SPEAKER ADAPTATION IN CONTINUOUS SPEECH RECOGNITION VIA ESTIMATION OF CORRELATED MEAN VECTORS

A DISSERTATION
SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY
in
ELECTRICAL AND COMPUTER ENGINEERING

by

William Anthony Michael Rozzi

Carnegie Mellon University
Pittsburgh, Pennsylvania
April 1991

Abstract

The present study addressed the problem of speaker adaptation in both feature-based and stochastic model-based continuous speech recognition systems. Effective speaker adaptation procedures must be able to adapt to the characteristics of a new speaker given speaker-specific training data in quantities which are well below those required for training speaker-dependent systems. The adaptation algorithm must be computationally efficient to allow for a short enrollment process. Since the basic recognition unit in continuous speech recognition systems is at the sub-word level, user feedback of unit labels is impractical. The adaptation algorithm should therefore operate in an unsupervised mode.

The approach taken in this thesis was to use multivariate parameter estimation procedures to update the mean values of the component densities which comprise a feature-based system's classifiers, or a stochastic model-based system's codebook. Emphasis was placed on obtaining low initial estimation error with a computationally efficient algorithm. Adaptive filtering techniques were exploited to derive an estimator which met these conditions. The Bayesian optimal (EMAP) estimator was first shown to be equivalent to a minimum mean-square error (MMSE) adaptive filter with time-varying data statistics. A stochastic gradient approximation of the MMSE formulation resulted in a least mean-square estimator, called LMS-C, which with proper initialization produced a faster rate of convergence than the Bayesian estimator. Computational requirements of the LMS-C estimate are approximately one-third of those of the EMAP estimate. Unlike the EMAP estimate, however, the LMS-C estimate is asymptotically biased. This misadjustment is negligible in the context of the speaker adaptation problem.

Expressions which define the LMS-C algorithm and its mean-square estimation error were derived and analyzed assuming correlated, jointly-gaussian data distributions. Compared with maximum likelihood (ML) estimation, the additional expense required for LMS-C (or EMAP) estimation was shown to be justified when the dogmatism of the data is neither very large nor very small, and training data is limited. Relative gains of LMS-C and EMAP estimates over ML estimates were shown to increase with increasing correlation between the data means and with increasing skew in the class' prior probabilities.

The general limitations of LMS-C, EMAP, and ML adaptation procedures were assessed in the

context of unsupervised speaker adaptation in the Carnegie Mellon ANGEL system, a novel feature-based system called PROPHET, and a semi-continuous version of the CMU SPHINX system. Comparisons between the ANGEL and PROPHET systems indicated the necessity for the adaptation data to obey the gaussian assumptions made in derivation of the estimation algorithms. When these assumptions were met (using computer-generated data), adaptation using the LMS-C or EMAP algorithms reduced front vowel classification error rates by 28% after the presentation of 10 unlabeled training samples. Five iterations through the training data were shown to reduce the error rate by an additional 10% over the one-iteration rate.

Unsupervised adaptation experiments with a synthetic HMM indicated that the EMAP and LMS-C estimates were able to produce an estimation error lower than the ML estimate only when the dogmatism of the data was low. It was shown that the unsupervised ML estimate, as specified by the HMM reestimation procedure, produced an estimation error which was initially much larger than the supervised form of this estimate. Due to the dependence of the EMAP and LMS-C estimates on the ML, performance of these two algorithms was also reduced. Repeated iteration of the forward-backward algorithm eventually reduced the unsupervised level of error to that of the supervised estimate. It was also shown that the unsupervised form of the ML estimate implicitly models the correlation of the data means which serves to reduce estimation error as the data means become more correlated.

Mean vector adaptation in SPHINX was less successful than with the feature-based systems because the dogmatism of the data in SPHINX was more than twice that of the feature-based systems. The SPHINX system's performance using LMS-C, EMAP, and ML codebook mean vector adaptation methods was compared with the system using no adaptation. Results showed an overall reduction of 2.0 to 3.4% in word error rate due to adaptation for a set of 11 speakers from the DARPA resource management task. Using a distance metric applied to the adapted codebooks, word error rates were reduced on average by 15% for those speakers automatically identified as good candidates for adaptation.

Acknowledgements

I am sincerely grateful to my advisor, Prof. Richard Stern, for his patience and guidance throughout the extended duration of this work. I also thank Prof. Virginia Stonick, Prof. Vijaya Kumar, and Prof. Raj Reddy for serving as members of my thesis committee.

I wish to thank the CMU Speech group for providing a stimulating work environment, and in particular Bob Weide and Ron Cole for many lessons about the characteristics of speech, and Xuedong Huang for his help and guidance with the semi-continuous SPHINX system.

I thank Bob Wohlford at Ameritech Services for his encouragement and support of portions of this work.

I am indebted to my parents for the sacrifices they have made for their children.

Finally, I wish to thank Sharon for providing me with the environment, encouragement and emotional support necessary to produce this document.

Chapter 1
Introduction

1.1. Background

Speech is the most natural form of communication. This communication is facilitated by the human capacity to readily adapt to differences in individuals' speech production. Variability in the structure of the vocal apparatus affect the acoustic realization of speech sounds, while other factors cause the inclusion, deletion, or substitution of "normal" speech sounds. While speech input to machines would make the most effective use of human communicative abilities, speech recognition systems are not as flexible as humans in adapting to the characteristics of unfamiliar speakers. Even with extensive training, current speaker-independent recognition systems may perform poorly for those speakers whose data are not well-modeled by the system parameters. Some form of automatic speaker adaptation is required for truly speaker-independent recognition.

Speaker adaptive recognition implies limits on the amount of available speaker-specific training data and the time allowed for the adaptation process. To allow a fast enrollment procedure, the adaptation algorithms must be able to operate on a relatively small amount of speaker-specific training data, and they must not incur a large computational load. The data requirements must be kept well below those for training speaker-dependent systems to justify the use of adaptation. Given this limited set of observations, the adaptation algorithm should be able to exploit any and all information available in the data quickly, and without sacrificing accuracy for speed.

A second restriction on the adaptation algorithm is imposed by the continuous speaking style. With continuous speech, user feedback of phonetic or other subword recognition unit labels is unrealistic, if not impossible. The adaptation algorithm must therefore operate in an unsupervised mode.

A brief overview of adaptation methods reported in the literature follows. This review is intended to be a representative sample of the techniques applied to speaker adaptation, and not a comprehensive list or discussion. Note that adaptation methods are strongly tied to the particular system's architecture, whether it be a template matching, stochastic model-based, feature-based, or neural network system. Differences in the baseline systems, plus differences in the degree of speaker independence of these systems make it difficult to directly compare performance gains due to adaptation.

1.1.1. Methods of Speaker Adaptation

Approaches to speaker adaptation fall into two broad categories, speaker mapping or selection, and parameter modification. Speaker selection and mapping techniques form a new speaker's recognition parameters by combining the parameters of a set of reference speakers, or selecting the parameters of a reference speaker whose parameters are close in some sense to the new speaker's. As opposed to forming the new speakers' parameters from the reference set, parameter modification approaches use speaker-specific observations to update the reference set.

1.1.1.1. Selection and Mapping Techniques

The mapping approach attempts to find a transformation between a set of reference parameters and the parameters obtained from a new speaker. The transformation may be in the form of a confusion matrix which maps individual templates onto the reference, or by cluster selection (multiple-template method) where a set of templates is selected based on minimum distance criteria. Representative selection and mapping techniques in hidden Markov model, dynamic time-warping, and neural network recognitions are presented in the remainder of this section.

In the BBN speaker-dependent continuous speech recognition system, Schwartz *et al.*, [1] used a probabilistic spectral transformation to map parameters of context-dependent HMMs for a prototype speaker to those of a new speaker. The SPHINX system uses a hidden Markov model (HMM) with discrete state output probability densities, where the output symbols are indices of vector quantization (VQ) codebook entries. With 15 seconds of normalization data, they achieved word recognition rates of 97%, a gain of 17% over the unadapted accuracy. This method did not perform as well with a less constrained grammar. Feng, *et al.* [2] subsequently improved performance in less constrained tasks through the use of a text-dependent alignment of the reference and new speaker's speech. Iterative application of this alignment procedure was shown to reduce word error rates even further [3].

Work by Nishimura and Sugawara [4] and Furui [5] has demonstrated the need for separate modeling of male and female speakers. Similar to the work of Feng, *et al.*, Nishimura and Sugawara applied a spectral mapping procedure to an isolated word recognition task using a discrete HMM. In cross-speaker adaptation of their speaker-dependent system, they noted that females used only 70% of the VQ codebook entries from male reference speakers. parameters for male and female speakers. Furui [5] also performed codebook mapping in a discrete HMM, using separate codebooks for males and females. In a 150-word isolated city name recognition task, codebook adaptation reduced error rates from 4.9% to 2.9% after presentation of 10 speaker-specific training tokens.

Lee [6, 7] investigated speaker adaptation in the SPHINX recognition system, which has demonstrated a speaker-independent word accuracy of better than 94% on a 1000-word task with a perplexity of 60[1]. Adaptation was attempted using codebook and model selection as well as reestimation of model parameters. Due to the speaker-independent nature of the system parameters, adaptation via the selection method showed no improvement. The reestimation method provided a 5-10% reduction in word error rate for the no-grammar case; with a grammar the improvement was smaller.

Adaptation by template selection is most appropriate for recognition systems based on template matching algorithms. Template matching algorithms seek to find the correspondence between an input utterance and a set of reference templates. Comparisons between spectra are made via dynamic time-warping (DTW) on a frame by frame basis. Adaptation may be achieved by using multiple reference templates for each word [8]. Rabiner, *et al.* [9] applied clustering techniques on a database of consisting of instances of isolated words from many speakers. The multiple clusters per word were assumed to represent the acoustic variability between speakers. Kijima, *et al.* [10] used a recognition rate criterion to select a set of speakers with characteristics similar to the new speaker. The adapted reference templates were formed by an average the templates of the selected reference speakers, or by the union of their templates. Use of averaged templates was shown to produce lower error rates than using multiple templates per word.

Other adaptation methods for template-based systems involve codebook mapping or spectral transformation. In an LPC vector-quantization-based isolated word recognition system, Shikano, *et al.* [11] performed adaptation by substitution of a reference speaker's codebook vectors for the vectors in the input speaker's codebook. Various substitution and learning algorithms were implemented; the best combination raised the recognition rate from 64% to 83%. Higuchi and Yato [12] performed adaptation by estimating frequency-axis scale factors which transformed a new speakers' vowel spectra onto the reference speaker's templates. As expected, gains were largest when the scale factors were much larger or smaller than 1.0.

Mapping techniques have also been applied to neural network phonetic classifiers. Hampshire and Waibel [13] demonstrated a time-delay neural network (TDNN) stop consonant classifier which learned to form weighted combinations of six reference classifier outputs to obtain recognition scores for new speakers. The adaptive recognition rate was significantly improved over that of a single TDNN trained on the speech from the six reference speakers.

[1]Perplexity can be thought of as the average branching factor of the grammar.

1.1.1.2. Parameter Modification Techniques

The techniques described above are used to select or build parameters for a new speaker from a set of reference parameters. Parameter modification approaches to speaker adaptation, on the other hand, update the current system parameters based on the observations. Interpolation between speaker-independent and speaker-specific parameters is a common theme amongst parameter modification approaches. Interpolation provides a certain degree of robustness to the new parameters by reducing the effects of insufficient training data. Adaptation by parameter modification in hidden Markov model and feature-based recognition systems is reviewed in the following.

Parameters in hidden Markov model-based systems represent statistically both the temporal and spectral variations between speakers. Hon and Huang [14] have shown that HMM word error rates are insensitive to changes in transition (temporal) probabilities. Adaptation of the spectral or observation parameters, however, has been shown to provide a significant reduction in word error rates. For discrete HMMs, the observation densities are represented by the locations of the codebook entries and the state-dependent probabilities assigned to them. In continuous-density HMMs (CDHMMs) using a gaussian model, the observation densities are represented by state-dependent mean vectors, covariance matrices, and mixture coefficients. Results reported in the literature demonstrate that adaptation of the densities' covariance matrices has little effect on recognition accuracy [15, 16]. Parameter adaptation in HMMs has therefore focused mainly on adaptation of mean vectors and mixture coefficients [15, 17, 16, 18].

HMM parameter modification typically involves incremental training followed by interpolation. Incremental training continues the reestimation process to locally adjust system parameters given speaker-specific observations. Interpolation methods may be based on Bayesian techniques [16, 17], or additional iterations of the forward-backward algorithm, as is the case with deleted interpolation [6]. For example, C-H Lee [16] used a Bayesian procedure to update means and covariance matrices in a CDHMM. This isolated-word system used 5-state word models with as many as 9 mixture components per state for a 39 word task. Diagonal covariance matrices were used to reduce the number of parameters to be estimated. For telephone bandwidth speech, a reduction in error rate of approximately 1/3 was obtained after one observation (a single word), and the error rate after 7 tokens approached speaker dependent performance.

In a 35-word, isolated-utterance recognition task, Martin *et al.* [19] adapted means and transition probabilities in a cross-speaker, supervised adaptation task. In separate experiments, models were adapted after correct recognition of an utterance or after any utterance. Since adaptation using only correctly-recognized utterances did not allow large modifications to the reference models, performance improvements in this case were limited. When adapting after all input tokens, the large model changes (necessary in cross-speaker adaptation) allowed performance to approach speaker-dependent levels after observation of 1-2 tokens.

Rtischev [17] and Huang [20] have investigated an alternate HMM architecture which bridges the gap between discrete and continuous HMMs. In this semi-continuous HMM (SCHMM), the VQ codebook entries become the mean vectors of mixture density components, and an associated covariance matrix is computed. In each state of the SCHMM, the continuous output density function is modeled as a mixture density where the continuous codebook's component densities are combined using the discrete HMM's output pdf as the mixture coefficients. The semi-continuous HMM reduces the number of parameters to be estimated with respect to the full CDHMM, but allows more accurate modeling of the states' output pdfs than the discrete HMM.

In adaptation experiments with a semi-continuous HMM, Rtischev [17] used the forward-backward algorithm to modify the prior codebook parameters after obtaining new speaker data. Using the SCHMM for adaptive training and the 5000 word, speaker-dependent CDHMM IBM system for decoding, Rtischev achieved appreciable reductions in error rate (≈ 50%) when adapting the reference speaker's models to the new speaker. Rtischev also investigated Bayesian adaptation of codebook parameters and obtained results (≈ 40% reduction in error rate) almost as good as with re-estimation. He also observed that continuous adaptation produced about the same results as adaptation after observation of all training data.

Huang, *et al.* [18] have investigated adaptation of codebook mean vectors and state mixture coefficients in a semi-continuous HMM version of the speaker-independent SPHINX system. Adaptation was performed by using the forward-backward algorithm to update the speaker-independent parameters given the speaker-specific observations. Speaker-adaptive word error rates on the order of 3% have been obtained after observation of 40 adaptation training sentences, for a test set of 4 speakers. This represented a reduction in word error rate of 25% over the baseline system. Separate speaker-independent parameters for male and female speakers were used in the baseline system. Use of these gender-specific models was claimed to reduce the baseline system error rates by 30% before adaptation [21].

Brown *et al.*, [22] used Bayesian estimation procedures to adapt HMM parameters in a continuous digit recognition system. Compared with maximum likelihood updating in tests on variable length digit strings, the Bayesian update reduced the error rate by 18% for isolated-digit training data, but raised it by 0.3% for digit-triplet training data. No reasons for this counterintuitive behavior were determined. Brown *et al.* also reported that performance was better for initialization with speaker independent parameters than with parameters derived from a short speaker-dependent training session.

In a feature-based English letter recognition system called FEATURE, Stern and Lasry [23] used Bayesian techniques to update classifier parameters. The features were assumed to be samples of jointly Gaussian random variables, and Gaussian classifiers in a decision-tree structure were used for classification. Extended MAP (EMAP) estimates [24] of classifier mean vectors were computed at each node visited during classification of an utterance. The term extended refers to the fact that the EMAP algorithm exploits the covariance between classes to update all classes at a node after each observation. This reduced the error rate by 49% after a supervised training session, and by 31% when used in an unsupervised mode.

1.2. Research Objectives

The present work adopted the parameter modification approach to speaker adaptation. The objective of this effort was to develop an accurate and efficient approximation of the EMAP mean vector estimation algorithm, and to determine its effectiveness in reducing error rates in both feature-based and stochastic model-based continuous speech recognition systems.

Prior work with the FEATURE system [23] and continuous-density hidden Markov model recognition systems [16], [18], [15], [17] demonstrated that a large portion of the differences between speakers may be modeled by a shift in the system mean values. Mean vectors are also less sensitive than other parameters (such as covariance matrices) to the effects of training on limited numbers of speaker-specific training data [15]. The work with FEATURE also demonstrated that the EMAP procedure possessed a number of desirable properties such as fast convergence. The cost of these properties was greater storage requirements and computational complexity. To satisfy the limited computation time and data constraints of speaker adaptive recognition, a faster implementation of the EMAP procedure was necessary.

Motivated by a desire to better understand the relationships between parameter estimation and adaptive filtering techniques, the specific goal of the present work was to recast the EMAP estimation procedure as an adaptive filtering problem to obtain a more efficient implementation. From estimation theory it is well-known that MAP and minimum mean square error (MMSE) estimation techniques are equivalent for normally-distributed data [25]. In adaptive filter theory, the computationally efficient least mean square (LMS) transversal filter is derived as an approximation of MMSE formulations. Expressing the Bayesian estimate as an MMSE adaptive filter led to the development of an estimation method (called LMS-C) which was more computationally efficient than the Bayesian optimal estimate and yet retained the desirable properties of the latter.

Initial work toward the goals of the present study involved extending the FEATURE results for isolated word recognition to the continuous speech case, in the context of the feature-based CMU ANGEL system. Continuous speech is subject to a host of coarticulation and end-of-utterance effects not present in isolated speech. The number of words in an utterance and the temporal locations of word boundaries, if any, are also unknown. The ANGEL system therefore used the phoneme, not the word, as the basic recognition unit, and a lexicon which described how words are composed from these units. The features used for phonetic classification in ANGEL were similar to those from FEATURE, but they tended to exhibit much greater variability due to the effects which are characteristic of connected speech. The effectiveness of mean vector adaptation was investigated in light of these differences. When the ANGEL system was retired in 1989, the focus of the feature-based applications shifted to applying LMS-C adaptation techniques in the context of alternate feature-based phonetic classifiers developed by the author.

The LMS-C algorithm was conceived in the context of feature-based recognition systems, and it is most easily described and understood in the context of such classification problems. With a slight change of perspective it is clear that the LMS-C algorithm may also be applied to adaptation in continuous-density hidden Markov model-based systems. In the HMM context, the component densities of gaussian mixtures replace the decision classes of the gaussian classifier. Since the HMM reestimation procedure computes maximum likelihood estimates of the HMM parameters, EMAP or LMS-C techniques can be used to interpolate these ML parameters with the speaker-independent or reference parameters.

The semi-continuous HMM reduces the number of parameters to be estimated with respect to the full continuous-density HMM. Because mixture components are shared among all states, the SCHMM can makes more effective use of training data. Previous research in SCHMM codebook adaptation used incremental training with maximum likelihood techniques, or Bayesian estimation with diagonal covariance matrices which ignore correlation information. Using a semi-continuous version of SPHINX as the baseline system, the present work investigates the use of correlation information and the LMS-C algorithm in the reestimation of codebook mean vectors.

1.3. Dissertation Outline

In Chapter 2, a fast multivariate parameter estimation procedure, in the form of an adaptive filter, is derived. The result is a least mean square algorithm, referred to as LMS-C, which produces a faster rate of convergence than the Bayesian MAP estimator at the expense of a finite bias or misadjustment. Expressions that specify the LMS-C adaptive estimate, as well as analytical expressions for the expected mean-square error and misadjustment are derived. LMS-C parameter values which allow this estimate to incorporate *a priori* statistics of the data are also specified. The ability of this adaptive filter implementation to estimate the mean values of normally distributed random vectors which are drawn from one of several possibly correlated classes is then evaluated. LMS-C learning curves are compared to the learning curves specified by analytical expressions of the mean-square error of both the EMAP and maximum likelihood (ML) estimators. The performance of each estimator is then demonstrated empirically using synthetic, normally distributed data exhibiting varying degrees of correlation and dogmatism. The computational requirements of each algorithm are also discussed.

The analysis in Chapter 2 assumes data are labeled, *i.e.* supervised adaptation. The results demonstrate that the LMS-C estimation algorithm can improve estimation accuracy with the same number or fewer observations than previous methods. Chapters 3 and 4 describe the effects the LMS-C estimation procedure has on the accuracy of both feature-based and stochastic model-based systems when operated in an unsupervised mode. Enabling these systems to quickly estimate the parameters that describe the acoustical characteristics of individual speakers should increase classification accuracy.

Chapter 3 describes adaptation at the phonetic classification level in feature-based continuous recognition systems. The chapter begins by briefly reviewing spectrogram reading and the Carnegie Mellon FEATURE and ANGEL recognition systems. The ANGEL system attempted to extend the work of the isolated-word FEATURE system to the continuous speech case. Chapter 3 continues with a description of our adaptation methodology in the context of feature-based phonetic classification. This is followed by a brief empirical study on the effect of adaptation on classification rate under various data conditions using a synthetic model from Chapter 2. Results from supervised adaptation of front vowel features in the ANGEL system are presented, along with a description of limitations of the ANGEL classification procedure. To address these limitations and demonstrate the effectiveness of LMS-C adaptation, an alternate classification structure, called PROPHET, is proposed. PROPHET uses simple rules derived from spectrogram reading combined with statistical classification techniques to make classification decisions. Unsupervised adaptation within PROPHET, using real and computer-generated data, is described.

Chapter 4 extends the application of the LMS-C algorithm to adaptation of continuous density hidden Markov models. Described are adaptation experiments with a semi-continuous version of the CMU SPHINX system, called SPHINX-SC. These experiments determine the extent to which the word error rate can be reduced by applying mean vector estimation techniques to adaptation of the mixture components in the semi-continuous codebook. Evaluations are based on the change in word error rate between the adapted and unadapted systems. The change in error rate varies widely between speakers. Methods are described which may be used to select the adapted codebooks of only those speakers expected to benefit from adaptation. Finally, to obtain a better understanding of the SPHINX-SC results, experiments with a synthetic hidden Markov model using computer-generated data are described.

Chapter 5 summarizes the results of this dissertation, and its conclusions. Suggestions for future research are also presented.

Chapter 2

Fast Estimation of Mean Vectors Using Adaptive Filtering

2.1. Overview

Estimation of mean vectors is an essential procedure in the training of both feature-based and stochastic model-based speech recognition systems. In both types, knowledge of the data's probabilistic structure is typically incomplete. In order to model this structure, the data is usually assumed to obey a particular set of parametric density functions based on general knowledge of the problem. Feature-based systems may use Gaussian or other parametric classifiers to analyze acoustic cues or feature vectors, while recognizers incorporating hidden Markov models often make use of continuous normal mixture densities to model the states' output probability distributions. Given these assumptions, the problem of accurately modeling the data reduces to one of reliably estimating the parameters of the assumed densities.

Current mean vector estimation algorithms trade accuracy or convergence rate for computational efficiency. While the maximum likelihood (ML) estimate of a multivariate mean vector is the most computationally efficient, it ignores information which may be available through knowledge of the correlation between decision classes. The extended MAP (EMAP) algorithm [24] increases adaptation speed by using this information to update the means for all classes after an observation from any class. Unfortunately, the EMAP algorithm is computationally expensive, especially when the product of the number of classes and dimensions is large. Given the constraints of limited time and limited training imposed by the speaker adaptation problem, a more computationally efficient form of the EMAP procedure is desired.

The learning and tracking abilities of adaptive filtering techniques make them attractive for estimation problems. The least mean-square (LMS) algorithm [26], which is a stochastic approximation of the minimum mean-square error (MMSE) algorithm, is the simplest and most widely-used algorithm for adjusting the weights in an adaptive system. Its simplicity and computational advantages make the LMS algorithm the first to be examined in these types of problems. Since MMSE and MAP estimates are equivalent for normally distributed data, the MAP algorithm can be cast into a form amenable to approximation by LMS techniques by deriving an MMSE estimate for mean vectors. By judicious initialization of LMS parameters, it is possible to incorporate *a priori* statistics into this algorithm, reducing the initial estimation error.

In this chapter, a fast multivariate parameter estimation procedure in the form of an adaptive filter is derived. As a basis for comparison, expressions of the ML and EMAP estimates and their mean-square error (MSE) are derived in Secs. 2.4 and 2.5. . The EMAP algorithm is recast as an MMSE adaptive filter by deriving the MMSE mean estimate and proving that it is equivalent to the EMAP expression in Sec. 2.6. This MMSE implementation is then approximated by an LMS procedure in Sec. 2.7. Expressions for the LMS MSE are derived, and methods for selection of parameters required by the LMS procedure are also discussed in this section. In Sec. 2.8 the ability of this adaptive filter implementation to estimate the mean values of normally distributed random vectors which are drawn from one of several possibly correlated classes is evaluated. Its performance is compared with the ML and optimal estimates, using computer-generated normally-distributed data exhibiting varying degrees of correlation and dogmatism. Finally, the computational requirements of each algorithm are compared in Sec. 2.9.

Derivation and analysis of the adaptive estimator are preceded by a review of basic adaptive filtering principles. Readers familiar with these concepts may wish to skip to Sec. 2.3.

2.2. Review of Adaptive Filtering Principles

An adaptive filter can be specified by a filter structure and an adaptation algorithm. For the purposes of the present study, the filter structure will be limited to the class of discrete-time, linear, shift-invariant systems. Such a structure typically has a finite number of internal parameters which can be modified to control the transfer function of the system. In Figure 2-1, which depicts a type of adaptive filter known as a joint process estimator, the internal parameters are represented by the matrix $\mathbf{H}$. The adaptation algorithm allows the filter to track some measure of the external environment, which in this example is the signal $\mathbf{a}_k$. In a joint process estimator, the coefficients in $\mathbf{H}$ are adjusted so as to minimize the difference between the output of the linear system, $\hat{\mu}$, and the desired output $\mathbf{d}_k$. In other words, the filter is adjusted to generate an estimate of $\mathbf{d}_k$ based on the samples of $\mathbf{a}_k$.

Consider a 1-dimensional example of a joint process estimator. In this case, the error signal may be expressed as $\varepsilon_k = \mathrm{d}_k - \hat{\mu}$ where $\hat{\mu} = \mathrm{h}_k \mathrm{a}_k$. It is usually assumed that the quantities ε_k, d_k, and a_k are statistically stationary. Define the mean-square error as $E[\varepsilon_k^2]$. The MSE may be rewritten as

$$\begin{aligned} E[\varepsilon_k^2] &= E[\mathrm{d}_k^2] + \mathrm{h}_k^2 E[\mathrm{a}_k^2] - 2E[\mathrm{d}_k \mathrm{a}_k]\mathrm{h}_k \\ &= E[\mathrm{d}_k^2] + \mathrm{h}_k^2 \mathrm{R} - 2\mathrm{Ph} \end{aligned} \tag{2.1}$$

where $\mathrm{R} = E[\mathrm{a}_k^2]$ and $\mathrm{P} = E[\mathrm{d}_k \mathrm{a}_k]$. The MSE expression is a quadratic function of the coefficient h_k. The MSE as a function of the filter coefficients is known as the performance surface. Adaptive algorithms specify methods of searching the performance surface for the coefficient value h^* which produces the minimum error.

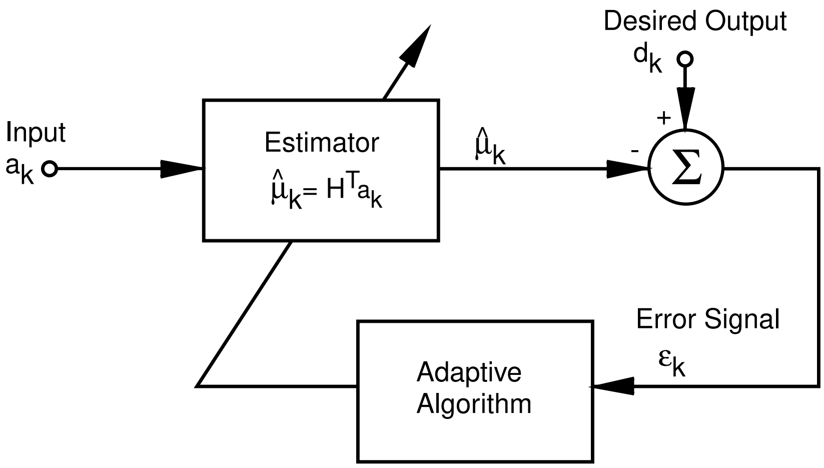

Figure 2-1: A joint process estimator adaptive filter.

If the ensemble statistics R and P are known, then Equation (2.1) may be solved directly for h^*. This solution, $h^* = R^{-1}P$, is the minimum mean-square error (MMSE) solution. An alternate method, the MMSE gradient algorithm, performs a gradient search for the optimal coefficient. The gradient algorithm updates the coefficient h_k according to

$$h_{k+1} = h_k - \frac{\beta}{2}\nabla_{h_k} E[\varepsilon_k^2] \tag{2.2}$$

where β an adaptation constant which limits the change in h_k at each step. It can be shown that equation (2.2) is stable and convergent for $\beta < 2/\lambda_{max}$ where λ_{max} is the maximum eigenvalue of R (equal to R in the 1-D case).

If ensemble statistics R and P are unavailable, then the MMSE solution must be approximated in some fashion. The simplest and most common approximation is the least mean-square (LMS) algorithm. The LMS algorithm substitutes instantaneous values for the gradient in (2.2), which effectively replaces ensemble averages with time averages. The LMS coefficient update equation is

$$\begin{aligned} h_{k+1} &= h_k - \frac{\beta}{2}\nabla_{h_k}\varepsilon_k^2 \\ &= (1 - \beta a_k^2)h_k + \beta a_k d_k \end{aligned} \tag{2.3}$$

The convergence of the expected value of the LMS coefficients is identical to that of the MMSE. Individual coefficient trajectories (on the performance surface) exhibit a random fluctuation about the optimal due to the random driving term in (2.3). This misadjustment can be controlled by varying the step size β.

An example of the behavior of the LMS algorithm is shown in Figure 2-2 for the one-dimensional example. The parabolic performance surface is the MSE as a function of the filter coefficient h, and the minimum occurs at $h^* = 1.0$. The coefficient trajectory (the zigzag line) represents the time history of the coefficient as it searches for the optimal value. When the coefficient is near h^*, the stochastic update specified by (2.3) causes it to vary randomly around the optimal value.

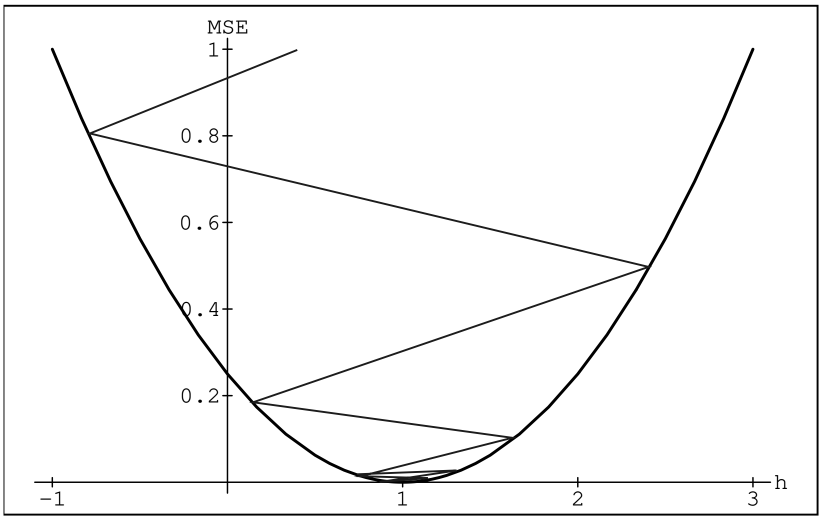

Figure 2-2: Example performance surface and an LMS coefficient trajectory.

Note that when the statistics R and P are constant, the performance surface and optimal coefficient value are also constant. It will be shown in Section 2.6 that the EMAP procedure is equivalent to MMSE adaptive filtering with a time-varying optimal coefficient matrix and performance surface. Then, in Section 2.7, an LMS procedure which performs a stochastic gradient search over this time-varying surface is derived.

2.3. Problem Statement and Assumptions

Consider a pattern-classification problem with C decision classes and D features, or a C-component mixture in D dimensions. Let the set of input data for the j^{th} class be $\{\mathbf{x}_{j,1}, \mathbf{x}_{j,2}, \ldots, \mathbf{x}_{j,n_j}\}$[2]. The data may be cepstral coefficients derived from a portion of the speech waveform in an HMM system, or it may be an acoustic feature vector in a feature-based recognition system. It is assumed that the random vector $\mathbf{x}_j$ is normally distributed about a mean vector μ_j with a covariance matrix Σ_j, *i.e.* $p(\mathbf{x}_j|\mu_j) \approx N(\mu_j,\Sigma_j)$, and that all data samples are independent. The class mean vectors $\{\mu_j\}$ are assumed to be jointly Gaussian random variables which are correlated across decision classes. Defining the overall mean vector as $\mu=[\mu_1^T, \mu_2^T, \ldots, \mu_C^T]^T$, this CD-dimensional vector is assumed to be normally distributed around the *a priori* mean vector μ_o with CD*x*CD covariance matrix Σ_o, so $p(\mu) \approx N(\mu_o,\Sigma_o)$. The statistics $\{\Sigma_j\}$, Σ_o, and μ_o are assumed to be known, or they can be estimated from a larger body of training data. Denote the set of observations from a single speaker as $\chi=\{\mathbf{x}_{1,1}, \ldots, \mathbf{x}_{1,n_1}, \mathbf{x}_{2,1}, \ldots, \mathbf{x}_{2,n_2}, \mathbf{x}_{C,1}, \ldots, \mathbf{x}_{C,n_C}\}$. The parameter estimation problem considered here is the production of reliable estimates of the parameter μ given the information provided by the samples in χ and the *a priori* statistics $\{\Sigma_j\}$, Σ_o, and μ_o.

[2]All notational conventions used in the present study are summarized in the Nomenclature section at the end of the thesis.

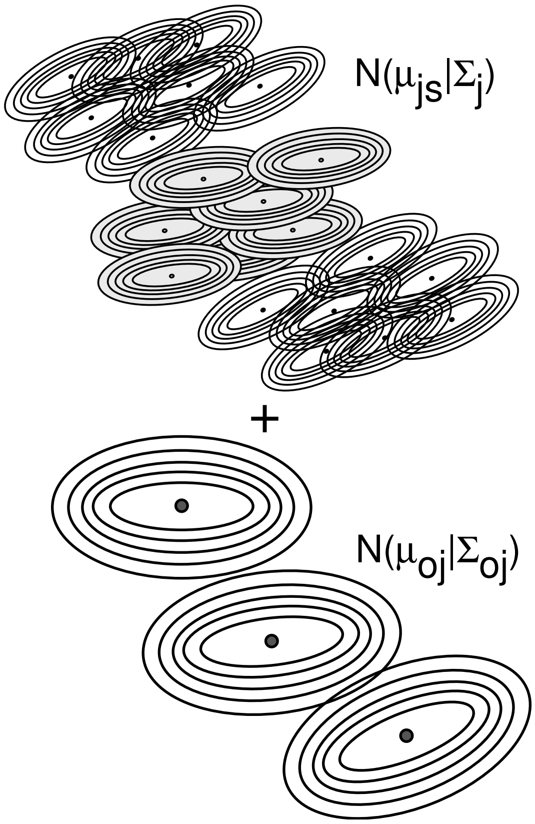

Figure 2-3: Schematic diagram illustrating assumed distributions of the data.

The distributions of the data defined above are illustrated schematically in Figure 2-3. The upper left portion shows three classes (j) of data from multiple speakers (s), where each ellipse represents $p(\mathbf{x}_{js}|\mu_{js})$. The lower left portion shows the distributions describing the dispersion of the speakers' means about μ_{oj}. The righthand portion of the figure shows how the individual speakers' distributions combine with the mean distributions to form the pooled distribution ot the data. Note that speaker-independent classifiers are designed to minimize errors given only these pooled distributions which may poorly represent an individual's data.

Knowledge of the correlation between decision classes is modeled by the mean crosscovariance matrix Σ_o. In a phonetic classification task this matrix might, for example, describe how the acoustic cues of the front vowels /*iy*/, /*ih*/, and /*eh*/,[3] related due to similar positioning of the vocal apparatus, manifest themselves in measures of those cues. The matrices $\{\Sigma_j\}$ model the within-class correlation between features. The matrix Σ is defined as a CD-dimensional block diagonal matrix with Σ_j as the j^{th} block. Σ is is block diagonal due to the assumption of independent observations. In the phonetic classifier example, this assumption means that individual realizations of phonemes are not correlated. The validity of this assumption depends on the degree to which contextual effects manifest themselves in the feature data.

[3]All speech-related terms are given in Appendix B.

Another property of the assumed distributions is dogmatism, which is of extreme importance to the estimation methods described in this chapter. Dogmatism is defined as the ratio of the within-class variance (Σ) to the between-speaker variance (Σ_o). In the one-dimensional case, dogmatism is computed as σ/σ_o. For the D-dimensional cases in the present study, dogmatism will be computed as

$$\delta = \frac{1}{CD}\sum_{i}^{CD}\frac{\sqrt{\Sigma_{ii}}}{\sqrt{\Sigma_{oii}}} \tag{2.4}$$

Dogmatism is a strong indicator of the expected degree of success of adaptation, as will be shown. Recall the pooled distributions of the data from Figure 2-3 and the 1-D representations in Figure 2-4. These distributions are dependent on both Σ and Σ_o. Consider the two extremes. When $\Sigma << \Sigma_o$ (the small-dogmatism case), most of the variation of the pooled data is due to the distribution of the mean vectors; observations under this condition convey a great deal of information about the mean value being estimated. When $\Sigma >> \Sigma_o$ (the large-dogmatism case), most of the pooled data variation is due to the within-class variance, and these observations provide little information regarding the individual's mean. Obviously as the dogmatism increases, adaptation becomes increasingly ineffective. It is actually advantageous to choose features in a speaker-independent recognition system which exhibit large dogmatism. Since the speaker-independent classifier is built from the pooled data, it is best if most of the variation of that data is accounted for by the within-class variance. In this case most speakers have very similar distributions of the data which are well-modeled by the speaker-independent classifier.

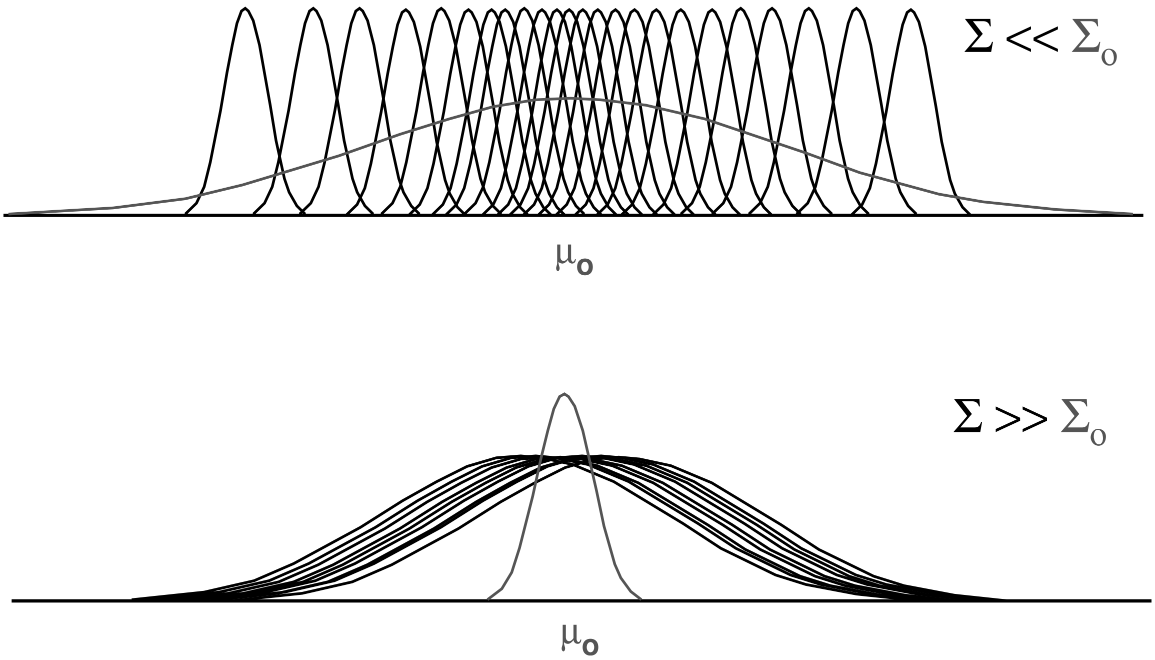

Figure 2-4: Distributions of the data exhibiting small and large dogmatism, in the upper and lower panels, respectively. The distributions in black represent distributions of the data of individual speakers while the lighter curves represent the distributions of the mean values of the speakers' data.

2.4. Maximum Likelihood Estimation

Maximum likelihood procedures treat the unknown parameters as fixed and seek to maximize the probability of the observed data with respect to the parameters. For mean vectors the ML estimate is the value of μ at which the likelihood function $p(\chi|\mu)$ is greatest. Given the assumption of independent observations, this likelihood may be written as

$$p(\chi|\mu) = \prod_{j=1}^{C}\prod_{k=1}^{n_j} p(\mathbf{x}_{j,k}|\mu_j) \tag{2.5}$$

Define the log-likelihood as $l(\mu) = log[p(\chi|\mu)]$, or

$$l(\mu) = \sum_{j=1}^{C}\sum_{k=1}^{n_j} log[p(\mathbf{x}_{j,k}|\mu_j)] \tag{2.6}$$

Taking the gradient with respect to μ of $l(\mu)$ and dropping constants which are independent of μ gives

$$\nabla_{\mu} l(\mu) = -\frac{1}{2}\nabla_{\mu}\sum_{j=1}^{C}\sum_{k=1}^{n_j}(\mathbf{x}_{j,k}-\mu_j)^T\Sigma_j^{-1}(\mathbf{x}_{j,k}-\mu_j) \tag{2.7}$$

Since $\nabla_{\mu} = [\nabla_{\mu_1}\nabla_{\mu_2}\ldots\nabla_{\mu_C}]$, we can consider each class m in turn. Looking at the double sum in $\nabla_{\mu_m} l(\mu)$, the summand can be ignored except when $j = m$. In this case,

$$\nabla_{\mu_m} l(\mu) = -\frac{1}{2}\nabla_{\mu}\sum_{k=1}^{n_m}(\mathbf{x}_{m,k}-\mu_m)^T\Sigma_m^{-1}(\mathbf{x}_{m,k}-\mu_m) \tag{2.8}$$

Setting this gradient to zero yields

$$\nabla_{\mu_m} l(\mu) = \sum_{k=1}^{n_m}\Sigma_m^{-1}(\mathbf{x}_{m,k}-\mu_m) \equiv 0 \tag{2.9}$$

Solving for μ_m results in the ML estimate

$$\hat{\mu}_m = \frac{1}{n_m}\sum_{k=1}^{n_m}\mathbf{x}_{m,k} \equiv \mathbf{a}_m \tag{2.10}$$

Note that this estimate has the same form as the familiar case in which the class means are assumed to be independent. The ML estimate has the desirable statistical properties of being unbiased and consistent,[4] and it requires very little computation. It also does not require the estimation of *a priori* statistics. However, since each element of each class' mean vector is estimated independently, it does not take advantage of information present given knowledge of the correlation between features and classes. Consequently, under certain data conditions it may converge more slowly than more sophisticated algorithms.

It can be seen that the correlation does not contribute to convergence in the ML estimate by inspection of the learning curves, which are defined as the mean-square error as a function of the number of observations. For the vector parameters considered here, the norm of the error vector will

[4]An estimator is unbiased if the expected value of the estimate is equal to the true parameter. An estimator is consistent if it converges to the true parameter as the number of observations increases.

be minimized. Defining the error vector as $\varepsilon = \mu - \hat{\mu}$, the mean-square error is then $E\{\|\varepsilon\|^2\} = E\{\|\mu - \hat{\mu}\|^2\}$, which for the ML estimate can be written as

$$\begin{aligned} MSE_{ML}(\mathbf{N}) &= E\{(\mu - \mathbf{a})^T(\mu - \mathbf{a})\} \\ &= E\{\mu^T\mu\} - 2E\{\mu^T\mathbf{a}\} + E\{\mathbf{a}^T\mathbf{a}\} \\ &= tr[E\{\mu\mu^T\}] - 2tr[E\{\mathbf{a}\mu^T\}] + tr[E\{\mathbf{a}\mathbf{a}^T\}] \\ &= tr[\Phi_{\mu\mu}] - 2tr[\Phi_{\mathbf{a}\mu}] + tr[\Phi_{\mathbf{aa}}\}] \end{aligned} \tag{2.11}$$

The matrix **N** is defined as a diagonal matrix with the number of observations per class along the diagonal, or in other words with $n_j\mathbf{I}$ as the C diagonal blocks of dimension D. The correlation matrices Φ_{aa}, $\Phi_{\mu\mu}$ and $\Phi_{a\mu}$ are defined as $E[\mathbf{aa}^T]$, $E[\mu\mu^T]$, and $E[\mathbf{a}\mu^T]$, respectively. As shown in Appendix A, these matrices may be expressed in terms of μ_o, Σ_o, and Σ as $\Phi_{\mathbf{aa}} = \Sigma_o + \mathbf{N}^{-1}\Sigma + \mu_o\mu_o^T$, $\Phi_{\mathbf{a}\mu} = \Sigma_o + \mu_o\mu_o^T$, and $\Phi_{\mu\mu} = \Sigma_o + \mu_o\mu_o^T$, respectively. Substituting these values into (2.11) yields

$$\begin{aligned} MSE_{ML}(\mathbf{N}) &= tr[\Sigma_o + \mathbf{N}^{-1}\Sigma + \mu_o\mu_o^T - (\Sigma_o + \mu_o\mu_o^T)] \\ &= tr[\mathbf{N}^{-1}\Sigma] \end{aligned} \tag{2.12}$$

which clearly shows that the between-class correlation (Σ_o) and within-class correlation (the off-diagonal elements of Σ) do not contribute to reducing the mean-square error.

2.5. Extended Maximum *A Posteriori* Estimation

In maximum *a posteriori* (MAP) parameter estimation, the unknown parameter is treated as a random quantity with a known *a priori* distribution and the probability of the parameter given the data is maximized with respect to the parameter. The term *extended* is historical and was applied specifically to the multi-class MAP mean vector estimate [24]. The EMAP estimate for μ is the value at which the *a posteriori* probability

$$p(\mu|\chi) = \frac{p(\chi|\mu)p(\mu)}{p(\chi)} \tag{2.13}$$

attains its maximum, where χ is the set of observations from a single speaker. Taking the gradient with respect to μ of the natural logarithm of $p(\mu|\chi)$,

$$\nabla_\mu log[p(\mu|\chi)] = \nabla_\mu log[p(\chi|\mu)] + \nabla_\mu log[p(\mu)] - \nabla_\mu log[p(\chi)] \tag{2.14}$$

The third term in (2.14) is zero because $log[p(\chi)]$ is independent of μ. The second term can be written as (ignoring constants independent of μ)

$$\begin{aligned} \nabla_\mu log[p(\mu)] &= -\frac{1}{2}\nabla_\mu(\mu - \mu_o)^T\Sigma_o^{-1}(\mu - \mu_o) \\ &= -\Sigma_o^{-1}(\mu - \mu_o) \end{aligned} \tag{2.15}$$

The first term in (2.14) is identical to (2.7). Looking at each class *m* separately, (2.9) gives

$$\begin{aligned} \nabla_{\mu_m} log[p(\chi|\mu)] &= \sum_{k=1}^{n_m} \Sigma_m^{-1}(\mathbf{x}_{k,m} - \mu_m) \\ &= \Sigma_m^{-1} n_m(\mathbf{a}_m - \mu_m) \end{aligned} \tag{2.16}$$

So, considering all classes again, (2.16) gives

$$\nabla_{\mu} log[p(\chi|\mu)] = \Sigma^{-1}\mathbf{N}(\mathbf{a}-\mu) \tag{2.17}$$

Setting the sum of (2.17) and (2.15) equal to zero yields,

$$\nabla_{\mu} log[p(\mu|\chi)] = \Sigma^{-1}\mathbf{N}(\mathbf{a}-\mu) - \Sigma_o^{-1}(\mu-\mu_o) \equiv 0 \tag{2.18}$$

Solving for μ results in the EMAP mean vector estimate:

$$\Sigma^{-1}\mathbf{N}(\mathbf{a}-\mu) = \Sigma_o^{-1}(\mu-\mu_o)$$
$$(\Sigma^{-1}\mathbf{N}+\Sigma_o^{-1})\mu = \Sigma^{-1}\mathbf{Na} + \Sigma_o^{-1}\mu_o$$
$$\hat{\mu} = (\Sigma^{-1}\mathbf{N}+\Sigma_o^{-1})\Sigma_o^{-1}\mu_o + (\Sigma^{-1}\mathbf{N}+\Sigma_o^{-1})\Sigma^{-1}\mathbf{Na}$$
$$\hat{\mu} = \Sigma(\Sigma+\mathbf{N}\Sigma_o)^{-1}\mu_o + \Sigma_o(\Sigma+\mathbf{N}\Sigma_o)^{-1}\mathbf{Na} \tag{2.19}$$

where the matrix identity $(\mathbf{A}^{-1}+\mathbf{B}^{-1})^{-1} = \mathbf{A}(\mathbf{A}+\mathbf{B})^{-1}\mathbf{B}$ was used.

The EMAP estimate is a linear combination of the prior mean μ_o and the observations (in **a**). The weights specify the optimal combination of *a priori* information with the knowledge gained from observations. Through Σ_o, the EMAP estimate is able to update all classes after presentation of a sample from any class. When there are few observations or the dogmatism is large, $\hat{\mu} \approx \mu_o$. When **N** becomes large (after many observations), the sample mean dominates. This estimate therefore behaves in an intuitively pleasing manner.

Similar to the ML algorithm, the EMAP mean estimate is consistent and unbiased. It converges at least as fast as the ML estimates of the mean vectors, and, depending on data conditions, it can converge much faster[5]. Drawbacks to this algorithm include estimation of the covariance matrix Σ_o, which requires data from more than C times D speakers to avoid singularity problems. This requirement created difficulties for the CMU FEATURE system. More importantly, the computation required to invert the matrix $(\Sigma+\mathbf{N}\Sigma_o)^{-1}$ after each observation may be prohibitive, especially in cases where the product of the number of features and classes is large. By rederiving this estimate using minimum mean-square error techniques, it is possible to express the EMAP estimate in a form which leads to an approximation which is as accurate but computationally more efficient.

2.6. Minimum Mean-Square Error Estimation

The MMSE estimate minimizes the expected value of the squared error between the parameter and its estimate. As above, the norm of the error vector will be minimized for the vector parameters case. Let the form of the MMSE estimate of μ be

$$\hat{\mu} = \mathbf{H}^T\mathbf{a} \tag{2.20}$$

where **H** is a CD*x*CD coefficient matrix to be determined.

[5]Derivation of learning curves for the EMAP algorithm is deferred to Section 2.6 on MMSE estimation.

The mean-square error vector $E\{\|\varepsilon\|^2\}=E\{\|\mu-\hat{\mu}\|^2\}$ is to be minimized with respect to $\mathbf{H}$. Expand this expression to obtain

$$E\{\|\varepsilon\|^2\}=E\{\mu^T\mu\}-2E\{\mu^T\mathbf{H}^T\mathbf{a}\}+E\{\mathbf{a}^T\mathbf{H}\mathbf{H}^T\mathbf{a}\} \tag{2.21}$$

Setting the gradient of $E\{\|\varepsilon\|^2\}$ with respect to $\mathbf{H}$ equal to zero yields

$$\nabla_{\mathbf{H}}E\{\|\varepsilon\|^2\}=-2\nabla_{\mathbf{H}}[E\{\mu^T\mathbf{H}^T\mathbf{a}\}]+\nabla_{\mathbf{H}}[E\{\mathbf{a}^T\mathbf{H}\mathbf{H}^T\mathbf{a}\}]\equiv 0 \tag{2.22}$$

Taking the gradient inside the expectation and using the identities $\nabla_{\mathbf{M}}\mathbf{a}^T\mathbf{M}\mathbf{b}=\mathbf{b}\mathbf{a}^T$ and $\nabla_{\mathbf{M}}\mathbf{a}^T\mathbf{M}^T\mathbf{M}\mathbf{a}=2\mathbf{a}\mathbf{a}^T\mathbf{M}$ results in

$$-E\{\mathbf{a}\mu^T\}+E\{\mathbf{a}\mathbf{a}^T\}\mathbf{H}=0 \tag{2.23}$$

which is easily solved for the optimal coefficient matrix

$$\mathbf{H}^*=\Phi_{\mathbf{aa}}^{-1}\Phi_{\mathbf{a}\mu} \tag{2.24}$$

The MMSE solution is in the form of the traditional Wiener-Hopf equation [26] except that the parameter Φ_{aa} (and therefore $\mathbf{H}^*$) varies with the number of data samples obtained. It is this dependence of the optimal coefficients on sample counts ($\mathbf{N}$) that forms the basis of the differences between standard adaptive filtering and our parameter estimation techniques. It results in a time-varying optimal solution and a time-varying performance surface and additional terms in the analysis of the LMS approximation to the MMSE solution. The additional terms cause the LMS convergence and stability properties to be dependent on the input data distribution in manners in which traditional LMS has no dependence. The differences between adaptive filtering and parameter estimation will be discussed further in Section 2.7.

2.6.1. Equivalence of the EMAP and MMSE Estimates

The MMSE estimate is the mean of the *a posteriori* density, *i.e.* the conditional mean of $p(\mu|\chi)$. The EMAP estimate is the value of μ at which $p(\mu|\chi)$ has its maximum. If the *a posteriori* pdf is a unimodal function which is symmetric about the conditional mean, as is the case with the multivariate normal density, then these two estimates are equivalent [25]. To demonstrate this equivalence, Φ_{aa} and $\Phi_{a\mu}$ must be expressed in terms of Σ, Σ_o, and μ_o. It was also found to be necessary for a bias term to be incorporated into the MMSE estimate to represent the contribution of the μ_o term in the EMAP expression. Without this bias, it is not possible to show equivalence. By prepending a constant to the sample mean vector, so that $\alpha=[1\ \mathbf{a}^T]^T$, the MMSE algorithm is allowed to compute the optimal weight for this bias term. The notation α to represent the sample mean plus bias will be used throughout the remainder of this thesis.

With the addition of the bias term in α, the data statistics now become

$$\Phi_{\alpha\alpha}=\begin{pmatrix} 1 & E[\mathbf{a}^T] \\ E[\mathbf{a}] & \Phi_{\mathbf{aa}} \end{pmatrix} \quad \text{and} \quad \Phi_{\alpha\mu}=\begin{pmatrix} E[\mu^T] \\ \Phi_{\mathbf{a}\mu} \end{pmatrix} \tag{2.25}$$

In Appendix A, $E[\mathbf{a}]$ is shown to be μ_o, and $E[\mu]$is μ_o. Using the block matrix inversion identity:

$$\begin{pmatrix} \mathbf{A} & \mathbf{C}^T \\ \mathbf{C} & \mathbf{B} \end{pmatrix}^{-1} = \begin{pmatrix} (\mathbf{A}-\mathbf{C}^T\mathbf{B}^{-1}\mathbf{C})^{-1} & -\mathbf{A}^{-1}\mathbf{C}^T\Delta^{-1} \\ -\Delta^{-1}\mathbf{B}^T\mathbf{A}^{-1} & \Delta^{-1} \end{pmatrix} \tag{2.26}$$

where $\Delta^{-1} = (\mathbf{B} - \mathbf{C}\mathbf{A}^{-1}\mathbf{C}^T)^{-1}$, $\Phi^{-1}_{\alpha\alpha}$ becomes

$$\Phi^{-1}_{\alpha\alpha} = \begin{pmatrix} 1+\mu_o{}^T\mathbf{M}^{-1}\mu_o & -\mu_o{}^T\mathbf{M}^{-1} \\ -\mathbf{M}^{-1}\mu_o & \mathbf{M}^{-1} \end{pmatrix} \tag{2.27}$$

where $\mathbf{M} = (\Sigma_o + \mathbf{N}^{-1}\Sigma)$. Substitution of Equation (2.27) into the optimal coefficient equation (2.24) (which holds for α as well as $\mathbf{a}$) results in

$$\mathbf{H}^* = \begin{pmatrix} \mu_o{}^T(\mathbf{I} - \mathbf{M}^{-1}\Sigma_o) \\ \mathbf{M}^{-1}\Sigma_o \end{pmatrix} \tag{2.28}$$

Using this coefficient matrix in (2.20) gives the expression for the MMSE estimate

$$\hat{\mu}_{MMSE} = \Sigma_o\,\mathbf{N}(\Sigma_o\,,+\Sigma)^{-1}\,(\mathbf{a} - \mu_o) + \mu_o \tag{2.29}$$

It can be shown that $\Sigma_o\mathbf{N}(\Sigma_o\mathbf{N}+\Sigma)^{-1} = I-\Sigma(\Sigma_o\mathbf{N}+\Sigma)^{-1}$. Substitution of this identity into (2.29) and rearranging gives the desired result, namely that the coefficients of $\mathbf{a}$ and μ_o in the MMSE and EMAP estimates are the same. Due to the equivalence of these two estimators for the data distributions considered here, analysis of the mean-square error or any approximations of either algorithm apply to both.

2.6.2. Mean-Square Error of the EMAP/MMSE estimate

The mean-square error of the EMAP and MMSE estimates can be obtained by substituting $\mathbf{H}^*$ for $\mathbf{H}$ in (2.21), or

$$MSE_{MMSE}(\mathbf{N}) = E[\mu^T\mu] - 2E[\mu^T\mathbf{H}^{*\,T}\alpha] + E[\alpha^T\mathbf{H}^{*\,T}\mathbf{H}^*\alpha] \tag{2.30}$$

Rearranging gives

$$MSE_{MMSE}(\mathbf{N}) = E[tr\{\mu^T\mu\}] - 2E[tr\{\mathbf{H}^{*\,T}\alpha\mu^T\}] + E[tr\{\mathbf{H}^{*\,T}\alpha\alpha^T\mathbf{H}^*\}] \tag{2.31}$$

Taking the expectation inside the trace yields

$$MSE_{MMSE}(\mathbf{N}) = tr\{\Phi_{\mu\mu}\} - 2tr\{\mathbf{H}^{*\,T}\Phi_{\alpha\mu}\} + tr\{\mathbf{H}^{*\,T}\Phi_{\alpha\alpha}\mathbf{H}^*\} \tag{2.32}$$

Substituting (2.24) for $\mathbf{H}^*$ in the last term of (2.32) results in

$$MSE_{MMSE}(\mathbf{N}) = tr\{\Phi_{\mu\mu}\} - tr\{\mathbf{H}^{*\,T}\Phi_{\alpha\mu}\} \tag{2.33}$$

Rewriting this in terms of Σ, Σ_o, and μ_o, and using (2.28) for $\mathbf{H}^*$ leads to the final expression of the mean-square error as

$$\begin{aligned} MSE_{MMSE}(\mathbf{N}) &= tr\{\Sigma_o + \mu_o\mu_o{}^T\} - tr\{\mu_o\mu_o{}^T + \Sigma_o\mathbf{N}(\Sigma_o\mathbf{N} + \Sigma)^{-1}\Sigma_o\} \\ &= tr\{\Sigma_o - \Sigma_o\mathbf{N}(\Sigma_o\mathbf{N} + \Sigma)^{-1}\Sigma_o\} \\ &= tr\{[\mathbf{I} - \Sigma_o\mathbf{N}(\Sigma_o\mathbf{N} + \Sigma)^{-1}\Sigma_o\} \\ &= tr\{\Sigma(\Sigma_o\mathbf{N} + \Sigma)^{-1}\Sigma_o\} \end{aligned} \tag{2.34}$$

which agrees with the result for $MSE_{EMAP}(\mathbf{N})$ in [24].

Analysis of this expression as a function of data correlation and dogmatism is given in Section 2.8. Briefly comparing the ML MSE expression in (2.34) with (2.12), it can be seen that the EMAP/MMSE expression has additional terms which produce a lower mean-squared error given the same number of samples. This reduction is the benefit of the additional computation required by these algorithms. As will be seen, the computational load can be lightened by LMS approximations to the MMSE gradient implementation.

2.6.3. The MMSE Gradient Algorithm

Conventional LMS adaptive filtering is derived from MMSE parameter estimation by performing the estimation iteratively using a modified gradient search procedure, and by approximating statistical averages used in the computation by their instantaneous values. The MMSE gradient algorithm can be written as

$$\mathbf{H}_{k+1} = \mathbf{H}_k - \frac{\beta}{2}\nabla_{\mathbf{H}_k}[E\{\varepsilon\}] \tag{2.35}$$

where β is the step size or adaptation constant. The index k is incremented after each new sample, from any class, is obtained.[6] From (2.22) and (2.23),

$$\nabla_{\mathbf{H}} E\{\varepsilon\} = -2\Phi_{\alpha\mu} + 2\Phi_{\alpha\alpha}\mathbf{H} \tag{2.36}$$

so (2.35) can be rewritten as

$$\mathbf{H}_{k+1} = [I - \beta\Phi_{\alpha\alpha}]\mathbf{H}_k + \beta\Phi_{\alpha\mu} \tag{2.37}$$

Convergence analysis of this gradient implementation is deferred to the discussion of LMS-C convergence in the next section.

2.7. Least Mean-Square Estimation

In this section we develop the mathematics that enable to represent the MAP and EMAP estimates in the form of adaptive filters. Like the MMSE estimate, the least mean-square mean vector estimate is of the form $\hat{\mu}_k = \mathbf{H}_k^T\alpha_k$, and is derived from the MMSE gradient algorithm by replacing ensemble averages with instantaneous values. The stochastic gradient $\nabla_{\mathbf{H}_k}\|\varepsilon\|^2$ can be obtained from (2.22) by dropping the expected value operation, so

$$\begin{aligned}\nabla_{\mathbf{H}_k}\|\varepsilon\|^2 &= -2\nabla_{\mathbf{H}}[\mu^T\mathbf{H}^T\alpha] + \nabla_{\mathbf{H}}[\alpha^T\mathbf{H}\mathbf{H}^T\alpha] \\ &= -2\alpha\mathbf{d}^T + 2\alpha\alpha^T\mathbf{H}\end{aligned} \tag{2.38}$$

where $\mathbf{d}$ is the so-called desired signal, or the signal from which the error vector $\varepsilon_k = \hat{\mu}_k - \mathbf{d}_k$ is derived. Substitution of this stochastic gradient approximation of $\nabla_{\mathbf{H}}\mathrm{E}\{\|\varepsilon\|^2\}$ into Equation (2.35) yields

$$\mathbf{H}_{k+1} = [\mathbf{I} - \beta\alpha_k\alpha_k{}^T]\mathbf{H}_k + \beta\alpha_k\mathbf{d}_k{}^T \tag{2.39}$$

Note that $\mathbf{H}_k$ is updated after the incorporation of the k^{th} sample into α_k.

---

[6] The parameter k equals $k = \sum_{i=1}^{C} n_i$.

As with conventional LMS adaptive filtering, it is necessary to specify several parameters in the LMS adaptive estimate: the step size parameter β, the desired signal $\mathbf{d}$, and the initial coefficient matrix $\mathbf{H}_0$. Appropriate choices for β and $\mathbf{H}_0$ may be determined through analysis of the LMS expected coefficient error equations. Inspection of empirical learning curves and a bit of insight lead to a suitable choice for $\mathbf{d}$.

2.7.1. Expected LMS Coefficient Error

Define the expected value of the coefficient matrix as $\mathbf{L}_k = E[\mathbf{H}_k]$. Then from (2.39),

$$\begin{aligned}\mathbf{L}_{k+1} &= E\{[\mathbf{I} - \beta\alpha_k\alpha_k{}^T]\mathbf{H}_k\} + E\{\beta\alpha_k\mathbf{d}_k{}^T\} \\ &= E\{[\mathbf{I} - \beta\alpha_k\alpha_k{}^T]\}\mathbf{L}_k + E\{\beta\alpha_k\mathbf{d}_k{}^T\} \\ &= [\mathbf{I} - \beta\Phi_{\alpha\alpha}]\mathbf{L}_k + \beta\Phi_{\alpha\mu} \end{aligned} \tag{2.40}$$

assuming that $\mathbf{H}_k$ is independent of α_k. This assumption is reasonable provided that the step size is small. For convenience, let the value of $\Phi_{\alpha\alpha}$ after the observation of k samples be written as $\mathbf{R}_k$, and let $\mathbf{P} = \Phi_{\alpha\mu}$. Equation (2.40) can now be written as

$$\mathbf{L}_{k+1} = [\mathbf{I} - \beta\mathbf{R}_k]\mathbf{L}_k + \beta\mathbf{P} \tag{2.41}$$

Define the expected coefficient error as $\mathbf{V}_k = \mathbf{L}_k - \mathbf{H}_k^*$, where the time dependence of $\mathbf{H}^*$ is explicitly shown, and recall that $\mathbf{P} = \mathbf{R}_k\mathbf{H}_k^*$. Expressing $\mathbf{P}$ in this fashion and subtracting $\mathbf{H}_{k+1}^* + \mathbf{H}_k^*$ from both sides of (2.41) allows the coefficient error to be written as

$$\begin{aligned}\mathbf{V}_{k+1} - \mathbf{H}_k^* &= [\mathbf{I} - \beta\mathbf{R}_k]\mathbf{V}_k - \mathbf{H}_{k+1}^* \\ \mathbf{V}_{k+1} &= [\mathbf{I} - \beta\mathbf{R}_k]\mathbf{V}_k - (\mathbf{H}_{k+1}^* - \mathbf{H}_k^*)\end{aligned} \tag{2.42}$$

Inspection of (2.42) reveals two aspects which preclude the expected coefficient error from being expressed recursively, as is possible in conventional adaptive filtering. One is the extra term $\mathbf{H}_{k+1}^* - \mathbf{H}_k^*$, and the other is the fact that the correlation matrix $\mathbf{R}_k$ is not constant over time. From (2.28),

$$\mathbf{H}_{k+1}^* - \mathbf{H}_k^* = \begin{pmatrix} -\mu_o{}^T(\mathbf{M}_{k+1}{}^{-1} - \mathbf{M}_k{}^{-1})\Sigma_o \\ (\mathbf{M}_{k+1}{}^{-1} - \mathbf{M}_k{}^{-1})\Sigma_o \end{pmatrix} \tag{2.43}$$

where $\mathbf{M}_k = \Sigma_o + \mathbf{N}_k^{-1}\mathbf{\Sigma}$ as before. Consider how $\mathbf{M}_k$ changes after the k+1$^{\text{th}}$ observation. Assume without loss of generality that this observation is from class i, and that n samples from class i have previously been observed. In this case, only the i^{th} diagonal block in $\mathbf{M}_k$ is updated by the amount $\frac{1}{n(n+1)}\Sigma_i$. When the dogmatism is small and/or the number of observations is large, it is obvious that $\mathbf{M}_{k+1} \approx \mathbf{M}_k$, so the difference in (2.43) is approximately zero and

$$\mathbf{V}_{k+1} = [\mathbf{I} - \beta\mathbf{R}_k]\mathbf{V}_k \tag{2.44}$$

Since $\mathbf{H}_{k+1}^* - \mathbf{H}_k^* = (\mathbf{R}_{k+1}^{-1} - \mathbf{R}_k^{-1})\mathbf{P}$, similar reasoning applies to $\mathbf{R}_k$ which is therefore assumed to be a constant $\mathbf{R}$. Given these assumptions,

$$\mathbf{V}_{k+1} = [\mathbf{I} - \beta\mathbf{R}]\mathbf{V}_k \tag{2.45}$$

Rotating to principal axes using the transformation $\overline{\mathbf{V}}_k = \mathbf{Q}\mathbf{V}_k$, where $\mathbf{Q}$ is the eigenvector matrix of $\mathbf{R}$, (2.45) can be expressed recursively:

$$\overline{\mathbf{V}}_{k+1} = [\mathbf{I} - \beta\Lambda]^k \overline{\mathbf{V}}_0 \tag{2.46}$$

where Λ is the eigenvalue matrix $diag[\lambda_1, \lambda_2, \ldots, \lambda_{CD}]$.

2.7.2. Selection of LMS Parameters

2.7.2.1. Coefficient Matrix Initialization

Appropriate choices for initialization of the coefficient matrix $\mathbf{H}$ can be determined from the expression of the expected coefficient error. Obviously $\mathbf{H}_0$ should be chosen to make the initial error $\overline{\mathbf{V}}_0$ as small as possible. This can be done by setting $\mathbf{H}_0 = \mathbf{H}^*_{\mathbf{I}}$, *i.e.*, the coefficients which would be specified by the optimal estimate after a single sample from each class, or

$$\mathbf{H}_0 = \begin{pmatrix} \mu_o{}^T(\Sigma+\Sigma_o)^{-1}\Sigma \\ (\Sigma+\Sigma_o)^{-1}\Sigma_o \end{pmatrix} \tag{2.47}$$

This choice incorporates more knowledge of the structure of the data into the LMS-C estimate than initializing $\mathbf{H}_0$ to zero or an identity matrix. In empirical simulations (see Section 2.8) this choice often reduces the initial mean square error to levels lower than that of the EMAP estimate. Also, in simulations of data conditions with large dogmatism which violate assumptions in this analysis, the LMS estimate tends to diverge.

2.7.2.2. Step-Size

In the adaptive filtering literature, the step size β is often a constant which is inversely proportional to $\lambda_{max} + \lambda_{min}$, where $\{\lambda_i\}$ are the eigenvalues of the (constant) data correlation matrix. For the multivariate parameter estimation problem considered here, the correlation matrix $\mathbf{R}_k$ is not constant, and this time dependence must be dealt with to ensure stability over time. The coefficient error analysis above showed that $\mathbf{R}_k$ changes slowly over time, but this change may be significant over a large number of iterations. Since $\mathbf{R}_k = \Sigma_o + \mathbf{N}^{-1}\Sigma + \mu_o\mu_o{}^T$, the range of $\mathbf{R}_k$ as $\mathbf{N}$ increases from $\mathbf{I}$ to ∞ is $\mathbf{R}_{\mathbf{I}} = \Sigma_o + \Sigma + \mu_o\mu_o{}^T$ to $R_\infty = \Sigma_o + \mu_o\mu_o{}^T$. Due to the positive definite nature of correlation matrices, all the eigenvalues of $\mathbf{R}_k$ are positive, and (since the trace equals the sum of the eigenvalues) $tr[\mathbf{R}_k] \ge \lambda_{k,max} + \lambda_{k,min}$. With these facts in mind, β was chosen to be inversely proportional to $tr[\mathbf{R}_{\mathbf{I}}] + tr[\mathbf{R}_\infty]$, and specifically

$$\beta = \frac{2}{2tr[\Sigma_o + \mu_o\mu_o{}^T] + tr[\Sigma] + 2} \tag{2.48}$$

Using the traces instead of the sums of the maximum and minimum eigenvalues may make β smaller than necessary, but this choice proved to be sufficient for convergence in empirical tests with both computer-generated and real data.

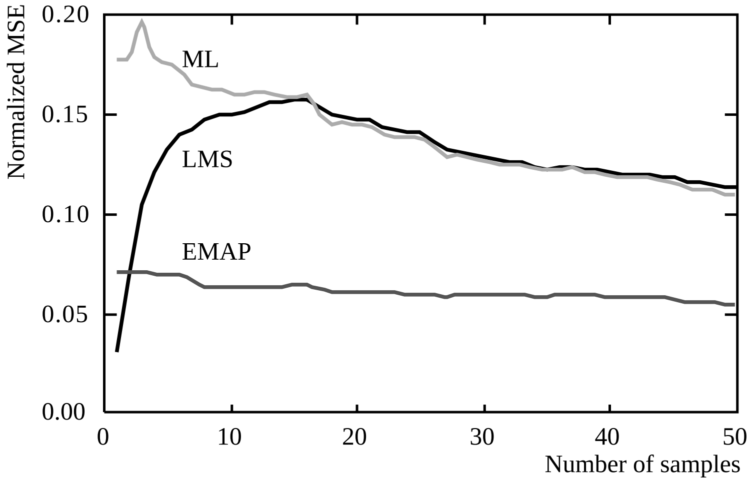

Figure 2-5: Mean-square error vs. number of samples for ML, EMAP, and LMS (with $\mathbf{d} = \mathbf{a}$) estimates for a 2-class, 1-feature case.

2.7.2.3. Desired Signal

Since the output of the LMS estimator is driven toward the desired signal **d**, selection of this signal is crucial for estimation accuracy. The true mean vector μ is the best choice for **d**, but it is obviously unavailable. Some estimate of the true mean, one that does not require excessive computation, must be used in its place. Using $\mathbf{d}_k = \mathbf{a}_k$ allows the LMS estimate to asymptotically converge to the sample mean, as does the MAP estimate, but the error is often larger than what is desirable. This behavior is illustrated in Figure 2-5 which shows the MSE (normalized by the magnitude of the true mean μ) versus number of adaptation samples. It is possible to reduce β to slow the LMS estimate's convergence with the sample mean, but this inhibits the estimate from quickly recovering when the initial error is high.

An alternative choice which does not require much computation is to form **d** as a weighted sum of the *a priori* mean and sample mean. The weights can be derived from the EMAP estimate by replacing **N** with a fixed matrix $\mathbf{N}_c$, or

$$\begin{aligned}\mathbf{d}_k &= \Sigma(\Sigma + \mathbf{N}_c\Sigma_o)^{-1}\mu_o + \Sigma_o(\Sigma + \mathbf{N}_c\Sigma_o)^{-1}\mathbf{N}_c a_k \\ &= \mathbf{c}_1 + \mathbf{C}_2\mathbf{a}_k \\ &= \mathbf{C}^T\alpha_k\end{aligned} \qquad (2.49)$$

where $\mathbf{N}_c = \eta\mathbf{I}$ and η is a constant. This choice for **d** is referred to as the CEMAP (constant-EMAP) estimate. The weights for μ_o and $\mathbf{a}_k$ in (2.49) are those which would be specified in the EMAP procedure if η samples had been obtained from each class. The reasoning behind this choice is as follows. Inspection of (2.19) shows that only **N** and **a** change with each iteration. If **N** is fixed to some value $\mathbf{N}_c$, then the CEMAP mean becomes a constant vector plus a constant matrix times the sample mean which, when $\mathbf{N} = \mathbf{N}_c$, is *identical* to the EMAP estimate. From the coefficient error analysis above, it is seen that the optimal coefficients change slowly when the dogmatism of the data is small, so the CEMAP estimate should be a reasonably close approximation for a fairly wide range of observations around this point. The concept of the CEMAP estimate is illustrated in Figure 2-6 which shows the EMAP and CEMAP MSE as a function of the number of samples. The two learning curves intersect at the point when $\mathbf{N} = \mathbf{N}_c$. The position of this intersection is specified by the parameter η. The LMS estimate using (2.49) as the desired signal is referred to as LMS-C, and has shown in simulation to perform as well or better than the EMAP estimate (see Section 2.8). It does, however, have a finite misadjustment as will be seen in the next section.

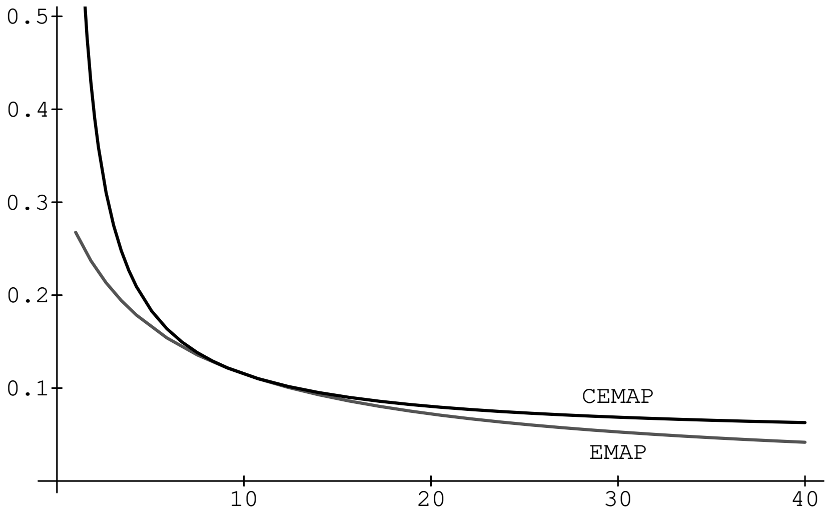

Figure 2-6: EMAP and CEMAP mean-square error vs. number of training samples with η = 10.

2.7.3. Asymptotic Mean-Square Error of the LMS-C Estimate

The gradient-based parameter estimation algorithms (Gradient MMSE and LMS-C) define a method of searching a time-varying performance surface for a time-varying optimal solution. This behavior is illustrated in Figure 2-7, which shows a time-varying performance surface at several instants in time, and the corresponding learning curve which is observed. Note that the minimum of the performance surface migrates across the coefficient (horizontal) axis as well as down the MSE (vertical) axis. The coefficient values at the minimum points are the coefficients which are specified by the EMAP algorithm. The EMAP algorithm theoretically tracks these points exactly, producing a smoothly decaying learning curve. The gradient algorithms, on the other hand, must iteratively search for the minimums as indicated by the zigzag line on the left. Only one point on the performance surface for time $k = k_o$ is visited, and the MSE at that point defines the MSE at time k_o on the learning curve.

In the steady state, the optimal solution converges to the minimum of the steady state surface ($k = \infty$). Due to the stochastic nature of the LMS-C algorithms gradient search, this solution continues

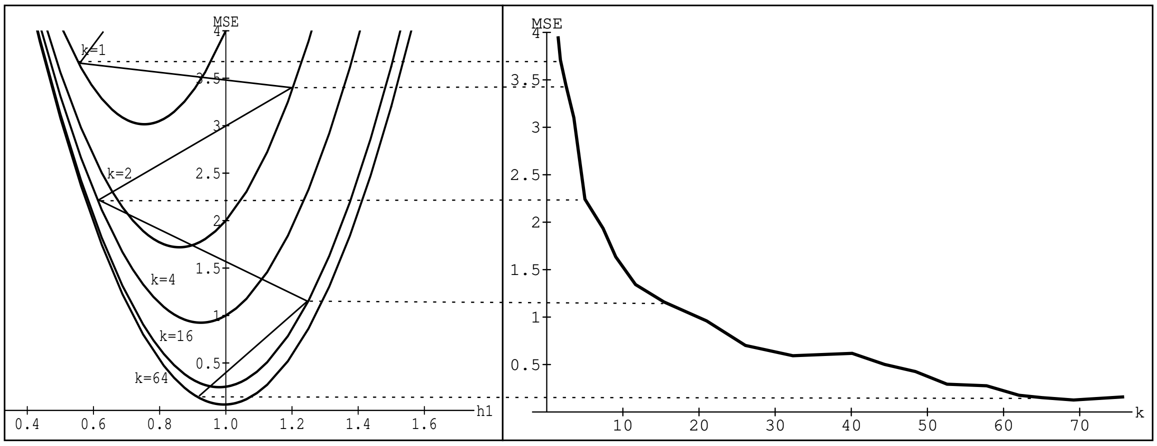

Figure 2-7: Illustration of a time-varying performance surface and the manner in which it is searched (left), and the resulting LMS-C learning curve (right).

to fluctuate about the minimum point, causing an asymptotic misadjustment. Since the LMS-C output is driven toward the desired signal, any misadjustment due to the CEMAP estimate will contribute to the misadjustment of the LMS-C estimate. Therefore the mean-square error of the CEMAP estimate is first derived, and a discussion of the LMS-C misadjustment follows.

2.7.3.1. Mean-Square Error of the CEMAP Estimate

The expected squared-norm of the error vector $E[\|\varepsilon\|^2]$ for the CEMAP estimate can be expressed as

$$\begin{aligned} MSE_{CEMAP}(\mathbf{N}) &= E[\mu\mu^T] - 2E[\mu^T\mathbf{C}^T\alpha] + E[\alpha^T\mathbf{C}\mathbf{C}^T\alpha] \\ &= tr[\Phi_{\mu\mu}] - 2tr[\mathbf{C}^T\Phi_{\alpha\mu}] + tr[\mathbf{C}^T\Phi_{\alpha\alpha}\mathbf{C}] \end{aligned} \tag{2.50}$$

Expanding $\mathbf{C}$ and the other statistics, plus noting that $\Phi_{\mathbf{aa}} = \Phi_{\mathbf{a}\mu} + \mathbf{N}^{-1}\Sigma$,

$$\begin{aligned} MSE_{CEMAP}(\mathbf{N}) = tr[&\Phi_{\mathbf{a}\mu} - 2\mathbf{c}_1\mu_o^T - 2\mathbf{C}_2\Phi_{\mathbf{a}\mu} + \mathbf{c}_1\mathbf{c}_1^T + 2\mathbf{c}_1\mu_o^T\mathbf{C}_2^T \\ &+ \mathbf{C}_2\Phi_{\mathbf{a}\mu}\mathbf{C}_2^T + \mathbf{C}_2\mathbf{N}^{-1}\Sigma\mathbf{C}_2^T] \end{aligned} \tag{2.51}$$

Only the last term in (2.51) is dependent on $\mathbf{N}$; the remainder is a fixed misadjustment. The transient term $T_{CEMAP}(\mathbf{N})$ can be rewritten using the definition of $\mathbf{C}_2$ from (2.49)

$$T_{CEMAP}(\mathbf{N}) = tr[\Sigma_o(\Sigma + \mathbf{N}_c\Sigma_o)^{-1}\mathbf{N}_c\mathbf{N}^{-1}\Sigma\mathbf{N}_c(\Sigma + \Sigma_o\mathbf{N}_c)^{-1}\Sigma_o] \tag{2.52}$$

Although straightforward, expressing the misadjustment M_{CEMAP} in terms of Σ, Σ_o, and $\mathbf{N}_c$ is tedious. The derivation is aided by the definitions $\mathbf{K} = (\Sigma + \Sigma_o\mathbf{N}_c)^{-1} = \mathbf{K}^T$ and $\mathbf{T} = \mu_o\mu_o^T$, and observations that $\mathbf{I} - \mathbf{C}_2 = \Sigma\mathbf{K}$, $\mathbf{I} - \mathbf{C}_2^T = \mathbf{K}^T\Sigma$, and $\mathbf{c}_1 = \Sigma\mathbf{K}\mu_o$ Rewriting all but the transient (last) term in (2.51) using these relations,

$$\begin{aligned} M_{CEMAP} &= tr[\Phi_{\mathbf{a}\mu}(\mathbf{I} - \mathbf{C}_2^T) - 2\mathbf{c}_1\mu_o^T(\mathbf{I} - \mathbf{C}_2^T) - \mathbf{C}_2\Phi_{\mathbf{a}\mu}(\mathbf{I} - \mathbf{C}_2^T) + \mathbf{c}_1\mathbf{c}_1^T] \\ &= tr[\{\Sigma\mathbf{K}\Phi_{\mathbf{a}\mu} - 2\Sigma\mathbf{K}\mu_o\mu_o^T\}\mathbf{K}^T\Sigma + \Sigma\mathbf{K}\mu_o\mu_o^T\mathbf{K}^T\Sigma] \\ &= tr[\Sigma\mathbf{K}(\Sigma_o + \mathbf{T})\mathbf{K}^T\Sigma - 2\Sigma\mathbf{K}\mathbf{T}\mathbf{K}^T\Sigma + \Sigma\mathbf{K}\mathbf{T}\mathbf{K}^T\Sigma] \\ &= tr[\Sigma\mathbf{K}\Sigma_o\mathbf{K}^T\Sigma] \end{aligned} \tag{2.53}$$

which results in

$$M_{CEMAP} = tr[\Sigma(\Sigma + \Sigma_o \mathbf{N}_c)^{-1}\Sigma_o(\Sigma + \mathbf{N}_c\Sigma_o)^{-1}\Sigma] \tag{2.54}$$

It is possible to show that $MSE_{CEMAP}(\mathbf{N}) = MSE_{EMAP}(\mathbf{N})$ when $\mathbf{N} = \mathbf{N}_c$, and also that the difference in MSE between these two estimates is small when the dogmatism of the data is small. The CEMAP MSE and misadjustment as a function of data properties and the parameter η are explored further in Section 2.8.

Returning to the question of the LMS-C estimate's asymptotic MSE, assume that the estimate has reached steady-state, at which point the coefficient matrix $\mathbf{H}_k$ is randomly fluctuating about $\mathbf{H}^*$. Equation (2.39) can also be expressed as

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \beta\alpha_k\varepsilon_k{}^T \tag{2.55}$$

where $\varepsilon_k = \hat{\mu}_k - \mathbf{d}_k$ is the error vector as before, so $\hat{\mu}_k$ can be written as

$$\hat{\mu}_k = \mathbf{H}_k^T\alpha + \beta\varepsilon\alpha^T\alpha \tag{2.56}$$

Since $\mathbf{H}_k$ is near the optimum, the noise in the LMS-C estimate is the second term in (2.56), $\beta\varepsilon\alpha^T\alpha = \beta\alpha^T\alpha\varepsilon$. The expected squared-norm of this term is the misadjustment of the LMS-C estimate, or

$$\begin{aligned} M_{LMS\text{-}C} &= E[\|\beta\alpha^T\alpha\varepsilon\|^2] \\ &= \beta^2 E[\alpha^T\alpha\varepsilon\varepsilon\alpha^T\alpha] \end{aligned} \tag{2.57}$$

When the weight matrix is near the optimum, the error signal ε is approximately uncorrelated with α.[7] Approximating the fourth-order statistic involving α with second-order statistics, (2.57) becomes

$$\begin{aligned} M_{LMS\text{-}C} &= \beta^2 E^2[\alpha^T\alpha]E[\varepsilon\varepsilon^T] \\ &= \beta^2 tr^2[\Phi_{\alpha\alpha}]E[\varepsilon\varepsilon^T] \\ &= \gamma^2 E[\varepsilon\varepsilon^T] \end{aligned} \tag{2.58}$$

At the optimum, $\hat{\mu} \approx \mu$ and therefore from the definition of ε and (2.50), $E[\varepsilon\varepsilon^T] \approx M_{CEMAP}$, making the LMS-C misadjustment a scaled version of that of the CEMAP estimate. Substituting the value for β from (2.48), the scale factor can be written as

$$\gamma = \frac{2(tr[\Sigma_o + \mu_o\mu_o{}^T] + 1)}{2tr[\Sigma_o + \mu_o\mu_o{}^T] + 2 + tr[\Sigma]} \leq 1 \tag{2.59}$$

where the $tr[\mathbf{N}^{-1}\Sigma]$ term in $tr[\Phi_{\alpha\alpha}]$ was ignored since it should be small at convergence. When the dogmatism is small, the scale factor γ is likely to be near 1.0 making the misadjustment of the LMS-C estimate approximately equal to that of the CEMAP. This result is not surprising since in the small-dogmatism case, as the LMS-C coefficient error decays to zero and the estimate is driven toward the desired signal, any misadjustment is largely due to the misadjustment of the desired signal. As will be seen in the analysis in the next section, the parameter η controls the tradeoff between this misadjustment and rate of convergence.

---

[7]This is easily proven by taking the expected value of $\varepsilon\alpha^T$ and evaluating it at $\mathbf{H} = \mathbf{H}^*$.

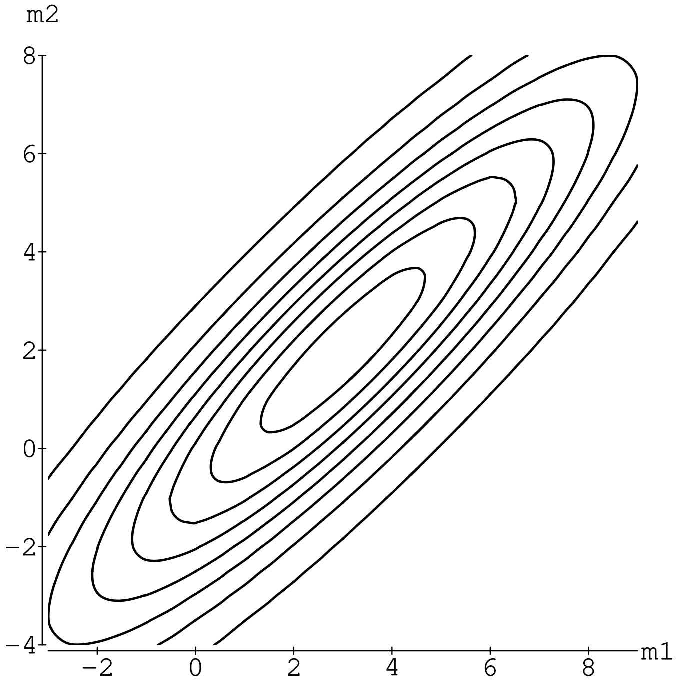

Figure 2-8: Contour plot of the *a priori* distribution of the mean vector $\mu = [m_1\ m_2]^T$ for $\mu_o = [3\ 2]^T$.

2.8. Analysis of the Estimates of Mean Vectors

The success of the estimation algorithms presented in the previous sections is highly dependent on a number of properties of the data. In this section, these procedures are analyzed theoretically and empirically with respect to dogmatism, correlation, and class-conditional *a priori* probability of observation. The theoretical analysis is based on the expressions for mean-square error versus the number of observations as derived in the previous sections. Empirical analyses are based on learning curves generated by Monte Carlo simulation methods. Examples in both sets of analyses are based on a 2-class, 1-feature model defined by the following statistics

$$\Sigma_o = \begin{pmatrix} \sigma^2/\delta^2 & \rho\sigma^2/\delta^2 \\ \rho\sigma^2/\delta^2 & \sigma^2/\delta^2 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix}, \quad \mathbf{N} = \begin{pmatrix} n & 0 \\ 0 & \theta n \end{pmatrix} \tag{2.60}$$

The parameters δ, ρ, and θ represent the dogmatism, correlation, and skew in the prior probabilities, respectively, and σ^2 is the variance of the speakers' class-conditional data distributions. This form for the data model allows each parameter to be varied independently of the others. The *a priori* mean distribution $N(\mu_o,\Sigma_o)$ is plotted in Figure 2-8 for $\mu_o = [3\ \ 2]^T$, $\rho = 0.9$, and $\sigma^2 = 1$. The ellipsoidal shape and positive slope of this distribution's major axis indicates the degree of correlation of the two class means m_1 and m_2. The within-speaker class-conditional probability density functions are centered around realizations of these random mean values, as illustrated in Figure 2-9 for $\mu = [3\ \ 2]^T$.

2.8.1. Theoretical Analysis

In this section the dependence of estimation error on the various statistical parameters of the data is discussed, as well as misadjustment of the CEMAP and LMS-C algorithms. Properties of the estimation error can be observed directly from the expressions describing the expected mean-square error versus the number of observations, which are given by Equations (2.12) and (2.34). while the CEMAP and LMS-C misadjustments are specified by Equations (2.54) and (2.57). Based on the derivation in Section 2.7, the LMS-C mean-square error is assumed to be equivalent to that of the EMAP/MMSE algorithm plus the asymptotic misadjustment. For convenience, the MSE expressions are repeated here:

$$\begin{aligned} &MSE_{ML}(\mathbf{N}) = tr[\mathbf{N}^{-1}\Sigma] \\ &MSE_{EMAP}(\mathbf{N}) = tr[\Sigma(\Sigma_o\mathbf{N}+\Sigma)^{-1}\Sigma_o] \\ &MSE_{LMS\text{-}C}(\mathbf{N}) = tr[\Sigma(\Sigma_o\mathbf{N}+\Sigma)^{-1}\Sigma_o] + \gamma^2 tr[\Sigma(\Sigma+\Sigma_o\mathbf{N}_c)^{-1}\Sigma_o(\Sigma+\mathbf{N}_c\Sigma_o)^{-1}\Sigma] \end{aligned} \tag{2.61}$$

where γ is specified by (2.59). Note that the statistic μ_o is only present in the expression for the LMS-C MSE and misadjustment (as part of γ). The contribution of μ_o to the MSE curves is negligible and will be ignored. Also, the variance σ^2 from (2.60) above acts only as a scale factor which can be assumed to be unity.

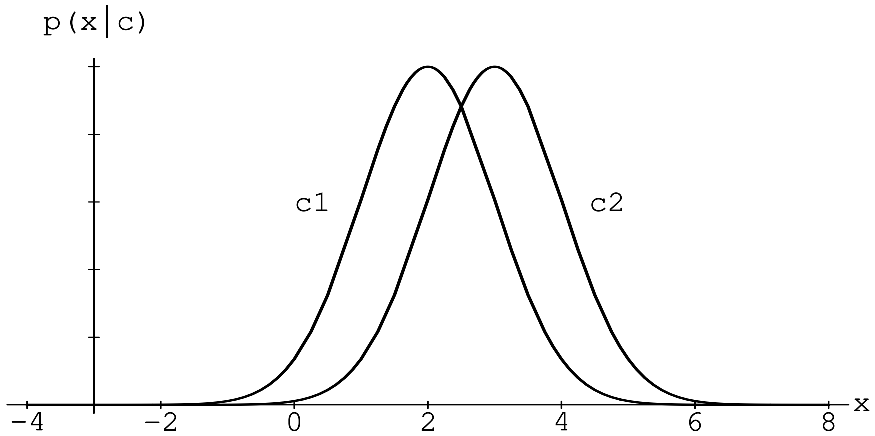

Figure 2-9: Class-conditional probability density functions for the example of $\mu = \mu_o$ and $\delta = 1$.

2.8.1.1. Effect of Dogmatism, Correlation, and Skew

The discussion in Section 2.3 leads to the expectation that when the dogmatism (δ) is small, any estimator based on observations should produce a reasonable estimate, while for large δ the best estimate is the constant μ_o. Plots of Equation (2.61) in Figure 2-10 for four values of δ verify this prediction. For very small δ, the MSE equations in (2.61) are all approximately equal to $tr[\mathbf{N}^{-1}\Sigma]$, which is the MSE of the ML estimate. As δ increases, the variation of μ around μ_o decreases, making μ_o a better estimate of μ. This has the effect of flattening the MSE curves for the EMAP and LMS-C estimates which initially give greater weight to μ_o. When δ is very large, the relative variation of μ

around μ_o is small, and the constant μ_o is the best estimate for μ. Inspection of (2.61) shows that when $\Sigma >> \Sigma_o$, the LMS-C and EMAP MSE expressions reduce to approximately $tr[\Sigma_o]$, which is the error incurred when the observations are ignored and μ_o is used as the estimate of μ. Figure 2-10(d) reflects this behavior of the EMAP and LMS-C estimates. The ML estimate does not depend on dogmatism, so it's curves are the same in all four graphs. Similar behavior is observed in empirical tests, as will be discussed below.

A consequence of the dogmatism-dependence of these estimates for speech-related problems is that as δ increases, the *a priori* or speaker-independent mean becomes a better estimate of the speaker-dependent mean, making adaptation less necessary. In fact, adaptation becomes more difficult as the dogmatism rises, since the convergence rate (the slope of the learning curve) decreases with increasing δ. As stated earlier, the most favorable conditions for adaptation are small-dogmatism cases ($\delta \le 1$). Of those cases, more benefit from EMAP or LMS-C adaptation will be observed for δ near unity.

One of the advantages of the EMAP and LMS-C algorithms is their ability to update all classes after observing a sample from any class. This reduces the MSE more quickly than methods like ML which do not model the between-class correlation. Figure 2-11 shows that the EMAP and LMS-C estimates converge more rapidly when there is greater correlation among features and/or decision classes. The figure also shows that the absolute MSE is always lower than the ML estimate. The reason for this behavior is that the ML learning curve is proportional to k^{-1}, while the other learning curves are proportional to $[k+c]^{-1}$, where c is a function of the correlation ρ. As ρ increases, c also increases, leading to faster convergence.

The utility of modeling between-class correlation is highest when the skew in the *a priori* class probabilities is large. Compare the EMAP and LMS-C learning curves to the ML curves in Figure 2-12. The figure shows the mean-square error for the three algorithms for $\theta = 0.053, 0.176, 0.43, \text{and } 1.0$, which corresponds to cases in which 5%, 15%, 30%, and 50%, respectively, of the observations are from Class 2 in the model of (2.60). Obviously, the gain is largest when θ is small. Since the ML algorithm estimates each element of each class' mean vector independently, it can only update a particular class after obtaining a sample from it, and the MSE for all other classes remains constant. This occurs because the ML MSE expression for the estimate of the mean vector for the j^{th} is equal to σ_j^2/k, while for the other algorithms there are additional terms which represent the contribution of the other classes in reducing the j^{th} class' error.

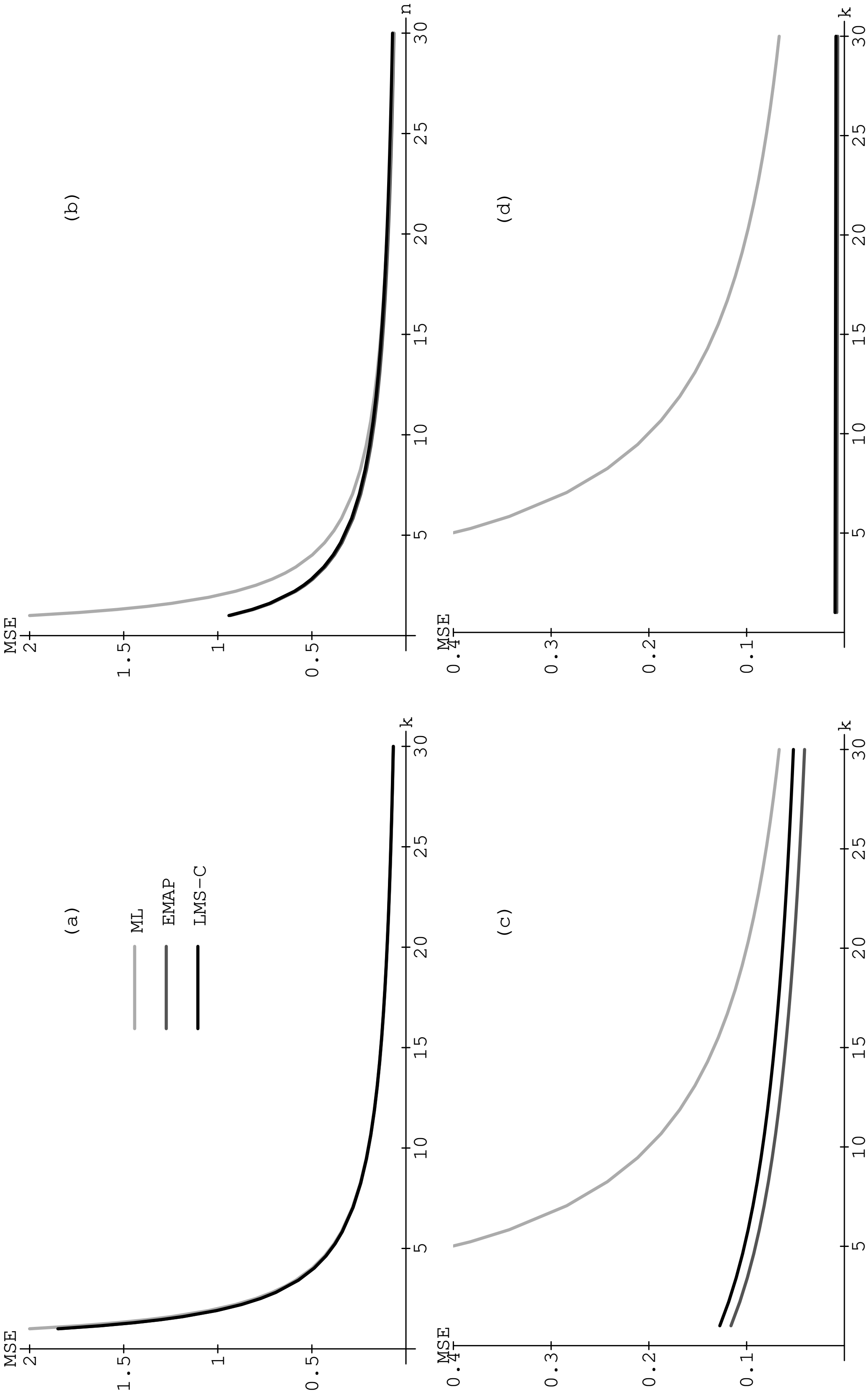

Figure 2-10: Mean-square error as specified by Equation (2.61) vs. number of samples for dogmatism values of δ = (a) 0.25, (b) 1, (c) 4, and (d) 16.

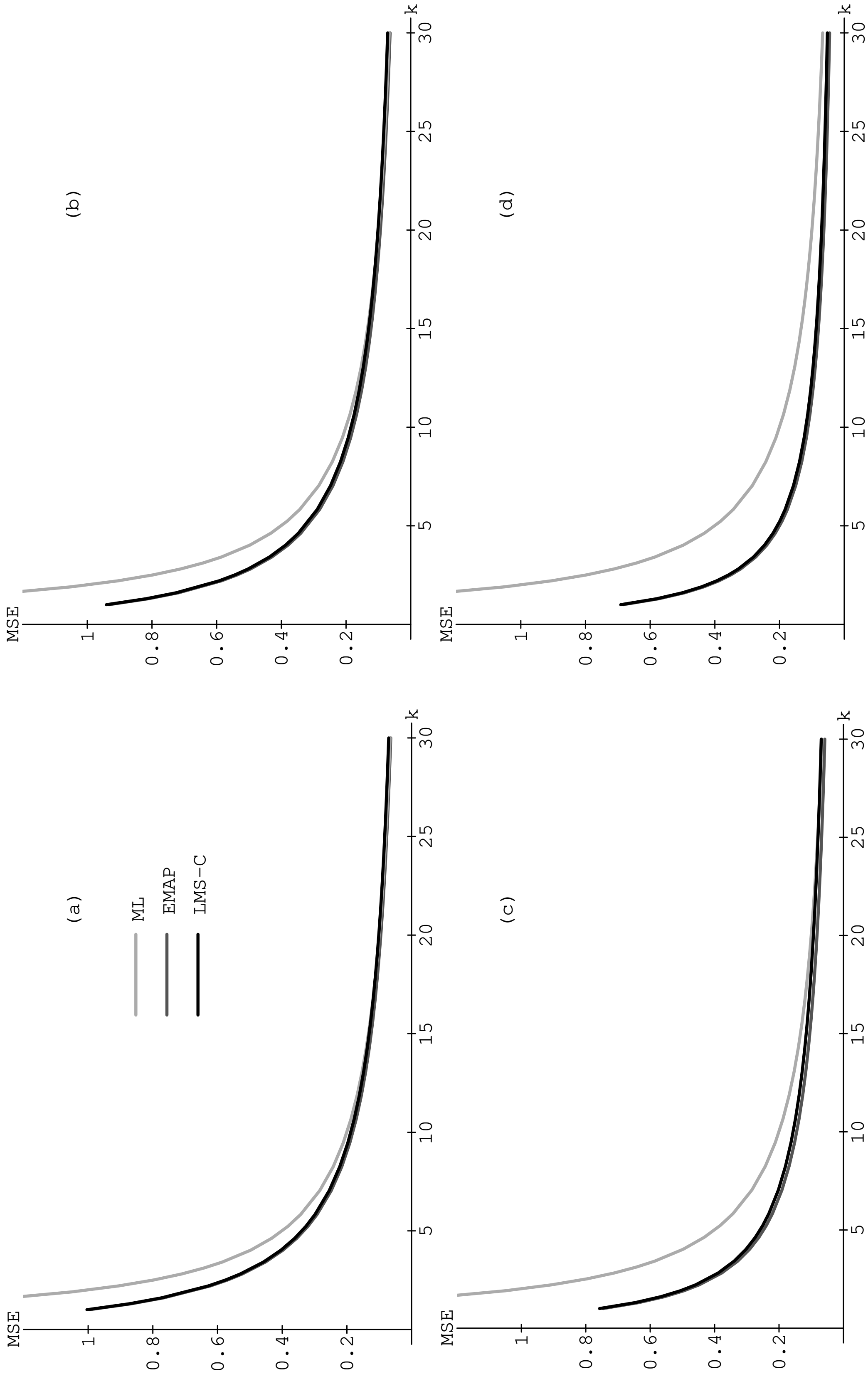

Figure 2-11: Mean-square error as specified by Equation (2.61) vs. number of samples for correlation values of ρ = (a) 0.1, (b) 0.5, (c) 0.9, and (d) 0.98.

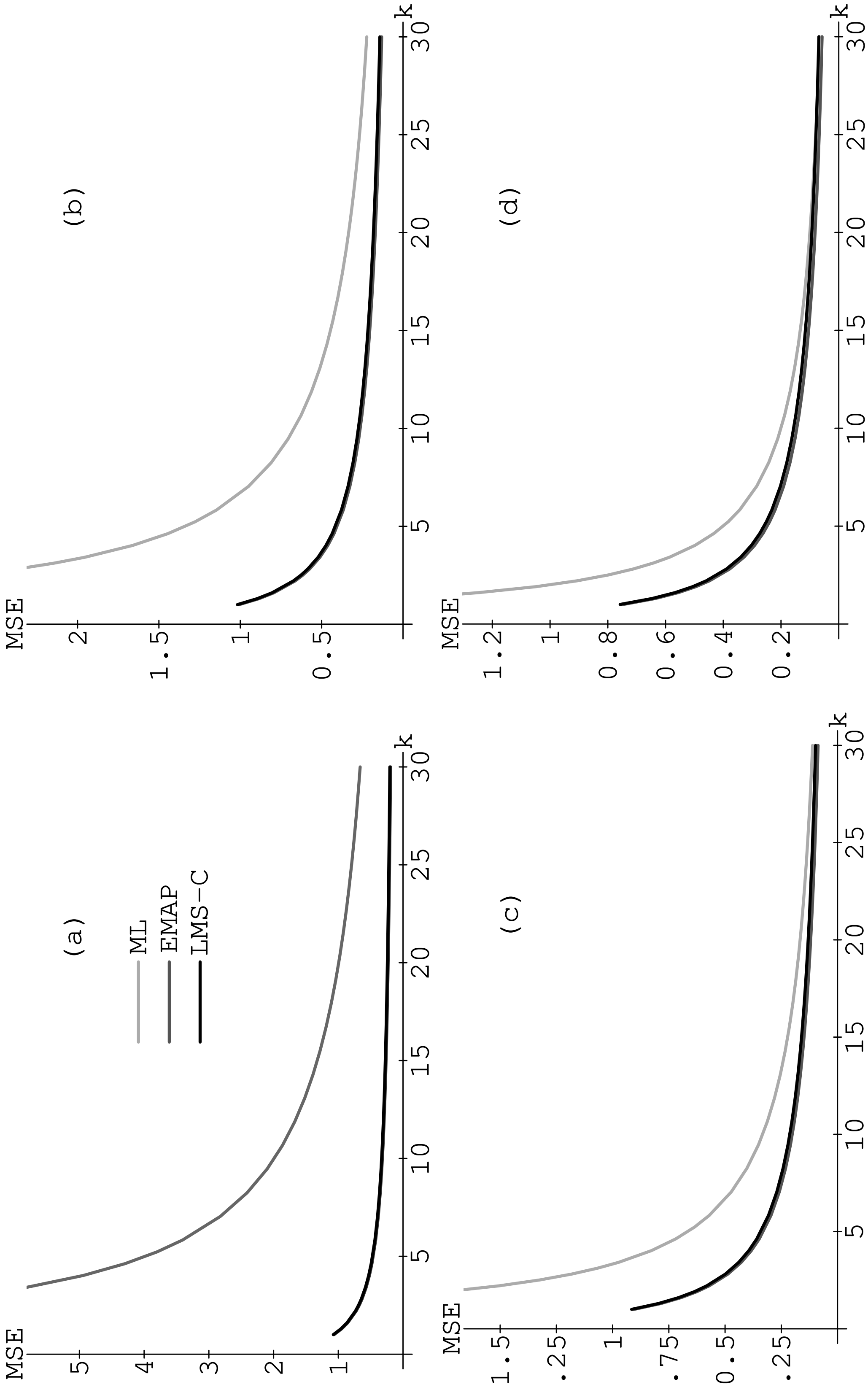

Figure 2-12: Mean-square error as specified by Equation (2.61) vs. number of samples with skew as a parameter. Values of θ are (a) 0.053, (b) 0.176, (c) 0.43, and (d) 1.0.

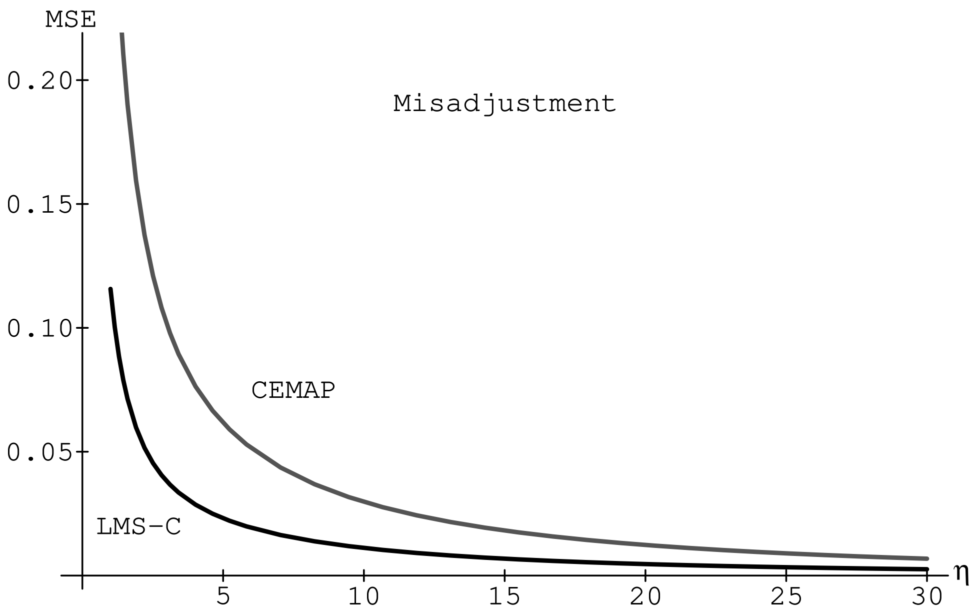

Figure 2-13: Theoretical LMS-C and CEMAP misadjustment for a 2-class, 1-feature case as a function of η, with δ=1, θ=1, and ρ=0.9.

2.8.1.2. CEMAP and LMS-C Misadjustment

As mentioned in Section 2.7, the parameter η controls the misadjustment in both the CEMAP and LMS-C algorithms. The constant matrix $\mathbf{N}_c$, which was defined as $\eta\mathbf{I}$, controls the relative weighting between the observations and μ_o in the CEMAP estimate $\mathbf{d}$. As more weight is given to the constant μ_o (by lowering η or $\mathbf{N}_c$), the misadjustment increases since the CEMAP (and therefore the LMS-C) estimates are driven toward this constant value and cannot converge to the observations. Figure 2-13 shows the relationship between η and misadjustment. The LMS-C misadjustment is smaller because the LMS-C algorithm weights the CEMAP estimate by γ^2 (see (2.59)). The scale factor γ has some dependence on μ_o, but this dependence decreases with decreasing δ and is negligible.

While it is best from a misadjustment standpoint to make η large, doing so greatly increases the contribution of μ_o in $\mathbf{d}$. Since giving more weight to the sample mean increases the initial error (see Figure 2-5), it is necessary to strike a balance between these two effects. Empirical tests or trial and error are necessary to determine optimal values of η.

2.8.2. Empirical Analysis

The empirical learning curves presented in this section are plots of the averaged ratio of the squared magnitudes of the error vector to the mean vector

$$MSE(k) = \frac{1}{T}\sum_{i=1}^{T}\frac{\|\hat{\mu}_i(k) - \mu_i\|^2}{\|\mu_i\|^2} \tag{2.62}$$

where T is the number of trials over which the results are averaged, and μ_i is the i^{th}-trial mean vector. This form of normalization, combined with the fact that random mean vectors are generated for each trial, has the effect of varying the vertical scale on the learning curves between experiments. Therefore, empirical analysis compares the relative reduction in MSE between algorithms and not the absolute error.

Each trial consists of a simulation of the observations that would be produced from several decision classes for a single hypothetical speaker. The "speaker's" mean vectors for a given trial are first generated by performing a Cholesky decomposition of Σ_o, generating a CD-dimensional vector of $N(0,1)$ i.i.d. random deviates, and multiplying this vector by the decomposed upper triangular matrix. Adding μ_o to this product produced a vector with the desired probability density function of the form $N(\mu_o,\Sigma_o)$. An equivalent procedure was used to produce a sequence of samples with probability density function $N(\mu,\Sigma)$. Σ and Σ_o were varied to obtain the desired data conditions, using the model in (2.60). The sample mean **a** was initialized to μ_o, *i.e.*, μ_o is treated as an initial data point. This prevented the learning curves from exhibiting an initial peak as components of **a** were replaced by observations. As the trial progressed, data were presented randomly from each class, according to the specified prior distribution, and the normalized MSE as a function of sample number was tabulated. The example learning curves in this section were typically averaged over five trials.

2.8.2.1. Dogmatism, Correlation, and Skew

Figure 2-14 depicts empirical learning curves for 4 values of dogmatism. Compare this figure with the predictions in Figure 2-10 above. The empirical curves behave as predicted: in the small-dogmatism case, all estimates are essentially the same. In the large-dogmatism case the EMAP and LMS-C estimates are fixed at μ_o and the ML MSE is large.[8] The most significant part of this figure is the behavior of the LMS-C algorithm for mid-range values of δ, where the LMS-C error is often half that of the EMAP algorithm. To obtain (theoretically) zero misadjustment, the optimal estimate must choose weights according to the constraints of the algorithm, which are to minimize MSE. These weights may not be optimal with respect to obtaining low initial error, but other weights (like those from LMS-C) violate the minimum-MSE constraints by adding asymptotic misadjustment. In other words, the EMAP coefficients are deterministic with respect to the matrix **N**: given the statistics of the data and the matrix **N**, the EMAP coefficients will not vary with the data. The LMS-C coefficients, however, are largely determined from the data. Although this gradient search of the performance surface introduces a misadjustment, it also produces coefficients which enable the LMS-C estimate to converge more quickly.

---

[8]Although in the model of (2.60) the ML MSE is independent of dogmatism, the varying test conditions produce different Σ_o matrices and hence different trial means, even with identical random seed initialization.

The LMS-C estimate's low initial error is due to a combination of factors which include the use of the CEMAP estimate as the desired signal, initialization of $\mathbf{H}_0$ as in (2.47), and the proper choice of η and β. This behavior is valuable since initial convergence is important for speaker adaptation. As expected, all three estimates converge as the number of samples increases, so the benefits of the LMS-C and EMAP algorithms decrease with time. In the long term, asymptotic misadjustment can be avoided by computing the optimal estimate once after the observation of many data samples since, unlike LMS-C, the Bayesian optimal estimate is not a recursive estimate.

Note the variation in absolute mean-square error (the vertical axes in Figure 2-14) as a function of dogmatism. When δ is very small, the observations convey a good deal of information about the location of the mean, making the MSE small. As δ increases, the data's information content (as far as μ is concerned) decreases, so the MSE is higher. At the same time, the amount of information about the mean conveyed by μ_o is increasing, so when the dogmatism becomes large the MSE is again small. In other words, at the extreme values of dogmatism the MSE is low because either the observations or the *a priori* mean are very reliable estimates of μ.

Figure 2-15 shows the learning curves with correlation as the parameter for $\rho = 0.98$, 0.9, 0.5, and 0.1, similar to Figure 2-11. Again, the estimators behave as expected with the gains of LMS-C and EMAP over ML methods increasing with increasing correlation. And again in these cases the LMS-C error is significantly smaller than either the EMAP or ML estimates. Note that the ML MSE is independent of ρ. The variation in the ML learning curves in Figure 2-15 occurs because different trial means are produced as ρ (and consequently Σ_o) is varied.

The empirical tests illustrated in Figure 2-16 investigated the degree to which modeling between-class correlation actually reduces MSE, and also whether the LMS-C algorithm is as effective as EMAP under these conditions. The test conditions are similar to the previous tests except that ρ was increased to 0.9 to better demonstrate the algorithms' properties. In these experiments, the frequency of samples from Class 2 was varied from 5% to 50% (the equal priors case), which is equivalent to varying θ from 0.053 to 1.0 as in Figure 2-12. The abrupt drops in the ML and LMS-C MSE in the $\theta = 0.053$ and 0.176 cases are due to the observation of samples from the less-likely class. It is obvious from these curves that the LMS-C method does retain some portion of the EMAP algorithm's ability to use correlation information to reduce MSE, but the shape of the $\theta = 0.053$ curve may indicate that it is more sensitive to skew in prior probabilities than the EMAP.

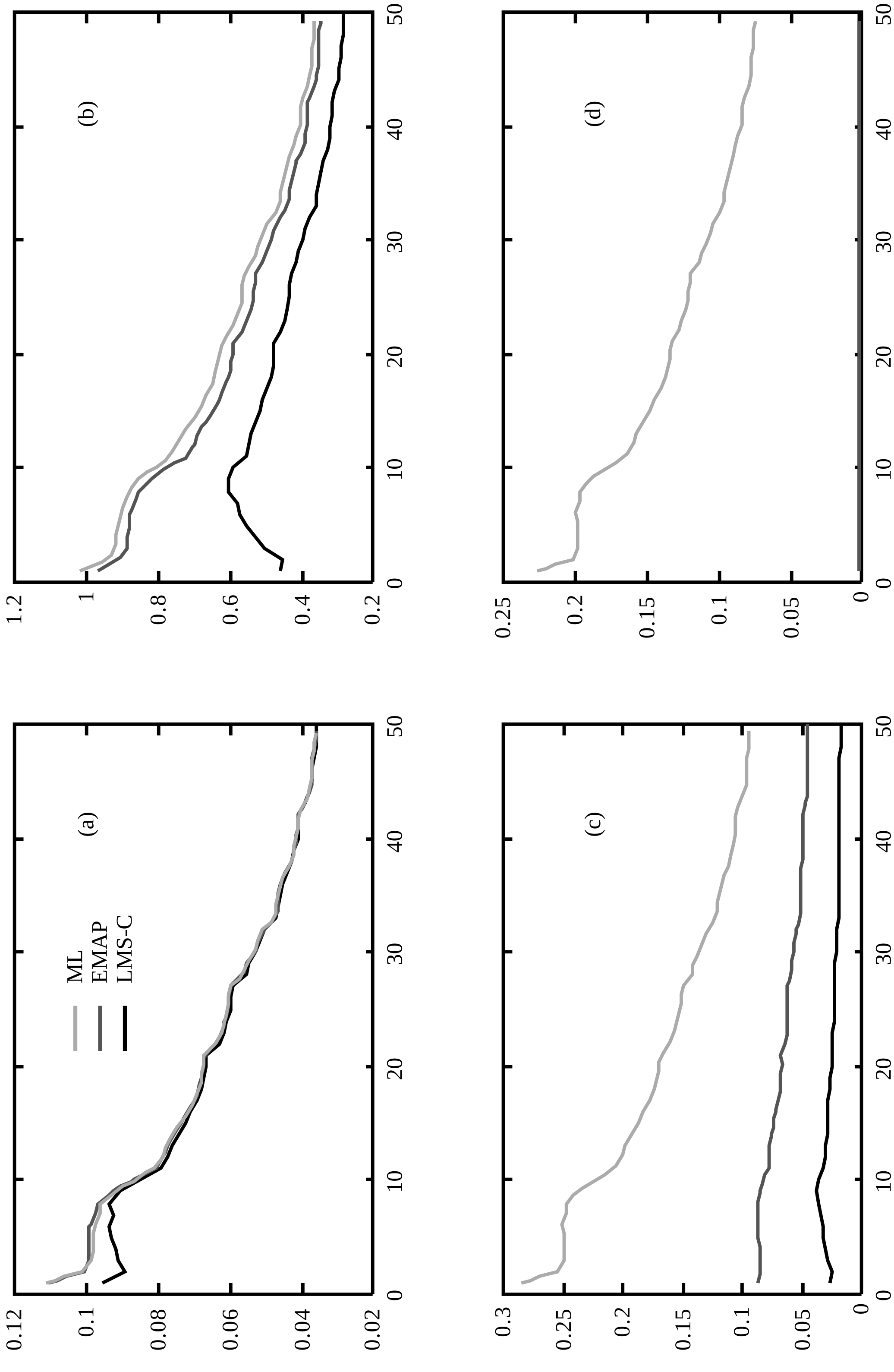

Figure 2-14: Empirical mean-square error vs. number of samples for dogmatism δ = (a) 0.25, (b) 1.0, (c) 4.0, and (d) 16.0; $\rho = 0.5$, $\theta = 1.0$.

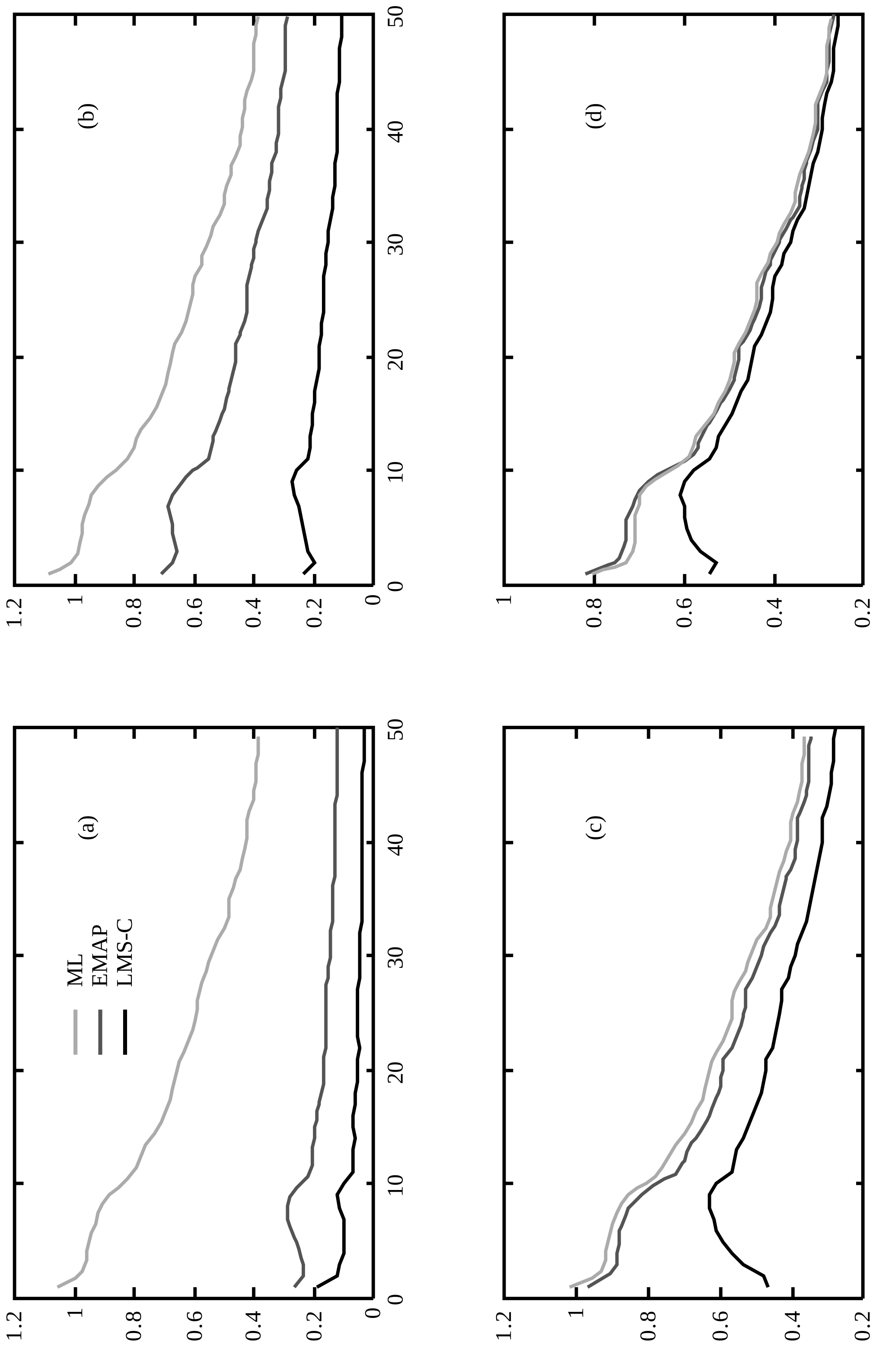

Figure 2-15: Empirical mean-square error vs. number of samples for correlation ρ = (a) 0.98, (b) 0.9, (c) 0.5, and (d) 0.1; $\delta = 1.0$, $\theta = 1.0$.

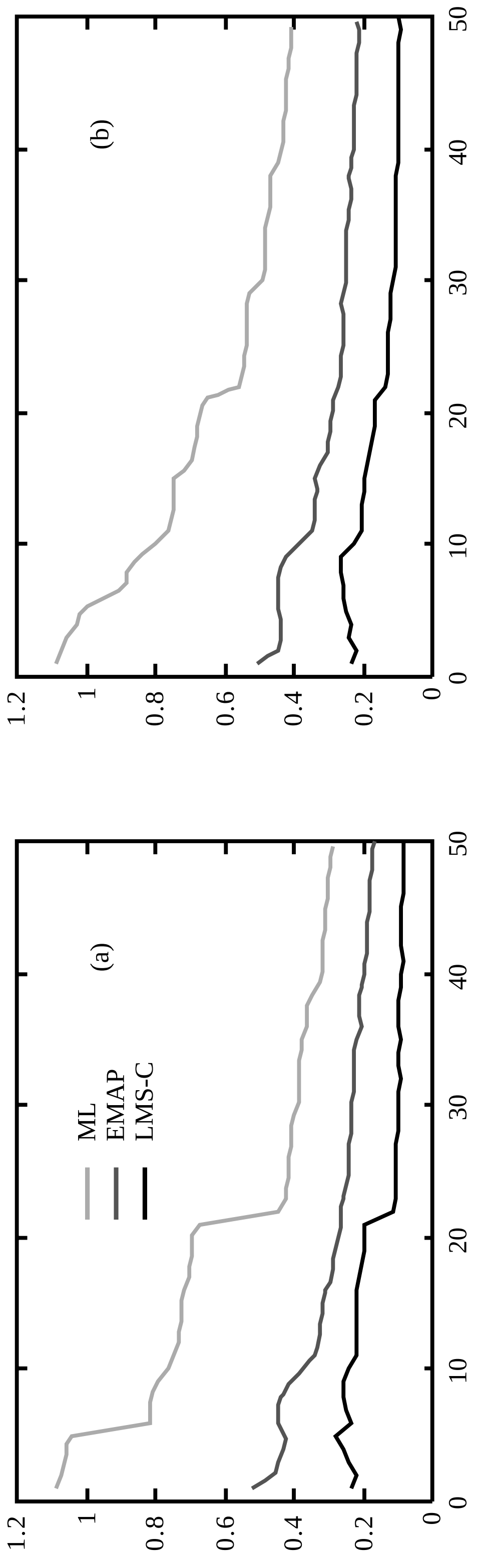

Figure 2-16: Empirical learning curves vs. number of samples with (a) 5%, (b) 15%, (c) 30%, and (d) 50% of the samples from Class 1, and $\delta = 1.0$, $\rho = 0.9$.

2.8.2.2. Initialization of the LMS-C Coefficient Matrix

The convergence analysis in Section 2.7 showed that the mean-squared coefficient error can be expressed as a recursion relation, so judicious choice of the initial coefficient matrix $\mathbf{H}_0$ can have a considerable impact on initial MSE. The curves in Figure 2-17 are typical results from an empirical study of different forms of LMS-C coefficient initialization. The forms of initialization under investigation were random, all zero, $[\mu_o|\mathbf{I}]^T$, and full initialization as specified by (2.47). In the first three cases the initial error is large with respect to the ML and EMAP error, although the estimate does converge after about 10 samples in these cases. As more information about the data is included in the initial coefficients, convergence is even faster - note the dip in 2-17(c) between 2 and 10 samples. When initialized using (2.47), a significant portion of the *a priori* knowledge of the data structure is incorporated and the LMS-C typically exhibits excellent convergence characteristics, as seen in Figure 2-17(d) and in the other tests discussed in this section.

In the derivation of the expected LMS coefficient error (Section 2.7) it was assumed that $\mathbf{H}_{k+1} \approx \mathbf{H}_k$. The sequence of coefficient matrices $\{\mathbf{H}_k\}$ can be inspected to verify this assumption. Figure 2-18 shows the norm of the difference between successive coefficient matrices, normalized by the magnitude of $\mathbf{H}_k$, averaged over 10 trials with $\delta = 4$, $\theta = 1$, and $\rho = 0.5$. The initial change is approximately 1.5% of the magnitude of the matrix, and after convergence the change is around 0.1%, so the assumptions about the rate of change in $\mathbf{H}_k$ appear to be valid. Figure 2-18 also validates the assumption that $\mathbf{H}_k$ is independent of the input data after convergence which was made in the derivation of the LMS-C misadjustment, as there are no large changes in the curve after 5 observations.

2.8.2.3. Convergence-point parameter η

Theoretical analysis showed that as η increases, the LMS-C and CEMAP misadjustment decreases. This parameter describes the point at which the theoretical CEMAP learning curve intersects the EMAP learning curve. It also controls the relative weighting of the prior and sample means in the CEMAP estimate. Therefore, if η is too large, the sample mean will dominate the LMS-C's desired signal, driving the LMS-C estimate too quickly toward the ML estimate, raising the estimation error. If η is too small, the prior mean μ_o dominates and the LMS-C estimation error is essentially a constant equal to $E[\|\mu - \mu_o\|^2]$. These effects are illustrated in Figure 2-19 for η equal to 1, 11, 21, and 31, and $\delta = 1$, $\rho = 0.5$. Clearly when $\eta = 1$ the output of the LMS-C estimator is quickly driven towards μ_o and the error becomes fixed at that level. When $\eta = 31$, the LMS-C converges to the CEMAP/EMAP mean after about 20 samples. In the other cases, the LMS-C estimate is balanced between the two extremes and is able to produce a more accurate mean value.

Another factor affecting the speed of convergence of the LMS-C estimate is the step size

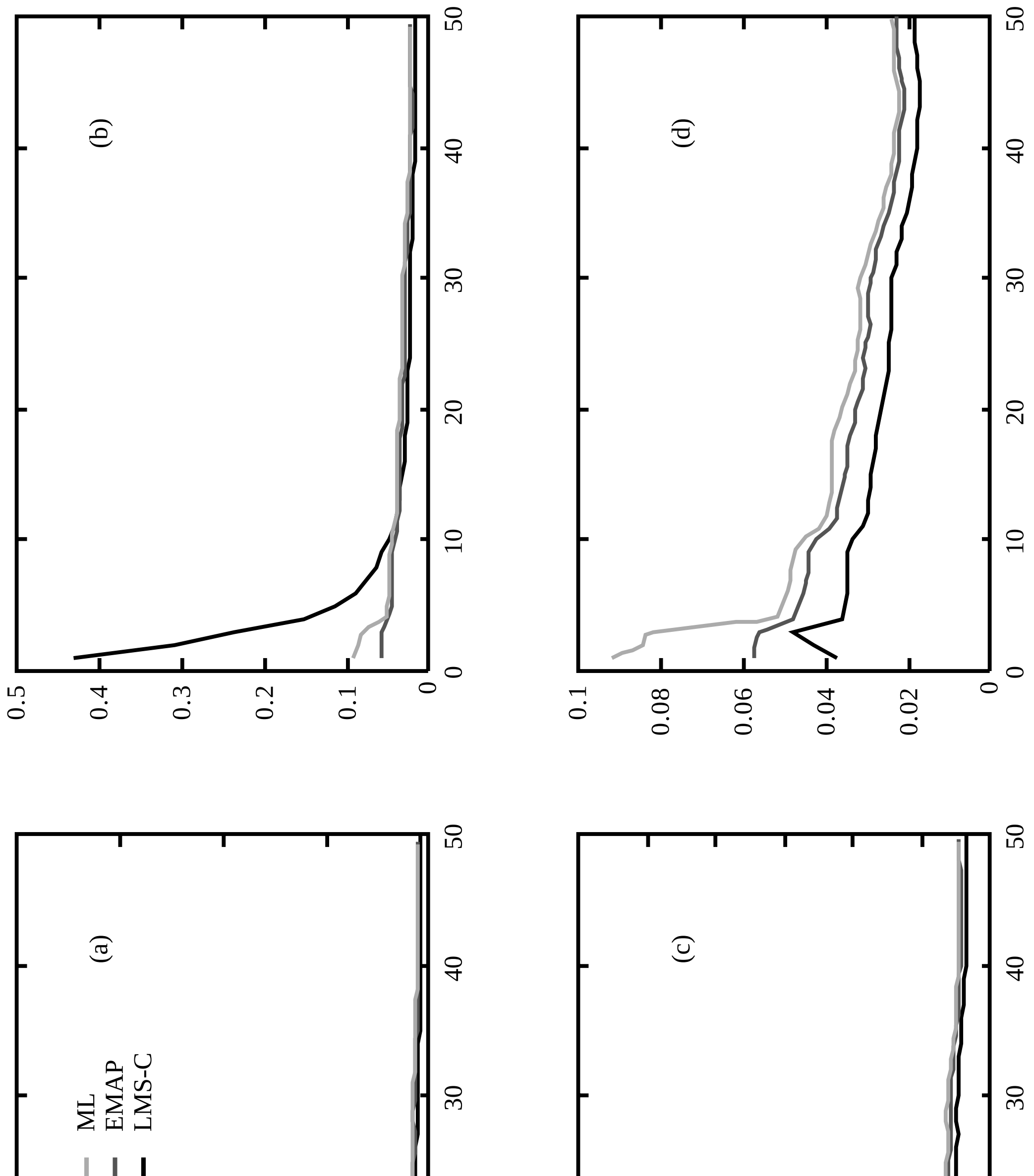

Figure 2-17: Empirical learning curves showing the effect of $\mathbf{H}_0$ initialization on LMS-C convergence: (a) random, (b) zero, (c) $[\mu_o|\mathbf{I}]$, and (d) Eqn. (2.47).

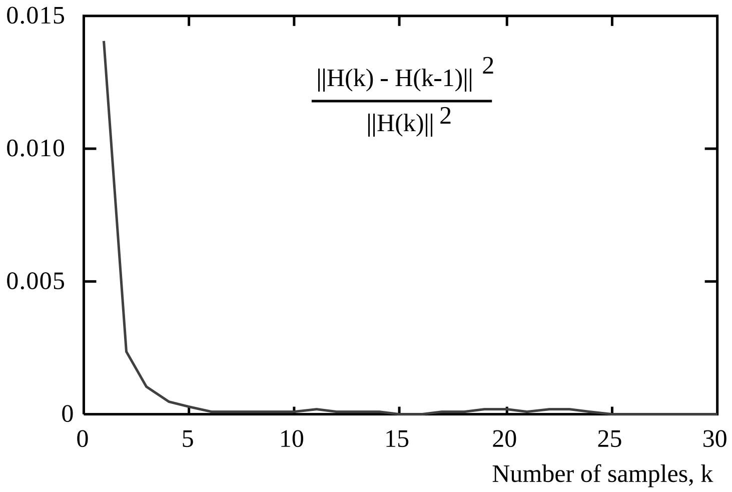

Figure 2-18: Normalized difference of successive LMS-C coefficient matrices, indicating the slow rate of change in the coefficients.

parameter β. The optimal value of this parameter is given by (2.48), but as in conventional adaptive filtering it is sometimes necessary to reduce this value to maintain stability. Figure 2-20 shows the effect on convergence speed of scaling the optimal step size. To better illustrate these effects by avoiding the effects of averaging, these graphs are the results from a single trial. The effect of scaling is best seen in the first three plots in the range of 1 to 10 samples. As the scale factor increases, the LMS estimate tracks the ML mean more closely, up to the point at which the algorithm becomes unstable (fourth plot). Scale factors between 0.5 and 1.0 were typically used in the tests reported here.

2.8.2.4. Non-Gaussian Distributions

The last set of empirical experiments studied the behavior of the three estimation algorithms when the assumption of Gaussian probability density functions was relaxed. Both triangular and uniform within-class distributions were investigated under various data conditions, and normal distributions were generated for comparison. The distribution parameters were chosen so that the amounts of dogmatism were accurate representations of the ratio of variances. As one might expect, estimation performance with the triangular distributions was similar to the normal distribution case, as shown in Figures 2-21 and 2-22, although the EMAP MSE appears to be closer to the ML MSE in the triangular data case. Degradation in performance was much larger when the uniform probability densities were used (Figure 2-23). After a few observations the ML and LMS-C learning curves converge to a constant MSE. The EMAP curve gradually approaches the ML curve since its coefficients gradually give more weight to the ML estimate.

From these tests, it appears that in instances in which the data distribution provides some infor-

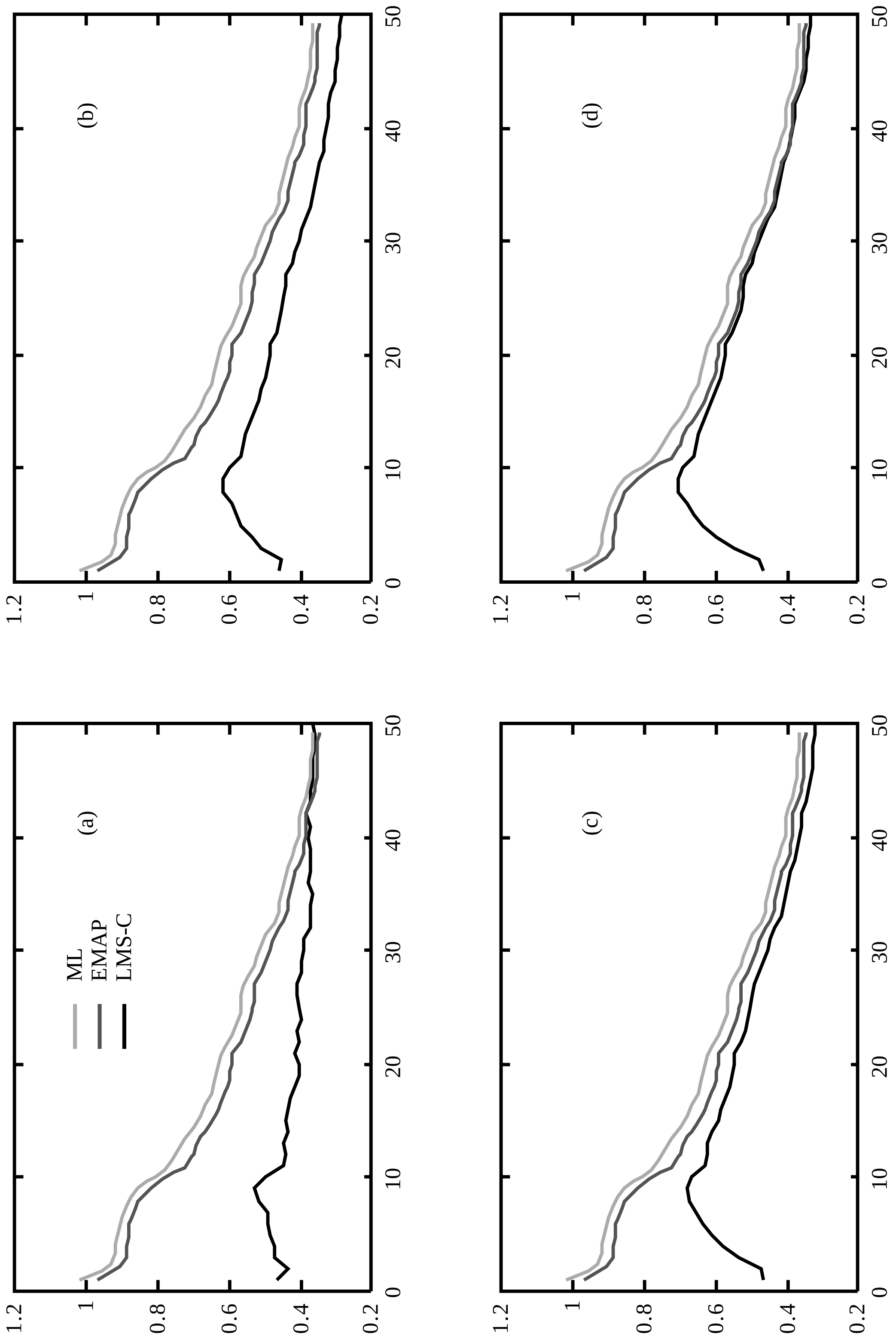

Figure 2-19: Empirical learning curves showing the effect of η on LMS-C convergence: η = (a) 1, (b) 11, (c) 21, and (d) 31; $\delta = 1.0$, $\rho = 0.5$.

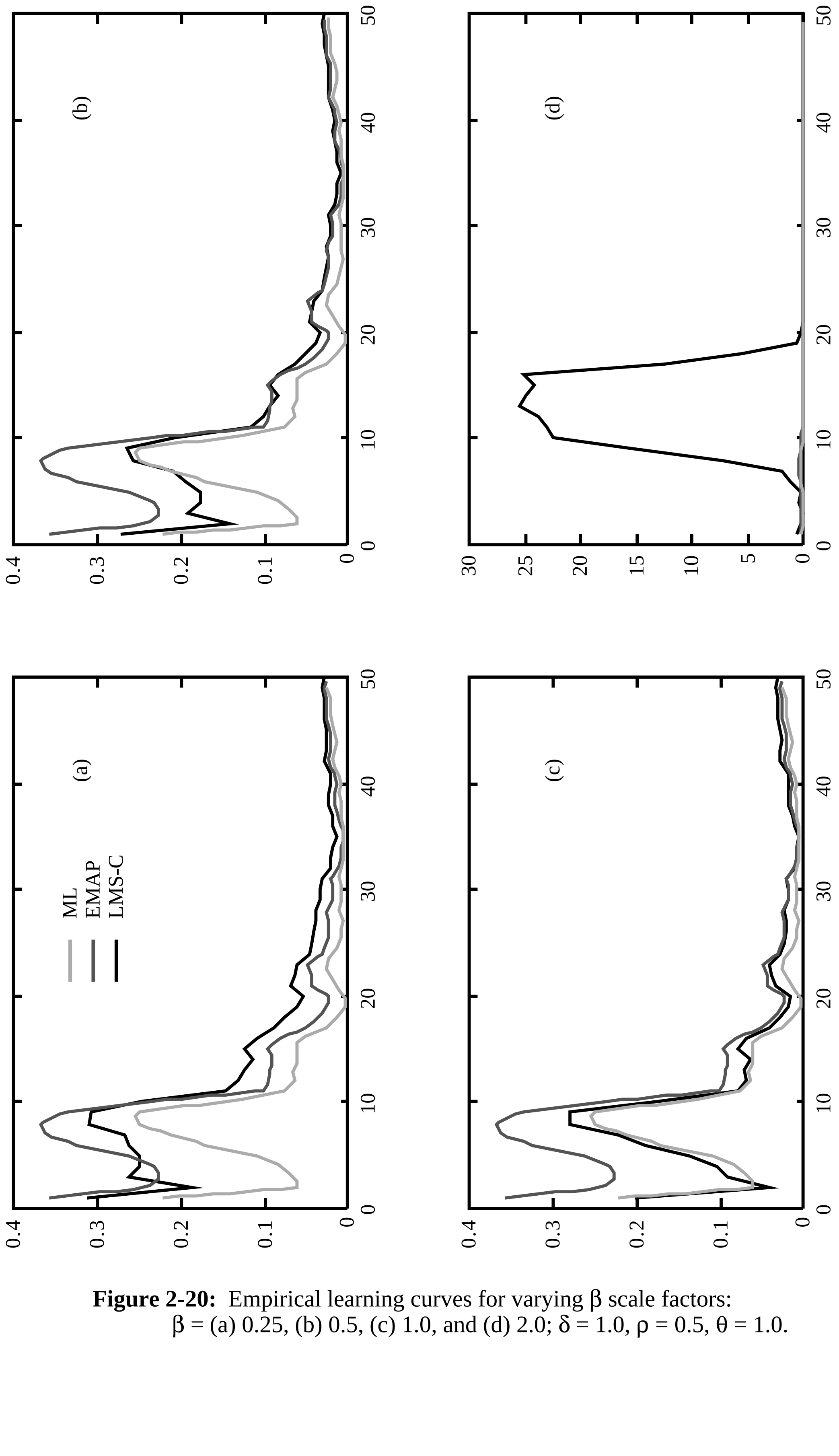

Figure 2-20: Empirical learning curves for varying β scale factors:
β = (a) 0.25, (b) 0.5, (c) 1.0, and (d) 2.0; δ = 1.0, ρ = 0.5, θ = 1.0.

mation on the location of the mean, *i.e.*, values near the mean are more likely, the LMS estimate is able to converge to the mean. Since the LMS-C algorithm effectively computes a time-averaged correlation matrix, it more accurately reflects the ensemble statistics when the data are not normally distributed. The EMAP algorithm does not have this capability due to its use of fixed ensemble statistics and a deterministic method of updating its coefficients. In cases where the data does not tend toward the mean (as in the uniform case), the estimates may not converge or show any reduction in MSE.

2.9. Computational Complexity

The computational requirements of each of the algorithms presented in this chapter are summarized in Table 2-1. The quantities in this table represent the number of multiplications and additions which are required to compute the updated estimate after obtaining a single sample, *i.e.* for one iteration. These data are illustrated in Figure 2-24 which plots the sum of multiplications and additions versus the product CD (the dimension of the adaptation statistics). As the Figure and Table indicate, the computational requirements of the LMS-C algorithm are about one-third those of the EMAP estimate. It should be pointed out that these calculations are based on non-optimized implementations of the EMAP and LMS-C algorithms; it may be possible to reduce the computation in each by the use of optimized matrix manipulation and inversion procedures.

2.10. Summary

In this chapter, the extended maximum *a posteriori* mean vector estimation procedure was recast as an adaptive filtering problem. The EMAP procedure was shown to be equivalent to an MMSE adaptive filtering with time-varying statistics and performance surface. An implementation of the MMSE algorithm was then developed using the stochastic gradient algorithm. The resulting LMS formulation for multivariate mean vector estimation was shown to be stable under conditions of low dogmatism.

Proper initialization of the LMS parameters is necessary to obtain good adaptation performance from this algorithm. Analysis of the expected error of the LMS coefficients under conditions of small dogmatism resulted in a recursive expression for that error. Minimization of the initial coefficient error was therefore necessary to keep the coefficient error small. Since the optimal coefficients are specified by the EMAP expression, the EMAP coefficients were used to initialize the LMS coefficient matrix **H**. Initializing the LMS coefficients in this manner incorporates *a priori* knowledge of the structure of the data into the LMS estimate, which was empirically shown to produce a lower initial mean-square error than other methods of initialization.

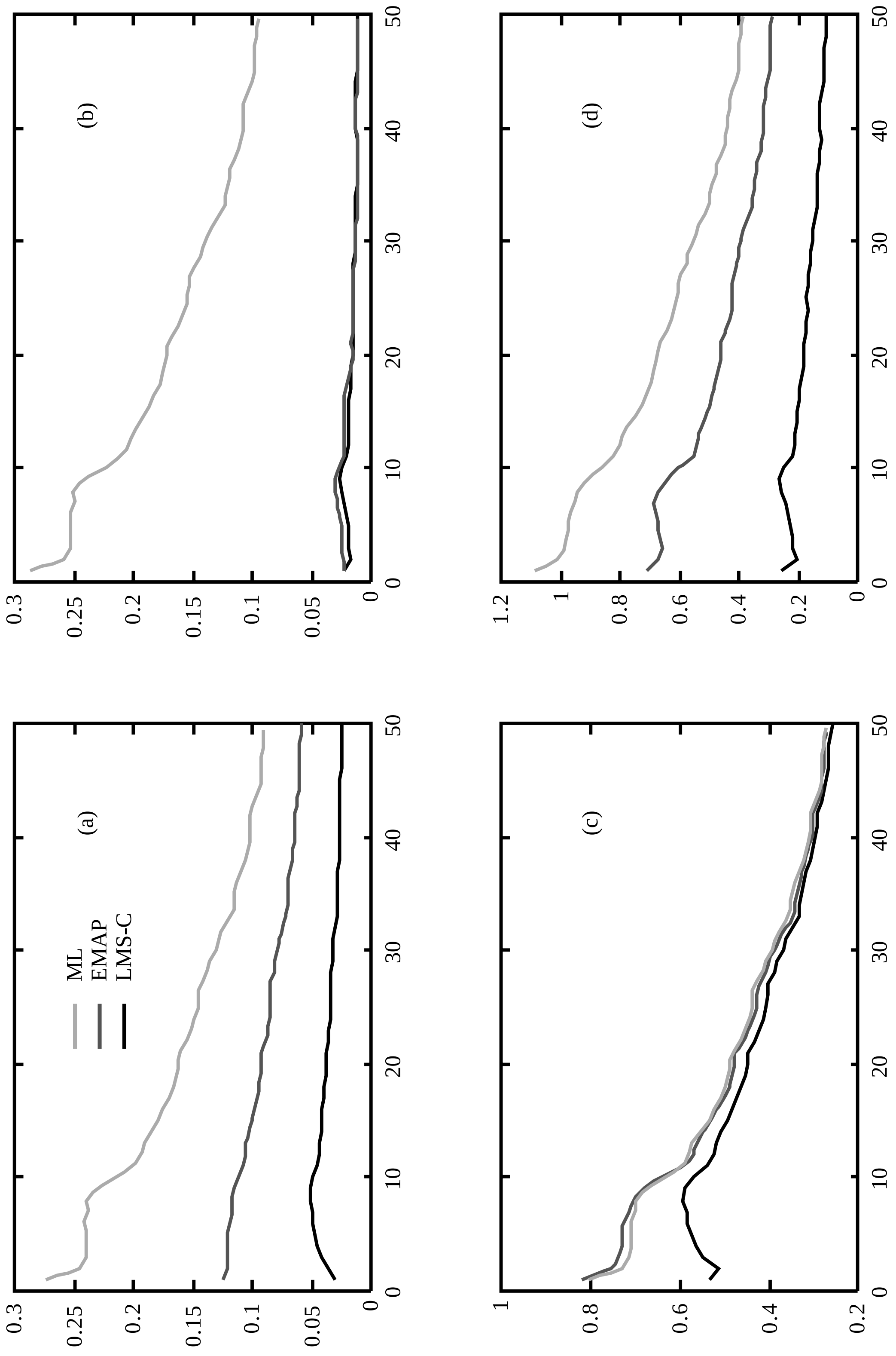

Figure 2-21: Empirical learning curves for Gaussian within-class distributions, for (δ,ρ) = (a) (4,0.1), (b) (4,0.9), (c) (1,0.1), and (d) (1,0.9); θ = 1.0.

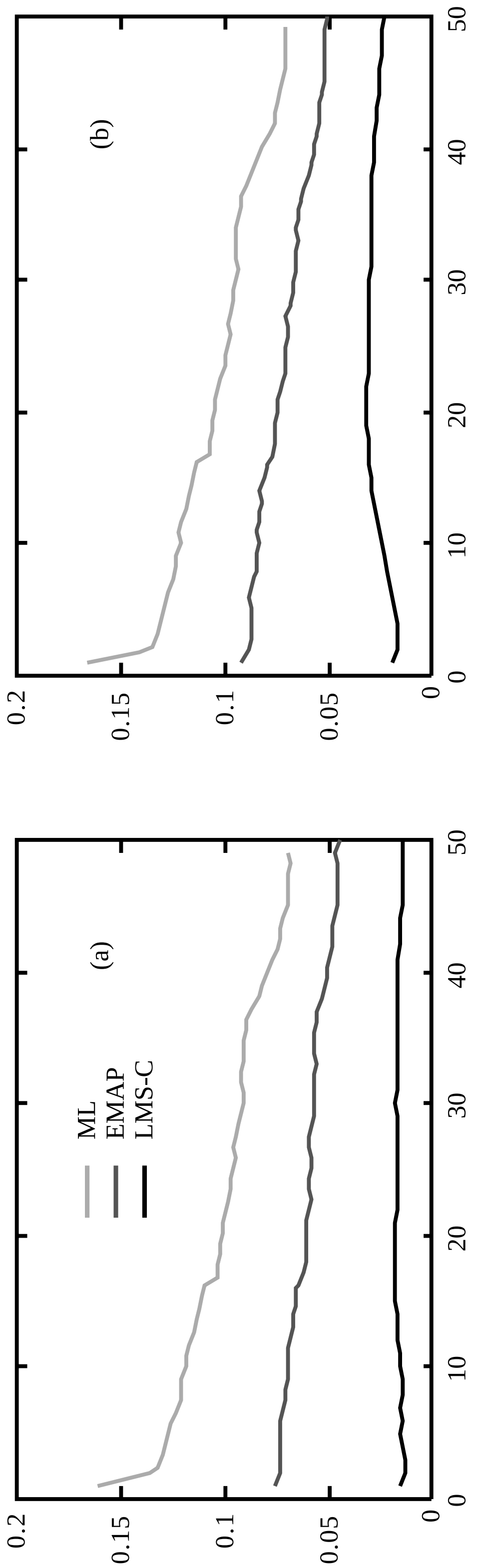

Figure 2-22: Empirical learning curves for triangular within-class distributions, for (δ,ρ) = (a) (4,0.1), (b) (4,0.9), (c) (1,0.1), and (d) (1,0.9); θ = 1.0.

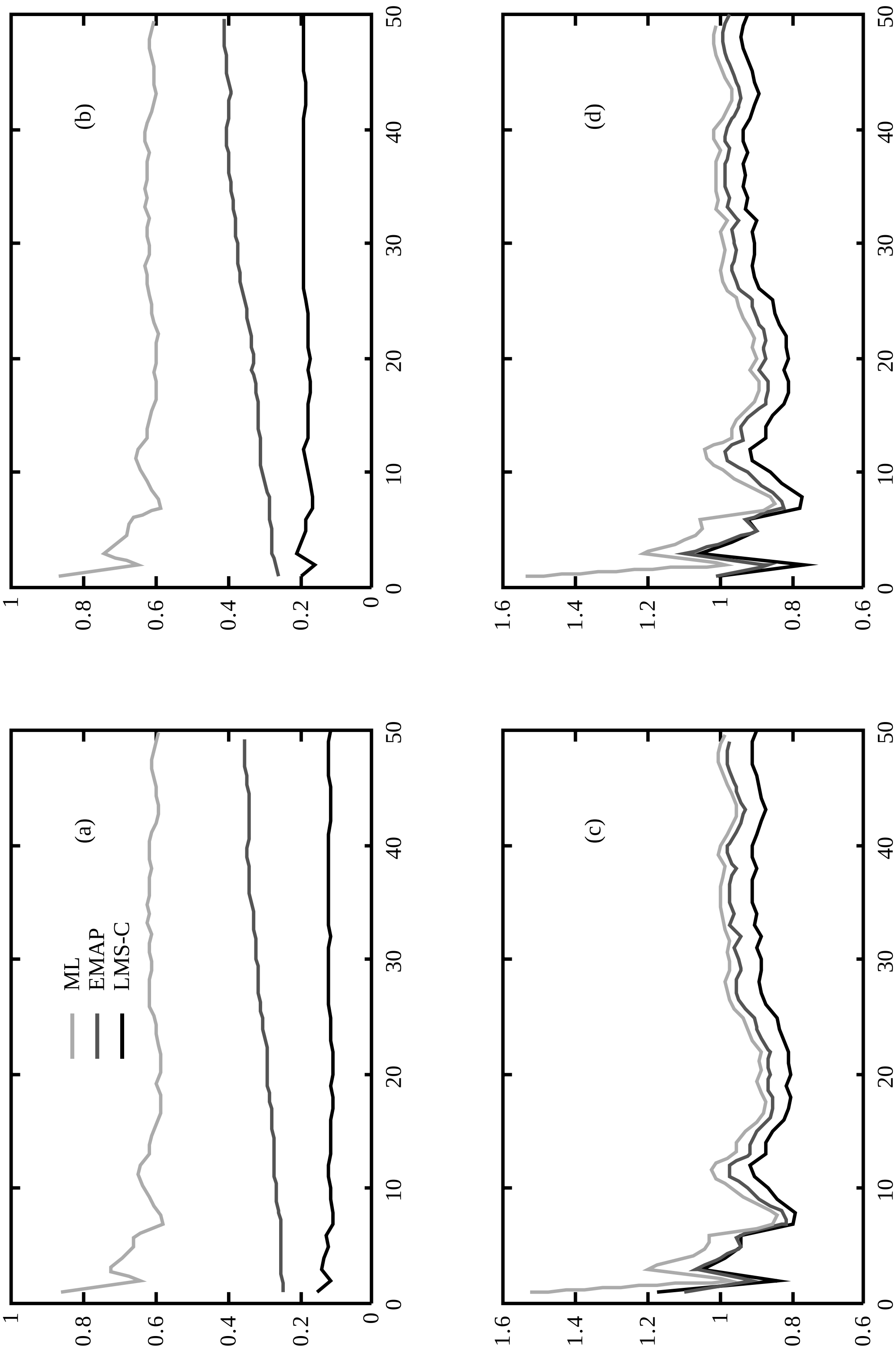

Figure 2-23: Empirical learning curves for uniform within-class distributions, for (δ,ρ) = (a) (4,0.1), (b) (4,0.9), (c) (1,0.1), and (d) (1,0.9); $\theta = 1.0$.

Algorithm	Additions	Multiplications
ML	D	2D
EMAP/MMSE	$3(CD)^3+3(CD)^2+CD$	$3(CD)^3+4(CD)^2+(D+1)CD$
CEMAP	$(CD)^2+D$	$(CD)^2+2D$
LMS-C	$(CD)^3+6(CD)^2+3CD+D-1$	$(CD)^3+6(CD)^2+6CD+2(D+1)$

Table 2-1: Computational requirements of estimation algorithms as a function of the numbers of classes (C) and dimensions (D).

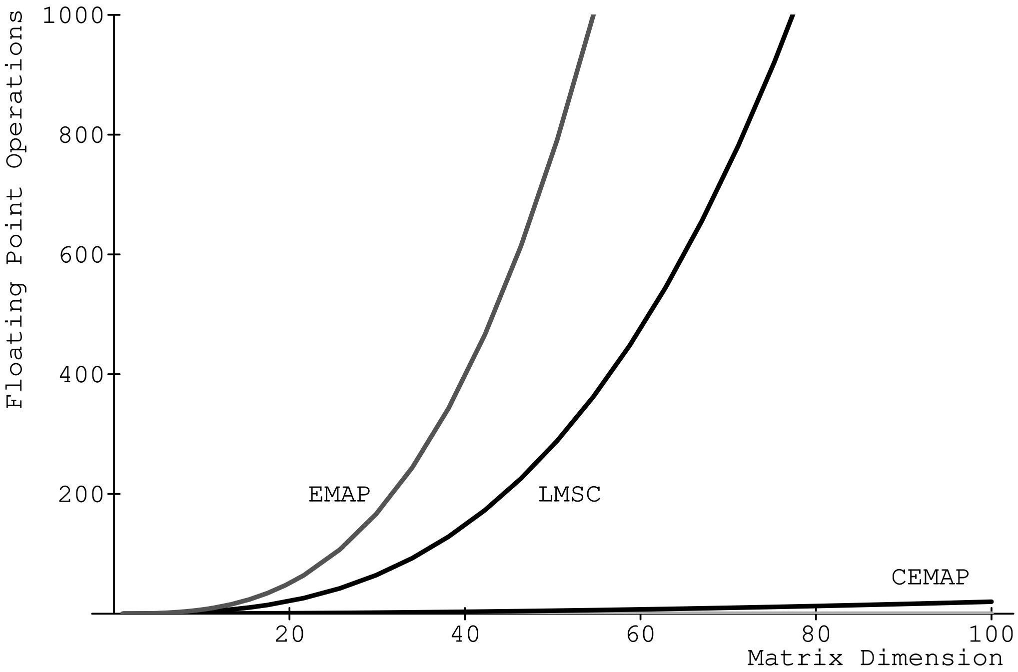

Figure 2-24: Number of floating point operations (in thousands) vs. matrix dimension for the estimation algorithms listed in Table 2-1. The ML curve is nearly coincident with the horizontal axis.

The CEMAP estimate was introduced as a fitting choice for the LMS desired signal due to its low computation and reasonable estimation accuracy. The CEMAP estimate uses fixed coefficients derived from the EMAP procedure to form the weighted combination of the *a priori* mean and the sample mean that would have been obtained after η samples had been observed by a particular speaker from each class of data. The parameter η determines these weighting factors. When the number of observations is near η, the CEMAP and EMAP mean-square errors are also close. Driving the LMS algorithm with this signal therefore reduces the estimation error with respect to other choices for the desired signal such as the ML mean. The parameter η was also shown to control the tradeoff between the misadjustment of the LMS-C estimates and its convergence properties.

The dependence of the ML, EMAP, and LMS-C estimators on the various statistical attributes of the data was investigated both analytically and empirically. Empirically, the LMS-C estimate was shown to produce a mean-square error which was often lower than that of the EMAP. This is possible because the LMS-C algorithm derives its coefficients from the data samples, unlike the EMAP estimate which has deterministic coefficients based on the number of observations. Although the stochastic gradient coefficient update introduces some asymptotic misadjustment, it appears to provide superior adaptation performance.

Analysis of the mean vector estimation problem in this chapter showed that when the dogmatism of the data is near 1.0, the LMS-C and EMAP estimators can substantially reduce estimation error with respect to the ML estimate when the adaptation training data is limited. The speaker adaptation problem is a limited-data setting. Chapters 3 and 4 investigate the effects which LMS-C and EMAP estimation have on the accuracy of both feature-based and stochastic model-based systems, respectively, when applied to the problem of speaker adaptation.

Chapter 3
Applications to Feature-Based Speech Recognition Systems

3.1. Overview

The general architecture of feature-based continuous speech recognition systems includes a feature extractor, a phonetic or other sub-word unit classifier, a word matcher, and a phrase hypothesization module, organized as what is essentially an expert system. The feature extractor codifys algorithms or heuristics which measure cues that represent the most important speaker-independent characteristics of the speech signal. Based on the observed values and temporal location of these features, the waveform is segmented and phonetic labels are applied by the classifier. Through application of knowledge about pronunciation, words are hypothesized from a phonetic network and placed into a word lattice with associated word scores. Parsing techniques can be applied to the word lattice to identify and score potential phrases and sentences which conform to the grammar of the given recognition task.

This chapter explores the application of the estimation algorithms from the previous chapter to feature-based continuous speech recognition systems. Specifically, it describes the effects on phonetic classification accuracy when these algorithms are used to update the system's acoustic-phonetic classifier parameters, and the extent to which correlation information can help to reduce error in the parameter estimation process. Unsupervised forms of the estimation algorithms are presented, as supervised feedback of phonetic labels is impractical in continuous speech recognition.

The chapter opens with a brief review of spectrogram reading, which highlights the types of information used in feature-based recognition. The Carnegie Mellon isolated-word FEATURE system, which successfully applied the EMAP algorithm for adaptation, is briefly described, followed by an overview of the CMU ANGEL continuous-speech recognition system. Section 3.4 explicitly describes the adaptation methodology adopted for the feature-based classification problem, and presents predictions of adaptation performance based on empirical tests using ANGEL front vowel statistics. Experiments described throughout this chapter focused on classification of the front vowels /*iy*/, /*ih*/, and /*eh*/, which exhibit a fair degree of correlation among formant frequencies and are sufficient to demonstrate the concepts set forth in this study.

Supervised adaptation results from ANGEL are described in Section 3.5. Results from these tests identified several problems which led to the development of an alternate feature-based system called PROPHET. As described in Section 3.6, the PROPHET system uses simple rules derived from the spectrogram reading process to compensate for some of the effects of continuous speech. PROPHET also incorporates unsupervised adaptation techniques. A lack of sufficient training data for estimation of Σ_o, however, made it necessary to investigate some aspects of feature-based adaptation using computer-generated data. These investigations are also described in Section 3.6.

At the time of the evaluation of PROPHET, the ANGEL system was no longer available for experimentation. The PROPHET experiments were therefore conducted using an approximation of the ANGEL system, which for expository convenience will be referred to as the SPIRIT system.

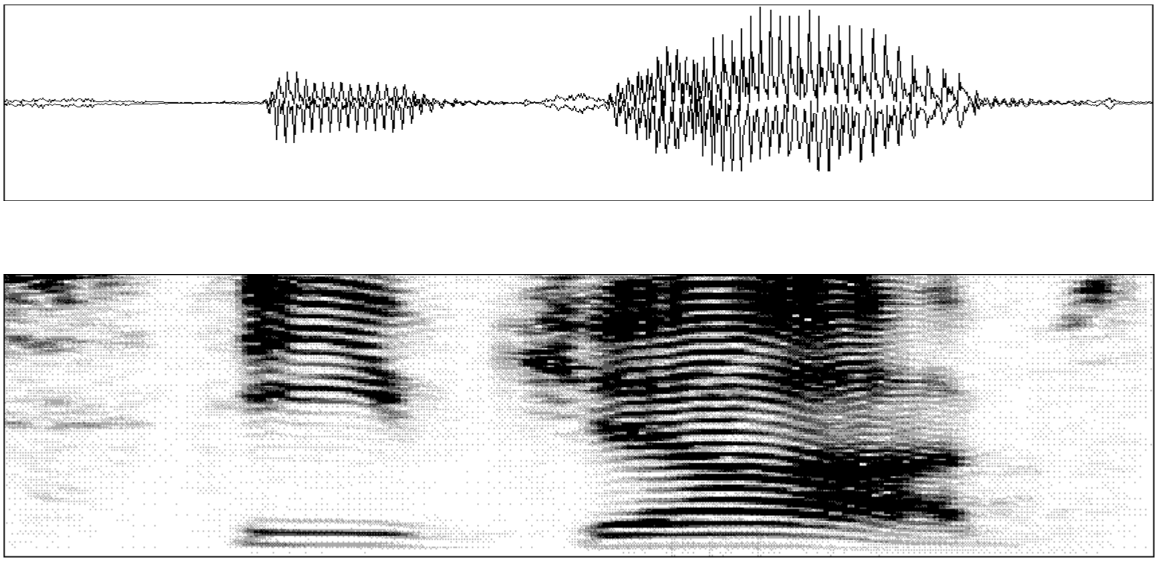

Figure 3-1: Waveform envelope and 0-4kHz spectrogram of the phrase "Steve Jobs".

3.2. Spectrogram Reading and Feature-Based Recognition

A spectrogram is a two-dimensional representation of the speech signal with time along the horizontal axis, frequency along the vertical axis, and density denoting amplitude (see Figure 3-1). In Cole *et al.* [27], researchers demonstrated that it is possible for an experienced person to identify the phonetic segments in a spectrogram with a high degree of accuracy. Spectrogram reading involves a complex decision process based on knowledge of speech production. It requires the detection of a relatively small set of fundamental cues in the spectrogram and associated displays of the waveform, zero crossings, or energy versus time. A detailed example can be found in a paper by Zue [28].

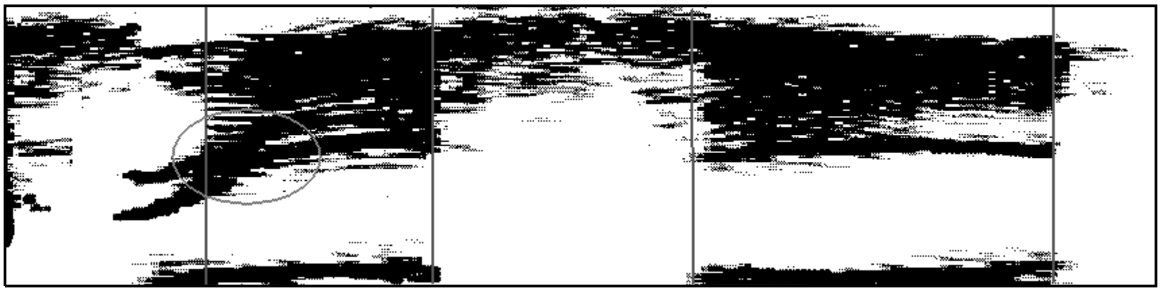

Figure 3-2: Spectrogram of the word *greasy* illustrating the phoneme /*iy*/ in neutral (right) and non-neutral (/*r*/, left) contexts.

Although every spectrogram reader uses visual cues in a unique manner, the basic steps are typically segmentation followed by labeling.

During the segmentation process boundaries are placed at points of large spectral or waveform change, which can indicate a change in the manner of articulation and the production of a different phoneme. Boundaries may also be placed on the basis of duration, as when an event is too long to be comprised of a single segment, or on the basis of formant trajectory where nonmonotonic movement can indicate the presence of multiple sonorants. When spectrographic information is ambiguous or insufficient for positive decisions, alternate segmentations may be provided. An experienced reader will identify the presence of more than 95% of the segments in an utterance.

Segments are labeled based on knowledge of a phoneme's characteristic spectral pattern, coarticulation effects, and phonological and phonotactic constraints. Produced in a neutral context, phonemes exhibit unique spectral patterns[9] which allows the reader to assign a segment label. In continuous speech, context can produce dramatic changes in these spectral patterns for which the expert must compensate. Knowledge of the legal subset of speech sounds in the language and the rules for combining them adds constraints to the labeling process. In English, for instance, not every phone can precede any other phone. Labeling performance of the best spectrogram readers exceeds 80% first choice accuracy.

When decoding continuous speech, knowledge of coarticulation or contextual effects is perhaps the most important. These effects occur due to the different degrees of sluggishness of the articulators, and can include formant motion, anticipatory coarticulation, combination in clusters, and end-of-utterance effects. The semi-vowels /*l*,*r*,*y*,*w*/ produce strong coarticulation effects, an example of which is given in Figure 3-2. The right-hand portion shows the phoneme /*iy*/ in a neutral context, while the left-hand portion shows an /*iy*/ in the context of a preceding /*r*/. In the /*r*/ context, the second and third formants are lowered in the first half of the /iy/ (within the gray ellipse), and then rise to attain their target values as in the neutral context. End-of-utterance effects are a weakening of

[9]See Cole [COLE80] or Zue [ZUE86B].

the speech signal, lengthening of sonorant regions, and potentially large amounts of glottalization or irregularities.

Feature-based recognition systems attempt to encode knowledge of the spectrogram-reading task, at the lowest levels of the system, through the choice of features extracted from the signal and the manner in which these cues are combined to assign labels to segments. The features are typically measures of formant values and their trajectories and amplitudes, waveform amplitude, spectral shape, segment durations, zero crossings, and energy in selected spectral bands. Chigier [29] provides a thorough discussion of feature selection for classification of stop consonants in the ANGEL system. The continuous-valued speech measures may be combined with a parametric or hand-crafted classifier, as in the ANGEL system, or they may be quantized to form descriptions of speech events for use in rule-based classification schemes. Lamel [30] developed an expert system for stop consonant classification which used a set of (human-generated) feature descriptions as input.

The features extracted from the acoustic signal will exhibit different degrees of speaker dependence due to natural variations in vocal apparatus and speaking style. Since one goal of speech recognition research is speaker independence, it is often necessary to incorporate a method of dynamic speaker adaptation to enable the system to learn an unfamiliar speaker's acoustic characteristics. In systems using parametric classifiers, adaptation can be performed by modification of those parameters.

One such adaptive system was the CMU FEATURE [23] system for speaker-independent, isolated letter recognition. FEATURE was based on a set of about 60 measures such as formant information, energy contours of unvoiced segments, and time to vowel onset. Because some measures are meaningful for only some letters, a decision-tree structure with adaptive gaussian classifiers at each node was used for classification. The EMAP algorithm was used to update all classes at each node after each observation. The average dogmatism of the data was around 0.40, and 0.14 at the lowest, so adaptation proved to be quite successful. The error rate in a supervised mode was reduced by 49% after four training tokens, and by 31% in an unsupervised adaptation mode.

Continuous speech has a number of effects or constraints not present in isolated speech which include inter-word contextual effects, lack of well-defined word boundaries, and the impossibility of perfect feedback. These effects distort feature measures and tend to increase their variability and consequently the dogmatism of the data. The lack of reliable feedback degrades adaptation algorithm performance as means are adapted on samples from the wrong class. Initial adaptation experiments by the author with the ANGEL system attempted to extend the FEATURE work to the continuous-speech case, in light of these limitations.

3.3. Overview of the ANGEL System

The Carnegie Mellon ANGEL system was a knowledge-based expert system, founded on the research by Cole, *et al.* [27] and designed for large-vocabulary, speaker-independent continuous speech recognition. The three main knowledge sources or modules in ANGEL were acoustic-phonetics, word hypothesization, and parsing. Data flowed from the low-level modules to the higher levels with little or no top-down feedback.

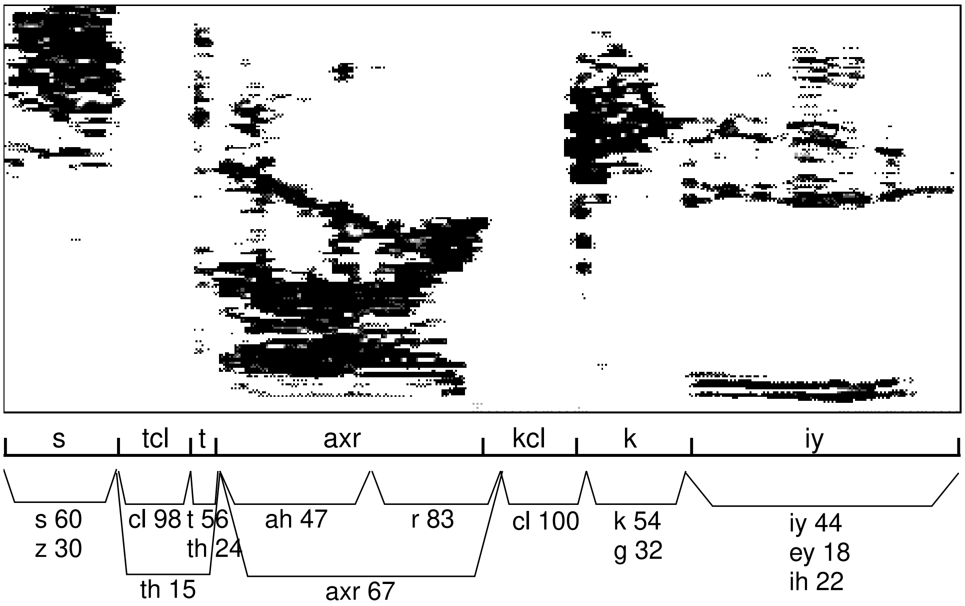

Figure 3-3: Spectrogram and associated phoneme network.

The acoustic-phonetic module was responsible for signal processing, feature extraction, segmentation, and phonetic classification. An example of the acoustic-phonetic module output is shown in Figure 3-3. The basic signal representation was DFT coefficients computed every 3 milliseconds, plus amplitude and pitch information. A broad-class phonetic network was generated from this representation using an algorithm which combines rules and statistics [31]. For segmentation, the DFT coefficients were smoothed in both time and frequency, and potential phonetic boundaries were tagged at points where there was a large Euclidean distance between adjacent frames. These boundaries defined seed regions which were combined on the basis of time-averaged spectra in an agglomerative hierarchical clustering procedure to obtain a dendrogram [32]. A gaussian classifier assigned broad class phonetic labels to the dendrogram segments, which were then pruned using a set of rules to form a preliminary phonetic network. A number of fine phonetic classification methods were explored, including hand-crafted classifiers, single gaussian classifiers per broad class, pairwise gaussian classifiers with a voting heuristic, and decision trees with gaussian classifiers at each node.

The phonetic network was passed to the word hypothesizer module to generate word candidates. As the network was traversed, it was matched against the task vocabulary stored in a lexicon to produce a word lattice containing begin and end times, and an acoustic-phonetic score. Each word in the lexicon was stored as a network of phonetic events; multiple paths indicated where multiple pronunciations were allowed. A word's score was the cumulative cost of traversing a path in its reference network. The parsing module applied semantic and syntactic constraints imposed by the language model to reduce the word lattice produced by the word module to a single sentence. These constraints were represented by second-order Markov models of sequences of syntactic and semantic categories of word candidates [33].

3.4. Adaptation Methodology and Empirical Predictions

Speaker adaptation experiments are performed within the acoustic-phonetic module of the ANGEL system. By statistically characterizing the speaker-to-speaker variability of training data and the variability of observations, the estimation procedures from the previous chapter can be applied to adjust the fine phonetic classifiers. Details of the supervised adaptation methodology are as follows. The unsupervised adaptation methods employed in the PROPHET and SPIRIT systems incorporate only minor modifications to the supervised approach and are discussed in Section 3.6.

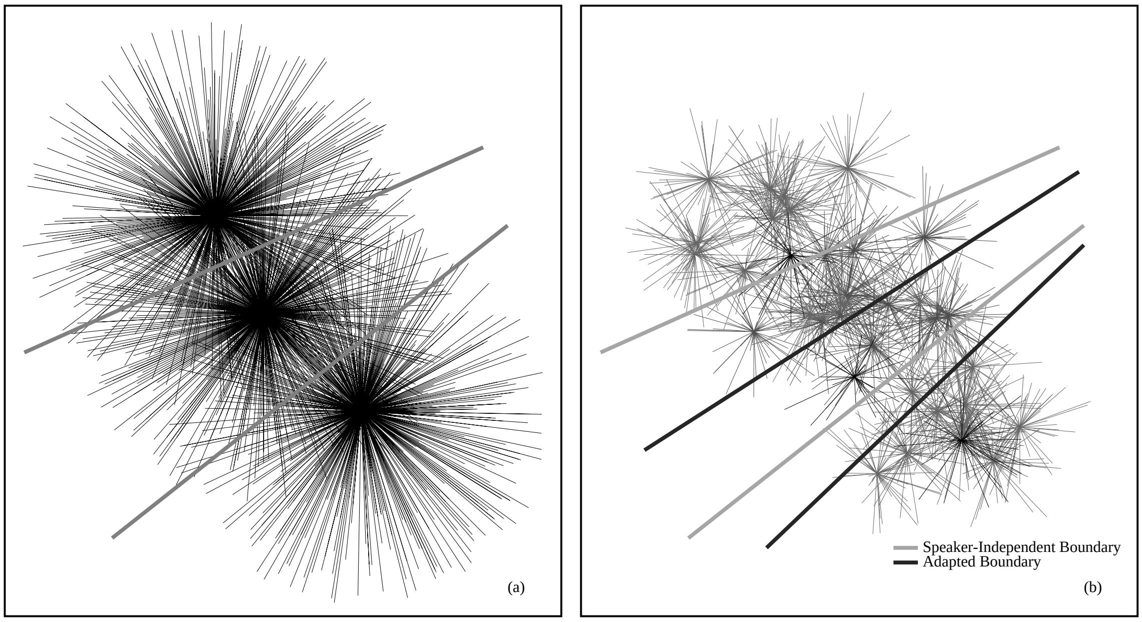

Figure 3-4: Three classes of data from multiple speakers: (a) pooled data and speaker-independent boundaries, and (b) individual speakers' data and a single speaker's adapted boundaries.

Let the set of observations χ be points in a D-dimensional hyperspace with feature values as the axes. Parametric or hand-crafted classifiers divide this space into mutually exclusive regions. Considering the data from C classes for many speakers, and assuming a multivariate Gaussian distribution, the data will appear as C possibly overlapping hyperellipsoidal clusters centered about the C class means μ_o, as illustrated in Figure 3-4(a) for C=3 and D=2. Speaker-independent classifier boundaries are placed to minimize classification errors for this pooled data (see the figure). The C clusters are actually composed of a number of smaller clusters, one per speaker, which are composed of points distributed about the speaker's mean μ. Figure 3-4(b) illustrates this point for a case with a dogmatism of 1.0. The gray distributions represent the data from multiple speakers for the three classes, while the black distributions represent the data from a hypothetical speaker of interest. The speaker-independent classifier boundaries cut through a large portion of this speaker's data distributions, which would result in a significant error rate. By replacing the *a priori* mean with an estimate of the speaker's mean, it is possible to shift the classifier boundaries to orientations which are a closer match to the speaker's data and reduce the error rate.

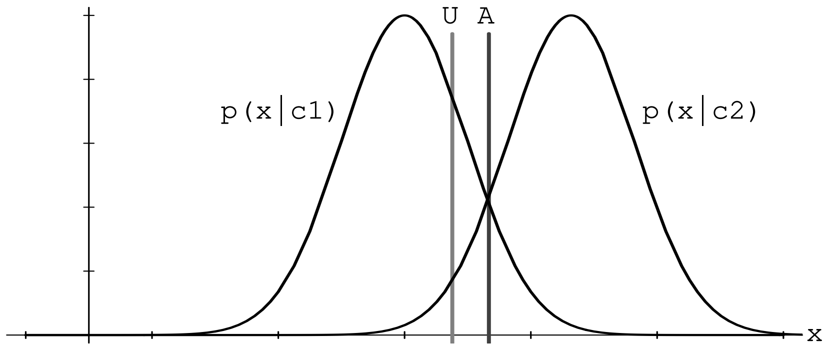

Figure 3-5: Hypothetical within-class data distributions with unadapted (U) and adapted (A) decision boundaries.

To provide some feeling for the potential gain due to this form of adaptation, consider again the 2-class, 1-feature case from Chapter 2. Assume that the data from a particular speaker are distributed as shown in Figure 3-5, and that the unadapted decision boundary is at the value indicated by U in the figure. The probability of error for this speaker would equal the area under the $p(x|c_1)$ to the right of U, plus the area under $p(x|c_2)$ to the left of U. Assuming equal prior probabilities $p(c_1) = 0.5 = p(c_2)$, the unadapted probability of error would then be

$$P_{\varepsilon,un} = 0.5\ \left[1 - erf\left(\frac{U-\mu_1}{\sigma}\right) + erf\left(\frac{U-\mu_2}{\sigma}\right)\right] \tag{3.1}$$

where $erf(x)$ is the standard normal error function

$$erf(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-u^2/2}du \tag{3.2}$$

If the speaker's mean values are estimated exactly, the adapted decision threshold will be midway between the two class means as indicated by the line at A. The probability of error now becomes

$$P_{\varepsilon,ad} = erf\left(\frac{\mu_1 - \mu_2}{2\sigma}\right) \tag{3.3}$$

and the percentage change in the error rate is

$$\Delta = \frac{P_{\varepsilon,un} - P_{\varepsilon,ad}}{P_{\varepsilon,un}} \tag{3.4}$$

An analytical solution for Δ in terms of dogmatism, correlation, and *a priori* means is not possible. Δ was therefore averaged over 1000 random values of μ for various values of correlation and dogmatism, assuming $|\mu_{o1} - \mu_{o2}| = 1$. The results are shown in Figure 3-6. As one would expect, the error rate reduction decreases with increasing dogmatism since the variation of means around μ_o becomes smaller, making the unadapted boundary location a better match to the data. The increase in adaptation gains with respect to correlation occurs because as the correlation rises there is a greater tendency for both class means to be located on the same side of the unadapted boundary, making the unadapted error rate very large.

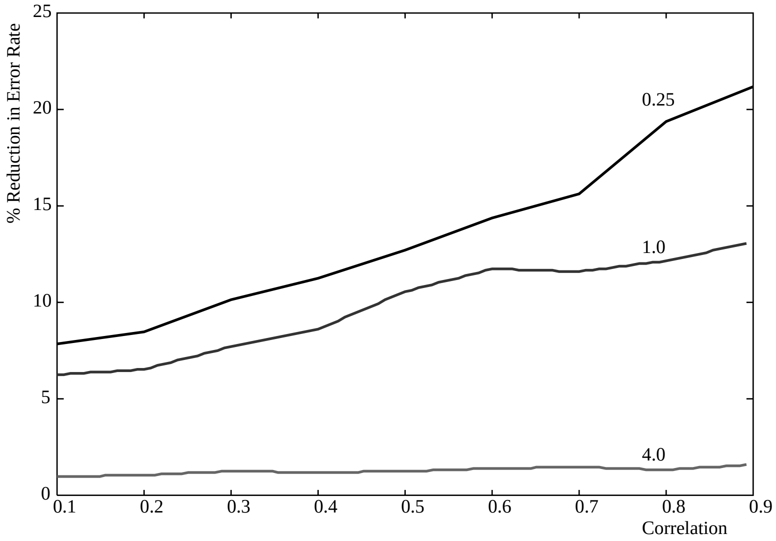

Figure 3-6: Percentage reduction in error rate vs. correlation for three values of dogmatism, using the model from Equation (2.60).

Application of the above methodology within the feature-based recognition systems considered in this chapter will be demonstrated for the class of front vowels /*iy*,*ih*,*eh*/ which are known to exhibit a fair degree of correlation between formant frequencies. For the feature data statistics in Appendix C, which were generated from context-independent front vowel formant data from the ANGEL system, the average correlation between features is approximately 0.5. The average dogmatism of these features is 2.0. Figure 3-7(a) shows the distributions of the first and second formant mean values as specified by the statistics, and 3-7(b) illustrates the within-class data distributions for $\mu = \mu_o$. In two dimensions these classes are highly confusable, and they remain so even as additional features are included. Note that 3-7(a) represents only the within-class correlation of the mean vector elements; six dimensions would be necessary to graphically represent the full interaction of the three class means.

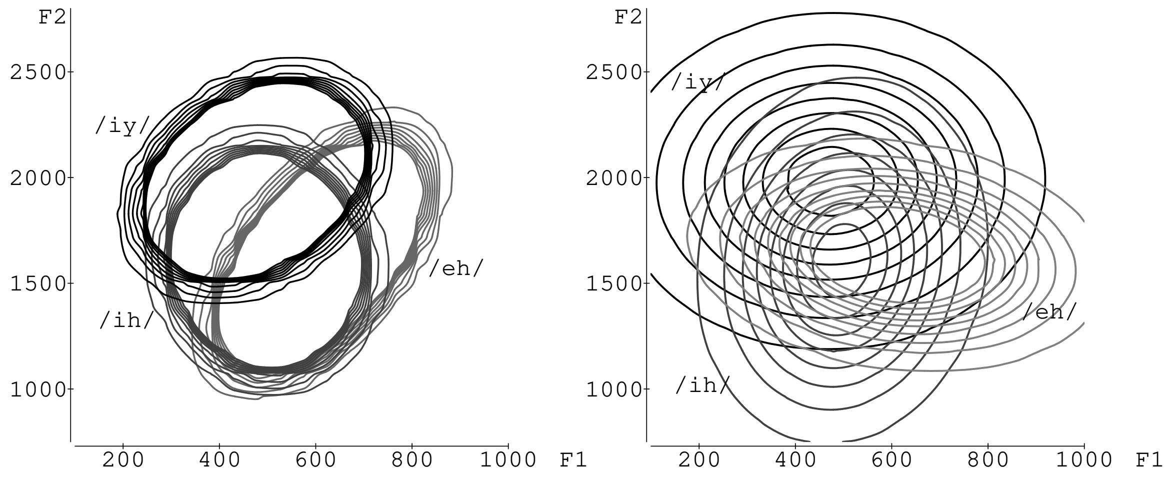

Figure 3-7: Distributions of the front-vowels first and second formants: (a) mean distributions, and (b) data distributions for $\mu = \mu_o$.

Using the expressions for mean-square error from Chapter 2, the learning curves as specified by Equation (2.61) and the front vowel statistics in Appendix D are shown in Figure 3-8. The statistics were generated with feature data from 179 speakers, 10 sentences per speaker, and a total of 6301 tokens. The average number of tokens per sentence is therefore approximately 3.5, so it appears from the graph that after 10 sentences (35 tokens) the EMAP or LMS-C mean-square estimation error should be less than one-third that of the unadapted mean. This assumes that the data obey the gaussian assumption upon which the theoretical analysis is based. Also, it should be noted that with the dogmatism of the data near 2.0, the difference between the adapted and unadapted means may be rather small, and (from the analysis above) the percentage reduction in error rate should tend to be less than that obtained with the FEATURE system.

To better predict what effect adaptation has on front-vowel classification rate given normally-distributed data, an empirical test was performed using the actual mean vectors of 9 speakers chosen at random from the set of 179. Five-hundred samples of computer-generated data were generated using ML estimates of the test speakers' mean vectors, and additional data were generated as adaptation data. The 500 samples were classified after both ML and EMAP[10] mean vector adaptation given perfect feedback of class membership. The 9-speaker average error rates are plotted versus the number of training tokens in Figure 3-9. The unadapted error rate was 39.5%, and the asymptote for the adapted means is around 26%, or a reduction of approximately 34%. More realistically, the EMAP error rate after 50 observations (closest to the number of front-vowel tokens in 10 sentences)

[10] The LMS-C algorithm had not been fully developed at the time of the ANGEL adaptation experiments.

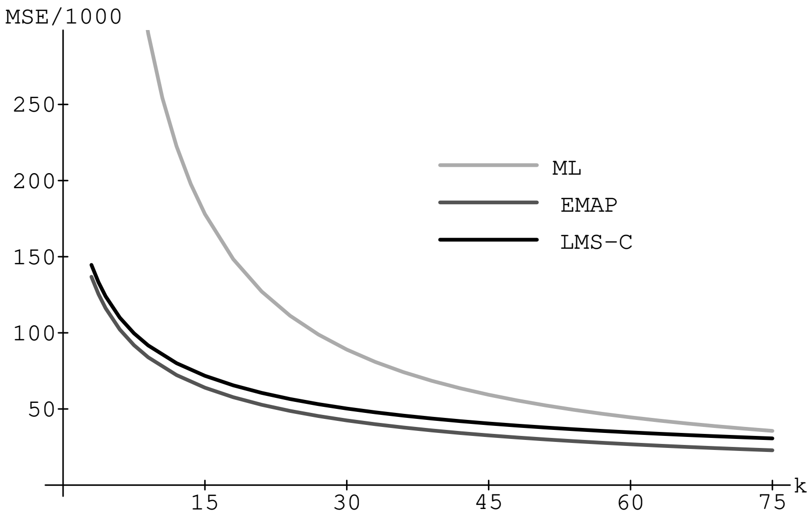

Figure 3-8: Learning curves as specified by Eqn. (2.61) for the ANGEL front vowel statistics.

was 35%, or an 11% reduction in error. Although actual tests with ANGEL use more than three features for classification, and imperfect label feedback, the simulations do provide some expectations for the upper bound on the improvement by applying these techniques to adaptation in the real system.

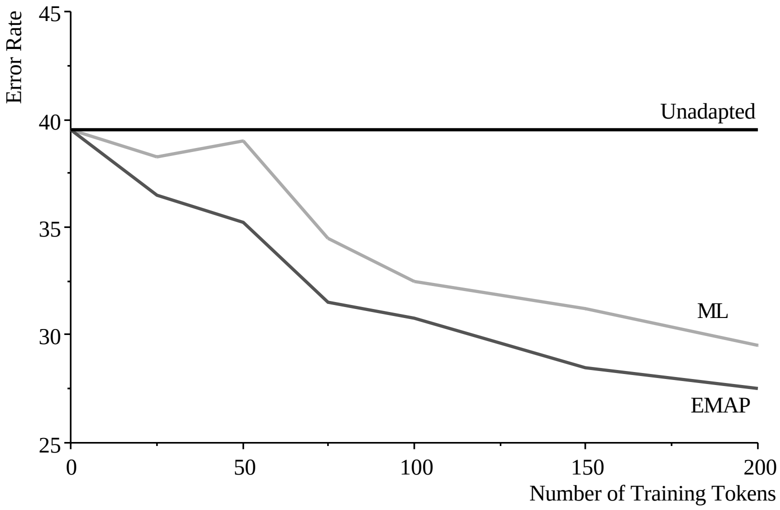

Figure 3-9: Empirical adaptive classification results with computer-generated, normally-distributed data generated from ANGEL front vowel statistics.

3.5. Adaptation Results from the ANGEL System

The first set of experiments with real data from the ANGEL system was conducted using perfect feedback of phonetic class membership and segment boundaries available from phonetic transcriptions produced by spectrogram readers. The data were from the three front vowels, independent of context, from 179 speakers with 10 sentences per speaker for a total of 6301 tokens. Each sentence was in turn held out and used for testing, while the remaining 9 sentences were used for adaptive training. Ten features were selected from an available set of 130 features using a Fisher ratio test as described in [29]. These features included segment duration, average pitch, average values of the first three formants, and a few measures of energy in selected frequency bands. Note that due to the use of manual segmentation and labels, results from these experiments should be regarded as an upper bound on the improvement in performance to be expected due to adaptation.

Adaptation Type	Error Rate	Percent Reduction
Unadapted	33.8%	--
ML	26.4%	21.8%

Table 3-1: Classification error rates for the ANGEL system front vowels after 10 and 20 adaptation sentences.

The maximum likelihood algorithm was used for adaptation to facilitate rotation through the test sentences.[11] As shown in Table 3-1, the unadapted and adapted error rates were 33.8% and 26.4%, respectively, which represents a reduction of 21.8% due to adaptation. These figures are comparable to the predictions obtained using computer-generated data. The larger gains with the real data may be attributable to the additional features which were used in the ANGEL tests. Given the imperfect segmentation in the system's acoustic-phonetic module and unsupervised adaptation methods, however, these gains will be lower in practice.

The context-independent result does not approach the gains obtained in the FEATURE system, which showed a 49% reduction in error rate in the supervised adaptation mode. In FEATURE, phonemes were produced in a limited number of contexts, so their acoustic realizations were likely to have been consistent. This consistency makes the within-class variance small with respect to the between-speaker variance, producing the low dogmatism which was observed at each node in the decision tree. Contextual effects in continuous speech cause feature values to be less consistent than the features from isolated words in the FEATURE system, increasing the variability and dogmatism of the features in ANGEL. Several experiments with context-dependent classification in ANGEL were

[11]The experiments were structured such that the contribution of a set of data could be easily removed from the statistics, which were originally estimated using ML techniques.

performed to determine whether explicitly modeling contextual effects would produce a lower error rate. Due to the increased number of classes, however, there were not enough training data for reliable estimation of the adaptation statistics. As a result, the context-dependent adapted error rates were higher than the unadapted rates and these tests were abandoned.

Other types of classification schemes within ANGEL were also investigated, including a decision tree structure similar to that in FEATURE. Unfortunately, classification rates at the highest decision nodes were on the order of 80% which were not believed to be high enough to justify use of this structure. Also, the dogmatism at these nodes was large enough to preclude the use of adaptation. Chigier [29] performed extensive optimization of a decision tree classifier for stop consonants within ANGEL, but was able to reduce the error rate by only 7.5% over that of a single gaussian classifier (37% vs. 40% error rates).

The ANGEL adaptation experiments were unsuccessful because the gaussian model does not adequately represent the features of continuous speech, making the classification scheme inappropriate for the actual distributions of the data encountered by the ANGEL system. Figure 3-10 shows four histograms of feature data for the vowel /*iy*/ and the gaussian densities which model it. The shape of a feature's distribution depends on the inherent variability of the feature and the manner in which they are measured. Many feature distributions in the ANGEL data were skewed, bimodal (usually a mixture of two normal distributions corresponding to the male and female subpopulations), or uniformly distributed, as indicated in 3-10 (a), (b), and (c). Some features were adequately modeled by a normal density; these features, like the formants shown in 3-10(d), can usually be related to anatomical characteristics. As was seen in the analysis of Chapter 2, the estimation algorithms require a unimodal probability density function for the data that is centered about the mean, and when this property is absent the estimates cannot converge. Chigier [29] also observed a poor match between stop features in ANGEL and gaussian densities used to model them.

A single gaussian classifier, therefore, was not an appropriate structure for classification of speech data in ANGEL. Some features were derived for specific discriminations and are not appropriate for all classes. A decision tree structure is a better choice for the phonetic classification problem, as FEATURE demonstrated. Given the increased variability of continuous speech feature data, however, this structure falls short of the requirements for highly accurate recognition. When a human expert decodes a spectrogram, some features are used qualitatively to narrow the candidate classes, and other quantitative features are adjusted to compensate for contextual effects. For vowel classification, compensated formant values are mentally adapted given previous data from the given speaker before the final labels are applied. If feature-based recognition is based on the same features as those used by an expert spectrogram reader, then a similar decision process should be used to obtain classification results which approach the human expert levels. This approach is demonstrated by the PROPHET system, which is described in the remainder of this chapter.

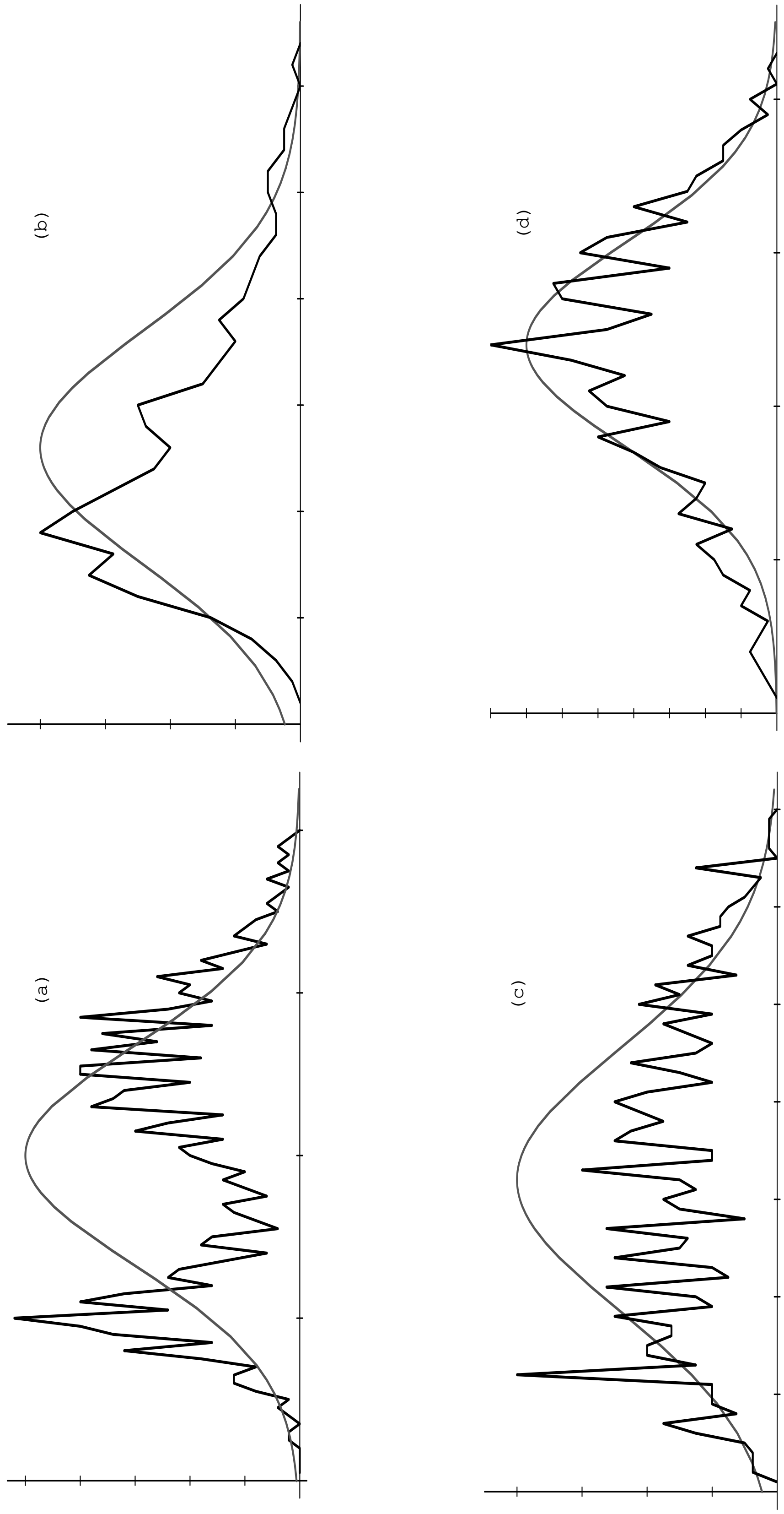

Figure 3-10: Histograms illustrating the fit of the gaussian model to ANGEL feature data for the vowel /*iy*/.

3.6. The PROPHET Phonetic Classifier

3.6.1. Overview of PROPHET

The PROPHET system attempts to address the shortcomings of the ANGEL phonetic classification module by demonstrating feature-based recognition techniques derived from expert spectrogram readers' decision processes. Aspects of these processes are modeled with a rule-based system which uses qualitative information to guide the application of statistical classifiers. Often a spectrogram reader uses qualitative descriptions of parameter values to prune the candidate decision classes for a segment. Afterwards, quantitative judgements are made to assign a phonetic label. For example, when classifying front vowels, a human expert may first look at the waveform to verify the presence of voicing or periodicity, and to determine whether the amplitude is sufficient for a vowel. If the formants are in a fronted pattern, then duration and formant values are used to assign a front-vowel label to the segment. The feature values may be adjusted to compensate for contextual effects or to adapt to the acoustic characteristics of the speaker, especially if the reader has previously decoded spectrograms from the speaker.

Many aspects of the spectrogram reader's decision process are modeled in PROPHET in a straightforward manner. Thresholds on feature values such as waveform amplitude or power, formant amplitudes, frontness, and probability of voicing are used to filter out any segments which do not satisfy the characteristics of the target classes (the front vowels). Specifically, segments with low power, low probability of voicing, or frontness less than 1.0 are ignored. Further, if any formants are weak, as is often the case with semivowels, that segment is rejected. Once a segment has been selected by this sorting procedure, a gaussian classifier is used to assign one of the target class labels. Only those features which are useful for discrimination between members of the target group are combined in the classifier. For the front vowels, these features are the first three formant frequencies, and segment duration. Note that the classifiers may have been previously updated given some amount of adaptation training data from the current speaker.

The lack of training data for context-dependent modeling implies a need to keep the number of decision classes small and to compensate for the effects of context and other sources of variability in the feature data, as opposed to modeling them explicitly. One method of context compensation investigated in PROPHET was target frequency identification, which is applied to formant frequencies within vowel segments. This process attempts to identify the value which the formant would have assumed if it were produced in a neutral context, much as a human spectrogram reader would do. To perform this identification, a set of rules are applied to a qualitative description of the formant trajectory (the time derivative of the formant frequency). A rough estimate of this derivative is obtained by computing the change in averaged frequencies between the first and middle thirds of the segment, and between the middle and last thirds. These changes are then quantized to obtain descriptions of the formant evolution in each half of the segment. For example, "RC" describes a formant which is rising in the first half and constant in the second half. The second and third formants in the circled region in Figure 3-2 would be described in such a manner.

The target frequency identification rules can be summarized as follows. If the formant is constant for any period, assume it has reached its target value and average over the constant region. If the formant rises then falls or vice versa, use the extremum as the target. If none of these rules apply, *i.e.* the formant is rising or falling throughout the segment, then another set of formant-specific rules is applied to consider the left and right contexts (assumed to be available) to explain this behavior and choose appropriate values. These context-dependent rules mostly apply to semivowel, nasal, or labial contexts. For example, if the second formant was rising throughout a segment and the left context was a /*w*/ (which pulls the second formant downward), then the values in the righthand portion of the segment are likely to be closer to the target values. Application of these rules should reduce the dogmatism to levels closer to those seen with the FEATURE system.

The PROPHET classification techniques described above place restrictions on the manner in which features are used in the decision process. These restrictions essentially state that a feature should be applied to make only the decision or discrimination for which it was originally derived. The restrictions help to overcome some of the problems with the gaussian model as applied to the feature data, as they allow a number of features to be omitted from the gaussian classifiers. To ensure a reasonable match between the remaining features' distributions and the gaussian adaptation model, additional restrictions are placed on feature selection. The features chosen are limited to be those which can be directly related to a speaker's physical characteristics, or to temporal variables. For example, formant frequencies, which are vocal tract resonance frequencies, exhibit a normal variation related to the variation in the human vocal apparatus. To avoid bimodal distributions, the male and female subpopulations are modeled separately, *i.e.* statistics are generated for each gender. Histograms of features in the PROPHET system, chosen under these restrictions, show the data to be normally distributed.

The ANGEL system was no longer available for experimentation by the time the PROPHET system was ready to be evaluated. For that reason, evaluations of the PROPHET system were made in comparison with an approximation of the ANGEL acoustic-phonetic module. This baseline system, which is referred to here as SPIRIT, is a single gaussian classifier which operates on segment-averaged feature values. Evaluations of the two systems, both with and without adaptation, are presented in the next section.

3.6.2. Experiments with PROPHET

The previous section described how the PROPHET system proposes to reduce variability in continuous speech and to improve the quality of the gaussian model. The experiments described in this section investigated classification rates with and without rules for segment selection and context compensation. Also described are adaptation experiments which compare error rates for the adapted systems using the estimation algorithms from Chapter 2 in an unsupervised adaptation mode. This is preceded by a description of the speech database and feature generation method.

3.6.2.1. Database and Generation of Features and Adaptation Statistics

The training, adaptation, and evaluation data were from the DARPA TIMIT [34] prototype CD-ROM which consists of 10 sentences from each of 420 speakers, 290 male and 130 female. Two sentences, referred to as dialect calibration sentences, are common to all speakers in the database. These sentences have been excluded to avoid biasing the adaptation statistics and recognition results. The remaining 33,600 sentences contain a total of 62,500 sonorant segments, 10,700 of which are labeled as front vowels. This is an average of 25 front vowels per speaker, consisting of 10.1 /*iy*/'s, 8.3 /*ih*/'s, and 6.6 /*eh*/'s.

The TIMIT database is arranged into eight subdirectories representing different U.S. English dialect regions. For each recorded sentence, there is an associated orthographic transcription, and a phonetic transcription which contains labels and segment boundaries produced by spectrogram readers at MIT. Since segmentation and broad phonetic class assignment are not under investigation here, the information in the TIMIT phonetic transcriptions are used as input to PROPHET. The TIMIT CD-ROM is a prototype database developed as part of a standardization process within the speech research community. As a prototype, the data are known to contain a small number segmental labeling errors. In the following it is assumed that the frequency of labeling errors for the classes considered is negligible.

The digitized speech was analyzed using the *formant* program from the *waves+* software package by Entropic Speech, Inc. [35]. The *formant* program incorporates an LPC-based formant-tracking algorithm [36] and a pitch-tracking algorithm [37] which also provides waveform power and probability of voicing information on a frame-by-frame basis. Formant frequencies are specified by the algorithm, and formant amplitudes can be computed from the LPC pole locations in the formant tracker output. Formant amplitude information was normalized by the sum of the formant amplitudes to reduce their dependence on changes in loudness or stress within an utterance. Waveform power was normalized to a maximum of 1.0 to reduce this dependence across utterances. The frame-by-frame measures produced by *formant* were then averaged over the extent of each phonetic segment. Frontness was computed from the segment-averaged formant frequencies, and duration was determined from the phonetic transcription. The processed feature values and phonetic transcription information for each utterance were stored in the database. A similar process was used to generate a second set of feature data files using the rule-based methods described in the previous section.

Statistics were generated using maximum likelihood estimates of the means and covariance matrices for the target classes /*iy*/, /*ih*/, and /*eh*/, and for an additional generic class *son* which is the union of all other sonorants. These statistics were computed for each speaker, and the number of tokens which contributed to each speaker's statistics was recorded. The token counts allow the contribution of an individual's data to be removed from the statistics. This facilitates rotation though the data (holding out one sentence for testing and training on the remainder) which was necessary given the limited amount of available data per speaker.

3.6.2.2. Experiments with TIMIT Data

The first set of experiments compared the error rates of the SPIRIT system, which uses no rules, with PROPHET which uses rules to select segments for classification and to identify target formant frequencies. The comparisons were performed between both unadapted systems and systems using ML, EMAP, and LMS-C adaptation in an unsupervised mode. Unlike the analysis in Chapter 2 and the ANGEL experiments from Section 3.5 which assumed labeled training tokens, the experiments reported here updated each class based on the probability that the current observation is a member of the class. In other words, during adaptive training, for each class j the ML mean $\mathbf{a}_j$ is updated as

$$\mathbf{a}_j = \frac{\sum_{k=1}^{n_j} f(\mathbf{x}_k|j)\mathbf{x}_k}{\sum_{k=1}^{n_j} f(\mathbf{x}_k|j)} \tag{3.5}$$

where $\mathbf{x}_k$ is the k^{th} *unlabeled* sample and $f(\mathbf{x}_k|j)$ is the probability that the sample is from Class j. The probabilities $f(\mathbf{x}_k|j)$ are assigned by the the classifier during adaptive training. The ML mean $\mathbf{a}_j$ is then used in the EMAP and LMS-C procedures as in the supervised mode.

Adaptation Type	Error Rate Without Rules (SPIRIT)	Error Rate With Rules (PROPHET)	Percent Change
Unadapted	55.3%	34.6%	-37.4%
ML	68.9% (+24.6%)	36.2% (+4.6%)	-47.5%
EMAP	71.6% (+29.5%)	24.6% (-28.9%)	-65.6%
LMS-C	58.7% (+6.1%)	24.6% (-28.9%)	-58.1%

Table 3-2: Error rates of the unadapted and adapted SPIRIT and PROPHET systems for the four-class experiment, which includes the *son* class.

The results for this first set of experiments are given in Table 3-2. The second column shows the error rates for the SPIRIT system (without rules) and the third column is the error rate using the PROPHET system (with rules). Entries in the last column show the percentage change in error rate between the SPIRIT and PROPHET systems, *i.e.* change across rows in the table. The numbers in parentheses in the second and third columns are the percentage change in error rate between the unadapted and adapted systems. The table indicates that the rules substantially reduced the error rate over that of the baseline system. Inspection of confusion matrices from each experiment indicated that the rules were effective in filtering out the non-front-vowel segments (the *son* class) from the front-vowel classifier regions. This ability to exclude the *son* class is crucial for the unsupervised form of adaptation used here, as comparison of columns two and three indicates. As the SPIRIT system adapts, the front-vowel classes learn the characteristics of the *son* class because of the large number of *son*'s which fall in these regions. This reduces the quality of the front-vowel classifier parameters, resulting in more *son* samples being labeled as one of the front vowels. The PROPHET system automatically places segments without all of the characteristics of front vowels into the *son* class. These samples cannot adversely affect the front-vowel parameters or classifier boundaries, the gains due to adaptation are therefore much greater.

To assess the utility of the target-frequency identification rules, a series of three-class experiments was performed. These experiments were identical to the four-class tests in Table 3-2 except that the *son* class was excluded. Results from the three-class experiments are listed in Table 3-3. It appears that the target frequency identification rules adversely affect the error rate for both adapted and unadapted systems. Further inspection of Table 3-3 shows that the ML systems' error rates are only slightly changed from the unadapted system, while the EMAP and LMS-C systems' error rates are substantially worse than the unadapted. This counterintuitive result is due to inaccuracies in the estimation of the covariance matrix Σ_o, as will soon be shown. Inaccurate estimation of Σ_o causes the coefficient matrices, and hence the mean estimates, in the EMAP and LMS-C procedures to be less than optimal for the data. This moves the classifier boundaries to orientations which are poor representations of the data, raising the classification error rate. Given these estimation problems, it cannot be determined whether the target frequency identification rules have any positive effect on the adapted error rates.

Adaptation Type	**Error Rate Without Rules (SPIRIT)**	**Error Rate With Rules (PROPHET)**	**Percent Change**
Unadapted	28.7%	36.7%	+27.9%
ML	29.3% (+2.1%)	36.1 (-1.6%)	+23.2%
EMAP	45.2% (+57.5%)	53.9% (+46.9%)	+19.2%
LMS-C	44.1% (+53.6%)	51.2% (+39.5%)	+16.1%

Table 3-3: Error rates of the unadapted and adapted SPIRIT and PROPHET systems for the three-class experiment (*son* class excluded).

That the covariance matrix Σ_o is poorly estimated can be deduced as follows. First, the error rates from the ML-adapted systems are only slightly changed from the unadapted error rates. This small change is expected given the relatively small number of observations. The main difference between the ML and the other estimators is that the ML estimate does not make use of the statistics μ_o, Σ_o, or Σ. The mean vector μ_o and covariance matrix Σ are reliably estimated because data from all speakers can be pooled to obtain them. An estimate of the covariance matrix Σ_o, on the other hand, is formed as the sum of the outer product of each speaker's mean vector and so is based on fewer summands. If these individuals' means are poorly estimated because few observations are available from a class, the resulting statistic (Σ_o) will not accurately represent the distribution of the true means.[12] This claim is supported by the results experiments using computer-generated data which are reported in the next section.

3.6.2.3. Experiments with Computer-Generated Data

The speech data from the TIMIT prototype CD-ROM were chosen for this thesis because they represent the largest body of phonetically-transcribed data available on-line. Unfortunately, there are not enough data available per speaker to address some of the adaptation issues in this study. Computer-generated data with statistics equal to those estimated from the TIMIT data will be used to investigate these issues. Tests on these computer-generated data will provide an upper bound on the expected performance with real data. Due to the good agreement between the SPIRIT system's three-class unadapted error rate (28.7%) and the corresponding computer simulations (29.8%), simulations reported here consider only this set of statistics.

For each experiment, the results were averaged over 25 trials. For each trial a true mean vector μ_t was generated according to the distribution $N(\mu_o,\Sigma_o)$ as specified by the front vowel statistics in Appendix C. Assuming equal prior class probabilities, 512 samples of evaluation data were generated according to $N(\mu_t,\Sigma)$. Additional adaptation training data were generated as needed. The adaptation training data were used to find an estimate of the trial mean using ML, EMAP, or LMS-C procedures. Each mean estimate was initialized with the *a priori* mean μ_o. After the adaptive training process, the 512 evaluation data were classified using the adapted mean vector and the *a priori* covariance matrix as the classifier parameters.

The first tests with computer-generated data were intended to show the effect on error rate of the type of feedback describing which class was correct that is supplied to the adaptation algorithms. In Chapter 2, it was assumed that the training samples were labeled. Since this "supervised" adap-

---

[12]It should be pointed out that the ML mean estimate performed reasonably well in classification tests because it was initialized with μ_o, which lessens the effect of a lack of data. This initialization wasn't used in estimation of Σ_o to avoid adding any bias to this statistic.

tation mode is unrealistic for phonetic classification in continuous speech, two methods of unsupervised adaptation were considered. The first method was the probabilistic update specified by Equation (3.5). The second method was similar in that it was based on the sample's probabilities of class membership; the difference was that only the most likely class was updated. These two approaches to unsupervised feedback to the classifier are referred to as soft-decision and hard-decision feedback, respectively.

Figure 3-11 shows the error rate for ML adaptation versus the number of adaptation training samples for the supervised and two unsupervised adaptation methods. As one would expect, the supervised adaptation error rates (lower curve) are consistently lower than the unsupervised methods. The soft-decision error rates are a few percent higher than the supervised rates, and the hard-decision rates are, at least asymptotically, only slightly higher than those of the soft-decision method. The rise in the hard-decision error rate after 10 training samples occurs because 10 tokens do not provide enough information to obtain a good mean vector estimate when some of the class membership decisions are wrong. The soft-decision curve does not exhibit this behavior because every class is updated, even if only slightly, after each observation. For these reasons, only the soft-decision method of unsupervised adaptation is considered in the remainder of this section.

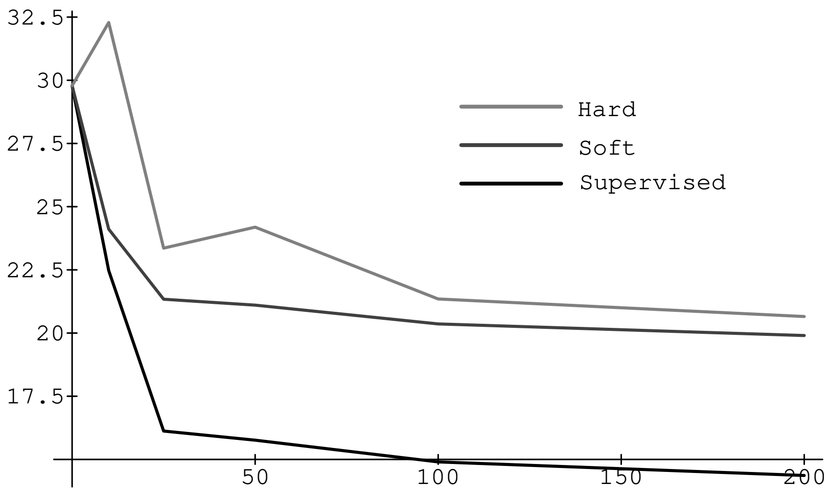

Figure 3-11: Error rates for ML adapted baseline system for three forms of feedback: hard-decision, soft-decision, and supervised.

Motivated by the reestimation procedures from hidden Markov model theory (see Section 4.2), a similar process of multiple iterations of the unsupervised adaptation algorithm was investigated for the feature-based classification problem. Each iteration consisted of initialization of the mean estimate to μ_o, followed by adaptive training. After each pass through the adaptation training data, the current classifier mean vectors were replaced by the new estimates, and the process was repeated. Since the probabilities assigned to each training sample (and the elements of the matrix **N**) are determined by the current classifier parameters, each iteration through the training data refines these probabilities until they converge. Iteration in this manner allows the unsupervised estimates to come closer to the level of error of the supervised estimates.

Figure 3-12 shows ML, EMAP, and LMS-C error rates as a function of the number of training samples after one (left) and five (right) iterations of the soft-decision adaptation algorithm. The LMS-C error rates are close to the EMAP rates when the number of training samples is small, and both are obviously better than the ML rates. Specifically, given 10 training samples and a single iteration of adaptive training, the ML estimate reduced the error rate by 19.1% (comparable to the ANGEL result of 21.8%). The EMAP and LMS-C estimates reduced the error rate by 29.3% and 26.3%, respectively. The asymptotic error rate for all estimators is approximately 20.5% after one iteration and 18.5% after five iterations. The supervised asymptotic error rate was determined to be 15% (see Figure 3-11), so the multiple iterations were able to reduce the error rate by an additional 10%, or about 36% of the difference between the one iteration and supervised levels.

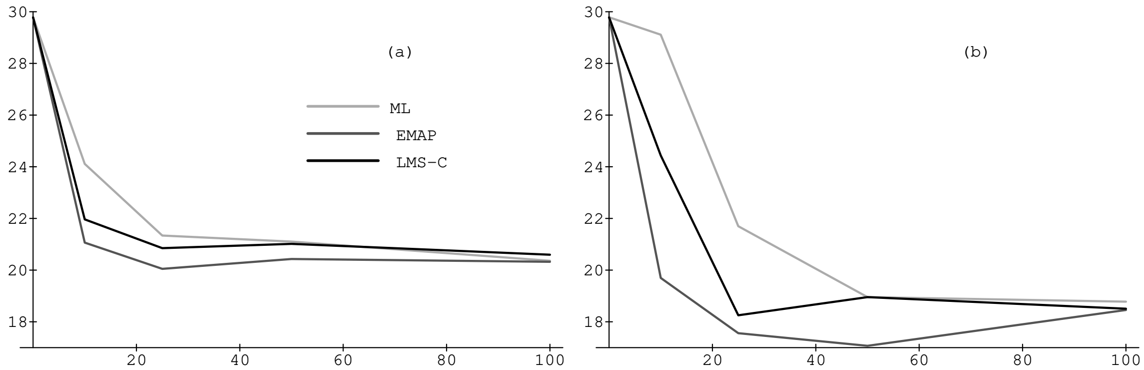

Figure 3-12: ML, EMAP, and LMS-C error rates after (a) one and (b) five iterations through the adaptation training data, as a function of the number of training tokens.

It is believed that similar gains would be obtained using real data in the three-class experiment that is summarized by Table 3-3 if an accurate estimate of Σ_o were available. As stated earlier, the estimate of Σ_o for the front vowels in the TIMIT prototype database did not represent the true distribution of the speakers' mean vectors. The simulations reported above support this claim. Histograms of the pooled data from real experiments were closely matched by the gaussian model. It is believed, based on the the physical relevance of the features at issue, that data from individual speakers are also normally distributed. The only remaining difference between the data from real experiments and simulations using computer-generated data is that the computer-generated data are known to obey the specified statistics, or conversely the statistics truly reflect the distribution of the data. Results from these experiments show that both EMAP and LMS-C perform well even when the amount of training data is small, and similar results are to be expected in real systems having ample training data.

Finally, note that the LMS-C parameters η and β can be adjusted to make the LMS-C error rate closer to the EMAP rate at other points along the horizontal axis. Since all three estimators converge as the number of training samples increases, it is best to optimize LMS-C performance for small numbers of training samples and to use ML adaptation when the training data are plentiful.

3.7. Summary

In this chapter, the estimation algorithms derived in Chapter 2 were applied to adaptation in feature-based continuous speech recognition systems. For the set of three front vowels, supervised adaptation in the ANGEL system produced a reduction in error rate of 21.8% using ML estimation. This figure represents an upper bound on adaptation performance; results with unsupervised adaptation will be lower. The ANGEL results were limited by the fact that many features are not adequately described by the gaussian distribution. As demonstrated in Chapter 2, performance of all three estimators under investigation is degraded when the data violates the gaussian assumption.

The PROPHET phonetic classification system was developed to address limitations observed in the ANGEL system. PROPHET embodies rules derived from the spectrogram-reading process to deal with the effects of context, and to make classification decisions using both gaussian and non-gaussian features. Restrictions placed on adaptation features, including separate modeling of male and female speakers, ensure that these features obey the gaussian assumptions of the adaptation algorithms.

Comparisons of the baseline SPIRIT system (substituted for ANGEL which was no longer available) and the PROPHET system results show that the strength of the PROPHET system lies in its ability to filter out segments (from the *son* class) which should not be considered as members of the target classes. The result was a 37% reduction in error rate for the unadapted systems. Results on the use of the context-compensation rules in PROPHET were inconclusive due to poor estimation of the statistic Σ_o which is crucial in the EMAP and LMS-C procedures. These results, based on the TIMIT prototype CD-ROM data, indicate that a certain minimum number of observations per speaker (from each class) are necessary to obtain reliable estimates of Σ_o.

Unsupervised adaptation using EMAP or LMS-C estimation algorithms within PROPHET was shown to be effective in reducing the classification error rate, and LMS-C performance was equal to that of the EMAP algorithm. Soft-decision feedback, which is similar to the reestimation procedure used in HMMs, was shown to be an effective method for unsupervised adaptation. In experiments using computer-generated data, the EMAP and LMS-C algorithms reduced the classification error rate by 29.3% and 26.3%, respectively, after presentation of 10 adaptation data samples. This represented a gain of approximately 40% with respect to the 19% reduction in error rate obtained with the ML-adapted mean vectors. The EMAP and LMS-C error rates after 10 observations were close to their asymptotic values.

Simulations showed that multiple iterations of the unsupervised adaptation algorithms reduce the error rate further, as the probabilities of class membership used in the soft-decision feedback method converge. When five iterations of this unsupervised adaptation were performed on the adaptation training data, the error rate was reduced by an additional 10% over that of one iteration. This difference represented 36% of the difference between the single-iteration and supervised-adaptation error rates.

Chapter 4

Application to Hidden Markov Model Speech Recognition Systems

4.1. Overview

Speech is an observable signal. Modeling the speech signal well allows the development of devices which recognize it, synthesize it, or identify the speakers who produced it. Statistical models of signals such as speech treat the observation sequence as a realization of a stochastic process and attempt to characterize the statistical properties of that process. The hidden Markov model (HMM) is a stochastic signal model that has been widely applied to the speech recognition problem. A number of variations of HMMs exist, the main difference between them being the form of the observation density functions. For HMMs using continuous density functions, estimation of possibly correlated mean vectors is an integral part of the parameter estimation process. This chapter investigates the application of the algorithms from Chapter 2 to the estimation of mean vectors in a hidden Markov model. More precisely, the estimates are used to update the HMM parameters given speaker-specific observations.

Section 4.2 reviews the theory of hidden Markov models, and Section 4.3 describes the restrictions placed on them for the speech recognition task. Discrete, continuous, and semi-continuous HMMs are presented. Vector quantization and the estimation of continuous mixture densities are briefly discussed as they apply to HMMs. Section 4.4 describes speaker adaptation experiments which were performed within a semi-continuous version of the CMU SPHINX system called SPHINX-SC. Adaptation in SPHINX-SC is achieved via modification of the mean vectors in its continuous-density codebook, using the ML, EMAP, and LMS-C algorithms. Several methods for automatically identifying good candidates for adaptation are also described. In Section 4.5 results from a set of experiments with computer-generated hidden Markov models are described to provide a better understanding of the SPHINX-SC results.

4.2. Review of Hidden Markov Model Theory

The hidden Markov model[13] is a model of a doubly-stochastic process. It describes the random traversal of a set of states, and the probabilistic emission of observations within each state. The state sequence is assumed to be unobservable, or hidden, and the observations may take on discrete or continuous values. Let the N distinct model states be denoted as $\{S_1, S_2, ..., S_N\}$ as depicted in Figure 4-1 for $N = 3$. State changes can be assumed to occur at times $t = 1, 2, ..., T$, where T is the length of the observation sequence; the actual state at time t will be referred to as q_t. Assuming a first-order Markov chain, the state transition probabilities can be written as

$$a_{ij} = Pr(q_t = S_j | q_{t-1} = S_i) \tag{4.1}$$

subject to

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{4.2}$$

and $a_{ij} \geq 0$. For a discrete-density HMM with the set of M output symbols $\{v_k\}$, the mapping from state j to observation, *i.e.* the j^{th}-state's output distribution, may be written as

$$b_j(k) = Pr(v_k | q_t = S_j) \tag{4.3}$$

For continuous-density HMMs, the state-dependent output distributions are written as

$$b_j(\mathbf{x}) = f(\mathbf{x} | q_t = S_j) \tag{4.4}$$

where $\mathbf{x}$ is the continuous-valued observation vector and $f(\cdot)$ is typically a finite mixture density. The initial state distribution is denoted as $\underline{\pi}$, where

$$\pi_i = Pr(q_1 = S_i) \tag{4.5}$$

An observation sequence $O = O_1, O_2, ..., O_T$ is generated from this model by first choosing an initial state $q_1 = S_i$ according to $\underline{\pi}$. An observation O_1 is generated according to $b_i(\mathbf{x})$, and a new state $q_{t+1} = S_j$ is chosen according to the transition probabilities $\{a_{ij}\}$. This process of generating an observation followed by a transition to another state is repeated until $t = T$.

Consider the following manufacturing model as an example of a hidden Markov process with continuous observations. Imagine there are three machines which produce ball bearings at different rates and with random diameters which obey the distributions in Figure 4-1. The bearings from all three machines are placed, as soon as they are produced, on a a single conveyor which transports them to a sorting and packing facility. From the packing facility's point of view, the machine which produced a particular bearing at time t is the hidden state $q_t = S_i$ and the bearing diameter is the observation O_t taken from the machine's output distribution $b_i(\mathbf{x})$. The probability at which the conveyor feeder switches from machine i to machine j is the transition probability a_{ij} which would be expressed in terms of the relative production rates of the three machines.

---

[13] The notation in this section is consistent with the review of HMM theory in [38].

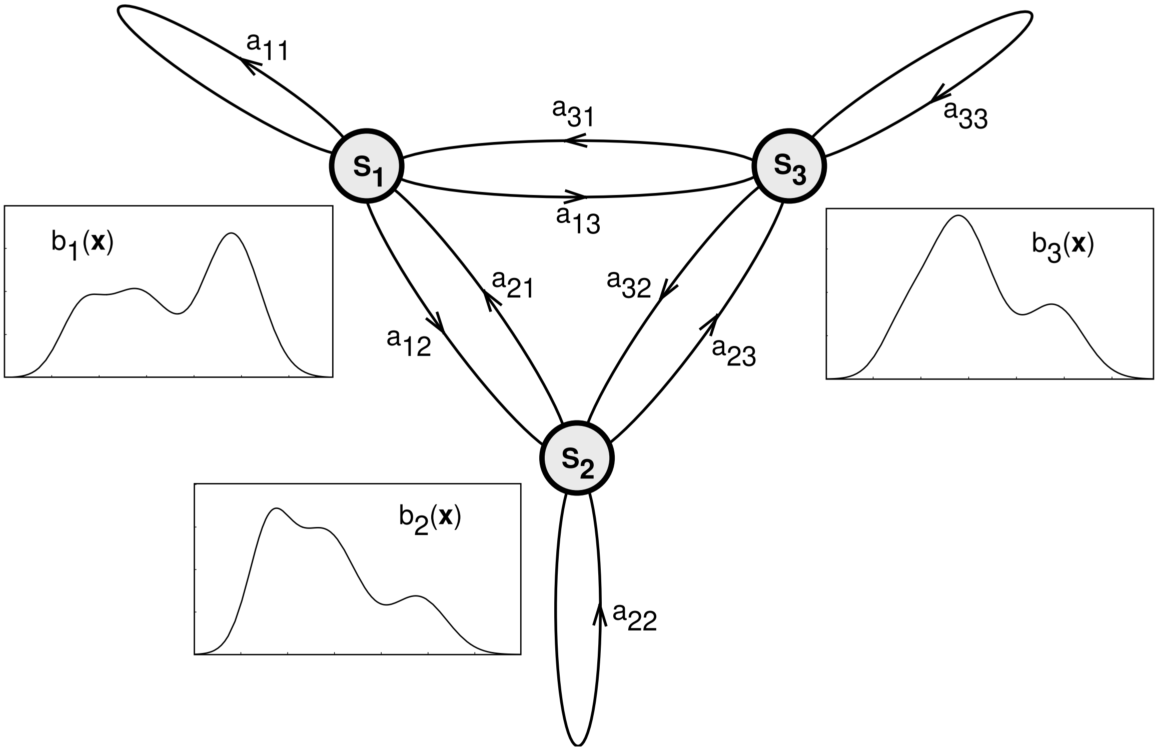

Figure 4-1: Example of a fully-connected, first-order Markov model with three states.

An example of a discrete-output HMM can be constructed as follows. Suppose that in the previous example, the bearings are sorted into bins containing bearings *k*mm ± 0.5mm in diameter for $k = 1, 2, ..., M$. Each state's continuous output distribution can be converted into a discrete density distribution by setting $b_i(v_k)$ equal to the integral of $b_i(\mathbf{x})$ over the width of the k^{th} bin. The observations are then the bins $\{v_k\}$ into which the current bearing is placed. The initial state distribution and transition probabilities would remain unchanged.

Three problems of interest for HMMs are those of evaluation, decoding, and training. Given the model (represented by λ) and an observation sequence O, the evaluation problem is to determine $Pr(O|\lambda)$, the probability that the observation sequence was produced by the model. This can also be viewed as a score of the match between a model and the observations, which is useful when comparing models. The decoding problem is that of determining the most likely state sequence Q given the model and observations. Solutions to this problem can be used for continuous speech recognition, where state sequences determine the word models which are traversed while scoring observations. The training problem describes how to adjust the model parameters λ to maximize $Pr(O|\lambda)$, which is useful for both the initial determination of the model parameters, and for speaker adaptation. Solution of the training problem is most important for the speaker adaptation techniques explored in this chapter. All three problems can be efficiently solved through the application of the forward-backward [39] and Viterbi [40] algorithms, as described in the following section.

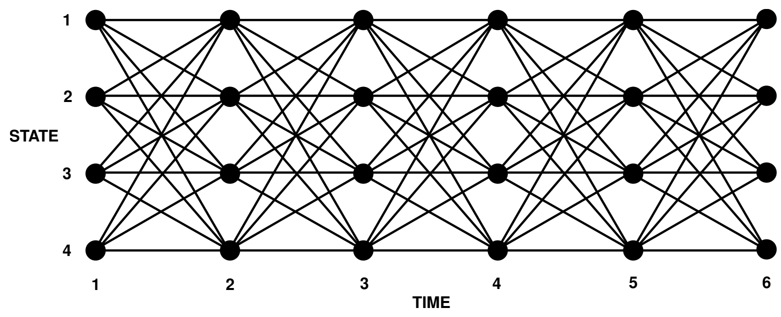

Figure 4-2: Example lattice of 4 HMM states and 6 observations upon which calculations in the forward-backward and Viterbi algorithms are based.

4.2.0.1. The Forward-Backward Algorithm

The forward-backward algorithm defines a set of recursive calculations over a lattice of observations and states (see Figure 4-2), providing an efficient representation for calculation of state sequence and observation probabilities. It provides significant computational savings over direct methods. The forward calculation defines a solution to the evaluation problem, and by the definition of a few additional variables in terms of the forward and backward vectors, it is possible to derive a set of reestimation formulas for the HMM parameters which can be used to solve the training problem.

Define the forward vector as $\underline{\alpha}_t$ where $\alpha_t(j)$ is the probability of the partial observation sequence $O_1, O_2, \cdots, O_t$ and being in state j at time t, given the model, or

$$\alpha_t[j] = P(O_1, O_2, \cdots, O_t, q_t = S_j \mid \lambda) \tag{4.6}$$

The forward vector is initialized as

$$\underline{\alpha}_1 = \mathbf{B}_1 \underline{\pi} \tag{4.7}$$

where $\mathbf{B}_t$ is defined as $diag[b_1[O_t], b_2[O_t], \ldots, b_N[O_t]]$ and $\underline{\pi}$ is the initial state distribution vector as above. By induction, $\alpha_t[j]$ can be written as

$$\alpha_{t+1}[j] = \sum_{i=1}^{N} \alpha_t[i] a_{ij} b_j[O_{t+1}] \tag{4.8}$$

or, in matrix terms

$$\underline{\alpha}_{t+1} = \underline{\alpha}_t^T \mathbf{A} \mathbf{B}_{t+1} \tag{4.9}$$

where $\mathbf{A}$ is the state transition matrix. Equation (4.8) can be interpreted as follows. The i^{th} term of the sum represents the probability of the joint event of producing all the observations up to time t and being in state i at time t, making the transition to state j at t+1, and producing the observation O_{t+1} at time t+1. Summing these products over all states i at time t gives the probability of producing all the observations up to time t+1 and being in state j at t+1 independent of the state at time t, which is the definition of α_{t+1}[j].

Similar to the forward vector, define the backward vector as $\underline{\beta}_t$, where the elements $\{\beta_t[i]\}$ are the probability of the partial observation sequence from t+1 to the end, given state i at t and the model, or

$$\beta_t[i] = P(O_{t+1}, O_{t+2}, \cdots, O_T| , q_t = S_i | \lambda) \tag{4.10}$$

Elements of the backward vector are arbitrarily initialized as unity,

$$\beta_T[i] = 1, \; 1 \le i \le N \tag{4.11}$$

and a recursion relation for $\underline{\beta}_t$ is

$$\underline{\beta}_t = \mathbf{AB}_{t+1}\underline{\beta}_{t+1} \tag{4.12}$$

for $t = T-1, T-2, \ldots, 1$. Looking at the i^{th} element of $\underline{\beta}_t$,

$$\beta_t[i] = \sum_{j=1}^{N} a_{ij} b_j[O_{t+1}]\beta_{t+1}[j] \tag{4.13}$$

Each summand in Eqn. (4.13) is the probability of a transition from state i at t to state j at t+1, emission of O_{t+1} from state j, and the production of the remainder of the observations. Summing these products over all states j at time t+1 gives the probability of producing the observations from time t+1 onward given state i at time t, *independent* of the sequence of following states.

A solution to the evaluation problem, determination of $Pr(O|\lambda)$, is obtained directly from the forward vector at time T. By definition, $\alpha_T[i]$ is the probability of the entire observation sequence and ending in state i. Summing these values over all states i gives the probability of the observation sequence and ending in any state, which is the quantity of interest:

$$Pr(O|\lambda) = \sum_{i=1}^{N} \alpha_T[i] \tag{4.14}$$

To solve the training problem, define an intermediate vector as $\underline{\gamma}_t$ where the element $\gamma_t[i]$ represents the probability of being in state i at time t given the model and observations, or

$$\gamma_t[i] = Pr(q_t = S_i | O, \lambda) \tag{4.15}$$

In terms of the forward-backward variables, this can be rewritten as

$$\gamma_t[i] = \frac{\alpha_t[i]\beta_t[i]}{\sum_{k=1}^{N} \alpha_t[k]\beta_t[k]} = \frac{\underline{\alpha}_t \otimes \underline{\beta}_t}{\underline{\alpha}_t^T \underline{\beta}_t} \tag{4.16}$$

where $\otimes$ represents the Schur product. The numerator of (4.16) is the probability of the observations up to time t and being in state i, times the probability of the observations from time t+1 to the end given state i at time t, all given the model parameters, or $Pr(q_t = S_i, O | \lambda)$. The denominator is actually $Pr(O|\lambda)$, which normalizes the expression to ensure that the elements of $\underline{\gamma}_t$ sum to 1.

Additionally define the intermediate matrix ξ_t where the element $\xi_t[i,j]$ is the probability of a transition from state i at time t to state j at t+1, given the model and observations, or

$$\xi_t[i,j] = Pr(q_t = S_i,\, q_{t+1} = S_j | O, \lambda) \tag{4.17}$$

This quantity can be expressed in terms of forward and backward variables as

$$\xi_t[i,j] = \frac{\alpha_t[i] a_{ij} b_j[O_{t+1}] \beta_{t+1}[j]}{\sum_{k=1}^{N} \alpha_t[k] \beta_t[k]} = \frac{\underline{\alpha}_t \underline{\beta}_{t+1}^T \otimes \mathbf{AB}_{t+1}}{\underline{\alpha}_t^T \underline{\beta}_t} \tag{4.18}$$

The numerator of (4.18) is $Pr(q_t = S_i, q_{t+1} = S_j, O|\lambda)$, the denominator is $Pr(O|\lambda)$, so by Bayes' theorem this is the desired probability.

Summing $\gamma_t[i]$ over time t yields the expected number of times the model is in state i or the expected number of times a transition out of state i is taken. Similarly, summing $\xi_t[i,j]$ over time yields the expected number of times the transition from state i to state j is taken. Note that these expectations are time averages, or frequencies of occurrence of the particular events. A set of HMM parameter reestimation formulas, defined in terms of these time-averaged frequencies, can be written as

$$\bar{\pi}_i = \gamma_1[i] \qquad \bar{a}_{ij} = \frac{\sum_{t=1}^{T} \xi_t[i,j]}{\sum_{t=1}^{T} \gamma_t[i]} \tag{4.19}$$

and

$$\bar{b}_j[k] = \frac{\sum_{t=1 \text{ and } O_t = v_k}^{T} \gamma_t[j]}{\sum_{t=1}^{T} \gamma_t[j]}$$

The reestimate $\bar{\pi}_i$ is the expected number of times in state i at time $t = 1$, $\bar{a}_{ij}$ is the expected number of transitions from S_i to S_j normalized by the expected number of transitions out of S_i, and $\bar{b}_j[k]$ is the expected number of times in S_j when v_k is emitted normalized by the number of times in S_j. If the current model parameters λ are iteratively replaced by those in (4.19) and the training process repeated, then the sequence of observation probabilities $\{Pr(O|\lambda_n)\}$ will increase until a local maximum is reached [39]. The HMM parameter estimates at these local maxima are maximum likelihood estimates since they maximize $\{Pr(O|\lambda)\}$ with respect to λ.

4.2.0.2. The Viterbi Algorithm

Although the solution of the decoding problem is not an issue for the adaptation work considered here, it is an integral part of the recognition algorithm under study and so for completeness it is outlined briefly. The Viterbi algorithm is a dynamic programming method which is similar in form to the forward computation in (4.8) except the summation is replaced by maximization. Define the best score and best state vectors as $\underline{\delta}_t$ and $\underline{\psi}_t$, respectively. The elements $\delta_t[i]$ represent the highest scoring single path, ending in state i, which produced the observation sequence

$$\delta_t[i] = \max_{\{q_k\}_{t-1}} Pr(q_1 q_2 \cdots q_t = i, O_1 O_2 \cdots O_t | \lambda) \tag{4.20}$$

where the maximization is over the state sequence up to time t-1. These elements are initialized as

$$\delta_1[i] = \pi_i b_j[O_1] \tag{4.21}$$

and again by induction

$$\delta_{t+1}[j] = max_i[\delta_t[i] a_{ij}] b_j[O_{t+1}] \tag{4.22}$$

The element $\psi_t[j]$ holds the index of the best state from the previous time instant (t-1) which passes through the current state j, or

$$\psi_t[j] = argmax_i[\delta_{t-1}[i] a_{ij}] \tag{4.23}$$

Upon completion of the calculation of (4.22) and (4.23) across the lattice of observations and states, the optimal state sequence Q^* may be decoded by backtracking through the ψ array

$$q_t^* = \psi_{t+1}[q_{t+1}^*] \tag{4.24}$$

where $t = T-1, T-2, \ldots, 1$. Equation (4.24) means that the previous best state is specified by the value of the ψ array at the current best state. Application of this algorithm to speech recognition will be discussed in the next section.

4.3. Hidden Markov Models for Speech Recognition

When modeling non-stationary signals such as speech, the general, fully-connected hidden Markov model (as in Figure 4-1) is modified to incorporate a number of constraints. The most basic modification is the assumption of a left-right model. In the fully-connected HMM, any state can be reached from any other state in one step, so the state transition matrix elements are strictly positive. In the left-right model (see Figure 4-3), the state index is not allowed to decrease with time, resulting in an upper-triangular state transition matrix. As a result, the state sequence must begin in state 1 and end in state N, which places constraints on the initial state probability vector $\underline{\pi}$ and requires $a_{NN} = 1$. Additionally, when modeling continuous speech, changes in the state sequence are often limited to a small number of steps, *i.e.* the state indices may change at most by some small number s. For example, if word and grammar models were constructed from phoneme models as in Figure 4-4, from the phoneme model it is obvious the maximum number of states which may be skipped in a single transition is three. Limiting this state jump size has the effect of making the state transition matrix banded in addition to being upper triangular; the number of diagonals above the main diagonal is equal to the maximum jump size.

Another consequence of the left-right model is that one training sequence is not sufficient to train the model: there can be only a very small number of observations from within a given state before transition to another state. To obtain reliable estimates of all model parameters, multiple training sequences are necessary. Fortunately, extension of the training algorithm to multiple training

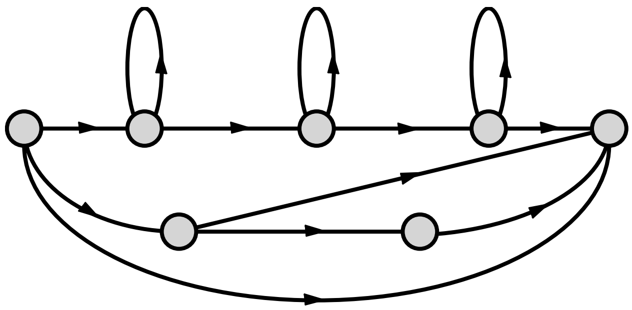

Figure 4-3: A left-right HMM with seven states.

sequences is straightforward. Since the reestimation formulas in (4.19) are written in terms of frequencies of certain events, the frequencies from multiple training sequences can be added together to get the total frequency from all sequences. So given some number of training sequences K, the only change to the reestimation formulas is an additional sum over k in the numerators and denominators.

When modeling restricted continuous speech tasks, that is tasks defined by fixed grammars, a large left-right model is constructed from phoneme and word models. Phoneme models are initialized with parameters derived from a large, task-independent database, and concatenated to form task-specific word models which may contain parallel paths to represent multiple pronunciations. The word models are then concatenated to form a sentence model or a finite state grammar network which represents the recognition task, as in Figure 4-4. Model parameters in the resulting network are updated using task-specific training sentences to refine the task-independent parameters.

During recognition, the Viterbi algorithm is used to match the observation sequence with the grammar network. The states' output probabilities and transition probabilities are accumulated in the Viterbi variable $\delta_t[i]$ as an utterance is processed, along with state indices in the ψ array. After completing calculation of these variables, the most likely state sequence is decoded. The word models which are traversed during the backtracking step is the hypothesized word sequence.

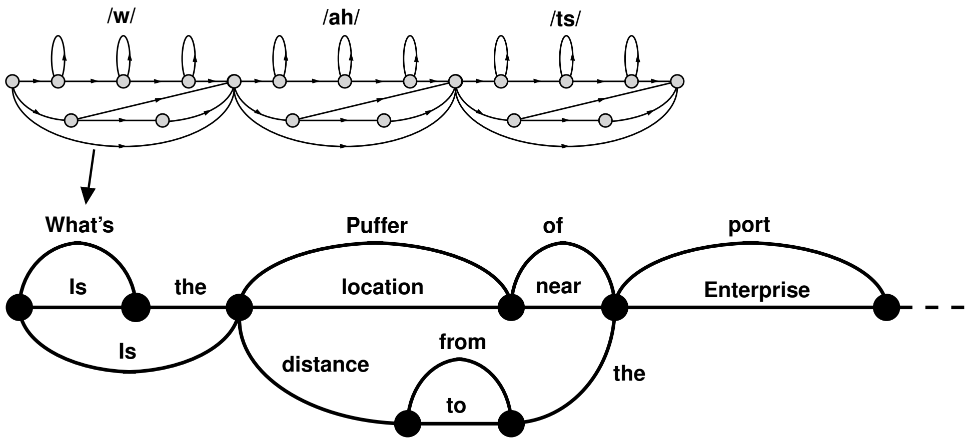

Figure 4-4: A finite-state grammar network for a continuous recognition task.

A fundamental difference between HMMs as applied to speech recognition is the representation of the speech signal, or equivalently, the manner in which probabilities are assigned to observations within a state. As was seen earlier in the manufacturing example, the observations may be modeled as either discrete or continuous, with corresponding output probability density functions $b_j[O_t]$. Discrete density HMMs (DDHMMs) represent the data from a speech frame as a symbol from a finite alphabet $\{v_k\}$. A vector quantization (VQ) codebook is used to perform the mapping between the continuous observation vector and the alphabet symbols. Continuous density HMMs (CDHMMs) typically use finite mixture densities to characterize the multi-dimensional output densities. A modification of the CDHMM, called the semi-continuous HMM (SCHMM), assumes that all component densities from the CDHMM mixtures are tied (*i.e.* all component densities for each model state are identical), and only the mixing coefficients vary from state to state. Details of these various representations follow.

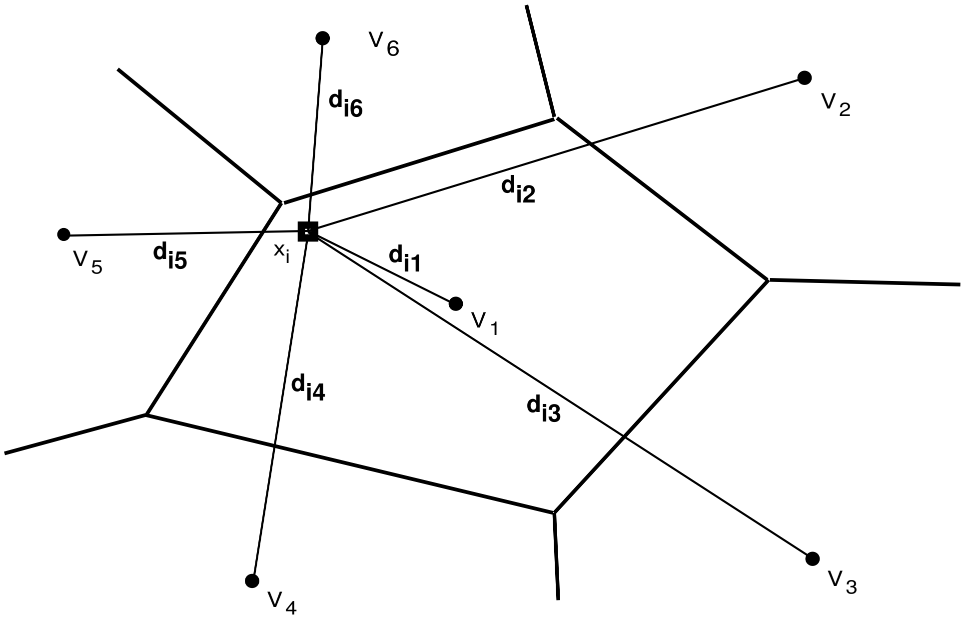

Figure 4-5: Vector quantization of a continuous-valued observations. All observations $\mathbf{x}_i$ are replaced by the closest prototype vector v_k.

Linear-predictive coding (LPC) or LPC cepstral coefficients, derived from a windowed portion of the speech signal, are commonly used as the representation of the observations $\{\mathbf{x}_i\}$. Prototype vectors for a VQ codebook are derived via cluster analysis of a large body of observation vectors. The number of prototypes (M) is chosen such that the training and storage requirements are minimized given a reasonable level of distortion or quantization noise. Partitioning the observation vector and using multiple codebooks has been shown to significantly reduce quantization noise [6]. Discrete HMMs use a VQ codebook to map the observation $\mathbf{x}_i$ into one of the prototype indexes k as

$$k = argmin_j[d(\mathbf{x}_i, v_j)] \qquad (4.25)$$

where $d(\cdot)$ is some suitably defined similarity measure. The sorting process in the second manufacturing example was an example of vector quantization in a discrete HMM, and VQ is illustrated in two-dimensions in Figure 4-5. With VQ and discrete output densities, the discrete HMM can easily model non-parametric densities, and computation of the output probability $b_j[O_t]$ is an efficient table lookup. There is an inherent loss of information associated with VQ, however, which may lower recognition rates.

To overcome distortions and loss of information caused by the vector quantization process, CDHMMs use continuous output density functions which obviate the need for VQ. While the CDHMMs directly model speech parameters, a finite mixture density must be estimated for each state in the model. The EM (estimation and maximization) algorithm [41], [42] specifies a general reestimation procedure for finite mixture densities, and a number of researchers have expressed the EM formulations in the context of specific HMMs [38], [43], [16]. Assuming M-component gaussian mixtures of the form

$$b_j[O_t] = \sum_{m=1}^{M} c_{jm} N(O_t, \mu_{jm}, \Sigma_{jm}) \tag{4.26}$$

the mixture parameter reestimation formulas can be expressed in terms of the forward-backward variables as ([44])

$$\bar{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t[j,k]}{\sum_{t=1}^{T} \sum_{m=1}^{M} \gamma_t[j,m]} \tag{4.27}$$

$$\bar{\mu}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t[j,k] O_t}{\sum_{t=1}^{T} \gamma_t[j,k]}$$

$$\bar{\Sigma}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t[j,k](O_t - \mu_{jk})(O_t - \mu_{jk})^T}{\sum_{t=1}^{T} \gamma_t[j,k]}$$

where

$$\gamma_t[j,k] = \frac{\alpha_t[j]\beta_t[j]}{\sum_{i=1}^{N} \alpha_t[i]\beta_t[i]} \frac{c_{jk} N(O_t, \mu_{jk}, \Sigma_{jk})}{\sum_{m=1}^{M} c_{jm} N(O_t, \mu_{jm}, \Sigma_{jm})} \tag{4.28}$$

is the probability of being in state j at time t and having the k^{th} component produce the observation. The formula for $\bar{\mu}_{jk}$ is the expected value of the contribution of the k^{th} component to the the observation vector, which is the number of times the system is in state j with the k^{th} component producing the observation, weighted by the observation, over the number of times the system is in state j. Interpretation of the reestimation formula for $\bar{\Sigma}_{jk}$ is essentially the same.

An important issue in CDHMM implementation is whether to use diagonal or full covariance matrices for the normal mixture densities. Full covariance matrices make fewer assumptions about the data and therefore model the distribution more accurately. This requires estimation of many more parameters, however, and additional training data and time. Brown [45] showed that error rates for a diagonal-covariance HMM were about twice that of a full covariance system. Huang [20], however, has reported marginally lower recognition rates with a full covariance Gaussian than with the diagonal HMM, albeit with additional constraints imposed by tied parameters.

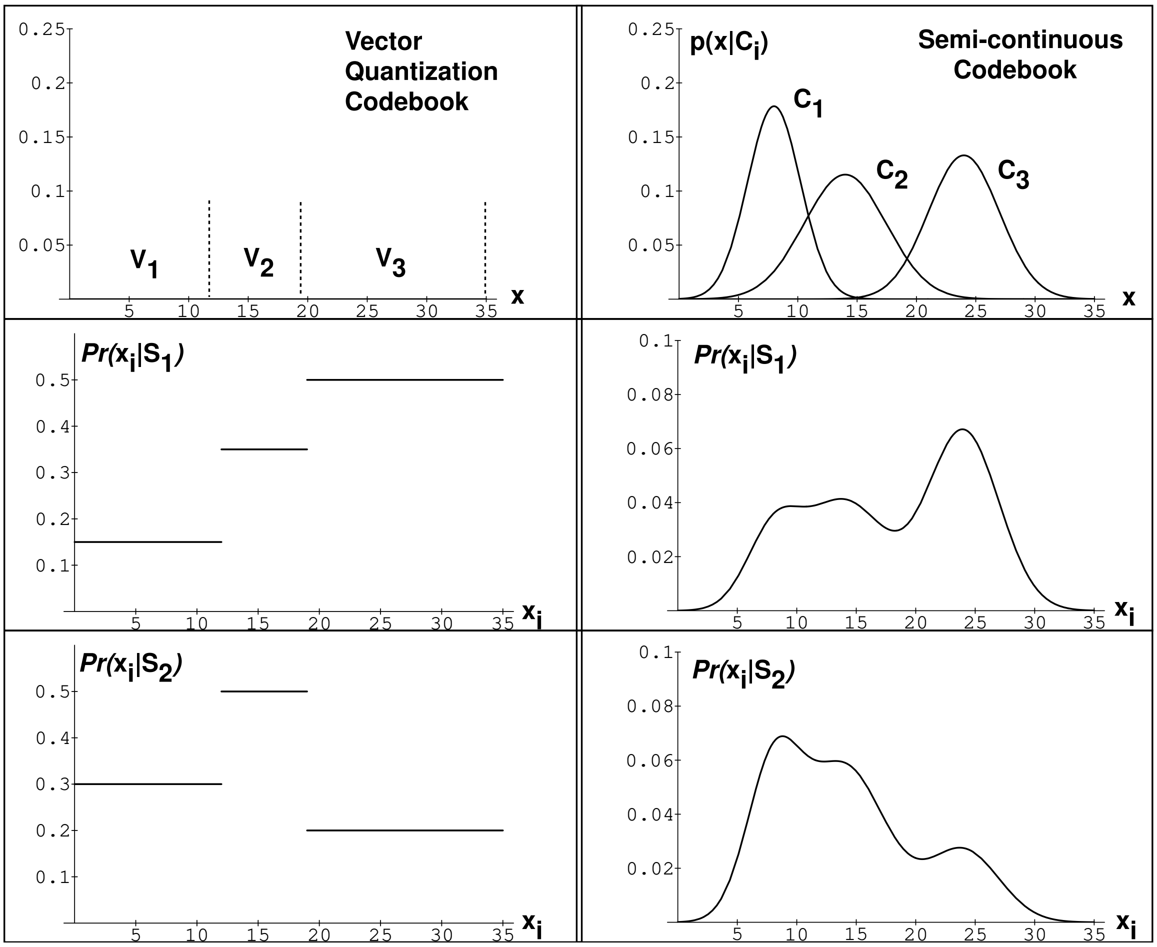

Figure 4-6: Output probability assignment in discrete and semi-continuous HMMs.

More recently, the discrete HMM has been extended to continuous mixture observation densities by replacing each of the VQ prototype vectors with a multivariate gaussian density and using the discrete output pdf as the mixture coefficients. This can also be viewed as a CDHMM in which the mixture components across all states are assumed to be tied. This semi-continuous HMM (SCHMM) architecture maintains the modeling ability of the CDHMM, reducing the DDHMM VQ distortion, but reduces the CDHMM computational load and number of free parameters. The SCHMM component density parameters are more robust since data from all states can be used in their estimation. An example of how the semi-continuous model can improve output probability modeling is shown in Figure 4-6 for $M=3$. The DDHMM prototype vectors $\{v_j\}$ define regions in the **x**-space such that any observation $\mathbf{x}_i$ falling in that region has the probability $Pr(\mathbf{x}_i | S_k) = b_k(v_j)$ assigned to it given state j, as illustrated in the two lower left graphs. The semi-continuous model uses the prototype vectors as the means of component densities, and the state output densities are built from this continuous codebook. As the lower-right graphs indicate, the semi-continuous output densities do not have the discontinuities of the DDHMM, and can more closely approximate the real data distributions which is crucial for good recognition performance.

Huang [43] has given expressions of the EM algorithm for the unified reestimation of the continuous codebook and the other SCHMM parameters. The mean and covariance formulas are

$$\bar{\mu}_j = \frac{\sum_{t=1}^{T} \zeta_t[j] O_t}{\sum_{t=1}^{T} \zeta_t[j]} \tag{4.29}$$

and

$$\bar{\Sigma}_j = \frac{\sum_{t=1}^{T} \zeta_t[j](O_t - \mu_j)(O_t - \mu_j)^T}{\sum_{t=1}^{T} \gamma_t[j]}$$

where $\zeta_t[j]$ is the probability of the j^{th} mixture component producing the observation at time t, which may be expressed in terms of the forward-backward variables as

$$\zeta_t[j] = \sum_{k=1}^{N} \sum_{i=1}^{N} \frac{\alpha_t[i] a_{ik} b_k[j] f(O_{t+1}|j) \beta_{t+1}[k]}{Pr(O|\lambda)} \tag{4.30}$$

where $b_k[j]$ is the j^{th}-element in the output distribution of the DDHMM from which the SCHMM is derived, and $f(O_{t+1}|j)$ is the probability of the observation O_{t+1} given the j^{th}-mixture component.

Adaptation can be performed within continuous or semi-continuous HMMs by reestimating a set of speaker-independent parameters given speaker-specific observation sequences. In the remainder of this chapter, speaker adaptation through reestimation of codebook mean vectors is investigated using a semi-continuous version of SPHINX, called SPHINX-SC. With statistics derived from a multiple-speaker database, adaptation via the LMS-C or EMAP algorithms can determine whether correlation information can improve the maximum likelihood reestimates $\{\bar{\mu}_j\}$ and increase recognition rates.

4.4. Adaptation in SPHINX-SC

The original CMU SPHINX system [6] demonstrated a speaker-independent word accuracy of better than 94 percent on the 997-word DARPA resource management task. This task, which has a perplexity of about 60, is defined by a grammar which allows the formation of, for example, queries about naval resources or display control. Performance of the SPHINX system has subsequently been improved by Huang, *et al.* [18]. An error rate of 4.3 percent is currently being reported.

The SPHINX system used a hidden Markov model with discrete state output probability densities and a vector quantization codebook. Huang [20] has investigated an approach in which VQ codebook entries become the mean vectors of mixture density components, and each state's discrete output probabilities become the mixture coefficients. This semi-continuous version of SPHINX was modified to determine how effective the EMAP and LMS-C estimation techniques were in reducing SPHINX-SC word error rates through codebook adaptation. Details of the SPHINX-SC system follow.

Speech data for training and evaluation is digitized with a 16KHz sampling rate, and pre-emphasized using a filter with transfer function $H(z)=1-0.97z^{-1}$. 14$^{\text{th}}$-order LPC analysis is performed on the speech data using a 20-msec Hamming window and a 10-msec increment. Twelve bilinear-transformed LPC cepstral coefficients are derived from these LPC coefficients. Finally, each speech frame is represented by three feature vectors consisting of the cepstral coefficients, first-differenced cepstral coefficients, and the power of each of these cepstral vectors. SPHINX-SC uses a set of three codebooks, one for each of the cepstral, differenced cepstral, and power cepstral data vectors, with 256 codewords each. Each codeword has an associated mean vector, covariance matrix, and determinant for use in output probability calculations. The state output density mixture coefficients were derived from the discrete output densities in SPHINX.

The complete SPHINX-SC experimental setup is depicted in Figure 4-7. The semi-continuous HMM phone models were trained in a two step process. A set of 48 context-independent models were trained on a large body of speech data and made available to the author. Given task-specific data, these models were refined to produce a larger set (1147 for SPHINX-SC) of context-dependent phone models, or triphones. Both context-dependent and context-independent training consist of multiple iterations of the forward-backward algorithm and reestimation. Within each iteration, an orthographic transcript is used to build an HMM for each training utterance. For each word in the transcription, an entry in the task's lexicon specifies the HMM phone model sequence which represents that word, *e.g.* *DECREASE => D IY K R IY S*. The corresponding phone models are concatenated to form a network specific to the current sentence. A forced recognition is then performed on the current utterance using the single-sentence HMM, and counts are accumulated. After each pass through the training utterances, these accumulated counts are used to update the model parameters as described in Sections 4.2 and 4.3.

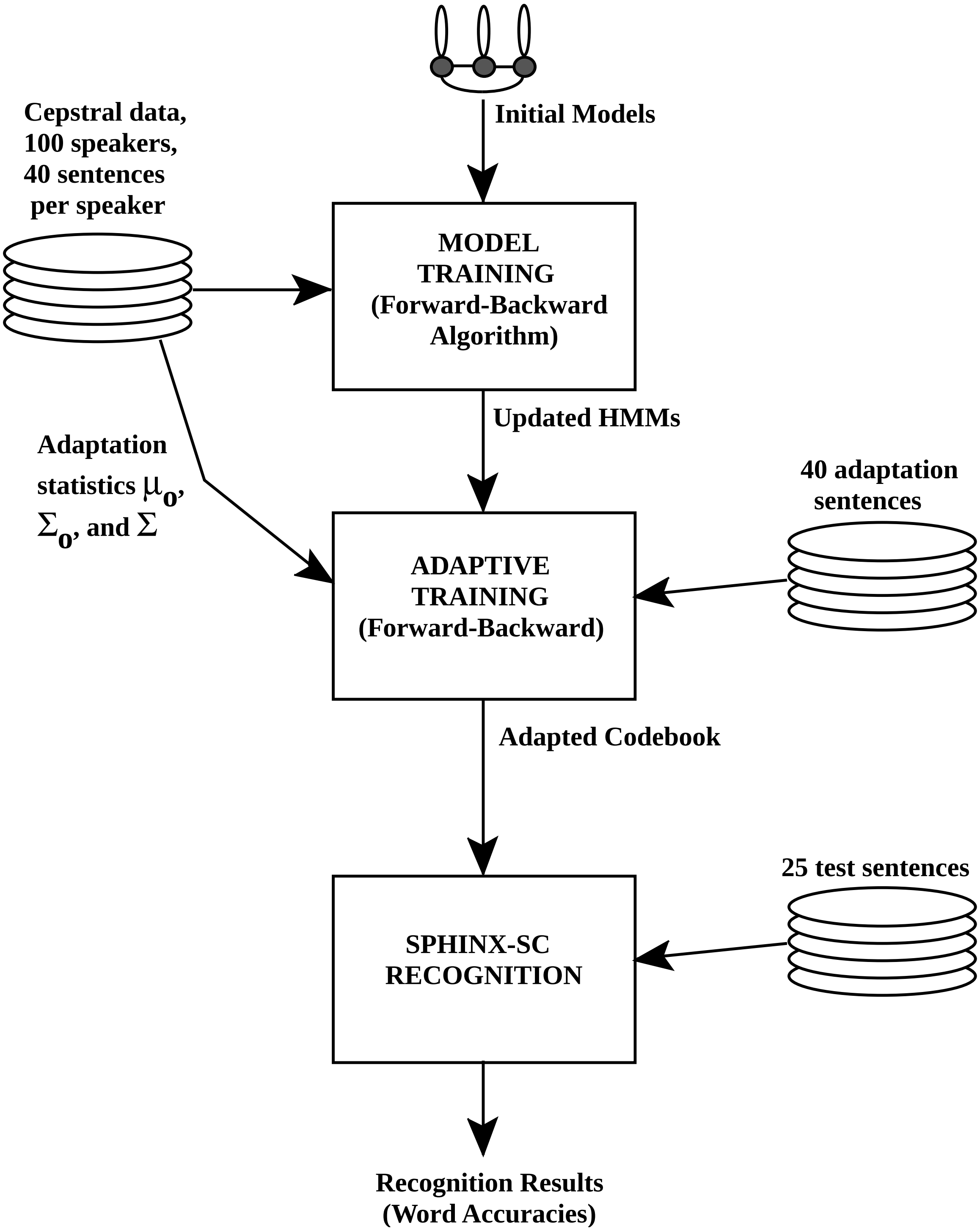

Figure 4-7: SPHINX-SC experimental setup.

Training data for the generation of models and adaptation statistics were 40 sentences from approximately 100 speakers in the TIRM [46] database. The adaptation data were 40 sentences from 11 different speakers in the TIRM database. The evaluation test data consisted of another 25 sentences from these 11 speakers. The cepstral and differenced cepstral vectors are 12-dimensional, and power cepstral vectors have 2 elements. Since the dimension of the *a priori* adaptation statistics is equal to the product of the number of classes and dimensions, at least $12 \cdot 256 = 3072$ speakers in the training set would have been required to estimate these parameters. Because of the limited availability of training data it was necessary to reduce this estimation problem to a set of parallel problems by considering only the correlation between a given codeword's features.

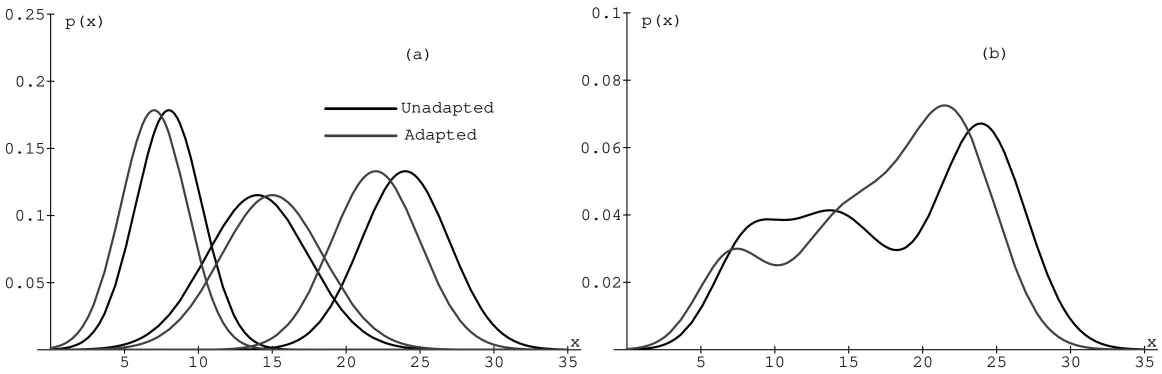

Figure 4-8: Adapted and unadapted (a) codebooks and (b) state output probability density functions for a 3-component example.

An early version of a semi-continuous form of SPHINX[14] was modified to allow ML, EMAP, or LMS-C adaptation of codebook mean vectors after observation of a speaker's 40 adaptation sentences. LMS-C and EMAP adaptation use the maximum likelihood reestimate $\bar{\mu}$ from (4.19) as $\mathbf{a}_j$, and the weighting terms in the $\mathbf{N}$ matrix are the counts $n_j = \sum_{t=1}^{T} \zeta_t[j]$ from (4.30). Codebook adaptation is performed simply by iterating the forward-backward algorithm over the adaptation sentences; only the mean vectors are updated after each F-B iteration. Figure 4-8 illustrates the effect this form of adaptation can have on the semi-continuous output densities.

Evaluations were based on comparisons of word error rates between systems using speaker independent codebooks and adapted codebooks, as shown in Tables 4-1 through 4-4. The tables list the number of correctly-recognized words, the number of insertions or extra words, and the number of reference words or the number actually spoken. Also listed are the percent correct, which is the ratio of correct to reference words, and the word error rate. The word error rate penalizes a system for insertions, and so is defined as

$$Error\,Rate = \frac{Reference - (Correct - Insertions)}{Reference} \cdot 100\% \tag{4.31}$$

The majority of the recognition errors were for the short function words such as *is, the, if, at, in, on, for* and *and*. These common words are often unstressed and poorly articulated or merged with

---

[14]There are a number of differences between the SPHINX-SC system used in the present evaluation and other semi-continuous versions of SPHINX described in the literature.

Speaker	Percent Correct	Correct	Insertions	Reference	Error Rate
bef03m	94.2%	213	0	236	5.8%
cmr02f	95.2%	220	8	231	8.2%
dms04f	98.9%	175	1	177	1.7%
dtb03m	97.3%	220	0	226	2.7%
dtd05	95.3%	222	0	233	4.7%
ers07m	95.3%	202	0	212	4.7%
hxs06	92.3%	205	10	222	12.2%
jws04m	93.7%	208	0	222	6.3%
pgh01m	96.1%	196	0	204	3.9%
rkm05m	88.5%	184	8	208	15.4%
tab07	96.1%	171	1	178	4.5%
TOTALS	94.74%	2216	28	2339	6.46%

Table 4-1: Summary of SPHINX-SC results with the speaker-independent (unadapted) semi-continuous codebook.

Speaker	Percent Correct	Correct	Insertions	Reference	Error Rate	Change in Error Rate
bef03m	93.8%	212	0	236	6.2%	+6.9%
cmr02f	95.7%	221	6	231	6.9%	-15.9%
dms04f	99.4%	176	1	177	1.1%	-35.3%
dtb03m	98.7%	223	0	226	1.3%	-51.9%
dtd05	95.3%	222	3	233	6.0%	+27.6%
ers07m	95.8%	203	0	212	4.2%	-10.6%
hxs06	91.0%	202	14	222	15.3%	+23.8%
jws04m	93.7%	208	0	222	6.3%	0.0%
pgh01m	96.1%	196	1	204	4.4%	+12.8%
rkm05m	91.8%	191	10	208	13.0%	-15.6%
tab07	97.2%	173	1	178	3.4%	-24.4%
TOTAL	95.21%	2227	36	2339	6.33%	--
CHANGE	-8.94%	+11	+8	--	-2.01%	--

Table 4-2: Summary of SPHINX-SC results with the ML adapted semi-continuous codebook.

other words in a sentence. Although significant reductions in error rate were observed for some speakers, the aggregate results were much lower, on the order of 2 to 4 percent. Standard statistical analysis [Gillick89] of the experiments reported in Tables 4-1 through 4-4 showed that while the recognition scores for the speakers after adaptation were significantly different from the error rates

Speaker	Percent Correct	Correct	Insertions	Reference	Error Rate	Change in Error Rate
bef03m	94.2%	213	0	236	5.8%	0.0%
cmr02f	95.7%	221	6	231	6.9%	-15.9%
dms04f	98.9%	175	1	177	1.7%	0.0%
dtb03m	97.8%	221	0	226	2.2%	-18.5%
dtd05	95.3%	222	1	233	5.2%	+10.6%
ers07m	95.8%	203	0	212	4.2%	-10.6%
hxs06	91.9%	204	13	222	14.0%	+14.8%
jws04m	93.7%	208	0	222	6.3%	0.0%
pgh01m	95.6%	195	1	204	4.9%	+25.6%
rkm05m	89.9%	187	6	208	13.0%	-15.6%
tab07	97.2%	173	1	178	3.4%	-24.4%
TOTALS	95.00%	2222	29	2339	6.24%	--
CHANGE	-4.94%	+6	+1	--	-3.41%	--

Table 4-3: Summary of SPHINX-SC results with the EMAP adapted semi-continuous codebook.

Speaker	Percent Correct	Correct	Insertions	Reference	Error Rate	Change in Error Rate
bef03m	94.2%	213	0	236	5.8%	0.0%
cmr02f	95.7%	221	6	231	6.9%	-15.9%
dms04f	98.9%	175	1	177	1.7%	0.0%
dtb03m	97.8%	221	0	226	2.2%	-18.5%
dtd05	95.3%	222	1	233	5.2%	+10.6%
ers07m	95.8%	203	0	212	4.2%	-10.6%
hxs06	91.9%	204	12	222	13.5%	+10.6%
jws04m	93.7%	208	0	222	6.3%	0.0%
pgh01m	95.6%	195	1	204	4.9%	+25.6%
rkm05m	89.4%	186	7	208	13.9%	-9.7%
tab07	96.6%	172	1	178	3.9%	-13.3%
TOTAL	94.91%	2220	29	2339	6.33%	--
CHANGE	-3.23%	+4	+1	--	-2.01%	--

Table 4-4: Summary of SPHINX-SC results with the LMS-C adapted semi-continuous codebook (Nc=30).

without adaptation with a confidence of 90 to 95 percent, the adapted systems were not significantly different from each other.

A number of variations of the adaptation paradigm described above were implemented in an effort to improve the adaptation results. They included multiple forward-backward iterations during adaptation, adaptation after each training sentence instead of after all sentences, and a threshold on the number of observations necessary before adapting a given codeword. The changes in recognition performance due to these variations and combinations of them were insignificant. As a result, given that the error rates for some individuals significantly decreased after adaptation, the remaining work with SPHINX-SC focused on determining methods of automatically identifying those speakers who benefit most from adaptation.

4.4.1. Identification of Adaptation Candidates

In each experiment, about half of the speakers showed improvement due to adaptation while the remainder showed no change or an increased error rate. Results may be improved if an automatic method of identifying speakers which are good candidates for adaptation can be found. Three metrics were investigated. The first was the change in the sum of the pairwise Euclidean distances between all codewords for the adapted and unadapted codebooks. Reasoning that if the normalized distance between the component densities in the VQ codebook increases the mixture components become less confusable, the speaker-adapted codebook was chosen only when adaptation produced an increase in this distance metric. Experimental results when this selection procedure was applied are given in Table 4-5. Listed are the number of adaptation candidates (out of 11) identified by this method, the number of those identified which actually showed improvement, the number of improved speakers which were missed, and the adapted and unadapted error rates when only the adapted speakers are included. For those speakers selected by this metric, the error rate was reduced on average by 15%.

Adaptation Type	Number Selected	Number Improved	Number Missed	Adapted Error Rate	Unadapted Error Rate	Percent Reduction
ML	3	3	3	4.99%	5.96%	16.2%
EMAP	4	3	2	4.26%	5.01%	15.0%
LMS-C	3	3	2	5.15%	5.96%	13.6%

Table 4-5: Recognition results for automatically selected speakers.

A second selection method was based on recognition rates. Of the 40 adaptation sentences per speaker, 35 were used for adaptive training. The recognition system was then run on the remaining 5 adaptation sentences using both the adapted and unadapted codebooks. If the error rate from the 5 sentences was lower with the adapted codebook, the speaker was declared to be a candidate for adaptation. In these cases, the adapted codebook was used in subsequent tests on the 25 test sentences. Unfortunately, there proved to be little correlation between the error rates from the 5 adaptation sentences and the 25 test sentences, and the overall error rate was unchanged.

The final selection method was the magnitude of the deviation of the adapted mean from the *a priori* mean μ_o. Although one might expect to obtain the largest gains when this deviation is largest, there was no definite correlation between this measure and the change in recognition rate after adaptation. The average dogmatism of the cepstral and difference cepstral codebooks was approximately 2.0 and 3.0, respectively. The variation around μ_o was therefore quite small, on the order of 10% of the magnitude of μ_o.

Adaptation in SPHINX-SC via any of the estimation procedures from Chapter 2 showed only a 2-3% reduction of the word error rate. If all speakers who are improved due to adaptation could be correctly identified, the net reduction would be on the order of 6%. To investigate the conditions necessary for successful application of the EMAP and LMS-C procedures in a semi-continuous HMM, experiments with computer-generated models and data were performed. These experiments are described in the remainder of the chapter.

4.5. HMM Experiments with Computer-Generated Data

To better understand why the SPHINX-SC experimental results using data from actual speakers were not more encouraging, a number of simulations were conducted using the 3-state, 3-component-mixture SCHMM shown in Figure 4-1. Experiments with this model allowed for investigation of adaptation performance with various values for dogmatism, correlation, and observation sequence length. Specific parameters and values considered were: dogmatism δ = (0.5, 1, 2, 4), correlation ρ = (0.1, 0.5, 0.9), and observation sequence length T = (32, 64, 128, 256, 512, 1024). The experiments were conducted as follows.

For each set of conditions, the experimental model was defined by the parameters $(\mathbf{A},\mathbf{B},\pi)$ and the *a priori* semi-continuous codebook (specified in Appendix D), and the desired data properties (δ, ρ) which were used to determine Σ_o. Results for each experiment were averaged over 25 trials. For each trial a true mean vector was generated according to $N(\mu_o,\Sigma_o)$. Using the elements of this mean vector as the codebook's mean values, an observation sequence of length T was generated according to the model parameters. This sequence was used to find an estimate of the trial (true) mean using ML, EMAP, or LMS-C procedures. Each mean estimate was initialized with the *a priori* mean μ_o, and then reestimated by repeated iterations of the forward-backward algorithm followed by EMAP or LMS-C updating when appropriate.[15] Ten iterations were conducted, and the estimated mean was replaced by the reestimate after each iteration. Only the codebook means were updated after each iteration. The mean-square error between the true (trial) mean and the reestimated mean for each

[15]The reestimate produced by the forward-backward algorithm is the ML estimate. No further processing was therefore necessary when performing ML adaptation.

iteration was recorded and averaged over each trials. The probability of the observation sequence given the model was also computed.

In every experiment, the results from ML, EMAP, and LMS-C adaptation were nearly identical, much like the results observed using real data. Figure 4-9 shows a family of curves which are typical of the results of these experiments. The curves in 4-9 represent the maximum likelihood estimate's MSE versus the number of iterations of the forward-backward algorithm for various observation lengths and $\delta = 0.5$ and $\rho = 0.1$. Results from most other experiments, and from EMAP and LMS-C adaptation, were similar to Figure 4-9. In general, the mean-square error decreased with each iteration of the forward-backward algorithm, and for each observation sequence length the estimates converged after approximately 6 iterations. Also, as the length of the observation sequence increased, the asymptotic MSE decreased since more data was available for adaptation.

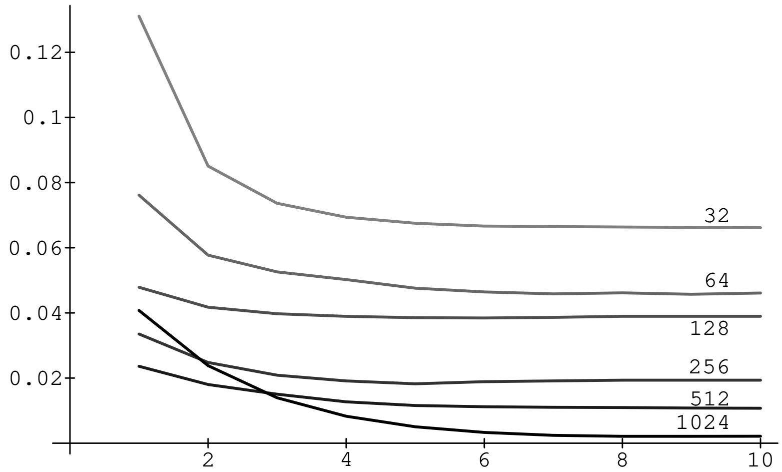

Figure 4-9: MSE as a function of number of forward-backward iterations, varying observation-sequence length as a parameter.

There were a few notable deviations from the typical results shown in Figure 4-9. In particular, for the larger values of dogmatism (2 and 4), the ML MSE initially increased with the number of forward-backward iterations when the observation length was short (32 or 64). This behavior is illustrated in Figure 4-10. The reason for this increase is as follows. From the analysis in Chapter 2, it is known that samples from the true mean's distribution are very close to the *a priori* mean μ_o when the dogmatism is large. Since the ML estimate is initialized with μ_o, the initial MSE is relatively low. As the forward-backward iterations continue, however, the ML estimate is unable to reliably estimate the true mean when the number of observations is small. When these poor reestimates replace the current mean and the process is repeated, the ML mean migrates away from μ_o (and the true mean) and the error increases. The EMAP and LMS-C estimators do not exhibit this behavior since they include a μ_o term which is weighted more heavily when the number of observations is small. Note that under the same large-dogmatism conditions, but with longer observation lengths, the ML estimate does converge.

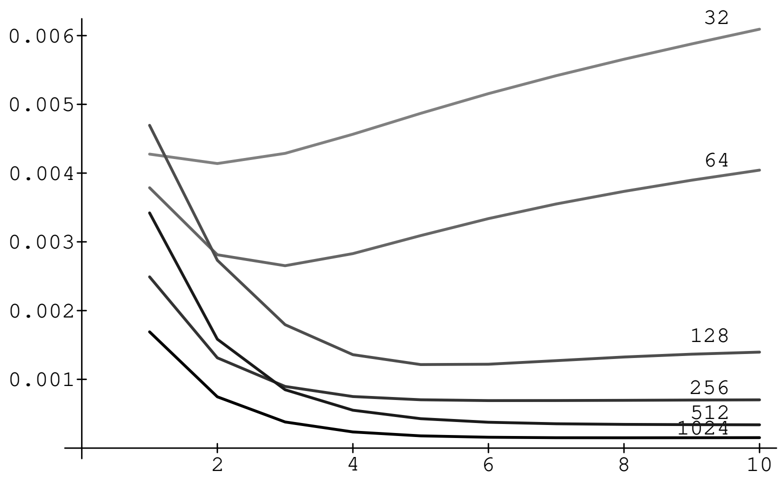

Figure 4-10: Family of curves of maximum likelihood MSE vs. forward-backward iteration for various observation sequence lengths and (δ,ρ) = (2.0,0.9). Note the increase in MSE with forward-backward iteration for lengths 32 and 64.

In addition to the divergence of the ML, there is also an occasional small fluctuation in mean-square error after convergence. This change occurs because the reestimation procedure is attempting to maximize $Pr(O|\lambda)$ which doesn't necessarily mean that the estimation squared-error will be minimized.

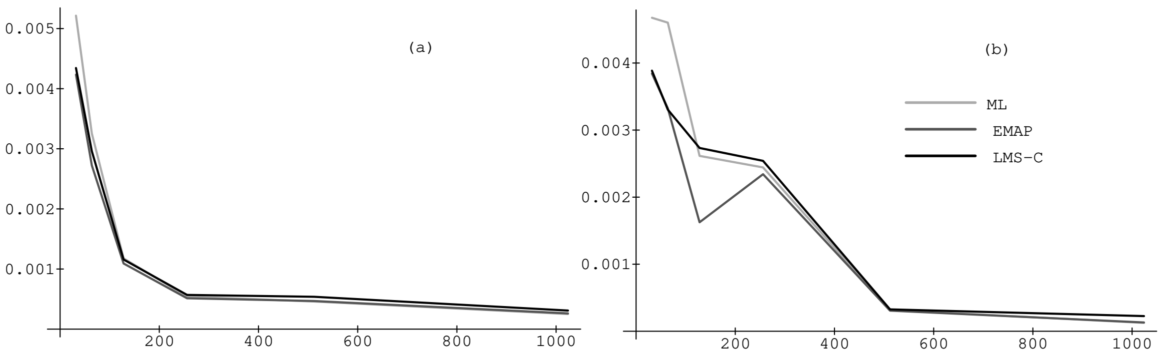

Figure 4-11: Asymptotic MSE (after 10 forward-backward iterations) vs. observation sequence length for ML, EMAP, and LMS-C estimates for (δ,ρ) = (a) (1.0,0.5) and (b) (0.5,0.9).

Figures 4-11 (a) and (b) show typical results using computer-generated data in the form of learning curves of MSE versus number of observations, as in Chapter 2. Shown in each graph is the asymptotic MSE (after 10 forward-backward iterations) for the ML, EMAP, and LMSC estimators. Under average conditions (4-11(a)), there is no difference between the four estimates. Under conditions most favorable for adaptation (4-11(b)), the differences are minor and occur only for shorter observation sequence lengths. Figure 4-12 further demonstrates this point for the learning curve for MSE versus forward-backward iteration. The difference between the ML and EMAP MSE in this graph is negligible for all iterations of the forward-backward algorithm.

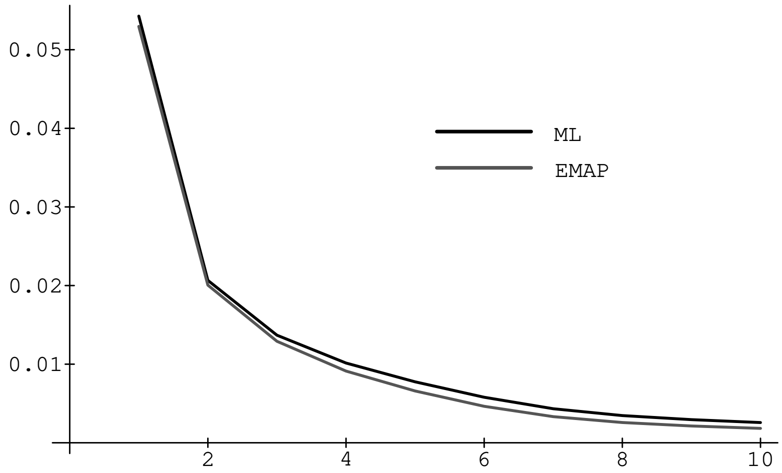

Figure 4-12: ML and EMAP MSE vs forward-backward iteration for observation sequence length of 256, $\delta = 0.5$, $\rho = 0.9$.

The behavior with respect to dogmatism of the learning curves for MSE as a function of forward-backward iteration is as one might expect given the analysis in Chapter 2. For small dogmatism, the curves showed a large initial error which converged fairly quickly. As the dogmatism increases, these curves become flatter since the trial means are closer to the prior mean μ_o. With respect to correlation, the MSE decreased as the means of the data became more correlated. Also, as the correlation increased the difference between the asymptotic MSE for short and long observation sequences was smaller, *i.e.*, shorter lengths were almost as effective as longer sequences. A crucial observation from the correlation experiments is that the results were the same for all estimators, *including the maximum likelihood estimate*. The ML estimate does not explicitly model correlation, yet (as Figure 4-13 shows), in the HMM framework this estimate demonstrates a dependence on correlation.

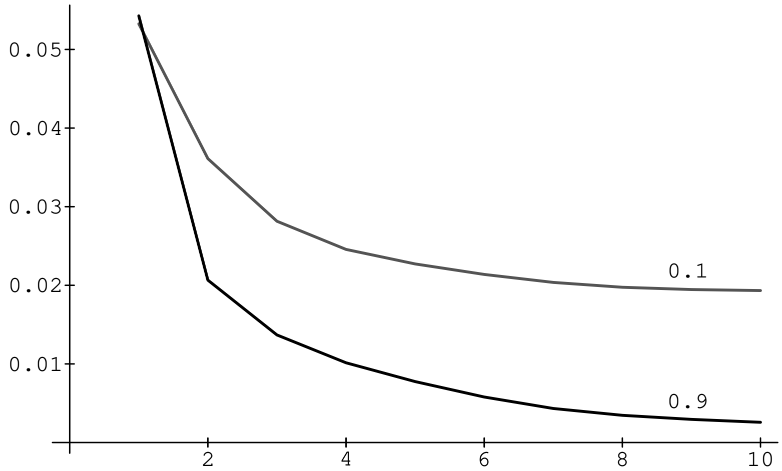

Figure 4-13: Maximum likelihood mean-square error vs forward-backward iteration for correlations of 0.1 and 0.9 with $\delta = 0.5$.

The reason for the dependence of the ML estimate on the correlation of the means of the data is that the HMM reestimation procedure implicitly models this correlation. What has been referred to as the maximum likelihood estimate in the context of the HMM studies here is the reestimation procedure which is specified by imposing Markov properties on the general EM procedure (see Equation (4.19)). It is not the same as the familiar sample mean (Equation (2.10)) because the HMM ML mean updates each mixture component according to the probability that the current observation was produced by that component, so each component is updated after each observation, and often with data attributable to other components. On the other hand, the sample mean (using supervised label feedback) updates only the true component or class. Since no data from other classes are involved, the sample mean estimate and MSE are therefore independent of correlation as Equation (2.12) indicates. The HMM ML estimate, due to its use of a probabilistic update, improves as correlation increases because as the means of the observations become more correlated, data from components other than the correct one convey more information about the correct component's mean.

An analytical expression for the HMM ML MSE most likely does not exist, but that it decreases with correlation can be described roughly and verified with empirical results. With the sample mean, each class mean is updated with data from only that class. When deriving the MSE, then, it is possible to consider each class separately since no cross-class terms (elements from the off-diagonal blocks of Σ_o) appear in the expected value. For the HMM ML reestimate, each class mean is a weighted summation of data from all other classes, and the MSE expression is no longer separable. Furthermore, the data from other classes introduce cross-class terms into the MSE expression which, as the empirical results demonstrate, help to reduce estimation error.

To substantiate these claims, two additional HMM experiments with computer-generated data were performed. The first experiment used a hard decision in the mean-reestimation procedure for the HMM, *i.e.*, only the most likely mixture component was updated with the current observation. The MSE should be greater using updates based on hard decisions than with probabilistic updating because only one component is updated per observation. The MSE should still decrease as correlation increases, however, since some useful information is obtained even when the hard decision is wrong (or in other words there are still some [but fewer] cross terms in the MSE expression). The second experiment was conducted using supervised feedback of component membership. In this case, like the sample mean, the HMM ML estimate should be independent of correlation since no cross-class terms appear in the MSE expression.

Figure 4-14 shows the results of these two experiments for $\delta = 0.5$, $\rho = 0.9$, and $T = 256$. The uppermost curve is the MSE for the hard-decision experiment, the middle curve is for probabilistic updating (the conventional HMM estimate), and the lower curve is from the supervised component feedback case. In all cases the empirical MSE behaves as predicted. Comparison of the hard decision

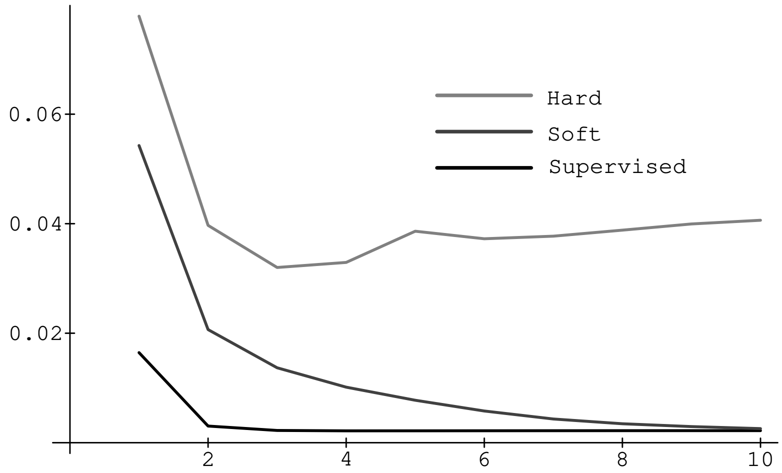

Figure 4-14: Maximum likelihood mean-square error for supervised feedback, soft decision, and hard decision of component membership with $(\delta,\rho,T) = (0.5,0.9,256)$.

and probabilistic curves demonstrates the importance of updating the component means after *every* observation. The figure also shows that the probabilistic ML MSE is initially much higher than with supervised feedback, but that repeated forward-backward reestimation allows the ML estimate MSE to eventually get down to supervised feedback MSE levels.

As a result of the increased ML MSE due to probabilistic decisions in the HMM reestimation procedure, performance of the more sophisticated EMAP and LMS-C estimates, which depend on the ML estimate, is degraded. This can be verified by performing EMAP adaptation in the supervised feedback experiment; the resulting MSE is clearly lower than the ML MSE. Supervised feedback, however, is unavailable in realistic situations. Since the EMAP and LMS-C performance is degraded, and the ML estimate makes use of correlation information, all three produce similar results in the context of hidden Markov models.

The above analysis suggests reasons for the marginal differences between the estimation procedures in the SPHINX-SC system. The reason that none of the estimators succeeded in substantially reducing the error rate is that the dogmatism was high, averaging 2.0 and 3.2 for the cepstral and differenced-cepstral codebooks. When dogmatism is this high, there's little difference between the prior (speaker independent) and trial (speaker dependent) means, and mean-vector adaptation is ineffective. The euclidean distances between adapted and unadapted means from SPHINX-SC codebooks were typically only 10%, 8%, and 1% of the magnitude of μ_o for the cepstral, difference cepstral, and power cepstral codebooks respectively. In addition, no further improvement in MSE was observed by further iterations of the forward-backward algorithm using the computer-simulated data. This was also observed in the SPHINX-SC tests.

4.6. Summary

In this chapter the adaptation procedures of Chapter 2 were applied to the recognition of a difficult phonetic-discrimination task using the SPHINX-SC system, which is based on semi-continuous HMMs. Adaptation in SPHINX-SC via reestimation of the mean vectors of the codebooks was shown to substantially reduce word error rates for certain individual speakers, but the reduction in error rate averaged over all speakers was only 3.4 percent or less, or approximately 4 percent when a second procedure is invoked that automatically evaluates the effect of the adaptation and rejects some speakers as inappropriate candidates for adaptation. Subsequent analyses using computer-simulated data identified the attributes of the original data that limited the effectiveness of this method of adaptation for this task.

No significant difference between the maximum likelihood and EMAP or LMS-C reestimation procedures was observed using the SPHINX-SC system. Results with computer-generated data indicate that the ML procedure's probabilistic update, necessary for unsupervised adaptation of HMM parameters, has both a positive and negative effect. The positive effect is that it allows all components to be updated with data from each observation. Updating a component mean with data attributable to other components implicitly includes a dependence on correlation not present in supervised ML (sample-mean) adaptation. This lowers estimation error when the means of the components become more correlated. The negative effect of the probabilistic update procedure is that the mean-square error is much higher than in the supervised case. Performance is therefore diminished for the ML estimate and the EMAP and LMS-C estimates which include the ML mean. Iteration of the forward-backward algorithm can reduce the mean-square error to supervised levels, but only when dogmatism is near or below 1.0 are there any significant difference between the ML, LMS-C, and EMAP estimates.

Chapter 5
Conclusions and Suggestions for Future Work

5.1. Overview of the Speaker Adaptation Problem

The present study addressed the problem of speaker adaptation in continuous speech recognition systems. Emphasis has been placed on adjusting the parameters of a set of reference templates to the characteristics of a new speaker. Speaker-adaptive systems circumvent the need for the lengthy enrollment procedures of speaker-dependent systems, and can improve speaker-independent system performance for speakers whose parameters markedly deviate from the speaker-independent parameters.

The desire for a speaker-adaptive, continuous-speech system places two important restrictions on the adaptation process. One is the necessity for a quick enrollment procedure. Implications of this restriction for adaptation are that the algorithms must be able to adapt on a relatively small amount of speaker-specific training data, and they must not incur a large computational load. The data requirements must be kept well below those for training speaker-dependent systems to justify the use of adaptation. Given this limited set of observations, the adaptation algorithm should be able to exploit any and all information available in the data quickly, and without sacrificing accuracy for speed.

The second restriction is imposed by the continuous speaking style. With continuous speech, user feedback of phonetic or other subword recognition unit labels is unrealistic, if not impossible. The adaptation algorithm must therefore operate in an unsupervised mode.

Based on prior work with adaptation in the FEATURE system and hidden Markov model-based systems, the approach taken in the present study was to update mean vectors of the feature-based system's parametric classifiers or the semi-continuous HMM's codebook. These parameters represent the most basic differences between speakers, and are less sensitive than other parameters to the effects of training on limited numbers of speaker-specific data. Due to the constraints imposed by speaker-adaptive recognition, the emphasis in this work was on computational complexity and initial convergence, *i.e.* to minimize the initial mean-square error as fast as possible.

5.2. Conclusions

The specific goal of the present study was to recast the extended maximum *a posteriori* (EMAP) estimation procedure in adaptive filtering terms, and to determine how successful this adaptive-filter implementation was at reducing error rates in continuous speech recognition systems. The first part of this goal was met through the development of a modified least mean-square (LMS) algorithm called LMS-C. The EMAP algorithm was demonstrated to be equivalent to a minimum mean-square error (MMSE) adaptive filter with a time-varying performance surface and optimal solution. A stochastic gradient approximation to the MMSE filter resulted in a LMS procedure which required only one-third the computation of the EMAP estimate.

Development of the expressions for the LMS-C estimate alone was not sufficient to provide good estimation performance. Coefficient initialization and choice of the LMS desired signal were shown to be crucial for low initial error. By analyzing the convergence properties of the EMAP estimate and its LMS implementation, it was possible to determine LMS-C parameters which incorporated *a priori* knowledge of the structure of the data into the LMS-C estimate. It was empirically demonstrated that the parameter values specified by the convergence analysis were necessary to minimize the initial estimation error. In simulations, use of these parameters produced a mean-square error (MSE) which was *lower* than the EMAP estimate.

Application of the LMS-C procedure to feature-based recognition was a qualified success. When using data which were known to obey the gaussian assumption, the LMS-C algorithm proved to be as effective as the EMAP procedure. Both algorithms reduced the error rate in a four-class phonetic classification example by 29 percent in an unsupervised mode. Additional tests with computer-generated data (generated using statistics of the actual front-vowel features) showed that repeated iterations of the unsupervised adaptation procedure were able to reduce classification error rates to levels closer to those obtained with supervised adaptation. Error-rate reductions after five iterations were on the order of 38 percent.

Application to hidden Markov model-based recognition systems proved to be more problematic. Distributions of the cepstral data in a semi-continuous version of the Carnegie Mellon SPHINX system (called SPHINX-SC) exhibited a large dogmatism, being on average greater than 2.0. Results from simulations, as discussed below, and analysis of the mean vector estimation procedures indicated that the expected gains due to mean vector adaptation are limited under these data conditions. The SPHINX-SC adaptation results showed an overall reduction of 2 to 3.4 percent in word-error rate. Simulations indicated that in HMM systems where the dogmatism of the data is below 1.0 the potential gains due to LMS-C or EMAP adaptation are larger.

A more detailed summary of the conclusions of this thesis is presented in the following sections, followed by suggestions for future research.

5.2.1. Development of the LMS-C Mean Vector Estimation Algorithm

The maximum likelihood (ML) algorithm satisfies the computational constraint imposed by speaker-adaptive recognition, but it does not always produce the most accurate estimate possible. The EMAP algorithm satisfies the accuracy constraint and exploits much of the information present in the data, but it does so at a large computational cost. Since maximum *a posteriori* and minimum mean-square error (MMSE) estimates are equivalent for normally-distributed data, the EMAP estimation procedure was formulated as an MMSE adaptive filter. Making a stochastic gradient approximation to this MMSE formulation resulted in an LMS procedure which performs a gradient search across the MMSE's time-varying performance surface. The LMS estimator was shown to be stable under conditions of low dogmatism (which are the conditions most suitable for adaptation by any method), and it requires only one-third of the computation of the EMAP estimate. Because the LMS-C coefficient update is based largely on observations, the LMS-C estimate can produce initial estimation errors lower than the EMAP or ML estimates with the proper initialization. This method of coefficient update, however, introduces a bias into the LMS estimate.

Proper choices for the LMS-C parameters were determined through analysis of the expected value of the LMS-C coefficient error. The time-varying nature of the EMAP's optimal coefficients introduced terms into the LMS-C convergence analysis which are not present in conventional LMS adaptive filtering. Further analysis showed that these additional terms are negligible when the dogmatism of the data is not large. The remainder of the LMS convergence analysis indicated that setting the initial LMS-C coefficient matrix elements equal to the initial values of the EMAP coefficients should minimize the LMS-C coefficient error. Initialization in this manner incorporates the statistics of the data into the LMS-C estimate. Empirical tests showed that initialization of the LMS-C weight matrix with the EMAP coefficients eliminated the large initial MSE which occurred with other forms of initialization.

The LMS-C adaptive filter's desired signal affects both the initial error and misadjustment of the LMS-C estimate. Close scrutiny of the EMAP coefficient update process indicated that the EMAP coefficients change slowly over time, and that a reasonably accurate, computationally-inexpensive estimate could be obtained by fixing the number of observations (**N**) matrix in the EMAP expression at some fixed value $\mathbf{N}_c = \eta \mathbf{I}$ where η is a constant called the convergence-point parameter. The resulting constant-EMAP (CEMAP) estimate's MSE is *identical* to that of the EMAP at the convergence point η, and is slightly higher on either side of this point. Using the CEMAP estimate as the desired signal in the LMS-C procedure produces better convergence properties than other choices (such as the ML estimate). The parameter η in the CEMAP procedure was shown to control the tradeoff between initial convergence and misadjustment in the LMS-C estimate.

The most important predictor of the performance of LMS-C and EMAP algorithms for mean-vector adaptation was determined to be the dogmatism of the data. For intermediate values of dogmatism (near 1.0), the EMAP and LMS-C algorithms can provide substantial gains in accuracy with respect to the ML estimator, as was demonstrated in empirical tests and in the feature-based experiments. When dogmatism is much less or much greater than 1.0, adaptation via EMAP or LMS-C provide no gains in estimation accuracy. The ML mean is sufficient when the dogmatism of the data is extremely low. When the dogmatism is large, as in the SPHINX-SC system, the *a priori* or speaker-independent mean μ_o is the best estimator.

Secondary predictors of adaptation performance are the degree of correlation between the class (or mixture component) means, and the skew in the prior probabilities of the classes. These measures are useful indicators only when the dogmatism of the data is at intermediate values. The relative gains of the EMAP and LMS-C algorithms over the ML estimate were shown to increase with increasing correlation between the class means. The EMAP and LMS-C estimates exploit this correlation to reduce estimation error when some classes have a low probability of occurrence.

5.2.2. Application to Feature-Based Speech Recognition Systems

When applied to unsupervised adaptation in feature-based continuous recognition systems, the LMS-C and EMAP estimators are capable of producing substantial reductions in classification error rates if a number of conditions are met. Foremost is that the dogmatism of the data must not be large; dogmatism near 1.0 appears to be sufficient. Comparisons between the acoustic-phonetic classification module in the ANGEL continuous speech recognition system and the rule-based classifier PROPHET indicate that another critical factor is that the adaptation data must obey the gaussian assumption. By applying simple rules derived from the spectrogram reading process, the PROPHET system was able to make some classification decisions, based on any non-gaussian features, to identify segments which were not members of the target classes (the front vowels). Segments identified as members of the target classes were classified using the gaussian features, and these feature values were used to update the classifier during unsupervised adaptive training. PROPHET's hybrid rule-based and statistical classification scheme improved classification results by 38 percent with respect to a gaussian classifier alone.

The hybrid classifier also provides a better opportunity for the adaptation algorithms to learn the characteristics of the target classes. The soft-decision method of unsupervised adaptation used in the feature-based experiments updated each class based on the probability that the current observation was a member of that class. The baseline gaussian classifier without rules (SPIRIT) was unable to filter out any of the non-target classes, and so the target class parameters were corrupted by the preponderance of non-target segments. In the PROPHET classifier, the majority of the non-target segments are filtered out before adaptation, so these data cannot adversely affect the target class parameters. LMS-C and EMAP adaptation in the PROPHET system produced a 29 percent reduction in error rate, while the ML-adapted PROPHET system and the all of the adapted SPIRIT systems showed an increase in error rate.

Experiments with classification of the three front vowels in the SPIRIT and PROPHET demonstrated the importance of obtaining quantities of data from each training speaker which are sufficient for estimation of the covariance matrix Σ_o. If too few samples are available from the decision classes for a large number of speakers, the statistic Σ_o will not accurately represent the distribution of the speakers' means. This leads to adaptation problems in the EMAP and LMS-C estimates which use Σ_o in the computation of their coefficients. The TIMIT CD-ROM used in the feature-based studies did not provide enough data to estimate this statistic for the classes under consideration. In the absence of a reliable estimate of this parameter, computer simulations of feature-based adaptation were conducted. Results with the computer-generated data provide an upper bound on the expected results with real data. These simulations are believed to be sound predictors of actual results since the real and simulation unadapted error rates (which depend only on μ_o and Σ and not on Σ_o) were in good agreement.

Adaptation experiments with computer-generated data showed the soft-decision feedback method to be effective for unsupervised adaptation in feature-based systems. With data generated using the statistics of the front vowel features, the EMAP and LMS-C algorithms reduced the classification error rate by approximately 28 percent after presentation of 10 adaptation data samples. This represented a gain of 40 percent with respect to the error rate using the ML-adapted mean vector. In addition, the EMAP and LMS-C error rates after 10 observations were almost at the asymptotic (many-observation) level of error. When five iterations of this unsupervised adaptation were performed on the adaptation training data, the error rate was reduced by an additional 10 percent over that of one iteration. This reduction represented 36 percent of the difference between the one iteration and supervised adaptation error rates.

5.2.3. Application to Hidden Markov Model-Based Systems

Results from studies with a computer-simulated HMM showed that the maximum likelihood mean estimate specified in the HMM reestimation procedure has different properties than the ML mean obtained in a supervised mode. One consequence of the unsupervised nature of this reestimate is that the MSE is much higher than that of the supervised mean. It was shown that this additional error may be reduced to the supervised level by repeated iterations of the forward-backward algorithm. In other words, by repeated application of the HMM reestimation procedure the unsupervised estimates converge to those which would be obtained given full knowledge of the component membership of each observation. Similar comments apply to the soft-decision mean in the feature-based system since that estimate, like the HMM reestimate, updates each class based on the probability that the observation is a member of that class.

The HMM mean reestimate also exhibits a dependence on the correlation of the data means. This dependence is not present in the supervised ML mean. Since each element of the reestimated ML mean vector is a weighted sum of data from all classes (or mixture components), it implicitly models correlation information which the EMAP and LMS-C estimators explicitly incorporate through the covariance matrix Σ_o. The dependence of the unsupervised ML mean on correlation of the data means was shown to reduce ML estimation error as the correlation increased, similar to the EMAP and LMS-C algorithms.

The properties of the ML mean as obtained via the HMM reestimation procedure, along with the large dogmatism (2.0 or greater) of the cepstral data combined to produce only limited gains due to adaptation in the SPHINX-SC system. The increased estimation error of the unsupervised ML reestimate increases EMAP and LMS-C error since they incorporate the ML mean in their respective estimation procedures. More importantly, with dogmatism at or above 2.0, the deviation of the individuals' means from μ_o is small, leaving little room for improvement by mean vector adaptation. Overall, the word error rate was reduced by 2 to 3.4 percent by adaptation using HMMs with semi-continuous codebooks.

Results from individual speakers showed a wider range of change in error rate. The most successful method of automatically identifying those speakers who are helped by adaptation, based on the change in euclidean distances between codebook components, averaged a 15 percent reduction for the selected speakers. This represented an overall reduction of approximately 4 percent. Simulations showed that under more favorable conditions, small dogmatism and high correlation, the potential for EMAP or LMS-C estimates to show any significant gains over the ML estimate is larger.

5.3. Directions for Future Research

There are a number of areas in which this work may be extended. The most obvious course of action is to assess whether other recognition systems would benefit from the techniques presented here. Significant improvements have been made with the semi-continuous SPHINX system since obtaining the version which became SPHINX-SC, including separate male and female models. The statistics of the data with this newer version of SPHINX-SC may be better suited for adaptation. Other HMM systems, possibly those developed for smaller tasks such as connected digit recognition, may also be more amenable to adaptation by the techniques developed here.

Although shifting mean vectors works well under the right conditions, greater gains may be obtained by simultaneously adapting other recognition parameters. These additional parameters may be the covariance matrices in feature-based classifiers or SCHMM codebooks, or transition and output probabilities in HMMs. Care must be taken to avoid adaptation of parameters which are sensitive to small training sample sizes.

Another promising area is the improvement of automatic methods for identifying those speakers who are good candidates for adaptation. It is unlikely that all speakers will benefit from adaptation. It may be possible to identify the characteristics of those speakers who do show some improvement. These characteristics are likely to be found in some comparative measure of the adapted and unadapted codebooks or classifier parameters. In the context of a system which is more amenable to adaptation these characteristics may possibly be more easily identified.

The techniques described here may also find applications in adaptation to environmental or other test conditions, such as noise level or microphone differences. It would be necessary to determine the effects that changes in these conditions have on the cepstral representation of speech data, and whether these effects can be modeled by shifts in the codebook means.

The algorithms themselves may also be developed further. Currently trial and error is necessary to choose the best settings for η and β in the LMS-C algorithm. In simulations, the best values for η were much smaller in the multivariate case than in one-dimensional tests. Further study of the algorithm and empirical results may lead to an automated procedure for selecting these parameters.

Finally, the EMAP algorithm specifies the coefficients which allow the mean estimate to converge with no bias. This set of coefficients minimizes the mean-squared error for all values of $\mathbf{N}$. Since speaker adaptation seeks low initial error, modification of the minimization criterion may lead to an estimator which accepts some asymptotic bias but optimally minimizes this initial error.

References

1. Schwartz,R., Chow,Y., and Kubala,F., ‘‘Rapid Speaker Adaptation Using a Probabilistic Spectral Mapping’’, *Proceedings of ICASSP87*, IEEE Acous., Speech, and Signal Proc. Society, 1987, pp. 15.3.1-4.

2. Feng,M.W., Kubala,F., Schwartz,R., and Makhoul,J., ‘‘Rapid Speaker Adaptation Using a Probabilistic Spectral Mapping’’, *Proceedings of ICASSP88*, IEEE Acous., Speech, and Signal Proc. Society, 1988, pp. 131-134.

3. Feng,M.W., Schwartz,R., Kubala,F., and Makhoul,J., ‘‘Iterative Normalization for Speaker-Adaptive Recognition in Continuous Speech Recognition’’, *Proceedings of ICASSP89*, IEEE Acous., Speech, and Signal Proc. Society, 1989, pp. 612-615.

4. Nishimura,M, and Sugawara,K., ‘‘Speaker Adaptation Method for HMM-Based Speech Recognition’’, *Proceedings of ICASSP88*, IEEE Acous., Speech, and Signal Proc. Society, 1988, pp. 207-210.

5. Furui,S., ‘‘Unsupervised Speaker Adaptation Method Based on Hierarchical Spectral Clustering’’, *Proceedings of ICASSP89*, IEEE Acous., Speech, and Signal Proc. Society, 1989, pp. 286-289.

6. Lee,K.F., *Large-Vocabulary Speaker-Independent Continuous Speech Recognition: The SPHINX System*, PhD dissertation, Computer Science Department, Carnegie Mellon University, 1988.

7. Lee,K.F., Hon,H.W., and Reddy.R., ‘‘An Overview of the SPHINX Speech Recognition System’’, *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. 38, No. 1, January 1990.

8. Rabiner,L.R., ‘‘On Creating Reference Templates for Speaker-Independent Recognition of Isolated Words’’, *IEEE Trans. ASSP*, Vol. ASSP-27, No. 4, 1979, pp. 236-249.

9. Rabiner,L.R., Levinson,S.E., Rosenberg,A.E., and Wilpon,J.G., ‘‘Speaker-Independent Recognition of Isolated Words Using Clustering Techniques’’, *IEEE Trans. ASSP*, Vol. ASSP-27, No. 4, 1979, pp. 236-249.

10. Kijima,Y., Nara,Y., Kobayashi,A., and Kimura,S., ‘‘Speaker Adaptation in Large-Vocabulary Voice Recognition’’, *Proceedings of ICASSP84*, IEEE Acous., Speech, and Signal Proc. Society, 1984, pp. 26.8.1-4.

11. Shikano,K., Lee,K.F., and Reddy,D.R., ‘‘Speaker Adaptation Through Vector Quantization’’, *Proc. International Conf. ASSP*, 1986, pp. 2643-2646.

12. Higuchi,N., and Yato,F., ‘‘Speaker Adaptation Methods using Selective Linear Prediction’’, *Proceedings of ICASSP86*, IEEE Signal Processing Society, 1986, pp. 49.6.1-4.

13. Hampshire,J.B., and Waibel,A.H., ‘‘The Meta-Pi Network: Connectionist Rapid Adaptation for High-Performance Multi-Speaker Phoneme Recognition’’, *Proceedings of ICASSP90*, IEEE Signal Processing Society, 1990, pp. 165-168.

14. Hon,H.W., and Huang,X.D., ‘‘personal communication’’.

15. Kenney,P., Lennig,M., and Mermelstein,P., ‘‘Speaker Adaptation in a Large Vocabulary Gaussian HMM Recognizer’’, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 1989.

16. Lee,C.H., Lin,C.H., and Juang,B.H., ‘‘A Study on Speaker Adaptation of Continuous Density HMM Parameters’’, *Proceedings of ICASSP90*, IEEE Signal Processing Society, 1990, pp. 145-148.

17. Rtischev,D., ‘‘Speaker Adaptation in a Large-Vocabulary Speech Recognition System’’, Master’s thesis, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1989.

18. Huang,X.D., Lee,K.F., Hon,H.W., and Huang,M., ‘‘Improved Acoustic Modeling for the SPHINX Speech Recognition System’’, *Proceedings of ICASSP91*, IEEE Signal Processing Society, 1991.

19. Martin,E.A., Lippmann,R.P., and Paul,D.P., ‘‘Dynamic Adaptation of Hidden Markov Models for Robust Isolated-Word Speech Recognition’’, *Proceedings of ICASSP88*, IEEE Acous., Speech, and Signal Proc. Society, 1988, pp. 52-54.

20. Huang,X.D., Hon,H.W., and Lee,K.F., ‘‘On Semi-Continuous Hidden Markov Modeling’’, *Proceedings of ICASSP89*, IEEE Signal Processing Society, 1989, pp. 689-692.

21. Huang,X.D., ‘‘personal communication’’.

22. Brown,P.F., Lee,C.H., and Spohr,J.C., ‘‘Bayesian Adaptation in Speech Recognition’’, *IEEE*, 1983, pp. 761-764.

23. Stern,R.M., and Lasry,M.J., ‘‘Dynamic Speaker Adaptation for Feature-Based Isolated Letter Recognition’’, *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. 35, 1987, pp. 751-763.

24. Lasry,M.J., and Stern,R.M., ‘‘A Posteriori Estimation of Correlated Jointly Gaussian Mean Vectors’’, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. PAMI-6, No. 4, 1984, pp. 530-535.

25. Van Trees,H.L., *Detection, Estimation, and Modulation Theory, Part I*, Wiley, New York, 1968.

26. Widrow,B. and Stearns,S.D., *Adaptive Signal Processing*, Prentice-Hall Inc., Englewood Cliffs NJ, 1985.

27. Cole,R.A., Rudnicky,A.I., Zue,V.W., and Reddy,D.R., *Speech as Patterns on Paper*, Lawrence Erlbaum Associates, 1980, ch. 1.

28. Zue,V.W., ‘‘The Use of Speech Knowledge in Automatic Speech Recognition’’, *Proceedings of the IEEE*, Vol. 73, No. 11, November 1985, pp. 1602-1615.

29. Chigier,B., ‘‘Classification of Stop Consonants in Natural Continuous Speech’’, Master’s thesis, Dept. of Electrical and Computer Engineering, Carnegie Mellon University, 1988.

30. Lamel,L.F., *Formalizing Knowledge used in Spectrogram Reading: Acoustic and perceptual evidence from stops*, PhD dissertation, Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1988.

31. Chigier,B. and Brennan,R.A., ‘‘Broad Class Network Generation Using a Combination of Rules and Statistics for Speaker Independent Continuous Speech’’, *Proceedings of ICASSP88*, IEEE Acoustics, Speech, and Signal Processing Society, 1988, pp. 449-452.

32. Duda,R.O., and Hart,P.E., *Pattern Classification and Scene Analysis,* Wiley, New York, 1973.

33. Stern,R.M., Ward,W.H., Hauptmann,A.G., and Leon,J., ‘‘Sentence Parsing with Weak Grammatical Constraints’’, *Proceedings of ICASSP87*, IEEE Acous., Speech, and Signal Proc. Society, 1987, pp. 380-383.

34. ‘‘The DARPA TIMIT prototype database’’, Distributed on CD-ROM by NIST.

35. Shore,J. and Burton,D., ‘‘ESPS/waves+ User’s Manual’’, Entropic Speech, Inc..

36. Talkin,D., ‘‘Speech Formant Trajectory Estimation using Dynamic Programming with Modulated Transition Costs’’, *JASA*, Vol. 82, 1987, pp. S55.

37. Secrest,B.G., and Doddington,G.R., ‘‘An Integrated Pitch Tracking Algorithm for Speech Systems’’, *Proceedings of ICASSP83*, IEEE Acoustics, Speech, and Signal Processing Society, 1983, pp. 1352-1355.

38. Rabiner,L.R., ‘‘A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition’’, *Proceedings of the IEEE*, Vol. 77, No. 2, 1989, pp. 257-286.

39. Baum,L.E., ‘‘An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of Markov Processes’’, *Inequalities*, Vol. 3, 1972, pp. 1-8.

40. Forney,G.D., ‘‘The Viterbi Algorithm’’, *Proceedings IEEE*, Vol. 61, March 1973, pp. 268-278.

41. Redner,R.A., and Walker,H.F., ‘‘Mixture Densities, Maximum Likelihood and the EM Algorithm’’, *SIAM Review*, Vol. 26, No. 2, 1984, pp. 195-239.

42. Dempster,A.P., Laird,N.M., and Rubin,D.B., ‘‘Maximum-likelihood from Incomplete Data via the EM Algorithm’’, *J. Royal Statist. Soc.*, Vol. 39, 1977, pp. 1-38.

43. Huang,X.D., *Semi-Continuous Hidden Markov Models for Speech Recognition*, PhD dissertation, University of Edinburgh, 1989.

44. Juang,B.H., Levinson,S.E., and Sondhi,M.M., ‘‘Maximum Likelihood Estimation for Multivariate Mixture Observations of Markov Chains’’, *IEEE Trans. Information Theory,* Vol. IT-32, No. 2, 1986, pp. 307-309.

45. Brown,P.F., *The Acoustic-Modeling Problem in Automatic Speech Recognition*, PhD dissertation, Computer Science Department, Carnegie Mellon University, 1987.

46. Price,P., Fisher,W.M., Bernstein,J., and Pallett,D.S., ‘‘The DARPA 1000-Word Resource Management Database for Continuous Speech Recognition’’, *Proceedings of ICASSP88*, IEEE Acoustics, Speech, and Signal Processing Society, 1988, pp. 651-654.

Glossary of Acronyms

BBN	Bolt, Beranek, and Newman, Inc.
CDHMM	continuous-density hidden Markov model
CD-ROM	compact disk read-only memory
CEMAP	constant-EMAP, EMAP estimate with fixed coefficients
CMU	Carnegie Mellon University
DARPA	Defense Advanced Research Project Administration
DDHMM	discrete-density hidden Markov model
DFT	discrete Fourier transform
EM	estimation and maximization
EMAP	extended maximum *a priori*
F-B	forward-backward
HMM	hidden Markov model
IBM	International Business Machines
LMS	least mean-square
LMS-C	LMS estimation using CEMAP estimate as the desired signal
LPC	linear predictive coding
MAP	maximum *a priori*
ML	maximum likelihood
MMSE	minimum mean-square error
MSE	mean-square error
NIST	National Institute of Standards and Technology
SCHMM	semi-continuous hidden Markov model
SPHINX-SC	a semi-continuous version of the SPHINX recognition system
TDNN	time-delay neural network
TIMIT	Texas Instruments/Massachusetts Institute of Technology
TIRM	Texas Instruments resource management
VQ	vector quantization

Nomenclature

General

a, **b**, **x**	Bold lowercase Roman symbols represent column vectors or random variables.
A, **M**	Bold uppercase Roman symbols represent matrices.
i, *j*, *t*	Italicized lowercase Roman symbols represent integer indices.
∇	gradient operator
$\|\cdot\|$	vector magnitude operator
$\mathbf{A} \otimes \mathbf{B}$	Schur product, defined as $[\mathbf{A} \otimes \mathbf{B}]_{ij} = a_{ij} b_{ij}$
$E[\cdot]$	expected value operator
$erf[\cdot]$	standard normal error function
I	identity matrix
$log[\cdot]$	natural logarithm operator
$N(\mu,\Sigma)$	Normal distribution with mean vector μ and covariance matrix Σ
$p(\cdot)$, $f(\cdot)$	probability density functions
$Pr(\cdot)$	probability of the specified event
$tr[\cdot]$	matrix trace operator
$U(a,b)$	uniform distribution over the interval $[a,b]$

Notation from Estimation and Classification Theory (Chapters 2 and 3)

a	maximum likelihood (sample mean) estimate of μ
C	number of decision classes
C	CEMAP coefficient matrix
CD	product of the number of classes and dimensions, $C \cdot D$
j, c_j	class index and j^{th} class
d	LMS desired signal
D	dimension of observation vector
H	MMSE and LMS-C coefficient matrix
$\mathbf{H}^*$	optimal value of coefficient matrix **H**
k	the total number of observations $\sum_{i=1}^{C} n_j$. As a subscript, k represents the value of the parameter after the k^{th} observation.
$l(\mu)$	log-likelihood function $log[p(\mu)]$
$\mathbf{L}_k$	expected value of LMS coefficient matrix, $E[\mathbf{H}_k]$
$MSE(\mathbf{N})$	mean-squared error matrix functional
n_j	number of observations from class j
N	diagonal matrix with $n_j\mathbf{I}$ as the diagonal blocks
$\mathbf{N}_c$	(fixed) count matrix in CEMAP procedure, $\eta\mathbf{I}$
$\mathbf{V}_k$	expected value of LMS coefficient matrix error, $E[\mathbf{H}_k - \mathbf{H}_k^*]$
$\overline{\mathbf{V}}_k$	$\mathbf{V}_k$ transformed into principal coordinates
x	random variable or observation data vector
α	sample mean prepended with a bias of 1.0, $[1 \quad \mathbf{a}^T]^T$

β	LMS step-size parameter
χ	set of observations from a single speaker
δ	dogmatism of the data
ε	error vector $\mu - \hat{\mu}$
$\Phi_{\mathbf{aa}}$	correlation matrix of **a**, $E[\mathbf{aa}^T]$
$\Phi_{\alpha\alpha}, \mathbf{R}_k$	correlation matrix of α, $E[\alpha\alpha^T]$. $\mathbf{R}_k$ explicitly indicates the dependence on the number of observations.
$\Phi_{\mathbf{a}\mu}$	crosscorrelation matrix of **a** and μ, $E[\mathbf{a}\mu^T]$
$\Phi_{\alpha\mu}, \mathbf{P}$	crosscorrelation matrix of α and μ, $E[\alpha\mu^T]$
$\Phi_{\mu\mu}$	correlation matrix of μ, $E[\mu\mu^T]$
γ	LMS-C misadjustment scale factor
η	CEMAP and LMS-C convergence point parameter
λ	eigenvalue
Λ, $\mathbf{Q}$	eigenvalue and eigenvector matrices of **R**
μ	mean vector of observations, to be estimated
$\hat{\mu}$	estimate of μ
μ_o	*a priori* mean vector $E[\mu]$
θ	skew in class prior probabilities
ρ	correlation of the data means
σ^2	data variance
Σ_j	covariance matrix of data from class (or mixture component) j
Σ_o	covariance matrix of μ

Notation from Classification Theory (Chapter 3)

/*a*/, /*a*/$_b$	phoneme *a* in any context or in the context of phoneme /*b*/ (see Appendix B for a complete list)
P_ε	probability of classification error (error rate)
Δ	percentage change in error rate

Notation from Hidden Markov Model Theory (Chapter 4)

	overlines indicate the reestimated parameter values
	underlines indicate vectors (used with Greek symbols)
$\mathbf{A}, a_{ij}$	state transition matrix and the $(i,j)^{th}$ element
$b_j[k], b_j[\mathbf{x}]$	discrete and continuous output distributions for state j
$\mathbf{B}_t$	diagonal matrix $diag[b_1[O_t], b_2[O_t], \ldots, b_N[O_t]]$
c_{jk}	k^{th} mixture component in state j
i, j	state indices
k	mixture component index
M	number of mixture components in SCHMM codebook
N	number of states in HMM
T	length of observation sequence O
O, O_t	observation sequence and observation at time t
Q, q_t	hidden state sequence and state at time t
Q^*, q_t^*	Viterbi decoded hidden state sequence and state at time t
S_j	HMM state j
v_k	discrete codebook VQ prototype vector
$\alpha_t[i]$	forward variable representing the probability of the partial observation sequence

	$O_1, O_2, \cdots, O_t$ and being in state *j* at time *t*, given the model
$\beta_t[i]$	backward variable representing the probability of the partial observation sequence from t+1 to the end, given state *i* at *t* and the model
$\delta_t[i]$	Viterbi variable representing the highest scoring single path, ending in state *i*, which produced the observation sequence
$\gamma_t[i]$	intermediate F-B variable representing the probability of being in state *i* at time *t* given the model and observations
$\gamma_t[j,k]$	intermediate F-B variable representing the probability of being in state *k* at time *t* and having the k^{th} component produce the observation, given the model and observations
λ	the set of current HMM parameters
$\bar{\mu}$	reestimated (adapted) mean vector
π	initial state probability distribution
$\xi_t[i,j]$	probability of transition from S_i to S_j at time *t*
$\psi_t[j]$	Viterbi variable holding the index of the best state from the previous time instant (*t-1*) which passes through the current state *j*
$\zeta_t[j]$	probability of the j^{th} component producing the observation at time *t*

Appendix A

Derivation of Expected Values of Selected Parameters

For notational simplicity, assume a one-class, D-dimensional case with $\mathbf{N} = n\mathbf{I}$. The results below apply equally well to the general C-class case. Denote the observations from a given speaker as $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.

Sample Mean, E[a]:

The sample mean $\mathbf{a}$ can be written as $\mathbf{a} = \mathbf{N}^{-1}\sum_{i=1}^{n}\mathbf{x}_i$. The global (across all speakers) expected value $E[\mathbf{a}]$ is the expected value of each speaker's sample mean $E[\mathbf{a}|\mu]$, or $E[\mathbf{a}] = E\{E[\mathbf{a}|\mu]\}$. $E[\mathbf{a}|\mu]$ can be rewritten as

$$\begin{aligned} E[\mathbf{a}|\mu] &= \mathbf{N}^{-1}E[\sum_{i=1}^{n}\mathbf{x}_i \,|\, \mu] \\ &= \mathbf{N}^{-1}\sum_{i=1}^{n}E[\mathbf{x}_i|\mu] \\ &= \mathbf{N}^{-1}\mathbf{N}\mu \\ &= \mu \end{aligned} \tag{A.1}$$

Substituting back, the expected value of the sample mean is found to be $E[\mathbf{a}] = E[\mu] = \mu_o$.

Correlation Matrix E[μμ^T]:

By definition, $\Sigma_o = E[(\mu-\mu_o)^T(\mu-\mu_o)]$ or $\Sigma_o = E[\mu\mu^T] - \mu_o\mu_o^T$. Obviously then $\Phi_{\mu\mu} = E[\mu\mu^T] = \Sigma_o + \mu_o\mu_o^T$.

Correlation Matrix E[aa^T]:

Again, the global average is the expected value over all speakers, so $E[\mathbf{a}\mathbf{a}^T] = E\{E[\mathbf{a}\mathbf{a}^T \,|\, \mu]\}$. Rewriting,

$$\begin{aligned} E[\mathbf{a}\mathbf{a}^T \,|\, \mu] &= E[\mathbf{N}^{-1}\sum_{i=1}^{n}\mathbf{x}_i \sum_{j=1}^{n}\mathbf{x}_j^T\mathbf{N}^{-1} \,|\, \mu] \\ &= \mathbf{N}^{-1}\sum_{i=1}^{n}\sum_{j=1}^{n}E[\mathbf{x}_i\mathbf{x}_j^T \,|\, \mu]\mathbf{N}^{-1} \end{aligned} \tag{A.2}$$

From the definition of $p(\mathbf{x}_i \,|\, \mu)$, which is $N(\mu,\Sigma)$,

$$E[\mathbf{x}_i\mathbf{x}_j^T \,|\, \mu] = \begin{cases} \Sigma + \mu\mu^T, & i = j, \\ \mu\mu^T, & otherwise. \end{cases} \tag{A.3}$$

The double sum in (A.2) contains n terms with $i = j$ and $n^2 - n$ other terms, so using (A.3) this sum can be written as $\mathbf{N}\mu\mu^T\mathbf{N} + \Sigma\mathbf{N}$. Continuing,

$$\begin{aligned} E[\mathbf{aa}^T \,|\, \mu] &= \mathbf{N}^{-1}[\mathbf{N}\mu\mu^T\mathbf{N} + \Sigma\mathbf{N}]\mathbf{N}^{-1} \\ &= \mu\mu^T + \mathbf{N}^{-1}\Sigma \end{aligned} \tag{A.4}$$

The global expected value is then

$$\begin{aligned} E[\mathbf{aa}^T] &= E[\mu\mu^T + \mathbf{N}^{-1}\Sigma] \\ &= E[\mu\mu^T] + \mathbf{N}^{-1}\Sigma \\ &= \Sigma_o + \mathbf{N}^{-1}\Sigma + \mu_o\mu_o{}^T \end{aligned} \tag{A.5}$$

Crosscorrelation Matrix E[a$\mu^{\mathbf{T}}$]:

Similar to the above, $E[\mathbf{a}\mu^T] = E\{E[\mathbf{a}\mu^T \,|\, \mu]\}$. Rewriting the inner expected value,

$$\begin{aligned} E[\mathbf{a}\mu^T \,|\, \mu] &= E[\mathbf{N}^{-1}\sum_{i=1}^{n}\mathbf{x}_i\mu^T \,|\, \mu] \\ &= \mathbf{N}^{-1}\sum_{i=1}^{n}E[\mathbf{x}_i\mu^T \,|\, \mu] \\ &= \mathbf{N}^{-1}\sum_{i=1}^{n}E[\mathbf{x}_i \,|\, \mu]\mu^T \\ &= \mathbf{N}^{-1}\sum_{i=1}^{n}\mu\mu^T \\ &= \mu\mu^T \end{aligned} \tag{A.6}$$

From the expression for $E[\mu\mu^T]$ above, the result can directly be written as $E[\mathbf{a}\mu^T] = \Sigma_o + \mu_o\mu_o{}^T$.

Appendix B

Definitions of Speech Terms and Listings of Phonetic Classes and Labels

Definitions

- Closure: A period of silence or very low amplitude energy in the speech signal.
- Formants: Vocal tract resonant frequencies which appear as lines within voiced segments on a spectrogram. The ordinal numbering of formants begins with the formant lowest in frequency.
- Front Vowel: A sonorant which, in neutral contexts, has the second formant closer to the third than to the first.
- Obstruent: A speech sound characterized by turbulent air flow caused by constrictions in the vocal tract. Obstruents appear as aperiodic patterns in the waveform and as diffuse energy in a spectrogram.
- Semivowel: Sonorants characterized by extreme positions of the articulators and large formant motion. Semivowels have strong coarticulation effects on adjacent phonemes. They are composed of the phonemes in Groups II, III, and IV in the table below.
- Sonorant: A speech sound characterized by voicing and no constrictions or turbulence in the vocal tract. Sonorants appear as a quasi-periodic pattern in the speech waveform and lines (formants) in a spectrogram.
- Voicing: Vibration of the vocal cords.

Phonetic Classes and Labels

The following is a list of phonetic labels used for phonetic transcription. Groups I through VI constitute the sonorant classes, Groups VII through IX are obstruents, and closures are in Group X.

```
I. Vowels
 iy   'beat'
 ih   'bit'
 ey   'bait'
 eh   'bet'
 ae   'bat'
 ux   high, front, rounded  allophone of /uw/ as in 'beauty'
 oe   mid-low, front, rounded allophone of /ow/ (and perhaps /uw/)
 ix   high, central vowel (unstressed), as in 'roses'
 ax   mid, central vowel (unstressed), as in 'the'
 ah   mid, central vowel (stressed), as in 'butt'
 uw   'boot'
```

```
 uh   'book'
 ao   'bought'
 aa   'cot'
 ay   'bite'
 oy   'boy'
 aw   'bough'
 ow   'boat'
 e    non-diphthongized /ey/
 o    mid-low, back, non-diphthongized allophone of /ow/

II. Liquids
 l    'led'
 r    'red'

III. Glides
 y    'yet'
 w    'wet'

IV. Syllabic resonants
 er   'bird'
 axr  unstressed allophone of /er/, as in 'diner'
 el   syllabic allophone of /l/, as in 'bottle'
 em   syllabic allophone of /m/, as in 'yes 'em' ('yes ma'am')
 en   syllabic allophone of /n/, as in 'button'
 eng  syllabic allophone of /ng/, as in 'Washington' (uncommon)

V. Nasals
 m    'mom'
 n    'non'
 ng   'sing' (only occurs in syllable-final position in English)

VI. Flaps and trills
 dx   alveolar flap (allophone of [t] & [d])
 nx   nasal flap (allophone of [n])
 lx   lateral flap (allophone of [l])

VII. Stops
 p   'pop'
 b   'bob'
 t   'tot'
 d   'dad'
 k   'kick'
 g   'gag'
 q   glottal stop - allophone of /t/, as in 'Atlanta' where the
       first /t/ can be realized as [q].  Also may occur between
       words in continuous speech, especially at vowel-vowel
       boundaries, and at the beginning of vowel-initial utterances.

VIII. Affricates
 ch   'church'
 jh   'judge'

IX. Fricatives
```

ph voiceless, bilabial fricative - allophone of /p/
bh voiced, bilabial fricative - allophone of /b/
f 'fief'
v 'verv'
th 'thief'
dh 'they'
s 'sis'
z 'zoo'
sh 'shoe'
zh 'measure'
kh voiceless, velar fricative - allophone of /k/
gh voiced, velar fricative - allophone of /g/
hh 'hay'
hv voiced allophone of [hh], occurs between vowels

X. Silence
cl at this level, we identify the closure with its stop, e.g. [pcl] means that the closure is for a [p], whether the [p] is released or not
epi closure resulting from coarticulation of fricative and nasal or lateral
bg silence at the beginning and end of an utterance
pau silence within an utterance that does not correspond to the closure for a stop or affricate; usually audible at sentence level
sil same as [pau], but shorter, and not audible at sentence level

XI. Other
ns a non-speech sound
h# exhalation at end of utterance
#h inhalation at beginning of utterance
voi voicing not associated with a stop closure
-h appended to stops to signify aspiration; appended to any voiced segment to signify devoicing
-n appended to sonorant segments to signify nasalization
-q appended to sonorant segments to signify glottalization/laryngealization
-b appended to stops to signify the release of a stop in an environment where stops are often not released, e.g., [k-b] as in 'black board' if the [k] in 'black' is released or in clause-final position.

Appendix C

Front Vowel Distribution Statistics

The following statistics were compiled from ANGEL front vowel formant data from 179 speakers, 10 sentences per speaker, for a total of 6301 training tokens. The measures were the first three formants from the classes /*iy*/, /*ih*/, /*eh*/, respectively.

```
μo =
   {473 1982 2529 497 1607 2362 612 1633 2382}

Σo =
   {6186  3745  5012  2467 -671  4017  2583  5801  7709 }
   {3745  24675 19974 3207 16745 19266 7240  25772 31329}
   {5012  19974 23809 3205 12751 20585 6507  21318 30026}
   {2467  3207  3205  4748 -700  7884  2700  2284  4850 }
   {-671  16745 12751 -700 31480 22192 2198  23757 27150}
   {4017  19266 20585 7884 22192 58083 3678  18776 37837}
   {2583  7240  6507  2700 2198  3678  5300  7903  11153}
   {5801  25772 21318 2284 23757 18776 7903  35589 39744}
   {7709  31329 30026 4850 27150 37837 11153 39744 57808}

Σ =
   {41111 840    2372   0     0      0      0      0     0      }
   {840   130621 96861  0     0      0      0      0     0      }
   {2372  96861  146890 0     0      0      0      0     0      }
   {0     0      0      18766 5442   11180  0      0     0      }
   {0     0      0      5442  155722 133887 0      0     0      }
   {0     0      0      11180 133887 234222 0      0     0      }
   {0     0      0      0     0      0      32138  -6705 -10599}
   {0     0      0      0     0      0      -6705  50111 35278 }
   {0     0      0      0     0      0      -10599 35278 80401 }
```

The average dogmatism specified by these statistics, computed as the average ratio of the square-root of the diagonal elements of Σ to Σ_o, is 2.0. The average mean crosscorrelation, computed as the average of

$$\rho_{ij} = \frac{\Sigma_{oij}}{\sqrt{\Sigma_{oii}}\sqrt{\Sigma_{ojj}}} \qquad (C.1)$$

for the off-diagonal elements of Σ_o is 0.48, and the maximum is 0.87.

The following statistics were generated from the TIMIT CD-ROM database, and were used in the three-class simulations described in Chapter 3. The measures are duration and the first three formants from the front vowels /*iy*/, /*ih*/, /*eh*/, respectively.

μ_o =

0.0928 352 1836 2683 0.0763 399 1490 2423 0.0898 491 1426 2433

Σ_o =

0.0004	0.2664	2.0849	2.7383	0.0001	0.2576	1.3178	1.8773	0.0002	0.3642	1.1194	1.9456
0.2664	1504	3430	5093	0.1126	849	1667	2585	0.0999	789	1600	2023
2.0849	3430	33778	40709	1.1639	4789	24950	33052	1.2497	6707	22868	33329
2.7383	5093	40709	85371	1.5213	5818	28928	52187	1.8998	7527	23916	55196
0.0001	0.1126	1.1639	1.5213	0.0003	0.9012	3.5566	5.6269	0.0003	0.9641	2.9651	4.9265
0.2576	849	4789	5818	0.9012	5121	15797	25040	0.9506	5342	14672	24320
1.3178	1667	24950	28928	3.5566	15797	69784	98867	3.6917	19058	60943	95301
1.8773	2585	33052	52187	5.6269	25040	98867	189276	6.0757	30465	90424	167751
0.0002	0.0999	1.2497	1.8998	0.0003	0.9506	3.6917	6.0757	0.0005	1.4861	4.4971	7.7452
0.3642	789	6707	7527	0.9641	5342	19058	30465	1.4861	8405	21256	36541
1.1194	1600	22868	23916	2.9651	14672	60943	90424	4.4971	21256	75120	111233
1.9456	2023	33329	55196	4.9265	24320	95301	167751	7.7452	36541	111233	232220

Σ =

0.0012	0.4865	3.8267	4.3899	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.4865	2670	566	1454	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3.8267	566	38698	36534	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
4.3899	1454	36534	160898	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0006	0.5323	2.0268	2.6252	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.5323	2721	1175	4138	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	2.0268	1175	37935	22806	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	2.6252	4138	22806	130441	0.0000	0.0000	0.0000	0.0000
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0008	0.5402	1.7505	2.3834
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.5402	2340	-184.9	2965
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.7505	-184.9	27531	14523
0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	2.3834	2965	14523	138274

Appendix D

Synthetic HMM Model Parameters

The following model parameters were used in the simulations of Chapter 4.

State Transition Matrix:

$$\mathbf{A} = \begin{vmatrix} 0.45 & 0.30 & 0.25 \\ 0.33 & 0.34 & 0.33 \\ 0.35 & 0.25 & 0.40 \end{vmatrix}$$

Initial State Vector:

$$\pi = [\ 0.4 \quad 0.25 \quad 0.35\ ]$$

Output Mixture Coefficients:

$$\mathbf{B} = \begin{vmatrix} 0.15 & 0.35 & 0.50 \\ 0.30 & 0.50 & 0.20 \\ 0.10 & 0.65 & 0.25 \end{vmatrix}$$

Semi-continuous Codebook parameters:

$$\mu_o = [\ 8.0 \quad 14.0 \quad 24.0\ ]$$

$$\Sigma = [\ 5.0 \quad 12.0 \quad 9.0\ ]$$

Σ_o was derived from the desired dogmatism δ and correlation ρ, and was set to

$$\Sigma_{oij} = \begin{cases} \Sigma_{ii} / \delta^2, & i = j, \\ \rho\sqrt{\Sigma_{ii}\Sigma_{jj}} / \delta^2, & otherwise. \end{cases}$$

Table of Contents

List of Figures

List of Tables