



uOttawa

L'Université canadienne
Canada's university

**FACULTÉ DES ÉTUDES SUPÉRIEURES
ET POSTDOCTORALES**



**FACULTY OF GRADUATE AND
POSTDOCTORAL STUDIES**

Jianping Deng

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Electrical Engineering)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Speech Feature Estimation Under the Presence of Noise with Switching Kalman Filter Methods

TITRE DE LA THÈSE / TITLE OF THESIS

Martin Bouchard

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

Tet Yeap

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Aysegul Cuhadar

Yongyi Mao

Wail Gueiaeb

Douglas O'Shaughnessy

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

**SPEECH FEATURE ESTIMATION UNDER THE PRESENCE OF NOISE
WITH SWITCHING KALMAN FILTER METHODS**

Jianping Deng

**Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of**

**Doctor of Philosophy
in Electrical and Computer Engineering**

**Ottawa-Carleton Institute of Electrical and Computer Engineering
School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa**

February 2007

©2007 Jianping Deng, Ottawa, Canada



Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-49335-9
Our file Notre référence
ISBN: 978-0-494-49335-9

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.



Canada

Acknowledgements

I would like to express my sincere gratitude and appreciation to my supervisors, Prof. M. Bouchard and Prof. T. H. Yeap, for their guidance and kind support throughout the duration of this research, especially during the development of this thesis.

I could have never achieved this goal without the steadfast encouragement of my mother.

Finally, I would like to acknowledge the Bell Canada Advanced Research Laboratory in the School of Information Technology and Engineering, University of Ottawa, for providing me with an opportunity to pursue a higher degree, a research assistantship, and the use of research facilities.

Abstract

The performance degradation of a speech recognizer in the presence of additive noise is one of the major problems that still remain unsolved in the application of speech recognition technology. This thesis develops speech enhancement schemes where a noisy speech signal is processed in the feature extraction stage. Since the most popular speech features for speech recognition are Mel-Frequency Cepstral Coefficients (MFCC), vectors of Mel-scaled log-spectrum coefficients or cepstrum coefficients are enhanced.

Three different speech feature enhancement schemes based on switching linear dynamic models (SLDMs) are proposed. The switching linear dynamic model describes the nonlinear and non-stationary time trajectory of speech features by switching among a set of linear dynamic models over time. With the resulting SLDMs as a speech model and a model for noise, speech and noise can be tracked jointly by means of switching Kalman filtering, which involves a weighted sum of filters operating interactively in parallel. Since the distortion caused by additive ambient noises is highly non-linear in the feature domain, the Extended Kalman Filter algorithm (EKF) and the Unscented Kalman Filter algorithm (UKF) have been used to deal with the nonlinear distortion caused by noise in the feature domain.

Comprehensive experiments have been carried out to evaluate the proposed schemes with commonly used databases. The simulation results are presented and compared with other model-based feature enhancement systems in the literature in terms of speech recognition accuracy. Compared with the best results based on the Aurora2 database that we could find in the literature, our approach offers a similar performance (85.96% vs 86.72%) when the EKF is used in our proposed method, resulting in a smaller complexity. When the UKF is used with our method, our approach achieves a better performance (89.60% vs 86.72%).

Abbreviations

AR	Autoregressive
CDCN	Codeword Dependent Cepstral Normalization
DCT	Discrete Cosine Transform
EKF	Extended Kalman Filter
EM	Expectation Maximization
GMM	Gaussian Mixture Model
GPB	Generalized Pseudo Bayesian
GPB1	First order Generalized Pseudo Bayesian
GPB2	Second order Generalized Pseudo Bayesian
HMM	Hidden Markov Model
HMME	Hidden Markov Mixtures of Experts
HTK	Hidden Markov Model Toolkit
IMM	Interacting Multiple Models
LDM	Linear Dynamic Model
LKF	Linearized Kalman Filter
LPC	Linear Predictive Coding
LPCC	Linear Predictive Cepstral Coefficients
LP-HMM	Linear Predictive Hidden Markov Model
MAP	Maximum A Posteriori
ME	Mixture of Experts
MFCC	Mel-frequency Cepstral Coefficients
ML	Maximum Likelihood
MLP	Multilayer Feedforward Neural Network
MMSE	Minimum Mean Square Error
PLP	Perceptual Linear Prediction
RBF	Radial Basis Function Neural Network
SLDM	Switching Linear Dynamic Model
SNR	Signal Noise Ratio
SPLICE	Stereo-based Piecewise Linear Compensation
UKF	Unscented Kalman Filter

UT

Unscented Transformation

VTS

Vector Taylor Series

Contents

Acknowledgements	i
Abstract	ii
Abbreviations	iii
Contents	v
List of Figures	x
List of Tables	xi

CHAPTER 1	Introduction	1
------------------	---------------------	----------

1.1	Motivation	2
1.2	Objectives	5
1.3	Contributions	6
1.4	Organization	8

CHAPTER 2	Framework for Speech Recognition	10
------------------	---	-----------

2.1	Speech Production System	10
2.2	Speech Recognition System	11
2.2.1	Standard Front-Ends	13
2.2.1.1	LPCC	14
2.2.1.2	MFCC	15

2.2.1.3	Additional Post-Processing	16
2.2.2	Acoustic Models	17
2.2.2.1	HMM model	18
2.2.2.2	GMM model	24
2.2.3	Search Algorithm and Scoring	25
2.3	Summary	27

CHAPTER 3	Review of Speech Enhancement	28
------------------	-------------------------------------	-----------

3.1	Speech Waveform Enhancement	28
3.1.1	Spectral Subtraction	28
3.1.2	Kalman Filter Based methods	30
3.1.3	Noise Masking	33
3.1.4	Statistical Model Based Approaches	34
3.1.4.1	Spectral Amplitude Model Based Methods	35
3.1.4.2	Hidden Markov Model Based Methods	35
3.2	Speech Feature Enhancement	37
3.2.1	Statistical-Model-Based Approach	38
3.2.1.1	CDCN	39
3.2.1.2	VTs-based Speech Feature Compensation	40
3.2.1.3	SPLICE	41
3.2.1.4	Extensions and Variations	43
3.2.2	Filter-Model-Based Approaches	45
3.3	Summary	46

CHAPTER 4	Experimental Tasks and Corpora	47
------------------	---------------------------------------	-----------

4.1	Experimental Conditions for Numbers v1.3 Database	47
4.1.1	Speech Database	47

4.1.2	Front-end Setting	48
4.1.3	HTK Reference Recognizer	48
4.2	Experimental Conditions for Aurora 2.0 Database	50
4.2.1	Speech Database	50
4.2.2	Front-end Setting	51
4.2.3	HTK Reference Recognizer	52
4.3	Scoring	52
4.4	Why Test on Both Number v1.3 and Aurora 2.0 Databases	53

CHAPTER 5	Speech Enhancement By Noise-Tracking-Only Method	55
------------------	---	-----------

5.1	State-Space for Noise-Tracking-only Method	56
5.2	Review of Switching Kalman Filter	58
5.2.1	Introduction	58
5.2.2	First Order Generalized Pseudo-Bayesian Approach	60
5.2.3	Second Order Generalized Pseudo-Bayesian Approach	62
5.2.4	Interacting Multiple Model (IMM) Approach	64
5.2.5	Comparison of GPB1, GPB2 and IMM Approximation	66
5.3	Switching Kalman Filter for Noise-Tracking-Only Method	68
5.3.1	Switching Extended Kalman Filter for Noise Tracking	68
5.3.2	Switching Unscented Kalman Filter for Noise Tracking	71
5.4	Speech Estimation by Minimum Mean Squared Estimation	74
5.5	Simulation Results and Analysis	76
5.4.1	Simulation Results on Database Number v1.3	76
5.4.2	Simulation Results on the Aurora 2.0 Database	80
5.6	Summary	84

CHAPTER 6	Linear Predictive HMM for Speech Feature Enhancement	85
------------------	---	-----------

6.1	Introduction	85
-----	--------------	----

6.2	Linear Predictive HMM (LP-HMM) as Speech Model	89
6.3	Joint-Speech-Noise-Tracking Based on Linear-Predictive-HMM	95
6.3.1	State-Space for Joint-Speech-Noise-Tracking	96
6.3.2	Switching Kalman Filter with the GPB1 algorithm	97
6.3.3	Switching Kalman Filter with the IMM algorithm	98
6.3.4	Extended Kalman Filter for Joint-Speech-Noise-Tracking	99
6.3.5	Unscented Kalman Filter for Joint-Speech-Noise-Tracking	101
6.4	Simulation Results and Analysis	103
6.4.1	Simulation Results on the Number v1.3 Database	103
6.4.2	Simulation Results on the Aurora 2 Database	107
6.5	Summary	110

CHAPTER 7	Linear Dynamic Models with Mixture of Experts Architecture for Speech Feature Enhancement	112
------------------	--	------------

7.1	Introduction	112
7.2	Mixture of Linear Dynamic Models as Speech Model	116
7.3	Speech Feature Enhancement Based on Mixture of Linear Dynamic Models	123
7.4	Simulation Results and Analysis	126
7.4.1	Simulation Results on the Number v1.3 Database	126
7.4.2	Simulation Results on the Aurora 2 Database	132
7.5	Summary	136

CHAPTER 8	Time-Variant Linear-Predictive-HMM for Speech Feature Enhancement	137
------------------	--	------------

8.1	Introduction	137
8.2	Time-variant Linear-Predictive-HMM as Speech Model	143

8.3	Speech Feature Enhancement Based on Time-Variant LP-HMM	147
8.4	Simulation Results and Analysis	150
8.4.1	Simulation Results on the Number v1.3 Database	150
8.4.2	Simulation Results on the Aurora 2 Database	153
8.5	Summary	158
CHAPTER 9	Conclusions and Future Work	159
9.1	Summary of Results and Conclusions	160
9.2	Recommendations for Future Work	165
BIBLIOGRAPHY		167

List of Figures

2.1	A general speech recognition system	12
2.2	Front-end processing using overlapping window functions	14
2.3	An example of Mel-spaced filterbank	15
2.4	Block diagram of the MFCC processor	16
2.5	Hidden Markov Model generating observation vectors	18
2.6	Recognition network	26
5.1(a)	GPB1 algorithm	67
5.1(b)	GPB2 algorithm	67
5.1(c)	IMM algorithm	68
5.2	Typical temporal sequence of log-spectrum coefficients	76
6.1	Graphical representation of an HMM with Gaussian output	86
6.2	Graphical representation of HMM-based switching LDM	87
6.3	Graphical representation of the switching LDM in [Dro04] and [Kim05]	87
7.1	Illustration of the mixture of experts (ME) structure	113
7.2(a)	Graphical representation of the switching LDM in [Dro04] and [Kim05]	115
7.2(b)	Graphical representation of the mixture of linear dynamic models	116
7.3	Block diagram of the proposed mixture of LDMs	116
8.1(a)	Graphical representation of the linear-predictive HMM	139
8.1(b)	Graphical representation of the mixture of linear dynamic models	140
8.1(c)	Graphical representation of the time-variant linear-predictive HMM	140
8.2	Block diagram of the proposed time-variant linear-predictive HMM	141
8.3	Block diagram of the transition network used in [Lie99][Wan04]	142
8.4	Block diagram of the proposed transition network	142

List of Tables

5.1	Recognition accuracy (%) for speech mixed with white noise when speech is enhanced in the log-spectrum domain	77
5.2	Recognition accuracy (%) for speech mixed with babble noise when speech is enhanced in the log-spectrum domain	77
5.3	Recognition accuracy (%) for speech mixed with white noise when speech is enhanced in the cepstrum domain	78
5.4	Recognition accuracy (%) for speech mixed with babble noise when speech is enhanced in the cepstrum domain	78
5.5	Accuracy (%), Aurora 2	81
5.6	Accuracy (%) baseline without noise compensation, Aurora 2	81
5.7	Accuracy (%) with speech enhanced by VTS	81
5.8	Accuracy (%) with speech enhanced by 100 particle filters	82
5.9	Accuracy (%) with speech enhanced by 16 Unscented Kalman filters	82
5.10	Summary of accuracy (%) on Aurora 2.0, set A	83
6.1	Recognition accuracy (%) for speech mixed with white noise when speech is enhanced in the log-spectrum domain	104
6.2	Recognition accuracy (%) for speech mixed with babble noise when speech is enhanced in the log-spectrum domain	105
6.3	Recognition accuracy (%) for speech mixed with white noise when speech is enhanced in the cepstrum domain	105
6.4	Recognition accuracy (%) for speech mixed with babble noise when speech is enhanced in the cepstrum domain	106
6.5	Accuracy (%) with speech enhanced by the switching Extended Kalman filters based on the proposed SLDM	108
6.6	Accuracy (%) with speech enhanced by the switching Unscented Kalman filters based on the proposed SLDM	109
6.7	Summary of accuracy (%) on Aurora 2.0, set A	109
6.8	Number of LDMs in the SLDM	110

7.1	Recognition accuracy (%) for speech mixed with white noise when speech is enhanced in the log-spectrum domain	128
7.2	Recognition accuracy (%) for speech mixed with babble noise when speech is enhanced in the cepstrum domain	129
7.3	Recognition accuracy (%) for speech mixed with white noise when speech is enhanced in the log-spectrum domain	129
7.4	Recognition accuracy (%) for speech mixed with babble noise when speech is enhanced in the cepstrum domain	130
7.5	CPU time (sec) for the enhancement of one utterance when speech is enhanced in the log-spectrum domain	130
7.6	Accuracy (%) with speech enhanced by the switching Extended Kalman filter based on the proposed SLDM	133
7.7	Accuracy (%) with speech enhanced by the switching Unscented Kalman filter based on the proposed SLDM	133
7.8	Summary of accuracy (%) on Aurora 2.0, set A	134
7.9	Number of LDMs in the SLDM	134
7.10	CPU time (sec) for the enhancement of one utterance when speech is enhanced by the switching EKF algorithm	135
8.1	Recognition accuracy (%) for speech mixed with white noise when speech is enhanced in the log-spectrum domain	152
8.2	Recognition accuracy (%) for speech mixed with babble noise when speech is enhanced in the log-spectrum domain	152
8.3	CPU time (sec) for the enhancement of one utterance when speech is enhanced by the switching EKF algorithm	153
8.4	Accuracy (%) with speech enhanced by the switching Extended Kalman Filter based on the proposed SLDM	155
8.5	Accuracy (%) with speech enhanced by the switching Unscented Kalman Filter based on the proposed SLDM	155
8.6	Summary of accuracy (%) on Aurora 2, set A	156
8.7	Number of LDMs in the SLDM	156
8.8	CPU time (sec) for the enhancement of one utterance	

when speech is enhanced by the switching EKF algorithm	157
8.9 CPU time (sec) for the enhancement of one utterance	
when speech is enhanced by the switching UKF algorithm	157
9.1 Recognition accuracy (%) for speech mixed with white noise	
when speech is enhanced by the switching EKF algorithm	161
9.2 Recognition accuracy (%) for speech mixed with babble noise	
when speech is enhanced by the switching EKF algorithm	161
9.3 CPU time (sec) for the enhancement of one utterance	
when speech is enhanced by the switching EKF algorithm	162
9.4 Summary of accuracy (%) on Aurora 2, set A, by the switching EKF algorithm	163
9.5 Summary of accuracy (%) on Aurora 2, set A, by the switching UKF algorithm	163
9.6 CPU time (sec) for the enhancement of one utterance	
when speech is enhanced by the switching EKF algorithm	164
9.7 CPU time (sec) for the enhancement of one utterance	
when speech is enhanced by the switching UKF algorithm	164

Chapter 1

Introduction

Speech is one of the most important tools that people use to communicate ideas. Advances in telecommunications over the last few decades have made speech communications anywhere a reality. The freedom and flexibility provided by speech communications introduce new challenges, one of the most prominent ones being the suppression of noise. For example, mobile phone users communicate in different environments with varying amounts and types of background noise and channel noise. These problems call for the use of speech enhancement.

Speech enhancement is defined as the attempt to improve the performance of speech communication systems when their input signal is corrupted by noise [Eph92]. Usually a speech enhancement system is based on certain assumptions (for example, the type of noise source, the way in which the noise interacts with the clean signal) and constraints (such as the number of input channels or microphones available for enhancement), which are typically dependent on the application and the environment. The interfering noise may result from background sources such as road noise or from the communication channel. Noise from background sources is generally modeled as additive noise whereas communication channel noise is modeled as convolutional noise. Environmental influences can also cause changes in speech articulation [Rab93]. For example, a speaker may shout to be heard in extreme noise. This phenomenon is called the Lombard effect [Rab93] and is difficult to model.

Broadly speaking, the field of speech enhancement is interested in addressing three (not necessarily compatible) objectives [Eph92]: (a) the improvement of perceptual quality of noisy speech, (b) the immunization of speech encoders against input noise, and (c) the improvement of performance of speech and speaker recognition systems in the presence of noise. If the first criterion for performance improvement is used, the enhancement

system attempts to improve the quality and/or intelligibility of the corrupted speech. These measures are related to listener fatigue and understanding, respectively. The ultimate test of the effectiveness of the enhancement technique is human evaluation. The enhancement schemes for performance improvement with the second objective typically have applications as front ends to speech coders. This thesis deals with the third objective: speech enhancement for the purpose of improving the accuracy of speech or speaker recognition systems. Our study is focused on telephone-driven applications, where a single acquisition channel is available. The interfering noise may be stationary or non-stationary, and only additive noise is considered. Noise is typically modeled independently of the speech. Communication channel noise and the Lombard effect are not considered. The enhancement system developed is to be used as a front end to a speech/speaker recognizer.

1.1 Motivation

It is well known that the performance of speech recognition systems degrades in the presence of noise [Gon95][Vii01]. Current state-of-the-art ASR (Automatic Speech Recognition) systems lag far behind human speech recognition in terms of noise robustness, and improving ASR in noisy environments remains a challenging problem. It should be noted that robust speech recognition is a complete research topic itself, and that using speech enhancement as a front-end to a recognizer is only one solution to the problem. A review of approaches to robust speech recognition can be found in [Gon95]. The approaches can be divided into the following three categories:

- (1) *Use of Noise Resistant Features* - This approach assumes that the system is independent of noise, with the same configuration used for both clean and noisy speech. Therefore these techniques attempt to derive features and parameters of speech that are noise resistant.
- (2) *Speech Model Compensation for Noisy Environments* - Model Compensation entails the transformation of speech recognition models created in a reference environment

(usually clean environment) to accommodate a specific noisy environment in order to recognize noisy speech. This technique, rather than attempting to derive a clean estimate of speech, allows for noise in the recognition process itself.

(3) *Speech Enhancement* - This involves the transformation of noisy speech into as close an approximation of the training environment as possible. Usually clean speech is used to train the acoustic models. By preprocessing speech, enhancement techniques are intended to recover either the waveform or specific parameters of clean speech. Therefore speech enhancement techniques are useful as a front end for speech recognition systems.

The approaches based on the usage of noise-resistant features have the advantage that they are applicable to wide ranges of noisy environments, since there is no assumption about the characteristics of the noise. But these techniques are not the most successful in the literature. Compared with them, speech model compensation approaches usually give better results. But the problem is that a speech recognition model compensated in one noise environment may only be used in that specific noise environment, making the system too specific for general use. Some earlier work, such as [Den01], also demonstrated a superior performance of the speech enhancement approach over the speech recognition model compensation one. Considering these factors, we choose speech enhancement as the concern of this thesis.

Research in speech enhancement over the last three decades has resulted in a number of different approaches. The techniques are categorized according to the type of models, if any, to model the speech and noise, and the amount of prior information included. In general, the use of more accurate speech and noise models and incorporation of relevant prior information leads to better enhancement at the expense of greater computational complexity. Among the most successful speech enhancement algorithms have been spectral subtraction [Bol79][Lim79], MMSE [Eph84], and Kalman filter [Pal87]. A detailed description is given in Chapter 3. It should be noted that these techniques are usually not directly related to improving speech recognition performance. Most of these techniques were originally developed for speech quality enhancement, and while the

distortion caused by those techniques is tolerable for humans, they may cause problems for recognition systems. On the other hand, distortion measures used by speech and speaker recognition systems may not correspond to a guaranteed quality or intelligibility improvement [Log97]. Towards solving the noise compensation problem in speech and speaker recognition systems, in the past few years a variety of techniques have been developed. One of the prevailing approaches is speech features enhancement, where a noisy speech signal is processed in the feature extraction stage. Instead of producing an enhanced waveform, an enhanced version of the recognition parameters is produced. It is believed that enhancement in the speech feature domain is most desirable if the purpose of speech enhancement is for robust speech recognition, since this is the domain as close as possible to the back end of the recognizer [Den04b]. Since the most popular speech features for speech recognition are Mel-Frequency Cepstral Coefficients (MFCC), vectors of Mel-scaled log-spectrum coefficients or cepstrum coefficients are enhanced, as they are an intermediate step in the computation of the MFCC coefficients. One of the difficulties with these approaches is that the distortion caused by additive ambient noises is highly non-linear in the feature domain. While numerical integration methods [Myr03] and particle filtering [Umb05] [Fuj05] can be used to solve the nonlinear estimation, in most of the approaches simpler estimators are obtained by linearization, e.g., by Taylor series approximation [Mor96] or statistical linear approximation [Kim98a]. Another difficulty is that the trajectory of speech features exhibits a complex and rich dynamic behavior that is both nonlinear and time varying. Since there is not an explicit dynamic model for a speech feature sequence, a Gaussian mixture model (GMM) is usually used. With GMM as the speech model, the noise is considered as the state variable to be estimated, whose observation is corrupted by speech. Clean speech estimation is then obtained by removing the noise sequence from the noisy signal sequence. While in some approaches noise is considered stationary, some approaches explicitly address non-stationary noise, among which are sequential expectation maximization (EM) [Kim98c][Afi04], sequential maximum a posteriori (MAP) [Den04a], Kalman filter [Kim98b], and particle filtering [Fuj05]. However, it is believed that joint noise and speech tracking should yield a better enhancement result, as is done in [Dro04], and that is the method followed by this thesis. The major obstacle with this approach is how to

represent the nonlinear and non-stationary time trajectory of speech features by a dynamic model, which is a problem that has not been well solved in the literature.

1.2 Objectives

The objective of this thesis is to develop model-based speech enhancement approaches to compensate the noise in the speech feature domain. Such systems include an a priori model for speech and a model for noise, as well, within the enhancement algorithm. Accurate estimation of clean speech requires that the models be robust and faithful representations of reality. So the proposed speech enhancement methods are composed of two parts: a speech model estimate and a filtering algorithm.

Inspired by the divide-and-conquer principle underlying the mixture-of-experts (ME) neural network [Wei95] and the autoregressive HMM (AR-HMM) architecture, which automatically segments a non-stationary time series into discrete stationary regimes [Pen98], a switching linear dynamic model is developed. The switching linear dynamic model describes the dynamics of a complex, non-stationary process by switching among a set of linear dynamic models over time. We investigate the appropriateness of the switching linear dynamic model to be the dynamic model of speech features in the log-spectra and cepstral domains. This includes the presentation of the theoretical aspects of the proposed architecture, the derivation of an appropriate learning algorithm, and the evaluation of experimental performance.

With the resulting model, speech and noise can be tracked jointly by means of switching Kalman filters. Unfortunately, exact inference in switching Kalman filters is intractable in practice. The optimal minimum mean square error estimate involves a bank of filters tuned to all the possible parameter histories, which makes the cost in computations grow exponentially with the data length [Sha93]. To solve this problem, we therefore need to use approximate inference. We investigate how to adapt the approximate inference to the switching linear dynamic model obtained and analyze the effectiveness of different approximation algorithms on improving the speech recognition accuracy.

1.3 Contributions

The main contributions of this thesis are listed as follows:

1. Three different switching linear dynamic models (SLDM), named in this thesis as linear-predictive-HMM, switching linear dynamic model with a mixture of experts (ME) structure, and time-variant linear-predictive-HMM, have been proposed to model the distribution of speech for the purpose of speech enhancement under a noisy environments. We investigate the appropriateness of the proposed switching linear dynamic models to be the dynamic models of speech features in the Mel-scaled log-spectra and cepstral domains.

In the linear-predictive-HMM, each multivariate linear prediction model is associated with a hidden state of a HMM to describe the temporal correlations among adjacent frames of speech features. Switching between the linear dynamic models is based on the internal information extracted from the estimation of present HMM state and state transition probability. In the switching linear dynamic model with a mixture of experts (ME) structure, the proposed model takes the input information into consideration when partitioning and switching between the complex input spaces. External information is presented in the form of inputs to a neural network, called a gating network. In the time-variant linear-predictive-HMM model, rather than holding a transition probability for the whole process as in the linear-predictive-HMM, a connectionist model is employed to learn the time variant transition probabilities from the dynamic situation of the time series. The new speech model integrates the following information: (1) input information, (2) predictor performance information, (3) previous state information. It can be interpreted as a combination of the switching linear dynamic model with a mixture of experts (ME) structure and the linear-predictive-HMM model previously proposed.

2. With the resulting SLDMs as a speech model and a model for noise, we propose to track the speech and noise jointly by means of switching Kalman filtering. We study

the efficiency of two different inference approximations, the Generalized Pseudo Bayesian (GPB) algorithm and the Interacting Multiple Model (IMM) algorithm, to handle the interactions between the parallel Kalman filters. The Extended Kalman Filter algorithm (EKF) and the Unscented Kalman Filter algorithm (UKF) are proposed to deal with the nonlinear distortion caused by noise in the feature domain.

3. Comprehensive experiments have been carried out to evaluate the proposed schemes with commonly used databases. The performances of the proposed schemes for speech enhancement in both the log-spectral domain and the cepstral domain are presented and compared in terms of speech recognition accuracy with other model-based feature enhancement systems found in the literature.

Parts of the technical contributions of this thesis have been published (or submitted) in the following papers:

- J. Deng, M. Bouchard, and T.H. Yeap, "Feature enhancement for noisy speech recognition with a time-variant linear predictive HMM structure", submitted to *IEEE Transactions on Audio, Speech and Language Processing*
- J. Deng, M. Bouchard, and T.H. Yeap, "Noisy speech feature estimation on the Aurora 2 database using a Switching linear dynamic model", invited paper, submitted to *Journal Of Multimedia*
- J. Deng, M. Bouchard, and T.H. Yeap, "Linear dynamic models with mixture of experts architecture for recognition of speech under additive noise conditions", *IEEE Signal Processing Letters*, Vol. 13, pp. 573-576, Sept. 2006.
- J. Deng, M. Bouchard, and T.H. Yeap, "Speech feature estimation under the presence of noise with a switching linear dynamic model", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'06*, Vol.1, pp 497-500, Toulouse, France, May 2006.
- J. Deng, M. Bouchard, and T.H. Yeap, " Noise compensation using interacting multiple Kalman filters", *Proceedings of INTERSPEECH'05-9th European Conference on Speech Communication and Technology*, pp. 2645-2648, Lisbon, Portugal, Sept. 2005.

- J. Deng, M. Bouchard, and T.H. Yeap, “Speech enhancement using a switching Kalman filter with a perceptual post-filter”, *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP’05*, Vol. 1, pp. 1121-1124, Philadelphia, March 2005.

1.4 Organization

Chapter 2 begins with a brief introduction to the anatomy and physiology of human speech production systems. Next, some background about speech recognition systems is provided. The theory of Hidden Markov models (HMM) and Gaussian Mixture Models (GMM) is outlined. How to model the HMMs and the GMMs with the expectation maximization algorithm (EM) is described.

Chapter 3 summarizes the main methods of speech enhancement presented in the literature, including the speech enhancement methods in the time domain and the speech enhancement methods in the feature domain.

In chapter 4, the speech recognition engines and the speech databases that are used in the experiments are addressed, as they shall be used to test the algorithms developed in this thesis.

A comparative study of model-based feature enhancement systems is given in chapter 5. At first, an overview of the switching Kalman filtering that is applied in the model-based feature enhancement system is given. The prevailing noise-tracking idea for speech enhancement is then reviewed, in which noise is considered as the state variable to be tracked. A switching Kalman filter is used to estimate the noise and then speech is enhanced by removing the noise estimate. We investigate the drawbacks of this noise-tracking idea, which leads to the introduction of switching models for speech in the following chapters, in order to track the speech and noise sequences jointly.

Chapter 5, 6, 7 and 8 contain the main contributions of this research thesis. In chapter 6, a speech enhancement scheme based on linear predictive hidden Markov models is developed. The experimental results with the proposed method are presented, and a discussion follows.

A mixture model with linear continuous state evolution processes is described in Chapter 7. The learning algorithms to estimate the model parameters are given. A speech enhancement scheme based on the proposed model is developed. Experimental evaluation of the proposed method for speech enhancement is carried out, and the results are analyzed.

A switching linear dynamic model based on the concept of hidden Markov models (HMM) but with time-variant transition probability is described in Chapter 8. The learning algorithms to estimate the model parameters are given. A speech enhancement scheme based on this new model is proposed. The proposed method is then evaluated based on the experimental results obtained.

In Chapter 9, the main results obtained in this thesis are summarized. A discussion on potential future work is given.

Chapter 2

Framework for Speech Recognition

Since the algorithms to be developed in this thesis will be evaluated in the context of continuous speech recognition, this chapter will give a short review on the production of speech in humans and provide an overview of the basic structure of the recognition system used for the experiments described in this thesis. The performance index for speech recognition systems will be given. Finally speech recognition engines constructed based on HTK (Hidden Markov Model Toolkit [HTK99]) will be described. It should be noted that the algorithms developed in this thesis are independent of the recognition engine used, as long as the speech features used in the recognition engine are Mel-frequency cepstral coefficients (MFCC); hence, the results and conclusions of this thesis could be applicable to other recognition systems, for example, speaker recognition systems, which share a similar front-end and a similar acoustic model with speech recognition systems.

2.1 Speech Production System

The process of speech communication involves the production of an acoustic wave by the speaker and the perception of the signal by the listener. Though the process of speech perception still largely remains a mystery to the scientific world, the process of speech production has been well researched and understood. When air is expelled from the lungs by the diaphragm through the trachea, the tensed vocal cords within the larynx are caused to vibrate. The airflow is chopped into quasi-periodic pulses which are then modulated in frequency when passing through the throat cavity, the mouth cavity and the nasal cavity. Depending on the positions of various articulators (i.e., velum, tongue, lip), different sounds are produced. A detailed description of speech production can be found in [Del93] [Rab93].

A sequence of various sound units constitutes a speech signal. The sound units, used as basic theoretical units for expressing linguistic meaning, are phonemes. A *phoneme* describes the linguistic meaning conveyed by a particular speech sound. The American English language consists of about 42 phonemes, which can be classified as vowels, semivowels, diphthongs and consonants (fricatives, nasals, affricates, aspiration and stops). Vowels are produced due to the periodic vibrations of the vocal cords in the larynx. Fricatives are caused by the turbulence of the air passing through narrow constrictions in the vocal tract, causing a random noise-like sound. Nasals are caused by acoustically coupling the nasal cavity to the pharyngeal cavity by lowering the velum. Building up pressure in front of the vocal tract and abruptly releasing it produces plosives. Thus, a phoneme basically represents a class of sounds with similar acoustical meaning.

2.2 Speech Recognition System

Figure 2.1 illustrates a contemporary ASR system [Ros04]. The system consists of five main blocks: the front-end, acoustic models, language model, lexicon and search algorithm. Incoming speech is subject to some form of front-end signal processing that outputs a sequence of acoustic vectors, $O = \{o_1, \dots, o_T\}$. This is called feature parameter extraction. The speech recognizer has to choose the word sequence, $W = \{w_1, \dots, w_L\}$, with the maximum posterior probability given the observation sequence as follows

$$\hat{W} = \arg \max_w P(W | O) = \arg \max_w \frac{P(O | W) p(W)}{P(O)} \quad (2.1)$$

where Bayes' formula has been applied to obtain the final form.

The term $P(O | W)$ is called the *acoustic model*. In the context of statistical methods for speech recognition, Hidden Markov Models (HMM) are currently the most popular and successful approach for characterizing the spectral properties of speech frames. For small-vocabulary systems with sufficient training data it is possible to build a HMM for each word, while in large-vocabulary systems, it is unlikely that there are enough training examples of each word in the dictionary. HMMs representing some sub-word units have

to be used. Linguistic units such as phonemes are the most commonly used sub-word speech units. In that case, with a *lexicon* to dictate how these sub-word units are linked together to form individual words, a composite HMM is constructed by joining in sequence the HMMs corresponding to sub-word units. The second term $P(W)$, generally called *language model*, contains the information about which word sequences are allowable and the probability of such sequences. The language model is called a *grammar* when only the syntactic constraints are used. It can reduce the computational load and improve the accuracy of the recognizer by limiting the word search space. It should be noted that the likelihood of the observation sequence, $P(O)$, may be omitted in the maximization since it is independent of the word sequence. A *Search algorithm* implements the maximization in Equation 2.1. The performance of speech recognition systems is then evaluated by comparing the recognized hypothesis to a reference transcription, to produce the word error rate.

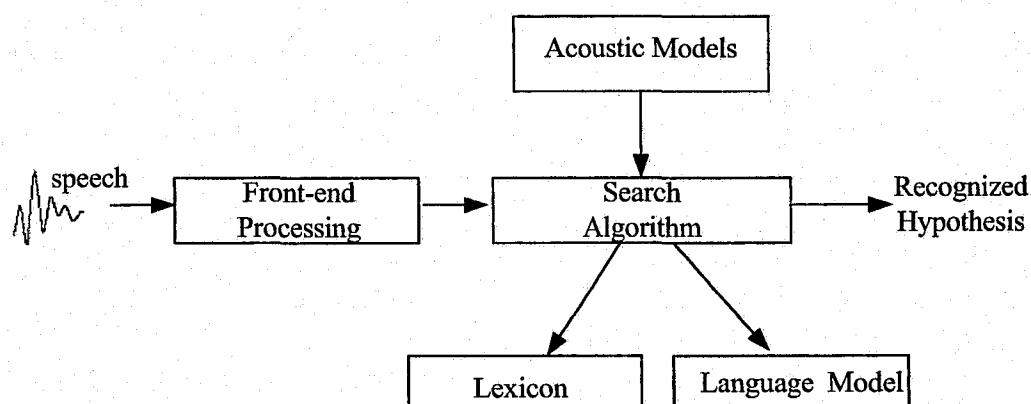


Figure 2.1 A general speech recognition system

The five blocks are introduced in more detail in the following sections. First, the front-end block is reviewed. The theory of Hidden Markov Models (HMM) is then presented along with schemes to optimize its parameters. The theory of Gaussian Mixture Models (GMM) is introduced as a particular type of HMM. Next an efficient search algorithm to find the maximum likelihood state sequence called the Viterbi algorithm [Rab93] is briefly described.

2.2.1 Standard Front-Ends

In a speech/speaker recognition system, first the acoustic waveform is recorded and sampled typically at 8 or 16 kHz to allow processing by a digital device. It is not easy to compare the sampled acoustic waveform directly due to the varying acoustic characteristics. Instead, the speech waveform is converted to some type of parametric representation, such as the short-term spectra, for further analysis and comparison.

The speech signal is assumed to be quasi-stationary so that its characteristics are fairly stationary when divided into short frames, often 25 ms. A window function (e.g., Hamming) is applied to each frame to reduce boundary effects in the subsequent processing. For each frame a new observation vector is produced. There are overlaps between the adjacent frames. This process is illustrated in Figure 2.2.

The most commonly used parametrical representations are Linear Predictive Cepstral Coefficients (LPCC) [Rab93], Mel-frequency cepstral coefficients (MFCC) [Dav80] and perceptual linear prediction (PLP) coefficients [Her90]. LPCC is based on the Linear Predictive coding (LPC) model, which is also known as autoregressive modeling or AR-modeling, and is widely used in many speech research areas. The LPC model uses an all-pole transfer function to model the vocal tract and estimate the power spectrum of the speech. A disadvantage of the LP model is that it approximates the speech at all frequencies in the same manner, which is not a suitable approach for hearing because it is more sensitive in the mid-range frequencies. Instead, MFCC is the most popular representation in speech recognition applications. The main idea of MFCC is to take advantage of characteristics derived from the physical acoustic properties of the human ear for estimating the audible spectrum. The LPCC and MFCC feature extractors will be described in more detail next.

2.2.1.1 LPCC

The linear predictive coding (LPC) performs spectral analysis on windows of speech using an all-pole modeling constraint [Mak75]. Within each window the sample at time t , $s(t)$ is assumed to be expressible as a linear combination of the past p samples plus some excitation term, $Gu(t)$:

$$s(t) \approx \sum_{i=1}^p a_i s(t-i) + Gu(t) \quad (2.2)$$

where a_i are assumed to be constant over the window of speech, G is the gain and $u(t)$ is the normalized excitation. The parameter p is called the LPC analysis order. The output of the LPC analysis on a window of speech is the vector of p coefficients (known as the autoregressive coefficients), $a_1 \dots a_p$, that specify the spectrum of an all-pole model that best matches the spectrum of frequencies inside the window. The cepstral coefficients, which have been shown to be a more robust, reliable feature set for speech/speaker recognition than the LPC coefficients, can be derived directly from the LPC coefficients as [HTK99]:

$$c_n = -a_n + \frac{1}{n} \sum_{i=1}^{n-1} (n-i) a_i c_{n-i} \quad 0 \leq n \leq p \quad (2.3)$$

where c_i is the i^{th} cepstral parameter calculated from the LPC parameters.

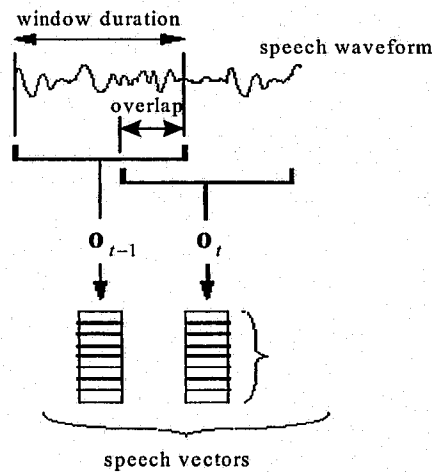


Figure 2.2 Front-end processing using overlapping window functions

2.2.1.2 MFCC

Psychophysical studies have shown that human perception of the frequency content of sounds does not follow a linear scale. Thus for each tone with an actual frequency f measured in Hz, a subjective pitch is measured on a scale called the 'mel' scale. The Mel-frequency scale is a linear frequency spacing below 1000 Hz and a logarithmic spacing above 1000 Hz. We can use the following approximate formula to compute the Mels for a given frequency f in Hz:

$$mel(f) = 2595 * \log_{10}(1 + f / 700) \quad (2.4)$$

One approach to simulating the subjective spectrum is to use a filter bank, spaced uniformly on the Mel scale, and each filter bank has a triangular bandpass frequency response as shown in Figure 2.3.

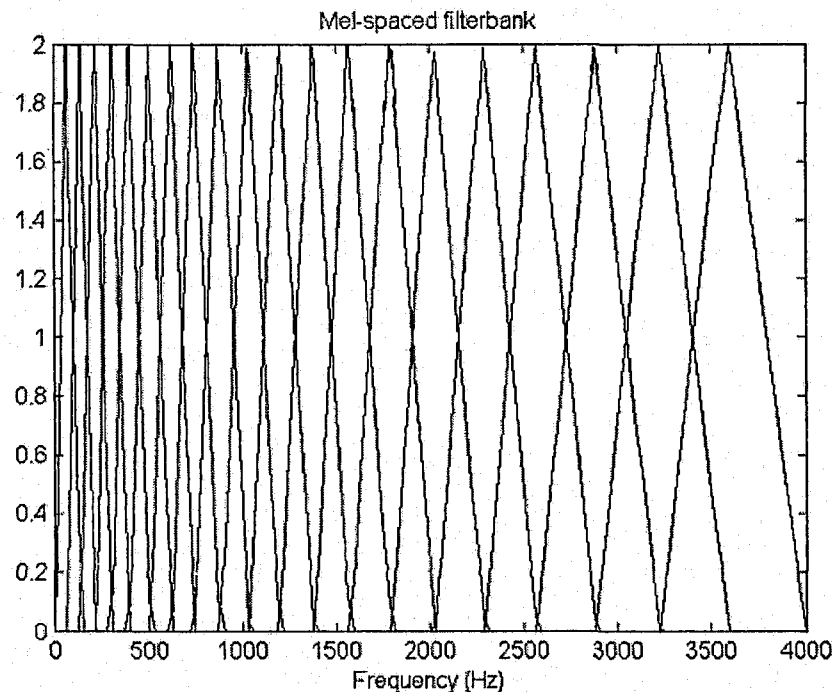


Figure 2.3 An example of Mel-spaced filterbank

To parameterize the speech signal into a sequence of Mel-frequency cepstral coefficients (MFCC), first each frame is transformed to the frequency domain using a Short Time

Fourier Transform (STFT). The squared magnitude of the STFT is computed and then multiplied by the Mel-frequency weighting functions and summed to obtain the Mel spectrum, whose components represent the energy in each of the Mel filters. To approximate human auditory processing more closely, the natural logarithm of each of the elements in the Mel-spectral vector is then computed, producing the log-Mel spectrum of the frame. In the final step the result is converted to Mel-frequency cepstra via a discrete cosine transform (DCT) to reduce the spatial correlation in the vectors. The MFCC feature extraction process is shown in Fig 2.4. The output is then truncated to the number of required observation vector elements. The 0th cepstral coefficient indicates the power of the speech signal. Therefore each input utterance is transformed into a sequence of acoustic vectors. By using the cepstral coefficients that are generally decorrelated, we obtain HMMs with diagonal covariance matrices.

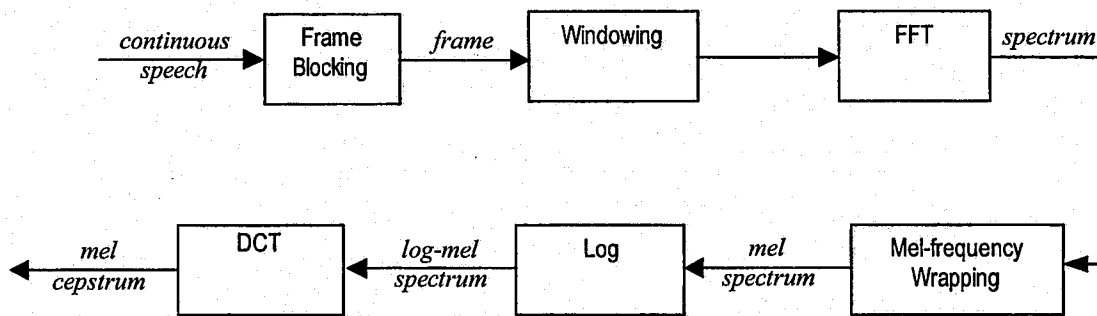


Figure 2.4 Block diagram of the MFCC processor

2.2.1.3 Additional Post-Processing

For systems that operate on variable communication channels such as public telephone networks, the convolutional noise is represented as an additive component in the cepstral domain. A simple method called Cepstral Mean Normalization (CMN) can reduce the effect of channel filtering greatly. The means of the features are estimated over a section of speech. Each feature has this mean subtracted away thereby removing any linear, time invariant static noise from the features [HTK99].

$$\hat{\mathbf{c}}_t = \mathbf{c}_t - \frac{1}{T} \sum_{i=0}^{T-1} \mathbf{c}_i \quad (2.5)$$

T is the total number of occurrences for a given feature in one utterance. After the normalized cepstral coefficients have been obtained, in the final step of a typical speech processing front-end, first-order (delta) and second-order (delta-delta) regression coefficients are appended to the observation vectors [HTK99]. The first and second derivatives (with respect to time) of the cepstral coefficients give an indication of the rates at which the cepstral coefficients change. It has been found that augmenting the cepstral coefficients with derivatives often improves the accuracy of speech recognition and speaker recognition systems, because these coefficients capture some of the correlation between feature vectors. The delta coefficients are given by [HTK99]

$$d_t = \frac{\sum_{\theta=-g}^g \theta (c_{t+\theta} - c_{t-\theta})}{2 \sum_{\theta=-g}^g \theta^2} \quad (2.6)$$

where d_t is a delta coefficient at time t computed in terms of the corresponding static coefficients $c_{t-\theta}$ to $c_{t+\theta}$, and g is the regression window size. The delta-delta regression coefficients (referred to as acceleration coefficients) are obtained by applying the same equation to the delta coefficients. Thus, a typical speech processing front-end produces a sequence of observation vectors that include the basic static parameters, the delta coefficients and the acceleration coefficients.

2.2.2 Acoustic Models

In speech recognition, it is assumed that speech is represented as a sequence of observation vectors derived from a probabilistic function of a first-order Markov chain. A brief summary of the fundamentals of HMMs is given next. Maximum likelihood (ML) parameter optimization for HMMs using the Baum-Welch algorithm [Bau70] is then addressed. Comprehensive descriptions of HMM theory and its application to speech modeling can be found in [Rab89][Rab93][Del93] and their references.

2.2.2.1 HMM Model

A speech unit modeled by a Markov chain is shown in Figure 2.6. The diagram, adopted from the Hidden Markov Model toolkit (HTK) [HTK99], has non-emitting entry and exit states, and three emitting states. Hence, the total number of states is $N = 6$. States are connected by a probabilistic ‘transitions’ matrix, $A = \{a_{ij}\}$, that capture the durational structure. For each state there is a transition probability which is formed by equations (2.7),(2.8).

$$P(q_t = j \mid q_{t-1} = i) = a_{ij} \quad 1 < i, j < N \quad (2.7)$$

$$\sum_{j=1}^N a_{ij} = 1 \quad (2.8)$$

where q_t is the actual state at time t and a_{ij} is the transition probability between state i and j . This matrix does not have to be full, whereby all transitions are permitted. As shown in Fig 2.5, the transition is normally constrained to be left-to-right. The non-emitting states allow expansion to composite models built from a number of individual HMMs for continuous speech recognition. Self-transitions from the non-emitting states are not allowed.

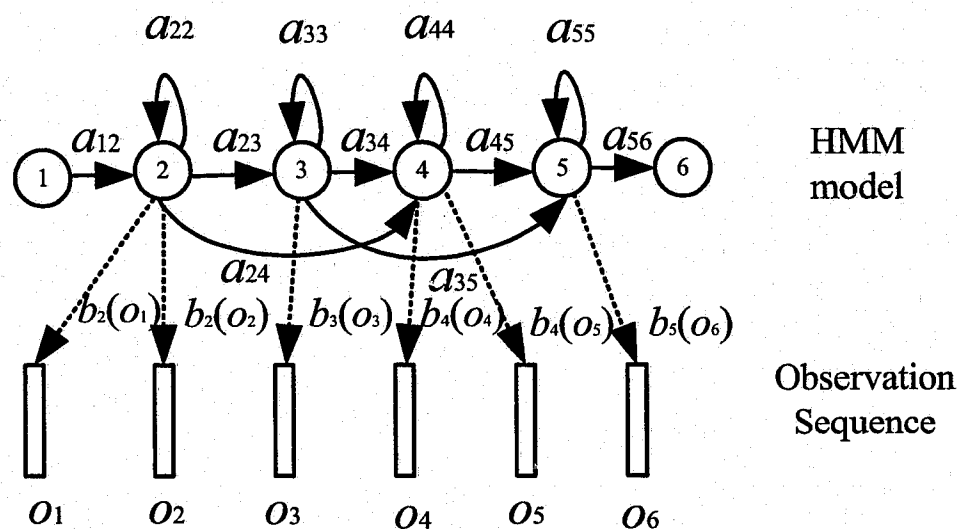


Fig 2.5 Hidden Markov Model generating observation vectors [HTK99]

The observation symbol probability distribution, $B = \{b_j(\mathbf{o}_t)\}$, in which

$$b_j(\mathbf{o}_t) = P(\mathbf{o}_t | q_t = j), \quad 1 \leq t \leq T \quad (2.9)$$

defines the symbol distribution in state $j, j=1,2 \dots N$. It may assume many different forms. Also we have the initial distribution for each state, where the initial state distribution is given as:

$$\pi_i = P(q_1 = i) \quad 1 \leq i \leq N \quad (2.10)$$

It can be seen from the above discussion that a typical HMM requires three sets of probability measures A, B and π . For convenience, we use the compact notation $\lambda = (A, B, \pi)$ to indicate the complete parameter set of the model. In order to best describe how a given observation sequence $O = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_T\}$ comes about, we need to choose $\lambda = (A, B, \pi)$ such that its likelihood, $P(O | \lambda)$, is locally maximized. To train an HMM, it is necessary to be able to calculate the probability of being in state i at time t

$$\gamma_t(i) = P(q_t = i | O) \quad (2.11)$$

It is also necessary to calculate the probability of state j succeeding state i

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j | O) \quad (2.12)$$

These two variables can be calculated using the forward-backward algorithm described next. Given the model λ , the forward variable $\alpha_t(i)$ is defined as

$$\alpha_t(i) = P(\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_t, q_t = i | \lambda) \quad (2.13)$$

Here is the procedure to compute this variable inductively:

- Initialization

$$\alpha_1(i) = \pi_i b_i(\mathbf{o}_1) \quad 1 \leq i \leq N \quad (2.14)$$

- Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(\mathbf{o}_{t+1}) \quad 1 \leq t \leq T-1 \quad (2.15)$$

- Termination

$$P(O | \lambda) = \sum_{i=1}^N \alpha_T(i) \quad (2.16)$$

In a similar manner, we can define the backward variable $\beta_t(i)$ as follows:

$$\beta_t(i) = P(\mathbf{o}_{t+1} \mathbf{o}_{t+2} \dots \mathbf{o}_T \mid q_t = i, \lambda) \quad (2.17)$$

This variable is inductively solved as:

- Initialization

$$\beta_T(i) = 1 \quad 1 \leq i \leq N \quad (2.18)$$

- Induction

$$\beta_t(i) = \sum_{j=1}^N b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j) a_{ij} \quad t = T-1, T-2, \dots, 1 \quad (2.19)$$

Given the forward and backward variables the discrete state posterior probability, $\gamma_t(i)$, is given by:

$$\gamma_t(i) = P(q_t = i \mid O) = \frac{\alpha_t(i) \beta_t(i)}{\sum_{i=1}^N \alpha_t(i) \beta_t(i)} \quad (2.20).$$

For re-estimation of the discrete state transition probability, the posterior probability of being in state i at time $t-1$ and in state j at time t is needed. $\xi_t(i, j)$ is given by:

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j \mid O, \lambda) = \frac{\alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^N \sum_{j=1}^N \alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)} \quad (2.21).$$

The HMM parameters $\lambda = (A, B, \pi)$ can then be trained using an expectation maximization (EM) algorithm [Dem77]. The Baum-Welch [Bau70] algorithm is an instance of the EM algorithm. The basic idea of the EM algorithm is, beginning with an initial model λ , to estimate a new model $\bar{\lambda}$, such that $P(O \mid \bar{\lambda}) > P(O \mid \lambda)$. The new model then becomes the initial model for the next iteration as the process is repeated until some convergence criterion is reached.

The joint likelihood of an observation sequence $O = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_T\}$ and hidden state sequence $\mathbf{q} = \{q_1, q_2, \dots, q_T\}$ can be written as

$$P_{\bar{\lambda}}(O, \mathbf{q}) = P_{\bar{\lambda}}(q_1) \prod_{t=2}^T P_{\bar{\lambda}}(q_t \mid q_{t-1}) \prod_{t=1}^T P_{\bar{\lambda}}(\mathbf{o}_t \mid q_t). \quad (2.22).$$

The joint log likelihood may be written as

$$\log P_{\bar{\lambda}}(O, \mathbf{q}) = \sum_{t=1}^T \log P_{\bar{\lambda}}(\mathbf{o}_t | q_t) + \sum_{t=2}^T \log P_{\bar{\lambda}}(q_t | q_{t-1}) + \log P_{\bar{\lambda}}(q_1) \quad (2.23).$$

The EM algorithm requires us to maximize, $Q(\lambda, \bar{\lambda})$, where $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$ is defined as the reestimated model. Hence,

$$\begin{aligned} Q(\lambda, \bar{\lambda}) &= \sum_{\mathbf{q}} P_{\lambda}(\mathbf{q} | O) \log P_{\bar{\lambda}}(O, \mathbf{q}) \\ &= \sum_{\mathbf{q}} P_{\lambda}(\mathbf{q} | O) \left(\log P_{\bar{\lambda}}(q_1) + \sum_{t=2}^T \log P_{\bar{\lambda}}(q_t | q_{t-1}) + \sum_{t=1}^T \log P_{\bar{\lambda}}(\mathbf{o}_t | q_t) \right) \end{aligned} \quad (2.24)$$

where $\sum_{\mathbf{q}}$ denotes a sum over all possible hidden state sequences. The above equation

consists of three terms; the first for the initial probability, π_i , the second for the state transition probability parameters, A , and the third for the observation model parameters B . It then can be rewritten as

$$Q(\lambda, \bar{\lambda}) = Q(\pi, \bar{\pi}) + Q(A, \bar{A}) + Q(B, \bar{B}) \quad (2.25).$$

The three terms in equation (2.25) can be maximized separately giving rise to parameter update equations for each different part of the model.

-Initial state probability

Using the notation introduced earlier, an update formula for the initial state distribution parameters can be found by maximizing

$$Q(\pi, \bar{\pi}) = \sum_i \gamma_1(i) \log \bar{\pi}_i \quad (2.26)$$

which gives the solution

$$\bar{\pi}_i = \gamma_1(i) \quad (2.27).$$

In the E-step, $\gamma_1(i)$ values are calculated using λ values and in the M-step, the new $\bar{\pi}_i$ values are calculated using the above update formula.

-State transition probabilities

Using notion (2.21), the second term in the equation (2.24) can be re-arranged as

$$Q(A, \bar{A}) = \sum_i \sum_j \left(\sum_t \xi_t(i, j) \right) \log(a_{ij}) \quad (2.28).$$

Maximizing this gives the update formula for the state transition matrix A , having elements a_{ij} as

$$\bar{a}_{ij} = \frac{\sum_i \xi_i(i, j)}{\sum_i \gamma_i(i)} \quad (2.29).$$

-Observation model

An update formula for the observation model parameters can be found by maximizing

$$Q(B, \bar{B}) = \sum_i \sum_{t=1}^T P_{\lambda}(q_t = i | O) \log P_{\lambda}(\mathbf{o}_t | q_t) \quad (2.30)$$

A typical choice is a multivariate Gaussian distribution and is formulated as:

$$b_j(\mathbf{o}_t) = \sum_{m=1}^M C_{jm} N(\mathbf{o}_t, \mu_{jm}, U_{jm}) \quad 1 \leq j \leq N \quad (2.31)$$

where $\mathbf{o}_t$ is the vector being modeled, C_{jm} is a mixture coefficient for the m^{th} mixture in state j and N is a Gaussian distribution with mean μ_{jm} and covariance matrix U_{jm} for the m^{th} mixture component in state j . $\gamma_t(j, n)$ is the probability of being in state j at time t with the n th mixture component accounting for $\mathbf{o}_t$, i.e.,

$$\gamma_t(j, n) = \frac{\alpha_t(j) \beta_t(j)}{\sum_{i=1}^N \alpha_t(i) \beta_t(i)} \left[\frac{C_{jn} N(\mathbf{o}_t, \mu_{jn}, U_{jn})}{\sum_{m=1}^M C_{jm} N(\mathbf{o}_t, \mu_{jm}, U_{jm})} \right] \quad (2.32).$$

The re-estimation procedure is as follows:

$$\begin{aligned} \bar{\mu}_{jn} &= \frac{\sum_{t=1}^T \gamma_t(j, n) \mathbf{o}_t}{\sum_{t=1}^T \gamma_t(j, n)} \\ \bar{U}_{jn} &= \frac{\sum_{t=1}^T \gamma_t(j, n) (\mathbf{o}_t - \mu_{jn})(\mathbf{o}_t - \mu_{jn})'}{\sum_{t=1}^T \gamma_t(j, n)} \\ \bar{C}_{jn} &= \frac{\sum_{t=1}^T \gamma_t(j, n)}{\sum_{t=1}^T \sum_{n=1}^M \gamma_t(j, n)} \end{aligned} \quad (2.33).$$

This procedure will be repeated and we can improve the probability of O being observed from the model until some limiting point is reached.

The above formulations are for single discrete observation sequences. For appropriate training of the model, we need to feed multi-observation sequences to the re-estimation procedure. The modification for the re-estimation procedure is straightforward: suppose we have the set of K observations of sequences, we need to maximize the product of each probability of individual observation:

$$P(\underline{O} | \lambda) = \prod_{k=1}^K P(O^{(k)} | \lambda) = \prod_{k=1}^K P_k \quad \text{with} \quad \underline{O} = [O^{(1)} O^{(2)} \dots O^{(K)}] \quad (2.34).$$

All of the parameters used for intermediate computations including forward variable and backward variables will be computed individually for each observation: $\alpha_i^{(k)}(i), \beta_i^{(k)}(i), \gamma_i^{(k)}(i)$. The final re-estimation formulation for ergodic, continuous observation HMM with multi-observation training can be shown as:

$$\begin{aligned} \bar{\pi}_i &= \frac{\sum_{k=1}^K \frac{1}{P_k} \gamma_1^{(k)}(i)}{\sum_{k=1}^K \frac{1}{P_k}} \\ \bar{a}_{ij} &= \frac{\sum_{k=1}^K \frac{1}{P_k} \sum_{t=1}^T \alpha_t^{(k)}(i) a_{ij} b_j(\mathbf{o}_{t+1}^{(k)}) \beta_{t+1}^{(k)}(j)}{\sum_{k=1}^K \frac{1}{P_k} \sum_{t=1}^T \alpha_t^{(k)}(i) \beta_t^{(k)}(j)} \\ \bar{\mu}_{jn} &= \frac{\sum_{k=1}^K \frac{1}{P_k} \sum_{t=1}^T \gamma_t^{(k)}(j, n) \mathbf{o}_t}{\sum_{k=1}^K \frac{1}{P_k} \sum_{t=1}^T \gamma_t^{(k)}(j, n)} \\ \bar{U}_{jn} &= \frac{\sum_{k=1}^K \frac{1}{P_k} \sum_{t=1}^T \gamma_t^{(k)}(j, n) (\mathbf{o}_t^{(k)} - \bar{\mu}_{jn}) (\mathbf{o}_t^{(k)} - \bar{\mu}_{jn})'}{\sum_{k=1}^K \frac{1}{P_k} \sum_{t=1}^T \gamma_t^{(k)}(j, n)} \\ \bar{C}_{jn} &= \frac{\sum_{k=1}^K \frac{1}{P_k} \sum_{t=1}^T \gamma_t^{(k)}(j, n)}{\sum_{k=1}^K \frac{1}{P_k} \sum_{t=1}^T \sum_{n=1}^M \gamma_t^{(k)}(j, n)} \end{aligned} \quad (2.35)$$

The procedure is iterated until the observation likelihood converges. The final result of the re-estimation procedure is a maximum likelihood estimation of the HMM.

2.2.2.2 GMM Model

The class of Gaussian mixture models also falls into the HMM category, since a Gaussian mixture model is equivalent to a one state HMM and no Markovian constraint placed on the evolution of mixtures as in the case of a multi-state HMM. The maximum likelihood (ML) estimation used to find the GMM parameters that maximize the likelihood of the training data will be summarized as follows [Rey92].

Assume the probability density function of x is to be represented by a weighted sum of M component densities as:

$$p(x) = \sum_{i=1}^M p_i b_i(x) \quad (2.36)$$

where x is a D -dimensional vector, $b_i(x), i=1, \dots, M$ are the component densities and $p_i, i=1, \dots, M$ are the mixture weights (i.e. *a priori* probability). Each component density is a multivariate Gaussian function of the form

$$b_i(x) = \frac{1}{(2\pi)^{D/2} |\Sigma_i|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i) \right\} \quad (2.37)$$

with mean vector μ_i and covariance matrix Σ_i . The mixture weights satisfy the constraint that $\sum_{i=1}^M p_i = 1$. The complete Gaussian mixture density parameters are collectively represented by:

$$\lambda = \{p_i, \mu_i, \Sigma_i\} \quad (2.38).$$

With observations $x = \{x_1, x_2, \dots, x_T\}$, the following estimates are evaluated on each EM iteration. These estimates guarantee a monotonic increase in the model's likelihood value:

$$\text{Mixture weights: } \bar{p}_i = \frac{1}{T} \sum_{t=1}^T p(i | x_t, \lambda) \quad (2.39)$$

$$\text{Means: } \bar{\mu}_i = \frac{\sum_{t=1}^T P(i | x_t, \lambda) x_t}{\sum_{t=1}^T P(i | x_t, \lambda)} \quad (2.40)$$

$$\text{Variance: } \bar{\Sigma}_i = \frac{\sum_{t=1}^T P(i | x_t, \lambda) (x_t - \mu_i)(x_t - \mu_i)'}{\sum_{t=1}^T P(i | x_t, \lambda)} \quad (2.41).$$

Then the *a posteriori* probability for an acoustic class is given by

$$P(i | x_t, \lambda) = \frac{p_i b_i(x_t)}{\sum_{k=1}^M p_k b_k(x_t)} \quad (2.42)$$

2.2.3 Search Algorithm and Scoring

The standard speech recognition solution involves finding the most likely sequence of HMMs. For continuous speech recognition applied to large-vocabulary tasks, the search algorithm needs to apply all available acoustic and linguistic knowledge to maximize the recognition accuracy. The recognition network consists of a set of nodes illustrated by Fig 2.7 [HTK99]. It can be viewed at three different levels: word, model (i.e. phoneme) and state. Within-HMM transitions are determined from the HMM parameters, between-model transitions are constant and word-level transitions are determined by the language model likelihood attached to the word level networks. The search algorithm for a speech recognizer, restricted by the lexicon and language models, concatenates HMMs corresponding to the phoneme sequences to form a composite model representing words and sentences.

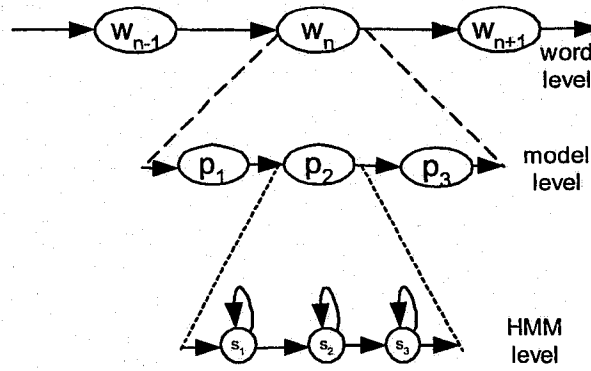


Fig 2.6 Recognition network

There is an algorithm to search for the maximum likelihood HMM state sequence called the *Viterbi algorithm* [Rab93]. To find the single best state sequence $q = q_1 q_2 \dots q_T$ for a given observation sequence $O = \mathbf{o}_1 \mathbf{o}_2 \dots \mathbf{o}_T$, we need to define the quantity

$$\delta_t(i) = \max P(q_1 q_2 \dots q_t = i, \mathbf{o}_1 \mathbf{o}_2 \dots \mathbf{o}_t | \lambda) \quad (2.43)$$

that is, $\delta_t(i)$ is the best score along a single path, at time t , which accounts for the first t observations and ends in state $q_t = i$. We have

$$\delta_{t+1}(j) = [\max_i \delta_t(i) a_{ij}] b_j(\mathbf{o}_{t+1}) \quad (2.44)$$

The complete procedure for finding the best state sequences can be written as follows: (ψ is the variable that keeps track of the argument which maximizes equation 2.43 for each t and j)

Initialization:

$$\begin{aligned} \delta_1(i) &= \pi_i b_i(\mathbf{o}_1) \quad 1 \leq i \leq N \\ \psi_1(i) &= 0 \end{aligned} \quad (2.45)$$

Recursion

$$\begin{aligned} \delta_t(i) &= \max_j [\delta_{t-1}(j) a_{ji}] b_i(\mathbf{o}_t) \quad 2 \leq t \leq T \\ \psi_t(i) &= \arg \max_j [\delta_{t-1}(j) a_{ji}] \quad 1 \leq j \leq N \end{aligned} \quad (2.46)$$

Termination

$$\begin{aligned} P^* &= \max_i [\delta_T(i)] \quad 1 \leq i \leq N \\ q_T^* &= \arg \max_i [\delta_T(i)] \end{aligned} \quad (2.47)$$

Path backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \quad t = T-1, T-2, \dots, 1 \quad (2.48)$$

It is straightforward to extend the Viterbi algorithm for continuous speech recognition. The paths are found using a *token passing algorithm* [You89]. In this algorithm, each state has a token associated with it. A token represents a partial path through the network extending from time 0 through time t . As each token passes through the network it must maintain a history recording its route. Normally, only word sequences are wanted and hence, only transitions out of the word end nodes need to be recorded. The word link record consists of the partial score and the link to the previous word link record is copied from the best exit token, along with the name of the word model where the token came from. A new word link record is generated every time instant by finding the token with the highest score from the exit node of each composite word model. The language model score and the insertion penalty are also stored to give the partial joint score. *Pruning* may be used to keep the number of active tokens from growing too large in large vocabulary continuous speech recognition. However, excessively tight pruning may result in search errors. After the final observation in the utterance has been processed, the final word link record holds the total score of the best hypothesis. The maximum likelihood word sequence is found by following back word link records using the links stored inside the records. For more details see [HTK99].

2.3 Summary

The framework for speech recognition has been briefly described in this chapter. The standard front-ends were also briefly reviewed. The theory of HMMs and their application to acoustic modeling was described. Language models and search algorithms for HMM based speech recognition were also presented.

Chapter 3

Review of Speech Enhancement

In this chapter, some approaches studied in recent years to enhance speech signals degraded by statistically independent additive noise when only one microphone is available will be reviewed. Some of the techniques described produce an enhanced speech waveform. Some other approaches however produce enhanced speech parameters suitable for being input directly to a clean speech recognizer.

3.1 Speech Waveform Enhancement

The observations, $y(t)$ ($t = 0, 1, \dots, N$), are assumed to be the sum of the desired signal $s(t)$ plus noise $d(t)$ as follows:

$$y(t) = s(t) + n(t), \quad t = 0, 1, \dots, N \quad (3.1)$$

$y(t)$ is the noisy speech signal, $s(t)$ is the clean speech signal and $n(t)$ is the noise signal. The speech waveform enhancement techniques described next aim to produce an enhanced speech waveform $\hat{y}(t)$.

3.1.1 Spectral Subtraction

Spectral subtraction [Bol79] is one of the most influential speech enhancement algorithms, favored for its simplicity and performance in suppressing stationary noise. We know that the power spectrum of uncorrelated additive signals is also additive and this holds approximately true for short-term estimates as well. Hence, if the power spectrum of the signal and noise could be obtained or estimated, a speech enhancement system may be constructed.

Using the notations in equation (3.1), the power spectrum of the noisy signal is given by

$$P_y(\omega) = P_s(\omega) + P_n(\omega) \quad (3.2).$$

The classic spectral subtraction technique [Bol79] estimates the speech spectrum as

$$P_s(\omega) = P_y(\omega) - P_n(\omega) \quad (3.3)$$

where $P_s(\omega)$ is the power spectrum estimate of the clean signal, $P_y(\omega)$ is the power spectrum estimate of the corrupted signal, and $P_n(\omega)$ is the smoothed power spectrum estimate of the noise, which is obtained by averaging the power spectrums during non-speech activity. Based on the fact that the amplitude of the short-time speech spectrum is perceptually more important than the phase, the enhanced speech signal is obtained by an inverse Fourier transform from $P_s(\omega)$ and the phase of the noisy signal:

$$\hat{s}(t) = \mathbf{F}^{-1}(\sqrt{|P_s(\omega)|} e^{j\theta_y(\omega)}) \quad (3.4)$$

where $\theta_y(\omega)$ is the phase spectrum of the noisy speech and $\mathbf{F}^{-1}$ denotes the inverse Fourier transform. Many variations of the spectral subtraction technique can be found in the literature [Lim79]. The most general form of spectral subtraction method is given by:

$$P_y^\gamma(\omega) = P_s^\gamma(\omega) + kP_n^\gamma(\omega) \quad (3.5)$$

where the parameters γ and k are chosen based on the compromise between the speech signal distortion and the level of residual noise. Their typical values can be found in [Lim79].

The major problem of the above method is that residual noise remains in the enhanced spectrum because only the average noise spectrum is subtracted. Fluctuations in the actual noise spectrum for each frame mean that spurious spectral components are introduced at the frame rate. This noise, known as ‘musical noise’, is mostly perceived in weak bands and weak speech frames. Musical noise decreases speech quality both for human listeners and recognition systems alike. Another problem is that the speech spectrum estimate in equation (3.3) may be negative. In this case some form of ad hoc solution must be applied which invariably involves a nonlinear operation, such as reflecting the negative spectral component or setting it to zero. A further problem is that the noise statistics must be known or estimated from the non-speech portions of the signals to be enhanced. Since no information about the speech is known, this may be non-trivial.

A related spectral domain method is Wiener filtering. The frequency response of this filter is

$$H(\omega) = \frac{P_s(\omega)}{P_s(\omega) + P_n(\omega)} \quad (3.6)$$

The analysis is performed on a per frame basis, with only the spectral magnitude being estimated and the noisy phase being used when the signal is reconstructed. The speech and noise power spectral density must be known a priori or estimates must be made from the noisy speech.

3.1.2 Kalman Filter Based Methods

Dynamic filtering techniques, such as Kalman filtering, also provide a good estimate of clean speech given noisy speech. The state space for the Kalman filter includes a process state equation which models the dynamics of the speech signal generation process, and an observation state equation which models the noisy signal. When the measurement noise process and the signal are jointly Gaussian, the Kalman filter can yield an minimum mean squared error (MMSE) estimate of the clean signal. The Kalman filter algorithm was first proposed for speech enhancement by Paliwal et al [Pal87]. It is shown in [Pal87] that, if the ideal autoregressive (AR) speech model is used, the Kalman filter performs better than the stationary or the nonstationary Wiener filters in terms of output SNR values. A summary of the standard Kalman filtering algorithm is shown below [Chu99].

A linear system with state-space description is written as the following state-space model:

$$\begin{cases} \mathbf{y}_{k+1} = \mathbf{A}_k \mathbf{y}_k + \mathbf{B}_k \mathbf{u}_k + \mathbf{\Gamma}_k \xi_k \\ \mathbf{w}_k = \mathbf{C}_k \mathbf{y}_k + \mathbf{D}_k \mathbf{u}_k + \eta_k \end{cases} \quad (3.7)$$

where $\mathbf{A}_k, \mathbf{B}_k, \mathbf{\Gamma}_k, \mathbf{C}_k, \mathbf{D}_k$ are $n \times n$, $n \times m$, $n \times p$, $q \times n$, $q \times m$ (known) constant matrices, respectively, with $1 \leq m, p, q \leq n$, $\{\mathbf{u}_k\}$ is a known sequence of m -vectors (called a deterministic input sequence), and $\{\xi_k\}$ and $\{\eta_k\}$ are, respectively, (unknown) system and observation noise sequences, with known statistical information such as mean, variance, and covariance. The estimate of the state vector depends on the statistical information of the noise sequences.

The derivation of the algorithm can be found in [Chu99]. Assume that $\{\xi_k\}$ and $\{\eta_k\}$ are sequences of zero-mean Gaussian white noise such that $Var(\xi_k) = \mathbf{Q}_k$ and $Var(\eta_k) = \mathbf{R}_k$ are positive definite matrices and $E(\xi_k \eta_l) = \mathbf{0}$ for all k and l . The initial state $\mathbf{y}_{-1}$ is also assumed to be independent of $\{\xi_k\}$ and $\{\eta_k\}$ in the sense that $E(\mathbf{y}_{-1} \xi_k^T) = \mathbf{0}$ and $E(\mathbf{y}_{-1} \eta_k^T) = \mathbf{0}$ for all k . The following Kalman filtering algorithm for the linear stochastic system of (3.7) is given by:

$$\begin{cases} \mathbf{P}(-1|-1) = Var(\mathbf{y}_{-1}) \\ \mathbf{P}(k|k-1) = \mathbf{A}_k \mathbf{P}(k|k-1) \mathbf{A}_k^T + \mathbf{\Gamma}_{k-1} \mathbf{Q}_{k-1} \mathbf{\Gamma}_{k-1}^T \\ \mathbf{G}_k = \mathbf{P}(k|k-1) \mathbf{C}_k^T (\mathbf{C}_k \mathbf{P}(k|k-1) \mathbf{C}_k^T + \mathbf{R}_k)^{-1} \\ \mathbf{P}(k|k-1) = (\mathbf{I} - \mathbf{G}_k \mathbf{C}_k) \mathbf{P}(k|k-1) \\ \hat{\mathbf{y}}(-1|-1) = E(\mathbf{y}_{-1}) \\ \hat{\mathbf{y}}(k|k-1) = \mathbf{A}_k \hat{\mathbf{y}}(k-1|k-1) \\ \hat{\mathbf{y}}(k|k) = \hat{\mathbf{y}}(k|k-1) + \hat{\mathbf{G}}_k [\mathbf{w}_k - \mathbf{C}_k \hat{\mathbf{y}}(k|k-1)] \\ k = 0, 1, \dots, N \end{cases} \quad (3.8)$$

where $\mathbf{P}(k|k-1) = E\{[\mathbf{y}_k - \hat{\mathbf{y}}(k|k-1)][\mathbf{y}_k - \hat{\mathbf{y}}(k|k-1)]^T\} = Var[\mathbf{y}_k - \hat{\mathbf{y}}(k|k-1)]$ is the predicted state error vector covariance matrix, and $\mathbf{P}(k|k) = E\{[\mathbf{y}_k - \hat{\mathbf{y}}(k|k)][\mathbf{y}_k - \hat{\mathbf{y}}(k|k)]^T\} = Var[\mathbf{y}_k - \hat{\mathbf{y}}(k|k)]$ is the estimate state error vector covariance matrix. $\mathbf{G}_k$ is called the Kalman gain matrix.

In speech enhancement, the clean speech signal $s(t)$ is modeled as a p -th order AR process as follows:

$$s(t) = \sum_{i=1}^p a_i s(t-i) + u(t) \quad (3.9)$$

where a_i is the i -th AR coefficient, $u(t)$ is the model noise process.

The measured speech signal $y(t)$ is

$$y(t) = s(t) + n(t) \quad (3.10)$$

$n(t)$ is the background noise. $u(t)$ and $n(t)$ are uncorrelated Gaussian white noise processes with means $\bar{u}$ and $\bar{n}$ and variances σ_u^2 and σ_n^2 .

The state space model can then be obtained:

$$\mathbf{x}(t) = \mathbf{F}\mathbf{x}(t-1) + \mathbf{G}u(t) \quad (3.11)$$

$$y(t) = \mathbf{H}\mathbf{x}(t) + n(t) \quad (3.12)$$

where

$$\mathbf{x}(t) = [s(t-p+1) \cdots s(t)]^T \quad (3.13)$$

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ a_p & a_{p-1} & a_{p-2} & \cdots & a_1 \end{bmatrix} \quad (3.14)$$

$$\mathbf{H} = \mathbf{G}^T = [0 \ 0 \ \cdots \ 0 \ 1] \quad (3.15)$$

where p is the order of the AR process, $\mathbf{F}$ is a $p \times p$ transition matrix, $\mathbf{G}$ and $\mathbf{H}$ are respectively the $p \times 1$ input vector and the $1 \times p$ observation row vector. A Kalman filtering algorithm for speech enhancement is summarized in [Gab01] as follows:

$$e(t) = y(t) - \mathbf{H}\hat{\mathbf{x}}(t|t-1) - \bar{n} \quad (3.16)$$

$$\mathbf{K}(t) = \mathbf{P}(t|t-1)\mathbf{H} \times (\mathbf{H}\mathbf{P}(t|t-1)\mathbf{H}^T + \sigma_n^2)^{-1} \quad (3.17)$$

$$\hat{\mathbf{x}}(t|t) = \hat{\mathbf{x}}(t|t-1) + \mathbf{K}(t)e(t) \quad (3.18)$$

$$\mathbf{P}(t|t) = (\mathbf{I} - \mathbf{K}(t)\mathbf{H})\mathbf{P}(t|t-1) \quad (3.19)$$

$$\hat{\mathbf{x}}(t+1|t) = \mathbf{F}\hat{\mathbf{x}}(t|t) + \mathbf{G}\bar{u} \quad (3.20)$$

$$\mathbf{P}(t+1|t) = \mathbf{F}\mathbf{P}(t|t)\mathbf{F}^T + \mathbf{G}\mathbf{G}^T\sigma_u^2 \quad (3.21)$$

As with any model-based enhancement, the parameter estimation problem remains a big issue for AR model based Kalman filters when only noisy speech is available. Gannot et al [Gan98] have proposed the use of the EM algorithm to iteratively estimate the parameters of noise and speech. Gibson et al [Gib91] have used a method which is a simplified version of the EM algorithm based on the maximum likelihood argument. However, noise variance was estimated during the silent period, which implies the use of voice activity detector. In [Gab01], the sequential estimators are derived for sub-optimal adaptive estimation of the unknown *a priori* driving process and additive noise statistics, simultaneously with the system state estimation, by reformulating and adapting the classical approach used for control applications. The algorithm provides improved state

estimates at little computational expense. A distinct advantage of the proposed algorithm is that a speech/noise detector is not required.

An alternative approach to AR model based Kalman filters is the nonlinear neural network model based on the Extended Kalman Filter [Wan98]. The underlying assumption is that a neural autoregressive model can better accommodate the nonlinear behaviors of speech than the linear autoregressive model. It was found that since one can only estimate the neural network model of speech from the noisy signal, then when the input SNR is high, the estimation is more accurate. However for low SNR, the estimation of the nonlinear speech model may deviate far from the true value. Thus the accuracy of the nonlinear speech model is sensitive to the input SNR [Man05].

3.1.3 Noise Masking

Noise masking is a psychological phenomenon of reducing the perceptibility of a signal in the presence of noise. That is, people cannot detect an acoustic stimulus whose level is lower than the masking threshold generated by another stimuli. Masking effects can be classified as time-domain (temporal) masking [Thi00] and frequency-domain (simultaneous) masking [Joh88]. Temporal masking effects can be further classified as forward masking and backward masking, respectively. In frequency masking, one sound cannot be perceived if another sound close in frequency has a high enough level. In temporal masking, a sound can mask another sound preceding it or following it, if there is a small time offset between the masking sound and the masked sound. The masking thresholds are energy dependent as well.

In [Vir99], Virag proposed to enhance speech in additive background noise at very low signal-to-noise ratios (SNR's) ($<5\text{dB}$) based on masking properties of the human auditory system. The proposed enhancement scheme is composed of the following main steps:

- Spectral decomposition;
- Speech/noise detection and estimation of noise during speech pauses;
- Calculation of the noise masking threshold;

- Adaptation in time and frequency of the subtraction parameters based on the noise masking threshold;
- Calculation of the enhanced speech spectral magnitude via parametric subtraction with adapted parameters;
- Inverse transform.

This approach exploits the masking properties of the human auditory system to overcome the problem of “musical residual noise” introduced by conventional subtractive-type enhancement methods.

In [Man04][Man05], several Kalman filter based speech enhancement approaches combined with masking properties of human auditory systems have been proposed where both time-domain and frequency domain masking properties are taken into account. The goal of the masking is to make the estimate state error energy smaller than the threshold.

The performance advantage brought about by masking may be two-fold. First, the environment-dependent residuals from the speech enhancement front-end are masked. Second, low-level speech events are masked so that the recognizer does not learn features that may not be distinguishable in noisy conditions. It seems that speech distortion and residual noise are two quantities that cannot be minimized simultaneously. The compromise between the speech signal distortion and the level of residual noise needs to be accommodated in speech enhancement.

3.1.4 Statistical Model Based Approaches

The class of statistical model based techniques estimate the clean speech signal from the noisy signals using statistical models of the speech and noise. Given a distortion measure and the joint statistics of the speech and noise, enhancement is performed using the estimator which minimizes the expected value of the distortion measure between the clean and estimated signals [Eph92].

3.1.4.1 Spectral Amplitude Model Based Methods

One of the well-known statistical model based approaches is the one proposed by Ephraim and Malah in 1984 [Eph84]. A minimum mean-squares error (MMSE) short-time spectral amplitude (STSA) estimator is derived by assuming *a priori* probability distributions of the speech and noise Fourier expansion coefficients. In the paper, the Fourier expansion coefficients of each process are modeled as statistically independent Gaussian random variables with zero mean. Strictly, this assumption only holds as the analysis frame length approaches infinity. However the authors justify its use on the grounds that it leads to simple estimators and good enhancement. The performance of the MMSE STSA method depends on the good estimate of the *a priori* SNR which is still an open issue in current research. It is demonstrated in [Eph84] that by using different STSA estimators for the *a priori* SNR, different enhancement methods are obtained. For example, using the “power spectral subtraction” method for estimating the *a priori* SNR results in an MMSE STSA estimator that is nearly equivalent to spectral subtraction estimator described in 3.1.1. The later work by Ephraim in 1985 [Eph85] looks at a MMSE log spectral estimator on the grounds that distortion measures which are based on the mean square error of the log spectrum are perceptually more suitable for speech processing. This results in a superior but computationally more complex enhancement scheme.

3.1.4.2 Hidden Markov Model Based Methods

A popular choice of statistical models for speech signals is HMMs. They are able to model the time-varying characteristics of the vocal tract. The HMM models described in Chapter 2 have dominated the field of speech processing in recent years not only for speech recognition but also for speech enhancement used before speech recognition. HMMs with mixtures of autoregressive (AR) output probability distributions are used to model the time-domain waveforms. Each mixture component of each HMM state represents observations that have similar autoregressive parameters, or equivalently similar spectral parameters. Those state-dependent autoregressive parameters are used to construct Wiener filters to perform enhancement similar to the Lim and Oppenheim

technique [Lim78]. The difference here though is that the speech autoregressive parameters are estimated from clean speech training data rather than from the noisy signals as in [Lim78].

Ephraim has developed various enhancement schemes based on autoregressive HMMs to enhance noisy speech [Eph89][Eph90]. In [Eph89], the expectation maximization (EM) algorithm [Dem77] is used to find the optimal MAP estimator of the clean speech. The algorithm works as follows: an estimate of the clean speech is calculated by filtering the noisy speech through a weighted sum of Wiener filters. This estimation is then used to find a probability sequence to supply a new sequence of Wiener filters and another estimate of the clean speech. This iterative process continues until a preset convergence criterion is reached. Experiments show that the algorithm is very dependent on the accuracy of choosing the correct state and mixture of each frame. For low SNRs, this can be difficult since a clean model is used to calculate the probability of each mixture of each state given initially the noisy observation.

Ephraim extended the work in [Eph90] by using the minimum mean square error (MMSE) method that gives better enhancement results compared with that of MAP estimation. In this later work, the problem of estimating the clean state and mixture component corresponding to each noisy observation frame is addressed and solved using a compensated HMM formed by combining speech and noise statistics. This alleviates the need for an iterative scheme and provides a better state-frame alignment and hence a better selection of filters. The MMSE estimator is given by the weighted sum of Wiener filters for each combination of speech and noise states, weighted by the probability of that combination. The MMSE based HMM is further modified by Sameti [Sam98] by incorporating multiple state mixture based models for speech and noise. In the improved MMSE enhancement method, a multiple state and mixture noise model was employed to accommodate non-stationarity in noise.

Lee et al [Lee96] have developed enhancement schemes based on variations to the classic AR-HMM. In [Lee96] and [Kim00] schemes based on the hidden filter HMM (HF-HMM)

are studied. This model is a special case of the autoregressive HMM with a frame size of one sample. It is assumed that such HMMs are better able to model quickly varying parts of speech signals since information is not lost at the frame boundaries. These schemes are state-dependent model switching methods and switching Kalman filters are used instead of Wiener filters.

One drawback with the HMM-based methods is the need to choose a sufficient number of states to model all the different regions of speech and noise statistics, which makes the implementation fairly complex in comparison to other methods.

3.2 Speech Feature Enhancement

The work described in section 3.1 applied the enhancement techniques mainly in the linear domain, either in the time-sample or frame-bound (linear) spectral domain, for the additive noise case. Although the adaptive enhancement schemes of section 3.1 pose little mathematical difficulties for additive noise, the cepstral or log-spectral domain is more desirable if the purpose of speech enhancement is for robust speech recognition, since this is the domain as close as possible to the back end of the recognizer [Den04b]. To get enhanced feature vectors that are as close to those of the undistorted, clean speech signal component as possible, one of the difficulties is that the distortion caused by additive ambient noises is highly non-linear in the Mel-scaled log-spectral or cepstral domain.

We are given a speech waveform which is corrupted by additive noise in the time domain as:

$$y(t) = x(t) + n(t) \quad (3.22)$$

Let us define $Y(\omega)$, $X(\omega)$ and $N(\omega)$ as the power spectral densities (PSD) of the signal $y(t)$, $x(t)$ and $n(t)$ respectively. Since the signal and noise are assumed statistically independent with each other, the following relationship holds for every frequency bank

$$Y(\omega) = X(\omega) + N(\omega) \quad (3.23)$$

If converted into the log-spectral domain (on a linear or Mel-scale), after some algebraic manipulation, the above relation becomes a nonlinear function as follows [Mor96]:

$$\mathbf{y}_t = f(\mathbf{x}_t, \mathbf{n}_t) = \mathbf{x}_t + \log(1 + \exp(\mathbf{n}_t - \mathbf{x}_t)) \quad (3.24)$$

in which $\mathbf{y}_t$, $\mathbf{x}_t$ and $\mathbf{n}_t$ represent the log spectrum of noisy speech, the hypothetical clean speech and noise at the t^{th} frame respectively. If equation (3.24) is converted into the cepstral domain, the observation model relates the cepstrum of the noisy speech $\mathbf{y}_t$ to the cepstrum of the clean speech $\mathbf{x}_t$ and to the cepstrum of the noise $\mathbf{n}_t$ at the t^{th} frame as [Ace93]:

$$\mathbf{y}_t = \mathbf{x}_t + \mathbf{C} \log(1 + \exp(\mathbf{C}^{-1}(\mathbf{n}_t - \mathbf{x}_t))) \quad (3.25)$$

where $\mathbf{C}$ is the DCT matrix and $\mathbf{C}^{-1}$ is the inverse DCT matrix.

To find the clean speech from the noisy observation, the enhancement systems need a model for speech, and often a model for noise as well. Mostly, the clean speech distribution is modeled by a Gaussian mixture model. With this speech model, the noise is considered as the state variable to be estimated, whose observation is corrupted by speech. Clean speech estimation is obtained by removing the estimated noise sequence from the noisy signal sequence.

Here we divide the speech feature enhancement approaches into two general categories, statistical-model-based approaches and filter-model-based approaches, and we review them respectively in the next sections.

3.2.1 Some Statistical-Model-Based Approaches

In this section, we will review some well-known statistical-model-based speech feature enhancement systems, including Codeword Dependent Cepstral Normalization (CDCN) proposed by Acero in [Ace93], Vector Taylor series (VTS) expansion proposed by Moreno in [Mor96], Stereo-based Piecewise Linear Compensation (SPLICE) proposed by Deng et al in [Den01] and some extensions and variations based on these approaches proposed in [Den04b].

3.2.1.1 CDCN

Codeword Dependent Cepstral Normalization (CDCN) [Ace93] is a compensation method that assumes an analytical model of the environmental effects on speech. In the standard form of CDCN, a mixture of Gaussians model is trained for clean speech

$$p(\mathbf{x}) = \sum_k p_k N(\boldsymbol{\mu}_{x,k}, \boldsymbol{\Sigma}_{x,k}) \quad (3.26)$$

Given a noise n , the noisy cepstra can be modeled using the following probability density function:

$$p(\mathbf{y}) = \sum_k p_k N(\mathbf{y}; \boldsymbol{\mu}_{x,k} + \mathbf{r}_k, \boldsymbol{\Sigma}_{x,k}) \quad (3.27)$$

$$\hat{\mathbf{r}}_k = \mathbf{C} \log(1 + \exp(\mathbf{C}^{-1}(\hat{\boldsymbol{\mu}}_n - \boldsymbol{\mu}_{x,k}))) \quad (3.28)$$

where $\mathbf{C}$ is the DCT matrix and $\mathbf{C}^{-1}$ is the inverse DCT matrix. $\hat{\boldsymbol{\mu}}_n$ is an estimated mean of the additive noise. It is seen from above that the CDCN algorithm does not model the effects of the environment on the variance of speech distributions, which limits its performance at low SNRs. $\hat{\boldsymbol{\mu}}_n$ can be estimated iteratively using the maximum likelihood criterion as

$$\hat{\boldsymbol{\mu}}_n = \arg \max_{\boldsymbol{\mu}_n} p(\mathbf{y} | \boldsymbol{\mu}_n, \mathbf{x}) \quad (3.29)$$

Finally, the MMSE estimate of the clean speech is given by

$$\hat{\mathbf{x}}_{t,MMSE} = \mathbf{y}_t - \sum_k p(k | \mathbf{y}_t) \hat{\mathbf{r}}_k \quad (3.30)$$

As the computational load of estimating the noise online is high, various simpler approximations are normally used. The simplest approximation is to make the correct factor a function of the instantaneous SNR,

$$\mathbf{x}_t = \mathbf{y}_t - w(\text{SNR}_t) \quad t = 0, 1, \dots, L-1 \quad (3.31)$$

The values of $w(\text{SNR}_t)$ are learned from a stereo database (i.e., a corpus in which there is a known frame-to-frame correspondence between the clean and noisy collected version of the data). Thus the feature space will be split into L regions and the compensation vector is a function of a codebook entry. $w(\text{SNR}_t)$ must be learned for each new noise condition.

3.2.1.2 VTS-based Speech Feature Compensation

While CDCN uses a 0th-order Taylor series approximation, a later approach VTS approximates the function f in equation (3.24) with a first-order vector Taylor series. VTS were originally proposed by Moreno [Mor96] to approximate the nonlinear environmental function. If we take a 1st-order VTS approximation for equation (3.24), it becomes

$$\mathbf{y} = \mathbf{x} + (\nabla_x f)' \mathbf{x} + (\nabla_n f)' \mathbf{n} + g(\mathbf{n}_0, \mathbf{x}_0) \quad (3.32)$$

$$g(\mathbf{n}_0, \mathbf{x}_0) = f(\mathbf{n}_0, \mathbf{x}_0) - (\nabla_x f)' \mathbf{x}_0 - (\nabla_n f)' \mathbf{n}_0 \quad (3.33)$$

where

$$f = \log(\mathbf{I} + \exp(\mathbf{n} - \mathbf{x})) \quad (3.34)$$

It is assumed that the probability density function (PDF) of the log-spectra of the clean speech signal can be represented by a summation of multivariate Gaussian distributions:

$$p(\mathbf{x}) = \sum_k p_k N(\boldsymbol{\mu}_{x,k}, \boldsymbol{\Sigma}_{x,k}) \quad (3.35)$$

where $\boldsymbol{\mu}_{x,k}$ is the mean vector of the k -th Gaussian mixture of the clean speech vector $\mathbf{x}$ and $\boldsymbol{\Sigma}_{x,k}$ is the corresponding covariance matrix of the clean speech and assumed to be diagonal. Also it is assumed that the statistics of the noise can be well represented by a single Gaussian $N(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$. The mean and the covariance of the noisy speech vector $\mathbf{y}$ becomes

$$\boldsymbol{\mu}_{y,k} = (\mathbf{I} + \nabla_x f)' \boldsymbol{\mu}_{x,k} + (\nabla_n f)' \boldsymbol{\mu}_n + g(\mathbf{n}_0, \mathbf{x}_0) \quad (3.36)$$

$$\boldsymbol{\Sigma}_{y,k} \approx (\mathbf{I} + \nabla_x f)' \boldsymbol{\Sigma}_{x,k} (\mathbf{I} + \nabla_x f) + (\nabla_n f)' \boldsymbol{\Sigma}_n (\nabla_n f) \quad (3.37)$$

Again the Expectation Maximization (EM) algorithm is used to re-estimate the unknown parameters, including the noise mean and variance. The VTS algorithm proceeds as follows [Kim98d]:

1. Obtain initial estimates of $\{\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n\}$
2. Estimate the parameters of the distribution of $\mathbf{y}$, $\boldsymbol{\mu}_{y,k}$ and $\boldsymbol{\Sigma}_{y,k}$ with equation (3.36) and (3.37)

3. Perform a single iteration of the EM algorithm to re-estimate the μ_n

$$\hat{\mu}_n = \frac{\sum_t \sum_k p(k | y_t) \tilde{\mu}_n(y_t, k)}{\sum_t \sum_k p(k | y_t)} \quad (3.38)$$

$$\hat{\Sigma}_n = \frac{\sum_t \sum_k p(k | y_t) [\tilde{\Sigma}_n(y_t, k) + \tilde{\mu}_n(y_t, k) \tilde{\mu}_n'(y_t, k)]}{\sum_t \sum_k p(k | y_t)} - \hat{\mu}_n \hat{\mu}_n' \quad (3.39)$$

where

$$\tilde{\mu}_n(y_t, k) = \bar{\Sigma}_{n,k} [\Sigma_n + \bar{\Sigma}_{n,k}]^{-1} \mu_n + \Sigma_n [\Sigma_n + \bar{\Sigma}_{n,k}]^{-1} \bar{\mu}_{n,k} \quad (3.40)$$

$$\tilde{\Sigma}_n(y_t, k) = [\Sigma_n^{-1} + \bar{\Sigma}_{n,k}^{-1}]^{-1} \quad (3.41)$$

$$\bar{\mu}_{n,k} = ((\nabla_n f)')^{-1} [y_t - (\mathbf{I} + \nabla_x f)' \mu_{x,k} - g(\mathbf{n}_0, \mathbf{x}_0)] \quad (3.42)$$

$$\bar{\Sigma}_{n,k} \approx ((\nabla_n f)')^{-1} (\mathbf{I} + \nabla_x f)' \Sigma_{x,k} (\mathbf{I} + \nabla_x f) (\nabla_n f)^{-1} \quad (3.43)$$

$$p(k | y_t) = \frac{p_k N(y_t; \mu_{y,k}, \Sigma_{y,k})}{\sum_l p_l N(y_t; \mu_{y,l}, \Sigma_{y,l})} \quad (3.44)$$

4. If the likelihood of the observed noisy data has not converged, return to step 2.

More detailed explanations are given in [Kim98d]. The above equations are general and easily applicable to any representation of signal in the log-spectral and cepstral domains. Once the parameters of the distribution of $\mathbf{n}$ are computed, an MMSE estimate is used to calculate the clean speech given the observed noisy speech as

$$\hat{\mathbf{x}}_{t, MMSE} = y_t - \sum_k p(k | y_t) f(\mu_{x,k}, \mu_n) \quad (3.45).$$

It is found in [Mor96] that both CDCN and VTS are highly efficient at medium levels of noise, i.e., at SNRs of 10 dB and above, but VTS performs slightly better.

3.2.1.3 SPLICE

From above, it is seen that both CDCN and VTS are algorithms that assume no a priori knowledge of the noise. These methods estimate the parameters by maximizing the likelihood of the observed feature sequence of noisy speech, given a Gaussian mixture distribution for the sequence of clean speech. Since the transformation that relates the

noisy speech feature to the clean speech feature is nonlinear, both CDCN and VTS approximate it with a truncated Taylor series in order to estimate it. However, both algorithms assume that the noise is stationary, and thus, both perform poorly when this assumption is violated. To further improve the performance of the system under non-stationary noise, the approach SPLICE, short for Stereo-based Piecewise Linear Compensation for Environment, was proposed by Deng et al in [Den01]. Remarkable improvements have been obtained with this method.

There are two basic modeling assumptions in SPLICE. The first one is that the noisy speech feature vector $\mathbf{y}$ follows a mixture of Gaussians:

$$p(\mathbf{y}) = \sum_k p(k)p(\mathbf{y} | k) = \sum_k p(k)N(\mathbf{y}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \quad (3.46)$$

The mixture component k takes a set of discrete values, one for each cepstral-space partitioning. A piecewise linear approximation between the clean speech MFCC vector $\mathbf{x}$ and its noisy counterpart $\mathbf{y}$ is made for each such partitioning. The distribution parameters in $p(\mathbf{y})$ are trained for each separate noisy condition, using noisy speech data. The second assumption in SPLICE is that the conditional PDF for clean speech $\mathbf{x}$ is also a Gaussian. The mean vector is assumed to be a shifted version of the noisy speech $\mathbf{y}$, that is,

$$p(\mathbf{x} | \mathbf{y}, k) = N(\mathbf{x}; \mathbf{y} + \mathbf{r}_k, \Gamma_k) \quad (3.47)$$

where $\mathbf{r}_k$ are the correction vectors that need to be trained using stereo training data.

With the above assumption, the MMSE estimate of clean speech feature vectors is given as

$$\hat{\mathbf{x}}_{MMSE} = \sum_k p(k | \mathbf{y})(\mathbf{y} + \mathbf{r}_k) = \mathbf{y} + \sum_k p(k | \mathbf{y})\mathbf{r}_k \quad (3.48)$$

where

$$p(k | \mathbf{y}) = \frac{p(k)p(\mathbf{y} | k)}{\sum_l p(l)p(\mathbf{y} | l)} \quad (3.49)$$

A fast implementation of SPLICE is to approximate the weights according to

$$p(\hat{k} | \mathbf{y}) = \begin{cases} 1 & \hat{k} = \arg \max_k p(k | \mathbf{y}) \\ 0 & \text{otherwise} \end{cases} \quad (3.50).$$

This approximate estimate consists of two sequential steps of operation: 1) finding the optimal “codeword” $\hat{k}$ using the VQ codebook based on the parameters (μ_k, Σ_k) and 2) adding the codeword-dependent vector $\mathbf{r}_k$ to the noisy speech vector. The correction vectors $\mathbf{r}_k$ are trained using stereo training data $(\mathbf{x}_t, \mathbf{y}_t)$, maximum likelihood training is used and the resulting formula is

$$\mathbf{r}_k = \frac{\sum_t p(k | \mathbf{y}_t)(\mathbf{x}_t - \mathbf{y}_t)}{\sum_t p(k | \mathbf{y}_t)} \quad (3.51).$$

An improved, noise-normalized SPLICE is proposed in [Den03]. In [Den03], the authors further improve the performance of SPLICE by estimating the nonstationary acoustic noise recursively. It has been noticed that the accuracy of the piecewise linear approximation for the nonlinear environmental model is a key factor in speech enhancement performance. Therefore the operating points for the truncated Taylor series expansion need to be determined appropriately. Instead of setting the operating points at some fixed values, a smoothed version of the previous estimate $\mathbf{n}_t$ is used to iteratively update the operating point of the Taylor series expansion, and to subsequently increase the accuracy for noise estimation.

3.2.1.4 Extensions and Variations

The previous speech models described in 3.2.1.1, 3.2.1.2 and 3.2.1.3 for speech enhancement use no dynamic or trajectory properties of speech. The clean speech model is a Gaussian mixture model and each frame of data is enhanced independently. A new algorithm for statistical speech feature enhancement was proposed in [Den04b]. In this paper, the prior clean speech model exploited takes into account both the static and dynamic properties of clean speech, in the domain of log mel-channel energy or equivalently in the domain of cepstrum.

The joint prior distributions (still in the form of Gaussian mixtures) of clean speech incorporate both the static and frame-differential dynamic cepstral parameters. Due to the inclusion of the delta feature, the speech frame is no longer independent of its previous

frame. This allows the trajectory information of speech to be captured as part of the prior information. Compared with the conventional approaches, which exploit only the static features in the Gaussian mixture model, the additional information source permits a more accurate characterization of the prior statistical properties of the clean speech sequence.

Full posterior probabilities for clean speech given the noisy observations are computed and a conditional minimum mean square error (MMSE) estimator for the clean speech feature is derived using the full posterior. The final form of the derived conditional MMSE estimator is shown to be a weighted sum of three separate terms. One of the three terms is shown to arrive naturally from the model in the absence of any prior information; the other two terms result from the speech model using only the static prior and only the dynamic prior, respectively. The experimental results demonstrate improvement in noise-robust recognition accuracy by incorporating the joint prior for both static and dynamic parameter distributions in the speech model, compared with using only the static or dynamic prior and with using no prior.

Another contribution of the paper [Den04b] is that a general form of nonlinear approximation technique has been developed in dealing with the nonlinearity in the acoustic distortion model. In that case, the Vector-Taylor-Series algorithm is a highly simplified and special case of this algorithm.

However, the inherent time delay with this approach is high. Firstly, the nonstationary noise needs to be recursively estimated using iterative stochastic approximation [Den03], which makes the algorithm actually a noise-tracking-only method. Next, to remove the noise estimate from the noisy speech, the intra-frame iterative refinement of a Taylor series expansion point is required to make the speech estimate converge. Comparatively, filter-based sequential estimation methods have a reasonable numerical complexity and a low time delay. That is why filter model-based approaches have attracted much attention recently.

3.2.2 Filter-Model-Based Approaches

Different from the statistical-model-based approaches, a state space needs to be built in filter-model-based approaches. The state space includes a process state equation describing the distribution of noise and an observation state equation modeling the noisy signal. In this case, the noise is identified as the state variable corrupted by speech. Kim et al used a simple random walk model as the process state equation for noise in [Kim98b]. Later on, Raj et al found that many types of noises, e.g., street, bar, subway, have a large component that is relatively slow evolving [Raj04]. This means that, even for non-stationary noise, the noise spectrum of the present frame can well be predicted from the previous frames. Noise can be represented by a continuous autoregressive model [Raj04], which achieves better accuracy than the random walk model used by Kim [Kim98b].

In Kim's approach [Kim98b], the nonlinear equation (3.24) is linearized by a statistical linear approximation (SLA) algorithm. Then the interacting multiple model (IMM) approach, which is based on switching Kalman filtering (more details in chapter 5), is applied for noise parameter tracking. After the environmental noise is estimated, the speech feature compensation procedure is completed by computing the clean feature estimate according to the minimum mean square error (MMSE) criterion. The MMSE procedure is the same as those in the statistical-model-based approaches.

An alternative to Kim's approach is proposed by Fujimoto et al [Fuj05] in order to compensate for the effects of slowly evolving environments. In their method, rather than linearizing the non-linear equation (3.24), a particle filter algorithm is used to track the noise. Particle filters are sequential Monte Carlo methods for nonlinear and/or non-Gaussian Bayesian tracking problems. Better performance can be observed in their experiments compared to VTS. Therefore, it is shown later on in [Umb05] that when the particle filter is used to address stationary noise it yields no additional performance advantage over the speech feature enhancement by the Vector Taylor Series

approximation method. Compared with statistical-model-based approaches like VTS, the main advantage of filter-based estimation methods is a low time delay.

3.3 Summary

In this chapter, some mainstream methods of speech enhancement presented in the literature were reviewed. Although speech enhancement can be carried out in the time domain, where the environmental noise and speech are considered to be additive, it is believed that enhancement in the speech feature domain is more desirable if the purpose of speech enhancement is for robust speech recognition. We revealed that the distortion caused by additive ambient noises is highly nonlinear in the feature domain. Then we presented some prevailing approaches for speech feature enhancement. The feature enhancement methods mentioned so far have separated noise tracking and speech feature enhancement. One of the problems with these approaches is that, since the clean speech model is a Gaussian mixture model, each frame of data is enhanced independently. Without post-processing, this can result in artifacts, such as sharp single frame transitions, that are not part of the original clean speech signal. We will further report on the filter-model-based approaches and compare their performances in chapter 5.

Chapter 4

Experimental Tasks and Corpora

Since speech recognition engines and speech databases shall be used to test the algorithms developed in this thesis, they will be addressed in a concise manner in this chapter.

The algorithms proposed in this thesis for estimating the clean speech feature vectors will be evaluated on two databases, one is the *Numbers* v1.3 database and the other one is the *Aurora 2* database. The reference recognizers are based on the HTK software package version 2.2 from Entropic [HTK99]. We choose HTK because it is the most widely used toolkit for building and manipulating Hidden Markov Models (HMMs) for speech recognition and is available for free download at <http://htk.eng.cam.ac.uk>.

The acoustic models used for recognition are trained on clean data only. With this type of training, the highest performance can only be obtained in case of testing on clean data. Since these models contain no information about possible noise distortions, the question arises about what performance can be achieved for a different type of noise or a different SNR.

4.1 Experimental Conditions for the *Numbers* v1.3 Database

4.1.1 Speech Database

The speech data is from the *Numbers* v1.3 Corpus provided by the Centre for Spoken Language Understanding (CSLU) from the Oregon Health & Science University (<http://www.cslu.ogi.edu/corpora/>). The utterances in this corpus include isolated digit strings, continuous digit strings, and ordinal/cardinal numbers. The data were collected from both analog and digital telephone lines. The analog data were recorded using an analog-to-digital conversion box. These files were recorded as 16-bit, 8 kHz and stored in

a linear (uniformly quantized) format. The digital data were recorded with the CSLU T1 digital data collection system. These files were sampled at 8 kHz and stored as 8-bit μ -law files. Then all of the data were linearly encoded into the 16-bit RIFF standard file format.

Since the *Numbers* database is used here for pilot experiments, to simplify our simulations, we only tested on 100 zip-code utterances (5 continuous digits). Utterances from 50 male and 50 female speakers were picked randomly from the database. Noise is artificially added to the speech by a computer at SNRs (signal-to-noise ratio) of 15 dB, 10 dB, 5 dB, and 0 dB. When the SNR is 20 dB, the speech recognizer is able to give an acceptable performance without using a speech enhancement front-end. The SNR is defined as the ratio of signal to noise energy. We add two kinds of noises, white Gaussian noise and babble noise from the ITU-T supplement P.23 database [ITU98], to the clean speech. White Gaussian noise is generally the easiest type of noise to be handled, while speech in babble noise is a difficult one to be enhanced. In total, the test set used in our evaluation experiments thus consists of $2 * 4 * 100 = 800$ utterances.

4.1.2 Front-End Setting

The front-end is a cepstral analysis scheme where 12 Mel frequency cepstral coefficients (MFCC) are determined for speech frames of 32ms length. The frame shift is 12 ms. These parameters are the default settings in the HTK and also typical values for speech feature extraction. The MFCC coefficient of order 0 is not included. The cepstral coefficients and their first and second order derivatives are used. Thus each feature vector consists of 36 components in total. Further details for the cepstral analysis scheme are:

- Application of a Hamming window
- FFT based Mel filter-bank with 20 frequency bands in the range from 1 Hz up to half of the sampling frequency

4.1.3 HTK Reference Recognizer

Phonemes are used to describe the digits. Totally 21 phonemes were modeled according to the British English BEEP pronouncing dictionary [HTK99]. The entries in the dictionary are:

eight ey t sp
five f ay v sp
four f ow r sp
nine n ay n sp
oh ow sp
one w ax n sp
SENT-END [] sil
SENT-START [] sil
seven s eh v e n sp
sil sil
six s i k s sp
three th r ii sp
two t oo sp
zero z ii r ow sp

Each phoneme was modeled by a HMM with the following parameters:

- 3 states per phoneme (5 states in HTK notation with 2 dummy states at beginning and end)
- simple left-to-right models without skips over states
- mixture of 8 Gaussians per state
- Diagonal covariance matrices
- Two pause models were defined. The first one called “sil” (silence model) consists of 3 states and models the pauses before and after the utterance. A mixture of 8 Gaussians models each state. The second pause model called “sp” is used to model pauses between words. It consists of a single state, so the “sp” model is a tee model, i.e., the entry non-emitting state has a transition to the exit non-emitting state. The emitting state of “sp” is tied with the middle state of the silence model.

- The training is done in several steps by applying the Baum-Welch re-estimation scheme several times.

The above settings are taken from a similar example provided in the HTK manual [HTK99]. Although the speech engine obtained may not be the optimal one, the speech recognizer obtained by using the above settings usually are able to give a reasonably good performance.

4.2 Experimental Conditions for the *Aurora* 2.0 Database

4.2.1 Speech Database

Different from the *Numbers* database where only clean speech is provided and noise needs to be added by a computer program, noisy speech is provided directly in the *Aurora* database. Next, we will give a short description of the *Aurora* database. For more details about how the Aurora working group has generated the noisy speech, please refer to [Hir00].

The *Aurora* 2.0 database (<http://www.icp.inpg.fr/ELRA/home.html>) takes the *TIDigits* database as its basis. This database contains the recordings of male and female US-American adults speaking isolated digits and sequences of up to 7 digits. The original 20 kHz *TIDigits* data have been down-sampled to 8 kHz with a low-pass filter extracting the spectrum between 0 and 4 kHz. An additional filtering is applied to consider the realistic frequency characteristics of terminals and equipment in the telecommunication area. A filter with frequency characteristics G.712 defined by the ITU is used to simulate the realistic frequency characteristics of a telecommunication channel.

Noises in the database represent the most probable application scenarios for telecommunication terminals and were added to the filtered *TIDigits* samples. Again, noise is filtered with the G.712 characteristic before being added. The SNR (signal-to-noise ratio) is defined as the ratio of signal to noise energy after filtering both signals with

the G.712 characteristic. The speech energy is determined by applying the ITU recommendation P.56 with the corresponding ITU software. The noise energy is calculated based on a noise segment of the same length as the speech signal, that is randomly cut out of the whole noise recording.

We work on the clean training mode in which the recognition engine is trained on clean data alone. So test set A in the *Aurora 2* database is tested under the clean training mode. 4004 utterances from 52 male and 52 female speakers are split into 4 subsets with 1001 utterances in each. Recordings of all speakers are present in each subset. One noise signal is added to each subset of 1001 utterances at SNRs of 20 dB, 15 dB, 10 dB, 5 dB and 0 dB. Furthermore the clean case without adding noise is taken as a sixth condition. Four noises, suburban train, babble, car and exhibition hall are added to the 4 subsets. In total, the test set used for our evaluation experiments thus consists of 4 times 6 times 1001 = 24024 utterances.

4.2.2 Front-End Setting

The *Aurora 2* supplied standard front-end is a cepstral analysis scheme where 13 Mel frequency cepstral coefficients (MFCCs), including the coefficient of order 0, are determined for a speech frame of 25 ms length. The frame shift is 10 ms. A feature vector with size 39 is defined by using 13 cepstral coefficients (with the 0th cepstral coefficient which indicates the power of speech signal), plus the corresponding delta and acceleration coefficients. This front-end has been standardized by the European Telecommunication Standards Institute (ETSI) as an outcome of the Aurora working group.

Further details of the cepstral analysis scheme are:

- Signal offset compensation with a notch filtering operation
- Application of a Hamming window
- FFT based Mel filterbank with 23 frequency bands in the range from 64 Hz up to half of the sampling frequency

4.2.3 HTK Reference Recognizer

The digits are modeled as whole word HMMs with the following parameters:

- 16 states per word (18 states in HTK notation with 2 dummy states at beginning and end)
- simple left-to-right models without skips over states
- mixture of 3 Gaussians per state
- diagonal covariance matrices
- Two pause models are defined. The silence model “sil” consists of 3 states. There is a mixture of 6 Gaussians in each state. The pause model “sp” consists of a single state which is tied with the middle state of the silence model.
- The training is done in several steps by applying the Baum-Welch re-estimation scheme several times.

For a fair comparison with other results based on *Aurora 2* database in the literature, we use the standard settings that are defined in the *Aurora 2* tasks for both speech feature extraction and speech recognition.

4.3 Scoring

The performance of a speech recognizer is evaluated by comparing the hypothesized word sequence to a reference transcription. These sequences are matched by performing an optimal string match using dynamic programming. Once the optimal alignment has been found, the number of substitution errors, S , deletion errors, D , and insertion errors, I , with respect to the true transcript can be determined. These are defined as follows:

Substitutions S : words which are substituted for other words;

Deletions D : words which are completely deleted from the transcripts;

Insertions I : words which are inserted into the hypothesized transcript that were not present in the true transcript.

Usually, the percentage *word error rate* (WER) is quoted. The WER is given proportional to the sum of these quantities [HTK99]:

$$WER = 1 - \frac{N - D - S - I}{N} \quad (4.1)$$

where N is the total number of words in the correct transcription.

The *Word Accuracy Rate* (WAR) is given by :

$$WAR = 1 - WER \quad (4.2)$$

4.4 Why Test on Both *Number* v1.3 and *Aurora 2.0* databases

The motivation for studying Kalman filter-based speech enhancement systems later in this thesis is that they can handle colored noise and have a reasonable numerical complexity. Additionally they have a low inherent time delay. A key issue in Kalman filtering is that the filtering algorithm relies on a set of parameters that are unknown and have to be replaced by estimated quantities. In order to obtain a Kalman filter output with high accuracy, it is important that the models be accurately estimated.

The algorithms developed in this thesis are made of two models: a speech model and a noise model. How to estimate the speech model will be elaborated in the following chapters. These parametric techniques can give a reasonably good speech model since the training set used to estimate the parameters in the speech model can often easily cover the acoustic space. However, because the noise types and levels are too numerous, it is too difficult to generate in advance a noise model with a full coverage of the acoustic space for the time-varying noise. Given only a limited amount of noisy speech, we find it hard to estimate accurately the parameters in the noise model.

For experiments that are conducted on the database *Number* v1.3, since noise is added to the clean speech manually, the true noise sequences are available to give an accurate estimation of the noise model. The accuracy and appropriateness of the proposed speech models for speech enhancement can be evaluated without considering the side effect of an inaccurate noise model.

For experiments that are conducted on the database *Aurora 2*, the true noise sequences are not available, since we have only noisy utterances at hand. The parameters for the noise model are trained on the first ten frames of the noisy utterances, which only gives a coarse estimation of the noise model. The accuracy and appropriateness of the proposed speech models for speech enhancement can then be evaluated under real-life situations.

Chapter 5

Speech Enhancement By Noise-Tracking-Only Methods

One of the prevailing approaches towards solving the noise robustness problem in speech recognition systems is speech feature enhancement, where a noisy speech signal is processed in the feature extraction stage. The speech features to be enhanced are usually the Mel-Frequency Cepstrum Coefficients (MFCC), which are used by most speech recognizers.

While in some current approaches noise is considered stationary, some approaches explicitly address non-stationary noise. The methods based on the sequential EM algorithm are then the prevailing ones. Previous work has shown that they can estimate the noise sequence effectively [Kim98c] [Afi04]. However, the inherent time delay with the sequential EM algorithms is high because frame-by-frame iterative estimation is required to make the noise parameters converge. Recently, filter-based sequential estimation methods have attracted much attention since they have a reasonable numerical complexity and a low time delay compared with a sequential EM algorithm. For instance, in [Kim98b], a Kalman filter is used to track the non-stationary noise characteristics. Based on a piecewise linear approximation of the nonlinear function, using a first order Taylor series expansion or a higher order Taylor series expansion, the Kalman filter can be used to address the time-varying noise compensation problem. Since the distortion caused by the additive environmental noise is highly non-linear in the feature domain, the particle filter algorithm [Fuj05] [Raj04] is a possible alternative method to Kim's linearized Kalman filter [Kim98b] method. The particle filter is a Bayesian estimation method based on the sequential Monte-Carlo method. Another possible alternative to Kim's linearized Kalman filter method is the unscented Kalman filter (UKF) [Jul97].

In this chapter, we will review Kim's linearized Kalman filter [Kim98b] method and propose to replace the linearized Kalman filter with the unscented Kalman filter [Jul97], in order to estimate the noise in a nonlinear system.

5.1 State-Space for Noise-Tracking-Only Method

Assuming that the clean speech signal $y(t)$ is corrupted by an independent ambient noise $n(t)$ in the time domain, the noisy speech $o(t)$ can be presented as:

$$o(t) = y(t) + n(t) \quad (5.1)$$

If we frame the speech signals and convert them into the log-spectral domain on a Mel-scale, the above relation becomes [Mor96]:

$$\exp(\mathbf{o}_t) = \exp(\mathbf{y}_t) + \exp(\mathbf{n}_t) \quad (5.2)$$

where $\mathbf{o}_t$, $\mathbf{n}_t$ and $\mathbf{y}_t$ represent the noisy speech log-spectrum, the noise log-spectrum and the hypothetical clean speech log-spectrum vectors at the t -th time frame, respectively. After some numerical calculation, we have [Mor96]:

$$\begin{aligned} \mathbf{o}_t &= \log(\exp(\mathbf{y}_t) + \exp(\mathbf{n}_t)) \\ &= \mathbf{y}_t + \log(\mathbf{I} + \exp(\mathbf{n}_t - \mathbf{y}_t)) \end{aligned} \quad (5.3)$$

With a discrete cosine transform (DCT), we can get the following parametric model in the cepstral domain [Ace93]:

$$\begin{aligned} \mathbf{o}_t &= \mathbf{C} \log(\exp(\mathbf{C}^{-1}\mathbf{y}_t) + \exp(\mathbf{C}^{-1}\mathbf{n}_t)) \\ &= \mathbf{y}_t + \mathbf{C} \log(\mathbf{I} + \exp(\mathbf{C}^{-1}(\mathbf{n}_t - \mathbf{y}_t))) \end{aligned} \quad (5.4)$$

where $\mathbf{o}_t$, $\mathbf{n}_t$ and $\mathbf{y}_t$ represent the noisy speech cepstrum, the noise cepstrum and the hypothetical clean speech cepstrum at the t -th time frame respectively. In practice, the DCT matrix $\mathbf{C}$ is not square, so the dimension of the cepstrum vector is smaller than the number of Mel-scaled filters. This means that we are losing resolution when going back to the frequency domain, and thus equation (5.4) represents only an approximation.

As in [Raj04], we can model the sequence of log-spectral vectors or cepstrum of noise as the output of a first-order auto-regressive (AR) system excited by a Gaussian process $\mathbf{v}$ with mean and covariance matrix $\mathbf{Q}$ as follows:

$$\mathbf{n}_t = \mathbf{A}\mathbf{n}_{t-1} + \mathbf{v}_t \quad (5.5)$$

Note that in the log-spectral domain, the $d \times d$ state $\mathbf{A}$ is not diagonal since there is considerable correlation among the components of a log-spectral feature vector. The state matrix and process noise parameters can be estimated from a segment of training samples of noise. It is found from our experiments that equation (5.5) can provide a better description of the log-spectral dynamics than a simple random walk model as used in [Kim98b] [Fuj05]

$$\mathbf{n}_t = \mathbf{n}_{t-1} + \mathbf{v}_t \quad (5.6)$$

where $\mathbf{A}$ equals an identity matrix $\mathbf{I}$. Different from the conventional approach where the speech signal is modeled as the output of a dynamic system, the noise compensation problem here is treated inversely by assuming that the speech is the nonlinear obscuring influence that prevents us from observing the noise.

Without losing generality, we could assume that the probability density function (pdf) of the clean speech feature vector $\mathbf{y}$ can be represented by a multivariate Gaussian mixture model (GMM) as:

$$p(\mathbf{y}) = \sum_{j=1}^M p(j) N(\mathbf{y}; \boldsymbol{\mu}_{\mathbf{y},j}, \boldsymbol{\Sigma}_{\mathbf{y},j}) \quad (5.7)$$

where M is the total number of mixture components and $p(j)$, $\boldsymbol{\mu}_{\mathbf{y},j}$ and $\boldsymbol{\Sigma}_{\mathbf{y},j}$ represent the given a priori probability, the mean and the covariance of the j th Gaussian distribution, respectively.

Let each mixture component form a Kalman filter, then Equations (5.5) and (5.3) form the state and observation equations of a dynamical system in the feature domain as:

$$\mathbf{n}_t = \mathbf{A}\mathbf{n}_{t-1} + \mathbf{v}_t \quad \mathbf{v} \sim N(0, \mathbf{Q}) \quad (5.8)$$

$$\mathbf{o}_t = f(\mathbf{n}_t, \mathbf{y}_t) \quad \mathbf{y}_j \sim N(\boldsymbol{\mu}_{\mathbf{y},j}, \boldsymbol{\Sigma}_{\mathbf{y},j}) \quad (5.9)$$

The above state space model is general and easily applicable to any representation of the signal in the log-spectral and cepstral domains. The difference is that, when the discrete cosine transform (DCT) is used to reduce the spatial correlation in the feature vectors, the matrices $\mathbf{Q}$, $\boldsymbol{\Sigma}_{\mathbf{y},j}$ could be diagonal.

As seen from equations (5.8) and (5.9), this is a switching state-space model. Therefore, switching Kalman filters should be applied. In the next section we will present the fundamental concepts of switching state-space models and give some in-depth review of the algorithms that can handle the interactions between the parallel Kalman filters in an efficient way.

5.2 Review of Switching Kalman Filter

5.2.1 Introduction

A switching model is a model with switching parameters, where the model parameters change among a set of different parameters, and where the change is allowed to occur at any time. A linear case of the switching model is

$$\mathbf{z}(t) = \Phi_{s_t=j} \mathbf{z}(t-1) + \mathbf{u}_{s_t=j} + \mathbf{W}_{s_t=j}(t) \quad \mathbf{W}_{s_t=j} \sim N(\mathbf{0}, \mathbf{Q}_j) \quad (5.10)$$

$$\mathbf{o}(t) = \mathbf{H}_{s_t=j} \mathbf{z}(t) + \mathbf{V}_{s_t=j}(t) \quad \mathbf{V}_{s_t=j} \sim N(\mathbf{0}, \mathbf{R}_j) \quad (5.11)$$

where $s_t = j$ denotes the mode of the model at time t , which is assumed to be one of the M possible modes.

For the fixed-parameter state space, a Kalman filter calculates the conditional mean and covariance of the hidden state $\mathbf{z}_t$, given the observation up to time t . It can be represented as:

$$\mathbf{z}_{t|t} = E[\mathbf{z}_t | \mathbf{o}_{1:t}] \quad (5.12)$$

$$\mathbf{V}_{t|t} = \text{Cov}[\mathbf{z}_t | \mathbf{o}_{1:t}] \quad (5.13).$$

Generalizing from the fixed-parameter case to the case with switching parameters like the model given in (5.10) and (5.11), the mean and covariance of $\mathbf{z}_t$ is conditioned not only on the observations up to time t but also on the entire switching history. The conditional mean and covariance become

$$E[\mathbf{z}_t | s_t = i_t, s_{t-1} = i_{t-1}, \dots, s_1 = i_1, \mathbf{o}_{1:t}] \quad (5.14)$$

and

$$\text{Cov}[\mathbf{z}_t | s_t = i_t, s_{t-1} = i_{t-1}, \dots, s_1 = i_1, \mathbf{0}_{1:t}] \quad (5.15)$$

where s_t represents the state at time t , $i_k (1 \leq i_k \leq M)$. If the switching evolution process $s_t = i_t, s_{t-1} = i_{t-1}, \dots, s_1 = i_1$, were known, the conventional state estimation techniques would directly apply [Chu99]. However, the switching evolution process is unknown. All the possibilities of the evolution have to be considered during the estimation. If the model switches among M different modes, at each time step t , any of the M models could be chosen and as such, there are potentially as many as M^t paths for the switching evolution up to time t . Therefore, the number of the elemental estimators has an exponential growth with respect to time [Sha93]. To make the state-estimation algorithm computationally feasible, approximations have to be made.

There are several general approaches to deal with this exponential growth. One is to approximate the mixture of M^t Gaussians with a mixture of r Gaussians. This is called the Generalized Pseudo Bayesian algorithm of order r (GPB(r)) [Sha93]. When $r=1$, we approximate a mixture of Gaussians with a single Gaussian approximation. When $r=2$, we collapse Gaussians that differ in their more recent history two steps ago. The more history we keep, the better the approximation. The generalized Pseudo-Bayesian approach (GPB) has been used for state estimation in the target tracking and econometrics literature for general dynamic systems [Sha93][Kim94]. The Interacting Multiple Models (IMM) [Sha93] is a good approximation to GPB2 (GPB with $r=2$) at the cost of only M (instead of M^2) filters (see Fig 3.1). Fig 3.1 shows a comparison of the GPB1, GPB2 and IMM filtering algorithms [Mur98a]. In the following, these three types of approximation and the associated suboptimal state estimation algorithms are discussed. More details can be found in [Sha93] [Mur98a][Maz03] and their references.

5.2.2 First Order Generalized Pseudo-Bayesian Approach

The structure of the first-order GPB (GPB1) algorithm is shown in Fig 5.1. At the start of each processing cycle, the information carried forward from the previous measurement interval is a single combined estimate; at each time t , the state estimate is carried out under each possible current model; the GPB1 estimator combining the output from all models into a single estimate at the end of each processing cycle, any information conditioned on previous model history events has been discarded.

Let us start by defining some notation. $\Pr$ is the shorthand notation for probability. The conditional mean and covariance of the hidden state $\mathbf{z}_t$, given the observations up to time t are denoted as:

$$\mathbf{z}_{t|t} = E[\mathbf{z}_t | \mathbf{o}_{1:t}] \quad (5.16)$$

$$\mathbf{V}_{t|t} = \text{cov}[\mathbf{z}_t | \mathbf{o}_{1:t}] \quad (5.17)$$

$$\mathbf{z}_{t|t}^j = E[\mathbf{z}_t | \mathbf{o}_{1:t}, s_t = j] \quad (5.18)$$

$$\mathbf{V}_{t|t}^j = \text{cov}[\mathbf{z}_t | \mathbf{o}_{1:t}, s_t = j] \quad (5.19)$$

$$P_{t|t}^j = \Pr(s_t = j | \mathbf{o}_{1:t}) \quad (5.20)$$

$$L_t^j = \Pr(\mathbf{o}_t | \mathbf{o}_{1:t-1}, s_t = j) \quad (5.21)$$

For each model, the Kalman filter algorithm is tailored to

$$\mathbf{z}_{t|t-1}^j = \Phi_j \mathbf{z}_{t-1|t-1} + \mathbf{u}_j \quad (5.22)$$

$$\mathbf{V}_{t|t-1}^j = \Phi_j \mathbf{V}_{t-1|t-1} \Phi_j^T + \mathbf{Q}_j \quad (5.23)$$

$$\mathbf{e}_t^j = \mathbf{o}_t - \mathbf{H}_j \mathbf{z}_{t|t-1}^j \quad (5.24)$$

$$\mathbf{K}_t^j = \mathbf{V}_{t|t-1}^j \mathbf{H}_j^T (\mathbf{H}_j \mathbf{V}_{t|t-1}^j \mathbf{H}_j + \mathbf{R}_j)^{-1} \quad (5.25)$$

$$\mathbf{z}_{t|t}^j = \mathbf{z}_{t|t-1}^j + \mathbf{K}_t^j \mathbf{e}_t^j \quad (5.26)$$

$$\mathbf{V}_{t|t}^j = (\mathbf{I} - \mathbf{K}_t^j \mathbf{H}_j) \mathbf{V}_{t|t-1}^j \quad (5.27)$$

While using the Kalman filter for the elemental estimator implementation, an assumption is made that posterior probabilities, $\Pr(\mathbf{o}_t | \mathbf{o}_{1:t-1}, s_t = j)$, are Gaussian and computed by

$$L_t^j = N(\mathbf{e}_t^j; \mathbf{0}, \mathbf{H}_j \mathbf{V}_{t|t-1}^j \mathbf{H}_j^T + \mathbf{R}_j) \quad (5.28)$$

Assume that the mixtures of Gaussians could be approximated as one Gaussian, the merged estimate is calculated according to

$$\mathbf{z}_{t|t} = \sum_j P_{t|t}^j \mathbf{z}_{t|t}^j \quad (5.29)$$

$$\mathbf{V}_{t|t} = \sum_j P_{t|t}^j \mathbf{V}_{t|t}^j + \sum_j P_{t|t}^j (\mathbf{z}_{t|t}^j - \mathbf{z}_{t|t}) (\mathbf{z}_{t|t}^j - \mathbf{z}_{t|t})^T \quad (5.30)$$

where $P_{t|t}^j = \Pr(s_t = j | \mathbf{o}_{1:t})$, called model probability, is calculated recursively according to Bayes rule as

$$\begin{aligned} \Pr(s_t = j | \mathbf{o}_{1:t}) &= \frac{1}{c} \Pr(\mathbf{o}_t | s_t = j, \mathbf{o}_{1:t-1}) \Pr(s_t = j | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{c} \Pr(\mathbf{o}_t | s_t = j, \mathbf{o}_{t-1}) \sum_i \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) \Pr(s_{t-1} = i | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{c} L_t^j \sum_i \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) \Pr(s_{t-1} = i | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{c} L_t^j \sum_i \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) P_{t-1|i}^i \end{aligned} \quad (5.31)$$

where c is the normalization constant

$$c = \sum_j L_t^j \sum_i \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) P_{t-1|i}^i \quad (5.32).$$

Each recursion of the GPB1 algorithm can be summarized as below for each time step:

1. For each current model, calculate the estimates $\mathbf{z}_{t|t}^j$ and $\mathbf{V}_{t|t}^j$ by running the KF listed in (5.22)-(5.27), given the estimate from previous time step $t-1$
2. Update model probabilities, $\Pr(s_t = j | \mathbf{o}_{1:t})$, according to (5.31)-(5.32)
3. Calculate the merged state and its covariance estimates, according to (5.29) and (5.30).

It is seen that GPB1 provides a very coarse approximation of the optimal system by merging outputs of all filters into a single estimate at each processing cycle. This approach is very efficient but we need to judge whether the price for loss of accuracy would be too great to be used for implementing our proposed algorithm.

5.2.3 Second-Order Generalized-Pseudo-Bayesian Approach

The second-order Generalized-Pseudo-Bayesian GPB2 estimator operates on similar principles to its first-order variant, except that the memory is allowed to extend to an additional processing cycle. In what follows, we review the GPB2 algorithm. The conditional mean and covariance of the hidden state $\mathbf{z}_t$, given the observation up to time t can be represented as:

$$\mathbf{z}_{t|t} = E[\mathbf{z}_t | \mathbf{o}_{1:t}] \quad (5.33)$$

$$\mathbf{V}_{t|t} = \text{cov}[\mathbf{z}_t | \mathbf{o}_{1:t}] \quad (5.34)$$

$$\mathbf{z}_{t|t}^j = E[\mathbf{z}_t | \mathbf{o}_{1:t}, s_t = j] \quad (5.35)$$

$$\mathbf{V}_{t|t}^j = \text{cov}[\mathbf{z}_t | \mathbf{o}_{1:t}, s_t = j] \quad (5.36)$$

$$\mathbf{z}_{t|t}^{i(j)} = E[\mathbf{z}_t | \mathbf{o}_{1:t}, s_{t-1} = i, s_t = j] \quad (5.37)$$

$$\mathbf{V}_{t|t}^{i(j)} = \text{cov}[\mathbf{z}_t | \mathbf{o}_{1:t}, s_{t-1} = i, s_t = j] \quad (5.38)$$

$$P_{t|t}^j = \Pr(s_t = j | \mathbf{o}_{1:t}) \quad (5.39)$$

$$P_{t-1,t|t}^{(i,j)} = \Pr(s_{t-1} = i, s_t = j | \mathbf{o}_{1:t}) \quad (5.40)$$

$$L_t^{i(j)} = \Pr(\mathbf{o}_t | \mathbf{o}_{1:t-1}, s_{t-1} = i, s_t = j) \quad (5.41)$$

$$W_{t|t}^{ij} = \Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t}) \quad (5.42)$$

We perform the following steps in sequence

$$\mathbf{z}_{t|t-1}^{i(j)} = \Phi_j \mathbf{z}_{t-1|t-1}^i + \mathbf{u}_j \quad (5.43)$$

$$\mathbf{V}_{t|t-1}^{i(j)} = \Phi_j \mathbf{V}_{t-1|t-1}^i \Phi_j^T + \mathbf{Q}_j \quad (5.44)$$

$$\mathbf{e}_t^{i(j)} = \mathbf{o}_t - \mathbf{H}_j \mathbf{z}_{t|t-1}^{i(j)} \quad (5.45)$$

$$\mathbf{K}_t^{i(j)} = \mathbf{V}_{t|t-1}^{i(j)} \mathbf{H}_j^T (\mathbf{H}_j \mathbf{V}_{t|t-1}^{i(j)} \mathbf{H}_j + \mathbf{R}_j)^{-1} \quad (5.46)$$

$$\mathbf{z}_{t|t}^{i(j)} = \mathbf{z}_{t|t-1}^{i(j)} + \mathbf{K}_t^{i(j)} \mathbf{e}_t^{i(j)} \quad (5.47)$$

$$\mathbf{V}_{t|t}^{i(j)} = (\mathbf{I} - \mathbf{K}_t^{i(j)} \mathbf{H}_j) \mathbf{V}_{t|t-1}^{i(j)} \quad (5.48)$$

The posterior probabilities $\Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t})$ and model probabilities $\Pr(s_t = j | \mathbf{o}_{1:t})$ are computed recursively by

$$L_t^{i(j)} = N(\mathbf{e}_t^{i(j)}; \mathbf{0}, \mathbf{H}_j \mathbf{V}_{t|t-1}^{i(j)} \mathbf{H}_j^T + \mathbf{R}_j) \quad (5.49)$$

$$P_{t-1,t|t}^{(i,j)} = \Pr(s_{t-1} = i, s_t = j | \mathbf{o}_{1:t}) = \frac{L_t^{i(j)} \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) P_{t-1|t-1}^i}{\sum_i \sum_j L_t^{i(j)} \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) P_{t-1|t-1}^i} \quad (5.50)$$

$$P_{t|t}^j = \Pr(s_t = j | \mathbf{o}_{1:t}) = \sum_i P_{t-1,t|t}^{(i,j)} \quad (5.51)$$

$$W_{t|t}^{i|j} = \Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t}) = P_{t-1,t|t}^{(i,j)} / P_{t|t}^j \quad (5.52)$$

The derivation of the mode update equation is as follows

$$\begin{aligned} \Pr(s_{t-1} = i, s_t = j | \mathbf{o}_{1:t}) &= \frac{1}{c} \Pr(s_{t-1} = i, s_t = j, y_t | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{c} \Pr(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) \Pr(s_{t-1} = i, s_t = j | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{c} \Pr(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) \Pr(s_{t-1} = i | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{c} L_t^{i(j)} \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) P_{t-1|t-1}^i \end{aligned} \quad (5.53)$$

where c is the normalization constant

$$c = \sum_i \sum_j L_t^{i(j)} \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) P_{t-1|t-1}^i \quad (5.54)$$

$P_{t|t}^j$ and $W_{t|t}^{i|j}$ can be derived according to Bayes rule and by straightforward conditional probability manipulation.

The merged estimate at node j is calculated according to

$$\mathbf{z}_{t|t}^j = \sum_i W_{t|t}^{i|j} \mathbf{z}_{t|t}^{i(j)} \quad (5.55)$$

$$\mathbf{V}_{t|t}^j = \sum_j W_{t|t}^{i|j} \mathbf{V}_{t|t}^j + \sum_j W_{t|t}^{i|j} (\mathbf{z}_{t|t}^{i(j)} - \mathbf{z}_{t|t}^j) (\mathbf{z}_{t|t}^{i(j)} - \mathbf{z}_{t|t}^j)^T \quad (5.56)$$

The state estimate is equivalent to

$$\mathbf{z}_{t|t} = \sum_j P_{t|t}^j \mathbf{z}_{t|t}^j \quad (5.57)$$

$$\mathbf{V}_{t|t} = \sum_j P_{t|t}^j \mathbf{V}_{t|t}^j + \sum_j P_{t|t}^j (\mathbf{z}_{t|t}^j - \mathbf{z}_{t|t}) (\mathbf{z}_{t|t}^j - \mathbf{z}_{t|t})^T \quad (5.58)$$

Hence the operation of the GPB 2 algorithm is such that the estimate from each model in the processing cycle is processed using each dynamic model, giving M^2 total elemental filters. At the end of each processing cycle, the M^2 estimates are combined down to M

estimates, combining estimates from different models in the previous processing cycle to leave one estimate for each model in the latest processing cycle. This is illustrated in Fig 5.1, which shows the structure of the algorithm. Compared with the structure of the GPB1 algorithm also shown in Fig 5.1, the GPB2 algorithm uses M^2 filters, thus it is able to maintain M estimates and propagate each estimate with each of the M filters at each processing interval.

The GPB2 algorithm can be summarized below:

1. Calculate $\mathbf{z}_{i|t}^{i(j)}$ and $\mathbf{V}_{i|t}^{i(j)}$ for $i, j = 1, 2, \dots, M$, using the KF in (5.43) to (5.48), using the merged state estimates at the previous time step, $\mathbf{z}_{t-1|t-1}^i$ and $\mathbf{V}_{t-1|t-1}^i$ as initial values.
2. Calculate the merging probabilities, $\Pr(s_{t-1} = i, s_t = j | \mathbf{o}_{1:t})$, according to (5.49)-(5.52).
3. Merge states for each model at the current time step using the merging probabilities according to (5.55) and (5.56), which will be used for the next time step recursion.
4. Update model probabilities, $\Pr(s_t = j | \mathbf{o}_{1:t})$, according to (5.49), (5.50) and (5.51).
5. Calculate the state and its covariance estimates, $\mathbf{z}_{t-1|t-1}$ and $\mathbf{V}_{t-1|t-1}$, according to (5.57) and (5.58)

5.2.4 Interacting Multiple Model (IMM) Approach

The IMM estimator is a methodology which achieves comparable performance to the GPB2 estimator using only M elemental filters, rather than M^2 as required by the latter.

A block diagram of the IMM algorithm is shown in Fig 5.1. There are M filters, each of which is supplied with a different input. However, rather than passing the output of each filter directly into the same filter at the next processing cycle, the algorithm mixes the estimates according to the Markov transition model in order to allow the system to react to changes to the model in force. In that way, the input to the j th filter becomes the best estimate of the state at time instant $t-1$, conditioned on the event that model j is in force at time instant t (the new sample time).

In this part, we adapt the general IMM state estimation approach to our nonlinear state-space model of feature dynamics. The following notations need to be first adopted

$$\tilde{\mathbf{z}}_{t-1|t-1}^j = E(\mathbf{z}_{t-1} | s_t = j, \mathbf{o}_{1:t-1}) \quad (5.59)$$

$$\tilde{\mathbf{V}}_{t-1|t-1}^j = \text{cov}(\mathbf{z}_{t-1} | s_t = j, \mathbf{o}_{1:t-1}) \quad (5.60)$$

$$W_{t-1|t-1}^{ij} = \Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t-1}) \quad (5.61)$$

The initialization values, $\tilde{\mathbf{z}}_{t-1|t-1}^j$ and $\tilde{\mathbf{V}}_{t-1|t-1}^j$, are obtained by merging according to

$$\tilde{\mathbf{z}}_{t-1|t-1}^j = \sum_i W_{t-1|t-1}^{ij} \mathbf{z}_{t-1|t-1}^i \quad (5.62)$$

$$\tilde{\mathbf{V}}_{t-1|t-1}^j = \sum_i W_{t-1|t-1}^{ij} \mathbf{V}_{t-1|t-1}^i + \sum_i W_{t-1|t-1}^{ij} (\mathbf{z}_{t-1|t-1}^i - \tilde{\mathbf{z}}_{t-1|t-1}^j)(\mathbf{z}_{t-1|t-1}^i - \tilde{\mathbf{z}}_{t-1|t-1}^j)^T \quad (5.63)$$

The mixing probability $W_{t-1|t-1}^{ij} = \Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t-1})$ is computed recursively with Bayes' rule

$$\begin{aligned} W_{t-1|t-1}^{ij} &= \Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t-1}) \\ &= \frac{\Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) \Pr(s_{t-1} = i | \mathbf{o}_{1:t-1})}{\sum_i \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) \Pr(s_{t-1} = i | \mathbf{o}_{1:t-1})} \end{aligned} \quad (5.64)$$

The Kalman filter algorithm for the IMM becomes:

$$\mathbf{z}_{t|t-1}^j = \Phi_j \tilde{\mathbf{z}}_{t-1|t-1}^j + \mathbf{u}_j \quad (5.65)$$

$$\mathbf{V}_{t|t-1}^j = \Phi_j \tilde{\mathbf{V}}_{t-1|t-1}^j \Phi_j^T + \mathbf{Q}_j \quad (5.67)$$

$$\mathbf{e}_t^j = \mathbf{o}_t - \mathbf{H}_j \mathbf{z}_{t|t-1}^j \quad (5.68)$$

$$\mathbf{K}_t^j = \mathbf{V}_{t|t-1}^j \mathbf{H}_j^T (\mathbf{H}_j \mathbf{V}_{t|t-1}^j \mathbf{H}_j + \mathbf{R}_j)^{-1} \quad (5.69)$$

$$\mathbf{z}_{t|t}^j = \mathbf{z}_{t|t-1}^j + \mathbf{K}_t^j \mathbf{e}_t^j \quad (5.70)$$

$$\mathbf{V}_{t|t}^j = (\mathbf{I} - \mathbf{K}_t^j \mathbf{H}_j) \mathbf{V}_{t|t-1}^j \quad (5.71)$$

And $\Pr(s_t = j | \mathbf{o}_{1:t})$ is updated as

$$\begin{aligned}
\Pr(s_t = j | \mathbf{o}_{1:t}) &= \frac{1}{c} \sum_i \Pr(s_{t-1} = i, s_t = j, \mathbf{o}_t | \mathbf{o}_{1:t-1}) \\
&= \frac{1}{c} \sum_i \Pr(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) \Pr(s_{t-1} = i, s_t = j | \mathbf{o}_{1:t-1}) \\
&= \frac{1}{c} \sum_i \Pr(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) \Pr(s_{t-1} = i | \mathbf{o}_{1:t-1})
\end{aligned} \tag{5.72}$$

where c is the normalization constant

$$c = \sum_j \sum_i \Pr(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) \Pr(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) \Pr(s_{t-1} = i | \mathbf{o}_{1:t-1}) \tag{5.73}$$

The calculation of $\Pr(\mathbf{o}_t | \mathbf{o}_{1:t-1}, s_{t-1} = i, s_t = j)$ in (5.72) is done with

$$\Pr(\mathbf{o}_t | \mathbf{o}_{1:t-1}, s_{t-1} = i, s_t = j) = N(\mathbf{e}_t^j; \mathbf{0}, \mathbf{H}_j \mathbf{V}_{t|t-1}^j \mathbf{H}_j^T + \mathbf{R}_j) \tag{5.74}$$

The state estimate is equivalent to

$$\mathbf{z}_{t|t} = \sum_j \Pr(s_t = j | \mathbf{o}_{1:t}) \mathbf{z}_{t|t}^j \tag{5.75}$$

$$\mathbf{V}_{t|t} = \sum_j \Pr(s_t = j | \mathbf{o}_{1:t}) \mathbf{V}_{t|t}^j + \sum_j \Pr(s_t = j | \mathbf{o}_{1:t}) (\mathbf{z}_{t|t}^j - \mathbf{z}_{t|t}) (\mathbf{z}_{t|t}^j - \mathbf{z}_{t|t})^T \tag{5.76}$$

The entire IMM algorithm can be summarized as follows:

1. Calculate mixing probability $\Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t-1})$ using (5.64)
2. Calculate merged initial conditions $\tilde{\mathbf{z}}_{t-1|t-1}^j$ and $\tilde{\mathbf{V}}_{t-1|t-1}^j$ for each current model according to (5.62) and (5.63), given the estimate from previous time step $t-1$
3. Calculate the estimates $\mathbf{z}_{t|t}^j, \mathbf{V}_{t|t}^j$ by running the KF in (5.65)-(5.71) with initial values obtained in step 2
4. Update model probabilities $\Pr(s_t = j | \mathbf{o}_{1:t})$ according to (5.72)-(5.74)
5. Calculate the state and its covariance estimates $\mathbf{z}_{t-1|t-1}$ and $\mathbf{V}_{t-1|t-1}$ according to (5.75) and (5.76).

5.2.5 Comparison of GPB1, GPB2 and IMM approximations

(1) Computational cost: Like the GPB1, the IMM approach has the same M possibilities considered at each time step, and hence it has the same computational cost. Compared

with the GPB2 that has M^2 possibilities considered at each time step, the IMM and GPB1 approach have a significantly lower computational cost (by a factor of M).

(2) Initial value: The IMM is different from the GPB1 approach in which all possibilities are initialized by the same value. The GPB2 approach takes into account all pairs of model switching from time $t-1$ to t , and each pair has its initial value from the previous model.

(3) Accuracy: The merging in the IMM approach takes place just before the estimation begins at each current model, which is different from the GPB2 in which the merging takes place after the estimation is completed at each current model. Since the merging in the GPB1 is more aggressive, it is expected to have a lower estimation accuracy than the IMM and the GPB2. The interacting multiple models (IMM) algorithm is a good approximation to the GPB2 at the cost of only M (instead of M^2) filters.

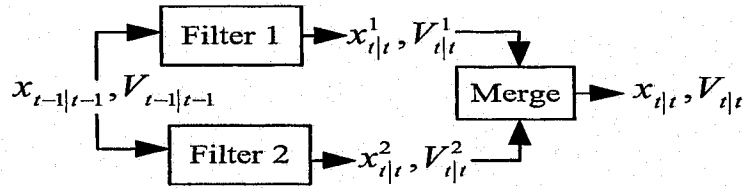


Fig 5.1 (a) GPB1 algorithm

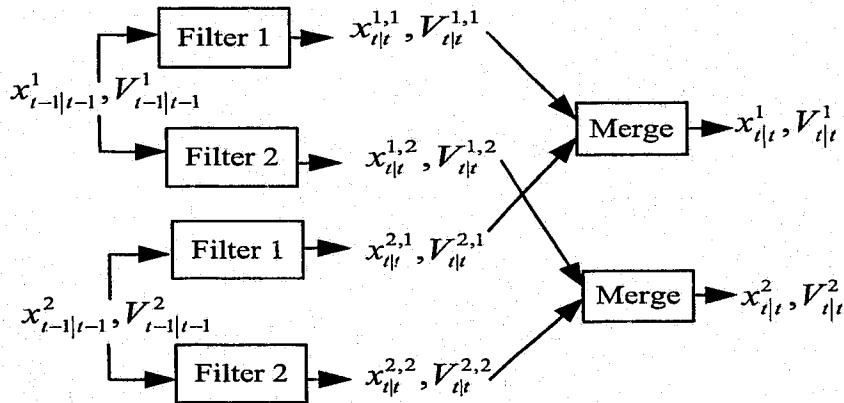


Fig 5.1 (b) GPB2 algorithm

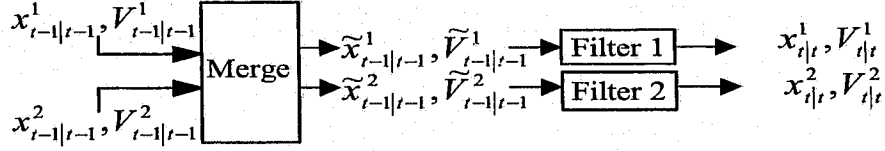


Fig 5.1 (c) IMM algorithm

It should be noted that, if there is a statistical independence assumption between s_t and s_{t-1} , the IMM approach is then identical to the GPB1 algorithm, since $W_{t-1|t-1}^{i|j} = \Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t-1}) = \Pr(s_{t-1} = i | \mathbf{o}_{1:t-1})$, $\tilde{\mathbf{z}}_{t-1|t-1}^j = \mathbf{z}_{t-1|t-1}$ and $\tilde{\mathbf{V}}_{t-1|t-1}^j = \mathbf{V}_{t-1|t-1}$.

5.3 Switching Kalman Filter for Noise-Tracking-Only Method

In section 5.1, we review some fundamental concepts of switching state-space models and the approximation algorithms that can handle the interactions between the parallel Kalman filters. Next we will show how to estimate the noise by using the switching Kalman filtering based on the state-space model given in equations (5.8) with (5.9).

5.3.1 Switching Extended Kalman Filter for Noise Tracking

Some notations are defined first:

$$\begin{aligned}
 \hat{\mathbf{n}}_{t|t}^j &= \mathbb{E}[\hat{\mathbf{n}}_t | \mathbf{o}_{1:t}, s_t = j] \\
 \mathbf{V}_{t|t}^j &= \text{cov}[\hat{\mathbf{n}}_t | \mathbf{o}_{1:t}, s_t = j] \\
 \hat{\mathbf{n}}_{t|t} &= \mathbb{E}[\hat{\mathbf{n}}_t | \mathbf{o}_{1:t}] \\
 \mathbf{V}_{t|t} &= \text{cov}[\hat{\mathbf{n}}_t | \mathbf{o}_{1:t}] \\
 L_t^j &= \Pr(\mathbf{o}_t | \mathbf{o}_{1:t-1}, s_t = j) \\
 P_t^j &= \Pr(s_t = j | \mathbf{o}_{1:t})
 \end{aligned} \tag{5.77}$$

Given a noisy speech feature vector $\mathbf{o}_t$ ($t=1, \dots, T$) and parameters $(\boldsymbol{\mu}_{y,j}, \boldsymbol{\Sigma}_{y,j}, j = 1 \dots M)$,

$\hat{\mathbf{n}}_{t|t}$ is found from the GPB1 algorithm using a bank of Extended Kalman Filters (EKF)

with the following steps performed in sequence:

$$\begin{aligned} & (\hat{\mathbf{n}}_{t|t}^j, \mathbf{V}_{t|t}^j, L_t^j) \\ &= EKF(\hat{\mathbf{n}}_{t-1|t-1}^j, \mathbf{V}_{t-1|t-1}^j, \mathbf{o}_t; \mathbf{Q}, \boldsymbol{\mu}_{y,j}, \boldsymbol{\Sigma}_{y,j}) \end{aligned} \quad (5.78)$$

$$P_t^j = \frac{L_t^j p(s_t = j)}{\sum_j L_t^j p(s_t = j)} \quad (5.79)$$

$$(\hat{\mathbf{n}}_{t|t}^j, \mathbf{V}_{t|t}^j) = \text{Collapse}(\hat{\mathbf{n}}_{t|t}^j, \mathbf{V}_{t|t}^j, P_t^j) \quad (5.80)$$

where $p(s_t = j) = p(j)$ is the a priori probability of the j th GMM mixture component and is assumed to be independent of the previous observation. Note that, because of the statistical independence assumption between $p(s_t)$ and $p(s_{t-1})$ in the collapsing step, the IMM approach is identical to the GPB1 algorithm [Kim98b].

The definition of the $EKF()$ function is as follows:

$$\hat{\mathbf{n}}_{t|t-1}^j = \mathbf{A} \hat{\mathbf{n}}_{t-1|t-1}^j \quad (5.81)$$

$$\mathbf{V}_{t|t-1}^j = \mathbf{A} \mathbf{V}_{t-1|t-1}^j \mathbf{A}^T + \mathbf{Q} \quad (5.82)$$

$$\mathbf{K}_t = \mathbf{V}_{t|t-1}^j \mathbf{H}^T (\mathbf{H} \mathbf{V}_{t|t-1}^j \mathbf{H}^T + \mathbf{G} \boldsymbol{\Sigma}_{y,j} \mathbf{G}^T)^{-1} \quad (5.83)$$

where

$$\mathbf{H} = \left. \frac{\partial f}{\partial \mathbf{n}} \right|_{\mathbf{n}_0, \boldsymbol{\mu}_{y,j}} \quad (5.84)$$

$$\mathbf{G} = \left. \frac{\partial f}{\partial \mathbf{y}} \right|_{\mathbf{n}_0, \boldsymbol{\mu}_{y,j}} \quad (5.85)$$

Filtered estimate of the state vector:

$$\mathbf{o}_{t|t-1} = f(\hat{\mathbf{n}}_{t|t-1}^j, \boldsymbol{\mu}_{y,j}) \quad (5.86)$$

$$\hat{\mathbf{n}}_{t|t}^j = \hat{\mathbf{n}}_{t|t-1}^j + \mathbf{K}_t (\mathbf{o}_t - \mathbf{o}_{t|t-1}) \quad (5.87)$$

Filtered state-error covariance matrix:

$$\mathbf{V}_{t|t}^j = \mathbf{V}_{t|t-1}^j - \mathbf{K}_t \mathbf{H}^T \mathbf{V}_{t|t-1}^j \quad (5.88)$$

If the above equations are applied to log-spectral features, the derivative matrices are

$$\mathbf{H} = \frac{\partial f}{\partial \mathbf{n}} = \text{diag} \left(1 - \frac{1}{1 + \exp(\mathbf{n}_0 - \boldsymbol{\mu}_{y,j})} \right) \quad (5.89)$$

$$\mathbf{G} = \frac{\partial f}{\partial \mathbf{y}} = \text{diag} \left(\frac{1}{1 + \exp(\mathbf{n}_0 - \boldsymbol{\mu}_{y,j})} \right) \quad (5.90)$$

In the cepstral domain, it becomes

$$\mathbf{H} = \frac{\partial f}{\partial \mathbf{n}} = \mathbf{I} - \mathbf{C} \text{diag} \left(1 - \frac{1}{1 + \exp(\mathbf{C}^{-1}(\mathbf{n}_0 - \boldsymbol{\mu}_{y,j}))} \right) \mathbf{C}^{-1} \quad (5.91)$$

$$\mathbf{G} = \frac{\partial f}{\partial \mathbf{y}} = \mathbf{C} \text{diag} \left(1 - \frac{1}{1 + \exp(\mathbf{C}^{-1}(\mathbf{n}_0 - \boldsymbol{\mu}_{y,j}))} \right) \mathbf{C}^{-1} \quad (5.92)$$

The choice of the Taylor series expansion point is a crucial issue and needs to be appropriately determined, as it affects the implementation of the algorithm greatly [Fre01]. For the clean speech $\mathbf{y}$, the operating point can be set naturally at the mean vector of the clean mixture model. If the operating point for the noise is set at $\mathbf{n}_0 = \boldsymbol{\mu}_n$ as in [Kim98b], the Kalman filter works on a stationary linear system, referred to as a linearized Kalman filter (LKF) in this chapter. In the extended Kalman filter, normally we set $\mathbf{n}_0 = \hat{\mathbf{n}}_{\ell t-1}^j$. Linearization is continually updated around the previous state estimate.

The posterior probabilities $L_t^j = \Pr(\mathbf{o}_t | \mathbf{o}_{1:t-1}, s_t = j)$ are updated according to:

$$L_t^j = N(\mathbf{o}_t; \mathbf{o}_{\ell t-1}, \mathbf{H} \mathbf{V}_{\ell t-1}^j \mathbf{H}^T + \mathbf{G} \boldsymbol{\Sigma}_{y,j} \mathbf{G}^T) \quad (5.93)$$

$P_t^j = \Pr(s_t = j | \mathbf{o}_{1:t})$ is calculated according to equations (5.79) and (5.93). The outputs of all filters are merged into a single estimate $\hat{\mathbf{x}}(k|k)$ and $\mathbf{P}_{\hat{\mathbf{x}}}(k|k)$ at time t according to:

$$\hat{\mathbf{n}}_{\ell t} = \sum_j P_t^j \hat{\mathbf{n}}_{\ell t}^j \quad (5.94)$$

$$\mathbf{V}_{\ell t} = \sum_j P_t^j \mathbf{V}_{\ell t}^j + \sum_j P_t^j (\hat{\mathbf{n}}_{\ell t}^j - \hat{\mathbf{n}}_{\ell t})(\hat{\mathbf{n}}_{\ell t}^j - \hat{\mathbf{n}}_{\ell t})^T \quad (5.95)$$

In [Kim98b], since the state estimations are associated with large state-error covariance matrices, to make the switching Kalman filter stable, a shrinking factor was introduced to reduce the Kalman gain. In our experiments, we use a more accurate noise model instead of the random-walk noise model in [Kim98b], so that the state-error covariance is decreased. In this case, the shrinking factor would bring negative effects to the results of our simulations. It was indeed found through our experiments that with the shrinking factor the recognition accuracy dropped by about 10%. Therefore no shrinking factor was used in our experiments.

Until now, we have obtained the noise sequence $\mathbf{n}$. To get the speech sequence, the noise sequence is to be removed from the noisy signal sequence $\mathbf{o}$ by using the approximated minimum mean squares estimation (MMSE) procedure that will be described in section 5.4.

5.3.2 Switching Unscented Kalman Filter (UKF) for Noise Tracking

Although the EKF algorithm presented in section 5.3.1 can handle the nonlinear distortion caused by noise in the feature domain, the flaws of the EKF have been pointed out by Julier and Uhlman in [Jul97]. A central and vital operation performance in the Kalman filter is the propagation of a Gaussian random variable (GRV) through the system dynamics. In the EKF, the state distribution is approximated by a GRV, which is then propagated analytically through the first-order linearization of the non-linear system. This can introduce large errors in the true mean and covariance of the transformed GRV, which may lead to sub-optimal performance and sometimes divergence [Wan00]. An improvement was introduced as the UKF proposed by Julier and Uhlman in [Jul97], and it addresses this problem by using a deterministic sampling approach. The state distribution is again approximated by a GRV, but is now represented using a set of carefully chosen sample points propagating through the nonlinear system. Julier and Uhlman have demonstrated the substantial performance gains of the UKF in the context of state-estimation for nonlinear control.

To further improve the performance, next we will apply the UKF to the noise-tracking-only method. The Unscented Kalman Filter is a straightforward extension of the unscented transformation (UT) to recursive estimation. The UT is a method for calculating the statistics of a random variable which undergoes a nonlinear transformation [Jul97]. It is described below in summary [Wan00]. Given an L -dimension random variable $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance matrix $\mathbf{P}_x$, the statistics of a random variable $\mathbf{y} = g(\mathbf{x})$ can be calculated by the following procedure: firstly, form a matrix χ of $2L+1$ sigma vectors χ_i with corresponding weights W_i , according to the following rules:

$$\begin{cases}
\chi_0 = \bar{\mathbf{x}} \\
\chi_i = \bar{\mathbf{x}} + \left(\sqrt{(L + \lambda) \mathbf{P}_x} \right)_i & i = 1, \dots, L \\
\chi_i = \bar{\mathbf{x}} - \left(\sqrt{(L + \lambda) \mathbf{P}_x} \right)_{i-L} & i = L + 1, \dots, 2L \\
W_0^{(m)} = \lambda / (L + \lambda) \\
W_0^{(c)} = \lambda / (L + \lambda) + (1 - \alpha^2 + \beta) \\
W_i^{(m)} = W_i^{(c)} = 1 / \{ 2(L + \lambda) \} & i = 1, \dots, 2L
\end{cases} \quad (5.96)$$

where $\lambda = \alpha^2 (L + \gamma) - L$ is a scaling parameter, α determines the spread of the sigma points around $\bar{\mathbf{x}}$ and is usually set to a small positive value, γ is a secondary scaling parameter which is usually set to 0, and β is used to incorporate a priori knowledge of the distribution of $\mathbf{x}$ (for Gaussian distributions, $\beta = 2$ is optimal). $\left(\sqrt{(L + \lambda) \mathbf{P}_x} \right)_i$ is the i^{th} column of the matrix square root. These sigma vectors are propagated through the nonlinear function,

$$\mathbf{Y}_i = g(\chi_i) \quad i = 0, \dots, 2L \quad (5.97)$$

and the mean and covariance for $\mathbf{y}$ are approximated by:

$$\begin{aligned}
\bar{\mathbf{y}} &\approx \sum_{i=0}^{2L} W_i^{(m)} \mathbf{Y}_i \\
\mathbf{P}_y &\approx \sum_{i=0}^{2L} W_i^{(c)} \{ \mathbf{Y}_i - \bar{\mathbf{y}} \} \{ \mathbf{Y}_i - \bar{\mathbf{y}} \}^T
\end{aligned} \quad (5.98)$$

It is seen that, by propagating a small set of deterministically chosen (Sigma) points through the nonlinear system, the posterior mean and covariance of a Gaussian random variable can be captured accurately. No explicit calculations of Jacobians or Hessians are necessary to implement this algorithm. For the Gaussian inputs, the UT results can achieve the accuracy of the third order. For non-Gaussian inputs, approximations are accurate to at least the second-order. The accuracy of third and higher order moments is determined by the choice of α and β [Jul00]. The EKF, in contrast, only achieves first-order accuracy.

Next we will show how to adapt the UKF algorithm to the state-space model given in equations (5.8) with (5.9). For frame t , some notations of the noise feature vector conditioned on the state $s_t = j$ are first defined:

$$\begin{aligned}
\hat{\mathbf{n}}_{t|t}^j &= E[\hat{\mathbf{n}}_t | \mathbf{o}_{1:t}, s_t = j] \\
\mathbf{V}_{t|t}^j &= \text{cov}[\hat{\mathbf{n}}_t | \mathbf{o}_{1:t}, s_t = j] \\
\hat{\mathbf{n}}_{t|t} &= E[\hat{\mathbf{n}}_t | \mathbf{o}_{1:t}] \\
\mathbf{V}_{t|t} &= \text{cov}[\hat{\mathbf{n}}_t | \mathbf{o}_{1:t}] \\
L_t^j &= \Pr(\mathbf{o}_t | \mathbf{o}_{1:t-1}, s_t = j) \\
P_t^j &= \Pr(s_t = j | \mathbf{o}_{1:t})
\end{aligned} \tag{5.99}$$

Initialize with:

$$\begin{aligned}
\hat{\mathbf{n}}_{0|0} &= E[\mathbf{n}_0] \\
\mathbf{V}_{0|0} &= E[(\mathbf{n}_0 - \hat{\mathbf{n}}_{0|0})(\mathbf{n}_0 - \hat{\mathbf{n}}_{0|0})^T]
\end{aligned} \tag{5.100}$$

Given a noisy speech feature vector $\mathbf{o}_t$ ($t=1, \dots, T$) and parameters $(\boldsymbol{\mu}_{y,j}, \boldsymbol{\Sigma}_{y,j}, j = 1 \dots M)$, $\hat{\mathbf{n}}_{t|t}$ is found from the GPB1 algorithm using a bank of unscented Kalman filters with the following steps performed in sequence:

$$(\hat{\mathbf{n}}_{t|t}^j, \mathbf{V}_{t|t}^j, L_t^j) = UKF(\hat{\mathbf{n}}_{t-1|t-1}, \mathbf{V}_{t-1|t-1}, \mathbf{o}_t; \mathbf{Q}, \boldsymbol{\mu}_{y,j}, \boldsymbol{\Sigma}_{y,j}) \tag{5.101}$$

$$P_t^j = \frac{L_t^j p(s_t = j)}{\sum_j L_t^j p(s_t = j)} \tag{5.102}$$

$$(\hat{\mathbf{n}}_{t|t}, \mathbf{V}_{t|t}) = \text{Collapse}(\hat{\mathbf{n}}_{t|t}^j, \mathbf{V}_{t|t}^j, P_t^j) \tag{5.103}$$

where $p(s_t = j) = p(j)$ is the a priori probability of the j th GMM mixture component and is assumed to be independent of the previous observation.

The *UKF()* function is given as follows:

Firstly, form $\hat{\mathbf{n}}_{t-1|t-1}^a, \mathbf{V}_{t-1|t-1}^a, \boldsymbol{\chi}^a$ as in [Wan00]:

$$\begin{aligned}
\hat{\mathbf{n}}_{t-1|t-1}^a &= [(\hat{\mathbf{n}}_{t-1|t-1})^T \quad \mathbf{0} \quad (\boldsymbol{\mu}_{y,j})^T]^T \\
\mathbf{V}_{t-1|t-1}^a &= [\mathbf{V}_{t-1|t-1} \quad \mathbf{0} \quad \mathbf{0}; \mathbf{0} \quad \mathbf{Q} \quad \mathbf{0}; \mathbf{0} \quad \mathbf{0} \quad \boldsymbol{\Sigma}_{y,j}] \\
\boldsymbol{\chi}^a &= [(\boldsymbol{\chi}^n)^T \quad (\boldsymbol{\chi}^v)^T \quad (\boldsymbol{\chi}^y)^T]^T
\end{aligned} \tag{5.104}$$

Sigma matrices: $i = 1, \dots, L$

$$\begin{aligned}
\boldsymbol{\chi}_0^a(t-1|t-1) &= \hat{\mathbf{n}}_{t-1|t-1}^a \\
\boldsymbol{\chi}_i^a(t-1|t-1) &= \hat{\mathbf{n}}_{t-1|t-1}^a + \left(\sqrt{(L + \lambda) \mathbf{V}_{t-1|t-1}^a} \right)_i \\
\boldsymbol{\chi}_{i+L}^a(t-1|t-1) &= \hat{\mathbf{n}}_{t-1|t-1}^a - \left(\sqrt{(L + \lambda) \mathbf{V}_{t-1|t-1}^a} \right)_i
\end{aligned} \tag{5.105}$$

Time update equations:

$$\begin{aligned}
\mathcal{X}_i^n(t|t-1) &= \mathbf{A} \mathcal{X}_i^n(t-1|t-1) + \mathcal{X}_i^v(t-1|t-1) \quad i = 0, \dots, 2L \\
\hat{\mathbf{n}}_{t|t-1} &= \sum_{i=0}^{2L} W_i^{(m)} \mathcal{X}_i^n(t|t-1) \\
\mathbf{V}_{t|t-1} &= \sum_{i=0}^{2L} W_i^{(c)} [\mathcal{X}_i^n(t|t-1) - \hat{\mathbf{n}}_{t|t-1}] [\mathcal{X}_i^n(t|t-1) - \hat{\mathbf{n}}_{t|t-1}]^T \\
Y_i(t|t-1) &= f(\mathcal{X}_i^n(t|t-1), \mathcal{X}_i^v(t|t-1)) \quad i = 0, \dots, 2L \\
\hat{\mathbf{o}}_{t|t-1} &= \sum_{i=0}^{2L} W_i^{(m)} Y_i(t|t-1)
\end{aligned} \tag{5.106}$$

Measurement update equations:

$$\mathbf{P}_{\hat{\mathbf{o}}\hat{\mathbf{o}}}(t) = \sum_{i=0}^{2L} W_i^{(c)} [Y_i(t|t-1) - \hat{\mathbf{o}}_{t|t-1}] [Y_i(t|t-1) - \hat{\mathbf{o}}_{t|t-1}]^T \tag{5.107}$$

$$\mathbf{P}_{\hat{\mathbf{n}}\hat{\mathbf{o}}}(t) = \sum_{i=0}^{2L} W_i^{(c)} [\mathcal{X}_i^n(t|t-1) - \hat{\mathbf{n}}_{t|t-1}] [Y_i(t|t-1) - \hat{\mathbf{o}}_{t|t-1}]^T \tag{5.108}$$

$$\mathbf{K}_t = \mathbf{P}_{\hat{\mathbf{n}}\hat{\mathbf{o}}}(t) \mathbf{P}_{\hat{\mathbf{o}}\hat{\mathbf{o}}}(t)^{-1} \tag{5.109}$$

Filtered estimate of the state vector:

$$\hat{\mathbf{n}}_{t|t}^j = \hat{\mathbf{n}}_{t|t-1} + \mathbf{K}_t(\mathbf{o}_t - \hat{\mathbf{o}}_{t|t-1}) \tag{5.110}$$

Filtered state-error covariance matrix:

$$\mathbf{V}_{t|t}^j = \mathbf{V}_{t|t-1} - \mathbf{K}_t \mathbf{P}_{\hat{\mathbf{o}}\hat{\mathbf{o}}}(t) \mathbf{K}_t^T \tag{5.111}$$

$$L_t^j = N(\mathbf{o}_t; \hat{\mathbf{o}}_{t|t-1}, \mathbf{P}_{\hat{\mathbf{o}}\hat{\mathbf{o}}}(t)) \tag{5.112}$$

Collapse is a moment matching function, defined as in [Sha93]:

$$\hat{\mathbf{n}}_{t|t} = \sum_j P_t^j \hat{\mathbf{n}}_{t|t}^j \tag{5.113}$$

$$\mathbf{V}_{t|t} = \sum_j P_t^j \mathbf{V}_{t|t}^j + \sum_j P_t^j (\hat{\mathbf{n}}_{t|t}^j - \hat{\mathbf{n}}_{t|t}) (\hat{\mathbf{n}}_{t|t}^j - \hat{\mathbf{n}}_{t|t})^T \tag{5.114}$$

For each frame of incoming noisy speech, the algorithm described above gives the estimate of the noise in the frame. The clean speech sequence can then be estimated by using the approximated minimum mean squares estimation (MMSE) procedure described next.

5.4 Speech Estimation by Minimum Mean Squares Estimation

The minimum mean squares estimation procedure developed in [Mor96] is given as follows:

$$\hat{\mathbf{y}}_t = \mathbf{o}_t - \sum_{j=1}^M p(j | \mathbf{o}_t, \hat{\mathbf{n}}_{t|t}) \log (1 + \exp (\hat{\mathbf{n}}_{t|t} - \boldsymbol{\mu}_{\mathbf{y},j})) \quad (5.115)$$

$\hat{\mathbf{n}}$ is the noise sequence obtained by the Switching Kalman filter method, $\mathbf{o}$ is the noisy signal sequence, and $\hat{\mathbf{y}}$ is the clean speech estimation. In the cepstrum domain, equation (5.115) becomes

$$\hat{\mathbf{y}}_t = \mathbf{o}_t - \sum_{j=1}^M p(j | \mathbf{o}_t, \hat{\mathbf{n}}_{t|t}) \mathbf{C} \log (1 + \exp (\mathbf{C}^{-1}(\hat{\mathbf{n}}_{t|t} - \boldsymbol{\mu}_{\mathbf{y},j}))) \quad (5.116)$$

where M is the number of mixture components and $p(j | \mathbf{o}_t, \hat{\mathbf{n}}_{t|t})$ is given by:

$$p(j | \mathbf{o}_t, \hat{\mathbf{n}}_{t|t}) = \frac{p(j)N(\mathbf{o}_t; \boldsymbol{\mu}_{\mathbf{o},j}, \boldsymbol{\Sigma}_{\mathbf{o},j})}{\sum_{i=1}^M p(i)N(\mathbf{o}_t; \boldsymbol{\mu}_{\mathbf{o},i}, \boldsymbol{\Sigma}_{\mathbf{o},i})} \quad (5.117)$$

where $\boldsymbol{\mu}_{\mathbf{o},i}$, $\boldsymbol{\Sigma}_{\mathbf{o},i}$ denote the mean vector and the covariance matrix of $\mathbf{o}$ which are compensated by a first order Vector Taylor Series (VTS) based approach [Mor96] with parameters $\boldsymbol{\mu}_{\mathbf{y},j}$, $\boldsymbol{\Sigma}_{\mathbf{y},j}$.

$$\begin{aligned} \mathbf{o} &= (\nabla_{\mathbf{y}} f) \mathbf{y} + (\nabla_{\mathbf{n}} f) \mathbf{n} + g(\mathbf{y}_0, \mathbf{n}_0) \\ g(\mathbf{y}_0, \mathbf{n}_0) &= f(\mathbf{y}_0, \mathbf{n}_0) - (\nabla_{\mathbf{y}} f) \mathbf{y}_0 - (\nabla_{\mathbf{n}} f) \mathbf{n}_0 \\ \boldsymbol{\mu}_{\mathbf{o},j} &= (\nabla_{\mathbf{y}} f) \boldsymbol{\mu}_{\mathbf{y},j} + (\nabla_{\mathbf{n}} f) \boldsymbol{\mu}_{\mathbf{n}} + g(\mathbf{y}_0, \mathbf{n}_0) \\ \boldsymbol{\Sigma}_{\mathbf{o},j} &\approx (\nabla_{\mathbf{y}} f)' \boldsymbol{\Sigma}_{\mathbf{y},j} (\nabla_{\mathbf{y}} f) + (\nabla_{\mathbf{n}} f)' \boldsymbol{\Sigma}_{\mathbf{n}} (\nabla_{\mathbf{n}} f) \end{aligned} \quad (5.118)$$

where $\mathbf{y}_0, \mathbf{n}_0$ are the first-order vector Taylor series expansion point and are set to $\boldsymbol{\mu}_{\mathbf{y},j}$, $\boldsymbol{\mu}_{\mathbf{n}}$, $j=1, \dots, M$, respectively in this chapter. It is suitable to perform VTS approximation at $(\boldsymbol{\mu}_{\mathbf{y},j}, \boldsymbol{\mu}_{\mathbf{n}})$, $j=1, \dots, M$, because the a posteriori probabilities $p(j | \mathbf{o}_t, \hat{\mathbf{n}}_{t|t})$ measure the contributions of all the operating points in the VTS approximations and weight the different linear approximations to result in a piecewise one. Note that in equation (5.118) biases are frame-dependent and different stochastic residuals for different frames may be introduced.

Since the distribution of the speech features in noise-tracking-only methods is described by a Gaussian mixture model without considering the temporal interrelations between close frames, each frame of data is enhanced independently. Without post-processing, this

can result in artifacts, such as sharp single frame transitions, that are not part of the original clean speech signal [Dro04]. This phenomenon can be seen in Figure 5.2. The residual noise has a great variance at the initial and final silences while the variance is smaller at non-masked speech parts. When speech is heavily masked by noise, the bias is large, while other parts exhibit a small bias.

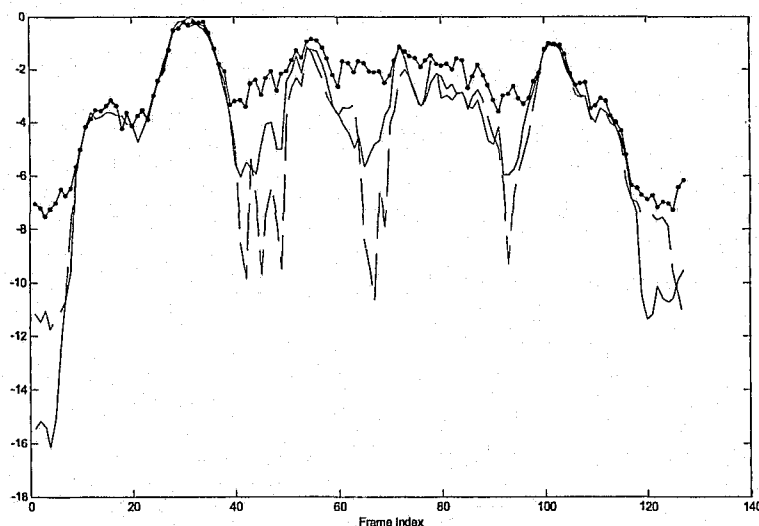


Fig 5.2 Typical temporal sequence of log-spectrum coefficients for clean speech (solid), noisy speech (dotted), compensated speech (dashed). C_8 is shown here.

5.5 Simulation Results and Analysis

5.5.1 Simulation Results on Database *Number* v1.3

Pilot experiments are first conducted on the database *Number* v1.3. The database description and experimental settings for the speech recognition engine are given in chapter 4. The estimated clean speech feature vectors will be mapped from the log-spectral domain into the MFCC domain using a DCT transform if they are processed in the Mel-scaled log-spectral domain, before being passed on directly to a speech recognizer for the recognition task.

With equation 5.7, clean speech feature vectors are modeled by a GMM model with 18 Gaussian mixtures. The GMM components have full covariance matrices if the features to be enhanced are in the log-spectral domain. Simulation results in terms of recognition accuracy for white noise and babble noise mixtures when speech is enhanced in the log-spectral domain are presented in Tables 5.1 and 5.2. For cepstrum vectors (MFCC), the GMM components have diagonal covariance matrices. For a fair comparison, we use a square DCT so that the cepstrum vectors have the same dimension as the log-spectral vectors. Simulation results in terms of recognition accuracy for white noise and babble noise mixtures when speech is enhanced in the cepstrum domain are presented in Tables 5.3 and 5.4.

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	50.6	73.4	88.0	93.0	76.3
Linearized Kalman filter	55.0	74.0	87.8	93.0	77.5
Extended Kalman filter	51.2	73.2	86.8	93.6	76.2
Unscented Kalman filter	56.0	74.6	88.6	94.0	78.3

Table 5. 1: Recognition accuracy (%) for speech mixed with white noise
when speech is enhanced in the log-spectrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	44.0	70.2	85.0	91.0	72.6
Linear Kalman filter	47.0	71.8	85.0	90.8	73.7
Extended Kalman filter	44.4	70.2	85.2	91.6	72.9
Unscented Kalman filter	46.2	72.4	85.2	91.0	73.7

Table 5. 2: Recognition accuracy (%) for speech mixed with babble noise
when speech is enhanced in the log-spectrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	50.6	73.4	88.0	93.0	76.3
Linearized Kalman filter	54.4	74.8	86.8	91.2	76.8
Extended Kalman filter	52.0	73.2	85.6	90.6	75.4
Unscented Kalman filter	54.8	74.6	87.4	91.8	77.2

Table 5.3: Recognition accuracy (%) for speech mixed with white noise
when speech is enhanced in the cepstrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	44.0	70.2	85.0	91.0	72.3
Linearized Kalman filter	47.2	72.4	87.0	89.6	74.1
Extended Kalman filter	44.4	71.4	84.8	88.6	72.3
Unscented Kalman filter	46.8	72.6	86.8	91.2	74.4

Table 5.4: Recognition accuracy (%) for speech mixed with babble noise
when speech is enhanced in the cepstrum domain

Since our corpus includes only 5-digit utterances, we use a grammar

`(([sil] $digit [sil] $digit [sil] $digit [sil] $digit [sil] $digit [sil]))`

instead of

`(([sil] <$digit [sil]>))`

which is used when the string length is unknown as in the *Aurora* database. The square brackets denote optional items and the angle braces denote one or more repetitions. We found that this experimental setting makes the *Number* database baseline recognition rates given in Table 5.1-5.4 without noise reduction much higher than the baseline rates of the *Aurora 2* database at low SNR.

The simulation results on the *Number* v1.3 database show that the Kalman filter approach with either the linearized Kalman filter, the extended Kalman filter or the unscented Kalman filter is able to provide a small improvement of the recognition performance for this database.

It was found that the extended Kalman filter cannot bring better results than approximating the observation equation (5.9) by a piecewise linear model and then using a linear Kalman filter. On the contrary, at low SNRs, the extended Kalman filter performs even a bit worse than a Kalman filter with a linearized state-space-model. Large variations of the parameter estimates along the time axis are responsible for this phenomenon. The estimated noise $\hat{n}$ tends to be biased, while here, since we use the true noise sequence for training, μ_n is the true value of the noise mean. Therefore, appropriately setting the Taylor expansion operating points at the mean vector of the clean speech mixture model and the mean vector of the noise brings better results.

The unscented Kalman Filter can resolve the Taylor-expansion-points problem of the extended Kalman filter and improve the approximation accuracy. For both types of noise, the UKF method showed its superiority to the EKF counterpart. At SNR higher than 10 dB where the speech is fairly clean, the recognition performance obtained with the EKF compensated speech is actually poorer than that obtained with the uncompensated noisy speech, while the enhancement algorithm continues to provide performance improvement with the UKF approach. However, the performance of the unscented Kalman filter is similar to the linearized Kalman filter. A possible reason is that the MMSE of speech is based on Taylor expansions, although the noise is tracked by a UKF method. Therefore, if we approximate the nonlinear acoustics model with appropriate Taylor expansion points, the Kalman filter can achieve equally good results as the UKF.

It is found that enhancement in the cepstrum domain does not result in any additional improvement in recognition accuracy. This is reasonable since the DCT transform is a linear transform and could not bring further performance improvement. However, since the covariance matrices of the clean speech and noise are diagonal, Kalman filtering algorithms could be implemented faster.

5.5.2 Simulation Results on the *Aurora* 2.0 database

After the pilot test, the experiments are then conducted on the *Aurora* 2.0 database. The database description and experimental settings for the speech recognition engine are given in chapter 4. We use the standard *Aurora* front-end settings. To conform to the observation model presented in (5.3), the feature generation was modified slightly from the reference implementation. In particular, we changed from using the spectral magnitude to using the power spectral density as the input to the Mel-frequency filter-bank to make equation (5.3) stand. Our settings are the same as those in [Dro04][Umb05]. The recognition rates (%) for clean speech are given in Table 5.5. The baseline results without noise processing are presented in Table 5.6.

All the experiments use the same global speech model, which is a GMM model with 16 Gaussian mixtures. For each noise type a specific noise model was built. The parameters for the noise model were trained on the first ten frames of the noisy utterances. The first ten frames are assumed to be speech free. All the experimental conditions were maintained the same as those in [Umb05]. As we found from the pilot test on the *Number* database that speech enhancement in the cepstral domain brought similar results to those in the log-spectral domain, in this section we tested the proposed algorithms in the log-spectral domain only.

Detailed recognition rates (%) for each of the four noise conditions and for each of the SNRs in the *Aurora* set-A are given in Tables 5.8-5.9. The Vector Taylor Series (VTS) algorithm [Mor96] reviewed in chapter 3 is used as a benchmark, since the VTS is the most influential speech feature enhancement algorithms. The VTS results in Table 5.7 were taken directly from [Myr03]. In the VTS, the non-linearity (5.9) is approximated by a 0-th order Taylor series approximation. An estimate of the noise is obtained and is then removed from the log-spectral feature vectors by using the MMSE method described in section 5.2.3. The results obtained by particle filters are also shown in Table 5.8 for the purposed of comparison. They were taken directly from [Umb05]. Our results using the proposed UKF approach are provided in Table 5.9.

	Subway	Babble	Car	Exhibition	Average
Clean	98.96	98.97	98.90	99.11	98.99

Table 5.5 Accuracy (%) , *Aurora 2*

	Subway	Babble	Car	Exhibition	Average
20dB	97.18	91.32	97.82	96.67	95.75
15dB	93.31	75.63	89.53	93.49	87.99
10dB	79.98	50.60	69.49	78.72	69.70
5dB	53.39	27.03	33.49	42.36	39.07
0dB	27.26	12.18	11.09	15.71	16.56
Average	70.22	51.35	60.28	65.39	61.81

Table 5.6 Accuracy (%) baselines without noise compensation, *Aurora 2*

	Subway	Babble	Car	Exhibition	Average
20dB	97.05	95.44	97.82	97.01	96.83
15dB	95.30	85.85	93.56	93.58	92.07
10dB	84.34	62.39	76.05	81.02	75.95
5dB	59.75	32.29	38.53	52.64	45.80
0dB	29.44	10.67	10.68	17.71	17.13
Average	73.12	57.32	63.33	68.39	65.55

Table 5.7 Accuracy (%) with speech enhanced by VTS – (results are from [Myr03])

	Subway	Babble	Car	Exhibition	Average
20dB	98.74	97.73	99.05	98.77	98.57
15dB	96.13	94.20	97.44	95.99	95.94
10dB	89.59	84.92	90.52	88.34	88.34
5dB	73.87	67.53	71.82	71.46	71.17
0dB	45.41	40.11	33.25	42.39	40.29
Average	80.74	76.89	78.41	79.39	78.85

Table 5.8 Accuracy (%) with speech enhanced by 100 Particle filters

(Results are from [Umb05])

	Subway	Babble	Car	Exhibition	Average
20dB	96.96	98.28	98.36	97.35	97.73
15dB	93.31	96.83	95.85	92.97	94.74
10dB	87.17	88.94	88.94	85.25	87.58
5dB	76.57	70.62	74.29	70.23	72.93
0dB	52.29	31.47	42.71	44.89	42.84
Average	81.26	77.23	80.03	78.14	79.16

Table 5.9 Accuracy (%) with speech enhanced by 16 Unscented Kalman filters

(Proposed in this chapter)

The simulation results for the *Aurora* 2.0 database are summarized in the following table. The results are averaged across 0 dB to 20 dB. It can be seen that the performance improvement provided by the enhancement algorithm is much more significant in this case.

	Subway	Babble	Car	Exhibition	Average
Baseline	70.22	51.35	60.28	65.39	61.81
VTs	73.12	57.32	63.33	68.39	65.55
Particle filter	80.74	76.89	78.41	79.39	78.85
Proposed UKF	81.26	77.23	80.03	78.14	79.16

Table 5.10 Summary of accuracy (%) on *Aurora* 2.0, set A

The results show that the noise tracking approaches with either the particle filter [Umb05] or the proposed Unscented Kalman filter clearly outperform the VTs method [Mor96]. Comparing the results of the UKF method with those obtained by a particle filter, a small performance improvement over the particle filtering approach is observed. Considering the complexity involved, we used only 16 UKF while 100 particle filters were used in [Umb05]. It has been shown in [Dro04] that with the increase in the number of Gaussian mixture components and the number of the corresponding filters, better results could be achieved. It is thus promising that if we increase the number of the Unscented Kalman filters to 100, we should obtain better results than the particle filters method.

We found that the performance improvement from using feature enhancement could be more easily observed from the *Aurora* database results than from the *Number* database results. One reason accounting for this is that, as explained before, the baseline recognition rate of the *Number* database is much higher. Another reason is that we used rather simple reference recognizer settings for the *Number* database: 3 HMM states were used to model one linguistic unit, while 16 states were used for one linguistic unit in the *Aurora* database testing. However, the results and conclusions based on the *Number* database still stand, because the algorithms are compared under the same unfavorable conditions.

In summary, from the simulation results, it is seen that the Kalman filter-based sequential estimation methods are effective at improving the recognition performance. The Unscented Kalman Filter is more appropriate (or at least as appropriate) for this application than the particle filtering approach. In the following chapters, we will further develop the Kalman filter-based speech enhancement approaches to compensate the noise in the speech feature domain.

5.6 Summary

A comparative study of filter-model-based noise-tracking-only methods was given in this chapter. A review of switching Kalman filtering was first given, which included the concept of switching state-space models, and an in-depth review of the approximate inference algorithms in switching state-space models, for which the exact inference is intractable. The prevailing noise-tracking idea for speech enhancement was then reviewed. Considering noise as the state variable to be tracked and whose observation is corrupted by speech, a switching Kalman filter can be used to estimate the noise, and then speech is enhanced by removing this estimated noise from the observation. The Extended Kalman Filter (EKF) algorithm and the Unscented Kalman Filter (UKF) algorithm were studied to compensate the nonlinear distortion caused by the environment noise in the feature domain. We revealed that the artifact problem associated with the noise-tracking-only approaches was brought about by the GMM speech model assumption. To further improve the performance of speech enhancement systems, a dynamic model taking the temporal interrelations between close frames of speech features into consideration will be introduced in the next chapter.

Chapter 6

Linear-Predictive-HMM for Speech Feature Enhancement

6.1 Introduction

In chapter 5 a noise-tracking-only method was reviewed, in which noise-tracking and clean speech estimation were two procedures carried out separately. However, it is believed that joint noise and speech tracking should yield a better enhancement result, as reported in [Dro04]. The major obstacle with the joint-speech-noise-tracking approach is how to represent the nonlinear and non-stationary time trajectory of speech features by a dynamic model.

In [Dro04], the time evolution of speech features is represented by a Switching Linear Dynamic Model (SLDM). The linear dynamic model is the simplest dynamic model with state evolving continuously through time, which can be described by the following equation:

$$\mathbf{x}_t = F\mathbf{x}_{t-1} + \mathbf{w}_t \quad (6.1)$$

Evolution from one time-state to the next is described by a linear transformation F and the addition of some noise. The benefit of using a linear dynamical system is the smooth trajectory modeling. Speech is a continuous process and ideally should be modeled as such. However, a linear dynamic model is not appropriate to describe the trajectories of speech features since the time movement of speech features is manifestly non-stationary due to the nature of speech production [Por82][Ken90]. To describe the complex dynamics of speech features, a more accurate model such as the switching linear dynamic model is desirable. The basic idea is that the model describes the dynamics of a complex, non-stationary process by switching among a set of linear dynamic models over time.

The standard hidden Markov model (HMM) is another popular and successful technique for modeling non-stationary time-series. As shown in Fig 6.1, the state variable in a

HMM is discrete and can be viewed as a switching variable between different stationary process regimes. The benefits of using a HMM include efficient training and recognition algorithms, in which automatic segmentation using the Viterbi algorithm has a very significant role.

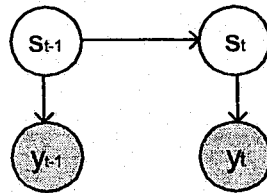


Fig 6.1 Graphical representation of an HMM with Gaussian output.

The shaded nodes are observed, and clear nodes are hidden

On the other hand, a major drawback with an HMM is that successive frames are assumed conditionally independent given the hidden state, as shown in Fig 6.1. It is not a good assumption since the speech feature vectors are strongly correlated [Dig92][Ost96]. This correlation between successive feature vectors is often called inter-frame correlation. Combining an HMM and a linear dynamic model may be able to address some shortcomings of this assumption [Por82] [Ken90]. The result is named as linear-predictive-HMM and it was applied for speech recognition in [Ken90]. The graphical representation of linear-predictive-HMM is shown in Figure 6.2. In Fig 6.2 (a), the linear-predictive-HMM has a first-order dependency, while Fig. 6.2 (b) shows a linear-predictive-HMM with a second-order evolution dimension. The shaded nodes are observed, and clear nodes are hidden. HMM states can be viewed as switching variables which choose the active parameters of the LDM. Therefore, linear-predictive-HMM is a SLDM that captures the smooth time evolution of the feature vectors conditioned on the hidden HMM state sequence.

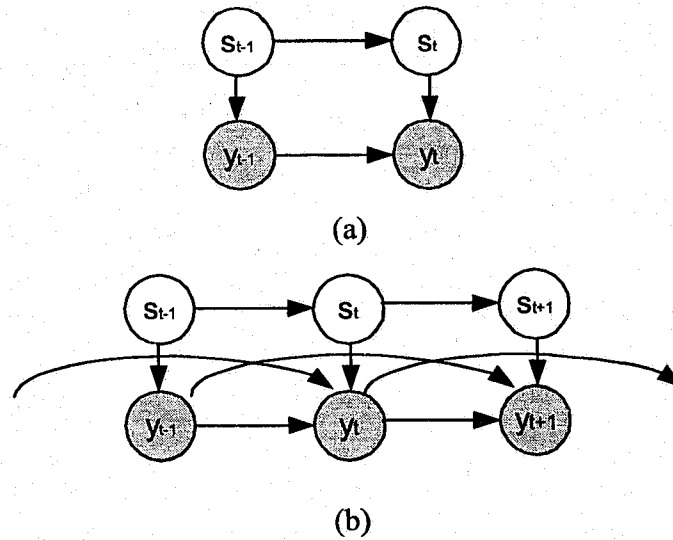


Fig. 6.2 Graphical representation of HMM-based switching LDM

In recent years, speech recognition using the SLDM in acoustic modeling has been studied in [Ken90][She94][Ros04]. However, the use of a switching linear dynamic model to represent speech in the context of speech feature enhancement was probably first proposed by Droppo et al [Dro04]. With a SLDM as the speech model and a noise model, Droppo et al estimate the clean speech component by a numerical integration method. Very recently, Kim et al [Kim05] use a SLDM with a similar structure to the one in Droppo's work, to model the speech and then estimate the clean speech component simultaneously with the noise component. The SLDM model in [Dro04] and [Kim05] has a structure which can be represented by the following figure.

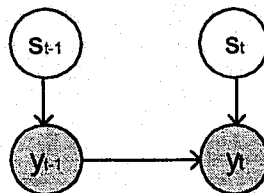


Fig.6.3 Graphical representation of the switching LDM in [Dro04] and [Kim05]

The SLDM shown in Fig. 6.3 also maintains the concept that as time progresses, the signal passes through several distinct discrete states. The difference between Fig. 6.2 and 6.3 is that the SLDM represented by Fig 6.3 assumes time dependence among the

continuous speech features in one state, but not among the discrete states. Therefore, it could be treated as a combination of a Gaussian mixture model (GMM) with a LDM. The linear-predictive-HMM represented by Fig. 6.2 takes further advantage of the temporal correlations among adjacent frames of the speech signal, by including transition probabilities among the discrete state variables. Compared with the GMM, the state transition probabilities of an HMM tend to eliminate single frame errors in the output. This advantage also makes HMM-based switching models have potentially greater descriptive power over the GMM-based switching models.

In this chapter, we propose the use of a linear-predictive-HMM as a model for speech feature enhancement instead of a model for speech recognition as in [Ken90]. By incorporating the speech model with a noise model into the state space equations, both the noise and clean speech components are estimated simultaneously during the speech enhancement. A filtering approach with the linear-predictive-HMM as the speech model permits the feature enhancement not only at the level of individual frames, but simultaneously at the level of local sequences of frames. Offsetting this advantage is the fact that exact inference in the SLDM-based Kalman filter is intractable. Approximate inference algorithms are required, which in turn complicates the filtering.

Our approach bears some similarity with the one proposed by Kim et al [Kim05]. However, our work was developed independently, and there are three important differences between our method and the method presented in [Kim05]:

First, the SLDM in [Kim05] uses a first-order linear predictive model to predict the t^{th} speech log-spectral vector. In our SLDM, we investigate the use of both first order and second order linear predictive models for log-spectral vector prediction. To the best of our knowledge, the explicit use of higher-order SLDM to model the feature vectors for speech enhancement has not been reported so far.

Second, the SLDM used in [Kim05] shares a similar structure to the one in Droppo's work [Dro04], while we use a different SLDM structure. Kim's SLDM is closely related

to the stochastic segment model [Dig93], in which the segments are assumed to be independent of one another. While in our SLDM, the speech vector is propagated over the segment boundaries, which should provide a better model for co-articulation.

Third, the inference process of the switching LDM parameters is different. In Kim's SLDM, the whole space of log spectra of clean speech is divided into a number of disjoint clusters. The statistics to estimate the SLDM parameters were computed based on a cluster index that was obtained by "hard" competition. We use a "soft" competition instead. Hence it is possible to estimate at each time t the responsibility of each model and therefore detect a mode transition.

It should be noted that, although linear-predictive-HMM can be used as an acoustic model for speech recognition, this model is only used for speech enhancement in this thesis. The enhanced speech feature vectors are then passed on to a speech recognizer for the recognition task. The speech recognizer is composed of the traditional HMMs with continuous Gaussian mixtures as reviewed in chapter 2. The number of hidden states in the linear-predictive-HMM could be understood as the number of LDMs in the SLDM, which has no connection with the number of HMM states used in the speech recognizer.

Another notable thing is that the state variable in a HMM is discrete, while the state variable in the LDMs and the noise model are continuous, these two types of state variables are considered differently in our applications. To distinguish them, we will label the discrete state as 'state (mode)' in the following sections, meaning different HMM states representing different dynamical "modes".

6.2 Linear-Predictive-HMM as a Speech Model

Linear-predictive-HMM can be trained efficiently in a maximum likelihood framework using the EM algorithm [Dem77]. Consider a Markov chain with M discrete hidden states (modes), $i, j \in (1, \dots, M)$ and a state (mode) transition matrix

$P(q_t = j | q_{t-1} = i) = a_{ij} = Z(i, j)$. Let $\mathbf{y}_t$ stand for the output of the process at time t given that the model is in state i at time t , $\mathbf{y}_t$ is defined as

$$\mathbf{y}_t = \boldsymbol{\mu}_{s_t=i} + \mathbf{B}_{1,s_t=i} \mathbf{y}_{t-1} + \dots + \mathbf{B}_{p,s_t=i} \mathbf{y}_{t-p} + \mathbf{e}_t \quad (6.2).$$

All the parameters of the linear dynamic model are time dependent via the discrete state (mode) variable s_t . If p is the autoregressive order and d is the dimension of $\mathbf{y}_t$, $\mathbf{y}_t$ and $\boldsymbol{\mu}$ are $d \times 1$ vectors, the $\mathbf{B}$'s are $d \times d$ matrices, and the residual $\mathbf{e}_t$ is a $d \times 1$ random vector having a Gaussian distribution with mean 0 and covariance matrix $\boldsymbol{\Sigma}_{s_t=i}$. It is assumed that the residual $\mathbf{e}_t$ at different times is independent. Equation 5.2 could be rewritten in the short-hand form as

$$\mathbf{y}_t = \boldsymbol{\mu}_{s_t=i} + \mathbf{B}_{s_t=i} \mathbf{x}_t + \mathbf{e}_t \quad (6.3)$$

here $\mathbf{x}_t$ is the dp -dimensional column vector $[\mathbf{y}_{t-1}, \dots, \mathbf{y}_{t-p}]^T$ and $\mathbf{B}_{s_t=i}$ is the $d \times dp$ matrix $(\mathbf{B}_{1,s_t=i} \dots \mathbf{B}_{p,s_t=i})$. The hidden filter model described in the time domain [She94] can be obtained by setting $d=1$ and $\boldsymbol{\mu} = 0$. For a stationary autoregressive process, the dc term $\boldsymbol{\mu}$ is necessarily 0; but for the non-stationary model there is no such restriction, and $\boldsymbol{\mu}$ is treated as a parameter to be trained in the same way as the regression coefficients. In this section, we show how to estimate the parameters of the model, namely $\mathbf{A}_i = \{\boldsymbol{\mu}_i, \mathbf{B}_i, \boldsymbol{\Sigma}_i\}$, as well as the transition probabilities Z of the hidden Markov chain and the initial probability π , from a sequence of observations $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T\}$ using an Expectation-Maximisation (EM) [Dem77] algorithm. For convenience, we use the compact notation $\lambda = (\mathbf{A}, Z, \pi)$ to indicate the complete parameter set of the model.

The EM algorithm requires us to maximize $Q(\lambda, \bar{\lambda})$, where $\bar{\lambda} = (\bar{\mathbf{A}}, \bar{Z}, \bar{\pi})$ is defined as the reestimated model. The joint likelihood of an observation sequence $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T\}$ and a hidden state (mode) sequence $\mathbf{q} = \{q_1, q_2, \dots, q_T\}$ can be written as

$$P_{\bar{\lambda}}(\mathbf{Y}, \mathbf{q}) = P_{\bar{\lambda}}(q_1) \prod_{t=2}^T P_{\bar{\lambda}}(q_t | q_{t-1}) \prod_{t=1}^T P_{\bar{\lambda}}(\mathbf{y}_t | q_t) \quad (6.4)$$

The joint log likelihood becomes

$$\log P_{\bar{\lambda}}(\mathbf{Y}, \mathbf{q}) = \log P_{\bar{\lambda}}(q_1) + \sum_{t=2}^T \log P_{\bar{\lambda}}(q_t | q_{t-1}) + \sum_{t=1}^T \log P_{\bar{\lambda}}(y_t | q_t) \quad (6.5)$$

The EM algorithm requires us to maximise

$$\begin{aligned} Q(\lambda, \bar{\lambda}) &= \sum_{\mathbf{q}} P_{\lambda}(\mathbf{q} | \mathbf{Y}) \log P_{\bar{\lambda}}(\mathbf{Y}, \mathbf{q}) \\ &= \sum_{\mathbf{q}} P_{\lambda}(\mathbf{q} | \mathbf{Y}) \left(\log P_{\bar{\lambda}}(q_1) + \sum_{t=2}^T \log P_{\bar{\lambda}}(q_t | q_{t-1}) + \sum_{t=1}^T \log P_{\bar{\lambda}}(y_t | q_t) \right) \end{aligned} \quad (6.6)$$

where $\sum_{\mathbf{q}}$ denotes a sum over all the possible hidden state (mode) sequences. The above

equation consists of three terms; the first for the initial probability, π_i , the second for the state (mode) transition probability parameters, Z and the third for the observation model parameters $\mathbf{A}$. It then can be rewritten as

$$Q(\lambda, \bar{\lambda}) = Q(\pi, \bar{\pi}) + Q(Z, \bar{Z}) + Q(\mathbf{A}, \bar{\mathbf{A}}) \quad (6.7)$$

$$Q(\pi, \bar{\pi}) = \sum_i P_{\lambda}(q_1 = i | \mathbf{Y}) \log P_{\bar{\pi}}(q_1 = i) \quad (6.8)$$

$$Q(Z, \bar{Z}) = \sum_i \sum_j \sum_t P_{\lambda}(q_t = j, q_{t-1} = i | \mathbf{Y}) \log P_{\bar{Z}}(q_t | q_{t-1}) \quad (6.9)$$

$$Q(\mathbf{A}, \bar{\mathbf{A}}) = \sum_i \sum_t P_{\lambda}(q_t = i | \mathbf{Y}) \log P_{\bar{\mathbf{A}}}(y_t | q_t) \quad (6.10)$$

The three terms in equation (6.7) can be maximized separately, giving rise to parameter update equations for each different part of the model.

The update formula for the initial state (mode) distribution parameters and the transition matrix can be found by maximizing equation (6.8) and (6.9) which gives

$$\bar{\pi}_i = \gamma_i(i) \quad (6.11)$$

$$\bar{a}_{ij} = \frac{\sum_t \xi_t(i, j)}{\sum_t \gamma_t(i)} \quad (6.12)$$

where

$$\gamma_t(i) = P(q_t = i | \mathbf{Y}), \quad t = 1, \dots, T \quad (6.13)$$

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j | \mathbf{Y}) \quad (6.14)$$

They are defined in the same way as in a standard HMM, and can be calculated by the forward-backward algorithm as described in chapter 2.

-Observation model

An update formula for the observation model parameters can be found by maximizing

$$Q(\mathbf{A}, \bar{\mathbf{A}}) = \sum_i \sum_t P_\lambda(q_t = i | \mathbf{Y}) \log P_{\bar{\lambda}}(\mathbf{y}_t | q_t) \quad (6.15)$$

For a Gaussian observation model

$$\Pr(\mathbf{y}_t | q_t = i) = (2\pi)^{-d/2} |\bar{\Sigma}_i|^{-1/2} \exp(-\frac{1}{2}[\mathbf{y}_t - \bar{\mathbf{B}}_i \mathbf{x}_t - \bar{\boldsymbol{\mu}}_i]' \bar{\Sigma}_i^{-1} [\mathbf{y}_t - \bar{\mathbf{B}}_i \mathbf{x}_t - \bar{\boldsymbol{\mu}}_i]) \quad (6.16)$$

By substituting this into equation (5.15) and dropping terms that are independent, the objective function becomes

$$Q(\mathbf{A}, \bar{\mathbf{A}}) = -\frac{1}{2} \sum_t \sum_i \gamma_t(i) \log |\bar{\Sigma}_i| - \frac{1}{2} \sum_t \sum_i \gamma_t(i) E_{ii}[(\mathbf{y}_t - \bar{\mathbf{B}}_i \mathbf{x}_t - \bar{\boldsymbol{\mu}}_i)' \bar{\Sigma}_i^{-1} (\mathbf{y}_t - \bar{\mathbf{B}}_i \mathbf{x}_t - \bar{\boldsymbol{\mu}}_i)] \quad (6.17)$$

The following expected sufficient statistics are introduced [Mur96b]:

$$\begin{aligned} S_{YY,i} &= \sum_t \gamma_t(i) E_{ii}[\mathbf{y}_t \mathbf{y}_t'] \\ S_{Y'Y,i} &= \sum_t \gamma_t(i) E_{ii}[\mathbf{y}_t' \mathbf{y}_t] \\ S_{Y,i} &= \sum_t \gamma_t(i) E_{ii}[\mathbf{y}_t] \\ S_{XX',i} &= \sum_t \gamma_t(i) E_{ii}[\mathbf{x}_t \mathbf{x}_t'] \\ S_{X,i} &= \sum_t \gamma_t(i) E_{ii}[\mathbf{x}_t] \\ S_{X'Y,i} &= \sum_t \gamma_t(i) E_{ii}[\mathbf{x}_t \mathbf{y}_t'] \\ S_{YX',i} &= \sum_t \gamma_t(i) E_{ii}[\mathbf{y}_t \mathbf{x}_t'] \end{aligned} \quad (6.18).$$

Maximizing $Q(\mathbf{A}, \bar{\mathbf{A}})$ with respect to $\mathbf{B}_i$ and setting the derivative to 0

$$\begin{aligned} \frac{\partial}{\partial \mathbf{B}_i} Q &= -\frac{1}{2} \sum_t \gamma_t(i) \cdot -2 \bar{\Sigma}_i^{-1} E_{ii}[(\mathbf{y}_t - \bar{\mathbf{B}}_i \mathbf{x}_t - \bar{\boldsymbol{\mu}}_i) \mathbf{x}_t'] \\ &= \bar{\Sigma}_i^{-1} \left\{ \left(\sum_t \gamma_t(i) E_{ii}[\mathbf{y}_t \mathbf{x}_t'] \right) - \bar{\mathbf{B}}_i \left(\sum_t \gamma_t(i) E_{ii}[\mathbf{x}_t \mathbf{x}_t'] \right) - \bar{\boldsymbol{\mu}}_i \left(\sum_t \gamma_t(i) E_{ii}[\mathbf{x}_t'] \right) \right\} \\ &= \bar{\Sigma}_i^{-1} \{ S_{YX',i} - \bar{\mathbf{B}}_i S_{XX',i} - \bar{\boldsymbol{\mu}}_i S_{X,i}' \} \\ \frac{\partial}{\partial \mathbf{B}_i} Q &= 0 \end{aligned} \quad (6.19);$$

the vector equation is obtained

$$\bar{\mathbf{B}}_i = (S_{YX',i} - \bar{\boldsymbol{\mu}}_i S'_{X,i}) S_{XX',i}^{-1} \quad (6.20)$$

Likewise, setting the derivatives of Q with respect to $\boldsymbol{\mu}_i$ equal to 0

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\mu}_i} Q &= -\frac{1}{2} \sum_i \gamma_i (i) \cdot -2 \boldsymbol{\Sigma}_i^{-1} E_{ii}[(\mathbf{y}_i - \bar{\mathbf{B}}_i \mathbf{x}_i - \bar{\boldsymbol{\mu}}_i)] \\ &= \bar{\boldsymbol{\Sigma}}_i^{-1} \{(\sum_i \gamma_i (i) E_{ii}[\mathbf{y}_i]) - \bar{\mathbf{B}}_i (\sum_i \gamma_i (i) E_{ii}[\mathbf{x}_i]) - \bar{\boldsymbol{\mu}}_i (\sum_i \gamma_i (i))\} \\ &= \bar{\boldsymbol{\Sigma}}_i^{-1} \{S_{Y,i} - \bar{\mathbf{B}}_i S_{X,i} - \bar{\boldsymbol{\mu}}_i \gamma_i\} \\ \frac{\partial}{\partial \boldsymbol{\mu}_i} Q &= 0 \end{aligned} \quad (6.21)$$

another vector equation is obtained

$$\bar{\boldsymbol{\mu}}_i = \frac{S_{Y,i} - \bar{\mathbf{B}}_i S_{X,i}}{\gamma_i} \quad (6.22)$$

where $\gamma_i = \sum_i \gamma_i (i)$. The new estimates for $\bar{\boldsymbol{\mu}}_i$ and $\bar{\mathbf{B}}_i$ can now be obtained by solving the pair of simultaneous equations.

$$\begin{aligned} \bar{\mathbf{B}}_i &= (S_{YX',i} - \bar{\boldsymbol{\mu}}_i S'_{X,i}) S_{XX',i}^{-1} \\ \bar{\boldsymbol{\mu}}_i &= \frac{S_{Y,i} - \bar{\mathbf{B}}_i S_{X,i}}{\gamma_i} \end{aligned} \quad (6.23)$$

Setting the derivatives of $Q(\mathbf{A}, \bar{\mathbf{A}})$ with respect to $\boldsymbol{\Sigma}_i$ gives the re-estimation formula as

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\Sigma}_i^{-1}} Q &= \frac{1}{2} (\sum_i \gamma_i (i)) \bar{\boldsymbol{\Sigma}}_i - \frac{1}{2} \sum_i \gamma_i (i) E_{ii}(\mathbf{y}_i - \bar{\mathbf{B}}_i \mathbf{x}_i - \bar{\boldsymbol{\mu}}_i)(\mathbf{y}_i - \bar{\mathbf{B}}_i \mathbf{x}_i - \bar{\boldsymbol{\mu}}_i)' = 0 \\ \bar{\boldsymbol{\Sigma}}_i &= \frac{1}{\gamma_i} \sum_i \gamma_i (i) E_{ii}(\mathbf{y}_i \mathbf{y}_i' - \mathbf{y}_i \mathbf{x}_i' \bar{\mathbf{B}}_i' - \mathbf{y}_i \bar{\boldsymbol{\mu}}_i' - \bar{\mathbf{B}}_i \mathbf{x}_i \mathbf{y}_i' + \bar{\mathbf{B}}_i \mathbf{x}_i \mathbf{x}_i' \bar{\mathbf{B}}_i' + \bar{\mathbf{B}}_i \mathbf{x}_i \bar{\boldsymbol{\mu}}_i' - \bar{\boldsymbol{\mu}}_i \mathbf{y}_i' + \bar{\boldsymbol{\mu}}_i \mathbf{x}_i' \bar{\mathbf{B}}_i' + \bar{\boldsymbol{\mu}}_i \bar{\boldsymbol{\mu}}_i') \end{aligned} \quad (6.24)$$

Since we have

$$\bar{\boldsymbol{\mu}}_i = \frac{S_{Y,i} - \bar{\mathbf{B}}_i S_{X,i}}{\gamma_i} \quad (6.25)$$

equation (6.24) is simplified further to

$$\bar{\boldsymbol{\Sigma}}_i = \frac{1}{\gamma_i} (S_{YY',i} - S_{YX',i} \bar{\mathbf{B}}_i' - \bar{\mathbf{B}}_i S_{XY',i} + \bar{\mathbf{B}}_i S_{XX',i} \bar{\mathbf{B}}_i' - \gamma_i \bar{\boldsymbol{\mu}}_i \bar{\boldsymbol{\mu}}_i') \quad (6.26)$$

Throughout the above analysis, we have assumed that the model is trained using a single token. However, a left-to-right model cannot, in general, be reliably trained with a single token. The multiple token training case analysis is similarly derived. Suppose we have L tokens $\mathbf{Y}^{(1)}, \dots, \mathbf{Y}^{(L)}$ of length $T^{(1)}, \dots, T^{(L)}$, we replace each “ t (time) summation” by a “ L

(token) summation and t (time) summation”: $\sum_t \Rightarrow \sum_{l=1}^L \sum_t^{T^{(l)}}$. The subsequent analysis is exactly the same as discussed previously.

A linear-predictive-HMM model trained using the EM algorithm is guaranteed to reach only a local maximum likelihood solution. Because there are many local maxima, such models are sensitive to how they are initialized. For this reason, we need to initialize the model rather carefully. Both the HMM and the linear dynamical model must be initialized. The key point is to start with a good partition of the data, each being modeled by a linear dynamic model [Gha98]. The initialization algorithm is summarized as follows:

1. A fuzzy C-means clustering technique [Bez81] is applied to cluster all the feature vectors into M segments. The process calculates the cluster membership degree $\alpha_j(t)$, and updates the cluster centers V_j iteratively to make the following objective function reach a minimum:

$$o = \sum_{j=1}^M \sum_{t=1}^T [\alpha_j(t)]^2 \|D_t - V_j\| \quad (6.27)$$

where $D_t = [\mathbf{y}_t; \mathbf{y}_{t-1}]$. The fuzzy membership degree to cluster j at a time point t $\alpha_j(t)$ can be interpreted as to what degree the data point belongs to a specific state (mode) j . Here the fuzzy-C clustering algorithm is used instead of the “hard” clustering, since it gives us the fuzzy membership degree $\alpha_j(t)$ used in step 2.

2. The aforementioned EM algorithm is performed to obtain the initial estimate for the LDM parameters in cluster j using formulas (6.23), (6.26) with $\alpha_j(t)$ instead of $\gamma_t(i)$.

3. Initial probability and transition probability are set to be $1/M$.

Now, the EM algorithm can be summarized as follows:

Step 1: Initialize $\lambda = (\mathbf{A}, Z, \pi)$

Step 2: Find the expected state (mode) occupancy: $\gamma_t(i) = P(q_t = i | \mathbf{Y})$, $t = 1, \dots, T$

Step 3: A new set of parameters $\bar{\lambda} = (\bar{\mathbf{A}}, \bar{\mathbf{Z}}, \bar{\pi})$ is found that maximizes the expected log-likelihood using the update formulas (6.13), (6.14), (6.23) and (6.26).

Step 4: Let $\lambda = \bar{\lambda}$; if convergence has not been achieved, go to step 2.

6.3 Joint-Speech-Noise-Tracking Based on Linear-Predictive-HMM

Given the linear-predictive-HMM as the speech model, we are going to develop a sequential approach to estimate the clean speech components from noisy observations. What distinguishes the present approach from the approach presented in chapter 5 is that both the clean speech components and the noise components are estimated simultaneously. The major obstacle to use the switching LDM for enhancement is the computational burden that it brings. Optimal minimum mean squared error estimators involve a bank of filters tuned to all the possible parameter histories, which makes the cost in computations grow exponentially with data length as we have explained in chapter 5. Tractable, approximate inference algorithms are therefore required. We will show how the Interacting Multiple Model (IMM) algorithm and the first order generalized pseudo Bayesian (GPB1) scheme can be adapted to the nonlinear state-space model of feature dynamics presented above, to provide a sub-optimal approximation solution. As a basis for these algorithms, we first present a nonlinear state-space model for noisy speech, clean speech, and noise in the feature domain.

The nonlinear acoustic model in the log spectrum feature space is given as:

$$\mathbf{o}_t = \mathbf{y}_t + \log(\mathbf{I} + \exp(\mathbf{n}_t - \mathbf{y}_t)) \quad (6.28)$$

where $\mathbf{o}_t$, $\mathbf{y}_t$ and $\mathbf{n}_t$ are the d -dimensional log spectrum of noisy speech, the hypothetical clean speech and noise at the t^{th} frame, respectively. In the cepstral feature space, it becomes

$$\mathbf{o}_t = \mathbf{y}_t + \mathbf{C} \log(\mathbf{I} + \exp(\mathbf{C}^{-1}(\mathbf{n}_t - \mathbf{y}_t))) \quad (6.29)$$

where $\mathbf{o}_t$, $\mathbf{y}_t$ and $\mathbf{n}_t$ represent the d -dimensional cepstrum of noisy speech, the hypothetical clean speech and noise at the t^{th} frame, respectively. Taking into account the time-varying characteristics of the background noise, we model the sequence of noise features as the

output of a first-order auto-regressive (AR) system excited by a zero mean Gaussian process $\mathbf{v}$ with a covariance matrix $\mathbf{Q}_n$ as [Raj04]:

$$\mathbf{n}_t = \mathbf{A}\mathbf{n}_{t-1} + \boldsymbol{\mu}_n + \mathbf{v}_t \quad (6.30)$$

6.3.1 State-Space for Joint-Speech-Noise-Tracking Method

Combining (6.2) with (6.28) and (6.30) leads to a nonlinear state space model in the log-spectral domain conditioned on $s_t = j$ as:

$$\mathbf{z}_t = \boldsymbol{\Phi}_j \mathbf{z}_{t-1} + \bar{\mathbf{u}}_j + \bar{\mathbf{e}}_{t,j} \quad (6.31)$$

$$\mathbf{o}_t = f(\mathbf{z}_t) = \mathbf{y}_t + \log(\mathbf{I} + \exp(\mathbf{n}_t - \mathbf{y}_t)) \quad (6.32)$$

The nonlinear state-space model in the cepstrum (MFCC) domain conditioned on $s_t = j$ is

$$\mathbf{z}_t = \boldsymbol{\Phi}_j \mathbf{z}_{t-1} + \bar{\mathbf{u}}_j + \bar{\mathbf{e}}_{t,j} \quad (6.33)$$

$$\mathbf{o}_t = f(\mathbf{z}_t) = \mathbf{y}_t + \mathbf{C} \log(\mathbf{I} + \exp(\mathbf{C}^{-1}(\mathbf{n}_t - \mathbf{y}_t))) \quad (6.34)$$

with the following definitions:

$$\mathbf{z}_t = \begin{bmatrix} \mathbf{y}_t^T & \mathbf{n}_t^T \end{bmatrix}^T \quad \bar{\mathbf{e}}_{t,j} = \begin{bmatrix} \mathbf{e}_{t,j}^T & \mathbf{v}_t^T \end{bmatrix}^T \quad (6.35)$$

$$\boldsymbol{\Phi}_j = \begin{bmatrix} \mathbf{B}_j & \mathbf{0} \\ \mathbf{0} & \mathbf{A} \end{bmatrix} \quad \bar{\mathbf{u}}_j = \begin{bmatrix} \boldsymbol{\mu}_j \\ \boldsymbol{\mu}_n \end{bmatrix} \quad \mathbf{Q}_j = \begin{bmatrix} \boldsymbol{\Sigma}_j & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_n \end{bmatrix} \quad (6.36)$$

The state-space model for linear-predictive-HMM with second-order dependency in the log-spectral domain is given as:

$$\mathbf{z}_t = \boldsymbol{\Phi}_j \mathbf{z}_{t-1} + \mathbf{u}_j + G\bar{\mathbf{e}}_{t,j} \quad (6.37)$$

$$\mathbf{o}_t = f(\mathbf{z}_t) = F\bar{\mathbf{y}}_t + \log(\mathbf{I} + \exp(\mathbf{n}_t - F\bar{\mathbf{y}}_t)) \quad (6.38)$$

In the cepstrum domain, it becomes

$$\mathbf{z}_t = \boldsymbol{\Phi}_j \mathbf{z}_{t-1} + \mathbf{u}_j + G\bar{\mathbf{e}}_{t,j} \quad (6.39)$$

$$\mathbf{o}_t = f(\mathbf{z}_t) = F\bar{\mathbf{y}}_t + \mathbf{C} \log(\mathbf{I} + \exp(\mathbf{C}^{-1}(\mathbf{n}_t - F\bar{\mathbf{y}}_t))) \quad (6.40)$$

with

$$\mathbf{z}_t = \begin{bmatrix} \bar{\mathbf{y}}_t \\ \mathbf{n}_t \end{bmatrix} \quad \bar{\mathbf{y}}_t = \begin{bmatrix} \mathbf{y}_t \\ \mathbf{y}_{t-1} \end{bmatrix} \quad \bar{\mathbf{e}}_{t,j} = \begin{bmatrix} \mathbf{e}_{t,j} & \mathbf{v}_t \end{bmatrix}^T$$

$$\Phi_j = \begin{bmatrix} C_j & 0 \\ 0 & A \end{bmatrix} \quad \mathbf{u}_j = [\boldsymbol{\mu}_j \quad 0 \quad \boldsymbol{\mu}_n]^T \quad (6.41)$$

$$C_j = \begin{bmatrix} B_j \\ I \end{bmatrix} \quad G = \begin{bmatrix} G_Y & 0 \\ 0 & I \end{bmatrix} \quad Q_j = \begin{bmatrix} \Sigma_j & 0 \\ 0 & Q_n \end{bmatrix}$$

$$F = [I \quad 0] \quad G_Y = [I \quad 0]^T$$

Both noise and clean speech components are incorporated in the state space, and are estimated simultaneously.

6.3.2 Switching Kalman Filter with the GPB1 algorithm

In this section, we will show how to adapt the first order generalized pseudo Bayesian scheme to the nonlinear state-space model of feature dynamics presented above. Let us define

$$\hat{\mathbf{z}}_{t-1|t-1} = E(\mathbf{z}_{t-1} | \mathbf{o}_{1:t-1}) \quad (6.42)$$

$$\hat{\mathbf{V}}_{t-1|t-1} = \text{cov}(\mathbf{z}_{t-1} | \mathbf{o}_{1:t-1}) \quad (6.43)$$

$$\hat{\mathbf{z}}_{it}^j = E[\mathbf{z}_t | \mathbf{o}_{1:t}, s_t = j] \quad (6.44)$$

$$\hat{\mathbf{V}}_{it}^j = \text{cov}[\mathbf{z}_t | \mathbf{o}_{1:t}, s_t = j] \quad (6.45)$$

$$L_t^j = p(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) \quad (6.46)$$

in which $\mathbf{o}_{1:t} = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_t\}$ denotes the noisy speech observation sequence up to time t .

Given a noisy speech feature vector $\mathbf{o}_t$ ($t=1, \dots, T$), $\hat{\mathbf{z}}_{it}^j$ is found from the GPB1 algorithm using a bank of Kalman filters with the following steps performed in sequence:

for the j th ($j=1, \dots, M$) Kalman filter:

$$(\hat{\mathbf{z}}_{it}^j, \mathbf{V}_{it}^j, L_t^j) = KF(\hat{\mathbf{z}}_{t-1|t-1}, \hat{\mathbf{V}}_{t-1|t-1}, \mathbf{o}_t; \Phi_j, \mathbf{u}_j, Q_j) \quad (6.47)$$

The model probabilities $p(s_t = j | \mathbf{o}_{1:t})$ are updated according to

$$\begin{aligned} p(s_t = j | \mathbf{o}_{1:t}) &= \frac{1}{\Omega} \sum_i p(s_{t-1} = i, s_t = j, \mathbf{o}_t | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{\Omega} \sum_i p(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) p(s_{t-1} = i, s_t = j | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{\Omega} \sum_i p(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) p(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) p(s_{t-1} = i | \mathbf{o}_{1:t-1}) \end{aligned} \quad (6.48)$$

where Ω is a scaling factor equal to

$$\Omega = \sum_j \sum_i p(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) p(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) p(s_{t-1} = i | \mathbf{o}_{1:t-1}) \quad (6.49)$$

The outputs of all filters are merged into a single estimate $\hat{\mathbf{z}}_{t|t}$ and $\hat{\mathbf{V}}_{t|t}$ at time t according to:

$$\hat{\mathbf{z}}_{t|t} = \sum_j p(s_t = j | \mathbf{o}_{1:t}) \hat{\mathbf{z}}_{t|t}^j \quad (6.50)$$

$$\hat{\mathbf{V}}_{t|t} = \sum_j p(s_t = j | \mathbf{o}_{1:t}) \hat{\mathbf{V}}_{t|t}^j + \sum_j p(s_t = j | \mathbf{o}_{1:t}) (\hat{\mathbf{z}}_{t|t}^j - \hat{\mathbf{z}}_{t|t}) (\hat{\mathbf{z}}_{t|t}^j - \hat{\mathbf{z}}_{t|t})^T \quad (6.51)$$

6.3.3 Switching Kalman Filter with the IMM algorithm

In this section, we will show how to adapt the IMM scheme to the nonlinear state-space model of feature dynamics presented in section 6.3.1. In the IMM algorithm, there are M filters, each of which is supplied with a different input. Let us define

$$\hat{\mathbf{z}}_{t-1|t-1} = E(\mathbf{z}_{t-1} | \mathbf{o}_{1:t-1}) \quad (6.52)$$

$$\hat{\mathbf{V}}_{t-1|t-1} = \text{cov}(\mathbf{z}_{t-1} | \mathbf{o}_{1:t-1}) \quad (6.53)$$

$$\tilde{\mathbf{z}}_{t-1|t-1}^j = E(\mathbf{z}_{t-1} | s_t = j, \mathbf{o}_{1:t-1}) \quad (6.54)$$

$$\tilde{\mathbf{V}}_{t-1|t-1}^j = \text{cov}(\mathbf{z}_{t-1} | s_t = j, \mathbf{o}_{1:t-1}) \quad (6.55)$$

$$W_{t-1|t-1}^{ij} = \Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t-1}) \quad (6.56)$$

$$L_t^j = p(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) \quad (6.57)$$

in which $\mathbf{o}_{1:t} = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_t\}$ denotes the noisy speech observation sequence up to time t .

The algorithm mixes the estimates according to the Markov transition probability, in order to allow the system to react to changes in the model in force. In that way, the input to the j^{th} filter becomes the best estimate of the state at time instant $t-1$, conditioned on the event that model j is in force at time instant t (the new sample time). $\tilde{\mathbf{z}}_{t-1|t-1}^j$ and $\tilde{\mathbf{V}}_{t-1|t-1}^j$ are then obtained according to

$$\tilde{\mathbf{z}}_{t-1|t-1}^j = \sum_i W_{t-1|t-1}^{ij} \hat{\mathbf{z}}_{t-1|t-1}^i \quad (6.58)$$

$$\begin{aligned} \tilde{\mathbf{V}}_{t-1|t-1}^j &= \sum_i W_{t-1|t-1}^{ij} \hat{\mathbf{V}}_{t-1|t-1}^i \\ &\quad + \sum_i W_{t-1|t-1}^{ij} (\hat{\mathbf{z}}_{t-1|t-1}^i - \tilde{\mathbf{z}}_{t-1|t-1}^j) (\hat{\mathbf{z}}_{t-1|t-1}^i - \tilde{\mathbf{z}}_{t-1|t-1}^j)^T \end{aligned} \quad (6.59)$$

The mixing probability $W_{t-1|t-1}^{ij} = p(s_{t-1}=i | s_t=j, \mathbf{o}_{1:t-1})$ is computed recursively with Bayes' rule

$$\begin{aligned} W_{t-1|t-1}^{ij} &= p(s_{t-1}=i | s_t=j, \mathbf{o}_{1:t-1}) \\ &= \frac{p(s_t=j | s_{t-1}=i, \mathbf{o}_{1:t-1}) p(s_{t-1}=i | \mathbf{o}_{1:t-1})}{\sum_i p(s_t=j | s_{t-1}=i, \mathbf{o}_{1:t-1}) p(s_{t-1}=i | \mathbf{o}_{1:t-1})} \end{aligned} \quad (6.60)$$

Given a noisy speech feature vector $\mathbf{o}_t$ ($t=1, \dots, T$), $\hat{\mathbf{z}}_{t|t}$ is found from the interacting multiple model algorithm using a bank of Kalman filters with the following steps performed in sequence:

for the j th ($j=1, \dots, M$) Kalman filter:

$$(\hat{\mathbf{z}}_{t|t}^j, \mathbf{V}_{t|t}^j, L_t^j) = KF(\tilde{\mathbf{z}}_{t-1|t-1}^j, \tilde{\mathbf{V}}_{t-1|t-1}^j, \mathbf{o}_t; \Phi_j, \mathbf{u}_j, \mathbf{Q}_j) \quad (6.61)$$

The model probabilities $p(s_t=j | \mathbf{o}_{1:t})$ are updated according to

$$\begin{aligned} p(s_t=j | \mathbf{o}_{1:t}) &= \frac{1}{\Omega} \sum_i p(s_{t-1}=i, s_t=j, \mathbf{o}_t | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{\Omega} \sum_i p(\mathbf{o}_t | s_{t-1}=i, s_t=j, \mathbf{o}_{1:t-1}) p(s_{t-1}=i, s_t=j | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{\Omega} \sum_i p(\mathbf{o}_t | s_{t-1}=i, s_t=j, \mathbf{o}_{1:t-1}) p(s_t=j | s_{t-1}=i, \mathbf{o}_{1:t-1}) p(s_{t-1}=i | \mathbf{o}_{1:t-1}) \end{aligned} \quad (6.62)$$

where Ω is a scale factor.

$$\Omega = \sum_j \sum_i p(\mathbf{o}_t | s_{t-1}=i, s_t=j, \mathbf{o}_{1:t-1}) p(s_t=j | s_{t-1}=i, \mathbf{o}_{1:t-1}) p(s_{t-1}=i | \mathbf{o}_{1:t-1}) \quad (6.63)$$

The estimated output at time t is calculated according to

$$\hat{\mathbf{z}}_{t|t} = \sum_j p(s_t=j | \mathbf{o}_{1:t}) \hat{\mathbf{z}}_{t|t}^j \quad (6.64)$$

6.3.4 Extended Kalman Filter for Joint-Speech-Noise-Tracking

If the extended Kalman filter (EKF) algorithm is used in equation (6.47), the algorithm becomes

$$(\hat{\mathbf{z}}_{t|t}^j, \mathbf{V}_{t|t}^j, L_t^j) = EKF(\hat{\mathbf{z}}_{t-1|t-1}^j, \hat{\mathbf{V}}_{t-1|t-1}^j, \mathbf{o}_t; \Phi_j, \mathbf{u}_j, \mathbf{Q}_j),$$

The $EKF()$ function is defined as:

$$\hat{\mathbf{z}}_{t|t-1}^j = \Phi_j \hat{\mathbf{z}}_{t-1|t-1}^j + \mathbf{u}_j \quad (6.65)$$

$$\hat{\mathbf{V}}_{t|t-1}^j = \Phi_j \hat{\mathbf{V}}_{t-1|t-1}^j \Phi_j^T + \mathbf{Q}_j \quad (6.66)$$

$$\mathbf{K}_t^j = \hat{\mathbf{V}}_{t|t-1}^j \mathbf{H}^T (\mathbf{H} \hat{\mathbf{V}}_{t|t-1}^j \mathbf{H} + \mathbf{R})^{-1} \quad (6.67)$$

$$\hat{\mathbf{z}}_{t|t}^j = \hat{\mathbf{z}}_{t|t-1}^j + \mathbf{K}_t^j (\mathbf{o}_t - f(\hat{\mathbf{z}}_{t|t-1}^j)) \quad (6.68)$$

$$\hat{\mathbf{v}}_{t|t}^j = (\mathbf{I} - \mathbf{K}_t^j \mathbf{H}) \hat{\mathbf{v}}_{t|t-1}^j \quad (6.69)$$

$$\text{where } \mathbf{H} = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{y}} \Big|_{(\hat{\mathbf{z}}_{t|t-1}^j)} & \frac{\partial f}{\partial \mathbf{n}} \Big|_{(\hat{\mathbf{z}}_{t|t-1}^j)} \end{bmatrix}$$

$$L_t^j = N(f(\hat{\mathbf{z}}_{t|t-1}^j), \mathbf{H} \hat{\mathbf{v}}_{t|t-1}^j \mathbf{H}^T) \quad (6.70).$$

If the speech enhancement is in the log-spectral domain, the derivative matrix is

$$\mathbf{H} = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{y}} \Big|_{(\hat{\mathbf{z}}_{t|t-1}^j)} & \frac{\partial f}{\partial \mathbf{n}} \Big|_{(\hat{\mathbf{z}}_{t|t-1}^j)} \end{bmatrix}, \text{ where}$$

$$\frac{\partial f}{\partial \mathbf{y}} = \text{diag} \left(\frac{1}{1 + \exp(\mathbf{n} - \mathbf{y})} \right) \quad (6.71)$$

$$\frac{\partial f}{\partial \mathbf{n}} = \text{diag} \left(1 - \frac{1}{1 + \exp(\mathbf{n} - \mathbf{y})} \right)$$

(6.72).

In the cepstral domain, it becomes

$$\frac{\partial f}{\partial \mathbf{y}} = \mathbf{C} \text{diag} \left(\frac{1}{1 + \exp(\mathbf{C}^{-1}(\mathbf{n} - \mathbf{y}))} \right) \mathbf{C}^{-1} \quad (6.73)$$

$$\frac{\partial f}{\partial \mathbf{n}} = \mathbf{I} - \mathbf{C} \text{diag} \left(\frac{1}{1 + \exp(\mathbf{C}^{-1}(\mathbf{n} - \mathbf{y}))} \right) \mathbf{C}^{-1} \quad (6.74)$$

For the state-space model with second-order linear-predictive-HMM, the extended Kalman filter algorithm is defined as:

$$\hat{\mathbf{z}}_{t|t-1}^j = \Phi_j \hat{\mathbf{z}}_{t-1|t-1}^j + \mathbf{u}_j \quad (6.75)$$

$$\hat{\mathbf{v}}_{t|t-1}^j = \Phi_j \hat{\mathbf{v}}_{t-1|t-1}^j + \mathbf{G} \mathbf{Q}_j \mathbf{G}^T \quad (6.76)$$

$$\mathbf{e}_t^j = \mathbf{o}_t - f(\hat{\mathbf{z}}_{t|t-1}^j) \quad (6.77)$$

$$\mathbf{K}_t^j = \hat{\mathbf{v}}_{t|t-1}^j \mathbf{H}^T (\mathbf{H} \hat{\mathbf{v}}_{t|t-1}^j \mathbf{H}^T)^{-1} \quad (6.78)$$

$$\hat{\mathbf{z}}_{t|t}^j = \hat{\mathbf{z}}_{t|t-1}^j + \mathbf{K}_t^j \mathbf{e}_t^j \quad (6.79)$$

$$\hat{\mathbf{v}}_{t|t}^j = (\mathbf{I} - \mathbf{K}_t^j \mathbf{H}) \hat{\mathbf{v}}_{t|t-1}^j \quad (6.80)$$

$$\text{where } \mathbf{H} = \begin{bmatrix} \frac{\partial f}{\partial \mathbf{y}} \Big|_{(\hat{\mathbf{z}}_{t|t-1}^j)} & \frac{\partial f}{\partial \mathbf{n}} \Big|_{(\hat{\mathbf{z}}_{t|t-1}^j)} \end{bmatrix}. \quad (6.81)$$

$$\begin{aligned} L_t^j &= p(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) \\ &= N(\mathbf{e}_t^j; \mathbf{0}, \mathbf{H} \hat{\mathbf{v}}_{t|t-1}^j \mathbf{H}) \end{aligned} \quad (6.82)$$

If the speech enhancement is in the log-spectral domain, the derivative matrix is

$$\mathbf{H} = \left[\frac{\partial f}{\partial \bar{\mathbf{y}}} \Big|_{(\hat{\mathbf{z}}_{t|t-1}^j)} \quad \frac{\partial f}{\partial \mathbf{n}} \Big|_{(\hat{\mathbf{z}}_{t|t-1}^j)} \right], \text{ where}$$

$$\frac{\partial f}{\partial \bar{\mathbf{y}}} = \text{diag} \left(\frac{1}{1 + \exp(\mathbf{n} - F\bar{\mathbf{y}})} \right) F \quad (6.83)$$

$$\frac{\partial f}{\partial \mathbf{n}} = \text{diag} \left(1 - \frac{1}{1 + \exp(\mathbf{n} - F\bar{\mathbf{y}})} \right) \quad (6.84)$$

In the cepstral domain, it becomes

$$\frac{\partial f}{\partial \bar{\mathbf{y}}} = \mathbf{C} \text{diag} \left(\frac{1}{1 + \exp(\mathbf{C}^{-1}(\mathbf{n} - F\bar{\mathbf{y}}))} \right) \mathbf{C}^{-1} F \quad (6.85)$$

$$\frac{\partial f}{\partial \mathbf{n}} = \mathbf{C} \text{diag} \left(1 - \frac{1}{1 + \exp(\mathbf{C}^{-1}(\mathbf{n} - F\bar{\mathbf{y}}))} \right) \mathbf{C}^{-1} \quad (6.86)$$

6.3.5 Unscented Kalman Filter for Joint-Speech-Noise-Tracking

If the unscented Kalman filter (UKF) algorithm is used, no explicit calculations of Jacobians or Hessians are needed. As reviewed in chapter 5, the unscented Kalman filter is a straightforward extension of the unscented transformation (UT) by applying the UT sigma point selection scheme. We will show next how to implement the unscented Kalman filter in the switching Kalman filter algorithm.

For the state-space with first order linear-predictive-HMM, if the unscented Kalman filter algorithm is used in equation (6.47), the algorithm becomes

$$(\hat{\mathbf{z}}_{t|t}^j, \mathbf{V}_{t|t}^j, L_t^j) = UKF(\hat{\mathbf{z}}_{t-1|t-1}, \hat{\mathbf{v}}_{t-1|t-1}, \mathbf{o}_t; \Phi_j, \mathbf{u}_j, \mathbf{Q}_j),$$

The *UKF()* function is given as follows: Firstly, form $\hat{\mathbf{z}}_{t-1|t-1}^a, \mathbf{v}_{t-1|t-1}^a, \chi^a$ as in [Wan00]:

$$\begin{aligned} \hat{\mathbf{z}}_{t-1|t-1}^a &= [\hat{\mathbf{z}}_{t-1|t-1}; \quad \mathbf{u}_j] \\ \mathbf{v}_{t-1|t-1}^a &= [\hat{\mathbf{v}}_{t-1|t-1} \quad \mathbf{0}; \quad \mathbf{0} \quad \mathbf{Q}_j] \\ \chi^a &= [(\chi^z)^T \quad (\chi^v)^T]^T \end{aligned} \quad (6.86)$$

Sigma matrices: $i = 1, \dots, L$, L is the dimension of $\hat{\mathbf{z}}_{t-1|t-1}^a$

$$\begin{aligned}\chi_0^a(t-1|t-1) &= \hat{\mathbf{z}}_{t-1|t-1}^a \\ \chi_i^a(t-1|t-1) &= \hat{\mathbf{z}}_{t-1|t-1}^a + \left(\sqrt{(L+\lambda)\mathbf{V}_{t-1|t-1}^a} \right)_i \\ \chi_{i+L}^a(t-1|t-1) &= \hat{\mathbf{z}}_{t-1|t-1}^a - \left(\sqrt{(L+\lambda)\mathbf{V}_{t-1|t-1}^a} \right)_i\end{aligned}\quad (6.87)$$

Time update equations:

$$\begin{aligned}\chi_i^z(t|t-1) &= \mathbf{A} \chi_i^z(t-1|t-1) + \chi_i^y(t-1|t-1) \quad i = 0, \dots, 2L \\ \hat{\mathbf{z}}_{t|t-1}^j &= \sum_{i=0}^{2L} W_i^{(m)} \chi_i^z(t|t-1) \\ \mathbf{V}_{t|t-1}^j &= \sum_{i=0}^{2L} W_i^{(c)} [\chi_i^z(t|t-1) - \hat{\mathbf{z}}_{t|t-1}^j][\chi_i^z(t|t-1) - \hat{\mathbf{z}}_{t|t-1}^j]^T \\ \mathbf{Y}_i(t|t-1) &= f(\chi_i^z(t|t-1)) \quad i = 0, \dots, 2L \\ \hat{\mathbf{o}}_{t|t-1}^j &= \sum_{i=0}^{2L} W_i^{(m)} \mathbf{Y}_i(t|t-1)\end{aligned}\quad (6.88)$$

Measurement update equations:

$$\mathbf{P}_{\hat{\mathbf{o}}\hat{\mathbf{o}}}(t) = \sum_{i=0}^{2L} W_i^{(c)} [\mathbf{Y}_i(t|t-1) - \hat{\mathbf{o}}_{t|t-1}^j][\mathbf{Y}_i(t|t-1) - \hat{\mathbf{o}}_{t|t-1}^j]^T \quad (6.89)$$

$$\mathbf{P}_{\hat{\mathbf{z}}\hat{\mathbf{o}}}(t) = \sum_{i=0}^{2L} W_i^{(c)} [\chi_i^z(t|t-1) - \hat{\mathbf{z}}_{t|t-1}^j][\mathbf{Y}_i(t|t-1) - \hat{\mathbf{o}}_{t|t-1}^j]^T \quad (6.90)$$

$$\mathbf{K}_t = \mathbf{P}_{\hat{\mathbf{z}}\hat{\mathbf{o}}}(t) \mathbf{P}_{\hat{\mathbf{o}}\hat{\mathbf{o}}}(t)^{-1} \quad (6.91)$$

Filtered estimate of the state vector:

$$\hat{\mathbf{z}}_{t|t}^j = \hat{\mathbf{z}}_{t|t-1}^j + \mathbf{K}_t(\mathbf{o}_t - \hat{\mathbf{o}}_{t|t-1}^j) \quad (6.92)$$

Filtered state-error covariance matrix:

$$\mathbf{V}_{t|t}^j = \mathbf{V}_{t|t-1}^j - \mathbf{K}_t \mathbf{P}_{\hat{\mathbf{o}}\hat{\mathbf{o}}}(t) \mathbf{K}_t^T \quad (6.93)$$

$$L_t^j = N(\mathbf{o}_t; \hat{\mathbf{o}}_{t|t-1}^j, \mathbf{P}_{\hat{\mathbf{o}}\hat{\mathbf{o}}}(t)) \quad (6.94)$$

For each frame of incoming noisy speech, the algorithm described above gives the estimate of the speech in the frame.

Compared with the EKF, the UKF can handle nonlinearities without using numerical derivatives and provides higher order approximation for both Gaussian and non-Gaussian distributions, as previously mentioned.

6.4 Simulation Results and Analysis

6.4.1 Simulation Results on the *Number* v1.3 Database

Pilot experiments were first conducted on the small database *Number* v1.3. The database description and experimental settings for the speech recognition engine are given in chapter 4. For experiments conducted on the database *Number* v1.3, we assume that an AR model of background noise is known. Clean speech features from 60 clean utterances are used to obtain the SLDM parameters. The number of hidden states (modes) used for the SLDM is decided by trial and error. Here we used 8 hidden states. If the log-spectral vectors are to be enhanced, the SLDM has full covariance matrices. For each frame, the noisy 20th order mel-scaled log spectrum is transformed to clean feature estimation. Finally, the estimated clean speech feature vectors are mapped from the log-spectral domain into the cepstral (MFCC) domain using a DCT transform, before being passed on directly to a speech recognizer for the recognition task. If the cepstrum vectors are to be enhanced, the SLDM has diagonal covariance matrices. For a fair comparison, we use a square DCT so that the cepstrum vectors have the same dimension as the log-spectrum vectors. The estimated clean speech feature vectors are then passed on directly to a speech recognizer for the recognition task.

From our experiments, we found that the extended Kalman filter can bring better results than approximating the observation equation (6.28),(6.29) by a piecewise linear model and then using the Kalman filter, as was done in Kim's previous work. An explanation is that, since joint noise and speech tracking yields a good estimate of hypothetical clean speech and noise, continuously updating the linearization around the previous speech and noise estimate will give a better approximation of the nonlinear function. So we use the extended Kalman filters in all of our experiments except when the Unscented Kalman Filter is used.

Tables 6.1 and 6.2 show the recognition accuracy for white noise and babble noise mixtures if the speech features are compensated in the log-spectral domain. Tables 6.3

and 6.4 show the recognition accuracy for white noise and babble noise mixtures if the speech features are compensated in the cepstral domain. The extended Kalman Filter algorithm is used to deal with the nonlinear distortion by noise in the feature domain. The results for the SLDM with first-order dependency are represented by ‘SLDM-chapter 6- $p=1$ ’ in the tables, while the results for SLDM with second-order dependency are represented by ‘SLDM-chapter 6- $p=2$ ’. Both the IMM algorithm and the GPB1 algorithm have been tested to provide approximative solutions to the switching nonlinear state-space model. For the purpose of comparison, the results with the noise-tracking-only method presented in [Kim98b], which we just reviewed in chapter 5, are listed in the tables as ‘[Kim98b]’. We also tested the SLDM method proposed in [Kim05] with a linearized Kalman filter, but the SLDM was modeled using the method described in [Raj04]. As we pointed out in section 6.1, the structure of the SLDM in [Kim05] and in [Raj04] is similar, the only difference being in the methods for finding the sufficient statistics to update the LDM parameters in the M step. The results are represented by ‘SLDM [Kim05]’ in the tables. Note that, as explained in Chapter 5, because of the statistical independence assumption between $p(s_t)$ and $p(s_{t-1})$, the interacting multiple model (IMM) approach is identical to the GPB1 approach for this SLDM structure [Kim98b]. However, the IMM approach is different from the GPB1 approach for our proposed SLDM structure.

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	50.6	73.4	88.0	93.0	76.3
[Kim98b]	55.0	74.0	87.8	93.0	77.5
SLDM [Kim05]	59.0	79.0	87.6	92.8	79.6
SLDM-chapter 6 $p=1$ -IMM	68.0	83.6	92.8	95.0	84.9
SLDM-chapter 6 $p=1$ -GPB1	64.8	84.2	93.0	95.0	84.3
SLDM-chapter 6 $p=2$ -IMM	71.0	86.6	93.4	95.6	86.7

Table 6. 1: Recognition accuracy (%) for speech mixed with white noise
when speech is enhanced in the log-spectrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	44.0	70.2	85.0	91.0	72.6
[Kim98b]	47.0	71.8	85.0	90.8	73.7
SLDM [Kim05]	49.6	73.2	84.6	91.0	74.6
SLDM-chapter 6 $p=1$ -IMM	59.4	74.8	90.4	93.4	79.5
SLDM-chapter 6 $p=1$ -GPB1	55.4	73.0	87.4	93.0	77.2
SLDM-chapter 6 $p=2$ -IMM	62.4	78.8	90.0	93.4	81.2

Table 6.2: Recognition accuracy (%) for speech mixed with babble noise
when speech is enhanced in the log-spectrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	50.6	73.4	88.0	93.0	76.3
[Kim98b]	54.4	74.8	86.8	91.2	76.8
SLDM [Kim05]	59.4	79.8	87.4	92.2	79.7
SLDM-chapter 6 $p=1$ -IMM	67.2	86.0	92.6	94.0	85.0
SLDM-chapter 6 $p=1$ -GPB1	66.8	84.6	92.6	94.2	84.6
SLDM-chapter 6 $p=2$ -IMM	69.2	87.6	92.8	94.4	86.0

Table 6.3: Recognition accuracy (%) for speech mixed with white noise
when speech is enhanced in the cepstrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	44.0	70.2	85.0	91.0	72.6
[Kim98b]	47.2	72.4	85.0	89.6	73.6
SLDM [Kim05]	50.4	73.2	84.0	91.0	74.7
SLDM-chapter 6 $p=1$ -IMM	59.4	77.4	89.0	92.4	79.6
SLDM-chapter 6 $p=1$ -GPB1	55.6	75.8	87.6	92.0	77.8
SLDM-chapter 6 $p=2$ -IMM	62.4	77.6	88.0	92.0	80.0

Table 6.4: Recognition accuracy (%) for speech mixed with babble noise when speech is enhanced in the cepstrum domain

The simulation results on the *Number* v1.3 database show that

- The switching linear dynamic model approach with either the SLDM by [Kim05] or the proposed SLDM is effective at improving the noise-robust recognition performance and provides better results compared with the method presented in [Kim98b], where the clean speech model is a Gaussian mixture model (GMM) and only noise is tracked with switching Kalman filters. It confirms that joint noise and speech tracking yields a better enhancement result.
- For both white noise and babble noise, the ‘SLDM-chapter 6- $p=1$ ’ and ‘SLDM-chapter 6- $p=2$ ’ show their superiority to their SLDM counterpart in [Kim05]. That is, the introduction of transition probabilities among linear dynamic models can improve the performance of SLDM systems.
- IMM vs. GPB1: from the comparison of approximation algorithms for switching Kalman filters presented in chapter 5, we saw that with the same computational complexity as the IMM algorithm, the merging in the GPB1 is more aggressive, and it is expected to have a lower estimation accuracy than the IMM. The experimental

results lead us to the same conclusions. The IMM algorithm provides better speech estimates than the GPB1 for both types of noise and for all SNRs.

- For SNRs between 0 dB and 10 dB, based on the same approximation algorithm, the ‘SLDM-chapter 6- $p=2$ ’ outperforms the ‘SLDM-chapter 6- $p=1$ ’. We can conclude that the SLDM with higher-order dependency is a better way to characterize the clean speech features distribution since it can model longer-term temporal dependencies. However, there is a tradeoff involved in that. As p increases, the number of model parameters grows linearly; this increase will lead to substantially slower filtering if p is too large.
- Speech enhancement in the cepstral domain brings similar results as those in the log-spectral domain. However, since the covariance matrices for linear dynamic models in the SLDM are diagonal in the cepstral domain, Kalman filtering algorithms could be implemented faster in that domain.

6.4.2 Simulation Results on the *Aurora 2* Database

The proposed algorithms were also tested on the *Aurora 2* database. As we found from the pilot test on the *Number* database that speech enhancement in the cepstral domain brought similar results to those in the log-spectral domain, in this section we tested the proposed algorithms in the log-spectral domain only. Since we expect a desirable balance between performance gain and the degree of computational complexity, for a large database like the *Aurora 2*, we tested the SLDM approach with first order dependency only and approximated the switching Kalman filter using the IMM algorithm. All the experiments used the same linear-predictive-HMM model with 16 hidden HMM states. With the increase of the number of hidden LDMs, the computational cost grows since the number of Kalman filters carried out in parallel in the switching Kalman filters increases. The number of hidden states was determined according to the simulation results in [Dro04], to achieve a desirable balance between performance gain and the cost of processing.

The *Aurora 2* database description and experimental settings for the speech recognition engine are given in chapter 4. We use the standard *Aurora* front-end settings. To conform to the observation model presented in (6.32), the feature generation was modified slightly from the reference implementation. In particular, we changed from using the spectral magnitude to using the power spectral density as the input to the Mel-frequency filter-bank to make equation (6.32) stand. Our settings are the same as those in [Dro04][Kim05].

All the experiments used the same speech model with 16 hidden states (modes). For each noise type a specific noise model was built. The parameters for the noise model were trained on the first ten frames of the noisy utterances. The first ten frames are assumed to be speech free. The experimental conditions to obtain the noise model were also maintained the same as those in [Dro04][Kim05].

Detailed recognition rate (%) for each of the four noise conditions and each of the noisy speech SNRs in the *Aurora 2* set-A are provided in Table 6.5, for the proposed SLDM approach. The SLDM has a first-order dependency and the switching extended Kalman filtering was approximated by the IMM algorithm. If the UKF is used in this approach, the results obtained are shown in Table 6.6.

	Subway	Babble	Car	Exhibition	Average
20 dB	98.25	98.67	98.78	97.78	98.37
15 dB	96.68	97.10	98.48	96.73	97.24
10 dB	91.10	93.53	94.63	91.61	92.72
5 dB	81.67	81.68	84.19	79.57	81.78
0 dB	56.80	47.37	55.98	51.16	52.83
Average	84.90	83.67	86.41	83.37	84.59

Table 6.5 Accuracy (%) with speech enhanced by the switching extended Kalman Filter based on the proposed SLDM

	Subway	Babble	Car	Exhibition	Average
20 dB	98.25	98.67	98.78	97.78	98.37
15 dB	96.76	97.07	98.39	96.42	97.16
10 dB	93.52	94.13	96.77	93.40	94.46
5 dB	86.37	85.12	89.66	86.70	86.96
0 dB	70.73	56.41	74.25	68.03	67.36
Average	89.13	86.28	91.57	88.47	88.86

Table 6.6 Accuracy (%) with speech enhanced by the switching unscented Kalman Filter based on the proposed SLDM

The simulation results on the *Aurora* 2.0 database are summarized in Table 6.7, with results averaged across 0 dB to 20 dB. For the purpose of comparison, the results reported in [Kim05] and [Dro04] are also listed in Table 6.7. The results obtained by our proposed linear-predictive-HMM based SLDM approach are presented as ‘SLDM chapter 6-EKF’ and ‘SLDM chapter 6-UKF’ in the table. The baseline results without noise processing are also given. The number of LDMs used in the SLDM to obtain the results presented in Table 6.7 are given in Table 6.8.

	Subway	Babble	Car	Exhibition	Average
Baseline	70.22	51.35	60.28	65.39	61.81
SLDM [Kim05]	80.55	67.52	86.19	77.82	78.02
SLDM [Dro04]	80.48	80.98	86.82	84.53	83.20
SLDM chapter 6-EKF	84.90	83.67	86.41	83.37	84.59
SLDM chapter 6-UKF	89.13	86.28	91.57	88.47	88.86

Table 6.7 Summary of accuracy (%) on *Aurora* 2.0, set A

Number of LDMs in the SLDM	
SLDM [Kim05]	128
SLDM [Dro04]	16
SLDM chapter 6	16

Table 6.8 Number of LDMs in the SLDM

Our SLDM approach based on the linear predictive HMM outperforms the SLDM approach from [Dro04],[Kim05]. More specifically, with the same number of LDMs in SLDM, the proposed SLDM approach outperforms the SLDM approach in [Dro04], especially when the environmental noise is babble noise. As we compare our results with those obtained in [Kim05], even with only 1/8 of the computational load used for the experiments in [Kim05], a better performance is obtained, especially when the environmental noise is babble noise or subway noise. If we replace the Extended Kalman filter with the Unscented Kalman filter, a clear performance improvement can be observed. It has been shown in [Dro04] that with the increase of the number of hidden LDMs, better results could be achieved, as with the increase of the number of LDMs, the SLDM can describe the acoustic space more accurately. Therefore, it is promising that if we increase the number of the linear dynamic models in the SLDM, we should obtain even better results than our present results.

6.5 Summary

In this chapter, a new Switching Linear Dynamic Model (SLDM) approach was proposed to compensate for the environment noise in the speech feature domain. Compared with previous work, the proposed SLDM approach based on linear-predictive-HMM can improve the noise-robust recognition accuracy even with a simple structure specification for the SLDM. One of the major advantages of the linear-predictive-HMM is its capability of capturing the temporal correlations among adjacent frames of speech in a systematic way. We have carried out experiments to evaluate the effects of incorporating the transition probabilities into the SLDM for the speech feature enhancement

application. Simulation results have confirmed the improved performance of the proposed approach over some previously published SLDM methods.

Chapter 7

Linear Dynamic Models with Mixture of Experts Architecture for Speech Feature Enhancement

In chapter 6 a switching linear dynamic model (LDM), named linear-predictive-HMM, was proposed to describe the time evolution of speech features in the context of feature enhancement for speech recognition. The basic idea is that the model can switch between different dynamic “modes” or “regimes”, each with local stationary statistics, so as to approximate the non-stationary dynamics of speech. The dynamics of the modes themselves are governed by a Markov chain. In the literature, there is another class of dynamic models, known as the mixture of experts (ME) [Jac91], which has been well studied for non-stationary time series prediction. Motivated by the capability of the ME in modeling the non-stationary dynamics and the close link between linear-predictive-HMM and ME, in this chapter we propose a novel approach to enhance speech features under noisy environments based on a switching linear dynamic model with the ME structure named as mixture of linear dynamic models.

It should be noted that there are two types of state variables in a switching linear dynamic model (LDM): discrete or continuous. They have different functions in the SLDM and in our applications. To distinguish them, we will label the discrete states as ‘state (mode)’ in the following sections, meaning different discrete states representing different dynamic “modes”.

7.1 Introduction

The mixture of experts (ME) is a powerful divide-and-conquer algorithm which was first introduced by Jacobs et al in 1991 [Jac91]. It consists of a set of competing experts moderated by a gate. The basic idea behind the mixture of experts is simple: rather than using a single global model, several local models (the experts) are used and each

specializes in modeling a small region of the input space. When the experts are predictors and the data set is a time series, the algorithm is capable of segmenting the nonstationary time series into stationary regimes with each region modeled by a predictive model. There is a large literature on models of this kind in econometrics, signal processing, and other fields. The basis concept of the ME can be visualized in figure 7.1 [Lie99]. The upper picture in figure 7.1 shows a nonstationary time series consisting of three dynamic modes. The experts of the ME system specialize on different dynamic modes. Figure 7.1 illustrates the corresponding probabilities of each expert. It is seen that the dynamics of a complex non-stationary physical process can then be described by switching among a set of expert models over time.

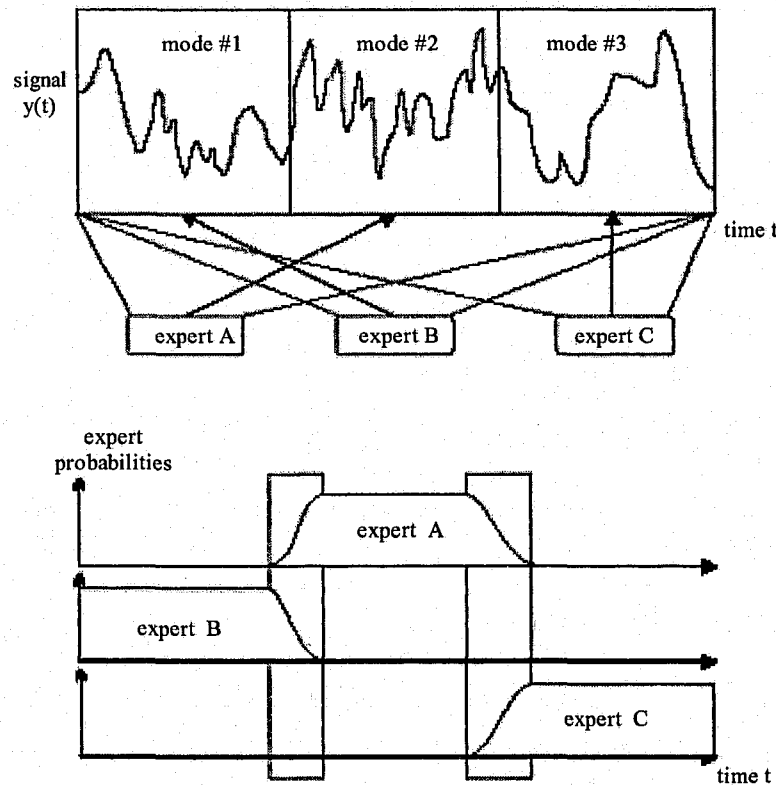


Fig 7.1 Illustration of the mixture of experts (ME) architecture

The central problem in using a set of experts is therefore the calculation of the activities of each expert, called the gating problem. Upon what information can the gate base its

decision? There are two choices: *external* information, presented in the form of input to the gating network, and *internal information*, extracted from the estimate of the present state (mode). The work of [Jac91][Wei95][Jor94][Wat95][Zha95] fall into the first case. The expert's activities are provided by a feedforward gating network that is directly connected to the inputs. The input space is split nonlinearly through the hidden units of the gating network, while in its original formulation [Jac91], the experts and gate saw the same input. The ME was applied to systems where different regimes did not overlap in the input space. However, for the prediction task, this is not always possible. Weigend et al [Wei95] got around this by using a longer tapped delay line at the gate than for the experts and applied it to predict the daily electricity demand of France, a real-world problem. Subsequently, Jordan and Jacobs [Jor94] introduced the architecture of a hierarchical mixture of linear experts, and Waterhouse and Robinson [Wat95] applied the hierarchical mixture of linear experts to time series prediction of acoustic vectors. The hierarchical mixture of experts has also been applied in a similar manner in speech recognition as a phonetic decision tree and performance improvements on the Wall Street Journal database test have been reported [Zha95].

In the second case of no connection between the input and the gate, the gate bases its decision on its estimate of the previous state (mode) and the transition from the previous state (mode) to the present state (mode). This concept is identical to a hidden Markov model [Rab89]. Based on this concept, a dynamic ME approach was proposed by Kohlmorgen et al [Koh00] and successfully applied to segment and identify the dynamic modes of electroencephalogram (EEG) data sets. In the fields of econometrics, Shi and Weigend applied ME to address the financial data prediction problem and showed promising results [Shi97]. Another application of the ME framework to the modeling of time series has been proposed in [Gha96]. The continuous signal sequence is processed by several experts and a discrete state (mode) variable is used to switch between these outputs. It has been revealed in [Gha96] that the switch state (mode) plays the role of a gating network and has Markovian dynamics.

Note that since the gate in the first case is directly connected to the inputs, this direct connection can facilitate the splitting, whereas in the second case the HMMs split the input space only indirectly [Wei95a]. The information from the external inputs and from the previous internal state (mode) of the gating network can also be combined: the gating network proposed in [Cac94] has recurrent connections from the output layer to the network's input layer, which allows the gate to calculate the present state based on the information from the last state and the present inputs. Bengio et al [Ben95] and Predovicu et al [Pre95] presented a framework called "input-output hidden Markov model" that allows the gate to access both external information (from inputs) and its own past outputs.

Up to now, research efforts on mixture of experts have shown that the ME is effective in modeling the temporal structure of non-stationary time-series. We notice that the linear-predictive-HMM proposed in chapter 6 is closely related to the concept of mixture of experts in the second case. In this chapter we will focus on mixture of experts with a structure falling into the first case. Since we use linear dynamic models as experts while in the ME, nonlinear experts are normally used; it is called mixtures of linear dynamic models here. There are important differences between a mixture of linear dynamic models with the ME architecture and the switching LDM presented in [Dro04] and [Kim05], as shown in figure 7.2. The shaded nodes are observed, and clear nodes are hidden. The prior probability of each model is fixed and input-independent in the switching LDM, while in the mixture of linear dynamic models the *a priori* probability is based on the knowledge of external inputs.

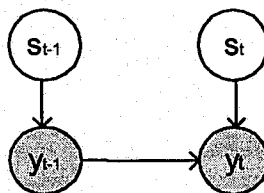


Fig 7.2 (a) Graphical representation of the switching LDM in [Dro04] and [Kim05]

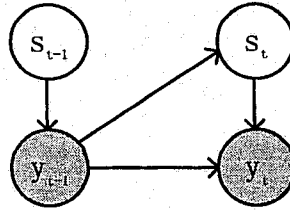


Fig 7.2 (b) Graphical representation of the mixture of linear dynamic models

7.2 Mixture of Linear Dynamic Models as a Speech Model

Consider a time series that consists of pairs of input and target data $\{(x_t, y_t)\}$. The target data might be a future value of a time series, e.g. y_t , and the input data might be a vector of past values, e.g. $x_t = [y_{t-1}]$ or $x_t = [y_{t-1}, y_{t-2}]$. This is the usual formulation of a time series prediction problem. At each time step t , $1 \leq t \leq T$, each expert i provides a prediction $y_{t,i}$ for the target y_t .

Figure 7.3 shows the architecture of a mixture of linear dynamic models with first order evolution in the LDM. The mixture of linear dynamic models consists of M linear dynamic models (LDM) and one gating network. Each LDM corresponds to a different state (mode); therefore there are M possible states (modes). Both the LDM and the gating network have access to the inputs: they can either share the same inputs $[y_{t-1}]$, or be given different sets of inputs: vector $[y_{t-1}]$ as the input to the LDMs, vector $[y_{t-1}, y_{t-2}]$ as the input to the gating network, which is represented by a dotted line in Figure 7.3.

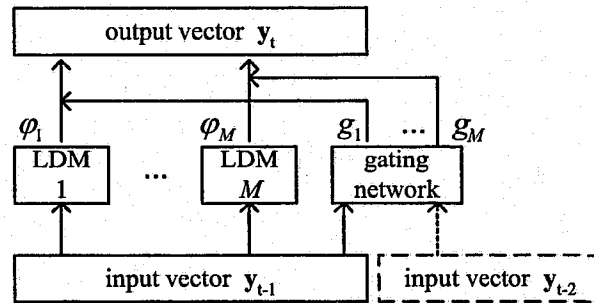


Fig. 7.3 Block diagram of the proposed mixture of LDMs

In the remaining part of this section, we presume that the LDMs and gating network see the same input $\mathbf{x}_t = [\mathbf{y}_{t-1}]$. The pair of input and target data is $\{\mathbf{x}_t, \mathbf{y}_t\}$. A multilayer feedforward neural network (MLP) or a radial basis function (RBF) neural network can be applied as the nonlinear gating network. The gating network has one output for each expert denoted by $g_i(\mathbf{x}_t, \boldsymbol{\theta}_g)$, which may be interpreted as the probability that the desired clean speech observation output $\mathbf{y}_t$ was generated by the i^{th} LDM (i.e., i^{th} possible state (mode)) given $\mathbf{x}_t$.

Denoting the MLP weights from the input to the hidden units as w_{jl} and the MLP weights from the hidden units to the output units as W_{ij} , we have the network (internal) outputs α_i as:

$$\alpha_i(\mathbf{x}_t, \boldsymbol{\theta}_g) = \sum_j W_{ij} \Psi_j \left(\sum_l w_{jl} \mathbf{x}_{t,l} \right) \quad i = 1, 2, \dots, M \quad (7.1)$$

where j is the index of the hidden units, and l is the index of the input units. The hidden units' activation function is the standard sigmoid function

$$\Psi(x) = \frac{1}{1 + \exp(-x)} \quad (7.2)$$

where $\boldsymbol{\theta}_g = \{W_{ij}, w_{jl}\}$ is the set of parameters characterizing the MLP gating network. If the gating network is a RBF network with M direct (internal) outputs α_i , we have

$$\alpha_i(\mathbf{x}_t, \boldsymbol{\theta}_g) = \sum_k w_{ki} \exp \left(- \frac{\|\mathbf{x}_t - \mathbf{c}_k\|^2}{2\sigma_k^2} \right), \quad i = 1, 2, \dots, M \quad (7.3)$$

where $\boldsymbol{\theta}_g = \{\mathbf{c}_k, \sigma_k, w_{ki}\}$ is the set of parameters characterizing the gating network: $\mathbf{c}_k$ is the RBF center for each hidden neuron k , σ_k is the variance for the Gaussian function in each hidden neuron k , and w_{ki} is the weight that connects the k^{th} hidden neuron to the node i in the RBF output layer.

The direct outputs α_i are exponentiated and normalized by their sum, giving the final gating outputs g_i :

$$g_i(\mathbf{x}_t, \boldsymbol{\theta}_g) = p(s_t = i | \mathbf{x}_t) = \frac{\exp(\alpha_i)}{\sum_{m=1}^M \exp(\alpha_m)} \quad (7.4)$$

where s_t is the unknown state (mode) at time t . The definition of g_i implies that they are strictly positive, smaller than 1 and that the sum over all the g_i is equal to 1. For the time series prediction, a linear dynamic model with the i^{th} state (mode) parameter set $\theta_i = \{\mathbf{B}_i, \mu_i, \Sigma_i\}$ obeys the following system equation:

$$\mathbf{y}_{t,i} = \varphi_i(\mathbf{x}_t, \theta_i) = \mu_i + \mathbf{B}_i \mathbf{x}_t + \mathbf{e}_{t,i}, \quad \mathbf{e}_{t,i} \sim N(\mathbf{0}, \Sigma_i) \quad (7.5)$$

where $\mathbf{B}_i$ represents the autoregressive prediction coefficients between consecutive frames of clean speech in the feature domain, μ_i is the estimated mean of $\mathbf{y}_{t,i}$, $\mathbf{e}_{t,i}$ is the autoregressive prediction residual with a zero mean Gaussian distribution and covariance matrix Σ_i . The overall output for a mixture of linear dynamic models is then defined as:

$$\mathbf{y}_t = \sum_{i=1}^M g_i(\mathbf{x}_t, \theta_g) \mathbf{y}_{t,i} = \sum_{i=1}^M g_i(\mathbf{x}_t, \theta_g) \varphi_i(\mathbf{x}_t, \theta_i) \quad (7.6).$$

We have the overall probability as

$$p(\mathbf{y}_t | \mathbf{x}_t) = \sum_{i=1}^M p(s_t = i | \mathbf{x}_t) P(\mathbf{y}_t | \mathbf{x}_t, s_t = i) = \sum_{i=1}^M g_i(\mathbf{x}_t, \theta_g) P_\lambda(\mathbf{y}_t | \mathbf{x}_t, s_t = i) \quad (7.7).$$

The Expectation Maximization (EM) algorithm [Dem77] is used here to train the proposed model. For details, see [Jor95] which provided a theoretical analysis for the convergence of the Expectation-Maximization (EM) algorithm for the training of mixture of experts. Assuming that at each time instant the underlying dynamics only come from a single model, the gating output g_i should be binary, i.e., zero or one. In other words, only one predictor remains active for each input pattern. With this assumption, the likelihood of the “complete data” can be written as

$$L = \prod_{t=1}^T \prod_{i=1}^M [g_i(\mathbf{x}_t, \theta_g) P_\lambda(\mathbf{y}_t | \mathbf{x}_t, s_t = i)]^{I_t^i} \quad (7.8)$$

where I_t^i is 1 if $\mathbf{y}_t$ is generated by the i^{th} LDM, otherwise it is 0. Note that whereas I is binary, its expectation h can take any value between 0 and 1. The EM algorithm allows us to replace the missing variable I_t^i by their expected values $h_t(i)$. Therefore, in the E-step, the current parameter estimation is held fixed to compute the expected state (mode) occupancy $h_t(i)$:

$$h_t(i) = P(s_t = i | \mathbf{y}_t, \mathbf{x}_t) = \frac{P(s_t = i | \mathbf{x}_t)P(\mathbf{y}_t | \mathbf{x}_t, s_t = i)}{\sum_j P(s_t = j | \mathbf{x}_t)P(\mathbf{y}_t | \mathbf{x}_t, s_t = j)} \quad (7.9).$$

Note that, the gating network output $g_i(\mathbf{x}_t, \boldsymbol{\theta}_g)$ is solely based on the knowledge of $\mathbf{x}_t$, not on the knowledge of $\mathbf{y}_t$, called the *a priori* probability, while the posterior probability $h_t(i)$ includes how close the output of expert i is to the target $\mathbf{y}_t$.

The probability of generating $\mathbf{y}_t$ by the i^{th} LDM at time t given $\mathbf{x}_t$ is

$$P(\mathbf{y}_t | \mathbf{x}_t, s_t = i) = (2\pi)^{-d/2} |\boldsymbol{\Sigma}_i|^{-1/2} \exp(-\frac{1}{2}[\mathbf{y}_t - \mathbf{B}_i\mathbf{x}_t - \boldsymbol{\mu}_i]^T \boldsymbol{\Sigma}_i^{-1} [\mathbf{y}_t - \mathbf{B}_i\mathbf{x}_t - \boldsymbol{\mu}_i]) \quad (7.10)$$

where d is the dimension of $\mathbf{y}_t$.

$$P(s_t = i | \mathbf{x}_t) = g_i(\mathbf{x}_t, \boldsymbol{\theta}_g) \quad (7.11)$$

The definition of the posteriors $h_t(i)$ here includes both the input and output values of each LDM. The EM algorithm requires us to maximize the expected log-likelihood $Q(\lambda, \bar{\lambda})$, where $\lambda = (\boldsymbol{\theta}_g, \boldsymbol{\theta}_i)$ $i=1,2,\dots,M$ is the model corresponding to an initial estimate of the parameters, and $\bar{\lambda} = (\bar{\boldsymbol{\theta}}_g, \bar{\boldsymbol{\theta}}_i)$ $i=1,2,\dots,M$ represents the parameters of the new model. Taking the expectation of the logarithm of equation (7.8) yields the cost function $Q(\lambda, \bar{\lambda})$ as

$$\begin{aligned} Q(\lambda, \bar{\lambda}) &= \sum_t \sum_i P_{\bar{\lambda}}(s_t = i | \mathbf{y}_t, \mathbf{x}_t) \log[P_{\bar{\lambda}}(s_t = i | \mathbf{x}_t) P_{\lambda}(\mathbf{y}_t | \mathbf{x}_t, s_t = i)] \\ &= \sum_t \sum_i h_t(i) \log[P_{\bar{\lambda}}(s_t = i | \mathbf{x}_t) P_{\bar{\lambda}}(\mathbf{y}_t | \mathbf{x}_t, s_t = i)] \end{aligned} \quad (7.12)$$

where $h_t(i)$ is the posterior probability that the LDM is in state (mode) i at time t . The above equation consists of two parts: the first one related to the parameters of the gating network, $\boldsymbol{\theta}_g$, the second one related to the parameters of the individual linear dynamic models, $\boldsymbol{\theta}_i$.

$$\begin{aligned} Q(\boldsymbol{\theta}_g, \bar{\boldsymbol{\theta}}_g) &= \sum_t \sum_i h_t(i) \log P_{\bar{\lambda}}(s_t = i | \mathbf{x}_t) \\ Q(\boldsymbol{\theta}_i, \bar{\boldsymbol{\theta}}_i) &= \sum_t h_t(i) \log P_{\bar{\lambda}}(\mathbf{y}_t | \mathbf{x}_t, s_t = i), \quad i=1,2,\dots,M \end{aligned} \quad (7.13)$$

The two parts can then be maximized separately giving rise to parameter update equations for the gating network and the LDMs.

- M step for linear dynamic models

A new set of LDM parameters is found by maximizing the expected log-likelihood $Q(\theta_i, \bar{\theta}_i)$. By substituting equation (7.10) into equation (7.13) and dropping terms that are independent, the objective function becomes

$$Q(\theta_i, \bar{\theta}_i) = -\frac{1}{2} \sum_t h_t(i) \log |\bar{\Sigma}_i| - \frac{1}{2} \sum_t h_t(i) E_{ii}(\mathbf{y}_t - \bar{\mathbf{B}}_i \mathbf{x}_t - \bar{\boldsymbol{\mu}}_i)' \bar{\Sigma}_i^{-1} (\mathbf{y}_t - \bar{\mathbf{B}}_i \mathbf{x}_t - \bar{\boldsymbol{\mu}}_i) \quad (7.14).$$

To present the result of this maximization, the following sufficient statistics are introduced [Mur98b]:

$$\begin{aligned} S_{YY',i} &= \sum_t h_t(i) E_{ii}[\mathbf{y}_t \mathbf{y}_t'] \\ S_{Y'Y,i} &= \sum_t h_t(i) E_{ii}[\mathbf{y}_t' \mathbf{y}_t] \\ S_{Y,i} &= \sum_t h_t(i) E_{ii}[\mathbf{y}_t] \\ S_{XX',i} &= \sum_t h_t(i) E_{ii}[\mathbf{x}_t \mathbf{x}_t'] \\ S_{X,i} &= \sum_t h_t(i) E_{ii}[\mathbf{x}_t] \\ S_{XY',i} &= \sum_t h_t(i) E_{ii}[\mathbf{x}_t \mathbf{y}_t'] \\ S_{YX',i} &= \sum_t h_t(i) E_{ii}[\mathbf{y}_t \mathbf{x}_t'] \end{aligned} \quad (7.15)$$

where " ' " denotes a transpose operation. Maximizing $Q(\theta_i, \bar{\theta}_i)$ with respect to $\mathbf{B}_i$ and setting the derivative to zero, the following equation is obtained:

$$\begin{aligned} \frac{\partial}{\partial \mathbf{B}_i} Q &= -\frac{1}{2} \sum_t h_t(i) \cdot -2 \bar{\Sigma}_i^{-1} E_{ii}[(\mathbf{y}_t - \bar{\mathbf{B}}_i \mathbf{x}_t - \bar{\boldsymbol{\mu}}_i) \mathbf{x}_t'] \\ &= \bar{\Sigma}_i^{-1} \left\{ \left(\sum_t h_t(i) E_{ii}[\mathbf{y}_t \mathbf{x}_t'] \right) - \bar{\mathbf{B}}_i \left(\sum_t h_t(i) E_{ii}[\mathbf{x}_t \mathbf{x}_t'] \right) - \bar{\boldsymbol{\mu}}_i \left(\sum_t h_t(i) E_{ii}[\mathbf{x}_t'] \right) \right\} \\ &= \bar{\Sigma}_i^{-1} \{ S_{YX',i} - \bar{\mathbf{B}}_i S_{XX',i} - \bar{\boldsymbol{\mu}}_i S_{X,i}' \} \\ \frac{\partial}{\partial \mathbf{B}_i} Q &= 0 \end{aligned} \quad (7.16)$$

The vector equation is obtained:

$$\bar{\mathbf{B}}_i = (S_{YX',i} - \bar{\boldsymbol{\mu}}_i S_{X,i}') S_{XX',i}^{-1} \quad (7.17).$$

Likewise, setting the derivatives of Q with respect to $\boldsymbol{\mu}_i$ equal to 0

$$\begin{aligned}
\frac{\partial}{\partial \bar{\mu}_i} Q &= -\frac{1}{2} \sum_i h_i(i) \cdot -2 \bar{\Sigma}_i^{-1} E_{ii}[(y_i - \bar{\mathbf{B}}_i \mathbf{x}_i - \bar{\mu}_i)] \\
&= \bar{\Sigma}_i^{-1} \{ (\sum_i h_i(i) E_{ii}[y_i]) - \bar{\mathbf{B}}_i (\sum_i h_i(i) E_{ii}[\mathbf{x}_i]) - \bar{\mu}_i (\sum_i h_i(i)) \} \\
&= \bar{\Sigma}_i^{-1} \{ S_{Y,i} - \bar{\mathbf{B}}_i S_{X,i} - \bar{\mu}_i h_i \} \\
\frac{\partial}{\partial \bar{\mu}_i} Q &= 0
\end{aligned} \tag{7.18}$$

Another vector equation is obtained

$$\bar{\mu}_i = \frac{S_{Y,i} - \bar{\mathbf{B}}_i S_{X,i}}{h_i} \tag{7.19}$$

$$\text{where } h_i = \sum_i h_i(i). \tag{7.20}$$

The new estimates for $\bar{\mu}_i$ and $\bar{\mathbf{B}}_i$ can now be obtained by solving the pair of simultaneous equations:

$$\begin{aligned}
\bar{\mathbf{B}}_i &= (S_{YX',i} - \bar{\mu}_i S_{X',i}) S_{XX',i}^{-1} \\
\bar{\mu}_i &= \frac{S_{Y,i} - \bar{\mathbf{B}}_i S_{X,i}}{h_i}
\end{aligned} \tag{7.21}$$

Setting the derivatives of $Q(\theta_i, \bar{\theta}_i)$ with respect to $\bar{\Sigma}_i$ gives the re-estimation formula as

$$\begin{aligned}
\frac{\partial}{\partial \bar{\Sigma}_i} Q &= \frac{1}{2} (\sum_i h_i(i)) \bar{\Sigma}_i - \frac{1}{2} \sum_i h_i(i) E_{ii}(\mathbf{y}_i - \bar{\mathbf{B}}_i \mathbf{x}_i - \bar{\mu}_i)(\mathbf{y}_i - \bar{\mathbf{B}}_i \mathbf{x}_i - \bar{\mu}_i)' \\
&= 0 \\
\bar{\Sigma}_i &= \frac{1}{h_i} \sum_i h_i(i) E_{ii}(\mathbf{y}_i \mathbf{y}_i' - \mathbf{y}_i \mathbf{x}_i' \bar{\mathbf{B}}_i' - \mathbf{y}_i \bar{\mu}_i' - \bar{\mathbf{B}}_i \mathbf{x}_i \mathbf{y}_i' + \bar{\mathbf{B}}_i \mathbf{x}_i \mathbf{x}_i' \bar{\mathbf{B}}_i' + \bar{\mathbf{B}}_i \mathbf{x}_i \bar{\mu}_i' - \bar{\mu}_i \mathbf{y}_i' + \bar{\mu}_i \mathbf{x}_i' \bar{\mathbf{B}}_i' + \bar{\mu}_i \bar{\mu}_i')
\end{aligned} \tag{7.22}$$

Since we have

$$\bar{\mu}_i = \frac{S_{Y,i} - \bar{\mathbf{B}}_i S_{X,i}}{h_i} \tag{7.23}$$

the equation (6.20) is further simplified to

$$\bar{\Sigma}_i = \frac{1}{h_i} (S_{YY',i} - S_{YX',i} \bar{\mathbf{B}}_i' - \bar{\mathbf{B}}_i S_{XY',i} + \bar{\mathbf{B}}_i S_{XX',i} \bar{\mathbf{B}}_i' - h_i \bar{\mu}_i \bar{\mu}_i') \tag{7.24}$$

- M step for nonlinear gating network

We then focus on the cost function $Q(\theta_g, \bar{\theta}_g)$. Since we use nonlinear hidden units, the weights of the network cannot be computed directly but require iterative techniques. The weight changes in the gating network are proportional to the gradient of the cost function

with respect to the intermediate variable α_i (prior to the exponentiation and normalization in the softmax part). To maximize the log-likelihood is equal to minimizing the negative loglikelihood. If $Q(\theta_g, \bar{\theta}_g)$ is interpreted as an error function to be minimized, we have

$$\frac{\partial}{\partial \alpha_i} Q = -(h_i(i) - g_i(\mathbf{x}_i, \bar{\theta}_g)) \quad (7.25)$$

$$\frac{\partial Q}{\partial \bar{\theta}_g} = \frac{\partial Q}{\partial \alpha_i} \frac{\partial \alpha_i}{\partial \bar{\theta}_g} = -(h_i(i) - g_i(\mathbf{x}_i, \bar{\theta}_g)) \frac{\partial \alpha_i}{\partial \bar{\theta}_g} \quad (7.26)$$

If $h_i(i)$ is treated as the target of the neural network, equation (7.24) is the same as those in the BackPropagation algorithm. Therefore, for both RBF and MLP, the parameters can be found by the gradient-descent method. The local-optimum property of the BackPropagation algorithm in this M-step makes the learning algorithm described in this section a generalized EM [Dem77]. An alternative way to update the parameters of the single-layer RBF network defined in equation (7.3) is to combine an unsupervised K-means clustering algorithm for adjusting the RBF center and variance parameters, and a supervised Least-Squares (LS) algorithm for updating the weights [Moo89]. The supervised training tries to match the gating network output $g_i(\mathbf{x}_i, \bar{\theta}_g)$ to the posterior probability $h_i(i)$. The advantages of updating the RBF in a hybrid manner are that it is fast and very easy to interpret. Let $\mathbf{h}$ denote a T -by- I matrix with element $h_{it} = h_i(i)$, T is the number of training samples, and I is the dimension of the RBF output layer which is equal to the number of experts. Φ is a T -by- K matrix, K is the number of hidden neurons, and its elements ϕ_{ik} are calculated as

$$\phi_{ik} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{c}_k\|^2}{2\sigma_k^2}\right) \quad (7.27)$$

where $\mathbf{c}_k$ is the RBF center for hidden neuron k , and σ_k is the variance for the Gaussian function in hidden neuron k . The weights $\mathbf{w}$ for the RBF output layer are updated by the Least-Squares algorithm as

$$\mathbf{w} = \text{pinv}(\Phi)\mathbf{h} \quad (7.28)$$

where pinv stands for the pseudoinverse.

A mixture of linear dynamic models trained using the EM algorithm is guaranteed to reach only a local maximum likelihood solution. Because there are many local maxima, such models are sensitive to how they initialized. For this reason, we need to initialize the model rather carefully. Both the gating network and the linear dynamical model must be initialized. The key point is to start with a good partition of the data, each being modeled by an LDM [Gha98]. Our initialization algorithm is summarized as follows:

1. Use the same initialization technique developed in chapter 6 to obtain the initial estimate for the LDM parameters.
2. For the RBF gating network, the initial RBF centers and variance parameters are obtained by the unsupervised K-means clustering algorithm. The weights in the RBF output layer are randomly set to small values. If the gating model is a MLP network, the weights in the MLP hidden layers are randomly set to small values.

We found through the experiments that, probably because the RBF gating network could be initialized in a more proper way than the MLP gating network, the mixture of linear dynamic models with the RBF gating network brought better results.

Now, the EM algorithm can be summarized as follows:

Step 1: Initialize $\lambda = (\theta_g, \theta_i)$ $i = 1, 2 \dots M$

Step 2: Find the expected state (mode) occupancy: $h_t(i)$, $t = 1, \dots, T$

Step 3: A new set of parameters $\bar{\lambda} = (\bar{\theta}_g, \bar{\theta}_i)$ $i = 1, 2 \dots M$ is found that maximizes the expected log-likelihood using the update formulas (7.20), (7.23), (7.25)-(7.27).

Step 4: let $\lambda = \bar{\lambda}$; if convergence has not been achieved, go to step 2.

7.3 Speech Feature Enhancement Based on a Mixture of Linear Dynamic Models

Assume that speech and noise are mixed linearly in the time domain; this corresponds to a nonlinear mixing in the log spectrum feature space as follows:

$$\mathbf{o}_t = \mathbf{y}_t + \log(1 + \exp(\mathbf{n}_t - \mathbf{y}_t)) \quad (7.29)$$

in which $\mathbf{o}_t$, $\mathbf{y}_t$, and $\mathbf{n}_t$ represent the log spectrum of the noisy speech, the hypothetical clean speech and the noise at the t^{th} frame, respectively. It corresponds to a nonlinear mixing in the cepstral feature (MFCC) space as:

$$\mathbf{o}_t = \mathbf{y}_t + \mathbf{C} \log(\mathbf{I} + \exp(\mathbf{C}^{-1}(\mathbf{n}_t - \mathbf{y}_t))) \quad (7.30)$$

where $\mathbf{C}$ is the DCT matrix and $\mathbf{C}^{-1}$ is the inverse DCT matrix. The observation model relates the cepstrum of the noisy speech $\mathbf{o}_t$ to the cepstrum of the clean speech $\mathbf{y}_t$, and to the cepstrum of the noise $\mathbf{n}_t$ at the t^{th} frame. The model of background noise can be written as:

$$\mathbf{n}_t = \mathbf{A} \mathbf{n}_{t-1} + \boldsymbol{\mu}_n + \mathbf{v}_t \quad (7.31)$$

where $\mathbf{A}$ represents the autoregressive prediction coefficients between consecutive frames of noise in the feature domain, $\boldsymbol{\mu}_n$ is the estimated mean of $\mathbf{n}_t$, $\mathbf{v}_t$ is the autoregressive prediction residual with a zero mean Gaussian distribution and covariance matrix $\mathbf{Q}_n$. Combining (7.5) with (7.29) or (7.30) and (7.31) leads to a nonlinear state space model conditioned on $s_t = i$ as:

$$\mathbf{z}_t = \boldsymbol{\Phi}_t \mathbf{z}_{t-1} + \bar{\mathbf{u}}_t + \bar{\mathbf{e}}_{t,i} \quad (7.32)$$

$$\mathbf{o}_t = f(\mathbf{z}_t) \quad (7.33)$$

with the following definitions:

$$\mathbf{z}_t = [\mathbf{y}_t^T \quad \mathbf{n}_t^T]^T \quad \bar{\mathbf{e}}_{t,i} = [\mathbf{e}_{t,i}^T \quad \mathbf{v}_t^T]^T \quad (7.34)$$

$$\boldsymbol{\Phi}_t = \begin{bmatrix} \mathbf{B}_i & \mathbf{0} \\ \mathbf{0} & \mathbf{A} \end{bmatrix} \quad \bar{\mathbf{u}}_t = \begin{bmatrix} \boldsymbol{\mu}_i \\ \boldsymbol{\mu}_n \end{bmatrix} \quad \mathbf{Q}_t = \begin{bmatrix} \boldsymbol{\Sigma}_i & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_n \end{bmatrix} \quad (7.35)$$

Both noise and clean speech components are incorporated in the state space model and can be estimated simultaneously by the switching Kalman filters. It is noted that since there is no transition probability between s_t and s_{t-1} , the interacting multiple model (IMM) approach is then identical to the GPB1 algorithm, as we have explained in chapter 5. Next, we will show how the GPB algorithm of order 1 can be adapted to the nonlinear state-space model of feature dynamics presented earlier, providing a sub-optimal approximation solution.

Let us define

$$\hat{\mathbf{z}}_{t-1|t-1} = E(\mathbf{z}_{t-1} | \mathbf{o}_{1:t-1}) \quad (7.36)$$

$$\hat{\mathbf{V}}_{t-1|t-1} = \text{cov}(\mathbf{z}_{t-1} | \mathbf{o}_{1:t-1}) \quad (7.37)$$

$$\hat{\mathbf{z}}_{t|t}^i = E[\mathbf{z}_t | \mathbf{o}_{1:t}, s_t = i] \quad (7.38)$$

$$\hat{\mathbf{V}}_{t|t}^i = \text{cov}[\mathbf{z}_t | \mathbf{o}_{1:t}, s_t = i] \quad (7.39)$$

in which $\mathbf{o}_{1:t} = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_t\}$ denotes the noisy speech observation sequence up to time t .

Given a noisy speech feature vector $\mathbf{o}_t$ ($t=1, \dots, T$), $\hat{\mathbf{z}}_{t|t}$ is found from the GPB1 algorithm using a bank of Kalman Filters (KF). For the i th ($i=1, \dots, M$) filter, the Kalman filter algorithm is given as:

$$(\hat{\mathbf{z}}_{t|t}^i, \mathbf{V}_{t|t}^i, L_t^i) = KF(\hat{\mathbf{z}}_{t-1|t-1}, \hat{\mathbf{V}}_{t-1|t-1}, \mathbf{o}_t; \Phi_i, \mathbf{u}_i, \mathbf{Q}_i) \quad (7.40)$$

where $KF()$ represents the filtering operations of the extended Kalman filter $EKF()$ or the unscented Kalman filter $UKF()$. For a detailed description of the EKF and the UKF algorithms, please refer to section 6.3.4 and 6.3.5 respectively.

The posterior probabilities $P(s_t = i | \mathbf{o}_{1:t})$ are updated according to:

$$P(s_t = i | \mathbf{o}_{1:t}) = \frac{1}{\Omega} P(\mathbf{o}_t | s_t = i, \mathbf{o}_{1:t-1}) P(s_t = i | \mathbf{o}_{1:t-1}) \quad (7.41)$$

where Ω is a scaling factor equal to

$$\Omega = \sum_j P(\mathbf{o}_t | s_t = j, \mathbf{o}_{1:t-1}) P(s_t = j | \mathbf{o}_{1:t-1}) \quad (7.42)$$

An assumption is made that $P(\mathbf{o}_t | s_t = i, \mathbf{o}_{1:t-1})$ is Gaussian and computed by

$$P(\mathbf{o}_t | s_t = i, \mathbf{o}_{1:t-1}) = N(\mathbf{o}_t; f(\hat{\mathbf{z}}_{t|t-1}^i), \mathbf{H} \hat{\mathbf{V}}_{t|t-1}^i \mathbf{H}) \quad (7.43)$$

and we assume

$$P(s_t = i | \mathbf{o}_{1:t-1}) \approx P(s_t = i | \hat{\mathbf{z}}_{1:t-1}) = P(s_t = i | \hat{\mathbf{y}}_{1:t-1}) \quad (7.44)$$

$P(s_t = i | \hat{\mathbf{y}}_{1:t-1})$ is calculated according to equation (7.1) or (7.3) and (7.4). It is seen that the previous estimation $\hat{\mathbf{y}}_{1:t-1}$ is used in addition to the present estimation error information in order to decide which state (mode) is present. Since $\hat{\mathbf{y}}_{1:t-1}$ is only an estimate of $\mathbf{y}_{1:t-1}$, it tends to be biased. Equation (7.50) is only an approximation. This has however not been observed to affect the performance of the switching Kalman filter

algorithm adversely in practice. We could also replace $\hat{\mathbf{y}}_{1:t-1}$ by $\gamma\hat{\mathbf{y}}_{1:t-1}$ which resulted in a slightly improved performance. Here, γ is a bias factor, which can be determined from experiments.

The outputs of all filters are merged into a single estimate $\hat{\mathbf{z}}_{t|t}$ and $\hat{\mathbf{V}}_{t|t}$ at time t according to:

$$\hat{\mathbf{z}}_{t|t} = \sum_i P(s_t = i | \mathbf{o}_{1:t}) \hat{\mathbf{z}}_{t|t}^i \quad (7.45)$$

$$\hat{\mathbf{V}}_{t|t} = \sum_i P(s_t = i | \mathbf{o}_{1:t}) \hat{\mathbf{V}}_{t|t}^i + \sum_i P(s_t = i | \mathbf{o}_{1:t}) (\hat{\mathbf{z}}_{t|t}^i - \hat{\mathbf{z}}_{t|t}) (\hat{\mathbf{z}}_{t|t}^i - \hat{\mathbf{z}}_{t|t})^T \quad (7.46)$$

For each frame of incoming noisy speech, the algorithm described above gives the estimate of the speech in the frame.

7.4 Simulation Results and Analysis

7.4.1 Simulation Results on the *Number* v1.3 Database

Pilot experiments were conducted firstly on the small database *Number* 1.3. The database description and experimental settings for the speech recognition engine are given in chapter 4. For the experiments conducted on the database *Number* 1.3, we assume that an AR model of the background noise is known. Clean speech features from 60 clean utterances were used to train the parameters in the mixture of linear dynamic models, as described in Section 7.2. The number of linear dynamic models is determined *a priori* and is 8 in our experiments. The switching between different models is determined by a RBF gating network. The number of hidden neurons for the RBF network was 8, determined by trial and error. For each frame, the noisy 20th order Mel-scaled log spectrum was transformed to clean feature estimation. Finally, the estimated clean speech feature vector was mapped from the log-Mel spectral domain into the cepstral domain (MFCC) using a DCT transform, before being passed on directly to a speech recognizer for the recognition task. If the cepstrum vectors are to be enhanced, the covariance matrices in the SLDM are diagonal matrices. For a fair comparison, we use a square DCT so that the cepstrum vectors have the same dimension as the log-spectral vectors. The

estimated clean speech feature vector will be passed on directly to the speech recognizer for the recognition task.

We found through the experiments that, since the joint noise and speech tracking yields a good estimate of the hypothetical clean speech and noise, continually updating the linearization around the previous speech and noise estimate will give a better approximation of the nonlinear function than approximating the observation equation (7.33) by a piecewise linear model at pre-selected points. Here we use the extended Kalman filters in all of our experiments with the database *Number* v1.3.

As we mentioned in section 7.2, there are two traditional ways to update the parameters of a RBF gating network. Comparing with the RBF-clustering algorithm, all the free network parameters in an RBF-LMS algorithm are adapted simultaneously, which seems to yield an improved overall solution. However, throughout our experiments, we could not observe this advantage. In the following tables, only the results by means of the RBF-clustering algorithm are given.

Tables 7.1 and 7.2 show the recognition accuracy for white noise and babble noise mixtures if the speech features are compensated in the log-spectral domain. Tables 7.3 and 7.4 show the recognition accuracy for white noise and babble noise mixtures if the speech features are compensated for in the cepstral domain. The Extended Kalman Filter algorithm (EKF) is used to deal with the nonlinear distortion by noise in the feature domain. The results for the proposed mixture of linear dynamic models with the ME architecture are entitled with 'SLDM-chapter 7' in the tables. Three schemes have been attempted. In the first scheme, to predict y_t the LDM and the gating network share the same input y_{t-1} . The results are presented by 'SLDM-chapter 7- $p=1, l=1$ -GPB1' in the tables. In another scheme, the input to the gating network is increased and becomes $[y_{t-1}; y_{t-2}]$, while the input of the LDMs is still y_{t-1} . The results are presented by 'SLDM-chapter 7- $p=1, l=2$ -GPB1' in the tables. In the third scheme, the input of the gating network and the input of the LDMs are both $[y_{t-1}; y_{t-2}]$. The results are presented by 'SLDM-chapter 7- $p=2, l=2$ -GPB1' in the tables. Since the dimensions of LDMs are kept

the same as the first scheme, the computational load for the Kalman filtering is the same as the first scheme. Only the mixing probability $P(s_t = j | \mathbf{o}_{1:t-1})$ is calculated differently. These extra computations are much fewer than if the dimension of both the LDMs and the gating network are increased as in the third scheme. The second scheme is unique to the mixture of linear dynamic models and it cannot be achieved by the SLDM proposed in chapter 6, and neither by the switching LDM used in [Kim05][Dro04]. The results obtained by the SLDM methods proposed in chapter 6 are also listed here, denoted by ‘SLDM-chapter 6- $p=1$ -IMM’. For the purpose of comparison, we also list the results with the method proposed in [Kim05], represented by ‘SLDM [Kim05]’ in the tables. The CPU time for the enhancement of one utterance (5-continuous-digits) by the switching EKF algorithm in the log-Mel domain on a Pentium4 2.4GHz personal computer under a Matlab environment is given in Table 7.5

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	50.6	73.4	88.0	93.0	76.3
SLDM [Kim05]	59.0	79.0	87.6	92.8	79.6
SLDM-chapter 6 $p=1$ -IMM	68.0	83.6	92.8	95.0	84.9
SLDM-chapter 7 $p=1, l=1$ -GPB1	65.2	85.0	92.8	94.0	84.3
SLDM-chapter 7 $p=1, l=2$ -GPB1	68.4	87.6	93.6	95.2	86.2
SLDM-chapter 7 $p=2, l=2$ -GPB1	68.4	85.0	92.8	96.0	85.6

Table 7.1: Recognition accuracy (%) for speech mixed with white noise
when speech is enhanced in the log-spectrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	44.0	70.2	85.0	91.0	72.6
SLDM [Kim05]	49.6	73.2	84.6	91.0	74.6
SLDM-chapter 6 $p=1$ -IMM	59.4	74.8	90.4	93.4	79.5
SLDM-chapter 7 $p=1, l=1$ -GPB1	59.6	76.6	88.0	93.6	79.5
SLDM-chapter 7 $p=1, l=2$ -GPB1	62.2	78.6	90.0	93.6	81.1
SLDM-chapter 7 $p=2, l=2$ -GPB1	62.8	78.2	87.4	92.6	80.3

Table 7.2: Recognition accuracy (%) for speech mixed with babble noise
when speech is enhanced in the log-spectrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	50.6	73.4	88.0	93.0	76.3
SLDM [Kim05]	59.4	79.8	87.4	92.2	79.6
SLDM-chapter 6 $p=1$ -IMM	67.2	86.0	92.6	94.0	85.0
SLDM-chapter 7 $p=1, l=1$ -GPB1	65.6	82.6	91.0	95.0	83.6
SLDM-chapter 7 $p=1, l=2$ -GPB1	65.4	83.6	91.8	94.0	83.7

Table 7.3: Recognition accuracy (%) for speech mixed with white noise
when speech is enhanced in the cepstrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	44.0	70.2	85.0	91.0	72.6
SLDM [Kim05]	50.4	73.2	84.0	91.0	74.6
SLDM-chapter 6 $p=1$ -IMM	59.4	77.4	89.0	92.4	79.6
SLDM-chapter 7 $p=1, l=1$ -GPB1	53.0	74.8	87.0	92.2	76.8
SLDM-chapter 7 $p=1, l=2$ -GPB1	53.2	73.2	84.8	91.6	75.7

Table 7.4: Recognition accuracy (%) for speech mixed with babble noise
when speech is enhanced in the cepstrum domain

SLDM-chapter 6 $p=1$ -GPB1	3.17
SLDM-chapter 6 $p=1$ -IMM	3.92
SLDM-chapter 7 $p=1, l=2$ -GPB1	3.20

Table 7.5 CPU time (sec) for the enhancement of one utterance
when speech is enhanced in the log-spectrum domain

The simulation results on the *Number* v1.3 database show that

- For both white noise and babble noise, the mixture of linear dynamic models approach appears to be effective at improving the recognition performance, and it can provide better results compared with its SLDM counterpart in [Kim05]. The difference between these two methods compared in Figure 7.2 is that the structure of the proposed mixture of linear dynamic models is able to access external information in order to estimate the *a priori* probability for each linear dynamic model. We thus conclude that the introduction of external information in calculating the state (mode)

probabilities for linear dynamic models can improve the performance of switching LDM systems in speech feature compensation for robust speech recognition.

- The results obtained in the log-Mel domain seem to be better than those in the cepstral domain. This phenomenon is unique to the mixture of linear dynamic models structure. In chapter 5 and chapter 6, we observed similar performances in these two areas. A possible explanation is that both GMM and HMM-based speech models don't make much use of the spatial correlation in modeling, while this information is useful for the SLDM mixture of linear dynamic models structure. At the same time, we notice that, when speech is enhanced in the log-Mel domain, the method 'SLDM-chapter 7- $p=1, l=2$ -GPB1' outperforms 'SLDM-chapter 7- $p=1, l=1$ -GPB1', especially at SNRs between 0 dB and 10 dB, while in the cepstral domain, this advantage cannot be observed. It seems that the gating network can take advantage of the spatial correlations among components of speech features to segment the input space. In future work, we could investigate further whether the method 'SLDM-chapter 7- $p=1, l=3$ -GPB1' for speech enhancement in the cepstral domain could compensate for the lack of information in the input-space segmentation and provide better results. We also notice that the method 'SLDM-chapter 7- $p=2, l=2$ -GPB1' didn't bring further performance improvement, compared with the method 'SLDM-chapter 7- $p=1, l=2$ -GPB1', meaning that the use of second order information at the input of the LDMs in the proposed ME-based structure is not useful.
- In the log-Mel domain, the method 'SLDM-chapter 7- $p=1, l=2$ -GPB1' gives a better performance than the linear predictive HMM approach represented by 'SLDM-chapter 6- $p=1$ -IMM'.
- From Table 7.5, it is seen that the processing time needed to estimate speech features with 'SLDM-chapter 7- $p=1, l=2$ -GPB1' is nearly the same as those with the 'SLDM-chapter 6- $p=1$ -GPB1' approach, and thus is less than the 'SLDM-chapter 6- $p=1$ -IMM' approach.

7.4.2 Simulation Results on the *Aurora 2* Database

The proposed algorithms were also tested on the *Aurora 2* database. The database description and experimental settings for the speech recognition engine are given in chapter 4. The feature generation was modified slightly from the reference implementation. In particular, we changed from using the spectral magnitude to using the power spectral density as the input to the Mel-frequency filter-bank to make equation (7.33) stand. These settings are the same as those in [Dro04],[Kim05].

All the experiments used the same speech model with 16 hidden linear dynamic models (LDMs). For each noise type, a specific noise model was built. The parameters for the noise model were trained on the first ten frames of the noisy utterances since usually the first ten frames are assumed to be speech free. The experimental conditions to obtain the noise model were also maintained the same as those in [Dro04][Kim05]. Feature vectors were enhanced in the log-spectral domain.

We used a radial basis function (RBF) network of the Moody-Darken type [Moo89] with 32 hidden neurons in the hidden layer as the gating network. The number of hidden neurons for the RBF network was determined by trial and error.

Detailed recognition rates (%) for each of the four noise conditions and each of the noisy speech SNRs in the *Aurora* set-A are provided in Table 7.6, for the proposed SLDM approach. In the experiment, the input to the gating network is $[y_{t-1}; y_{t-2}]$, while the input of the LDMs is y_{t-l} , which corresponds to the scheme ‘SLDM-chapter 7- $p=1, l=2$ -GPB1’ in section 7.5.1. When the unscented Kalman filter was used in this approach, the results obtained are shown in Table 7.7.

SNR	Subway	Babble	Car	Exhibition	Average
20 dB	98.13	98.46	98.84	97.87	98.33
15 dB	96.78	97.07	97.97	96.20	97.01
10 dB	91.71	93.47	94.48	91.02	92.67
5 dB	81.18	81.26	82.64	79.05	81.03
0 dB	56.77	47.34	52.43	48.50	51.26
Average	84.91	83.52	85.27	82.52	84.06

Table 7.6 Accuracy (%) with speech enhanced by the switching extended Kalman filter based on the proposed SLDM

SNR	Subway	Babble	Car	Exhibition	Average
20dB	98.13	98.46	98.84	97.87	98.33
15dB	96.46	97.38	98.33	96.42	97.15
10dB	92.74	94.32	95.75	92.58	93.84
5dB	84.70	86.08	86.96	83.97	85.43
0dB	64.13	57.33	65.07	57.64	61.04
Average	87.23	86.71	88.99	85.70	87.16

Table 7.7 Accuracy (%) with speech enhanced by the switching unscented Kalman filter based on the proposed SLDM

The simulation results on the *Aurora* 2.0 database are summarized in the Table 7.8 with results averaged across 0 dB to 20 dB. For the purpose of comparison, the *Aurora* 2.0 database results reported in [Kim05] and [Dro04] are also listed in Table 7.8, represented as ‘SLDM[Kim05]’ and ‘SLDM[Dro04]’ respectively. The results obtained by our proposed approach based on the SLDM with ME structure are presented as ‘SLDM chapter 7- $p=1, l=2$ -GPB1-EKF’ and ‘SLDM chapter 7- $p=1, l=2$ -GPB1-UKF’ in the table. Our previous results in chapter 6 are represented as ‘SLDM chapter 6- $p=1$ -IMM -EKF’

and 'SLDM chapter 6- $p=1$ -IMM -UKF' in the table. The baseline results without noise processing are also given. The number of linear dynamic models needed to obtain the results presented in Table 7.8 are given in Table 7.9. Table 7.10 shows the CPU time for the enhancement of one utterance (5-continuous-digits) by the switching EKF algorithm on a Pentium4 2.4GHz personal computer under a Matlab environment.

	Subway	Babble	Car	Exhibition	Average
Baseline	70.22	51.35	60.28	65.39	61.81
SLDM [Kim05]	80.55	67.52	86.19	77.82	78.02
SLDM [Dro04]	80.48	80.98	86.82	84.53	83.20
SLDM chapter 6 $p=1$,IMM-EKF	84.90	83.67	86.41	83.37	84.59
SLDM chapter 7 $p=1,l=2$ -GPB1-EKF	84.91	83.52	85.27	82.52	84.06
SLDM chapter 6 $p=1$,IMM-UKF	89.13	86.28	91.57	88.47	88.86
SLDM chapter 7 $p=1,l=2$ -GPB1-UKF	87.23	86.71	88.99	85.70	87.16

Table 7.8 Summary of accuracy (%) on *Aurora 2*, set A.

Number of LDMs in the SLDM	
SLDM [Kim05]	128
SLDM [Dro04]	16
SLDM chapter 6	16
SLDM chapter 7	16

Table 7.9 Number of LDMs in the SLDM

SLDM-chapter 6 $p=1$ -GPB1-EKF	11.31
SLDM-chapter 6 $p=1$ -IMM-EKF	19.17
SLDM-chapter 7 $p=1, l=2$ -GPB1-EKF	11.33

Table 7.10 CPU time (sec) for the enhancement of one utterance
when speech is enhanced by the switching EKF algorithm

The results show that our SLDM approach outperforms the SLDM approach from [Dro04],[Kim05]. More specifically, with the same number of LDMs in the SLDM, the proposed SLDM approach clearly outperforms the SLDM approach in [Dro04], especially when the environmental noise is babble noise. As we compare our results with those obtained in [Kim05], even with only 1/8 of the computational load used for the experiments in [Kim05], a better performance is obtained, especially when the environmental noise is babble noise and subway noise. If we replace the Extended Kalman filter with the Unscented Kalman filter, a clear performance improvement can also be observed. It has been shown in [Dro04] that with the increase of the number of LDMs in the SLDM, better results could be achieved. Therefore, it is promising that if we increase the number of the linear dynamic models in the SLDM, we should obtain even better results than our present results.

Comparing our results with those obtained in chapter 6, we find that our SLDM approach in chapter 6 performs a little bit better than the SLDM approach proposed in chapter 7. This conclusion is different from the one obtained based on the *Number* v1.3 database results. We explain the disagreement as follows: in the ME structure-based SLDM approach, the previous filtered estimation was used to predict the present mode of the SLDM. The filtered estimation tends to be biased, especially under the experimental conditions of the *Aurora 2*, where the noise model only gives a very coarse estimation of the true noise. In other words, when the side effect of inaccurate noise modeling cannot

be ignored, the approach in chapter 7 no longer performed better than the one in chapter 6.

7.5 Summary

In summary, this chapter presented a new speech features enhancement method based on a switching linear dynamic model with a mixture of experts (ME) architecture, named as a mixture of linear dynamic models. The advantage of the proposed mixture of linear dynamic models is its capability to access external information in order to estimate the prior probability for each linear dynamic model. We have carried out experiments to evaluate the effects of introducing the external information into the SLDM for the speech feature enhancement application. Simulation results have confirmed the improved performance of the proposed approach over some previously published SLDM methods, and the improvement of performance over the SLDM approaches proposed in chapter 6 under ideal noise training conditions (*Number* v1.3 database).

Chapter 8

Time-Variant Linear-Predictive-HMM for Speech Feature Enhancement

In chapter 6 a switching linear dynamic model (LDM), named linear-predictive-HMM, was proposed to model the distribution of speech features. One problem with the linear-predictive-HMM is that it is unable to describe the model transitions at each time point precisely, since the transition property is defined over the whole process. In this chapter, we propose a novel approach to enhance speech features under noisy environments based on a SLDM named here as a time-variant linear-predictive-HMM. It has a similar structure to the linear-predictive-HMM in chapter 6, but the transition properties between the LDMs are time-variant. We will show that the time-variant linear-predictive-HMM model can be interpreted as a combination of the linear-predictive-HMM approach in chapter 6 and the mixture of linear dynamic models approach in chapter 7.

Again, it should be noted that there are two types of state variables in a switching linear dynamic model (LDM): discrete or continuous. They have different functions in the SLDM and in our applications. To distinguish them, we will label the discrete state as ‘state (mode)’ in the following sections, meaning different discrete states representing different dynamical “modes”.

8.1 Introduction

The idea of introducing time-variant transition probabilities to a Hidden Markov Model (HMM) was first proposed in [Lie99]. It was named as Hidden Markov Mixtures of Experts (HMME) and was applied to the analysis of human electroencephalogram (EEG) signals. This HMME model was further studied in [Koh00] to identify the non-stationary dynamics in physiological recordings. A similar model was proposed in [Wan04] for time series signal prediction and was named as the Time-line Hidden Markov Experts

(THME). In the HMME and the THME, rather than holding a transition probability for the whole process, the transition property between the experts is localized at each time point by a connectionist model named as the transition network.

In this chapter, we will introduce the time-variant transition probabilities to the linear-predictive-HMM that we have studied in chapter 6 for the purpose of speech feature enhancement under noisy environments. The main innovations of the current work are as follows: 1) previous work [Lie99][Koh00][Wan04] had studied the univariate, time domain signal. This is the first time that the switching linear dynamic model with time-variant transition probabilities is applied to model the multivariate speech feature vectors; 2) a new topology for the transition network is proposed; and 3) it is the first time that a time-variant linear-predictive-HMM is proposed to enhance speech features under noisy environments.

It is noticeable that there are important differences between the time-variant linear-predictive-HMM and the linear-predictive-HMM described in chapter 6 and the mixture of linear dynamic models with the ME architecture studied in chapter 7, as shown in figure 8.1. The shaded nodes in figure 8.1 are observed, and the clear nodes are hidden. According to the previous linear-predictive-HMM model in figure 8.1(a), if there is no noise and $\mathbf{x}_{1:t}$ can be observed, the calculation of the posterior probabilities $p(s_t = j | \mathbf{x}_{1:t})$ makes use of the predictor performance information $p(\mathbf{x}_t | s_t = j, \mathbf{x}_{1:t-1})$, the previous state (mode) information $p(s_{t-1} = i | \mathbf{x}_{1:t-1})$, and the relation

$$p(s_t = j | \mathbf{x}_{1:t}) = \frac{1}{\Omega} p(\mathbf{x}_t | s_t = j, \mathbf{x}_{1:t-1}) \sum_i p(s_t = j | s_{t-1} = i, \mathbf{x}_{1:t-1}) p(s_{t-1} = i | \mathbf{x}_{1:t-1}) \quad (8.1)$$

where Ω is a scaling factor equal to

$$\Omega = \sum_j p(\mathbf{x}_t | s_t = j, \mathbf{x}_{1:t-1}) \sum_i p(s_t = j | s_{t-1} = i, \mathbf{x}_{1:t-1}) p(s_{t-1} = i | \mathbf{x}_{1:t-1}) \quad (8.2)$$

Here the transition probability $p(s_t = j | s_{t-1} = i, \mathbf{x}_{1:t-1})$ is time-independent, i.e. $p(s_t = j | s_{t-1} = i, \mathbf{x}_{1:t-1}) = p(s_t = j | s_{t-1} = i)$. The mixture of linear dynamic models in

figure 8.1(b) uses the input information $p(s_t = j | \mathbf{x}_{1:t-1})$ and the predictor performance information. The posterior probability $p(s_t = j | \mathbf{x}_{1:t})$ is then calculated as:

$$p(s_t = j | \mathbf{x}_{1:t}) = \frac{1}{\Omega} p(\mathbf{x}_t | s_t = j, \mathbf{x}_{1:t-1}) p(s_t = j | \mathbf{x}_{1:t-1}) \quad (8.3)$$

where Ω is a scaling factor equal to

$$\Omega = \sum_j p(\mathbf{x}_t | s_t = j, \mathbf{x}_{1:t-1}) p(s_t = j | \mathbf{x}_{1:t-1}) \quad (8.4)$$

The new speech model in figure 8.1(c) can be interpreted as a combination of the mixture of linear dynamic models and the linear-predictive-HMM. It has integrated the following information: (1) input information (2) predictor performance information (3) previous state (mode) information, to calculate the posterior probabilities $p(s_t = j | \mathbf{x}_{1:t})$:

$$p(s_t = j | \mathbf{x}_{1:t}) = \frac{1}{\Omega} p(\mathbf{x}_t | s_t = j, \mathbf{x}_{1:t-1}) \sum_i p(s_t = j | s_{t-1} = i, \mathbf{x}_{1:t-1}) p(s_{t-1} = i | \mathbf{x}_{1:t-1}) \quad (8.5)$$

where Ω is a scaling factor equal to

$$\Omega = \sum_j p(\mathbf{x}_t | s_t = j, \mathbf{x}_{1:t-1}) \sum_i p(s_t = j | s_{t-1} = i, \mathbf{x}_{1:t-1}) p(s_{t-1} = i | \mathbf{x}_{1:t-1}) \quad (8.6)$$

The transition probability $p(s_t = j | s_{t-1} = i, \mathbf{x}_{1:t-1})$ has its full meaning since it also depends on the values $\mathbf{x}_{1:t-1}$. The model shown in figure 8.1(c) can be treated as a combination of the models in figure 8.1(a) and (b). Therefore the new model is more general than the previous ones.

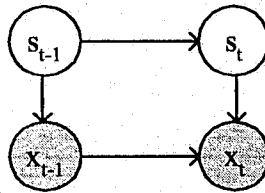


Fig 8.1(a) Graphical representation of the linear-predictive HMM proposed in chapter 6

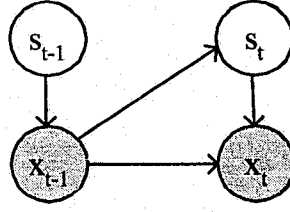


Fig 8.1(b) Graphical representation of the mixture of linear dynamic models proposed in chapter 7

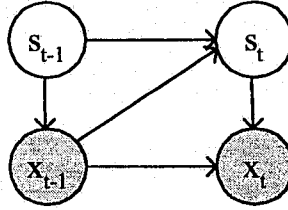


Fig 8.1(c) Graphical representation of the time-variant linear-predictive HMM

Consider a set of M linear dynamic models $\{\theta_i\}_{i=1}^M$ at time step t , $1 \leq t \leq T$. The i^{th} linear model with $i \in (1, \dots, M)$ provides a prediction $\mathbf{x}_{t,i}$ for the target $\mathbf{x}_t$ with

$$\mathbf{x}_{t,i} = \varphi_i(\mathbf{x}_{t-1}, \theta_i) = \boldsymbol{\mu}_i + \mathbf{B}_i \mathbf{x}_{t-1} + \mathbf{e}_{t,i}, \quad \mathbf{e}_{t,i} \sim N(\mathbf{0}, \boldsymbol{\Sigma}_i) \quad (8.7)$$

The parameters for each LDM are $\theta_i = \{\boldsymbol{\mu}_i, \mathbf{B}_i, \boldsymbol{\Sigma}_i\}$, where $\mathbf{B}_i$ represents the linear regression coefficients between consecutive frames of clean speech in the feature domain, $\boldsymbol{\mu}_i$ is the estimated mean of $\mathbf{x}_{t,i}$, and $\mathbf{e}_{t,i}$ is the prediction residual with a zero mean Gaussian distribution and covariance matrix $\boldsymbol{\Sigma}_i$.

If each LDM is associated with a hidden state (mode) in a Markov chain, the state (mode) transition along the Markov chain is the process of activating different LDMs with different probabilities. If the transition probabilities between the hidden states (modes) are based on a transition matrix defined over the whole process

$$a_{ij} = P(q_t = j | q_{t-1} = i) \quad (8.8),$$

we then have the linear-predictive-HMM presented in chapter 6. If the transition probabilities between the hidden states (modes) are based on a time-variant transition matrix we have

$$P(q_t = j | q_{t-1} = i, \bar{X}_t, \theta_g) = a_t^{ij}(\bar{X}_t, \theta_g). \quad (8.9)$$

which means that the transition property at each time point depends on the actual location $\bar{X}_t$ in the space, resulting in the time-variant linear-predictive-HMM. $\bar{X}_t$ is the vector of past $\mathbf{x}_t$ values, $\bar{X}_t = [\mathbf{x}_{t-1}, \mathbf{x}_{t-2}, \dots, \mathbf{x}_{t-\tau}]$, and θ_g contains the parameters of a connectionist model, such as a neural network, that can model the input-related state (mode) transition probabilities.

The structure for the time-variant linear-predictive-HMM is shown in Figure 8.2.

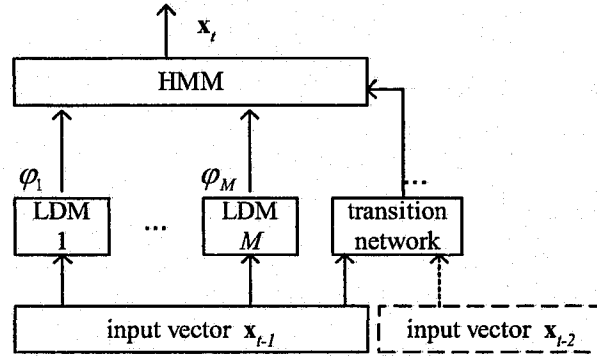


Fig. 8.2 Block diagram of the proposed time-variant linear-predictive-HMM

The interesting point in this context is the input-dependent transition network $a_t^{ij}(\bar{X}_t, \theta_g)$, which is usually modeled by a neural network. We named it as a transition network in figure 8.2. The structure of the transition network is shown in figure 8.3. The outputs of the neural network have dimensions $M \times M$, representing the transition probabilities between LDMs.

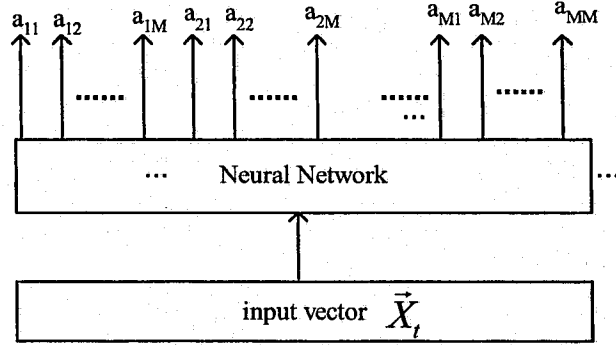


Fig. 8.3 Block diagram of the transition network used in [Lie99][Wan04]

One problem with the above transition network is that, as the number of LDMs increases, the output dimensions of the neural network increase quadratically. The input and output relationship $a_t^{ij}(\vec{X}_t, \theta_g)$ then becomes complex and a huge neural network with several thousand weights may be needed. Instead, we propose to use a group of small networks as shown in Figure 8.4. One of the advantages of using a group of smaller networks is that the function of the transition network can be better interpreted. If the i th small network is treated as the precondition $q_{t-1} = i$, the input and output relationship for the network i can be easily interpreted as

$$a_t^{ij}(\vec{X}_t, \theta_g) = P(q_t = j | q_{t-1} = i, \vec{X}_t, \theta_g)$$

Furthermore, the new topology results in neural networks with a smaller total number of weights than the topology shown in figure 8.3, and consequently faster training was achieved in our experiments.

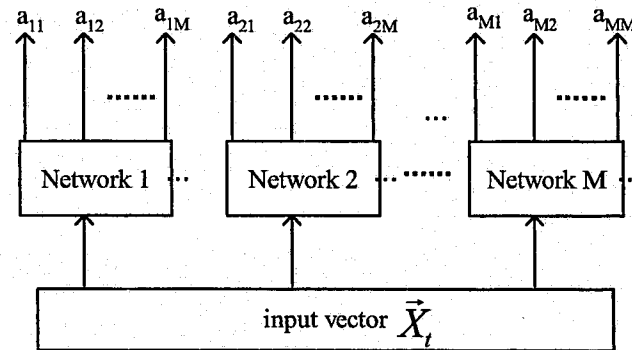


Fig. 8.4 Block diagram of the proposed transition network

A multilayer feedforward neural network (MLP) or a radial basis function (RBF) neural network can be applied as the nonlinear transition network. For the i^{th} network, we have the network (internal) outputs $\alpha_{ij} = \psi(\vec{X}_t, \theta_{g,i})$, where $\theta_{g,i}$ is the set of parameters characterizing the i^{th} network. The direct outputs α_{ij} are exponentiated and normalized by their sum, giving the final gating outputs $a_t^{ij}(\vec{X}_t, \theta_{g,i})$:

$$a_t^{ij}(\vec{X}_t, \theta_{g,i}) = P(q_t = j | q_{t-1} = i, \vec{X}_t, \theta_{g,i}) = \frac{\exp(\alpha_{ij})}{\sum_{j=1}^M \exp(\alpha_{ij})} \quad (8.10).$$

The definition of a_t^{ij} implies that they are strictly positive, smaller than 1 and that the sum of all a_t^{ij} over j is equal to 1.

8.2 Time-Variant Linear-Predictive-HMM as a Speech Model

In this section, we show how to estimate the parameters of the time-variant linear-predictive-HMM model, namely $\theta_i = \{\mu_i, \mathbf{B}_i, \Sigma_i\}, i \in (1, \dots, M)$, as well as the parameters for the transition network $\theta_g = \{\theta_{g,i}\}_{i=1}^M$ and the initial probability π , from a sequence of speech feature observations $\mathbf{X} = \{\mathbf{x}_1 \mathbf{x}_2 \dots \mathbf{x}_T\}$ using an Expectation-Maximisation (EM) [Dem77] algorithm. For convenience, we use the compact notation $\lambda = (\{\theta_i\}_{i=1}^M, \{\theta_{g,i}\}_{i=1}^M, \pi)$ to indicate the complete parameter set of the model.

The EM algorithm requires us to maximize $Q(\lambda, \bar{\lambda})$, where $\bar{\lambda} = (\{\bar{\theta}_i\}_{i=1}^M, \{\bar{\theta}_{g,i}\}_{i=1}^M, \bar{\pi})$ is defined as the reestimated model. The joint likelihood of an observation sequence $\mathbf{X} = \{\mathbf{x}_1 \mathbf{x}_2 \dots \mathbf{x}_T\}$ and a hidden state (mode) sequence $\mathbf{q} = \{q_1 q_2 \dots q_T\}$ can be written as

$$P_{\bar{\lambda}}(\mathbf{X}, \mathbf{q}) = P_{\bar{\lambda}}(q_1) \prod_{t=2}^T P_{\bar{\lambda}}(q_t | q_{t-1}, \mathbf{X}) \prod_{t=1}^T P_{\bar{\lambda}}(\mathbf{x}_t | q_t) \quad (8.11)$$

The joint log likelihood thus becomes

$$\log P_{\bar{\lambda}}(\mathbf{X}, \mathbf{q}) = \log P_{\bar{\lambda}}(q_1) + \sum_{t=2}^T \log P_{\bar{\lambda}}(q_t | q_{t-1}, \mathbf{X}) + \sum_{t=1}^T \log P_{\bar{\lambda}}(\mathbf{x}_t | q_t) \quad (8.12).$$

The EM algorithm requires us to maximize

$$\begin{aligned} Q(\lambda, \bar{\lambda}) &= \sum_{\mathbf{q}} P_{\lambda}(\mathbf{q} | \mathbf{X}) \log P_{\bar{\lambda}}(\mathbf{X}, \mathbf{q}) \\ &= \sum_{\mathbf{q}} P_{\lambda}(\mathbf{q} | \mathbf{X}) \left(\log P_{\bar{\lambda}}(q_1) + \sum_{t=2}^T \log P_{\bar{\lambda}}(q_t | q_{t-1}, \mathbf{X}) + \sum_{t=1}^T \log P_{\bar{\lambda}}(\mathbf{x}_t | q_t) \right) \end{aligned} \quad (8.13),$$

where $\sum_{\mathbf{q}}$ denotes a sum over all the possible hidden state (mode) sequences. The above equation consists of three terms; the first for the initial probability π_i , the second for the state (mode) transition probability parameters, and the third for the observation model parameters θ_i . It can then be rewritten as

$$Q_1 = \sum_i P_{\lambda}(q_1 = i | \mathbf{X}) \log P_{\bar{\lambda}}(q_1 = i) \quad (8.14)$$

$$Q_2 = \sum_i \sum_j \sum_t P_{\lambda}(q_t = j, q_{t-1} = i | \mathbf{X}) \log P_{\bar{\lambda}}(q_t | q_{t-1}, \mathbf{X}) \quad (8.15)$$

$$Q_3 = \sum_i \sum_t P_{\lambda}(q_t = i | \mathbf{X}) \log P_{\bar{\lambda}}(\mathbf{x}_t | q_t) \quad (8.16)$$

The three terms in equation (8.16) can be maximized separately, giving rise to parameter update equations for each different part of the model.

The update formula for the initial state (mode) distribution parameters and the transition network can be found by maximizing equations (8.17),(8.18). Adding the Lagrange

multiplier δ and with the constraint that $\sum_{i=1}^M \pi_i = 1$, we have

$$\frac{\partial}{\partial \pi_i} [Q_1 + \delta (\sum_{i=1}^M \pi_i - 1)] = 0 \quad (8.17)$$

The update formula for $\bar{\pi}_i$ becomes

$$\bar{\pi}_i = P(q_1 = i | \mathbf{X}) = \gamma_1(i) \quad (8.18)$$

In a similar way, we use the Lagrange multiplier δ and the constraint that $\sum_{j=1}^M a_t^{ij} = 1$,

$$\frac{\partial}{\partial a_t^{ij}} [Q_2 + \delta (\sum_{j=1}^M a_t^{ij} - 1)] = 0 \quad (8.19).$$

We get the following formula:

$$\frac{1}{\alpha_t^{ij}} P(q_{t-1} = i, q_t = j | \mathbf{X}) + \delta = 0 \quad (8.20)$$

The transition network update equation is obtained as

$$\bar{\alpha}_t^{ij} = \frac{\xi_t(i, j)}{\gamma_t(i)} \quad (8.21)$$

where

$$\gamma_t(i) = P(q_t = i | \mathbf{X}, \boldsymbol{\theta}_g), \quad t = 1, \dots, T \quad (8.22)$$

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j | \mathbf{X}, \boldsymbol{\theta}_g) \quad (8.23).$$

These are defined in the same way as in a standard HMM, and can be calculated by the forward-backward algorithm [Rab93].

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^N \alpha_t(i) a_{t+1}^{ij}(\boldsymbol{\theta}_g) \right] b_j(\mathbf{o}_{t+1}) \quad 1 \leq t \leq T-1 \quad (8.24)$$

$$\beta_t(i) = \sum_{j=1}^N b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j) a_{t+1}^{ij}(\boldsymbol{\theta}_g) \quad t = T-1, T-2, \dots, 1 \quad (8.25)$$

$$\gamma_t(i) = P(q_t = i | \mathbf{X}, \boldsymbol{\theta}_g) = \frac{\alpha_t(i) \beta_t(i)}{\sum_{i=1}^M \alpha_t(i) \beta_t(i)} \quad (8.26)$$

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j | \mathbf{X}, \boldsymbol{\theta}_g) = \frac{\alpha_t(i) a_{t+1}^{ij}(\boldsymbol{\theta}_g) b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^M \sum_{j=1}^M \alpha_t(i) a_{t+1}^{ij}(\boldsymbol{\theta}_g) b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)} \quad (8.27)$$

We then focus on the update formula for $\bar{\boldsymbol{\theta}}_g$. Since we use a neural network as the connectionist model, with nonlinear hidden units, the weights of the transition network cannot be computed directly but require iterative techniques. The weight changes in the i^{th} , $i \in (1, \dots, M)$, transition network are proportional to the gradient of the cost function with respect to the intermediate variable $\alpha_{ij}(\bar{X}_t, \boldsymbol{\theta}_{g,i})$ (prior to the exponentiation and normalization). Maximizing the log-likelihood is equal to minimizing the negative loglikelihood. If $Q(\boldsymbol{\theta}_{g,i}, \bar{\boldsymbol{\theta}}_{g,i})$ is interpreted as an error function to be minimized, we have

$$\frac{\partial}{\partial \alpha_{ij}} Q = -(\bar{\alpha}_t^{ij} - \alpha_{ij}(\bar{X}_t, \bar{\boldsymbol{\theta}}_{g,i})) \quad (8.28)$$

$$\frac{\partial Q}{\partial \bar{\theta}_{g,i}} = \frac{\partial Q}{\partial \alpha_{ij}} \frac{\partial \alpha_{ij}}{\partial \bar{\theta}_{g,i}} = -(\bar{a}_t^{ij} - a_{ij}(\bar{X}_t, \bar{\theta}_{g,i})) \frac{\partial \alpha_{ij}}{\partial \bar{\theta}_{g,i}} \quad (8.29)$$

Therefore, for both RBF and MLP networks, the parameters can be found by the gradient-descent method. The local-optimum property of the gradient-descent algorithm in this M-step makes the learning algorithm described above a generalized EM algorithm [Dem77].

-Observation model

An update formula for the observation model parameters can be found by maximizing

$$Q_3 = \sum_i \sum_t P_\lambda(q_t = i | \mathbf{X}) \log P_{\bar{\lambda}}(\mathbf{x}_t | q_t) \quad (8.30)$$

For a Gaussian observation model we have

$$P_{\bar{\lambda}}(\mathbf{x}_t | q_t = i) = (2\pi)^{-d/2} |\bar{\Sigma}_i|^{-1/2} \exp(-\frac{1}{2}[\mathbf{x}_t - \bar{\mathbf{B}}_i \mathbf{x}_{t-1} - \bar{\boldsymbol{\mu}}_i]' \bar{\Sigma}_i^{-1} [\mathbf{x}_t - \bar{\mathbf{B}}_i \mathbf{x}_{t-1} - \bar{\boldsymbol{\mu}}_i]) \quad (8.31)$$

By substituting this into (8.16) and dropping terms that are independent, the objective function becomes

$$Q_3 = -\frac{1}{2} \sum_t \sum_i \gamma_t(i) \log |\bar{\Sigma}_i| - \frac{1}{2} \sum_t \sum_i \gamma_t(i) E_{ii}[(\mathbf{x}_t - \bar{\mathbf{B}}_i \mathbf{x}_{t-1} - \bar{\boldsymbol{\mu}}_i)' \bar{\Sigma}_i^{-1} (\mathbf{x}_t - \bar{\mathbf{B}}_i \mathbf{x}_{t-1} - \bar{\boldsymbol{\mu}}_i)] \quad (8.32)$$

This objective function is the same as the one for linear-predictive-HMMs in chapter 6. The detailed optimization procedure to find the update equation for $\bar{\mathbf{B}}_i$, $\bar{\boldsymbol{\mu}}_i$ and $\bar{\Sigma}_i$ is thus not reproduced here.

The time-variant linear-predictive-HMM trained using the EM algorithm is guaranteed to reach only a local maximum likelihood solution. Because there are many local maxima, such models are sensitive to how they are initialized. For this reason, we need to initialize the model rather carefully. Both the parameters of the transition network and the linear dynamical models must be initialized. The initialization algorithm is summarized as follows:

1. Use the technique developed in chapter 6 to obtain the initial estimate for the parameters of a linear-predictive-HMM, with the transition property defined over the whole process. The parameters for the linear dynamical models $\{\boldsymbol{\theta}_i\}_{i=1}^M$ are then treated as the initial value for the time-variant linear-predictive-HMM.

2. The *Viterbi algorithm* [Rab93] is used to search for the maximum likelihood linear-predictive-HMM hidden state (mode) sequence $\mathbf{q} = \{q_1, q_2, \dots, q_T\}$.
3. For a RBF gating network, the initial RBF centers and variance parameters are obtained by an unsupervised K-means clustering algorithm. The initial weights in the RBF output layer are randomly set to small values. If the gating model is a MLP network, the initial weights in the MLP hidden layers are randomly set to small values.
4. Conditioned on $q_{t-1} = i$ obtained in step 2, the i^{th} transition network is trained with a series of input-output pairs $\{ \bar{X}_t, a_t^j(\bar{X}_t, \theta_g) = P(q_t = j | q_{t-1} = i) \}$. Note that at the initialization stage the transition probabilities $P(q_t = j | q_{t-1} = i)$ are input-independent and were found in step 1.
5. The parameters of the transition network obtained in step 4 are used as the initial setups for the transition network.

Now, the EM algorithm can be summarized as follows:

Step 1: Initialize $\lambda = (\{\theta_i\}_{i=1}^M, \theta_g, \pi)$

Step 2: Find the expected state (mode) occupancy: $\gamma_t(i) = P(q_t = i | \mathbf{X})$, $t = 1, \dots, T$

Step 3: A new set of parameters $\bar{\lambda} = (\{\bar{\theta}_i\}_{i=1}^M, \bar{\theta}_g, \bar{\pi})$ is found that maximizes the expected log-likelihood.

Step 4: let $\lambda = \bar{\lambda}$. If convergence has not been achieved, go to step 2.

8.3 Speech Feature Enhancement Based on Time-Variant Linear-Predictive-HMM

Given a time-variant linear-predictive-HMM as the speech model, a sequential approach to estimate the clean speech components from noisy observations will now be developed. As we found in the previous chapters that speech enhancement in the cepstral domain brought similar results as those in the log-spectral domain, in this chapter we only describe and test the proposed approach in the log-spectral domain.

Given that the speech model is at mode j at time t , $\mathbf{x}_t$ is defined as

$$\mathbf{x}_{t,j} = \boldsymbol{\mu}_j + \mathbf{B}_j \mathbf{x}_{t-1} + \mathbf{e}_{t,j}, \quad \mathbf{e}_{t,j} \sim N(\mathbf{0}, \boldsymbol{\Sigma}_j) \quad (8.33)$$

The nonlinear acoustic model in the log spectrum feature space is given as:

$$\mathbf{o}_t = \mathbf{x}_t + \log(\mathbf{I} + \exp(\mathbf{n}_t - \mathbf{x}_t)) \quad (8.34)$$

where $\mathbf{o}_t$, $\mathbf{x}_t$ and $\mathbf{n}_t$ are the log spectrum of noisy speech, the hypothetical clean speech and the noise at the t^{th} frame, respectively. Taking into account the time-varying characteristics of the background noise, we model the sequence of noise features as the output of a first-order auto-regressive (AR) system excited by a zero mean Gaussian process $\mathbf{v}_t$ with a covariance matrix $\mathbf{Q}_n$ as [Raj04]:

$$\mathbf{n}_t = \mathbf{A} \mathbf{n}_{t-1} + \boldsymbol{\mu}_n + \mathbf{v}_t \quad (8.35)$$

Combining (8.41) with (8.42) and (8.43) leads to a nonlinear state space model in the log-spectral domain, conditioned on $s_t = j$:

$$\mathbf{z}_t = \boldsymbol{\Phi}_j \mathbf{z}_{t-1} + \bar{\mathbf{u}}_j + \bar{\mathbf{e}}_{t,j} \quad (8.36)$$

$$\mathbf{o}_t = f(\mathbf{z}_t) = \mathbf{x}_t + \log(\mathbf{I} + \exp(\mathbf{n}_t - \mathbf{x}_t)) \quad (8.37)$$

with the following definitions:

$$\mathbf{z}_t = [\mathbf{x}_t^T \quad \mathbf{n}_t^T]^T \quad \bar{\mathbf{e}}_{t,j} = [\mathbf{e}_{t,j}^T \quad \mathbf{v}_t^T]^T \quad (8.38)$$

$$\boldsymbol{\Phi}_j = \begin{bmatrix} \mathbf{B}_j & \mathbf{0} \\ \mathbf{0} & \mathbf{A} \end{bmatrix} \quad \bar{\mathbf{u}}_j = \begin{bmatrix} \boldsymbol{\mu}_j \\ \boldsymbol{\mu}_n \end{bmatrix} \quad \mathbf{Q}_j = \begin{bmatrix} \boldsymbol{\Sigma}_j & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_n \end{bmatrix} \quad (8.39)$$

Since we have found in chapter 6 that the Interacting Multiple Model (IMM) algorithm has a better estimation accuracy than the first order generalized pseudo Bayesian (GPB1) algorithm, in this chapter the switching Kalman filters approximated by the IMM algorithm are used to estimate the clean speech components from the noisy observations.

In the IMM algorithm, there are M filters, each of which is supplied with a different input.

Let us define

$$\hat{\mathbf{z}}_{t-1|t-1} = E(\mathbf{z}_{t-1} | \mathbf{o}_{1:t-1}) \quad (8.40)$$

$$\hat{\mathbf{V}}_{t-1|t-1} = \text{cov}(\mathbf{z}_{t-1} | \mathbf{o}_{1:t-1}) \quad (8.41)$$

$$\tilde{\mathbf{z}}_{t-1|t-1}^j = E(\mathbf{z}_{t-1} | s_t = j, \mathbf{o}_{1:t-1}) \quad (8.42)$$

$$\tilde{\mathbf{V}}_{t-1|t-1}^j = \text{cov}(\mathbf{z}_{t-1} | s_t = j, \mathbf{o}_{1:t-1}) \quad (8.43)$$

$$W_{t-1|t-1}^{ij} = \Pr(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t-1}) \quad (8.44)$$

$$L_t^j = p(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) \quad (8.45)$$

in which $\mathbf{o}_{1:t} = \{\mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_t\}$ denotes the noisy speech observation sequence up to time t .

The algorithm mixes the estimates according to the Markov transition probability, in order to allow the system to react to changes in the model in force. In that way, the input to the j^{th} filter becomes the best estimate of the state at time instant $t-1$, conditioned on the event that model j is in force at time instant t (the new sample time). $\tilde{\mathbf{z}}_{t-1|t-1}^j$ and $\tilde{\mathbf{v}}_{t-1|t-1}^j$ are then obtained according to

$$\tilde{\mathbf{z}}_{t-1|t-1}^j = \sum_i W_{t-1|t-1}^{ij} \hat{\mathbf{z}}_{t-1|t-1}^i \quad (8.46)$$

$$\begin{aligned} \tilde{\mathbf{v}}_{t-1|t-1}^j &= \sum_i W_{t-1|t-1}^{ij} \hat{\mathbf{v}}_{t-1|t-1}^i \\ &\quad + \sum_i W_{t-1|t-1}^{ij} (\hat{\mathbf{z}}_{t-1|t-1}^i - \tilde{\mathbf{z}}_{t-1|t-1}^j)(\hat{\mathbf{z}}_{t-1|t-1}^i - \tilde{\mathbf{z}}_{t-1|t-1}^j)^T \end{aligned} \quad (8.47).$$

The mixing probability $W_{t-1|t-1}^{ij} = p(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t-1})$ is computed recursively with Bayes' rule

$$\begin{aligned} W_{t-1|t-1}^{ij} &= p(s_{t-1} = i | s_t = j, \mathbf{o}_{1:t-1}) \\ &= \frac{p(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) p(s_{t-1} = i | \mathbf{o}_{1:t-1})}{\sum_i p(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) p(s_{t-1} = i | \mathbf{o}_{1:t-1})} \\ &\approx \frac{p(s_t = j | s_{t-1} = i, \hat{\mathbf{x}}_{1:t-1}) p(s_{t-1} = i | \mathbf{o}_{1:t-1})}{\sum_i p(s_t = j | s_{t-1} = i, \hat{\mathbf{x}}_{1:t-1}) p(s_{t-1} = i | \mathbf{o}_{1:t-1})} \end{aligned} \quad (8.48)$$

$p(s_t = j | s_{t-1} = i, \hat{\mathbf{x}}_{1:t-1})$ is the output of the nonlinear transition network. It is seen that the input-related transition probability is used in addition to the previous state (mode) information, in order to decide which state (mode) is present. Since $\hat{\mathbf{x}}_{1:t-1}$ is only an estimate of $\mathbf{x}_{1:t-1}$, it tends to be biased. Equation (8.55) is thus only an approximation. However in practice this has not been observed to affect the performance of the switching Kalman filter algorithm adversely in our experiments.

Given a noisy speech feature vector $\mathbf{o}_t$ ($t=1, \dots, T$), $\hat{\mathbf{z}}_{t|t}$ is found from the interacting multiple model algorithm using a bank of Kalman filters with the following steps performed in sequence:

Obtain $\hat{\mathbf{z}}_{t|t}^j$ from the j^{th} ($j=1, \dots, M$) filter by

$$(\hat{\mathbf{z}}_{t|t}^j, \mathbf{V}_{t|t}^j, L_t^j) = KF(\tilde{\mathbf{z}}_{t-1|t-1}^j, \tilde{\mathbf{V}}_{t-1|t-1}^j, \mathbf{o}_t; \Phi_j, \mathbf{u}_j, \mathbf{Q}_j) \quad (8.49)$$

where $KF()$ represents the filtering operations of the extended Kalman filter $EKF()$ function or the unscented Kalman filter $UKF()$ function. For a detailed description of the EKF and the UKF algorithm, please refer to section 6.3.4 and 6.3.5 respectively.

The model probabilities $p(s_t = j | \mathbf{o}_{1:t})$ are updated according to

$$\begin{aligned} p(s_t = j | \mathbf{o}_{1:t}) &= \frac{1}{\Omega} \sum_i p(s_{t-1} = i, s_t = j, \mathbf{o}_t | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{\Omega} \sum_i p(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) p(s_{t-1} = i, s_t = j | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{\Omega} \sum_i p(\mathbf{o}_t | s_{t-1} = i, s_t = j, \mathbf{o}_{1:t-1}) p(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) p(s_{t-1} = i | \mathbf{o}_{1:t-1}) \\ &= \frac{1}{\Omega} L_t^j \sum_i p(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) p(s_{t-1} = i | \mathbf{o}_{1:t-1}) \end{aligned} \quad (8.50)$$

where Ω is a scale factor.

$$\Omega = \sum_j L_t^j \sum_i p(s_t = j | s_{t-1} = i, \mathbf{o}_{1:t-1}) p(s_{t-1} = i | \mathbf{o}_{1:t-1}) \quad (8.51)$$

The estimated output at time t is calculated according to

$$\hat{\mathbf{z}}_{t|t} = \sum_j p(s_t = j | \mathbf{o}_{1:t}) \hat{\mathbf{z}}_{t|t}^j \quad (8.52)$$

For each frame of incoming noisy speech, the algorithm described above gives the estimate of the speech in the frame.

8.4 Simulation Results and Analysis

8.4.1 Simulation Results on the *Number* v1.3 Database

Pilot experiments were first conducted on the small database *Number* v1.3. The database description and experimental settings for the speech recognition engine are given in chapter 4. For the experiments conducted on the database *Number* v1.3, we assume that an AR model of background noise is known. Clean speech features from 60 clean utterances were used to obtain the SLDM parameters. The total number of LDMs used in

the SLDM was 8. The SLDM has full covariance matrices. A feed-forward Multi-Layer Perceptron (MLP) neural network was used to model the transition probabilities between the LDMs. The MLP network has a topology as shown in figure 8.3. The input dimensions for the network are 40 ($\bar{X}_l = [\mathbf{x}_{l-1}, \mathbf{x}_{l-2}]$). There are 8 hidden neurons in the hidden layer and the output dimensions are 64. For each frame, the noisy 20th order Mel-scaled log spectrum was transformed to clean feature estimation. Finally, the estimated clean speech feature vectors were mapped from the log-spectral domain into the cepstral domain using a DCT transform, before being passed on directly to a speech recognizer for the recognition task.

Tables 8.1 and 8.2 show the recognition accuracy for white noise and babble noise mixtures if the speech features are compensated in the log-spectral domain. The results of the proposed time-variant linear predictive HMM approach are represented by ‘SLDM-chapter 8- $p=1, l=2$ -IMM’, meaning that the SLDM have first-order dependency and the input of the transition network is $\bar{X}_l = [\mathbf{x}_{l-1}, \mathbf{x}_{l-2}]$. For the purpose of comparison, the results obtained by the SLDM methods proposed in chapter 6 and chapter 7 are also listed here, denoted by ‘SLDM-chapter 6- $p=1$ -IMM’ and ‘SLDM-chapter 7- $p=1, l=2$ -GPB1’. The results by the SLDM method proposed in [Kim05] are represented by ‘SLDM [Kim05]’ in the tables. Table 8.3 shows the average CPU time for the enhancement of one utterance (5-continuous-digits) on a Pentium4 2.4 GHz personal computer under a Matlab environment.

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	50.6	73.4	88.0	93.0	76.3
SLDM [Kim05]	59.0	79.0	87.6	92.8	79.6
SLDM chapter 6 $p=1$ -IMM	68.0	83.6	92.8	95.0	84.9
SLDM-chapter 7 $p=1, l=2$ -GPB1	68.4	87.6	93.6	95.2	86.2
SLDM chapter 8 $p=1, l=2$ -IMM	66.6	82.8	93.0	94.0	84.1

Table 8. 1: Recognition accuracy (%) for speech mixed with white noise
when speech is enhanced in the log-spectrum domain

SNR	0 dB	5 dB	10 dB	15 dB	Average
No noise reduction	44.0	70.2	85.0	91.0	72.6
SLDM [Kim05]	49.6	73.2	84.6	91.0	74.6
SLDM chapter 6 $p=1$ -IMM	59.4	74.8	90.4	93.4	79.5
SLDM-chapter 7 $p=1, l=2$ -GPB1	62.2	78.6	90.0	93.6	81.1
SLDM chapter 8 $p=1, l=2$ -IMM	62.8	79.4	88.2	93.0	80.9

Table 8.2: Recognition accuracy (%) for speech mixed with babble noise
when speech is enhanced in the log-spectrum domain

SLDM-chapter 6 $p=1$ -IMM	3.92
SLDM-chapter 7 $p=1, l=2$ -GPB1	3.20
SLDM-chapter 8 $p=1, l=2$ -IMM	4.31

Table 8.3 CPU time (sec) for the enhancement of one utterance
when speech is enhanced by the switching EKF algorithm

The simulation results on the small *Number* v1.3 database show that, for white noise, the introduction of time-variant transition probabilities among linear dynamic models didn't improve the performance of SLDM systems. The time-variant linear predictive HMM proposed gives a similar performance to its SLDM counterpart, the linear-predictive-HMM with time-invariant transition property, proposed in chapter 6. However, for babble noise, the introduction of time-variant transition probabilities among linear dynamic models makes the newly proposed SLDM approach give better performance. It demonstrates that a connectionist network could be used to model the transition property between the LDMs, which gives a greater potential descriptive power for the SLDM. We also notice, although we got some performance advantage from the time-variant linear predictive HMM approach, the approach proposed in chapter 7 still gave the best performance among these approaches.

8.4.2 Simulation Results on the *Aurora 2* Database

The proposed speech enhancement scheme was also tested on the *Aurora 2* database. The database description and experimental settings for the speech recognition engine are given in chapter 4. Feature vectors were enhanced in the log-spectral domain. For each noise type a specific noise model was built. The parameters for the noise model were trained on the first ten frames of the noisy utterances, which are assumed to be speech free. To conform to the observation model presented in (8.37), the feature generation was

modified slightly from the *Aurora 2* reference implementation. In particular, we changed from using the spectral magnitude to using the power spectral density as the input to the Mel-frequency filter-bank. All the experimental settings were maintained the same as those in [Dro04], [Kim05].

A global SLDM model with 16 hidden linear dynamic models (LDMs) was trained to describe the time movement of the speech features. One problem with this model is how to train a series of input-related transition probability matrices. When we used the *Number v1.3* as the testing database, there were only 8 LDMs in the SLDM. The input-related transition probability matrices had a size of 8×8 . Therefore a transition network with a structure shown in figure 8.3 can still be updated rapidly in the EM framework. However, for the *Aurora 2.0* database testing, the number of LDMs in the SLDM is 16, and the transition probability matrices have a size of 16×16 . A huge neural network with several thousand weights is needed to model the complex input and output relationship. Training such a large connectionist model would consume a lot of computational resources and make the updating process very slow. As previously mentioned, we build up instead a transition network with the structure presented in figure 8.4. The transition network is composed of 16 radial basis function (RBF) networks of the Moody-Darken type [Moo89], each with 8 hidden neurons in the hidden layer. The number of hidden neurons are obtained by trial and error. The input dimensions are 46 ($\vec{X}_t = [\mathbf{x}_{t-1}, \mathbf{x}_{t-2}]$) and the output dimensions for each RBF network are 16. It should be noted that, although time-variant linear predictive HMMs take more training time during the EM learning because of the introduction of neural networks, this training is off-line and the speech model obtained will be kept fixed during the whole database testing.

Detailed recognition rates (%) for each of the four noise conditions and each of the noisy speech SNRs in the *Aurora 2* set-A are provided in Table 8.4 for the proposed SLDM approach. The SLDM has a first-order dependency and the switching extended Kalman filtering was approximated by the IMM algorithm. If the EKF in this approach is replaced by the UKF algorithm, the results obtained are shown in Table 8.5.

SNR	Subway	Babble	Car	Exhibition	Average
20dB	98.46	98.58	98.93	97.81	98.45
15dB	96.50	97.37	98.36	96.91	97.29
10dB	91.68	94.11	94.93	91.30	93.01
5dB	82.93	83.95	85.77	78.90	82.89
0dB	63.43	52.72	62.45	54.03	58.16
Average	86.60	85.34	88.09	83.79	85.96

Table 8.4 Accuracy (%) with speech enhanced by switching extended Kalman Filter based on the proposed SLDM

SNR	Subway	Babble	Car	Exhibition	Average
20dB	98.46	98.67	98.93	97.81	98.45
15dB	96.77	97.64	98.30	96.73	97.36
10dB	94.18	94.74	96.59	93.78	94.82
5dB	88.84	86.28	90.49	88.43	88.51
0dB	71.09	60.07	75.54	68.70	68.85
Average	89.87	87.48	91.97	89.09	89.60

Table 8.5 Accuracy (%) with speech enhanced by switching unscented Kalman Filter based on the proposed SLDM

The simulation results on the *Aurora* 2.0 database are summarized in Table 8.6, with results averaged across 0 dB to 20 dB. For the purpose of comparison, the results reported in [Kim05] and [Dro04] are also listed in Table 8.6. The results obtained by our proposed time-variant linear-predictive-HMM are presented as ‘SLDM chapter 8- $p=1$, $l=2$ -IMM-EKF’ and ‘SLDM chapter 8- $p=1$, $l=2$ -IMM -UKF’, while the results by its counterpart, the linear predictive HMM with fixed transition property, are represented as ‘SLDM chapter 6- $p=1$ -IMM -EKF’ and ‘SLDM chapter 6- $p=1$ -IMM -UKF’ in the tables. The results obtained by our proposed SLDM approach with ME structure are also included, represented as ‘SLDM chapter 7- $p=1$, $l=2$ -GPB1-EKF’ and ‘SLDM chapter 7- $p=1$, $l=2$ -GPB1-UKF’. The baseline results without noise processing are also given. The number

of LDMs used in the SLDM to obtain the results presented in Table 8.6 are given in Table 8.7. Tables 8.8-8.9 show the average CPU time for the enhancement of one utterance (5-continuous-digits) on a Pentium4 2.4 GHz personal computer under a Matlab environment.

	Subway	Babble	Car	Exhibition	Average
Baseline	70.22	51.35	60.28	65.39	61.81
SLDM [Dro04]	80.55	67.52	86.19	77.82	78.02
SLDM [Kim05]	80.48	80.98	86.82	84.53	83.20
SLDM chapter 6 $p=1$ -IMM-EKF	84.90	83.67	86.41	83.37	84.59
SLDM chapter 7 $p=1, l=2$ -GPB1-EKF	84.91	83.52	85.27	82.52	84.06
SLDM chapter 8 $p=1, l=2$ -IMM -EKF	86.60	85.34	88.09	83.79	85.96
SLDM chapter 6 $p=1$ -IMM -UKF	89.13	86.28	91.57	88.47	88.86
SLDM chapter 7 $p=1, l=2$ -GPB1-UKF	87.23	86.71	88.99	85.70	87.16
SLDM chapter 8 $p=1, l=2$ -IMM -UKF	89.87	87.48	91.97	89.09	89.60

Table 8.6 Summary of accuracy (%) on Aurora 2, set A.

Number of LDMs in the SLDM	
SLDM [Kim05]	128
SLDM [Dro04]	16
SLDM chapter 6	16
SLDM chapter 7	16
SLDM chapter 8	16

Table 8.7 Number of LDMs in the SLDM

SLDM-chapter 6 $p=1$ -IMM-EKF	19.17
SLDM-chapter 7 $p=1, l=2$ -GPB1-EKF	11.33
SLDM-chapter 8 $p=1, l=2$ -IMM-EKF	20.70

Table 8.8 CPU time (sec) for the enhancement of one utterance
when speech is enhanced by the switching EKF algorithm

SLDM-chapter 6 $p=1$ -IMM-UKF	789
SLDM-chapter 7 $p=1, l=2$ -GPB1-UKF	775
SLDM-chapter 8 $p=1, l=2$ -IMM-UKF	811

Table 8.9 CPU time (sec) for the enhancement of one utterance
when speech is enhanced by the switching UKF algorithm

The experimental results from the *Aurora 2* database show that all of our SLDM approaches outperform the SLDM approach from [Dro04],[Kim05]. It can also be observed that the Unscented Kalman filter performs better than the Extended Kalman filter, at the cost of more memory and processing time.

Among the three SLDM approaches that we proposed, the approach in chapter 8 gave the best performance, while the approach proposed in chapter 6 performs as well as the one proposed in chapter 7. This conclusion a priori disagrees with the one we just obtained based on the results of the *Number v 1.3* database. The conclusive disagreement can be explained as follows: in chapter 7, the previous filtered estimation was used to predict the present mode of the SLDM. The filtered estimation tends to be biased especially under

the experimental conditions of the *Aurora 2* database, where the noise model only gives a very coarse estimation of the true noise. Therefore, if the side effect of inaccurate noise modeling cannot be ignored, the approach in chapter 7 no longer performs better than the one in chapter 6 and chapter 8. The SLDM proposed in chapter 8 can be interpreted as a combination of the SLDM in chapter 6 and the SLDM in chapter 7. Theoretically, it tends to have a potentially greater descriptive power than each individual one, at the expense of a greater computational complexity. On the other hand, the complex SLDM structure makes the modeling process fairly complicated and can introduce more training errors into the speech model, in comparison to each individual one. This drawback can be revealed from the *Number v1.3* database experimental results. Therefore, for the SLDM approach proposed in chapter 8, there is a compromise between the achievable performance advantage and the modeling error involved. For example, suppose that the proposed SLDM model could be trained with some improved, advanced neural network algorithm and that the EM algorithm could converge well enough, then we would get a better speech model than our present one, and this would further improve the performance of the new approach.

8.6 Summary

In this chapter, a new SLDM approach was proposed to compensate for the environmental noise in the speech feature domain. Compared with the SLDM approaches proposed in chapter 6 and chapter 7, the newly proposed SLDM approach can further improve the speech enhancement accuracy, since the model transition probabilities between the LDMs can be described more precisely at each time point. We have carried out experiments to evaluate the effects of incorporating an input-dependent transition network into the SLDM for the speech feature enhancement application. Simulation results have confirmed the improved performance of the proposed approach over the comparable SLDM methods previously published, and the improvement of performance over the SLDM approaches proposed in chapter 6 and chapter 7 under realistic noise training conditions (*Aurora 2* database).

Chapter 9

Conclusions and Future Work

Our research has focused on the enhancement of speech in the cepstral and log-spectral domain for the purpose of robust speech recognition. One of the difficulties with this kind of approach is that the distortion caused by additive ambient noises is highly non-linear in the feature domain. The Extended Kalman Filter algorithm (EKF) and the Unscented Kalman Filter algorithm (UKF) can be used to deal with the nonlinear distortion introduced by the noise in the feature domain.

Another difficulty is that the trajectory of speech features exhibits a complex and rich dynamic behavior and may not be explicitly described by a global dynamical model. How to model the non-stationary time evolution trajectory of speech features is a problem that we attempted to solve in chapter 6, chapter 7 and chapter 8.

In chapter 6, a switching model based on the concept of hidden Markov models (HMM) was studied. In chapter 7, a switching model with a mixture of experts (ME) structure was proposed. We showed that the different variants of these two models could be interpreted in a unified way in chapter 8. The method in chapter 6 made use of a bank of M different linear dynamic models, each of which tracks the speech dynamics in a different regime. The behavior of the system is characterized by switching between the linear dynamic models. If each linear prediction model is associated with a hidden state of a HMM, and the switching between the linear dynamic models is based on the internal information extracted from the estimation of the present HMM state and the state transition probability, it gives the linear-predictive-HMM described in chapter 6. If the switching between the linear dynamic models is based on the external information presented in the form of inputs to a gating network, it gives the mixture of linear dynamic models presented in chapter 7. The learning and inference algorithms for modeling the time evolution of speech features with the proposed switching model were presented. Based on

the systems obtained, switching Kalman filters were then adapted to compensate for the noisy speech.

A problem with the linear-predictive-HMM model presented in chapter 6 is that it is unable to describe the transitions at each time point precisely since the state transition property is defined over the whole process. To further improve the performance of the linear-predictive-HMM model for speech enhancement, a new linear-predictive-HMM model with a time-variant HMM state transition probability was proposed for the purpose of speech enhancement under noisy environments in chapter 8. Rather than holding a transition probability for the whole process, a neural network model was employed to learn the time variant transition probability from the dynamic situation of the time series. With an input-dependent transition matrix, this new model is more general than the related linear-predictive-HMM model in chapter 6, which doesn't make use of the input information in the switching process. The new model is also more general than the mixture of linear dynamic models approach in chapter 7, which doesn't make use of the previous state information. Thus the time-variant linear-predictive-HMM model can be interpreted as a combination of the linear-predictive-HMM approach of chapter 6 and the mixture of linear dynamic models approach of chapter 7. A speech enhancement scheme based on the new model and switching Kalman filters was developed in chapter 8.

9.1 Summary of Results and Conclusions

In chapters 6,7 and 8, the performances of the proposed schemes for speech enhancement were presented and compared with other model-based feature enhancement systems in the literature. In this section, the performances of the proposed schemes will be summarized and compared with the state-of-the-art performance in the literature.

(1) Experimental results on the *Number* v1.3 database

Tables 9.1-9.2 show the best results that have been obtained with the SLDM proposed in chapter 6, chapter 7 and chapter 8, when speech is enhanced in the log-spectral domain and the LDMs in the SLDM have first-order dependency. The three entries have the same number of LDMs. Table 9.3 shows the average CPU time for the enhancement of one utterance (5-continuous-digits) on a Pentium4 2.4 GHz personal computer under a Matlab environment.

	0 dB	5 dB	10 dB	15 dB	Average
SLDM-chapter 6 $p=1$ -IMM	68.0	83.6	92.8	95.0	84.9
SLDM-chapter 7 $p=1, l=2$ -GPB1	68.4	87.6	93.6	95.2	86.2
SLDM-chapter 8 $p=1, l=2$ -IMM	66.6	82.8	93.0	94.0	84.1

Table 9.1 Recognition accuracy (%) for speech mixed with white noise
when speech is enhanced by switching EKF algorithm

SNR	0 dB	5 dB	10 dB	15 dB	Average
SLDM-chapter 6 $p=1$ -IMM	59.4	74.8	90.4	93.4	79.5
SLDM-chapter 7 $p=1, l=2$ -GPB1	62.2	78.6	90.0	93.6	81.1
SLDM-chapter 8 $p=1, l=2$ -IMM	62.8	79.4	88.2	93.0	80.9

Table 9.2 Recognition accuracy (%) for speech mixed with babble noise
when speech is enhanced by switching EKF algorithm

SLDM-chapter 6 $p=1$ -IMM	3.92
SLDM-chapter 7 $p=1, l=2$ -GPB1	3.20
SLDM-chapter 8 $p=1, l=2$ -IMM	4.31

Table 9.3 CPU time (sec) for the enhancement of one utterance
when speech is enhanced by the switching EKF algorithm

(2) Experimental results on the *Aurora* database

Tables 9.4-9.5 show the best results that have been obtained with the SLDM proposed in chapter 6, chapter 7 and chapter 8 when speech is enhanced in the log-spectral domain, and the LDMs in the SLDM have first-order dependency. These three entries have the same number of LDMs. The results are averaged across 0 dB to 20 dB. Tables 9.6-9.7 show the average CPU time for the enhancement of one utterance (5-continuous-digits) on a Pentium4 2.4 GHz personal computer under a Matlab environment.

To the best of our knowledge, the best experimental results based on the *Aurora 2* database Set A in the literature are given by L. Deng et al in [Den04b]. The method proposed in [Den04b] is based on a noise-tracking EM algorithm and was reviewed in section 3.2.1.4 of chapter 3. Their results are represented by ‘State-of-the-Art’ in Tables 9.4-9.5. Different from the other noise-tracking-only approaches, L. Deng’s approach takes into account the dynamic properties of speech by using the frame-differential (“delta”) features. However, the inherent time delay with this approach may be high as we explained in chapter 3.

	Subway	Babble	Car	Exhibition	Average
State-of-the-Art	85.92	83.77	90.47	86.71	86.72
SLDM-chapter 6 $p=1$ -IMM-EKF	84.90	83.67	86.41	83.37	84.59
SLDM-chapter 7 $p=1, l=2$ -GPB1-EKF	84.91	83.52	85.27	82.52	84.06
SLDM-chapter 8 $p=1, l=2$ -IMM-EKF	86.60	85.34	88.09	83.79	85.96

Table 9.4 Summary of accuracy (%) on Aurora 2, set A, by the switching EKF algorithm

	Subway	Babble	Car	Exhibition	Average
State-of-the-Art	85.92	83.77	90.47	86.71	86.72
SLDM-chapter 6 $p=1$ -IMM-UKF	89.13	86.28	91.57	88.47	88.86
SLDM-chapter 7 $p=1, l=2$ -GPB1-UKF	87.23	86.71	88.99	85.70	87.16
SLDM-chapter 8 $p=1, l=2$ -IMM-UKF	89.87	87.48	91.97	89.09	89.60

Table 9.5 Summary of accuracy (%) on Aurora 2, set A, by the switching UKF algorithm

SLDM-chapter 6 $p=1$ -IMM-EKF	19.17
SLDM-chapter 7 $p=1, l=2$ -GPB1-EKF	11.33
SLDM-chapter 8 $p=1, l=2$ -IMM-EKF	20.70

Table 9.6 CPU time (sec) for the enhancement of one utterance
when speech is enhanced by the switching EKF algorithm

SLDM-chapter 6 $p=1$ -IMM-UKF	789
SLDM-chapter 7 $p=1, l=2$ -GPB1-UKF	775
SLDM-chapter 8 $p=1, l=2$ -IMM-UKF	811

Table 9.7 CPU time (sec) for the enhancement of one utterance
when speech is enhanced by the switching UKF algorithm

(3) Analysis

The experimental results of the large *Aurora 2* database showed that the proposed approach in chapter 8 gave the best performance. Compared with the best results found in the literature, our ‘SLDM-chapter 8- $p=1, l=2$ -IMM-EKF’ approach offers a similar performance (86.72% vs. 85.96%) but at a lower cost, while our ‘SLDM-chapter 8- $p=1, l=2$ -IMM-UKF’ approach achieves a better performance (89.60% vs. 86.72%), arguably with similar processing time. The approach proposed in chapter 6 performs as well as the one proposed in chapter 7. However, from the pilot experimental results of the small *Number v1.3* database, we found that the approach proposed in chapter 7 performed better than the approaches proposed in chapter 6 and chapter 8. This disagreement has

been analyzed in detail in chapter 8, and the differences in performance come from the different conditions on the noise modeling between the *Number* v1.3 database (good noise model available, not too realistic) and the *Aurora 2* database (only approximate noise model available, realistic). To decide which SLDM approach should be used for a given application, we thus need to take the experimental conditions into consideration, and we also need to consider a desirable balance between the performance gain and the degree of algorithm complexity. When we can only have a very coarse estimation of the noise model, the approach of chapter 8 is more preferable. On the other hand, if we have a good noise model at hand, approach in chapter 7 is more desirable. Although the approach in chapter 8 always performs better than the one in chapter 6, the complex SLDM structure in chapter 8 makes the modeling process fairly complicated and may introduce more training errors into the speech model if it gets trapped in a local maximum. If the background noise has a more stationary property, compared with the approach in chapter 8, the approach in chapter 6 is then a better choice because of its simplicity.

It also has been observed in our experiments that the unscented Kalman filter performed systematically better than the extended Kalman filter, at the cost of more memory and processing time.

9.2 Recommendations for Future Work

The direct area of future work emerging from this thesis is the applications of the SLDM for speech enhancement in the time domain.

Although this thesis dealt with the problem of speech enhancements for the purpose of improving the accuracy of speech or speaker recognition systems, the proposed SLDM approaches could be applied in the time domain to improve the perceptual quality of noisy speech. The reasons why the SLDM would be suitable for non-stationary speech signal analysis in the time domain can be briefly explained as follows. Among most of the speech waveform enhancement methods reviewed in chapter 3, the speech time signal is split up into a number of overlapping blocks. Assuming that signals in each block are

stationary, an autoregressive (AR) model is then applied. A problem with this approach, however, is that the framing often results in a relatively poor temporal resolution for quickly varying speech sounds such as plosives [She94]. Alternatively, it has been found by [Lee96][Kim00] that the SLDM are better able to model quickly varying parts of speech signals, since information is not lost at the frame boundaries. Applications of the SLDM for speech enhancement in the time domain would thus be a worthwhile investigation.

Based on the above, we have preliminarily studied the hidden filter model suggested in [Lee96][Kim00], which has an inference process similar to the linear-predictive-HMM in chapter 6, except that it was working on the univariate time signal. The Kalman filtering problem for such models includes a weighted sum of filters operating interactively in parallel, which is the same as the switching Kalman filters problem that we have elaborated in this thesis. We found from the experiments that this approach provides a better performance than the conventional Kalman filter approaches in terms of ITU-T P.862 PESQ scores [ITU01], which have a close match with subjective speech quality scores. Since the SLDM approaches proposed in chapter 7 and chapter 8 are closely related with the one in chapter 6, it is promising that they could also be applied to enhance speech waveforms for the purpose of improving the perceptual quality of noisy speech.

Bibliography

- [Ace90] A. Acero, *Acoustical and environmental robustness in automatic speech recognition*, Kluwer Academic Publishers, Norwell, 1993.
- [Afi04] M. Afify and O. Siohan, "Sequential estimation with optimal forgetting for robust speech recognition", *IEEE Trans. Speech and Audio Processing*, Vol. 12, pp.19-26, 2004.
- [Bau70] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains", *Annals of Mathematical Statistics*, Vol. 41, pp.164-171, 1970.
- [Ben95] Y. Bengio, and P. Frasconi, An input output HMM architecture. *Advances in Neural Information Processing Systems 7 (NIPS*94)*, pp. 427-434, MIT Press, Cambridge, MA, 1995.
- [Bez81] J. Bezdek, *Pattern recognition with fuzzy objective function algorithms*, Plenum press, New York, 1981
- [Bol79] S. F. Boll, "Suppression of Acoustic Noise in Speech Using Spectral Subtraction," *IEEE Trans. Acoustics, Speech, Signal Processing*, Vol. ASSP-27, No. 2, pp. 113-120, April 1979.
- [Cac94] T. W. Cacciatore, and S. J. Nowlan, Mixtures of controllers for jump linear and nonlinear plants. *Advances in Neural Information Processing Systems 6 (NIPS*93)*, pp. 719-726, Morgan Kaufmann, MIT Press, 1994.
- [Chu99] C. K. Chui, G. Chen, *Kalman Filtering with Real-Time Applications*, 3rd Edition, Springer-Verlag, 1999
- [Cou00] C. Couvreur, and H. Van Hamme, "Model-based feature enhancement for noisy speech recognition", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'00*, Vol. 3, pp. 1719-1722, June, 2000.
- [Dav80] S. B. Davis and P. Mermelstein. "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences", *IEEE Transactions on Acoustics, Speech and Signal Processing*, Vol. 28, pp. 357-366, 1980.
- [Del93] J. R. Deller, J. G. Proakis, J. H. Hansen, *Discrete-time processing of speech signals*, Macmillan Publishing Company, 1st ed., 1993.

- [Dem77] A. P. Dempster, N. M. Laird, and D.B. Rubin, "Maximum likelihood for incomplete data via the EM algorithm", *Journal of the Royal Statistical Society B*, pp.1-38, 1977
- [Den01] L. Deng, A. Acero, J. Jiang, J. Droppo, and X. D. Huang, "High-performance robust speech recognition using stereo training data", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'01*, Vol 1, pp.301-304, 2001.
- [Den03] L. Deng, J. Droppo and A. Acero, "Recursive estimation of nonstationary noise using iterative stochastic approximation for robust speech recognition", *IEEE Trans. Speech and Audio Processing*, Vol. 11, pp. 568-579, May, 2003.
- [Den04a] 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 Trans. Speech and Audio Processing*, Vol. 12, pp.133-143, March, 2004.
- [Den04b] L. Deng, J. Droppo and A. Acero, "Estimating cepstrum of speech under the presence of noise using a joint prior of static and dynamic features", *IEEE Trans. Speech and Audio Processing*, Vol. 12, pp. 218-233, May 2004.
- [Dig92] V. Digalakis. *Segment-Based Stochastic Models of Spectral Dynamics for Continuous Speech Recognition*, PhD thesis, Boston University, 1992.
- [Dig93] V. Digalakis, J.R. Rohlicek, and M. Ostendorf, "ML estimation of a stochastic linear system with the EM algorithm and its application to speech recognition", *IEEE Transactions on Speech and Audio Processing*, Vol. 1, No. 4, pp. 431-442, 1993.
- [Dro04] J. Droppo and A. Acero, "Noise robust speech recognition with a switching linear dynamic model", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'04*, Vol. 1, pp. I-953- I-956, May 2004
- [Eph84] Y. Ephraim and D. Malah, "Speech Enhancement Using a Minimum Mean-Square Error Short-Time Spectral Amplitude Estimator," *IEEE Trans. Acoustics, Speech and Signal Processing*, Vol. 32, No. 6, pp. 1109-1121, December, 1984.
- [Eph89] Y. Ephraim, D. Malah and B. Juang, "On the Application of Hidden Markov Models for enhancing Noisy Speech," *IEEE Trans. Acoustics, Speech and Signal Processing*, Vol. 37, No. 12, pp.1846-1856, December, 1989.
- [Eph90] Y. Ephraim, "A Minimum Mean square error approach for speech enhancement" *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98*, pp. 829-832, 1990.
- [Eph92] Y. Ephraim, "Statistical-Model-Based Speech Enhancement Systems," *Proceedings*

- of the *IEEE*, Vol. 80, No. 10, pp. 1526-1555, Dec. 1992.
- [Fre01] B. Frey, L. Deng, A. Acero, and T. Kristjansson, "ALGONQUIN: Iterating Laplace's method to remove multiple types of acoustic distortion for robust speech recognition", *Proc. Eur. Conf. Speech Communication*, Aalborg, Denmark, pp. 901-904, Sept. 2001.
 - [Fuj05] M. Fujimoto and S. Nakamura, "Particle filter based non-stationary tracking for robust speech recognition", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'05*, pp. 257-260, April, 2005
 - [Gab01] M. Gabrea, "Adaptive Kalman Filtering-Based Speech enhancement Algorithm," *Canadian Conference on Electrical and Computer Engineering 2001*, Fredericton, New-Brunswick, Vol.1, pp.521-526, May, 2001.
 - [Gan98] S. Gannot, D. Burshtein and E. Weinstein, "Iterative and Sequential Kalman Filter-Based Speech Enhancement Algorithms," *IEEE Trans. Speech and Audio Processing*, Vol. 6, No. 4, pp. 373-385, July, 1998.
 - [Gha96] Z. Ghahramani, and G. Hinton, Switching state-space models, *Technical Report CRGTR-96-3*, Department of Computer Science, University of Toronto, 1996
 - [Gib91] J. D. Gibson, B. Koo and S. D. Gray, "Filtering of colored noise for speech enhancement", *IEEE transactions on Signal processing*, Vol. 39, No. 8, pp. 1732-1741, 1991.
 - [Gon95] Y. Gong, "Speech recognition in noisy environment: A survey" *Speech Communication*, Vol. 16, pp.261-291, 1995.
 - [Hay96] S. Haykin, *Adaptive Filter Theory*, Third Edition, Prentice Hall, 1996.
 - [Her90] H. Hermansky. "Perceptual linear prediction (PLP) of speech", *Journal of the Acoustic Society of America*, Vol. 87, pp. 1738-1752, 1990.
 - [Hir00] H. Hirsch and D. Pearce, "The AURORA experimental framework for the performance evaluation of speech recognition systems under noisy conditions," *Proceedings of ISCA ITRW ASR2000 on Automatic Speech recognition: Challenges for the next Millennium*, Paris, France, Sept. 2000.
 - [HTK99] S. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev and P. Woodland, *The HTK book – version 2.2*, <http://htk.eng.cam.ac.uk/docs/docs.shtml> , 1999.
 - [ITU01] "Perceptual evaluation of speech quality (PESQ), and objective method for end-to-end speech quality assessment of narrow band telephone networks and speech codecs", ITU-T Recommendation P.862, Feb. 2001
 - [ITU98] ITU-T Rec. P. Supplement 23, "ITU-T coded-speech database," International

- Telecommunication Union, 1998.
- [Jac91] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts", *Neural Computation*, Vol. 3, pp.79-87, 1991.
 - [Joh88] J. D. Johnston, "Transform Coding of Audio Signals Using Perceptual Noise Criteria," *IEEE Journal Selected Areas in Communications*, Vol. 6, No. 2, pp.314-323, February, 1988.
 - [Jor94] M. I. Jordan and R. A. Jacobs, "Hierarchical mixtures of experts and the EM algorithm", *Neural Computation*, Vol.6, pp. 181-214, 1994.
 - [Jor95] M. I. Jordan and L. Xu, "Convergence results for the EM approach to mixtures of experts architectures", *Neural Networks*, Vol. 8, pp.1409-1431, 1995.
 - [Jul00] S. J. Julier, J. K. Uhlmann, H.F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators", *IEEE Transactions on Automatic Control*, Vol. 45, pp. 477 - 482, March 2000
 - [Jul97] S. J. Julier, and J. K. Uhlmann, "A new extension of the Kalman Filter to Nonlinear Systems." *Proceedings of AeroSense: The 11th Int. Symp. On Aerospace/Defence Sensing, Simulation and Controls*, 1997 (Available at <http://citeseer.ist.psu.edu/julier97new.html>)
 - [Ken90] P. Kenny, M. Lenning, and P. Mermelstein, "A linear predictive HMM for vector-valued observations with applications to speech recognition", *IEEE Trans. Acoustics, Speech and Signal Processing*, Vol. 38, pp. 220-225, Feb. 1990.
 - [Ken92] R. D. Kent and C. Read, *The Acoustic Analysis of speech*, Singular Publishing Group Inc, 1992
 - [Kim00] J. B. Kim, K. Y. Lee, and C. W. Lee, "On the application of the Interacting Multiple Model Algorithm for Enhancing Noisy Speech," *IEEE Trans. Speech and Audio Processing*, Vol. 8, No 3, pp. 349-352, May 2000.
 - [Kim05] N. S. Kim, W. Lim, and R. Stern, "Feature compensation based on switching linear dynamic model", *IEEE Signal Processing Letters*, Vol. 12, pp. 473-476, June, 2005.
 - [Kim94] C. J. Kim, "Dynamic linear models with Markov switching", *Journal of Econometrics*, Vol. 60, pp.1-22, 1994.
 - [Kim98a] N. S. Kim, "Statistical linear Approximation for environment compensation", *IEEE Signal Processing Letters*, Vol. 5, pp.8-10, Jan. 1998.
 - [Kim98b] N. S. Kim, "IMM-based estimation for slowly evolving environments", *IEEE Signal Processing Letters*, Vol. 5, pp. 146-149, June, 1998.

- [Kim98c] N. S. Kim, "Nonstationary environment compensation based on sequential estimation", *IEEE Signal Processing Letters*, Vol.5, No 3, pp.57-59, March 1998.
- [Kim98d] D. Y. Kim, C. K. Un, N. S. Kim, "Speech recognition in noisy environments using first-order vector Taylor series", *Speech Communication*, Vol. 24, pp. 39-49, 1998.
- [Koh00] J., Kohlmorgen, K.-R., Müller, J., Rittweger, K. Pawelzik, "Identification of Nonstationary Dynamics in Physiological Recordings", *Biological Cybernetics*, Vol. 83, pp.73-84, 2000.
- [Lee96] K. Y. Lee and K. Shirai, "Efficient Recursive Estimation For Speech Enhancement In Colored Noise," *IEEE Signal processing Letters*, Vol. 3, pp.196-199, July 1996.
- [Lie99] S. Liehr, K. Pawelzik, J. Kohlmorgen, and K.-R. Muller, "Hidden Markov Mixtures of Experts with an Application to EEG Recordings from Sleep", *Theory in Biosciences* , Vol. 118, pp. 246-260, 1999.
- [Lim78] J. S. Lim, A. V. Oppenheim, "All-pole modeling of degraded speech", *IEEE Trans. Acoustics, Speech, Signal Processing*, Vol. 26, June, 1978, pp.197-210
- [Lim79] J. S. Lim, A. V. Oppenheim, "Enhancement and Bandwidth Compression of Noisy Speech", *Proceedings of the IEEE*, Vol. 67, No. 12, pp. 1586-1604, Dec. 1979
- [Log97] B. T. Logan, *Adaptive model-based speech enhancement*, PhD thesis, Cambridge University, 1997.
- [Mak75] J. Makhoul, "Linear prediction: A tutorial review", *Proceedings of the IEEE*, Vol. 63, pp. 561-580, 1975.
- [Man04] N. Ma, M. Bouchard and R. Goubran, "Perceptual Kalman filtering for speech enhancement in colored noise," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'04*, Vol. 1, pp.717-720, Montreal, May 2004.
- [Man05] N. Ma, *Speech Enhancement Algorithms Using Kalman Filtering and Masking Properties of Human Auditory Systems*, Ph.D Thesis, University of Ottawa, 2005.
- [Maz03] Z. Ma, L. Deng, "Efficient decoding strategies for conversational speech recognition using a constrained nonlinear state space model", *IEEE Trans. Speech and Audio Processing*, Vol. 11, pp. 590-602, Nov. 2003
- [Moo89] J. Moody and C. J. Darken, "Fast learning in network of locally-tuned processing units", *Neural Computation*, Vol. 3, pp.281-294, 1989.
- [Mor96] P. J. Moreno, *Speech recognition in noisy environments*, Ph.D Thesis, Carnegie Mellon University, 1996
- [Mur98a] K. P. Murphy, "Switching Kalman filters", *Technical Report*, Compaq Cambridge

Research Laboratory, 1998

- [Mur98b] K. P. Murphy, "Fitting a conditional linear Gaussian distribution", *Technical Report*, Compaq Cambridge Research Laboratory, 1998
- [Myr03] T. A. Myrvoll and S. Nakamura, "Optimal filtering of noise cepstral coefficients for robust ASR", *IEEE ASRU workshop '03*, pp. 381-386, Dec. 2003
- [Ost96] M. Ostendorf, V. Digalakis, and O. Kimball, "From HMM's to segment models: A unified view of stochastic modeling for speech recognition", *IEEE Transactions on Speech and Audio Processing*, Vol. 4, No. 5, pp. 360-378, 1996.
- [Pal87] K. Paliwal and A. Basu, "A speech enhancement method based on Kalman filtering," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'87*, Vol. 12, pp. 297-300, April, 1987.
- [Pen98] W. D. Penny and S. J. Roberts, "Dynamic models for non-stationary signal segmentation", *Technical Report TR-98-14*, Computers and Biomedical Research, pp. 483-502, 1999.
- [Por82] A. B. Poritz, "Linear predictive hidden markov models and the speech signal", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'82*, pp 1291-1294, 1982.
- [Pre95] M. M. Predoviciu, and M. I. Jordan, Learning the parameters of HMMs with auxilliary input, *Technical report*, MIT, 1995.
- [Rab89] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Application in Speech recognition", *Proceedings of the IEEE*, Vol. 77, pp. 257-286, Feb. 1989
- [Rab93] L. R. Rabiner and B. H. Juang, *Fundamentals of speech recognition*, Prentice-Hall, 1993
- [Raj04] B. Raj, R. Singh, and R. Stern "On tracking noise with linear dynamical system models", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'04*, Vol. 1, pp. 965- 968, May 2004.
- [Rey92] D. A. Reynolds, *A Gaussian Mixture Modelling Approach to Text-independent Speaker Identification*, PhD thesis, Georgia Institute of Technology, September 1992.
- [Ros01] A. V. I. Rosti and M. J. F. Gales, "Generalised linear Gaussian models", *Technical Report CUED/F-INFENG/TR.420*, Cambridge University Engineering Department, 2001. Available at <http://mi.eng.cam.ac.uk/reports/>.
- [Ros04] A. V. I. Rosti, *Linear Gaussian Models for Speech Recognition*, PhD thesis, Cambridge University Engineering Department, May, 2004

- [Sam98] H. Sameti, H. Sheikhzadeh, L. Deng, "HMM-Based Strategies for Enhancement of Speech Signals Embedded in Nonstationary Noise," *IEEE Trans. Speech and Audio Processing*, Vol. 6, No. 5, pp. 445-454, September, 1998.
- [Sha93] Y. B. Shalom and X. R. Li, *Estimation and Tracking*, Boston Artech house, 1993.
- [She94] H. Sheikhzadeh, L. Deng, "Waveform-based speech recognition using hidden filter models: parameter selection and sensitivity to power normalization", *IEEE trans. Speech and audio processing*, Vol. 2, No. 1, pp. 80-89, 1994
- [Shi97] S. Shi and A. S. Weigend, "Taking time seriously, hidden markov experts applied to financial engineering", *Proceedings of the IEEE/IAFC conference on computational intelligence of financial engineering*, pp. 244-252, 1997.
- [Thi00] T. Thiede, W. C. Treurniet, R. Bitto, et. al., "PEAQ-the ITU standard for objective measurement of perceived audio quality," *Journal of Audio Eng. Soc.*, Vol. 48, January, 2000.
- [Umb05] R. H. Umbach, J. Schmalenstroeer, "A comparison of particle filtering variants for speech feature enhancement", *INTERSPEECH'05*, pp. 913-916, Sept, 2005
- [Vii01] O. Viikki, *Speech communication (Special issue on noise robust Automatic speech recognition)*, O. Viikki, Ed., Vol.34, 2001.
- [Vir99] N. Virag, "Single Channel Speech Enhancement Based on Masking Properties of the Human Auditory System," *IEEE Trans. Speech and Audio Processing*, Vol. 7, No. 2, pp.126-137, March, 1999.
- [Wan00] E. A. Wan, R. V. D. Merwe, "The unscented Kalman filter for nonlinear estimation", *IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium AS-SPCC.*, pp.153 -158, Oct. 2000
- [Wan04] X. Wang, P. Whigham, D. Deng, M. Purvis, "Time-line Hidden Markov Experts for time series prediction", *Neural Information Processing-Letters and Reviews* , Vol. 3, No.2, pp.39-48, 2004
- [Wan98] E. A. Wan, A. T. Nelson, "Removal of Noise from Speech Using the Dual EKF Algorithm", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98*, Vol. 1, pp. 381-384, 1998.
- [Wat95] S. R. Waterhouse, and A. J. Robinson, "Non-linear prediction of acoustic vectors using hierarchical mixture of experts", *Advances in Neural Information Processing Systems 7 (NIPS'94)*, pp. 835-842. MIT Press, Cambridge, MA, 1995.
- [Wei95] A. S. Weigend, M. Mangeas, A. N. Srivastava, "Nonlinear gated experts for time series: discovering regimes and avoiding overfitting", *International Journal of*

- Neural Systems*, Vol. 6, pp. 373-399, 1995.
- [Wei95] A. S. Weigend and A. N. Srivastava, "Predicting conditional probability distributions: A connectionist approach", *International Journal of Neural Systems*, Vol. 6, pp. 109-118, 1995
 - [Wei95a] A. S. Weigend, M. Mangeas, A.N. Srivastava, "Nonlinear gated experts for time series: discovering regimes and avoiding overfitting", *International Journal of Neural Systems*, Vol. 6, pp. 373-399, 1995.
 - [Xie03] N. Xie and H. Leung, "A multiple-model prediction approach for sea clutter modeling", *IEEE Trans. Geoscience and Remote Sensing*, Vol. 41, pp. 1491-1502, June, 2003.
 - [You89] S. J. Young, N. H. Russell, and J. H. S. Thornton, "Token passing: a simple conceptual model for connected speech recognition systems". *Technical Report CUED/FINFENG/TR.38*, Cambridge University Engineering Department, 1989.
 - [Zha95] Y. Zhao, R. Schwartz, J. Sroka, and J. Makhoul, "Hierarchical mixtures of experts methodology applied to continuous speech recognition", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'95*, pp. 3443-3446, 1995.