VARIANCE COMPENSATION WITHIN THE MLLR FRAMEWORK

M.J.F. Gales & P.C. Woodland

CUED/F-INFENG/TR 242

February 1996

Cambridge University Engineering Department
Trumpington Street
Cambridge CB2 1PZ
England

Email: mjfg,pcw@eng.cam.ac.uk

Abstract

Speaker adaptation techniques try to obtain near speaker dependent (SD) performance with only small amounts speaker specific data, and are often based on initial speaker independent (SI) recognition systems. One of the key issues faced in speaker adaptation is to adapt a large number of parameters with only a small amount of data. This report examines the Maximum Likelihood Linear Regression (MLLR) technique for speaker adaptation and presents a number of enhancements to the basic MLLR approach. MLLR estimates linear transformations for the models parameters to maximise the likelihood of the adaptation data. Previously, MLLR has been applied to the mean parameters in mixture Gaussian HMM systems. In this report MLLR is extended to also adapt the Gaussian variances. Re-estimation formulae are derived for variance transforms. A second approach called Normalised Domain MLLR is also introduced that can be used efficiently for HMMs with both diagonal and full covariance matrices. A number of issues concerning the use of regression class trees in MLLR are also discussed. MLLR with variance compensation is evaluated on several large vocabulary recognition tasks. The use of mean and variance MLLR adaptation was found to give an additional 1% to 8% decrease in word error rate over mean-only MLLR adaptation.

Contents

1 Introduction

Current state-of-the-art speaker independent (SI) speech recognition systems are capable of achieving impressive performance. However for some speakers, particularly those that are not well represented by the training data, performance can be relatively poor e.g. for non-native speakers using a system trained on speech from natives. For complex speech recognition systems the amount of data to train a speaker dependent (SD) system is often excessive and hence it is very desirable to be able to improve the performance of an SI system towards SD levels while only using a small amount of speaker-specific data. This has lead to the interest in techniques for speaker adaptation.

One of the key issues is to faced in speaker adaptation is to adapt a large number of parameters with only a small amount of data. Techniques that only update distributions for which observations occur in the adaptation data, such as the use maximum a-posteriori (MAP) estimation [6, 8], require a relatively large amount of adaptation data to be effective. An alternative approach is to estimate set of transformations that can be applied to the model parameters. If these transformations can capture general relationships between the speaker independent model set and the current speaker they can be effective in adapting all the HMM distributions. One such transformation approach is maximum likelihood linear regression (MLLR) [10, 11, 13] which estimates a set of linear transformations for the mean parameters of a mixture Gaussian HMM system to maximise the likelihood of the adaptation data. It should be noted that while MLLR was initially developed for speaker adaptation, since it reduces the mismatch between a set of models and adaptation data it can also be used to perform environmental compensation by reducing a mismatch due to channel or additive noise effects.

This report extends the MLLR approach in two ways. Firstly MLLR adaptation of the Gaussian variance vectors is described. The re-estimation formulae used are derived and experiments are presented that show the utility of mean and variance MLLR-based adaptation. Due to computational requirements, standard MLLR is applicable only to HMMs with diagonal covariance matrices. An alternative MLLR-like method is introduced that can also be efficiently applied to HMMs with full covariance matrices. This is termed Normalised Domain MLLR. The transformations in MLLR are shared amongst a set of Gaussian distributions to ensure robust estimation. This sharing is based on the definition of "regression classes" and are defined using a tree structure to group all the Gaussians in the HMM system. A number of issues relating to the use of regression class trees in MLLR are presented here.

Speaker adaptation techniques may operate in a number of *modes*. If the true transcription of the adaptation data is known then it is termed supervised adaptation, whereas if the adaptation data is unlabelled the adaptation is unsupervised. Situations in which all the adaptation data is available in one block (e.g. from an system enrolment session) and the the system adapted once before use is termed static adaptation. Alternatively the data may become available as the system is used and the system adapted incrementally. The MLLR techniques described in this report are applicable to all these adaptation modes, and a number of experimental results are presented using unsupervised incremental adaptation for several large vocabulary tasks from the 1994 ARPA Continuous Speech Recognition (CSR) evaluation.

The report starts by examining several transformation approaches for speaker adaptation based on maximising the likelihood of the adaptation data. It then describes the standard MLLR adaptation of the means and the extension of the technique to adapting the variances. The adaptation techniques are then evaluated on a series of tasks, which include both clean and noise corrupted data, from the 1994 ARPA CSR evaluation.

2 Transformation Techniques in Speaker Adaptation

A number of different types of transformation have been proposed for speaker adaptation. Most attempt to modify the parameters of an SI model set to approach those of an SD set. The transformations are normally estimated to reduce the mismatch between the adaptation data and the models using either a least squares criterion (e.g. [7])[1] or a maximum likelihood (ML) criterion used in MLLR. Furthermore there are a number of options in choosing the form of the transformation and the HMM parameters to which it applies. As in any HMM training problem, it is essential to ensure that the transformation parameters are robustly estimated given the available adaptation data. One approach to ensure robust estimation is to vary the number of transformations depending on the available data so that if insufficient data is available more Gaussians will share the same transformation. This is further discussed in Section 5.

A number of different transformation types, based on the maximum likelihood criterion, have been examined in the literature. The simplest approach is to use diagonal transformation matrices with an offset vector (e.g. [2]). This approach can also be used to transform the variances [2]. The new model mean, $\hat{\mu}$, and new variance $\hat{\mathbf{\Sigma}}$ ($\mathbf{\Sigma}$ is a diagonal matrix with elements σ_{ii}^2) are given by

$$\begin{aligned} \hat{\mu}_i &= a_{ii}\mu_i + b_i && (1)\\ \hat{\sigma}_{ii}^2 &= a_{ii}^2\sigma_{ii}^2 && (2) \end{aligned}$$

where μ is the SI mean, $\mathbf{\Sigma}$ the SI variance, $\mathbf{A}$ is the diagonal transformation matrix with elements a_{ii} and $\mathbf{b}$ is the bias. Here the variance transformation, where used, is constrained to be the same as the mean transform.

MLLR [12] removes the restriction of a diagonal transformation for the means. Using a similar notation to above

$$\begin{aligned} \hat{\mu} &= \mathbf{A}\mu + \mathbf{b} && (3)\\ \hat{\mathbf{\Sigma}} &= \mathbf{\Sigma} && (4) \end{aligned}$$

This has been found to outperform the simple diagonal transformation case [13, 15]. Here the limitations are that there are no computationally efficient solutions when full covariance matrices are used in the speaker independent system, so only diagonal covariance models may be adapted. Furthermore, the variances are not modified. An alternative to using a full transformation matrix is to use a block diagonal transformation matrix [9, 15].

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_s & \mathbf{0}^n & \mathbf{0}^n \\ \mathbf{0}^n & \mathbf{A}_\Delta & \mathbf{0}^n \\ \mathbf{0}^n & \mathbf{0}^n & \mathbf{A}_{\Delta^2} \end{pmatrix} \qquad (5)$$

This reduces the number of transformation matrix parameters required to be learnt and hence reduces the adaptation data required per transform.

To try and overcome the inability to transform the variances for the full and block diagonal transformation matrix cases, the use of the Stochastic Additive Transform (SAT) [15] has been proposed. Here in addition to the linear-regression adaptation of the means, the mean and variance are transformed by

$$\begin{aligned} \hat{\mu}_i &= \mu_i + \mu_{b_i} && (6)\\ \hat{\sigma}_{ii}^2 &= \sigma_{ii}^2 + \sigma_{b_{ii}}^2 && (7) \end{aligned}$$

where the additive bias $\mathbf{b}$ has a mean, μ_b, and variance, $\mathbf{\Sigma}_b$, associated with it. Although this transform allows the variances to be modified, the resultant "variances" may not be positive-definite, as the bias variance will only be based on the distributions for which there are observations. This is not a major problem as a variance floor may be applied. However, it would be preferable to have a transformation of the variances that guarantees the likelihood does not decrease and modifies the variances in a well-motivated fashion ensuring that the resulting variances are positive.

3 Maximum Likelihood Linear Regression

The aim of MLLR is to obtain the transformation matrix that maximises the likelihood of the adaptation data. MLLR has been applied to a range of speaker adaptation tasks [9], both supervised and unsupervised, static and incremental. This section gives the basic theory for MLLR

[1] The least squares criterion is really a constrained case of the maximum likelihood criterion [9].

adaptation of the means [13] and shows how the linear regression transformation matrices and biases are trained. In addition, the memory requirements to store the statistics used to determine the transformations are discussed. The derivations and notation used in this section follow those in [9].

3.1 Estimation of the Transformation Matrix

The transformation matrix is used to give a new estimate of the mean, where

$$\hat{\mu}_m = \hat{\mathbf{W}}_m \xi_m \tag{8}$$

and $\hat{\mathbf{W}}_m$ is the $n \times (n+1)$ transformation matrix (n is the dimensionality of the data) and ξ_m is the extended mean vector

$$\xi_m = \begin{bmatrix} w & \mu_1 & \dots & \mu_n \end{bmatrix}^T \tag{9}$$

It is simple to see that

$$\hat{\mathbf{W}}_m = \begin{bmatrix} \hat{\mathbf{b}}_m & \hat{\mathbf{A}}_m \end{bmatrix} \tag{10}$$

By optimising the parameters of $\hat{\mathbf{W}}_m$ it is possible to simultaneously obtain both the required transformation matrix and bias.

In order to solve this maximisation problem an *Expectation-Maximisation* (EM) technique [1] is used. The standard auxiliary function $\mathcal{Q}(\mathcal{M}, \hat{\mathcal{M}})$ is adopted,

$$\mathcal{Q}(\mathcal{M}, \hat{\mathcal{M}}) = \tag{11}$$

$$K_1 - \frac{1}{2}\mathcal{L}(\mathbf{O}_T|\mathcal{M}) \sum_{m=1}^{M}\sum_{\tau=1}^{T} L_m(\tau)\left[K_m + \log(|\mathbf{\Sigma}_m|) + (\mathbf{o}(\tau) - \hat{\mu}_m)^T \mathbf{\Sigma}_m^{-1}(\mathbf{o}(\tau) - \hat{\mu}_m)\right]$$

where K_1 is a constant dependent only on the transition probabilities, K_m is the normalisation constant associated with Gaussian component m, $\mathbf{O}_T = \{\mathbf{o}(1), \dots, \mathbf{o}(T)\}$ is the adaptation data and

$$L_m(\tau) = p(q_m(\tau)|\mathcal{M}, \mathbf{O}_T) \tag{12}$$

where $q_m(\tau)$ indicates Gaussian component m at time τ. Increasing the value of this auxiliary function is guaranteed to increase the likelihood of the adaptation data. To enable robust transformations to be trained the transformation matrices are tied across a number of Gaussians (a transformation per Gaussian is equivalent to conventional re-training of the means). The set of Gaussians that share a transform are referred to as a *regression class*. The selection of regression classes depends on the available adaptation data and uses a regression class tree. This tree contains all the Gaussians in the system and statistics are gathered at the leaves (which may each contain a number of Gaussians and defines the *base classes*). The most specific transform that can be robustly estimated is then generated for all the Gaussians in the system. This is described in further detail in section 5.

A particular transformation $\hat{\mathbf{W}}_m$ will be tied across R Gaussian components, $\{m_1, \dots, m_R\}$, as determined by the regression class tree. For the Gaussian output probability density functions considered, $\hat{\mathbf{W}}_m$ may be found by solving

$$\sum_{\tau=1}^{T}\sum_{r=1}^{R} L_{m_r}(\tau)\mathbf{\Sigma}_{m_r}^{-1}\mathbf{o}(\tau)\xi_{m_r}^T = \sum_{\tau=1}^{T}\sum_{r=1}^{R} L_{m_r}(\tau)\mathbf{\Sigma}_{m_r}^{-1}\hat{\mathbf{W}}_m\xi_{m_r}\xi_{m_r}^T \tag{13}$$

For the full covariance matrix case this is computationally very expensive[2], however, for the diagonal covariance matrix case a closed-form solution is computationally feasible [13].

The left-hand side of equation 13 is independent of the transformation matrix and will be referred to as $\mathbf{Z}$, where

$$\mathbf{Z} = \sum_{r=1}^{R}\sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{\Sigma}_{m_r}^{-1}\mathbf{o}(\tau)\xi_{m_r}^{T} \tag{14}$$

A new variable $\mathbf{G}^{(i)}$ is defined with elements

$$g_{jq}^{(i)} = \sum_{r=1}^{R} v_{ii}^{(r)} d_{jq}^{(r)} \tag{15}$$

where

$$\mathbf{V}^{(r)} = \sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{\Sigma}_{m_r}^{-1} \tag{16}$$

and

$$\mathbf{D}^{(r)} = \xi_{m_r}\xi_{m_r}^{T}. \tag{17}$$

$\hat{\mathbf{W}}_m$ is calculated using

$$\hat{\mathbf{w}}_i^T = \mathbf{G}^{(i)-1}\mathbf{z}_i^T \tag{18}$$

where $\hat{\mathbf{w}}_i$ is the i^{th} vector of $\hat{\mathbf{W}}_m$ and $\mathbf{z}_i$ is the i^{th} vector of $\mathbf{Z}$.

3.2 Statistics Required to Estimate Transformation

It is interesting to examine the memory requirements for calculating the mean transformation matrices. The statistics to compute the transformations may be stored either at the Gaussian component level or at a regression class level. The most memory efficient technique is dependent on the ratio of the regression classes to the number of components. In addition, it will depend on the form of the transformation sequence described in Section 5.

1. **Gaussian Component Level**. This requires $\sum_{\tau=1}^{T} L_m(\tau)\mathbf{o}(\tau)$ and $\sum_{\tau=1}^{T} L_m(\tau)$ to be stored, a cost of $(n+1)$ floats[3] per component.

2. **Regression Class Level**. The statistics, $\mathbf{G}^{(i)}$ and $\mathbf{Z}$ may be stored at the regression-class level. This has a memory cost of n sets of $(n+1)\times(n+1)$ symmetric matrices and one $n\times(n+1)$ full matrix, a cost for each regression class of $\mathcal{O}(n^3)$. This assumes that the regression classes have been pre-defined. The regression-class tree may be used to generate classes dynamically [9], so it is not known a-priori which regression classes will be used to estimate the transform. This is not a problem, as $\mathbf{G}^{(i)}$ and $\mathbf{Z}$ for the chosen regression class may be obtained from its child classes. Letting the parent node R have children $\{R_1,\ldots,R_C\}$ then

$$\mathbf{Z} = \sum_{c=1}^{C} \mathbf{Z}^{(R_c)} \tag{19}$$

[2]The closed-form solution requires solving $n\times(n+1)$ simultaneous equations of the form

$$z_{kl} = \sum_{p=1}^{n}\sum_{q=1}^{n+1} \hat{w}_{pq}\left(\sum_{r=1}^{R} v_{kp}^{(r)} d_{ql}^{(r)}\right)$$

for $k=1\ldots n, l=1\ldots(n+1)$, where $\mathbf{Z}$, $\mathbf{V}^{(r)}$ and $\mathbf{D}^{(r)}$ are defined in equations 14, 16 and 17 respectively.

[3]A float is used as the unit of storage.

where

$$\mathbf{Z}^{(R_c)} = \sum_{r=1}^{R_c} \sum_{\tau=1}^{T} L_{m_r}(\tau) \boldsymbol{\Sigma}_{m_r}^{-1} \mathbf{o}(\tau) \xi_{m_r}^T \tag{20}$$

Similarly

$$\mathbf{G}^{(i)} = \sum_{c=1}^{C} \mathbf{G}^{(iR_c)} \tag{21}$$

where

$$g_{jq}^{(iR_c)} = \sum_{r=1}^{R_c} v_{ii}^{(r)} d_{jq}^{(r)} \tag{22}$$

It is clear that it is only necessary to store data for each transform at the most specific regression classes possible. This set of regression classes are called the *base classes* which form the leaves of the regression class tree.

3.3 Multiple Iterations of MLLR

As MLLR is dependent on the frame/state component alignment, $L_m(\tau)$ in equation 11, performance can sometimes be improved using multiple iterations of MLLR [13]. There are additional implementation issues when multiple iterations are used, particularly if the use of the regression class tree changes. As shown in Appendix B, for a given regression class it is unimportant how many times the means have been transformed, the final transformation will always yield the same value irrespective of whether the original or the latest model set is transformed, provided the frame/state component alignment is the same. However if the regression class tree is used dynamically, the situation may occur where due to changes in the alignments there is insufficient data to generate a transformation for a specific class. There is now no way to compensate for the transform previously applied to that class[4]. By always transforming the original model parameters this problem of building errors into the system will never occur.

4 MLLR Adaptation of the Variances

This section describes the basis of a transformation of the Gaussian variances in the MLLR framework. The means and variances are adapted in two separate stages. Initially new means are found and then, given these new means, the variances are also updated.

4.1 Estimation of the Transformation Matrix

The HMMs are modified in two steps such that

$$\mathcal{L}(\mathbf{O}_T|\check{\mathcal{M}}) \geq \mathcal{L}(\mathbf{O}_T|\hat{\mathcal{M}}) \geq \mathcal{L}(\mathbf{O}_T|\mathcal{M}) \tag{23}$$

where the models $\hat{\mathcal{M}}$ have just the means parameters updated to $\hat{\mu}_1, \ldots, \hat{\mu}_M$ and the models $\check{\mathcal{M}}$ have both the means and the variances $\hat{\boldsymbol{\Sigma}}_1, \ldots, \hat{\boldsymbol{\Sigma}}_M$ updated.

The Gaussian variance vectors are updated by

$$\hat{\boldsymbol{\Sigma}}_m = \mathbf{B}_m^T \hat{\mathbf{H}}_m \mathbf{B}_m \tag{24}$$

[4] Although the original parameters are transformed, the frame/state alignments are found using the latest model set. This needs both model sets to be kept in memory which adds additional memory requirements.

where $\hat{\mathbf{H}}_m$ is the linear transformation to be estimated and $\mathbf{B}_m$ is the inverse of the Choleski factor of $\mathbf{\Sigma}_m^{-1}$, so

$$\mathbf{\Sigma}_m^{-1} = \mathbf{C}_m \mathbf{C}_m^T \tag{25}$$

and

$$\mathbf{B}_m = \mathbf{C}_m^{-1} \tag{26}$$

The same auxiliary function as that used for the standard ML estimation of the HMM parameters is employed

$$\begin{aligned}&\mathcal{Q}(\mathcal{M}, \check{\mathcal{M}}) = \\ &\quad K_1 - \frac{1}{2}\mathcal{L}(\mathbf{O}_T|\mathcal{M}) \sum_{m=1}^{M} \sum_{\tau=1}^{T} L_m(\tau) \left[K_m + \log(|\hat{\mathbf{\Sigma}}_m|) + (\mathbf{o}(\tau) - \hat{\mu}_m)^T \hat{\mathbf{\Sigma}}_m^{-1} (\mathbf{o}(\tau) - \hat{\mu}_m)\right]\end{aligned} \tag{27}$$

It is hard to directly optimise this expression for both the mean transformation matrix and the variance transform. However it is sufficient to ensure that

$$\mathcal{Q}(\mathcal{M}, \check{\mathcal{M}}) \geq \mathcal{Q}(\mathcal{M}, \hat{\mathcal{M}}) \tag{28}$$

to satisfy equation 23.

Rewriting equation 27 using equations 25 and 24 leads to

$$\begin{aligned}\mathcal{Q}(\mathcal{M}, \check{\mathcal{M}}) = K_1 - \frac{1}{2}\mathcal{L}(\mathbf{O}_T|\mathcal{M}) \sum_{m=1}^{M} \sum_{\tau=1}^{T} L_m(\tau) \Big[K_m + \log(|\mathbf{\Sigma}_m|) + \log(|\hat{\mathbf{H}}_m|) \\ + (\mathbf{C}_m^T \mathbf{o}(\tau) - \mathbf{C}_m^T \hat{\mu}_m)^T \hat{\mathbf{H}}_m^{-1} (\mathbf{C}_m^T \mathbf{o}(\tau) - \mathbf{C}_m^T \hat{\mu}_m)\Big]\end{aligned} \tag{29}$$

The maximisation of equation 29 has a simple closed form solution and leads to a re-estimation formula for $\hat{\mathbf{H}}_m$ analogous to the standard ML estimate of the covariance matrix

$$\begin{aligned}\hat{\mathbf{H}}_m &= \frac{\sum_{\tau=1}^{T} L_m(\tau)(\mathbf{C}_m^T \mathbf{o}(\tau) - \mathbf{C}_m^T \hat{\mu}_m)(\mathbf{C}_m^T \mathbf{o}(\tau) - \mathbf{C}_m^T \hat{\mu}_m)^T}{\sum_{\tau=1}^{T} L_m(\tau)} \\ &= \frac{\mathbf{C}_m^T \left[\sum_{\tau=1}^{T} L_m(\tau)(\mathbf{o}(\tau) - \hat{\mu}_m)(\mathbf{o}(\tau) - \hat{\mu}_m)^T\right] \mathbf{C}_m}{\sum_{\tau=1}^{T} L_m(\tau)}\end{aligned} \tag{30}$$

Up to this point the tying of the variance transformation matrices has been ignored. However, if the transform is to be shared over a number of Gaussian components, $\{m_1, \ldots, m_R\}$, then the re-estimation formula becomes

$$\hat{\mathbf{H}}_m = \frac{\sum_{r=1}^{R} \left\{\mathbf{C}_{m_r}^T \left[\sum_{\tau=1}^{T} L_{m_r}(\tau)(\mathbf{o}(\tau) - \hat{\mu}_{m_r})(\mathbf{o}(\tau) - \hat{\mu}_{m_r})^T\right] \mathbf{C}_{m_r}\right\}}{\sum_{r=1}^{R} \sum_{\tau=1}^{T} L_{m_r}(\tau)} \tag{31}$$

It is preferable to obtain all the transformation statistics, both for the mean and variance adaptation, in a single pass. Hence, as $\hat{\mu}_m$ is not known when the statistics are being accumulated, it is necessary to rearrange this expression as

$$\hat{\mathbf{H}}_m = \frac{\sum_{r=1}^{R} \left\{\mathbf{C}_{m_r}^T \left[\sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau)\mathbf{o}(\tau)^T - \hat{\mu}_{m_r}\overline{\mathbf{o}}_{m_r}^T - \overline{\mathbf{o}}_{m_r}\hat{\mu}_{m_r}^T + \hat{\mu}_{m_r}\hat{\mu}_{m_r}^T \sum_{\tau=1}^{T} L_{m_r}(\tau)\right] \mathbf{C}_{m_r}\right\}}{\sum_{m=1}^{R} \sum_{\tau=1}^{T} L_{m_r}(\tau)} \tag{32}$$

where

$$\overline{\mathbf{o}}_{m_r} = \sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau) \tag{33}$$

The estimate of $\hat{\mathbf{H}}_m$ defined in equation 32 results in a full covariance matrix, yielding a full covariance matrix for the new estimate of the covariance, $\hat{\mathbf{\Sigma}}_{m_r}$, even if the original covariance matrices were diagonal. However, due to the derivation of these matrices, it is not necessary to store a full symmetric covariance matrix for each individual component. The original diagonal covariances may be left unchanged and the likelihood calculated as

$$\begin{aligned}\mathcal{L}(\mathbf{o}(\tau)|\mathcal{M}_{m_r}) = K_{m_r} - \frac{1}{2}\Big(&\log(|\mathbf{\Sigma}_{m_r}|) + \log(|\hat{\mathbf{H}}_m|) \\ &+(\mathbf{C}_{m_r}^T\mathbf{o}(\tau) - \mathbf{C}_{m_r}^T\hat{\mu}_{m_r})^T\hat{\mathbf{H}}_m^{-1}(\mathbf{C}_{m_r}^T\mathbf{o}(\tau) - \mathbf{C}_{m_r}^T\hat{\mu}_{m_r})\Big)\end{aligned} \tag{34}$$

Alternatively $\hat{\mathbf{H}}_m$ may be forced to be a diagonal transformation by setting the off-diagonal elements to zero which results in $\hat{\mathbf{\Sigma}}_m$ being a diagonal covariance matrix. This is still guaranteed to increase the likelihood of the adaptation data. If these diagonal transformations are used then the number of transformation parameters is small, n, compared to the number typically used to adapt the means, $(n+1)\times n$.

4.2 Statistics Required to Estimate Variance Transformations

The statistics for calculating the variance transformation may, again, be stored at either the Gaussian component level or at the regression class level.

1. **Gaussian Component Level**. In addition to the statistics required to estimate the mean transformation it is necessary to store $\sum_{\tau=1}^{T} L_m(\tau)\mathbf{o}(\tau)\mathbf{o}(\tau)^T$. If a full covariance transformation, $\hat{\mathbf{H}}_m$, is to be calculated, then a full symmetric matrix must be stored for each component. For many systems this is impractical, as it has a memory requirement of $\mathcal{O}(n^2)$ per Gaussian.

2. **Regression Class Level** The statistics may be stored at the regression class level. There are two options available. Firstly all statistics to estimate both the mean and variance transformation matrices are accumulated in a single pass of the adaptation data. Alternatively, two passes of the adaptation data is used. In the first pass the statistics for the mean transformation are obtained. In the second pass of the adaptation data the statistics to obtain the variance transform are accumulated.

   (a) **Single-Pass**. To accumulate the statistics to estimate the transformation matrices for both means and variances in a single pass is complex as the new estimate of the mean, $\hat{\mu}_{m_r}$, is not known when the transformation statistics are required to be collected. Hence, it is necessary to modify the elements in equation 32 so that they are independent of the mean transformation, $\hat{\mathbf{W}}_m$. This is explained in Appendix A and has a memory requirement of $\mathcal{O}(n^3)$ per regression class with the additional constraint that the original covariance matrices are diagonal.

   (b) **Two-Pass**. Statistics to calculate the transformation matrix for the means are obtained in the first pass. In the second pass, the mean transformation is known, so it is only necessary to store

   $$\mathbf{Z}^{(R_c)} = \sum_{r=1}^{R_c} \mathbf{C}_{m_r}^T\left[\sum_{\tau=1}^{T} L_{m_r}(\tau)(\mathbf{o}(\tau)-\hat{\mu}_{m_r})(\mathbf{o}(\tau)-\hat{\mu}_{m_r})^T\right]\mathbf{C}_{m_r} \tag{35}$$

   and the regression class occupancy to calculate equation 31. This has a memory requirement of $\mathcal{O}(n^2)$ per regression class, but is more computationally expensive as a second pass through the adaptation data is required.

The choice of where statistics are to be stored is dependent on the number of Gaussian components compared to the number of regression classes and the allowable computational load for the adaptation. In addition, it will depend on the form of the transformation sequence described in Section 5.

The ML estimate of the mean transformation matrix, equation 13, is a function of the current estimate of the covariance matrix. Thus the ML estimate of $\hat{\mathbf{W}}_m$ will alter as the variances change. If $\hat{\mathbf{H}}_m$ is made diagonal, it is possible to iteratively improve the likelihood. One of the drawbacks of the variance transformation described is that it does not simultaneously optimise the mean and the variance transformations. Therefore this iterative scheme is required to obtain a "true" ML estimate of the mean and variance transformations.

5 Regression Class Trees

MLLR uses a regression class tree [13] to order the Gaussians in the system so that the set transformations to be estimated can be chosen according to the amount and type of adaptation data available. The optimal method for selecting the regression classes, in the sense of maximising the auxiliary function, would be to cluster the Gaussian components such that they maximise the average auxiliary function for all "training" speakers. Consider a binary regression class tree. For a possible split of R into $\{R_1, R_2\}$ using data from a set of training speakers, $\mathbf{S} = \{S_1, \ldots, S_S\}$ it is necessary to optimise

$$\{R_1, R_2\} = \arg\max_{\mathbf{R}} \left\{ \sum_{s=1}^{S} \mathcal{Q}(\mathcal{M}, \hat{\mathcal{M}}_{s,r}) \right\} \tag{36}$$

where $\hat{\mathcal{M}}_{s,r}$ is the model set using component grouping r out of all possible groupings $\mathbf{R}$ of the components and the training data from speaker s. Unfortunately this is not usually computationally feasible. It is therefore necessary to make some simplifying assumptions. The assumption used in the original MLLR work [9], is that components that are close together in acoustic space should be transformed in the same way. This alters the problem from selecting regression classes into one of clustering the Gaussian components. This is the assumption used to generate the regression class trees for all the experiments reported in Section 7.

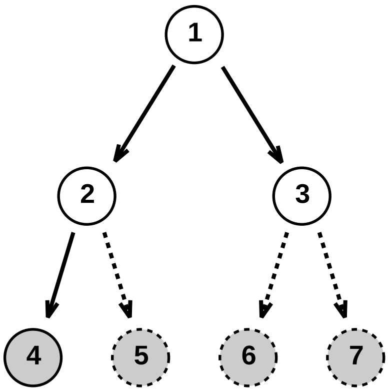

Figure 1: A binary four base class regression class tree

A simple binary regression class tree with four base classes is shown in figure 1. Usually the base classes represent a set of acoustically similar Gaussians.[5] In the diagram a solid arrow and class indicates that there is sufficient data for a transformation matrix to be generated using the

[5]In the limit, of course, there is a separate base class per-Gaussian. However this requires a large number of statistics to be stored if the regression class tree is to be used dynamically.

data associated with that class, a dotted line and class indicates insufficient data. Two methods of traversing the regression class tree are considered, a *top-down* approach and a *bottom-up* approach.

5.1 Top-Down Traversal

A simple top-down scheme starts the search at the root node and then:

For a node A
    generate transform **W** *from data for A*
    update all components in A using **W**
    if A has children B and C
        if B has sufficient data
            search through node B
        if C has sufficient data
            search through node C

For a particular search through the tree there are some implementation issues. For simplicity, only mean adaption will be considered here, but the same options are also available for the variance adaption. For the regression class tree in Figure 1 traversing the regression tree using the above method results in the following transformation sequence.

$$\left\{ \begin{array}{lcl} D_1 & \rightarrow & \{C_4, C_5, C_6, C_7\} \\ D_2 & \rightarrow & \{C_4, C_5\} \\ D_4 & \rightarrow & \{C_4\} \\ D_3 & \rightarrow & \{C_6, C_7\} \end{array} \right\} = \left\{ \begin{array}{lcl} D_4 & \rightarrow & \{C_4\} \\ D_2 & \rightarrow & \{C_5\} \\ D_3 & \rightarrow & \{C_6, C_7\} \end{array} \right\} = T_1 \tag{37}$$

where D_j denotes the transformation statistics, $\mathbf{G}^{(i)}$ and $\mathbf{Z}$, associated with node j and C_j indicates the set of components associated with regression class, or node, j. Thus in the diagram $D_2 = \{D_4, D_5\}$ and $C_2 = \{C_4, C_5\}$. Examining the transformation sequence, repeated transformations, such as the case for C_4, are unimportant as only the most specific transformation determines the final mean value (see Appendix B). Thus if the transformation data is based on the original means, only the last transform is important and the series of transforms reduces to the original search technique described.

It is not necessary to transform the original model means: for instance the transformation may be based on the mean after a transformation of the parent node. In this case, both the mean to be adapted and the data to calculate the transform will be altered. Let $\hat{D}_i^j$ indicate the data obtained using the latest estimates of the means of Gaussians C_i after they have been transformed at node j, i.e. $\hat{C}_i^j$. A particular mean may be transformed many times, for example $\hat{C}_i^{j,k}$ indicates that the components have been transformed at regression class j and then at regression class k. As shown in Appendix B all top-down transformations will yield this same transformation sequence, T_1.

5.2 Bottom-Up Traversal

A simple bottom-up scheme starts the search at a leaf node and then:

For a node A
    If A has sufficient data
        generate transform **W** *from data for A*
        update all components in A using **W**
    else
        search through parent node B

Bottom-up methods of traversing the regression class tree which are based on the original means also yield the same transform, T_1, provided each Gaussian component is transformed only once as described (see Appendix B). A bottom-up transform using the original means and data where each component may be transformed many times, ie the parent node is transformed as well, will in the limit be equivalent to a single global transformation at the root node. This transformation sequence is

$$\{\ D_1 \quad \rightarrow \quad \{C_4, C_5, C_6, C_7\}\ \} = T_2 \tag{38}$$

For bottom-up schemes using the latest estimate of the mean the transformation sequence is

$$\left\{ \begin{array}{lll} D_4 & \rightarrow & \{C_4\} \\ \hat{D}_2^4 & \rightarrow & \{\hat{C}_4^4, C_5\} \\ D_3 & \rightarrow & \{C_6, C_7\} \\ \hat{D}_1^{423} & \rightarrow & \{\hat{C}_4^{4,2}, \hat{C}_5^2, \hat{C}_6^3, \hat{C}_7^3\} \end{array} \right\} = T_3 \tag{39}$$

Of course it is not necessary to perform the last transformation, as all the means have already been modified, however it is still guaranteed to increase the likelihood. The case where the final transform is not used will be referred to as transform T_4. Thus

$$\left\{ \begin{array}{lll} D_4 & \rightarrow & \{C_4\} \\ \hat{D}_2^4 & \rightarrow & \{\hat{C}_4^4, C_5\} \\ D_3 & \rightarrow & \{C_6, C_7\} \end{array} \right\} = T_4 \tag{40}$$

The statistics for estimating the transforms when either a T_3 or T_4 transformation sequence is required must be stored at the component level, not the regression base class level. This is because both $\mathbf{G}^{(r)}$ and $\mathbf{Z}^{(r)}$ are dependent on the current estimate of the mean. Thus, if latest estimates are used in the transformation, e.g. using $\hat{D}_2^4$, the regression class level statistics will alter.

There are various different "assumptions" being made when applying these transforms. For example using T_1 it is assumed that the transform for C_5 is similar to that of C_4. Alternatively, using T_4 it is assumed that a safer transformation for C_5 is to smooth its adaptation data with the identity matrix, as C_4 has already been adapted. This is a "safer" transform, but is unlikely to achieve the same improvement in the auxiliary function.

The method used to traverse the regression class tree for the variances need not be the same as that used for the means, nor the same threshold used.

For the experiments described in section 7 only transformation sequences T_1 and T_4 are considered.

6 Normalised Domain MLLR

The scheme described in the previous section only allows adaptation of models with diagonal covariance matrices in a computationally efficient manner. To extend the MLLR framework to handle Gaussians with full covariance matrices efficiently, the transformation must be modified.

An alternative linear regression transform of the mean is given by

$$\hat{\mu}_m = \mathbf{C}_m^{T-1} \tilde{\mathbf{A}}_m \mathbf{C}_m^T \mu_m + \mathbf{b}_m \tag{41}$$

where again

$$\mathbf{\Sigma}_m^{-1} = \mathbf{C}_m \mathbf{C}_m^T \tag{42}$$

This form of transformation is similar to the standard MLLR transform except that the transformation is applied in a *normalised* domain, in which all Gaussians have the same, identity matrix, covariance matrix.

Consider the Gaussian component mean in the normalised domain, $\tilde{\mu}$. The new adapted mean in the normalised domain, $\check{\mu}_m$, is found by

$$\check{\mu}_m = \tilde{\mathbf{A}}_m \tilde{\mu}_m + \tilde{\mathbf{b}}_m = \tilde{\mathbf{W}}_m \tilde{\xi}_m \tag{43}$$

where

$$\tilde{\xi}_m = [\ w \quad \tilde{\mu}_1 \quad \dots \quad \tilde{\mu}_n\ ]^T \tag{44}$$

and

$$\tilde{\mu}_m = \mathbf{C}_m^T \mu_m \tag{45}$$

Using this new modified transform the auxiliary function, equation 11, may be written as

$$\begin{aligned} \mathcal{Q}(\mathcal{M}, \hat{\mathcal{M}}) = K_1 - \frac{1}{2}\mathcal{L}(\mathbf{O}_T|\mathcal{M}) \sum_{m=1}^{M}\sum_{\tau=1}^{T} L_m(\tau) \Big[K_m + \log(|\mathbf{\Sigma}_m|) & \qquad (46) \\ +(\mathbf{C}_m^T\mathbf{o}(\tau) - \tilde{\mathbf{W}}_m\tilde{\xi}_m)^T(\mathbf{C}_m^T\mathbf{o}(\tau) - \tilde{\mathbf{W}}_m\tilde{\xi}_m)\Big] & \end{aligned}$$

Noting that $\log(|\mathbf{\Sigma}_m|)$ is not a function of $\tilde{\mathbf{W}}_m$, the optimisation is identical to the standard MLLR case, except that all covariance matrices are now the identity matrix. Again tying the transformations over R components $\{m_1, \dots, m_R\}$, equation 13 becomes

$$\begin{aligned} \sum_{\tau=1}^{T}\sum_{r=1}^{R} L_{m_r}(\tau)\mathbf{C}_{m_r}^T\mathbf{o}(\tau)\tilde{\xi}_{m_r}^T &= \sum_{\tau=1}^{T}\sum_{r=1}^{R} L_{m_r}(\tau)\tilde{\mathbf{W}}_m\tilde{\xi}_{m_r}\xi_{m_r}^T \\ &= \tilde{\mathbf{W}}_m \sum_{\tau=1}^{T}\sum_{r=1}^{R} L_{m_r}(\tau)\tilde{\xi}_{m_r}\tilde{\xi}_{m_r}^T \qquad (47) \end{aligned}$$

The optimisation is similar to the least squares regression problem [7]. This is straightforward to solve, provided that the number of Gaussian components assigned to the particular class is greater than the dimensionality of the feature vector plus one. It should be noted that for this case there are effectively different transforms for every Gaussian component.

Though this form of transformation appears attractive, as it extends the class of models that may be easily adapted using MLLR, its performance in preliminary experiments was poor compared to the standard MLLR form. The problem with this transformation is felt to be the uneven nature of the errors that result from its application. For components with large variances, $\mathbf{C}_m$ will be very small. Thus in the normalised domain, where the transformation $\tilde{\mathbf{W}}_m$ is calculated, the mean will be small. When mapped back to the standard domain, errors in $\tilde{\mu}_m$ will be dramatically increased, as $\mathbf{C}_m^{-1}$ will be correspondingly large. So for components with large variances and little or no adaptation data, the transformation may be very poor. This problem may be partially overcome by altering the clustering to form the regression class tree. However no experimental results with the normalised domain MLLR are given in this report.

7 Experiments and Results

To evaluate the variance adaption scheme the ARPA 1994 CSRNAB development and evaluation data was used. A variety of tasks, with data recorded in both clean[6] and noise corrupted environments, were examined. For all the tasks selected, the results were required causally and the transcription of the adaptation data was unknown. Thus unsupervised incremental adaptation was used on a per-speaker level, the speaker boundaries were known.

1. **Spoke 4: Incremental Speaker Adaptation**. This is a 5k task recorded in a clean environment with around one hundred sentences per speaker. The task is to perform unsupervised incremental adaptation with a relatively large amount of adaptation data per speaker.

[6] Here the term "clean" refers to the training and test conditions being from the same microphone type with a high signal-to-noise ratio.

2. **Hub 1: Unlimited Vocabulary NAB News Baseline**. This is an unlimited vocabulary task with approximately 15 sentences per-speaker. The data was recorded in a clean environment.

3. **Spoke 10: Noisy Channel**. The S10 spoke is a 5k task with car noise artificially added onto clean speech. Three noise levels are given, however only one was evaluated for this report. Approximately ten sentences were uttered by each speaker.

4. **Spoke 5: Microphone Independence**. This task uses data recorded with "unknown" microphones. It is a 5k vocabulary task with about twenty sentences uttered by each speaker.

The baseline system used for the recognition task was a gender-independent cross-word-triphone mixture-Gaussian tied-state HMM system and was the same as the "hmm-1" model set used in the CUED HTK 1994 evaluation system [18]. The set of speech parameters consisted of 12 MFCCs, C_1 to C_{12}, along with normalised log-energy and the first and second differentials of these parameters. This yielded a 39-dimensional feature vector. The acoustic training data consisted of 36493 sentences from the SI-284 WSJ0 and WSJ1 sets, and the LIMSI WSJ lexicon and phone set were used. The standard HTK system was trained using decision-tree-based state clustering [19] to define 6399 speech states. A 12 component mixture Gaussian distribution was then trained for each tied state, a total of about 6 million parameters. For the all the spoke tasks, S4, S5 and S10, the standard MIT Lincoln Labs 5k trigram language model was specified. Decoding used the single-pass dynamic-network decoder [16]. For the H1 task a 65k word list and dictionary was used.

For all the noise corrupted tasks, S5 and S10, the model set parameters were initially modified using parallel model combination (PMC) [4]. For computational efficiency the PMC Log-Add approximation [4] with simple convolutional noise estimation [5] was used to modify the means of the models. PMC cannot be applied to models built with the standard HTK front-end: C_0 replaces normalised log-energy and simple differences, not linear regression, are used to generate the dynamic parameters. Furthermore, Cepstral mean normalisation (CMN) is not used for the PMC models. As CMN was not applied, it was necessary to first compensate for the different global signal levels of the WSJ0 and WSJ1 databases. Therefore, the C_0 feature vector coefficient of the WSJ0 data was offset to have the same average value as the WSJ1 database. To generate the PMC models, the standard HTK model set was initially estimated as described above. The model set was then altered in a single pass [3] to be based on the PMC parameter set. An additional smoothing pass of Baum-Welch re-estimation was also performed.

So that meaningful comparison to be made with other published results, all the experiments described here were implemented using unsupervised incremental adaptation. Furthermore, none of the systems were optimised for the test data. The grammar scale factors and insertion penalties were determined by the standard clean speaker independent system, or in the case of the PMC model sets, by optimising them on the ARPA 1994 CSRNAB S0 development data [3].

For all tasks full transformation matrices for the means and diagonal transforms for the variances were used. The transformation sequence for the variances was the same as that for the means and the minimum class occupancy counts were also set to be the same for both the mean and variance transformation matrices. The regression class trees used throughout this work were based on clean speech and did not transform the silence models. The model parameters were updated after every additional two sentences of adaptation data.

7.1 Spoke 4: Incremental Speaker Adaptation

Figure 2 shows how the auxiliary function value of the adaptation data varies with the number of adaptation updates for a MLLR mean adapted model set and a MLLR mean and variance adapted system[7]. From the graph it can be seen that the use of variance adaption, at least in terms of the the auxiliary function, showed a distinct improvement over the standard mean adaptation case,

[7] Since this is an incremental task the alignments for the mean adapted and the mean-and-variance adapted systems will be different after the first model update.

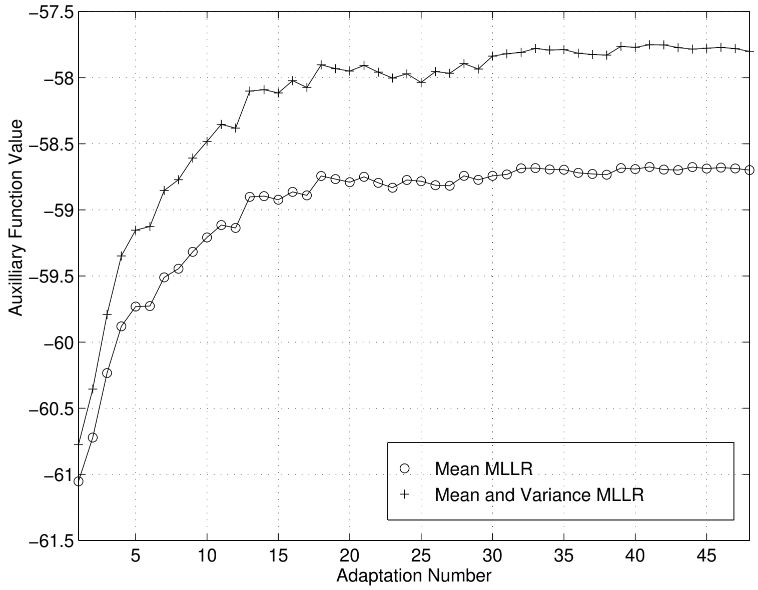

Figure 2: Auxiliary function value obtained using the T_4 transformation with and without adapting the variances for speaker 4tb

-57.8 compared with -58.7. Since the system is being run in an incremental adaptation mode the likelihoods do not increase monotonically as new adaptation data is added at each update. Figure 2 shows that the variance adaption was increasing the likelihood of the adaptation data, but this does not necessarily indicate a reduction in word error rate on the test data.

Transform		Speaker				Average
		4tb	4tc	4td	4te	
None [13]		5.6	6.3	14.2	4.8	7.7
Mean	T_1	5.1	5.8	12.1	4.0	6.7
	T_4	4.7	5.8	12.2	4.1	6.7
Mean and	T_1	4.7	5.5	12.1	4.1	6.6
Variance	T_4	4.3	5.5	12.1	4.1	6.5

Table 1: Incremental adaptation results on the S4 evaluation data

Table 1 shows the word error rate for the four speakers[8]. The use of mean MLLR adaptation reduced the error rate by 13%. A further decrease of 3% was obtained by adapting the variances. A similar small gain in performance, 2%, was obtained with variance compensation using the T_1 transformation set. This transformation order type T_1 had a slightly higher error rate than the T_4 transformation type.

7.2 Hub 1: Unlimited Vocabulary NAB News Baseline

MLLR can improve the performance using a very small amount of adaptation data [12]. MLLR variance adaption was therefore examined on the H1 task, where relatively few adaptation sentences

[8]The results given here differ from those submitted for the CSRNAB 1994 ARPA S4 evaluation as for time reasons the recognition was performed using lattices generated by the unadapted system and the adaptation was performed every other sentence as opposed to every sentence for the evaluation. These differences will affect the absolute performance levels, but it is felt that the relative performances will still be valid

were available, approximately 15 per speaker[9].

Transform Set	Comp Means	Comp Variances	Error (%) Development	Error (%) Evaluation
None [18]	$\times$	$\times$	9.5	9.2
T_1	$\surd$	$\times$	8.0	8.3
	$\surd$	$\surd$	7.9	8.1
T_4	$\surd$	$\times$	8.2	8.3
	$\surd$	$\surd$	7.9	8.2

Table 2: Incremental adaptation results on H1 development and evaluation data

Table 2 shows the results on both the development and evaluation data for the H1 task. Again, the use of both mean and mean-and-variance adaptation with MLLR improved the performance. On the development data mean adaptation yielded about a 15% reduction in error rate and variance adaption a further 2% reduction. The improvements on the evaluation data were smaller than 10% reduction in error rate using mean adaptation MLLR and a further 1% adapting the variances with the T_4 transformation set. With the T_1 transformation set a 2% gain in performance was obtained over the mean compensation case.

7.3 Spoke 10: Noisy Channel

Transform Set	Comp Means	Comp Variances	Error (%)
None [3]	$\times$	$\times$	10.7
T_1	$\surd$	$\times$	9.3
	$\surd$	$\surd$	8.9
T_4	$\surd$	$\times$	9.5
	$\surd$	$\surd$	9.1

Table 3: Incremental adaptation results on S10 evaluation Level 3 data

The use of variance adaption was then examined for an additive noise task the ARPA CSRNAB 1994 S10 data. Car noise was artificially added to clean speech at various signal-to-noise ratios (SNRs). A total of just over one hundred sentences from ten speakers was specified as the test set. For the actual evaluation three SNRs were given. For this work only the *Level 3* noise condition was considered. This is the noisiest condition and has a SNR of approximately 10dB.

Initially, PMC Log-Add adaption [4] with simple convolutional noise estimation [5] was applied. This compensation technique only modifies the means of the model set and gave a word error rate of 10.7%. The means of this PMC adapted model set were then adapted using MLLR mean adaptation in an unsupervised incremental fashion. Adapting the means reduced the error rate by 13% for the T_1 transformation set. The test was then repeated adapting both the means and the variances. Adapting the variances further reduced the error rate by 4%. Using the T_4 transformation set the use of MLLR had a smaller effect, 11% simply adapting the means, though again, the use of variance adaption reduced the error rate by 4%. The performance of the T_1 transform on this task may be seen to be better than that of the T_4 transform, though in both cases variance adaption showed some improvement. This indicates, not surprisingly, that the choice of transform has some

[9]Again the results presented here cannot be directly compared with those of published for the evaluation. Here only a triphone model set with a tri-gram language model was used, compared with a 4-gram language model and a quin-phone model set in the evaluation.

effect on the system performance. It is interesting to compare these results with those of SRI [14], 12.2% [17], and the best PMC result, adapting both means and variance, of 10.1% [3].

7.4 Spoke 5: Microphone Independence

Transform Set	Comp Means	Comp Variances	Error (%)
None	$\times$	$\times$	10.6
T_1	$\surd$	$\times$	8.6
	$\surd$	$\surd$	8.0
T_4	$\surd$	$\times$	8.7
	$\surd$	$\surd$	8.0

Table 4: Incremental adaptation results on ARPA CSRNAB 1994 S5 evaluation data

Finally MLLR was applied to the S5 task. For this task, ten sentences from twenty speakers were recorded over unknown microphones were used for the test set. The microphone was known to be constant for a given speaker.

Initially the PMC Log-Add adaption [4] with simple convolutional noise estimation was applied [5]. On the evaluation test set this gave a word error rate of 10.6%. This performance was disappointing compared with the best published evaluation results of 9.7% [17] and is attributed to the poor modelling of the convolutional noise. These PMC models were then adapted using MLLR. The use of MLLR mean adaptation reduced the error rate, using the T_4 transformation set by 18%. Variance adaptation further reduced the error rate by 8%. Again, slight variations in performance were observed when the transformation set was changed. The best performance using the MLLR mean adaptation was with the T_1 transformation. However, there was no difference observed for any of the transformations when variance adaptation was applied.

8 Conclusions

A new technique for adapting the variances of a set of continuous density HMMs within the MLLR framework has been described. The variance transformation may yield a full or diagonal transform, even if the original covariance matrices were diagonal. In either case, it is guaranteed to increase the likelihood of the adaptation data. The memory requirements to estimate the transformation was shown to reasonable even for full covariance matrix transformations. The computational load of calculating the actual transformation matrix is small.

The technique was evaluated on a variety of large vocabulary recognition tasks. On all tasks variance adaption was found to improve the performance, with gains ranging from 1% to 8%. Though these improvements are small compared to those obtained adapting the means, the results are consistently better at little additional cost in computation. The number of additional parameters introduced is very small. For the full MLLR mean transform a total of $(n+1)\times n$ parameters are used per transform and for the diagonal transformation matrix adaption of the variance only n parameters are used per transform.

For all the experiments described the variance adaptation required the same number of frames to be present in a regression class in order for that class to be adapted, despite the small number of parameters used for the variance transformation. Additional improvements may be possible by reducing the number of frames required at a regression class for the variance adaptation. However as the variance adaptation is based on second order statistics, care must be taken to ensure that the transform is robust. In addition the variance transformation matrices were constrained to be diagonal. As this is not a necessary constraint, further gains in performance may be obtained using a full variance transformation matrix.

Acknowledgements

The MLLR variance adaptation code was based on code originally written by Chris Leggetter. Mark Gales is a Research Fellow at Emmanuel College, Cambridge.

A Single-Pass Statistics Required for the Variance Transformation

This appendix gives the statistics required to estimate the mean and variance transformation matrices in a single pass of the adaptation data. Additionally it is assumed that the model covariance matrices are diagonal. This is a normal requirement of the MLLR mean transformation.

The standard mean transformation statistics are required. The estimate of the variance transformation may be written as (see equation 32)

$$\hat{\mathbf{H}}_m = \frac{\sum_{r=1}^{R}\left\{\mathbf{C}_{m_r}^T\left[\sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau)\mathbf{o}(\tau)^T - \hat{\mu}_{m_r}\bar{\mathbf{o}}_{m_r}^T - \bar{\mathbf{o}}_{m_r}\hat{\mu}_{m_r}^T + \hat{\mu}_{m_r}\hat{\mu}_{m_r}^T\sum_{\tau=1}^{T} L_{m_r}(\tau)\right]\mathbf{C}_{m_r}\right\}}{\sum_{m=1}^{R}\sum_{\tau=1}^{T} L_{m_r}(\tau)} \tag{48}$$

where $\bar{\mathbf{o}}_{m_r} = \sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau)$. To find the statistics that need to be stored to calculate the variance transformation, each element of equation 48 is considered separately. The denominator is simply the occupancy of that regression class and is independent of the mean transformation. The first term of the numerator is also independent of the mean transform. Examining the second numerator expression, the third term will simply be the transpose of the second, this may be written as

$$\sum_{r=1}^{R}\mathbf{C}_{m_r}^T\hat{\mu}_{m_r}\sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau)^T\mathbf{C}_{m_r} = \sum_{r=1}^{R}\mathbf{C}_{m_r}^T\left(\hat{\mathbf{A}}_m\mu_{m_r} + \hat{\mathbf{b}}_m\right)\sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau)^T\mathbf{C}_{m_r} \tag{49}$$

Examining the individual elements of equation 49[10]. Let

$$\mathbf{F}_m = \sum_{r=1}^{R}\mathbf{C}_{m_r}^T\left\{\hat{\mathbf{A}}_m\mu_{m_r}\sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau)^T\right\}\mathbf{C}_{m_r} \tag{50}$$

This may be re-expressed as

$$\mathbf{f}_{m:i} = \hat{\mathbf{a}}_{m:i}\left\{\sum_{r=1}^{R} c_{m_r ii}\mu_{m_r}\left(\sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau)^T\right)\mathbf{C}_{m_r}\right\} \tag{51}$$

where $\mathbf{f}_{m:i}$ is the i^{th} row of $\mathbf{F}_m$ and similarly $\hat{\mathbf{a}}_{m:i}$ to $\hat{\mathbf{A}}_m$. The only term that is a function of the mean transform is removed from within the summation, so may be applied after the summation. Considering the second term in equation 49

$$\sum_{r=1}^{R}\mathbf{C}_{m_r}^T\left\{\hat{\mathbf{b}}_m\sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau)^T\right\}\mathbf{C}_{m_r} = \hat{\mathbf{B}}_m\sum_{r=1}^{R}\left\{\mathbf{c}'_{m_r}\left(\sum_{\tau=1}^{T} L_{m_r}(\tau)\mathbf{o}(\tau)^T\right)\mathbf{C}_{m_r}\right\} \tag{52}$$

where $\hat{\mathbf{B}}_m = \mathrm{diag}(\hat{\mathbf{b}}_m)$ and $\mathbf{c}'_{m_r} = \mathrm{diag}(\mathbf{C}_{m_r})$. Again, $\hat{\mathbf{B}}_m$, the only term dependent on the mean transform, is outside the summation. Using the two expressions, equations 51 and 52, statistics may be stored at the regression class level and then transformed when the mean transform is known. Examining the final term in the numerator of equation 32, there are no statistics that may be stored at the regression class level, however the storage at the Gaussian component level increases only by a single value the occupancy, $\sum_{\tau=1}^{T} L_{m_r}(\tau)$. In the same fashion as the mean transform, if the regression class tree is to be used dynamically, then the statistics must be stored at base class level.

The storage requirements are $\mathcal{O}(n^3)$ for each base class, which is the same as for the mean transformation estimation. This is for the case where a full covariance transformation matrix is to

[10] There is an explicit assumption here that the same base classes are used for the variance adaption as the mean adaption.

be estimated. For the diagonal transform case it is only necessary to store the leading diagonal of $\mathbf{F}$, hence the cost will be $\mathcal{O}(n^2)$ per base class. Both the previous cases have been for a full transform of the mean vector. Using a diagonal mean transform and a diagonal covariance transform further reduces the storage requirements and results in a similar transform to [2]. It is therefore possible to obtain estimates for full covariance transforms in a memory efficient fashion.

B Transformation Equalities

This appendix derives the various transformation equalities used in this work. The simplest equalities, those based on the original mean transforms, are trivial to show as all previous transforms of the means are ignored as both the data and the transforms are based on the original values.

For the modification of the current estimate of the mean the proof is slightly more complex. It is necessary, for example, to show that

$$\hat{D}_4^{1,2} \quad \rightarrow \quad \{\hat{C}_4^{1,2}\} \quad = \quad D_4 \quad \rightarrow \quad \{C_4\} \tag{53}$$

The auxiliary function used in the derivation of the MLLR transformation is

$$\mathcal{Q}(\mathcal{M},\hat{\mathcal{M}}) = K_1 - \frac{1}{2}\mathcal{L}(\mathbf{O}_T|\mathcal{M})\sum_{m=1}^{M}\sum_{\tau=1}^{T} L_m(\tau)\left[K_m + \log(|\boldsymbol{\Sigma}_m|) + (\mathbf{o}(\tau)-\hat{\mu}_m)^T\boldsymbol{\Sigma}_m^{-1}(\mathbf{o}(\tau)-\hat{\mu}_m)\right] \tag{54}$$

Two cases need to be considered for estimating $\hat{\mu}_m$. Firstly there is the direct transform whereby

$$\hat{\mu}_m = \hat{\mathbf{W}}_m \left[\begin{array}{c} 1 \\ \mu_m \end{array}\right] \tag{55}$$

Extending the notation such that

$$\hat{\mu}_m^4 = \hat{\mathbf{W}}_m^4 \left[\begin{array}{c} 1 \\ \mu_m \end{array}\right] = \left[\begin{array}{cc} \mathbf{b}_m^4 & \mathbf{A}_m^4 \end{array}\right]\left[\begin{array}{c} 1 \\ \mu_m \end{array}\right] = \mathbf{b}_m^4 + \mathbf{A}_m^4\mu_m \tag{56}$$

where $\hat{\mathbf{W}}_m^4$ is the transform associated with component m. In this notation

$$\hat{\mu}_m^{1,2} = \mathbf{b}_m^2 + \mathbf{A}_m^2\mathbf{b}_m^1 + \mathbf{A}_m^2\mathbf{A}_m^1\mu_m \tag{57}$$

This is the mean used in the estimation of transform $\hat{\mathbf{W}}_m^4$. However the complete transform may be written as

$$\begin{aligned} \hat{\mathbf{W}}_m^4\hat{\mu}_m^{1,2} &= \left(\mathbf{b}_m^4 + \mathbf{A}_m^4\mathbf{b}_m^2 + \mathbf{A}_m^4\mathbf{A}_m^2\mathbf{b}_m^1\right) + \mathbf{A}_m^4\mathbf{A}_m^2\mathbf{A}_m^1\mu_m \\ &= \hat{\mathbf{W}}_m\left[\begin{array}{c} 1 \\ \mu_m \end{array}\right] \end{aligned} \tag{58}$$

Thus if there is a unique solution in a ML sense then the two transformations will be the same.

The above "proof" has been couched in terms of the means. For the variances the proof is simpler due to the nature of the variance transform (there is no offset).

The above equality rules may be applied to the binary regression class tree shown in figure 1. The transformation sequence obtained using the described scheme is

$$\left\{\begin{array}{lcl} D_1 & \rightarrow & \{C_4, C_5, C_6, C_7\} \\ D_2 & \rightarrow & \{C_4, C_5\} \\ D_4 & \rightarrow & \{C_4\} \\ D_3 & \rightarrow & \{C_6, C_7\} \end{array}\right\} = \left\{\begin{array}{lcl} D_4 & \rightarrow & \{C_4\} \\ D_2 & \rightarrow & \{C_5\} \\ D_3 & \rightarrow & \{C_6, C_7\} \end{array}\right\} = T_1 \tag{59}$$

where D_j denotes the transformation statistics, $\mathbf{G}^{(i)}$ and $\mathbf{Z}$, associated with node j and C_j indicates the set of components associated with regression class, or node, j. Thus in the diagram $D_2 = \{D_4, D_5\}$ and $C_2 = \{C_4, C_5\}$.

It is not necessary to transform the original model means: for instance the transformation may be based on the mean after a previous pass of adaptation. In this case, both the mean to be adapted and the data to calculate the transform will be altered. Let $\hat{D}_i^j$ indicate the data obtained using the latest estimates of the means of Gaussians C_i after they have been transformed at node j, i.e. $\hat{C}_i^j$. A particular mean may be transformed many times, for example $\hat{C}_i^{j,k}$ indicates that the components have been transformed at regression class j and then at regression class k. Using a top-down transformation sequence and adapting the latest estimate of the mean yields

$$\left\{\begin{array}{lcl} D_1 & \rightarrow & \{C_4, C_5, C_6, C_7\} \\ \hat{D}_2^1 & \rightarrow & \{\hat{C}_4^1, \hat{C}_5^1\} \\ \hat{D}_4^{1,2} & \rightarrow & \{\hat{C}_4^{1,2}\} \\ \hat{D}_3^1 & \rightarrow & \{\hat{C}_6^1, \hat{C}_7^1\} \end{array}\right\} = \left\{\begin{array}{lcl} D_4 & \rightarrow & \{C_4\} \\ D_2 & \rightarrow & \{C_5\} \\ D_3 & \rightarrow & \{C_6, C_7\} \end{array}\right\} = T_1 \tag{60}$$

Since only the most specific transform determines the final values (see above) the same effective transforms results. Hence all top down schemes result in the same effective transformation[11].

[11] This is strictly only true when variance adaption is not applied or is applied after the complete transformation of all the means.

References

[1] A P Dempster, N M Laird, and D B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39:1–38, 1977.

[2] V V Digalakis, D Rtischev, and L G Neumeyer. Speaker adaptation using constrained estimation of Gaussian mixtures. *IEEE Transactions SAP*, 3:357–366, 1995.

[3] M J F Gales. *Model-Based Techniques for Noise Robust Speech Recognition*. PhD thesis, Cambridge University, 1996.

[4] M J F Gales and S J Young. A fast and flexible implementation of parallel model combination. In *Proceedings ICASSP*, pages 133–136, 1995.

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

[6] J L Gauvain and C H Lee. Maximum a-posteriori estimation for multivariate Gaussian mixture observations of Markov chains. *IEEE Transactions SAP*, 2:291–298, 1994.

[7] A J Hewett. *Training and Speaker Adaptation in Template-Based Speech Recognition*. PhD thesis, Cambridge University, 1989.

[8] C H Lee, C H Lin, and B H Juang. A study of speaker adaptation of continuous density HMM parameters. In *Proceedings ICASSP*, pages 145–148, 1990.

[9] C J Leggetter. *Improved Acoustic Modelling for HMMs using Linear Transformations*. PhD thesis, Cambridge University, 1995.

[10] C J Leggetter and P C Woodland. Speaker adaptation of continuous density HMMs using linear regression. In *Proceedings ICSLP*, pages 451–454, 1994.

[11] C J Leggetter and P C Woodland. Flexible speaker adaptation for large vocabulary speech recognition. In *Proceedings Eurospeech*, pages 1155–1158, 1995.

[12] C J Leggetter and P C Woodland. Flexible speaker adaptation using maximum likelihood linear regression. In *Proceedings ARPA Workshop on Spoken Language Systems Technology*, pages 110–115, 1995.

[13] C J Leggetter and P C Woodland. Maximum likelihood linear regression for speaker adaptation of continuous density HMMs. *Computer Speech and Language*, 9:171–186, 1995.

[14] L Neumeyer and M Weintraub. Robust speech recognition in noise using adaptation and mapping techniques. In *Proceedings ICASSP*, pages 141–144, 1995.

[15] L R Neumeyer, A Sankar, and V V Digalakis. A comparative study of speaker adaptation techniques. In *Proceedings Eurospeech*, pages 1127–1130, 1995.

[16] J J Odell, V Valtchev, P C Woodland, and S J Young. A one pass decoder design for large vocabulary recognition. In *Proceedings ARPA Workshop on Human Language Technology*, pages 405–410, 1994.

[17] D S Pallett, J G Fiscus, W M Fisher, J S Garofolo, B A Lund, A Martin, and M A Przybocki. 1994 benchmark tests for the ARPA spoken language program. In *Proceedings ARPA Workshop on Spoken Language Systems Technology*, pages 5–36, 1995.

[18] P C Woodland, J J Odell, V Valtchev, and S J Young. The development of the 1994 HTK large vocabulary speech recognition system. In *Proceedings ARPA Workshop on Spoken Language Systems Technology*, pages 104–109, 1995.

[19] S J Young, J J Odell, and P C Woodland. Tree-based state tying for high accuracy acoustic modelling. In *Proceedings ARPA Workshop on Human Language Technology*, pages 307–312, 1994.