.2 respectively. Maximum likelihood estimation of the model/transform parameters was also presented in these sections.

Chapter 4
Reverberant Environmental Robustness

In section 3.3, the impact of reverberation on ASR systems was briefly discussed. To combat reverberation distortion, three approaches have been developed in the literature: robust signal processing techniques such as linear filtering; feature enhancement schemes and model compensation schemes. These approaches were briefly reviewed in section 3.3.3. In this chapter, a model-based approach to reverberant environmental robustness is proposed and discussed in detail. Section 4.1 will compare the additive, convolutional and reverberant noises and the corresponding mismatch functions. Based on these mismatch functions, a reverberant model compensation scheme will be developed in section 4.2. ML estimation of noise model parameters will be also presented. To enable acoustic model training on found data, an approach to training acoustic models adaptively on multi-condition reverberant data will be explored in section 4.3.

4.1 Reverberant Environment

When speech signals are recorded using a close-talking microphone, they are relatively clean: the background noises can be suppressed to a great extent and there is only one direct acoustic path from the speaker's mouth to the microphone. However, when the speech signals are produced in an enclosed room (e.g., office, living room or lecture hall) and captured by a distant microphone, there are multiple acoustic paths since the wavefront of the speech is reflected at the walls and other objects in the room. These reflections, which are delayed and attenuated copies of the original clean signals, are also captured by the distant microphone, resulting in *reverberation* distortion. In contrast to the short-term convolutional distortion (a.k.a. convolutional noise), such as microphone channel characteristic differences, the length of reverberation (ranging from 200ms to 1s, or even longer) is much longer than the length of analysis window (typically 25ms) commonly used in automatic speech recognition systems. Therefore the reverberation has a dispersive effective on the extracted feature vectors: the features are smeared along the time axis so that the current feature vector depends on the previous several feature vectors. Clearly, this invalidates the conditional independent assumption and significantly degrades the performance of HMM-based speech recognition systems. The recognition performance can be made even worse, when additive noise, such as background noise and interfering speakers' talking,is also picked up by the microphone, resulting in *additive noise or background noise* distortion. In this section, the effects of reverberant, convolutional and additive noise on ASR systems are discussed in detail.

4.1.1 Additive, Convolutional and Reverberant noise

Among the three main noise types, additive noise is the most common source of distortions. It can occur in almost any environment. Additive noise is the sound from the background, captured by the microphone, and linearly mixed with the target speech signals. It is statistically independent and additive to the original speech signals. Additive noise can be stationary (e.g., aircraft/train noise and noise caused by ventilation fan), slowly changing (e.g., noise caused by background music), or transient (e.g., in car noise caused by traffic or door slamming).

Convolutional noise is another major source of distortion that can adversely impact the ASR performance. This kind of distortion is mainly due to the transmission channel, such as microphones and telecommunication lines. A convolutional distortion is usually characterised by the impulse response of the corresponding transmission system. If the length of the impulse response is short compared to the analysis window, the distortion mainly causes spectral tilt

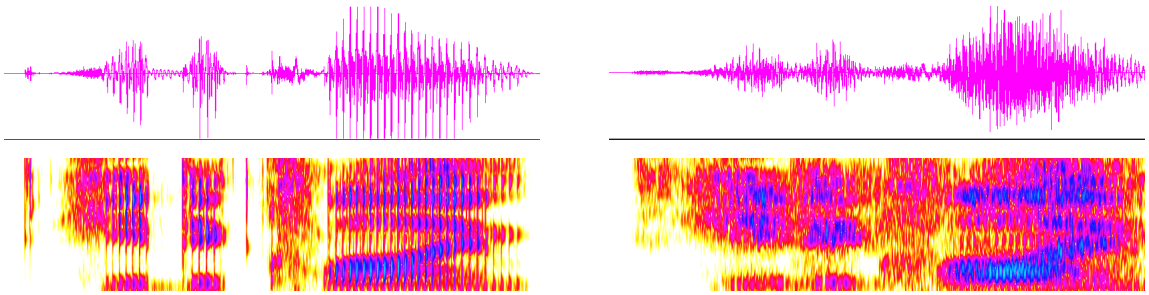


Figure 4.1: Waveforms and spectrograms of clean and reverberation noise corrupted utterance saying "he declined". Top: waveforms; Bottom: spectrograms ; Left: clean signals; right: reverberation noise corrupted signals.

[4]. This is because the convolution in the time domain is transfered to multiplication in the frequency domain and addition in the cepstral domain.

When the length of impulse response is significantly longer, the nature of its impact on the speech recognition systems is very different. In line with the conventional terminology usage in the literature, distortion caused by the short-time impulse response (less than 30ms) is referred to as "convolutional distortion" while the distortion caused by convolution with a system which has a longer impulse response is referred to as "reverberation" in this thesis. Reverberation is usually caused by multiple acoustic paths in an enclosed environment, such as a room environment. The length of a room impulse response (RIR) is usually measured by the so called reverberation time, T_{60}, which is the time needed for the power of reflections of a direct sound to decay by 60dB. The T_{60} value is usually significant longer than the analysis window. For example, T_{60} normally ranges from 200ms to 600ms in a small office room, 400ms to 800ms in a living room, while in a lecture room or concert hall environment, it can range from 1s to 2s or even longer. Figure 4.1 shows the impact of a reverberation noise (T_{60}=400ms) on both waveform and spectrogram. The detrimental effect of reverberation noise is clearly displayed. On the waveform, the reverberant voice is stretching out between syllables to fill in the gaps. This results in that segments with relatively weak energy are substantially masked by the previous segments with strong energy[177]. On the spectrogram, reverberation also causes a temporal smearing. The clear frequency structure in the clean signal's spectrogram is partially destroyed.

As the reverberation time is significantly longer than the analysis window, the effect of reverberation on the short time Fourier transform (STFT) based feature vector (e.g., MFCC and PLP) cannot be described as a function of the current clean feature vector only. Instead, the reverberant noise corrupted feature vector depends on a few previous clean feature vectors.

In the next section, the functional relationship between clean and corrupted feature vectors is described by the so called mismatch functions. To model the joint effect of reverberation and additive noise, the interaction between reverberation and additive noise is also considered.

4.1.2 Mismatch Functions

As discussed in section 3.3, the standard form used to describe the additive noise $n(\tau)$ and short-term convolutional noise $h(\tau)$ corrupting the clean speech $x(\tau)$ is expressed as:

$$y(\tau) = h(\tau) * x(\tau) + n(\tau)\,, \tag{4.1}$$

where the length of $h(\tau)$ is less than the length of analysis window used for feature extraction. Based on this time-domain mismatch function, the mismatch function in the cepstral domain, relating the corrupted speech static MFCCs $\boldsymbol{y}_t^{\mathtt{s}}$ to the clean speech vector, $\boldsymbol{x}_t^{\mathtt{s}}$, is written as:

$$\begin{aligned}\boldsymbol{y}_t^{\mathtt{s}} =& \mathbf{C}\log\left(\exp(\mathbf{C}^{-1}(\boldsymbol{x}_t^{\mathtt{s}} + \boldsymbol{\mu}_{\mathtt{h}})) + \exp(\mathbf{C}^{-1}\boldsymbol{n}_t^{\mathtt{s}})\right)\\ =& \mathbf{f}(\boldsymbol{x}_t^{\mathtt{s}}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{n}_t^{\mathtt{s}})\,,\end{aligned} \tag{4.2}$$

where $\mathtt{s}$ denotes the static parameters, $\boldsymbol{n}_t$ is the noise MFCCs, $\boldsymbol{\mu}_{\mathtt{h}}$ the MFCC representation of the convolutional noise, and $\mathbf{C}$ the DCT matrix. Normally, the convolutional noise $\boldsymbol{\mu}_{\mathtt{h}}$ is assumed to be an unknown constant, while the additive noise $\boldsymbol{n}_t$ is Gaussian distributed, i.e., $\boldsymbol{n}_t \sim \mathcal{N}(\boldsymbol{\mu}_{\mathtt{n}}, \boldsymbol{\Sigma}_{\mathtt{n}})$. It is further assumed that additive noise is stationary, and thus the expected values of noise dynamics, such as $\mathcal{E}(\Delta\boldsymbol{n}_t)$, $\mathcal{E}(\Delta^2\boldsymbol{n}_t)$, are zero. The noise parameters $\boldsymbol{\Phi}_{\mathtt{n}}$ for Eq. (4.2) are $\boldsymbol{\mu}_{\mathtt{h}}$, $\boldsymbol{\mu}_{\mathtt{n}}$ [1] and $\boldsymbol{\Sigma}_{\mathtt{n}}$. Here, the subscript or superscript $\mathtt{s}$ and Δ indicate the static and delta parameters respectively. The continuous time approximation [86] can be used to derive a mismatch function for the corresponding dynamic parameter mismatch function, e.g.,

$$\Delta\boldsymbol{y}_t \approx \frac{\partial \boldsymbol{y}_t^{\mathtt{s}}}{\partial t} = \frac{\partial \boldsymbol{y}_t^{\mathtt{s}}}{\partial \boldsymbol{x}_t^{\mathtt{s}}}\frac{\partial \boldsymbol{x}_t^{\mathtt{s}}}{\partial t} \approx \frac{\partial \boldsymbol{y}_t^{\mathtt{s}}}{\partial \boldsymbol{x}_t^{\mathtt{s}}}\Delta\boldsymbol{x}_t \tag{4.3}$$

As discussed in section 3.3.1.1, a few time domain mismatch functions can be derived to describe the effect of additive noise in a reverberant environment. Since there are usually many background noise sources in a typical reverberant environment, the mismatch function in Eq. (3.38) for the diffuse noise source is widely used in the literature (e.g., [135, 150, 241]). This time domain mismatch function is also investigated in this chapter. For convenience, it is duplicated in the following equation:

$$z(\tau) = h_r(\tau) * x(\tau) + n(\tau) \tag{4.4}$$

[1] Since $\boldsymbol{\mu}_{\Delta\mathtt{n}}$ and $\boldsymbol{\mu}_{\Delta^2\mathtt{n}}$ are zero, $\boldsymbol{\mu}_{\mathtt{n}}$ is used to refer the static noise mean $\boldsymbol{\mu}_{\mathtt{sn}}$ for convenience.

where $n(\tau)$ and $h_r(\tau)$ are the additive and reverberant noise terms, $z(\tau)$, $x(\tau)$ and $n(\tau)$ are the corrupted, clean and additive noise signals respectively.

Following [134], it is assumed that the cross-term and cross-band correlation can be ignored. The effect of reverberant distortion in the cepstral domain can be approximated as a combination of $n+1$ frame-level distortion terms, $\boldsymbol{\mu}_{\mathtt{l}} = [\boldsymbol{\mu}_{\mathtt{l}0}^{\mathsf{T}}, \cdots, \boldsymbol{\mu}_{\mathtt{l}n}^{\mathsf{T}}]^{\mathsf{T}}$, acting on a set of preceding clean MFCC features, $\boldsymbol{x}_t^{\mathtt{s}}, \cdots, \boldsymbol{x}_{t-n}^{\mathtt{s}}$, i.e.,

$$\begin{aligned}\boldsymbol{z}_t^{\mathtt{s}} &= \mathbf{C}\log\left(\sum_{\delta=0}^{n}\exp\left(\mathbf{C}^{-1}(\boldsymbol{x}_{t-\delta}^{\mathtt{s}}+\boldsymbol{\mu}_{\mathtt{l}\delta})\right)+\exp\left(\mathbf{C}^{-1}\boldsymbol{n}_t^{\mathtt{s}}\right)\right)\\ &= \mathbf{g}(\boldsymbol{x}_t^{\mathtt{s}}, \cdots, \boldsymbol{x}_{t-n}^{\mathtt{s}}, \boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{n}_t^{\mathtt{s}})\end{aligned} \tag{4.5}$$

Note this equation was derived in Eq. (3.42), section 3.3.1.1. The detailed derivation can be found in the Appendices. Similar to the mismatch function in Eq. (4.2), the additive noise is assumed to be stationary and follows a Gaussian distribution. As a result, the noise parameters $\boldsymbol{\Phi}$ for Eq. (4.5) are $\boldsymbol{\mu}_{\mathtt{l}}$, $\boldsymbol{\mu}_{\mathtt{n}}$ and $\boldsymbol{\Sigma}_{\mathtt{n}}$. Using the continuous time approximation, a mismatch function for the dynamic parameters can be derived. For example, the delta cepstral coefficients can be approximated by

$$\Delta\boldsymbol{z}_t \approx \sum_{\delta=0}^{n}\frac{\partial\boldsymbol{z}_t^{\mathtt{s}}}{\partial\boldsymbol{x}_{t-\delta}^{\mathtt{s}}}\Delta\boldsymbol{x}_{t-\delta} \tag{4.6}$$

Apart from the continuous time approximation, it is also possible to use the extended VTS technique [38] to derive the mismatch functions for dynamic parameters. As an example, the delta cepstral coefficients can be expressed by the following equation using the extended VTS technique:

$$\Delta\boldsymbol{z}_t = \mathbf{D}\begin{bmatrix}\boldsymbol{z}_{t+w}^{\mathtt{s}}\\ \vdots\\ \boldsymbol{z}_{t-w}^{\mathtt{s}}\end{bmatrix} = \mathbf{D}\begin{bmatrix}\mathbf{g}(\boldsymbol{x}_{t+w}^{\mathtt{s}}, \ldots, \boldsymbol{x}_{t+w-n}^{\mathtt{s}}, \boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{n}_{t+w}^{\mathtt{s}})\\ \vdots\\ \mathbf{g}(\boldsymbol{x}_{t-w}^{\mathtt{s}}, \ldots, \boldsymbol{x}_{t-w-n}^{\mathtt{s}}, \boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{n}_{t-w}^{\mathtt{s}})\end{bmatrix}, \tag{4.7}$$

where $\mathbf{D}$ is the projection matrix which transforms a sequence of static cepstral coefficients to the corresponding delta coefficients. Note that, the above equation uses the following equations derived from Eq. (4.5):

$$\boldsymbol{z}_{t-i} = \mathbf{g}(\boldsymbol{x}_{t-i}^{\mathtt{s}}, \ldots, \boldsymbol{x}_{t-i-n}^{\mathtt{s}}, \boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{n}_{t-i}^{\mathtt{s}}) \quad i = -w, \cdots, +w \tag{4.8}$$

It is obvious from the equation that using extended VTS to derive the mismatch function for delta coefficients requires much more computation. Hence, in this work, the continuous time approximation is used to derive the mismatch functions for dynamic cepstral coefficients.

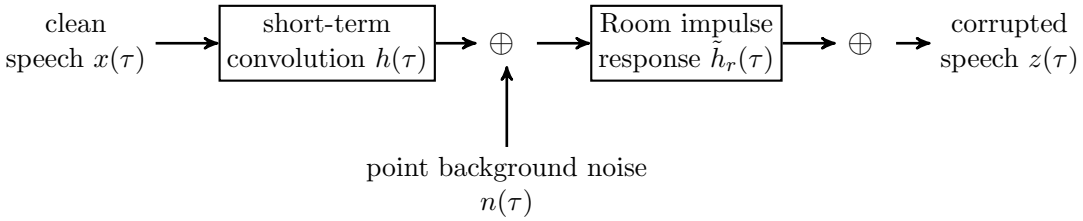


Figure 4.2: Point background noise source in a reverberant environment and the background noise source is located far away from the microphone.

4.1.3 Alternative Mismatch Functions

In this section, some alternative forms of mismatch functions are discussed. These mismatch functions can be also used in the reverberant noise model compensation framework which will be described in section 4.2.

4.1.3.1 Additive Noise From a Point Source

The time domain mismatch function in Eq. (4.4) was derived in section 3.3.1.1 when either the background noise is a diffuse noise source or the background noise is sufficiently close to the microphone. In section 3.3.1.1, an alternative scenario, in which a point additive noise source located far from microphones, was discussed. As a summary, the time domain mismatch function can be expressed as (see section 3.3.1.1 and Figure 4.2)

$$\begin{aligned} z(\tau) &= \tilde{h}_r(\tau) * (h(\tau) * x(\tau) + n(\tau)) \\ &= \tilde{h}_r(\tau) * y(\tau) \end{aligned} \tag{4.9}$$

where $y(\tau) = h(\tau) * x(\tau) + n(\tau)$ and $h(\tau)$ is a short-term convolution, and $\tilde{h}_r(\tau)$ is the RIR measured at the place where the background noise is emitted. Correspondingly, the cepstral domain mismatch function for this scenario is

$$\begin{aligned} \boldsymbol{z}_t^{\mathtt{s}} &= \mathbf{C} \log \left(\sum_{\delta=0}^{n} \exp(\mathbf{C}^{-1}(\boldsymbol{y}_{t-\delta}^{\mathtt{s}} + \tilde{\boldsymbol{\mu}}_{\mathtt{l}\delta})) \right) \\ &= \tilde{\mathbf{g}} \left(\boldsymbol{y}_t^{\mathtt{s}}, \ldots, \boldsymbol{y}_{t-n}^{\mathtt{s}}, \tilde{\boldsymbol{\mu}}_{\mathtt{l}} \right) \end{aligned} \tag{4.10}$$

where $\tilde{\boldsymbol{\mu}}_{\mathtt{l}}$ is another set of frame-level distortion terms, $\boldsymbol{y}_t^{\mathtt{s}}$ is the static MFCC of $y(\tau)$, and

$$\begin{aligned} \boldsymbol{y}_t^{\mathtt{s}} &= \mathbf{C} \log \left(\exp(\mathbf{C}^{-1}(\boldsymbol{x}_t^{\mathtt{s}} + \boldsymbol{\mu}_{\mathtt{h}})) + \exp(\mathbf{C}^{-1}\boldsymbol{n}_t^{\mathtt{s}}) \right) \\ &= \mathbf{f}(\boldsymbol{x}_t^{\mathtt{s}}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{n}_t^{\mathtt{s}}) \end{aligned} \tag{4.11}$$

Here, the noise model parameters $\tilde{\boldsymbol{\Phi}}$ are $\boldsymbol{\mu}_{\mathtt{l}}$, $\boldsymbol{\mu}_{\mathtt{n}}$, $\boldsymbol{\mu}_{\mathtt{h}}$ and $\boldsymbol{\Sigma}_{\mathtt{n}}$.

4.1.3.2 Domain of Noise Combination

The mismatch function in Eq. (4.5) combines the energy from the Mel filter bank in the magnitude domain (see the underlying assumptions in Appendix A). It is possible to combine the energy in other domains [62] or optimise the phase factor as in [44, 47, 154]. This is referred to as mismatch function optimisation. In fact, implementations of MFCC parameters differ in that the Mel filter can be performed either on the magnitude or on the power domain. Statistics about the speech and noise are usually only available in one domain. However, as pointed out in [62], scaling these parameters accordingly, assuming that the variation of the parameters within each of the Mel-spaced frequency bins is small, will yield the desired domain statistics. Under this assumption, scaling statistics to the other domain is possible. This only requires a simple change: the DCT matrix $\mathbf{C}$ is replaced by $\frac{1}{\gamma}\mathbf{C}$. Applying this substitution to Eq. (4.5), a more general form of mismatch function can be expressed as:

$$\boldsymbol{z}_t^{\mathbf{s}} = \frac{1}{\gamma}\mathbf{C}\log\left(\sum_{\delta=0}^{n}\exp\left(\gamma\mathbf{C}^{-1}(\boldsymbol{x}_{t-\delta}^{\mathbf{s}}+\boldsymbol{\mu}_{1\delta})\right)+\exp\left(\gamma\mathbf{C}^{-1}\boldsymbol{n}_t^{\mathbf{s}}\right)\right) \tag{4.12}$$

where $\gamma = 1$ represents that noise and speech are linearly additive in the magnitude domain, and $\gamma = 2$ the power domain. Setting γ to a large value is to approximate the MAX assumption [178] used in some masking schemes[204, 238].

4.2 Reverberant Model Compensation

From the mismatch functions in Eqs. (4.5) and (4.10), it can be observed that a reverberant and additive noise corrupted static speech frame is a function of a window of $n+1$ static clean speech frames $\boldsymbol{x}_t^{\mathbf{s}},\ldots,\boldsymbol{x}_{t-n}^{\mathbf{s}}$ and the additive noise $\boldsymbol{n}_t^{\mathbf{s}}$ (or $\boldsymbol{n}_t^{\mathbf{s}},\ldots,\boldsymbol{n}_{t-n}^{\mathbf{s}}$ to yield $\boldsymbol{y}_t^{\mathbf{s}},\ldots,\boldsymbol{y}_{t-n}^{\mathbf{s}}$). Therefore, additional model statistics are needed to model this dependency.

4.2.1 Model Statistics

Figure 4.3 shows the generating process of reverberant observations (ignoring the dynamic parameters) according to the mismatch function in Eq. (4.5). For simplicity, a scenario in which $n = 1$ is shown in the figure. Inference on this dynamic Bayesian network (DBN) is not practical as the number of states and components affecting the current state grows exponentially. Approximations to this form are possible. For example, in [213] the Viterbi decoding algorithm is modified where model parameters are adapted based on the current best partial path. The model adaptation is done at each frame, which results in a large amount of computation. Alternatively, the model parameters can be adapted prior to recognition,

based on the estimated preceding states. This can be done by either using the intra-phoneme preceding states or the context of biphone [198] or triphone [103] models. However, it is difficult to infer a long preceding state sequence in this way, especially when tied-state cross word triphone models are used.

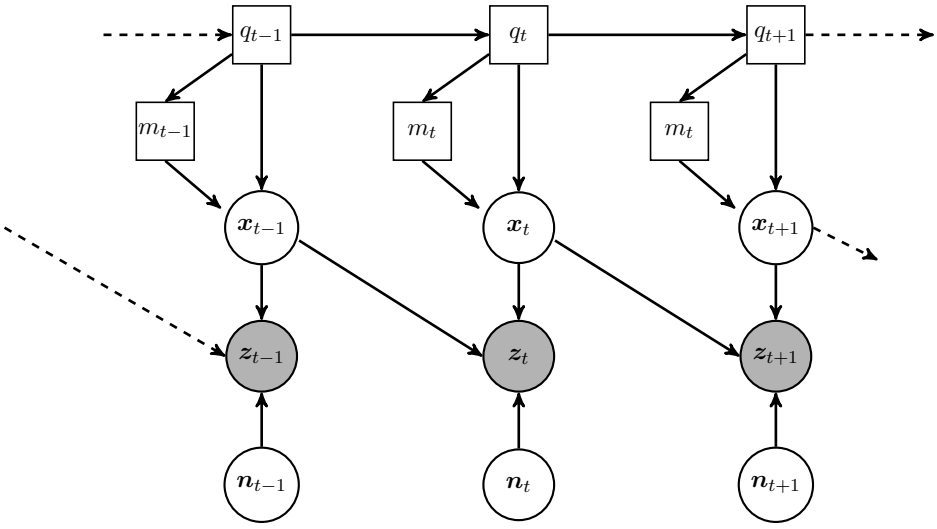


Figure 4.3: Reverberant dynamic Bayesian network, ignoring the dynamic cepstral coefficients. q_t and m_t denote the state and component at time t,$n = 1$.

In this work, another form of approximation is used. The DBN is shown in Figure 4.4. In this approximation, rather than an explicit dependence on the previous observation or states, the observation vector $\boldsymbol{z}_t$ is assumed to depend on an extended observation vector $\overline{\boldsymbol{x}}_t$. In this way, the standard Viterbi algorithm can be used for inference. This approximation results in two forms of smoothing. First statistics are smoothed over all possible previous states. This effect is moderated for the context dependent models as the left context automatically limits the range of possible states. The second impact is the smoothing over components for the previous state. It is worth noting that this is exactly the same form of approximation that is used in deriving the standard dynamic parameters.

It is also important to decide which form of the probability distribution, $p(\overline{\boldsymbol{x}}_t|m_t = m)$, to use. To ensure that if there is no reverberant noise, the compensated model becomes the original model, the following form is used:

$$\overline{\boldsymbol{x}}_t = \begin{bmatrix} \boldsymbol{x}_t^{\mathsf{s}} \\ \Delta \boldsymbol{x}_t \\ \Delta^2 \boldsymbol{x}_t \\ \tilde{\boldsymbol{x}}_t \end{bmatrix} = \mathbf{W} \begin{bmatrix} \boldsymbol{x}_{t+w}^{\mathsf{s}} \\ \dots \\ \boldsymbol{x}_{t-n-w}^{\mathsf{s}} \end{bmatrix} \sim \mathcal{N}(\overline{\boldsymbol{\mu}}_{\mathbf{x}}^{(m)}, \overline{\boldsymbol{\Sigma}}_{\mathbf{x}}^{(m)}) \tag{4.13}$$

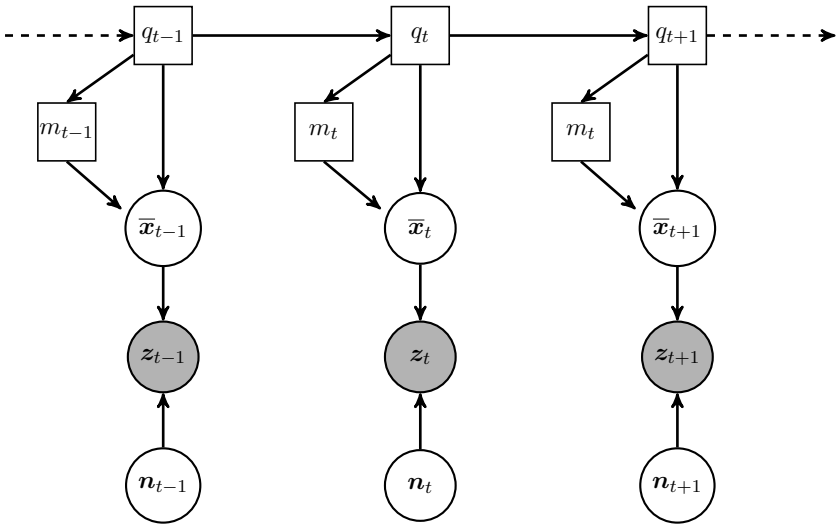


Figure 4.4: Approximate reverberant environment dynamic Bayesian network. $\overline{\boldsymbol{x}}_t$ is an extended observation vector which denotes the adjacent observation vectors $[\boldsymbol{x}_{t-w-n}^{\mathsf{T}}, \cdots, \boldsymbol{x}_{t+w}^{\mathsf{T}}]^{\mathsf{T}}$ when the t-th state and Gaussian component are q and m respectively.

where w is the window size to calculate the dynamic parameters, $\tilde{\boldsymbol{x}}_t$ can be any vector, provided $\mathbf{W}$ is square and invertible, and $\overline{\boldsymbol{\Sigma}}_{\mathbf{x}}^{(m)}$ can be approximated by a diagonal covariance matrix. In this work, $\mathbf{W}$ is produced by

$$\mathbf{W} = \overline{\mathbf{W}} \otimes \mathbf{I}_d = \begin{pmatrix} \overline{w}_{11}\mathbf{I}_d & \cdots & \overline{w}_{1d}\mathbf{I}_d \\ \vdots & \cdots & \vdots \\ \overline{w}_{d1}\mathbf{I}_d & \cdots & \overline{w}_{dd}\mathbf{I}_d \end{pmatrix} \tag{4.14}$$

where $\otimes$ denotes the Kronecker product, d is the dimension of $\boldsymbol{x}_t^{\mathbf{s}}$, $\overline{w}_{ij}$ is the (i,j)-th element of $\overline{\mathbf{W}}$; $\overline{\mathbf{W}}$ uses the following form:

$$\begin{array}{c|cccccc} & \tau = w & \cdots & \tau = 0 & \cdots & \tau = -w & \cdots \quad \tau = -(n+w) \\ i = 0 & 0 & \cdots & 1 & \cdots & 0 & \\ i = 1 & & & \boldsymbol{\alpha}^{(1)} & & & \mathbf{0} \\ i = 2 & & & \boldsymbol{\alpha}^{(2)} & & & \\ \hline i = 3 & & & & & & \\ \vdots & & & \left(\cos\left(\frac{i\pi}{L}(\tau + 0.5)\right)\right)_{i,\tau} & & & \\ i = L-1 & & & & & & \end{array}$$

where $L = n + 2w + 1$, $\boldsymbol{\alpha}^{(1)}$ and $\boldsymbol{\alpha}^{(2)}$ are the vectors which are used to compute delta and

delta-delta coefficients. This is equivalent to the following equations in a scalar form:

$$\begin{aligned}
\overline{x}_{t,j} &= x_{t,j} \quad \forall j \\
\overline{x}_{t,d\times i+j+1} &= \boldsymbol{\alpha}^{(1)\mathsf{T}} \begin{pmatrix} x^{\mathtt{s}}_{t+w,j} \\ \vdots \\ x^{\mathtt{s}}_{t-w,j} \end{pmatrix} \quad \forall j \text{ and } i = 1, \cdots, 2 \\
\overline{x}_{t,d\times i+j+1} &= \sum_{\tau=-n-w}^{w} \cos\left(\frac{i\pi}{L}(\tau+0.5)\right) x_{t+\tau,j} \quad \forall j \text{ and } i = 3, \cdots, d\times 2w+n+1
\end{aligned}$$

where $x^{\mathtt{s}}_{t+i,j}$ is the j-th element of $\boldsymbol{x}^{\mathtt{s}}_{t+i}$, $\overline{x}_{t,d\times i+j+1}$ is the $d\times i+j+1$ -th element of $\overline{\boldsymbol{x}}_t$.

Using this form of $\mathbf{W}$ matrix ensures that the extended model statistics always agree with the standard model statistics, and the compensated acoustic model falls back to the VTS-compensated model when there is no reverberation noise.

Using this representation, it is simple to derive the clean speech statistics. For example, the mean and covariance of spliced frames $\boldsymbol{x}^{\mathtt{s}}_{\mathtt{e}} = [\boldsymbol{x}^{\mathtt{s}\mathsf{T}}_t, \cdots, \boldsymbol{x}^{\mathtt{s}\mathsf{T}}_{t-n}]^{\mathsf{T}}$, [1] $\boldsymbol{\mu}^{(m)}_{\mathtt{sxe}}$ and $\boldsymbol{\Sigma}^{(m)}_{\mathtt{sxe}}$, can be found using

$$\boldsymbol{\mu}^{(m)}_{\mathtt{sxe}} = \mathbf{P}_{\mathtt{s}}\overline{\boldsymbol{\mu}}^{(m)}_{\mathtt{x}} \; ; \quad \boldsymbol{\Sigma}^{(m)}_{\mathtt{sxe}} = \mathbf{P}_{\mathtt{s}}\overline{\boldsymbol{\Sigma}}^{(m)}_{\mathtt{x}}\mathbf{P}^{\mathsf{T}}_{\mathtt{s}} \tag{4.15}$$

where $\mathbf{P}_{\mathtt{s}}$ is the matrix that projects $\overline{\boldsymbol{x}}_t$ to $\boldsymbol{x}_{\mathtt{e}}$. Since the delta of $\boldsymbol{x}_{\mathtt{e}}$, $\Delta\boldsymbol{x}_{\mathtt{e}}$, is also a linear combination of $\overline{\boldsymbol{x}}_t$, the mean and covariance of $\Delta\boldsymbol{x}_{\mathtt{e}}$, $\boldsymbol{\mu}^{(m)}_{\Delta\mathtt{xe}}$ and $\boldsymbol{\Sigma}^{(m)}_{\Delta\mathtt{xe}}$ can be obtained in a similar way, as shown in the following equations:

$$\boldsymbol{\mu}^{(m)}_{\Delta\mathtt{xe}} = \mathbf{P}_{\Delta}\overline{\boldsymbol{\mu}}^{(m)}_{\mathtt{x}} \; ; \quad \boldsymbol{\Sigma}^{(m)}_{\Delta\mathtt{xe}} = \mathbf{P}_{\Delta}\overline{\boldsymbol{\Sigma}}^{(m)}_{\mathtt{x}}\mathbf{P}^{\mathsf{T}}_{\Delta} \, , \tag{4.16}$$

where $\mathbf{P}_{\Delta}$ is the matrix which is used to project the extended vector $\overline{\boldsymbol{x}}_t$ to $\Delta\boldsymbol{x}_t$. Similar methods can be used to derive the model parameters of delta-delta cesptral coefficients.

It is clear from Eqs. (4.15) and (4.16) that the model statistics are $\mathcal{M} = \{\overline{\boldsymbol{\mu}}^{(m)}_{\mathtt{x}}, \overline{\boldsymbol{\Sigma}}^{(m)}_{\mathtt{x}}\}$. The simplest way to estimate $\mathcal{M}$ is to use clean speech. However, this requires access to clean data recorded in the target task, which is not always available in practice. An alternative method is to adaptively estimate $\mathcal{M}$ from multi-condition data. Adaptive training of $\mathcal{M}$ will be discussed in section 4.3.

The above expressions describe the derivation of the "clean" statistics required by the mismatch function in Eq. (4.5). Given the noise model parameters $\boldsymbol{\Phi}_{\mathtt{n}} = (\boldsymbol{\mu}_{\mathtt{n}}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\Sigma}_{\mathtt{n}})$, it is possible to derive the "noisy" statistics required by the mismatch function in Eq. (4.10). To

[1] Here, $\mathtt{e}$ is used to denote $\boldsymbol{x}^{\mathtt{s}}_{\mathtt{e}}$ is an extended vector, which is obtained by cancatenating a window of static features, $\boldsymbol{x}^{\mathtt{s}}_t, \cdots, \boldsymbol{x}^{\mathtt{s}}_{t-n}$. Note that $\boldsymbol{x}^{\mathtt{s}}_{\mathtt{e}}$ is part of the vector projected by the $\mathbf{W}$ matrix in Eq. (4.13). This is because the extended vector $\overline{\boldsymbol{x}}_t$ is a projection of both static and dynamic cepstral coefficients from time t to $t-n$. To be able to derive the dynamic parameters, the extended vector needs to contain extra static frames, i.e., from time $t+w$ to $t-n-w$.

avoid computing a large number of Jacobian matrices, a linear approximation is used. For example, the mean of $\boldsymbol{y}^{\mathtt{s}}_{t-\delta}$, $\boldsymbol{\mu}^{(m)}_{\mathtt{sy}\delta}$, is given by

$$\begin{aligned}\boldsymbol{\mu}^{(m)}_{\mathtt{sy}\delta} &= \mathbf{f}(\boldsymbol{\mu}^{(m)}_{\mathtt{sx}}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\mu}_{\mathtt{n}}) \\ &\approx \mathbf{f}(\boldsymbol{\mu}^{(m)}_{\mathtt{sx}}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\mu}_{\mathtt{n}}) + \mathbf{J}^{(m)}_{\mathtt{x}}(\boldsymbol{\mu}^{(m)}_{\mathtt{sx}\delta} - \boldsymbol{\mu}^{(m)}_{\mathtt{sx}})\end{aligned} \tag{4.17}$$

where

$$\mathbf{J}^{(m)}_{\mathtt{x}} = \frac{\partial \mathbf{f}(\boldsymbol{x}^{\mathtt{s}}, \mathbf{h}, \boldsymbol{n}^{\mathtt{s}})}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{\mu}^{(m)}_{\mathtt{sx}}, \boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\mu}_{\mathtt{n}}}$$

and $\boldsymbol{\mu}^{(m)}_{\mathtt{sx}\delta}$ is the mean of $\boldsymbol{x}^{\mathtt{s}}_{t-\delta}$, conditioning on the component m. $\boldsymbol{\mu}^{(m)}_{\mathtt{sx}\delta}$ is a subvector of $\boldsymbol{\mu}^{(m)}_{\mathtt{sxe}}$, which can be obtained by projecting $\overline{\boldsymbol{\mu}}^{(m)}_{\mathtt{x}}$ using Eq. (4.15). Once $\boldsymbol{\mu}^{(m)}_{\mathtt{sy}\delta}(\delta = -w, \ldots n+w)$ are known, the noisy delta statistics, $\boldsymbol{\mu}^{(m)}_{\Delta\mathtt{ye}}$, can be obtained in the same way as $\boldsymbol{\mu}^{(m)}_{\Delta\mathtt{xe}}$. Hence, $\mathcal{M} = \{\overline{\boldsymbol{\mu}}^{(m)}_{\mathtt{y}}, \overline{\boldsymbol{\Sigma}}^{(m)}_{\mathtt{y}}\}$ are used as the model statistics.

4.2.2 Model Compensation

In the previous section, approximate dynamic Bayesian networks were introduced which require extended model statistics to take account of the effect of previous clean speech frames in the reverberant mismatch function. Since the mismatch functions are highly nonlinear, additional approximations are needed to derive a model compensation form. In this thesis, the vector Taylor series (VTS) expansion[5], discussed in section 3.3, was extended to handle reverberant noise.

4.2.2.1 Reverberant VTS and Reverberant VTS-Joint Compensation

Using the mismatch functions in Eqs. (4.10 and 4.5), which take reverberant noise into account, it is possible to extend the use of VTS to handle reverberant noise as well. Given the extended model statistics $\mathcal{M} = \{\overline{\boldsymbol{\mu}}^{(m)}_{\mathtt{x}}, \overline{\boldsymbol{\Sigma}}^{(m)}_{\mathtt{x}}\}$ (or $\mathcal{M} = \{\overline{\boldsymbol{\mu}}^{(m)}_{\mathtt{y}}, \overline{\boldsymbol{\Sigma}}^{(m)}_{\mathtt{y}}\}$ for Eq. (4.10)) described in the previous section, the mismatch function in Eq. (4.5) can be expanded around the model parameters $\mathcal{M}$ and the current noise parameters $\boldsymbol{\Phi}$, i.e.,

$$\begin{aligned}\boldsymbol{z}^{\mathtt{s}}_t | m &= \mathbf{g}(\boldsymbol{x}^{\mathtt{s}}_{\mathtt{e}}, \boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{n}^{\mathtt{s}}_t) \qquad \text{where } \boldsymbol{x}^{\mathtt{s}}_{\mathtt{e}} \sim \mathcal{N}(\boldsymbol{\mu}^{(m)}_{\mathtt{sxe}}, \boldsymbol{\Sigma}^{(m)}_{\mathtt{sxe}}) \\ &\approx \mathbf{g}(\boldsymbol{\mu}^{(m)}_{\mathtt{sxe}}, \boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{\mu}_{\mathtt{n}}) + [\mathbf{J}^{(m)}_{\mathtt{xe}}, \mathbf{J}^{(m)}_{\mathtt{ne}}] \begin{bmatrix} \boldsymbol{x}^{\mathtt{s}}_{\mathtt{e}} - \boldsymbol{\mu}^{(m)}_{\mathtt{sxe}} \\ \boldsymbol{n}^{\mathtt{s}}_t - \boldsymbol{\mu}_{\mathtt{n}} \end{bmatrix}\end{aligned} \tag{4.18}$$

where

$$\mathbf{J}^{(m)}_{\mathtt{xe}} = [\mathbf{J}^{(m)}_{\mathtt{x}0}, \cdots, \mathbf{J}^{(m)}_{\mathtt{x}n}]\,; \quad \mathbf{J}^{(m)}_{\mathtt{ne}} = \mathbf{I} - \sum_{\delta=0}^{n} \mathbf{J}^{(m)}_{\mathtt{x}\delta} \tag{4.19}$$

and

$$\mathbf{J}_{\mathtt{x}i}^{(m)} = \frac{\partial \mathbf{g}}{\partial \boldsymbol{x}_{t-i}}\Big|_{\boldsymbol{\mu}_{\mathtt{xe}}^{(m)},\boldsymbol{\mu}_{\mathtt{l}},\boldsymbol{\mu}_{\mathtt{n}}} = \mathbf{C}\mathbf{F}_i^{(m)}\mathbf{C}^{\mathsf{T}}, \qquad \mathbf{J}_{\mathtt{ne}}^{(m)} = \mathbf{C}\mathbf{F}_{\mathtt{n}}^{(m)}\mathbf{C}^{\mathsf{T}} \tag{4.20}$$

$\mathbf{F}_i^{(m)}$ and $\mathbf{F}_{\mathtt{n}}^{(m)}$ are two diagonal matrices whose q-th diagonal items, $f_{iq}^{(m)}$ and $f_{\mathtt{n},q}^{(m)}$, are

$$f_{iq}^{(m)} = \frac{\exp(\boldsymbol{d}_q^{\mathsf{T}}(\boldsymbol{\mu}_{\mathtt{x}i}^{(m)} + \boldsymbol{\mu}_{\mathtt{l}i}))}{\sum_{j=0}^{n}\exp(\boldsymbol{d}_q^{\mathsf{T}}(\boldsymbol{\mu}_{\mathtt{x}j}^{(m)} + \boldsymbol{\mu}_{\mathtt{l}j})) + \exp(\boldsymbol{d}_q^{\mathsf{T}}\boldsymbol{\mu}_{\mathtt{n}})} \tag{4.21}$$

$$f_{\mathtt{n},q}^{(m)} = \frac{\exp(\boldsymbol{d}_q^{\mathsf{T}}\boldsymbol{\mu}_{\mathtt{n}})}{\sum_{j=0}^{n}\exp(\boldsymbol{d}_q^{\mathsf{T}}(\boldsymbol{\mu}_{\mathtt{x}j}^{(m)} + \boldsymbol{\mu}_{\mathtt{l}j})) + \exp(\boldsymbol{d}_q^{\mathsf{T}}\boldsymbol{\mu}_{\mathtt{n}})} \tag{4.22}$$

where $\boldsymbol{d}_q^{\mathsf{T}}$ are the q-th row vector in the matrix $\mathbf{C}^{-1}$. The static model parameters are therefore compensated by

$$\boldsymbol{\mu}_{\mathtt{sz}}^{(m)} = \mathcal{E}\{\boldsymbol{z}_t^{\mathtt{s}}|m\} = \mathbf{g}(\boldsymbol{\mu}_{\mathtt{sxe}}^{(m)}, \boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{\mu}_{\mathtt{n}}) \tag{4.23}$$

For the dynamic model parameters, the continuous time approximation[86] can be applied

$$\boldsymbol{\mu}_{\Delta\mathtt{z}}^{(m)} \approx \mathcal{E}\left(\frac{\partial \boldsymbol{z}_t}{\partial t}\Big|m\right) = \mathcal{E}\left(\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{x}_{\mathtt{e}}}\frac{\partial \boldsymbol{x}_{\mathtt{e}}}{\partial t}\Big|m\right) = \mathbf{J}_{\mathtt{xe}}^{(m)}\boldsymbol{\mu}_{\Delta\mathtt{xe}}^{(m)} \tag{4.24}$$

The delta-delta parameters can be compensated in a similar fashion. This compensation form will be referred to as Reverberant VTS-Joint (RVTSJ) since it allows the joint estimation of additive and reverberant noise, which will be demonstrated later.

A similar approximation can be also applied for the mismatch function in Eq. (4.10). Performing an expansion of the function $\tilde{\mathbf{g}}()$ around $\boldsymbol{\mu}_{\mathtt{sye}}^{(m)}, \tilde{\boldsymbol{\mu}}_{\mathtt{l}}$ yields

$$\boldsymbol{z}_t^{\mathtt{s}}|m = \tilde{\mathbf{g}}(\boldsymbol{y}_{\mathtt{e}}^{\mathtt{s}}, \tilde{\boldsymbol{\mu}}_{\mathtt{l}}) \approx \tilde{\mathbf{g}}(\boldsymbol{\mu}_{\mathtt{sye}}^{(m)}, \tilde{\boldsymbol{\mu}}_{\mathtt{l}}) + \mathbf{J}_{\mathtt{ye}}^{(m)}(\boldsymbol{y}_{\mathtt{e}}^{\mathtt{s}} - \boldsymbol{\mu}_{\mathtt{sye}}^{(m)}) \tag{4.25}$$

where $\boldsymbol{y}_{\mathtt{e}}^{\mathtt{s}}$ consists of the stacked noisy frames $\boldsymbol{y}_t^{\mathtt{s}}, \ldots, \boldsymbol{y}_{t-n}^{\mathtt{s}}$ and

$$\mathbf{J}_{\mathtt{ye}}^{(m)} = [\mathbf{J}_{\mathtt{y}0}^{(m)}, \cdots, \mathbf{J}_{\mathtt{y}n}^{(m)}]\ ; \quad \mathbf{J}_{\mathtt{y}\delta}^{(m)} = \frac{\partial \tilde{\mathbf{g}}}{\partial \boldsymbol{y}_{t-i}^{\mathtt{s}}}\Big|_{\boldsymbol{\mu}_{\mathtt{sye}}^{(m)},\tilde{\boldsymbol{\mu}}_{\mathtt{l}}} \tag{4.26}$$

With this expansion, the static mean is compensated using

$$\boldsymbol{\mu}_{\mathtt{sz}}^{(m)} = \tilde{\mathbf{g}}(\boldsymbol{\mu}_{\mathtt{sye}}^{(m)}, \tilde{\boldsymbol{\mu}}_{\mathtt{l}}) \tag{4.27}$$

Based on the same continuous time approximation, the delta parameters are compensated using the following equation:

$$\boldsymbol{\mu}_{\Delta\mathtt{z}}^{(m)} \approx \mathcal{E}\left(\frac{\partial \boldsymbol{z}_t}{\partial t}|m\right) = \mathcal{E}\left(\frac{\partial \boldsymbol{z}_t}{\partial \boldsymbol{y}_{\mathtt{e}}}\frac{\partial \boldsymbol{y}_{\mathtt{e}}}{\partial t}|m\right) = \mathbf{J}_{\mathtt{ye}}^{(m)}\boldsymbol{\mu}_{\Delta\mathtt{ye}}^{(m)} \tag{4.28}$$

The delta-delta parameters are compensated in the same way. This form of compensation will be referred to as RVTS.

It is possible to compensate the variances as well. However, in initial investigations, it was found that variance compensation is quite sensitive to the errors in the supervision hypothesis and a reliable compensation is hard to obtain. Thus, in this work the variance compensation is done using standard VTS, i.e., $\mathbf{\Sigma}_{\mathtt{z}}^{(m)} = \mathbf{\Sigma}_{\mathtt{y}}^{(m)}$, where the VTS noise model parameters $\mathbf{\Phi}_{\mathtt{n}}$ are estimated via standard VTS noise estimation. Variance compensation for reverberant environments will be left to future work in this thesis.

4.2.3 Noise Estimation

In the previous section, two model compensation forms, RVTS and RVTSJ, were described. The noise parameters, $\mathbf{\Phi}$ for RVTSJ or $\tilde{\mathbf{\Phi}}$ for RVTS, now need to be determined. Though there exists a simple method to determine the frame-level distortion terms [103] based on the known reverberation time T_{60}, it is preferable to use the ML estimate of noise parameters, as it yields a consistent fit with the reverberant data. ML estimation of noise parameters also helps to address the incorrectness of the mismatch function.

Due to the nature of the RVTS mismatch function in Eq. (4.10), it is required to convert the clean statistics to noisy statistics using Eq. (4.17). The joint estimation of reverberant and additive noise parameters is complicated: additional assumptions (e.g., $\mathbf{J}_{\mathtt{n}}^{(m)}$ is independent of $\boldsymbol{\mu}_{\mathtt{h}}, \boldsymbol{\mu}_{\mathtt{n}}$) are needed to avoid expensive computation of a large number of Jacobian matrices. In [73], a sequential ML estimation of noise parameters for the RVTS was presented, where the additive and convolutional noise parameters, $\boldsymbol{\mu}_{\mathtt{n}}, \boldsymbol{\mu}_{\mathtt{h}}$ and $\mathbf{\Sigma}_{\mathtt{n}}$, were first estimated using standard VTS noise estimation, then the noisy statistics $\boldsymbol{\mu}_{\mathtt{ye}}^{(m)}$ were obtained, followed by reverberant noise mean $\tilde{\boldsymbol{\mu}}_{\mathtt{l}}$ estimation. In this thesis, the estimation of reverberant noise was extended for RVTSJ mismatch function, which allows joint estimation of reverberant and additive noise more easily. As both estimation algorithms share many of the same attributes, the following presentation will focus on noise estimation for RVTSJ.

The estimation of the reverberant and additive noise means is done using the EM framework, similar to the convolutional and additive noise mean estimation using EM. The following auxiliary function is maximised:

$$\mathcal{Q}(\hat{\boldsymbol{\mu}}_{\mathtt{l}}, \hat{\boldsymbol{\mu}}_{\mathtt{n}}) = \sum_{t,m} \gamma_t^{(m)} \log p(\boldsymbol{z}_t; \boldsymbol{\mu}_{\mathtt{z}}^{(m)}, \mathbf{\Sigma}_{\mathtt{z}}^{(m)}) + \mathcal{R}(\hat{\boldsymbol{\mu}}_{\mathtt{l}}, \hat{\boldsymbol{\mu}}_{\mathtt{n}}) \tag{4.29}$$

where $\hat{\boldsymbol{\mu}}_{\mathtt{l}}, \hat{\boldsymbol{\mu}}_{\mathtt{n}}$ are the current noise mean estimates, $\boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{\mu}_{\mathtt{n}}$ are corresponding old estimates; $\gamma_t^{(m)}$ is the posterior of component m at time t, given the current hypothesis and current

noise estimates $\boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{\mu}_{\mathtt{n}}$, $\mathcal{R}(\hat{\boldsymbol{\mu}}_{\mathtt{l}}, \hat{\boldsymbol{\mu}}_{\mathtt{n}})$ is a regularisation term to improve the stability of noise estimation. In this work, the following form of regularisation was used:

$$\mathcal{R}(\hat{\boldsymbol{\mu}}_{\mathtt{l}}, \hat{\boldsymbol{\mu}}_{\mathtt{n}}) = \alpha \left((\hat{\boldsymbol{\mu}}_{\mathtt{l}} - \boldsymbol{\mu}_{\mathtt{l}})^{\mathsf{T}} (\hat{\boldsymbol{\mu}}_{\mathtt{l}} - \boldsymbol{\mu}_{\mathtt{l}}) + (\hat{\boldsymbol{\mu}}_{\mathtt{n}} - \boldsymbol{\mu}_{\mathtt{n}})^{\mathsf{T}} (\hat{\boldsymbol{\mu}}_{\mathtt{n}} - \boldsymbol{\mu}_{\mathtt{n}}) \right)$$

where α is a tuning parameter, used to improve the stability of noise estimation. The purpose of this regularisation is to restrict step size of each EM update.

Performing a first-order expansion of $\boldsymbol{\mu}_{\mathtt{z}}^{(m)}$ using the current estimates, $\boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{\mu}_{\mathtt{n}}$ [1], yields:

$$\begin{bmatrix} \hat{\boldsymbol{\mu}}_{\mathtt{sz}}^{(m)} \\ \hat{\boldsymbol{\mu}}_{\Delta\mathtt{z}}^{(m)} \end{bmatrix} \approx \begin{bmatrix} \boldsymbol{\mu}_{\mathtt{sz}}^{(m)} \\ \boldsymbol{\mu}_{\Delta\mathtt{z}}^{(m)} \end{bmatrix} + \begin{bmatrix} \mathbf{J}_{\mathtt{le}}^{(m)} & \mathbf{J}_{\mathtt{ne}}^{(m)} \\ \mathbf{J}_{\Delta\mathtt{le}}^{(m)} & \mathbf{J}_{\Delta\mathtt{ne}}^{(m)} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\mu}}_{\mathtt{l}} - \boldsymbol{\mu}_{\mathtt{l}} \\ \hat{\boldsymbol{\mu}}_{\mathtt{n}} - \boldsymbol{\mu}_{\mathtt{n}} \end{bmatrix} \tag{4.30}$$

where

$$\mathbf{J}_{\mathtt{le}}^{(m)} = [\mathbf{J}_{\mathtt{l}0}^{(m)}, \cdots, \mathbf{J}_{\mathtt{l}n}^{(m)}] \quad \mathbf{J}_{\mathtt{l}\delta}^{(m)} = \frac{\partial \mathbf{g}}{\partial \boldsymbol{\mu}_{\mathtt{l}\delta}} \Big|_{\boldsymbol{\mu}_{\mathtt{xe}}^{(m)}, \boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{\mu}_{\mathtt{n}}}$$

and

$$\begin{aligned}
\mathbf{J}_{\Delta\mathtt{le}}^{(m)} &= [\mathbf{J}_{\Delta\mathtt{l},0}^{(m)}, \cdots, \mathbf{J}_{\Delta\mathtt{l},n}^{(m)}] \\
\mathbf{J}_{\Delta\mathtt{ne}}^{(m)} &= \frac{\partial \boldsymbol{\mu}_{\Delta\mathtt{z}}^{(m)}}{\partial \boldsymbol{\mu}_{\mathtt{n}}} = \mathbf{C}\mathrm{diag}\left(\sum_{i=0}^{n} \boldsymbol{\Lambda}_i^{(m)} \mathbf{C}^{-1} \boldsymbol{\mu}_{\Delta\mathtt{x}i}^{(m)} \right) \mathbf{C}^{-1} \\
\mathbf{J}_{\Delta\mathtt{l},\delta}^{(m)} &= \frac{\partial \boldsymbol{\mu}_{\Delta\mathtt{z}}^{(m)}}{\partial \boldsymbol{\mu}_{\mathtt{l}\delta}} = \mathbf{C}\mathrm{diag}\left(\sum_{i=0}^{n} \boldsymbol{\Xi}_{i,\delta}^{(m)} \mathbf{C}^{-1} \boldsymbol{\mu}_{\Delta\mathtt{x}i}^{(m)} \right) \mathbf{C}^{-1}
\end{aligned}$$

$\boldsymbol{\Lambda}_i^{(m)}$ and $\boldsymbol{\Xi}_{i,\delta}^{(m)}$ are diagonal matrices with the size $d_L \times d_L$ and d_L is the number of filter banks. The q-th diagonal elements of $\boldsymbol{\Lambda}_i^{(m)}$ and $\boldsymbol{\Xi}_{i,\delta}^{(m)}$ are given by:

$$(\boldsymbol{\Lambda}_i^{(m)})_{qq} = -f_{iq}^{(m)} \cdot f_{\mathtt{n},q}^{(m)} \tag{4.31}$$

$$(\boldsymbol{\Xi}_{i,\delta}^{(m)})_{qq} = f_{iq}^{(m)} (\mathbf{1}[i = \delta] - f_{\delta q}^{(m)}) \tag{4.32}$$

Here, $\mathbf{1}[\cdot]$ is an indicator function, $f_{iq}^{(m)}$ and $f_{\mathtt{n},q}^{(m)}$ are defined in Eqs (4.21 and 4.22) respectively. Note $\mathbf{J}_{\Delta\mathtt{le}}^{(m)}$ and $\mathbf{J}_{\Delta\mathtt{ne}}^{(m)}$ are nonzero matrices due to the fact $\boldsymbol{\mu}_{\Delta\mathtt{z}}^{(m)}$ is a function of $\mathbf{J}_{\mathtt{le}}^{(m)}$ and $\mathbf{J}_{\mathtt{ne}}^{(m)}$, which in turn depend on the noise parameters.

Differentiating the auxiliary function and equating to zero gives the following update:

$$\begin{aligned}
\begin{bmatrix} \hat{\boldsymbol{\mu}}_{\mathtt{l}} \\ \hat{\boldsymbol{\mu}}_{\mathtt{n}} \end{bmatrix} = &\left(\sum_{t,m} \gamma_t^{(m)} \mathbf{J}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{z}}^{(m)-1} \mathbf{J}^{(m)} + \alpha \mathbf{I} \right)^{-1} \times \\
&\left(\sum_{t,m} \gamma_t^{(m)} \mathbf{J}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{z}}^{(m)-1} \left(\boldsymbol{\mu}_{\mathtt{z}}^{(m)} - \mathbf{J}^{(m)} \begin{bmatrix} \boldsymbol{\mu}_{\mathtt{l}} \\ \boldsymbol{\mu}_{\mathtt{n}} \end{bmatrix} \right) \right)
\end{aligned} \tag{4.33}$$

[1] To simplify notations, only the delta parameters are shown. Including delta-delta parameters is similar.

where

$$\mathbf{J}^{(m)} = \begin{bmatrix} \mathbf{J}_{\mathtt{le}}^{(m)} & \mathbf{J}_{\mathtt{ne}}^{(m)} \\ \mathbf{J}_{\Delta\mathtt{le}}^{(m)} & \mathbf{J}_{\Delta\mathtt{ne}}^{(m)} \end{bmatrix}.$$

Note in the noise estimation for RVTS, a similar expression was used. However, for RVTS, only $\tilde{\boldsymbol{\mu}}_{\mathtt{l}}$ was updated while $\boldsymbol{\mu}_{\mathtt{n}}$ was fixed at the value estimated in VTS. Due to the VTS approximation, it is necessary to check the auxiliary value to ensure it does not decrease, otherwise a back-off strategy similar as the one described in [158] is used. When a new noise estimate is obtained, it is possible to re-expand the mismatch function and update the noise estimate accordingly. This process can be repeated several times until the auxiliary function does not increase.

Since the auxiliary function is highly nonlinear, it is crucial to have a good initialisation. The initialisation scheme in this work uses an initial (rough) estimate of T_{60} value, similar to the one used in [103]. The initialisation scheme is slightly modified so that the initial compensated mean vectors are approximately the same as the VTS compensated means. For RVTS, the initial frame-level distortion terms are given by

$$\tilde{\boldsymbol{\mu}}_{\mathtt{l}\delta} = \mathbf{C}[\,\delta\eta + \rho \;\ldots\; \delta\eta + \rho]^{\mathsf{T}} \tag{4.34}$$

where

$$\eta = -3\log(10)\frac{\Delta}{T_{60}}\;; \quad \rho = -\log\left(\frac{1 - e^{(n+1)\eta}}{1 - e^{\eta}}\right) \tag{4.35}$$

and Δ is the shift of analysis window (10ms in this work). Note here the cepstral coefficients are extracted from the magnitude spectrum rather than power spectrum; therefore Eq. (4.35) is slightly modified from the one in [103]: since the T_{60} measures the time needed to attenuate 60dB in the power domain, a $3\log(10)$ coefficient is used in calculating the frame shift rate η in the magnitude domain. For RVTSJ, $\boldsymbol{\mu}_{\mathtt{n}}$ is initialised as the additive noise mean estimated by standard VTS noise estimation, while for $\boldsymbol{\mu}_{\mathtt{l}\delta}$

$$\boldsymbol{\mu}_{\mathtt{l}\delta} = \boldsymbol{\mu}_{\mathtt{h}} + \mathbf{C}[\,\delta\eta + \rho \;\ldots\; \delta\eta + \rho]^{\mathsf{T}} \tag{4.36}$$

and $\boldsymbol{\mu}_{\mathtt{h}}$ is the convolutional noise estimated in VTS noise estimation.

In summary, the noise estimation algorithm for RVTSJ is detailed in Algorithm 4.1. A similar algorithm is also used for RVTS-based noise estimation. In practice, it is found that 4 EM iterations (N_{EM}) and $2-3$ inner iterations (the "repeat" part) in Algorithm 4.1 work well.

In section 4.2.2 and section 4.2.3, the RVTS and RVTSJ compensation schemes and noise estimation algorithms have been presented. It is interesting to compare these two schemes.

Algorithm 4.1: Noise parameter estimation for RVTSJ

Given:

a sequence of observation vectors $\{\mathcal{Z} = (\cdots, \boldsymbol{z}_t, \cdots)\}$;
a supervision hypothesis $\mathcal{H}$;
VTS-based noise model parameters $\boldsymbol{\Phi}_{\mathtt{n}}$;
an initial T_{60} value;

Output:

the RVTSJ-based noise parameters $\boldsymbol{\Phi}$;

- Initialise the reverberant noise parameters $\boldsymbol{\Phi}$ using $\boldsymbol{\Phi}_{\mathtt{n}}$ and the initial T_{60} value (c.f., Eq. (4.36)) ;

for *i=1,...,* N_{EM} **do**

- Compensate the acoustic model using the current noise model $\boldsymbol{\Phi}$;
- Align the hypothesis $\mathcal{H}$ to obtain the component level occupancy $\gamma_t^{(m)}$;
- Compute the initial auxiliary function

$$\mathcal{Q} = \sum_{t,m} \gamma_t^{(m)} \log p(\boldsymbol{z}_t; \boldsymbol{\mu}_{\mathtt{z}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{z}}^{(m)})$$

repeat

- using the component occupancy and the current noise parameters, collect sufficient statistics:

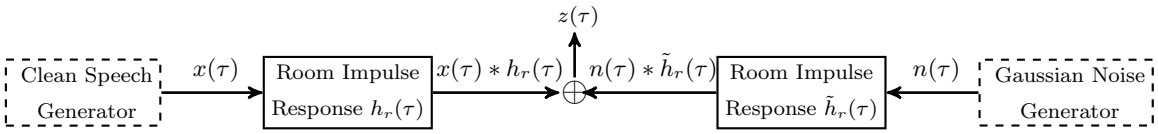
$$\left(\sum_{t,m} \gamma_t^{(m)} \mathbf{J}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{z}}^{-1} \mathbf{J}^{(m)}\right) \quad \text{and} \quad \left(\sum_{t,m} \gamma_t^{(m)} \mathbf{J}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{z}}^{-1} \boldsymbol{\mu}_{\mathtt{z}}^{(m)}\right)$$

- update noise parameters $\boldsymbol{\Phi}$ to $\hat{\boldsymbol{\Phi}}$ (c.f. Eq. (4.33));
- check auxiliary to make sure $\mathcal{Q}(\hat{\boldsymbol{\Phi}}) \geq \mathcal{Q}(\boldsymbol{\Phi})$, otherwise find $\lambda \in [0, 1]$ such that $\mathcal{Q}((1-\lambda)\hat{\boldsymbol{\Phi}} + \lambda\boldsymbol{\Phi}) \geq \mathcal{Q}(\boldsymbol{\Phi})$;
- update the noise parameters ;
- re-compensate the acoustic model ;

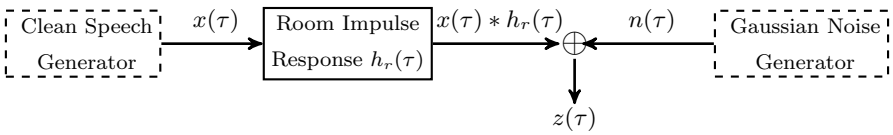
until *until convergence of* $\mathcal{Q}$;

end

The differences in these two approaches are due to the underlying assumption about the nature of background noise. RVTS is used to model the noise term $n_r(\tau) \approx \tilde{h}_r(\tau) * n(\tau)$, where $\tilde{h}_r(\tau)$ is a long-term convolutional distortion. In consequence, this additive noise is assumed to be correlated across different frames. In contrast, the RVTSJ models the additive noise without cross frame correlation. Figure 4.5 compares the assumption of addtive background noise in RVTS and RVTSJ. In theory, RVTS is able to capture the temporal correlation of background noise while RVTSJ is better for the additive noise without temporal correlation. In practice, to avoid the expensive calculation of a large number of Jacobian matrices, a sequential noise estimation for RVTS is used, i.e., standard VTS is used to estimate the additive noise model



(a) Frame-correlated additive noise modelled by RVTS.



(b) Frame-uncorrelated additive noise modelled by RVTSJ.

Figure 4.5: Comparing the additive noise modelling forms of RVTS and RVTSJ.

parameters without considering the reverberation effect. As a result, the additive noise model parameters in RVTS are usually larger than the observed values, since part of the reverberant speech is taken as the additive noise in the standard VTS mismatch function. In contrast, RVTSJ model compensation scheme allows joint estimation of the reverberant and additive noise. Therefore, it is expected RVTSJ can achieve better recognition accuracy than RVTS. This will be demonstrated by the experiments presented in chapter 7.

4.3 Reverberation Adaptive Training

In the previous sections, the model compensation and noise estimation have been already discussed, in which the underlying acoustic model $\mathcal{M}$ is assumed to be known. The simplest method to estimate $\mathcal{M}$ is to simply use the clean data to train an acoustic model. However, when a large quantity of found data is available, the recognition performance will be limited if only clean data is used. Moreover, both model compensation schemes, RVTS and RVTSJ, make many approximations, which may cause a "residual" mismatch between the compensated models and the observed data, and this "residual" mismatch must be modelled by the acoustic model. Alternatively, an adaptive training framework can be applied in which both the acoustic model and the noise model are trained in a ML framework on multi-condition data. This is a powerful technique to "factor out"[69] the unwanted acoustic factors, for example, the reverberation and background noise in this work. This is expected to yield a canonical model $\mathcal{M}_c$ that models only the relevant phone variations. This adaptive training framework also enables acoustic models to be trained on found data, which is more accessible than data without noise. In this work, the adaptive training framework is extended to handle to reverberant and additive noise distortions. This will be referred to as reverberant adaptive

training (RAT). As the RVTSJ modelling scheme allows a joint estimation of the reverberant and additive noise, this work focuses on RAT using the RVTSJ model compensation scheme.

In adaptive training, both the canonical model $\mathcal{M}_{\mathtt{c}}$ [1] and a set of noise model parameters $\{\boldsymbol{\Phi}^{(r)}\}$ are iteratively estimated using EM, where $\boldsymbol{\Phi}^{(r)}$ is the noise model parameter of the r-th homogeneous block. First, given the current canonical model, the noise models $\boldsymbol{\Phi}$ are estimated for each utterance [2], using the noise estimation algorithm presented in the last section. Then the canonical model $\mathcal{M}_{\mathtt{c}}$ is updated given the current noise models. Multiple iterations may be performed to interleave optimisation in the EM framework.

In RAT, the canonical model $\mathcal{M}_{\mathtt{c}}$ is estimated by maximising the following auxiliary function:

$$\mathcal{Q}(\hat{\mathcal{M}}_{\mathtt{c}};\mathcal{M}_{\mathtt{c}}) = \sum_{r,t,m} \gamma_t^{(mr)} \log p(\boldsymbol{z}_t^{(r)}; \hat{\boldsymbol{\mu}}_{\mathtt{z}}^{(mr)}, \boldsymbol{\Sigma}_{\mathtt{z}}^{(mr)}) + \mathcal{R}_{\mathtt{c}}(\hat{\mathcal{M}}_{\mathtt{c}}, \mathcal{M}_{\mathtt{c}}) \tag{4.37}$$

where $\mathcal{M}_{\mathtt{c}}$ is the current canonical model and $\hat{\mathcal{M}}_{\mathtt{c}}$ is the new canonical model to be updated; r is the index of utterance; $\boldsymbol{z}_t^{(r)}$ is the t-th observation vector in the r-th utterance; $\gamma_t^{(mr)}$ is the posterior of the component m at time t for the r-th utterance, calculated using the current model $\mathcal{M}_{\mathtt{c}}$ and the current noise model $\boldsymbol{\Phi}^{(r)} = (\boldsymbol{\mu}_{\mathtt{l}}^{(r)}, \boldsymbol{\mu}_{\mathtt{n}}^{(r)})$; the compensated mean vector $\hat{\boldsymbol{\mu}}_{\mathtt{z}}^{(m)}$ is given by:

$$\hat{\boldsymbol{\mu}}_{\mathtt{z}}^{(m)} = \begin{bmatrix} \hat{\boldsymbol{\mu}}_{\mathtt{sz}}^{(m)} \\ \hat{\boldsymbol{\mu}}_{\Delta\mathtt{z}}^{(m)} \end{bmatrix} = \begin{bmatrix} \mathbf{g}(\mathbf{P}_{\mathtt{s}}\hat{\overline{\boldsymbol{\mu}}}_{\mathtt{x}}^{(m)}, \boldsymbol{\mu}_{\mathtt{l}}^{(r)}, \boldsymbol{\mu}_{\mathtt{n}}^{(r)}) \\ \mathbf{J}_{\mathtt{xe}}^{(m)}\mathbf{P}_{\Delta}\hat{\overline{\boldsymbol{\mu}}}_{\mathtt{x}}^{(m)} \end{bmatrix} . \tag{4.38}$$

In line with RVTSJ model compensation, the standard VTS variance compensation is used to calculate $\boldsymbol{\Sigma}_{\mathtt{z}}^{(m)}$. As a result, the model variance $\boldsymbol{\Sigma}_{\mathtt{x}}^{(m)}$ is not adaptively trained in RAT and the canonical model consists of the extended mean vectors, i.e., $\mathcal{M}_{\mathtt{c}} = \{\overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)}\}$. A regularisation term $\mathcal{R}_{\mathtt{c}}(\hat{\mathcal{M}}_{\mathtt{c}};\mathcal{M}_{\mathtt{c}})$ is used in Eq. (4.37) to ensure the stability of canonical model parameters update. The following form of regularisation is used in this work:

$$\mathcal{R}_{\mathtt{c}}(\hat{\mathcal{M}}_{\mathtt{c}};\mathcal{M}_{\mathtt{c}}) = \beta \sum_{m} (\hat{\overline{\boldsymbol{\mu}}}_{\mathtt{x}}^{(m)} - \overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)})^{\mathsf{T}} (\hat{\overline{\boldsymbol{\mu}}}_{\mathtt{x}}^{(m)} - \overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)}) \tag{4.39}$$

and β is a tunable parameter.

Approximations are needed to update the canonical model parameters. Again, the vector Taylor series expansion technique can be applied to expand $\hat{\boldsymbol{\mu}}_{\mathtt{z}}^{(m)}$ using the current canonical model estimates:

$$\hat{\boldsymbol{\mu}}_{\mathtt{z}}^{(m)} \approx \begin{bmatrix} \boldsymbol{\mu}_{\mathtt{sz}}^{(m)} \\ \boldsymbol{\mu}_{\Delta\mathtt{z}}^{(m)} \end{bmatrix} + \begin{bmatrix} \mathbf{J}_{\mathtt{xe}}^{(mr)} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{\mathtt{xe}}^{(mr)} \end{bmatrix} \begin{bmatrix} \mathbf{P}_{\mathtt{s}} \\ \mathbf{P}_{\Delta} \end{bmatrix} (\hat{\overline{\boldsymbol{\mu}}}_{\mathtt{x}}^{(m)} - \overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)}) \tag{4.40}$$

[1] The subscript $\mathtt{c}$ is used to emphasis that the underlying acoustic models are adaptively trained canonical models.

[2] It is assumed in this work, each utterance has a unique noise condition. Thus a homogeneous block is defined at the utterance level.

where $\mathbf{J}_{\mathtt{xe}}^{(mr)}$ is the Jacobian matrix calculated by:

$$\mathbf{J}_{\mathtt{xe}}^{(mr)} = \frac{\partial \mathbf{g}(\boldsymbol{\mu}_{\mathtt{xe}}, \boldsymbol{\mu}_{\mathtt{l}}, \boldsymbol{\mu}_{\mathtt{n}})}{\partial \boldsymbol{\mu}_{\mathtt{xe}}}\Big|_{\mathbf{P}_{\mathtt{s}}\overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)}, \boldsymbol{\mu}_{\mathtt{l}}^{(r)}, \boldsymbol{\mu}_{\mathtt{n}}^{(r)}} \tag{4.41}$$

Differentiating the auxiliary function and equating to zero gives the following update:

$$\begin{aligned}\hat{\overline{\boldsymbol{\mu}}}_{\mathtt{x}}^{(m)} = \overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)} + \left(\sum_r \gamma^{(mr)} \mathbf{K}^{(mr)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{z}}^{(mr)-1} \mathbf{K}^{(mr)\mathsf{T}} + \beta \mathbf{I}\right)^{-1} \\ \times \left(\sum_r \gamma^{(mr)} \mathbf{K}^{(mr)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{z}}^{(mr)-1} (\boldsymbol{\Gamma}_{\mathtt{z}}^{(mr)} - \boldsymbol{\mu}_{\mathtt{z}}^{(mr)})\right)\end{aligned} \tag{4.42}$$

where

$$\mathbf{K}^{(mr)} = \begin{bmatrix} \mathbf{J}_{\mathtt{xe}}^{(mr)} \mathbf{P}_{\mathtt{s}} \\ \mathbf{J}_{\mathtt{xe}}^{(mr)} \mathbf{P}_{\Delta} \end{bmatrix}$$

and $\gamma^{(mr)} = \sum_t \gamma_t^{(mr)}$; $\boldsymbol{\Gamma}_{\mathtt{z}}^{(mr)} = \frac{1}{\gamma^{(mr)}} \sum_t \gamma_t^{(mr)} \boldsymbol{z}_t^{(r)}$. Due to the VTS approximation in Eq. (4.40), the update formula in Eq. (4.42) does not guarantee the increase in likelihood. Therefore, the following update formula with a step size ζ is used:

$$\begin{aligned}\hat{\overline{\boldsymbol{\mu}}}_{\mathtt{x}}^{(m)} = \overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)} + \zeta \left(\sum_r \gamma^{(mr)} \mathbf{K}^{(mr)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{z}}^{(mr)-1} \mathbf{K}^{(mr)\mathsf{T}} + \beta \mathbf{I}\right)^{-1} \\ \times \left(\sum_r \gamma^{(mr)} \mathbf{K}^{(mr)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{z}}^{(mr)-1} (\boldsymbol{\Gamma}_{\mathtt{z}}^{(mr)} - \boldsymbol{\mu}_{\mathtt{z}}^{(mr)})\right)\end{aligned} \tag{4.43}$$

For every update, the step size ζ is set as 1 initially. If the initial update does not increases the auxiliary function, a simple back-off procedure, similar to the one in [158], is used to reduce the step size until the auxiliary function increases. Since the auxiliary function of the canonical model involves all the utterances, to make this back-off procedure possible, it is necessary to save $\boldsymbol{\Gamma}_{\mathtt{z}}^{(mr)}$ for each component m and each utterance r, provided $\gamma^{(mr)}$ is not zero. This is impractical for a large-scale task. An approximation of the auxiliary function in Eq. (4.37) is used: for each component m, the summation over all the utterances is done on a subset $\mathcal{R}_m = \{r|\gamma^{(mr)} \geq \theta_m\}$, where θ_m is chosen such that the top N utterances are in this subset. $N = 156$ is used in this work and it was found this yields a good increase in likelihood. [1]

Since the auxiliary function is highly non-linear, it is also crucial to have a good initialisation of the canonical model parameters. The simplest method is to use the multi-condition

[1] Note that Eq. (4.43) involves a $D \times D$ matrix inversion, of which the main part is a sum of N matrices. As such, N must be equal to or larger than D for the worse-case scenario in which each of the N matrices degrades to a rank-1 matrix. For the experiments carried out in this thesis, 39-dimensional feature vectors were used, thus $D = 39$; setting $N = 4D = 156$ guarantees the matrix is invertible.

Algorithm 4.2: Reverberant adaptive training procedure.

Given:

a set of training utterances $\{\mathcal{Z}^{(r)}|r=1,\cdots,R\}$;

Output:

a set of RVTSJ noise model parameters $\{\mathbf{\Phi}^{(r)}|r=1,\cdots,R\}$ and the canonical model $\mathcal{M}_{\mathtt{c}}$, such that the auxiliary function in Eq. (4.37) is jointly maximised;

- Perform VTS-based adaptive training to initialise canonical model parameters $\mathcal{M}_{\mathtt{c}}=\{\boldsymbol{\mu}_{\mathtt{x}}^{(m)},\boldsymbol{\Sigma}_{\mathtt{x}}^{(m)}\}$, which is described in detail in [122];

for *m = 1, . . . , M* **do**

- Initialise the extended mean model parameters $\overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)}$ via solving the optimisation problem in Eq. (4.45);

end

for *i=1, . . . , N_{model}* **do**

for *r=1,. . . , R* **do**

Given the current canonical model $\mathcal{M}_{\mathtt{c}}=\{\overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)},\boldsymbol{\Sigma}_{\mathtt{x}}^{(m)}\}$, re-estimate the RVTSJ noise model parameters $\mathbf{\Phi}^{(r)}$ N_{trans} times (c.f. Algorithm 4.1) ;

end

for *j=1,. . . , N_{EM}* **do**

Given the noise models $\{\mathbf{\Phi}^{(r)}\}$ and the current canonical model $\mathcal{M}_{\mathtt{c}}$, collect sufficient statistics $\gamma^{(mr)}$ and $\mathbf{\Gamma}_{\mathtt{z}}^{(mr)}$ for each utterance r and each component m;

for *m=1,. . . , M* **do**

- Given sufficient statistics $\{\gamma^{(mr)}\}$, $\{\mathbf{\Gamma}_{\mathtt{z}}^{(mr)}\}$ and the noise model parameters $\{\mathbf{\Phi}^{(r)}\}$, update $\overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)}$ for each component m using Eq. (4.42);
- Check the auxiliary function in Eq. (4.37). Reduce the step size ζ and redo the update until auxiliary is increasing;

end

- Gradually reduce the parameters β;

end

end

data as if it were clean, i.e.,

$$\overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)}=\mathbf{W}\cdot\mathcal{E}\left\{\left[\begin{array}{c}\boldsymbol{z}_{t+w}^{\mathtt{s}}\\ \vdots\\ \boldsymbol{z}_{t-n-w}^{\mathtt{s}}\end{array}\right]|m\right\} \tag{4.44}$$

However, it was found that this form of initialisation is suboptimal in the initial experiments, since the extended statistics $\{\overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)}\}$ are initialised using reverberant observations. In this work, the following strategy is used. The standard model parameters, $\{\boldsymbol{\mu}_{\mathtt{x}}^{(m)}\}$, are set as the parameter obtained by the VTS-based adaptive training (VAT); it is assumed that $\boldsymbol{\mu}_{\mathtt{x}\delta}^{(m)},\delta=0\ldots n$ is a smooth trajectory starting from $\boldsymbol{\mu}_{\mathtt{x}0}^{(m)}=\boldsymbol{\mu}_{\mathtt{x}}^{(m)}$; therefore a reconstruction error needs to be minimised. In this work, $\overline{\boldsymbol{\mu}}_{\mathtt{x}}^{(m)}$ is initialised to the value which minimises the

following constrained optimisation problem:

$$\begin{aligned} \min \quad & \sum_{\delta=1}^{n} w_\delta \|\mathbf{Q}_\delta \overline{\boldsymbol{\mu}}_{\mathbf{x}}^{(m)} - \boldsymbol{\mu}_{\mathbf{x}}^{(m)}\|^2_{\boldsymbol{\Sigma}^{(m)-1}} \\ \text{s.t.} \quad & \mathbf{Q}_0 \overline{\boldsymbol{\mu}}_{\mathbf{x}}^{(m)} = \boldsymbol{\mu}_{\mathbf{x}}^{(m)} \end{aligned} \tag{4.45}$$

where $\|\boldsymbol{v}\|_{\boldsymbol{\Sigma}} = \boldsymbol{v}^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{v}$; $\mathbf{Q}_\delta$ is the matrix that maps $\overline{\boldsymbol{\mu}}_{\mathbf{x}}^{(m)}$ to $\boldsymbol{\mu}_{\mathbf{x}\delta}^{(m)}$, and w_δ is the weight associated with the reconstruction error of $\boldsymbol{\mu}_{\mathbf{x}\delta}^{(m)}$. In this work, the weight is set as $w_\delta = 10^{-\delta\frac{3\Delta}{T_{60}}}$, where Δ is the shift of the analysis window, and T_{60} the median of reverberation time in the multi-condition training data (for example, 400ms in this work). The constrained optimisation in Eq. (4.45) is a standard linearly constrained quadratic programming problem and can be solved analytically.

The RAT training algorithm is summarised in Algorithm 4.2. In practice, the number of iterations in this algorithm are set as: $N_{\text{trans}} \sim 2$, $N_{\text{model}} \sim 2$ or 3 and $N_{\text{EM}} \sim 4$.

4.4 Summary

This chapter presents a model-based approach to robust speech recognition in reverberant environments. Due to the reverberation effect, exact inference is not tractable. The proposed scheme in this chapter approximates the distribution of preceding frames using a conditional distribution of an extended vector for every Gaussian component. In this way, conventional Viterbi decoding can still be used after model compensation. Given these extended model statistics, VTS was reformulated to handle the nonlinear reverberant mismatch functions. This yields the RVTS and RVTSJ model compensation schemes, where RVTS is used to model frame correlated additive noise while RVTSJ is used to model the frame uncorrelated additive noise. ML estimation of the reverberant noise model is also presented.

The extended model statistics can be estimated from the clean training data. However, this limits the acoustic models can be only trained on clean data. Training of acoustic models on multi-condition data is possible. However, models trained on multi-condition data implicitly model all the variability exhibited in the training data. Another limitation of multi-style trained models is that it is no longer possible to use the mismatch function to compensate the acoustic model for a particular environment, as the mismatch function uses a clean speech distribution. Motivated by the success of VTS-based adaptive training (VAT), the second part of this chapter investigated model-based approaches to adaptive training in reverberant environments. RVTSJ is used to compensate acoustic models in both training and testing, yielding the reverberant adaptive training (RAT) scheme. Maximum likelihood (ML) estimation of the canonical model parameters in the EM framework is described. This RAT

scheme provides a powerful method to factor out the effects of reverberant and background noise from the canonical acoustic models.

Chapter 5
Acoustic Factorisation Framework

In the previous chapters, the impacts of various acoustic factors on speech recognition systems, such as speaker characteristics (section 3.2),channel distortion, background noise (section 3.3) and reverberation (chapter 4), were discussed. Adaptation schemes which are designed to improve acoustic model robustness against these factors were presented, with a focus on model-based approaches. So far, the adaptation schemes mostly focused on robustness against a single factor. In practice, speech recognition systems often need to deal with complex acoustic environments where there are multiple acoustic factors simultaneously affecting speech signals. For example, a typical speech recogniser is often required to operate in a wide range of environments for a large number of possible users. It should have the ability to be adapted to a particular speaker and environment condition rapidly. The conventional approach is to build a model transform for every target condition. This requires adaptation data for all possible operating conditions. The acoustic factorisation framework, first proposed in 2001 [69], addresses this problem in a more effective way. In this framework, a transform is associated with one distinct acoustic factor and therefore is referred to as a *factor transform*. These factor transforms are combined to yield the final model transform for the target acoustic condition. They can be estimated from the relevant data and can be combined to construct

complex model transforms for conditions never seen before. To achieve factorisation, it is crucial that each factor transform only models the impact of its associated factor, and keeps it independent from others. This chapter will discuss three possible options to construct such factor transforms. The first, data-constrained, approach entirely relies on balanced data as an implicit constraint, while the second and the third approach use additional constraints to enforce the independence: the second, transform constrained, approach utilises the knowledge of how acoustic factors impact speech signals while the third approach is based on an explicit constraint derived from a mathematical analysis of interdependence between factor transforms.

5.1 Acoustic Factorisation

Before presenting the details of the acoustic factorisation framework, it is worthwhile pointing out the fundamental assumptions in this framework.First, acoustic factors are assumed to impact speech signals *differently*. For example, background noise normally masks the speech signals in some acoustic regions with low speech energy, while speaker characteristics usually differ in the frequency warping. This gives an opportunity to separate the variability caused by different acoustic factors. Another crucial assumption is that acoustic factors impact speech signals *independently*. For many pairs or groups of acoustic factors, this is a reasonable assumption, as acoustic factors are usually driven by physical factors, which often have no connection. One exception is the Lombard effect [120], in which a speaker can change the way he or she speaks in a very noisy environment; therefore the speaker and environment characteristics become related. In this thesis, it is assumed that the noise level is not sufficiently high so that the Lombard effect is not significant.

Based on these assumptions, the acoustic factorisation considered here is an extension to the previously described model-based framework. In this framework, intrinsic and extrinsic variabilities are represented by a canonical model $\mathcal{M}_{\mathtt{c}}$ and a set of transforms $\mathcal{T}$, respectively. Consider a complex acoustic environment, simultaneously affected by two acoustic factors $\mathtt{s}_i$ and $\mathtt{n}_j$ respectively, where $\mathtt{s}$ and $\mathtt{n}$ are the types of acoustic factors and i, j are the indices within each acoustic factor [1]. The canonical model is adapted to represent this condition by the transform $\mathcal{T}^{(i,j)}$:

$$\mathcal{M}^{(i,j)} = \mathcal{F}(\mathcal{M}_c, \mathcal{T}^{(i,j)}) \tag{5.1}$$

[1]Note the typewriter font (e.g., $\mathtt{s}$) is used to denote the acoustic factor, while the italic font (e.g., i) is used to denote the index

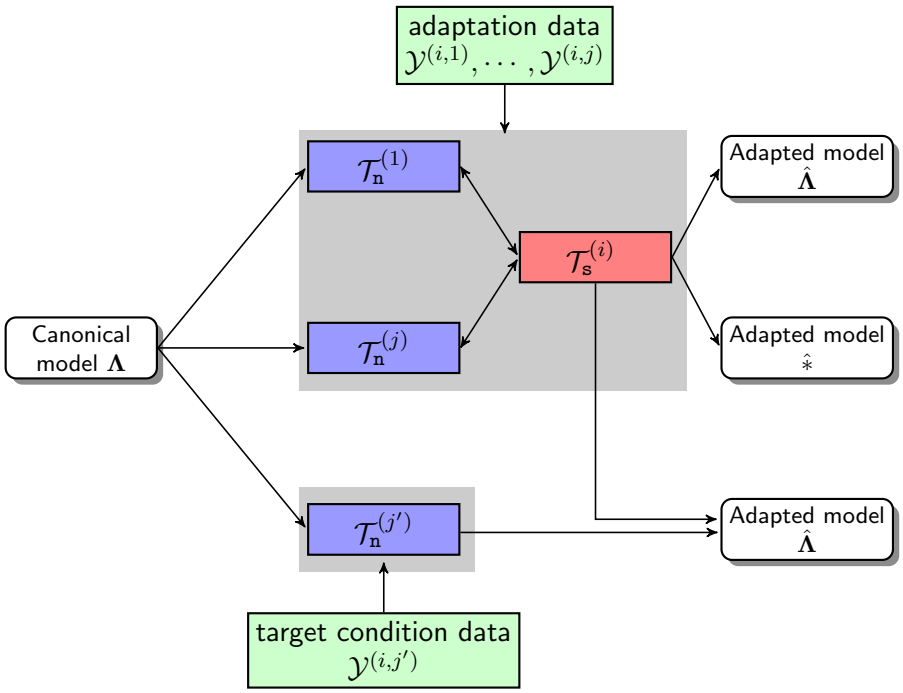


Figure 5.1: Speaker and noise adaptation in the factorisation mode.

where $\mathcal{M}^{(i,j)}$ is the adapted acoustic model for condition $(\mathtt{s}_i, \mathtt{n}_j)$, $\mathcal{T}^{(i,j)}$ the transform for the target condition, and $\mathcal{F}$ is a mapping function. The transform is normally estimated using the ML criterion:

$$\mathcal{T}^{(i,j)} = \arg\max_{\mathcal{T}} \left\{ p(\mathcal{O}^{(i,j)} | \mathcal{M}_c, \mathcal{T}) \right\} \tag{5.2}$$

where $\mathcal{O}^{(i,j)}$ is a sequence of feature vectors observed in the acoustic condition $(\mathtt{s}_i, \mathtt{n}_j)$. It is possible to directly estimate the transform $\mathcal{T}^{(i,j)}$ from $\mathcal{O}^{(i,j)}$ and use $\mathcal{T}^{(i,j)}$ in the target condition $(\mathtt{s}_i, \mathtt{n}_j)$. However, in a rapidly changing environment, there is not enough data to build model transforms for every target condition. As model transforms are built on the same kind of data to which they are applied, this adaptation strategy will be referred to as *batch* adaptation.

To effectively deal with complex acoustic environments, the concept of acoustic factorisation was proposed in [69], where the final transform can be decomposed into a few *factor* transforms, each of which is constrained to be related to an individual acoustic factor. In the above example, this requires that the transform $\mathcal{T}^{(i,j)}$ can be expressed as:

$$\mathcal{T}^{(i,j)} = \mathcal{T}_{\mathtt{s}}^{(i)} \otimes \mathcal{T}_{\mathtt{n}}^{(j)} \tag{5.3}$$

where $\otimes$ denotes transform composition and its exact form dependents on the nature of factor transforms; $\mathcal{T}_{\mathtt{s}}^{(i)}$ and $\mathcal{T}_{\mathtt{n}}^{(j)}$ are the factor transforms associated with $\mathtt{s}_i$ and $\mathtt{n}_j$, respectively. The

factorisation attribute in Eq. (5.3) offers flexibility for acoustic models to be used in complex acoustic environments. This can be demonstrated by considering R utterances $\mathcal{O}^{(1)}, \cdots, \mathcal{O}^{(R)}$, produced by a set of speakers $(\mathtt{s}_1, \cdots, \mathtt{s}_I)$ in a range of different noise $(\mathtt{n}_1, \cdots, \mathtt{n}_J)$ conditions. In batch mode adaptation, it is required to estimate a transform $\mathcal{T}^{(r)}$ for every utterance r. This requires $\mathcal{T}^{(r)}$ should be robustly estimated using a single utterance, which is often not possible in practice. However, under the acoustic factorisation, $\mathcal{T}^{(r)}$ can be decomposed to factor transforms $\mathcal{T}_{\mathtt{s}}^{(s_r)}$ and $\mathcal{T}_{\mathtt{n}}^{(n_r)}$, where $s_r \in \{1, \cdots, I\}$ and $n_r \in \{1, \cdots, J\}$ are the speaker and noise indices of utterance r respectively. Hence it only needs to estimate I speaker factor transforms and J noise factor transforms. Utterances produced by the i-th speaker can be pooled to estimate $\mathcal{T}_{\mathtt{s}}^{(i)}$ while utterances recorded in the j-th noise condition can be pooled to estimate $\mathcal{T}_{\mathtt{n}}^{(j)}$. This enables more powerful transforms with more parameters to be used for an individual factor as data from multiple conditions can be used. Furthermore, for the data produced by a seen speaker $\mathtt{s}_i$ in unseen noise condition $\mathtt{n}_{j'}$, it is only necessary to estimate the noise transform $\mathcal{T}_{\mathtt{n}}^{(j')}$ and combine this transform with the existing speaker transform $\mathcal{T}_{\mathtt{s}}^{(i)}$. Figure 5.1 shows the concept of acoustic factorisation for the speaker and noise adaptation.

In summary, algorithm 5.1 illustrates how the speaker and noise adaptation can be performed in this framework. This will be referred to as factorised adaptation mode. Note that the canonical model, $\mathcal{M}_c$, is assumed to have been trained. Having obtained the speaker and noise transforms for the adaptation data, the transform for a new acoustic condition $(\mathtt{s}_i, \mathtt{n}_{j'})$, can be obtained simply by estimating the noise transform

$$\mathcal{T}_{\mathtt{n}}^{(j')} = \arg\max_{\mathcal{T}} \left\{ p(\mathcal{O}|\mathcal{M}_c, \mathcal{T}_{\mathtt{s}}^{(i)} \otimes \mathcal{T}) \right\} \tag{5.6}$$

Given the speaker transform and the noise transforms, the acoustic model is adapted to the test condition using the transform $\mathcal{T}^{(i,j')} = \mathcal{T}_{\mathtt{s}}^{(i)} \otimes \mathcal{T}_{\mathtt{n}}^{(j')}$.

The factorisation attribute relies on the "orthogonality" of factor transforms. For example, the speaker transform only models speaker attributes and the noise transform the noise attributes. This is illustrated in detail in Figure 5.2, using the speaker and environment adaptation as an example. Assuming the impact of speaker and environment on the acoustic model can be represented by model transforms whose parameters are $\boldsymbol{\lambda}_{\mathtt{s}}$ and $\boldsymbol{\lambda}_{\mathtt{n}}$ respectively, two ellipses in figure 5.2 illustrate the speaker and environment coverage in the adaptation data, where point b and c represent two conditions observed. To adapt the model to the target condition a, only the parameter $\boldsymbol{\lambda}_{\mathtt{s}}^{(b)}$ and $\boldsymbol{\lambda}_{\mathtt{n}}^{(c)}$ are needed, which can be estimated from adaptation data. This is due to the *independence* between two factor transforms, as shown in Figure 5.2: when the operating condition is moved from point b towards point a, $\boldsymbol{\lambda}_{\mathtt{n}}$ gets a small update $\Delta\boldsymbol{\lambda}_{\mathtt{n}}$ to reflect the environment transition; due to the independence between $\boldsymbol{\lambda}_{\mathtt{s}}$ and $\boldsymbol{\lambda}_{\mathtt{n}}$, speaker transform will not be affected by $\Delta\boldsymbol{\lambda}_{\mathtt{n}}$.

Algorithm 5.1: Adaption under the acoustic factorisation framework.

Input:

Acoustic data $\mathcal{O}^{(1)}, \cdots, \mathcal{O}^{(R)}$ observed in a range of acoustic conditions $(\mathtt{s}_i, \mathtt{n}_j)$: $i = 1, \cdots, I, j = 1, \cdots, J$;

Output:

Model transforms $\mathcal{T}^{(1)}, \cdots, \mathcal{T}^{(R)}$ such that the likelihood of observed acoustic data is maximised;

Initialisation:

Initialise the factor transform;

Iterating:

while *not converged* **do**

1. Estimate speaker transforms while keeping noise transforms fixed:

$$\mathcal{T}_{\mathtt{s}}^{(i)} = \arg\max_{\mathcal{T}} \left\{ \sum_{r:s_r=i} \log p(\mathcal{O}^{(r)}|\mathcal{M}_c, T \otimes T_{\mathtt{n}}^{(n_r)}) \right\} \quad \forall i = 1, \cdots, I; \tag{5.4}$$

2. Estimate noise transform while keeping speaker transforms fixed:

$$\mathcal{T}_{\mathtt{n}}^{(j)} = \arg\max_{\mathcal{T}} \left\{ \sum_{r:n_r=j} \log p(\mathcal{O}^{(r)}|\mathcal{M}_c, \mathcal{T}_{\mathtt{s}}^{(s_r)} \otimes \mathcal{T}) \right\} \quad \forall j = 1 \cdots J; \tag{5.5}$$

end

Composition:

Combine factor transforms to form the final transformation:

$$\mathcal{T}^{(r)} = \mathcal{T}^{(s_r)} \otimes \mathcal{T}^{(n_r)} \; r = 1, \cdots, R$$

where s_r an n_r are the speaker and noise indices of the r-th utterance, respectively.

5.2 Approaches to Factorisation

As discussed before, acoustic factorisation offers flexibility for adapting canonical models to complex acoustic conditions. To make the factorisation work, it is important to keep an independence between factor transforms. The simplest way is to rely on *balanced* data to force a factor transform to learn the impact caused by its associated factor, while keeping it invariant to other factors. Works in the literature that fall in this category will be briefly reviewed in section 5.2.1. A second approach is to use different forms of factor transforms. It is hoped by using factor transforms designed for different acoustic factors, they are able to model the specific factor for which they have been designed to model. This argument is presented in section 5.2.2. Unlike the first two approaches relying on an *implicit* constraint to enforce independence, the third one is based on an *explicit* constraint. Section 5.2.3 first analyses

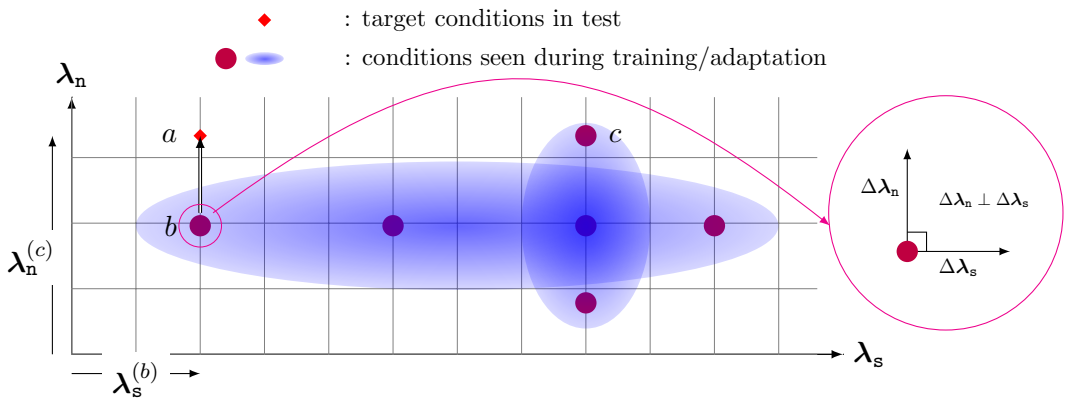


Figure 5.2: Factorised adaptation to the target speaker and environment: transform independence. It is assumed that the impact of speaker and environment on the acoustic model can be represented by model transforms $\boldsymbol{\lambda}_{\mathbf{s}}$ and $\boldsymbol{\lambda}_{\mathbf{n}}$ respectively; two ellipses illustrate the speaker and environment coverage in the adaptation data; point b and c represent two observed conditions and point a is the target condition.

the dependence between factor transforms, and derives an explicit constraint. Finally, the advantages and disadvantages of each approach are compared and discussed in section 5.3.

5.2.1 Data Constrained Approach

When factor transforms are estimated on balanced and representative data, the ML estimators in Eqs (5.4-5.5) already form an implicit constraint. Take the speaker and noise factorisation example illustrated in Figure 5.1 again: as the speaker transform is set to maximise the likelihood of data produced by the same speaker, for example $\mathbf{s}_i$, under many noise conditions, $\{n_r|r : s_r = i\}$, the estimated speaker transform optimally models the speaker characteristics irrespective of the noise factors, as long as the noise conditions of data $\{\mathcal{O}^{(r)}|r : s_r = i\}$ are distributed in a balanced way. In the same manner, the noise transform estimated in Eq. (5.5) is believed to model only the noise attribute irrespective of speaker characteristics. As the speaker/noise characteristic is an intrinsic attribute of speaker/environment, it can be argued that speaker and noise factor transforms are independent of each other.

As this data constrained approach is the simplest method of factorisation, most of the factorisation works in the literature fall in this category. In [263], structured transforms, clustered mean interpolation combined with CMLLR, are used to remove the effects of unwanted acoustic factors. However, these component transforms are not constrained to specific acoustic factors. Interestingly, structured transform have been applied to polyglot speech synthesis recently [265], in which CMLLR transforms are used to modelling the speaker factor, while cluster means are interpolated to yield a particular language space. Under this modelling

scheme, the probability distribution of the m-th Gaussian component is adapted to the i-th speaker, $\mathtt{s}_i$ and the j-th language, $\mathtt{l}_j$, by:

$$p(\boldsymbol{o}_t|i,j,m) = |\mathbf{A}_{\mathtt{s}}^{(i)}|\mathcal{N}(\mathbf{A}_{\mathtt{s}}^{(i)}\boldsymbol{o}_t + \boldsymbol{b}_{\mathtt{s}}^{(i)}; \sum_{c=1}^{C} \lambda_{\mathtt{l},c}^{(j)}\boldsymbol{\mu}_c^{(m)}, \boldsymbol{\Sigma}^{(m)}) \tag{5.7}$$

where $\mathtt{l}$ denotes the language factor; cluster means $\{\boldsymbol{\mu}_c^{(m)}|c=1,\cdots,C\}$ and variances $\boldsymbol{\Sigma}^{(m)}$ form the canonical model parameter; $\mathbf{W}_{\mathtt{s}}^{(i)} = [\mathbf{A}_{\mathtt{s}}^{(i)}, \boldsymbol{b}_{\mathtt{s}}^{(i)}]$ is the speaker transform, and cluster weight vector $\boldsymbol{\lambda}_{\mathtt{l}}^{(j)} = [\lambda_{\mathtt{l},1}^{(j)}, \cdots, \lambda_{\mathtt{l},C}^{(j)}]^{\mathsf{T}}$ is used to interpolate cluster means to yield a language specific mean vector.

In [217, 218], the impacts of speaker and noise factors are represented by two sets of CMLLR transforms in a cascaded fashion, i.e., the likelihood of a particular observation vector $\boldsymbol{o}_t$ produced by speaker i under environment j emitted by the m-th Gaussian component is:

$$\begin{aligned} p(\boldsymbol{o}_t|i,j,m) &= |\mathbf{A}_{\mathtt{s}}^{(i)}||\mathbf{A}_{\mathtt{n}}^{(j)}|\mathcal{N}(\hat{\boldsymbol{x}}_t; \boldsymbol{\mu}_{\mathtt{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{x}}^{(m)}) \\ \hat{\boldsymbol{x}}_t &= \mathbf{A}_{\mathtt{s}}^{(i)}(\mathbf{A}_{\mathtt{n}}^{(j)}\boldsymbol{o}_t + \boldsymbol{b}_{\mathtt{n}}^{(j)}) + \boldsymbol{b}_{\mathtt{s}}^{(i)} \end{aligned} \tag{5.8}$$

where $\boldsymbol{\mu}_{\mathtt{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{x}}^{(m)}$ are the canonical model parameter, and $\mathbf{W}_{\mathtt{s}} = [\mathbf{A}_{\mathtt{s}}^{(i)}, \boldsymbol{b}_{\mathtt{s}}^{(i)}]$ and $\mathbf{W}_{\mathtt{n}} = [\mathbf{A}_{\mathtt{n}}^{(j)}, \boldsymbol{b}_{\mathtt{n}}^{(j)}]$ are the speaker and noise transforms respectively. Using this cascaded CMLLR transform, the noise and speaker distorted speech vector $\boldsymbol{o}_t$ is normalised to $\hat{\boldsymbol{x}}_t$, where the transforms are estimated from the corresponding speaker or noise data. This scheme was evaluated on the AURORA-2 task and was shown to be able to port a speaker transform from one noise condition to another noise condition. In [219], another factorisation form was proposed, where speaker and noise adaptation was performed using CMLLR and MLLR respectively. The proposed factorisation scheme along with an environment clustering scheme were shown to be effective on a Bing voice search task. Although effective, there is no built-in mechanism in the above schemes to prevent the speaker transform from modelling the noise variability. Thus these schemes rely on balanced data to separate the speaker and noise variability. In the extreme case, if a speaker's data are all in a single noise condition, the speaker and noise transform will model the combined effect, which means they cannot be factorised from each other.

In speaker recognition or verification, it is important to separate the speaker variability from the session and/or environment variability. Therefore, the concept of acoustic factorisation is also used in the speaker recognition/verification area. The Joint Factor Analysis (JFA)[127, 128] model is an example. In JFA, a supervector $\boldsymbol{\mu}$ is formed by stacking the mean vectors of a speaker and session adapted GMM and the variability of this D-dimensional supervector is explained by the speaker and session factors, $\boldsymbol{\lambda}_{\mathtt{s}}$ and $\boldsymbol{\lambda}_{\mathtt{n}}$, which sit in lower

dimensional subspaces:

$$\boldsymbol{\mu} = \boldsymbol{\mu}_0 + \mathbf{M}_{\mathbf{s}}\boldsymbol{\lambda}_{\mathbf{s}} + \mathbf{M}_{\mathbf{n}}\boldsymbol{\lambda}_{\mathbf{n}} + \mathbf{D}\boldsymbol{\lambda}_z \tag{5.9}$$

where $\boldsymbol{\mu}_0$ is the supervector formed by concatenating the mean vectors of a universal background model (UBM); $\mathbf{M}_{\mathbf{s}} \in \mathcal{R}^{D\times d_{\mathbf{s}}}$ and $\mathbf{M}_{\mathbf{n}} \in \mathcal{R}^{D\times d_{\mathbf{n}}}$ (also known as eigenvoice[126] and eigenchannel [127] matrices) define the speaker and session subspaces with $d_{\mathbf{s}}, d_{\mathbf{n}} \ll D$; $\boldsymbol{\lambda}_{\mathbf{s}}$ and $\boldsymbol{\lambda}_{\mathbf{n}}$ are $d_{\mathbf{s}}$ and $d_{\mathbf{n}}$-dimensional vectors representing the speaker and session factors respectively; $\mathbf{D}$ is a diagonal matrix and $\mathbf{D}\boldsymbol{\lambda}_z$ models the residual such that JFA can asymptotically converge to the speaker-dependent model if there is enough speaker data. To apply JFA to speaker recognition, the speaker and session subspaces ($\mathbf{M}_{\mathbf{s}}$ and $\mathbf{M}_{\mathbf{n}}$) are first estimated on appropriately labelled training corpora, and then for each test utterance, $\boldsymbol{\lambda}_{\mathbf{s}}$ and $\boldsymbol{\lambda}_{\mathbf{n}}$ are estimated by maximising likelihood. The feature vector of test utterances are scored against the session compensated speaker-adapted GMM whose supervector is given by $\boldsymbol{\mu} - \mathbf{M}_{\mathbf{n}}\boldsymbol{\lambda}_{\mathbf{n}}$. JFA relies on the assumption that $\mathbf{M}_{\mathbf{s}}$ and $\mathbf{M}_{\mathbf{n}}$ expand different subspaces so that maximising the likelihood with respect to $\boldsymbol{\lambda}_{\mathbf{s}}$ can only explain the speaker variability. However, as $\mathbf{M}_{\mathbf{s}}$ and $\mathbf{M}_{\mathbf{n}}$ are in general not orthogonal, it is still possible to move the speaker factor $\boldsymbol{\lambda}_{\mathbf{s}}$ to compensate the session variability, and thus allows session information to be contained in speaker factors. This is also observed in [40]. Using sub-spaces to represent different factors has also been applied to emotional speech synthesis [141]. Acoustic factorisation in [141] allows a flexible control of the speaker and emotion factors to synthesis natural and emotional voices. In section 6.3, a similar scheme is applied to the speaker and noise factorisation problem for speech recognition, where the subspaces are constrained to be orthogonal. This orthogonal subspace condition is derived by an explicit independence constraint between factor transforms, which will be developed and discussed in section 5.2.3.

5.2.2 Transform Constrained Approach

As discussed in the beginning of this chapter, it is assumed that acoustic factors impact the speech signal *differently* and *independently*. Intuitively, if the impact (or distortion) of each acoustic factor can be summarised in a functional form such as a mismatch function, it should be possible to use predictive transforms to model the impact of acoustic factors. Maximum likelihood estimation of transform parameters using Eqs (5.4-5.5) amounts to fitting the data to the distortion model. It is thus sensible to expect the estimated parameters to reflect the nature of that acoustic factor and thus be independent of each other.

This idea can be illustrated using a speaker and noise factorisation problem. Consider a speaker-independent and noise-free speech vector $\boldsymbol{x}_t$ drawn from a canonical model

$\mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_{\mathtt{x}}, \boldsymbol{\Sigma}_{\mathtt{x}})$. The distortion caused by i-th speaker and j-th noise condition is assumed to be exactly modelled by the following mismatch function:

$$\begin{aligned} \boldsymbol{y}_t^{(i,j)} &= \mathbf{f}(\boldsymbol{x}_t + \boldsymbol{b}_{\mathtt{s}}^{(i)}, \boldsymbol{n}_t^{(j)}) \\ &= \mathbf{C} \log \left(\exp(\mathbf{C}^{-1}(\boldsymbol{x}_t + \boldsymbol{b}_{\mathtt{s}}^{(i)})) + \exp(\mathbf{C}^{-1}\boldsymbol{n}_t^{(j)}) \right) \end{aligned} \tag{5.10}$$

where $\boldsymbol{y}_t^{(i,j)}$ is the vector observed in the acoustic condition; $(\mathtt{s}_i, \mathtt{n}_j)$, $\boldsymbol{b}_{\mathtt{s}}^{(i)}$ is the i-th speaker bias; $\boldsymbol{n}_t^{(j)}$ is the additive noise generated in the j-th noise condition, which is assumed to follow a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{\mathtt{n}}^{(j)}, \boldsymbol{\Sigma}_{\mathtt{n}}^{(j)})$; $\mathbf{C}$ is the DCT matrix, and $\boldsymbol{y} = \mathbf{f}(\boldsymbol{x}, \boldsymbol{n})$ describes how the noise vector $\boldsymbol{n}$ distorts the clean speech $\boldsymbol{x}$, resulting in the noisy speech vector $\boldsymbol{y}$ in the cesptral domain. Note this mismatch function is similar to Eq. (3.33), while the channel distortion is ignored here to concentrate on the effect of speaker and additive noise.

By combining the VTS transform and the speaker bias transform, a model transform which maps the canonical model $\mathcal{M}_{\mathtt{c}} = \mathcal{N}(\boldsymbol{\mu}_{\mathtt{x}}, \boldsymbol{\Sigma}_{\mathtt{x}})$ to the adapted model $\mathcal{M}^{(i,j)} = \mathcal{N}(\boldsymbol{\mu}_{\mathtt{y}}^{(i,j)}, \boldsymbol{\Sigma}_{\mathtt{y}}^{(i,j)})$ can be derived:

$$\begin{aligned} \boldsymbol{\mu}_{\mathtt{y}}^{(i,j)} &= \mathbf{f}(\boldsymbol{\mu}_{\mathtt{x}} + \boldsymbol{b}_{\mathtt{s}}^{(i)}, \boldsymbol{\mu}_{\mathtt{n}}^{(j)}) \\ \boldsymbol{\Sigma}_{\mathtt{y}}^{(i,j)} &= \mathbf{J}_{\mathtt{x}} \boldsymbol{\Sigma}_{\mathtt{x}} \mathbf{J}_{\mathtt{x}}^{\mathsf{T}} + \mathbf{J}_{\mathtt{n}} \boldsymbol{\Sigma}_{\mathtt{n}}^{(j)} \mathbf{J}_{\mathtt{n}}^{\mathsf{T}} \end{aligned} \tag{5.11}$$

where $\mathbf{J}_{\mathtt{x}}$ and $\mathbf{J}_{\mathtt{n}}$ are the Jacobian matrices,

$$\mathbf{J}_{\mathtt{x}} = \frac{\partial \boldsymbol{y}_t^{(i,j)}}{\partial \boldsymbol{x}_t}\Big|_{(\boldsymbol{\mu}_{\mathtt{x}} + \boldsymbol{b}_{\mathtt{s}}^{(i)}, \boldsymbol{\mu}_{\mathtt{n}}^{(j)})} \qquad \mathbf{J}_{\mathtt{n}} = \frac{\partial \boldsymbol{y}_t^{(i,j)}}{\partial \boldsymbol{n}_t}\Big|_{(\boldsymbol{\mu}_{\mathtt{x}} + \boldsymbol{b}_{\mathtt{s}}^{(i)}, \boldsymbol{\mu}_{\mathtt{n}}^{(j)})} \tag{5.12}$$

It can be seen from the compensation formula that the speaker transform $\mathcal{T}_{\mathtt{s}}^{(i)} = \boldsymbol{b}_{\mathtt{s}}^{(i)}$ linearly transforms the model parameters, while a nonlinear transform $\mathcal{T}_{\mathtt{n}}^{(j)}$ parameterised by $(\boldsymbol{\mu}_{\mathtt{n}}^{(j)}, \boldsymbol{\Sigma}_{\mathtt{n}}^{(j)})$ is used for the noise factor. Intuitively, as the model parameter adaptation function in Eq. (5.11) matches the mismatch function in Eq. (5.10) which is used in the generation process, and $\mathcal{T}_{\mathtt{s}}^{(i)}$ and $\mathcal{T}_{\mathtt{n}}^{(j)}$ modify model parameters in different ways (linear vs. nonlinear transforms) they should be "orthogonal" to each other. In fact, it is demonstrated in section 6.2 that the speaker bias estimator is (approximately) independent of the noise estimator, and vice versa. This is the desired factorisation property which ensures that estimating speaker bias parameters on the data from the same speaker under different noise condition will remain the same. Moreover, as the functional form of distortion, or the mismatch function, is known exactly in this example, the speaker bias estimator will converge to the true parameter used in the generation process.

In practice, only limited information about how an acoustic factor impacts speech signals is known. In the above speaker and noise factorisation example, the noise mismatch function is only an approximation while the impact of speaker differences is more complex than a

simple bias. However, it is reasonable to expect that combining a VTS transform (which is designed to model the effect of noise) and an adaptive linear transform (which has been proven to effectively model the speaker factor) can yield independence between transforms. As an application of this transform constrained approach, a speaker and noise factorisation scheme is developed in section 6.2. Results presented in section 8.1 will demonstrate that this transform constrained approach indeed yields a flexible yet effective factorised speaker and noise adaptation.

5.2.3 Explicitly Constrained Approach

The factorisation schemes presented in the previous sections rely on various assumptions to *implicitly* maintain the independence between factors: the data-constrained approach assumes that data is distributed over acoustic factors in a balanced manner, while the transform-constrained approach assumes that acoustic factors distort the speech signals in different ways, and the relationship between those factors and the observed speech vectors, usually summarised in a so called mismatch function, is known. However, in many applications, the data is not distributed in a balanced fashion, or the distortion cannot be summarised by a mismatch function. As an alternative to the implicit constraint approaches adopted in the previous sections, the interdependence between factor transforms is analysed mathematically in this section and an explicit constraint is given.

As discussed in section 5.1, factor transforms are normally ML estimated on their corresponding data. Taking the speaker and noise factorisation problem discussed in section 5.1 as an example, the factor transforms are iteratively optimised using the following steps:

1. Given the current noise factor transforms $\mathcal{T}_{\mathtt{n}}^{(1)}, \cdots, \mathcal{T}_{\mathtt{n}}^{(J)}$, update each of the speaker factor transforms by:

$$\hat{\mathcal{T}}_{\mathtt{s}}^{(i)} = \boldsymbol{h}_{\mathtt{s}_i}(\mathcal{T}_{\mathtt{n}}^{(1)}, \cdots, \mathcal{T}_{\mathtt{n}}^{(J)}) \qquad i = 1, \cdots, I \; ; \tag{5.13}$$

2. Given the current speaker factor transforms, $\mathcal{T}_{\mathtt{s}}^{(1)}, \cdots, \mathcal{T}_{\mathtt{s}}^{(I)}$, update each of the noise factor transform by:

$$\hat{\mathcal{T}}_{\mathtt{n}}^{(j)} = \boldsymbol{h}_{\mathtt{n}_j}(\mathcal{T}_{\mathtt{s}}^{(1)}, \cdots, \mathcal{T}_{\mathtt{s}}^{(I)}) \qquad j = 1, \cdots, J \; ; \tag{5.14}$$

3. Goto step 1 until converge;

Here, to emphasise that updated speaker (noise) factor transforms depend on the noise (speaker) factor transforms, $\boldsymbol{h}_{\mathtt{s}_i}$ and $\boldsymbol{h}_{\mathtt{n}_j}$ are defined as functions of the optimal speaker (noise) factor transforms given all the noise (speaker) factor transforms, i.e.,

$$\boldsymbol{h}_{\mathtt{s}_i}(\mathcal{T}_{\mathtt{n}}^{(1)}, \cdots, \mathcal{T}_{\mathtt{n}}^{(J)}) = \arg\max_{\mathcal{T}} \sum_{r:s_r=i} \mathcal{L}(\mathcal{O}^{(r)}; \mathcal{M}, \mathcal{T} \otimes \mathcal{T}_{\mathtt{n}}^{(n_r)}) \tag{5.15}$$

$$\boldsymbol{h}_{\mathtt{n}_j}(\mathcal{T}_{\mathtt{s}}^{(1)}, \cdots, \mathcal{T}_{\mathtt{s}}^{(I)}) = \arg\max_{\mathcal{T}} \sum_{r:n_r=j} \mathcal{L}(\mathcal{O}^{(r)}; \mathcal{M}, \mathcal{T}_{\mathtt{s}}^{(s_r)} \otimes \mathcal{T}) \tag{5.16}$$

where s_r and n_r are the speaker ($\mathtt{s}$) and noise ($\mathtt{n}$) index of utterance r respectively; $\mathcal{L}(\mathcal{O}^{(r)}; \mathcal{M}, \mathcal{T})$ is the log-likelihood function of $\mathcal{O}^{(r)}$ given the canonical model $\mathcal{M}_{\mathtt{c}}$ and the model transform $\mathcal{T}$. It is clear from the definition of $\boldsymbol{h}_{\mathtt{s}_i}$, an optimal speaker factor transform is a function of a set of noise factor transforms, which may break down the factorisation property in Eq. (5.3). In an extreme case, if there is only one noise source in a particular speaker's utterances or vice versa, it is not possible to separate one factor from the other.

To mitigate this problem, it is necessary to ensure the independence between factor transforms. The independence property means the optimal factor transforms do not change when other factor transforms vary. This requires the following condition of the function $\boldsymbol{h}_{\mathtt{s}_i}$ and $\boldsymbol{h}_{\mathtt{n}_j}$:

$$\frac{\partial \boldsymbol{h}_{\mathtt{s}_i}}{\partial \mathcal{T}_{\mathtt{n}}^{(j)}} = \mathbf{0} \quad , \quad \frac{\partial \boldsymbol{h}_{\mathtt{n}_j}}{\partial \mathcal{T}_{\mathtt{s}}^{(i)}} = \mathbf{0} \qquad \forall i, j \tag{5.17}$$

The independence constraint in Eq. (5.17) ensures the optimal factor transform will not be affected by other factor transforms, which allows it to change only when the corresponding acoustic factor changes. The condition in Eq. (5.17) implies that for any observed utterance $\mathcal{O}$, the optimal factor transforms $\mathcal{T}_{\mathtt{s}}^*$ and $\mathcal{T}_{\mathtt{n}}^*$ are independent of each other, where $\mathcal{T}_{\mathtt{s}}^* = \arg\max_{\mathcal{T}_{\mathtt{s}}} \mathcal{L}(\mathcal{O}; \mathcal{M}_{\mathtt{c}}, \mathcal{T}_{\mathtt{s}} \otimes \mathcal{T}_{\mathtt{n}})$ and $\mathcal{T}_{\mathtt{n}}^* = \arg\max_{\mathcal{T}_{\mathtt{n}}} \mathcal{L}(\mathcal{O}; \mathcal{M}_{\mathtt{c}}, \mathcal{T}_{\mathtt{s}} \otimes \mathcal{T}_{\mathtt{n}})$. This constraint can be translated into a constraint in the second order derivatives of the log-likelihood function as the following proposition demonstrates.

Proposition 1. *Let $\boldsymbol{x}^*(\boldsymbol{y})$ be the local maximiser of function $f(\boldsymbol{x}, \boldsymbol{y})$ given $\boldsymbol{y}$. Assuming the function f has its second order derivatives $\frac{\partial^2 f}{\partial \boldsymbol{x} \partial \boldsymbol{y}}$ defined everywhere, the derivatives of $\boldsymbol{x}^*$ with respect to $\boldsymbol{y}$ is*

$$\frac{\partial \boldsymbol{x}^*(\boldsymbol{y})}{\partial \boldsymbol{y}} = -\left(\frac{\partial^2 f}{\partial \boldsymbol{x} \partial \boldsymbol{x}}\right)^{-1} \frac{\partial^2 f}{\partial \boldsymbol{x} \partial \boldsymbol{y}}\Big|_{\boldsymbol{x}=\boldsymbol{x}^*(\boldsymbol{y})} \tag{5.18}$$

A sufficient condition for $\frac{\partial \boldsymbol{x}^*(\boldsymbol{y})}{\partial \boldsymbol{y}} = \mathbf{0}$ is :

$$\frac{\partial^2 f}{\partial \boldsymbol{x} \partial \boldsymbol{y}} = \mathbf{0} \quad \forall \boldsymbol{x}, \boldsymbol{y} \tag{5.19}$$

This condition also implies $\frac{\partial \boldsymbol{y}^*(\boldsymbol{x})}{\partial \boldsymbol{x}} = \mathbf{0}$.

Proof. Define another function $g(\boldsymbol{y}) = \frac{\partial f(\boldsymbol{x},\boldsymbol{y})}{\partial \boldsymbol{x}}|_{\boldsymbol{x}=\boldsymbol{x}^*(\boldsymbol{y})}$. On one hand,

$$\frac{\partial g(\boldsymbol{y})}{\partial \boldsymbol{y}} = \left[\frac{\partial^2 f}{\partial \boldsymbol{x}\partial \boldsymbol{y}} + \frac{\partial^2 f}{\partial \boldsymbol{x}\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{y}}\right]_{\boldsymbol{x}=\boldsymbol{x}^*(\boldsymbol{y})}$$

On the other hand, as $\boldsymbol{x}^*(\boldsymbol{y})$ is a local maximiser of $f(\cdot, \boldsymbol{y})$, thus $\frac{\partial^2 f}{\partial \boldsymbol{x}\partial \boldsymbol{x}}|_{\boldsymbol{x}=\boldsymbol{x}*(\boldsymbol{y})} \succ \mathbf{0}$ and

$$g(\boldsymbol{y}) \stackrel{\triangle}{=} \frac{\partial f(\boldsymbol{x}, \boldsymbol{y})}{\partial \boldsymbol{x}}|_{\boldsymbol{x}=\boldsymbol{x}^*(\boldsymbol{y})} = \mathbf{0}$$

Therefore, $\frac{\partial \boldsymbol{x}^*(\boldsymbol{y})}{\partial \boldsymbol{y}} = -\left(\frac{\partial^2 f}{\partial \boldsymbol{x}\partial \boldsymbol{x}}\right)^{-1}\frac{\partial^2 f}{\partial \boldsymbol{x}\partial \boldsymbol{y}}\Big|_{\boldsymbol{x}=\boldsymbol{x}^*(\boldsymbol{y})}$ and $\frac{\partial^2 f}{\partial \boldsymbol{x}\partial \boldsymbol{y}} = \mathbf{0}$ implies $\frac{\partial \boldsymbol{x}^*(\boldsymbol{y})}{\partial \boldsymbol{y}} = \mathbf{0}$. A similar argument can be made to establish $\frac{\partial^2 f}{\partial \boldsymbol{y}\partial \boldsymbol{x}} = \mathbf{0}$ implies $\frac{\partial \boldsymbol{y}^*(\boldsymbol{x})}{\partial \boldsymbol{x}} = \mathbf{0}$. □

Proposition 1 ensures that if the second order derivative of the log-likelihood function is zero everywhere, i.e.,

$$\frac{\partial^2 \mathcal{L}(\mathcal{O}; \mathcal{M}_{\mathtt{c}}, \mathcal{T}_{\mathtt{s}} \otimes \mathcal{T}_{\mathtt{n}})}{\partial \mathcal{T}_{\mathtt{s}} \partial \mathcal{T}_{\mathtt{n}}} = \mathbf{0} \tag{5.20}$$

the independence constraint in Eq. (5.3) is satisfied. The log-likelihood function is usually maximised via EM using the auxiliary function in the following form:

$$\mathcal{Q} = \sum_{m,t} \gamma_t^{(m)} \log p(\boldsymbol{o}_t; \boldsymbol{\mu}_{\mathtt{c}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)}, \mathcal{T}_{\mathtt{s}} \otimes \mathcal{T}_{\mathtt{n}}) \tag{5.21}$$

where $\boldsymbol{o}_t$ is the observation vector at time t, $\gamma_t^{(m)}$ is the posterior of $\boldsymbol{o}_t$ belonging to the m-th component, $\boldsymbol{\mu}_{\mathtt{c}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)}$ are the canonical mean and variance of component m. Assuming $\gamma_t^{(m)}$ does not vary with respect to the change of $\mathcal{T}_{\mathtt{s}}$ and $\mathcal{T}_{\mathtt{n}}$, the following constraint is used to enforce the independence:

$$\frac{\partial^2 \mathcal{Q}}{\partial \mathcal{T}_{\mathtt{s}} \partial \mathcal{T}_{\mathtt{n}}} = \mathbf{0} \tag{5.22}$$

It is worthwhile pointing out that though the independence is expressed as the second order derivative with respect to factor transforms, it is possible to achieve the property in Eq. (5.22) by enforcing constraints on the canonical model. For example, in section 6.3 where multiple subspaces are used in the canonical model to represent different acoustic factors, the subspaces are constrained to be orthogonal to ensure the property in Eq. (5.22) holds.

Compared with the first two approaches, this approach is based on the analysis of dependence of factor transforms and built on a mathematically sound ground, which ensures the factorisation property in Eq. (5.3) is satisfied. The downside is that the independence constraint complicates the transform parameter estimation. An application of this approach to speaker and noise factorisation is presented in section 6.3.

5.3 Summary

This chapter has presented the concept of acoustic factorisation and how it can be used in the model-based framework to robust speech recognition. In contrast to conventional schemes which use model transforms to model the combined effects of multiple acoustic factors, acoustic factorisation schemes separate the variations caused by various acoustic factors, each modelled by a single factor transform. This is a useful extension to the model-based framework for robust speech recognition in complex environments which has multiple acoustic factors simultaneously affecting the speech signals.

Constructing factor transforms which allows factorisation is the main focus of this chapter. The key requirement to achieve factorisation is to keep factor transforms independent of each other. Following this principle, three different approaches to constructing factor transforms are discussed. The first approach is to rely on balanced data to enforce the independence constraint. However, in many applications, especially in complex and rapidly changing environments, it is likely that the observed data will be highly unbalanced. The second approach is based on the fact that in some applications, the impact of acoustic factors on the observation vectors is approximately known, usually summarised in a mismatch function. Therefore it is possible to design transforms in different forms, each is tuned to a particular factor. As an acoustic factor has a different impact, the transforms will model different aspects of the observed signals, which means the interdependence between them is small. A simple speaker and noise factorisation example is used to demonstrate this. The first and the second approach require either data or the prior knowledge to implicitly keep the factor transforms independent. The third approach is designed for applications in which there is no balanced data and the impact of various acoustic factors cannot be easily summarised in a mismatch function. Based on the mathematical analysis of the interdependence between factor transforms, an explicit constraint to keep them "orthogonal" is proposed. The advantage of this approach is that it makes no assumption about the data or using specific form of factor transforms. In theory, it can be integrated into many factorised adaptation schemes.

This chapter presents the theory developed within the acoustic factorisation framework. In the next chapter, it will be applied to the speaker and noise factorisation problem. Using the transform constrained approach, a new adaptation scheme, "Joint", which combines the VTS and MLLR transforms are compared with a conventional approach, which treats the speaker and noise distortion as an combined effect. The explicit constrained approach is applied on CAT transforms to yield factor CAT (fCAT). Experimental evaluation will be presented in Chapter 8.

CHAPTER 6
Speaker and Noise Factorisation

In the previous chapter, the acoustic factorisation framework for robust speech recognition was presented and several approaches to constructing factorised transforms were discussed. As an application of this framework, this chapter will consider speaker and background noise factorisation for speech recognition. Speaker differences and background noise are two major factors which significantly impact the performance of speech recognition systems. A large amount of research has been devoted to combat the detrimental effects of these two factors. Some of the most widely used techniques were reviewed in Chapter 3. Most of them focused on compensating for the effects of a single acoustic factor, either speaker or noise. It is possible to combine techniques of speaker adaptation and noise compensation to yield speaker and noise compensation schemes. This will be discussed in section 6.1. The transform constrained approach is applied to speaker and noise factorisation in section 6.2: two standard techniques, the model-based VTS transform for noise compensation, which is a nonlinear transform, and the MLLR transform for speaker adaptation, which is a linear transform, are combined in an appropriate order. This yields the "Joint" speaker and noise factorisation scheme. A simple example is also used to illustrate why this yields factorisation in section 6.2. The transform constrained approach uses prior knowledge of the acoustic factors to build acoustic factorised

schemes. However, this knowledge may not be available in many applications. In this case, the explicit constrained approach can be used to construct acoustic factorisation schemes. In section 6.3, a general linear transform, fCAT, is used for speaker and noise compensation, while an explicit constraint is imposed on the model parameters. It is shown that due to this constraint, speaker and noise transforms are "orthogonal" to each other.

6.1 Speaker and Noise compensation

In section 3.2 and 3.3, speaker adaptation and noise robustness techniques have been discussed. It is possible to combine them to adapt the speech recognition systems to both speaker and noise factors. There are generally two approaches in the literature for joint speaker and environment adaptation. The first one is to use feature enhancement techniques to denoise the observation before back-end model adaptation [31]. The other approach, discussed in [66], is a full model-based approach: acoustic models are first compensated for the effect of noise using a predictive transform, then linear transform-based adaptation can be performed to reduce the residual mismatch, including those caused by speaker differences. Both approaches are designed to model the combined the effects of speaker and noise factors, instead of separating them. This section presents a model-based speaker and noise compensation scheme, in which a predictive transform, model-based VTS, is combined with an adaptive transform, MLLR, thus it is referred to as "VTS-MLLR".

6.1.1 Speaker and Noise Factors

Before presenting the speaker and noise compensation schemes, it is useful to briefly review the impact of speaker and noise factors on the observed feature vectors. Additive and convolutional noise corrupt "clean" speech, resulting in the noisy, observed, speech. In the Mel-cepstrum domain, the mismatch function relating the clean speech static $\boldsymbol{x}^{\mathtt{s}}$ and the noisy speech static $\boldsymbol{y}^{\mathtt{s}}$ is approximated by:

$$\begin{aligned}\boldsymbol{y}^{\mathtt{s}} &= \boldsymbol{x}^{\mathtt{s}} + \boldsymbol{h} + \mathbf{C}\log\left(\mathbf{1} + \exp\left(\mathbf{C}^{-1}(\boldsymbol{n}^{\mathtt{s}} - \boldsymbol{x}^{\mathtt{s}} - \boldsymbol{h})\right)\right) \\ &= \mathbf{f}(\boldsymbol{x}^{\mathtt{s}}, \boldsymbol{h}, \boldsymbol{n}^{\mathtt{s}})\,, \end{aligned} \tag{6.1}$$

where $\mathtt{s}$ is used to denote static parameters, $\boldsymbol{n}^{\mathtt{s}}$ and $\boldsymbol{h}$ are the static additive noise and convolutional noise, respectively, and $\mathbf{C}$ is the DCT matrix. It is assumed that for the j-th noise condition: $\boldsymbol{n}$ is Gaussian distributed with a mean $\boldsymbol{\mu}_{\mathtt{n}}^{(j)}$ and a diagonal covariance $\boldsymbol{\Sigma}_{\mathtt{n}}^{(j)} = \mathrm{diag}(\boldsymbol{\Sigma}_{\mathtt{sn}}^{(j)}, \boldsymbol{\Sigma}_{\Delta\mathtt{n}}^{(j)}, \boldsymbol{\Sigma}_{\Delta^2\mathtt{n}}^{(j)})$; $\boldsymbol{h} = \boldsymbol{\mu}_{\mathtt{h}}^{(j)}$ is an unknown constant variable. Model-based VTS

compensation [5, 153] approximates the mismatch function by a first-order vector Taylor series, expanded at $(\boldsymbol{\mu}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{h}}^{(j)}, \boldsymbol{\mu}_{\mathtt{n}}^{(j)})$ [1] for each component m. Under this approximation,

$$p(\boldsymbol{y}^{\mathtt{s}}|m, n_r = j) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{\mathtt{vts,sy}}^{(mj)}, \boldsymbol{\Sigma}_{\mathtt{vts,sy}}^{(mj)}) \tag{6.2}$$

where the compensated mean $\boldsymbol{\mu}_{\mathtt{vts,sy}}^{(mj)}$ and covariance matrix $\boldsymbol{\Sigma}_{\mathtt{vts,sy}}^{(mj)}$ are given by:

$$\begin{aligned} \boldsymbol{\mu}_{\mathtt{vts,sy}}^{(mj)} &= \mathbf{f}(\boldsymbol{\mu}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{h}}^{(j)}, \boldsymbol{\mu}_{\mathtt{n}}^{(j)}) \,, \\ \boldsymbol{\Sigma}_{\mathtt{vts,sy}}^{(mj)} &= \mathrm{diag}\left(\mathbf{J}_{\mathtt{x}}^{(mj)} \boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)} \mathbf{J}_{\mathtt{x}}^{(mj)\mathsf{T}} + \mathbf{J}_{\mathtt{n}}^{(mj)} \boldsymbol{\Sigma}_{\mathtt{sn}}^{(j)} \mathbf{J}_{\mathtt{n}}^{(mj)\mathsf{T}}\right) \end{aligned} \tag{6.3}$$

and $\boldsymbol{\mu}_{\mathtt{sx}}^{(m)}$ and $\boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}$ are the static mean and covariance of component m, $\mathbf{J}_{\mathtt{x}}^{(mj)}$ and $\mathbf{J}_{\mathtt{n}}^{(mj)}$ are the derivatives of $\boldsymbol{y}^{\mathtt{s}}$ with respect to $\boldsymbol{x}^{\mathtt{s}}$ and $\boldsymbol{n}^{\mathtt{s}}$ respectively, evaluated at $\boldsymbol{\mu}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{h}}^{(j)}, \boldsymbol{\mu}_{\mathtt{n}}^{(j)}$. With the continuous time approximation [86], the delta parameters under VTS compensation scheme are compensated by:

$$\begin{aligned} \boldsymbol{\mu}_{\mathtt{vts,\Delta y}}^{(mj)} &= \mathbf{J}_{\mathtt{x}}^{(mj)} \boldsymbol{\mu}_{\mathtt{\Delta x}}^{(m)} \,, \\ \boldsymbol{\Sigma}_{\mathtt{vts,\Delta y}}^{(mj)} &= \mathrm{diag}\left(\mathbf{J}_{\mathtt{x}}^{(mj)} \boldsymbol{\Sigma}_{\mathtt{\Delta x}}^{(m)} \mathbf{J}_{\mathtt{x}}^{(mj)\mathsf{T}} + \mathbf{J}_{\mathtt{n}}^{(mj)} \boldsymbol{\Sigma}_{\mathtt{\Delta n}}^{(j)} \mathbf{J}_{\mathtt{n}}^{(mj)\mathsf{T}}\right) \end{aligned} \tag{6.4}$$

where $\boldsymbol{\mu}_{\mathtt{\Delta x}}^{(m)}$ and $\boldsymbol{\Sigma}_{\mathtt{\Delta x}}^{(m)}$ are the mean and covariance matrix of clean delta parameters. The delta-delta parameters are compensated in a similar way. For notational convenience, only the delta parameters will be considered in the following.

To adapt the speaker independent model to the target speaker i, the MLLR mean transform [148] in the following form is often used:

$$\boldsymbol{\mu}^{(mi)} = \mathbf{A}^{(i)} \boldsymbol{\mu}^{(m)} + \boldsymbol{b}^{(i)}, \tag{6.5}$$

where $[\mathbf{A}^{(i)}, \boldsymbol{b}^{(i)}]$ is the linear transform for speaker i, $\boldsymbol{\mu}^{(m)}$ and $\boldsymbol{\mu}^{(mi)}$ the speaker independent and speaker dependent mean for the component m respectively.

6.1.2 VTS-MLLR

VTS-MLLR is a simple combination of two successful schemes in speaker adaptation and environment robustness. In this scheme, acoustic models are first compensated for the additive and channel distortion using a predictive transform, model-based VTS [5, 153]. This is followed by an adaptive transform, MLLR, applied for each speaker to reduce the residual mismatch, which includes the mismatch caused by speaker differences. Let $\boldsymbol{o}_t = [\boldsymbol{y}^{\mathtt{s}\mathsf{T}}, \mathtt{\Delta}\boldsymbol{y}^{\mathsf{T}}]^{\mathsf{T}}$ the observation vector, denote the noise model parameters for j-th noise condition as: $\boldsymbol{\Phi}^{(j)} =$

[1] As noise is assumed to be stationary $\boldsymbol{\mu}_{\mathtt{n}}$ is used to referred to the static noise mean instead.

$(\boldsymbol{\mu}_{\mathtt{n}}^{(j)}, \boldsymbol{\mu}_{\mathtt{h}}^{(j)}, \boldsymbol{\Sigma}_{\mathtt{n}}^{(j)})$. For the r-th utterance, the m-th Gaussian component is compensated in the following form:

$$p(\boldsymbol{o}_t|m, s_r, n_r) = \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_{\mathtt{o}}^{(mr)}, \boldsymbol{\Sigma}_{\mathtt{o}}^{(mr)}) \,. \tag{6.6}$$

where

$$\boldsymbol{\mu}_{\mathtt{o}}^{(mr)} = \begin{bmatrix} \boldsymbol{\mu}_{\mathtt{sy}}^{(mr)} \\ \boldsymbol{\mu}_{\Delta\mathtt{y}}^{(mr)} \end{bmatrix}, \quad \boldsymbol{\Sigma}_{\mathtt{o}}^{(mr)} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathtt{sy}}^{(mr)} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{\Delta\mathtt{y}}^{(mr)} \end{bmatrix} \tag{6.7}$$

Note s_r and n_r are the indices of speaker and environment condition of the r-th utterance. In this thesis, a block-diagonal MLLR [1] is used to linearly transform the VTS-compensated model parameters, i.e.,

$$\begin{aligned} \boldsymbol{\mu}_{\mathtt{sy}}^{(mn_r)} &= \mathbf{A}_{\mathtt{s}}^{(s_r)} \boldsymbol{\mu}_{\mathtt{vts,sy}}^{(mn_r)} + \boldsymbol{b}_{\mathtt{s}}^{(s_r)} \\ \boldsymbol{\mu}_{\Delta\mathtt{y}}^{(mn_r)} &= \mathbf{A}_{\Delta}^{(s_r)} \boldsymbol{\mu}_{\mathtt{vts},\Delta\mathtt{y}}^{(mn_r)} + \boldsymbol{b}_{\Delta}^{(s_r)} \end{aligned} \tag{6.8}$$

and

$$\boldsymbol{\Sigma}_{\mathtt{sy}}^{(mn_r)} = \boldsymbol{\Sigma}_{\mathtt{vts,sy}}^{(mn_r)}, \qquad \boldsymbol{\Sigma}_{\Delta\mathtt{y}}^{(mn_r)} = \boldsymbol{\Sigma}_{\mathtt{vts},\Delta\mathtt{y}}^{(mn_r)} \tag{6.9}$$

Here, $\mathbf{W}_{\mathtt{s}}^{(i)} = [\mathbf{A}_{\mathtt{s}}^{(i)}, \boldsymbol{b}_{\mathtt{s}}^{(i)}]$ and $\mathbf{W}_{\Delta}^{(i)} = [\mathbf{A}_{\Delta}^{(i)}, \boldsymbol{b}_{\Delta}^{(i)}]$ are the i-th speaker linear transform for the static and delta features, respectively. The combined MLLR transform will be written as $\mathbf{K}^{(i)} = (\mathbf{W}_{\mathtt{s}}^{(i)}, \mathbf{W}_{\Delta}^{(i)})$. The VTS compensated means and covariance matrices for the m-th component in the j-th noise condition $\boldsymbol{\mu}_{\mathtt{vts,y}}^{(mj)}, \boldsymbol{\Sigma}_{\mathtt{vts},\Delta\mathtt{y}}^{(mj)}$ are given in Eqs. (6.3 - 6.4).

For notational convenience, the following discussion will only consider a single speaker; therefore the speaker index i is omitted. The extension to multiple speakers is straightforward. Without loss of generality, it is also assumed that each utterance has a unique noise condition, i.e., $r = n_r$.

6.1.3 Transform Estimation

There are two sets of transform parameters to be estimated in the VTS-MLLR schemes: the linear transform $\mathbf{K}$ and the noise model parameters $\boldsymbol{\Phi} = \{\boldsymbol{\Phi}^{(r)}\}$, where $\boldsymbol{\Phi}^{(r)}$ is the noise model parameters of r-th noise utterance , $\boldsymbol{\Phi}^{(r)} = (\boldsymbol{\mu}_{\mathtt{n}}^{(r)}, \boldsymbol{\mu}_{\mathtt{h}}^{(r)}, \boldsymbol{\Sigma}_{\mathtt{n}}^{(r)})$. These parameters can be optimised using EM. This yields the following auxiliary function the VTS-MLLR compensation scheme

$$\mathcal{Q}(\mathbf{K}, \boldsymbol{\Phi}) = \sum_{r,m,t} \gamma_t^{(mr)} \log \mathcal{N}(\boldsymbol{o}_t^{(r)}; \boldsymbol{\mu}_{\mathtt{o}}^{(mr)}, \boldsymbol{\Sigma}_{\mathtt{o}}^{(mr)}) \,, \tag{6.10}$$

[1] It is possible to use full-transforms, however in this work to be consistent with the factorisation approach only block-diagonal transforms are considered.

where the summation over r involves all the utterances belonging to the same speaker, $\gamma_t^{(mr)}$ is the posterior probability of component m at time t of the r-th utterance given the current transform parameters $(\hat{\mathbf{K}}, \hat{\mathbf{\Phi}})$, $\boldsymbol{o}_t^{(r)}$ is the t-th observation vector of the r-th utterance.

To estimate $\mathbf{K}$ and $\mathbf{\Phi}$ for the "VTS-MLLR" scheme, a *block coordinate descent* strategy is adopted: first, for each speaker, $\mathbf{K}$ is initialised as $[\mathbf{I}, \mathbf{0}]$, and $\mathbf{\Phi}$ as the standard VTS-based noise estimates for each utterance; then $\mathbf{K}$ is optimised at the speaker level while keeping the noise model parameter fixed at the current noise estimates $\hat{\mathbf{\Phi}}$; finally, given the updated speaker transform, $\hat{\mathbf{K}}$, the noise parameter $\mathbf{\Phi}$ is re-estimated. This process is repeated N_{EM} times.

As the VTS-compensated static and dynamic parameters are transformed independently by $\mathbf{W}_{\mathtt{s}}$ and $\mathbf{W}_{\Delta}$ respectively, the estimation of $\mathbf{K} = (\mathbf{W}_{\mathtt{s}}, \mathbf{W}_{\Delta})$ can be done separately. Given the noise estimates for each utterance, $\mathbf{\Phi}^{(r)}$, the transform $\mathbf{W}_{\mathtt{s}}$ needs to be estimated at the speaker level, involving multiple utterances thus associated with different noise conditions. The transform estimation statistics in [65] are modified to reflect the changing noise conditions:

$$\begin{aligned} \boldsymbol{k}_i &= \sum_r \sum_m \sum_t \frac{\gamma_t^{(mr)} y_{t,i}^{(r)}}{\sigma_{\mathtt{vts},i}^{(mr)2}} \boldsymbol{\xi}_{\mathtt{vts,sy}}^{(mr)}, \\ \mathbf{G}_i &= \sum_r \sum_m \frac{\gamma^{(mr)}}{\sigma_{\mathtt{vts},i}^{(mr)2}} \boldsymbol{\xi}_{\mathtt{vts,sy}}^{(mr)} \boldsymbol{\xi}_{\mathtt{vts,sy}}^{(mr)\mathsf{T}}, \end{aligned} \tag{6.11}$$

with $y_{t,i}^{(r)}$ being the i-th element of $\boldsymbol{y}_t^{(r)}$, $\boldsymbol{\xi}_{\mathtt{vts,sy}}^{(mr)} = [\boldsymbol{\mu}_{\mathtt{vts,sy}}^{(mr)\mathsf{T}}, 1]^{\mathsf{T}}$ and $\sigma_{\mathtt{vts},i}^{(mr)2}$ the i-th diagonal item of $\mathbf{\Sigma}_{\mathtt{vts,sy}}^{(mr)}$, $\gamma^{(mr)} = \sum_t \gamma_t^{(mr)}$. Given these statistics, the i-th row of $\mathbf{W}_{\mathtt{s}}$, $\mathbf{w}_i^{\mathsf{T}}$, is obtained by $\mathbf{w}_i^{\mathsf{T}} = \boldsymbol{k}_i^{\mathsf{T}} \mathbf{G}_i^{-1}$. Estimating of $\mathbf{W}_{\Delta}$ is done similarly.

Given the current linear speaker transform $\hat{\mathbf{K}}$, the parameters of the noise transform can be updated. This requires the noise estimation approaches in, for example [153, 158, 159] to be modified to reflect that the compensated model will have the speaker transform applied. To estimate the additive and convolutional noise mean, a first-order VTS approximation is made, e.g., the mean and covariance for the static feature are approximated as follows:

$$\begin{aligned} \boldsymbol{\mu}_{\mathtt{sy}}^{(mr)} \approx & \hat{\boldsymbol{\mu}}_{\mathtt{sy}}^{(mr)} + \hat{\mathbf{A}}_{\mathtt{s}} \hat{\mathbf{J}}_{\mathtt{h}}^{(mr)} (\boldsymbol{\mu}_{\mathtt{h}}^{(r)} - \hat{\boldsymbol{\mu}}_{\mathtt{h}}^{(r)}) + \hat{\mathbf{A}}_{\mathtt{s}} \hat{\mathbf{J}}_{\mathtt{n}}^{(mr)} (\boldsymbol{\mu}_{\mathtt{n}}^{(r)} - \hat{\boldsymbol{\mu}}_{\mathtt{n}}^{(r)}) \\ \mathbf{\Sigma}_{\mathtt{y}}^{(mr)} \approx & \text{diag}\left(\hat{\mathbf{J}}_{\mathtt{x}}^{(mr)} \mathbf{\Sigma}_{\mathtt{x}}^{(m)} \hat{\mathbf{J}}_{\mathtt{x}}^{(mr)\mathsf{T}} + \hat{\mathbf{J}}_{\mathtt{n}}^{(mr)} \mathbf{\Sigma}_{\mathtt{n}}^{(r)} \hat{\mathbf{J}}_{\mathtt{n}}^{(mr)\mathsf{T}} \right) \end{aligned} \tag{6.12}$$

where $\hat{\mathbf{J}}_{\mathtt{x}}^{(mr)}$,$\hat{\mathbf{J}}_{\mathtt{h}}^{(mr)}$, $\hat{\mathbf{J}}_{\mathtt{n}}^{(mr)}$ and $\hat{\boldsymbol{\mu}}_{\mathtt{y}}^{(mr)}$ are the Jacobian matrices and the compensated mean based on the current noise estimation $\hat{\boldsymbol{\mu}}_{\mathtt{h}}^{(r)}, \hat{\boldsymbol{\mu}}_{\mathtt{n}}^{(r)}$ and the current linear transform $\hat{\mathbf{K}}$. Because of this VTS approximation, the auxiliary is now a quadratic function of the noise means. Hence $\boldsymbol{\mu}_{\mathtt{h}}^{(r)}, \boldsymbol{\mu}_{\mathtt{n}}^{(r)}$ can be obtained via solving a linear equation, in a similar fashion to the

scheme described in [158]. After estimating the noise mean , the noise variances $\boldsymbol{\Sigma}_{\mathbf{n}}^{(r)}$ can be estimated using the second order method, in the same way as [159]. At each iteration a check that the auxiliary function increases is performed and the estimates backed-off if necessary [158][1].

6.2 Transform Constrained Factorisation

In VTS-MLLR, the speaker linear transform is applied on top of the noise-compensated models. This means that it will represent attributes of both speaker and noise factors, as the VTS compensated model will depend on the noise condition. Thus the VTS-MLLR scheme will not have the required factorisation attribute, i.e., the linear transform in VTS-MLLR does not solely represent the speaker characteristics. As discussed in chapter 5, one possible way to yield acoustic factorisation is to combine different forms of transforms which match the generation process for each acoustic factor. Here, as an example, VTS model compensation and MLLR speaker adaptation are combined in a different way than VTS-MLLR. This scheme is named as "Joint", in which the speaker transform is applied to the underlying "clean" speech model prior to the application of VTS. Intuitively, the MLLR transform should therefore not depend on the nature of the noise, thus will yield an adaptation scheme which allows factorisation.

6.2.1 "Joint" Speaker and Noise adaptation

As the speaker adaptation in Joint is applied to the clean speech models, this adaptation stage can be expressed for speaker i as

$$\boldsymbol{\mu}_{\mathbf{x}}^{(mi)} = \mathbf{A}^{(i)}\boldsymbol{\mu}_{\mathbf{x}}^{(m)} + \boldsymbol{b}^{(i)}, \quad \boldsymbol{\Sigma}_{\mathbf{x}}^{(sm)} = \boldsymbol{\Sigma}_{\mathbf{x}}^{(m)}, \tag{6.13}$$

where $\boldsymbol{\mu}_{\mathbf{x}}^{(mi)}$ and $\boldsymbol{\Sigma}_{\mathbf{x}}^{(sm)}$ are the compensated clean speech distribution parameters for component m of speaker i.

For standard VTS compensation, the compensation and Jacobian are based on the speaker independent distribution $\mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathbf{x}}^{(m)})$. For the Joint scheme these terms need to be based on the speaker compensated distribution $\mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}}^{(mi)}, \boldsymbol{\Sigma}_{\mathbf{x}}^{(mi)})$. Substituting the speaker dependent mean $\mathbf{W}_{\mathbf{s}}\boldsymbol{\xi}_{\mathbf{sx}}^{(m)}$ (for clarity of notation, the speaker index i will be dropped) into Eq. (6.3)

[1]Since the second order optimisation assumes the approximation in Eq. (6.12), there is no guarantee that the auxiliary function in Eq. (6.10) will be non-decreasing.

yields a new, "Joint", compensation scheme:

$$\begin{aligned}\boldsymbol{\mu}_{\mathtt{sy}}^{(mr)} &= \mathbf{f}(\mathbf{W}_{\mathtt{s}}\boldsymbol{\xi}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{h}}^{(r)}, \boldsymbol{\mu}_{\mathtt{n}}^{(r)}) \,, \\ \boldsymbol{\Sigma}_{\mathtt{sy}}^{(mr)} &= \text{diag}\left(\mathbf{J}_{\mathtt{x},\mathtt{w}}^{(mr)}\boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}\mathbf{J}_{\mathtt{x},\mathtt{w}}^{(mr)\mathsf{T}} + \mathbf{J}_{\mathtt{n},\mathtt{w}}^{(mr)}\boldsymbol{\Sigma}_{\mathtt{sn}}^{(r)}\mathbf{J}_{\mathtt{n},\mathtt{w}}^{(mr)\mathsf{T}}\right)\end{aligned} \tag{6.14}$$

where $\boldsymbol{\xi}_{\mathtt{sx}}^{(m)} = [\boldsymbol{\mu}_{\mathtt{sx}}^{(m)\mathsf{T}}, 1]^{\mathsf{T}}$, and

$$\mathbf{J}_{\mathtt{x},\mathtt{w}}^{(mr)} = \frac{\partial \boldsymbol{y}^{\mathtt{s}}}{\partial \boldsymbol{x}^{\mathtt{s}}}|_{\mathbf{W}_{\mathtt{s}}\boldsymbol{\xi}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{h}}^{(r)}, \boldsymbol{\mu}_{\mathtt{n}}^{(r)}}, \quad \mathbf{J}_{\mathtt{n},\mathtt{w}}^{(mr)} = \mathbf{I} - \mathbf{J}_{\mathtt{x},\mathtt{w}}^{(mr)} \,. \tag{6.15}$$

In this work, the MLLR mean transform is constrained to have a block diagonal structure, where the blocks correspond to the static and delta parameters. With this block diagonal structure [1] and the continuous time approximation, the compensated delta parameters are given by:

$$\begin{aligned}\boldsymbol{\mu}_{\Delta\mathtt{y}}^{(mr)} &= \mathbf{J}_{\mathtt{x},\mathtt{w}}^{(mr)}(\mathbf{A}_{\Delta}\boldsymbol{\mu}_{\Delta\mathtt{x}}^{(m)} + \boldsymbol{b}_{\Delta}) \,, \\ \boldsymbol{\Sigma}_{\Delta\mathtt{y}}^{(mr)} &= \text{diag}\left(\mathbf{J}_{\mathtt{x},\mathtt{w}}^{(mr)}\boldsymbol{\Sigma}_{\Delta\mathtt{x}}^{(m)}\mathbf{J}_{\mathtt{x},\mathtt{w}}^{(mr)\mathsf{T}} + \mathbf{J}_{\mathtt{n},\mathtt{w}}^{(mr)}\boldsymbol{\Sigma}_{\Delta\mathtt{n}}^{(r)}\mathbf{J}_{\mathtt{n},\mathtt{w}}^{(mr)\mathsf{T}}\right)\end{aligned} \tag{6.16}$$

where $\boldsymbol{\mu}_{\Delta\mathtt{x}}^{(m)}$, $\boldsymbol{\Sigma}_{\Delta\mathtt{x}}^{(m)}$ are the m-th component parameters for the clean delta features respectively, and $\boldsymbol{\Sigma}_{\Delta\mathtt{n}}^{(r)}$ is the variance of $\Delta\mathtt{n}$, the noise delta.

The above Joint scheme uses a speaker transform, $\mathbf{K} = (\mathbf{W}_{\mathtt{s}}, \mathbf{W}_{\Delta})$ to *explicitly* adapt the models to the target speaker. In contrast to the VTS-MLLR scheme, the speaker transform is applied *before* the noise transform.

6.2.2 Transform Estimation

Similar to the VTS-MLLR schemes, there are two sets of transform parameters to be estimated in the Joint scheme: the linear transform $\mathbf{K}$ and the noise model parameters $\boldsymbol{\Phi} = \{\boldsymbol{\Phi}^{(r)}\}$. These transform parameters are optimised to maximise the auxiliary function in Eq. (6.10) in the EM framework. Again, the block coordinate descent strategy is adopted. The speaker transform $\mathbf{K}$ is initialised as $[\mathbf{I}, \mathbf{0}]$ per speaker, and $\boldsymbol{\Phi}$ as the standard VTS-based noise estimates for each utterance. This is followed by alternately optimising one of $\mathbf{K}$ and $\{\boldsymbol{\Phi}^{(r)}\}$, while keeping the other fixed.

Estimating the noise parameters, given the current speaker transform $\hat{\mathbf{K}}$ is a simple extension of VTS-based noise estimation in [153, 158]: prior to the noise estimation, the clean speech mean is transformed to the speaker-dependent clean speech mean. However, estimating the speaker transform $\mathbf{K}$ is not straight-forward, since the transform is applied to the "clean" speech and then nonlinear VTS compensation applied. To address this non-linearity,

[1] It is possible to extend the theory to handle full transforms, however this is not addressed in this thesis.

a first-order vector Taylor series approximation can again be employed to express $\boldsymbol{\mu}_{\mathtt{y}}^{(mu)}$ and $\boldsymbol{\Sigma}_{\mathtt{y}}^{(mu)}$ as functions of the current, $\hat{\mathbf{K}}$, and new,$\mathbf{K}$, estimates of the speaker transform. For the static parameters, this can be expressed as

$$\begin{aligned}\boldsymbol{\mu}_{\mathtt{sy}}^{(mu)} &\approx \mathbf{f}(\hat{\mathbf{W}}_{\mathtt{s}}\boldsymbol{\xi}_{\mathtt{sx}}^{(m)}, \boldsymbol{\mu}_{\mathtt{h}}^{(r)}, \boldsymbol{\mu}_{\mathtt{n}}^{(r)}) + \mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)}(\mathbf{W}_{\mathtt{s}} - \hat{\mathbf{W}}_{\mathtt{s}})\boldsymbol{\xi}_{\mathtt{sx}}^{(m)} \\ \boldsymbol{\Sigma}_{\mathtt{sy}}^{(mr)} &\approx \mathrm{diag}\left(\mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)}\boldsymbol{\Sigma}_{\mathtt{sx}}^{(m)}\mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)\mathsf{T}} + \mathbf{J}_{\mathtt{n},\hat{\mathtt{w}}}^{(mr)}\boldsymbol{\Sigma}_{\mathtt{n}}^{(r)}\mathbf{J}_{\mathtt{n},\hat{\mathtt{w}}}^{(mr)\mathsf{T}}\right)\end{aligned} \tag{6.17}$$

while for the delta parameters,

$$\begin{aligned}\boldsymbol{\mu}_{\Delta\mathtt{y}}^{(mr)} &\approx \mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)}\mathbf{W}_{\Delta}\boldsymbol{\xi}_{\Delta\mathtt{x}}^{(m)} \\ \boldsymbol{\Sigma}_{\Delta\mathtt{y}}^{(mr)} &\approx \mathrm{diag}\left(\mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)}\boldsymbol{\Sigma}_{\Delta\mathtt{x}}^{(m)}\mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)\mathsf{T}} + \mathbf{J}_{\mathtt{n},\hat{\mathtt{w}}}^{(mr)}\boldsymbol{\Sigma}_{\Delta\mathtt{n}}^{(r)}\mathbf{J}_{\mathtt{n},\hat{\mathtt{w}}}^{(mr)\mathsf{T}}\right)\end{aligned} \tag{6.18}$$

Due to the approximation in Eq. (6.18), the optimisation of $\mathbf{W}_{\mathtt{s}}$ and $\mathbf{W}_{\Delta}$ again becomes two separate but similar problems. The estimation of $\mathbf{W}_{\mathtt{s}}$, given the current noise estimation $\hat{\boldsymbol{\Phi}}$ and the VTS approximation in Eq. (6.17), uses the following, approximate, auxiliary function (up to some constant term):

$$q(\mathbf{W}_{\mathtt{s}}; \hat{\mathbf{W}}_{\mathtt{s}}) = \sum_{r,m,t} \gamma_t^{(mr)} \log \mathcal{N}(\boldsymbol{z}_t^{(mr)}; \mathbf{W}\boldsymbol{\xi}_{\mathtt{sx}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{full}}^{(mr)}) \tag{6.19}$$

where

$$\begin{aligned}\boldsymbol{z}_t^{(mr)} &= \mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)-1}(\boldsymbol{y}_t^{\mathtt{s}(r)} - \hat{\boldsymbol{\mu}}_{\mathtt{sy}}^{(mr)} + \mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)}\hat{\mathbf{W}}_{\mathtt{s}}\boldsymbol{\xi}_{\mathtt{sx}}^{(m)}) \\ \boldsymbol{\Sigma}_{\mathtt{full}}^{(mu)} &= \mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mu)-1}\hat{\boldsymbol{\Sigma}}_{\mathtt{sy}}^{(mu)}\mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mu)-\mathsf{T}}\end{aligned}$$

and $\hat{\boldsymbol{\mu}}_{\mathtt{sy}}^{(mu)}$, $\hat{\boldsymbol{\Sigma}}_{\mathtt{sy}}^{mu)}$ are the compensated static model parameters using the current transforms $\hat{\mathbf{W}}_{\mathtt{s}}$ and $\hat{\boldsymbol{\Phi}}^{(r)}$. As $\boldsymbol{\Sigma}_{\mathtt{full}}^{(mr)}$ is a full matrix, this optimisation is equivalent to the MLLR estimation with full covariance matrices [225]. Let $\mathbf{p}_i^{(mr)\mathsf{T}}$ be the i-th row vector of $\boldsymbol{\Sigma}_{\mathtt{full}}^{(mr)-1}$, $p_{ij}^{(mr)}$ the j-th element of $\mathbf{p}_i^{(mr)}$, and

$$\begin{aligned}\boldsymbol{k}_i &= \sum_{r,m,t} \gamma_t^{(mr)}\mathbf{p}_i^{(mr)\mathsf{T}}\boldsymbol{z}_t^{(mr)}\boldsymbol{\xi}_{\mathtt{sx}}^{(m)} - \sum_{j\neq i}\mathbf{G}_{ij}\mathbf{w}_j\,, \\ \mathbf{G}_{ij} &= \sum_{m,r}\gamma^{(mr)}p_{ij}^{(mr)}\boldsymbol{\xi}_{\mathtt{sx}}^{(m)}\boldsymbol{\xi}_{\mathtt{sx}}^{(m)\mathsf{T}}\,.\end{aligned} \tag{6.20}$$

Differentiating the auxiliary with respect to $\mathbf{w}_i^{\mathsf{T}}$ yields

$$\frac{\partial q(\mathbf{W}_{\mathtt{s}}; \hat{\mathbf{W}}_{\mathtt{s}})}{\partial \mathbf{w}_i^{\mathsf{T}}} = -\mathbf{w}_i^{\mathsf{T}}\mathbf{G}_{ii} + \boldsymbol{k}_i^{\mathsf{T}}\,. \tag{6.21}$$

The update formula for $\mathbf{w}_i$ depends on all the other row vectors through $\boldsymbol{k}_i$. Thus an iterative procedure is required [225]: first $\mathbf{G}_{ij}$ is set as $\mathbf{0}$ for all $j \neq i$ to get an initial $\mathbf{w}_i$; then $\mathbf{w}_i$

and $\boldsymbol{k}_i$ are updated on a row-by-row basis. Normally, one or two passes through all the row vectors is sufficient.

For estimation of $\mathbf{W}_{\Delta}$, another auxiliary function is used:

$$q_{\Delta}(\mathbf{W}_{\Delta}; \hat{\mathbf{W}}_{\mathtt{s}}) = \sum_{r,m,t} \gamma_t^{(mr)} \log \mathcal{N}(\Delta \boldsymbol{z}_t^{(mr)}; \mathbf{W}_{\Delta} \boldsymbol{\xi}_{\Delta \mathtt{x}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{full},\Delta}^{(mr)}) \tag{6.22}$$

where

$$\begin{aligned} \Delta \boldsymbol{z}_t^{(mr)} &= \mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)-1} \Delta \boldsymbol{y}_t^{(r)} \\ \boldsymbol{\Sigma}_{\mathtt{full},\Delta}^{(mr)} &= \mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)-1} \hat{\boldsymbol{\Sigma}}_{\Delta \mathtt{y}}^{(mr)} \mathbf{J}_{\mathtt{x},\hat{\mathtt{w}}}^{(mr)-\mathsf{T}} . \end{aligned}$$

This has the same form as the auxiliary function in Eq. (6.19). Thus the same procedure can be applied to estimate $\mathbf{W}_{\Delta}$.

As a first-order approximation, Eq. (6.17), is used to derive the approximate auxiliary functions. Optimising $\mathbf{K}$ via $q(\mathbf{W}_{\mathtt{s}}; \hat{\mathbf{W}}_{\mathtt{s}})$ and $q_{\Delta}(\mathbf{W}_{\Delta}; \hat{\mathbf{W}}_{\mathtt{s}})$ is not guaranteed to increase $\mathcal{Q}(\mathbf{K}, \hat{\boldsymbol{\Phi}})$ or the log-likelihood of the adaptation data. To address this problem, a simple back-off approach similar to the one used in [158], is adopted in this work. Note the back-off approach, i.e., step 3 in the following procedure, guarantees that the auxiliary function is non-decreasing. The estimation of the Joint speaker transform is thus:

1. Collect sufficient statistics $\boldsymbol{k}_i$ and $\mathbf{G}_{ij}$ based on the current transform $\hat{\mathbf{W}}_{\mathtt{s}}$ and $\hat{\boldsymbol{\Phi}}$. Similar statistics are also collected for $\mathbf{W}_{\Delta}$.

2. Use the row-iteration method to find the $\check{\mathbf{K}} = (\check{\mathbf{W}}_{\mathtt{s}}, \check{\mathbf{W}}_{\Delta})$ such that

$$\begin{aligned} \check{\mathbf{W}}_{\mathtt{s}} &= \arg\max_{\mathbf{W}_{\mathtt{s}}} q(\mathbf{W}_{\mathtt{s}}; \hat{\mathbf{W}}_{\mathtt{s}}) \\ \check{\mathbf{W}}_{\Delta} &= \arg\max_{\mathbf{W}_{\Delta}} q_{\Delta}(\mathbf{W}_{\Delta}; \hat{\mathbf{W}}_{\mathtt{s}}) \end{aligned}$$

3. Find $\alpha \in [0, 1]$, such that $\mathbf{K} = \alpha \hat{\mathbf{K}} + (1 - \alpha) \check{\mathbf{K}}$ satisfy $\mathcal{Q}(\mathbf{K}, \hat{\boldsymbol{\Phi}}) \geq \mathcal{Q}(\hat{\mathbf{K}}, \hat{\boldsymbol{\Phi}})$

4. Update current estimate $\hat{\mathbf{K}} \leftarrow \mathbf{K}$, and go to step 1 N_q times. It is observed in the experiments that setting $N_q = 5$ is enough for the auxiliary to converge in most of the cases.

The above procedure allows the speaker transform to be estimated. The noise transforms can then be re-estimated and the whole process repeated. However it is worth noting that there is no unique optimal value for the speaker and noise transforms. There is no way to distinguish between the speaker bias, $\boldsymbol{b}$, from the convolutional noise mean, $\boldsymbol{\mu}_{\mathtt{h}}^{(r)}$. This is not an issue as the parameters of the speaker model are estimated given the set of noise parameters. This ensures that all the convolutional noise means are consistent with one another.

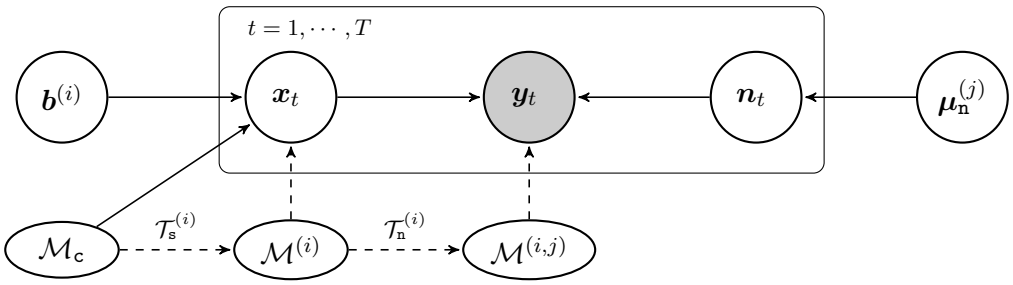


Figure 6.1: Generation of speaker and noise dependent observation vectors and corresponding model compensation. Solid lines show the DBN of generating observation vector $\boldsymbol{y}_t$, and the dashed lines represent the model compensation via transforms.

6.2.3 The Factorisation Property of Joint

The previous section presented the Joint scheme and its transform estimation algorithm. The speaker transform is applied prior to noise compensation. Intuitively, this will factorise the speaker and noise variabilities. In this section, the Joint scheme is applied to a very simple toy problem, where the functional form used to generate speaker and noise distorted observation vectors is assumed to be exactly known. It can be shown in this example ML speaker and noise parameter estimators are independent of each other, which ensures factorisation. Moreover, as the generation process is known exactly, the ML speaker (or noise) estimator will converge to the true speaker (or noise) parameters.

Consider the following generation process of an observation vector (MFCCs) of $\boldsymbol{y}_t \in \mathcal{R}^d$ by the i-th speaker in the j-th noise condition. Firstly, for the i-th speaker, a clean speech vector $\boldsymbol{x}_t$ is sampled from a Gaussian distribution, i.e.,

$$\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_{\mathtt{x}} + \boldsymbol{b}^{(i)}, \boldsymbol{\Sigma}_{\mathtt{x}}) \; ; \tag{6.29}$$

where $\mathcal{M}_{\mathtt{c}} = (\boldsymbol{\mu}_{\mathtt{x}}, \boldsymbol{\Sigma}_{\mathtt{x}})$ is the speaker-independent mean and covariance, which form the canonical model parameters; $\boldsymbol{b}^{(i)}$ is the bias vector associated with the i-th speaker. In the mean time, a noise vector is independently sampled from the j-th noise condition :

$$\boldsymbol{n}_t \sim \mathcal{N}(\boldsymbol{\mu}_{\mathtt{n}}^{(j)}, \boldsymbol{\Sigma}_{\mathtt{n}}); \tag{6.30}$$

The observation vector $\boldsymbol{y}_t$ is then generated according to a deterministic mismatch function, similar to the one introduced in section 3.3.2:

$$\boldsymbol{y}_t = \mathbf{f}(\boldsymbol{x}_t, \boldsymbol{n}_t) = \mathbf{C} \log \left(\exp(\mathbf{C}^{-1}\boldsymbol{x}_t) + \exp(\mathbf{C}^{-1}\boldsymbol{n}_t) \right) \tag{6.31}$$

Algorithm 6.1: ML estimation of speaker and noise transform parameters

Input: a set of vectors $\{\boldsymbol{y}_t | t = 1, \ldots, T\}$ observed in the acoustic condition $(\mathtt{s_i}, \mathtt{n_j})$.

Output: ML estimators, $\hat{\boldsymbol{b}}^{(i)}$ and $\hat{\boldsymbol{\mu}}_{\mathtt{n}}^{(j)}$ which maximise the likelihood of observed data

$\hat{\boldsymbol{b}}^{(i)}$ is initialised by $\boldsymbol{b}_0$ and $\hat{\boldsymbol{\mu}}_{\mathtt{n}}^{(j)}$ is initialised by $\boldsymbol{\mu}_0$

for $n_1 = 1, \ldots, N_1$ **do**

• Based on the current estimator, calculate $\mathbf{f}_0 = \mathbf{f}(\boldsymbol{\mu}_{\mathtt{x}} + \boldsymbol{b}_0, \boldsymbol{\mu}_0)$, $\overline{\boldsymbol{y}} = \frac{1}{T}\sum \boldsymbol{y}_t$ and the following matrices:

$$\mathbf{Z}_{\mathtt{by}} = (\mathbf{J}_{\mathtt{x0}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{x0}})^{-1} \mathbf{J}_{\mathtt{x0}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \tag{6.23}$$

$$\mathbf{Z}_{\mathtt{bn}} = -(\mathbf{J}_{\mathtt{x0}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{x0}})^{-1} \mathbf{J}_{\mathtt{x0}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{n0}} \tag{6.24}$$

$$\mathbf{Z}_{\mathtt{ny}} = (\mathbf{J}_{\mathtt{n0}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{n0}})^{-1} \mathbf{J}_{\mathtt{n0}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \tag{6.25}$$

$$\mathbf{Z}_{\mathtt{nb}} = -(\mathbf{J}_{\mathtt{n0}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{n0}})^{-1} \mathbf{J}_{\mathtt{n0}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{x0}} \tag{6.26}$$

where $\boldsymbol{\Sigma}_{\mathtt{y}}$ is the compensated covariance based on the current estimator, $\mathbf{J}_{\mathtt{x0}}$ and $\mathbf{J}_{\mathtt{n0}}$ are the Jacobian matrices calculated at current expansion point.

for $n_2 = 1, \ldots, N_2$ **do**

• Update $\hat{\boldsymbol{b}}^{(i)}$ by:

$$\hat{\boldsymbol{b}}^{(i)} \leftarrow \boldsymbol{b}_0 + \mathbf{Z}_{\mathtt{by}}(\overline{\boldsymbol{y}} - \mathbf{f}_0) + \mathbf{Z}_{\mathtt{bn}}(\hat{\boldsymbol{\mu}}_{\mathtt{n}}^{(j)} - \boldsymbol{\mu}_0) \tag{6.27}$$

• Update $\hat{\boldsymbol{\mu}}_{\mathtt{n}}^{(j)}$ by:

$$\hat{\boldsymbol{\mu}}_{\mathtt{n}}^{(j)} \leftarrow \boldsymbol{\mu}_0 + \mathbf{Z}_{\mathtt{ny}}(\overline{\boldsymbol{y}} - \mathbf{f}_0) + \mathbf{Z}_{\mathtt{ny}}(\hat{\boldsymbol{b}}^{(i)} - \boldsymbol{b}_0) \tag{6.28}$$

end

• Update expansion point: $\boldsymbol{b}_0 \leftarrow \hat{\boldsymbol{b}}^{(i)}, \boldsymbol{\mu}_0 \leftarrow \hat{\boldsymbol{\mu}}_{\mathtt{n}}^{(j)}$

end

This process is illustrated in Figure 6.1. For the purpose of illustration, the speaker-independent mean $\boldsymbol{\mu}_{\mathtt{x}}$ and all the variances are assumed to be known, while the speaker bias $\boldsymbol{b}^{(i)}$ and noise mean $\boldsymbol{\mu}_{\mathtt{n}}^{(j)}$ need to be estimated based on a set of observation $\{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t, \ldots, \boldsymbol{y}_T\}$. Denote the true parameters used in the generation process as $\boldsymbol{b}^*$ and $\boldsymbol{\mu}_{\mathtt{n}}^*$.

Using the knowledge of this generation process as a prior, it is possible to design model transforms to compensate the canonical model $\mathcal{M}_{\mathtt{c}}$. The canonical model $\mathcal{M}_{\mathtt{c}}$ is transformed via a speaker transform $\mathcal{T}_{\mathtt{s}}^{(i)}$ (which is a speaker bias transform) followed by a noise transform $\mathcal{T}_{\mathtt{n}}^{(j)}$ (which is a VTS transform presented in section 3.3.2). The compensated model $\mathcal{M}^{(i,j)}$ is again a Gaussian distribution, whose parameters $\boldsymbol{\mu}_{\mathtt{y}}^{(i,j)}, \boldsymbol{\Sigma}_{\mathtt{y}}^{(i,j)}$ are given by:

$$\begin{aligned} \boldsymbol{\mu}_{\mathtt{y}}^{(i,j)} &= \mathbf{f}(\boldsymbol{\mu}_{\mathtt{x}} + \boldsymbol{b}^{(i)}, \boldsymbol{\mu}_{\mathtt{n}}^{(j)}) \\ \boldsymbol{\Sigma}_{\mathtt{y}}^{(i,j)} &= \mathbf{J}_{\mathtt{x}} \boldsymbol{\Sigma}_{\mathtt{x}} \mathbf{J}_{\mathtt{x}}^{\mathsf{T}} + \mathbf{J}_{\mathtt{n}} \boldsymbol{\Sigma}_{\mathtt{n}} \mathbf{J}_{\mathtt{n}}^{\mathsf{T}} \end{aligned} \tag{6.32}$$

where $\mathbf{J}_{\mathtt{x}}$ and $\mathbf{J}_{\mathtt{n}}$ are the Jacobian matrices, $\mathbf{J}_{\mathtt{x}} = \mathbf{C} \cdot \mathrm{diag}(\boldsymbol{u}) \cdot \mathbf{C}^{-1}, \mathbf{J}_{\mathtt{x}} = \mathbf{I} - \mathbf{J}_{\mathtt{n}}$, $\mathbf{C}$ and $\mathbf{C}^{-1}$

are the DCT matrix and its inverse matrix, and $\boldsymbol{u}$ is a d_L-dimensional (d_L is the number of channels) vector whose q-th element is

$$u_q = \frac{e_{\mathtt{x},q}}{e_{\mathtt{x},q} + e_{\mathtt{n},q}} \tag{6.33}$$

$e_{\mathtt{x},q} = e^{\mathbf{c}_q^{-\mathsf{T}}(\boldsymbol{\mu}_\mathtt{x} + \boldsymbol{b}^{(i)})}$ and $e_{\mathtt{n},q} = e^{\mathbf{c}_q^{-\mathsf{T}}\boldsymbol{\mu}_\mathtt{n}^{(j)}}$ are the speech and noise energy in the q-th channel.

Similar to the noise estimation for VTS, the speaker and noise ML estimators, $\hat{\boldsymbol{\mu}}_\mathtt{s}^{(i)}$ and $\hat{\boldsymbol{\mu}}_\mathtt{n}^{(j)}$ can be obtained by the algorithm 6.1. Note in practice, it is usually sufficient to update the expansion point only once (e.g., [153]) . At a first glance of this algorithm, it appears that the ML estimators $\hat{\boldsymbol{b}}^{(i)}$ and $\hat{\boldsymbol{\mu}}_\mathtt{n}^{(j)}$ are interdependent via matrices $\mathbf{Z}_\mathtt{bn}$ and $\mathbf{Z}_\mathtt{nb}$. However, if the MAX approximation in [178] is made, i.e.,

$$e_{\mathtt{x},q} + e_{\mathtt{n},q} \approx \max(e_{\mathtt{x},q}, e_{\mathtt{n},q})$$

it can be shown that $\mathbf{Z}_\mathtt{bn}$ and $\mathbf{Z}_\mathtt{nb}$ are approximately zero matrices (see appendix B for details). The MAX approximation assumes that channel energy is dominated by either speech or noise, thus the speaker characteristics and noise distortion impact the observation differently. This means the variabilities caused by speaker and noise can be separated. For the example discussed in this section, there are a few interesting consequences from following the MAX assumption:

- The ML estimator of speaker parameters, $\hat{\boldsymbol{b}}^{(i)}$, is independent of the noise estimator and vice versa. This means the two estimators are decoupled. When the noise parameters used in the generation process change, it is not possible to adjust speaker parameters to fit the data better. Therefore, the speaker parameters are only used to explain the variations caused by the speaker factor.

- The ML estimator of speaker parameters, $\hat{\boldsymbol{b}}^{(i)}$ will asymptotically converge to the true speaker parameters, $\boldsymbol{b}^*$, which is used to generate $\{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_T\}$, provided that the initialisation is sufficiently good. This can be seen from the following fact:

$$\begin{aligned}
\mathcal{E}\{\hat{\boldsymbol{b}}^{(i)}\} &= \boldsymbol{b}_0 + \mathbf{Z}_\mathtt{by}(\mathcal{E}\{\overline{\boldsymbol{y}}\} - \mathbf{f}_0) \\
&\approx \boldsymbol{b}_0 + \mathbf{Z}_\mathtt{by}(\mathbf{f}(\boldsymbol{\mu}_\mathtt{x} + \boldsymbol{b}^*, \boldsymbol{\mu}_\mathtt{n}^*) - \mathbf{f}(\boldsymbol{\mu}_\mathtt{x} + \boldsymbol{b}_0, \boldsymbol{\mu}_0)) \\
&\approx \boldsymbol{b}_0 + \mathbf{Z}_\mathtt{by}\mathbf{J}_{\mathtt{x}0}(\boldsymbol{b}^* - \boldsymbol{b}_0) + \mathbf{Z}_\mathtt{by}\mathbf{J}_{\mathtt{n}0}(\boldsymbol{\mu}_\mathtt{n}^* - \boldsymbol{\mu}_0) \\
&\approx \boldsymbol{b}_0 + (\boldsymbol{b}^* - \boldsymbol{b}_0) = \boldsymbol{b}^*
\end{aligned}$$

  where the derivation to the last line used the approximation $\mathbf{Z}_\mathtt{by}\mathbf{J}_{\mathtt{n}0} = \mathbf{Z}_\mathtt{bn} \approx \mathbf{0}$, and $\mathbf{Z}_\mathtt{by}\mathbf{J}_{\mathtt{x}0} = \mathbf{I}$. Similar formulations can be derived for the noise estimator. Therefore, the

ML estimator $\hat{\boldsymbol{b}}^{(i)}$ derived from samples $\{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_T\}$ draw from the acoustic condition $(\mathtt{s}_i, \mathtt{n}_j)$ can be used for speaker adaptation in the other noisy environments, such as $(\mathtt{s}_i, \mathtt{n}_{j'})$.

It is also interesting to compare the causes of these two phenomena. The first is possible because speaker and noise factors have different impacts on the observations (induced by the MAX approximation), while the second one is because that the same functional form is used for data generation and model compensation. Because of the form of transforms used in this example, the speaker and noise transforms are "orthogonal" to each other, thus enabling acoustic factorisation. Inspired by this example, it is expected that the Joint adaptation scheme can be used in a factorised adaptation, since the Joint scheme combines the MLLR transform and VTS transforms in an order matching the generation process.

6.3 Explicitly Constrained Factorisation

In section 6.1, a conventional approach, VTS-MLLR is used for speaker and noise compensation, where no attempt is made to separate the speaker and noise variations. In section 6.2, a different scheme, Joint, is proposed in which the speaker and noise factors are modelled by specific model transforms. These model transforms are tuned to fit the variations caused by the corresponding factors. The acoustic factorisation property is thus achieved via implicit constraints.

In section 5.2.3, it was argued that acoustic factorisation can be achieved by imposing an explicit constraint. In this section, an explicit constraint is applied to yield factorised speaker and noise adaptation. It is also possible to apply the constraint to extend the previously proposed Joint scheme. However, as a proof of concept study, this work will assume the knowledge of the noise mismatch function is not known in advance and it will modify a simple linear transform, namely factored CAT (fCAT) [141], to evaluate the effectiveness of the proposed independence constraint for acoustic factorisation. fCAT is an extension of the standard cluster adaptive training (CAT) [70] or eigenvoice[136], which enables adapting the acoustic models to multiple factors. In fCAT, to compensate the variability of speaker and noise in the r-th utterance, the m-th Gaussian component is adapted by:

$$p(\boldsymbol{o}_t^{(r)}; \boldsymbol{\mu}_{\mathsf{c}}^{(m)}, \boldsymbol{\Sigma}_{\mathsf{c}}^{(m)}, m, s_r, n_r) = \mathcal{N}(\boldsymbol{o}_t^{(r)}; \boldsymbol{\mu}^{(mr)}, \boldsymbol{\Sigma}_{\mathsf{c}}^{(m)}) \tag{6.34}$$

where $\boldsymbol{o}_t^{(r)} \in \mathcal{R}^D$ is the t-th observation vector in r-th utterance, $\boldsymbol{\mu}_{\mathsf{c}}^{(m)}, \boldsymbol{\Sigma}_{\mathsf{c}}^{(m)}$ are the canonical mean and variance, $s_r \in \{1 \dots I\}$ and $n_r \in \{1 \dots J\}$ are the speaker and noise condition

indices of utterance r respectively and

$$\boldsymbol{\mu}^{(mr)} = \boldsymbol{\mu}_{\mathtt{c}}^{(m)} + \mathbf{M}_{\mathtt{s}}^{(m)} \boldsymbol{\lambda}_{\mathtt{s}}^{(s_r, q_m)} + \mathbf{M}_{\mathtt{n}}^{(m)} \boldsymbol{\lambda}_{\mathtt{n}}^{(n_r, p_m)} , \tag{6.35}$$

$\mathbf{M}_{\mathtt{s}}^{(m)} \in \mathcal{R}^{D \times d_{\mathtt{s}}}$, $\mathbf{M}_{\mathtt{n}}^{(m)} \in \mathcal{R}^{D \times d_{\mathtt{n}}}$ are the component m's speaker and noise cluster parameters, $d_{\mathtt{s}}$ and $d_{\mathtt{n}}$ are the number of speaker and noise clusters respectively; $q_m \in \{1, \ldots, Q\}$ ($p_m \in \{1, \ldots, P\}$) maps the component index m to the speaker (noise) regression class index; $\boldsymbol{\lambda}_{\mathtt{s}}^{(i,q)}$ is the speaker cluster weight vector associated with q-th speaker base class for i-th speaker; $\boldsymbol{\lambda}_{\mathtt{n}}^{(j,p)}$ is the noise cluster weight vector associated with r-th noise base class for j-th noise condition. Among these parameters, $\{\boldsymbol{\mu}_{\mathtt{c}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)}, \mathbf{M}_{\mathtt{s}}^{(m)}, \mathbf{M}_{\mathtt{n}}^{(m)}\}$ form the canonical model parameters $\mathcal{M}_{\mathtt{c}}$, while $\{\boldsymbol{\lambda}_{\mathtt{s}}^{(i,q)} | q = 1 \ldots Q\}$ and $\{\boldsymbol{\lambda}_{\mathtt{n}}^{(j,p)} | p = 1 \ldots P\}$ are the parameters of i-th speaker transform $\mathcal{T}_{\mathtt{s}}^{(i)}$ and j-th noise transform $\mathcal{T}_{\mathtt{n}}^{(j)}$ respectively. Note this modelling form is similar to the one used in JFA[127] for speaker recognition.

The canonical and factor transform parameter estimation is done with EM via maximising the following auxiliary function:

$$\mathcal{Q}(\mathcal{M}_{\mathtt{c}}, \{\mathcal{T}_{\mathtt{s}}^{(i)}\}, \{\mathcal{T}_{\mathtt{n}}^{(j)}\}) = \sum_{r,t,m} \gamma_t^{(mr)} \log\left(\boldsymbol{o}_t^{(r)}; \boldsymbol{\mu}^{(mr)}, \boldsymbol{\Sigma}^{(mr)}\right) \tag{6.36}$$

in which $\gamma_t^{(mr)}$ is the posterior probability of component m at time t in r-th utterance. In the normal fCAT, this auxiliary function is maximised without any constraint. As it is discussed in section 5.2.3, a sufficient condition to keep the optimal factor transforms independent of each other is:

$$\frac{\partial^2 \mathcal{Q}}{\partial \mathcal{T}_{\mathtt{s}}^{(i)} \partial \mathcal{T}_{\mathtt{n}}^{(j)}} = \sum_{r: s_r = i \wedge n_r = j} \sum_{t,m} \gamma_t^{(mr)} \mathbf{M}_{\mathtt{s}}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)-1} \mathbf{M}_{\mathtt{n}}^{(m)} = \mathbf{0} \quad \forall i, j \tag{6.37}$$

Hence a sufficient condition for the independence constraint in Eq. (6.37) to hold for every possible observation sequence is

$$\mathbf{M}_{\mathtt{s}}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)-1} \mathbf{M}_{\mathtt{n}}^{(m)} = \mathbf{0} \quad \forall m. \tag{6.38}$$

This constraint can be also expressed in another form. Define

$$\mathbf{M}_{\mathtt{s}} = \begin{bmatrix} \vdots \\ \mathbf{M}_{\mathtt{s}}^{(m)} \\ \vdots \end{bmatrix} \quad ; \quad \boldsymbol{\Sigma}_{\mathtt{c}} = \mathrm{diag}(\cdots, \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)}, \cdots) \; ; \quad \mathbf{M}_{\mathtt{n}} = \begin{bmatrix} \vdots \\ \mathbf{M}_{\mathtt{n}}^{(m)} \\ \vdots \end{bmatrix} \tag{6.39}$$

where $\mathbf{M}_{\mathtt{s}}$ and $\mathbf{M}_{\mathtt{n}}$ represent the speaker and noise subspaces. The independence constraint becomes

$$\mathbf{M}_{\mathtt{s}}^{\mathsf{T}} \boldsymbol{\Sigma}_c^{-1} \mathbf{M}_{\mathtt{n}} = \mathbf{0} \tag{6.40}$$

According to the above constraint, in a normed vector space induced by the inner product function $\boldsymbol{x} \cdot \boldsymbol{y} = \boldsymbol{x}^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{y}$, the speaker subspace $\mathbf{M}_{\mathtt{s}}$ is orthogonal to the noise subspace $\mathbf{M}_{\mathtt{n}}$. This orthogonality guarantees that for a given data point, there is a unique speaker-noise factorisation, thus it is possible to separate speaker factor even if there is only one data point.

6.3.1 Parameter Estimation

The fCAT model is trained to maximise the likelihood of training data which consists of various speaker and noise combinations. The canonical model parameter updates are very similar to those in [141]. The main difference is the constraint in Eq. (6.38) need to be maintained, thus updating the speaker subspace $\mathbf{M}_{\mathtt{s}}$ requires solving the following constrained optimisation:

$$\begin{aligned} &\max_{\mathbf{M}_{\mathtt{s}}} -\frac{1}{2}\mathrm{tr}\left(\mathbf{M}_{\mathtt{s}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{-1} \mathbf{M}_{\mathtt{s}} \mathbf{G}_{\mathtt{s}}\right) + \mathrm{tr}\left(\boldsymbol{\Sigma}_{\mathtt{c}}^{-1} \mathbf{M}_{\mathtt{s}} \mathbf{K}_{\mathtt{s}}\right) \\ &\text{s. t.} \quad \mathbf{M}_{\mathtt{s}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{-1} \mathbf{M}_{\mathtt{n}} = \mathbf{0} \end{aligned} \tag{6.41}$$

where

$$\mathbf{G}_{\mathtt{s}} = \mathrm{diag}(\cdots, \mathbf{G}_{\mathtt{s}}^{(\mathtt{m})}, \cdots) \qquad \mathbf{K}_{\mathtt{s}} = [\cdots, \mathbf{K}_{\mathtt{s}}^{(\mathtt{m})}, \cdots]$$

and

$$\begin{aligned} \mathbf{G}_{\mathtt{s}}^{(m)} &= \sum_{t,r} \gamma_t^{(mr)} \boldsymbol{\lambda}_{\mathtt{s}}^{(s_r, q_m)} \boldsymbol{\lambda}_{\mathtt{s}}^{(s_r, q_m)\mathsf{T}} \\ \mathbf{K}_{\mathtt{s}}^{(m)} &= \sum_{t,r} \gamma_t^{(mr)} \boldsymbol{\lambda}_{\mathtt{s}}^{(s_r, q_m)} (\boldsymbol{o}_t^{(r)} - \boldsymbol{\mu}_c^{(m)} - \mathbf{M}_{\mathtt{n}}^{(m)} \boldsymbol{\lambda}_{\mathtt{n}}^{(n_r, p_m)})^{\mathsf{T}} \end{aligned}$$

Using the method of Lagrange multipliers [29], it is shown in Appendix C that the solution is given by:

$$\mathbf{M}_{\mathtt{s}} = \left[\mathbf{I} - \mathbf{M}_{\mathtt{n}} \left(\mathbf{M}_{\mathtt{n}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{-1} \mathbf{M}_{\mathtt{n}}\right)^{-1} \mathbf{M}_{\mathtt{n}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{-1}\right] \mathbf{K}_{\mathtt{s}}^{\mathsf{T}} \mathbf{G}_{\mathtt{s}}^{-1} \tag{6.42}$$

Note $\mathbf{M}_{\mathtt{s}} = \mathbf{K}_{\mathtt{s}}^{\mathsf{T}} \mathbf{G}_{\mathtt{s}}^{-1}$ if the constraint is removed. It is also necessary to point out that since the speaker (or the noise) variability only lies in a subspace, its dimension d_s (or d_n) must be smaller than the full dimension, $M \cdot D$, where M is the number of Gaussian components in the system. Therefore $\mathbf{M}_{\mathtt{n}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{-1} \mathbf{M}_{\mathtt{n}}$ is a full-rank matrix with the size $d_s \times d_s$, thus it is always invertible. A similar equation is adopted for re-estimating for the noise subspace $\mathbf{M}_{\mathtt{n}}$.

There are three main stages involved to train a fCAT model. In the first stage, speaker and noise transforms are initialised. It is assumed that the speaker label and the noise type label are known for each utterance. In this work, the eigen-decomposition[136] approach

was used for speaker transforms initialisation, while the noise transforms were initialised by a one-of-K vectors, with the corresponding noise type weighted as 1. In the second, standard CAT training stage, the speaker transforms, the speaker subspace parameters and the canonical mean/variance are iteratively updated, while in the third training stage, 5 sets of parameters: the speaker transforms, speaker subspace parameters, noise transforms, noise subspace parameters and the canonical mean/variances are re-estimated iteratively. The overall fCAT training procedure, starting from a well-trained CAT model, is summarised in the following:

1. Initialise the noise transforms by setting the weight corresponding to the current utterance's noise type as 1, all the other weights as 0; initialise the speaker transforms using the transforms obtained during standard CAT training.

2. Given the current speaker subspace (obtained by the standard CAT), and the speaker/noise transforms the noise subspace is estimated to maximise the log-likelihood function while maintaining the independence constraint in Eq. (6.38).

3. The noise transforms are updated while keeping all the other parameters fixed.

4. The speaker subspace is updated while keeping all the other parameter fixed. Again, the independence constraint in Eq. (6.38) needs to be maintained.

5. Update speaker transforms while keeping all the other parameters fixed.

6. Goto step 2 $N_1(\sim 2)$ times.

7. The canonical mean and variances are updated given the current speaker/environment transforms and clusters.

8. Goto step 2 $N_2(\sim 5)$ times.

6.4 Summary

This chapter considers speaker and noise compensation schemes within the acoustic factorisation framework. Conventional approaches to speaker and noise compensation usually combine an adaptive linear transform with a noise-specific, usually nonlinear, transform to model the combined effect of speaker and noise factors. An example of a conventional approach, the VTS-MLLR scheme, was presented in section 6.1, where the MLLR transform is tied to both speaker and noise factors. This was compared with the Joint scheme in section 6.2, in which the MLLR transform is designed to represent the speaker factor only. A simple example is

also used to illustrate why the factorisation attribute is expected in the Joint scheme. The effectiveness of the Joint scheme and the benefit of acoustic factorisation will be demonstrated by the experiments presented in section 8.1.

The Joint scheme is designed based on the transform constrained approach to achieve factorisation. It leverages knowledge of how noise impacts the speech features, which is summarised in a mismatch function. In practice, the mismatch function is not always known. In that case, the explicit constrained approach proposed in the previous chapter can be used to design the factorised transforms. In this chapter, it is also applied to the speaker and noise factorisation problem. As a proof-of-concept study, a simple linear transform, fCAT, is used to represent both speaker and noise variability. Without the explicit constraint, fCAT for speaker and noise factorisation solely relies on the balanced data to factorise. Applying the proposed independence constraints to fCAT yields orthogonal constraints for the speaker and noise subspaces. The effectiveness of this constraint for fCAT will be demonstrated by the experiments in section 8.2.

CHAPTER 7
Experiments on Reverberant Robustness

In this chapter, the effectiveness of the proposed model-based approaches to robust speech recognition in reverberant environments will be investigated. Evaluations are based on two recognition tasks: a reverberant version of the standard AURORA4 task and a Multi-Channel Wall Street Journal Audio Visual (MC-WSJ-AV) task [161]. In the first task, background noise and the reverberant noise are artificially added to a medium vocabulary Wall Street Journal (WSJ) task to simulate background noise in various reverberant environments. In the second task, single distant microphone data recorded in real reverberant environments are used as the test data. Results and discussions on these two tasks are presented in Section 7.1 and Section 7.2 respectively.

7.1 The Reverberant AURORA4 Task

To evaluate the proposed noise robustness schemes for reverberant environments, the standard AURORA4 task is used in this thesis to create a reverberant and background noise corrupted task, the reverberant AURORA4 task. Training and test data in the AURORA4 task were filtered by the tool in [102] to simulate various reverberant environments.

Microphone Channels	Background Noise Type						
	no	car	babble	restaurant	street	airport	train
Primary	01	02	03	04	05	06	07
Secondary	08	09	10	11	12	13	14

Table 7.1: The definition of test sets in the standard AURORA4 task. Set A (01) is the clean data; Set B (02-08) are distorted by 6 different types of background noises; Set C (08) and Set D (09 - 14) are recorded by secondary microphones, and contains channel mismatch; Background noises are also used to distort the data in Set D.

7.1.1 Datasets

The standard AURORA4 task and the reverberant AURORA4 task are both created by artificially corrupting the clean data in the WSJ0 corpus. The creation of these two tasks are briefly described in the following.

There are two training sets in the standard AURORA4 [188] task: the clean and multi-condition training sets. Both of these two sets comprise 7138 utterances (about 14 hours) from 83 speakers. 16kHz data were used in all the experiments in this thesis. In the clean training dataset, the 7138 training utterances were recorded using a primary microphone, which is a close-talking microphone. For the multi-condition data, half of them came from desk-mounted, secondary microphones. The multi-condition data had 6 different types of noise added, with the SNR ranging from 20dB to 10dB. There are 4 test sets defined in the standard AURORA4 task. 330 utterances from 8 speakers, recorded by the close talking microphone, form task 01 (set A). 6 types of noises, the same as those in multi-condition training data, were added to the clean data, with randomly selected SNRs (from 15dB to 5dB, average 10 dB). These form tasks 02 to 07 (set B). Recordings of these utterances by desk-mounted secondary microphones were also provided in task 08 (set C). Noise were added to set C to form tasks 09 to 14 (set D). The definition of test sets in the standard AURORA4 task is summarised in Table 7.1.

The reverberant AURORA4 corpus was created by filtering the clean data in the standard AURORA4 corpus with a few RIRs and a range of reverberation time T_{60}. Five test sets were created using two RIRs, office1 and office2, which were measured in two different office environments respectively. The five test sets are:

- r01 test set: the clean test data in the standard AURORA4 task were filtered by the RIR office1 to form the test set r01;
- r02 test set: the clean test data in the standard AURORA4 task were filtered by the RIR office1; the restaurant noise signals in task 04 in the standard AURORA4 task

Reverberant Environment	No Background Noise	Restaurant Noise	
		Uncorrelated	Correlated
office1	r01	r02	r03
office2	r04	r05	

Table 7.2: Definition of 5 test sets in the Reverberant AURORA4 task. The reverberation time T_{60} is fixed as 400ms.

were added to the RIR filtered signals to form the test set r02. The background noise in r02 is referred to as *frame-uncorrelated background noise* (see Section 3.3.1.1);

- r03 test set: the 04 data (restaurant background noise) in the standard AURORA4 task were filtered by the RIR office1 to form the test set r03. The background noise in r03 is referred to as *frame-correlated background noise* (see Section 3.3.1.1);
- r04 test set: the clean test data in the standard AURORA4 task were filtered by the RIR office2 to form the test set r04;
- r05 test set: similar as the r02 test set, but the RIR office2 was used.

In creating these 5 test sets, the reverberation time T_{60} was fixed as 400ms. The office1 environment differs from the office2 environment due to the differences in the shape of RIRs office1 and office2 (e.g., some frequency band in the RIR office1 may attenuate quicker in RIR office2). Since it is expected that the shape of RIRs and the frame correlation are two independent factors, a test set r06, which would be frame-correlated restaurant noise added to the office02 reverberant environment, was not investigated in this work. The definition of reverberant test sets is summarised in the Table 7.2.

For the training set definition, the clean training set in the standard AURORA4 task was also used as the clean training set in the reverberant AURORA4 task. To create a multi-condition reverberation training set, the office1 RIR mentioned before and a RIR measured in a living room environment, referred to as liv1, were used to filter the standard AURORA4 clean training set, with the reverberant time T_{60} ranging from 200ms to 600ms. The 6 background noise types in the standard AURORA4 task were also added to RIR-filtered signals, yielding frame-uncorrelated background noise in the multi-condition training set. The ratio of reverberant speech to the background noise, also known as RNR, ranges from 10dB to 20dB. Note the test sets r04 and r05 are unseen environments for this multi-condition training set, as the RIR office2 was not used in creating the multi-condition training data.

Microphone Channels	Background Noise Type						
	no	car	babble	restaurant	street	airport	train
Primary	7.1	38.5	58.5	56.0	64.7	53.5	64.0
Secondary	47.1	62.4	70.4	71.2	79.8	69.7	76.8

Table 7.3: Recognition performance (in WER %) of the clean-trained acoustic model on the standard AURORA4 test data.

Reverberant Environment	No Background Noise	Restaurant Noise	
		Uncorrelated	Correlated
`office1`	70.3	97.3	98.1
`office2`	60.1	97.6	

Table 7.4: Recognition performance (in WER %) of the clean-trained acoustic model on the reverberant AURORA4 test data.

7.1.2 Baseline Experiments

The HTK front-end was used to derive a 39-dimensional feature vector, consisting of 12 MFCCs, extracted from the magnitude spectrum, appended with zeroth cepstrum, delta and delta-delta coefficients. The clean training data was used to build the decision tree. A cross-word tri-phone model set with 3140 distinct states and 16 component per state was ML-trained on the clean data, and this model topology was used throughout all the experiments. The standard 5k-vocabulary bi-gram LM for the AURORA4 task was used in decoding. The language scale factor was fixed as 16 in all the experiments.

An initial decoding was run on the 14 tasks in the standard AURORA4 corpus and the 5 test sets in the reverberant AURORA4 corpus using acoustic models trained on the clean data without any model adaptation. The large mismatch between the clean training condition and the noisy test condition can be seen from Table 7.3 and Table 7.4. On the clean data, the `01` task in the standard AURORA4 corpus, a WER of 7.1% can be achieved, while on all the other 13 tasks in AURORA4, performance is severely degraded and the average WER on 14 tasks is substantially increased to 58.5%. The impact of reverberant environments is even more serious, as demonstrated in Table 7.4.

Baseline systems on both the standard AURORA4 task and the reverberant AURORA4 task were established by using the standard VTS-based model adaptation scheme. Two model sets were used in the baseline systems. The first one is the clean trained acoustic model. The second is a VTS-based adaptively trained acoustic model, denoted as "VAT". Adaptation in all the experiments in the standard AURORA4 and reverberant AURORA4 tasks were performed in an unsupervised mode. All noise parameters were estimated at the utterance level. Initially, the clean trained acoustic model or the VAT acoustic models were

Procedure 7.1: VTS-based unsupervised noise estimation and model compensation.

Given:

a set of test utterances $\{\mathcal{O}^{(r)}|r = 1, \cdots, R\}$ and the acoustic model $\mathcal{M}$

Output:

VTS-based noise model parameters $\{\mathbf{\Phi}_{\text{vts}}^{(r)}|r = 1, \cdots, R\}$ and decoding hypotheses $\{\mathcal{H}^{(r)}|r = 1, \cdots, R\}$

for *r=1,..., R* **do**

- Initialise $\mathbf{\Phi}_{\text{vts}}^{(r)}$ using the first and last 20 frames of $\mathcal{O}^{(r)}$;

for *i=1,..., I (∼ 2)* **do**

- $\mathcal{M}$ is compensated to $\mathcal{M}^{(r)}$ using the standard VTS model compensation based on the current noise model $\mathbf{\Phi}_{\text{vts}}^{(r)}$;
- Hypothesis $\mathcal{H}^{(r)}$ is obtained by using the compensated acoustic model $\mathcal{M}^{(r)}$ and the standard bi-gram LM;
- Noise model parameters $\mathbf{\Phi}_{\text{vts}}^{(r)}$ are re-estimated N_{EM} times.

end

end

compensated using VTS, based on an initial estimation of the additive noise, using the first and last 20 frames of each utterance. These compensated models were used to decode an initial hypothesis. With this initial hypothesis, the noise models were re-estimated N_{EM} (4 in this task) times. New hypotheses were then obtained by another pass of decoding. This process was optionally repeated a few times (2 in this task). Further VTS iterations did not give any performance gains on either the standard AURORA4 data or the reverberant AURORA4 data. This VTS-based noise estimation and model compensation procedure is described in the procedure 7.1.

Table 7.5 shows the performance of unsupervised adaptation of the clean-trained and VAT trained acoustic models on the 4 sets (set A-D) of the standard AURORA4 task. Using the unsupervised VTS adaptation, the system robustness is significantly improved on all sets (A-D). As a result, the average WER was improved to 17.8%. This demonstrates the usefulness of VTS adaptation on background and convolutional noise corrupted data. Using the VAT acoustic model and VTS adaptation can further improve the recognition performance to 15.9%. A detailed breakdown of the performance on each of the 14 tasks can be found in Table 8.1 in the next Chapter.

Unsupervised VTS adaptation of the clean trained acoustic model is also performed on test data in the reverberant AURORA4 task. Table 7.6 compares the performance of this system on the clean data (`01` task), background noise corrupted data (`04` task), data in reverberant environments without background noise (`r01` and `r04` tasks), and data in reverberant environment with frame-uncorrelated (`r02` task) and frame-correlated background noise (`r03`

task). Note in a non-reverberant environment, the background noise is assumed to be frame-uncorrelated. The substantial impact of reverberant noise is evident by comparing WERs in Table 7.5 and Table 7.6. On the task `r01`, which only contains reverberant noise, the VTS-compensated acoustic model gives a WER of 43.8%, while on its non-reverberant counterpart data, the `01` task in the standard AURORA4 corpus, the WER performance is 6.9%. On the task `02`, which contains both reverberant noise and frame un-correlated background noise, the VTS model compensation gives a WER of 51.6%, while on its non-reverberant counterpart, the `04` task in the standard AURORA4 task, the WER is 19.5%. It is also noted that there is a large performance variation among the data with different RIRs. For example, the WER on the `r01` data is 43.8%, while the WER on `r04` data is 30.9%, though both data were corrupted by the same reverberation time $T_{60} = 400$ms. This illustrates that the shape of RIRs can also have an impact on the recognition performance. Given the recognition performance in Table 7.5, it is clear that the VTS-based model adaptation is not able to achieve high performance in reverberant environments.

7.1.3 Model Compensation and Noise estimation

In this section, a set of experiments were run to investigate various aspects of the proposed RVTS and RVTSJ model compensation schemes. These include: static and dynamic parameter compensation; ML noise parameter estimation; sensitivity to the reverberant time T_{60}; power-domain mismatch function and combination with linear transforms. This section will focus on the effectiveness of model compensation and only the clean-trained acoustic model will be used in the experiments. The effectiveness of reverberant adaptive training will be investigated in Section 7.1.4.

7.1.3.1 Static and Dynamic Parameter Compensation

In this experiment, the clean-trained acoustic model was compensated using an initial noise model. The noise parameters were obtained using an estimated T_{60} value as described in Section 4.2.3. There are a few methods can be used to estimate the T_{60} value from reverberant

Systems	Adaptation	A	B	C	D	Avg.
clean-trained	—	7.1	55.9	47.1	71.7	58.5
	VTS	6.9	15.1	11.8	23.3	17.8
VAT	VTS	8.5	13.7	11.8	20.1	15.9

Table 7.5: Recognition performance (in WER%) of unsupervised VTS adaptation of the clean-trained and VAT trained acoustic models on the 4 sets (set A-D) of standard AURORA4 task. The standard bi-gram LM was used in decoding.

Reverberant Environment	Background Noise		
	No	Frame-Uncorrelated	Frame-correlated
No	6.9 (01)	19.5 (04)	—
office1	43.8 (r01)	51.6 (r02)	48.9 (r03)
office2	30.9 (r04)	48.8 (r05)	—

Table 7.6: Recognition performance (in WER %) of unsupervised VTS adaptation of the clean-trained acoustic model on the clean data (01 task), background noise corrupted data (04 task), data in reverberant environments without background noise (r01 and r04 tasks), and data in reverberant environment with frame-uncorrelated (r02 task) and frame-correlated background noise (r03 task). The standard AURORA4 bi-gram LM is used in decoding.

signals, e.g., [103]. In this experiment, it is assumed that a correct T_{60} estimate is available: T_{60} is fixed as 400ms, the same value as was used in artificially corrupting the test data. The impact of using an incorrect T_{60} value on the recognition performance will be investigated later.

Table 7.7 compares the WERs (in %) of VTS, RVTS and RVTSJ compensated acoustic models. Note that the proposed RVTS and RVTSJ schemes only compensate the mean parameters while the variance parameters were compensated by the standard VTS scheme. It can be seen from this table that when only static mean parameters are compensated, both RVTS and RVTSJ schemes gain over the standard VTS scheme on the background noise free task r01. However, on the tasks r02 and r03, which have background noise distortion, RVTS and RVTSJ performance is degraded, especially for RVTSJ. Extending the model compensation to the dynamic mean parameters substantially improves the performance on all tasks. This demonstrates that dynamic parameters compensation is essential in reverberant environments. When both static and dynamic mean parameters are compensated, RVTSJ and RVTS performed similarly on task r01, while RVTSJ is slightly better than RVTS on tasks r02 and r03. Note the task r03 was created using frame correlated background noise, which matches the mismatch function used for RVTS model compensation. Since compensating all the mean parameters always gives the best performance, this configuration is used in the all the following experiments.

It is also observed in this table that RVTS and RVTSJ using this simple noise model initialisation have already significantly [1] improved the performance over VTS model compensation (average WER 42.7% of RVTS and 41.2% of RVTSJ vs 48.1% of VTS). This is expected as the RVTS and RVTSJ model compensation is designed for reverberant environments, while standard VTS only handles the background noise and convolutional noise. It is

[1] All statistical significance tests in this thesis were based on a matched pair-wise significance test at a 99.5% confidence level.

Schemes	Compensation		test			Avg.
	mean	var.	r01	r02	r03	
VTS	$\boldsymbol{\mu}_{\text{VTS}}$	$\boldsymbol{\Sigma}_{\text{VTS}}$	43.8	51.6	48.9	48.1
RVTS	$\boldsymbol{\mu}_{\mathtt{sz}}$		39.5	52.3	50.8	47.5
	$\boldsymbol{\mu}_{\mathtt{sz}}$, $\boldsymbol{\mu}_{\Delta\mathtt{z}}$	$\boldsymbol{\Sigma}_{\text{VTS}}$	33.6	52.6	52.1	46.1
	$\boldsymbol{\mu}_{\mathtt{sz}}$, $\boldsymbol{\mu}_{\Delta\mathtt{z}}$, $\boldsymbol{\mu}_{\Delta^2\mathtt{z}}$		29.7	49.7	48.7	42.7
RVTSJ	$\boldsymbol{\mu}_{\mathtt{sz}}$		41.2	86.5	86.0	71.2
	$\boldsymbol{\mu}_{\mathtt{sz}}$, $\boldsymbol{\mu}_{\Delta\mathtt{z}}$	$\boldsymbol{\Sigma}_{\text{VTS}}$	33.2	69.5	67.0	56.6
	$\boldsymbol{\mu}_{\mathtt{sz}}$, $\boldsymbol{\mu}_{\Delta\mathtt{z}}$, $\boldsymbol{\mu}_{\Delta^2\mathtt{z}}$		29.4	47.3	46.9	41.2

Table 7.7: Recognition performance (in WER %) of static, delta and delta-delta mean parameter compensation using RVTS and RVTSJ. The clean-trained acoustic model was used. The noise model parameters is initialised by the known reverberation time $T_{60} = 400$ms.

also noted that reference[103] reports a WER of 39.8% on a test set similar to the `r01` task, in which the noise model parameters are also initialised from the estimated T_{60} value. As a contrast, the RVTS and RVTSJ schemes achieve WERs of 29.7% and 29.4% respectively on the `r01` task.

7.1.3.2 ML Noise Estimation

Experiments in Table 7.7 used a simple reverberant noise estimation, which only has one free parameter, the T_{60} value. As demonstrated in the development of VTS model compensation schemes (e.g., [153, 159]), using ML estimated noise model parameters always yields large gains over using simple initial estimates of noise. Hence, it is also preferable to use ML estimated noise for RVTS and RVTSJ compensation. The VTS hypothesis (the first row in Table 7.7) was taken as the initial supervision; noise parameters were re-estimated while the model variance was locked at the VTS compensated variance. A few EM iterations were used to re-estimate the reverberant noise model parameters. The supervision hypothesis was also updated to yield better noise estimation before the final decoding. The detailed RVTS/RVTSJ noise estimation and model compensation procedure is illustrated in procedure 7.2, with the number of hypothesis update iterations $I = 1$ and $N_{\text{EM}} = 4$.

The recognition performance of RVTS and RVTSJ model compensation with unsupervised noise estimation are presented in Table 7.8. In the first rows of block 2 and 3 in Table 7.8 ("Init"), noise model parameters were initialised based on an estimated T_{60} value, while in the second rows of block 2 and 3 ("ML"), the noise model parameters were ML estimated given the VTS hypothesis. Additionally, the hypothesis was also updated to re-estimate the noise parameters (the third rows of block 2 and 3, "upd. hyp."). As expected, ML estimation of noise yields consistent gains over initial noise estimation. RVTSJ outperforms RVTS in

Procedure 7.2: RVTS/RVTSJ model compensation and unsupervised noise estimation procedure.

Given:

a set of test utterances $\{\mathcal{O}^{(r)}|r = 1, \cdots, R\}$;
VTS noise model parameters $\{\mathbf{\Phi}_{\text{vts}}^{(r)}|r = 1, \cdots, R\}$;
VTS supervision hypotheses $\{\mathcal{H}^{(r)}|r = 1, \cdots, R\}$;
and an estimate of T_{60} value;

Output:

RVTS and RVTSJ noise model parameters $\{\mathbf{\Phi}^{(r)}|r = 1, \cdots, R\}$ and final decoding hypotheses $\{\mathcal{H}^{(r)}|r = 1, \cdots, R\}$

for *r=1,..., R* **do**

- Initialise $\mathbf{\Phi}^{(r)}$ using the known T_{60} value;

for *i=1,..., I* **do**

- Noise model parameters $\mathbf{\Phi}^{(r)}$ are re-estimated N_{EM} times;
- $\mathcal{M}$ is compensated to $\mathcal{M}^{(r)}$ using RVTS or RVTSJ based on the current noise model $\mathbf{\Phi}^{(r)}$;
- Decoding hypothesis $\mathcal{H}^{(r)}$ is obtained by using acoustic model $\mathcal{M}^{(r)}$ and the standard LM;

end

end

all three task. This is due to the sequential approach to noise estimation in RVTS, where the additive noise was estimated using the VTS-style mismatch function, then the frame-level distortion terms were estimated given the additive noise. Because of this sequential approach, the additive noise was used to model some attributes of reverberation, yielding inaccurate noise estimates. Joint estimation of both additive and reverberant noise alleviates this issue by taking the effect of both reverberant and additive noise into account. It is also observed from Table 7.8, though the supervision WERs have been significantly reduced by RVTS/RVTSJ compensation (from an average 48.1% to 39.1% for RVTS and 36.3% and RVTSJ), updating the supervision hypotheses does not yield significant gains. This may suggest that RVTS and RVTSJ schemes are relatively insensitive to the errors in the supervision hypothesis. It is also noted that RVTS and RVTSJ yield large gains on reverberant noise only set `r01`, about 15-20% absolute gains, while the gains on sets `r02-r03` are reduced to about 10% absolute. This may suggest that the mismatch function for background noise in reverberant environments is not as accurate as the one for reverberant noise only, and the temporal correlation of background noise is either discarded in RVTSJ or not well estimated in RVTS due to the employed sequential noise estimation procedure.

Schemes	test			Avg.
	r01	r02	r03	
VTS	43.8	51.6	48.9	48.1
RVTS (Init.)	29.7	49.7	48.7	42.7
+ML	27.3	46.0	44.0	39.1
+upd. hyp.	26.7	45.8	43.9	38.9
RVTSJ (Init.)	29.4	47.3	46.9	41.2
+ML	24.8	43.6	40.8	36.3
+upd. hyp.	24.1	43.1	40.4	35.8

Table 7.8: Recognition performance (in WER%) of RVTS and RVTSJ compensation of the clean-trained acoustic model with unsupervised noise model estimation. "Init." means noise model parameters are initialised based on the correct T_{60} value; "ML" stands for the noise model parameters are ML estimated using the VTS hypothesis; "upd. hyp." denotes that the hypothesis was updated to re-estimate the noise parameters.

7.1.3.3 Sensitivity to Reverberant Time Estimation

The above experiments assume that the correct reverberation time T_{60} is known. In practice, this information is often not available, and an estimate of T_{60} is used. It is therefore interesting to investigate the sensitivity of the proposed model compensation schemes with respect to the variations of T_{60} values. Another set of experiments were run on task r01 and the initial T_{60} ranges from 100ms to 800ms, with a step size of 100ms. Results are presented in Table 7.9. Note the supervision hypothesis was fixed as the VTS hypothesis. The results demonstrate that when noise models are simply initialised from the T_{60} value, both RVTS and RVTSJ yield the best performance (after ML estimation of noise parameters) when the estimate of T_{60} is around its correct value. There is a small performance variation when the estimate of T_{60} deviates from its correct value. After ML estimation, the variations are reduced. This indicates that the proposed RVTS and RVTSJ schemes with ML noise estimation are insensitive to the error in T_{60} estimation.

T_{60}	RVTS		RVTSJ	
	Init.	ML	Init.	ML
100	31.0	28.2	30.2	25.9
200	28.4	28.0	27.5	24.9
300	29.1	27.6	28.4	24.9
400	29.7	27.3	29.4	24.8
600	31.7	27.6	30.8	24.9
800	32.4	27.4	31.7	25.3

Table 7.9: Recognition performance (in WER %) of the RVTS and RVTSJ model compensation using a range of initial T_{60} values on the r01 task. Noise model parameters are either initialised using the given T_{60} value ("Init.") or ML estimated ("ML").

7.1.3.4 Power-Domain Mismatch Function

The above experiments assume the noise and speech are additive in the magnitude domain, as it was empirically found magnitude domain combination yielded better results [72] for additive noise corrupted data. It is also interesting to examine this conclusion for reverberant noise corrupted data. VTS and RVTSJ adaptation experiments were re-run on tasks `r01` and `r03` using the power domain mismatch functions ($\gamma = 2$) with the same setup. Results are shown in Table 7.10. Consistent with the findings in [72], magnitude domain combination performs better.

Schemes	domain	test		Avg.
		r01	r03	
VTS	power	46.7	61.1	53.9
	magnitude	43.8	48.9	46.4
RVTSJ	power	30.5	53.1	41.8
	magnitude	24.8	40.8	32.8

Table 7.10: VTS and RVTSJ model compensation recognition performance on tasks `r01` and `r03` using the power-domain and magnitude-domain mismatch functions. Noise parameters are ML estimated. Note the average WERs are calculated on `r01` and `r03`; thus they are not comparable with those in the previous tables.

7.1.3.5 Combination with Linear Transforms

To further improve the performance, a linear transform, the CMLLR transform [67], was combined with previous model compensation schemes. A global CMLLR transform was estimated for each speaker using the VTS hypothesis. Results are shown in Table 7.11. As expected, adding linear transforms to further reduce the mismatch yielded large gains. The best performance was achieved by RVTSJ combined with CMLLR adaptation, the average WER 28.4% on tasks `r01` and `r03`. This was a 38.8% relative error reduction, compared with the VTS model compensation scheme. The combination of RVTSJ and CMLLR transform also outperforms the Direct CMLLR scheme proposed in [73], in which a linear transform per speaker was employed to project several neighbouring frames. This demonstrates that the use of nonlinear mismatch functions is helpful for the reverberant noise distortion.

7.1.4 Reverberant Adaptive Training

In the previous section, the clean-trained acoustic model was used in the experiment. It is demonstrated in [214] that when multi-condition data is available, multi-style trained acoustic models can yield significant gains over adapting clean-trained acoustic models. However, a

Schemes		test		Avg.
		r01	r03	
VTS	ML	43.8	48.9	46.4
	+CMLLR	32.0	45.2	38.6
RVTS	ML	27.3	44.0	35.7
	+CMLLR	22.8	41.0	31.9
RVTSJ	ML	24.8	40.8	32.8
	+CMLLR	20.2	36.5	28.4

Table 7.11: Recognition performance (in WER%) of combing VTS, RVTS and RVTSJ model compensation schemes with CMLLR adaptation. For RVTS and RVTSJ model compensation, noise parameters and the CMLLR transform were ML estimated based on the VTS hypothesis. Average WERs are calculated on r01 and r03.

System	Adaptation	office1		office2		Avg.
		r01	r02	r04	r05	
Clean-trained	VTS	43.8	51.6	30.9	48.8	43.8
	RVTSJ	24.8	43.6	25.4	43.2	34.3
MST	–	19.9	38.3	37.4	63.6	39.8
	CMLLR	14.1	30.1	14.7	48.4	26.8
VAT	VTS	15.1	29.0	18.7	44.5	26.8
	RVTSJ	14.9	29.6	18.3	43.6	26.6
RAT	RVTSJ	13.7	28.5	15.0	42.2	24.9

Table 7.12: Performance (in WER%) of clean-trained, multi-style trained and adaptively trained acoustic models in reverberant environments. Reverberant environment "office1" was seen in the multi-condition training data while "office2" was not seen; r01 and r04 tasks do not contain background noise, while r02 and r04 tasks contain frame-uncorrelated background noise. Average WERs are calculated on r01, r02, r04 and r05.

limitation of multi-style trained acoustic models is that it may not generalise well to reverberant conditions unseen in the training data. As an alternative to this multi-style training (MST), reverberant adaptive training (RAT) is proposed in section 4.3.

The MST acoustic model was built first. Starting from this MST model, the VAT system was built, which in turn serves as an initialisation for the RAT acoustic model. Following the procedure in Algorithm 4.2, a RAT acoustic model was built, where the number of iterations were set as $N_{\text{model}} = 3$, $N_{\text{trans}} = 2$ and $N_{\text{EM}} = 4$. Within each N_{EM} iterations of canonical model update, β is reduced from 8 to 1, halving at every iteration. As a comparison, the extended mean parameters $\overline{\boldsymbol{\mu}}_{\mathbf{x}}^{(m)}$ which were obtained by solving the optimisation problem in Eq. (4.45) were appended with the VAT acoustic model. In this way, the RVTSJ adaptation can be also be performed using the VAT acoustic model.

An initial decoding using the MST acoustic model without adaptation was run. Results

are shown in the first row of the second block in Table 7.12. Compared with adapting the clean-trained acoustic models (details were described in the previous section and the WERs are duplicated in the first block of Table 7.12), the MST model considerably improves the WERs in the "`office1`" environment. However, when operated in an environment not observed during training, "`office2`", WER increased. This demonstrates multi-style training introduces a bias toward the training environments and may not generalise well to unseen conditions. The MST system was also adapted by a CMLLR transform for each speaker, using the initial MST decoding hypothesis as the supervision hypothesis. As shown in the fourth row in Table 7.12, CMLLR adapted MST system yields large error reductions in all conditions. This is consistent with the findings reported in [169, 214]. Note that, though powerful, the CMLLR transform needs to be estimated on multiple utterances in the same condition. As a result, the CMLLR scheme is not as flexible as VTS and RVTSJ schemes which can be estimated on a single utterance.

RVTSJ adaptation experiments using the VAT and RAT models were also performed in an unsupervised mode. The adaptation procedure is the same as the one used in the previous section, except that the MST+CMLLR hypothesis was used as supervision, as the initial investigation shows that this is helpful for the recognition performance. Results are displayed in the third and fourth blocks of Table 7.12. Compared with the CMLLR adapted MST acoustic model, the VAT acoustic model with VTS adaptation performed worse when there is only reverberation distortion, but gave gains when the background noise is also present. This is due to the fact that VTS is designed to compensate the impact of background noise, while the CMLLR can be used for general adaptation. RVTSJ adaptation of the VAT acoustic model was also performed, which gave small gains over VTS adapted VAT system in average. RVTSJ adaptation of the RAT acoustic model further improves the performance. Compared with the VAT acoustic model, RAT yields 0.5% to 1.3% absolute gains for the "`office1`" environment, and 2.3% to 3.7% for the "`office2`" environment. This demonstrates that reverberant adaptive training produces a canonical model neutral to the reverberant and background noise distortions to some extent. On average, the RAT system gave the best performance, yielding 1.9% absolute gains over the CMLLR adapted MST system.

7.2 MC-WSJ-AV Task

In the previous section, the proposed reverberant robustness schemes were evaluated on the reverberant AURORA4 task where the reverberant and background noises are artificially added. In this section, these schemes will be evaluated on the Multi-Channel Wall Street Journal Audio Visual (MC-WSJ-AV) task [161], where data was recorded in "real" reverberant

environments. The MC-WSJ-AV task consists of a collection of read Wall Street Journal (WSJ) sentences taken from the development and evaluation sets of the WSJCAM0 database [203]. The data were recorded in a number of instrumented meeting rooms constructed within the framework of the European Augmented Multi-party Interaction (AMI) project. Three different scenarios were considered during the recording of the database, where only the "single stationary speaker" scenario is considered in this thesis. Furthermore only the test speakers recorded at University of Edinburgh (UEDIN) were used. Data were recorded simultaneously by a headset microphone, a lapel microphone and two 8-element circular microphone arrays placed on the table. The configuration examined here is primarily the single distant microphone (`sdm`), where element 1 of microphone array 1 was used. One development (`dev1`) and two evaluation sets (`ev1` and `ev2`) were used in the experiments. Each set consists of 5 test speakers, with around 17 adaptation sentences per speaker for speaker/environment adaptation. The number of sentences associated with each of the test sets was `dev1` 178, `evl1` 188 [1], and `evl2` 183. It was measured in [151] that the ratio of reverberant speech to the background noise (RNR) of test data is 15-20dB in average. Estimates of the room reverberation time range from 380ms [140] to 700ms [161].

The training data were taken from the WSJCAM0 database, which include 7861 utterances from 92 speakers, all with British English accents. The WSJCAM0 data were recorded by close talking microphones in quiet environments. As such, it was used as clean training data in the experiments. Multi-condition training data was also created using a script provided by the 2013 REVERB (REverberant Voice Enhancement and Recognition Benchmark) challenge [1]. Clean training data in WSJCAM0 were convolved with various RIRs and background noise signals were subsequently added. The RIRs were measured in 6 different reverberant environments: 3 rooms with different volumes (small, medium and large size), 2 types of distances between a speaker and a microphone array (near=50cm and far=200cm). Only the element 1 in microphone arrays was used in the experiment, as the single distant microphone is the main focus of this work. Stationary background noise, which was mainly caused by air conditioning systems in rooms, was measured under the same conditions with the same arrays as used for RIR measurement. These RIRs and background noise signals were randomly selected to corrupt each utterance in the WSJCAM0 training set, forming the multi-condition reverberant WSJCAM0 training set. The background noise was added at a RNR of 20dB. Reverberation time of small, medium, and large-size rooms for the far field microphone array are about 250ms, 500ms, 700ms respectively.

[1] The data for one sentence was missing for this set. Therefore results presented in this thesis are slightly mismatched with the numbers quoted in [161] and [151].

There are several published works reporting their results on the MC-WSJ-AV tasks,e.g., [140, 151, 161] and [73]. References [140, 161] mainly focus on microphone array techniques to reduce the impact of reverberation while[151] and [73] focus on the single distant microphone scenario. All the previous works used clean training sets to build acoustic models.

7.2.1 Experimental Setup

Initial experiments were performed using the acoustic model built from the clean training data. The same feature extraction procedure as the one used for the reverberant AURORA4 task was used. Cross-word triphone models with 3066 distinct states were constructed with 12 Gaussian components per state. The extended model statistics were estimated from the cleaning training data in the same way as the reverberant AURORA4 task. The parameters $n = 10$ and $w = 4$ were used. These model statistics were also adaptively trained from the multi-condition reverberant WSJCAM0 set using RAT in the second set of experiments. A trigram language model was used for decoding.

Adaptation in this task was performed in a supervised mode. VTS, RVTS/RVTSJ, CMLLR transforms were estimated from the adaptation read utterances at the speaker level. Multiple EM iterations were performed to reduce the performance differences due to the number of iterations. A global class was used for CMLLR transform. The T_{60} value was initialised as 400ms in all the experiments.

7.2.2 Results

Table 7.13 compares recognition performance of VTS, RVTS and RVTSJ adaptation of clean trained acoustic models on this task. Similar to the reverberant AURORA4 task, using RVTS/RVTSJ compensation based on the initial reverberant noise model parameters already yields gains over the baseline VTS scheme. Using ML-estimated noise model parameters yields additional gains, giving about 20% relative reduction over the baseline VTS scheme. RVTS and RVTSJ yield similar WERs in this task, with RVTSJ slightly better. This is expected, as the background noise level is relatively low in this task. Further gains were obtained when RVTS/RVTSJ was combined with CMLLR adaptation. However, the differences between VTS and RVTS/RVTSJ are reduced. The best WER performance is obtained by combining RVTSJ with CMLLR adaptation, which yields 5.5% absolutely gains over VTS+CMLLR. It is noted reference [151] reports their best WERs on `evl1` set are around 39-40%, using a feature enhancement technique. This represents the state-of-the-art performance of feature enhancement-based approach on this task. Also note [151] uses a bigram language model and

Scheme		Test Set			Avg
		dev1	evl1	evl2	
VTS	ML	58.7	53.7	69.5	60.6
	+CMLLR	45.4	38.9	49.1	44.4
RVTS	Init	52.7	46.5	56.0	51.7
	ML	49.8	42.9	52.1	48.2
	+CMLLR	43.8	35.5	43.6	40.9
RVTSJ	Init	50.3	44.5	55.1	49.6
	ML	48.2	40.9	53.8	47.5
	+CMLLR	40.2	32.9	43.6	38.9

Table 7.13: VTS, RVTS and RVTSJ MC-WSJ-AV dev and eval results (in WER %). Clean trained acoustic models are used in the experiments.

Adaptation	Test Set			Avg
	headset	lapel	sdm	
—	46.6	77.4	97.4	73.8
CMLLR	9.8	16.1	90.0	38.6
VTS	14.8	22.8	58.7	32.2
RVTSJ	15.4	23.2	48.2	28.9

Table 7.14: Performance (in WER%) of CMLLR, VTS, and RVTSJ on MC-WSJ-AV dev1 set. Three types of data are used: headset, lapel, and single distant microphone (sdm) .

does not use any adaptation data, thus the comparison is not completely fair. A comprehensive comparison of state-of-the-art model-based and feature-based techniques on this task will be left to future work.

The proposed RVTS and RVTSJ schemes are specifically designed to handle reverberation. It is thus interesting to examine their performance on data recorded by less reverberant channels, such as the lapel and headset. Headset, lapel and single distant microphone (sdm) data in set dev1 were used in the second set of experiments. Results are presented in Table 7.14. Due to differences in accent, speaking rate and styles, acoustic models trained on WSJCAM0 do not perform well even on the headset data. However, a CMLLR adaptation can largely compensate for these differences, and reduce the WER to 9.8%. As headset data is relatively clean, there is no advantage to use VTS and/or RVTSJ adaptation, as indicated by the corresponding WERs. A similar trend is also observed for lapel data. However, for the sdm data, due to reverberation, CMLLR adaptation alone is not able to recover the WER performance. In contrast, VTS reduced the WER to 58.7% and RVTSJ further reduced the WER to 48.2%. This results demonstrated that RVTSJ is specifically designed to handle the reverberation effect. In contrast, the CMLLR adaptation, which is a general purposed linear transform scheme, is not able to handle reverberation effectively.

In the third set of experiments, acoustic models were built on the multi-condition reverberant WSJCAM0 set. The MST model was firstly built. Following the same procedure for the reverberant AURORA4 task, the VAT and RAT systems were subsequently built. Results are presented in Table 7.15. Note, although adaptation for test set was performed on the speaker level, utterance level adaptation was performed during the adaptive training. As expected, CMLLR adaptation of MST acoustic model achieves good performance on all three sets. On average, it reduces WER more than 50% relatively (from 78.6% to 35.5%). Compared with the numbers in the first block of Table 7.13, the VAT system yields consistent and large gains over clean-trained acoustic models,reducing average WER from 60.6% to 40.3%. RVTSJ adaptation of the RAT system yields further gains, though much smaller. Both adaptively trained systems give large gains over clean-trained acoustic models. Again, the best performance was achieved by combing RVTSJ adaptation with CMLLR transform.

Systems	Adaptation	Test Set			Avg .
		dev1	evl1	evl2	
MST	—	84.6	70.8	80.5	78.6
	CMLLR	36.5	29.3	40.7	35.5
VAT	VTS	38.0	34.9	49.6	40.3
RAT	RVTSJ	37.4	30.3	46.0	37.9
	+CMLLR	33.6	27.5	40.3	33.8

Table 7.15: Performance (in WER%) of Multi-style trained (MST) and adaptively trained (VAT and RAT) acoustic models in reverberant environments on sdm test sets in the MC-WSJ-AV task.

7.3 Summary

This chapter has examined the proposed model-based techniques to handle reverberant environments. Two model compensation techniques, RVTS and RVTSJ adaptation, were investigated on two tasks, the reverberant AURORA4 task and the MC-WSJ-AV task. By controlling how the background noise and reverberation are added to the clean signals, a series of test sets were created for the reverberant AURORA4 task. This task was used to examine the RVTS and RVTSJ schemes thoroughly. Results demonstrated that RVTS and RVTSJ performed better than VTS even when the reverberant noise model parameters are initialised using a simple method. ML estimation of noise model parameters gave further gains. It is also showed that RVTSJ performed consistently better than RVTS on three test sets. This is due to the joint estimation of reverberant and background noise in RVTSJ. Results also demonstrated that RVTS and RVTSJ adaptation are insensitive to the supervision error

and reverberation time estimation. It is also shown that combining RVTS or RVTSJ model compensation, which are nonlinear adaptation schemes, with CMLLR adaptation, which is a linear transform based scheme, is very helpful. Adaptive training schemes were also examined in this task. CMLLR adaptation of MST systems gave very large gains, while both the VAT and RAT models gave better performance than CMLLR adapted MST systems, with RAT slightly better. Compared with the MST system, the RAT system generalised better to unseen conditions during training.

The proposed model-based schemes were also evaluated on the MC-WSJ-AV task, where the test data was recorded in the "real" reverberant environment. A similar trend to the reverberant AURORA4 task was observed. Combing RVTSJ and CMLLR adaptation schemes using the RAT system yields the best recognition accuracies on all three sets.

Chapter 8

Experiments on Acoustic Factorisation

This chapter will present experimental results on acoustic factorisation. Three approaches to acoustic factorisation are discussed in Chapter 5, namely the data constrained, transform constrained and explicitly constrained approaches. These approaches are applied to yield speaker and noise factorised adaptation for robust speech recognition in Chapter 6. In this chapter, the "Joint" adaptation scheme, proposed in section 6.2, which is a transform constrained approach to acoustic factorisation, is compared with a simple speaker and noise compensation scheme "VTS-MLLR" to demonstrate the flexibility of the acoustic factorisation framework and the factorisation property can be achieved by a carefully designed transform constrained approach. "Joint" and "VTS-MLLR" schemes are compared in section 8.1 on a standard noise robustness benchmark task, the AURORA4 task. Different from the first two approaches, the explicitly constrained approach does not make assumptions about how acoustic factors impact speech signals and aim to keep transform independence even when data is highly unbalanced. To evaluate the effectiveness of the explicit independence constraint presented in section 5.2.3, the fCAT adaptation scheme discussed in section 6.3 is examined on a noise corrupted WSJ task, with the focus on test data which is unbalanced distributed over the speaker and noise factors.

8.1 The AURORA4 Task

The AURORA4 [188] corpus has been used to test the baseline systems in the previous Chapter. Here, it is used to evaluate the effectiveness of speaker and noise factorisation using the transform constrained approach. The detailed description of the AURORA4 corpus was presented in Section 7.1.1 and the test set definition is summarised in Table 7.1.

All the acoustic models used in experiments were cross-word triphone models with 3140 distinct tied-states and 16 components per state. The standard bi-gram language model provided for the AURORA4 experimental framework was used in decoding and a language scale factor 16 was used. For all the experiments unsupervised adaptation was performed. Where MLLR adaptation was performed, block-diagonal transforms with two regression classes (one speech, one silence) were used. It is assumed that each utterance is in a unique noise condition (noise types plus SNR) in the AURORA4 task, therefore the VTS-based noise estimation was performed on a per-utterance basis. The speaker adaptation was performed on the speaker level. To minimise differences due to the different forms of adaptation, multiple EM iterations were performed when estimating transforms.

8.1.1 Baseline Systems

In order to evaluate the effectiveness of the proposed speaker and noise adaptation scheme, a series of baseline systems were built. The first two systems are a clean-trained acoustic model without adaptation ("clean") and a VAT-based adaptively trained acoustic model ("VAT"). Details of building these baseline systems can be found in Section 7.1.2. As a comparison of model compensation versus feature compensation approaches, the ETSI advanced feature (AFE) was used to build the third system. This system is referred to as "AFE" [230].

Results of these baselines are presented in Table 8.1. Using the VTS-based noise adaptation, the clean-trained model achieved a WER of 17.8%. Compared with other feature-based or model-based noise robustness schemes on AURORA4 (e.g., [43]), it is clear this provides a reasonable baseline on this task. As expected, the use of the adaptively trained acoustic model (the VAT system) gave gains over the clean system on noisy data: the average WER was further reduced from 17.8% to 15.9%. However, a small degradation on the clean set (8.5% of VAT vs. 6.9% of clean) can be seen. This may be explained as VTS is not able to completely remove the effects of noise. Thus the "pseudo" clean speech parameters estimated by adaptive training will have some residual noise effects and so will be slightly inconsistent with the clean speech observation in set A. It is also interesting to look at the performance of the AFE system. With AFE, multi-style training achieved a WER of 21.4%. Note that, the

Systems	Adaptation	A	B							C	D							Avg.
		01	02	03	04	05	06	07	Avg.	08	09	10	11	12	13	14	Avg.	
clean-trained	–	7.1	38.5	58.5	56.0	64.7	53.5	64.0	55.9	47.1	62.4	70.4	71.2	79.8	69.7	76.8	71.7	58.5
	VTS	6.9	9.0	14.2	19.5	16.3	14.0	17.7	15.1	11.8	13.5	22.8	26.8	25.4	23.5	27.6	23.3	17.8
VAT	VTS	8.5	9.8	13.2	16.0	14.6	12.0	16.4	13.7	11.8	12.4	19.6	23.1	23.2	18.8	23.8	20.1	15.9
AFE	—	8.8	13.1	15.9	20.0	18.4	15.1	17.4	16.7	19.1	24.1	27.8	31.1	30.9	28.6	29.4	28.6	21.4

Table 8.1: Recognition performance (WER, in %) of three baseline systems: the clean-trained system, the adaptively trained VAT system and multiple-style trained AFE system.

multi-style model using MFCC features achieved a WER of 27.1%. However, the large performance gap (21.4% vs. 15.9%) between AFE and its counterpart of model-based schemes, the VAT system, demonstrates the usefulness of model-based schemes for this task.

8.1.2 Batch-mode Speaker and Noise Adaptation

The above experiments built a series of baseline systems, where only noise adaptation was performed. In the following experiments, acoustic models were adapted to both the target speaker and environment. In the first set of experiments, speaker and noise adaptation was performed in a batch mode (referred to as "`bat`"), i.e. the adaptation experiments were run where speaker and noise (utterance-level) transforms were estimated for each speaker for each task[1]. The "Joint" and "VTS-MLLR" schemes were first examined using the clean-trained acoustic model. Following the same procedure used in the baseline systems, one VTS iteration was run for each utterance to generate the supervision hypothesis. The generated noise models were also taken to initialise the noise parameters $\boldsymbol{\Phi}$. The speaker level transform, $\mathbf{K}$, was initialised as the identity transform. Then, as discussed in Section 6.2.2, the block coordinate descent optimisation strategy is applied for "Joint" and "VTS-MLLR". Multiple iterations, $N_{\text{EM}} = 4$, were used to update the speaker transform and noise models. This batch-mode speaker and noise adaptation procedure is illustrated in Figure 8.1. As an additional contrast, an MLLR transform was applied on top of the "Joint", again estimated at the speaker level, yielding another scheme "Joint-MLLR". The results of these batch-mode speaker and noise adaptation experiments are presented in Table 8.2. Significant performance gains[2] were obtained using both "Joint" (14.6%) and "VTS-MLLR" (14.7%), compared to the baseline VTS performance (17.8%). The best performance was obtained using the "Joint-MLLR" scheme (14.1%), which indicates that there is still some residual mismatch after "Joint" adaptation and a general linear transform can be used to reduce this mismatch. These experiments serve as a contrast to the factorisation experiments in the next section.

8.1.3 Speaker and Noise Factorisation

To investigate the factorisation of speaker and noise transforms, a second set of experiments were conducted. Again, the noise transforms were estimated for each utterance. However in contrast to the batch-mode adaptation, the speaker transforms were estimated from either

[1]The speaker transforms were estimated for each speaker on each noise condition (01-14), and were used only in the noise condition where speaker transforms were estimated from. The noise transform was always estimated for every utterance.

[2]All statistical significance tests are based on a matched pair-wise significance test at a 95% confidence level.

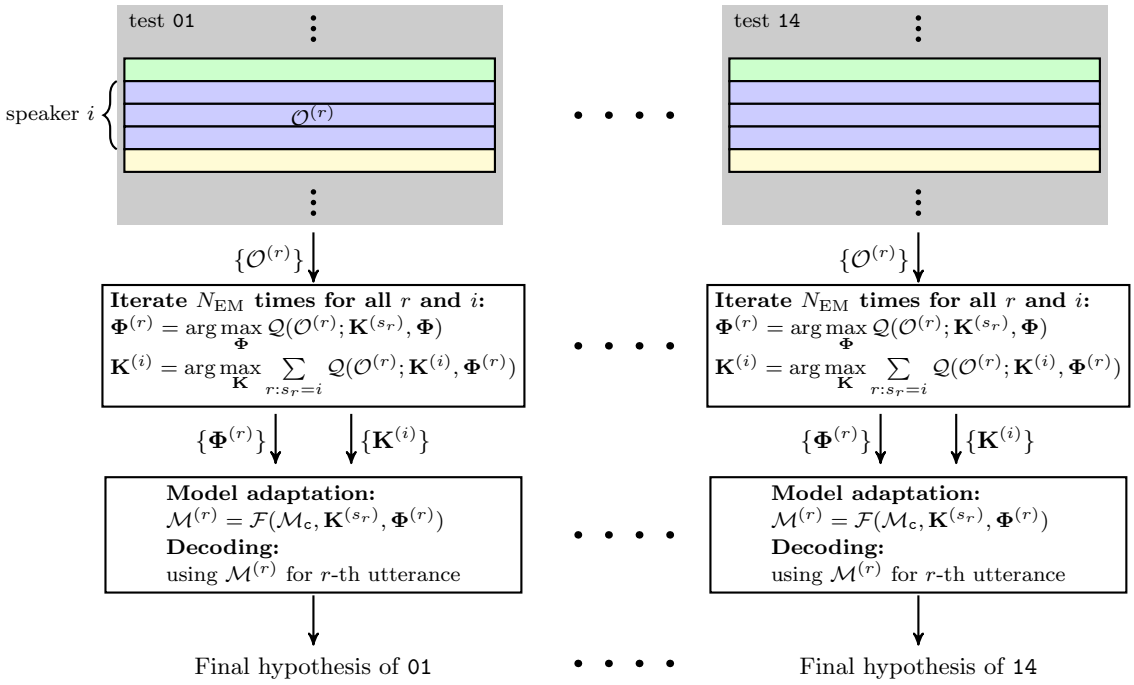


Figure 8.1: Illustration of batch-mode speaker and noise adaptation and decoding procedure.

Adaptation	A	B	C	D	Avg.
VTS	6.9	15.1	11.8	23.3	17.8
VTS-MLLR	5.0	12.1	9.0	19.8	14.7
Joint	5.0	12.1	8.6	19.7	14.6
Joint-MLLR	5.0	11.5	8.1	19.1	14.1

Table 8.2: Performance (in WER%) of speaker and noise adaptation in a batch mode. The clean-trained acoustic model was used. The speaker level linear transform was estimated on each of the 14 tasks and applied to the same task.

01 or 04[1]. These speaker transforms were then fixed and used for all the test sets, just the utterance-level noise transforms were re-estimated. The same setup as the previous experiments was used to estimate the speaker transform from either 01 or 04[2]. This factorisation mode allows very rapid adaptation to the target condition.

Table 8.3 presents the results of the speaker and noise factorisation experiments using clean-trained acoustic models. It is seen that speaker transforms estimated from either 01

[1]In principle, it is possible to estimate speaker transform from any of the 14 test sets. Unfortunately, utterances from three speakers in set C and set D were recorded by a handset microphone which limits the speech signal to telephone bandwidth. This also means it is not useful to estimate speaker transforms from set C or set D.

[2]When the clean-trained acoustic models were adapted by VTS (line 1, Table 8.3), 04 was the worst performed in set B. This trend was also observed in line 1, Table 8.4.

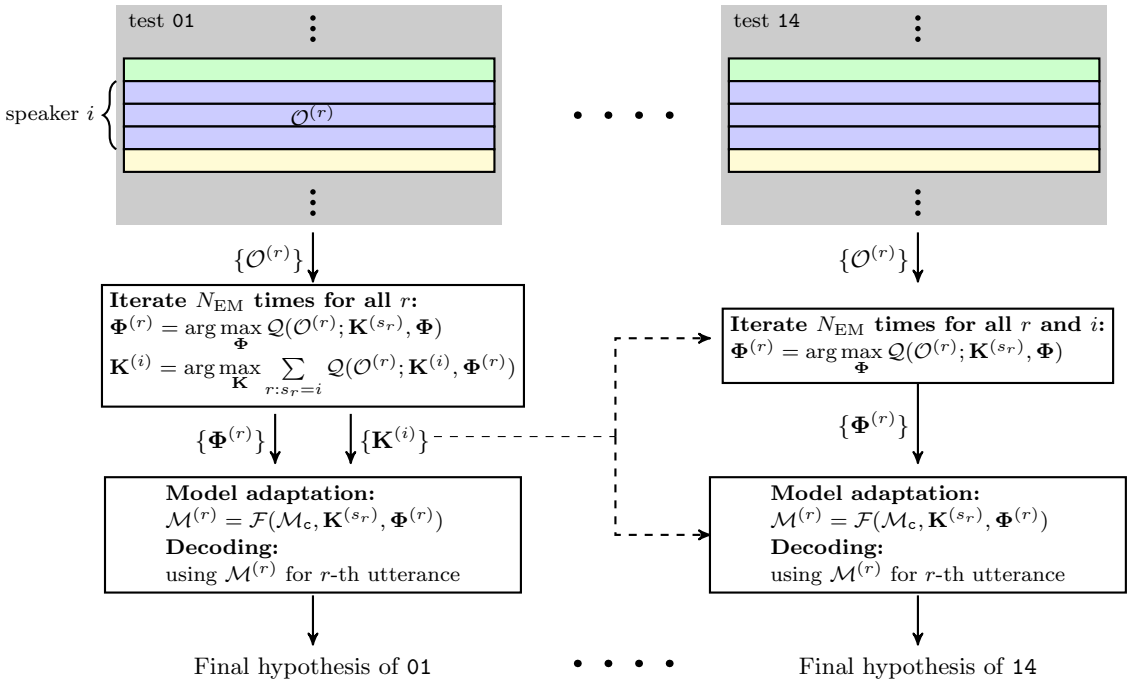


Figure 8.2: Illustration of factorised speaker and noise adaptation and decoding procedure. The dashed line shows that speaker transforms estimated from 01 are used for other test sets.

(clean) or 04 (restaurant) improve the average performance over all conditions (16.7% and 15.4% compared with 17.8%). This indicates that it is possible to factorise the speaker and noise transform to some extent. For the speaker transform estimated using 01, the "clean" data, gains in performance (compared with VTS adaptation only) for all the four sets were obtained. Interestingly the average performance was improved by estimating the speaker transform in a noisy environment, 04. Other than on the clean set A this yielded lower WERs than the clean estimated model for all of the B test sets. This indicates that although the speaker and noise transforms can be factorised to some extent, the linear transform for the speaker characteristics derived from the "Joint" scheme is still modelling some limitations in the VTS mismatch function to fully reflect the noise environment. It is also interesting to compare the results with the batch-mode system from Table 8.2. For test set B the average WER for the batch-mode "Joint" scheme was 12.1%, compared to 12.5% when the speaker MLLR transform was estimated using 04 and then fixed for all the test sets. This indicates that for these noise conditions the factorisation was fairly effective. However for the clean set A, the performance difference between the batch-mode and the factorisation mode was greater. This again indicates that the speaker transform was modelling some of the limitations of the VTS mismatch function. Results of speaker and noise factorisation using the "VTS-

Scheme	Spk. Est.	A	B	C	D	Avg.
VTS	—	6.9	15.1	11.8	23.3	17.8
VTS-MLLR	01	5.0	20.2	16.5	28.0	22.2
	04	10.2	19.7	19.7	28.0	22.5
Joint	01	5.0	14.1	10.4	22.3	16.7
	04	7.0	12.5	11.0	20.4	15.4

Table 8.3: Performance (in WER%) of speaker and noise adaptation in a factorised mode. The clean-trained acoustic model was used. The speaker level linear transform was estimated on either 01 and 04 task and applied to all the 14 tasks.

MLLR" scheme are also presented in Table 8.3. It is clear that the "VTS-MLLR" scheme does not have the desired factorisation attribute, as the linear transforms estimated from one particular noise conditions cannot generalise to other conditions. Hence, the "VTS-MLLR" scheme is not further investigated for factorised speaker and noise adaptation.

The above experiments demonstrate the factorisation attribute of "Joint" when the clean-trained acoustic models were used. To examine whether this attribute is still valid for adaptively trained acoustic models, a second set of experiments was run. VAT acoustic models were adapted by "Joint", in both batch and factorisation modes. For the latter, 01 and 04 were again used for speaker transform estimation. Results on all 14 subsets are presented in Table 8.4. Since the acoustic models are adaptively trained, improved performance is expected, compared with those in Table 8.3. Note that factorisation mode adaptation on 01(04) using the speaker transform estimated from 01(04) is equivalent to the batch-mode adaptation, thus gives identical results to bat on 01(04). The same trends as those observed in the previous experiments can be seen: a batch-mode "Joint" adaptation yielded large gains over VTS adaptation only (13.4% vs. 15.9%, average on all 4 sets), while using the speaker transform estimated on 04 achieved a very close performance, 14.1%. The advantages of using the "Joint" scheme were fairly maintained with the adaptively trained acoustic models.

It is also of interest to look at the experiments of the speaker and noise adaptation with the AFE acoustic models. Speaker adaptation for AFE model was done via an MLLR mean transform with the same block diagonal structure, again estimated at the speaker level. The AFE model was first used to generate the supervision hypothesis, following the MLLR adaptation, and then the final hypothesis was generated. Though multiple iterations of hypothesis generation and transform re-estimation could be used, it was found in the experiments the gain was minimal. In the batch-mode adaptation, speaker transforms were estimated for every single set, while for the factorisation mode, speaker transforms were estimated from 01 or 04. Results of these experiments are summarised in the first block of

Systems	Adaptation		A	B							C	D							Avg.
	Scheme	Spk. Est.	01	02	03	04	05	06	07	Avg.	08	09	10	11	12	13	14	Avg.	
AFE	—	—	8.8	13.1	15.9	20.0	18.4	15.1	17.4	16.7	19.1	24.1	27.8	31.1	30.9	28.6	29.4	28.6	21.4
	MLLR	bat	7.0	8.9	13.1	16.6	15.3	12.2	14.4	13.4	10.5	14.6	19.8	23.0	21.9	18.5	21.4	19.9	15.5
		01	7.0	18.5	20.6	25.8	24.1	20.7	21.5	21.9	21.0	28.1	32.9	37.7	35.6	32.0	33.1	33.2	25.6
		04	8.7	10.5	14.3	16.6	16.2	13.0	15.1	14.3	16.3	19.2	24.7	26.4	27.0	23.4	26.5	24.5	18.4
VAT	VTS	—	8.5	9.8	13.2	16.0	14.6	12.0	16.4	13.7	11.8	12.4	19.6	23.1	23.2	18.8	23.8	20.1	15.9
	Joint	bat	5.6	6.7	10.5	13.4	12.1	9.5	13.8	11.0	8.8	10.3	17.8	20.7	20.8	16.5	20.8	17.8	13.4
		01	5.6	7.6	12.9	17.7	14.2	11.7	16.0	13.4	11.1	11.5	19.6	24.8	22.6	19.7	23.9	20.3	15.6
		04	6.9	7.4	11.2	13.4	12.6	10.3	14.1	11.5	10.4	11.0	18.2	21.2	20.4	17.9	22.2	18.5	14.1

Table 8.4: Performance (in WER%) of speaker and noise adaptation in a factorised mode. The AFE acoustic model and the adaptively trained VAT acoustic model were used. "`bat`" means batch mode adaptation. In factorised adaptation, the speaker level linear transform was estimated on either `01` and `04` task and applied to all the 14 tasks.

Table 8.4. It can be seen that the speaker transform estimated from 01 did not generalise well to other noisy sets (WER increased from 21.4% to 25.6%), while the one estimated from 04 can generalise to other noise conditions. This suggests that for feature normalisation style trained acoustic models, the linear transform estimated from one noise data can be applied to other noise conditions for the same speaker. However examining the results in more detail shows that this factorisation using AFE is limited. A 19% relative degradation (18.4% of the factorisation mode vs 15.5% of the batch-mode) was observed. This compares to only 5% relative degradation for the "Joint" scheme. It is worth noting that batch-mode AFE with MLLR (15.5%) is still significantly worse that the "Joint" scheme run in a factorised mode on the 04 data (14.1%).

8.2 The Noise-Corrupted WSJ task

In AURORA4 task, data is well-balanced between speaker and noise factors. The number of utterances from a speaker in 6 different noise conditions are the same. In order to evaluate performance of factorised adaptation when data is distributed among acoustic factors in an unbalanced manner, a new task, the noise-corrupted WSJ task, is created to simulate practical scenarios, in which utterances from a particular noise conditions may dominate a speaker's data.

In the same manner as in creating the AURORA4 task, artificial background noise is added to the original WSJ task [52] to create this task. Both WSJ0 and WJS1 training data were used. There are in total 36,515 utterances in the training set, produced by 284 speakers. To simulated the background noise, 6 types of noise (similar to those used in AURORA4 task) were added to the training set to form a multi-condition training set with 7 noise conditions (including the clean condition). As an initial investigation, the training set was created in a speaker-noise balanced manner, i.e., utterances from each speaker were exposed to 7 noise conditions with equal probability. The SNRs in training data ranged from 20dB to 10dB. 3 evaluation sets were defined to simulate different scenarios in which factorised adaptation can be used. The first one, based on the original WSJ1 non-verbalised 5k-vocabulary development set (wsj1_dt), had 10 speakers, each comprising 40 adaptation read utterances and roughly 50 test utterances. These utterances were distorted in the same way as training utterances, except only 6 noisy environment conditions were created with the SNR ranging from 15dB to 5dB. The second one, based on the WSJ0 non-verbalised 5k-vocabulary development set (wsj0_dt), simulated a more realistic operation scenario, where a majority (75%) of adaptation utterances (400 in total) were distorted by the same noise source ("restaurant noise"), while the 410 test utterances were distorted by the 6 noise sources

with a uniform distribution. The third set, based on the WJS0 evaluation set (`wsj0_et`, or the `Nov'92` non-verbalised 5k closed test set which was used in the November 1992 ARPA WSJ evaluation) simulated the *practical enrollment* scenario: all the 321 adaptation utterances were distorted by a single noise source ("restaurant noise"), while the noise conditions of the 330 test utterances were uniformly distributed. The SNRs for the adaptation and test utterances in the three evaluation sets were ranging from 15dB to 5dB. Adaptation for the first two sets (`wsj1_dt` and `wsj0_dt`) ran in a unsupervised adaptation mode, while `wsj0_et` set ran in a supervised mode. It was assumed all the speaker and noise identities were known in advance.

8.2.1 Experimental Setup

A 39 dimensional front-end feature vector was used for the experiments, consisting of 12 MFCCs appended with the zeroth cepstral coefficients, delta and delta-delta coefficients. Cepstral mean normalisation was performed for each utterance. To verify the baseline CAT system on this task, a cross-word triphone acoustic model with 3955 tied states, 16 components per state was first ML trained on the *clean* data. Initialised by this ML system, a standard CAT acoustic model with 8 clusters was also trained, in which a 32-node regression tree was used for CAT adaptation on the speaker level. Using the clean-trained ML system, a MST system is built using single pass re-training (SPR) technique on the multi-condition training set. Initialised by this MST model, a standard CAT model with 8 clusters were also trained, where a 32-node regression tree was used for CAT adaptation on the speaker level. On top of this CAT model, fCAT model was trained to take the noise variability into account, in which the noise space consists of 7 clusters and a 256-node noise regression tree was used. Different from the experiments in the previous section, utterances corrupted with the same noise environment are considered as in the same homogeneous block. Another fCAT model was also trained without the orthogonal constraint, which represents the data constraint approach to factorisation. A 5k-vocabulary non-verbalised bi-gram language model was used throughout the experiments.

8.2.2 Results

The first set of experiments were performed on the *clean* data, in which the non-speech variability is mainly the speaker differences. The baseline performance of clean-trained system on clean test data are presented in Table 8.5. Because the training data and the test data come from the same *clean* condition, the accuracy of the unadapted ML system is already high. Using the ML acoustic model, an initial hypothesis for each of the adaptation utterances in

the unsupervised adaptation tasks, `wsj1_dt` and `wsj0_dt`, was generated. For the supervised adaptation task, `wsj0_et`, the ML acoustic model was used to generate phone-level alignments against the reference hypothesis. Using the supervision hypothesis for adaptation utterances, an interpolation weight vector is estimated for each speaker, which enables adaptive decoding using the CAT acoustic model. Compared with the performance of the ML system (shown in first row of Table 8.5), the CAT system further improves the performance (shown in the second row of Table 8.5). This demonstrates that the baseline CAT system is effective for speaker adaptation.

System	`wsj1_dt`	`wsj0_dt`	`wsj0_et`
ML	8.5	5.8	4.8
CAT	6.9	4.8	4.0

Table 8.5: Performance (in WER%) of the clean-trained ML and CAT systems on the *clean* test data. For the CAT system, cluster weights are estimated at the speaker level.

On the noise corrupted data, the main non-speech variabilities are the speaker and noise differences. Recognition of the noise corrupted data was first performed by unadapted decoding using the MST model. The first row of Table 8.6 shows the performance on three test sets. The MST model was also used in decoding the adaptation utterances in set `wsj0_dt` and `wsj1_dt` to provide the supervision hypothesis for the following adaptation run. For supervised adaptation with `wsj0_et`, the MST model was used to generate the phone alignment against the reference hypothesis. Compared with the WERs in Table 8.5, the impact of background noise is evident. It also indicates that multi-style training is not able to model all the variabilities seen in the training data.

To run adapted decoding using CAT, speaker transforms were first estimated on the adaptation utterances, and then applied on the test withs. The second row of table 8.6 shows the performance of adapted decoding using the CAT model. As expected, speaker adaptation improved the performance. To perform factorised adaptation, speaker and environment transforms were iteratively estimated on the adaptation data. In the very first iteration, environment weight vectors were set as the one-of-N vector according to the known environment type, and the speaker transforms were estimated while the component posterior was calculated using the MST model. Given the set of speaker transforms, a set of environment transforms were estimated for each environment condition appeared in the adaptation data. A few(~ 5) iterations [1] were performed. Then the speaker transform estimated from adapta-

[1]It is possible to only estimate the speaker transform while borrowing the environment transforms obtained during training as in [218]. In this work, iterative estimation of both transforms was used to investigate what a fCAT model can learn on each factor.

System	wsj1_dt	wsj0_dt	wsj0_et
MST	23.3	15.7	14.9
CAT	20.3	13.3	13.0
fCAT(w/o constr.)	19.5	13.6	14.4
fCAT	19.7	12.9	12.3

Table 8.6: Performance (in WER%) of multi-style or adaptively trained MST, CAT and fCAT systems on noise-corrupt data. "w/o constr." means the orthogonal constraint is not maintained in training a fCAT system. Utterances in test set `wsj1_dt` are distorted by 6 noise types with an equal probability; A majority (75%) of utterances in set `wsj0_dt` are distorted by the "restaurant" noise, and all the utterance in set `wsj0_et` are distorted by the "restaurant" noise.

tion data in conjunction with the environment transforms obtained during training were used in decoding. This enables the comparison of speaker transforms estimated by different fCAT systems.

fCAT systems trained with and without the independence constraint are compared in the second block of Table 8.6. On the first set, `wsj1_dt`, which is speaker-environment balanced, both systems achieved gains over the speaker-adapted CAT system (19.5% and 19.7% vs 20.3%). This illustrates the benefit of adapting acoustic models to both speaker and environment. However, when used in more realistic scenarios where the speaker-environment distribution of adaptation data is imbalanced, i.e., on the `wsj0_dt` and `wsj1_et` sets, factorised adaptation using fCAT model trained without independence constraint degraded performance, compared with the performance of speaker adaptation using CAT (13.6% vs 13.3% and 14.4% vs 13.0%). This demonstrates that the speaker transform which was estimated on imbalanced data modelled not only the speaker but also the environment variability. In contrast, factorised adaptation using fCAT trained with the independence constraint achieved gains over speaker adaptation on both sets (12.9% vs 13.3% and 12.3% vs 13.0%). The gain of this fCAT system on `wsj0_et` set is more interesting, as there is only one environment condition in the adaptation data. Other adaptation schemes using linear transforms on this set can only learn the combination effect of speaker and noise factors, not able to distinguish one factor from the other. With fCAT, as the subspace expanded by the speaker cluster is orthogonal to the subspace expanded by the environment cluster, speaker transforms learned on adaptation data can only explain the speaker distortion. This enables speaker variabilities to be factorised out from the adaptation data.

8.3 Summary

This chapter has described experiments concerning several approaches to acoustic factorisation. In the first set of experiments, the proposed "Joint" adaptation scheme, which is a transform constraint approach to factorisation, were compared with two alternatives, the "VTS-MLLR" and "AFE-MLLR" schemes, on the standard AURORA4 task. Experimental results demonstrate that if operated in a batch mode, both "VTS-MLLR" and "Joint" give gains over noise adaptation alone. However, only "Joint" supports the factorisation mode adaptation, which allows a very rapid speaker and noise adaptation. The "Joint" scheme was also compared with the scheme that uses feature-based approach to noise compensation, "AFE-MLLR". Results show "AFE-MLLR" does not achieve the same level of performance as "Joint". These results confirm that by using different forms of transform as an implicit constraint, transforms associated with different acoustic factors can be kept independent of each other.

In the second part of this chapter, the effectiveness of the proposed explicit independence constraint was evaluated on the noisy WSJ task, with the focus on test data which has a unbalanced distribution. Experimental results demonstrated that similar performance can be achieved with or without the independence constraint on a speaker-environment balanced set. However, when a majority of data is distorted by a single noise condition, the speaker weights estimated from the fCAT system trained without the independence constraint are biased towards the dominant noise environment, and thus cannot be used in a factorisation mode. However, the fCAT system with the proposed explicit independence constraint is able to factorise out the speaker variability from the environment variability even when the adaptation data is distorted by the same noise type. This demonstrates that the usefulness of the proposed explicit independence constraint. This is especially true when no prior knowledge about the concerned acoustic factors is available and the data has a unbalanced distribution over concerned acoustic factors.

Chapter 9
Conclusion

Several model-based approaches to robust speech recognition in diverse environments were investigated in this thesis. First, a novel model-based scheme for robust speech recognition in reverberant environments was proposed. In this scheme, extended model statistics were used to represent the distribution of previous clean speech vectors. Given these extended statistics and certain noise model parameters, the vector Taylor series expansion technique was extended to compensate the acoustic models for the effects caused by both background noise and reverberation. Maximum likelihood estimation of noise model parameters and adaptive training of the extended statistics were also developed. This contribution will be summarised in section 9.1. Second, to adapt the acoustic models flexibly to diverse environments, several model adaptation schemes were investigated within the acoustic factorisation framework. The investigation demonstrates that it is essential to keep the independence between factor transforms to make acoustic factorisation work. Several approaches can be used to enforce this independence. They were discussed in this thesis and also applied to speaker and noise factorisation problem in chapter 6. This contribution will be summarised in section 9.2. Some possible future works will be discussed in section 9.3.

9.1 Reverberant Environment Robustness

Speech recognition in reverberant environments is a challenging problem due to the highly dynamic nature of reverberated speech. As discussed in section 3.3.3, there are mainly two approaches to acoustic model adaptation for reverberant environments, depending on whether acoustic models are adapted statically or dynamically. Due to the reverberation effect, the current feature vector is influenced by the feature vectors of several preceding frames. This seriously invalidates the conditional independence assumption in HMM models. The advantage of dynamic model adaptation schemes such as[212, 232] lies in that the compensated acoustic model is not a traditional HMM, so those schemes are able to model the dynamic nature of reverberation to certain extent. However, this comes at an expensive price: as the acoustic model needs to be dynamically modified, usually at the frame level, decoding is computationally expensive. This makes the application of this approach in large vocabulary ASR systems impractical. On the other hand, the static model adaptation approach modifies acoustic models before recognition, and a standard HMM model set is used in decoding. Hence, the standard Viterbi search algorithm can still be employed. This static model adaptation approach for reverberant environments was explored in [103, 198]. One limitation of the methods presented in [103, 198] is that the mean vectors of the previous several frames needs to be inferred before recognition. This was achieved either by using a left-to-right state sequence or via the context of the acoustic units. The inference process using the methods in [103, 198] is rather unreliable, especially for speech frames with a large distance from the current frame. It also smooths the trajectory of speech as the previous mean vectors are inferred from the state level. This reduces the discriminant power of the compensated acoustic model. One novelty of the proposed scheme in this thesis is that extended statistics for each Gaussian component are appended with the standard acoustic model. These extended statistics represent distributions of the preceding clean speech vectors conditioned on the corresponding Gaussian component. This avoids inferring the previous hidden states and the distributions of previous clean speech vectors are modelled at the Gaussian component level, rather than at the state level. Given the distribution of previous clean speech vectors, the model parameters are compensated using the VTS approximation, rather than the log-normal approximation in [103, 198]. This yields the RVTS and RVTSJ compensation schemes, depending on how the interaction between background noises and reverberation is modelled. A maximum likelihood estimation of reverberant noise and background noise parameters can be also derived. In addition, a reverberant adaptive training scheme is used to estimate the extended statistics from the multi-condition data in a full ML framework.

Experimental results on an artificially corrupted corpus and a real noise corpus demonstrate that the proposed RVTS and RVTSJ schemes performed significantly better than the standard VTS scheme. Experimental results also show that RVTSJ was slightly better than RVTS because the reverberant noise and background noise parameters are jointly estimated in the RVTSJ scheme. It is also demonstrated that reverberant adaptively trained acoustic models performed better than multi-style trained acoustic models.

9.2 Extensions to Acoustic Factorisation Framework

Acoustic factorisation is a framework in which the variabilities of different acoustic factors are separately modelled. This offers additional flexibility in adapting acoustic models to diverse acoustic conditions. The second part of this thesis made several extensions to this framework. It is pointed out in section 5.1, the independence between factor transforms is essential to factorisation. Previous works normally rely on the attributes of acoustic data to achieve the independence while imposing no constraints on acoustic models or transforms. This thesis has proposed two extensions to the acoustic factorisation framework.

The first extension is a transform constraint approach, in which each acoustic factor is associated with a different form of factor transform. The factor transform is designed to model the nature of the corresponding acoustic factor. In this way, the maximum likelihood estimated factor transforms reflect the nature of impacts caused by the corresponding factor, thus they should be independent of each other. Applying this approach to the speaker and noise factorisation problem, a so-called "Joint" scheme is derived by combining a linear transform, the MLLR mean transform, and a non-linear transform, the model-based VTS transform. The MLLR mean transform is applied prior to the VTS transform, which matches the generation process of the speaker and noise corrupted speech, thus the MLLR mean transform will only model the speaker variability. This approach is compared with the conventional VTS-MLLR scheme, in which VTS transform is applied after MLLR. In this way, the MLLR transform is coupled with both speaker and noise conditions, therefore it cannot be used in factorised adaptation. One advantage of using the "Joint" scheme is that a speaker transform estimated from one noise condition can be used in another noise condition, which allows a very rapid speaker adaptation in noisy environments. This is demonstrated by the experimental results on the AURORA4 task. Compared with the VTS-MLLR scheme, which can only be used in batch adaptation, "Joint" supports both batch and factorised adaptation, and the performance of factorised adaptation is comparable to batch adaptation.

The transform constrained approach is based on partial knowledge of how one or several acoustic factors distort the clean speech signals. For example, the proposed "Joint" scheme

relies on a mismatch function which describes how speaker and noise factors affect the clean speech. In many practical applications, the nature of the distortions caused by concerned acoustic factors is not known. In this scenario, it is still possible to use a data constrained approach to construct factorised adaptation. However, there is no guarantee that factor transforms are independent of each other using the data constrained approach.In contrast to the data constrained and transform constrained approaches, which enforce the independence between factor transforms *implicitly*, an *explicit* independence constraint, based on a mathematical analysis of the interdependence between ML-estimated factor transforms, was proposed in this thesis. It was also applied to speaker and noise factorisation for speech recognition, in which the cluster adaptive training (CAT) scheme was generalised to the factorised CAT (fCAT) scheme. An orthogonal constraint of the speaker and noise subspaces was derived as a result of the proposed independence constraint. Experimental results on an artificially corrupted WSJ task demonstrated that fCAT with the independence constraint is able to separate the speaker and noise variability, even when all the data are corrupted by the same type of noise.

9.3 Future Work

There are a number of possible directions to be further explored along the line of research works presented in this thesis. Some of these directions are summarised in the following:

- **Reverberant Environment Robustness:**
  There are a few issues and limitations of the proposed model-based schemes that need to be addressed. For example, only the mean vector is compensated for the effect of both reverberation and background noise in the proposed scheme, while the variance is compensated using standard VTS scheme, which does not take reverberation into account. As reverberant speech is highly dynamic and usually highly correlated with the original clean speech, the variance compensation is important to reverberant model robustness.

  Another limitation of the proposed scheme is that it is not easy to be combined with front-end schemes, as it makes strong assumptions of the relationship between corrupted speech and the clean speech. However, in a state-of-art ASR system, a set of feature transforms, e.g., CMN/CVN and HLDA, are usually applied after feature extraction. Moreover, for acoustic model robustness in reverberant environments, front-end processing schemes such as linear filtering, feature de-reverberation, have been demonstrated to be very effective, especially when a microphone array is used. Therefore, it is also

interesting to combine the proposed model-based robustness schemes with the front-end based processing schemes.

- **Acoustic Factorisation:**
  There are also several research directions which can be explored to extend the acoustic factorisation framework. For example, the proposed explicit independence constraint is only applied to a simple model, the fCAT model. It will be interesting to apply this constraint to more complex and powerful adaptation schemes or combine this explicit constraint approach with the transform constrained approach.

  The concept of acoustic factorisation can be applied to a wider domain. For example, speaker recognition will benefit if it is possible to factorise speaker and session variability. In fact, as already discussed in Chapter 6, the proposed fCAT is similar to the famous Joint Factor Analysis (JFA) technique for speaker recognition, and fCAT can be also applied to enforce the orthogonality between noise and speaker subspace. Acoustic factorisation is also an important topic for speech synthesis. Besides intelligibility of synthesised speech, more and more applications of speech synthesis put emphasis on the quality of naturalness and expressiveness. This requires a versatile speech synthesis system which is able to simulate effects of many acoustic factors, such as speakers, languages, and emotion. Using independent factor transforms constructed under the acoustic factorisation framework is therefore an appealing approach to this problem.

  Another research direction to apply the concept of acoustic factorisation to the recently popular deep neural network approach for speech recognition. Rather that using a single neural network to model all the desired and unwanted variabilities, it would be attractive to use separate modules or sub-networks to represent various acoustic factors. In this case, the independence between each modularised sub-network is still the key to factorisation.

Appendix A
Derivation of mismatch functions

In chapter 4, the impact of environments on the ASR systems are discussed. In particular, the impact of background noises, short-term convolutional noises and reverberation on the extracted feature vectors for ASR systems are usually summarised as mismatch functions. In this appendix, the feature extraction procedure used in ASR systems is first described in detail. This is followed by the derivation of mismatch functions for additive noise and reverberant noise respectively.

A.1 Feature extraction procedure

In the feature extraction stage, the input speech signal (in the time domain) is analysed and a sequence of feature vectors is extracted for better speech representation. Among various representation forms used, Mel frequency cepstrum coefficients (MFCC)[39] is the most popular and widely used one. The procedure to extract MFCC features is standardised by the ETSI document in [51]. In order to focus on the functional relationship between features extracted in different stages, the following procedure describes a slightly modified procedure. Here, it is assumed the input signal is not corrupted by noise.

step 1. The input signal is offset compensated, pre-emphasised. This gives a time signal $x(t)$.

step 2. $x(t)$ is framed and weighted by an analysis window function $w(t)$ with a finite length L_w. After this step, a frame dependent and windowed signal segment $\tilde{x}(m,t)$ is obtained, where

$$\tilde{x}(m,t) = w(t)x(t+mB) \qquad t = 0, \cdots, L_w - 1 \tag{A.1}$$

and $w(t) = 0$ if $t < 0$ or $t > L_w$, B is the frame shift.

step 3. A short-time discrete Fourier transform (STFT) is applied on each signal segment, i.e.,

$$\tilde{X}(m,b) = \sum_{t=0}^{L_w-1} \tilde{x}(m,t) e^{-j\frac{2\pi b}{L_b}t} \tag{A.2}$$

where b is frequency bin index, $b = 0, \cdots, L_b - 1$.

step 4. A bank of L_f overlapping triangular filters, spaced equally on the Mel scale, are used to filter the power spectrum, i.e.,

$$X(m,k) = \sum_{b=0}^{L_b-1} \Lambda_k(b) |\tilde{X}(m,b)|^2 \tag{A.3}$$

where $k = 0, \cdots, L_f$, $\Lambda_k(b) - 1$ is the k-th filter weight for the b-th frequency bin and $\sum_{b=0}^{L_b-1} \Lambda_k(b) = 1$. The above equation assumes the power spectrum is used. In many implementations, the magnitude spectrum is used, i.e.,

$$X(m,k) = \sum_{b=0}^{L_b-1} \Lambda_k(b) |\tilde{X}(m,b)| \tag{A.4}$$

step 5. Mel spectral coefficients are nonlinear transformed by the logarithmic function, yielding the logarithmic mel-spectral features:

$$x_{m,k}^{(1)} = \log X(m,k) \tag{A.5}$$

step 6. The logarithmic mel-spectral features are decorrelated by a discrete cosine transform (DCT), yielding the well-known MFCCs:

$$x_{m,c} = \sum_{k=0}^{L_f-1} x_{m,k}^{(1)} \cos\left[\frac{c\pi}{L_c}(k+0.5)\right] \tag{A.6}$$

where $c = 0, \cdots, L_c - 1$ and L_c is the number of cepstral coefficients. Usually, L_c is less than the number of filters L_f.

As a summary, Table A.1 lists the meaning of symbols used in the above procedure, and some commonly used parameters in MFCC feature extraction.

Symbol	Meaning	Typical value/Note
m	frame index	—
$\tilde{x}(m,t)$	framed and window weighted time signals	$t = 0, \ldots, L_w - 1$
$w(t)$	window function	
L_w	length of window function	$L_w = 0.025f$
B	frame shift	$B = 0.01f$
f	signal sampling frequency	f=16/11/8kHz
$\tilde{X}(m,b)$	STFT of input signals	$b = 1 \cdots, L_b$
L_b	number of frequency bins	$L_b = 256$ or 512
$X(m,k)$	Mel power/magnitude spectral coefficients	$k = 1 \cdots, L_f$
L_f	number of Mel filters	$L_f = 24$
$x^{(1)}_{m,k}$	logarithmic mel-spectral features	$k = 1 \cdots, L_f$
$x_{m,c}$	MFCC	$c = 1 \cdots, L_c$
L_c	number of cepstral coefficients	$L_c = 13$

Table A.1: Meanings of symbols and typical values used in MFCC feature extraction.

A.2 Mismatch function for additive noise and convolutional distortion

In the case of additive noise $n(t)$ and short-term convolutional distortion $h(t)$ presented in the environments, the relationship of the corrupted time signal $y(t)$ between $n(t)$ and $h(t)$ can be expressed by:

$$y(t) = h(t) * x(t) + n(t) = \sum_{\tau=0}^{L_h-1} h(\tau)x(t-\tau) + n(t) \; . \tag{A.7}$$

Note that it is assumed that the convolutional distortion is stationary, i.e., the convolutional characteristics $h(\tau)$ does not vary with respect to t. It is also assumed $h(\tau)$ is a short-term signal compared with the length of analysis window, i.e., $L_h \ll L_w$.

Hence, the m-th windowed time signal segment $\tilde{y}(m,t)$ become

$$\begin{aligned} \tilde{y}(m,t) &= w(t)n(t+mB) + w(t) \sum_{\tau=0}^{L_h-1} h(\tau)x(t+mB-\tau) \\ &\approx \tilde{n}(m,t) + h(t) * \tilde{x}(m,t) \end{aligned} \tag{A.8}$$

The approximation is possible due to the fact $L_h \ll L_w$.

Due to the relationship in the time domain as it is shown in Eq. (A.8), the STFT representation of $y(t)$ can be written as:

$$\tilde{Y}(m,b) = \tilde{N}(m,b) + \tilde{H}(b)\tilde{X}(m,b) \tag{A.9}$$

where $N(m,b)$ and $H(b)$ are the STFT representation of $\tilde{n}(m,t)$ and $\tilde{n}(m,t)$ respectively.

In the mel-spectral filter bank domain, depending on the which spectrum (power vs. magnitude) is used for the Mel filter bank, different assumptions can be made to link the noisy speech mel-spectral coefficients with the ones of noise:

- If the power spectrum is used,

$$\begin{aligned} Y(m,k) &= \sum_{b=0}^{L_b-1} \Lambda_k(b)|\tilde{H}(b)\tilde{X}(m,b) + \tilde{N}(m,b)|^2 \\ &= \sum_{b=0}^{L_b-1} \Lambda_k(b)\left[|\tilde{N}(m,b)|^2 + |\tilde{X}(m,b)\tilde{H}(b)|^2 + 2\cos(\theta_b)|\tilde{N}(m,b)||\tilde{X}(m,b)\tilde{H}(b)|\right] \\ &= N(m,k) + X(m,k)H(k) + 2\alpha_k\sqrt{N(m,k)X(m,k)H(k)} \end{aligned} \tag{A.10}$$

  where $N(m,k)$, $X(m,k)$ and $H(k)$ are the k-th Mel filter output of signals $n(t)$, $x(t)$ and $h(t)$ at frame m respectively, θ_b is a (random) angle between two complex signals $\tilde{H}(b)\tilde{X}(m,b)$ and $\tilde{N}(m,b)$ and

$$\alpha_k = \frac{\sum_{b=0}^{L_b-1}\Lambda_k(b)\cos(\theta_b)|\tilde{N}(m,b)\tilde{H}(b)\tilde{X}(m,b)|}{\sqrt{N(m,k)H(k)X(m,k)}}$$

- Similarly, if the magnitude spectrum is used,

$$\begin{aligned} Y^2(m,k) &= \left(\sum_{b=0}^{L_b-1} \Lambda_k(b)|\tilde{H}(b)\tilde{X}(m,b) + \tilde{N}(m,b)|\right)^2 \\ &\approx \sum_{b=1}^{L_b-1} \Lambda_k(b)|\tilde{H}(b)\tilde{X}(m,b) + \tilde{N}(m,b)|^2 \qquad \text{(A.11)} \\ &= \sum_{b=0}^{L_b-1} \Lambda_k(b)\left[|\tilde{N}(m,b)|^2 + |\tilde{X}(m,b)\tilde{H}(b)|^2 + 2\cos(\theta_b)|\tilde{N}(m,b)||\tilde{X}(m,b)\tilde{H}(b)|\right] \\ &= N^2(m,k) + X^2(m,k)H^2(k) + 2\alpha_k N(m,k)X(m,k)H(k) \qquad \text{(A.12)} \end{aligned}$$

  and

$$\alpha_k = \frac{\sum_{b=0}^{L_b-1}\Lambda_k(b)\cos(\theta_b)|\tilde{N}(m,b)\tilde{H}(b)\tilde{X}(m,b)|}{N(m,k)H(k)X(m,k)}$$

  The approximation in Eq. (A.11) is possible due to $\sum_{b=0}^{L_b-1}\Lambda_k(b) = 1$ and the assumption that the variation of $\tilde{Y}(m,b)$ within each of the mel filter k is small[62].

With the above the mismatch functions in the mel-spectral domain, it is easy to verify that in the MFCC domain:

- If the Mel filter bank is operated on the power spectrum:

$$\exp\left(\mathbf{C}^{-1}\boldsymbol{y}_m\right) = \exp\left(\mathbf{C}^{-1}(\boldsymbol{x}_m + \boldsymbol{\mu}_{\mathtt{h}})\right) + \exp\left(\mathbf{C}^{-1}\boldsymbol{n}_m\right) + 2\boldsymbol{\alpha}\cdot\exp\left(\frac{1}{2}\mathbf{C}^{-1}(\boldsymbol{x}_m + \boldsymbol{\mu}_{\mathtt{h}} + \boldsymbol{n}_m)\right) \tag{A.13}$$

  where log and exp are element-wise operations, $\boldsymbol{x}_m$, $\boldsymbol{\mu}_{\mathtt{h}}$ and $\boldsymbol{n}_m$ are the MFCC representations of noisy speech, convolutional distortion and noise at frame m, $\boldsymbol{\alpha} = (\alpha_0, \cdots, \alpha_{L_f-1})^{\mathsf{T}}$, $\mathbf{C}$ is the DCT matrix with $(\mathbf{C})_{ij} = \cos(\frac{i\pi}{L_c}(j+0.5))$, $\mathbf{C}^{-1}$ is the inverse matrix of $\mathbf{C}$. In case that $\mathbf{C}$ is a non-square matrix, $\mathbf{C}^{-1}$ is the Moore-Penrose inverse of $\mathbf{C}$.

- If the Mel filter bank is operated on the magnitude spectrum:

$$\exp\left(2\mathbf{C}^{-1}\boldsymbol{y}_m\right) = \exp\left(2\mathbf{C}^{-1}(\boldsymbol{x}_m + \boldsymbol{\mu}_{\mathtt{h}})\right) + \exp\left(2\mathbf{C}^{-1}\boldsymbol{n}_m\right) + 2\boldsymbol{\alpha}\cdot\exp\left(\mathbf{C}^{-1}(\boldsymbol{x}_m + \boldsymbol{\mu}_{\mathtt{h}} + \boldsymbol{n}_m)\right) \tag{A.14}$$

In the above equations (A.13,A.14), the "phase factor" $\boldsymbol{\alpha}$ ([44, 47, 154, 155]) is included. However, it is hard to accurately estimate. In the standard approach, this phase factor is either assumed to be 1.0 (the noise and the filtered speech always has the same phase) or 0.0 (the noise and the filtered speech are orthogonal). This gives three mismatch functions:

- If the filter bank is operated on the power spectrum and $\boldsymbol{\alpha} = \mathbf{0}$ or if the filter bank is operated on the magnitude spectrum and $\boldsymbol{\alpha} = \mathbf{1}$:

$$\boldsymbol{y}_m = \mathbf{C}\log\left(\exp(\mathbf{C}^{-1}(\boldsymbol{x}_m + \boldsymbol{\mu}_{\mathtt{h}})) + \exp(\mathbf{C}^{-1}\boldsymbol{n}_t)\right) \tag{A.15}$$

- If the filter bank is operated on the magnitude spectrum and $\boldsymbol{\alpha} = \mathbf{0}$:

$$\boldsymbol{y}_m = \frac{1}{2}\mathbf{C}\log\left(\exp(2\mathbf{C}^{-1}(\boldsymbol{x}_m + \boldsymbol{\mu}_{\mathtt{h}})) + \exp(2\mathbf{C}^{-1}\boldsymbol{n}_t)\right) \tag{A.16}$$

- If the filter bank is operated on the power spectrum and $\boldsymbol{\alpha} = \mathbf{1}$:

$$\boldsymbol{y}_m = 2\mathbf{C}\log\left(\exp(\frac{1}{2}\mathbf{C}^{-1}(\boldsymbol{x}_m + \boldsymbol{\mu}_{\mathtt{h}})) + \exp(\frac{1}{2}\mathbf{C}^{-1}\boldsymbol{n}_t)\right) \tag{A.17}$$

Alternatively, [62] proposed another form of mismatch function:

$$\boldsymbol{y}_m = \frac{1}{\gamma}\mathbf{C}\log\left(\exp(\gamma\mathbf{C}^{-1}(\boldsymbol{x}_m + \boldsymbol{\mu}_{\mathtt{h}})) + \exp(\gamma\mathbf{C}^{-1}\boldsymbol{n}_t)\right) \tag{A.18}$$

Clearly, the above three mismatch functions are the special forms of Eq. (A.18). The Mismatch function in Eq. (A.18) allows combining the energy of speech and noise in various domains (power, magnitude or others).

Eq. (A.15) is the standard mismatch function that has been widely used. Despite the mismatch function has the same form for filter bank operated in power and magnitude spectrum, it is worthwhile to point out different assumptions and approximations have been made to achieve the same form of mismatch function.

A.3 Mismatch function for reverberation and additive noise

As discussed in section 3.3.1.1, when both the reverberation and additive noise are presented in the environments, the relationship between the room impulse response (RIR) $h_r(t)$ additive noise $n(t)$ clean speech signal $x(t)$ and corrupted signal $z(t)$ can be written as:

$$z(t) = \sum_{\tau=0}^{L_r-1} h_r(\tau)x(t-\tau) + n(t) \tag{A.19}$$

Note the length of RIR $L_r = L_H \cdot B$ is much longer than the length of analysis window L_w, therefore the approximation in Eq. (A.8) cannot be used anymore.

Instead, $h_r(t)$ can be decomposed into a series of delayed impulse responses, i.e.,

$$h_r(t) = \sum_{l=0}^{L_H-1} h'_l(t+lB) \tag{A.20}$$

where $h'_l(\tau) = 0$ if $\tau \notin [0, L_w)$. Using this decomposition, it can be shown that

$$\begin{aligned}
\tilde{z}(m,t) &= w(t) \sum_{\tau=0}^{L_r-1} h_r(\tau)x(t+mB-\tau) + w(t)n(t+mB) \\
&= w(t) \sum_{l=0}^{L_H-1} \sum_{\tau=0}^{L_w-1} h'_l(\tau+lB)x(t-\tau+mB) + w(t)n(t+mB) \\
&\approx \sum_{l=0}^{L_H-1} h'_l(\tau) * \tilde{x}(m-l,t) + \tilde{n}(m,t)
\end{aligned} \tag{A.21}$$

Therefore in the STFT domain:

$$\tilde{Z}(m,b) = \sum_{l=0}^{L_H-1} \tilde{H}_l(b)\tilde{X}(m-l,b) + \tilde{N}(m,b) \tag{A.22}$$

where $\tilde{Z}$, $\tilde{H}_l$, $\tilde{X}$, $\tilde{N}$ are the STFT representation of time signal $\tilde{z}(m,t)$, $\tilde{h}'_l(\tau)$, $\tilde{x}(m,t)$ and $\tilde{n}(m,t)$ respectively.

Furthermore,

$$
\begin{aligned}
|\tilde{Z}(m,b)|^2 &= \sum_{l=0}^{L_H-1}\sum_{l'=0}^{L_H-1} \alpha_{\mathtt{x}}(l,l',b)|\tilde{H}_l(b)\tilde{H}_{l'}(b)\tilde{X}(m-l,b)\tilde{X}(m-l,b')| \\
&\quad + 2\sum_{l=0}^{L_H-1} \alpha_{\mathtt{xn}}(l,b)|\tilde{H}_l(b)\tilde{X}(m-l,b)\tilde{N}(m,b)| + |\tilde{N}(m,b)|^2 \\
&\approx \sum_{l=0}^{L_H-1} |\tilde{H}_l(b)\tilde{X}(m-l,b)|^2 + 2\sum_{l=0}^{L_H-1} \alpha_{\mathtt{xn}}(l,b)|\tilde{H}_l(b)\tilde{X}(m-l,b)\tilde{N}(m,b)| + |\tilde{N}(m,b)|^2
\end{aligned}
\tag{A.23}
$$

where $\alpha_{\mathtt{xn}}(l,b)$ the phase factor between $\tilde{H}_l(b)\tilde{X}(m-l,b)$ and $\tilde{N}(m,b)$, $\alpha_{\mathtt{x}}(l,l',b)$ the phase factor between $\tilde{H}_l(b)\tilde{X}(m-l,b)$ and $\tilde{H}_{l'}(b)\tilde{X}(m-l',b)$. The approximation was made based on the assumption that the cross frame phase factor is zero, i.e., $\alpha_{\mathtt{x}}(l,l',b) \approx 0, \forall l \neq l'$

Assuming that the Mel filter bank is operating on the magnitude spectrum, using the same approximation in the previous section, it can be shown that

$$
\begin{aligned}
Z^2(m,k) &= \left(\sum_{b=0}^{L_b-1} \Lambda_k(b)|\tilde{Z}(m,b)|\right)^2 \approx \sum_{b=0}^{L_b-1} \Lambda_k(b)|\tilde{Z}(m,b)|^2 \\
&= \sum_{l=0}^{L_H-1} \left(H_l(k)X(m-l,k)\right)^2 + N^2(m,k) + 2\alpha\left(\sum_{l=0}^{L_H-1} H_l(k)X(m-l,k)\right)N(m,k) \\
&= \left(\sum_{l=0}^{L_H-1} H_l(k)X(m-l,k)\right)^2 + N^2(m,k) + 2\alpha\left(\sum_{l=0}^{L_H-1} H_l(k)X(m-l,k)\right)N(m,k)
\end{aligned}
\tag{A.24}
$$

where

$$
\begin{aligned}
H_l(k) &= \frac{\sum_{b=0}^{L_b-1} \Lambda_k(b)|\tilde{H}_l(b)\tilde{X}(m-l,b)|^2}{\sum_{b=0}^{L_b-1} \Lambda_k(b)|\tilde{X}(m-l,b)|^2} \\
\alpha &= \frac{\sum_{b=0}^{L_b-1} \Lambda_k(b) \sum_{l=0}^{L_H-1} \alpha_{\mathtt{xn}}(l,b)|\tilde{H}_l(b)\tilde{X}(m-l,b)\tilde{N}(m,b)}{\left(\sum_{l=0}^{L_H-1} H_l(k)X(m-l,k)\right)N(m,k)}
\end{aligned}
$$

Depending on the assumption of phase factor α, different approximations can be made:

- If α is assumed to be 1,

$$
Z(m,k) = \sum_{l=0}^{L_H-1} H_l(k)X(m-l,k) + N(m,k) \tag{A.25}
$$

- If α is assumed to be 0,

$$
Z(m,k)^2 = (\sum_{l=0}^{L_H-1} H_l(k)X(m-l,k))^2 + N^2(m,k) \tag{A.26}
$$

This leads to two mismatch functions for reverberation and additive noise case used in this thesis:

$$\boldsymbol{z}_m = \mathbf{C} \log \left(\sum_{l=0}^{L_H-1} \exp(\mathbf{C}^{-1}(\boldsymbol{x}_{m-l} + \boldsymbol{h}_l)) + \exp(\mathbf{C}^{-1}\boldsymbol{n}_m) \right) \tag{A.27}$$

or

$$\boldsymbol{z}_m = \frac{1}{2}\mathbf{C} \log \left(\sum_{l=0}^{L_H-1} \exp(2\mathbf{C}^{-1}(\boldsymbol{x}_{m-l} + \boldsymbol{h}_l)) + \exp(2\mathbf{C}^{-1}\boldsymbol{n}_m) \right) \tag{A.28}$$

APPENDIX B

Implications of MAX Assumption

In section 6.2.3, a toy problem is examined to illustrate the speaker and noise factorisation can be achieved using the transform constrained approach. The MAX assumption proposed in [178] was used to illustrate the independence between speaker and noise estimates. In this appendix, a proof is provide for the implications of MAX assumption used in section 6.2.3.

Let the speech vector $\boldsymbol{x}$ and the noise vector $\boldsymbol{n}$ follow Gaussian distributions $\mathcal{N}(\boldsymbol{\mu}_{\mathrm{x}}, \boldsymbol{\Sigma}_{\mathrm{x}})$ and $\mathcal{N}(\boldsymbol{\mu}_{\mathrm{n}}, \boldsymbol{\Sigma}_{\mathrm{n}})$ respectively, and the corrupted speech vector $\boldsymbol{y}$ is given by:

$$\boldsymbol{y} = \mathbf{f}(\boldsymbol{x}, \boldsymbol{n}) = \mathbf{C} \log \left(\exp(\mathbf{C}^{-1}\boldsymbol{x}) + \exp(\mathbf{C}^{-1}\boldsymbol{n}) \right) \tag{B.1}$$

where $\boldsymbol{x} \in \mathcal{R}^d$, $\mathbf{C}$ is the (truncated) DCT matrix with the size $d \times d_L$, and d_L is the number of filter banks; $\mathbf{C}^{-1}$ is the Moore-Penrose pseudo inverse matrix of $\mathbf{C}$, i.e., $\mathbf{C}^{\mathsf{T}}$.

Using the VTS approximation, it can be shown that

$$\boldsymbol{\mu}_{\mathrm{y}} = \mathcal{E}\{\boldsymbol{y}\} = \mathbf{f}(\boldsymbol{\mu}_{\mathrm{x}}, \boldsymbol{\mu}_{\mathrm{n}}) \tag{B.2}$$

$$\boldsymbol{\Sigma}_{\mathrm{y}} = \mathcal{E}\{\boldsymbol{y}\boldsymbol{y}^{\mathsf{T}}\} - \boldsymbol{\mu}_{\mathrm{y}}\boldsymbol{\mu}_{\mathrm{y}}^{\mathsf{T}} = \mathbf{J}_{\mathrm{x}}\boldsymbol{\Sigma}_{\mathrm{x}}\mathbf{J}_{\mathrm{x}}^{\mathsf{T}} + \mathbf{J}_{\mathrm{n}}\boldsymbol{\Sigma}_{\mathrm{n}}\mathbf{J}_{\mathrm{n}}^{\mathsf{T}} \tag{B.3}$$

where $\mathbf{J}_{\mathrm{x}}$ and $\mathbf{J}_{\mathrm{n}}$ are the Jacobian matrix calculated by

$$\mathbf{J}_{\mathrm{x}} = \mathbf{C} \cdot \mathrm{diag}(\boldsymbol{u})\mathbf{C}^{-1} \qquad \mathbf{J}_{\mathrm{n}} = \mathbf{I} - \mathbf{J}_{\mathrm{x}} \tag{B.4}$$

and $\boldsymbol{u}$ is a d_L-dimensional vector with its q-th element given by

$$u_q = \frac{e_{\mathtt{x},q}}{e_{\mathtt{x},q} + e_{\mathtt{n},q}} \tag{B.5}$$

and

$$e_{\mathtt{x},q} = \exp(\mathbf{c}_q^{\mathsf{T}} \boldsymbol{\mu}_{\mathtt{x}}) \qquad e_{\mathtt{n},q} = \exp(\mathbf{c}_q^{\mathsf{T}} \boldsymbol{\mu}_{\mathtt{n}}) \tag{B.6}$$

The MAX approximation assumes that for each filter bank, either the speech energy or the noise energy dominates, i.e.,

$$e_{\mathtt{y},q} = e_{\mathtt{x},q} + e_{\mathtt{n},q} \approx \max(e_{\mathtt{x},q}, e_{\mathtt{n},q}) \tag{B.7}$$

The following will demonstrate that this leads the following approximations:

$$\mathbf{Z}_{\mathtt{bn}} = -(\mathbf{J}_{\mathtt{x}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{x}})^{-1} \mathbf{J}_{\mathtt{x}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{n}} \approx 0 \tag{B.8}$$

$$\mathbf{Z}_{\mathtt{nb}} = -(\mathbf{J}_{\mathtt{n}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{n}})^{-1} \mathbf{J}_{\mathtt{n}}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{y}}^{-1} \mathbf{J}_{\mathtt{x}} \approx 0 \tag{B.9}$$

Following the MAX approximation, it is obvious that $u_q = 0$ or $u_q = 1$. Therefore, there exists a rotation matrix $\mathbf{R} \in \mathcal{R}^{d_L \times d_L}$ such that:

$$\mathrm{diag}(\boldsymbol{u}) = \mathbf{R} \begin{bmatrix} \mathbf{I}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{R}^{\mathsf{T}} \tag{B.10}$$

where $\mathbf{I}_1$ is an identity matrix with the size d_1, and d_1 is the number of speech-dominated filter banks. Note since $\mathbf{R}$ is a rotation matrix, $\mathbf{R}\mathbf{R}^{\mathsf{T}} = \mathbf{I}$. Therefore, it can be shown that

$$\mathbf{J}_{\mathtt{x}} = \mathbf{C}\mathbf{R} \begin{bmatrix} \mathbf{I}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} (\mathbf{C}\mathbf{R})^{\mathsf{T}}, \qquad \mathbf{J}_{\mathtt{n}} = \mathbf{C}\mathbf{R} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_2 \end{bmatrix} (\mathbf{C}\mathbf{R})^{\mathsf{T}} \tag{B.11}$$

where $\mathbf{I}_2$ is also an identity matrix, whose size is $d_L - d_1$.

In this way, the covariance matrix of $\boldsymbol{y}$ becomes

$$\begin{aligned}
\boldsymbol{\Sigma}_{\mathtt{y}} &= \mathbf{J}_{\mathtt{x}} \boldsymbol{\Sigma}_{\mathtt{x}} \mathbf{J}_{\mathtt{x}}^{\mathsf{T}} + \mathbf{J}_{\mathtt{n}} \boldsymbol{\Sigma}_{\mathtt{n}} \mathbf{J}_{\mathtt{n}}^{\mathsf{T}} \\
&= \mathbf{C}\mathbf{R} \left(\begin{bmatrix} \mathbf{I}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} (\mathbf{C}\mathbf{R})^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{x}} \mathbf{C}\mathbf{R} \begin{bmatrix} \mathbf{I}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \right) (\mathbf{C}\mathbf{R})^{\mathsf{T}} \\
&\quad + \mathbf{C}\mathbf{R} \left(\begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_2 \end{bmatrix} (\mathbf{C}\mathbf{R})^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{n}} \mathbf{C}\mathbf{R} \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_2 \end{bmatrix} \right) (\mathbf{C}\mathbf{R})^{\mathsf{T}} \\
&= \mathbf{C}\mathbf{R} \begin{bmatrix} \tilde{\boldsymbol{\Sigma}}_{\mathtt{x},11} & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\Sigma}}_{\mathtt{n},22} \end{bmatrix} (\mathbf{C}\mathbf{R})^{\mathsf{T}}
\end{aligned} \tag{B.12}$$

where

$$\tilde{\boldsymbol{\Sigma}}_{\mathtt{x}} = (\mathbf{C}\mathbf{R})^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{x}} \mathbf{C}\mathbf{R} \tag{B.13}$$

$$\tilde{\boldsymbol{\Sigma}}_{\mathtt{n}} = (\mathbf{C}\mathbf{R})^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{n}} \mathbf{C}\mathbf{R} \tag{B.14}$$

and $\tilde{\mathbf{\Sigma}}_{\mathtt{x},11}$ and $\tilde{\mathbf{\Sigma}}_{\mathtt{n},22}$ are the corresponding sub-matrices of $\tilde{\mathbf{\Sigma}}_{\mathtt{x}}$ and $\tilde{\mathbf{\Sigma}}_{\mathtt{n}}$ respectively. It is then possible to show that

$$\begin{aligned}
\mathbf{J}_{\mathtt{x}}^{\mathsf{T}}\mathbf{\Sigma}_{\mathtt{y}}^{-1}\mathbf{J}_{\mathtt{n}} = \mathbf{CR}\begin{bmatrix}\mathbf{I}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}(\mathbf{CR})^{\mathsf{T}}\cdot \\
\left(\mathbf{CR}\begin{bmatrix}\tilde{\mathbf{\Sigma}}_{\mathtt{x},11} & \mathbf{0}\\ \mathbf{0} & \tilde{\mathbf{\Sigma}}_{\mathtt{n},22}\end{bmatrix}(\mathbf{CR})^{\mathsf{T}}\right)^{-1}\cdot\mathbf{CR}\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}_2\end{bmatrix}(\mathbf{CR})^{\mathsf{T}} \\
\approx \mathbf{CR}\begin{bmatrix}\mathbf{I}_1 & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\cdot\begin{bmatrix}\mathbf{\Sigma}_{\mathtt{x},11}^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{\Sigma}_{\mathtt{n},11}^{-1}\end{bmatrix}\cdot\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{I}_2\end{bmatrix}(\mathbf{CR})^{\mathsf{T}} \\
= \mathbf{0}
\end{aligned} \tag{B.15}$$

Note the second equation uses the approximation

$$(\mathbf{C\Sigma C}^{\mathsf{T}})^{-1} \approx \mathbf{C\Sigma}^{-1}\mathbf{C}^{\mathsf{T}}\ , \tag{B.16}$$

in which $\mathbf{\Sigma}$ is a d_L-by-d_L invertible matrix, and $\mathbf{C}$ is a truncated DCT matrix with the size $d \times d_L$.

Therefore,

$$\mathbf{Z}_{\mathtt{bn}} = -(\mathbf{J}_{\mathtt{x}}^{\mathsf{T}}\mathbf{\Sigma}_{\mathtt{y}}^{-1}\mathbf{J}_{\mathtt{x}})^{-1}\mathbf{J}_{\mathtt{x}}^{\mathsf{T}}\mathbf{\Sigma}_{\mathtt{y}}^{-1}\mathbf{J}_{\mathtt{n}} \approx 0 \tag{B.17}$$

$$\mathbf{Z}_{\mathtt{nb}} = -(\mathbf{J}_{\mathtt{n}}^{\mathsf{T}}\mathbf{\Sigma}_{\mathtt{y}}^{-1}\mathbf{J}_{\mathtt{n}})^{-1}\mathbf{J}_{\mathtt{n}}^{\mathsf{T}}\mathbf{\Sigma}_{\mathtt{y}}^{-1}\mathbf{J}_{\mathtt{x}} \approx 0 \tag{B.18}$$

APPENDIX C

Maximum Likelihood Estimation for fCAT

In section 6.3, the factor CAT (fCAT) model is used as an application of the explicitly constrained approach to speaker and noise factorisation. In this chapter, details of maximum likelihood estimation for fCAT, including the estimation of both canonical model parameters and the transform parameters, are presented.

Given R utterances in the training data, $\{\mathbf{O}^{(r)}|r=1,\cdots,R\}$, the log-likelihood function of the canonical model $\mathcal{M}_{\mathtt{c}}$ and a set of speaker/noise transforms ($\boldsymbol{\mathcal{T}}_{\mathtt{s}}=\{\mathcal{T}_{\mathtt{s}}^{(i)}|i=1,\cdots,I\}$ and $\boldsymbol{\mathcal{T}}_{\mathtt{n}}=\{\mathcal{T}_{\mathtt{n}}^{(j)}|j=1,\cdots,J\}$) can be expressed by:

$$\mathcal{L}(\mathcal{M}_{\mathtt{c}},\boldsymbol{\mathcal{T}}_{\mathtt{s}},\boldsymbol{\mathcal{T}}_{\mathtt{n}})=\sum_{r}\sum_{\boldsymbol{\theta}^{(r)}}p(\boldsymbol{\theta}^{(r)}|\mathcal{M}_{\mathtt{c}},\mathcal{T}_{\mathtt{n}}^{(n_r)},\mathcal{T}_{\mathtt{s}}^{(s_r)})\prod_{t}p(\boldsymbol{o}_t^{(r)};\theta_t^{(r)},\mathcal{M}_{\mathtt{c}},\mathcal{T}_{\mathtt{s}}^{(s_r)},\mathcal{T}_{\mathtt{n}}^{(n_r)}) \tag{C.1}$$

where s_r and n_r are the indices of the speaker and noise conditions of the r-th utterance, $\boldsymbol{\theta}^{(r)}=[\theta_1^{(r)},\cdots,\theta_T^{(r)}]$ denotes the r-th sequence of underlying Gaussian components, and the m-th Gaussian distribution in the s_r-th speaker and n_r-th noise condition is given by:

$$p(\boldsymbol{o}_t|\theta_t=m,\mathcal{M}_{\mathtt{c}},\mathcal{T}_{\mathtt{s}}^{(s_r)},\mathcal{T}_{\mathtt{n}}^{(n_r)})=\mathcal{N}(\boldsymbol{o}_t,\boldsymbol{\mu}^{(mr)},\boldsymbol{\Sigma}_{\mathtt{c}}^{(m)}) \tag{C.2}$$

with

$$\boldsymbol{\mu}^{(mr)}=\boldsymbol{\mu}_{\mathtt{c}}^{(m)}+\mathbf{M}_{\mathtt{s}}^{(m)}\boldsymbol{\lambda}_{\mathtt{s}}^{(s_r,q_m)}+\mathbf{M}_{\mathtt{n}}^{(m)}\boldsymbol{\lambda}_{\mathtt{n}}^{(n_r,p_m)}\,. \tag{C.3}$$

Here, $q_m \in \{1, \ldots, Q\}$ ($p_m \in \{1, \ldots, P\}$) maps the component index m to the speaker (noise) regression class index; $\boldsymbol{\lambda}_{\mathtt{s}}^{(i,q)}$ is the speaker cluster weight vector associated with q-th speaker base class for i-th speaker; $\boldsymbol{\lambda}_{\mathtt{n}}^{(j,p)}$ is the noise cluster weight vector associated with r-th noise base class for j-th noise condition. $\mathbf{M}_{\mathtt{s}}^{(m)} \in \mathcal{R}^{d \times d_{\mathtt{s}}}$, $\mathbf{M}_{\mathtt{n}}^{(m)} \in \mathcal{R}^{d \times d_{\mathtt{n}}}$ are the component m's speaker and noise cluster or subspace parameters, $d_{\mathtt{s}}$ and $d_{\mathtt{n}}$ are the number of speaker and noise clusters respectively. Among these parameters, $\{\boldsymbol{\mu}_{\mathtt{c}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)}, \mathbf{M}_{\mathtt{s}}^{(m)}, \mathbf{M}_{\mathtt{n}}^{(m)}\}$ form the canonical model parameters $\mathcal{M}_{\mathtt{c}}$, while $\{\boldsymbol{\lambda}_{\mathtt{s}}^{(i,q)} | q = 1 \ldots Q\}$ and $\{\boldsymbol{\lambda}_{\mathtt{n}}^{(j,p)} | p = 1 \ldots P\}$ are the parameters of i-th speaker transform $\mathcal{T}_{\mathtt{s}}^{(i)}$ and j-th noise transform $\mathcal{T}_{\mathtt{n}}^{(j)}$ respectively. The explicit independence constraint also requires the speaker and noise subspace are orthogonal to each other:

$$\mathbf{M}_{\mathtt{s}}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)-1} \mathbf{M}_{\mathtt{n}}^{(m)} = \mathbf{0} \tag{C.4}$$

The canonical model parameters and the transform parameters are adaptively estimated on the multi-conditional data via maximising the following auxiliary function:

$$\mathcal{Q} = \sum_{r,t,m} \gamma_t^{(mr)} \log \mathcal{N}(\boldsymbol{o}_t^{(r)}; \boldsymbol{\mu}^{(mr)}, \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)}) \; . \tag{C.5}$$

Five sets of parameters are iteratively maximised with respect to this auxiliary function. These are 1) speaker subspace $\{\mathbf{M}_{\mathtt{s}}^{(m)}\}$; 2) noise subspace $\{\mathbf{M}_{\mathtt{n}}^{(m)}\}$; 3) speaker transforms $\boldsymbol{\mathcal{T}}_{\mathtt{s}}$; 4) noise transforms $\boldsymbol{\mathcal{T}}_{\mathtt{n}}$; 5) canonical means and variances $\{\boldsymbol{\mu}_{\mathtt{c}}^{(m)}, \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)}\}$. The following sections will give detailed estimation formula for each of these sets of parameters.

C.1 Estimation of Cluster Parameters

Fixing all the other parameters, maximising the speaker cluster parameters $\mathbf{M}_{\mathtt{s}}^{(m)}$ amounts to the following constrained optimisation:

$$\begin{aligned} &\max_{\mathbf{M}} -\frac{1}{2}\mathrm{tr}\left(\mathbf{M}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)-1} \mathbf{M} \mathbf{G}_{\mathtt{s}}^{(m)}\right) + \mathrm{tr}\left(\boldsymbol{\Sigma}_{\mathtt{c}}^{(m)-1} \mathbf{M} \mathbf{K}_{\mathtt{s}}^{(m)}\right) \\ &\text{s. t.} \quad \mathbf{M}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathtt{c}}^{(m)-1} \mathbf{M}_{\mathtt{n}}^{(m)} = \mathbf{0} \end{aligned} \tag{C.6}$$

where

$$\begin{aligned} \mathbf{G}_{\mathtt{s}}^{(m)} &= \sum_{t,r} \gamma_t^{(mr)} \boldsymbol{\lambda}_{\mathtt{s}}^{(s_r,q_m)} \boldsymbol{\lambda}_{\mathtt{s}}^{(s_r,q_m)\mathsf{T}} \\ \mathbf{K}_{\mathtt{s}}^{(m)} &= \sum_{t,r} \gamma_t^{(mr)} \boldsymbol{\lambda}_{\mathtt{s}}^{(s_r,q_m)} (\boldsymbol{o}_t^{(r)} - \boldsymbol{\mu}_c^{(m)} - \mathbf{M}_{\mathtt{n}}^{(m)} \boldsymbol{\lambda}_{\mathtt{n}}^{(n_r,p_m)})^{\mathsf{T}} \end{aligned}$$

and $\gamma_t^{(mr)}$ is the posterior probability of Gaussian component m at time t for the r-th utterance, calculated using the current parameters. Since these optimisation problem can be individually done for each m, the superscript m will be dropped in this section for notation convenience.

Using the method of Lagrange multipliers [29], a Lagrange dual function for this optimisation problem can be defined as follows:

$$\mathcal{D}(\mathbf{M}, \mathbf{\Gamma}) = -\frac{1}{2}\mathrm{tr}\left(\mathbf{M}^{\mathsf{T}}\mathbf{\Sigma}_{\mathsf{c}}^{-1}\mathbf{M}\mathbf{G}_{\mathsf{s}}\right) + \mathrm{tr}\left(\mathbf{\Sigma}_{\mathsf{c}}^{-1}\mathbf{M}\mathbf{K}_{\mathsf{s}}\right) + \mathrm{tr}\left(\mathbf{\Gamma}^{\mathsf{T}}\mathbf{M}^{\mathsf{T}}\mathbf{\Sigma}_{\mathsf{c}}^{-1}\mathbf{M}_{\mathsf{n}}\right) \tag{C.7}$$

where $\mathbf{\Gamma} \in \mathcal{R}^{d_s \times d_n}$ is the dual variable of $\mathbf{M}$. Since the concerned constrained optimisation problem is convex with only the equality constraints, the *strong duality* holds, and the optimal $\mathbf{M}$ can be obtained by maximising the dual function. Since the dual function is a convex function of $\mathbf{M}$, there is a unique optimal $\mathbf{M}^*$, which maximises $\mathcal{D}(\mathbf{M}, \mathbf{\Gamma})$. Differentiating $\mathcal{D}$ with respect to $\mathbf{M}$ and equating to zero gives the following equation:

$$\mathbf{\Sigma}^{-1}(\mathbf{K}^{\mathsf{T}} + \mathbf{M}_{\mathsf{n}}\mathbf{\Gamma}^{\mathsf{T}}) = \frac{1}{2}\mathbf{\Sigma}^{-1}\mathbf{M}^*(\mathbf{G} + \mathbf{G}^{\mathsf{T}}) = \mathbf{\Sigma}^{-1}\mathbf{M}^*\mathbf{G}\ . \tag{C.8}$$

The second equation follows the fact that $\mathbf{G}$ is a symmetric matrix. The optimiser is thus given:

$$\mathbf{M}^* = (\mathbf{K}^{\mathsf{T}} + \mathbf{M}_{\mathsf{n}}\mathbf{\Gamma}^{\mathsf{T}})\mathbf{G}^{-1}\ . \tag{C.9}$$

On the other hand $\mathbf{M}^*$ needs to satisfy the constraint $\mathbf{M}^{*\mathsf{T}}\mathbf{\Sigma}^{-1}\mathbf{M}_{\mathsf{n}} = \mathbf{0}$, therefore

$$\mathbf{\Gamma} = -\mathbf{G}^{-1}\mathbf{K}\mathbf{\Sigma}^{-1}\mathbf{M}_{\mathsf{n}}\left(\mathbf{M}_{\mathsf{n}}\mathbf{\Sigma}^{-1}\mathbf{M}_{\mathsf{n}}\right)^{-1} \tag{C.10}$$

and

$$\mathbf{M}^* = \left[\mathbf{I} - \mathbf{M}_{\mathsf{n}}\left(\mathbf{M}_{\mathsf{n}}^{\mathsf{T}}\mathbf{\Sigma}_{\mathsf{c}}^{-1}\mathbf{M}_{\mathsf{n}}\right)^{-1}\mathbf{M}_{\mathsf{n}}^{\mathsf{T}}\mathbf{\Sigma}_{\mathsf{c}}^{-1}\right]\mathbf{K}_{\mathsf{s}}^{\mathsf{T}}\mathbf{G}_{\mathsf{s}}^{-1} \tag{C.11}$$

Similar formula can be used to estimate the noise cluster parameters.

C.2 Estimation of Cluster Weights

Fixing all the other parameters, maximising the speaker cluster weights $\boldsymbol{\lambda}_{\mathsf{s}}^{(i,q)}$ amounts to maximising the following function:

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\lambda}_{\mathsf{s}}^{(i,q)}) &= -\frac{1}{2}\sum_{r:s_r=i}\sum_{m\in q}\sum_t \gamma_t^{(mr)}(\boldsymbol{o}_t^{(r)} - \boldsymbol{\mu}^{(mr)})^{\mathsf{T}}\mathbf{\Sigma}_{\mathsf{c}}^{(m)-1}(\boldsymbol{o}_t^{(r)} - \boldsymbol{\mu}^{(mr)}) \\ &= -\frac{1}{2}\mathrm{tr}\left(\mathbf{G}_{\mathbf{w},\mathbf{s}}^{(i,q)}\boldsymbol{\lambda}_{\mathsf{s}}^{(i,q)}\boldsymbol{\lambda}_{\mathsf{s}}^{(i,q)\mathsf{T}}\right) + \mathbf{k}_{\mathbf{w},\mathbf{s}}^{(i,q)\mathsf{T}}\boldsymbol{\lambda}_{\mathsf{s}}^{(i,q)}\end{aligned} \tag{C.12}$$

where the sufficient statistics, $\mathbf{k}_{\mathbf{w},\mathbf{s}}^{(i,q)}$ and $\mathbf{G}_{\mathbf{w},\mathbf{s}}^{(i,q)}$, are given by

$$\mathbf{G}_{\mathbf{w},\mathbf{s}}^{(i,q)} = \sum_{r:s_r=i} \sum_{m \in q} \sum_{t} \gamma_t^{(mr)} \mathbf{M}_{\mathbf{s}}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{c}}^{(m)-1} \mathbf{M}_{\mathbf{s}}^{(m)} \tag{C.13}$$

$$\mathbf{k}_{\mathbf{w},\mathbf{s}}^{(i,q)} = \sum_{r:s_r=i} \sum_{m \in q} \sum_{t} \gamma_t^{(mr)} \mathbf{M}_{\mathbf{s}}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{c}}^{(m)-1} (\boldsymbol{o}_t^{(r)} - \boldsymbol{\mu}_{\mathbf{c}}^{(m)} - \mathbf{M}_{\mathbf{n}}^{(m)} \boldsymbol{\lambda}_{\mathbf{n}}^{(n_r,p_m)}) \tag{C.14}$$

Therefore, the optimal $\boldsymbol{\lambda}_{\mathbf{s}}^{(i,q)}$ is given by:

$$\boldsymbol{\lambda}_{\mathbf{s}}^{(i,q)} = \mathbf{G}_{\mathbf{w},\mathbf{s}}^{(i,q)-1} \mathbf{k}_{\mathbf{w},\mathbf{s}}^{(i,q)} \tag{C.15}$$

Similar formula can be used to estimate the noise cluster weights.

C.3 Estimation of Canonical Mean and Variance

Fixing all the other parameters fixed, maximising the canonical mean and variance, $\mu_{\mathbf{c}}^{(m)}$ and $\boldsymbol{\Sigma}_{\mathbf{c}}^{(m)}$, amounts to the following optimisation problem:

$$\mathcal{Q}(\boldsymbol{\mu}_{\mathbf{c}}^{(m)}, \boldsymbol{\Sigma}_{\mathbf{c}}^{(m)}) = -\frac{1}{2}\gamma^{(m)} \left(\log |\boldsymbol{\Sigma}_{\mathbf{c}}^{(m)}| + \boldsymbol{\mu}_{\mathbf{c}}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{c}}^{(m)-1} \boldsymbol{\mu}_{\mathbf{c}}^{(m)} \right) - \mathrm{tr}(\boldsymbol{\Gamma}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{c}}^{(m)-1}) + \boldsymbol{\mu}_{\mathbf{c}}^{(m)\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{c}}^{(m)-1} \boldsymbol{\eta}^{(m)} \tag{C.16}$$

where the sufficient statistics, $\gamma^{(m)}$ and $\boldsymbol{\eta}^{(m)}$, are given by

$$\gamma^{(m)} = \sum_{t,r} \gamma_t^{(mr)} \tag{C.17}$$

$$\boldsymbol{\eta}^{(m)} = \sum_{t,r} \gamma_t^{(mr)} (\boldsymbol{o}_t^{(r)} - \mathbf{M}_{\mathbf{s}}^{(m)} \boldsymbol{\lambda}_{\mathbf{s}}^{(s_r,q_m)} - \mathbf{M}_{\mathbf{n}}^{(m)} \boldsymbol{\lambda}_{\mathbf{n}}^{(n_r,p_m)}) \tag{C.18}$$

$$\boldsymbol{\Gamma}^{(m)} = \sum_{t,r} \gamma_t^{(mr)} (\boldsymbol{o}_t^{(r)} - \mathbf{M}_{\mathbf{s}}^{(m)} \boldsymbol{\lambda}_{\mathbf{s}}^{(s_r,q_m)} - \mathbf{M}_{\mathbf{n}}^{(m)} \boldsymbol{\lambda}_{\mathbf{n}}^{(n_r,p_m)}) (\boldsymbol{o}_t^{(r)} - \mathbf{M}_{\mathbf{s}}^{(m)} \boldsymbol{\lambda}_{\mathbf{s}}^{(s_r,q_m)} - \mathbf{M}_{\mathbf{n}}^{(m)} \boldsymbol{\lambda}_{\mathbf{n}}^{(n_r,p_m)})^{\mathsf{T}} \tag{C.19}$$

Therefore, the optimal solution is given by:

$$\boldsymbol{\mu}_{\mathbf{c}}^{(m)} = \frac{\boldsymbol{\eta}^{(m)}}{\gamma^{(m)}} \tag{C.20}$$

$$\boldsymbol{\Sigma}_{\mathbf{c}}^{(m)} = \frac{\boldsymbol{\Gamma}^{(m)}}{\gamma^{(m)}} - \boldsymbol{\mu}_{\mathbf{c}}^{(m)} \boldsymbol{\mu}_{\mathbf{c}}^{(m)\mathsf{T}} \tag{C.21}$$

References

[1] The REVERB challenge. `http://reverb2014.dereverberation.com/`. Accessed on September 1st, 2013. 7.2

[2] O. Abdel and H. Jiang. Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code. In *Proceedings of ICASSP*, 2013. 2.3.1.3

[3] K. Abed-Meraim, E. Moulines, and P. Loubaton. Prediction error method for second-order blind identification. *IEEE Transactions on Signal Processing*, 45(3):694–705, 1997. 3.3.3.1

[4] A. Acero. *Acoustical and environmental robustness in automatic speech recognition*. Kluwer Academic Pub, 1993. 3, 3.3.1, 3.3.1, 3.3.1.1, 4.1.1

[5] A. Acero, L. Deng, T. Kristjansson, and J. Zhang. HMM adaptation using vector Taylor series for noisy speech recognition. In *Proceedings of ICSLP*, 2000. 3.3.2.2, 3.3.2.3, 3.3.2.3, 4.2.2, 6.1.1, 6.1.2

[6] A. Agarwal and Y. M. Cheng. Two-stage Mel-warped Wiener filter for robust speech recognition. In *Proceedings of ASRU*, pages 67–70, 1999. 3.3.2.1

[7] S. M. Ahadi and P. C. Woodland. Combined Bayesian and predictive techniques for rapid speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 11(3):187–206, 1997. 3.2.1

[8] C. Alberti, M. Bacchiani, A. Bezman, C. Chelba, A. Drofa, H. Liao, P. Moreno, T. Power, A. Sahuguet, M. Shugrina, et al. An audio indexing system for election video material. In *Proceedings of ICASSP*, pages 4873–4876, 2009. 1

[9] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul. A compact model for speaker adaptive training. In *Proceedings of ICSLP*, pages 1137–1140, 1996. 3, 3.4, 3.4.1, 3.4.1.1, 3.4.1.1

[10] J. A. Arrowood. *Using observation uncertainty for robust speech recognition.* PhD thesis, Georgia Institute of Technology, 2003. 3.3.2.2, 3.3.2.2

[11] B. S. Atal. Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. *The Journal of the Acoustical Society of America*, 55, 1974. 2.2.1, 3.4

[12] Y. Avargel and I. Cohen. System identification in the short-time fourier transform domain with crossband filtering. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1305–1319, 2007. 3.3.3.1

[13] M. Bacchiani, F. Beaufays, J. Schalkwyk, M. Schuster, and B. Strope. Deploying goog-411: early lessons in data, measurement, and testing. In *Proceedings of ICASSP*, pages 5260–5263, 2008. 1

[14] J. Baker. The DRAGON system–An overview. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23(1):24–29, 1975. 1

[15] J. Baker, L. Deng, J. Glass, S. Khudanpur, C.-H. Lee, N. Morgan, and D. O'Shaughnessy. Developments and directions in speech recognition and understanding. *IEEE Signal Processing Magazine*, 26(3):75–80, 2009. 1

[16] L. E. Baum and J. A. Eagon. An inequality with application to statistical estimation for probabilistic functions of markov processes and to a model for ecology. *Bulletin of the American Mathematical Society*, 73:360–363, 1967. 2.3.3.1

[17] M. J. Beal. *Variational algorithms for approximate Bayesian inference.* PhD thesis, University of London, 2003. 3.2.1

[18] J. Benesty, J. Chen, and Y. Huang. *Microphone array signal processing.* Springer, 2008. 3.3.1

[19] Yoshua Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003. 2.4.1

[20] M. C. Benitez, J. C. Segura, A. Torre, J. Ramirez, and A. Rubio. Including uncertainty of speech observations in robust speech recognition. In *Proceedings of ICSLP*, 2004. 3.3.2.2

[21] J. Bernardo and A. F. Smith. *Bayesian theory.* Wiley, 2009. 3.2.1

[22] J. A. Bilmes. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Technical report, International Computer Science Institute, 1998. 2.3.3.1

[23] C. M. Bishop. *Pattern recognition and machine learning*, volume 1. Springer New York, 2006. 2.3.1.3, 3.3.2.2

[24] S. F. Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 27(2):113–120, 1979. 3.3.2.1

[25] S. E. Bou-Ghazale and J. H. L. Hansen. Duration and spectral based stress token generation for HMM speech recognition under stress. In *Proceedings of ICASSP*, 1994. 3.3.1

[26] S. E. Bou-Ghazale and J. H. L. Hansen. A comparative study of traditional and newly proposed features for recognition of speech under stress. *IEEE Transactions on Speech and Audio Processing*, 8(4):429–442, 2000. 3.3.1

[27] H. Bourlard and N. Morgan. Continuous speech recognition by connectionist statistical methods. *IEEE Transactions on Neural Networks*, 4(6):893–909, 1993. 2.3.1.3

[28] H. A. Bourlard and N. Morgan. *Connectionist speech recognition: A hybrid approach*. Springer, 1994. 2.3.1.3

[29] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004. 6.3.1, C.1

[30] P. F. Brown. *The acoustic-modeling problem in automatic speech recognition*. PhD thesis, Carnegie Mellon University, 1987. 2.3.3, 2.3.3.2

[31] L. Buera, A. Miguel, O. Saz, A. Ortega, and E. Lleida. Unsupervised data-driven feature vector normalization with acoustic model adaptation for robust speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(2):296 –309, 2010. 6.1

[32] N. A. Campbell. Canonical variate analysis a general model formulation. *Australian Journal of Statistics*, 26(1):86–96, 1984. 2.2.1

[33] S. S. Chen, E. Eide, M. J. F. Gales, R. A. Gopinath, D. Kanvesky, and P. Olsen. Automatic transcription of broadcast news. *Speech Communication*, 37(1):69–87, 2002. 1

[34] K. K. Chin, H. Xu, M. J. F. Gales, C. Breslin, and K. Knill. Rapid joint speaker and noise compensation for robust speech recognition. In *Proceedings of ICASSP*, pages 5500–5503, 2011.

[35] T. Chou. Frequency-independent beamformer with low response error. In *Proceedings of ICASSP*, pages 2995–2998, 1995. 3.3.3.1

[36] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989. 2.3.1.3

[37] G. E. Dahl, D. Yu, L. Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):30–42, 2012. 2.3.1.3

[38] R. C. van Dalen and M. J. F. Gales. Extended VTS for noise-robust speech recognition. In *Proceedings of ICASSP*, 2009. 3.3.1.1, 3.3.2.3, 4.1.2

[39] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980. 2.2, A.1

[40] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2011. 5.2.1

[41] M. Delcroix, T. Nakatani, and S. Watanabe. Static and dynamic variance compensation for recognition of reverberant speech with dereverberation preprocessing. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(2):324–334, 2009. 3.3.3.3, 3.3.3.3

[42] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977. 2.3.3.1

[43] K. Demuynck, X. Zhang, D. Van Compernolle, and H. Van hamme. Feature verus model based noise robustness. In *Proceedings of Interspeech*, pages 721–724, 2010. 8.1.1

[44] L. Deng. Roles of high-fidelity acoustic modeling in robust speech recognition. In *Proceedings of ASRU*, 2007. 4.1.3.2, A.2

[45] L. Deng, A. Acero, M. Plumpe, and X. Huang. Large vocabulary speech recognition under adverse acoustic environments. In *Proceedings of ICSLP*, 2000. 3.3.2.2, 3.3.2.2, 3.3.2.3, 3.4, 3.4.2

[46] L. Deng, J. Droppo, and A. Acero. Exploiting variances in robust feature extraction based on a parametric model of speech distortion. In *Proceedings of Interspeech*, 2002. 3.3.2.2

[47] L. Deng, J. Droppo, and A. Acero. Enhancement of log Mel power spectra of speech using a phase-sensitive model of the acoustic environment and sequential estimation of the corrupting noise. *IEEE Transactions on Speech and Audio Processing*, 12(2): 133–143, 2004. 3.3.1.1, 4.1.3.2, A.2

[48] L. Deng, J. Droppo, and A. Acero. Dynamic compensation of HMM variances using the feature enhancement uncertainty computed from a parametric model of speech distortion. *IEEE Transactions on Speech and Audio Processing*, 13(3):412–421, 2005. 3.3.2.2, 3.3.2.2, 3.3.2.2

[49] V. D. Diakoloukas and V. V . Digalakis. Maximum-likelihood stochastic-transformation adaptation of hidden Markov models. *IEEE Transactions on Speech and Audio Processing*, 7(2):177–187, 1999. 3.2.2

[50] V. Digalakis, D. Rtischev, and L. Neumeyer. Speaker adaptation using constrained estimation of Gaussian mixtures. *IEEE Transactions on Speech and Audio Processing*, pages 357–366, 1995. 3.2.2.3

[51] ETSI Standard Document. Speech processing, transmission and quality aspects (stq); distributed speech recognition; front-end feature extraction algorithm; compression algorithms. *ETSI ES 201 108 v1.1.2*, 2000. A.1

[52] P. Douglas and J. M. Baker. The design for the Wall Street Journal-based CSR corpus. In *Proceedings of the Workshop on Speech and Natural Language*, pages 357–362, 1992. 1, 8.2

[53] J. Droppo, A. Acero, and L. Deng. Uncertainty decoding with SPLICE for noise robust speech recognition. In *Proceedings of ICASSP*, 2002. 3.3.2.2, 3.3.2.2

[54] J. Du and Q. Huo. A feature compensation approach using piecewise linear approximation of an explicit distortion model for noisy speech recognition. In *Proceedings of ICASSP*, pages 4721–4724, 2008. 3.3.2.3

[55] Y. Ephraim and D. Malah. Speech enhancement using a minimum mean-square error log-spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(2):443–445, 1985. 3.3.2.2

[56] Y. Ephraim, D. Malah, and B-H Juang. On the application of hidden Markov models for enhancing noisy speech. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(12):1846–1856, 1989. 3.3.2.1

[57] J. S. Erkelens and R. Heusdens. Correlation-based and model-based blind single-channel late-reverberation suppression in noisy time-varying acoustical environments. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(7):1746–1765, 2010. 3.3.3.2

[58] P. Fousek. *Extraction of features for automatic recognition of speech based on spectral dynamics*. PhD thesis, Czech Tech Univ, Prague, 2007. 2.3.1.3

[59] S. Furui. Speaker independent isolated word recognition using dynamic features of speech spectrum. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34: 52–59, 1986. 2.2.1

[60] S. Furui. Unsupervised speaker adaptation based on hierarchical spectral clustering. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(12):1923–1930, 1989. 3.2.3

[61] M. F. J. Gales and S. J. Young. Robust speech recognition in additive and convolutional noise using parallel model combination. *Computer Speech and Language*, 9(4):289–307, 1995. 3.3.2.3

[62] M. J. F. Gales. *Model-based techniques for noise robust speech recognition*. PhD thesis, Cambridge University, 1995. 3.3.1, 3.3.1.1, 3.3.2.3, 3.3.2.3, 3.3.2.3, 3.3.2.3, 3.3.2.3, 4.1.3.2, A.2, A.2

[63] M. J. F. Gales. The generation and use of regression class trees for MLLR adaptation. *University of Cambridge, Tech. Rep. CUED/F-INFENG/TR263*, 1996. 3.2.4

[64] M. J. F. Gales. Transformation smoothing for speaker and environmental adaptation. In *Proceedings of European Conference on Speech Communication Technology*, 1997. 3.2.3

[65] M. J. F. Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech and Language*, 12:75–98, 1998. 6.1.3

[66] M. J. F. Gales. Predictive model-based compensation schemes for robust speech recognition. *Speech Communication*, 25(1-3):49–74, 1998. 3.3.2.3, 6.1

[67] M. J. .F. Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech and Language*, 12(2):75–98, 1998. 3.2, 3.2.2.1, 3.2.2.1, 3.2.2.2, 3.2.2.2, 3.2.2.3, 3.2.2.3, 3.4.1, 3.4.1.1, 7.1.3.5

[68] M. J. F. Gales. Cluster adaptive training of hidden Markov models. *IEEE Transactions on Speech and Audio Processing*, 8(4):417–428, 2000. 3.4.1, 3.4.1.2

[69] M. J. F. Gales. Acoustic factorisation. In *Proceedings of ASRU*, 2001. 1, 4.3, 5, 5.1

[70] M. J. F. Gales. Cluster adaptive training of hidden Markov models. *IEEE Transactions on Speech and Audio Processing*, 8(4):417–428, 2002. 3.2, 3.2.3, 3.4, 6.3

[71] M. J. F. Gales. Model-based approaches to handling uncertainty. In *Robust Speech Recognition of Uncertain or Missing Data: Theory and Applications*, pages 101–125. Springer, 2011. (document), 3, 3.1, 3.3.1, 3.3.2.2, 3.3.2.2, 3.3.3.3, 3.3.3.3

[72] M. J. F. Gales and F. Flego. Discriminative classifiers with adaptive kernels for noise robust speech recognition. *Computer Speech and Language*, 24(4):648–662, 2010. 3.3.1.1, 7.1.3.4

[73] M. J. F. Gales and Y.-Q. Wang. Model-based approaches to handling additive noise in reverberant environments. In *Joint Workshop on Hands-free Speech Communication and Microphone Arrays*, 2011. 4.2.3, 7.1.3.5, 7.2

[74] M. J. F. Gales and P. C . Woodland. Mean and variance adaptation within the MLLR framework. *Computer Speech and Language*, 10(4):249–264, 1996. 3.2, 3.2.2.1, 3.2.2.2, 3.2.2.2

[75] M. J. F. Gales and S. Young. An improved approach to the hidden markov model decomposition of speech and noise. In *Proceedings of ICASSP*, volume 1, pages 233–236, 1992. 3.3.2.3

[76] M. J. F. Gales and S. Young. The application of hidden Markov models in speech recognition. *Foundations and Trends in Signal Processing*, 1(3):195–304, 2008. 1

[77] M. J. F. Gales, D. Y. Kim, P. .C. Woodland, H. Y. Chan, D. Mrva, R. Sinha, and S. E. Tranter. Progress in the CU-HTK broadcast news transcription system. *IEEE Transactions on Audio, Speech, and Language Processing*, 2007. 1

[78] S. Gannot and M. Moonen. Subspace methods for multimicrophone speech dereverberation. *EURASIP Journal on Applied Signal Processing*, 2003:1074–1090, 2003. 3.3.3.1

[79] 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*, 2(2):291–298, 1994. 3.2, 3.2.1

[80] J. F. Gemmeke, U. Remes, and K. J. Palomäki. Observation uncertainty measures for sparse imputation. In *Proceedings of Interspeech*, pages 2262–2265, 2010. 3.3.2.2

[81] B. W. Gillespie and A. L. Atlas. Strategies for improving audible quality and speech recognition accuracy of reverberant speech. In *Proceedings of ICASSP*, 2013. 3.3.3.1

[82] L. Gillick and S. J. Cox. Some statistical issues in the comparison of speech recognition algorithms. In *Proceedings of ICASSP*, pages 532–535, 1989. 2.4.3

[83] Y. Gong. Speech recognition in noisy environments: A survey. *Speech Communication*, 16(3):261–291, 1995. 3, 3.3.1

[84] I. J. Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:237–264, 1953. 2.4.1

[85] P. S. Gopalakrishnan, D. Kanevsky, A. Nadas, and D. Nahamoo. A generalization of the Baum algorithm to rational objective functions. In *Proceedings of ICASSP*, 1989. 2.3.3.2

[86] R. A. Gopinath et al. Robust speech recognition in noise – performance of the IBM continuous speech recogniser on the ARPA noise spoke task. In *Proc. APRA Workshop on Spoken Language System Technology*, pages 127–130, 1995. 3.3.1.1, 3.3.2.3, 4.1.2, 4.2.2.1, 6.1.1

[87] M. Gurelli and C.L. Nikias. EVAM: An eigenvector-based algorithm for multichannel blind deconvolution of input colored signals. *IEEE Transactions on Signal Processing*, 43(1):134–149, 1995. 3.3.3.1

[88] T. Hain, P. C. Woodland, G. Evermann, M. J. F. Gales, X. Liu, G. L. Moore, D. Povey, and L. Wang. Automatic transcription of conversational telephone speech. *IEEE Transactions on Speech and Audio Processing*, 13(6):1173–1185, 2005. 1, 2.4.2

[89] J. H. L. Hansen. Analysis and compensation of speech under stress and noise for environmental robustness in speech recognition. *Speech Communication*, 20:151–173, 1996. 3.3.1

[90] S. Haykin. *Array signal processing*. Prentice-Hall, Inc., 1985. 3.3.3.1

[91] T. J. Hazen. *The use of speaker correlation information for automatic speech recognition*. PhD thesis, Massachusetts Institute of Technology, 1998. 3.2.3

[92] T. J. Hazen and J. R. Glass. A comparison of novel techniques for instantaneous speaker adaptation. In *Proceedings of European Conference on Speech Communication Technology*, 1997. 3.2.3

[93] G. Heigold, T. Deselaers, R. Schlüter, and H. Ney. Modified MMI/MPE: A direct evaluation of the margin in speech recognition. In *Proceedings of the International Conference on Machine Learning*, pages 384–391, 2008. 2.3.3.2

[94] G. Heigold, H. Ney, R. Schluter, and S. Wiesler. Discriminative training for automatic speech recognition: modeling, criteria, optimization, implementation, and performance. *IEEE Signal Processing Magazine*, 29(6):58–69, 2012. 2.3.3.2

[95] H. Hermansky. Perceptual linear predictive (PLP) analysis of speech. *The Journal of the Acoustical Society of America*, 87(4):1738–1752, 1990. 2.2

[96] H. Hermansky and N. Morgan. RASTA processing of speech. *IEEE Transactions on Speech and Audio Processing*, 2(4):578–589, 1994. 3.3.2.1

[97] H. Hermansky, N. Morgan, A. Bayya, and P. Kohn. RASTA-PLP speech analysis technique. In *Proceedings of ICASSP*, pages 121–124, 1992. 3.3.2.1

[98] H. Hermansky, D. P. W. Ellis, and S. Sharma. Tandem connectionist feature extraction for conventional HMM systems. In *Proceedings of ICASSP*, pages 1635–1638, 2000. 2.2.1, 2.3.1.3, 3.3.2.1

[99] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, and T. N. Sainath. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. 2.3.1.3

[100] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006. 2.3.1.3

[101] H. G. Hirsch and C. Ehrlicher. Noise estimation techniques for robust speech recognition. In *Proceedings of ICASSP*, pages 153–156, 2005. 3.3.2.1

[102] H.-G. Hirsch and H. Finster. The simulation of realistic acoustic input scenarios for speech recognition systems. In *Proceedings of Interspeech*, pages 2697–2700, 2005. 7.1

[103] H. G. Hirsch and H. Finster. A new approach for the adaptation of HMMs to reverberation and background noise. *Speech Communication*, 50(3):244–263, 2008. 3.3.3.3, 3.3.3.3, 3.3.3.3, 4.2.1, 4.2.3, 4.2.3, 4.2.3, 7.1.3.1, 9.1

[104] J. R. Hopgood and P. J. W. Rayner. Blind single channel deconvolution using nonstationary signal processing. *IEEE Transactions on Speech and Audio Processing*, 11(5): 476–488, 2003. 3.3.3.1

[105] R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge University Press, 2012. 3.3.1.1

[106] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991. 2.3.1.3

[107] Y. Hu and Q. Huo. An HMM compensation approach using unscented transformation for noisy speech recognition. *Chinese Spoken Language Processing*, pages 346–357, 2006. 3.3.2.3

[108] Y. Hu and Q. Huo. Irrelevant variability normalization based HMM training using VTS approximation of an explicit model of environmental distortions. In *Proceedings of Interspeech*, pages 1042–1045, 2007. 3.3.2.3, 3.4, 3.4.2

[109] X. Huang, A. Acero, and H. Hon. *Spoken language processing: a guide to theory, algorithm, and system development*. Prentice Hall, 2001. 2.3.1.2, 3.2

[110] X. D. Huang and K. F. Lee. On speaker-independent, speaker-dependent, and speaker-adaptive speech recognition. *IEEE Transactions on Speech and Audio Processing*, 1(2): 150–157, 1993. 3.2

[111] Q. Huo and C.-H. Lee. On-line adaptive learning of the continuous density hidden Markov model based on approximate recursive Bayes estimate. *IEEE Transactions on Speech and Audio Processing*, 5(2):161–172, 1997. 3.2.1

[112] M.-Y. Hwang and X.-D. Huang. Shared-distribution hidden Markov models for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 1(4):414–420, 1993. 2.3.1.2, 2.3.1.2

[113] V. Ion and R. Haeb-Umbach. A novel uncertainty decoding rule with applications to transmission error robust speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(5):1047–1060, 2008. 3.3.2.2

[114] F. Jelinek. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64(4):532–556, 1976. 1, 2.3.1

[115] H. Jiang. Discriminative training of HMMs for automatic speech recognition: A survey. *Computer Speech and Language*, 24(4):589–608, 2010. 2.3.3.2

[116] B. H. Juang, W. Hou, and C.-H. Lee. Minimum classification error rate methods for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 5(3):257–265, 1997. 2.3.3.2

[117] S. J. Julier and J. K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004. 3.3.2.3

[118] J. C. Junqua. The Lombard reflex and its role on human listeners and automatic speech recognizers. *The Journal of the Acoustical Society of America*, 93, 1993. 3.3.1

[119] J. C. Junqua and Y. Anglade. Acoustic and perceptual studies of Lombard speech: Application to isolated-words automatic speech recognition. In *Proceedings of ICASSP*, pages 841–844, 1990. 3.3.1

[120] J.-C. Junqua and Y. Anglade. Acoustic and perceptual studies of Lombard speech: Application to isolated-words automatic speech recognition. In *Proceedings of ICASSP*, 1990. 5.1

[121] K. Kalgaonkar, M. L. Seltzer, and A. Acero. Noise robust model adaptation using linear spline interpolation. In *Proceedings of ASRU*, pages 199–204, 2009. 3.3.2.3

[122] O. Kalinli, M. L. Seltzer, and A. Acero. Noise adaptive training using a vector Taylor series approach for noise robust automatic speech recognition. In *Proceedings of ICASSP*, 2009. 3.3.2.3, 3.4, 3.4.2, 3.4.2, 4.2

[123] H. Kameoka, T. Nakatani, and T. Yoshioka. Robust speech dereverberation based on non-negativity and sparse nature of speech spectrograms. In *Proceedings of ICASSP*, pages 45–48, 2009. 3.3.3.2

[124] S. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401, 1987. 2.4.1

[125] P. Kenny, M. Lennig, and P. Mermelstein. Speaker adaptation in a large-vocabulary Gaussian HMM recognizer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):917–920, 1990. 3.2.2

[126] P. Kenny, G. Boulianne, and P. Dumouchel. Eigenvoice modeling with sparse training data. *IEEE Transactions on Speech and Audio Processing*, 13(3):345–354, 2005. 5.2.1

[127] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel. Joint factor analysis versus eigenchannels in speaker recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1435–1447, 2007. 5.2.1, 5.2.1, 6.3

[128] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel. Speaker and session variability in GMM-based speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(4):1448–1460, 2007. 5.2.1

[129] D. Y. Kim, C. Kwan Un, and N. S. Kim. Speech recognition in noisy environments using first-order vector Taylor series. *Speech Communication*, 24(1):39–49, 1998. 2, 3.3.2.2, 3.3.2.3, 3.3.2.3

[130] K. Kinoshita, M. Delcroix, T. Nakatani, and M. Miyoshi. Suppression of late reverberation effect on speech signal using long-term multiple-step linear prediction. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(4):534–545, 2009. 3.3.3.1

[131] R. Kneser and H. Ney. Improved backing-off for N-gram language modeling. In *Proceedings of ICASSP*, pages 181–184, 1995. 2.4.1

[132] B. O. Koopman. On distributions admitting a sufficient statistic. *Transactions of the American Mathematical Society*, 39(3):399–409, 1936. 3.2.1

[133] T. Kosaka and S. Sagayama. Tree-structured speaker clustering for fast speaker adaptation. In *Proceedings of ICASSP*, 1994. 3.2.3

[134] A. Krueger and R. Haeb-Umbach. Model-based feature enhancement for reverberant speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 18 (7):1692–1707, 2010. 3.3.3.2, 4.1.2

[135] A. Krueger and R. Haeb-Umbach. A model-based approach to joint compensation of noise and reverberation for speech recognition. In *Robust Speech Recognition of Uncertain or Missing Data*, pages 257–290. Springer, 2011. 4.1.2

[136] R. Kuhn, P. Nguyen, J. C. Junqua, L. Goldwasser, N. Niedzielski, S. Fincke, K. Field, and M. Contolini. Eigenvoices for speaker adaptation. In *Proceedings of ICSLP*, 1998. 3.2.3, 6.3, 6.3.1

[137] R. Kuhn, J. C. Junqua, P. Nguyen, and N. Niedzielski. Rapid speaker adaptation in eigenvoice space. *IEEE Transactions on Speech and Audio Processing*, 8(6):695–707, 2000. 3.4.1.2

[138] K. Kumar, B. Raj, R. Singh, and R. M. Stern. An iterative least-squares technique for dereverberation. In *Proceedings of ICASSP*, pages 5488–5491, 2011. 3.3.3.2, 3.3.3.2

[139] N. Kumar. *Investigation of silicon auditory models and generalization of linear discriminant analysis for improved speech recognition*. PhD thesis, Johns Hopkins University, 1997. 2.2.1

[140] K. Kumatani, J. McDonough, B. Rauch, D. Klakow, P. N. Garner, and W. Li. Beamforming with a maximum negentropy criterion. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(5):994–1008, 2009. 7.2

[141] J. Latorre, V. Wan, M. J. F. Gales, L.-Z. Chen, K.-K. Chin, K. Knill, and M. Akamine. Speech factorization for HMM-TTS based on cluster adaptive training. In *Proceedings of Interspeech*, 2012. 5.2.1, 6.3, 6.3.1

[142] K. Lebart, J.-M. Boucher, and P. N. Denbigh. A new method based on spectral subtraction for speech dereverberation. *Acta Acustica united with Acustica*, 87(3):359–366, 2001. 3.3.3.2

[143] C.-H. Lee and Q. Huo. On adaptive decision rules and decision parameter adaptation for automatic speech recognition. *Proceedings of the IEEE*, 88(8):1241–1269, 2000. 3

[144] C.-H. Lee, F. K. Soong, and K. K. Paliwal. *Automatic speech and speaker recognition: advanced topics*. Springer, 1996. 1

[145] K. F. Lee. On large-vocabulary speaker-independent continuous speech recognition. *Speech Communication*, 7(4):375–379, 1988. 2.3.1.2

[146] L. Lee and R. C. Rose. Speaker normalization using efficient frequency warping procedures. In *Proceedings of ICASSP*, pages 353–356, 1996. 2.2.1, 3.4

[147] C. J. Leggetter. *Improved acoustic modelling for HMMs using linear transformations*. PhD thesis, University of Cambridge, 1995. 3.2.4

[148] C. J. Leggetter and P. C. Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech and Language*, 9(2):171–185, 1995. 3.2, 3.2.2.1, 3.2.2.1, 3.2.4, 6.1.1

[149] V. Leutnant and R. Haeb-Umbach. An analytic derivation of a phase-sensitive observation model for noise robust speech recognition. In *Proceedings of Interspeech*, pages 2395–2398, 2009. 3.3.1.1

[150] V. Leutnant, A. Krueger, and R. Haeb-Umbach. A statistical observation model for noisy reverberant speech features and its application to robust ASR. In *International Conference on Signal Processing, Communication and Computing (ICSPCC)*, pages 142–147, 2012. 4.1.2

[151] V. Leutnant, A. Krueger, and R. Haeb-Umbach. Bayesian feature enhancement for reverberation and noise robust speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(8):1640–1652, 2013. 3.3.3.2, 7.2, 1, 7.2.2

[152] J. Li and P. Stoica. *Robust adaptive beamforming*. Wiley, 2006. 3.3.3.1

[153] J. Li, D. Yu, Y. Gong, and A. Acero. High-performance HMM adaptation with joint compensation of additive and convolutive distortions via vector Taylor series. In *Proceedings of ASRU*, 2007. 3.3.2.3, 6.1.1, 6.1.2, 6.1.3, 6.2.2, 6.2.3, 7.1.3.2

[154] J. Li, L. Deng, D. Yu, Y. Gong, and A. Acero. HMM adaptation using a phase-sensitive acoustic distortion model for environment-robust speech recognition. In *Proceedings of ICASSP*, pages 4069–4072, 2008. 4.1.3.2, A.2

[155] J. Li, L. Deng, D. Yu, Y. Gong, and A. Acero. A unified framework of HMM adaptation with joint compensation of additive and convolutive distortions. *Computer Speech and Language*, 23(3):389–405, 2009. 3.3.1.1, 3.3.1.1, 3.3.2.3, 3.3.2.3, A.2

[156] J. Li, D. Yu, Y. Gong, and L. Deng. Unscented transform with online distortion estimation for hmm adaptation. In *Proceedings of Interspeech*, pages 1660–1663, 2010. 3.3.2.3

[157] H. Liao. Speaker adaptation of context dependent deep neural networks. In *Proceedings of ICASSP*, 2013. 2.3.1.3

[158] H. Liao and M. J. F. Gales. Joint uncertainty decoding for robust large vocabulary speech recognition. Technical Report CUED/F-INFENG/TR552, University of Cambridge, 2006. 3.3.2.2, 3.3.2.3, 3.3.2.3, 3.4.2, 4.2.3, 4.3, 6.1.3, 6.1.3, 6.2.2, 6.2.2

[159] H. Liao and M. J. F. Gales. Adaptive training with joint uncertainty decoding for robust recognition of noisy data. In *Proceedings of ICASSP*, pages 389–392, 2007. 3.3.2.2, 3.4, 3.4.2, 3.4.2, 6.1.3, 6.1.3, 7.1.3.2

[160] H. Liao and MJF Gales. Issues with uncertainty decoding for noise robust automatic speech recognition. *Speech Communication*, 50(4):265–277, 2008. 3.3.2.2, 3.4.2

[161] M. Lincoln, I. McCowan, J. Vepa, and H. K. Maganti. The multi-channel Wall Street Journal audio visual corpus (MC-WSJ-AV): Specification and initial experiments. In *Proceedings of ASRU*, 2005. 7, 7.2, 1

[162] Y. Linde, A. Buzo, and R. Gray. An algorithm for vector quantizer design. *IEEE Transactions on Communication*, 28(1):84–95, 1980. 3.2.4

[163] L Liporace. Maximum likelihood estimation for multivariate observations of Markov sources. *IEEE Transactions on Information Theory*, 28(5):729–734, 1982. 2.3.1.3

[164] R. Lippmann, E. Martin, and D. Paul. Multi-style training for robust isolated-word speech recognition. In *Proceedings of ICASSP*, volume 12, pages 705–708, 1987. 3.3.2.3

[165] X. Liu, M. J. F. Gales, K. C. Sim, and K. Yu. Investigation of acoustic modeling techniques for lvcsr systems. In *Proceedings of ICASSP*, 2005. 2.2.1

[166] L. Lu. *Subspace Gaussian Mixture Models for Automatic Speech Recognition*. PhD thesis, University of Edinburgh, 2013. 3.3.2.3

[167] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, 1999. 2.4.1

[168] W. J. Masek and M. S. Paterson. A faster algorithm computing string edit distances. *Journal of Computer and System sciences*, 20(1):18–31, 1980. 2.4.3

[169] M. Matassoni et al. Hidden Markov model training with contaminated speech material for distant-talking speech recognition. *Computer Speech and Language*, 16(2):205–223, 2002. 7.1.4

[170] E. McDermott, T. J. Hazen, J. Le Roux, A. Nakamura, and S. Katagiri. Discriminative training for large-vocabulary speech recognition using minimum classification error. *IEEE Transactions on Audio, Speech, and Language Processing*, 15(1):203–223, 2007. 2.3.3.2

[171] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur. Recurrent neural network based language model. In *Proceedings of Interspeech*, pages 1045–1048, 2010. 2.4.1

[172] M. Miyoshi and Y. Kaneda. Inverse filtering of room acoustics. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(2):145–152, 1988. 3.3.3.1

[173] P. Moreno. *Speech recognition in noisy environments*. PhD thesis, Carnegie Mellon University, 1996. 3.3.2.2

[174] P. J. Moreno, B. Raj, and R. M. Stern. A vector taylor series approach for environment-independent speech recognition. In *Proceedings of ICASSP*, pages 733–736, 1996. 3.3.2.2, 3.3.2.2, 2, 3.3.2.2, 3.3.2.3

[175] N. Morgan and H. Bourlard. Factoring networks by a statistical method. *Neural Computation*, 4(6):835–838, 1992. 2.3.1.3

[176] K. P. Murphy. Switching kalman filters. Technical report, UC Berkeley, 1998. 3.3.3.2

[177] Anna K Nábělek, Tomasz R Letowski, and Frances M Tucker. Reverberant overlap and self-masking in consonant identification. *The Journal of the Acoustical Society of America*, 86, 1989. 4.1.1

[178] A. Nádas, D. Nahamoo, and M. A. Picheny. Speech recognition using noise-adaptive prototypes. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(10): 1495–1503, 1989. 4.1.3.2, 6.2.3, B

[179] R. M. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer, 1998. 3.2.1

[180] L. Neumeyer and M. Weintraub. Probabilistic optimum filtering for robust speech recognition. In *Proceedings of ICASSP*, 1994. 3.3.2.2

[181] H. Ney and S. Ortmanns. Dynamic programming search for continuous speech recognition. *IEEE Signal Processing Magazine*, 16(5):64–83, 1999. 2.4.2

[182] H. Ney, U. Essen, and R. Kneser. On the estimation of small probabilities by leaving-one-out. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12): 1202–1212, 1995. 2.4.1

[183] P. Nguyen, C. Wellekens, and J. C. Junqua. Maximum likelihood eigenspace and MLLR for speech recognition in noisy environments. In *Proceedings of European Conference on Speech Communication Technology*, 1999. 3.2

[184] Y. Normandin, R. Cardin, and R. De Mori. High-performance connected digit recognition using maximum mutual information estimation. *IEEE Transactions on Speech and Audio Processing*, 2(2):299–311, 1994. 2.3.3.2, 2.3.3.2

[185] J. J. Odell, V. Valtchev, P. C. Woodland, and S. J. Young. A one pass decoder design for large vocabulary recognition. In *Proceedings of the workshop on human language technology*, pages 405–410, 1994. 2.4.2

[186] J. Olsen, Y. Cao, G. Ding, and X. Yang. A decoder for large vocabulary continuous short message dictation on embedded devices. In *Proceedings of ICASSP*, pages 4337–4340, 2008. 1

[187] M. Omologo, M. Matassoni, P. Svaizer, and D. Giuliani. Microphone array based speech recognition with different talker-array positions. In *Proceedings of ICASSP*, volume 1, pages 227–230, 2007. 3.3.3.1

[188] N. Parihar and J. Picone. Aurora working group: DSR front end LVCSR evaluation AU/384/02. Technical report, Institution for Signal and Information Process, Mississippi State University, 2002. 7.1.1, 8.1

[189] J. Park, F. Diehl, M. J. F. Gales, M. Tomalin, and P. C. Woodland. The efficient incorporation of MLP features into automatic speech recognition systems. *Computer Speech and Language*, 25(3):519–534, 2011. 2.3.1.3

[190] D. Pearce, H. G. Hirsch, et al. The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In *Proceedings of ICSLP*, 2000. 3.3.2.3

[191] D. Povey. *Discriminative training for large vocabulary speech recognition*. PhD thesis, Cambridge University, 2004. 2.3.3, 2.3.3.2, 2.3.3.2

[192] D. Povey and P. C. Woodland. Minimum phone error and i-smoothing for improved discriminative training. In *Proceedings of ICASSP*, 2002. 2.3.3.2

[193] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah. Boosted MMI for model and feature-space discriminative training. In *Proceedings of ICASSP*, pages 4057–4060, 2008. 2.3.3.2

[194] S. J. Press. *Subjective and objective Bayesian statistics: principles, models, and applications*. Wiley, 2009. 3.2.1

[195] P. Price, W. M. Fisher, J. Bernstein, and D. S. Pallett. The DARPA 1000-word resource management database for continuous speech recognition. In *Proceedings of ICASSP*, pages 651–654, 1988. 1

[196] L. Rabiner and B.-H. Juang. *Fundamentals of speech recognition*. Prentice-Hall, 1993. 2.3, 2.3.2

[197] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. 2.3.3

[198] C. K. Raut, T. Nishimoto, and S. Sagayama. Model adaptation for long convolutional distortion by maximum likelihood based state filtering approach. In *Proceedings of ICASSP*, 2006. 3.3.3.3, 3.3.3.3, 3.3.3.3, 3.3.3.3, 4.2.1, 9.1

[199] S. Renals, N. Morgan, H. Bourlard, M. Cohen, and H. Franco. Connectionist probability estimators in HMM speech recognition. *IEEE Transactions on Speech and Audio Processing*, pages 161–174, 1994. 2.3.1.3, 2.3.1.3

[200] H. Robbins. *An empirical Bayes approach to statistics*. Springer, 1985. 3.2.1

[201] H. Robbins. The empirical Bayes approach to statistical decision problems. In *Annual Math Statistics*, pages 49–68. 1985. 3.2.1

[202] A. J. Robinson. An application of recurrent nets to phone probability estimation. *IEEE Transactions on Neural Networks*, 5(2):298–305, 1994. 2.3.1.3

[203] T. Robinson, J. Fransen, D. Pye, J. Foote, and S. Renals. WSJCAM0: A British English speech corpus for large vocabulary continuous speech recognition. In *Proceedings of ICASSP*, pages 81–84, 1995. 7.2

[204] R. C. Rose, E. M. Hofstetter, and D. A. Reynolds. Integrated models of signal and background with application to speaker identification in noise. *IEEE Transactions on Speech and Audio Processing*, 2(2):245–257, 1994. 4.1.3.2

[205] D. B. Rubin and D. T. Thayer. EM algorithms for ML factor analysis. *Psychometrika*, 47(1):69–76, 1982. 3.4.2

[206] D. E. Rumelhart, G. E. Hintont, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. 2.3.1.3

[207] A. Sankar and C.-H. Lee. A maximum-likelihood approach to stochastic matching for robust speech recognition. *IEEE Transactions on Speech and Audio Processing*, 4(3): 190–202, 1996. 3.1, 3.2.2, 3.3.1.2

[208] G. Saon, S. Dharanipragada, and D. Povey. Feature space gaussianization. In *Proceedings of ICASSP*, 2004. 2.2.1, 3.4

[209] J. Schalkwyk, D. Beeferman, F. Beaufays, B. Byrne, C. Chelba, M. Cohen, M. Kamvar, and B. Strope. "your word is my command": Google search by voice: A case study. In *Advances in Speech Recognition*, pages 61–90. 2010. 1

[210] R. Schlüter. *Investigations on discriminative training criteria*. PhD thesis, RWTH, Aache, Germany, 2000. 2.3.3, 2.3.3.2, 2.3.3.2

[211] R. Schwartz and Y.-L. Chow. The N-best algorithms: an efficient and exact procedure for finding the N most likely sentence hypotheses. In *Proceedings of ICASSP*, pages 81–84, 1990. 2.3.3.2

[212] A. Sehr, R. Maas, and W. Kellermann. Reverberation model-based decoding in the logmelspec domain for robust distant-talking speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(7):1676–1691, 2010. 3.1, 3.3.3.3, 1, 2, 9.1

[213] A. Sehr, R. Maas, and W. Kellermann. Frame-wise HMM adaptation using state-dependent reverberation estimates. In *Proceedings of ICASSP*, 2011. 4.2.1

[214] A. Sehr et al. Multi-style training of HMMs with stereo data for reverberation-robust speech recognition. In *Joint Workshop on Hands-free Speech Communication and Microphone Arrays*, 2011. 7.1.4, 7.1.4

[215] F. Seide, G. Li, X. Chen, and D. Yu. Feature engineering in context-dependent deep neural networks for conversational speech transcription. In *Proceedings of ASRU*, pages 24–29, 2011. 2.3.1.3

[216] F. Seide, G. Li, and D. Yu. Conversational speech transcription using context-dependent deep neural networks. In *Proceedings of Interspeech*, pages 437–440, 2011. 2.3.1.3

[217] M. L. Seltzer and A. Acero. Factored adaptation for separable compensation of speaker and environmental variability. In *Proceedings of ASRU*, pages 146–151, 2011. 5.2.1

[218] M. L. Seltzer and A. Acero. Separating speaker and environmental variability using factored transforms. In *Proceedings of Interspeech*, pages 1097–1100, 2011. 5.2.1, 1

[219] M. L. Seltzer and A. Acero. Factored adaptation using a combination of feature-space and model-space transforms. In *Proceedings of Interspeech*, 2012. 3.1, 5.2.1

[220] M. L. Seltzer, B. Raj, and R. M. Stern. Likelihood-maximizing beamforming for robust hands-free speech recognition. *IEEE Transactions on Speech and Audio Processing*, 12 (5):489–498, 2004. 3.3.3.1

[221] M. L. Seltzer, A. Acero, and K. Kalgaonkar. Acoustic model adaptation via linear spline interpolation for robust speech recognition. In *Proceedings of ICASSP*, pages 4550–4553, 2010. 3.3.2.3

[222] K. Shinoda. Speaker adaptation with autonomous control using tree structure. In *Proceedings of European Conference on Speech Communication Technology*, 1995. 3.2.4

[223] K. Shinoda and C.-H. Lee. Structural MAP speaker adaptation using hierarchical priors. In *Proceedings of ASRU*, pages 381–388, 1997. 3.2.1

[224] Y. Shinohara and M. Akamine. Bayesian feature enhancement using a mixture of unscented transformation for uncertainty decoding of noisy speech. In *Proceedings of ICASSP*, pages 4569–4572, 2009. 3.3.2.3

[225] K. C. Sim and M. J. F. Gales. Adaptation of precision matrix models on LVCSR. In *Proceedings of ICASSP*, 2005. 3.2.2.1, 6.2.2, 6.2.2

[226] R. Sinha, S. E. Tranter, M. J. F. Gales, and P. C. Woodland. The Cambridge University March 2005 speaker diarisation system. In *Proceedings of Interspeech*, pages 2437–2440, 2005. 3.1

[227] O. Siohan, Y. Gong, and J. P. Haton. A Bayesian approach to phone duration adaptation for lombard speech recognition. In *Proceedings of European Conference on Speech Communication Technology*, 1993. 3.3.1

[228] F. K. Soong and E.-F. Huang. A tree-trellis based fast search for finding the n-best sentence hypotheses in continuous speech recognition. In *Proceedings of ICASSP*, pages 705–708, 1991. 2.4.2

[229] H. W. Sorenson. *Kalman filtering: theory and application*, volume 38. IEEE, 1985. 3.3.3.2

[230] ETSI standard doc. Speech processing, transmission and quality aspects (STQ); distributed speech recognition; advanced frontend feature extraction algorithm; compression algorithms. Technical Report ES 202 050 v1.1.3, ETSI, 2003. 3.3.2.1, 8.1.1

[231] V. Stouten, H. Van hamme, and P. Wambacq. Accounting for the uncertainty of speech estimates in the context of model-based feature enhancement. In *Proceedings of ICSLP*, 2004. 3.3.2.2

[232] T. Takiguchi and M. Nishimura. Acoustic model adaptation using first order prediction for reverberant speech. In *Proceedings of ICASSP*, 2004. 3.3.3.3, 9.1

[233] A. de la Torre, D. Fohr, and J.-P. Haton. Statistical adaptation of acoustic models to noise conditions for robust speech recognition. In *Proceedings of ICSLP*, 2002. 3.3.1.1

[234] S. E. Tranter, K. Yu, A. Reynolds, G. Evermann, D. Y. Kim, and P. C. Woodland. An investigation into the interaction between speaker diarization systems and automatic speech transcription. *University of Cambridge, Tech. Rep. CUED/F-INFENG/TR464*, 2003. 3.1

[235] S. E. Tranter, K. Yu, G. Evermann, and P. C. Woodland. Generating and evaluating segmentations for automatic speech recognition of conversational telephone speech. In *Proceedings of ICASSP*, 2004. 3.1

[236] V. Valtchev, J. J. Odell, P. C. Woodland, and S. J. Young. MMIE training of large vocabulary recognition systems. *Speech Communication*, 22(4):303–314, 1997. 2.3.3, 2.3.3.2

[237] R. C. van Dalen and M. J. F. Gales. Importance sampling to compute likelihoods of noise-corrupted speech. *Computer Speech and Language*, 27:322–349, 2012. 3.3.1.1, 3.3.2.3

[238] A. P. Varga and R. K. Moore. Hidden markov model decomposition of speech and noise. In *Proceedings of ICASSP*, pages 845–848, 1990. 4.1.3.2

[239] S. V. Vaseghi and B. P. Milner. Noisy speech recognition based on HMMs, Wiener filters and re-evaluation of most likely candidates. In *Proceedings of ICASSP*, pages 103–106, 1993. 3.3.2.1

[240] A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967. 2.4.2, 2.4.2, 2.4.2

[241] Y.-Q. Wang and M. J. F. Gales. Improving reverberant VTS for hands-free robust speech recognition. In *Proceedings of ASRU*, pages 113–118, 2011. (document), 4.1.2

[242] Y.-Q. Wang and M. J. F. Gales. Speaker and noise factorisation on the AURORA4 task. In *Proceedings of ICASSP*, pages 4584–4587, 2011. (document)

[243] Y.-Q. Wang and M. J. F. Gales. Speaker and noise factorisation for robust speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(7), 2012. (document)

[244] Y.-Q. Wang and M. J. F. Gales. Model-based approaches to adaptive training in reverberant environments. In *Proceedings of Interspeech*, 2012. (document)

[245] Y.-Q. Wang and M. J. F. Gales. TANDEM system adaptation using multiple linear feature transforms. In *Proceedings of ICASSP*, 2013. 2.3.1.3

[246] Y.-Q. Wang and M. J. F. Gales. An explicit independence constraint for factorised adaptation in speech recognition. In *Proceedings of Interspeech*, 2013. (document)

[247] Y.-Y. Wang, D. Yu, Y.-C. Ju, and A. Acero. An introduction to voice search. *IEEE Signal Processing Magazine*, 25(3):28–38, 2008. 1

[248] S. Watanabe, Y. Minami, A. Nakamura, and N. Ueda. Application of variational Bayesian approach to speech recognition. In *Proceedings of Advances in NIPS*, pages 1237–1244, 2002. 3.2.1, 3.2.1

[249] T. Watanabe, K. Shinoda, K. Takagi, and K.-I. Iso. High speed speech recognition using tree-structured probability density function. In *Proceedings of ICASSP*, 1995. 3.2.4

[250] B. Widrow, K. Duvall, R. Gooch, and W. Newman. Signal cancellation phenomena in adaptive antennas: Causes and cures. *IEEE Transcription on Antennas and Propagation*, 30(3):469–478, 1982. 3.3.3.1

[251] J. Wishart. The generalised product moment distribution in samples from a normal multivariate population. *Biometrika*, 20:32–52, 1928. 3.2.1

[252] M. Woelfel and J. McDonough. *Distant speech recognition*. Wiley, 2009. 3.3.1, 3.3.3.1

[253] M. Wolfel and F. Faubel. Considering uncertainty by particle filter enhanced speech features in large vocabulary continuous speech recognition. In *Proceedings of ICASSP*, 2007. 3.3.2.2

[254] P. C. Woodland. Speaker adaptation for continuous density HMMs: A review. In *Proc. ISCA ITR-Workshop on Adaptation Methods for Speech Recognition*, 2001. 3, 3.2

[255] P. C. Woodland and D. Povey. Large scale discriminative training of hidden Markov models for speech recognition. *Computer Speech and Language*, 16(1):25–47, 2002. 2.3.3.2

[256] P. C. Woodland, J. J. Odell, V. Valtchev, and S. J. Young. Large vocabulary continuous speech recognition using HTK. In *Proceedings of ICASSP*, 1994. 2.3.1.2, 3.2

[257] P. C. Woodland, M. J. F. Gales, D. Pye, and S. J. Young. The development of the 1996 HTK broadcast news transcription system. In *DARPA speech recognition workshop*, pages 73–78, 1997. 2.2

[258] T. Yoshioka, A. Sehr, M. Delcroix, K. Kinoshita, R. Maas, T. Nakatani, and W. Kellermann. Making machines understand us in reverberant rooms: robustness against reverberation for automatic speech recognition. *IEEE Signal Processing Magazine*, 29(6): 114–126, 2012. 3.3.1, 3.3.3.3

[259] S. J. Young, J. J. Odell, and P. C. Woodland. Tree-based state tying for high accuracy acoustic modelling. In *Proc. of the ARPA workshop on Human Language Technology*, pages 307–312, 1994. 2.3.1.2, 2.3.1.2

[260] S. J. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. C. Woodland. *The HTK Book, version 3.3*. Cambridge University Engineering Department, Cambridge, U. K. , 2005. 2.4.2, 3.2.4

[261] D. Yu and M. L. Seltzer. Improved bottleneck features using pretrained deep neural networks. In *Proceedings of Interspeech*, pages 237–240, 2011. 2.3.1.3

[262] D. Yu, K. Yao, H. Su, G. Li, and F. Seide. KL-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition. In *Proceedings of ICASSP*, 2013. 2.3.1.3

[263] K. Yu and M. J. F. Gales. Adaptive training using structured transforms. In *Proceedings of ICASSP*, 2004. 5.2.1

[264] G. Zavaliagkos, R. Schwartz, and J. Makhoul. Batch, incremental and instantaneous adaptation techniques for speech recognition. In *Proceedings of ICASSP*, volume 1, pages 676–679, 1995. 3.1

[265] H. Zen, N. Braunschweiler, S. Buchholz, M. J. F. Gales, K. Knill, S. Krstulovic, and J. Latorre. Statistical parametric speech synthesis based on speaker and language factorization. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(6):1713–1724, 2012. 5.2.1

[266] Y. Zhao and B. H. Juang. Nonlinear compensation using the Gauss-Newton method for noise-robust speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(8):2191–2206, 2012. 3.3.2.3, 3.3.2.3, 3.3.2.3