Enhancements to Transformation-Based Speaker Adaptation: Principal Component and Inter-Class Maximum Likelihood Linear Regression

Sam-Joo Doh

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, Pennsylvania

July 2000

To my family:

Myung-Sook, Hansol, and Unsol

Abstract

In this thesis we improve speech recognition accuracy by obtaining better estimation of linear transformation functions with a small amount of adaptation data in speaker adaptation. The major contributions of this thesis are the developments of two new adaptation algorithms to improve maximum likelihood linear regression. The first one is called *principal component MLLR* (*PC-MLLR*), and it reduces the variance of the estimate of the MLLR matrix using principal component analysis. The second one is called *inter-class MLLR*, and it utilizes relationships among different transformation functions to achieve more reliable estimates of MLLR parameters across multiple classes.

The main idea of PC-MLLR is that if we estimate the MLLR matrix in the eigendomain, the variances of the components of the estimates are inversely proportional to their eigenvalues. Therefore we can select more reliable components to reduce the variances of the resulting estimates and to improve speech recognition accuracy. PC-MLLR eliminates highly variable components and chooses the principal components corresponding to the largest eigenvalues. If all the component are used, PC-MLLR becomes the same as conventional MLLR. Choosing fewer principal components increases the bias of the estimates which can reduce recognition accuracy. To compensate for this problem, we developed *weighted principal component MLLR (WPC-MLLR)*. Instead of eliminating some of the components, all the components in WPC-MLLR are used after applying weights that minimize the mean square error. The component corresponding to a larger eigenvalue has a larger weight than the component corresponding to a smaller eigenvalue.

As more adaptation data become available, the benefits from these methods may become smaller because the estimates using conventional MLLR become more reliable. However, if we have a larger amount of adaptation data, we would use a larger number of MLLR classes, making the amount of adaptation data for each MLLR class smaller. Therefore PC-MLLR and WPC-MLLR can be useful.

It is useful to consider relationships among different parameters when a small amount of adaptation data is available. Most previous studies use correlations or regression models among the recognition model parameters in a Bayesian framework. In this thesis, inter-class MLLR utilizes relationships among different transformation functions. Inter-class transformations given by linear regressions are used to modify the baseline mean vectors in the neighboring classes so that the neighboring classes can contribute to the estimates the MLLR parameters of the target class. If the inter-class transformations are identity functions, inter-class MLLR becomes the same as conventional single-class MLLR. This idea also can be applied to other types of transformation-based adaptation and general parameter estimation problems.

In inter-class MLLR several neighboring classes are considered for each target class. In this procedure, some neighboring classes may be closer to the target class than other neighboring classes. Therefore we apply different weights to the neighboring classes to accommodate their different contributions to the target class. Considering the weighted least squares estimation, the weight for each neighboring class is inversely proportional to the variance of the error that is produced in estimating the target parameters. Therefore a neighboring class with a smaller variance has a larger weight.

As more adaptation data become available, fewer neighboring classes can be used for a target class. For a large amount of adaptation data, we may not use any neighboring class at all. In this case inter-class adaptation becomes the same as conventional multi-class MLLR. To limit the number of neighboring classes, the neighboring classes are sorted for each target class according to their variances of the errors. Adaptation data from the closest neighboring class are used first, then from the next closest neighboring class until sufficient data are used.

In our experiments PC-MLLR improves recognition accuracy over conventional MLLR, and WPC-MLLR provided further improvement. Inter-class MLLR also provides improvements in recognition accuracy over conventional MLLR. Inter-class MLLR is better than WPC-MLLR in supervised adaptation. In unsupervised adaptation, however, inter-class MLLR is worse with a very small amount of test data, and becomes better with more test data. We believe WPC-MLLR is more effective in a highly unreliable case like unsupervised adaptation with a very small amount of test data. Both methods do not provide improvement over conventional MLLR on a complex task like the DARPA broadcast news task. These methods seems to be more effective when there is a larger mismatch between training and test conditions in a small task.

We also tried to combine WPC-MLLR and inter-class MLLR by first modifying the baseline mean vectors using inter-class transformations, and then using WPC-MLLR. However, this combination did not provide further improvement in accuracy. We believe that the benefits of WPC-MLLR become smaller after inter-class transformations such as the eigenvalues become more compact. The developed methods are aimed for the case when only a small amount of adaptation data is available. For a larger amount of adaptation data, we can combine the adapted means from these methods with the sample means to get further improvement in recognition accuracy.

Acknowledgement

First I would like to thank my thesis advisor, Richard Stern, for his guide and support. He has helped me with patience and insights during my CMU years. His dedication to work has been the greatest influence on this thesis. I am also grateful to the other members of my thesis committee, Prof. Tsuhan Chen, Prof. Amro El-Jaroudi, and Dr. Mosur Ravishankar, for their comments and feedback. They always welcome my questions and gives me a lot of insights.

I would also like to thank the current and former members of the CMU Speech Group for making a great research environment. I think I am really lucky to work with these people. Eric Thayer and Mosur Ravishankar helped me understand the SPHINX system. My colleagues in the Robust Speech Recognition Group inspire me with friendship. Rita and Bhiksha gave me several ideas for this thesis. Evandro, Juan and Mike who have shared the office were never tired of my interruptions. Pedro, Matt, and Uday help me a lot in my early days at CMU. Jon also gave me a great feedback on my work.

I would also like to thank my former colleagues in Korea Telecom Research and Development Group. Thanks to all friends in Pittsburgh, particularly those who have worked for Korean Student Association of Pittsburgh.

I am extremely indebted to my parents for all that they have thought me. I hope that my mom in heaven is also happy with my work. I would also like to thank my parents-in-law for their support. Finally, I would like to thank my wife, Myung-Sook, and my children, Hansol and Unsol. Myung-Sook always believes in me and gives endless love. Hansol and Unsol are the source of joy in my life. They patiently supported me throughout these long years. This thesis is dedicated to my family.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

Speech recognition systems have achieved increasingly good recognition accuracy in recent years. Nevertheless, the accuracy of speech recognition systems is still often severely degraded when there are mismatches between testing and training conditions, or the recognition model does not represent the test data properly. Adaptation is a process that reduces the difference between training and testing conditions, usually by using a small amount of adaptation (or training) data. If we have a large amount of data from a test condition, we can simply use a regular re-training technique. In practical applications, however, it is frequently difficult to obtain a large amount of training data from a new test condition such as a different speaker, microphone, telephone channel, or noisy environment. That is why we need adaptation.

The general goals of adaptation are as follows:

- Fast adaptation using small amount of adaptation data
- Good asymtotic property, *i.e.* more data, better accuracy
- Simple computation
- Good unsupervised adaptation

We seek to achieve both fast and asymptotic properties in adaptation algorithms. Fast adaptation means that good recognition accuracy can be achieved with small amounts of adaptation data. It is “fast” in sense that it does not have to wait long to get more adaptation data. In order to obtain fast adaptation, we usually need to either reduce the number of the parameters to be estimated, or perform smoothing of the parameters. If we assume that there is a function that transforms the baseline recognition models to the adapted recognition models, we can estimate the parameters of this transformation function rather than the recognition models. In this case, the tranformation function usually has much fewer parameters than the recognition models. Therefore we can achieve reliable estimation of the transformaiton function with a small amount of adaptation data. This method is called *transformation-based adaptation*.

This method causes some detailed information of the individual model parameters to be lost, which generally impairs recognition accuracy. For example, when we adapt a speaker-independent (SI) speech recognition system for a new speaker, the speaker-adaptive (SA) system may not achieve the level of accuracy obtained with the speaker-dependent (SD) system. Therefore, as we have more adaptation data, we would like to achieve better recognition accuracy asymtotic to a condition-specific system (*e.g.* SD system). If adaptation requires too much computation, it may not be very useful. Therefore we would like to develop a computationally simple algorithm.

Sometimes adaptation data with correct transcripts are available (supervised adaptation), sometimes not (unsupervised adaptation). In the case of unsupervised adaptation we adapt the system without any labelled adaptation data. Unsupervised adaptation algorithms typically adapt on the basis of the (errorful) decoding of the incoming speech produced by the baseline recognition system instead of completely correct transcripts. An obvious goal of unsupervised adaptation is to reduce the gap in recognition accuracy between supervised and unsupervised adaptation to the extent possible.

Most current speech recognition systems use hidden Markov models (HMMs) to represent the statistical characteristic of speech. The most important parameters in HMM are output distributions, which are usually represented by Gaussian mixture densities. In this thesis we focus on the adaptation of recognition model parameters, especially Gaussian mean vectors which are essential in the Gaussian mixture densities. We focus on speaker adaptation even though the developed techniques can be applied to adaptation for other sources of mismatches.

Goal and outline

The goal of this thesis is to improve recognition accuracy by obtaining better estimation of the transformation function with the limited amount of adaptation data. We present two new adaptation algorithms to improve maximum likelihood linear regression (MLLR) adaptation. One is called *principal component MLLR* (*PC-MLLR*) which reduces the variance of the estimate of the MLLR matrix using principal component analysis. The other is called *inter-class MLLR* which utilizes relationships among different transformation functions to achieve more reliable estimates of MLLR parameters across multiple classes.

The outline of the thesis is as follows. In Chapter 2 we present a brief review of automatic speech recognition and speaker adaptation algorithms. We especially describe two major adaptation approaches, Bayesian adaptation and transformation-based adaptation. We also describe the extensions of those approaches to overcome their disadvantages. We then describe the experimental system to be used for the evaluation of the algorithms to be presented. In Chapter 3 we review classical linear regression and MLLR, and describe that we can consider MLLR as weighted least square estimation. We present principal component MLLR in Chapter 4, and inter-class MLLR in Chapter 5 along with their experimental results. Finally we summarize our work and present ideas for future work in Chapter 6.

Chapter 2
Background Information

2.1 Introduction

In this chapter we will present the background information to the thesis. We will first briefly explain automatic speech recognition and hidden Markov models (HMM), then review speaker adaptation techniques. The two most common approaches used to modify the parameters characterizing speech in recognition system are Bayesian adaptation, which uses a maximum *a posteriori* probability (MAP) criterion, and transformation-based adaptation, which usually uses a maximum-likelihood (ML) criterion.

In Bayesian adaptation, the parameters to be estimated are assumed to be random variables. Each parameter is adapted if there are adaptation data which are associated with that parameter. If only a small amount of adaptation data is available, there will not be sufficient data available to meaningfully adapt many of the parameters, which results in little change in recognition accuracy. In contrast, if there are enough adaptation data to provide information about all of the parameters to be adapted, the adapted models will converge to speaker dependent (SD) models.

Transformation-based adaptation procedures usually use a simple linear transformation of the recognition model parameters to reduce the number of the parameters to be estimated. They can improve in recognition accuracy when only a small amount of adaptation data is present. Nevertheless, a simple transformation cannot capture all the details about the information contained in the adaptation data, which has an adverse impact on its asymptotic properties.

We will also review the some of the techniques developed to overcome the disadvantages of Bayesian and transformation-based approaches. Finally we will describe the SPHINX-3 system and the test corpora used for experiments to evaluate the algorithms to be developed in this thesis.

2.2 Automatic Speech Recognition

We consider automatic speech recognition as a pattern matching problem [47]. As shown in Fig. 2.1, a speech recognition system receives an input speech waveform and converts it to a set of feature vectors so that it can better represent speech characteristics, then compares it with reference patterns to find the closest pattern. If the input speech in generated by a different speaker or under different environmental condition from the reference patterns, the system may not find the correct pattern.

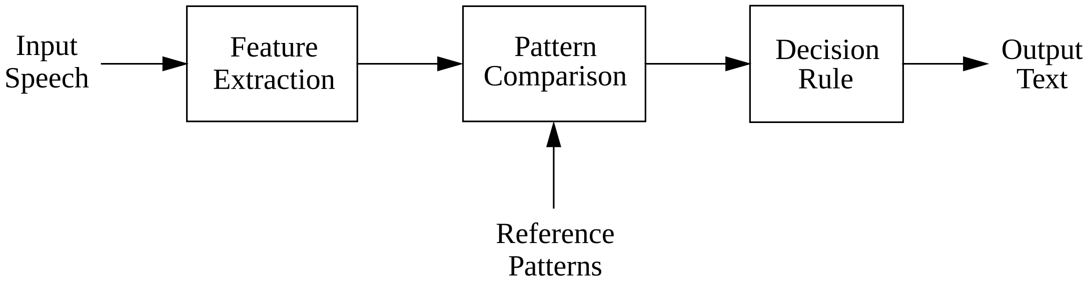


Figure 2.1 A block diagram of speech recognition system viewing as a pattern matching problem

2.2.1 Acoustic Modeling

In automatic speech recognition we try to find a best matching reference pattern for the input speech (or test pattern). We can think of speech as a sequence of phonemes (or sub-phonemes) which have their own distinctive patterns (inter-phoneme differences). However the exact shape of speech signal changes even for a same speaker repeating a same word (intra-phoneme variations). The duration of each phoneme also changes every time it is spoken (temporal variation).

Dynamic Time Warping (*DTW*) has been developed to overcome the temporal variation [50]. It stores a set of reference patterns, and aligns test and reference patterns so that they their smallest "distance" between them according to a predetermined metric. After comparing the test word with all the reference patterns, the reference pattern with the minimum distance is identified as the recognized output. Because

DTW compares a test pattern with one reference pattern at a time, it is difficult to incorporate with many reference patterns which may (for example) represent the acoustic variations of a same phoneme (or word) over different reference speakers or environmental conditions.

Hidden Markov Models (*HMM*s) can deal with this problem. The HMM uses a stochastic model instead of a set of reference patterns. A HMM consists of a number of states which have their own transition and output probabilities as illustrated in Fig. 2.2. We collect a lot of reference patterns for each phoneme (or recognition unit), and build the probabilistic distributions from them. Because each phoneme is characterized by probabilistic distributions, it can cover the intra-phoneme variations.

An input feature vector $\mathbf{o}_t$ at time frame t is associated with state i with a probability which can be calculated from the transition and the output probabilities. The next input feature vector at time frame t+1 may be associated with the same state i again (with self-transition probability p_{ii}) or state j (with transition probability p_{ij}, $i \neq j$). In this way a sequence of input feature vectors is associated with the states. Several different sequences of states can correspond to a particular input sequence as shown in Fig. 2.3. If p_{ii} is large, the next state will more likely be the same state i, so state i will be associated with more input feature vectors representing a phonetic segment that is longer in time. If p_{ii} is small, the input feature vectors for that segment tend to be shorter in time. Because a transition is given by a probability, a HMM can represent input feature vectors with different lengths.

A different HMM with different transitions and output probabilities can represent a different phoneme. For a given test pattern (or a sequence of input feature vectors) $O = [o_1 \;\; o_2 \;\; \ldots \;\; o_T]$, the likelihood $P(O|\lambda_m)$ of the pattern O being generated from a model λ_m representing the mth phoneme is calculated. If a particular model λ_m has larger likelihood than all other models, the test pattern is determined to be the mth phoneme.

$$P(O|\lambda_m) > P(O|\lambda_n) \qquad \text{for all } n \neq m$$

Most state-of-the-art HMM systems use Gaussian mixtures to represent the output probabilities.

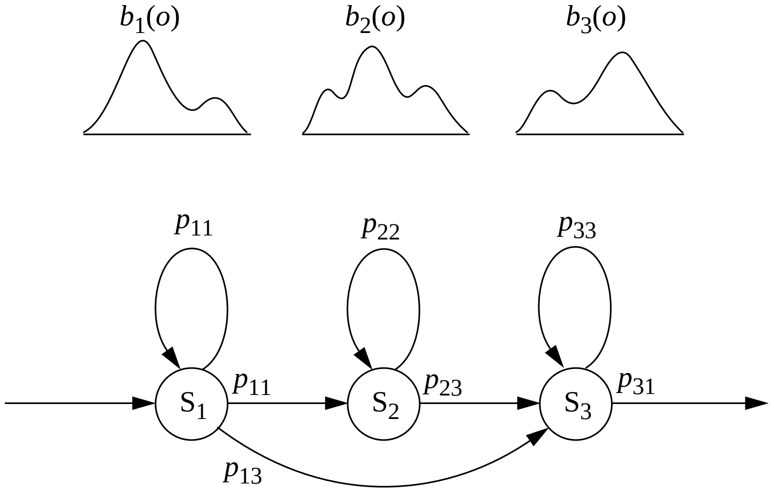
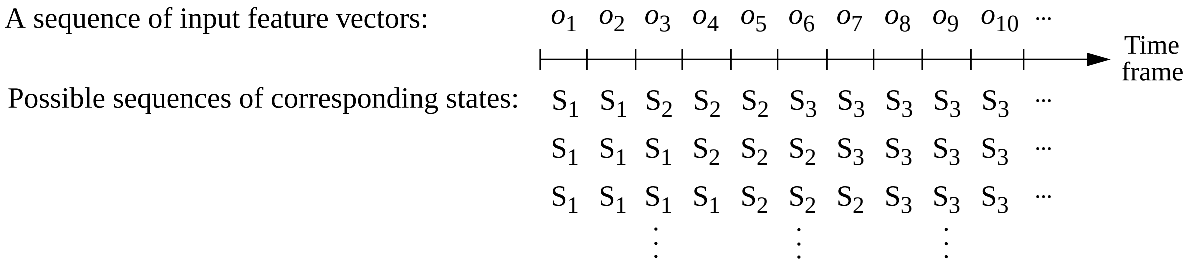


Figure 2.2 An example of hidden Markov models (HMM). A 3-state model is depicted, which represents a phoneme. Models are concatenated to represent a word or sentence.

S_i = State i
p_{ij} = Transition probability from state i to state j
b_i = Output probability for state i,



A sequence of input feature vectors: o_1 o_2 o_3 o_4 o_5 o_6 o_7 o_8 o_9 o_{10} ···

Time frame

Possible sequences of corresponding states: S_1 S_1 S_2 S_2 S_2 S_3 S_3 S_3 S_3 S_3 ···
S_1 S_1 S_1 S_2 S_2 S_2 S_3 S_3 S_3 S_3 ···
S_1 S_1 S_1 S_1 S_2 S_2 S_2 S_3 S_3 S_3 ···
⋮ ⋮ ⋮

Figure 2.3 Sequences of input feature vectors and corresponding states

$$b_s(o) = \sum_k w_{sk} N(o; \mu_{sk}, C_{sk})$$

where $b_s(o)$ is an output probability for state s, w_{sk} is a mixing weight, μ_{sk} is a mean vector, and C_{sk} is a covariance matrix for k^{th} Gaussian in state s. More details about HMMs can be found in [24, 30, 47].

2.2.2 Language Modeling

In many applications, a speech recognition system is intended to recognize a sentence or a sequence of words. In this case, the recognized sentence $\hat{W}$ will be the sequence of words $W = [w_1 \; w_2 \; \dots \; w_n]$ which obtain the highest likelihood given the observation (O), *i.e.*

$$\hat{W} = \underset{W}{\arg max} \; P(W|O)$$

From Bayes' rule, this can be written as

$$\hat{W} = \underset{W}{\arg max} \; \frac{P(O|W)P(W)}{P(O)}$$

Because $P(O)$ does not depend on W,

$$\hat{W} = \underset{W}{\arg max} \; P(O|W)P(W)$$

$P(O|W)$ represents the acoustic likelihood which can be obtained using HMMs as explained in the previous subsection. $P(W)$ represents the likelihood of the sequence of words which can be obtained from *language modeling*[30]. A popular language modeling method is to estimate the probability of a word considering its history.

$$P(W) = P(w_1, w_2, \dots, w_n) = \prod_{i=1}^{n} P(w_i | w_{i-1}, w_{i-2}, \dots, w_1)$$

In practice, we usually consider only a limited history for language modeling. For example, the popular bigram and trigram models consider only one previous word and two previous words, respectively.

We apply *language model weights* to the probability from the language model to control the relative contributions of the acoustic and language models. For a higher language model weight, a recognition system relies more on the language model.

2.3 Review of Speaker Adaptation

For adaptation we can either modify the input feature vectors or the recognition model parameters as shown in Fig. 2.4 [36]. Cepstral mean normalization (CMN) (or cepstral mean subtraction [48]), code-dependent cepstral normalization (CDCN) [1], and vocal tract length normalization [20] are some of the examples of the method which modifies input feature vectors. Sankar and Lee [51] proposed a maximum-likelihood stochastic matching approach to estimate an inverse distortion function. These methods modify all input feature vectors, which may require too much computation. On the contrary when we modify recognition model parameters, we can modify the model parameters once, and use them for all input feature vectors. A different phoneme may have a different mismatch. In this case, modifying all the input feature vectors in the same way will not be appropriate. The adaptation of recognition model parameters is useful because we can easily modify different phonemes in different ways.

In this thesis, we focus on modifying the model parameters and specifically Gaussian mean vectors so that they can represent a new condition (speaker or environment) better. In this section we will explain speaker adaptation techniques, focussing on the two most common approaches and their extensions.

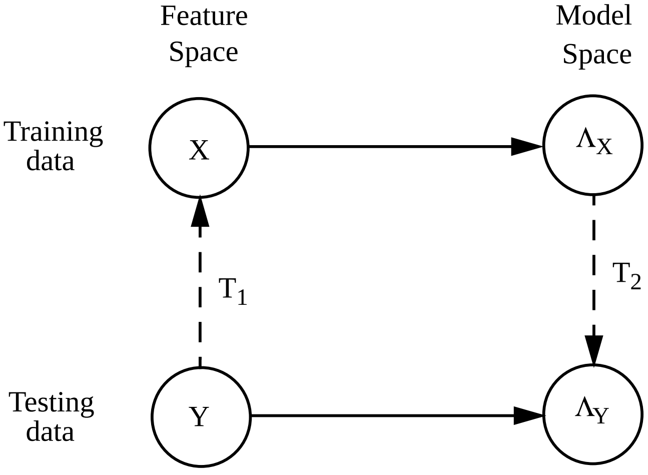


Figure 2.4 Adaptation in feature space (test data or input feature vectors), or in model space (recognition model parameters)

2.3.1 Bayesian Adaptation

In Bayesian or maximum *a posteriori* probability (MAP) adaptation, we assume that the target parameter that we try to estimate is a random variable, and that some prior knowledge about the target parameter is available. For a data vector ***X,*** the MAP estimate of a parameter λ is given by

$$\lambda_{MAP} = argmax_{\lambda}(P(\lambda|X)) = argmax_{\lambda}(P(X|\lambda)P(\lambda))$$

where $P(\lambda)$ represents the prior distribution of the model parameter λ, and can be obtained from a large collection of training data.

Suppose we have N samples $x_1, \ldots, x_N$ which are independently drawn from a Gaussian distribution with mean μ and variance σ^2,

$$P(x|\mu) \sim N(\mu, \sigma^2)$$

and we try to estimate the mean value μ (the target parameter) whose prior distribution is known to have a Gaussian distribution

$$P(\mu) \sim N(\mu_o, {\sigma_o}^2)$$

The MAP estimate of a mean μ is obtained by (*e.g.* [15])

$$\begin{aligned} \hat{\mu}_{MAP} &= \frac{N{\sigma_o}^2}{N{\sigma_o}^2 + \sigma^2}\bar{x} + \frac{\sigma^2}{N{\sigma_o}^2 + \sigma^2}\mu_o \\ &= \beta\bar{x} + (1-\beta)\mu_o \qquad (2.1) \end{aligned}$$

where $\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x_n$ is the sample mean, and $\beta = \frac{N{\sigma_o}^2}{N{\sigma_o}^2 + \sigma^2}$ is a weighting factor.

In the above equation, the MAP estimate can be interpreted as a weighted average (or an interpolation) of the sample mean $\bar{x}$ and the prior mean μ_o. When there are no samples of data available ($N = 0$), $\beta = 0$

and the MAP estimate is simply the prior mean ($\hat{\mu}_{MAP} = \mu_o$). In other words, there is no adaptation. As we obtain more samples ($N \to \infty$), β becomes one and the MAP estimate converges to the sample mean ($\hat{\mu}_{MAP} \to \bar{x}$). Fig. 2.5 illustrates the MAP estimates for different amount of samples for three Gaussian mean values. Note that the estimate $\hat{\mu}_{1,\,MAP}$ is close to its prior mean value μ_{1o} because it is obtained using a small number of samples, and $\hat{\mu}_{3,\,MAP}$ is close to its sample mean value $\bar{x}_3$ because it is obtained using a large number of samples.

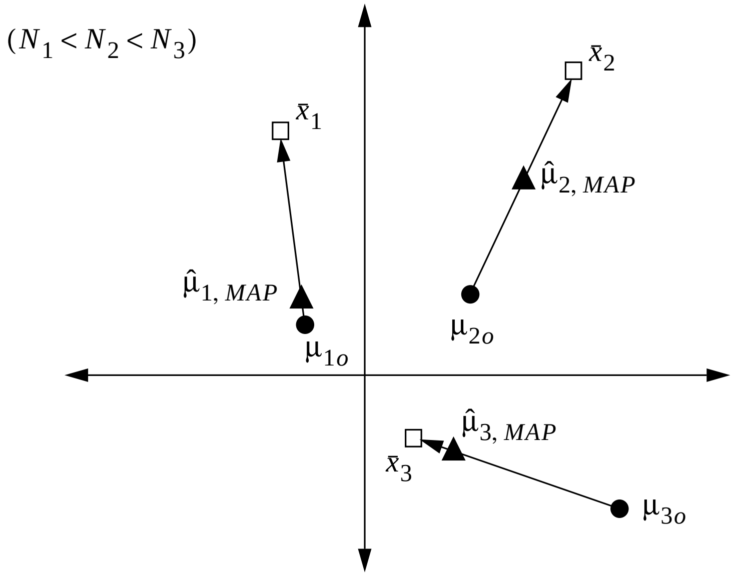


Figure 2.5 An example of MAP estimates of Gaussian mean vectors with different amount of adaptation data

If we apply this formulation to speaker adaptation, we can consider the prior mean as a SI model (or baseline model) parameter, and the samples as adaptation data. When a large amount of adaptation data is available ($N \to \infty$), adapted model parameters converge to the corresponding SD model parameters, the ML estimates. Hence, MAP estimation has good asymptotic properties. When only a small amount of adaptation data is provided, the number of samples N for the some of the model parameters will be zero, and the parameters are consequently not adapted. This is one of the major disadvantages of Bayesian adaptation. A large amount of adaptation data is needed to provide a significant change in the model parameters and recognition accuracy. It is also noted that if the prior variance σ_o^2 is large (an “uninformative prior”) then the MAP estimate is approximately equal to the ML estimate because we don’t have useful prior information.

An early seminal study on MAP adaptation for speech recognition by Lee *et al.* [37] described how to reestimate the mean and covariance parameters of continuous density HMMs. Gauvain and Lee [18] further extended this work to estimate other HMM parameters. The Bayesian approach has been applied to semi-continuous HMMs and discrete density HMMs by Huo *et al.* [26], among many others.

2.3.2 Transformation-Based Adaptation

In transformation-based adaptation, we assume that a set of the target parameters is updated by the same transformation. When we try to adapt recognition model parameters (especially Gaussian mean vectors in continuous HMMs), we estimate the transformation function using a small amount of adaptation data instead of estimating the individual recognition model parameters. Then we update the recognition model parameters using the estimated transformation. We can think of the transformation as an intermediate parameter as shown in Fig. 2.6. Because the number of transformation parameters is usually much smaller than the number of the target parameters, the estimation of the transformation parameters can be more reliable when only a small amount of adaptation data is available. So transformation-based adaptation is very useful in that case. However, because it can’t capture the details of the individual target parameters, it is less useful when a larger amount of adaptation data is available.

Transformation-based adaptation most frequently uses an ML criterion to estimate the transformation parameters. The ML estimate of a model parameter λ is given by

$$\lambda_{ML} = argmax_{\lambda}(P(\boldsymbol{X}|\lambda)) \tag{2.2}$$

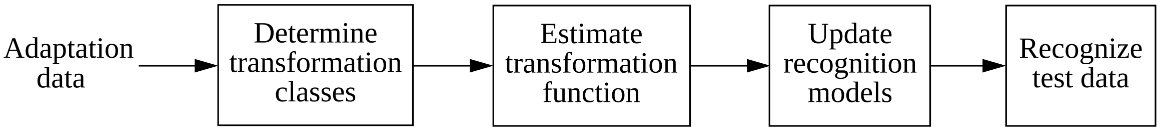

Adaptation data
Determine transformation classes
Estimate transformation function
Update recognition models
Recognize test data

Figure 2.6 A block diagram of transformation-based adaptation

where $\boldsymbol{X}$ refers to the adaptation data and $P(\boldsymbol{X}|\lambda)$ refers to the likelihood function. In ML estimation, the parameter of interest is assumed to be a deterministic but unknown constant. It does not take advantage of the prior knowledge that we may have about the parameters.

Fig. 2.7 depicts an example of transformation-based adaptation. Let's assume we have the same baseline means and samples as we had in the example in the previous section. Now we assume these means are updated by a same function, for example, a same shift vector $\boldsymbol{b}$ (*i.e.* $\hat{\mu}_k = \mu_k + \boldsymbol{b}$ where $\hat{\mu}_k$ is the adapted mean and μ_k is the baseline mean). First, we estimate the shift vector $\hat{\boldsymbol{b}}$ (shown as dotted lines) by using adaptation data. Then we adapt the means using the shift vector $\hat{\boldsymbol{b}}$. Notice that even though μ_{1o} has a small number of samples, it has the same shift as other means, and the resulting means (shown as triangles) are different from those in the example in the previous section.

In transformation-based adaptation, we can classify target parameters (usually Gaussian mean vectors) in different ways. We may classify them as a single transformation class, or multiple transformation classes. Once we determine the transformation classes, we consider each class independently. We estimate transformation function parameters for a class using only the samples which correspond to that class. If we have more transformation classes then we will have fewer samples for each class. Fig. 2.8 shows examples of different classifications. Each dot represents a Gaussian mean vector which is associated with a different phoneme or sub-phoneme. If we assume a shift vector $\boldsymbol{b}$ for each transformation class (*i.e.* $\hat{\mu}_k = \mu_k + \boldsymbol{b}, k \in \text{a transformation class}$), we may update the mean vectors as shown in Fig. 2.8 (a) or as in Fig. 2.8 (b). For the single-class case (a), we use all the samples to estimate $\hat{\boldsymbol{b}}$, so we may get a reliable

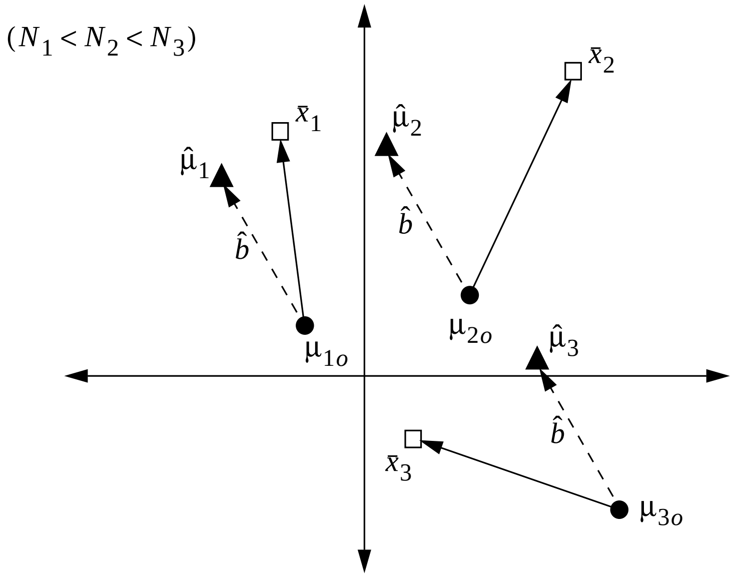
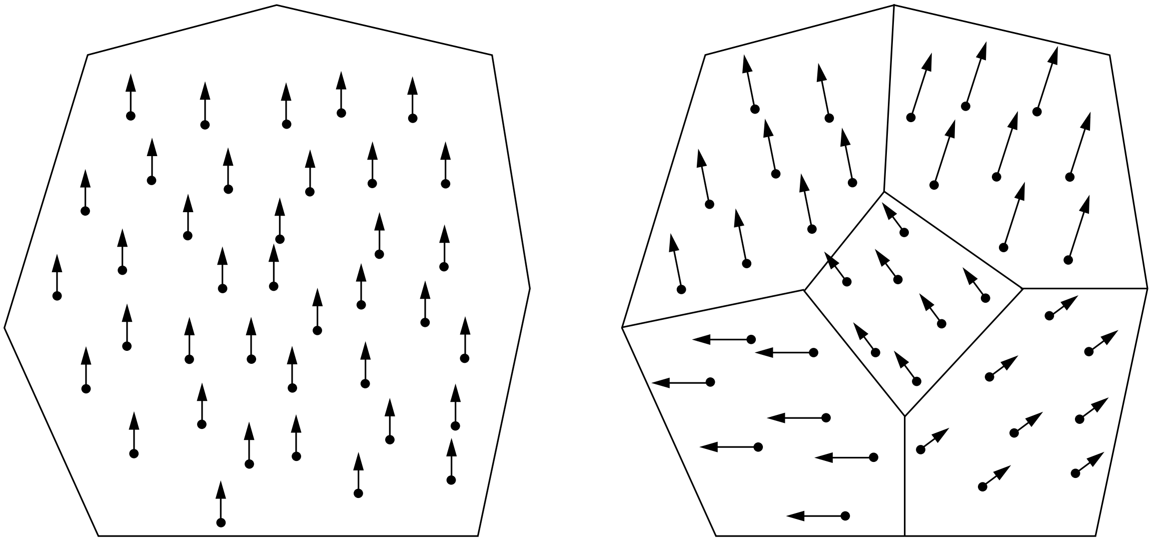


Figure 2.7 An example of transformation-based adaptation of Gaussian mean vectors assuming $\hat{\mu}_k = \mu_{ko} + \boldsymbol{b}$

estimation (or small estimation error) of $\hat{\boldsymbol{b}}$ even with a small number of samples. However, this $\hat{\boldsymbol{b}}$ may not describe the movement of the individual Gaussian mean vector well. For the multiple-class case (b), the shift vector $\hat{\boldsymbol{b}}_c$ for a transformation class c will describe the movement of the individual Gaussian mean vector better than the single-class case (a). However, because each shift vector $\hat{\boldsymbol{b}}_c$ is estimated using smaller number of samples which correspond to the class c, $\hat{\boldsymbol{b}}_c$ may not be estimated as reliably. We usually choose a small number of transformation classes (or a single class) for a small number of adaptation samples. As we have more adaptation samples we choose larger number of transformation classes. The optimal number of transformation classes will depend on the number of adaptation samples available.

The performance of transformation-based adaptation depends on the quality of the transformation model. Consider the example in Fig. 2.9. There are two baseline mean vectors (μ_{1o}, μ_{2o}). The true values of the adapted mean vectors ($\tilde{\mu}_1$, $\tilde{\mu}_2$) are shown as empty circles. We actually don't know the true values



(a) Single class

(b) multiple classes

Figure 2.8 An example of transformation-based adaptation with different number of transformation classes (a) single class, (b) multiple class

and try to estimated the adapted mean vectors using adaptation samples (or sample mean vectors $\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2$). Let's assume that the baseline mean vectors are updated by the same shift vector $\boldsymbol{b}$ (*i.e.* $\hat{\mu}_k = \mu_{ko} + \boldsymbol{b}$, k = 1,2). Then the estimation of the shift vector $\boldsymbol{b}$ will be given as $\hat{\boldsymbol{b}}$, and the adapted mean vectors as $\hat{\mu}_1, \hat{\mu}_2$. In this example, even though the sample mean vectors are located close to their true mean vectors, the estimations of the adapted mean vectors are far from the true vectors. It is because the assumption of transformation function, a same shift vector in this example, is not good for the given mean vectors. So a good model for the transformation function is important. A good model can describe the movement of the target parameters well with a small number of transformation parameters.

One of the most successful implementations of transformation-based adaptation is the Maximum Likelihood Linear Regression (MLLR) method developed by Leggetter and Woodland [40]. MLLR applies a linear transformation on recognition model parameters (which are usually Gaussian mean vectors) to obtain an adapted model. If μ_k is the mean vector of a Gaussian mixture component in an SI model, the

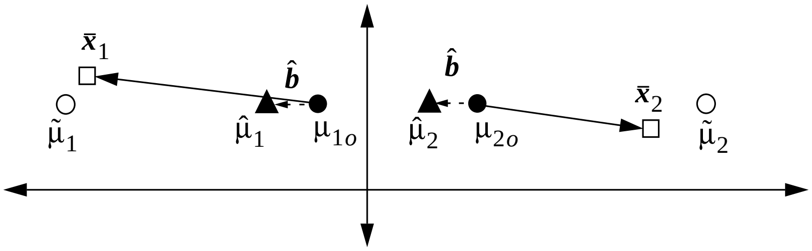


Figure 2.9 An example of transformation-based adaptation with an improper transformation model

corresponding mean of the adapted mean vector is given by the following transformation.

$$\hat{\mu}_k = \boldsymbol{A}\mu_k + \boldsymbol{b} \tag{2.3}$$

where $\boldsymbol{A}$ is a multiplication matrix and $\boldsymbol{b}$ is a shift vector. These transformation parameters are estimated by ML procedures. We will discuss MLLR further in Chapter 3.

2.3.3 Comparison of Bayesian and Transformation-Based Adaptation

In the previous sections we showed two major approaches for speaker adaptation, Bayesian adaptation and transformation-based adaptation. In typical cases, transformation-based adaptation is better for a small amount of adaptation data, and Bayesian adaptation is better for a larger amount of adaptation data. Bayesian adaptation tends to have better asymtotic properties than transformation-based adaptation. Fig. 2.10 shows an example of the recognition results using both methods. The experiements were performed using the non-native speakers from 1994 DARPA Wall Street Journal task. In Fig. 2.10, the zero adaptation sentence indicates the case of the baseline system (no adaptation). It is noted that transformation-based adaptation uses a single class for all cases, and may improve the recognition accuracy if more transformation classes are used.

It should be noted that the names for these adaptations may be misleading. In Bayesian adaptation, we usually make use of the prior distributions of the target parameters and apply MAP criterion on them.

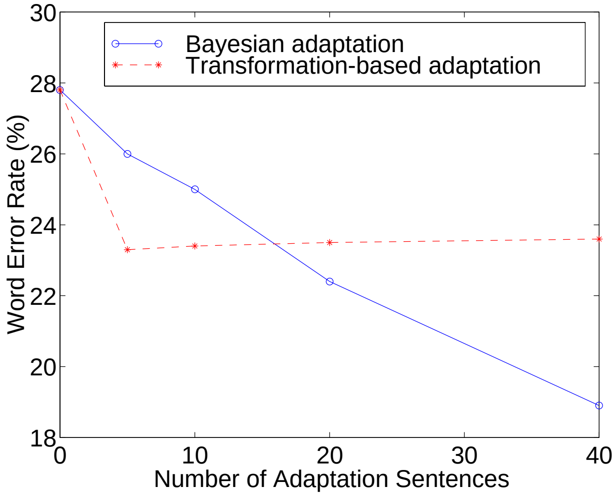


Figure 2.10 A comparison of recognition error rates of Bayesian and transformation-based adaptation for different amount of adaptation data

However, we can apply MAP criteria to a transformation function too. For example, the MAP criterion has been applied to linear regression in the MAPLR method [7]. In transformation-based adaptation, we usually assume that the number of transformation parameters is much smaller than the target parameters. However, we can increase the number of transformation functions. In an extreme case, we can apply transformation function for each target parameter (even though it is not transformation-based adaptation in an usual sense). This will give the same result as the sample mean gives, *i.e* an asymptotic result. In Fig. 2.10 transformation-based adaptation produces a larger error rate than Bayesian adaptation for a large amount of adaptation. However, in the case of a very large amount of available data, the transformation-based estimate as well as the MAP estimate will become the ML estimate if a sufficient number of transformation classes are used.

There are three important sources of information that can be used for adaptation.

(1) Adaptation data

(2) Individual prior distributions of either target parameters or transformation function parameters

(3) Relationships among different parameters

Bayesian adaptation uses the information sets (1) and (2), and attempts to estimate the means by combining *a priori* information and the adaptation data. Transformation-based adaptation uses information sets (1) and (3). It assumes a common transformation function and uses adaptation data to estimate the parameters of that function.

2.3.4 Extensions to Bayesian and Transformation-Based Adaptation

In conventional MAP adaptation parameters are estimated independently, so a parameter is not adapted if there are no adaptation data associated with it. Several methods have been proposed to overcome this disadvantage, which mostly utilize the relationships among different parameters.

Lasry and Stern [35] introduced the extended MAP (EMAP) adaptation which makes use of information about correlations among parameters. A parameter can be adapted even though there is no adaptation data directly associated with it. Zavaliagkos [56] was the first to apply EMAP adaptation to large scale HMM speech recognizers. While the EMAP approach produces an elegant analytical adaptation equation that makes appropriate use of correlations among adaptation parameters, solution of the equation depends on the inversion of a large matrix. Rozzi and Stern [49] developed a least mean square (LMS) algorithm for an efficient computation of the EMAP algorithm at the expense of a finite misadjustment.

Huo and Lee [27] proposed an algorithm based on pairwise correlation of parameter pairs to reduce computational complexity. Afify *et al* [3] also used pairwise correlation of parameters. For a target parameter, they got several estimates using correlations with different parameters, and combine them to get the final estimate for the target parameter.

Chen and DeSouza [6] also used the correlation between speech units for a similar speaker adaptation algorithm which they refer to as *Adaptation By Correlation* (ABC). The estimates are derived using least squares theory, and it is very similar to EMAP estimation. They reported that ABC is more stable than

MLLR when the amount of adaptation data is very small on the Wall Street Journal task, and that ABC enhances MLLR when they applied ABC with MLLR on the F0 condition of the ARPA HUB 4-96 evaluation data (which deals with the recognition of clean prepared speech from broadcast news shows).

Sagayama *et al* [53, 54] suggested the Vector Field Smoothing (VFS) method to overcome insufficient training data. An unobserved vector is estimated by interpolating the neighboring vectors, and trained vectors are smoothed to reduce estimation errors.

Cox [10] predicted unheard sounds using correlation models. Ahadi and Woodland [2] proposed the regression-based model prediction (RMP) technique. They applied linear regression to estimate parametric relationships among the model parameters, and used it to update those parameters for which there was insufficient adaptation data.

Zavaliagkos [56] suggested a method to tie adaptation across clusters of models. As is done in transformation-based adaptation, he applied a simple transformation on the parameters in a cluster. A shared mean shift and a shared variance scaling are assumed, and estimated using the MAP framework. Other research groups have proposed hybrid algorithms to combine the transformation-based and Bayesian adaptation approaches [8, 11].

Shinoda and Lee [52] proposed a structured MAP (SMAP) adaptation, in which the transformation parameters are estimated in a hierarchical structure. Despite its name, the SMAP algorithm does not make use of prior information about speaker variability. The MAP approach was introduced to achieve a better interpolation of the parameters at each level. Parameters at a given level of the hierarchical structure are used as the priors for the next lower child level. The resulting transformation parameters are a combination of the transformation parameters at all levels. The weights for the combinations are changed according to the amount of adaptation data present. The main benefit of the SMAP adaptation is that automatic control is obtained over the effective cluster size in a fashion that depends on the amount of adaptation data.

Many of these techniques can be thought of as an interpolation or smoothing of the MAP estimation procedure using either parameter correlations (as in the EMAP, ABC algorithms), neighborhoods in vector space (as in the VFS algorithm), regression models (as in the RMP algorithm), or parameter tying (as in the hybrid algorithm). In these ways they attempt to estimate unobserved parameters, and to get robust estimates when very small amounts of adaptation data are available. However, interpolation and smoothing can cause potentially useful detailed information to become lost.

Correlation and regression among Gaussian mean vectors have been used mostly in a Bayesian framework. It describes shift but not rotation of mean vectors. Because there are thousands of Gaussians in speech recognition systems, considering only a few neighboring Gaussians may not have much effect on Bayesian estimates of mean vectors, and consideration of larger numbers of neighboring Gaussians may require too much computation. It is also difficult to apply correlations among multi Gaussian mixtures because there are no explicit correspondence among Gaussians. In this thesis, we will describe a new algorithm in Chapter 5 which utilizes relationships among different transformation functions in transformation-based adaptation.

2.4 Experimental Environment

2.4.1 The SPHINX-3 Speech Recognition System

The SPHINX-3 system is used as the baseline speech recognition system to evaluate the algorithms to be developed in this thesis. Even though we report results only using SPHINX-3 in this thesis, the algorithms can be applied to any continuous HMM-based speech recognition system. SPHINX-3 is a large-vocabulary, speaker-independent, HMM-based continuous speech recognition system developed at Carnegie Mellon University. Even though SPHINX-3 can handle both continuous and semi-continuous models, only the continuos models are used in this thesis. The output probabilities in the continuous model are represented by Gaussian mixtures. For adaptation, the mean vectors of the Gaussian mixtures are modified in this thesis.

As shown in Fig. 2.1 the input speech is converted to feature vectors. In this thesis mel-frequency cepstral coefficients (MFCC) are used as static features for speech recognition [57]. First-order and second-order time derivatives of the cepstral coefficients are then obtained, and power information is included as a fourth feature. These features form a 39-dimensional vector (12+12+12+3) which is obtained every 10 msec. The input speech is digitized at sampling rate of 16 kHz.

Triphone models are used as the recognition units to consider the influence by the neighboring phonemens. Because the number of states of HMM for the triphone is too large, tied states, or senones are used to reduce the number [29]. The output distribution of a state is represented by the Gaussian mixtures. 6000 senones and 16 Gaussians per senone are used for Wall Street Journal task in this thesis.

2.4.2 Test Corpora

- **Wall Street Journal (s3-94, s0-94)**

  The main experimental task in this thesis is the Wall Street Journal task (WSJ) focussing especially on the non-native English speakers from Spoke 3 in the 1994 DARPA evaluation (s3-94). Spoke 3 is designed “to evaluate a rapid enrollment speaker adaptation algorithm on difficult speakers [45].” It consists of 10 speakers reading 20 test sentences each. Each speaker reads news articles from the WSJ corpus. The primary test in Spoke 3 uses 40 adaptation sentences for each speaker in supervised mode. In this thesis, however, we are interested in much smaller amounts of adaptation data, and we use only 1 or 3 adaptation sentences for each speaker. The following is some of the examples of the test sentences.

(1) NET INCOME FOR THE QUARTER EXCEEDED FOUR POINT FIVE MILLION DOLLARS OR ABOUT A DOLLAR THIRTY FIVE A SHARE

(2) THE MAJOR FACTOR IN OCTOBER WAS A SMALLER RISE IN NEW LOANS FOR CAR PURCHASES

(3) HERE ARE SOME OF THEIR ANSWERS

(4) MR. EVANS SAID THE THRIFT WILL CONTINUE TO EXAMINE POSSIBLE ACQUISITIONS OF OTHER THRIFTS OR THRIFT BRANCHES

(5) THE NEW AFFILIATE WILL INCLUDE FOUR PRODUCING DIVISIONS AND SIX EXPLORATION DIVI-

SIONS THE COMPANY SAID

(6) THE NIKKEI INDEX ADDED TWO HUNDRED EIGHTY SEVEN POINT TWENTY ONE POINTS TO CLOSE AT TWENTY SEVEN THOUSAND SEVEN HUNDRED AND THREE POINT NINETY ONE

(7) AS PREVIOUSLY REPORTED THE UTILITY HAS URGED REGULATORS TO APPROVE A FLAT TWENTY SEVEN PERCENT RATE INCREASE

The baseline speech recognition system was trained using the standard speaker-independent training corpus which contains clean speech data with 20K-word vocabulary from 321 native speakers, referred to as SI-284 and SI-37. A 5000-word trigram language model was used in recognition test.

The Spoke 0 (s0-94) corpus is also used to evaluates the speaker adaptation algorithms for native speakers. We use 20 speakers and 10 test sentences each. The Spoke 0 is similar to the Spoke 3 except that they are from native speakers.

- **The Telefónica (TID) corpus**

This corpus was recorded over the GSM cellular telephone network in Spain [25]. It is a small vocabulary set (approximately 75 words in lexicon) which contain mainly numbers. It includes approximately 6000 training utterances and 3000 testing utterances (with approximately 14500 tokens in the test) set spoken by approximately 1700 speakers. It was originally recorded at 8000 kHz. It has been manually transcribed and annotated for acoustic environment, speaker's dialect and other conditions. It contains a broad sample of speaker and environmental conditions.

- **Broadcast News from the 1998 DARPA Evaluation (Hub 4)**

The Hub 4 corpus contains recorded speech from TV broadcast news [46]. It is a mixture of several different conditions, *e.g.* planned/spontaneous, clean/noisy, full/narrow bandwidth, and native/non-native. It has a much larger vocabulary than the previous corpora. The baseline system is trained using about 100 hours of speech, including all conditions together. About one hour of evaluation data in

1998 are used for the test in this thesis. There are no separate adaptation data provided for this task. Typically the speech data are first segmented automatically, and clustered into similar conditions. Then adaptation is performed in unsupervised mode for each cluster.

2.5 Summary

In this chapter we have presented a brief review of automatic speech recognition, speaker adaptation, and the experimental environment used in this thesis. Hidden Markov models (HMMs) represent the characteristic of speech in statistical way. The most important parameters in HMMs are their output distributions, which are usually represented by Gaussian mixture densities. We can modify these parameters to adapt a baseline model to a new test condition. We presented the basic characteristics of Bayesian and transformation-based adaptation, and their extensions to overcome their disadvantages. In this thesis we will focus on transformation-based adaptation because it gives better recognition accuracy with limited amount of adaptation data. In the next chapter we will review maximum likelihood linear regression (MLLR) which is the most widely used method these days.

Chapter 3
Maximum Likelihood Linear Regression

Maximum Likelihood Linear Regression (MLLR) is currently one of the most popular speaker adaptation techniques. It assumes that the baseline mean vectors are transformed to adapted mean vectors by linear regression. In this chapter we will first review classical linear regression which relates a pair of samples. We will explain that estimates using different criteria can generate the same result. We will then describe MLLR and compare MLLR with classical linear regression. We will also describe some approaches to implementation that reduce the computational cost of MLLR estimates.

3.1 Classical Linear Regression

Consider the linear regression between samples x_i and y_i as shown in Fig. 3.1. (We use scalar values for simplicity.) For each pair of samples (x_i, y_i), we assume

$$y_i = ax_i + b + e_i, \quad i = 1, 2, ..., N \tag{3.1}$$

where e_i is an error term and N is the number of sample pairs. Once we estimate (a, b), we can predict y_i given x_i according to the linear equation.

$$\hat{y}_i = ax_i + b \tag{3.2}$$

Linear regression parameters (a, b) can be estimated in different ways. In the following three subsections, we describe the similarity of different methods.

3.1.1 Least Squares Estimation

We estimate (a_e, b_e) to minimize the squared error, *i.e.* minimize

$$Q_e = \sum_{i=1}^{N} e_i^2 = \sum_{i=1}^{N} (y_i - a_e x_i - b_e)^2 \tag{3.3}$$

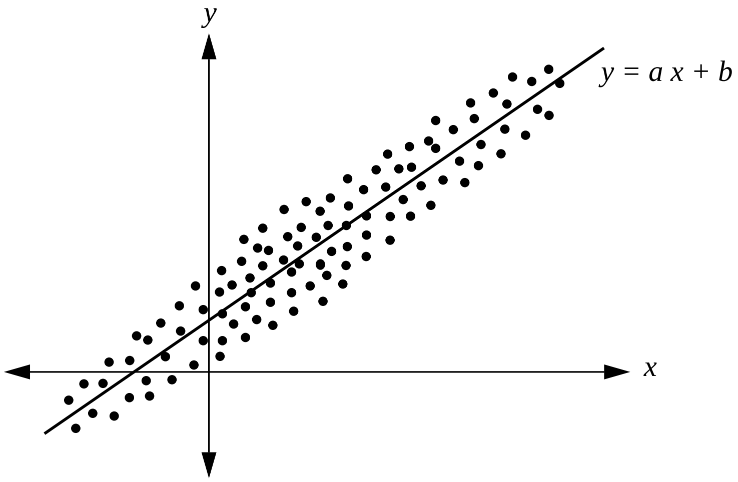


Figure 3.1 Linear regression between sample x_i and y_i

3.1.2 Maximum Likelihood Estimation

We assume that the error e_i has Gaussian distribution, or $e_i \sim N(0, \sigma_e^2)$. Then we estimate (a_m, b_m) to maximize the probability of the error. Specifically we maximize

$$f(e) = \prod_{i=1}^{N} f_i(e_i) = k \cdot \exp\left[\frac{-1}{2\sigma_e^2} \sum_{i=1}^{N} (y_i - a_m x_i - b_m)^2\right] \tag{3.4}$$

which is accomplished by minimizing

$$Q_m = \frac{1}{\sigma_e^2} \sum_{i=1}^{N} (y_i - a_m x_i - b_m)^2 \tag{3.5}$$

We can see that Eqs. (3.3) and (3.5) give the same result if σ_e^2 is constant. The solution is [5]

$$a_e = a_m = \frac{\sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{N} (x_i - \bar{x})^2} \tag{3.6}$$

$$b_e = b_m = \bar{y} - a_e\bar{x} \tag{3.7}$$

where $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$ is the sample mean of x_i, and $\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$ is the sample mean of y_i.

If e_i has a different variance σ_{ei}^2 for a different sample pair (x_i, y_i), then Eq. (3.5) becomes

$$Q'_e = \sum_{i}^{N} (y_i - a'_e x_i - b'_e)^2 / \sigma_{ei}^2 \tag{3.8}$$

which is different from Eq. (3.3). However, there is another method that can be applied for this case, which is called the '*weighted least squares*' [*e.g.* 42, 43]. It minimizes the weighted sum of squared errors.

$$Q_w = \sum_{i}^{N} w_i e_i^2 = \sum_{i}^{N} w_i (y_i - a_w x_i - b_w)^2 \tag{3.9}$$

where w_i is a weight. We usually choose the inverse of the variance $(1/\sigma_{ei}^2)$ for the weight w_i. Then it becomes the same as Eq. (3.8). The solution is

$$a_w = \frac{\sum_{i=1}^{N} w_i (x_i - \bar{x}_w)(y_i - \bar{y}_w)}{\sum_{i=1}^{N} w_i (x_i - \bar{x}_w)^2} \tag{3.10}$$

$$b_w = \bar{y}_w - a_w \bar{x}_w \tag{3.11}$$

where $\bar{x}_w, \bar{y}_w$ are the weighted sample means, $\bar{x}_w = \frac{\sum_{i=1}^{N} w_i x_i}{\sum_{i=1}^{N} w_i}$ and $\bar{y}_w = \frac{\sum_{i=1}^{N} w_i y_i}{\sum_{i=1}^{N} w_i}$.

3.1.3 Estimation Using Correlation

It is interesting to note that linear regression is closely related to correlation. Consider two random variables (X,Y) which have a bivariate Gaussian distribution with mean vector $[\mu_X \ \ \mu_Y]^T$ and covariance matrix $C_X = \begin{bmatrix} \sigma_X^2 & \sigma_{XY} \\ \sigma_{XY} & \sigma_Y^2 \end{bmatrix}$. We can get the expected value of y_i given sample x_i using the conditional probability density function *f*(y|x) [*e.g.* 33].

$$\hat{y}_i = E(y_i | x_i) = \mu_Y + \frac{\sigma_{XY}}{\sigma_X^2}(x_i - \mu_X) \tag{3.12}$$

This can be written in the form

$$\hat{y}_i = a_c x_i + b_c \tag{3.13}$$

when

$$a_c = \frac{\sigma_{XY}}{\sigma_X^2} = \frac{E\{(X-\mu_X)(Y-\mu_Y)\}}{E\{(X-\mu_X)^2\}} \tag{3.14}$$

$$b_c = \mu_Y - a_c \mu_X \tag{3.15}$$

We can see that (a_c, b_c) is very similar to (a_e, b_e) and (a_m, b_m) in Eqs. (3.6) and (3.7), especially for a large *N*.

3.1.4 Estimation of x_i using y_i

In this section, we have been considered the linear regression of y_i on x_i, or the estimation of y_i using given x_i. For sample pairs (x_i, y_i), we can also think of linear regression in the reverse way, *i.e.* the estimation of x_i using given y_i. Let

$$\hat{x}_i = cy_i + d$$

using linear regression parameters (c, d). It may look like

$$y_i = \frac{1}{c}\hat{x}_i - \frac{d}{c}$$

and is comparable to Eq. (3.2). However it should be noted that $a \neq \frac{1}{c}$ and $b \neq -\frac{d}{c}$.

3.2 Maximum Likelihood Linear Regression (MLLR)

Maximum Likelihood Linear Regression (MLLR) [40] assumes that Gaussian mean vectors are updated by linear transformation. If μ_k is a baseline mean vector and $\hat{\mu}_k$ is the corresponding adapted mean vector for a new speaker or environmental condition, the relation between these two vectors is given by linear regression parameters $\boldsymbol{A}$ and $\boldsymbol{b}$,

$$\hat{\mu}_k = \boldsymbol{A}\mu_k + \boldsymbol{b}, \quad k \in \text{ a regression class} \tag{3.16}$$

or

$$\begin{bmatrix} \hat{\mu}_{k1} \\ \hat{\mu}_{k2} \\ \cdot \\ \hat{\mu}_{kD} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1D} \\ a_{21} & a_{22} & \dots & a_{2D} \\ \cdot & \cdot & \dots & \cdot \\ a_{D1} & a_{D2} & \dots & a_{DD} \end{bmatrix} \begin{bmatrix} \mu_{k1} \\ \mu_{k2} \\ \cdot \\ \mu_{kD} \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \cdot \\ b_D \end{bmatrix} \tag{3.17}$$

where μ_k, $\hat{\mu}_k$, and $\boldsymbol{b}$ are D-dimensional vectors, $\boldsymbol{A}$ is a D x D matrix, and D is the number of components in the feature vector. We estimate the regression parameters $\boldsymbol{A}$ and $\boldsymbol{b}$ from adaptation data so that the likelihood of the adaptation data is maximized (hence the term Maximum Likelihood Linear Regression). Then we update the mean vectors within the regression class using Eq. (3.16).

We normally obtain better speech recognition accuracy when ***A*** is a full matrix (compared to when it is a diagonal matrix) [39]. A full matrix means that each component of an adapted mean vector is a weighted combination of the components of the baseline mean vector, or each component of $\hat{\mu}_k$ depends on all the components of μ_k.

$$\hat{\mu}_{kr} = \sum_{i=1}^{D} (a_{ri}\mu_{ki}) + b_r, \qquad r = 1, 2, ..., D$$

On the other hand, we typically assume that each component of a mean vector is independent in a speech recognition system. We can explain this as follows. The baseline mean vectors move into a different location for the adapted mean vectors as shown in Fig. 3.2. It also changes the overall shape of the Gaussian mean vectors. We can characterize this movement by the combination of rotation, expansion/contraction, and shift. The MLLR parameters (***A***, ***b***) can describe these movements if ***A*** is a full matrix, and MLLR is not related to the independence of each component within a mean vector.

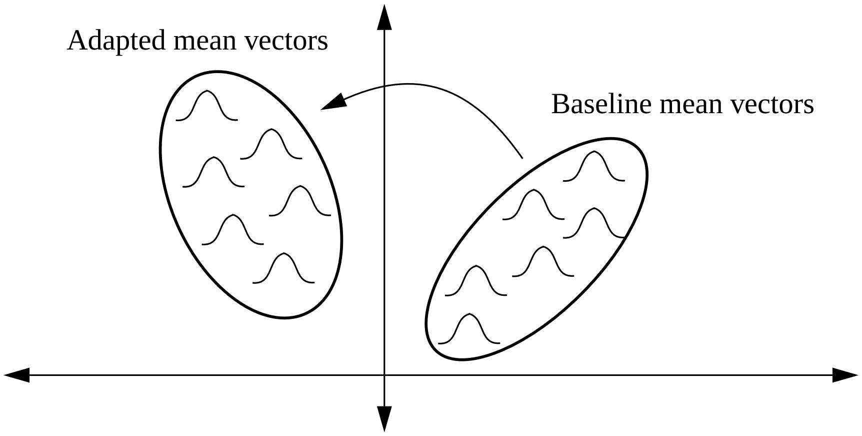


Figure 3.2 An example of the transformation of Gaussian mean vectors

In applying MLLR to speech recognition using HMMs, we try to estimate the MLLR parameters $\boldsymbol{A}$ and $\boldsymbol{b}$ which maximize the likelihood of the adaptation data (or observation sequence) $\boldsymbol{O} = (\boldsymbol{o}_1, \boldsymbol{o}_1, ..., \boldsymbol{o}_T)$ from the Gaussian mixtures

$$N(\hat{\mu}_k, \boldsymbol{C}_k) = N(\boldsymbol{A}\mu_k + \boldsymbol{b}, \boldsymbol{C}_k), \qquad 1 \le k \le K$$

and the HMM parameter set λ. The index *k* represents all the Gaussians which belong to the regression class. Again, $\boldsymbol{o}_t$, μ_k, $\hat{\mu}_k$, and $\boldsymbol{b}$ are *D*-dimensional vectors, $\boldsymbol{A}$ is a *D* x *D* matrix. It has been shown that we can estimate $\boldsymbol{A}$ and $\boldsymbol{b}$ by minimizing [40]

$$Q = \sum_t^T \sum_k^K \gamma_t(k)(\boldsymbol{o}_t - \boldsymbol{A}\mu_k - \boldsymbol{b})^{\mathrm{T}} \boldsymbol{C}_k^{-1} (\boldsymbol{o}_t - \boldsymbol{A}\mu_k - \boldsymbol{b}) \tag{3.18}$$

where $\gamma_t(k) = \gamma_t(s, k)$ is the *a posteriori* probability of being in the HMM state *s* at time *t* with the *k*th Gaussian, given the observation sequence $\boldsymbol{O}$ and the recognition model λ. If we compare this equation to Eq. (3.8), we can see they are very similar with μ_k corresponding to x_i and $\boldsymbol{o}_t$ corresponding to y_i, except that Eq. (3.18) has the probability $\gamma_t(k)$ and is written in matrix form. Therefore we can think of Eq. (3.18) as a weighted least squares between a baseline mean vector μ_k and observation $\boldsymbol{o}_t$ assuming $\boldsymbol{C}_k$ is the variance of the error

$$\boldsymbol{o}_t = \boldsymbol{A}\mu_k + \boldsymbol{b} + \boldsymbol{e}_t, \quad t = 1, 2, ..., N, \text{ and } \quad k = 1, 2, ..., K \tag{3.19}$$

with the probability $\gamma_t(k)$. We may interpret $\gamma_t(k)$ as a sample count, or we even think $\boldsymbol{W} = \gamma_t(k)\boldsymbol{C}_k^{-1}$ as a weight which corresponds to w_i in Eq. (3.9).

3.3 Implementation of MLLR

To solve Eq. (3.18), we determine the partial delivatives with respect to $\boldsymbol{A}$ and $\boldsymbol{b}$.

$$\frac{\partial Q}{\partial \boldsymbol{A}} = \sum_{t}^{T}\sum_{k}^{K}\gamma_t(k)\boldsymbol{C}_k^{-1}(\boldsymbol{o}_t - \boldsymbol{A}\mu_k - \boldsymbol{b})\mu_k^{\mathrm{T}} = 0 \tag{3.20}$$

$$\frac{\partial Q}{\partial \boldsymbol{b}} = \sum_{t}^{T}\sum_{k}^{K}\gamma_t(k)\boldsymbol{C}_k^{-1}(\boldsymbol{o}_t - \boldsymbol{A}\mu_k - \boldsymbol{b}) = 0 \tag{3.21}$$

If the covariance matrix $\boldsymbol{C}_k$ is diagonal as is commonly the case for speech recognition systems using cepstral features or their equivalent, we can write the above equations as

$$\sum_{t}^{T}\sum_{k}^{K}\gamma_t(k)\begin{bmatrix}\sigma_{k1}^{-2} & 0 & \dots & 0\\ 0 & \sigma_{k2}^{-2} & \dots & 0\\ \cdot & \cdot & \dots & \cdot\\ 0 & 0 & \dots & \sigma_{kD}^{-2}\end{bmatrix}\left\{\begin{bmatrix}o_{t1}\\ o_{t2}\\ \cdot\\ o_{tD}\end{bmatrix} - \begin{bmatrix}a_{11} & a_{12} & \dots & a_{1D}\\ a_{21} & a_{22} & \dots & a_{2D}\\ \cdot & \cdot & \dots & \cdot\\ a_{D1} & a_{D2} & \dots & a_{DD}\end{bmatrix}\begin{bmatrix}\mu_{k1}\\ \mu_{k2}\\ \cdot\\ \mu_{kD}\end{bmatrix} - \begin{bmatrix}b_1\\ b_2\\ \cdot\\ b_D\end{bmatrix}\right\}\begin{bmatrix}\mu_{k1}\\ \mu_{k2}\\ \cdot\\ \mu_{kD}\end{bmatrix}^{\mathrm{T}} = \begin{bmatrix}0\\ 0\\ \cdot\\ 0\end{bmatrix} \tag{3.22}$$

$$\sum_{t}^{T}\sum_{k}^{K}\gamma_t(k)\begin{bmatrix}\sigma_{k1}^{-2} & 0 & \dots & 0\\ 0 & \sigma_{k2}^{-2} & \dots & 0\\ \cdot & \cdot & \dots & \cdot\\ 0 & 0 & \dots & \sigma_{kD}^{-2}\end{bmatrix}\left\{\begin{bmatrix}o_{t1}\\ o_{t2}\\ \cdot\\ o_{tD}\end{bmatrix} - \begin{bmatrix}a_{11} & a_{12} & \dots & a_{1D}\\ a_{21} & a_{22} & \dots & a_{2D}\\ \cdot & \cdot & \dots & \cdot\\ a_{D1} & a_{D2} & \dots & a_{DD}\end{bmatrix}\begin{bmatrix}\mu_{k1}\\ \mu_{k2}\\ \cdot\\ \mu_{kD}\end{bmatrix} - \begin{bmatrix}b_1\\ b_2\\ \cdot\\ b_D\end{bmatrix}\right\} = \begin{bmatrix}0\\ 0\\ \cdot\\ 0\end{bmatrix} \tag{3.23}$$

Let's consider only an r^{th} component of the vectors and r^{th} row of matrices in the above equations for simplicity.

$$\sum_{t}^{T}\sum_{k}^{K}\gamma_t(k)\sigma_{kr}^{-2}\left\{o_{tr} - \begin{bmatrix}a_{r1} & a_{r2} & \dots & a_{rD}\end{bmatrix}\begin{bmatrix}\mu_{k1}\\ \mu_{k2}\\ \cdot\\ \mu_{kD}\end{bmatrix} - b_r\right\}\begin{bmatrix}\mu_{k1}\\ \mu_{k2}\\ \cdot\\ \mu_{kD}\end{bmatrix}^{\mathrm{T}} = 0 \tag{3.24}$$

$$\sum_{t}^{T}\sum_{k}^{K}\gamma_t(k)\sigma_{kr}^{-2}\left\{o_{tr} - \begin{bmatrix}a_{r1} & a_{r2} & \dots & a_{rD}\end{bmatrix}\begin{bmatrix}\mu_{k1}\\ \mu_{k2}\\ \cdot\\ \mu_{kD}\end{bmatrix} - b_r\right\} = 0 \tag{3.25}$$

Because only o_{tr} is dependent on the variable *t*, we can rewrite Eqs. (3.24) and (3.25) as

$$\sum_{k}^{K}\gamma(k)\sigma_{kr}^{-2}\left\{\bar{o}_{kr}-\begin{bmatrix}a_{r1} & a_{r2} & \dots & a_{rD}\end{bmatrix}\begin{bmatrix}\mu_{k1}\\ \mu_{k2}\\ \cdot\\ \mu_{kD}\end{bmatrix}-b_r\right\}\begin{bmatrix}\mu_{k1}\\ \mu_{k2}\\ \cdot\\ \mu_{kD}\end{bmatrix}^{\mathrm{T}} = 0 \tag{3.26}$$

$$\sum_{k}^{K}\gamma(k)\sigma_{kr}^{-2}\left\{\bar{o}_{kr}-\begin{bmatrix}a_{r1} & a_{r2} & \dots & a_{rD}\end{bmatrix}\begin{bmatrix}\mu_{k1}\\ \mu_{k2}\\ \cdot\\ \mu_{kD}\end{bmatrix}-b_r\right\} = 0 \tag{3.27}$$

where $\gamma(k) \equiv \sum_{t}^{T}\gamma_t(k)$ is the accumulation of the probability and $\bar{o}_{kr} \equiv \left(\sum_{t}^{T}\gamma_t(k)o_{tr}\right) / \left(\sum_{t}^{T}\gamma_t(k)\right)$ is the weighted sample mean value of the r^{th} component of the k^{th} Gaussian.

For the implementation of MLLR, it is better to use Eqs. (3.26) and (3.27) than Eqs. (3.24) and (3.25) because they require much less computation. It is noted that $\bar{o}_{kr}$ is the same equation that we use for mean re-estimation in training. So we can share a program for MLLR and mean re-estimation, which makes it easy to implement.

It is interesting that Eqs. (3.26) and (3.27) are the same solutions for the weighted least squares between a baseline mean vector μ_k and sample mean $\bar{o}_{kr}$ with weight $w_k = \gamma(k)\sigma_{kr}^{-2}$. Considering all components of $\bar{o}_{kr},\ r = 1, 2, ..., D$, we can say

$$\bar{\boldsymbol{o}}_k = \boldsymbol{A}\mu_k + \boldsymbol{b} + \boldsymbol{e}_k, \quad k = 1, 2, ..., K \tag{3.28}$$

where $\bar{\boldsymbol{o}}_k$ is the sample mean vector and $\boldsymbol{e}_k$ is an error vector of the Gaussian mixture density. Therefore we can interpret MLLR as an relationship either between μ_k and $\boldsymbol{o}_t$ (Eq. (3.19)), or between μ_k and $\bar{\boldsymbol{o}}_k$ (Eq. (3.28)).

3.4 Summary

In this chapter, we have reviewed classical linear regression and MLLR. We explained that the least squares estimation, the maximum likelihood estimation, and the estimation using correlation produce similar results. We showed that MLLR can be interpreted as a weighted least squares either between the baseline mean vectors and the input feature vectors, or between the baseline mean vectors and sample mean vectors. We also showed that we can compute the MLLR estimate efficiently by using a sample mean value for each Gaussian. In the following chapters, we will describe two new techniques to improve recognition accuracy in the MLLR framework.

Chapter 4
Principal Component MLLR

4.1 Introduction

In MLLR adaptation, we first estimate the matrix $\boldsymbol{A}$ and the vector $\boldsymbol{b}$ from adaptation data, and then update the mean vectors using Eq. (3.16). If the amount of adaptation data is small then the estimates of $\boldsymbol{A}$ and $\boldsymbol{b}$ may not be reliable (or have large variances). In this case, even though the estimates are obtained from adaptation data, they may not accurately describe the statistics of the test data, resulting in low recognition accuracy. Therefore we would like to obtain more reliable estimates of $\boldsymbol{A}$ and $\boldsymbol{b}$. We may achieve this by reducing the number of parameters.

Gales *et al.* [17] constrained the transformation matrix $\boldsymbol{A}$ to a block diagonal structure, with feature components assumed to have correlation only within a block. This reduces the number of parameters to be estimated. They used three blocks consisting of the static, delta, and delta-delta feature components. However, the block diagonal matrices did not provide better recognition accuracy than the full matrix in their test. This is because they already had enough adaptation data to estimate the full matrix, or because three blocks may not have been optimal.

Principal component analysis (*PCA*) [31] has been used to reduce the dimensionality of the data. The original data which consisted of interrelated variables are transformed into a new set of uncorrelated variables by the eigenvectors of the covariance matrix of the original data set. Nouza [44] used PCA for feature selection in a speech recognition system. Kuhn *et al*. [34] introduced "*eigenvoices*" to represent the prior knowledge of speaker variation. Hu [23] applied PCA to describe the correlation between phoneme classes for speaker normalization. While the general motivation for these approaches was similar to the approach described in this thesis, none of them are directly related to MLLR.

In this chapter, we will describe *principal component MLLR* (*PC-MLLR*) which applies PCA to the MLLR framework to reduce the variance of the estimate of matrix $\boldsymbol{A}$. The main idea is that if we estimate the matrix $\boldsymbol{A}$ in the eigendomain, the variances of the components of the estimates are inverse proportional to their eigenvalues. Therefore we can select more reliable components to reduce the overall variances. In this chapter we first review classical principal component regression and its application to MLLR. We then describe a refinement to the method which we refer to as *weighted principal component MLLR* (*WPC-MLLR*). Finally we will compare speaker adaptation performance obtained using the methods described.

4.2 Principal Component Regression

Principal component regression was developed in classical linear regression theory (*e.g.* [42, 43]). For example, consider N pairs of samples ($\mathbf{x}_i$, y_i), where $\mathbf{x}_i$ is a D-dimensional row vector, and y_i and $e_{\mathrm{x}i}$ are scalars. e_{x_i} is assumed to have a Gaussian distribution with zero mean and variance $\sigma_{e_{\mathrm{x}}}^2$. We assume that $\mathbf{x}_i$ and y_i are related by a linear regression vector $\boldsymbol{a}_{\mathbf{x}}$ (the shift term is assumed to be zero for simplicity).

$$\begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_N \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1D} \\ x_{21} & x_{22} & \dots & x_{2D} \\ \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \dots & \cdot \\ x_{N1} & x_{N2} & \dots & x_{ND} \end{bmatrix} \begin{bmatrix} a_{\mathrm{x}1} \\ a_{\mathrm{x}2} \\ \cdot \\ \cdot \\ a_{\mathrm{x}D} \end{bmatrix} + \begin{bmatrix} e_{\mathrm{x}1} \\ e_{\mathrm{x}2} \\ \cdot \\ \cdot \\ e_{\mathrm{x}N} \end{bmatrix} \tag{4.1}$$

or

$$\boldsymbol{y} = \boldsymbol{X} \cdot \boldsymbol{a}_{\mathbf{x}} + \boldsymbol{e}_{\mathbf{x}} \tag{4.2}$$

Letting $\boldsymbol{V}_{\boldsymbol{X}}$ be the orthonormal matrix (*i.e.* $\boldsymbol{V}_{\boldsymbol{X}}\boldsymbol{V}_{\boldsymbol{X}}^{\mathrm{T}} = \boldsymbol{I}$) whose columns are the eigenvectors of the correlation matrix $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$, and $\Lambda_{\boldsymbol{X}}$ be the diagonal matrix consisting of the corresponding eigenvalues λ_{Xj},

$$(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})\boldsymbol{V}_{\boldsymbol{X}} = \boldsymbol{V}_{\boldsymbol{X}}\Lambda_{\boldsymbol{X}}$$

Defining the new variables $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{V}_{\boldsymbol{X}}$ and $\alpha_{\mathbf{x}} = \boldsymbol{V}_{\boldsymbol{X}}^{\mathrm{T}} \boldsymbol{a}_{\mathbf{x}}$, from Eq. (4.2), we obtain

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{X}\boldsymbol{V}_{\boldsymbol{X}} \cdot \boldsymbol{V}_{\boldsymbol{X}}^{\mathrm{T}}\, \boldsymbol{a}_{\mathbf{x}} + \boldsymbol{e}_{\mathbf{x}} \\ &= \boldsymbol{Z} \cdot \alpha_{\mathbf{x}} + \boldsymbol{e}_{\mathbf{x}} \end{aligned} \tag{4.3}$$

or

$$\begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_N \end{bmatrix} = \begin{bmatrix} z_{11} & z_{12} & \dots & z_{1D} \\ z_{21} & z_{22} & \dots & z_{2D} \\ . & . & \dots & . \\ . & . & \dots & . \\ z_{N1} & z_{N2} & \dots & z_{ND} \end{bmatrix} \begin{bmatrix} \alpha_{\mathrm{x}1} \\ \alpha_{\mathrm{x}2} \\ . \\ . \\ \alpha_{\mathrm{x}D} \end{bmatrix} + \begin{bmatrix} e_{\mathrm{x}1} \\ e_{\mathrm{x}2} \\ . \\ . \\ e_{\mathrm{x}N} \end{bmatrix} \tag{4.4}$$

The estimate for $\alpha_{\mathbf{x}}$ is then

$$\begin{aligned} \hat{\alpha}_{\mathbf{x}} &= (\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z})^{-1}\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{y} \\ &= \Lambda_{\boldsymbol{X}}^{-1}\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{y} \end{aligned} \tag{4.5}$$

because

$$\begin{aligned} \boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z} &= (\boldsymbol{V}_{\boldsymbol{X}}^{\mathrm{T}}\boldsymbol{X}^{\mathrm{T}})(\boldsymbol{X}\boldsymbol{V}_{\boldsymbol{X}}) \\ &= \boldsymbol{V}_{\boldsymbol{X}}^{\mathrm{T}} \cdot (\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X})\boldsymbol{V}_{\boldsymbol{X}} \\ &= \boldsymbol{V}_{\boldsymbol{X}}^{\mathrm{T}} \cdot \boldsymbol{V}_{\boldsymbol{X}}\Lambda_{\boldsymbol{X}} \\ &= \Lambda_{\boldsymbol{X}} \end{aligned}$$

It is interesting to note that the variance of the individual components of the estimate $\hat{\alpha}_{\mathbf{x}}$ is inversely proportional to the corresponding eigenvalues of $\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}$ [42].

$$\begin{aligned} \hat{\alpha}_{\mathbf{x}} &= \Lambda_{\boldsymbol{X}}^{-1}\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{y} \\ &= \Lambda_{\boldsymbol{X}}^{-1}\boldsymbol{Z}^{\mathrm{T}}(\boldsymbol{Z} \cdot \alpha_{\mathbf{x}} + \boldsymbol{e}_{\mathbf{x}}) \\ &= \Lambda_{\boldsymbol{X}}^{-1} \cdot \boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z} \cdot \alpha_{\mathbf{x}} + \Lambda_{\boldsymbol{X}}^{-1}\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{e}_{\mathbf{x}} \\ &= \Lambda_{\boldsymbol{X}}^{-1} \cdot \Lambda_{\boldsymbol{X}} \cdot \alpha_{\mathbf{x}} + \Lambda_{\boldsymbol{X}}^{-1}\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{e}_{\mathbf{x}} \end{aligned}$$

$$= \alpha_{\mathbf{x}} + \Lambda_{\boldsymbol{X}}^{-1}\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{e}_{\mathbf{x}} \tag{4.6}$$

Therefore,

$$\begin{aligned} var(\hat{\alpha}_{\mathbf{x}}) &= E\{(\hat{\alpha}_{\mathbf{x}} - \alpha_{\mathbf{x}})(\hat{\alpha}_{\mathbf{x}} - \alpha_{\mathbf{x}})^{\mathrm{T}}\} \\ &= E\left\{(\Lambda_{\boldsymbol{X}}^{-1}\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{e}_{\mathbf{x}})(\Lambda_{\boldsymbol{X}}^{-1}\boldsymbol{Z}^{\mathrm{T}}\boldsymbol{e}_{\mathbf{x}})^{\mathrm{T}}\right\} \\ &= \Lambda_{\boldsymbol{X}}^{-1}\boldsymbol{Z}^{\mathrm{T}} \cdot E\{\boldsymbol{e}_{\mathbf{x}} \cdot \boldsymbol{e}_{\mathbf{x}}^{\mathrm{T}}\} \cdot \boldsymbol{Z}\Lambda_{\boldsymbol{X}}^{-1} \end{aligned}$$

If we assume $E\{\boldsymbol{e}_{\mathbf{x}} \cdot \boldsymbol{e}_{\mathbf{x}}^{\mathrm{T}}\} = \sigma_{e_{\mathrm{x}}}^2 \cdot \boldsymbol{I}_N$ where $\boldsymbol{I}_N$ is N X N identity matrix,

$$\begin{aligned} var(\hat{\alpha}_{\mathbf{x}}) &= \sigma_{e_{\mathrm{x}}}^2 \cdot \Lambda_{\boldsymbol{X}}^{-1} \cdot \boldsymbol{Z}^{\mathrm{T}}\boldsymbol{Z} \cdot \Lambda_{\boldsymbol{X}}^{-1} \\ &= \sigma_{e_{\mathrm{x}}}^2 \cdot \Lambda_{\boldsymbol{X}}^{-1} \cdot \Lambda_{\boldsymbol{X}} \cdot \Lambda_{\boldsymbol{X}}^{-1} \\ &= \sigma_{e_{\mathrm{x}}}^2 \cdot \Lambda_{\boldsymbol{X}}^{-1} \end{aligned}$$

or, for individual $\hat{\alpha}_{\mathrm{x}j}$,

$$var(\hat{\alpha}_{\mathrm{x}j}) = \sigma_{e_{\mathrm{x}}}^2 / \lambda_{Xj}, \qquad 1 \le j \le D \tag{4.7}$$

For example, consider a case where a sample $\mathbf{x}_i$ is a 2-dimensional vector (D = 2) in Eq. (4.1). We can estimate the linear regression parameters $(\hat{a}_{\mathrm{x}1}, \hat{a}_{\mathrm{x}2})$ in the original domain, and $(\hat{\alpha}_{\mathrm{x}1}, \hat{\alpha}_{\mathrm{x}2})$ in the eigen-domain. If $\mathbf{x}_i$ is distributed as in Fig. 4.1, $\hat{\alpha}_{\mathrm{x}1}$ which is associated with a larger eigenvalue, has a smaller variance than $\hat{\alpha}_{\mathrm{x}2}$. Therefore, the estimate $\hat{\alpha}_{\mathrm{x}1}$ is more reliable than the estimate $\hat{\alpha}_{\mathrm{x}2}$.

Because we know that some components of the estimates are more reliable than the others, we can obtain more reliable estimates overall by ignoring less reliable components and selecting only the top p < D principal components. If $\boldsymbol{V}_{\boldsymbol{X}}$ and $\Lambda_{\boldsymbol{X}}$ are sorted in descending order of the eigenvalues, we can pick the first p components of $\boldsymbol{Z}$ and $\alpha_{\mathbf{x}}$. Eq. (4.4) then becomes

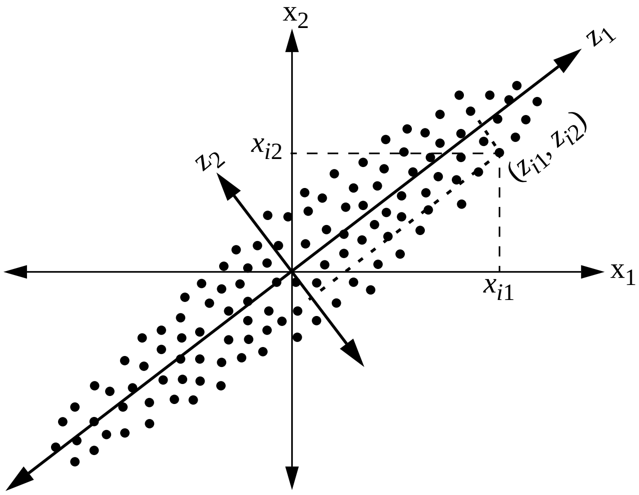


Figure 4.1 Distribution of x_i samples in the original domain (x_1, x_2) and in the eigendomain (z_1, z_2)

$$\begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ y_N \end{bmatrix} = \begin{bmatrix} z_{11} & \dots & z_{1p} \\ z_{21} & \dots & z_{2p} \\ \cdot & \dots & \cdot \\ \cdot & \dots & \cdot \\ z_{N1} & \dots & z_{Np} \end{bmatrix} \begin{bmatrix} \alpha_{x1} \\ \cdot \\ \cdot \\ \alpha_{xp} \end{bmatrix} + \begin{bmatrix} e_{x1} \\ e_{x2} \\ \cdot \\ \cdot \\ e_{xN} \end{bmatrix} \tag{4.8}$$

Now we have only p parameters to estimate rather than D parameters. After we estimate $\hat{\alpha}_{\mathbf{x}(p)} = [\hat{\alpha}_{x1} \; \dots \; \hat{\alpha}_{xp}]^T$ from Eq. (4.8), we can transform $\hat{\alpha}_{\mathbf{x}(p)}$ to obtain $\hat{\boldsymbol{a}}_{\mathbf{x}} = [\hat{a}_{x1} \; \hat{a}_{x2} \; \dots \; \hat{a}_{xD}]^T$ for the original linear regression. If we consider all the components of $\hat{\alpha}_{\mathbf{x}}$, $\hat{\boldsymbol{a}}_{\mathbf{x}} = \boldsymbol{V} \cdot \hat{\alpha}_{\mathbf{x}}$ from $\alpha_{\mathbf{x}} = \boldsymbol{V}_X^T \, \boldsymbol{a}_{\mathbf{x}}$. $\hat{\boldsymbol{a}}_{\mathbf{x}}$ obtained this way is same with $\hat{\boldsymbol{a}}_{\mathbf{x}}$ obtained directly from Eq. (4.2) in the original domain. If we consider only p components, we can let

$$\hat{\boldsymbol{a}}_{\mathbf{x}} = \boldsymbol{V}_{(p)} \cdot \hat{\alpha}_{\mathbf{x}(p)}$$

or

$$\begin{bmatrix} \hat{a}_{x1} \\ \hat{a}_{x2} \\ \cdot \\ \cdot \\ \hat{a}_{xD} \end{bmatrix} = \begin{bmatrix} v_{11} & \dots & v_{1p} \\ v_{21} & \dots & v_{2p} \\ \cdot & \dots & \cdot \\ \cdot & \dots & \cdot \\ v_{D1} & \dots & v_{Dp} \end{bmatrix} \begin{bmatrix} \hat{\alpha}_{x1} \\ \cdot \\ \cdot \\ \hat{\alpha}_{xp} \end{bmatrix}$$

The subscript (*p*) denotes that only the principal components corresponding to the *p* largest eigenvalues are used, so $\boldsymbol{V}_{(p)}$ consists of *p* eigenvectors in its column. The resulting $\hat{\boldsymbol{a}}_{\mathbf{x}}$ will have smaller variance than the estimate from conventional linear regression [42].

Even though choosing $p < D$ components reduces the variance, it makes $\hat{\boldsymbol{a}}_{\mathbf{x}}$ biased. As *p* becomes smaller, the bias becomes larger. The best value for *p* will depend on the eigenvalues. If the several largest eigenvalues are much larger than the others, then *p* can be small. If the smaller eigenvalues have relatively large values then *p* must be larger. We determine the best *p* value in advance using development data which are similar to the actual test data.

4.3 Principal Component MLLR

The formulation of *Principal Component MLLR* (*PC-MLLR*) [12] is very similar to that discussed in the previous section, except that we also consider the *a posteriori* probability $\gamma_t(k)$ as well as the baseline variance σ_k^2 and the shift vector $\boldsymbol{b}$.

Let's consider an arbitrary *r*th component $\hat{\mu}_{kr}$ of an adapted mean vector $\hat{\mu}_k$ and the corresponding row vector $\boldsymbol{a}_r^{\mathrm{T}}$ of the matrix $\boldsymbol{A}$. From Eqs. (3.16) and (3.17),

$$\hat{\mu}_{kr} = \boldsymbol{a}_r^{\mathrm{T}}\mu_k + b_r = \begin{bmatrix} a_{r1} & a_{r2} & \ldots & a_{rD} \end{bmatrix} \begin{bmatrix} \mu_{k1} \\ \mu_{k2} \\ \cdot \\ \mu_{kD} \end{bmatrix} + b_r \tag{4.9}$$

From Eq. (3.18), if the covariance matrix $\boldsymbol{C}_k$ is diagonal with *r*th component σ_{kr}^2,

$$Q = \sum_{t=1}^{T}\sum_{k=1}^{K}\gamma_t(k)\sigma_{kr}^{-2}(o_{tr} - \boldsymbol{a}_r^{\mathrm{T}}\mu_k - b_r)^2, \qquad 1 \le r \le D \tag{4.10}$$

As we explained in Section 3.3, we solve Eqs. (3.26) and (3.27) to estimate $\boldsymbol{a}_r^{\mathrm{T}}$. Let

$$w_{rk} = \sum_{t=1}^{T} \gamma_t(k)\sigma_{kr}^{-2} = \gamma(k)\sigma_{kr}^{-2} \tag{4.11}$$

Rewriting Eqs. (3.26) and (3.27), we obtain

$$\sum_{k=1}^{K} w_{rk}\{\bar{o}_{kr} - \boldsymbol{a}_r^{\mathrm{T}}\mu_k - b_r\}\mu_k^{\mathrm{T}} = 0 \tag{4.12}$$

$$\sum_{k=1}^{K} w_{rk}\{\bar{o}_{kr} - \boldsymbol{a}_r^{\mathrm{T}}\mu_k - b_r\} = 0 \tag{4.13}$$

From Eq. (4.13),

$$b_r = \frac{\sum_{k=1}^{K} w_{rk}\bar{o}_{kr}}{\sum_{k=1}^{K} w_{rk}} - \boldsymbol{a}_r^{\mathrm{T}} \cdot \frac{\sum_{k=1}^{K} w_{rk}\mu_k}{\sum_{k=1}^{K} w_{rk}} \tag{4.14}$$

or,

$$b_r = o_{r,\,avg} - \boldsymbol{a}_r^{\mathrm{T}}\mu_{avg} \tag{4.15}$$

where $o_{r,\,avg} = \frac{\sum_{k=1}^{K} w_{rk}\bar{o}_{kr}}{\sum_{k=1}^{K} w_{rk}}$ and $\mu_{avg} = \frac{\sum_{k=1}^{K} w_{rk}\mu_k}{\sum_{k=1}^{K} w_{rk}}$ are weighted averages.

Putting Eq. (4.15) into Eq. (4.12), we obtain

$$\sum_{k=1}^{K} w_{rk}\{(\bar{o}_{kr} - o_{r,\,avg}) - \boldsymbol{a}_r^{\mathrm{T}}(\mu_k - \mu_{avg})\}\mu_k^{\mathrm{T}} = 0 \tag{4.16}$$

or

$$\sum_{k=1}^{K} w_{rk}\{\tilde{o}_{kr} - \boldsymbol{a}_r^{\mathrm{T}}\tilde{\mu}_k\}\mu_k^{\mathrm{T}} = 0 \qquad (4.17)$$

where $\tilde{o}_{kr} = \bar{o}_{kr} - o_{r,\,avg}$ (a scalar), $\tilde{\mu}_k = \mu_k - \mu_{avg}$ (a D dimensional vector).

Rewriting Eq. (4.13), we obtain

$$\sum_{k=1}^{K} w_{rk}\{\tilde{o}_{kr} - \boldsymbol{a}_r^{\mathrm{T}}\tilde{\mu}_k\} = 0$$

Because μ_{avg} is constant, this expression can be written

$$\sum_{k=1}^{K} w_{rk}\{\tilde{o}_{kr} - \boldsymbol{a}_r^{\mathrm{T}}\tilde{\mu}_k\}\mu_{avg}^{\mathrm{T}} = 0 \qquad (4.18)$$

Subtracting Eq. (4.18) from Eq. (4.17), we can get

$$\sum_{k=1}^{K} w_{rk}\{\tilde{o}_{kr} - \boldsymbol{a}_r^{\mathrm{T}}\tilde{\mu}_k\}(\mu_k - \mu_{avg})^{\mathrm{T}} = 0$$

or

$$\sum_{k=1}^{K} w_{rk}\{\tilde{o}_{kr} - \boldsymbol{a}_r^{\mathrm{T}}\tilde{\mu}_k\}\tilde{\mu}_k^{\mathrm{T}} = 0 \qquad (4.19)$$

Eq. (4.19) can be written in matrix form instead of summation form:

$$\begin{bmatrix}\tilde{\mu}_1 & \tilde{\mu}_2 & \dots & \tilde{\mu}_K\end{bmatrix}\begin{bmatrix} w_{r1} & 0 & . & . & 0 \\ 0 & w_{r2} & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & 0 & . & . & w_{rK}\end{bmatrix}\begin{bmatrix}\tilde{\mu}_1^{\mathrm{T}} \\ \tilde{\mu}_2^{\mathrm{T}} \\ . \\ . \\ \tilde{\mu}_K^{\mathrm{T}}\end{bmatrix} \cdot a_r = \begin{bmatrix}\tilde{\mu}_1 & \tilde{\mu}_2 & \dots & \tilde{\mu}_K\end{bmatrix}\begin{bmatrix} w_{r1} & 0 & . & . & 0 \\ 0 & w_{r2} & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & 0 & . & . & w_{rK}\end{bmatrix}\begin{bmatrix}\tilde{o}_{kr} \\ \tilde{o}_{kr} \\ . \\ . \\ \tilde{o}_{kr}\end{bmatrix}$$

Let $\boldsymbol{W}_r$ be a K dimensional diagonal matrix with elements w_{rk}, $\tilde{\boldsymbol{O}}_r$ be a K dimensional vector with elements $\tilde{o}_{kr}$, and $\boldsymbol{M}$ be a K x D matrix with row $\tilde{\mu}_k^{\mathrm{T}}$.

$$\boldsymbol{M}^{\mathrm{T}}\boldsymbol{W}_r\boldsymbol{M}\cdot\boldsymbol{a}_r = \boldsymbol{M}^{\mathrm{T}}\boldsymbol{W}_r\tilde{\boldsymbol{O}}_r \tag{4.20}$$

or, the estimate $\hat{\boldsymbol{a}}_r$ is given by

$$\hat{\boldsymbol{a}}_r = (\boldsymbol{M}^{\mathrm{T}}\boldsymbol{W}_r\boldsymbol{M})^{-1}\boldsymbol{M}^{\mathrm{T}}\boldsymbol{W}_r\tilde{O}_r \tag{4.21}$$

We can see this as the weighted least squares estimation for the linear regression [42]

$$\tilde{\boldsymbol{O}}_r = \boldsymbol{M}\boldsymbol{a}_r + \boldsymbol{e}_r \tag{4.22}$$

where $\boldsymbol{e}_r$ is a K dimensional error vector.

Let's consider the eigenvectors and eigenvalues for the above case as we did in the previous section. Defining $\boldsymbol{V_M}$ and $\Lambda_{\boldsymbol{M}}$ as orthonormal and diagonal D x D matrices which contain the eigenvectors and eigenvalues of $(\boldsymbol{M}^{\mathrm{T}}\boldsymbol{W}_r\boldsymbol{M})$, *i.e.*

$$(\boldsymbol{M}^{\mathrm{T}}\boldsymbol{W}_r\boldsymbol{M})\boldsymbol{V_M} = \boldsymbol{V_M}\Lambda_{\boldsymbol{M}} \text{ and } \boldsymbol{V_M}\boldsymbol{V_M}^{\mathrm{T}} = \boldsymbol{I}_D$$

Using the new variables $\Omega \equiv \boldsymbol{M}\boldsymbol{V_M}$ and $\alpha_{\boldsymbol{r}} \equiv \boldsymbol{V}_{\boldsymbol{M}}^{\mathrm{T}}\boldsymbol{a}_r$, we can write Eq. (4.22) as

$$\tilde{\boldsymbol{O}}_r = \Omega\alpha_r + \boldsymbol{e}_r$$

and write Eq. (4.21) as

$$\boldsymbol{V_M}\hat{\boldsymbol{a}}_r = (\boldsymbol{V_M}\Lambda_{\boldsymbol{M}}\boldsymbol{V}_{\boldsymbol{M}}^{\mathrm{T}})^{-1}(\Omega\boldsymbol{V}_{\boldsymbol{M}}^{\mathrm{T}})^{\mathrm{T}}\boldsymbol{W}_r\tilde{\boldsymbol{O}}_r$$

This expression reduces to

$$\hat{\alpha}_r = \Lambda_{\boldsymbol{M}}^{-1}\Omega^{\mathrm{T}}\cdot\boldsymbol{W}_r\cdot\tilde{\boldsymbol{O}}_r \tag{4.23}$$

We can calculate the variance of the estimate $\hat{\alpha}_r$ in the same fashion as we did in the previous section.

$$\begin{aligned}\hat{\alpha}_r &= \Lambda_{\boldsymbol{M}}^{-1}\Omega^{\mathrm{T}}\cdot\boldsymbol{W}_r\cdot\tilde{\boldsymbol{O}}_r \\ &= \Lambda_{\boldsymbol{M}}^{-1}\Omega^{\mathrm{T}}\cdot\boldsymbol{W}_r\cdot(\Omega\alpha_r + \boldsymbol{e}_r)\end{aligned}$$

$$= \alpha_r + \Lambda_M^{-1}\Omega^{\mathrm{T}} W_r e_r$$

and,

$$\begin{aligned} var(\hat{\alpha}_r) &= E\{(\hat{\alpha}_r - \alpha_r)(\hat{\alpha}_r - \alpha_r)^{\mathrm{T}}\} \\ &= E\left\{(\Lambda_{\boldsymbol{M}}^{-1}\Omega^{\mathrm{T}}\boldsymbol{W}_r\boldsymbol{e}_r)(\Lambda_{\boldsymbol{M}}^{-1}\Omega^{\mathrm{T}}\boldsymbol{W}_r\boldsymbol{e}_r)^{\mathrm{T}}\right\} \\ &= \Lambda_{\boldsymbol{M}}^{-1}\Omega^{\mathrm{T}}\boldsymbol{W}_r \cdot E\{\boldsymbol{e}_r \cdot \boldsymbol{e}_r^{\mathrm{T}}\} \cdot \boldsymbol{W}_r\Omega\Lambda_{\boldsymbol{M}}^{-1} \end{aligned}$$

If we ignore the effect of different probability $\gamma(k)$ in $\boldsymbol{W}_r$, and assume $E\{\boldsymbol{e}_r \cdot \boldsymbol{e}_r^{\mathrm{T}}\} \cdot \boldsymbol{W}_r \approx \kappa \cdot \boldsymbol{I}_K$ where κ is a constant and $\boldsymbol{I}_K$ is the K x K identity matrix, the variance becomes

$$var(\hat{\alpha}_r) \approx \kappa \cdot \Lambda_{\boldsymbol{M}}^{-1} \tag{4.24}$$

Therefore the variance of the i^{th} components of the estimate $\hat{\alpha}_r$ is approximately inversely proportional to the corresponding eigenvalue λ_{Mj} of $(\boldsymbol{M}^{\mathrm{T}}\boldsymbol{W}_r\boldsymbol{M})$. As before, we can choose the largest $p < D$ principal components to reduce the variances of the estimates.

Fig. 4.2 shows a comparison of the variance of $\hat{\alpha}_r$ and the eigenvalues obtained from actual data. The eigenvalues are obtained from the baseline recognition model for the Wall Street Journal (WSJ) task. The inverses of the eigenvalues are shown in order of their magnitudes. The variances of the components of the MLLR matrix are also obtained from the same task. We estimate single-class MLLR matrices in the eigendomain from several different speakers and adaptation data, and calculate the variances of the components of the estimates. The variances of each component of the row vector $\hat{\alpha}_r$ of the MLLR matrix are averaged over all the rows, and shown in the order of the corresponding eigenvalues. Both values are normalized so that the largest values become one. This comparison confirms that the variances of the components the MLLR matrix are indeed inversely proportional to their eigenvalues.

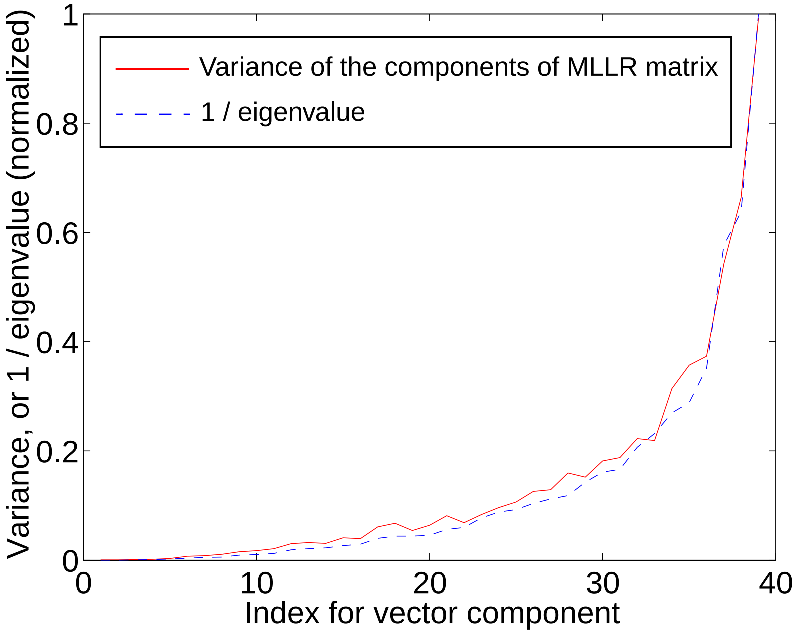


Figure 4.2 Comparison of the variance of the components of MLLR matrix and the inverse of its eigenvalues.

It should be noted that the matrix $\boldsymbol{W}_r$ includes the inverse variance σ_{kr}^{-2} as well as the probability $\gamma(k)$ in Eq. (4.11). The variances are different for each different r^{th} component, so in principle different eigenvectors should be used for the different row vectors $\boldsymbol{a}_r$ of matrix $\boldsymbol{A}$. Because the probabilities will change for different adaptation data, we should calculate new eigenvectors for each new speaker (or environment). However, the eigenvectors and eigenvalues from a small amount of adaptation data may not represent the statistics of the new speaker accurately. It also requires much computation because they need to be re-calculated for new adaptation data. Therefore, in this thesis, we pre-calculate the eigenvectors and eigenvalues, and use them for all adaptation data. We also use the same eigenvectors for each different row of matrix $\boldsymbol{A}$ for computational simplicity. We first calculate a global weighted covariance matrix Φ

$$\Phi = \sum_{r=1}^{D} (\boldsymbol{M}^{\mathrm{T}} \boldsymbol{W}_r \boldsymbol{M})$$

using the Gaussian mixture weights in the baseline speech recognition system instead of $\gamma(k)$. We then generate a single set of eigenvalues and eigenvectors from Φ.

4.4 Weighted Principal Component MLLR

Even though eliminating some of the less important components in PC-MLLR reduces the variance of the estimation error, it makes $\hat{\boldsymbol{a}}_r$ biased, which tends to reduce recognition accuracy. In this section we introduce a modification referred to as *Weighted Principal Component MLLR* (*WPC-MLLR*) [12] in an attempt to ameliorate this problem. WPC-MLLR applies weights to the MLLR estimates to reduce their mean square error.

Weighting the j^{th} component of $\hat{\alpha}_r$ by ω_j, we obtain

$$\hat{\alpha}'_{rj} \equiv \omega_j \hat{\alpha}_{rj} = \omega_j \alpha_{rj} + \omega_j \Lambda_M^{-1} \Omega^{\mathrm{T}} W_r e_r, \quad j = 1, 2, ..., D$$

Considering Eq. (4.24), the mean square error of $\hat{\alpha}'_{rj}$ is

$$\begin{aligned} MSE(\hat{\alpha}'_{rj}) &= E(\hat{\alpha}'_{rj} - \alpha_{rj})^2 \\ &\approx (\omega_j - 1)^2 \alpha_{rj}^2 + \omega_j^2 \kappa \lambda_{Mj}^{-1} \end{aligned}$$

and, the weight ω_j that minimizes the MSE can be obtained by solving

$$\frac{\partial}{\partial \omega_j} MSE(\hat{\alpha}'_{rj}) \approx 2(\omega_j - 1)\alpha_{rj}^2 + 2\omega_j \kappa \lambda_{Mj}^{-1} = 0 \qquad (4.25)$$

Hence, ω_j becomes

$$\omega_j = \frac{\lambda_{Mj}\alpha_{rj}^2}{\lambda_{Mj}\alpha_{rj}^2 + \kappa} = \frac{\lambda_{Mj}}{\lambda_{Mj} + \kappa / \alpha_{rj}^2}, \quad j = 1, 2, ..., D \qquad (4.26)$$

The weight ω_j approaches 1 for a large eigenvalue λ_{Mj} and approaches 0 for a small eigenvalue λ_{Mj}.

This is intuitively appealing because we would want to apply larger weights to components of $\hat{\alpha}_{rj}$ with smaller variance, and smaller weights to components with larger variance. In this method, instead of discarding the less significant components of $\hat{\alpha}_{rj}$, we use all components but with weighting. Unfortunately, we don't know the correct value of the parameter α_{rj}. We may use the estimated value $\hat{\alpha}_{rj}$, or the average value of $\hat{\alpha}_{rj}$ from prior experiments. In our experiment we ignore the difference of α_{rj}, and calculate the weight

$$\omega_j = \frac{\lambda_{Mj}}{\lambda_{Mj} + \kappa'} \tag{4.27}$$

We then normalize the weights so that the largest weight (ω_0) is always one.

$$\omega'_j = \omega_j / \omega_0, \quad j = 1, 2, ..., D \tag{4.28}$$

The value κ' is chosen as it produces the best recognition accuracy in the prior experiments.

The steps for the adaptation can be summarized as follows:

(1) Pre-calculate eigenvectors, eigenvalues, and weights from the baseline model

(2) Transform the baseline mean vectors by their eigenvectors using $\Omega = \boldsymbol{MV_M}$

(3) Estimate $\hat{\alpha}_r$ in the eigendomain

(4) Apply the weights to $\hat{\alpha}_r$, *i.e.* $\hat{\alpha}'_{rj} = \omega'_j \hat{\alpha}_{rj}$ for $r, j = 1, 2, ..., D$

(5) Re-calculate the shift term b_r with $\hat{\alpha}'_r$ using Eq. (4.15)

(6) Transform $\hat{\alpha}'_r$ into the original domain to produce the multiplication matrix $\boldsymbol{A}$ using $\boldsymbol{a}_r = \boldsymbol{V_M}\hat{\alpha}'_r$

(7) Adapt the baseline mean vectors by using $\hat{\mu}_k = \boldsymbol{A}\mu_k + \boldsymbol{b}$.

4.5 Experiments

We evaluate the developed algorithms using several corpora from the DARPA 1994 Wall Street Journal, the Telefónica, and the DARPA 1998 broadcast news tasks.

4.5.1 Non-native Speakers from the Wall Street Journal Task

As described in Section 2.4, sentences from the DARPA 1994 Wall Street Journal are used for the evaluation. Table 4.1 summarizes the word error rates obtained for Spoke 3 non-native speakers (s3-94) using each adaptation method. There are 10 speakers and about 20 test sentences each. For supervised adaptation, 1 or 3 adaptation sentences are used for each speaker. The adaptation sentences are separate data from the test data, and correctly transcribed by humans. The single-class full matrix ***A*** and shift vector ***b*** are estimated in MLLR because we use a small amount of adaptation data. The recognition tests are performed 4 times for each speaker after adaptation with different adaptation sentences, and average word error rates (WER) are shown in Table 4.1. In this experiment, PC-MLLR provides a relative improvement over conventional MLLR of 4.5% with 1 adaptation sentence and 6.1% with 3 adaptation sentences. WPC-MLLR provides greater improvement than PC-MLLR, resulting in relative improvements of 10.0% for both cases over conventional MLLR.

Adaptation Method	1 Adaptation Sentence	3 Adaptation Sentences
Baseline (unadapted)	27.3%	27.3%
Conventional MLLR	24.1%	23.1%
PC-MLLR	23.0% (4.5%)	21.7% (6.1%)
WPC-MLLR	21.7 (10.0%)%	20.8% (10.0%)

Table 4.1 Word error rates for s3-94 data after supervised adaptation using different MLLR methods (Relative percentage improvement over conventional MLLR is shown in parenthesis)

Fig. 4.3 depicts the word error rates (WER) plotted as a function of the number (p) of principal components used in PC-MLLR. As expected, the WER for PC-MLLR decreases and then increases as the number of components p increases. If p is too small, the estimates become highly biased, producing high WER. As the number of components increases to 39 (*i.e.* $p \rightarrow D$), the WER obtained with PC-MLLR increases, asymptoting to that obtained with conventional MLLR.

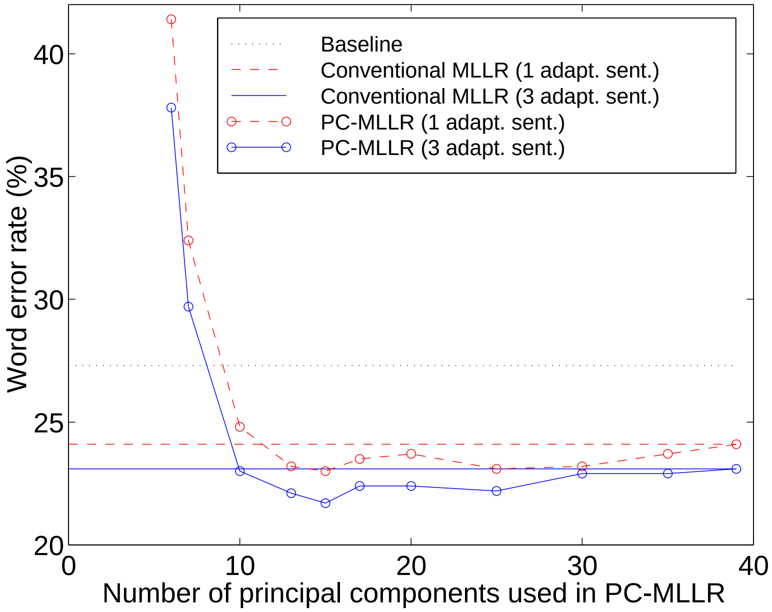


Figure 4.3 Word error rates as a function of the number of principal components used in PC-MLLR for s3-94 in supervised adaptation

Fig. 4.4 plots the ratio of sum of the p largest eigenvalues to the sum of all 39 eigenvalues. The optimum value of p will depend on the eigenvalue spread. In this experiment, we get the best recognition accuracy for PC-MLLR when p equals 15, with the ratio equal to 0.983. The ratio drops rapidly when p becomes smaller than 10, as does recognition accuracy.

Fig. 4.5 depicts word error rates (WER) plotted as a function of the average weight used in PC-MLLR. The weights are controlled by changing κ' in Eq. (4.27). Fig. 4.6 describes the relative weights of each vector component for various values of κ'. Larger weights are always applied to components corresponding to larger eigenvalues (lower indices in Fig. 4.6). Fig. 4.5 shows a similar pattern to that of Fig. 4.3 for

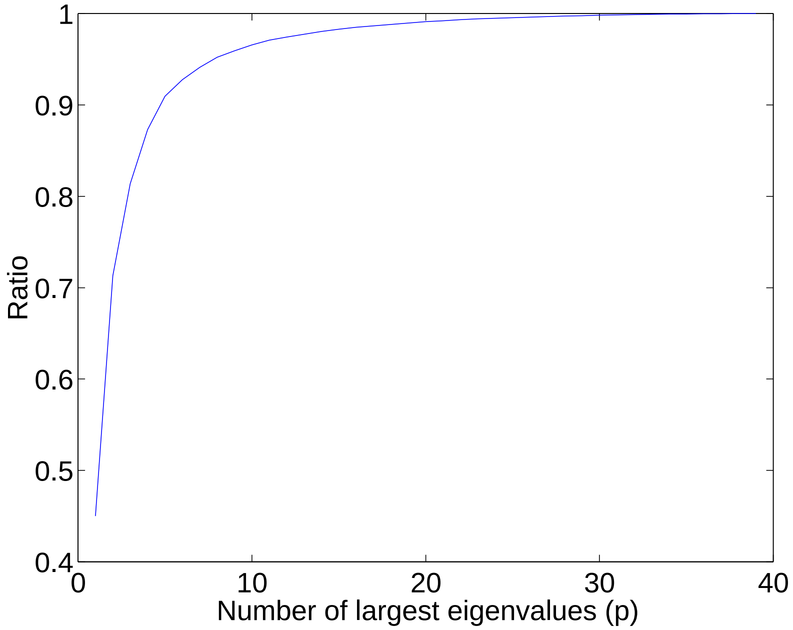


Figure 4.4 Ratio of the sum of the *p* largest eigenvalues to the sum of all 39 eigenvalues.

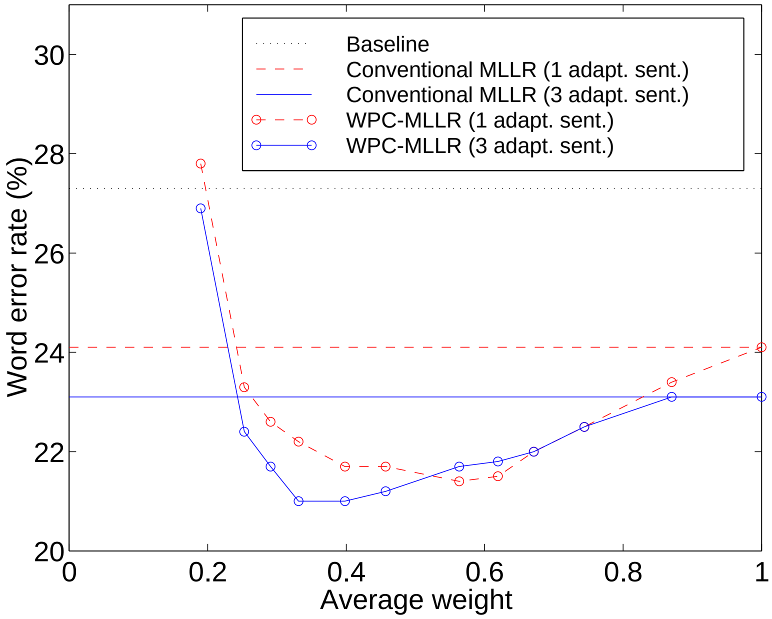


Figure 4.5 Word error rates as a function of the average weight used in WPC-MLLR for s3-94 in supervised adaptation

PC-MLLR, *i.e.* the WER becomes high for a small weight, and converges to the WER of conventional MLLR as the weight becomes one (equal weights).

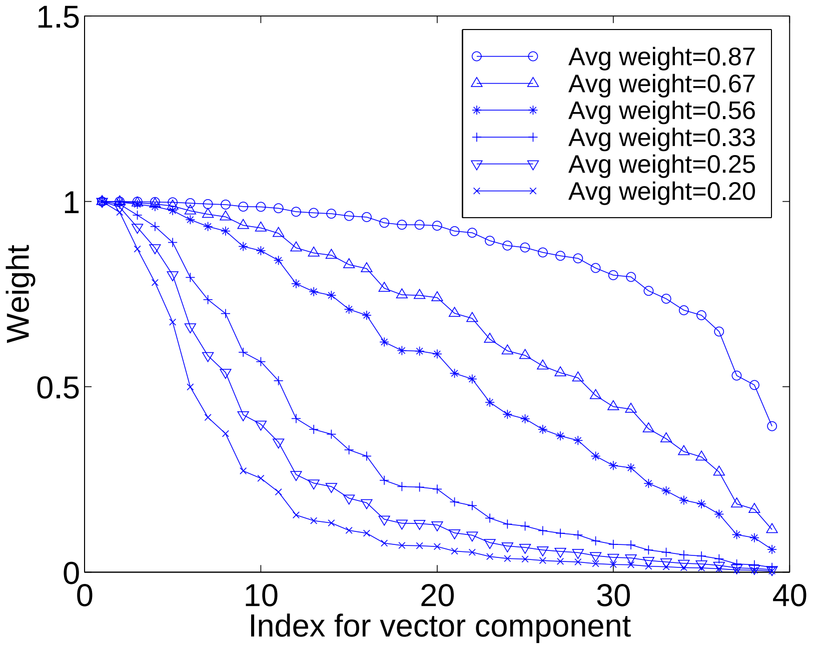


Figure 4.6 The shape of the weights used in WPC-MLLR

Table 4.2 summarizes the results for unsupervised adaptation with Spoke 3 data. In unsupervised adaptation, the test data are used for adaptation along with the recognized transcripts by the baseline system. Different number of sentences are used for adaptation. “1 test sentence” in Table 4.2 means that the adaptation is performed using a 1 test sentence at a time, and recognized the sentence again using the adapted model. In “3 test sentences”, 3 test sentences are used at a time.

Adaptation Method	1 Test Sentence	3 Test Sentences	20 Test Sentences
Baseline (unadapted)	27.3%	27.3%	27.3%
Conventional MLLR	26.7%	25.9%	23.9%
WPC-MLLR	21.4% (19.9%)	20.5% (20.8%)	20.1% (15.9%)

Table 4.2 Word error rates for s3-94 data after unsupervised adaptation using WPC-MLLR (Relative percentage improvement over conventional MLLR is shown in parenthesis)

WPC-MLLR in this case provides more improvements than in the supervised adaptation in Table 4.1. We think it is because conventional MLLR has larger variances of the estimates and provides higher error rates in unsupervised adaptation than in supervised adaptation. WPC-MLLR provides a smaller improvement in the "20 test sentence" case than in the "1 or 3 test sentence" cases as conventional MLLR provides better accuracy in the "20 test sentence" case.

As noted in Section 4.3, we pre-calculated eigenvectors from the baseline recognition model and use the same eigenvectors for different speakers in these experiments. In other experiments using different eigenvectors considering the *a posteriori* probability $\gamma(k)$ from the adaptation data, we observed similar or a little worse recognition accuracy. This may be because the adaptation data are insufficient to estimate proper eigenvectors.

4.5.2 Native Speakers from the Wall Street Journal Task

Table 4.3 shows word error rates for native speakers from Spoke 0 at the 1994 DARPA WSJ task (s0-94) after supervised adaptation with 5 sentences. WPC-MLLR provides 6.0% of improvement in word error rate compared to conventional MLLR, which is less improvement than was observed for non-native speakers. We think that this is because native speakers are much closer to the baseline model than non-native speakers, so speaker adaptation has less effect.

Note that a small language model weight is used for the s0-94 data to emphasize the effect of adaptation on the acoustic models. When a larger language model weight is used, the word error rate for the baseline model becomes 5.9%, and conventional MLLR provides 5.7% of word error rate. Unfortunately WPC-MLLR does not provide further improvement in our experiment. The language model may be too dominant to get further improvement in this case.

Adaptation Method	**5 Adaptation Sentences**
Baseline (unadapted)	21.9%
Conventional MLLR	18.3%
PC-MLLR	18.0% (1.6%)
WPC-MLLR	17.2% (6.0%)

Table 4.3 Word error rates for s0-94 data after supervised adaptation using different MLLRs (Relative percentage improvement over conventional MLLR is shown in parenthesis)

4.5.3 The Telefónica (TID) Corpus

The experimental results with the TID data are summarized in Table 4.4. We cluster test data for each speaker. Each speaker has 5.55 utterances on average, and each utterance is about 2-4 seconds long. Adaptation is done in unsupervised mode. In this task, the statistical significance test using tools developed at NIST [19] indicates that if two algorithms produce 0.2% absolute or larger difference in WER then they are statistically different [25]. Even though the improvements after adaptation is small, WPC-MLLR provides a further improvement over conventional MLLR.

Adaptation Method	5.55 Test Utterances
Baseline (unadapted)	6.9%
Conventional MLLR	6.8%
WPC-MLLR	6.5% (4.4%)

Table 4.4 Word error rates for TID data after unsupervised adaptation using WPC-MLLR (Relative percentage improvement over conventional MLLR is shown in parenthesis)

4.5.4 The 1998 DARPA Broadcast News Task (Hub 4)

As described in Chapter 2, Hub 4 corpus is a mixture of several different conditions and does not have separate adaptation data. First the speech data are segmented automatically, and clustered into similar conditions. The adaptation is then done in unsupervised mode for each cluster. Table 4.5 shows the experimental results. Conventional MLLR provides only a 5.8% improvement over the baseline system, and WPC-MLLR does not provide further improvement. This may be because of several factors. This corpus is much more complex than the previous corpora, containing larger vocabulary, a mixture of different conditions, and incorrect clustering. It seems that WPC-MLLR is less effective on a complex task like this.

Adaptation Method	WER
Baseline (unadapted)	20.8%
Conventional MLLR	19.6%
WPC-MLLR	19.6%

Table 4.5 Word error rates for Hub 4 data after unsupervised adaptation using WPC-MLLR

4.6 Summary

In this chapter, we applied principal component analysis to the MLLR framework for speaker adaptation (PC-MLLR). By eliminating highly variable components and choosing the *p* principal components corresponding to the largest eigenvalues we can reduce the variance of the estimates and improve speech recognition accuracy. The best value for *p* depends on the eigenvalues. Choosing fewer principal components increases the bias of the estimates which can reduce recognition accuracy. To compensate for this problem, we developed Weighted Principal Component MLLR (WPC-MLLR). We applied weights to the MLLR estimates so that they minimize the mean square error. WPC-MLLR provided better WER than PC-MLLR. However, it does not provide further improvement over conventional MLLR on Hub 4 corpus.

Chapter 5
Inter-Class MLLR

5.1 Introduction

In this chapter we describe *inter-class MLLR* which utilizes relationships among different transformation functions to achieve more reliable estimates of MLLR parameters across multiple classes with a small amount of adaptation data. We introduce the *inter-class transformation* which provides *a priori* knowledge to relate transformation functions between the target and neighboring classes. The inter-class transformations modify the baseline mean vectors in the neighboring classes so that the adaptation of the neighboring classes are formulated by the transformation function of the target class. Therefore several neighboring classes as well as the target class can be considered to estimate the transformation function of the target class. We apply this procedure to the MLLR framework and develop inter-class MLLR. This procedure also can be extended to any form of transformation-based adaptation.

As we explained in Chapter 2, conventional transformation-based adaptation typically proceeds as follows. The target parameters (Gaussian mean vectors in this thesis) are classified into transformation classes. The target parameters in a transformation class are assumed to be updated by a same transformation function. A form for the transformation function is pre-chosen (usually a simple linear function). The parameters of the transformation function are then estimated for each class by using the adaptation data associated with the class. Finally, the model parameters in each class are modified by applying the estimated class-dependent transformation function to them. In this process, each class is considered independently and does not affect each other.

As an example, suppose that the Gaussian means underlying a set of mixture densities are as shown by the dots in Fig. 5.1. These means can be classified either as a single class (Method I) or as two classes (or multiple classes) (Method II). For Method I we assume that all the Gaussian means (μ_k) are transformed to the adapted means $(\hat{\mu}_k)$ by a single transformation function $f(\cdot)$:

$$\hat{\mu}_k = f(\mu_k), \qquad k \in \text{all Gaussians} \tag{5.1}$$

For Method II we assume two functions $f_1(\cdot)$ and $f_2(\cdot)$, one for each class:

$$\hat{\mu}_k = f_1(\mu_k), \qquad k \in \text{Class 1} \tag{5.2}$$

$$\hat{\mu}_k = f_2(\mu_k), \qquad k \in \text{Class 2} \tag{5.3}$$

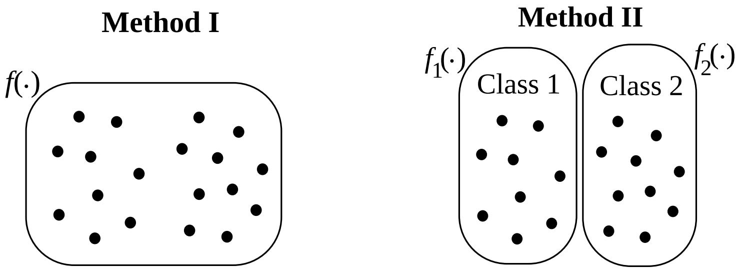


Figure 5.1 Classification of Gaussian mean vectors as either a single class (Method I) or as two classes (Method II)

If only a small amount of adaptation data is present, single-class adaptation (Method I) may be more effective than multi-class adaptation (Method II). With multi-class adaptation, we have a smaller amount of data available per class, so we may not obtain reliable estimates of the functions $f_1(\cdot)$ and $f_2(\cdot)$. In single-class adaptation, all data are used to estimate the function $f(\cdot)$ so that function can be estimated more reliably. However multi-class adaptation may be more effective than single-class adaptation, if enough data are available to estimate the transformation function reliably for each class because the parameters of each class are modified in a fashion that is appropriate for the individual class. Therefore there is a trade-off between reliability of estimation versus detail with which the transformation is performed. The optimum number of classes will depend on the amount of available adaptation data.

Gales [16] used regression class trees to find the optimal number of regression classes in conventional MLLR. He developed a binary regression class tree and dynamically selected the classes so that either each class had sufficient data or the overall likelihood of the adaptation data was maximized. He also described a method in which MLLR was applied to the root node and the sub-nodes of the regression class tree iteratively. Chien *et al.* [9] described a hierarchical adaptation procedure that used a tree structure to control the transformation sharing and to obtain reliable transformation parameters.

As we have noted above, new parameters are estimated independently for each class in conventional multi-class regression. Nevertheless, under many circumstances, estimation accuracy (and hence recognition accuracy) can be improved if information is shared across parameter classes. Correlations among model parameters (usually Gaussian mean vectors), for example, has been used by several research groups to improve estimation of parameter values with sparse data. Most previous work using parameter correlation has been done within a Bayesian framework as described in Section 2.6. Within the MLLR framework, Bocchieri *et al.* [4] used correlation between the shift terms (*i.e.* the $\boldsymbol{b}$ term in $\hat{\mu}_k = \boldsymbol{A}\mu_k + \boldsymbol{b}$) to refine further the transformation function. Correlation information was limited to the shift terms in their work, and the relation between the multiplication term (*i.e.* the $\boldsymbol{A}$ term in $\hat{\mu}_k = \boldsymbol{A}\mu_k + \boldsymbol{b}$) was not considered even though the multiplication term is more important in MLLR.

In this thesis we utilize the inter-class relationships among transformation functions, not model parameters. In the following sections, we will explain about using inter-class transformations as *a priori* knowledge in transformation-based adaptation, then apply this idea on MLLR to obtain more reliable estimates of MLLR parameters including the multiplication term as well as the shift term across multiple classes.

5.2 The Inter-class Transformation

In this section we introduce the inter-class transformation which relates transformation functions between the target and neighboring classes. We explain the inter-class transformation using the example in the previous section. We also illustrate the usage of the inter-class transformation by a series of simulations with artificial data.

Let us consider some of the issues involving inter-class relationships with transformation-based adaptation using “Method II” in Fig. 5.1 as an example. Let’s think of estimating $f_1(\cdot)$ for Class 1 (*the target class*). We assume there is a known relation between the two classes, which is given by the *inter-class transformation function* $g_{12}(\cdot)$, that we can rewrite the transformation of Class 2 (*the neighboring class*) in Eq. (5.3) using the function $f_1(\cdot)$ of Class 1 and a modified Gaussian mean $\mu_k^{(12)} \equiv g_{12}(\mu_k)$.

$$\hat{\mu}_k = f_1(\mu_k^{(12)}), \qquad k \in \text{Class 2} \tag{5.4}$$

We can say that the two functions $f_1(\cdot)$ and $f_2(\cdot)$ are related by the inter-class transformation $g_{12}(\cdot)$ as shown in Fig. 5.2. Since the transformation for Class 2 is now formulated by $f_1(\cdot)$, data from Class 2 can contribute to the estimation of $f_1(\cdot)$. Therefore we can estimate $f_1(\cdot)$ using all the data from Classes 1 and 2 with Eqs. (5.2) and (5.4).

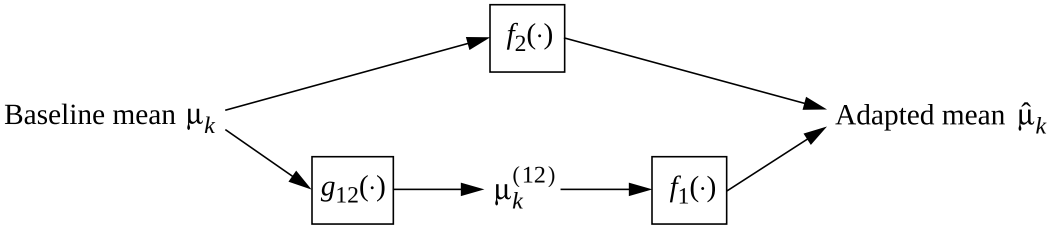


Figure 5.2 A baseline mean vector is transformed to an adapted mean vector by $f_2(\cdot)$, or the combination of $g_{12}(\cdot)$ and $f_1(\cdot)$. We can say that $f_1(\cdot)$ and $f_2(\cdot)$ are related by inter-class transformation $g_{12}(\cdot)$.

This approach can also be applied to the estimation of the function $f_2(\cdot)$ of Class 2 (Now this is *the target class*). In this case we use the *inter-class transformation function* $g_{21}(\cdot)$ which modifies the baseline mean vectors in Class 1 (*the neighboring class*) so that the adaptation for Class 1 can be formulated by $f_2(\cdot)$, *i.e.* for $\mu_k^{(21)} \equiv g_{21}(\mu_k)$,

$$\hat{\mu}_k = f_2(\mu_k^{(21)}), \qquad k \in \text{Class 1} \tag{5.5}$$

Now we can estimate $f_2(\cdot)$ using all the data from Classes 1 and 2 with Eqs. (5.3) and (5.5).

The functions $f_1(\cdot)$ and $f_2(\cdot)$ estimated by this method will be more reliable than those estimated by conventional multi-class methods because more adaptation data are used. They also provide more effective adaptation to each class than the class-independent function $f(\cdot)$ used in Method I because while the adaptation is primarily based on observations within a given class, it benefits from relevant information from other classes as well. This method can be extended to any form of transformation-based adaptation.

Before adaptation, multiple transformation classes are defined, and the inter-class transformation functions are estimated using training data. In adaptation, the model parameters (Gaussian mean vectors in the above example) in a target class are used as is, and the model parameters in other *neighboring* classes are modified by the inter-class transformation functions. The incoming adaptation data from all classes are then combined to estimate the transformation functions for the target class. This process is repeated for all other target classes.

In this process, the number of neighboring classes to be used depends on the amount of adaptation data. The neighboring classes are ranked in order of their "*closeness*" to the target class. Adaptation data are selected from classes of decreasing proximity to the target class until there are sufficient data to estimate the target function. If only a very small amount of data is available, then all neighboring classes may be used. As more data becomes available the number of neighboring classes used declines. In the limit no neighboring classes are used, and inter-class adaptation asymptotes to conventional multi-class adaptation. Choosing the identity function as the inter-class function is equivalent to conventional single-class adaptation.

The use of the inter-class transformation is illustrated by a series of simulations with artificial data shown in Fig. 5.3. There are two sets of sample pairs, (x_{1i}, y_{1i}) shown as an 'o' for Class 1, (x_{2i}, y_{2i}) shown as an '+' for Class 2. Sample pairs in each class are related by linear regression as illustrated in Fig. 5.3 (a).

$$y_{1i} = f_1(x_{1i}) = a_1 x_{1i} + b_1, \qquad \text{for Class 1}$$

$$y_{2i} = f_2(x_{2i}) = a_2 x_{2i} + b_2, \qquad \text{for Class 2}$$

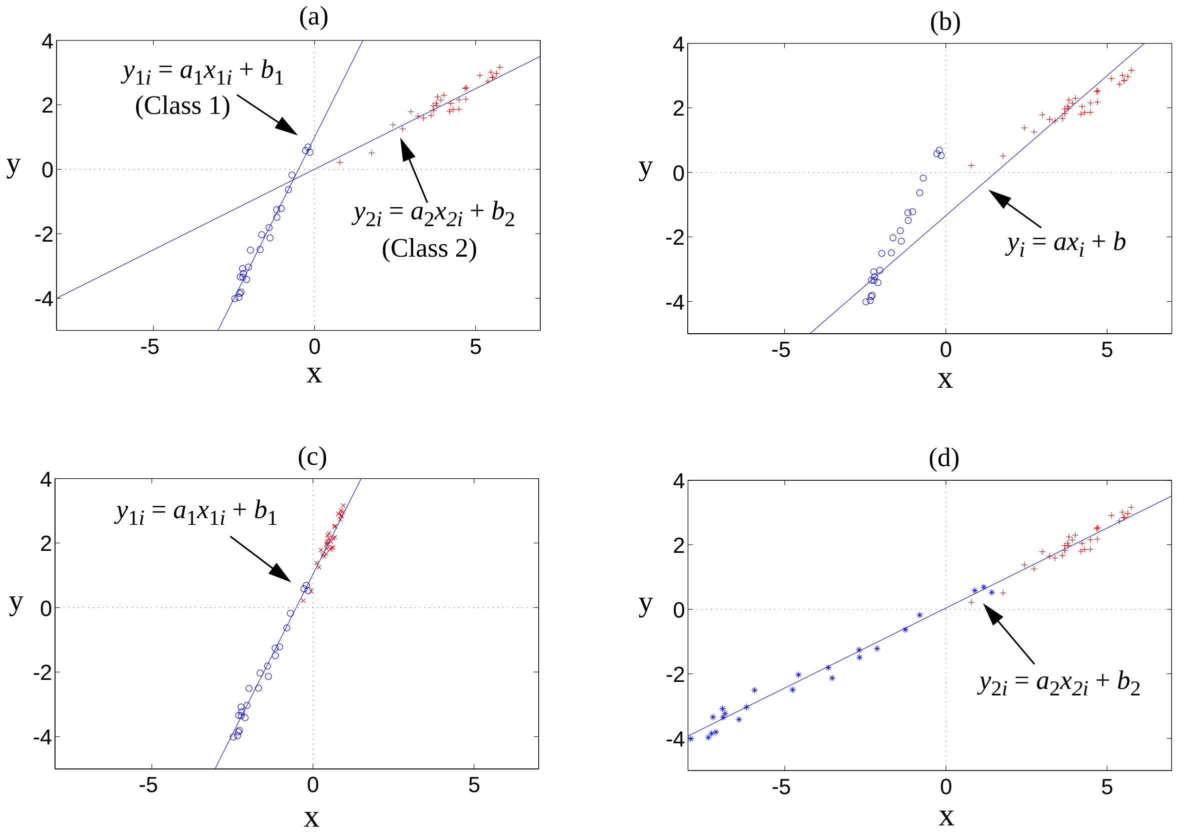


Figure 5.3 Simulations of linear regression (a) Original Class 1 and Class 2, (b) Single class, (c) Inter-class transformation for Class 1 (Class 2 samples are moved), (d) Inter-class transformation for Class 2 (Class 1 samples are moved)

We would like to estimate the regression functions $f_1(\cdot)$ and $f_2(\cdot)$ using the available samples. If all the samples from Class 1 and Class 2 are considered together as a single class, then a single function $f(\cdot)$ will be given as Fig. 5.3 (b). We can see that the estimates of (a, b) are different from both (a_1, b_1) and (a_2, b_2).

$$
\begin{aligned}
y_{1i} &= f(x_{1i}) = a x_{1i} + b_1, && \text{for Class 1} \\
y_{2i} &= f(x_{2i}) = a x_{2i} + b_2, && \text{for Class 2}
\end{aligned}
$$

If inter-class relationships are considered, better estimates can be obtained. Let's assume that the *inter-class transformation function* $g_{12}(\cdot)$ is known to be in the form of a linear regression.

$$x_{2i}^{(12)} = g_{12}(x_{2i}) = t_{12}x_{2i} + d_{12}, \qquad \text{for Class 2}$$

Then x_{2i} in Class 2 is transformed to $x_{2i}^{(12)}$ as shown ‘x’ in Fig. 5.3 (c), and $f_1(\cdot)$ can be estimated using all the samples. We can see that the estimated regression line in Fig. 5.3 (c) is very close to the original regression line in Fig. 5.3 (a). The estimation of $f_2(\cdot)$ can be done in same way. Now x_{1i} is transformed to $x_{1i}^{(21)}$ by $g_{21}(\cdot)$ as shown ‘*’ in Fig. 5.3 (d).

$$x_{1i}^{(21)} = g_{21}(x_{1i}) = t_{21}x_{1i} + d_{21}, \qquad \text{for Class 1}$$

The actual values used in this example are shown in Table 5.1.

Fig. 5.4 illustrates another advantage of the inter-class transformations, also using simulated data. We assume that there are 5 classes and that each class has 5 Gaussians. We have adaptation samples from new Gaussians whose mean values are related to the original mean values by linear regression (LR) with known inter-class functions. New Gaussian mean values were estimated using linear regression for the conventional single-class and multi-class adaptation procedures, as well as for inter-class adaptation. We calculated the mean square estimation error (MSE) as a function of the number of available samples. As can be seen in Fig. 5.4, the inter-class LR always provides lower MSE than the other methods in this simulation. We note that while single-class LR provides better MSE than multi-class LR when the number of samples is small, multi-class LR surpasses it as the number of samples increases.

Variables	Values	Variables	Values
a_1	2	b_1	1
a_2	0.5	b_2	0
t_{12}	0.25	d_{12}	-0.5
t_{21}	4	d_{21}	4

Table 5.1 The actual values used in the simulation in Fig. 5.3

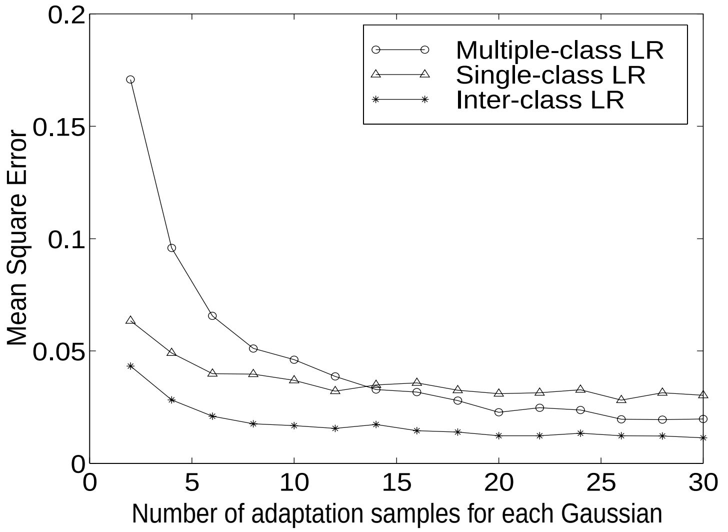


Figure 5.4 Mean-square errors from simulated estimates of Gaussian means using single-class, multi-class, and inter-class LR.

5.3 Inter-Class MLLR

We apply the method described in the previous section to the MLLR framework, and develop a new adaptation procedure which is called *inter-class MLLR* [13]. In inter-class MLLR the inter-class transformation is given by another linear regression. The inter-class transformation provides *a priori* knowledge among the regression classes, and helps to estimate the MLLR parameters.

Consider multi-class MLLR where we try to estimate the MLLR parameters $(\boldsymbol{A}_m, \boldsymbol{b}_m)$ for the regression class m. In this section we first explain how to estimate $(\boldsymbol{A}_m, \boldsymbol{b}_m)$ using the inter-class transformations. We then explain how to estimate the inter-class transformations from training data. We also describe the experimental results to evaluate inter-class MLLR.

5.3.1 Estimation of the MLLR Parameters ($\boldsymbol{A}_m$, $\boldsymbol{b}_m$)

In this subsection we explain how to estimate the MLLR parameters (the multiplication matrix $\boldsymbol{A}_m$ and the shift vector $\boldsymbol{b}_m$) for a regression class m (*the target class*) assuming the inter-class transformations are known.

Let's first consider conventional multi-class MLLR. We estimate the MLLR parameters **(**$\boldsymbol{A}_m$, $\boldsymbol{b}_m$) using the following equation with adaptation data from the class m only.

$$\hat{\mu}_k = \boldsymbol{A}_m \mu_k + \boldsymbol{b}_m, \qquad k \in \text{Class } m \tag{5.6}$$

As shown in Section 3.2, ($\boldsymbol{A}_m$, $\boldsymbol{b}_m$) are estimated by maximizing the likelihood in Class m, or minimizing $Q_{\mathrm{C}}(m)$.

$$Q_{\mathrm{C}}(m) = \sum_{k \in \text{ Class } m} \sum_t \gamma_t(k)(\boldsymbol{o}_t - \boldsymbol{A}_m \mu_k - \boldsymbol{b}_m)^{\mathrm{T}} \boldsymbol{C}_k^{-1} (\boldsymbol{o}_t - \boldsymbol{A}_m \mu_k - \boldsymbol{b}_m) \tag{5.7}$$

where $\gamma_t(k)$ is the *a posteriori* probability, $\boldsymbol{o}_t$ is the input feature vector at time t (adaptation data), and $\boldsymbol{C}_k$ is the covariance matrix of Gaussian k.

Consider the neighboring class $n \neq m$. In conventional MLLR, the MLLR class n has its own MLLR parameters ($\boldsymbol{A}_n$, $\boldsymbol{b}_n$) which are independent of the target class m.

$$\hat{\mu}_k = \boldsymbol{A}_n \mu_k + \boldsymbol{b}_n, \qquad k \in \text{Class } n \tag{5.8}$$

In inter-class MLLR, we assume that the inter-class transformation function $g_{mn}(\cdot)$ is given by a second linear regression with ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$), which relates the neighboring class n to the target class m with the modified mean vector $\mu_k^{(mn)}$.

$$\mu_k^{(mn)} \equiv \boldsymbol{T}_{mn} \mu_k + \boldsymbol{d}_{mn}, \qquad k \in \text{Class } n \tag{5.9}$$

Rewriting Eq. (5.8) with $\mu_k^{(mn)}$ and the MLLR parameters ($\boldsymbol{A}_m$, $\boldsymbol{b}_m$) of the target class m, we assume

$$\hat{\mu}_k = \boldsymbol{A}_m(\boldsymbol{T}_{mn} \mu_k + \boldsymbol{d}_{mn}) + \boldsymbol{b}_m$$

$$= \boldsymbol{A}_m \mu_k^{(mn)} + \boldsymbol{b}_m \qquad\qquad k \in \text{Class } n \tag{5.10}$$

(*A*$_m$, *b*$_m$) are the unknown parameters in Eq. (5.10). They can be estimated from the neighboring class *n* using $Q_n(m)$ which is similar to $Q_{\mathrm{C}}(m)$ except $\mu_k^{(mn)}$ and the class *n*.

$$Q_n(m) = \sum_{k \in \text{Class } n} \sum_t \gamma_t(k)(\boldsymbol{o}_t - \boldsymbol{A}_m \mu_k^{(mn)} - \boldsymbol{b}_m)^{\mathrm{T}} \boldsymbol{C}_k^{-1} (\boldsymbol{o}_t - \boldsymbol{A}_m \mu_k^{(mn)} - \boldsymbol{b}_m) \tag{5.11}$$

Now, considering $Q_{\mathrm{C}}(m)$ and $Q_n(m)$, **(*A*$_m$, *b*$_m$)** are estimated by maximizing the overall likelihood from the target class *m* and neighboring classes *n*, or by minimizing $Q_{\mathrm{I}}(m)$.

$$\begin{aligned} Q_{\mathrm{I}}(m) = & \sum_{k \in \text{Class } m} \sum_t \gamma_t(k)(\boldsymbol{o}_t - \boldsymbol{A}_m \mu_k - \boldsymbol{b}_m)^{\mathrm{T}} \boldsymbol{C}_k^{-1} (\boldsymbol{o}_t - \boldsymbol{A}_m \mu_k - \boldsymbol{b}_m) \\ & + \sum_{n \in \text{Neighbors}} \sum_{k \in \text{Class } n} \sum_t \gamma_t(k)(\boldsymbol{o}_t - \boldsymbol{A}_m \mu_k^{(mn)} - \boldsymbol{b}_m)^{\mathrm{T}} \boldsymbol{C}_k^{-1} (\boldsymbol{o}_t - \boldsymbol{A}_m \mu_k^{(mn)} - \boldsymbol{b}_m) \end{aligned} \tag{5.12}$$

or,

$$Q_{\mathrm{I}}(m) = Q_{\mathrm{C}}(m) + \sum_{n \in \text{Neighbors}} Q_n(m) \tag{5.13}$$

where $Q_{\mathrm{C}}(m)$ is the contribution of the target class *m*, and $Q_n(m)$ is the contribution of the neighboring class *n*. This enables us to obtain more reliable estimates of (*A*$_m$, *b*$_m$) while preserving the characteristics of the target class *m*.

5.3.2 Estimation of the Inter-Class Transformation Parameters (*T*$_{mn}$, *d*$_{mn}$)

In the previous subsection we assume that the inter-class transformation parameters (*T*$_{mn}$, *d*$_{mn}$) are known. (*T*$_{mn}$, *d*$_{mn}$) can be estimated from training data before adaptation. In this subsection we explain how to estimate (*T*$_{mn}$, *d*$_{mn}$).

Let (*A*$_{m,s}$, *b*$_{m,s}$) be the speaker-dependent MLLR parameters and $\hat{\mu}_{k,s}$ be the adapted mean vector for the target class *m* and a training speaker *s*.

$$\hat{\mu}_{k,s} = \boldsymbol{A}_{m,s}\mu_k + \boldsymbol{b}_{m,s}, \qquad k \in \text{Class } m \tag{5.14}$$

We assume that there are several training speakers with sufficient data to obtain reliable estimates of ($\boldsymbol{A}_{m,s}$, $\boldsymbol{b}_{m,s}$) for each regression class *m* and the training speaker *s*. The MLLR parameters ($\boldsymbol{A}_{m,s}$, $\boldsymbol{b}_{m,s}$) are first estimated from training data using conventional multi-class MLLR. The inter-class transformation parameters ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) are then estimated using ($\boldsymbol{A}_{m,s}$, $\boldsymbol{b}_{m,s}$).

Rewriting (5.10) for the training speaker *s*, we obtain

$$\hat{\mu}_{k,s} = \boldsymbol{A}_{m,s}(\boldsymbol{T}_{mn}\mu_k + \boldsymbol{d}_{mn}) + \boldsymbol{b}_{m,s}, \qquad k \in \text{Class } n, \quad s \in \text{Training speakers} \tag{5.15}$$

where ($\boldsymbol{A}_{m,s}$, $\boldsymbol{b}_{m,s}$) are known parameters. Moving ($\boldsymbol{A}_{m,s}$, $\boldsymbol{b}_{m,s}$) to the left-hand side, and defining

$$\hat{\mu}_{k,s}^{(mn)} \equiv \boldsymbol{A}_{m,s}^{-1}(\hat{\mu}_{k,s} - \boldsymbol{b}_{m,s}) \tag{5.16}$$

Eq. (5.15) becomes

$$\hat{\mu}_{k,s}^{(mn)} = \boldsymbol{T}_{mn}\mu_k + \boldsymbol{d}_{mn}, \qquad k \in \text{Class } n, \quad s \in \text{Training speakers} \tag{5.17}$$

If we let $\hat{\mu}_{k,s}$ be the adapted mean vector for the training data $\boldsymbol{o}_{t,s}$ from speaker *s*, then $\hat{\mu}_{k,s}^{(mn)}$ will be the adapted mean vector for the modified training data $\boldsymbol{o}_{t,s}^{(mn)} \equiv \boldsymbol{A}_{m,s}^{-1}(o_t - \boldsymbol{b}_{m,s})$. Eq. (5.17) can be considered as conventional MLLR for $\boldsymbol{o}_{t,s}^{(mn)}$ with the MLLR parameters ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$). Therefore ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) can be estimated using conventional MLLR, or by minimizing $Q_{\mathrm{T}}(mn)$ with all training speakers' data.

$$Q_{\mathrm{T}}(mn) = \sum_{s}\sum_{k \in \text{Class } n}\sum_{t} \gamma_{t,s}(k)(\boldsymbol{o}_{t,s}^{(mn)} - \boldsymbol{T}_{mn}\mu_k - \boldsymbol{d}_{mn})^{\mathrm{T}}\boldsymbol{C}_k^{-1}(\boldsymbol{o}_{t,s}^{(mn)} - \boldsymbol{T}_{mn}\mu_k - \boldsymbol{d}_{mn}) \tag{5.18}$$

($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) are estimated for each the target class *m* and the neighboring class *n*. In this procedure, we assume that the baseline variances are unchanged. These ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) provide *a priori* knowledge between the target class *m* and the neighboring class *n* independent of speaker. Once ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) are estimated, they are used in Eqs. (5.9) and (5.13) for adaptation to a new speaker.

$(\boldsymbol{T}_{mn}, \boldsymbol{d}_{mn})$ can also be estimated in different way. Let $(\boldsymbol{T}_{mn,s}, \boldsymbol{d}_{mn,s})$ be the speaker-dependent inter-class transformation parameters for a training speaker s. $(\boldsymbol{T}_{mn,s}, \boldsymbol{d}_{mn,s})$ are estimated in the same way as above, except that the training data are used only from a single speaker s, not from all training speakers. The $(\boldsymbol{T}_{mn,s}, \boldsymbol{d}_{mn,s})$ are then averaged over all speakers to get the global $(\boldsymbol{T}_{mn}, \boldsymbol{d}_{mn})$.

$$(\boldsymbol{T}_{mn}, \boldsymbol{d}_{mn}) = \underset{s \in \text{ All speakers}}{\boldsymbol{average}} \; (\boldsymbol{T}_{mn,\, s}, \boldsymbol{d}_{mn,\, s}) \tag{5.19}$$

5.3.3 Experiments with Non-Native Speakers from the 1994 DARPA WSJ Task

We evaluated inter-class MLLR using the Spoke 3 data (s3-94) in the 1994 DARPA Wall Street Journal (WSJ) evaluation as before. There are 10 non-native speakers with 20 test sentences each. The recognition tests were performed four times with different adaptation data from each speaker. Table 5.2 shows the 13 phonetic-based regression classes used for inter-class MLLR, which are similar to those used by Leggetter [39]. The recognition models for silence and noise are not considered for adaptation. The inter-class transformation functions were trained using adaptation data from other 9 test speakers (s3_9) using Eq. (5.18). Weighted principal component MLLR was used to obtain more reliable estimates of the speaker-dependent regression parameters $\boldsymbol{A}_{m,\, s}$ and $\boldsymbol{b}_{m,\, s}$ in Eq. (5.15) before $\boldsymbol{T}_{mn}$ and $\boldsymbol{d}_{mn}$ are estimated.

Table 5.3 and Fig. 5.5 summarize the word error rates (WER) obtained using the three types of inter-class MLLR along with conventional single-class MLLR in supervised adaptation. $\boldsymbol{T}_{mn}$ is assumed to be a full matrix (which allows the parameters to be rotated, scaled, and shifted), diagonal matrix (which allows them to be scaled and shifted only), or the identity matrix (which allows them to be shifted only). The shift vector $\boldsymbol{d}_{mn}$ is used in all three cases. However, the full matrix $\boldsymbol{A}$ and shift vector $\boldsymbol{b}$ are used in all cases, for both conventional and inter-class MLLR. All the neighboring classes are considered in estimating each target class in inter-class MLLR because only small amounts of adaptation data are used. Silence and noise phones are not adapted in these experiments.

Class	**Members**
Very front vowels	IH, IX, IY
Near front vowels	AE, EH
Front dipthongs	AY, EY
Back dipthongs	AW, OW, OY
Near back vowels	AA, AH, UH, ER, AXR
Very back vowels	AO, AX, UW
Liquids	L, R, W, Y
Nasals	M, N, NG
Strong fricatives	DH, JH, TS, V, Z, ZH
Weak fricatives	CH, F, HH, S, SH, TH
Closures	BD, DD, GD, KD, PD, TD
Unvoiced stops	K, P, T
Voiced stops	B, D, DX, G

Table 5.2 Phonetic-based regression classes used in inter-class MLLR (from Leggetter [39] with minor modifications)

Inter-class MLLR with a full matrix $\boldsymbol{T}_{mn}$ and shift vector $\boldsymbol{d}_{mn}$ reduces the WER by 15.4% with 1 adaptation sentence and 15.2% with 3 adaptation sentences over conventional MLLR, which is greater than the reduction from other types of inter-class MLLR. These improvements are statistically significant according to the matched pair sentence segment (word error) test from NIST [19]. The improvement in WER from adaptation is less than that reported by other sites in the 1994 DARPA evaluation primarily because much smaller amounts of adaptation data are used in this experiment.

Adaptation Method	**1 Adaptation Sentence**	**3 Adaptation Sentences**
Baseline (unadapted)	27.3%	27.3%
Conventional MLLR (one class)	24.1%	23.1%
Inter-class MLLR (shift only)	25.1% (-4.1%)	21.8% (5.6%)
Inter-class MLLR (diag. + shift)	21.4% (11.2%)	20.4% (11.7%)
Inter-class MLLR (full + shift)	20.4% (15.4%)	19.6% (15.2%)

Table 5.3 Word error rates for non-native speakers (s3-94) after supervised adaptation using inter-class MLLR (Relative percentage improvement over conventional MLLR is shown in parenthesis)

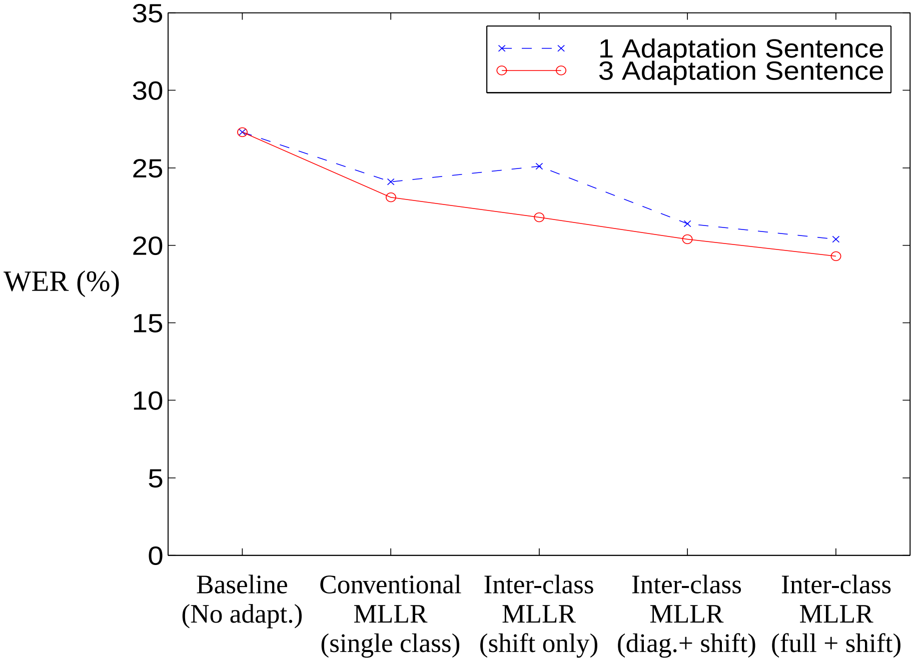


Figure 5.5 Word error rates for non-native speakers (s3-94) after supervised adaptation using inter-class MLLR

Table 5.4 shows word error rates for unsupervised adaptation. The transcripts of the test data as recognized by the baseline system are used for adaptation. The number of the test sentences used in these experiments are same as those used in the previous chapter. Again inter-class MLLR provides 10.1 to 19.3% of relative improvement in word error rate over conventional MLLR.

Adaptation Method	**1 Test Sentence**	**3 Test Sentences**	**20 Test Sentences**
Baseline (unadapted)	27.3%	27.3%	27.3%
Conventional MLLR (one class)	26.7%	25.9%	23.9%
Inter-class MLLR (full + shift)	24.0% (10.1%)	20.9% (19.3%)	20.1% (15.9%)

Table 5.4 Word error rates for non-native speakers (s3-94) after unsupervised adaptation using inter-class MLLR (Relative percentage improvement over conventional MLLR is shown in parenthesis)

As we described in Subsection (5.3.2), ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) can also be estimated by averaging speaker-dependent ($\boldsymbol{T}_{mn,s}$, $\boldsymbol{d}_{mn,s}$). Using inter-class MLLR with the speaker-averaged full matrix $\boldsymbol{T}_{mn}$ and the shift vector $\boldsymbol{d}_{mn}$ results in a WER of 20.7% for 1 adaptation sentence and 19.9% for 3 adaptation sentences. This is a little worse than the result in Table 5.3 which used inter-class transformation obtained from Eq. (5.18).

5.3.4 Experiments with Native Speakers from the 1994 DARPA WSJ Task

We also evaluated inter-class MLLR on the native speakers from the 1994 DARPA Wall Street Journal task (S0-94) in the same fashion as we did in the previous chapter. The inter-class transformations are trained using 25 native speakers from the standard WSJ training corpus (wsj-25). Inter-class MLLR with a full matrix and shift vector provides 3.8% of improvement in word error rate as shown in Table 5.5, which is similar to the result obtained with PC-MLLR.

Adaptation Method	5 Adaptation Sentences
Baseline (unadapted)	21.9%
Conventional MLLR (one class)	18.3%
Inter-class MLLR (shift only)	17.7% (3.3%)
Inter-class MLLR (diag. + shift)	17.8% (2.7%)
Inter-class MLLR (full + shift)	17.6% (3.8%)

Table 5.5 Word error rates for non-native speakers (s0-94) after supervised adaptation using inter-class MLLR (Relative percentage improvement over conventional MLLR is shown in parenthesis)

The improvement is much smaller than the case of non-native speakers, and different forms of inter-class MLLR provide similar results. Note again that a small language model weight is used for the s0-94 data to emphasize the effect of adaptation on the acoustic models as in PC-MLLR. When a larger language model weight is used, inter-class MLLR does not provide improvement over conventional MLLR as before.

5.4 Enabling Unequal Contributions from Neighboring Classes

In this section we describe two methods to consider the different contributions from the neighboring classes [14]. One is to apply weights to the neighboring classes. Another is to limit the number of the neighboring classes to be used as more adaptation data are available.

5.4.1 Application of Weights on Neighboring Classes

Inter-class MLLR considers several neighboring classes while estimating the MLLR parameters of a target class. In this procedure, some neighboring classes may be *"closer"* to the target class than other neighboring classes. Therefore it may be helpful to apply different weights to the various neighboring classes to accommodate their different contributions to the target class.

We use estimation error as a measure of *"closeness"*. As we explained in Chapter 3, we can think of MLLR as a weighted least squares estimation between a baseline mean vector μ_k and the input feature vector (or adaptation data) $\boldsymbol{o}_t$. Considering Eq. (3.19) $(\boldsymbol{o}_t = \boldsymbol{A}\mu_k + \boldsymbol{b} + \boldsymbol{e}_t)$, we can rewrite Eq. (5.10) for speaker *s* as

$$\boldsymbol{o}_{t,s} = \boldsymbol{A}_{m,s}\mu_k^{(mn)} + \boldsymbol{b}_{m,s} + \boldsymbol{e}_{k,t,s}^{(mn)}, \qquad k \in \text{Class } n \tag{5.20}$$

where $\boldsymbol{e}_{k,t,s}^{(mn)}$ is the linear approximation error observed for Gaussian *k* of class *n* and speaker *s* using the parameters ($\boldsymbol{A}_{m,s}$, $\boldsymbol{b}_{m,s}$). A large error means that neighboring class *n* is not helpful in estimating the target class parameters ($\boldsymbol{A}_{m,s}$, $\boldsymbol{b}_{m,s}$), or that neighboring class *n* is not *"close"* to the target class.

In the training corpus, we know the correct ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) and ($\boldsymbol{A}_{m,s}$, $\boldsymbol{b}_{m,s}$) for each training speaker *s*, so $(\boldsymbol{e}_{k,t,s}^{(mn)})$ is easily calculated for each time *t*.

$$\boldsymbol{e}_{k,t,s}^{(mn)} = \boldsymbol{o}_{t,s} - \boldsymbol{A}_{m,s}\mu_k^{(mn)} - \boldsymbol{b}_{m,s}, \qquad k \in \text{Class } n \tag{5.21}$$

The variance of the error is obtained using the Baum-Welch method as the Gaussian mixtures are reestimated in regular retraining [47]. For Gaussian *k* and classes *m*, *n*, the variance of the error is

$$\boldsymbol{C}_{\boldsymbol{e}_k^{(mn)}} = \frac{\sum_s \sum_t \gamma_{t,s}(k) \cdot \left(\boldsymbol{e}_{k,t,s}^{(mn)} - \mu_{\boldsymbol{e}_{k,s}^{(mn)}}\right)\left(\boldsymbol{e}_{k,t,s}^{(mn)} - \mu_{\boldsymbol{e}_{k,s}^{(mn)}}\right)^{\mathrm{T}}}{\sum_s \sum_t \gamma_{t,s}(k)} \tag{5.22}$$

where $\gamma_{t,s}(k)$ is the *a posteriori* probability of being in Gaussian *k* at time *t*, and $\mu_{\boldsymbol{e}_{k,s}^{(mn)}}$ is the mean of the error for Gaussian *k* and classes *m* and *n*, both for speaker *s*. We assume $\boldsymbol{C}_{\boldsymbol{e}_k^{(mn)}}$ is diagonal. We also estimate the variance $(\boldsymbol{C}_{\boldsymbol{e}_k})$ of the error $(\boldsymbol{e}_{k,t,s})$ in the target class from

$$\boldsymbol{o}_{t,s} = \boldsymbol{A}_{m,s}\mu_k + \boldsymbol{b}_{m,s} + \boldsymbol{e}_{k,t,s}, \qquad k \in \text{Class } m \tag{5.23}$$

Recalling the weighted least squares estimation in Chapter 3, this variance can be used as a weight in inter-class MLLR instead of the variance of the Gaussian mixture of the baseline model. With this assumptions Eq. (5.13) which estimates ($\boldsymbol{A}_m$, $\boldsymbol{b}_m$) becomes as follows.

$$\begin{aligned} Q_{\mathrm{W}}(m) = & \sum_{k \in \text{Class } m} \sum_t \gamma_t(k)(\boldsymbol{o}_t - \boldsymbol{A}_m\mu_k - \boldsymbol{b}_m)^{\mathrm{T}} \boldsymbol{C}_{\boldsymbol{e}_k}^{-1}(\boldsymbol{o}_t - \boldsymbol{A}_m\mu_k - \boldsymbol{b}_m) \\ & + \sum_{n \in \text{Neighbors}} \sum_{k \in \text{Class } n} \sum_t \gamma_t(k)(\boldsymbol{o}_t - \boldsymbol{A}_m\mu_k^{(mn)} - \boldsymbol{b}_m)^{\mathrm{T}} \boldsymbol{C}_{\boldsymbol{e}_k^{(mn)}}^{-1}(\boldsymbol{o}_t - \boldsymbol{A}_m\mu_k^{(mn)} - \boldsymbol{b}_m) \end{aligned} \tag{5.24}$$

Since the inverse of the variance is used as the weight, a Gaussian with a large variance of the estimation error will get a small weight. The variance can also be used to sort neighboring classes, in choosing the top N neighboring classes as described in the previous section.

In our experiment, we do not have enough training data to accurately estimate the variance of the error, so we estimate the average ratio of the baseline variance to the variance of the error for classes m and n. The ratios w_m for the target class m, and w_{mn} for the neighboring class n become

$$w_m = \underset{k \in \text{Class } m}{\boldsymbol{average}} \left(\frac{\sigma_{k,i}^2}{\sigma_{\boldsymbol{e}_k, i}^2} \right) \tag{5.25}$$

$$w_{mn} = \underset{k \in \text{Class } n}{\boldsymbol{average}} \left(\frac{\sigma_{k,i}^2}{\sigma_{\boldsymbol{e}_k^{(mn)}, i}^2} \right) \tag{5.26}$$

where $\sigma_{k,i}^2$, $\sigma_{\boldsymbol{e}_k,i}^2$, and $\sigma_{\boldsymbol{e}_k^{(mn)},i}^2$ are the diagonal elements of $\boldsymbol{C}_k$, $\boldsymbol{C}_{\boldsymbol{e}_k}$, and $\boldsymbol{C}_{\boldsymbol{e}_k^{(mn)}}$, respectively.

The weights w_m and w_{mn} are combined as follows.

$$Q'_{\mathrm{W}}(m) = w_m Q_{\mathrm{C}}(m) + \sum_{n \in \text{Neighbors}} w_{mn} Q_n(m) \tag{5.27}$$

Table 5.6 and 5.7 show the word error rates after applying weights with Eq. (5.27) for supervised and unsupervised adaptations. The supervised adaptation shows a small improvement (0.8 to 1.3% relative) over inter-class MLLR without weights in Table 5.3. However the improvements are statistically insignificant because the amount of the test data was small.

Adaptation Method	**1 Adaptation Sentence**	**3 Adaptation Sentences**
Conventional MLLR (one class)	24.1%	23.1%
Inter-class MLLR without weights (full + shift)	20.4% (15.4%)	19.6% (15.2%)
Inter-class MLLR with weights (full + shift)	20.2% (16.2%)	19.3% (16.5%)

Table 5.6 Word error rates for non-native speakers (s3-94) after supervised adaptation using inter-class MLLR with weights (Relative percentage improvement over conventional MLLR is shown in parenthesis)

Adaptation Method	**1 Test Sentence**	**3 Test Sentences**	**20 Test Sentences**
Conventional MLLR (one class)	26.7%	25.9%	23.9%
Inter-class MLLR without weights (full + shift)	24.0% (10.1%)	20.9% (19.3%)	20.1% (15.9%)
Inter-class MLLR with weights (full + shift)	24.3% (9.0%)	20.9% (19.3%)	19.9% (16.7%)

Table 5.7 Word error rates for non-native speakers (s3-94) after unsupervised adaptation using inter-class MLLR with weights (Relative percentage improvement over conventional MLLR is shown in parenthesis)

For the unsupervised case, even though the differences were minor, applying weights was not helpful for 1 test sentence case, and was helpful for 20 test sentence case. We speculate that the limited benefit in unsupervised adaptation is a consequence of transcription errors by the baseline system. In other words, unsupervised adaptation needs more smoothing than supervised adaptation to average out the effect of incorrect transcriptions. The application of weights, that are smaller weights for the neighboring classes than for the target class, enables unsupervised adaptation to focus on the target class and results in less smoothing. It also reduces the effective amount of input data (because the contributions of neighboring classes are multiplied by their weights). This could be a problem especially when the amount of the data is small. As we have more data, we can have enough smoothing to obtain benefits from the weights.

5.4.2 Limiting the Number of Neighboring Classes

As we mentioned in Section 5.2, the number of neighboring classes to be used depends on the amount of adaptation data. As we have more adaptation data, we can use fewer neighboring classes for a target class. For a large amount of adaptation data, we may not use any neighboring class, and inter-class adaptation becomes the same as conventional multi-class adaptation.

The variance of the error $\boldsymbol{C}_{\boldsymbol{e}_k^{(mn)}}$ obtained from Eq. (5.22) is used to measure the closeness of neighboring class *n* to the target class *m*. For each target class, neighboring classes are sorted according to their variance of the error. Adaptation data from the closest neighboring class are accumulated first in Eq. (5.27), then from the next closest neighboring class until sufficient data are used. We can consider $w_{mn}\gamma_t$ in Eq. (5.27) as an effective count of adaptation data. We set a threshold and accumulate $w_{mn}\gamma_t$ until it exceeds the threshold. We can control the threshold and the number of classes accumulated to get better recognition accuracy. The best threshold can be obtained from prior experiments.

Fig. 5.6 shows the word error rates as a function of the threshold for non-native speakers (s3-94) after supervised adaptation. The value next to each data point is the average number of classes used to estimate

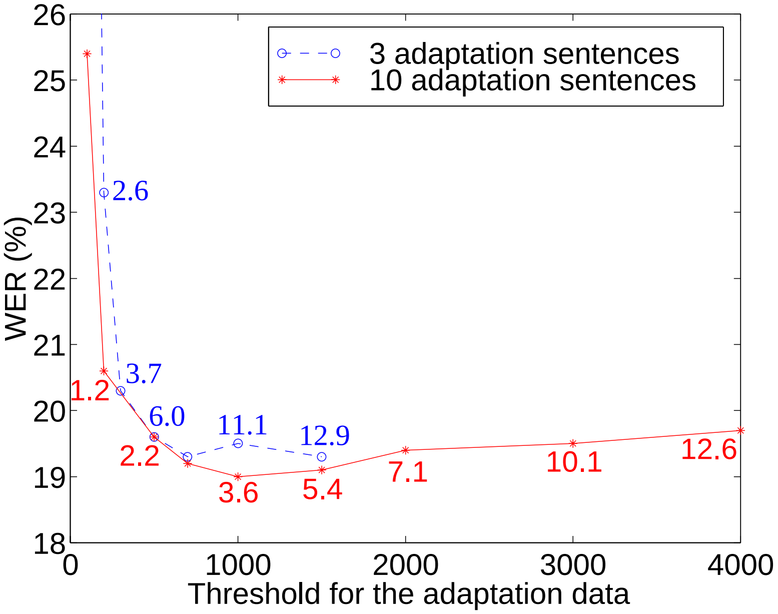


Figure 5.6 Word error rates as a function of the threshold for adaptation data in inter-class MLLR. The value next to each data point is the average number of classes used.

the MLLR parameters of the target class. For a very small threshold, only 1 class (target class itself) is accumulated because the amount of adaptation from the target class exceeds the threshold. For a very large threshold, all the classes are accumulated.With the threshold value 1500, almost all classes (the average number = 12.9 out of 13) are used in 3 adaptation sentence case, while only 5.4 classes are used in 10 adaptation sentence case. It is because there are more adaptation data from each class in 10 adaptation sentence case than in 3 adaptation sentence case. If the threshold is too small, the error rate becomes high because too little adaptation data are used. In 10 adaptation sentence case, the best WER is obtained at a threshold value of 1000, which corresponds to the use of about only 3.6 MLLR classes. In this case the adaptation data are sufficient with a small number of classes, and adding more adaptation data from "*farther*" classes is not helpful. In 3 adaptation sentence case, WER continues to improve until all the data are used, and all MLLR classes are contributing to the parameter estimates.

5.5 Further Issues

5.5.1 Inter-Class Transformation from Different Training Data

The performance of inter-class MLLR will depends on the *quality* of the inter-class transformation. If the inter-class transformation does not match to the characteristics of the test data, it may not be helpful. Table 5.8 shows the word error rates for the inter-class transformation functions trained from different data with non-native speakers (s3-94) after supervised adaptation.

First we train the inter-class transformation functions using 25 native speakers from the standard WSJ training corpus (wsj_25). It provides 22.0% and 21.1% of WER which is higher than the previous results (20.4% and 19.6% of WER) in Table 5.3 where the inter-class transformation functions are trained from other test speakers (s3_9). We think it is because s3_9 (non-native speakers) has closer characteristic to the test data than wsj_25 (native speakers).

As a side experiment, we performed inter-class MLLR using only 1 or 3 adaptation sentences, but obtaining estimates of ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) using a large amount of adaptation data (200 sentences) from the same speaker (s3_1) with the test data for each speaker. It provides very good WER because the inter-class transformation ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) can provide a good *a priori* knowledge for the test speaker.

Adaptation Method	**1 Adaptation Sentence**	**3 Adaptation Sentences**
Conventional MLLR (one class)	24.1%	23.1%
Inter-class MLLR (trained from wsj_25)	22.0% (8.7%)	21.1% (8.7%)
Inter-class MLLR (trained from s3_1)	16.5% (31.5%)	16.8% (27.3%)

Table 5.8 Word error rates for non-native speakers (s3-94) after supervised adaptation using inter-class transformation functions trained from different data (Relative percentage improvement over conventional MLLR is shown in parenthesis)

This is a kind of "*oracle*" experiment because we use the information of the test speaker identity which may not be known in an actual application. The inter-class transformation ($\boldsymbol{T}_{mn}$, $\boldsymbol{d}_{mn}$) in the previous section does not use the information of the speaker identity (speaker independent). The test speaker is not normally available to train adaptation parameters, but in circumstances where he or she is, substantial improvements in accuracy can be obtained. One approach to improved recognition accuracy may be to prepare several sets of inter-class transformations representing different type of speakers, and selecting an appropriate set for each test speaker.

5.5.2 Combination with Principal Component MLLR

In the previous chapter, we described weighted principal component MLLR (WPC-MLLR) which provided a good improvement in WER. We also performed experiments to combine inter-class MLLR with WPC-MLLR.

We first transformed the baseline mean vectors using the inter-class transformation as Eq. (5.9). We then calculated eigenvalues and eigenvectors from the transformed mean vectors, and applied WPC-MLLR. However this does not provides further improvement. We notice that the distribution of the eigenvalues becomes compact in this case, *i.e.* the ratio of the biggest eigenvalue and smallest eigenvalues from the transformed mean vectors becomes much smaller than that from the baseline mean vectors, from about 10^4 to 10^2. We believe that this reduces the benefit of WPC-MLLR.

5.5.3 Combination with Sample Means

As we described in Chapter 2, Bayesian adaptation combines the baseline mean with the sample mean. We can apply this idea using the adapted mean from inter-class MLLR as the baseline mean. Considering Eq. (2.1), the new adapted mean $\hat{\mu}'_k$ can be written as an MAP-like combination [11].

$$\hat{\mu}'_k = \beta\bar{\mu}_k + (1-\beta)\hat{\mu}_k \tag{5.28}$$

where $\bar{\mu}_k$ is a sample mean, $\hat{\mu}_k$ is an adapted mean from inter-class MLLR, β is the interpolation factor. Instead of using variances to calculate β as shown in Eq. (2.1), we use the parameters k_1 and k_2 parameters assuming the variances of the estimates $\bar{\mu}_k$ and $\hat{\mu}_k$ are inverse proportional to the amount of the adaptation data.

$$\beta = \frac{k_1 N_1}{k_1 N_1 + k_2 N_2} \tag{5.29}$$

where N_1 is the effective number of data for $\bar{\mu}_k$ and N_2 is the effective number of data for $\hat{\mu}_k$. The optimal k_1 and k_2 can be found from prior experiments.

As Table 5.9 shows the MAP-like combination provides a good improvement with 200 adaptation sentences. Combining with the sample mean will improve asymtotic property. However this method will not be useful when the amount of adaptation is very small because, in this case, the sample mean will not be reliable.

Adaptation Method	**200 Adaptation Sentence**
Inter-class MLLR (all neighboring classes)	19.8%
Inter-class MLLR (limited number of neighboring classes)	17.7%
Inter-class MLLR + Sample mean	11.9%

Table 5.9 Word error rates for non-native speakers (s3-94) after supervised adaptation using MAP-like combination with inter-class MLLR

We can also apply MAP-like combination of the adapted mean $\hat{\mu}_k$ and the baseline mean μ_k, which may improve the reliability of the final adapted mean $\hat{\mu}''_k$ for very small amount of adaptation data.

$$\hat{\mu}''_k = \beta'\mu_k + (1 - \beta')\hat{\mu}_k \tag{5.30}$$

We can even think of MAP-like combination of all three means: baseline mean μ_k, sample mean $\bar{\mu}_k$, and adapted mean $\hat{\mu}_k$ from inter-class MLLR.

$$\hat{\mu}'''_k = \beta_1\mu_k + \beta_2\bar{\mu}_k + (1 - \beta_1 - \beta_2)\hat{\mu}_k \tag{5.31}$$

In our experiments, combining with the baseline mean was not helpful. We believe that this is because $\bar{\mu}_k$ and $\hat{\mu}_k$ are good enough to produce a reliable estimate with the adaptation data used.

5.5.4 Clustering of MLLR Classes

It is important to have a good clustering of transformation classes in inter-class MLLR. In this thesis, we use pre-clustered phonetic-based MLLR classes. If we can find better clustering, we may improve recognition accuracy.

A large number of the classes will provides more details of Gaussian mean vectors at the expense of more computation. It also requires more training data to get good estimates of the inter-class transformation functions. Therefore we should choose a proper number of the classes for a given application. In our experiments using 2 MLLR classes, inter-class MLLR provides 24.2% of WER for 1 adaptation sentence case, and 21.5% of WER for 3 adaptation sentence case (supervised adaptation). It is worse than the results (full + shift) in Table 5.3 which use 14 MLLR classes.

5.5.5 Application to the Broadcast News Task (Hub 4)

We evaluated inter-class MLLR on the broadcast news task (Hub4), which has a large vocabulary and several mixed conditions. We trained the inter-class transformation functions from baseline broadcast speech (clean+native+prepared) because this speech has enough training data. In out experiments, inter-class MLLR does not provide improvement over conventional MLLR, as was the case for WPC-MLLR. It seems that inter-class MLLR is also less effective on a complex task. Because we trained the inter-class transformation from baseline broadcast speech, it may not be good for other conditions (noisy, non-native, spontaneous, band-limited). For baseline broadcast speech, conventional MLLR may capture most of the benefit to be expected from MLLR, so inter-class MLLR does not provide further improvement.

5.6 Summary

In this chapter, we introduced a new adaptation method called *inter-class MLLR*. It utilizes relationships among different classes to achieve both detailed and reliable adaptation with limited data. In this method, the inter-class relation is given by another linear regression which is estimated from training data.When we estimate MLLR parameters of a target class for test speech, we first transform the baseline mean vectors in neighboring classes using inter-class regression so that they can contribute to the estimation. We can then use adaptation data from neighboring classes as well as the target class.

In practice, each neighboring class may make a different contribution to the target class. For this reason we can apply weights to the neighboring classes that consider their differences. We measure estimation errors for the target class parameters in the neighboring classes, and use the inverse of the variance of the estimation error as the weight. A neighboring class with a large variance of the error will get a small weight. This procedure is similar to the weighted least squares estimation in classical linear regression.

As we have more adaptation data, we can rely on fewer neighboring classes for a target class. For a large amount of adaptation data, we may not need any neighboring class, and inter-class adaptation becomes the same as conventional multi-class adaptation.

The inter-class transformation represents *a priori* knowledge. Therefore, if it is different from the characteristics of the test data, it may not be useful. Combining inter-class MLLR with WPC-MLLR does not provides further improvement in our experiments. In this case, the smallest eigenvalue becomes larger than in WPC-MLLR alone. We think that this reduces the benefit of WPC-MLLR.

We can apply MAP-like combinations of the baseline mean, sample mean, and adapted mean from inter-class MLLR to get better recognition accuracy. Clustering regression classes is also important. We did not achieve a further improvement on broadcast news task (Hub 4) with inter-class MLLR. These areas need further study.

Chapter 6
Conclusions

6.1 Summary and Contributions

Adaptation is a process that reduces the differences between training and testing conditions, usually by using a small amount of adaptation data. In transformation-based adaptation, the recognition model parameters are clustered into transformation classes and several parameters in a transformation class are assumed to be updated by the same transformation function. The transformation function is estimated from the adaptation data from the transformation class. Since transformation classes are considered independently, it is difficult to achieve reliable estimates of the transformation functions across multiple classes with a small amount of adaptation data. If the estimates are not reliable, they will not be helpful for recognition.

Maximum likelihood linear regression (MLLR) is currently one of the most popular speaker-adaptation technique. It assumes that the baseline mean vectors are transformed to adapted mean vectors by linear regression. We have noted that MLLR can be considered as the weight least squares either between the baseline mean vectors and the input feature vectors, or between the baseline mean vectors and the sample mean vectors.

In this thesis we have tried to improve speech recognition accuracy by obtaining better estimation of linear transformation functions with a small amount of adaptation data in speaker adaptation. The major contributions of this thesis are the developments of two new adaptation algorithms to improve maximum likelihood linear regression. The first one is called *principal component MLLR* (*PC-MLLR*), and it reduces the variance of the estimate of the MLLR matrix using principal component analysis. The second one is called *inter-class MLLR,* and it utilizes relationships among different transformation functions to achieve more reliable estimates of MLLR parameters across multiple classes. In the following subsections we summarize these methods.

6.1.1 Principal Component MLLR

The main idea of PC-MLLR is that if we estimate the MLLR matrix in the eigendomain, the variances of the components of the estimates are inversely proportional to their eigenvalues. Therefore we can select more reliable components to reduce the variances of the resulting estimates and to improve speech recognition accuracy. PC-MLLR eliminates highly variable components and chooses the principal components corresponding to the largest eigenvalues. If all the component are used, PC-MLLR becomes the same as conventional MLLR. Choosing fewer principal components increases the bias of the estimates which can reduce recognition accuracy. To compensate for this problem, we developed *weighted principal component MLLR (WPC-MLLR)*. Instead of eliminating some of the components, all the components in WPC-MLLR are used after applying weights that minimize the mean square error. The component corresponding to a larger eigenvalue has a larger weight than the component corresponding to a smaller eigenvalue.

As more adaptation data become available, the benefits from these methods may become smaller because the estimates using conventional MLLR become more reliable. However, if we have a larger amount of adaptation data, we would use a larger number of MLLR classes, making the amount of adaptation data for each MLLR class smaller. Therefore PC-MLLR and WPC-MLLR can be useful.

In our experiments PC-MLLR improves recognition accuracy over conventional MLLR, and WPC-MLLR provided further improvement. However, they do not provide improvement over conventional MLLR on a complex task like the DARPA broadcast news. These methods seems to be more effective when there is a larger mismatch between training and test conditions in a small task.

6.1.2 Inter-Class MLLR

It is useful to consider relationships among different parameters when a small amount of adaptation data is available. Most previous studies use correlations or regression models among the recognition model parameters in a Bayesian framework. In this thesis, inter-class MLLR utilizes relationships among different transformation functions. Inter-class transformations given by linear regressions are used to modify the baseline mean vectors in the neighboring classes so that the neighboring classes can contribute to the estimates the MLLR parameters of the target class. If the inter-class transformations are identity functions, inter-class MLLR becomes the same as conventional single-class MLLR. This idea also can be applied to other types of transformation-based adaptation and general parameter estimation problems.

In inter-class MLLR several neighboring classes are considered for each target class. In this procedure, some neighboring classes may be closer to the target class than other neighboring classes. Therefore we apply different weights to the neighboring classes to accommodate their different contributions to the target class. Considering the weighted least squares estimation, the weight for each neighboring class is inversely proportional to the variance of the error that is produced in estimating the target parameters. Therefore a neighboring class with a smaller variance has a larger weight.

As more adaptation data become available, fewer neighboring classes can be used for a target class. For a large amount of adaptation data, we may not use any neighboring class at all. In this case inter-class adaptation becomes the same as conventional multi-class MLLR. To limit the number of neighboring classes, the neighboring classes are sorted for each target class according to their variances of the errors. Adaptation data from the closest neighboring class are used first, then from the next closest neighboring class until sufficient data are used.

In our experiments, inter-class MLLR also provides improvements in recognition accuracy compared to conventional MLLR. Inter-class MLLR provides greater recognition accuracy than WPC-MLLR in supervised adaptation. In unsupervised adaptation, however, inter-class MLLR is worse with a very small amount of test data, and becomes better with more test data. We believe that WPC-MLLR is more effective in highly unreliable cases like unsupervised adaptation with a very small amount of test data.

We tried to combine WPC-MLLR and inter-class MLLR by first modifying the baseline mean vectors using inter-class transformations, and then using WPC-MLLR. However, this combination did not provide further improvement in accuracy. We believe that the benefits of WPC-MLLR become smaller after inter-class transformations such as the eigenvalues become more compact.

6.2 Suggestions for Future Work

In this thesis we used principal component analysis and inter-class relationships to improve speaker adaptation. We think that the following areas need further study for better understanding of the problems and further improvement in recognition accuracy.

6.2.1 Prediction of the Inter-Class Transformations

In this thesis the inter-class transformations are estimated from many training speakers forming speaker-independent prior information. These transformations are applied to all test speakers, assuming that the test speakers have the same statistics as the training data. However each test speaker may have different statistics. As in the case of any other procedure based on *a priori* information, these transformations may not be helpful if the test speaker has different statistics from the training speakers. Our experiments show that the inter-class transformations from training speakers that are similar to the test speakers provide better recognition accuracy than transformations from training speakers with much different statistics. It is also observed that the inter-class transformations from the same speaker with the same test speaker provide the best recognition accuracy.

Therefore we may build several sets of the inter-class transformations which can represent a variety of test speakers, and select an appropriate set of the inter-class transformations for a test speaker. We may also modify the inter-class transformations so that they better represent the statistics of the test speaker. For this procedure we will need to estimate or predict the characteristics of the test speaker. Similar topics have been studied in Bayesian framework where the prior distributions of model parameters are used [28].

6.2.2 Control of Weights in Inter-Class MLLR

Using neighboring classes to estimate the parameters in the target class has both advantages and disadvantages. Neighboring classes are helpful because they add more data for the estimation, but they are not helpful if they have different statistics from those of the target class. If the disadvantages are bigger than the advantages then it is better not to include the neighboring class. In Section 5.4 we described how we can choose a smaller number of neighboring classes as we obtain more adaptation data. We also described application of weights to the neighboring classes to incorporate the differences of their contributions to the target class. We can consider discarding a neighboring class as applying zero weight to it. We may find better weights which can automatically control this procedures to select the neighboring classes. However, the better weights may not provide much further improvements in WER as the application of weights in our experiments provided the minor improvements.

6.2.3 Other Issues

There are some more issues we can consider for the further improvement in recognition accuracy even though we think these are less important than the previous issues.

Combination of inter-class MLLR and WPC-MLLR: In our experiments we did not achieve further improvement by combining WPC-MLLR and inter-class MLLR. However we still think it is possible to improve recognition accuracy by applying these methods together.

Combination of inter-class MLLR and sample mean: We obtained a large improvement in recognition accuracy by combining the adapted means from inter-class MLLR with the sample means when a larger amount of adaptation data is available. We used MAP-like interpolation for this combination. We may find a better model to combine the adapted means from inter-class MLLR with the baseline means and the sample means.

Clustering of MLLR classes: In our experiments we used pre-determined MLLR classes. We may achieve better recognition accuracy with more MLLR classes at the expense of more computation. We may control the number of the MLLR classes dynamically for a new task or system. Finding a better metric to cluster the MLLR classes also needs further study.

Application to more complex tasks: We achieved improvements in recognition accuracy mainly for non-native speakers in a relatively simple task. Further work is needed to get a better understanding of the nature of problem of a complex task like the DARPA broadcast news, and to achieve improvement in recognition accuracy.

Testing with other sources of degradation: In this thesis we focused on adaptation for different speakers. Further study is needed to evaluate the effects of the algorithms developed on other sources of degradation. We believe that different microphones and channels have similar effects as different speakers. However background noise may be different.

6.3 Conclusions

In this thesis, we developed two new adaptation algorithms to improve speech recognition accuracy in maximum likelihood linear regression. The first one is called *principal component MLLR* (*PC-MLLR*), and it reduces the variance of the estimate of the MLLR matrix using principal component analysis. The second one is called *inter-class MLLR*, and it utilizes relationships among different transformation functions to achieve more reliable estimates of MLLR parameters across multiple classes.

Inter-class MLLR requires training data to estimate inter-class transformations, while principal component MLLR does not. We may improve recognition accuracy further in inter-class MLLR by obtaining better inter-class transformations for the test data. We may also improve recognition accuracy by combining the inter-class MLLR and principal component MLLR. The developed methods are aimed for the case when only a small amount of adaptation data is available. For a larger amount of adaptation data, we can combine the adapted means from these methods with the sample means to get further improvement in recognition accuracy.

References

[1] A. Acero, *Acoustical and Environmental Robustness in Automatic Speech Recognition*, Kluwer Academic Publishers, 1992.

[2] S. M. Ahadi and P. C. Woodland, “Combined Bayesian and predictive techniques for rapid speaker adaptation of continuous density hidden Markov models,” *Computer Speech and Language*, vol. 11, pp.187-206, July 1997.

[3] M. Afify, Y. Gong and J.-P. Haton, “Correlation based predictive adaptation of hidden Markov models,” *Proc. of Eurospeech*, p. 2059-2062, 1997.

[4] E. Bocchieri *et al*., “Correlation modeling of MLLR transform biases for rapid HMM adaptation to new speakers,” *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 2343-2346, 1996.

[5] S. Chatterjee and B. Price, *Regression Analysis by Example*, John Wiley & Sons, 1977.

[6] S. S. Chen and P. DeSouza, “Speaker adaptation by correlation (ABC),” *Proc. of Eurospeech*, pp. 2111-2114, 1997.

[7] C. Chesta, O. Siohan, and C.-H. Lee, “Maximum *a posteriori* linear regression for hidden Markov model adaptation,” *Proc. of Eurospeech*, pp.211-214, 1999.

[8] J.-T. Chien, C.-H. Lee and H.-C. Wang “A hybrid algorithm for speaker adaptation using MAP transformation and adaptation,” *IEEE Signal Processing Letters*, vol. 4, no. 6, p. 167-169, June 1997.

[9] J.-T. Chien, J.-C. Junqua, and P. Gelin, “Extraction of reliable transformation parameters for unsupervised speaker adaptation,” *Proc. of Eurospeech*, pp. 207-210, 1999.

[10] S. Cox, “Predictive speaker adaptation in speech recognition,” *Computer Speech and Language*, vol. 9, pp.1-17, 1995.

[11] V. V. Digalakis and L. G. Neumeyer “Speaker adaptation using combined transformation and Bayesian methods,” *IEEE Trans. on Speech and Audio Processing*, vol. 4, no. 4, pp. 294-300, July 1996.

[12] S.-J. Doh and R. M. Stern, “Weighted principal component MLLR for speaker adaptation,” *Proc. of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 1999.

[13] S.-J. Doh and R. M. Stern, “Inter-class MLLR for speaker adaptation,” *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1755-1758, 2000.

[14] S.-J. Doh and R. M. Stern, “Using class weighting in inter-class MLLR,” *to be presented at International Conference on Spoken Language Processing,* Beijing, China, October 2000.

[15] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, 1973.

[16] M. J. F. Gales, “The generation and use of regression class trees for MLLR adaptation,” *Technical Report*, CUED/F-INFENG/TR 263, Cambridge University Engineering Department, August 1996.

[17] M.J.F. Gales, D. Pye and P. C. Woodland, “Variance Compensation Within the MLLR Framework for Robust Speech Recognition and Speaker Adaptation,” *Proc. of International Conference on Spoken Language Processing,* pp.1832-1835, 1996.

[18] J.-L. Gauvain and C.-H., Lee, “Maximum A Posteriori Estimation For Multivariate Gaussian Mixture Observations Of Markov Chains,” *IEEE Trans. on Speech and Audio Processing*, vol. 2, n. 2, pp. 291-298, April 1994.

[19] L. Gillick and S. J. Cox, “Some Statistical Issues in the Comparison of Speech Recognition Algorithms,” *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 532-535, 1989.

[20] E. B. Gouvea, *Acoustic-Feature-Based Frequency Warping for Speaker Normalization*, Ph.D thesis, Carnegie Mellon University, July 1998.

[21] H. Hattori and S. Sagayama, “Vector Field Smoothing Principle For Speaker Adaptation”, *Proc. of International Conference on Spoken Language Processing*, pp. 381-384, 1992.

[22] S. Homma, K. Aikawa and S. Sagayama, “Improved Estimation of Supervision in Unsupervised Speaker Adaptation,” *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1023-1026, 1997.

[23] Z. Hu, *Understanding and Adapting to Speaker Variability Using Correlation-Based Principal Analysis*, Ph.D. thesis, Oregon Graduate Institute of Science and Technology, 1999.

[24] X. D. Huang, Y. Ariki, and M. A. Jack, *Hidden Markov Models for Speech Recognition*, Edinburgh University Press, 1990.

[25] J. M. Huerta, *Speech Recognition in Mobile Environments*, Ph.D thesis, Carnegie Mellon University, April 2000.

[26] Q. Huo, C. Chan and C.-H.Lee "Bayesian Adaptive Learning of the Parameters of Hidden markov Model for Speech Recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 3, no.5, p. 334-345, 1995.

[27] Q. Huo and C.-H. Lee "On-line adaptive learning of the correlated continuous density hidden Markov models for speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 6, no. 4, pp. 386-397, July 1998.

[28] Q. Huo and C.-H. Lee, "A Bayesian predictive classification approach to robust speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 8, no. 2, pp. 200-204, March 2000.

[29] M.-Y. Hwang, *Subphonetic Acoustic Modeling for Speaker-Independent Continuous Speech Recognition*, Ph.D thesis, Carnegie Mellon University, CMU-CS-TR-93-230, December, 1993.

[30] F. Jelinek, *Statistical Methods for Speech Recognition*, The MIT Press, 1998.

[31] I. T. Jolliffe, *Principal Component Analysis*, Springer-Verlag, 1986.

[32] J.-C. Junqua and J.-P. Haton, *Robustness In Automatic Speech Recognition Fundamentals and Applications*, Kluwer Academic Publishers, 1996

[33] S. M. Kay, *Fundamentals of Statistical Signal Processing: estimation theory*, PTR Prentice Hall, 1993

[34] R. Kuhn *et al*., "Eigenvoices for speaker adaptation," *Proc. of International Conference on Spoken Language Processing*, pp.1771-1774, 1998.

[35] M. J. Lasry and R. M. Stern, "A posteriori estimation of correlated jointly Gaussian mean vectors," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 4, pp. 530-535, July 1984.

[36] C.-H. Lee, "On feature and model compensation approach to robust speech recognition," *Proc. of ESCA-NATO Workshop on Robust Speech Recognition for Unknown Communication Channels*, pp.45-54, France, April 1997.

[37] C.-H. Lee, C.-H. Lin and B.-H. Juang, "A study on speaker adaptation of the parameters of continuous

density hidden Markov models", *IEEE Trans. on Signal Processing*, vol. 39, n. 4, pp. 806-814, April 1991.

[38] C.-H. Lee, F. K. Soong and K. K. Paliwal, edit, *Automatic Speech and Speaker Recognition Advanced Topics*, Kluwer Academic Publishers, 1996

[39] C. J. Leggetter, *Improved Acoustic Modeling for HMMs Using Linear Transformations*, Ph.D. thesis, Cambridge University, 1995.

[40] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer Speech and Language*, vol. 9, pp.171-185, 1995.

[41] T. Matsui, T. Matsuoka and S. Furui, "Smoothed N-Best-Based Speaker Adaptation for Speech Recognition," *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 1015-1018, 1997.

[42] R. H. Myers, *Classical And Modern Regression With Applications*, PWS-KENT Publishing Company, Boston, 1990.

[43] J. Neter, *Applied Linear Statistical Models*, Richard D. Irwin, Inc., 1985

[44] J. Nouza, "Feature Selection Methods for Hidden Markov Model-Based Speech Recognition," *Proc. of the 13th International Conference on Pattern Recognition*, Vol.2, pp.186-190, 1996.

[45] D. S. Pallett *et al*, "1994 Benchmark Tests for the ARPA Spoken Language Program," *Proc. of Spoken Language Systems Technology Workshop*, pp. 5-36, January 1995.

[46] D. S. Pallett *et al*, "1998 Broadcast news benchmark test results: English and non-English word error rate performance measures," *Proc. of DARPA Broadcast News Workshop*, February1999.

[47] L. R. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, 1993

[48] M. G. Rahim and B.-H. Juang, "Signal bias removal by maximum likelihood estimation for robust telephone speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 4, no. 1, pp. 19-30, January 1996.

[49] W. A. Rozzi and R. M. Stern, "Speaker adaptation in continuous speech recognition via estimation of

correlated mean vectors," *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 865-868, May 1991.

[50] H. Sakoe and S. Chiba, "Dynamic programming optimization for spoken word recognition," *IEEE Trans. on Acoustics, Speech and Signal Processing*, ASSP-34 (1), pp. 52-59, February 1994.

[51] 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. 190-202, May 1996.

[52] K. Shinoda and C.-H. Lee, "Unsupervised adaptation using structured Bayes approach," *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp.793-796, 1998.

[53] H. Hattori and S. Sagayama, "Vector field smoothing principle for speaker adaptation", *Proc. of International Conference on Spoken Language Processing*, pp. 381-384, 1992.

[54] J.-I. Takahashi and S. Sagayama, "Vector-field-smoothed Bayesian learning for fast and incremental speaker/telephone-channel adaptation", *Computer Speech and Language*, vol. 11, pp.127-146, April 1997.

[55] S. Takahashi and S. Sagayama, "Tied-structure HMM based on parameter correlation for efficient model training," *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 467-470, 1996.

[56] G. Zavaliagkos, *Maximum A Posteriori Adaptation Techniques For Speech Recognition*, Ph.D. thesis, Northeastern University, May 1995.

[57] URL: http://www.speech.cs.cmu.edu/