Internal Use Only（非公開）

TR－SLT－0016

Robust speech recognition: noise adaptive speech recongition, Sequential Monte Carlo and generative factor analyzed HMM

ロバスト音声認識: 雑音適応音声認識、シーケンシャルモンテカルロおよび生成要素分析 HMM

Kaisheng Yao, Kuldip K. Paliwal, and Satoshi Nakamura
カイシェン ヤオ　クルディップ パリワル　中村 哲

2002年8月14日

This report presents works on robust speech recognition carried out by the authors. The reports have two major works. The first is noisy speech recognition in non-stationary environments, which include noise adaptive speech recognition based on sequential Kullback proximal algorithm, and sequential noise compensation based on sequential Monte Carlo method. The second part is on acoustic modeling, with a work denoted as generative factor analyzed HMM.

（株）国際電気通信基礎技術研究所
音声言語コミュニケーション研究所
〒619－0288「けいはんな学研都市」光台二丁目2番地2 TEL：0774－95－1301

Advanced Telecommunication Research Institute International
Spoken Language Translation Research Laboratories
2-2-2 Hikaridai "Keihanna Science City" 619-0288,Japan
Telephone:+81-774-95-1301
Fax :+81-774-95-1308

Chapter 1

Noise adaptive speech recognition

1.1 Abstract

We present a noise adaptive speech recognition approach to noisy speech recognition in non-stationary environments. Environment effects include channel distortion and additive background noises. Given previously estimated environment parameter sequences, the Viterbi process provides approximated joint likelihood of active partial paths and observation sequence at current frame. The joint likelihood after normalization provides approximation to the posterior probabilities of state sequences for an EM-type recursive process based on sequential Kullback proximal algorithm to estimate the current environment parameter. The combined process can easily be applied to perform continuous speech recognition in presence of non-stationary environments. Experiments conducted in simulated and real environments showed that the noise adaptive speech recognition provides significant improvements in word accuracy as compared to the baseline system (without environment compensation) and the normal noise compensation system (which assumes the stationary environments).

1.2 Introduction

Speech recognition has to be carried out often in situations where there exists environment distortions, such as channel distortion, background noise, competing speech and room reverberation, which cause mismatches between pre-trained models and real testing data. These mismatches between training and testing conditions can be viewed in the signal-space, the feature-space, or the model-space [1]. Varieties of methods have been proposed to combat environment effects, and in general, there are three approaches in this research. The first approach is based on front-end signal processing, where the signal input for feature extraction has higher signal-to-noise ratio (SNR) after processing than that without the processing, e.g., speech enhancement [2]. The second approach is robust feature extraction, which tries to extract features, to some extent, invariant to environment effects, e.g., Perceptual Linear Prediction (PLP) [3] and combination of static, dynamic and acceleration features [4]. The third approach is denoted as model-based approach. The model-based approach assumes parametric models representing environment effects on speech features. Processing of the model-based approach can be either in modifying the Hidden Markov

Model (HMM) parameters in the model-space, e.g., Parallel Model Combination (PMC) [5] and stochastic matching [1], or modifying the input features for recognition, e.g., Code-Dependent Cepstral Normalization (CDCN) [6], Vector Taylor Series (VTS) [7], and Frequency-Domain ML feature estimation [8]. This approach has been shown promising to compensate noise effects [9].

Environment robustness can also be achieved by applying adaptation methods. Linear transformation matrices for adaptation can be estimated by maximum likelihood linear regression (MLLR) [10] and maximum a posterior (MAP) [11] approaches given adaptation utterances. As suggested by their names, the difference between them is the estimation criterion.

In the above approaches, most researches are focused on stationary environment conditions. In this situation, environment or adaptation parameters are often estimated before speech recognition from a small set of environment adaptation data for modifying HMM parameter or input features. However, it is known that the environment statistics may vary during recognition. As a result, the environment or adaptation parameters estimated prior to speech recognition are no longer relevant to the subsequent inputs.

Recently, a number of techniques have been proposed to combat time-varying environment effects. They can be categorized into two approaches. In the first approach, time-varying environment sources are modeled by HMMs or Gaussian mixtures that were trained by prior measurement of environments, so that environment compensation is a task of identification of the underlying state sequences of the environment HMMs [5][12][13] by MAP estimation in a batch mode. For example, in [5], ergodic HMM represents different SNR conditions, so that a composed HMM with speech models can have expanded states that possibly represent speech states at different SNR conditions. This approach requires to make a model representing different conditions of environments (SNRs, types of noise, etc), so that statistics at some states or mixtures obtained before speech recognition are close to the real testing environments.

In the second approach, environment parameters are assumed to be time-varying. The environment parameters can be estimated based on maximum likelihood, e.g., sequential EM algorithm [14][15]. In [14], the sequential EM algorithm is applied to estimate time-varying noise parameter in cepstral domain. Frequency domain EM algorithm [8] has been extended to sequential estimation of time-varying noise parameter in linear frequency domain [15]. The environment parameters can also be estimated by Bayesian methods [16][17]. In [16], a Laplace transform is used to approximate the joint distribution of speech, additive noise and channel distortion by vector Taylor series approximation. In [17], sequential Monte Carlo method is used to estimate environment parameters.

In this paper, we investigate a method assuming time-varying environment parameter for noisy speech recognition in non-stationary environments. In particular, a noise adaptive speech recognition approach [18] is proposed based on the following two novel points. Firstly, noise parameter is estimated sequentially, i.e., frame-by-frame, which can possibly handle non-stationary environments. Secondly, time-varying environment parameter estimation makes use of a Viterbi process from the recognition process to approximate the posterior probabilities along state sequences for the time-varying parameter estimation.

This paper is organized as follows. Section 2.3 reviews the model-based noisy speech recognition. In particular, Section 1.3.2 shows that the parametric modeling of the noise effects on speech features can be seen as mapping between two spaces, where one space is considered as the original training data space and the second space is in the testing environments. Based on this

understanding, it is shown that the environment parameter for mapping between the two spaces can be learned from data. Accordingly, the speech models to be transformed can also be trained from noisy speech.

The process must be carried out sequentially in order to track the time-varying environment parameter. In Section 1.4, the time-recursive environment parameter estimation is described. In particular, the sequential Kullback proximal algorithm [19], which is an extension of the sequential EM algorithm, is applied. Compared to the sequential EM algorithm, the sequential Kullback proximal algorithm gives flexibility in controlling its convergence rate. Section 1.4.2 justifies the Viterbi approximation of the posterior probabilities of state sequences given observation sequences. Section 1.5 provides experimental results carried out on TI-Digits and Aurora 3 database to show the efficacy of the method. Conclusions are in Section 1.6.

1.3 Model based noisy speech recognition

1.3.1 MAP Decision rule for automatic speech recognition

The speech recognition problem can be described as follows. Given a set of trained models $\Lambda_X = \{\lambda_{x_m}\}$ where λ_{x_m} is the mth subword HMM unit trained from X, and an observation vector sequence $Y(T) = (y(1), y(2), \cdots, y(T))$, the aim is to recognize the word sequence $W = (W(1), W(2), \cdots, W(L))$ embedded in $Y(T)$. Each speech unit model λ_{x_m} is a Υ-state CDHMM with state transition probability $a_{iq}(0 \leq a_{iq} \leq 1)$ and each state i is modeled by a mixture of Gaussian probability density functions $\{b_{ik}(\cdot)\}$ with parameter $\{w_{ik}, \mu_{ik}, \Sigma_{ik}\}_{k=1,2,\cdots,M}$, where M denotes the number of Gaussian mixture components in each state. $\mu_{ik} \in R^{D\times 1}$ and $\Sigma_{ik} \in R^{D\times D}$ are respectively the mean vector and covariance matrix of each Gaussian mixture component. D is the feature vector size. w_{ik} is the mixture weight.

In speech recognition, the model Λ_X are used to decode $Y(T)$ using the maximum a posterior (MAP) decoder

$$\begin{aligned} \hat{W} &= \arg\max_W P(W|\Lambda_X, Y(T)) \\ &= \arg\max_W P(Y(T)|\Lambda_X, W) P_\Gamma(W) \end{aligned} \tag{1.1}$$

where the first term is the likelihood of observation sequence $Y(T)$ given that the word sequence is W, and the second term is denoted as the language model. However, in many situations, there exists mismatches due to environments, e.g., additive noise and channel distortion, and accordingly, there is a mismatch in the likelihood of $Y(T)$ given Λ_X evaluated by (1.1).

1.3.2 Model-based noisy speech recognition

In the model-based approach to noisy speech recognition, models representing environment effects on speech features are used. In particular, the following function was proposed in [5][6] to represent environment effects on speech features. (A simple derivation is shown in Appendix .1).

$$y_j^l(t) = x_j^l(t) + h_j^l(t) + \log(1 + \exp(n_j^l(t) - x_j^l(t) - h_j^l(t))) \tag{1.2}$$

where $y_j^l(t)$ is the jth component in the vector of observation power $y^l(t)$ in the log-spectral domain at frame t. Superscript l denotes log-spectral domain. Similarly for $x_j^l(t)$, $h_j^l(t)$ and $n_j^l(t)$ to denote the j-th component in vectors of speech power, channel distortion power and additive noise power at frame t. Subscript j ranges from 1 to J, where J denotes number of filter banks.

Note that cepstral vector $y(t)$, $x(t)$, $h(t)$ and $n(t)$ are obtained by discrete Cosine transform (DCT) on $y^l(t)$, $x^l(t)$, $h^l(t)$ and $n^l(t)$, respectively. Training data of $\{x(t) : t = 1, \cdots, T\}$ is used to train the acoustic model Λ_X. If data of $\{h(t) : t = 1, \cdots, T\}$ and $\{n(t) : t = 1, \cdots, T\}$ are available, a model Λ_N can be trained, so that, by explicit use of function (1.2), (1.1) can be carried out as,

$$\hat{W} = \arg\max_W P(Y(T)|\Lambda_X, \Lambda_N, W) P_\Gamma(W) \tag{1.3}$$

In case that $\{h(t) : t = 1, \cdots, T\}$ and $\{n(t) : t = 1, \cdots, T\}$ are stationary or available before recognition, Λ_N can be estimated prior to speech recognition.

1.3.3 Environment parameter estimation for the model-based noisy speech recognition

Function (1.2) represents a parametric mapping between $x_j^l(t)$ and $y_j^l(t)$. Figure 1.1 shows the function when $x_j^l(t) = 1.0$, $h_j^l(t) = 0.0$ and $n_j^l(t)$ ranges from -10.0 to 10.0. Through the figure, it is seen that the function is smooth and convex as a function of $n_j^l(t)$ given $x_j^l(t)$ and $h_j^l(t)$. The function approximates the masking effects of $n_j^l(t)$ on $x_j^l(t)$. Function (1.2) will output either $x_j^l(t) + h_j^l(t)$ or $n_j^l(t)$ depending on whether $x_j^l(t) + h_j^l(t)$ is much larger than $n_j^l(t)$ or $n_j^l(t)$ is much larger than $x_j^l(t) + h_j^l(t)$. When $x_j^l(t) + h_j^l(t) \approx n_j^l(t)$, the observation $y_j^l(t)$ is non-linearly related to $x_j^l(t) + h_j^l(t)$ and $n_j^l(t)$.

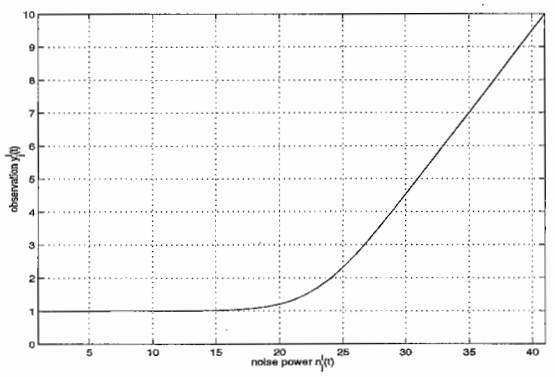

Figure 1.1: Plot of function $y_j^l(t) = x_j^l(t) + h_j^l(t) + \log(1 + \exp(n_j^l(t) - x_j^l(t) - h_j^l(t)))$. $x_j^l(t) = 1.0$, and $h_j^l(t) = 0.0$. $n_j^l(t)$ ranges from -10.0 to 10.0.

Accordingly, environment compensation includes two steps, the environment parameter estimation step and an acoustic model (or feature) adaptation step. In the environment parameter estimation step, Λ_N is estimated based on $y_j^l(t)$ and $x_j^l(t)$. Note that, if assuming that $x_j^l(t)$ is clean speech (as that in PMC [5]) and the environment is stationary, environment parameter Λ_N can be estimated directly from the explicit noise-along segments. In such a case, $h_j^l(t)$ and $n_j^l(t)$

are not function of $x_j^l(t)$. In contrast to the approach, in this work, parameters of $h_j^l(t)$ and $n_j^l(t)$ need to be estimated given $x_j^l(t)$ and $y_j^l(t)$. With the estimated parameter of $h_j^l(t)$ and $n_j^l(t)$, function (1.2) is applicable to the situation that environment segments are not explicitly available or the acoustic models of $x_j^l(t)$ are trained from noisy speech.

As shown in Figure 1.1, the noise power is masked by speech power in the situation that the noise power is smaller than a certain value. This non-linearity of the function (1.2) may result in estimates of the $h_j^l(t)$ and $n_j^l(t)$ that are different from true channel distortion and noise. In this sense, it is better to view the estimates as parameters for the non-linear mapping by (1.2), instead of explicit meaning of environment parameter. However, in the sequel, we still denote Λ_N as the environment model, though the estimate may not be true environment parameter.

Normally, a direct observation of $x_j^l(t)$ is not available, so the parameters of $h_j^l(t)$ and $n_j^l(t)$ are estimated from Λ_X (the model of $x_j^l(t)$), and $y_j^l(t)$ in either a supervised (with correct transcript) or unsupervised (correct transcript is not known) way.

In the acoustic model (or feature) adaptation step, the estimated parameter of $h_j^l(t)$ and $n_j^l(t)$ are used in function (1.2) to transform Λ_X (which substitutes $x_j^l(t)$ in function (1.2)) in the model space, so that the transformed model $\hat{\Lambda}_Y$ is close to $\{y(t) : t = 1, \cdots, T\}$. Similarly, the transformation can be carried out in the feature space to make $\{y(t) : t = 1, \cdots, T\}$ close to Λ_X.

1.4 Noise adaptive speech recognition

Furthermore, consider that the noisy environment may change during the recognition process. Λ_N (in (1.3)) thus have to be estimated sequentially, i.e., frame-by-frame.

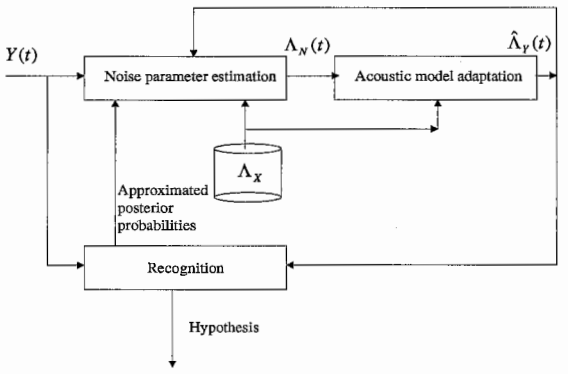

Figure 1.2: Diagram of the noise adaptive speech recognition. Λ_X, $\Lambda_N(t)$ and $\hat{\Lambda}_Y(t)$ are the original acoustic model, noise model at frame t, and adapted acoustic model at frame t, respectively. $Y(t)$ is the input noisy speech observation sequence till frame t. Recognition module provides approximated posterior probabilities of state sequences given noisy observation sequences till frame t to the noise parameter estimation module, which output $\Lambda_N(t)$ to adapt acoustic model Λ_X to $\hat{\Lambda}_Y(t)$.

In this work, a noise adaptive speech recognition is proposed to do sequential estimation of the time-varying environment parameter for noisy speech recognition. It works in the model

space, i.e., modifying HMM parameters. Its diagram is shown in Figure 1.2. Details are shown in the following sections. In Section 1.4.1, the objective function for time-varying environment parameter estimation is defined. The Viterbi approximation of the posterior probabilities of state sequences given noisy observation sequences is described in Section 1.4.2. Section 1.4.3 provides the detailed implementation.

1.4.1 Objective function for time-varying environment parameter estimation

Denote the estimated environment parameter sequence till frame $t-1$ as $\Lambda_N(t-1) = (\hat{\lambda}_N(1), \hat{\lambda}_N(2), \cdots, \hat{\lambda}_N(t-1))$. Given the current observation sequence $Y(t) = (y(1), y(2), \cdots, y(t))$ till frame t, the environment parameter estimation procedure will find $\hat{\lambda}_N(t)$ as the current environment parameter estimate, which satisfies,

$$l_t(\hat{\lambda}_N(t)) \geq l_t(\hat{\lambda}_N(t-1)) \tag{1.4}$$

where

$$\begin{aligned} l_t(\hat{\lambda}_N(t)) &= \log P(Y(t)|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t))) \\ &= \log \sum_{S(t)} P(Y(t), S(t)|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t))) \end{aligned} \tag{1.5}$$

and

$$\begin{aligned} l_t(\hat{\lambda}_N(t-1)) &= \log P(Y(t)|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t-1))) \\ &= \log \sum_{S(t)} P(Y(t), S(t)|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t-1))) \end{aligned} \tag{1.6}$$

$S(t) = (s(1), s(2), \cdots, s(t))$ is the state sequence till frame t. The formula shows that the updated environment parameter sequence $(\Lambda_N(t-1), \hat{\lambda}_N(t))$ will not decrease the likelihood of observation sequence $Y(t)$, over that given by the previous estimate of the environment parameter $\hat{\lambda}_N(t-1)$ concatenated with the previously estimated environment parameter sequence $\Lambda_N(t-1)$.

Since $S(t)$ is hidden, at each frame, we iteratively maximize the lower bound of the log-likelihood according to Jensen's inequality, i.e.,

$$\begin{aligned} &\log P(Y(t)|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t))) = \\ &\qquad \log \sum_{S(t)} P(Y(t), S(t)|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t))) \\ &\geq \sum_{S(t)} P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \lambda^{\star}_N(t))) \\ &\qquad \log \frac{P(Y(t), S(t)|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t)))}{P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \lambda^{\star}_N(t)))} \\ &= \sum_{S(t)} P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \lambda^{\star}_N(t))) \\ &\qquad \log\{P(Y(t), S(t)|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t)))\} + Z \end{aligned} \tag{1.7}$$

where Z is not a function of $\hat{\lambda}_N(t)$.

Define auxiliary function as

$$\begin{aligned} & Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) = \\ & \quad \sum_{S(t)} P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \lambda_N^\star(t))) \\ & \quad \log\{P(Y(t), S(t)|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t)))\} \end{aligned} \tag{1.8}$$

It provides the objective function to be maximized by sequential EM algorithm [20].

The algorithm is carried out by iterations between the procedure to calculate the posterior probabilities $P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \lambda_N^\star(t)))$, and maximization of the objective function to obtain $\hat{\lambda}_N(t)$. For each iteration, estimated $\hat{\lambda}_N(t)$ is for initialization of $\lambda_N^\star(t)$ in the next iteration.

Forgetting factor $\rho(0 < \rho \leq 1.0)$ can be adopted to improve convergence rate by reducing the effects of past observations relative to the new input, so that the auxiliary function is modified to [20]

$$\begin{aligned} & Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) = \\ & \quad \sum_{\tau=1}^{t} \rho^{t-\tau} \sum_{s(\tau)} P(s(\tau)|Y(\tau), \Lambda_X, (\Lambda_N(\tau-1), \lambda_N^\star(\tau))) \\ & \quad \log\{P(Y(\tau), s(\tau)|\Lambda_X, (\Lambda_N(\tau-1), \hat{\lambda}_N(\tau)))\} \end{aligned} \tag{1.9}$$

The objective function by sequential Kullback proximal algorithm [19] (derived in Appendix .2) is obtained by adding a Kullback-Leibler (K-L) divergence between $P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t-1)))$ and $P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t)))$ into the above objective functions. So the new objective function is given by,

$$\begin{aligned} & Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) - \\ & \quad (\beta_t - 1) \sum_{S(t)} P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t-1))) \\ & \quad \log \frac{P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t-1)))}{P(S(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t)))} \end{aligned} \tag{1.10}$$

where $\beta_t \in R^+$ works as a relaxation factor. The sequential EM algorithm is a special case of this algorithm and corresponds to setting β_t equal to 1.0 in the algorithm. The algorithm holds the objective in (1.4) (Proofs are in Appendix .3).

As shown in Appendix .4, the sequential Kullback proximal algorithm can be viewed as a constrained optimization problem. When $\beta_t \geq 1.0$, the estimate is constrained optimization of the auxiliary function (1.8) with a regularization term $I_t(\lambda_N^\star(t); \hat{\lambda}_N(t))$. The larger the β_t, the stronger the constraint. Thus, the estimate by the sequential Kullback proximal algorithm could be smooth in this situation. When $0 < \beta_t < 1.0$, the estimate by sequential Kullback proximal algorithm is a constrained maximization of the K-L divergence $I_t(\lambda_N^\star(t); \hat{\lambda}_N(t))$ with constraint from $-Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) \leq C$. The constraint on $-Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t))$ will be very tight when $\beta_t \to 1.0$, which results in estimate by sequential EM algorithm. When $\beta_t \to 0.0$, the constraint on $-Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t))$ is so loose that the estimate is far away from that given by sequential EM algorithm.

1.4.2 Approximation of the posterior probability

Normally, time-varying environment parameter estimation is carried out separately from the recognition process, as that in [14][15], by sequential EM algorithm with summation over all state/mixture sequences of a separately trained acoustic model. In fact, the joint likelihood of observation sequence $Y(t)$ and state sequence $S(t)$ can be approximately obtained from the Viterbi process, i.e.,

$$\begin{aligned} P(Y(t), S(t)|\Lambda_X, \Lambda_N(t)) \approx a_{s^\star(t-1)s(t)} b_{s(t)}(y(t)) \\ P(Y(t-1), S^\star(t-1)|\Lambda_X, \Lambda_N(t-1)) \end{aligned} \tag{1.11}$$

where the previous state $s^\star(t-1)$ for decision of $S^\star(t-1)$ is given as,

$$\begin{aligned} s^\star(t-1) = \arg\max_{s(t-1)} a_{s(t-1)s(t)} \cdot \\ P(Y(t-1), S(t-1)|\Lambda_X, \Lambda_N(t-1)) \end{aligned}$$

By normalizing the joint likelihood with respect to the sum of those from all active partial state sequences in the recognition stage, an approximation of the posterior probability of state sequence can be obtained. Thus in (1.7) and (1.10), instead of summing over all state/mixture sequences, the summation is over all *active partial state sequence* (path) till frame t provided by Viterbi process. By Jensen's inequality (1.7), the summation still provides the lower bound of the log-likelihood. This approximation makes it easy to combine time-varying environment parameter estimation with the Viterbi process. We thus denote this scheme of time-varying environment parameter estimation as noise adaptive speech recognition since the same Viterbi process is shared by the recognition process and time-varying environment parameter estimation process.

1.4.3 Implementation

Time-varying environment parameter estimation is carried out in the log-spectral domain. The environment model $\hat{\lambda}_N(t)$ is a single Gaussian with concatenation of the time-varying channel distortion mean vector $\mu_h^l(t) \in R^{J\times 1}$ and time-varying noise mean vector $\mu_n^l(t) \in R^{J\times 1}$, which need to be estimated, and constant diagonal covariance $\mathbf{\Sigma}_N^l \in R^{2J\times 2J}$. At each frame, the pre-trained mean vector $\mu_{ik}^l \in R^{J\times 1}$ in each mixture k of state i in acoustic models is transformed by a non-linear transformation in the log-spectral domain,

$$\hat{\mu}_{ik}^l(t) = \mu_{ik}^l + \mu_h^l(t) + \log(1 + \exp(\mu_n^l(t) - \mu_{ik}^l - \mu_h^l(t))) \tag{1.12}$$

Cepstral mean vector $\hat{\mu}_{ik}(t) \in R^{D\times 1}$ of the adapted model $\hat{\Lambda}_Y(t)$ is obtained by DCT on the above transformed mean vector $\hat{\mu}_{ik}^l(t)$. Note that function (1.12) is an approximation to function (1.2) with the assumption that the "channel" $h_j^l(t)$ and "noise" $n_j^l(t)$ have very small variance.

Thus, by (1.12), the likelihood density function is related to environment parameters as that shown in (1.3). The log-likelihood density function for mixture k in state i is given by

$$\begin{aligned} \log b_{ik}(y(t)) = -\frac{D}{2}\log(2\pi) \\ -\frac{1}{2}\log|\mathbf{\Sigma}_{ik}| - \frac{1}{2}(y(t) - \hat{\mu}_{ik}(t))^T \mathbf{\Sigma}_{ik}^{-1}(y(t) - \hat{\mu}_{ik}(t)) \end{aligned} \tag{1.13}$$

where superscript T denotes transpose operation.

Let $\hat{\lambda}_N(t)$ denote the mean vector $\mu_h^l(t)$ and $\mu_n^l(t)$. $\hat{\lambda}_N(0)$ is the initial parameter. The time-varying channel parameter $\mu_h^l(t)$ and noise parameter $\mu_n^l(t)$ are estimated by the sequential Kullback proximal algorithm shown below (Detailed derivation is in Appendix .5). Given $Y(t)$, the recursive update of $\hat{\lambda}_N(t)$ is given as,

$$\hat{\lambda}_N(t) \leftarrow \hat{\lambda}_N(t-1) - \frac{\frac{\partial Q_t(\hat{\lambda}_N(t-1);\hat{\lambda}_N)}{\partial \hat{\lambda}_N}}{\beta_t \frac{\partial^2 Q_t(\hat{\lambda}_N(t-1);\hat{\lambda}_N)}{\partial \hat{\lambda}_N^2} + (1-\beta_t)\frac{\partial^2 l_t(\hat{\lambda}_N)}{\partial \hat{\lambda}_N^2}}|_{\hat{\lambda}_N=\hat{\lambda}_N(t-1)} \tag{1.14}$$

where the first-order derivative of the auxiliary function with respect to the environment parameter is given as,

$$\frac{\partial Q_t(\hat{\lambda}_N(t-1);\hat{\lambda}_N)}{\partial \hat{\lambda}_N} = \sum_{s(t)}\sum_{k(t)} P(s(t)k(t)|Y(t),\Lambda_X,(\Lambda_N(t-1),\hat{\lambda}_N(t-1))) \frac{\partial \log b_{s(t)k(t)}(y(t))}{\partial \hat{\lambda}_N} \tag{1.15}$$

and similarly for its second order derivative with respect to the environment parameter

$$\frac{\partial^2 Q_t(\hat{\lambda}_N(t-1);\hat{\lambda}_N)}{\partial \hat{\lambda}_N^2} = \rho \cdot \frac{\partial^2 Q_{t-1}(\hat{\lambda}_N(t-2);\hat{\lambda}_N)}{\partial \hat{\lambda}_N^2} + \sum_{s(t)}\sum_{k(t)} P(s(t)k(t)|Y(t),\Lambda_X,(\Lambda_N(t-1),\hat{\lambda}_N(t-1))) \frac{\partial^2 \log b_{s(t)k(t)}(y(t))}{\partial \hat{\lambda}_N^2} \tag{1.16}$$

The second order derivative of the log-likelihood $l_t(\hat{\lambda}_N)$ with respect to the environment parameter is given as,

$$\frac{\partial^2 l_t(\hat{\lambda}_N)}{\partial \hat{\lambda}_N^2} = \sum_{s(t)}\sum_{k(t)} P(s(t)k(t)|Y(t),\Lambda_X,(\Lambda_N(t-1),\hat{\lambda}_N(t-1))) [(\frac{\partial \log b_{s(t)k(t)}(y(t))}{\partial \hat{\lambda}_N})^2 + \frac{\partial^2 \log b_{s(t)k(t)}(y(t))}{\partial \hat{\lambda}_N^2}] - (\frac{\partial Q_t(\hat{\lambda}_N(t-1);\hat{\lambda}_N)}{\partial \hat{\lambda}_N})^2 \tag{1.17}$$

By (1.13), it has,

$$\frac{\partial \log b_{s(t)k(t)}(y(t))}{\partial \hat{\lambda}_N} = \mathbf{G}_{\hat{\lambda}_N} \frac{\partial \hat{\mu}^l_{s(t)k(t)}(t)}{\partial \hat{\lambda}_N}$$

$$\frac{\partial^2 \log b_{s(t)k(t)}(y(t))}{\partial \hat{\lambda}_N^2} = \mathbf{H}_{\hat{\lambda}_N}(\frac{\partial \hat{\mu}^l_{s(t)k(t)}(t)}{\partial \hat{\lambda}_N})^2 + \mathbf{G}_{\hat{\lambda}_N}\frac{\partial^2 \hat{\mu}^l_{s(t)k(t)}(t)}{\partial \hat{\lambda}_N^2}$$

where the jjth element in diagonal matrices $\mathbf{G}_{\hat{\lambda}_N} \in R^{J\times J}$ and $\mathbf{H}_{\hat{\lambda}_N} \in R^{J\times J}$ are given as $G_{\hat{\lambda}_N jj} = \sum_{d=1}^{D}[z_{dj}\frac{(y_t(d)-\hat{\mu}_{s(t)k(t)d}(t-1))}{\Sigma^2_{s(t)k(t)d}}]$ and $H_{\hat{\lambda}_N jj} = \sum_{d=1}^{D}[-\frac{1}{\Sigma^2_{s(t)k(t)d}}z_{dj}^2]$, respectively. z_{dj} is the DCT coefficient.

By (1.12), the j-th element in the vector of the first- and second-order differential coefficients of $\hat{\mu}^l_{s(t)k(t)}(t)$ with respect to the channel parameter and noise parameter are respectively given as

$$\frac{\partial \hat{\mu}^l_{s(t)k(t)j}(t)}{\partial \mu^l_{hj}(t)} = 1 - \frac{\exp(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t))}{1+\exp(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t))} \tag{1.18}$$

$$\frac{\partial \hat{\mu}^l_{s(t)k(t)j}(t)^2}{\partial^2 \mu^l_{hj}(t)} = \frac{\exp(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t))}{(1+\exp(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t)))^2} \tag{1.19}$$

and

$$\frac{\partial \hat{\mu}^l_{s(t)k(t)j}(t)}{\partial \mu^l_{nj}(t)} = \frac{\exp(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t))}{1+\exp(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t))} \tag{1.20}$$

$$\frac{\partial \hat{\mu}^l_{s(t)k(t)j}(t)^2}{\partial^2 \mu^l_{nj}(t)} = \frac{\exp(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t))}{(1+\exp(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t)))^2} \tag{1.21}$$

The posterior probability at state $s(t)$ and mixture $k(t)$ given observation sequence $Y(t)$ and noise parameter sequence $(\Lambda_N(t-1), \hat{\lambda}_N(t))$ is approximated by Viterbi process as described in subsection 1.4.2.

Implementation on MFCC generated from spectral amplitude

As shown in Appendix .1, MFCC can also be generated from amplitude of the FFT coefficients. In such a case, the environment effects can be approximated by (15). Accordingly, (1.18) to (1.21) are modified to

$$\frac{\partial \hat{\mu}^l_{s(t)k(t)j}(t)}{\partial \mu^l_{hj}(t)} = 1 - \frac{\exp(2(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t)))}{1+\exp(2(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t)))} \tag{1.22}$$

$$\frac{\partial \hat{\mu}^{2l}_{s(t)k(t)j}(t)}{\partial^2 \mu^l_{hj}(t)} = \frac{2\exp(2(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t)))}{(1+\exp(2(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t))))^2} \tag{1.23}$$

and

$$\frac{\partial \hat{\mu}^l_{s(t)k(t)j}(t)}{\partial \mu^l_{nj}(t)} = \frac{\exp(2(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t)))}{1 + \exp(2(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t)))} \tag{1.24}$$

$$\frac{\partial \hat{\mu}^{2l}_{s(t)k(t)j}(t)}{\partial^2 \mu^l_{nj}(t)} = \frac{2\exp(2(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t)))}{(1 + \exp(2(\mu^l_{nj}(t) - \mu^l_{s(t)k(t)j} - \mu^l_{hj}(t))))^2} \tag{1.25}$$

1.5 Experimental results

In order to show the efficacy of the proposed method for noisy speech recognition, we conducted experiments in simulated noisy environments and real noisy environments. In the first set of experiments shown in Section 1.5.1, acoustic models were trained from clean speech. By varying contaminating noise power, we showed that the noise adaptive speech recognition can estimate the time-varying noise parameter during the recognition stage. The second set of experiments shown in Section 1.5.2 were conducted on AURORA 3 database, which contains continuous digits utterances collected in real car environments.

1.5.1 Experiments on acoustic models trained from clean speech

Experimental setup

Three systems were compared in the experiments conducted on subsets of TI-Digits database. The first was the baseline without noise compensation, denoted as Baseline, and the second was the system with noise compensation by (1.12) assuming stationary noise, i.e., $\mu^l_n(t)$ was kept as constant once initialized, denoted as Normal. The third was the noise adaptive recognition system by (1.14). It is denoted according to the relaxation factor β_t set. Forgetting factor ρ in (1.9) and (1.16) was set to 0.995 empirically. These systems were compared in the view of the averaged relative error rate reduction (ERR) in noises, which is calculated as the average of the relative error rate reductions in the noise.

Digits and silence were respectively modeled by 10-state and 3-state whole word HMMs with 4 diagonal Gaussian mixtures in each state. The window size was 25.0ms with a 10.0ms shift. Twenty-six filter banks were used in the binning stage. The features were MFCC + C0, with feature vector dimension D equal to 13.

Experiments were performed on TI-Digits database down-sampled to 16kHz. Five hundred clean speech utterances from 15 speakers were used for training and 111 utterances unseen in the training set were used for testing.

Four seconds of contaminating noise was used in each experiment to obtain noise mean vector for Normal. It was also for initialization of $\mu^l_n(0)$ in the noise adaptive system. Baseline performance in clean condition was 97.89% word accuracy (WA).

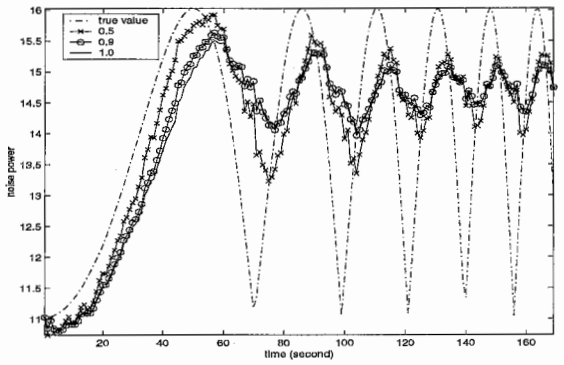

Figure 1.3: Estimation of the time-varying parameter $\mu_n^l(t)$ by the noise adaptive systems in the 12th filter bank. Estimates are labeled according to the relaxation factor β_t. The dashed-dotted curve shows evolution of the true noise power in the filter bank.

Table 1.1: Word Accuracy (in %) in simulated non-stationary noise, achieved by the noise adaptive system as a function of β_t in comparison with baseline without noise compensation (Baseline), and noise compensation assuming stationary noise (Normal).

Baseline	Normal	0.5	0.9	1.0
34.34	58.73	95.48	95.48	95.48

Speech recognition in simulated non-stationary noise

White noise signals were multiplied by a Chirp signal, so that the noise power, e.g., in the 12th filter bank, changed continuously as the dash-dotted curve shown in Figure 1.3. The SNR ranged from 0dB to 20.4dB. We also plotted the estimated noise power versus time in the filter bank by the noise adaptive system.

Observations are as follows. First, the noise adaptive system can track the evolution of the true noise power. Second, the results show that the smaller the relaxation factor β_t, the faster the convergence rate in estimation process. For example, estimation by $\beta_t = 0.5$ shows much better tracking performance than that by setting $\beta_t = 1.0$.

In terms of performance, Table 1.1 shows that the noise adaptive system achieves significant performance improvement over "Baseline" and "Normal".

Speech recognition in real noise

Speech signals were contaminated by non-stationary Babble noise in different SNRs. Recognition performances are shown in Table 1.2, together with "Baseline" and "Normal". It is observed that, in all SNR conditions, the noise adaptive system can further improve system performance, compared to that obtained by "Normal", over "Baseline". For example, in 21.5dB, the "Baseline" achieved 34.04% WA, and "Normal" attained 95.18%. The noise adaptive system with $\beta_t = 1.0$

Table 1.2: Word Accuracy (in %) in Babble noise, achieved by the noise adaptive system as a function of β_t in comparison with baseline without noise compensation (Baseline), and noise compensation assuming stationary noise (Normal). Relative error rate reduction (ERR) as a function of β_t over Normal are in the last row.

SNR (dB)	Baseline	Normal	0.5	0.9	1.0
29.5	96.69	96.69	97.59	97.89	97.89
21.5	34.04	95.18	96.39	96.69	96.69
13.6	25.30	83.13	90.96	91.27	91.27
7.6	16.27	73.19	75.60	75.30	75.30
ERR (in %)			26.9	30.9	30.9

Table 1.3: Word Accuracy (in %) in the Chirp-signal-multiplied Babble noise, achieved by the noise adaptive system as a function of β_t in comparison with baseline without noise compensation (Baseline), and noise compensation assuming stationary noise (Normal). Relative error rate reduction (ERR) as a function of β_t over Normal are in the last row.

SNR (dB)	Baseline	Normal	0.5	0.9	1.0
12.4	28.31	64.14	93.07	92.77	92.17
6.9	17.17	50.00	82.83	82.23	81.93
4.4	16.87	48.49	74.10	71.99	71.69
-1.6	14.76	37.65	47.59	50.0	51.51
ERR (in %)			53.0	52.4	52.3

achieved 96.69% WA. As a whole, the adaptive system with β_t set to 0.5, 0.9, and 1.0, achieved, respectively, 26.9%, 30.9%, and 30.9% relative error rate reduction (ERR) over that by "Normal".

We then increased the non-stationarity of the Babble noise by multiplying the noise signal with the Chirp signal as that in subsection 1.5.1. Results are shown in Table 1.3. It is observed that the Relative error rate reduction (ERR) of the noise adaptive system are larger than those in Table 1.2.

We also tested systems in highly non-stationary Machine-gun noise. Through results shown in Table 2.2, we observe that the noise adaptive system can improve recognition performance in the noise.

1.5.2 Experiments on acoustic models trained from noisy speech

Experimental setup

In this section, we show the validity of the noise adaptive speech recognition in the situation that the acoustic models were trained from noisy speech. Environment effects include channel distortion and background noises. Note that the acoustic models were trained from noisy speech.

Table 1.4: Word Accuracy (in %) in Machine-gun noise, achieved by the noise adaptive system as a function of β_t in comparison with baseline without noise compensation (Baseline), and noise compensation assuming stationary noise (Normal). Relative error rate reduction (ERR) as a function of β_t over Normal are in the last row.

SNR (dB)	Baseline	Normal	0.5	0.9	1.0
33.3	91.87	93.37	96.69	95.48	97.59
28.8	87.95	90.60	94.28	95.18	94.28
22.8	78.61	81.33	87.05	83.43	82.83
20.9	77.41	79.82	83.73	85.24	76.51
ERR (in %)			34.8	29.7	23.6

In this situation, many model-based methods, e.g. PMC, can not work, since they require acoustic models trained from clean speech. A normal way to do environment robustness is employing adaptation methods such as MLLR [10].

Thus, we compared three systems. The first system, denoted as "Baseline", was the system neither with MLLR nor noise adaptive speech recognition. The second system, denoted as "+MLLR", was with acoustic models adapted by supervised MLLR. The third system, denoted as "+ Noise adaptive recognition", was with acoustic models firstly adapted by the supervised MLLR, and then with noise adaptive speech recognition by (1.14). The only difference between the second and the third system is if noise adaptive speech recognition was applied.

The experiments were conducted on the AURORA 3 database, which is a subset of the SpeechDat-Car (SDC) corpus collected in cars through close-talking microphones and hands-free microphones with different driving conditions, e.g., High-speed, Low-speed, and various placing configurations, such as climate control on/off and sunroof open/closed, etc. We showed experimental results on Spanish, Finnish, and Danish subset of the database.

In all of the tested languages, three sets of evaluations are provided. The Well-matched (WM) evaluation has training and testing set from utterances through both microphone types and all driving conditions. The Medium-mismatched (MM) evaluation utilizes training data from hands-free microphones using all driving conditions except for the High Speed driving condition. The testing set has data from hands-free microphones and the High Speed driving condition only. High-mismatched (HM) evaluation utilizes data from close-talking microphones and all driving conditions. The testing set in the evaluation has data through hands-free microphones and all driving conditions except the Stopped Motor driving condition.

Window size for FFT was 25 ms, and time-shift was 10 ms. The number of filter banks and dimension of static cepstral coefficients were 26 and 13, respectively. MFCCs were generated from linear-spectral amplitude[1]. Regression window length for both of the first- and second-order

[1]The MFCCs in this work has slight difference from traditional MFCCs in the sense that a median filtering is introduced after FFT. The median filter was added in each frequency bin after FFT to filter the sequence of linear spectral amplitude coefficients along frame index. The median of the input sequence was extracted in order to smooth the sequence. The filter length was set according to the spectral frequency. The higher the frequency, the longer the median filter length. In this work, the longest filter length was 10 frames and the shortest filter length

MFCC coefficients was 2. Feature dimension was 39.

The HMM back-end was defined by AURORA 3 task as 18 states with 3 Gaussian mixtures in each state for speech models, and five state HMM with 6 mixtures in each state for silence model. A three state tee model with six Gaussian mixtures in each state was used to model short-pause between speech events[2].

Block diagonal matrix for MLLR was used in this work where each sub-matrix for static, first- and second-order coefficients was full. MLLR was supervised with 22 adaptation utterances in each evaluation set. Noise adaptive speech recognition took one iteration at each frame. It had forgetting factor $\rho = 0.995$ and relaxation factor $\beta_t = 0.95$. At the beginning for each evaluation, channel parameter $\mu_h^l(t)$ and noise parameter $\mu_n^l(t)$ were initialized to be zero vector. The noise adaptive speech recognition was with beam-width of 300 for the evaluation. Note that, since the spectral amplitude was used in this work, the noise adaptive speech recognition made use of (1.22) to (1.25).

Experimental results

Table 1.5: Word accuracy of the Finnish set. WM, MM, and HM each denotes Well-matched evaluation, Medium-mismatched evaluation, and High-mismatched evaluation. Baseline denotes system without acoustic model adaptation. +MLLR denotes system with supervised MLLR adaptation of acoustic models. +Noise adaptive recognition denotes system with combination of the supervised MLLR and the noise adaptive speech recognition.

	WM	MM	HM
Baseline	93.68%	76.35%	68.79%
+MLLR	93.50%	78.05%	85.64%
+Noise adaptive recognition	93.77%	83.35 %	85.27%

Recognition accuracies by the three systems are shown in Table 1.5 to Table 1.7 for Finnish, Danish, and Spanish, respectively. It is observed that MLLR was effective to improve system performance in the three sets of evaluations. Though adding MLLR slightly decreased word accuracies in Well-matched evaluations in the Finnish and Danish set, it effectively improved recognition accuracies in other evaluations sets. For example, in Well-matched evaluation in the Spanish set, word accuracy increased from 92.97% attained by the "Baseline" to 93.54% by "+MLLR".

By noise adaptive speech recognition on the acoustic models adapted by MLLR, a competitive performance improvements were observed in the evaluations. Relative error rate reductions (ERR) of the "+noise adaptive recognition" over system "+MLLR" are shown in Table 1.8. For example, in the Medium-mismatched evaluation in the Spanish set shown in Table 1.7, word accuracy was improved from 85.30% by the system "+MLLR" to 89.52% by "+ noise adaptive recognition",

was 3 frames.

[2]A segmentation module by GMMs was applied before acoustic model training[21]. It works as a speech/non-speech classifier, which remove some very noisy segments. GMMs had 32 Gaussian mixtures.

Table 1.6: Word accuracy of the Danish set. WM, MM, and HM each denotes Well-matched evaluation, Medium-mismatched evaluation, and High-mismatched evaluation. Baseline denotes system without acoustic model adaptation. +MLLR denotes system with supervised MLLR adaptation of acoustic models. +Noise adaptive recognition denotes system with combination of the supervised MLLR and the noise adaptive speech recognition.

	WM	MM	HM
Baseline	89.61%	73.87%	60.20%
+MLLR	89.28%	76.75%	76.38%
+Noise adaptive recognition	89.59%	77.00 %	77.44%

Table 1.7: Word accuracy of the Spanish set. WM, MM, and HM each denotes Well-matched evaluation, Medium-mismatched evaluation, and High-mismatched evaluation. Baseline denotes system without acoustic model adaptation. +MLLR denotes system with supervised MLLR adaptation of acoustic models. +Noise adaptive recognition denotes system with combination of the supervised MLLR and the noise adaptive speech recognition.

	WM	MM	HM
Baseline	92.97%	77.57%	61.13%
+MLLR	93.54%	85.30%	68.02%
+Noise adaptive recognition	93.93%	89.52%	72.05%

which corresponds to 28.71% of relative error rate reduction of the "+noise adaptive recognition" over "+MLLR". Through Table 1.8, it is seen that, with combination of the noise adaptive speech recognition and supervised MLLR, system performances could be further improved over those attained by supervised MLLR, though a slight decrease of the word accuracy was observed in the High-mismatched evaluation in the Finnish set.

1.6 Conclusions

We have the following observations on the results: 1) Our derivation is based on the assumption that the environments are non-stationary. The assumption fits the real situations. In the non-stationary environments, we observed improvements over noise compensation assuming stationary environments. 2) As shown in Table 1.2, the highest ERR of the adaptive system over "Normal" was achieved at β_t equal to 1.0 and 0.9, whereas it achieved the highest ERR at $\beta_t = 0.5$, when the non-stationarity of the Babble noise was increased by multiplying it with a Chirp signal. Also, we observed that the highest ERR was achieved at $\beta_t = 0.5$ in Machine-gun noise, which is more non-stationary than Babble noise. It seems that the more non-stationary the noise is, the smaller the β_t to be set[3].

[3]The β_t cannot be too small, since, otherwise, the estimation error after convergence might be large [19].

Table 1.8: Relative error rate reduction (ERR) of the noise adaptive speech recognition system with combination of the supervised MLLR, over systems with acoustic models adapted by supervised MLLR, in Finnish, Danish, and Spanish sets of the AURORA 3 database. WM, MM, and HM each denotes Well-matched evaluation, Medium-mismatched evaluation, and High-mismatched evaluation.

	WM	MM	HM
Finnish	4.15%	24.15%	-2.58%
Danish	2.89%	1.08%	4.49%
Spanish	6.04%	28.71%	12.60%

Our results on Aurora 3 database also show that the noise adaptive speech recognition is applicable when speech models were trained from noisy speech. In this situation, the "environment" parameter estimated may not have the explicit meaning of environment parameter, but works as the parameter for the parametric mapping (1.2).

Our results also show that it is possible to improve system robustness to environment effects by combining adaptation methods, e.g., MLLR, with the noise adaptive speech recognition. They can possibly boost each other. MLLR can adapt the static, first- and second-order feature coefficients in a batch or incremental way before recognition. The noise adaptive speech recognition further adapts static feature coefficients of the acoustic models in an un-supervised way frame-by-frame during the recognition stage.

The above results have shown that the noise adaptive speech recognition improves system performances in non-stationary environments. Results also show a possible relationship between the best relaxation factor β_t of the recursive environment parameter estimation and the environments. Further improvement in this research can be achieved via incorporation of adaptation for the dynamic features and refinement of acoustic models.

Chapter 2

Sequential Monte Carlo method

2.1 Abstract

We present a sequential Monte Carlo method applied to additive noise compensation for robust speech recognition in time-varying noise. The method generates a set of samples according to the prior distribution given by clean speech models and noise prior evolved from previous estimation. An explicit model representing noise effects on speech features is used, so that an extended Kalman filter is constructed for each sample, generating the updated continuous state estimate as the estimation of the noise parameter, and prediction likelihood for weighting each sample. Minimum mean square error (MMSE) inference of the time-varying noise parameter is carried out over these samples by fusion the estimation of samples according to their weights. A residual resampling selection step and a Metropolis-Hastings smoothing step are used to improve calculation efficiency. Experiments were conducted on speech recognition in simulated non-stationary noises, where noise power changed artificially, and highly non-stationary Machinegun noise. In all the experiments carried out, we observed that the method can have significant recognition performance improvement, over that achieved by noise compensation with stationary noise assumption.

2.2 Introduction

Speech recognition in noise has been considered to be essential for its real applications. There have been active research efforts in this area. Among many approaches, model-based approach assumes explicit models representing noise effects on speech features. In this approach, most researches are focused on stationary or slow-varying noise conditions. In this situation, environment noise parameters are often estimated before speech recognition from a small set of environment adaptation data. The estimated environment noise parameters are then used to compensate noise effects in the feature or model space for recognition of noisy speech.

However, it is well-known that noise statistics may vary during recognition. In this situation, the noise parameters estimated prior to speech recognition of the utterances is possibly not relevant to the subsequent frames of input speech if environment changes.

A number of techniques have been proposed to compensate time-varying noise effects. They can be categorized into two approaches. In the first approach, time-varying environment sources

are modeled by Hidden Markov Models (HMM) or Gaussian mixtures that were trained by prior measurement of environments, so that noise compensation is a task of identification of the underlying state sequences of the noise HMMs, e.g., in [12], by maximum a posterior (MAP) decision. This approach requires making a model representing different conditions of environments (signal-to-noise ratio, types of noise, etc.), so that statistics at some states or mixtures obtained before speech recognition are close to the real testing environments. In the second approach, environment model parameters are assumed to be time-varying, so it is not only an inference problem but also related to environment statistics estimation during speech recognition. The parameters can be estimated by Maximum Likelihood estimation, e.g., sequential EM algorithm [14][19][22]. They can also be estimated by Bayesian methods. In the Bayesian methods, all relevant information on the set of environment parameters and speech parameters, which are denoted as $\Theta(t)$ at frame t, is included in the posterior distribution given observation sequence $Y(0:t)$, i.e., $p(\Theta(t)|Y(0:t))$. Except for a few cases including linear Gaussian state space model (Kalman filter), it is formidable to evaluate the distribution updating analytically. Approximation techniques are required. For example, in [16], a Laplace transform is used to approximate the joint distribution of speech and noise parameters by vector Taylor series. The approximated joint distribution can give analytical formula for posterior distribution updating.

We report an alternative approach for Bayesian estimation and compensation of noise effects on speech features. The method is based on sequential Monte Carlo method [23]. In the method, a set of samples is generated hierarchically from the prior distribution given by speech models. A state space model representing noise effects on speech features is used explicitly, and an extended Kalman filter (EKF) is constructed in each sample. The prediction likelihood of the EKF in each sample gives its weight for selection, smoothing, and inference of the time-varying noise parameter, so that noise compensation is carried out afterwards. Since noise parameter estimation, noise compensation and speech recognition are carried out frame-by-frame, we denote this approach as sequential noise compensation.

2.3 Speech and noise model

Our work is on speech features derived from Mel Frequency Cepstral Coefficients (MFCC). It is generated by transforming signal power into log-spectral domain, and finally, by discrete Cosine transform (DCT) to the cepstral domain. The following derivation of the algorithm is in log-spectral domain. Let t denote frame (time) index.

In our work, speech and noise are respectively modeled by HMMs and a Gaussian mixture. For speech recognition in stationary additive noise, the following formula [22] has been shown to be effective in compensating noise effects. For Gaussian mixture k_t at state s_t, the Log-Add method transforms the mean vector $\mu^l_{s_t k_t}$ of the Gaussian mixture by,

$$\hat{\mu}^l_{s_t k_t} = \mu^l_{s_t k_t} + \log(1 + \exp(\mu^l_n - \mu^l_{s_t k_t})) \tag{2.1}$$

where μ^l_n is the mean vector in the noise model. $s_t \in \{1, \cdots, S\}, k_t \in \{1, \cdots, M\}$. S and M each denote the number of states in speech models and the number of mixtures at each state. Superscript l indicates that parameters are in the log-spectral domain.

After the transformation, the mean vector $\hat{\mu}^l_{s_t k_t}$ is further transformed by DCT, and then plugged into speech models for recognition of noisy speech. In case of time-varying noise, the μ^l_n should be a function of time, i.e., $\mu^l_n(t)$. Accordingly, the compensated mean is $\hat{\mu}^l_{s_t k_t}(t)$.

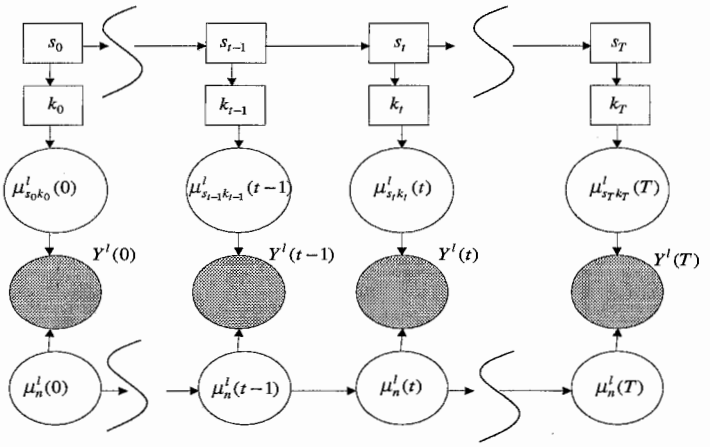

Figure 2.1: The graphical model representation of the dependences of the speech and noise model parameters. s_t and k_t each denote the state and Gaussian mixture at frame t in speech models. $\mu^l_{s_t k_t}(t)$ and $\mu^l_n(t)$ each denote the speech and noise parameter. $Y^l(t)$ is the noisy speech observation.

The following analysis can be viewed in Figure 2.1. In Gaussian mixture k_t at state s_t of speech model, speech parameter $\mu^l_{s_t k_t}(t)$ is assumed to be distributed in Gaussian with mean $\mu^l_{s_t k_t}$ and variance $\Sigma^l_{s_t k_t}$. On the other hand, since the environment parameter is assumed to be time varying, the evolution of the environment mean vector can be modeled by a random walk function, i.e.,

$$\mu^l_n(t) = \mu^l_n(t-1) + v(t) \tag{2.2}$$

where $v(t)$ is the environment driving noise in Gaussian distribution with zero mean and variance V.

Then, we have,

$$\begin{aligned} & p(s_t, k_t, \mu^l_{s_t k_t}(t), \mu^l_n(t) | s_{t-1}, k_{t-1}, \mu^l_{s_{t-1} k_{t-1}}(t-1), \mu^l_n(t-1)) \\ = \; & a_{s_{t-1} s_t} p_{s_t k_t} N(\mu^l_{s_t k_t}(t); \mu^l_{s_t k_t}, \Sigma^l_{s_t k_t}) N(\mu^l_n(t); \mu^l_n(t-1), V) \end{aligned} \tag{2.3}$$

where $a_{s_{t-1} s_t}$ is the state transition probability from s_{t-1} to s_t, and $p_{s_t k_t}$ is the mixture weight. The above formula gives the prior distribution of the set of speech and noise model parameter $\Theta(t) = \{s_t, k_t, \mu^l_{s_t k_t}(t), \mu^l_n(t)\}$.

Furthermore, given observation $Y^l(t)$, assume that the transformation by (2.1) has modeling and measurement uncertainty in Gaussian distribution, i.e.,

$$Y^l(t) = \mu^l_{s_t k_t}(t) + \log\left(1 + \exp\left(\mu^l_n(t) - \mu^l_{s_t k_t}(t)\right)\right) + w_{s_t k_t}(t) \tag{2.4}$$

where $w_{s_t k_t}(t)$ is Gaussian with zero mean and variance $\Sigma^l_{s_t k_t}$, i.e., $N(\cdot; 0, \Sigma^l_{s_t k_t})$. Thus, the likelihood of observation $Y^l(t)$ at state s_t and mixture k_t is

$$p(Y^l(t)|\Theta(t)) = N(Y^l(t); \mu^l_{s_t k_t}(t) + \log\left(1 + \exp\left(\mu^l_n(t) - \mu^l_{s_t k_t}(t)\right)\right), \Sigma^l_{s_t k_t}) \tag{2.5}$$

Refereeing to (2.3) and (2.5), the posterior distribution of $\Theta(t)$ given $Y^l(t)$ is

$$p(s_t, k_t, \mu^l_{s_t k_t}(t), \mu^l_n(t)|Y^l(t)) \propto \\ p(Y^l(t)|\Theta(t)) a_{s_{t-1}s_t} p_{s_t k_t} N(\mu^l_{s_t k_t}(t); \mu^l_{s_t k_t}, \Sigma^l_{s_t k_t}) N(\mu^l_n(t); \mu^l_n(t-1), V) \tag{2.6}$$

The time-varying noise parameter is estimated by MMSE, given as,

$$\hat{\mu}^l_n(t) = \int_{\mu^l_n(t)} \mu^l_n(t) \sum_{s_t, k_t} \int_{\mu^l_{s_t k_t}(t)} p(\Theta(t)|Y^l(0:t)) d\mu^l_{s_t k_t}(t) d\mu^l_n(t) \tag{2.7}$$

However, it is difficult to obtain the posterior distribution $p(\Theta(t)|Y^l(0:t))$ analytically, since $p(\mu^l_{s_t k_t}(t), \mu^l_n(t)|Y^l(t))$ is non-Gaussian in $\mu^l_{s_t k_t}(t)$ and $\mu^l_n(t)$ due to the non-linearity in (2.4). It is thus difficult, if possible, to assign conjugate prior of $\mu^l_n(t)$ to the likelihood function $p(Y^l(t)|\Theta(t))$. Another difficulty is that the speech state and mixture sequence is hidden in (2.7). We thus rely on the solution by computational Bayesian approach [23].

2.4 Time-varying noise parameter estimation by sequential Monte Carlo method

We apply the sequential Monte Carlo method [23] for posterior distribution updating. At each frame t, a proposal importance distribution is sampled whose target is the posterior distribution in (2.7), and it is implemented by sampling from lower distributions in hierarchy. The method goes through the sampling, selection, and smoothing steps frame-by-frame. MMSE inference of the time-varying noise parameter is a by-product of the steps, carried out after the smoothing step.

In the sampling step, the prior distribution given by speech models is set to the proposal importance distribution, i.e., $q(\Theta(t)|\Theta(t-1)) = a_{s_{t-1}s_t} p_{s_t k_t} N(\mu^l_{s_t k_t}(t); \mu^l_{s_t k_t}, \Sigma^l_{s_t k_t})$. The samples are then generated by sampling hierarchically of the prior distribution described as follows: set $i = 1$ and perform the following steps:

1. sample $s_t^{(i)} \sim a_{s^{(i)}_{t-1} s_t}$
2. sample $k_t^{(i)} \sim p_{s_t^{(i)} k_t}$
3. sample $\mu^{l(i)}_{s_t^{(i)} k_t^{(i)}}(t) \sim N(; \mu^l_{s_t^{(i)} k_t^{(i)}}, \Sigma^l_{s_t^{(i)} k_t^{(i)}})$, and set $i = i + 1$
4. repeat step 1 to 3 until $i = N$

where superscript (i) denotes the index of samples and N denotes the number of samples. Each sample represents certain speech and noise parameter, which is denoted as $\Theta^{(i)}(t) = (s_t^{(i)}, k_t^{(i)}, \mu^{l(i)}_{s_t^{(i)} k_t^{(i)}}(t), \mu^{l(i)}_n$ The weight of each sample is given by $\prod_{\tau=1}^{t} \frac{p(\Theta(\tau)^{(i)}|Y^l(\tau))}{q(\Theta(\tau)^{(i)}|\Theta(\tau-1)^{(i)})}$. Refereeing to (2.6), the weight is calculated by

$$\beta^{(i)}(t) = p(Y^l(t)|\Theta^{(i)}(t)) N(\mu^{l(i)}_n(t); \mu^{l(i)}_n(t-1), V) \check{\beta}^{(i)}(t-1) \tag{2.8}$$

where $\check{\beta}^{(i)}(t-1)$ is the sample weight from previous frame. The remaining part in the right side of above equation, in fact, represents the prediction likelihood of the state space model given by (2.2) and (2.4) for each sample (i). This likelihood can be obtained analytically since after linearization of (2.4) with respect to $\mu_n^l(t)$ at $\mu_n^{l(i)}(t-1)$, an extended Kalman filter (EKF) can be obtained, where the prediction likelihood of the EKF gives the weight, and the updated continuous state of EKF gives $\mu_n^{l(i)}(t)$.

In practice, after the above sampling step, the weights of all but several samples may become insignificant. Given the fixed number of samples, this will results in degeneracy of the estimation, where not only some computational resources are wasted, but also estimation might be biased because of losing detailed information on some parts important to the parameter estimation. A selection step by residual resampling [23] is adopted after the sampling step. The method avoids the degeneracy by discarding those samples with insignificant weights, and in order to keep the number of the samples constant, samples with significant weights are duplicated. Accordingly, the weights after the selection step are also proportionally redistributed. Denote the set of samples after the selection step as $\tilde{\Theta}(t) = \{\tilde{\Theta}^{(i)}(t); i = 1 \cdots N\}$ with weights $\tilde{\beta}(t) = \{\tilde{\beta}^{(i)}(t); i = 1 \cdots N\}$.

After the selection step at frame t, these N samples are distributed approximately according to the posterior distribution in (2.7). However, the discrete nature of the approximation can lead to a skewed importance weights distribution, where the extreme case is all the samples have the same $\tilde{\Theta}(t)$ estimated. A Metropolis-Hastings smoothing [24] step is introduced in each sample where the step involves sampling a candidate $\Theta^{\star(i)}(t)$ given the current $\tilde{\Theta}^{(i)}(t)$ according to the proposal importance distribution $q(\Theta^{\star}(t)|\tilde{\Theta}^{(i)}(t))$. The Markov chain then moves towards $\Theta^{\star(i)}(t)$ with acceptance possibility as $\min\{1, \frac{p(\Theta^{\star(i)}|Y^l(t))q(\tilde{\Theta}^{(i)}|\Theta^{\star(i)})}{p(\tilde{\Theta}^{(i)}|Y^l(t))q(\Theta^{\star(i)}|\tilde{\Theta}^{(i)})}\}$, otherwise it remains at $\tilde{\Theta}^{(i)}$. To simplify calculation, we assume that the importance distribution $q(\Theta^{\star}(t)|\tilde{\Theta}^{(i)}(t))$ is symmetric, and after some mathematical manipulation, it is shown that the acceptance possibility is given by $\min\{1, \frac{\beta^{\star(i)}(t)}{\tilde{\beta}^{(i)}(t)}\}$. Denote the obtained samples as $\check{\Theta}(t) = \{\check{\Theta}^{(i)}(t); i = 1 \cdots N\}$ with weights $\check{\beta}(t) = \{\check{\beta}^{(i)}(t); i = 1 \cdots N\}$.

Noise parameter $\mu_n^l(t)$ is estimated via MMSE over the samples, i.e.,

$$\hat{\mu}_n^l(t) = \sum_{i=1}^{N} \frac{\check{\beta}^{(i)}(t)}{\sum_{j=1}^{N} \check{\beta}^{(j)}(t)} \check{\mu}_n^{l(i)}(t)$$

where $\check{\mu}_n^{l(i)}(t)$ is the updated continuous state of the EKF in the sample after the smoothing step. Once the estimate $\hat{\mu}_n^l(t)$ has been obtained, it is plugged into (2.1) to do non-linear transformation of clean speech models.

2.5 Experimental results

2.5.1 Experimental setup

Experiments were performed on the TI-Digits database down-sampled to 16kHz. Five hundred clean speech utterances from 15 speakers and 111 utterances unseen in the training set were used for training and testing, respectively. Digits and silence were respectively modeled by 10-state and 3-state whole word HMMs with 4 diagonal Gaussian mixtures in each state.

The window size was 25.0ms with a 10.0ms shift. Twenty-six filter banks were used in the binning stage. The features were MFCC + Δ MFCC. The baseline system had a 98.7% Word Accuracy under clean conditions.

We compared three systems. The first was the baseline trained on clean speech without noise compensation, and the second was the system with noise compensation by (2.1) assuming stationary noise [22]. They were each denoted as Baseline and Stationary Compensation. The sequential method was un-supervised, i.e., without training transcript, and it was denoted according to the number of samples and variance of the environment driving noise V. Four seconds of contaminating noise was used in each experiment to obtain noise mean vector μ_n^l in (2.1) for Stationary Compensation. It was also for initialization of $\mu_n^l(0)$ in the sequential method. The initial $\mu_n^{l(i)}(0)$ for each sample was sampled from $N(\mu_n^l(0), 0.01) + N(\mu_n^l(0) + \zeta(0), 10.0)$, where $\zeta(0)$ was flat distribution in $[-1.0, 9.0]$.

2.5.2 Speech recognition in simulated non-stationary noise

White noise signal was multiplied by a Chirp signal and a rectangular signal, so that the noise power of the contaminating White noise changed continuously, denoted as experiment A, and dramatically, denoted as experiment B. As a result, signal-to-noise ratio (SNR) of the contaminating noise ranged from 0dB to 20.4dB. We plotted the noise power in 12th filter bank versus frames in Figure 2.2, together with the estimated noise power by the sequential method with number of samples set to 120 and environment driving noise variance set to 0.0001. As a comparison, we also plotted the noise power and its estimate by the method with the same number of samples but larger driving noise variance to 0.001.

By Figure 2.2 and Figure 2.3, we have the following observations. First, the method can track the evolution of the noise power. Second, the larger driving noise variance V will make faster convergence but larger estimation error of the method. In terms of recognition performance, Table 2.1 shows that the method can effectively improve system robustness to the time-varying noise. For example, with 60 samples, and the environment driving noise variance V set to 0.001, the method can improve word accuracy from 75.30% achieved by "Stationary Compensation", to 94.28% in experiment A. The table also shows that, the word accuracies can be improved by increasing number of samples. For example, given environment driving noise variance V set to 0.0001, increasing number of samples from 60 to 120, can improve word accuracy from 77.11% to 85.84% in experiment B.

2.5.3 Speech recognition in real noise

In this experiment, speech signals were contaminated by highly non-stationary Machinegun noise in different SNRs. The number of samples was set to 120, and the environment driving noise variance V was set to 0.0001. Recognition performances are shown in Table 2.2, together with "Baseline" and "Stationary Compensation". It is observed that, in all SNR conditions, the method can further improve system performance, compared to that obtained by "Stationary Compensation", over "Baseline". For example, in 8.86dB SNR, the method can improve word accuracy from 75.60% by "Stationary Compensation" to 83.13%. As a whole, the method can have a relative 39.9% word error rate reduction compared to "Stationary Compensation".

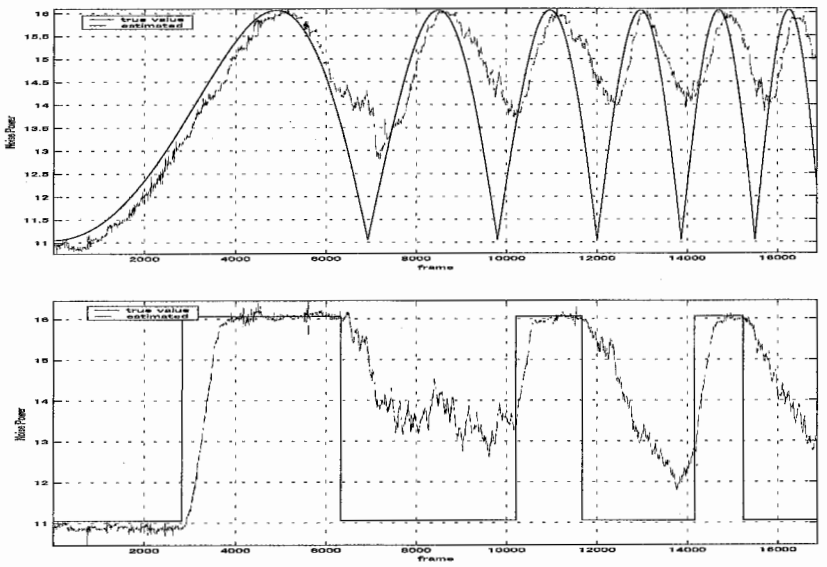

Figure 2.2: Estimation of the time-varying parameter $\mu_n^l(t)$ by the sequential Monte Carlo method at 12th filter bank in experiment A. Number of samples is 120. Environment driving noise variance is 0.0001. Solid curve is the true noise power. Dash-dotted curve is the estimated noise power.

2.6 Summary

We have presented a sequential Monte Carlo method for Bayesian estimation of time-varying noise parameter, which is for sequential noise compensation applied to robust speech recognition. The method uses samples to approximate the posterior distribution of the additive noise and speech parameters given observation sequence. Once the noise parameter has been inferred, it is plugged into a non-linear transformation of clean speech models. Experiments conducted on digits recognition in simulated non-stationary noises and real noises have shown that the method is very effective to improve system robustness to time-varying additive noise.

Table 2.1: Word Accuracy (in %) in simulated non-stationary noises, achieved by the sequential Monte Carlo method in comparison with baseline without noise compensation, denoted as Baseline, and noise compensation assuming stationary noise, denoted as Stationary Compensation.

Experiment	Baseline	Stationary Compensation	# samples = 60		# samples = 120	
			V		V	
			0.001	0.0001	0.001	0.0001
A	48.19	75.30	94.28	93.98	94.28	94.58
B	53.01	78.01	82.23	77.11	85.84	85.84

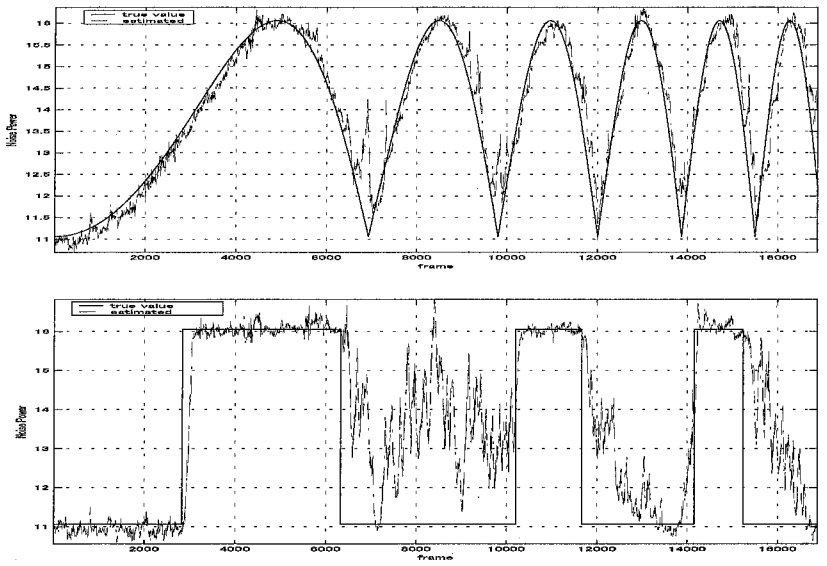

Figure 2.3: Estimation of the time-varying parameter $\mu_n^l(t)$ by the sequential Monte Carlo method at 12th filter bank in experiment A. Number of samples is 120. Environment driving noise variance is 0.001. Solid curve is the true noise power. Dash-dotted curve is the estimated noise power.

Table 2.2: Word Accuracy (in %) in Machinegun noise, achieved by the sequential Monte Carlo method in comparison with baseline without noise compensation, denoted as Baseline, and noise compensation assuming stationary noise, denoted as Stationary Compensation.

SNR (dB)	Baseline	Stationary Compensation	#samples = 120, $V = 0.0001$
28.86	90.36	92.77	97.59
14.88	64.46	76.81	88.25
8.86	56.02	75.60	83.13
1.63	50.0	68.98	72.89

Chapter 3

Generative Factor Analyzed HMM for Automatic Speech Recognition

3.1 Abstract

We present a generative factor analyzed Hidden Markov model (GFA-HMM) for automatic speech recognition. Traditional HMM models observation vectors by mixtures of Gaussian (MoG) associated with discrete-valued hidden state sequences. In this work, in addition to the hidden state sequences, the observation vectors are also represented by continuous-valued latent vectors. On contrary to principle component analysis (PCA) and factor analysis (FA), the continuous-valued latent vectors are dependent on acoustic units for recognition. Accordingly, the distributions of the latent vectors are MoGs associated to discrete state sequence. To model the correlations of the latent vectors, a further latent representation is introducted by factor analysis. Our experiments on digits recognition showed that GFA-HMM could consistently outperform traditional HMM with the same amount of training data.

3.2 Introduction

In automatic speech recognition (ASR), system is presented with multi-dimensional observation vectors which may have varies order of statistics. Normally, mixture of Gaussian (MoG) with diagonal covariances are used to model the distributions of the observation vectors, which result in implicit modeling of the correlations of the vectors. To account for dynamics of the distribution, the MoGs are assumed to be dependent on discrete state sequence, which accounts for semantic information of the speech. This gives rise to the traditional Hidden Markov model (HMM) for ASR.

Recently, continuous-valued latent representation of the observed vectors is found to be useful for pattern recognition, since they can provide compact representation of the correlations of the observation vectors. For example, the latent representation can be carried out by principle component analysis (PCA) [25] and factor analysis (FA) [26], which find various applications in image processing [27] and speech recognition [28]. The above works assume that the vectors in the introduced latent space, denoted as X, are distributed in Gaussian $N(\cdot; 0, I)$, i.e., zero mean

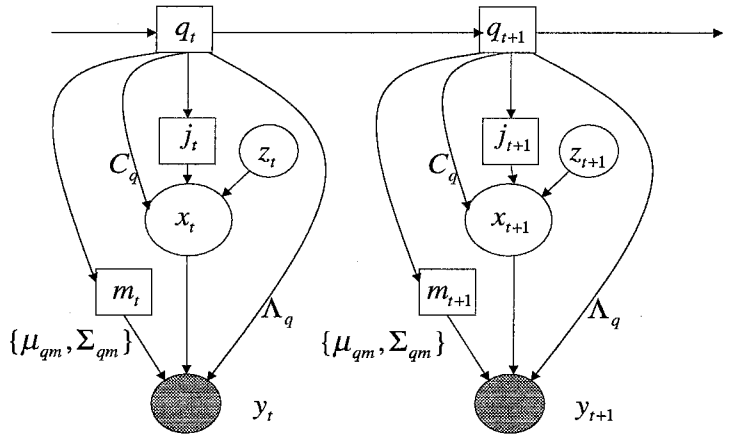

Figure 3.1: Graphical model of the generative factor analyzed HMM.

and unit diagonal covariance.

Instead of the single Gaussian distribution $N(\cdot; 0, I)$, of the latent vectors, other important works of continuous-latent representation include independent component analysis (ICA) [29] assume non-Gaussian distribution of the latent vectors. As a result, higher-order statistics of the observation vectors can be modeled.

Since speech observation vector is dynamic, it might be beneficial to consider the vector in the continuous-valued latent space as acoustic-unit dependent, and, as a result, it is no longer distributed in $N(\cdot; 0, I)$ as PCA and FA. Furthermore, instead of assuming independence of the vector elements as in ICA, these elements in the latent vectors can be considered as correlated with each other. To model the correlations, a further latent representation can be introduced.

Based on above considerations, in this work, we propose to model speech observation by a generative factor-analyzed Hidden Markov models (GFA-HMMs). The key to our approach lies in the introduction of *acoustic-unit-dependent* latent representation vector $x_t \in R^L$ of the observation vector $y_t \in R^N$ and a further *acoustic-unit-independent* latent representation vector $z_t \in R^K$ of the latent vector x_t. Since z_t is acoustic-unit independent, it could be considered as *generative* source to acoustic-unit dependent latent vector x_t via a acoustic-unit dependent loading matrix. The model is related to factor analysis since there are noises in both of the observation vector y_t and latent vector x_t.

3.3 Generative factor analyzed HMM

Figure 3.1 shows the graphical model of the generative factor analyzed HMM. Round circle and rectangular square each denotes continuous- and discrete-valued node. Shaded nodes denote observations. $q_t = \{1, \cdots, S\}$ denote discrete state at time t. $Q(T) = (q_1, \cdots, q_t, q_{t+1}, \cdots, \cdots, q_T)$ is the discrete state sequence with first-order state transition probability a_{pq} from state p to state q, which accounts for semantic sequence in speech. Two continuous-valued variables, x_t and y_t, are dependent on the discrete state sequence, whereas z_t is independent of the discrete state sequence.

The continuous-valued nodes, y_t, x_t, and z_t are hierarchical. In the highest hierarchy, vector

x_t is considered to be generated from z_t through factor analyzer [28] by C_q, the state-dependent loading matrix with dimension $L \times K$ in state q, i.e.,

$$z_t \sim p(z_t) = N(z_t; 0, I) \tag{3.1}$$

$$x_t = C_q z_t + \zeta_{qt} \tag{3.2}$$

where vector ζ_{qt} denotes noise in space X. The noise is modeled by mixture of Gaussian $\{N(\zeta_{qt}; \xi_{qj}, V_{qj})\}_{j=1,\cdots,M_q^x}$, with component weight c_{qj}. V_{qj} is diagonal. M_q^x denotes number of the mixture components for state q in space X.

Since elements in diagonal covariance V_{qj} are not restricted to have the same value, the above functions are factor analysis on x_t in each component $j \in \{1, \cdots, M_q^x\}$ at state q. The observation y_t is related to x_t by the following model.

$$x_t \sim Model^{FA-HMM} \tag{3.3}$$

$$y_t = \Lambda_q x_t + v_{qt} \tag{3.4}$$

where the observation noise v_{qt} is distributed according to mixtures of Gaussian $\{N(v_{qt}; \mu_{qm}, \Sigma_{qm})\}_{m=1,\cdots,M_q^y}$ with mixture weight π_{qm}. M_q^y is the number of mixture components in state q in space Y. Σ_{qm} is diagonal with σ_{qmn}^2 for element (n,n). Value of σ_{qmn}^2 is not restricted to be the same for $\forall n \in \{1, \cdots, N\}$. Λ_q is state-dependent loading matrix with dimension of $N \times L$.

It is seen through Figure 3.1 that, without the link from x_t to y_t and the link of Λ_q, the model is the traditional HMM. FA-HMM [28] can be obtained from Figure 3.1 by adding a direct link from z_t to y_t and deleting link from x_t to y_t.

Functions (3.1) and (3.2) are compact representation of x_t. Since z_t is semantic independent, it can be considered as source stimulus. By the state-dependent C_q working as a "vocal tract filter", semantic dependent x_t is generated. For this reason, we denote our model as generative factor-analyzed HMM.

3.4 Maximum likelihood parameter estimation of the GFA-HMM

Since the sequences $Q(T)$, $X(T)$, $Z(T)$, $M(T)$, and $J(T)$ are hidden, maximum likelihood estimation of the model parameter Θ may be carried out iteratively by EM algorithm [30]. In the EM algorithm, the auxiliary function is defined as the average of the joint log-likelihood calculated on current model parameter $\hat{\Theta}$ over posterior probabilities of the hidden sequences calculated from previous model parameter Θ, i.e.,

$$\begin{aligned} Q(\Theta, \hat{\Theta}) &= E_\Theta[\log \prod_{t=1}^{T} p(y_t, x_t, z_t, q_t, m_t, j_t | \hat{\Theta})] \\ &= E_\Theta[\log \prod_{t=1}^{T} \{\hat{a}_{q_{t-1}q_t} \hat{\pi}_{q_t m_t}\}^{\delta_{qm}(t)}] \\ &+ E_\Theta[\log \prod_{t=1}^{T} \{p(z_t) \hat{c}_{q_t j_t} p(x_t | z_t, q_t, j_t; \hat{\Theta})\}^{\delta_{qj}(t)}] \end{aligned}$$

$$+ \quad E_\Theta[\log \prod_{t=1}^{T} \{p(y_t|x_t, q_t, m_t; \hat{\Theta})\}^{\delta_{qm}(t)}] \tag{3.5}$$

where $\delta_{qmj}(t)$ is calculated given previous model parameter Θ. Since components in the right of (3.5) are functions of $\{\hat{a}_{qp}, \hat{\pi}_{qm}\}$, $\{\hat{C}_q, \hat{c}_{qj}, \hat{\xi}_{qj}, \hat{V}_{qj}\}$ and $\{\hat{\mu}_{qm}, \hat{\Sigma}_{qm}\}$, respectively, parameter estimation can be carried out on them separately. Solving the auxiliary function requires their posterior statistics.

3.4.1 Posterior statistics

The likelihood at state q, mixture component m in space Y, and mixture component j in space X at time t is given as,

$$p(y_t|q, m, j; \Theta) = N(y_t; \mu_{qm} + \Lambda_q \xi_{qj}; \Sigma_{qm} + \Lambda_q (V_{qj} + C_q C_q^T) \Lambda_q^T) \tag{3.6}$$

where superscript T denotes transpose. By marginalizing, likelihood $p(y_t|q, m)$ and $p(y_t|q)$ can be obtained.

Given previous model parameter Θ, the posterior probability of being in state q, mixture component m in space Y and mixture component j in space X at time t, $\gamma_{qmj}(t) = p(qmj|Y(T); \Theta)$, can be obtained by forward-backward algorithm with the above likelihood, i.e.,

$$\gamma_{qmj}(t) = \frac{\pi_{qm} c_{qj} p(y_t|q, m, j; \Theta)}{\sum_q \sum_m \sum_j \pi_{qm} c_{qj} p(y_t|q, m, j; \Theta)} \tag{3.7}$$

Marginalizing of the above posterior probability can give posterior probability in state q and mixture component m, $\gamma_{qm}(t)$, and the posterior probability in state q, $\gamma_q(t)$.

Regarding the posterior distribution of the continuous-valued hidden sequence $X(T)$, according to Bayes rule, it is given as,

$$p(x_t|y_t, q, m, j; \Theta) = \frac{p(y_t|x_t, q, m; \Theta) p(x_t|q, j; \Theta)}{p(y_t|q, m, j; \Theta)} \tag{3.8}$$

Since each component in the above function is Gaussian, the posterior distribution is Gaussian as well. It can be verified that the posterior distribution, $p(x_t|y_t, q, m, j; \Theta)$ is given as $N(x_t; \phi^x_{qmj}(t), \Psi^x_{qmj})$, where,

$$\begin{aligned} \phi^x_{qmj}(t) &= E_\Theta[x_t|y_t, q, m, j] \\ &= \Psi^x_{qmj}[(V_{qj} + C_q C_q^T)^{-1} \xi_{qj} + \Lambda_q^T \Sigma_{qm}^{-1} (y_t - \mu_{qm})] \qquad (3.9) \\ \Psi^x_{qmj} &= E_\Theta[\delta x_t \delta x_t^T|y_t, q, m, j] \\ &= [(V_{qj} + C_q C_q^T)^{-1} + \Lambda_q^T \Sigma_{qm}^{-1} \Lambda_q]^{-1} \qquad (3.10) \end{aligned}$$

Denote $E_\Theta[x_t x_t^T|y_t, q, m, j]$ as $\Phi^x_{qmj}(t)$. It is given as, $\Psi^x_{qmj} + \phi^x_{qmj}(t)\phi^x_{qmj}(t)^T$. Marginalizing of (3.9) and (3.10) can give posterior mean $\phi^x_{qm}(t)$ as $\sum_j \gamma_{qmj}(t)\phi^x_{qmj}(t)$, and posterior variance $\Psi^x_{qm}(t) = \sum_j \gamma_{qmj} \Psi^x_{qmj}(t)$. Similarly for $\phi^x_q(t)$ and $\Psi^x_q(t)$.

Since $p(z_t) \sim N(z_t; 0, I)$ and $p(x_t|z_t, q, m, j; \Theta)$ is Gaussian, the posterior distribution of z_t is also Gaussian $N(z_t; \phi^z_{qmj}(t), \Psi^z_{qmj}(t))$. Thus, only the first- and second-order statistics are needed.

This simplifies calculation of the posterior statistics for z_t, since the posterior mean vector of x_t, $\phi^x_{qmj}(t)$, can be taken as the "observation vector" of x_t to Function (3.1) and (3.2). Thus, in the same way as (3.9) and (3.10), the posterior statistics of z_t is

$$\begin{aligned}\phi^z_{qmj}(t) &= E_\Theta[z_t|x_t, y_t, q, m, j] \\ &= \Psi^z_{qmj}(t) C_q^T V_{qj}^{-1}(\phi^x_{qm}(t) - \xi_{qj}) \qquad (3.11)\\ \Psi^z_{qmj}(t) &= E_\Theta[\delta z_t \delta z_t^T|x_t, y_t, q, m, j] \\ &= [I + C_q^T V_{qj}^{-1} C_q]^{-1} \qquad (3.12)\end{aligned}$$

Denote $E_\Theta[z_t z_t^T|y_t, q, m, j]$ as $\Phi^z_{qmj}(t)$, which is given as $\Psi^z_{qmj}(t) + \phi^z_{qmj}(t)\phi^z_{qmj}(t)^T$.

3.4.2 Parameter estimation

EM algorithm for updating model parameters $\hat{\Theta}$ involves summation over the above posterior statistics. Re-estimation formulae are shown in appendix.

3.5 Experimental results

3.5.1 Experimental setup

The proposed GFA-HMM was compared with the traditional HMM in this paper by experiments on Aurora 2 database [31], a down-sampled TI-Digits database to 8kHz sampling rate.

Features for recognition were 39-dimensional MFCC plus C0 and its first- and second-order coefficients. One thousand utterances in clean training set of the database were used for training acoustic models. Testing was conducted with 1,000 clean utterances from the testing set of the database.

Acoustic models in all of the systems were trained by EM algorithm with six iterations. In all the acoustic models, state number was ten for digits and three for silence model.

Traditional HMM could only adjust its number of mixture components M^y_q. Accordingly, number of free parameters (NoFP) for a model was $S \times (2N) \times M^y_q$. The structure of GFA-HMM is more flexible. In this work, we varied the number of mixture component in space X, M^x_q and the dimension of space X, L. The dimension of space Z was set to one, and the number of mixture component M^y_q was set to one. Latent parameters $\{\Lambda_q, \xi_{qj}, V_{qj}, C_q\}$ are shared among states for each acoustic models. For a word model by GFA-HMM, NoFP was $S \times (2N) + (N+1) \times L + (2 \times L) \times M^x_q$.

Mixture components in space Y and X were incrementally obtained by mixture splitting [32]. In the training stage, V_{qj} and Σ_{qm} were floored to 1.0 and 0.001, respectively.

3.5.2 Results

Table 3.1 shows results by traditional HMM. The highest word accuracy (W.A.) was 88.93% by setting M^y_q to 4. In such a case, NoFP arrived to 2496 for a word model.

Performances by GFA-HMM were shown in Table 3.2. It is seen that GFA-HMM could achieve higher recognition accuracy over traditional HMM with the same amount of training data. For

Table 3.1: Number of free parameters (NoFP) for a word model and word accuracy (in %) in testing set by traditional HMM as a function of number of mixture component in space Y, M_q^y.

M_q^y	1	2	3	4
NoFP	624	1248	1872	2496
W.A.	88.05	85.04	87.96	88.93

example, word accuracy increased consistently by increasing mixture component in space X while keeping $L = 1$. The best W.A. was 90.93% by setting $L = 1$ and $M_q^x = 4$. Moreover, the NoFPs could be much lower than those by traditional HMMs. For example, in this situation, the NoFP was 672, whereas the NoFP was 2496 for traditional HMM to achieve its best performance.

Table 3.2: Number of free parameters (NoFP) for a word model and word Accuracy (in %) in testing sets by generative factor-analyzed HMM (GFA-HMM) as a function of number of mixture components in space X, M_q^x, and dimension in space X, L.

Dimension L	M_q^x	1	2	3	4
1	NoFP	666	668	670	672
	W.A.	88.80	89.73	90.30	90.93
2	NoFP	708	712	716	720
	W.A.	86.44	89.09	89.73	89.66

3.6 Conclusions and discussions

We propose to model speech observation vectors by a generative factor-analyzed HMM. Continuous-valued latent representation, which is also dependent on discrete hidden state sequence, of the observation vectors is introduced in this model. The model can achieve more compact represen-taion of observations compared to traditional HMM. Our experimental results on digits recognition show that the proposed model could achieve better performance than traditional HMM with the same amount of training data.

The model can be considered as a generalization of several models recently proposed for speech recognition. Without the latent representation of x_t by z_t, our model reduces to the factor analyzed HMM by [33]. The model reduces to the model by [28] without latent representation of x_t by z_t and a further assumption of single Gaussian $N(x_t; 0, I)$ in space X. Without the continuous latent representation of y_t, the model is the traditional HMM.

Further work will investigate various sharing schemes of the latent representation parameters and automatic decision of the dimension of the latent vector and number of mixture components.

.1 Approximation of the environment effects on speech features

The derivation is on normal Mel-Filter Cepstral Coefficients (MFCCs). In each filter bank in the linear frequency domain, channel distortion and additive noise have effects on speech power, which can be approximated by [5][6]

$$\sigma_y^2(j) = \sigma_s^2(j) \times \sigma_h^2(j) + \sigma_n^2(j) \tag{13}$$

where $\sigma_y^2(j)$, $\sigma_s^2(j)$, $\sigma_h^2(j)$ and $\sigma_n^2(j)$ each denotes observation power, speech power, channel distortion power, and additive noise power in filter bank j.

MFCC extracts the power or the amplitude for later stages by logarithm compression to log-spectral domain and discrete Cosine transform (DCT) to the cepstral domain. In case that the amplitude is used, following equations can be used to approximate the channel and noise effects on speech in the log-spectral domain.

$$\begin{aligned} & \log(\sqrt{\sigma_s^2(j) \times \sigma_h^2(j) + \sigma_n^2(j)}) \\ = \; & \frac{1}{2}[\log \sigma_s^2(j) + \log \sigma_h^2(j) + \log(1 + \frac{\sigma_n^2(j)}{\sigma_s^2(j) \times \sigma_h^2(j)})] \\ = \; & \frac{1}{2}\log \sigma_s^2(j) + \frac{1}{2}\log \sigma_h^2(j) + \\ & \frac{1}{2}\log(1 + \exp 2 \times \frac{1}{2}(\log \sigma_n^2(j) - \log \sigma_s^2(j) - \log \sigma_h^2(j))) \end{aligned} \tag{14}$$

Substitute $x_j^l = \frac{1}{2}\log \sigma_s^2(j)$, and similarly n_j^l, h_j^l, and y_j^l for $\frac{1}{2}\log \sigma_n^2(j)$, $\frac{1}{2}\log \sigma_h^2(j)$ and $\frac{1}{2}\log \sigma_y^2(j)$, respectively. The above function can be written as,

$$y_j^l = x_j^l + h_j^l + \frac{1}{2}\log(1 + \exp(2(n_j^l - x_j^l - h_j^l))) \tag{15}$$

In a similar way, the following function is obtained to approximate the environment effects in log-spectral domain on speech feature extracted from the power in the linear frequency domain.

$$y_j^l = x_j^l + h_j^l + \log(1 + \exp(n_j^l - x_j^l - h_j^l)) \tag{16}$$

.2 The objective function of the sequential Kullback proximal algorithm

The sequential Kullback proximal algorithm [19] is a sequential version of the Kullback proximal algorithm [34] for maximum-likelihood estimation. In the sequential Kullback proximal algorithm [19], the cost function for the iterative procedure is given as the log-likelihood function (shown in function (1.5)) regularized by a Kullback-Leibler divergence, i.e.,

$$\begin{aligned} & l_t(\hat{\lambda}_N(t)) - \beta_t I_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) = \\ & \quad l_t(\hat{\lambda}_N(t)) - I_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) - (\beta_t - 1) I_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) \end{aligned} \tag{17}$$

where $I_t(\lambda_N^\star(t); \hat{\lambda}_N(t))$ is the Kullback-Leibler divergence between the posterior distribution of state sequences given observation sequence $Y(t)$ and noise parameter sequence $(\Lambda_N(t-1), \lambda_N^\star(t))$ and that by the noise parameter sequence $\hat{\lambda}_N(t)$ till frame t, which is given as,

$$\begin{aligned}
&I_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) = \\
&\quad \sum_{S(t)} P(S(t)|Y(t), (\Lambda_N(t-1), \lambda_N^\star(t))) \\
&\quad \log \frac{p(S(t)|Y(t), (\Lambda_N(t-1), \lambda_N^\star(t)))}{P(S(t)|Y(t), (\Lambda_N(t), \hat{\lambda}_N(t)))} \\
&= l_t(\hat{\lambda}_N(t)) + \sum_{S(t)} P(S(t)|Y(t), (\Lambda_N(t-1), \lambda_N^\star(t))) \\
&\quad \log \frac{P(S(t)|Y(t), (\Lambda_N(t-1), \lambda_N^\star(t)))}{P(Y(t), S(t)|(\Lambda_N(t), \hat{\lambda}_N(t)))} \\
&= -Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) + l_t(\hat{\lambda}_N(t)) \\
&+ \sum_{S(t)} P(S(t)|Y(t), (\Lambda_N(t-1), \lambda_N^\star(t))) \\
&\quad \log P(S(t)|Y(t), (\Lambda_N(t-1), \lambda_N^\star(t)))
\end{aligned} \tag{18}$$

where

$$\begin{aligned}
&Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) = \\
&\quad \sum_{S(t)} p(S(t)|Y(t), (\Lambda_N(t-1), \lambda_N^\star(t))) \\
&\quad \log p(Y(t), S(t)|(\Lambda_N(t), \hat{\lambda}_N(t)))
\end{aligned} \tag{19}$$

Substituting above equation into (18), we obtain,

$$\begin{aligned}
&l_t(\hat{\lambda}_N(t)) - \beta_t I_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) \\
&= Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) - (\beta_t - 1) I_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) + Z
\end{aligned} \tag{20}$$

where Z is a function without relation to $\hat{\lambda}_N(t)$. We thus obtain (1.10) as the objective function for the sequential parameter estimation.

.3 Properties of the sequential Kullback proximal algorithm

.3.1 Sequential EM algorithm is a particular case of the sequential Kullback proximal algorithm

When $\beta_t = 1.0$, according to (20), the objective function $l_t(\hat{\lambda}_N(t)) - \beta_t I_t(\lambda_N^\star(t); \hat{\lambda}_N(t))$ to be maximized is equivalent to maximization of $Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t))$, which is the objective function to be maximized by sequential EM algorithm.

.3.2 Monotonic likelihood property

According to the objective function defined by the sequential Kullback proximal algorithm, it has,

$$\begin{aligned} & l_t(\hat{\lambda}_N(t)) - l_t(\hat{\lambda}_N(t-1)) \\ \geq\ & \beta_t I_t(\hat{\lambda}_N(t-1); \hat{\lambda}_N(t)) - \beta_t I_t(\hat{\lambda}_N(t-1); \hat{\lambda}_N(t-1)) \\ =\ & \beta_t I_t(\hat{\lambda}_N(t-1); \hat{\lambda}_N(t)) \end{aligned} \tag{21}$$

Since $\beta_t \in R^+$, $I_t(\hat{\lambda}_N(t-1); \hat{\lambda}_N(t-1)) = 0$ and $I_t(\hat{\lambda}_N(t-1); \hat{\lambda}_N(t)) \geq 0$, we prove that the sequential Kullback proximal algorithm can achieve the objective function (1.4).

.4 Sequential Kullback proximal algorithm can be viewed as a constrained maximum problem

When $\beta_t \geq 1.0$, since $I_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) \geq 0.0$, maximization of the function (20) corresponds to the constrained maximum problem, i.e.,

$$\hat{\lambda}_N(t) = \arg\max_{\lambda_N(t)} Q_t(\lambda_N^\star; \lambda_N(t)) \tag{22}$$

subject to

$$I_t(\lambda_N^\star(t); \lambda_N(t)) \leq C \tag{23}$$

where $C \in R^+$. The larger the β_t, the stronger the constraint in (23). This means that estimate by the sequential Kullback proximal algorithm will be "pulled close to" $\lambda_N^\star(t)$.

Another situation is that $0 < \beta_t < 1.0$. In such a case, $\beta_t - 1.0 \leq 0.0$. Define $\xi_t = 1.0 - \beta_t$. Function (20) can be written as,

$$I_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) + \frac{1}{\xi_t} Q_t(\lambda_N^\star(t); \hat{\lambda}_N(t)) + \frac{Z}{\xi_t} \tag{24}$$

Maximization of the function thus can be seen as a constrained optimization given by,

$$\hat{\lambda}_N(t) = \arg\max_{\lambda_N(t)} I_t(\lambda_N^\star(t); \lambda_N(t)) \tag{25}$$

subject to

$$-Q_t(\lambda_N^\star(t); \lambda_N(t)) \leq C \tag{26}$$

where $C \in R^+$. Since $\xi_t \to 0.0$ when $\beta_t \to 1.0$, constraint in (26) thus will be very tight in this situation. On the contrary, when $\beta_t \to 0.0$, the constraint will be very loose. As a result, the estimate by sequential Kullback proximal algorithm will be far away from that provided by sequential EM algorithm.

.5 Derivation of the Sequential Kullback proximal algorithm applied for environment parameter estimation

The first- and second-order differential of the K-L divergence of (18) are given respectively as,

$$\frac{\partial I_t(\lambda_N^\star(t);\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)} = \tag{27}$$
$$-\frac{\partial Q_t(\lambda_N^\star(t);\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)} + \frac{\partial l_t(\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)}$$
$$\frac{\partial^2 I_t(\lambda_N^\star(t);\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)^2} = \tag{28}$$
$$-\frac{\partial^2 Q_t(\lambda_N^\star(t);\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)^2} + \frac{\partial^2 l_t(\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)^2}$$

Assume that $\frac{\partial I_t(\lambda_N^\star(t);\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)}|_{\hat{\lambda}_N(t)=\lambda_N^\star(t)} = 0$ has been achieved, and it thus holds

$$\frac{\partial l_t(\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)}|_{\hat{\lambda}_N(t)=\lambda_N^\star(t)} = \frac{\partial Q_t(\lambda_N^\star(t);\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)}|_{\hat{\lambda}_N(t)=\lambda_N^\star(t)} \tag{29}$$

With the second-order Taylor series expansion of the objective function (17) at $\lambda_N^\star(t)$, the updating of environment parameter is given as,

$$\hat{\lambda}_N(t) \longleftarrow \hat{\lambda}_N(t-1) - \frac{\frac{\partial(l_t(\hat{\lambda}_N(t))-\beta_t I_t(\lambda_N^\star(t);\hat{\lambda}_N(t)))}{\partial \hat{\lambda}_N(t)}}{\frac{\partial^2(l_t(\hat{\lambda}_N(t))-\beta_t I_t(\lambda_N^\star(t);\hat{\lambda}_N(t)))}{\partial \hat{\lambda}_N(t)^2}}|_{\hat{\lambda}_N(t)=\hat{\lambda}_N(t-1)} \tag{30}$$

By (28) and (29), the updating is given as,

$$\hat{\lambda}_N(t) \longleftarrow \hat{\lambda}_N(t-1) - \tag{31}$$
$$\frac{\frac{\partial Q_t(\lambda_N^\star(t);\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)}}{\beta_t \frac{\partial^2 Q_t(\lambda_N^\star(t);\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)^2} + (1-\beta_t)\frac{\partial^2 l_t(\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)^2}}|_{\hat{\lambda}_N(t)=\hat{\lambda}_N(t-1)}$$

The derivation of the updating formulae for the auxiliary function $Q_t(\hat{\lambda}_N(t-1);\hat{\lambda}_N(t))$ can be seen in [20]. We briefly describe the derivation in this paper. Since

$$\begin{aligned} Q_t(\hat{\lambda}_N(t-1);\hat{\lambda}_N(t)) = & \\ & \sum_{S(t)} P(S(t)|Y(t),\Lambda_X,(\Lambda_N(t-1),\hat{\lambda}_N(t-1)))\cdot \\ & \log[P(S(t-1)|Y(t-1),\Lambda_X,\Lambda_N(t-1))\cdot \\ & a_{s(t-1)s(t)}b_{s(t)}(y(t))] \\ = & \sum_{S(t-1)} P(S(t-1)|Y(t-1),\Lambda_X,\Lambda_N(t-1)) \end{aligned}$$

$$\begin{aligned} & \log b_{s(t-1)}(y(t-1)) \\ + & \sum_{s(t)} P(s(t)|Y(t), \Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t-1))) \\ & \log b_{s(t)}(y(t)) + Z \end{aligned}$$

where Z is not a function of $\hat{\lambda}_N(t)$. Denote $Q_{t-1}(\hat{\lambda}_N(t-2); \hat{\lambda}_N(t)) = \sum_{S(t-1)} P(S(t-1)|Y(t-1), \Lambda_X, \Lambda_N(t-1)) \log b_{s(t-1)}(y(t-1))$. Assume that $\hat{\lambda}_N(t-1)$ has made $\frac{\partial Q_{t-1}(\hat{\lambda}_N(t-2); \hat{\lambda}_N(t))}{\partial \hat{\lambda}_N}|_{\hat{\lambda}_N(t)=\hat{\lambda}_N(t-1)} = 0$. We thus obtain the first- and second-order derivative of the auxiliary function with respect to the noise parameter, which are shown in (1.15) and (1.16), respectively.

The remaining is how to calculate $\frac{\partial^2 l_t(\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)^2}$. Define forward accumulated likelihood at state i and mixture m as $\alpha_t(i, m; \hat{\lambda}_N(t)) = P(Y(t), s(t) = i, k(t) = m|\Lambda_X, (\Lambda_N(t-1), \hat{\lambda}_N(t)))$, and accordingly the forward accumulated likelihood at state i, $\alpha_t(i; \hat{\lambda}_N(t)) = \sum_m \alpha_t(i, m; \hat{\lambda}_N(t))$. They have relations shown below as

$$\alpha_t(i, m; \hat{\lambda}_N(t)) = \sum_{l=1}^{\Upsilon} \alpha_{t-1}(l; \hat{\lambda}_N(t-1)) a_{li} w_{im} b_{im}(y(t)) \tag{32}$$

Since $l_t(\hat{\lambda}_N(t)) = \log \sum_{im} \alpha_t(i, m; \hat{\lambda}_N(t))$, it has

$$\begin{aligned} & \frac{\partial l_t(\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)}|_{\hat{\lambda}_N(t)=\hat{\lambda}_N(t-1)} \\ = & \frac{\partial \log \sum_{i=1}^{\Upsilon} \sum_{m=1}^{M} \alpha_t(i, m; \hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)}|_{\hat{\lambda}_N(t)=\hat{\lambda}_N(t-1)} \\ = & \frac{\sum_{i=1}^{\Upsilon} \sum_{m=1}^{M} \frac{\partial \alpha_t(i,m;\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)}}{\sum_{j=1}^{\Upsilon} \sum_{m=1}^{M} \alpha_t(j, m; \hat{\lambda}_N(t))}|_{\hat{\lambda}_N(t)=\hat{\lambda}_N(t-1)} \end{aligned} \tag{33}$$

By (1.13) and (32), it has,

$$\begin{aligned} & \frac{\partial \alpha_t(i, m; \hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)} = \\ & \alpha_t(i, m; \hat{\lambda}_N(t)) \frac{\partial \log b_{im}(y_t)}{\partial \hat{\lambda}_N(t)} \end{aligned} \tag{34}$$

Substituting the above equation into (33), we have,

$$\begin{aligned} & \frac{\partial l_t(\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)}|_{\hat{\lambda}_N(t)=\hat{\lambda}_N(t-1)} = \\ & \sum_{i=1}^{\Upsilon} \sum_{m=1}^{M} \gamma_t(i, m; \hat{\lambda}_N(t)) \frac{\partial \log b_{im}(y_t)}{\partial \hat{\lambda}_N(t)}|_{\hat{\lambda}_N(t)=\hat{\lambda}_N(t-1)} \end{aligned}$$

where $\gamma_t(i, m; \hat{\lambda}_N(t)) = \frac{\alpha_t(i,m;\hat{\lambda}_N(t))}{\sum_{lm} \alpha_t(l,m;\hat{\lambda}_N(t))}$, representing the posterior probability at state i and mixture m given observation sequence $Y(t)$ and noise parameter sequence $(\Lambda_N(t-1), \hat{\lambda}_N(t))$. We

thus obtain,

$$\frac{\partial^2 l_t(\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)^2} = \sum_{i=1}^{\Upsilon}\sum_{m=1}^{M} \frac{\partial \gamma_t(i,m;\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)} \frac{\partial \log b_{im}(y_t)}{\partial \hat{\lambda}_N(t)} + \sum_{i=1}^{\Upsilon}\sum_{m=1}^{M} \gamma_t(i,m;\hat{\lambda}_N(t)) \frac{\partial^2 \log b_{im}(y_t)}{\partial \hat{\lambda}_N^2(t)} \tag{35}$$

Notice that $\gamma_t(i,m;\hat{\lambda}_N(t)) = \frac{\alpha_t(i,m;\hat{\lambda}_N(t))}{\sum_{i=1}^{\Upsilon}\sum_{k=1}^{M}\alpha_t(i,k;\hat{\lambda}_N(t))}$, and refereeing to (34), we have,

$$\frac{\partial \gamma_t(i,m;\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)} = \gamma_t(i,m;\hat{\lambda}_N(t))[\frac{\partial \log b_{im}(y_t)}{\partial \hat{\lambda}_N(t)} - \sum_{l=1}^{\Upsilon}\sum_{k=1}^{M} \gamma_t(l,k;\hat{\lambda}_N(t)) \frac{\partial \log b_{lk}(y_t)}{\partial \hat{\lambda}_N(t)}] \tag{36}$$

Substituting above equation into (35), we have,

$$\frac{\partial^2 l_t(\hat{\lambda}_N(t))}{\partial \hat{\lambda}_N(t)^2} = \sum_{i=1}^{\Upsilon}\sum_{m=1}^{M} \gamma_t(i,m;\hat{\lambda}_N(t))[(\frac{\partial \log b_{im}(y_t)}{\partial \hat{\lambda}_N(t)})^2 + \frac{\partial^2 \log b_{im}(y_t)}{\partial \hat{\lambda}_N(t)^2}] - (\sum_{i=1}^{\Upsilon}\sum_{m=1}^{M} \gamma_t(i,m;\hat{\lambda}_N(t)) \frac{\partial \log b_{im}(y_t)}{\partial \hat{\lambda}_N(t)})^2 \tag{37}$$

.6 Gaussian PDFs

Since $p(z_t) \sim N(0,I)$ and $p(x_t|q,z_t,j)$ is Gaussian, by marginalizing of z_t, the density of x_t in state q and mixture component j is $N(x_t;\xi_{qj},V_{qj}+C_qC_q^T)$.

Since the density of y_t given x_t in state q and observation component m is given by

$$p(y_t|x_t,q,m) = \frac{1}{\sqrt{(2\pi)^N|\Sigma_{qm}|}} \exp(-\frac{1}{2}(y_t-\mu_{qm}-\Lambda_q x_t)^T\Sigma_{qm}^{-1}(y_t-\mu_{qm}-\Lambda_q x_t)) \tag{38}$$

, by marginalizing of x_t, we obtain the density function for y_t at state q, mixture component m in space Y and mixture component j in state X as that in (3.6).

.7 EM algorithm for GFA-HMM

By setting the first order derivative of the auxiliary function with respect to $\hat{\mu}_{qm}$, $\hat{\Sigma}_{qm}$ and $\hat{\Lambda}_q$ to zero, we obtained the following re-estimation formulae.

$$\sum_t \gamma_{qm}(t)\Sigma_{qm}^{-1}\hat{\Lambda}_q\Phi_{qm}^x(t)$$

$$= \sum_t \gamma_{qm}(t)\Sigma_{qm}^{-1}(y_t - \mu_{qm})\phi_{qm}^x(t)^T \tag{39}$$

$$\hat{\mu}_{qm} = \frac{1}{\sum_t \gamma_{qm}(t)} \sum_t \gamma_{qm}(t)[y_t - \Lambda_q \phi_{qm}^x(t)] \tag{40}$$

$$\hat{\Sigma}_{qm} = diag\frac{1}{\sum_t \gamma_{qm}(t)} \sum_t \gamma_{qm}(t)[(y_t - \mu_{qm})(y_t - \mu_{qm})^T \tag{41}$$

$$- \Lambda_q \phi_{qm}^x(t)(y_t - \mu_{qm})^T - (y_t - \mu_{qm})\phi_{qm}^x(t)^T \Lambda_q^T + \Lambda_q \Phi_{qm}^x(t)\Lambda_q^T]$$

The loading matrix $\hat{\Lambda}_q$ has to be estimated row by row [33]. The nth row vector $\hat{\lambda}_{qn}$ of the new loading matrix $\hat{\Lambda}_q$ can be written as,

$$\hat{\lambda}_{qn} = \mathbf{k}_{qn}^T G_{qn}^{-1} \tag{42}$$

where the L by L matrices G_{qn} and L dimensional vector are defined as follows,

$$G_{qn} = \sum_t \gamma_{qm}(t)\frac{1}{\sigma_{qmn}^2}\Phi_{qm}^x(t) \tag{43}$$

$$\mathbf{k}_{qn} = \sum_t \gamma_{qm}(t)\frac{1}{\sigma_{qmn}^2}(y_{tn} - \mu_{qmn})\phi_{qm}^x(t)^T \tag{44}$$

where y_{tn} and μ_{qmn} are, respectively, the nth element of the current observation and the observation noise mean vectors.

Taking $\phi_{qmj}^x(t)$ as the "observation vector" in space X, re-estimation formulae for $\{\hat{C}_q, \hat{\xi}_{qj}, \hat{V}_{qj}\}$ are similarly derived as above. The formulae are given as,

$$\hat{C}_q[\sum_t \sum_j \gamma_{qj}(t)\Phi_{qj}^z(t)] = \sum_t \sum_j [\phi_{qj}^x(t) - \xi_{qj}]\phi_{qj}^z(t)^T \tag{45}$$

$$\hat{\xi}_{qj} = \frac{1}{\sum_t \gamma_{qj}(t)} \sum_t \gamma_{qj}(t)(\phi_{qj}^x(t) - C_q \phi_{qj}^z(t)) \tag{46}$$

$$\hat{V}_{qj} = diag\frac{1}{\sum_t \gamma_{qj}(t)} \sum_t \gamma_{qj}(t)\{[\phi_{qj}^x(t) - \xi_{qj} - C_q\phi_{qj}^z(t)] [\phi_{qj}^x(t) - \xi_{qj} - C_q\phi_{qj}^z(t)]^T + C_q \Psi_{qj}^z(t) C_q^T\} \tag{47}$$

The loading matrix $\hat{C}_q$ should be calculated row-by-row similarly as that for $\hat{\Lambda}_q$ by (39).

Maximization auxiliary function (3.5) with respect to mixture component weights arrives at the following updating formulae,

$$\hat{\pi}_{qm} = \frac{\sum_t \sum_j \gamma_{qmj}(t)}{\sum_t \sum_m \sum_j \gamma_{qmj}(t)} \tag{48}$$

$$\hat{c}_{qj} = \frac{\sum_t \sum_m \gamma_{qmj}(t)}{\sum_t \sum_m \sum_j \gamma_{qmj}(t)} \tag{49}$$

Bibliography

[1] A. Sankar and C-H Lee, "A maximum–likelihood approach to stochastic matching for robust speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 4, no. 3, pp. 190–201, 1996.

[2] Y. Ephraim and D. Malah, "Speech enhancement using a minimum mean square error short-time spectral amplitude estimator," *IEEE trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-32, no. 6, pp. 1109–1121, December 1984.

[3] H. Hermansky, "Perceptual linear predictive (plp) analysis for speech," *J.Acoust.Soc.Amer.*, vol. 87, no. 4, pp. 1738–1752, 1990.

[4] B. A. Hanson and T. H. Applebaum, "Robust speaker–independent word recognition using static, dynamic and acceleration features: Experiments with lombard and noisy speech," in *ICASSP*, 1990, pp. 857–860.

[5] M.J.F.Gales and S.J.Young, "Robust speech recognition in additive and convolutional noise using parallel model combination," *Computer Speech and Language*, vol. 9, pp. 289–307, 1995.

[6] A. Acero, *Acoustical and Environmental Robustness in Automatic Speech Recognition*, Ph.D. thesis, Carnegie Mellon University, September 1990.

[7] P.J. Moreno, B. Raj, and R. M. Stern, "A vector taylor series approach for environment-independent speech recognition," in *ICASSP*, 1996, vol. 2, pp. 733–736.

[8] Y. Zhao, "Frequency-domain maximum likelihood estimation for automatic speech recognition in additive and convolutive noises," *IEEE Trans. on Speech and Audio Processing*, vol. 8, no. 3, pp. 255–266, May 2000.

[9] S. V. Vaseghi and B. P. Milner, "Noise compensation methods for hidden markov model speech recognition in adverse environments," *IEEE. Trans. on Speech and Audio Processing*, vol. 5, no. 1, pp. 11–21, January 1997.

[10] P. C. Woodland, M. J. F. Gales, and D. Pye, "Improving environmental robustness in large vocabulary speech recognition," in *ICASSP*, 1996, pp. 65–68.

[11] C-H. Lee and J-L. Gauvain, "Bayesian adaptive learning and map estimation of hmm," in *Automatic Speech and Speaker Recognition*, pp. 83–107. Kluwer Academic Publishers, 1996.

[12] A. Varga and R.K. Moore, "Hidden markov model decomposition of speech and noise," in *ICASSP*, 1990, pp. 845–848.

[13] T. Takiguchi, S. Nakamura, and K. Shikano, "Speech recognition for a distant moving speaker based on hmm composition and separation," in *ICASSP*, 2000, pp. 1403–1406.

[14] N.S. Kim, "Nonstationary environment compensation based on sequential estimation," *IEEE Signal Processing Letters*, vol. 5, no. 3, pp. 57–59, March 1998.

[15] Y. Zhao, S. Wang, and K-C. Yen, "Recursive estimation of time-varying environments for robust speech recognition," in *ICASSP*, 2001, pp. 225–228.

[16] 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," in *EUROSPEECH*, 2001, pp. 901–904.

[17] K. Yao and S. Nakamura, "Sequential noise compensation by sequential monte carlo method," in *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

[18] K. Yao, K. Paliwal, and S. Nakamura, "Noise adaptive speech recognition in time-varying noise based on sequential kullback proximal algorithm," in *ICASSP*, 2002, pp. 189–192.

[19] K. Yao, K. K. Paliwal, and S. Nakamura, "Sequential noise compensation by a sequential kullback proximal algorithm," in *EUROSPEECH*, 2001, pp. 1139–1142.

[20] V. Krishnamurthy and J. B. Moore, "On-line estimation of hidden markov model parameters based on the kullback-leibler information measure," *IEEE Trans. on Signal Processing*, vol. 41, no. 8, pp. 2557–2573, August 1993.

[21] K. Yao, D.-L. Zhu, and S. Nakamura, "Evaluation of a noise adaptive speech recognition system on aurora 3 database," in *ICSLP*, 2002.

[22] K. Yao, B. E. Shi, S. Nakamura, and Z. Cao, "Residual noise compensation by a sequential em algorithm for robust speech recognition in nonstationary noise," in *ICSLP*, 2000, vol. 1, pp. 770–773.

[23] J. S. Liu and R. Chen, "Sequential monte carlo methods for dynamic systems," *J. Am. Stat. Assoc*, vol. 93, pp. 1032–1044, 1998.

[24] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[25] I.T. Jolliffe, *Principle Component Analysis*, Springer-Verlag, 1986.

[26] D. Rubin and D. Thayer, "EM algorithms for ML factor analysis," *Psychometrika*, vol. 47, no. 1, pp. 69–76, 1982.

[27] M. E. Tipping and C. M. Bishop, "Mixtures of probabilistic principal component analysis," Tech. Rep. NCRG-97-003, Aston University, 1997.

[28] L. K. Saul and M. G. Rahim, "Maximum likelihood and minimum classification error factor analysis for automatic speech recognition," *IEEE Trans. on SAP*, vol. 8, no. 2, pp. 115 – 125, March 2000.

[29] T.W. Lee, *Independent component analysis, theory and applications*, Kluwer Academic Publishers, 1998.

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Stat. Soc.*, vol. 3g, pp. 1–38, 1977.

[31] D. Pearce, "Aurora project: Experimental framework for the performance evaluation of distributed speech recognition front-ends," in *ISCA ITRW ASR2000*, Sep. 2000.

[32] Steve Young, *The HTK BOOK*, Cambridge University, 2.1 edition, 1997.

[33] A-V.I. Rosti and M.J.F. Gales, "Factor analysed Hidden Markov models," in *ICASSP*, 2002.

[34] S. Chréten and A. O. Hero III, "Kullback proximal point algorithms for maximum–likelihood estimation," *IEEE. Trans. on IT*, vol. 46, no. 5, pp. 1800–1810, August 2000.