Adaptive Training and Noise Estimation for Model-Based Noise Compensation for ASR

F. Flego, M.J.F. Gales
CUED/F-INFENG/TR.653
September 2010
Cambridge University Engineering Department
Trumpington Street
Cambridge, CB2 1PZ
England
E-mail: {ff257,mjfg}@eng.cam.ac.uk
1 Introduction

Speech recognition in noisy environments is a difficult task. To address this problem a number of approaches have been proposed, which generally fall into two categories: front-end and back-end based schemes. Front-end pre-processing of the source signal or related speech features provides the clean trained acoustic models with a “cleaned” representation of the corrupted speech [18]; in back-end adaptation, which is commonly referred to as model compensation, the (clean) model parameters are “compensated” so that they match the target acoustic condition [1, 14, 15].

Model compensation techniques can be further classified into “adaptive” and “predictive”. In the first case the compensation parameters are directly obtained from the adaptation data. Among others, popular schemes based on this approach are Maximum Likelihood Linear Regression (MLLR), Constrained MLLR (CMLLR), and Noisy CMLLR (NCMLLR) [11].

This technical report focuses on predictive model compensation approaches for noise robust speech recognition. In particular it considers the popular Joint Uncertainty Decoding (JUD) [17] and Vector Taylor Series (VTS) [1] compensation schemes. These techniques use a highly non-linear mismatch function to “predict” the corrupted speech model parameters given a clean model set and an explicit estimate of the noise parameters. Though this increases the model complexity, the advantage over adaptive approaches is that fewer parameters, in general, need to be estimated.

Expectation Maximisation (EM) can be used to estimate the noise parameters from the data and second-order gradient-ascent approaches have been widely used to optimise the related auxiliary function [14, 17].

Adaptive training [2] can also be applied using the predictive schemes above. This provides JUD Adaptive Training (JAT) and VTS Adaptive Training (VAT). JAT and VAT proved to be efficient approaches for training canonical model parameters using data affected by different noise conditions. In this case, noise transform parameters are estimated for each noise-homogeneous block of training data. Model parameters are then compensated using these noise transforms, and an EM-based approach is used to refine the canonical model parameters. The process is then iterated until convergence. Similar to the noise estimation approach, the auxiliary function for JAT or VAT can be maximised using a second-order approach [17, 4, 10].

An alternative technique to the second-order approaches above (generalised EM) can be used to maximise the auxiliary function. This is obtained by introducing an additional latent variable to the original auxiliary function and re-applying the EM scheme (multi-level EM). This alternative EM-based scheme has been applied for both noise [9, 13] and canonical model estimation [9].

The EM-based approach above can be re-formulated within a Factor Analysis (FA) framework [8, 23]. This framework was initially used to describe NCMLLR compensation and EM-based solutions were obtained for both transform and canonical model parameter estimation [11]. Also JUD and VTS compensation can be represented using an FA framework, where the mismatch function is written as a generative model and standard FA results are exploited to obtain EM-based estimates for both noise [5] and JAT and VAT parameters [5]. One of the advantages of using such a framework is that discriminative training can be easily extended to yield discriminative adaptive training. In particular, the Minimum Phone Error (MPE) criterion [21] was used to train acoustic models using NCMLLR [11] transforms, and JUD and VTS compensation [5].

The above JUD and VTS based approaches for (discriminative) adaptive training and noise estimation, based on both second-order and EM-based schemes, use a linearised version of the mismatch function so that a tractable auxiliary function is obtained. Depending on the maximisation scheme used, further approximations are then applied. For the second-order based schemes these approximations yield simple gradient and Hessian forms. Back-off procedures are generally applied in this case to ensure that the “real” auxiliary function is maximised. This differs from the case of the EM-based approaches, where back-off the new estimates would counteract the computational advantages provided by the closed-form solutions.

In this work a detailed review of the above schemes is given, and the effects of the above approximations discussed in detail. For the EM-based approaches a unified FA framework, which extends that in [5], is proposed. This framework, as it will be shown, allows the necessary constraints on the generative model parameters to be easily set so that correct update expressions
are obtained. This turns out to be particularly important for the noise estimation case, which was shown to perform poorly when the above approximations are not taken account [30].

The different approaches discussed are tested using two different training configurations: the Aurora4 database and TRE-L database, which consists of 486 hours of both artificially corrupted and real noisy data.

The technical report is organised as follows. Section 2 briefly presents the basic concepts of JUD and VTS model compensation. The FA framework used to represent such compensation approaches is then described in Section 3. Section 4 describes JUD and VTS Adaptive training based on both the second-order and the FA-related EM approaches. The extension to JUD and VTS discriminative adaptive training is discussed in Section 5. Noise estimation based on both the second-order and EM schemes is then investigated in Section 6 and the results obtained applying the various methods described are discussed in Section 7.

2 Model based Compensation

For predictive model-based compensation approaches it is first necessary to specify a mismatch function relating the clean speech model parameters, the noise model, and the corrupted speech model. For this purpose the parameters involved in the compensation comprise statics, delta and acceleration coefficients. For example the corrupted speech vector can be written as \( y^T = [y_1, y_2, y_3] \).

The form of mismatch function used here for the statics can be written as

\[
\begin{align*}
y_s &= x_s + h_s + C \log \left( 1 + \exp \left( C^{-1}(z_s - x_s - h_s) \right) \right) = f(x_s, h_s, z_s)
\end{align*}
\]

where \( x_s \) and \( y_s \) are the clean and corrupted speech static features, \( C \) is the DCT matrix, \( z_s \) and \( h_s \) are the additive noise and convolutional noise static vectors respectively.

The clean speech for each model component \( m \) is Gaussian distributed as \( x \sim \mathcal{N}(\mu_{x}^{(m)}, \Sigma_{x}^{(m)}) \) and is independent of the noise. Similarly the additive noise distribution is \( z \sim \mathcal{N}(\mu_{z}, \Sigma_{z}) \) but \( z \) is assumed to be stationary implying that its mean is completely specified by the static part, i.e. only \( \mu_{z} \neq 0 \). Given the assumption of independence between the additive noise dimensions, the covariance is diagonal and comprises the statics as well as the dynamic coefficients.

The convolutional noise term, \( h \), is considered constant for a given noise condition and is therefore completely specified by its static mean, \( \mu_{h} \). The set of noise parameters is indicated by \( \mathcal{M}_{n} = \{ \mu_{z}, \Sigma_{z}, \mu_{h} \} \).

2.1 JUD and VTS

Both JUD and VTS can be described in terms of a model based on the joint distribution of the clean speech and the corrupted speech [15]. Initially JUD will be described. The acoustic space is partitioned into \( R \) base-classes and the clean and corrupted speech joint distribution is assumed to be jointly Gaussian for each class \( r \) and can be expressed as

\[
\begin{bmatrix}
x \\
y
\end{bmatrix} \sim \mathcal{N}\left( \begin{bmatrix}
\mu_{x}^{(r)} \\
\mu_{y}^{(r)}
\end{bmatrix}; \begin{bmatrix}
\Sigma_{x}^{(r)} & \Sigma_{xy}^{(r)} \\
\Sigma_{yx}^{(r)} & \Sigma_{y}^{(r)}
\end{bmatrix}\right)
\]

where the subscript \( y \) indicates parameters associated with the corrupted speech distribution and subscript \( x \) the clean speech distribution. The likelihood of the corrupted speech is then approximated by

\[
p(y|m) \approx \int p(y|x, r_m)p(x|m)dx
\]

where each of the \( M \) system components, \( m \), is assigned to a given base-class \( r_m \), indicated only with \( r \) in the following for simplicity. The statistics for the conditional distribution in the expression above can be obtained from the joint distribution parameters of Equation 2 which need
to be estimated. For this, the first-order VTS approximation is applied to Equation 1 for each base-class \( r \) providing
\[
y_z \approx f(\mu_s^{(r)}, \mu_h, \mu_z) + J_z^{(r)}(x_z - \mu_s^{(r)}) + J_x^{(r)}(h_z - \mu_h) + J_z^{(r)}((z_z - \mu_z) \quad (4)
\]
with \( J_z^{(r)} = I - J_x^{(r)} \) and the Jacobian is defined as
\[
J_x^{(r)} = \frac{\partial y_z}{\partial x_z} \mid \tau^{(r)}
\]
where the expansion point, \( \tau^{(r)} = \{\mu_s^{(r)}, \mu_h, \mu_z\} \), is obtained from the noise parameters \( \mathcal{M}_n \) and the base-class clean speech parameters \( \{\mu_s^{(r)}, \Sigma_s^{(r)}\} \).

To obtain the expressions for the compensated dynamic parameters the continuous time approximation is used [7]. The linearised mismatch function for the delta coefficients for example can be written as
\[
y_\Delta \approx J_x^{(r)} x_\Delta + J_z^{(r)} z_\Delta \quad (6)
\]
and a similar form can be written for the acceleration coefficients. These linear expressions are used to compute the parameters of Equation 2 where the covariances, given that the static, delta and delta-delta parameters are assume to be uncorrelated to each other, result in a block-diagonal form. For simplicity only the static parts of such parameters are computed in the following and the static symbol \( s \) is dropped for clarity. From Equation 4 it is possible to write\(^1\)
\[
\mu_y^{(r)} = \mathcal{E}\{y\} = f(\mu_x^{(r)}, \mu_h, \mu_z) \quad (7)
\]
\[
\Sigma_y^{(r)} = \mathcal{E}\{yy^T\} - \mu_y^{(r)}\mu_y^{(r)T} = J_x^{(r)}\Sigma_x^{(r)}J_x^{(r)T} + J_z^{(r)}\Sigma_z^{(r)}J_z^{(r)T} \quad (8)
\]
\[
\Sigma_{y_x}^{(r)} = \Sigma_{y_x}^{(r)T} = \mathcal{E}\{yx^T\} - \mu_y^{(r)}\mu_x^{(r)T} = J_x^{(r)}\Sigma_x^{(r)} \quad (9)
\]
where the expectations are computed with respect to the current values of the clean speech and noise parameters. The joint distribution parameters above can now be used in Equation 3 and the following form for JUD compensation is obtained
\[
p(y|m) = |A^{(r)}| \mathcal{N}(A^{(r)}y + b^{(r)}; \mu_x^{(m)}, \Sigma_x^{(m)} + \Sigma_y^{(r)}) \quad (10)
\]
\[
= \mathcal{N}(y; H^{(r)}\mu_x^{(m)} + g^{(r)}, H^{(r)}\Sigma_x^{(m)}H^{(r)T} + \Psi^{(r)}) \quad (11)
\]
where the noise transform for each base-class \( r \), \( \mathcal{T}^{(r)} = \{A^{(r)}, b^{(r)}, \Sigma_y^{(r)}\} \), is specified\(^2\) by the parameters of the joint distribution as follows
\[
A^{(r)} = \Sigma_x^{(r)}\Sigma_y^{(r)} = J_x^{(r)-1} \\
b^{(r)} = \mu_x^{(r)} - A^{(r)}\mu_y^{(r)} \\
\Sigma_y^{(r)} = A^{(r)}\Sigma_y^{(r)}A^{(r)T} - \Sigma_x^{(r)} \\
\]
An equivalent JUD expression where the means are also compensated is given in Equation 11 which is based on the following set of parameters
\[
H^{(r)} = A^{(r)T} = J_x^{(r)} \\
g^{(r)} = -H^{(r)}b^{(r)} \\
\Psi^{(r)} = \Sigma_y^{(r)} - H^{(r)}\Sigma_x^{(r)}H^{(r)T} \quad (13)
\]
\(^1\)The equivalent expressions for the dynamics parameters can be easily obtained considering Equation 6 instead.
\(^2\)Symbols \( \mathcal{T}^{(r)} \) and \( \mathcal{T} \) are used to indicate a transform for a given base-class and all the transforms, respectively.
In the following sections both Equations 10 and 11 are used, the choice depending on the parameters to be estimated and on the approach used, which makes one or the other expression more convenient.

When the number of base-classes is equal to the number of components, \( R = M \), both Equations 10 and 11 yield VTS compensation [15]. However, though in some cases it may be more advantageous to implement VTS as a feature transform and a bias on the variances as in Equation 10, the likelihood for VTS is usually computed as

\[
p(y|m) = \mathcal{N}(y; \mu^{(m)}_y, \Sigma^{(m)}_y) \tag{14}
\]

with parameters obtained substituting index \( r \) with \( m \) in Equations 7 and 8.

For VTS the number of compensation parameters is typically much larger than for JUD, increasing their estimation cost, as well as requiring that all the acoustic model parameters be compensated.

2.1.1 Diagonal JUD and VTS compensation

To reduce the recognition cost, acoustic models with diagonal covariance matrices are normally used. For diagonal JUD model compensation this requires diagonalising each covariance term of the joint distribution, yielding

\[
\begin{bmatrix} x \\ y \end{bmatrix}^r \sim \mathcal{N}\left( \begin{bmatrix} \mu_x^{(r)} \\ \mu_y^{(r)} \end{bmatrix}; \begin{bmatrix} \text{diag}(\Sigma_x^{(r)}) & \text{diag}(\Sigma_y^{(r)}) \\ \text{diag}(\Sigma_x^{(r)}) & \text{diag}(\Sigma_y^{(r)}) \end{bmatrix} \right) \tag{15}
\]

The effect of the above diagonalisation is that the following diagonal expressions are obtained for terms \( A^{(r)} \) and \( \Sigma_b^{(r)} \) of Equation 10

\[
\begin{align*}
A^{(r)} &= \text{diag}(\Sigma_x^{(r)}) \text{diag}(\mathbf{J}_x^{(r)} \Sigma_x^{(r)})^{-1} \\
\Sigma_b^{(r)} &= A^{(r)} \text{diag}(\Sigma_y^{(r)}) A^{(r)\text{T}} - \text{diag}(\Sigma_y^{(r)})
\end{align*} \tag{16}
\]

and similar expressions can be found for \( \mathbf{H}^{(r)} \) and \( \Psi^{(r)} \) in Equation 11. When these transform parameters are used, the compensated covariance matrix will also be diagonal. The resulting approach is very efficient as the transform parameters \( T \) are computed on a base-class basis, and only a diagonal bias is applied to the acoustic models.

The standard diagonal VTS compensation form used in the literature is obtained by diagonalising the resulting compensated matrix in Equation 14 yielding (note that the same result is provided by diagonal JUD above setting \( R = M \))

\[
p(y|m) = \mathcal{N}(y; \mu^{(m)}_y, \text{diag}(\Sigma^{(m)}_y)) \tag{17}
\]

An alternative approach for diagonal JUD and VTS compensation can also be obtained by diagonalising the Jacobians in the linearised mismatch function in Equation 4 as follows

\[
y \approx f(\mu_x^{(r)}, \mu_h, \mu_z) + \text{diag}(\mathbf{J}_x^{(r)})(x - \mu_x^{(r)}) + \mathbf{J}_h^{(r)}(h - \mu_h) + \text{diag}(\mathbf{J}_z^{(r)})(z - \mu_z) \tag{18}
\]

yielding the following diagonal compensated covariance (in this case \( \Sigma^{(r)}_x \) need to be diagonal)

\[
\Sigma^{(r)}_y = \text{diag}(\mathbf{J}_x^{(r)} \Sigma_x^{(r)} \mathbf{J}_y^{(r)\text{T}}) + \text{diag}(\mathbf{J}_z^{(r)} \Sigma_z \mathbf{J}_y^{(r)\text{T}}) \tag{19}
\]

Though this approach is more computationally efficient compared to the above schemes, it is not used in this work since it is less accurate. This is evident if Equation 19 is compared to Equation 8 which provides the full \( \Sigma_y \) used in Equations 16 and 17. In the latter case the diagonalisation is applied at a later stage thus retaining important information provided by the
off-diagonal terms of the Jacobians.

An alternative diagonal JUD form, VTS-JUD, has also been proposed in [29] and its form is given by

\[
p(y|m) = \mathcal{N}(y; H^{(r)}\mu_z^{(m)} + g^{(r)}, \text{diag}(H^{(r)}\Sigma_z^{(m)}H^{(r)T} + \Psi^{(r)}))
\]

where the joint distribution parameters are block-diagonal as in Equation 2 and only the resulting compensated covariance is diagonalised.

VTS-JUD was found to be more accurate and to provide better recognition results when compared to diagonal JUD [29]. However the latter form is preferred in this work since, as shown in next section, VTS-JUD is not suitable for EM-based parameter estimation and is more computationally expensive.

3 Linear Generative Processes

Factor Analysis (FA) is a statistical method for modelling the covariance structure of high dimensional observed data \( y \) using a small number of latent (hidden) variables \( x \). It can be used as a structured covariance matrix scheme, as an alternative to using full covariance matrices and various extended FA forms have been used to allow more extensive parameter tying [8, 23, 26]. The FA generative model can be written as

\[
y = \Lambda x + \epsilon
\]

where \( \Lambda \) is the loading matrix and \( x \) is Gaussian distributed as \( x \sim \mathcal{N}(\mu_x, \Sigma_x) \). The error term \( \epsilon \) is independent of \( x \) and its distribution \( \epsilon \sim \mathcal{N}(\mu_e, \Sigma_e) \) is based on constant parameters.

Considering, for example, the linearised mismatch function in Equation 4 for VTS compensation (setting \( r = m \), it is possible to write the following generative model

\[
y|m = J_x^{(m)}x + J_z^{(m)}z + J_\mu^{(m)}\mu + d^{(m)}
\]

with

\[
x \sim \mathcal{N}(\mu_x^{(m)}, \Sigma_x^{(m)});
\]

\[
z \sim \mathcal{N}(\mu_z, \Sigma_z);
\]

\[
d^{(m)} = f(\mu_x^{(m)}, \mu_h, \mu_z) - J_x^{(m)}\mu_x^{(m)} - J_z^{(m)}\mu_z - J_\mu\mu
\]

where the symbol “\(^{(m)}\)” is used to differentiate between parameters to be estimated, as for example \( \mu_x^{(m)} \), from parameters based on current estimates such as \( \mu_x \).

The generative model above represents a generalisation of Equation 21. The first two terms are the clean speech and noise latent variables whose squared loading matrices are not used for covariance modelling, but instead to map the clean speech to the noise corrupted speech distribution. The third term is related to \( \mu_h \) which, as previously discussed, completely specifies the convolutional noise distribution. The bias function \( d^{(m)} \) incorporates all the remaining mismatch function terms.

Equation 22 can be used to obtain EM-based update formulae to update the clean speech parameters using the current noise parameters for the \( z \) distribution, or to update the noise parameters using the current clean noise parameters for the \( x \) distribution. In both cases a generative model form similar to Equation 21 is obtained.

The following two criteria need to be satisfied by the above generative model in order to be used for parameter estimation:

1. **Fixed loading matrix and bias term** are required so that the log-likelihood is increased at each EM iteration. This is not the case for Equation 22 where the loading matrix associated for example with the clean speech is the Jacobian. The value of this Jacobian, as shown in
Equation 5, is a function of the clean speech, additive and convolutional noise means. If any of these parameters change, the Jacobian will also change and the solutions obtained do not guarantee an increase in the log-likelihood.

2. For efficient decoding a diagonal corrupted-speech covariance matrix is required. However, the loading matrices in Equation 22 are full and the following full compensated covariance is obtained

\[ \Sigma_{y}^{(m)} = J^{(m)}_{x} \Sigma^{(m)}_{x} J^{(m)T}_{x} + J^{(m)}_{z} \Sigma^{(m)}_{z} J^{(m)T}_{z} \]  

(24)

To address this issue it is not possible to only diagonalise the result in the above expression, as this would be inconsistent with the joint \( \{x, y\} \) (or \( \{z, y\} \)) distribution derived from the generative model\(^3\).

The effects of the two requirements above are different depending on whether clean speech or noise parameter estimation is applied and on the compensation form used. Note that given the second requirement it is not possible to obtain a valid generative model for VTS-JUD compensation described in Section 2.1.1, therefore only diagonal JUD and VTS compensation are considered in the following.

The generative model which yields diagonal JUD compensation can be obtained from Equations 11 and 13. Using \( \Lambda^{(m)}_{x} \) and \( \Lambda^{(m)}_{z} \) for the loading matrices, this can be written as

\[ y|m, r = \Lambda^{(r)}_{x} x + \Lambda^{(r)}_{z} z + J^{(r)}_{z} \mu_{h} + d^{(r)} \]  

(25)

\[ x \sim N(\mu^{(m)}_{x}, \Sigma^{(m)}_{x}) = \text{diag}(\Lambda^{(r)}_{x})J^{(r)}_{x} \mu_{x} \]  

\[ z \sim N(\mu^{(m)}_{z}, \Sigma^{(m)}_{z}) = \text{diag}(\Lambda^{(r)}_{z})J^{(r)}_{z} \mu_{z} \]  

(26)

d^{(r)} = J^{(r)}_{z} \mu_{h} - \Lambda^{(r)}_{z} \mu_{z} + J^{(r)}_{z} \mu_{z} - \Lambda^{(r)}_{z} \mu_{z}

where, similarly to shared factor analysed HMM (FAHMM) and factor-analysis invariant to linear transformations (FACILT) [23, 27], the components of the clean speech and noise processes in the generative model share the constant base-class related parameters. A similar form is also used for Noisy-CMLLR (NCMLLR) transforms [12]. In the specific case the shared loading matrix \( \Lambda^{(r)}_{x} \) is function of the diagonal JUD parameters in Equation 16, \( \Lambda^{(r)}_{x} = H^{(r)} = \Lambda^{(r)}_{r} \), and is diagonal.

The distribution used for the latent variables \( \{x, z\} \) differs from that used in Equation 23. In particular the \( z \) covariance depends on the system component covariance \( \Sigma^{(m)}_{z} \) and on other base-class related constant terms, while the covariance for \( z \) is a more complex expression directly involving \( \Sigma_{z} \). The form of these distributions and the diagonal \( \Lambda^{(r)}_{x} \) allow a diagonal compensated covariance, \( \Sigma^{(m)}_{y} \), to be obtained, thus satisfying the second requirement discussed previously.

When Equation 25 is used to update the clean speech parameters \( \{\hat{\mu}^{(m)}_{x}, \hat{\Sigma}^{(m)}_{x}\} \), the current noise parameters are used to compute \( d^{(r)} \) and for the \( z \) distribution in Equation 26. These parameters will be constant during the estimation process and, given that the base-class parameters are also constant, the first requirement is satisfied. This is true for any \( \Lambda^{(r)}_{x} \) which can thus be set to the full Jacobian, \( \Lambda^{(r)}_{x} = J^{(r)}_{x} \).

When the noise parameters are estimated instead, it is not possible to obtain simple EM update expressions if the additive noise distribution in Equation 26 is used. Setting \( \Lambda^{(r)}_{z} = \text{diag}(J^{(r)}_{z}) \) addresses this issue providing the simple additive noise statistics\(^4\) as that of Equation 23. However, the generative model parameters are now a function of the noise parameters to be estimated which

---

\(^3\)This can be seen considering that the result of the marginalisation involves cross-terms as \( \hat{\Sigma}^{(m)}_{x} = J^{(m)}_{x} \hat{\Sigma}^{(m)} \) which are not diagonal.

\(^4\)Note that for clean speech parameter estimation this is not an issue, since the additive noise covariance in Equation 26 can be modified to account for all the constant terms at the right side of \( \hat{\Sigma}^{(m)}_{x} \), thus providing a clean speech distribution as that of Equation 23.
implies that the first requirement is not satisfied. In fact, under these conditions the solutions obtained do not guarantee to increase the log-likelihood and a modified scheme is proposed in Section 6.2 to address this issue.

For diagonal VTS compensation, two generative model forms can be used. The first is obtained from the VTS linearised mismatch function yielding the same expressions as those in Equation 25 and 26 after the base-class index $r$ is substituted by the system component index $m$. However, a different distribution for $x$ is obtained given by

$$x \sim \mathcal{N}(\hat{\mu}_x, \Lambda_x^{(m)-1}\text{diag}(\Lambda_x^{(m)}\Sigma_x^{(m)}\Lambda_x^{(m)}_x)\Lambda_x^{(m)}_x)$$  \hspace{1cm} (27)$$

Similar to the JUD case, to obtain simple distributions for $\{x, z\}$, the loading matrix relative to the process whose variable are being estimated needs to be diagonalised. Therefore, for clean speech parameter estimation, for example, it is necessary to set $\Lambda_x^{(m)} = \text{diag}(J_x^{(m)})$ and $\Lambda_z^{(m)} = J_z^{(m)}$.

An alternative VTS generative model form is obtained by setting $R = M$ in the JUD generative process in Equation 25. In this case the clean speech process loading matrix is always $\Lambda_x^{(m)} = \text{diag}(J_x^{(m)})$. The noise loading matrix, as in the previous case, needs to be diagonal when noise estimation is applied. In that case $\Lambda_z^{(m)} = \text{diag}(J_z^{(m)})$ is set. When used for clean speech parameter estimation, this generative model form for VTS compensation introduces fewer approximations when $\hat{\Sigma}_x^{(m)}$ is computed, compared to that of Equation 27 (for noise estimation the two forms are identical). The difference can be shown to be

$$\Delta \hat{\Sigma}_x^{(m)} = \text{diag}(J_x^{(m)}\Sigma_x^{(m)}J_x^{(m)}_x) - \text{diag}(J_x^{(m)}\Sigma_x^{(m)}\text{diag}(J_x^{(m)}))$$  \hspace{1cm} (28)$$

which shows that the information relative to the Jacobian off-diagonal terms is exploited when the VTS generative model based on Equation 25 is used. This is the reason why the latter form is used in this work.

However, both the generative model forms for VTS compensation cannot satisfy the first requirement that fixed loading matrices and bias term need to be used. In fact the latter are now a function of the parameters to be estimated, for both noise or clean speech parameter estimation. The solutions obtained will therefore not guarantee to increase the log-likelihood.

The generative processes discussed in this section have already been used in [3, 5] to obtain EM-based solutions for JUD and VTS discriminative adaptive training, and in [5] for EM-based JUD noise parameters estimation. Here a unified framework based on Equation 25 is used to address in detail the various issues discussed in this section and to extend the results of [3, 5]. JUD and VTS EM-based adaptive training is discussed in Section 4.2 and noise estimation in Section 6.2.

4 Adaptive Training

Adaptive training is an elegant approach to training acoustic models on non-homogeneous data, i.e. data where there are multiple speakers or acoustic noise conditions. The general process is to use transforms to model each speaker or acoustic condition during the training phase. Depending on the nature of these transforms, different forms of adaptive training can be found in the literature. MLLR and Constrained MLLR (CMLLR) based approaches were proposed in [2, 6]. A Stochastic Matching based technique was illustrated in [22, 25] and Noisy CMLLR (NCMLLR) transforms were employed in [11]. Recently, these adaptive training approaches have been applied to predictive

$^5$In this case, the base-class parameters $\{\mu_x^{(r)}, \Sigma_x^{(r)}\}$ are replaced by the current clean model parameters $\{\mu_x^{(m)}, \Sigma_x^{(m)}\}$.
model-based compensation schemes such as JUD, referred to as Joint Adaptive Training (JAT) [17], and VTS, referred to in this work as VTS Adaptive Training (VAT) [4, 9, 10].

The general training process for adaptive training based on adaptive or predictive transforms, aims to optimise the following standard auxiliary function

\[
Q(\hat{\mathcal{M}}, \hat{\mathcal{M}}_n; \mathcal{M}, \mathcal{M}_n) = E(\Theta|Y, \mathcal{M}_n, \mathcal{M}_n) \left[ \log p(Y|\Theta, \hat{\mathcal{M}}, \hat{\mathcal{M}}_n) \right]
\]

\[
= \sum_{s=1}^{S} \sum_{t=1}^{T(s)} \sum_{m=1}^{M} \gamma_{t}^{(sm)} \log p(y_{t}^{(s)}|\hat{\mathcal{M}}, \hat{\mathcal{M}}_n, m)
\]

(29)

where the expectation is computed over all possible state sequences \(\Theta\) in the state sequence space for utterance set \(Y\), and \(\hat{\mathcal{M}}\) and \(\hat{\mathcal{M}}_n\) are the new clean speech and noise models. Training data \(Y = \{y_{1}^{(1)}, \ldots, y_{T(s)}^{(S)}\}\) is partitioned into \(S\) homogeneous blocks, each of length \(T(s)\), and \(\gamma_{t}^{(sm)}\) is the posterior probability that observation \(y_{t}^{(s)}\) is generated by component \(m\), which belongs to regression class \(r\). This is derived by aligning the training data transcriptions using the current clean speech models \(\mathcal{M}\) compensated with transforms based on the current noise model \(\mathcal{M}_n\).

Note that, although not explicitly shown, a different noise model \(\mathcal{M}_n\) can be associated with each utterance \(s\), for which the related noise transform, \(T(s)\) is being estimated. In this case, where this does not create ambiguity, the same symbol \(\mathcal{M}_n\) is used in the following to indicate the set of noise models.

The general approach adopted to optimise the auxiliary function of Equation 29 is based on an iterative training procedure [2]:

1. train a multi-style (speaker/environment) system for initial canonical model parameters;
2. initialise the set of transforms \(T\) to the identity matrix;
3. given the current canonical model and transforms \(\{\mathcal{M}, T\}\) estimate new noise parameters \(\mathcal{M}_n\);
4. given the current canonical model \(\mathcal{M}\), and transforms updated using the new noise parameters \(\mathcal{M}_n\), estimate new canonical model parameters \(\mathcal{M}\);
5. repeat steps (3) and (4) until convergence;

In this work, transform parameters are obtained using the predictive approaches described in Section 2.1. Although adaptive training using full covariance matrices is possible, it is not efficient and may not be feasible for medium/large-sized HMM systems. Therefore only the diagonal JUD and VTS approaches will be considered for adaptive training. In the following section, two adaptive training schemes for canonical model parameter estimation are considered: a second-order gradient ascent scheme and an EM-based approach. Noise transform estimation is discussed in Section 6.

### 4.1 Second-order gradient ascent approach

This section describes second-order approaches for maximising the auxiliary function in Equation 29 with respect to the canonical model parameters \(\mathcal{M}\), while the noise parameters are fixed and set to the current estimates, \(\mathcal{M}_n = \mathcal{M}_n\). This is generally obtained using the following general second-order gradient ascent update formula

\[
\begin{bmatrix}
\frac{\partial Q}{\partial \mu^{(m)}_x} \\
\frac{\partial Q}{\partial \sigma_{x}^{(m)} \mu^{(m)}_x}
\end{bmatrix}
= \zeta \left[
\begin{bmatrix}
\frac{\partial^2 Q}{\partial \mu^{(m)}_x^2} & \frac{\partial^2 Q}{\partial \mu^{(m)}_x \partial \sigma_{x}^{(m)}} \\
\frac{\partial^2 Q}{\partial \sigma_{x}^{(m)} \partial \mu^{(m)}_x} & \frac{\partial^2 Q}{\partial \sigma_{x}^{(m)} \partial \sigma_{x}^{(m)}}
\end{bmatrix}
\right]^{-1}
\begin{bmatrix}
\frac{\partial Q}{\partial \mu^{(m)}_x} \\
\frac{\partial Q}{\partial \sigma_{x}^{(m)} \mu^{(m)}_x}
\end{bmatrix}
\]

(30)

where the \(D\)-dimensional vector \(\sigma_{x}^{(m)}\) is used to indicate the elements of the diagonal clean speech covariance matrix \(\Sigma_{x}^{(m)}\), and the subscript \(\mathcal{M}\) indicates that the partial derivatives are evaluated...
at the current clean model parameters \( \{ \mu_s^{(m)}, \sigma_s^{(m)2} \} \). Each Hessian term in Equation 30 has a block diagonal structure, one block for each of the statics, delta and delta-delta parameters. In the following only the update expressions for the statics are considered, as the results for the dynamic coefficients will be similar. The learning rate parameter \( \zeta \) is used to control the convergence rate of the algorithm.

There are two main issues when the second-order approach above is used. The first issue is the inherent approximation introduced by the approach itself, the solutions of which are not guaranteed to maximise the auxiliary function. This problem is generally addressed by controlling the parameter \( \zeta \) and verifying that after each estimate, the solutions actually result in increases in \( Q() \).

The second issue is related to the form used for model compensation. Depending on whether JUD or VTS is used, some approximations may be necessary to obtain the derivatives in Equation 30. This is analysed in detail in the following sections.

4.1.1 JUD

When using JUD compensation, the resulting training procedure is Joint Adaptive Training (JAT) [17]. In order to obtain the canonical model update expressions, the diagonal JUD compensation form of Equation 16 is used here to evaluate the auxiliary function in Equation 29.

The partial derivatives in Equation 30 may be computed by expanding each term to show the dependency on the transform parameters \( T \) explicitly. Considering for example \( \partial Q/\partial \mu_s^{(m)} \) it is possible to write

\[
\frac{\partial Q}{\partial \mu_s^{(m)}} = \frac{\partial Q}{\partial \mu_s^{(m)}} \bigg|_{T^{(r)}=\text{const.}} + \frac{\partial Q}{\partial T^{(r)}} \frac{\partial T^{(r)}}{\partial \mu_s^{(m)}}
\]

(31)

where, as in Equation 30, the derivatives are evaluated at the current model parameters (symbol \( \mathcal{M} \) not shown for simplicity), and the notation \( T^{(r)} = \text{const.} \) indicates that the derivative is computed by fixing the transform parameters of base-class \( r \) selected by \( m \) for each utterance \( s \).

For diagonal JUD the parameters \( T^{(r)} = \{ A^{(r)}, b^{(r)}, \Sigma_b^{(r)} \} \) (or equivalently \( \{ H^{(r)}, g^{(r)}, \Psi^{(r)} \} \)) are diagonal and constant, as they are based on the fixed base-class parameters \( \{ \mu_s^{(r)}, \sigma_s^{(r)2} \} \). The term \( \partial T^{(r)}/\partial \mu_s^{(m)} \) in Equation 31 is therefore null, yielding

\[
\frac{\partial Q}{\partial \mu_s^{(m)}} = \frac{\partial Q}{\partial \mu_s^{(m)}} \bigg|_{T^{(r)}=\text{const.}}
\]

(32)

with the same result applying to the remaining derivatives of Equation 30. The following expressions for each gradient component \( i \) are therefore obtained

\[
\frac{\partial Q}{\partial \mu_{x_i}^{(m)}} = \sum_{s=1}^{S} \sum_{t=1}^{T^{(r)}} \omega_{ti}^{(sm)} \left( y_{ti}^{(s)} b_i^{(s)} - \hat{\mu}_{x_i}^{(m)} \right)
\]

(33)

\[
\frac{\partial Q}{\partial \sigma_{x_i}^{(m)2}} = \sum_{s=1}^{S} \sum_{t=1}^{T^{(r)}} \frac{1}{2} \omega_{ti}^{(sm)} \left( \frac{(y_{ti}^{(s)} b_i^{(s)})^2 - \hat{\mu}_{x_i}^{(m)2}}{\sigma_{x_i}^{(m)2} + \sigma_{bi}^{(sr)2}} - 1 \right)
\]

(34)

with \( \omega_{ti}^{(sm)} = \gamma_t^{(sm)}/(\hat{\sigma}_{x_i}^{(m)2} + \sigma_{bi}^{(sr)2}) \). Similar expressions are obtained for the Hessian terms which are diagonal, providing independent update expressions for each canonical model parameter dimension \( i \), \( \{ \hat{\mu}_{x_i}^{(m)}, \hat{\sigma}_{x_i}^{(m)2} \} \). Note that, as the intuition behind adaptive training suggests, each term in the summation is weighted by \( \omega_{ti}^{(sm)} \). This term decreases for observations with lower SNR, since the uncertainty bias term \( \sigma_{bi}^{(sr)2} \) becomes larger.

To ensure that JAT based on the second-order approach of Equation 30 functions correctly, it is necessary that the Hessian matrix be negative-definite. If this requirement is not met the parameters may move towards a local minimum, rather than a maximum. Since the second
derivative of \(Q()\) w.r.t. the mean is always negative, it is sufficient to control the second derivative w.r.t. the variance. This is accomplished by means of a stabilising parameter that can be varied across different iterations of parameter re-estimation to improve the estimation speed [17].

Note that although exact auxiliary function derivatives can be computed for JAT, the approximate nature of the second-order approach implies it is still necessary to ensure that the solutions result in increases in \(Q()\).

### 4.1.2 VTS

When using VTS compensation, the resulting training procedure is VTS Adaptive Training (VAT). As was shown in Section 2.1.1, diagonal VTS is equivalent to diagonal JUD when setting \(R = M\) in Equation 16, or using the direct VTS form in Equation 17. In the following discussion, the terms VAT,JUD, and VAT,VTS, are used to refer to the former and latter approaches respectively.

The two compensation forms are identical when the current and newly-estimated clean speech parameters are the same. However, when VAT is applied to estimate new canonical model parameters, the two forms differ. Considering for example the updated compensated mean, it is possible to write\(^6\)

\[
\begin{align*}
\text{VAT,JUD:} & \quad \hat{\mu}_x^{(sm)} = \sum_j \mathbf{H}^{(sm)}(\hat{\mu}_x^{(m)} - \mu_x^{(m)}) + \mu_x^{(sm)} \\
\text{VAT,VTS:} & \quad \hat{\mu}_x^{(sm)} = \sum_j \mathbf{J}_x^{(sm)}(\hat{\mu}_x^{(m)} - \mu_x^{(m)}) + \mu_x^{(sm)}
\end{align*}
\]

where \(\mathbf{H}^{(sm)} = \text{diag}(\mathbf{J}_x^{x(sm)})\), and \(\mu_x^{(sm)}\) are defined in Equation 7. Similar differences can be shown to exist for the compensated covariances, \(\Sigma_x^{(sm)}\).

Different canonical model update expressions are obtained for VAT,JUD and VAT,VTS. For VAT,JUD, the transforms \(T\) now depend on the canonical model parameters to be estimated \(\mathcal{M}\), rather than on fixed base-class clean parameters as was the case for JAT. This implies, that the term \(\partial T^{(m)} / \partial \hat{\mu}_x^{(m)}\) in Equation 31 is non-zero, and more complex expressions are obtained for the gradient and Hessian in Equation 30. In practice, the following approximation is introduced for ease of implementation

\[
\frac{\partial Q}{\partial \hat{\mu}_x^{(m)}} \approx \frac{\partial Q}{\partial \hat{\mu}_x^{(m)}} \bigg|_{T^{(m)} = \text{const.}}
\]

and similar approximations are applied to the remaining derivatives. This makes it possible for the same results obtained for JAT in Equations 33 and 34 to be used for VAT,JUD.

Considerations similar to those discussed above apply for VAT,VTS, where the Jacobians play a similar role to that of the transforms \(T\) in VAT,JUD. As a consequence, a similar expression to that in Equation 31 is obtained by considering terms \(\{\mathbf{J}_x^{(m)}, \mathbf{J}_z^{(m)}\}\) instead of \(T^{(m)}\). Given that for VAT,VTS the Jacobians are function of the canonical model means to be estimated, terms such as \(\partial \mathbf{J}_x^{(m)} / \partial \hat{\mu}_x^{(m)}\), must be considered in order to derive the gradient and Hessian expressions.

However, a similar approximation to that in Equation 37 is applied here to obtain simple update expressions such as

\[
\frac{\partial Q}{\partial \hat{\mu}_x^{(m)}} \approx \frac{\partial Q}{\partial \hat{\mu}_x^{(m)}} \bigg|_{\mathbf{J}_x^{(m)} = \text{const.}}
\]

with similar approximations being applied to obtain the remaining auxiliary function derivatives in Equation 30. Applying such approximations yields the following expressions for the gradient

\(^6\)Equation 35 is obtained from Equation 11 where \(\mathbf{H}^{(r)} \mu_x^{(m)} + \mathbf{g}^{(r)}\) is considered. Equation 36 is obtained by setting \(r = m\) in the mismatch function of Equation 4 and computing the expectation with respect to the new clean speech variable \(\bar{x}\).
\[ \frac{\partial Q}{\partial \tilde{\mu}_x^{(m)}} = \sum_{s=1}^{S} \sum_{t=1}^{T_{(s)}} \gamma_{t}^{(s)} \sum_{l=1}^{D} \frac{y_{lt} - \mu_{yl}^{(s)}}{\sigma_{yl}^{(s)}^2} y_{sl}^{(s)} \]  
\[ \frac{\partial Q}{\partial \sigma_{xl}^{(m)^2}} = -\frac{1}{2} \sum_{s=1}^{S} \sum_{t=1}^{T_{(s)}} \gamma_{t}^{(s)} \sum_{l=1}^{D} \frac{\gamma_{t}^{(s)}}{\sigma_{yl}^{(s)}^2} \left[ 1 - \frac{\left( y_{lt} - \mu_{yl}^{(s)} \right)^2}{\sigma_{yl}^{(s)}^2} \right] \]  

where \( D \) is the dimension of the (static) feature vector and \( J_{sl}^{(sr)} \) represents the element in the \( l \)-th row and \( i \)-th column of \( J_{sl}^{(sr)} \). Similar expressions can be obtained for the Hessian matrix which becomes block diagonal.

It is possible to show that the difference between, for example, the \( \frac{\partial Q}{\partial \tilde{\mu}_x^{(m)}} \) term obtained in VAT \(_{\text{JAT}}\) and that obtained using VAT \(_{\text{VTS}}\) can be expressed (for each data block \( s \)) as

\[ \Delta^{(sm)} = \frac{\partial Q}{\partial \tilde{\mu}_x^{(m)}} \left( J_{x}^{(sm)} - \text{diag}(J_{x}^{(sm)}) \right) \]  

For very high or low SNRs \( J_{x} \) will either tend to the identity matrix or the null matrix [17], and the expression in Equation 41 will tend to zero, resulting in the approximations introduced by the two VAT schemes being similar. This result does not however hold for SNR values in other ranges. In this case the VAT \(_{\text{VTS}}\) approach is more accurate since it also exploits the information provided by the off-diagonal terms of the Jacobians.

However, independent of the noise conditions, both VAT \(_{\text{JAT}}\) and VAT \(_{\text{VTS}}\) provide solutions which do not optimise the “true” auxiliary function. When considering the means as an example, this function is computed using Equation 7 based on \( \tilde{\mu}_x^{(m)} \), rather than using Equation 35 or Equation 36 in the case of VAT \(_{\text{JAT}}\) or VAT \(_{\text{VTS}}\), respectively.

One potential solution to this problem would be to back-off the updated canonical model parameters until the “true” auxiliary function is actually increased. Unfortunately, this would necessitate the storage of the statistics relative to each data block \( s \). Although this could be achieved for small medium training sets, it was shown to be unnecessary in [4, 10].

As for the JAT approach, in the case of VAT the Hessian matrix must also be negative-definite to guarantee a local maximum of the auxiliary function is reached. For VAT \(_{\text{JAT}}\), each of the Hessian matrix terms are diagonal, and the same JAT stabilisation scheme used in [4] is used in this work.

In the case of VAT \(_{\text{VTS}}\), a straightforward approach [10] may be taken to ensure that the Hessian is negative definite, where the off-diagonal terms of the Hessian are set to zero for simplicity\(^7\). In this work the same stabilisation approach used for VAT \(_{\text{JAT}}\) is applied to the diagonal terms of the Hessian only.

### 4.2 EM-based approach

This section describes an EM-based approach for estimating the canonical model parameters \( \hat{\mathcal{M}} \) for adaptive training based on the diagonal predictive model compensation approaches of Section 2.1. Similarly to the approach used for adaptive training based on NCMLLR transforms [11], the EM update formulae are obtained using the FA framework, where the generative model represents JUD and VTS compensation as described in Section 3.

Following the same approach as in [8, 23], it is possible to write the following expression for the log-likelihood in Equation 29 (not showing indexes \( m, s \) and the noise parameters which are fixed and set to the current estimates, \( \mathcal{M}_n = \hat{\mathcal{M}}_n \))

\[ \log p(y | \hat{\mathcal{M}}) = \log \int_{\mathcal{M}} p(y, x | \mathcal{M}) dx \geq \int_{\mathcal{M}} p(x | \mathcal{M}) \log p(y, x | \mathcal{M}) dx + \mathcal{H}(p(x | \mathcal{M})) \]  

\(^7\)In [10] both the noise and the canonical model are updated at each iteration. This may improve convergence speed although it is not clear whether there will be stability issues with more complex training sets.
where \( H(p|x; M) \) is the entropy of the clean speech distribution and is independent of \( \hat{M} \). Substituting the above expression into Equation 29 yields the following lower bound for the original auxiliary function

\[
Q(\hat{M}; M) = E_{X, \Theta | Y, M} \left[ \log p(X, Y, \Theta | M) \right]
\]  

(43)

where the expectation \( E[\cdot] \) is computed with respect to both the clean speech features \( X \) and the state sequence \( \Theta \), given the corrupted speech \( Y \), and the current canonical model parameters \( M \).

The auxiliary function in Equation 43 may therefore be optimised to obtain the updated canonical model parameters. To achieve this, Equation 43 can be expanded as follows

\[
Q(\hat{M}; M) = \sum_{s=1}^{S} \sum_{t=1}^{T^{(s)}} \sum_{m=1}^{M} \gamma_t^{(sm)} E_{x \mid y_t^{(s)}} \left[ \log p(y_t^{(s)} | x, s, m, \hat{M}) + \log p(x | m, \hat{M}) \right]
\]  

(44)

and, under the hypothesis that the first term in the r.h.s is independent of \( \hat{M} \), it is possible to find simple expressions for the auxiliary function derivatives w.r.t \( \{\hat{\mu}_x(m), \Sigma_x(m)\} \). Equating them to zero yields the following standard update formulae for the canonical model parameters

\[
\hat{\mu}_x(m) = \frac{\sum_{s,t} \gamma_t^{(sm)} E_{x \mid y_t^{(s)}} \left[ x \right] y_t^{(s)} \mid s, m, \hat{M}}{\sum_{s,t} \gamma_t^{(sm)}}
\]  

(45)

\[
\Sigma_x(m) = \text{diag} \left( \frac{\sum_{s,t} \gamma_t^{(sm)} E_{x \mid y_t^{(s)}} \left[ xx^T \right] y_t^{(s)} \mid s, m, \hat{M}}{\sum_{s,t} \gamma_t^{(sm)}} - \hat{\mu}_x(m) \hat{\mu}_x(m)^T \right)
\]  

(46)

In contrast with the second-order approaches of Section 4.1 which always require regularisation procedures, the above solutions guarantee to maximise the auxiliary function. However, this may result in slower convergence and multiple EM iterations may be required given a set of state/component posteriors \( \gamma_t^{(sm)} \).

The sections which follow describe how the expectations in Equations 45 and 46 can be obtained from the diagonal JUD or VTS generative models described in Section 3. These approaches also make it possible to easily extend discriminative training to function with JUD or VTS transforms. This is discussed in Section 5.

### 4.2.1 JUD

EM-based diagonal JUD Adaptive Training (JAT-EM) based on a generative model was proposed in [3], which exploits the results obtained for NCMLLR based adaptive training [11]. Here the same approach is used, but is however applied in the context of the more general framework defined in Section 3. Considering constant noise parameters, the generative process for JAT-EM is obtained by re-arranging the terms of Equation 25 (for a given training data block \( s \)) to yield

\[
y \mid m, r = H^{(sr)} x + e^{(sr)},
\]  

(47)

where the loading matrix \( H^{(sr)} = A^{(sr)-1} \) is obtained from Equation 16 and is diagonal. The clean speech variable and error term are Gaussian distributed as \( x \sim N(\mu_x(m), \Sigma_x(m)) \), and \( e^{(sr)} \sim N(\mu_{e}^{(sr)}, \Sigma_{e}^{(sr)}) \), with mean and covariance given by

\[
\mu_x^{(sr)} = f(\mu_x(m), \mu_b, \mu_d) - H^{(sr)} \mu_x
\]  

(48)

\[
\Sigma_x^{(sr)} = \text{diag}(J_x^{(sr)} \Sigma_x^{(sr)} J_x^{(sr)T} + J_z^{(sr)} \Sigma_z^{(sr)} J_z^{(sr)T} - H^{(sr)} \text{diag}(\Sigma_x^{(sr)}) H^{(sr)T})
\]  

(49)

The corrupted speech \( y \) in Equation 47 is Gaussian distributed, and the standard results for jointly Gaussian multivariate variables may therefore be applied to derive the expected values in
Equations 45 and 46. These are given by

\[
\mathcal{E}[x|y^{(s)}_t, s, m, \mathcal{M}] = \mu_x^{(m)} + \sum_{y} \sum_{y^{(s)}_t} \left( y^{(s)}_t - \mu_y^{(s)} \right) = \mu_{x|y}^{(s)} (50)
\]

\[
\mathcal{E}[x^T|y^{(s)}_t, s, m, \mathcal{M}] = \sum_{x} - \sum_{y} \sum_{y^{(s)}_t} \left( y^{(s)}_t - \mu_y^{(s)} \right) + \text{diag}(\mu_{x|y}^{(s)} \mu_{x|y}^{(s)}^T) (51)
\]

where the joint distribution parameters of the clean and corrupted speech are obtained from the generative model. These are defined as

\[
\mu_y^{(s)} = H^{(sr)} \mu_x^{(m)} + \mu_y^{(sr)} (52)
\]

\[
\Sigma_y^{(sm)} = H^{(sr)} \Sigma_x^{(m)} H^{(sr)T} + \Sigma_y^{(sr)} (53)
\]

\[
\Sigma_{yx}^{(sm)} = \Sigma_{xy}^{(sm)} = H^{(sr)} \Sigma_x^{(m)} (54)
\]

The JAT-EM approach described here addresses both issues of Section 3, since the FA parameters are constant and diagonal. Furthermore, the solutions provided by Equations 45 and 46 are exact since the conditional probability in the right hand side of Equation 44 is obtained from the generative model whose parameters, for fixed \( x \), are not a function of \( \mathcal{M} \). This guarantees that the training data log-likelihood will increase at each EM iteration.

### 4.2.2 VTS

When using a diagonal VTS generative model to obtain the statistics in Equations 45 and 46, the resulting training procedure is EM-based diagonal VTS Adaptive Training (VAT-EM). Two VTS generative models were described in Section 3, one directly obtained from VTS compensation (related to Equation 27) and the other obtained from the diagonal JUD generative model. The latter approach was employed in [3] and is also used in this work since, as Equation 28 shows, it provides a better approximation to the corrupted speech covariances.

However, in both cases, the loading matrix and error term parameters depend on the canonical model parameters which are being estimated. This contradicts the first requirement defined in Section 3, with the consequence being that the conditional probability in the right hand side of Equation 44 can no longer be ignored when the derivatives with respect to \( \mathcal{M} \) are computed.\(^8\)

The aforementioned issue is similar to that of second-order VAT which lead to the approximation in Equation 37. Here, as in [3], the generative process parameters will be considered constant and the update expressions given by Equations 45 and 46 are used, with statistics given by Equations 50 and 51. However, in this case the joint distribution parameters are computed at the component level.

The effect of the assumptions described above is that the solutions obtained will be approximate and the value of the auxiliary function is no longer guaranteed to increase. However, experiments reported in [3] for EM-based VTS discriminative training (illustrated in the next section), showed that convergence may still be achieved. Convergence is also observed in this work where VAT-EM is run on different training data sets.

Experiments using EM-based updated formulae for VAT are also reported in [9], though in this case not described using a FA framework.\(^9\)

### 4.3 Adaptive training schemes performance

A preliminary comparison of the alternative JAT and VAT approaches previously described is conducted in this section with the aim of selecting the approaches which will be used in the experimental section. This contrast is based on the speed of convergence during training of canonical

\(^8\)Note that in this case the generative model variables \( x \) and \( y^{(s)} \) are not independent and additional terms should be considered when \( \Sigma_y^{(sm)} \) and \( \Sigma_{yx}^{(sm)} \) are computed.

\(^9\)It is not clear what form of diagonalization is applied to the joint distribution parameters in [9] and, consequently, whether a valid joint distribution is obtained.
models and the related decoding performance. Two different training datasets are used for the experiments: the TREL training set, which consists of 486 hours of both artificially corrupted and real noisy data; and the Aurora4 dataset (12 hours). Further details on these datasets are provided in the experimental section.

For both JAT and VAT a multi-style system is used as the initial system and 4 iterations of canonical model parameter estimation are interleaved with noise transform updates. The TREL training data log-likelihood for each iteration is plotted in Figure 1 where the JAT and JAT-EM approaches are compared in the figure on the left, and the VAT_{VTS}, VAT_{MSE} and VAT-EM schemes are shown in the figure on the right.

![Figure 1: TREL database: Training data log-likelihood for each iteration of canonical model re-estimation based on the JAT and JAT-EM schemes (left), and on the VAT_{VTS}, VAT_{MSE} and VAT-EM schemes (right); transform parameters are updated every 4 iterations.](image)

The results show that JAT-EM consistently converges sooner than JAT, and this is reflected in the decoding performance of JAT-EM which showed a relative improvement of almost 20% over JAT.

For VTS based adaptive training schemes, VAT-EM yielded the lowest increase in log-likelihood, while VAT_{VTS} yielded the best convergence rate. The decoding performance of the VAT approaches, similarly to the JAT case, is correlated with the related log-likelihood trend, though the resulting differences are smaller. The reason for this is that the VAT systems required the introduction of some approximation which turned out to have an impact on their performance.

An equivalent comparison was carried out on the Aurora4 dataset. This configuration showed similar trends in the training log-likelihood value for the JAT and JAT-EM schemes, which also performed similarly in terms of decoding accuracy. However, the VAT schemes instead showed similar training and decoding trends as those observed on the TREL configuration.

Compared to Aurora4, the TREL dataset is characterised by a larger amount of data, with real noisy recordings and a mismatch in noise conditions between training and testing conditions. The additional complexity of this dataset made it possible to investigate the differences of the training approaches considered in more depth.

## 5 Discriminative Adaptive Training

The Minimum Phoneme Error (MPE) [20] criterion is a popular form of discriminative training. It is a specific instance of minimum Bayes’ risk training where the following criterion is minimised

\[
J_{\text{ape}}(\mathcal{M}) = \sum_{k=1}^{S} \sum_{H} P(H|Y^{(s)}, \mathcal{M}) \mathcal{L}(H, H_{\text{ref}}^{(s)})
\]  

(55)
where \( \mathcal{H}^{(s)} \) is the reference hypothesis for each block of training data \( Y^{(s)} \) and the summation for \( \mathcal{H} \) is over a restricted set of confusible hypotheses defined by a lattice for each block. The “loss” function \( \mathcal{L}(\mathcal{H}, \mathcal{H}^{(s)}) \) is measured at the phone level between a hypothesis and the reference. To optimize the above expression the following weak-sense auxiliary function is often used [20]:

\[
Q_{\text{aux}}(\hat{\mathcal{H}}; \mathcal{H}) = Q_d(\hat{\mathcal{H}}; \mathcal{H}) - Q_d(\hat{\mathcal{H}}; \mathcal{H}) + Q_S(\hat{\mathcal{H}}; \mathcal{H}) + Q_T(\hat{\mathcal{H}}; \Phi)
\]

(56)

where \( Q_d() \) and \( Q_d() \) are the numerator and denominator auxiliary functions, \( Q_S(\hat{\mathcal{H}}; \mathcal{H}) \) is a smoothing function which is needed to guarantee the convexity of \( Q_{\text{aux}} \) and \( Q_T(\hat{\mathcal{H}}; \Phi) \) is an I-smoothing prior which is used to reduce the risk of parameter over-training [20]. These terms have the form of a Normal-Wishart distribution

\[
Q_S(\hat{\mathcal{H}}; \mathcal{H}) = -\frac{1}{2} \sum_{m} D^{(m)} \left\{ \log |\Sigma_x^{(m)}| + \text{trace} \left[ \Sigma_x^{(m)} \Sigma_x^{(m)-1} \right] \right. \\
+ \left. (\mu_x^{(m)} - \hat{\mu}_x^{(m)})^T \Sigma_x^{(m)-1} (\mu_x^{(m)} - \hat{\mu}_x^{(m)}) \right\}
\]

(57)

and a similar expression can be written for \( Q_T(\hat{\mathcal{H}}; \Phi) \) with I-smoothing constant and priors given by \( \Phi = \{\tau_p, \mu_p^{(m)}, \Sigma_p^{(m)}\} \). In this work dynamic MMI estimates are used as the I-smoothing prior.

The extension of discriminative training to JUD and VTS discriminative adaptive training was derived in [3] where the \( Q_d() \) and \( Q_d() \) terms in Equation 56 are obtained from the EM-based adaptive training auxiliary function in Equation 44. This yields the following expressions for the new canonical model parameters estimates\(^{11}\)

\[
\mu_x^{(m)} = \frac{\sum_{s,t} \gamma_{t}^{(sm)} \mathcal{E}[y_t^{(s)}, s, m, \mathcal{H}] + D^{(m)} \mu_x^{(m)} + \tau_p \mu_p^{(m)}}{\sum_{s,t} \gamma_{t}^{(sm)} + D^{(m)} + \tau_p}
\]

\[
\Sigma_x^{(m)} = \frac{\sum_{s,t} \gamma_{t}^{(sm)} \mathcal{E}[xx^T | y_t^{(s)}, s, m, \mathcal{H}] + D^{(m)} \Sigma_x^{(m)} + \tau_p \mu_p^{(m)}}{\sum_{s,t} \gamma_{t}^{(sm)} + D^{(m)} + \tau_p} - \hat{\mu}_x^{(m)} \hat{\mu}_x^{(m)^T}
\]

(58)

(59)

where the expectations are obtained from Equations 50 and 51, as detailed in Section 4.2.1 for JUD and Section 4.2.2 for VTS compensation. The terms \( L_x^{(m)} \) and \( L_p^{(m)} \) are defined as follows

\[
L_x^{(m)} = \Sigma_x^{(m)} + \mu_x^{(m)} \mu_x^{(m)^T}
\]

\[
L_p^{(m)} = \Sigma_p^{(m)} + \mu_p^{(m)} \mu_p^{(m)^T}
\]

(60)

The occupancy probability \( \gamma_{t}^{(sm)} \) is defined as \( \gamma_{t}^{(sm)} = \gamma_{t}^{(sm), n} - \gamma_{t}^{(sm), d} \) where the numerator and denominator “posteriors” are obtained by running the forward-backward algorithm using the set of hypotheses \( \mathcal{H} \) and the reference \( \mathcal{H}_{\text{ref}}^{(s)} \) respectively. Note that in order for VTS and JUD discriminative adaptive training to work, each feature vector sequence \( Y^{(s)} \) needs to represent an acoustically homogeneous block of data, on which noise compensation can be applied consistently.

The constant \( D^{(m)} \) is a component-specific smoothing constant, and it is a critical value in MPE discriminative training. It is responsible for making the weak-sense auxiliary function concave whilst ensuring rapid and stable updates. Following [20], this is set on a per-component basis as

\[
D^{(m)} = \max \left\{ E \sum_{s=1}^{S} \sum_{t=1}^{T} \gamma_{t}^{(sm), 2} D^{(m)} \right\}
\]

(61)

\(^{10}\)The numerator and denominator terms refer to the phone-marked lattices used to obtain the state/component “occupancies” based on the reference, and on the most likely hypotheses, respectively.

\(^{11}\)This is also the approach used in [12] for NCMLLR based discriminative adaptive training.
$D_{\text{min}}^{(m)}$ is the minimum value required to ensure that the covariance matrix of component $m$ is semi-positive definite and $E$ is an empirically set constant. The advantage of this form of discriminative adaptive training over NCMLLR [12] is that predictive approaches such as JUD and VTS require less data to estimate their parameters and can therefore be applied on shorter utterances [3].

A second-order gradient ascent approach could be used to maximise Equation 56, as it was for ML adaptive training in Section 4.1. However in this case the Hessian of the complete expression in Equation 56 needs to be negative definite to ensure that the auxiliary function is concave. This can be achieved, considering for example the simpler diagonal covariance case, by setting

$$D^{(m)} > (\sigma_{x_i}^{(m)})^2 \left\{ \frac{\partial^2 Q_a}{\partial (\sigma_{x_i}^{(m)})^2} \bigg|_\mathcal{M} + \frac{\partial^2 Q_d}{\partial (\sigma_{x_i}^{(m)})^2} \bigg|_\mathcal{M} + \frac{\partial^2 Q_t}{\partial (\sigma_{x_i}^{(m)})^2} \bigg|_\mathcal{M} \right\}$$ (62)

where a simple expression can be obtained for the derivative of the smoothing function. Though the first two terms can be obtained as in Section 4.1, exact expressions are obtained only when JUD compensation is applied (see Section 4.1.1). For the VTS case approximations need to be applied as shown in Equations 37 and 38.

Furthermore the second-order approach is not guaranteed to converge and, in contrast to the ML case, the auxiliary function $Q_{\text{aux}}()$ is only locally a strict lower bound for the objective function.

The above issues mean that the second-order approach may require careful fine-tuning of the learning rate as well as back-off procedures (where large amount of statistics need to be stored), in contrast to the simpler EM-based scheme. For this reasons in this work the EM-based approach, which has been found to yield robust estimates, is used for MPE VTS and JUD adaptive training. \textsuperscript{12}

Ideally, as for the ML adaptive training case, new estimates for both the transforms and the canonical model parameters would be estimated using MPE. However the lack of the correct transcription during testing in an unsupervised fashion makes the MPE approach more difficult to apply for transform parameter estimation. Therefore here a simplified approach is employed, where the ML scheme is used to obtain the initial transforms [28]. These are then fixed and discriminative adaptive training is applied iteratively to refine the estimate of the canonical model only.

6 Noise Estimation

The approaches discussed in previous sections assumed that the noise model $\mathcal{M}_a = \{\mu_t, \Sigma_t, \mu_b\}$ necessary to compute the transform parameters $T$ is known. In practice this is often estimated from the test data and a possible approach is to use ML noise estimation, which maximises the auxiliary function in Equation 29 with respect to the noise parameters while fixing the (canonical) model parameters $\mathcal{M} = \mathcal{M}$.

As in the case of the adaptive training approaches described above, it is possible to maximise the auxiliary function Equation 29 for diagonal JUD and VTS using either a second-order gradient-ascent scheme [10, 16] or an EM-based approach [5, 9, 13]. Both schemes are described in the following sections using the compensation framework of Section 2.1.

As stated in Section 2, the additive noise is assumed to be stationary, and the convolutional noise constant. This implies that only the static coefficients need to be estimated for the noise means. Also it is assumed that the vector components are uncorrelated and the additive noise covariance cross-terms are consequently set to zero. For the above reasons $\mathcal{M}_n = \{\mu_t, \Sigma_t, \mu_b\}$ is

\textsuperscript{12} An alternative approach could be obtained using a similar framework as in [30] where $Q_{\text{aux}}()$ and $Q_t$ are function of the corrupted speech parameters. However, though this approach would give closed-form update formulae for the corrupted speech parameters and ensure that $\Sigma_{x_i}^{(m)}$ are semi-positive definite, no constrain would be set on $\Sigma_{x_i}^{(m)}$ which may result negative-definite.
used in the following to indicate the static noise components, and only their update expressions are given. The expressions for the dynamic components of $\Sigma_z$ can then be obtained in a similar way.

### 6.1 Second-order gradient ascent approach

In this section a second-order gradient ascent scheme to estimate the new noise parameters $\hat{\mathcal{M}}_n = \{\hat{\mu}_z, \hat{\Sigma}_z, \hat{\mu}_h\}$ is described [10, 16]. Using a similar approach to that of Section 4.1, the following auxiliary function can be derived for JUD compensation (cf. Equation 11) where a single block of homogeneous data is considered and the model parameters are fixed ($\mathcal{M} = \mathcal{M}$ not shown for simplicity)

$$Q(\hat{\mathcal{M}}_n; \mathcal{M}_n) = -\frac{1}{2} \sum_{t=1}^{T} \sum_{m=1}^{M} \gamma_t^{(m)} \log \left\{ \mathcal{N}(y_t; \textbf{H}^{(r)} \mu_\gamma^{(m)} + \textbf{g}^{(r)}, \textbf{H}^{(r)} \Sigma_\gamma^{(m)} \textbf{H}^{(r)T} + \Psi^{(r)} \right\}$$ (63)

where the diagonal transform parameters $\mathcal{T}^{(r)} = \{\textbf{H}^{(r)}, \textbf{g}^{(r)}, \Psi^{(r)}\}$ are obtained as specified in Section 2.1.1 but based on the following base-class parameters: \[^{13}\]

$$\begin{align*}
\mu^{(r)}_x &= f(\mu_x^{(r)}, \mu_h, \mu_z) + J^{(r)}_x(\mu_h - \mu_h) + J^{(r)}_z(\mu_z - \mu_z) \\
\Sigma^{(r)}_y &= \text{diag}(J^{(r)}_x \Sigma_x^{(r)} J^{(r)}_x^T + J^{(r)}_z \Sigma_z J^{(r)}_z^T)
\end{align*}$$ (64) (65)

The auxiliary function for estimating the noise parameters when VTS compensation is applied can be obtained from Equation 63 by setting $R = M$. The same result is obtained using diagonal VTS compensation (cf. Equation 17), which computes $p(y|m) = \mathcal{N}(y; \mu_y^{(m)}, \Sigma_y^{(m)})$ with parameters given by Equations 64 and 65 evaluated at the component level. Therefore only noise estimation equations based on the more general JUD compensation approach are given below.

Equation 63 can be maximised using a second-order gradient ascent approach similar to that of Equation 30. However in contrast to the canonical model estimation case, for noise estimation the transforms $\mathcal{T}$ and gradients $\mathbf{J}$ always depend on the noise parameters to be estimated. This makes the exact derivation of the partial derivatives difficult to obtain. A possible solution is to apply the approximation that the transforms and gradients are independent of the noise parameters [10, 16]. This can be written as

$$\frac{\partial Q}{\partial \mathcal{M}_n} \approx \frac{\partial Q}{\partial \mathcal{M}_n} \Big|_{\mathcal{T} = \text{const.}}, \frac{\partial Q}{\partial \mathcal{M}_n} \Big|_{\mathbf{J} = \text{const.}},$$ (66)

where the transforms $\mathcal{T}$ and the Jacobians $\mathbf{J}$ are fixed when the derivatives with respect to the noise parameters are computed.

The consequence of the above approximation is that the noise update formulae obtained are not guaranteed to increase the auxiliary function in either the JUD or the VTS case. Furthermore, as discussed in Section 4.1, the second-order approach too is not guaranteed to increase the auxiliary function. Therefore it is very important to check the consistency of the solutions obtained in this case. To address this problem, estimation of noise means and additive noise variance is interleaved, at each iteration backing-off the new estimates to the current noise parameters until the “true” auxiliary function is increased [16]. This value is obtained using Equations 64 and 65 evaluated at the updated noise parameters (which also implies updating $f()$ and $J$).

In the next two sections schemes to estimate the noise means $\{\hat{\mu}_z, \hat{\mu}_h\}$ and noise variance $\Sigma_z$ are presented.

\[^{13}\]These expressions are obtained by computing the expectations in Equations 7 and 8 with respect to the new values of the noise variables.
6.1.1 Noise means estimation: \( \hat{\mu}_z, \hat{\mu}_h \)

When the noise means are estimated the noise variance is fixed. The derivatives of the auxiliary function in Equation 63 with respect to \( \{\mu_z, \mu_h\} \) are computed and the result is set to zero. Using the approximation in Equation 66, and setting \( \Sigma_z = \Sigma_z \) in Equation 65, the derivative with respect to \( \mu_z \) is given by

\[
\frac{\partial Q}{\partial \mu_z} = \sum_{m=1}^{M} \sum_{t=1}^{T} \gamma_t^{(m)} \left( J_2^{(r)^T} \Sigma_y^{(m)-1} (y_t - \mu_y^{(m)}) \right) = -d - E \hat{\mu}_z - F \hat{\mu}_h
\]

where the vector \( d \) and matrices \( E, F \) are functions of the current noise parameters and are defined as

\[
d = \sum_{m=1}^{M} J_2^{(r)^T} \Sigma_y^{(m)-1} \sum_{t=1}^{T} \gamma_t^{(m)} (y_t - H_t^{(r)}(\mu_x^{(m)} - \mu_x^{(r)})) - f(\tau^{(r)}) + J_1^{(r)} \mu_h + J_2^{(r)} \mu_z
\]

\[
E = \sum_{m=1}^{M} J_2^{(r)^T} \Sigma_y^{(m)-1} J_2^{(r)}, \quad F = \sum_{m=1}^{M} J_2^{(r)^T} \Sigma_y^{(m)-1} J_1^{(r)}
\]

Similar expressions for \( \partial Q / \partial \mu_h \) as a function of a vector \( u \) and matrices \( V, W \) can be obtained\(^{14}\). Equating the derivatives to zero provides a system of linear equations whose solution is

\[
\hat{\mu}_z = (V - WF^{-1}E)^{-1}(u - WF^{-1}d)
\]

A similar expression is obtained for \( \hat{\mu}_h \).

Noise estimation based on diagonal JUD compensation is particularly efficient since the diagonal transform parameters are computed at the base-class level. This is the approach used in [3, 5].

When diagonal VTS noise estimation is used the same expressions originally proposed in [16] are obtained. These extend the point estimate technique used in [19] for MMSE-based feature compensation. Though at the expense of higher complexity since parameters are now computed for each system component, the second-order VTS noise estimation approach has proved to be very effective [3, 10, 16]\(^{15}\).

6.1.2 Noise variance estimation: \( \Sigma_z \)

To estimate the additive noise covariance matrix, the noise means are kept constant and the following second-order approach is used

\[
\Sigma_z = \Sigma_n + \zeta \left[ \frac{\partial^2 Q}{\partial \Sigma_n \partial \Sigma_n} \right]_{M_n} \left[ \frac{\partial Q}{\partial \Sigma_n} \right]_{M_n}
\]

where \( \zeta \) is the learning rate, and the derivatives are evaluated at the current noise parameters \( M_n \).

As for the noise means estimation approach, the approximation in Equation 66 is applied to compute the first and second derivatives in Equation 71. \( \Sigma_z \) is diagonal, and the following

\(^{14}\)The full derivation for VTS compensation can be found in [15].

\(^{15}\)A slightly simplified approach is used in [10] where the matrices \( F \) and \( V \) above are set to zero. In addition both the noise means and variance are estimated at the same time and it is not clear whether some back-off related approach is applied to guarantee that the auxiliary function is actually maximised.
expressions can be obtained for each diagonal element

\[
\frac{\partial Q}{\partial \sigma_{zi}^2} = -\frac{1}{2} \sum_{m=1}^{M} \sum_{d=1}^{D} \frac{J_d^{(r)}(m)^2}{\sigma_{yd}^2} \sum_{t=1}^{T} \gamma_t(m) \left\{ 1 - \frac{(y_{td} - \mu_{yd})^2}{\sigma_{yd}^2} \right\}
\]  

(72)

\[
\frac{\partial^2 Q}{\partial (\sigma_{zi}^2)^2} = \sum_{m=1}^{M} \sum_{d=1}^{D} \left( \frac{J_d^{(r)}(m)^2}{\sigma_{yd}^2} \right)^2 \sum_{t=1}^{T} \gamma_t(m) \left\{ 1 - \frac{(y_{td} - \mu_{yd})^2}{\sigma_{yd}^2} \right\}
\]  

(73)

where \( J_{d}^{(r)} \) indicates the element on the \( d \)-th row and \( i \)-column of \( J_{d}^{(r)} \), and \( D \) is the dimension of the feature vector. Given fixed noise means, multiple iterations of additive noise variance estimation can be run, back-off the covariance estimate to ensure the auxiliary function is actually increased at each iteration.

Depending on the form of compensation used, the above expressions will provide noise variance estimation based on diagonal JUD [3] or on diagonal VTS [16, 10].

JUD-based noise estimation based on the second-order approach above was initially used in [3], which extends the work in [16] where a numerical method is used to estimate the derivatives. In both approaches a back-off procedure is applied to ensure the auxiliary function is maximised.

6.2 EM based approach

An alternative to the second-order approach for estimating the noise parameters is possible. In [13] EM-based solutions are obtained for all static noise parameters, which are used for both MMSE-based feature enhancement and VTS model compensation. A similar approach was also employed in [9] where EM-based VAT is implemented.

Both approaches above can be reformulated in terms of the framework defined in Section 3 where JUD and VTS compensation are expressed in terms of linear generative processes [5]. This framework allows a better investigation of the limitations and constraints that need to be set so that the noise parameters are correctly estimated.

Applying the same steps used to obtain the equations for JAT-EM and VAT-EM in Section 4.2 it is possible to derive the following auxiliary function

\[
Q(\hat{\mathbf{M}}_a; \mathbf{M}_a) = \sum_{t=1}^{T} \sum_{m=1}^{M} \gamma_t(m) \mathcal{E} \left[ \log p(y_t|z, m, \hat{\mathbf{M}}_a) + \log p(z|m, \hat{\mathbf{M}}_a) \right]
\]  

(74)

To obtain the update expression for the noise parameters the derivative of the auxiliary function with respect to the noise parameters \( \{\hat{\mu}_z, \Sigma_z\} \) is set to zero. Under the hypothesis that the first term in the right hand side of the expression above is independent of the additive noise parameters the following update formulae are obtained

\[
\hat{\mu}_z = \frac{\sum_t \gamma_t(m) \mathcal{E} \left[ y_t | z, m, \mathbf{M}_a \right]}{\sum_t \gamma_t(m)}
\]

(75)

\[
\Sigma_z = \text{diag} \left( \frac{\sum_t \gamma_t(m) \mathcal{E} \left[ z z^\top | y_t, m, \mathbf{M}_a \right] - \hat{\mu}_z \hat{\mu}_z^\top}{\sum_t \gamma_t(m)} \right)
\]

(76)

The convolutional noise mean is an additional parameter to be estimated and is obtained by taking the derivative of \( Q() \) with respect to \( \hat{\mu}_b \). An approach interleaving the estimation of additive and convolutional noise parameters is described below.

To compute the expectations in Equations 75 and 76 the JUD or VTS generative model described in Section 3 can be used. Here the more general diagonal JUD generative model in Equation 25 is used, VTS being a special case of this.
Considering fixed clean model parameters, it is possible to rewrite Equation 25 to give
\[
y|m, r = \Lambda^r_z z + C^r \mu_h + \epsilon^r
\]  
(77)
where \(C^r = J^r_z\) and \(\Lambda^r_z = \text{diag}(J^r_z)\). The noise and error terms are Gaussian distributed as 
\[z \sim \mathcal{N}(\mu_z, \Sigma_z), \; \epsilon^r \sim \mathcal{N}(\mu^r_e, \Sigma^r_e),\]  
with parameters defined as
\[
\mu^r_e = f(\mu^r_z, \mu_h, \mu_z) + H^r(\mu^r_z - \mu^r_z) - \Lambda^r_z \mu_z - C^r \mu_h
\]  
(78)
\[
\Sigma^r_e = H^r(\Sigma^r_z - \text{diag}(\Sigma^r_z))^r + \text{diag}(J^r_z \Sigma^r_z J^r_z)^T
\]  
(79)

Two requirements were discussed in Section 3 for the above generative model to be consistent and to provide simple solutions. Setting the loading matrix above to be diagonal provides diagonal compensated covariances thus satisfying the second requirement.

The first requirement requires that the generative model parameters \(\Lambda^r_z, C^r\) and the error term parameters are constant, so that the update formulae above guarantee an increase of the auxiliary function. However this is not possible since these parameters always depend on the noise parameters to be estimated.

A possible solution is to consider the generative model parameters to be constant, as in the VAT-EM case in Section 4.2.2. However, and in contrast to the VAT-EM case, preliminary experiments, as well as results reported in [30], demonstrated that this approximation has a large impact on the accuracy of the estimates. Though backing-off the noise estimates until they actually increase the auxiliary function could address the above issue, the additional computational load required would counteract the advantages provided by closed-form solutions.

The approach in [5], which relates to Jacobian compensation [24], is used here. This keeps the parameters of \(\Lambda^r_z, C^r\) and the error term constant throughout the noise estimation process so that Equation 77 represents a valid generative model\(^{16}\) and the estimates obtained from Equations 75 and 76 guarantee the log-likelihood will increase.

The necessary statistics are obtained from the following expressions, which are valid for jointly Gaussian multivariate variables
\[
\mathcal{E}[z|y_t, m, M_n] = \mu^r_z + \Sigma^r_z y_t^T = \mu^r_z
\]  
(80)
\[
\mathcal{E}[zz^T|y_t, m, M_n] = \Sigma^r_z - \Sigma^r_z y_t y_t^T + \text{diag}(\mu^r_z y_t y_t^T)
\]  
(81)
where the noise and corrupted speech joint distribution parameters are obtained from the generative model based on the current noise parameters \(M_n\). These are defined as
\[
\mu^r_y = \Lambda^r_z \mu_z + C^r \mu_h + \mu^r_e
\]  
(82)
\[
\Sigma^r_y = \Lambda^r_z \Sigma^r_z \Lambda^r_z^T + \Sigma^r_e
\]  
(83)
\[
\Sigma^r_yz = \Sigma^r_yz^T = \Lambda^r_z \Sigma_z
\]  
(84)
where the covariances are diagonal.

Multiple iterations can be run updating the additive noise parameters \(\{\mu_z, \Sigma_z\}\) in the above expressions at each iteration. Once convergence is reached, the convolutional noise mean is estimated. For this the term \(p(y_t|z, m, M_n)\), which depends on \(\mu_h\) in Equation 74, is considered. This conditional probability is obtained from the generative model and is Gaussian distributed as follows
\[
y_t|z, m, M_n \sim \mathcal{N}(\mu^r_z, \Sigma^r_z)
\]  
(85)
with parameters given by
\[
\mu^r_y = \Lambda^r_z \mu_z + C^r \mu_h + \mu^r_e, \quad \Sigma^r_yz = \Sigma^r_e
\]  
(86)
\(^{16}\)The constant generative parameters make the term \(p(y_t|z, m, M_n)\) independent of \(\{\mu_z, \Sigma_z\}\) in Q().
Substituting Equation 85 into the auxiliary function and setting the derivative with respect to \( \hat{\mu}_h \) to zero gives the following update formula

\[
\hat{\mu}_h = \left( \sum_{t,m} \gamma_t^{(m)} C^{(r)T} \Sigma_{\epsilon t}^{(m)-1} C^{(r)} \right)^{-1} \left( \sum_{t,m} \gamma_t^{(m)} C^{(r)T} \Sigma_{\epsilon t}^{(m)-1} \left( y_t - A_z^{(r)} \hat{\mu}_z^{(m)} - \mu_\epsilon^{(m)} \right) \right)
\]  

(87)

where the full matrix inversion in the first set of brackets needs to be computed only once since it depends on constant terms.

Multiple iterations can be run, until convergence, to refine the estimate of the convolutional noise mean, while \( \{ \mu_x, \Sigma_{\epsilon t} \} \) are fixed. At each iteration Equation 82 is updated with the updated value of \( \mu_h \), and is used in turn to compute Equation 80. This gives the new value of \( \mu_{z[y]}^{(m)} \) in Equation 87.

Estimation of additive and convolutional noise parameters is interleaved until the auxiliary function value provided by the complete set of updated noise parameters fails to increase by a certain threshold.

When the above expressions are used for noise estimation based on diagonal VTS compensation (\( R = M \)), the terms related with \( H^{(r)} \) in Equations 78 and 79 cancel out providing the same final form that can be obtained using the VTS generative model described in Section 3.

The advantage of the EM-based approach over the second-order approach for noise estimation is that the former is more computationally efficient and therefore faster. This was made possible by fixing the generative model parameters, allowing simple update expressions to be obtained. However this introduces a mismatch between the fixed parameters, which are based on the initial noise values, and those used for decoding, which are computed using the updated noise estimates. To address this issue a modified compensation form is proposed in the following section.

### 6.2.1 FA-Style Compensation

Both the second-order and EM-based noise estimation approaches described in previous sections are based on maximising an auxiliary function for a given form of model compensation. If a different form of compensation is used during recognition a mismatch occurs, with detrimental effects on recognition accuracy.

This is not a problem for the second-order approaches discussed previously, as the noise parameters are backed-off until they maximise the “true” auxiliary function. However if standard JUD compensation based on noise parameters obtained by applying the EM-based approach described in the previous section is used, a large mismatch occurs. Back-off could still be applied in this case, but the large mismatch would require several iterations that would negate the computational benefits provided by the EM-based method.

To address the above problem the modified FA-JUD compensation form [5], which is also related to Jacobian compensation [24], is used. This is obtained using the generative model in Equation 77 based on the same initial set of noise parameters \( M_n \) used for noise estimation. Given the updated noise parameters \( M_n = \{ \hat{\mu}_z, \Sigma_{\epsilon t}, \hat{\mu}_h \} \), it is interesting to contrast FA-JUD compensation with second-order JUD. Considering, for example, the compensated mean \( \mu_y^{(m)} \), it is possible to write

\[
\text{JUD: } \hat{\mu}_y^{(m)} = \hat{A}^{(r)-1}(\mu_x^{(m)} - \mu_\epsilon^{(r)}) + f(\mu_x^{(r)}, \hat{\mu}_h, \hat{\mu}_z) \quad (88)
\]

\[
\text{FA-JUD: } \hat{\mu}_y^{(m)} = A^{(r)} \hat{\mu}_z + C^{(r)} \hat{\mu}_h + \mu_\epsilon^{(m)} \quad (89)
\]

where the JUD feature transform matrix \( \hat{A}^{(r)} \) is obtained from \( \hat{M}_n \), while the FA-JUD parameters are computed using \( M_n \), as detailed in the previous section. Thus FA-style compensation has the following advantages and limitations

21
1. FA-JUD is *computationally efficient*. The same linear expression is used for both noise estimation, Eq. 77, and model compensation, Eq. 89. By fixing the generative model parameters, the non-linear term $f(\cdot)$ is computed only once for each base-class $r$, whereas it must be recomputed for each new noise estimate with second-order JUD in Eq. 88. Furthermore, EM-based noise estimation, and multiple EM iterations if desired, can be applied without the need of expensive back-off procedures.

2. The compensated FA-JUD and JUD means in Equations 88 and 89 are the same when $M_n = M_n$\(^{17}\). However, FA-JUD will result in a *worse approximation* to the mismatch function since the "distance" between the noise parameters being estimated $M_n$ and those used for the expansion point $M_b$ increases.

FA-VTS compensation is obtained by setting $R = M_n$. The combined EM-based noise estimation and FA-style model compensation scheme will be referred to as FA-JUD or FA-VTS below.

Compared to the second-order approach, FA-JUD and FA-VTS are more computationally efficient. However this computational efficiency comes at the expense of a sensitivity to the expansion point used. As shown in the experimental section this represents a problem when a large mismatch exists between the actual noise parameters and the expansion point used (obtained from the current noise estimates). In the next section a combined FA-JUD/JUD incremental decoding scheme which addresses this problem is presented.

There are some important differences between the above approach and Jacobian compensation [24]. In Jacobian compensation: a) only the additive noise effects are taken into account and no channel distortion is considered; b) the expansion point is fixed at training time and is obtained from the noise parameters used to derive noisy training data from clean speech, which restricts the class of possible noise types which the system can deal with during decoding to those which are not too "different" to the noise used during training; c) no "Jacobian compensation"-related noise estimation is used, since the noise parameters are known in advance from training. Only the means are updated in [24] since no significant improvement is observed when the variances are compensated too. This may be due to the fact that the mismatch function in Equation I was shown not to work well on multi-style trained systems [3].

### 6.2.2 FA-JUD/JUD Incremental Decoding

The FA-JUD (or FA-VTS) approach described in the previous section has the advantage of being a fast noise estimation scheme. This comes at the expense of being sensitive to the nature of the noise $M_n$ used for the generative model parameters. When the "distance" between the new noise values being estimated $M_n$ and $M_b$ becomes too large, FA-JUD will not accurately estimate the noise model or the compensation parameters. If it is possible to detect when this happens, noise estimation can be backed-off to the second-order JUD approach. This provides reliable noise estimates which can be used to update the FA-JUD parameters. This mode of operation is suitable for incremental style adaptation where the statistics obtained from decoding a given utterance are used to refine the noise model which is used for decoding the next utterance. By detecting how far the new estimate is from that used to obtain the generative model parameters, it is possible to detect when the noise has drifted too far.

The auxiliary function can be used to give a measure of this noise "distance" [5]. Given $M_n$ and noise parameters $M_{n,i}$ estimated from utterance $i$, the following expression may be used as a measure of this distance
\[
\Delta Q = Q(M_{n,i}, M_n) - Q(M_n, M_n)
\]  
(90)

Alternative schemes are possible. For example the KL divergence between models compensated with $M_{n,i}$ and $M_n$ may be used. However Eq. 90 is fast to compute, as it is a by-product of the noise estimation process. This combined FA-JUD/JUD incremental adaptation scheme is shown

\(^{17}\) A different form of $\Sigma^{(m)}_n$ is defined in the next section so that FA-JUD and JUD also provide the same covariances when $M_n = M_b$. 

22
in Figure 2 and can be summarised as follows:

1. **Initialisation**: set the current noise model \( \mathcal{M}_n \) to an initial set of noise parameters \( \mathcal{M}_n = \mathcal{M}_{n,0} \) and update the FA-JUD parameters accordingly.

2. **For each utterance**: \( i = 1 \) to \( N \), \( \mathbf{Y}^{(i)} \):

   (a) **Hypothesis generation**: using models adapted with \( \mathcal{M}_{n,i-1} \) to decode utterance \( i \), accumulating statistics;

   (b) **Noise estimation (a)**: use FA-JUD to estimate noise parameters \( \mathcal{M}_{n,i} \) and compute \( \Delta Q \);

   (c) **Noise estimation (b)**: if \( \Delta Q > \alpha \) use JUD to re-estimate \( \mathcal{M}_{n,i} \) and set \( \mathcal{M}_n = \mathcal{M}_{n,i} \), updating the FA-JUD parameters accordingly;

   (d) Set \( i = i + 1 \).

For consistency, when the current noise parameters \( \mathcal{M}_n \) are updated at step (c) the same second-order JUD approach, rather than FA-JUD, should be applied in the following step (a). However this will provide the EM-based noise estimation block with statistics \( \gamma_i^{(m)} \) different than those obtained if FA-JUD compensation were used. As was shown in the previous section the reason for this is that though when \( \mathcal{M}_{n,i-1} = \mathcal{M}_n \) JUD and FA-JUD provide the same compensated means, \( \Sigma_i^{(m)} \) will be different. This issues are easily solved by modifying Equation 70 as follows

\[
\Sigma_i^{(m)} = \Sigma_i^{(m)} + \Delta^{(r)}
\]

(91)

where the constant parameter \( \Delta^{(r)} \) is based on the current noise parameters and is defined as

\[
\Delta^{(r)} = \text{diag}(J_z^{(r)} \Sigma_z^{(r)} J_z^{(r)T}) - \text{diag}(J_z^{(r)} \Sigma_z^{(r)} \text{diag}(J_z^{(r)})^T)
\]

(92)

If Equation 91 is used FA-JUD and second-order JUD are identical when \( \mathcal{M}_{n,i-1} = \mathcal{M}_n \). This allows us to always apply FA-JUD compensation in step (a) above, providing consistency when noise-estimation is backed-off to the second-order JUD approach. This is the approach used in [5] and in this work, which allows tracking the noise conditions and automatically choosing between the two noise estimation approaches: the faster and less accurate FA-JUD, or the slower and more accurate JUD. The trade-off between speed and accuracy can be controlled by setting the threshold \( \alpha \).

Note that statistics can be accumulated from several consecutive utterances and a smoothing factor can be used to control the degree of influence of past utterances [3]. This is applied to the statistics using

\[
\mathcal{O}^{(i)} = \beta \mathcal{O}^{(i-1)} + \tilde{\mathcal{O}}^{(i)}
\]

(93)

where, for utterance \( i \), \( \tilde{\mathcal{O}}^{(i)} \) are the statistics collected and \( \mathcal{O}^{(i)} \) are the statistics actually used to estimate the noise transform. However in order to precisely evaluate the advantages of the combined incremental approach above in this work the accumulated statistics are reset after each utterance is processed (\( \beta = 0 \)).
7 Experiments and Results

To evaluate the proposed approaches two different configurations were used in this work: the Aurora4 database based on WSJ0, and an in-car recognition task provided by TREL [3]. Further details of each task and the related experiments are discussed below. In this section common aspects are discussed.

The parameterisation used consisted of 12 MFCCs plus zeroth cepstrum, delta and delta-delta coefficients. Decision-tree-clustered states and cross-word triphone models were used to train the systems. This resulted in approximately 50,000 and 8,000 model components for Aurora4 and TREL, respectively. For JUD and JAT systems the components were clustered into 512 base-classes for Aurora4 and 64 for TREL, which resulted in roughly 1/100 the number of components for the two configurations. This consistently reduced the computational load compared to VTS compensation.

For adaptive training, given the results presented in Section 4.3, the JAT-EM and VAT<sub>TS</sub> schemes are used, and they will simply be referred to as JAT and VAT, respectively. For these schemes noise parameter estimation is required during both training and recognition. However the computational load is less important during the training phase as this is done off-line. Thus to compute the set of training noise transforms the more accurate second-order JUD and VTS noise estimation approaches are used. For testing both the second-order and EM-based noise estimation approaches are evaluated.

The above VAT and JAT systems also provide the initial canonical models and training noise transforms for the EM-based MPE adaptively trained systems. In this case, as discussed in Section 5, the set of transforms are fixed and multiple iterations of canonical models estimation are applied<sup>18</sup>.

The performance of the systems were evaluated using both batch- and incremental-mode adaptation on a per-utterance basis. Batch-mode adaptation is a standard form used for example in [10, 11, 15], but may not be suitable for some applications as it introduces latencies when multiple hypothesis and transforms updates are required. Incremental-mode adaptation was found to reduce such latencies but requires test data to be collected in a causal fashion [4].

During recognition for batch-mode, the first and last 20 frames (20±20) of each utterance were used to obtain the initial additive noise estimates, while the convolutional noise mean was set to zero (μ<sub>n</sub> = 0). In the case of the clean systems the initial hypothesis was provided by compensating the models with the above noise estimates and then decoding. When adaptively trained systems were used the initial hypothesis was obtained using an uncompensated multi-style system.

For the incremental experiments initial transforms for the first utterance of each speaker are required. These were obtained using a single iteration of batch-mode adaptation on the first utterance.

7.1 Aurora4

Two Aurora4 training sets, each of 7138 utterances (12 hours) and recorded at 16kHz, were considered. The first used data recorded with a single close-talk microphone (Mic1). This was the training data for the clean system. The second set comprises multi-condition and multi-channel (Mic1 and Mic2) data and was used to train the multi-style and adaptively trained systems. The recognition task is a 5K-word dictation task with 14 test sets, each comprising 330 utterances. Sets 01-07 were recorded with Mic1 where different noise (Car, Babble, Restaurant, Street, Airport and Train station noise, respectively) with random SNR from 5 to 15 dB were added to sets 02-07. The same approach was used to obtain sets 08-14 but Mic2 was used instead. In the following, letters A, B, C and D will be used to indicate test sets 01, 02-07, 08 and 09-14 respectively.

Given that the noisy test sets are obtained by adding noise with SNRs varying randomly from utterance to utterance, Aurora4 is not suitable for incremental decoding which requires utter-

---

<sup>18</sup>In addition systems obtained using the other VAT and JAT schemes from Section 4 were employed as initial canonical model parameters for MPE/discriminative training. However, in accordance with the results of Section 4.3, the MPE systems obtained in this case performed worse.
ances to be recorded in a causal fashion. Therefore only batch adaptation is applied and, given the complexity of the tasks, 4 iterations of noise estimation are performed using the fixed initial hypothesis. Then, after the hypothesis is updated, another 4 noise estimation iterations are performed to provide the final transforms used for decoding.

The ARPA 1994 CSRNAB Spoke 10 (S10) task was also used to evaluate how the systems trained using the above Aurora4 training configuration perform on unseen channel and noise conditions\(^{10}\) when the JUD and VTS-based approaches are applied. The task is a 5K closed-vocabulary task where car noise has been artificially added to clean speech. The standard MIT Lincoln Labs 5K trigram language model was used. The development set was used, which consists of 108 sentences spoken by 10 different speakers recorded in clean conditions. Noise-corrupted data were generated by adding the car noise at average SNRs of 18, 24, 30 dB.

The first set of experiments compare the performance of batch noise adaptation on clean and adaptively trained systems. Results are shown in Table 1. The first row of the table shows the WERs obtained by applying JUD adaptation on the different tasks. Almost no improvement is obtained on the clean task A compared to using uncompensated clean models which provided 7.1% WER. On the noisy tasks JUD gives 19.90% average WER compared to 58.52% WER when no compensation was applied.

Further gains are obtained if the multi-style training data is used to adaptively train a JAT system. In this case a small deterioration is observed compared to JUD on clean microphone-matched task A, and on clean task C, which is affected by channel distortion. The reason for this is that the canonical model parameters are trained on the residual obtained after applying the noise transforms to the multi-style training data, resulting in broader covariances compared to clean trained systems. Though this behaviour is similar to that of multi-style trained models, the advantage of JAT is that noise compensation can be applied. This gave 17.87% WER providing 10% overall relative improvement over the JUD adapted clean models.

The same considerations mentioned above for JUD and JAT apply to VTS and VAT whose performance is shown in the last two rows of Table 1. These results can be considered as a lower bound on the WER for JAT, which can be achieved using JAT by increasing the number of base-classes to the limit \(R = M\). Though this makes the resulting approaches more computationally expensive, a higher accuracy is obtained resulting in a consistent reduction of the overall WER to 15.98% in the case of VAT.

The second set of experiments aims to compare systems trained with multi-style data using either Maximum Likelihood (ML) or MPE discriminative (adaptive) training. Results are shown in Table 2. The first block in the table compares a standard multi-style system (MST ML) with a system trained using discriminative training (MST MPE). No compensation is applied in either case, and consistent gains are provided by MPE over ML training in all tasks.

The advantages of using the MPE criterion during training are also evident when MPE adaptive training is used. In fact, MPE performed better than ML in all tasks, giving an overall relative

---

\(^{10}\)The ITU-T P.341 filter simulating wide-band hands-free telephony terminals was applied to the Aurora4 data. Also the car noise used to contaminate the Aurora4 training data was different to that used for S10.
Table 2: Maximum Likelihood (ML) and MPE discriminatively trained systems: unadapted multi-style (MST) system; JAT and VAT systems; and system trained on multi-style data pre-processed with the Advanced Aurora Front-End (MST-AFE).

<table>
<thead>
<tr>
<th>System</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>MST</td>
<td>ML</td>
<td>10.8</td>
<td>18.8</td>
<td>30.3</td>
<td>37.4</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>9.6</td>
<td>16.7</td>
<td>29.5</td>
<td>35.1</td>
</tr>
<tr>
<td>JAT</td>
<td>ML</td>
<td>7.9</td>
<td>14.4</td>
<td>13.0</td>
<td>23.8</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>7.5</td>
<td>13.3</td>
<td>12.4</td>
<td>22.8</td>
</tr>
<tr>
<td>VAT</td>
<td>ML</td>
<td>8.1</td>
<td>13.5</td>
<td>11.6</td>
<td>20.5</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>7.2</td>
<td>12.8</td>
<td>11.5</td>
<td>19.7</td>
</tr>
<tr>
<td>MST-AFE</td>
<td>ML</td>
<td>8.8</td>
<td>16.6</td>
<td>19.1</td>
<td>28.6</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>8.1</td>
<td>14.9</td>
<td>16.7</td>
<td>26.3</td>
</tr>
</tbody>
</table>

For comparison purposes, two systems using the ETSI advanced front-end [18] on the same MST training data were evaluated on this task. Results are reported in the last block of Table 2 and confirm the advantages of MPE training (MST-AFE MPE) which gave about 9% relative improvement over ML training (MST-AFE ML). However, though MST-AFE performed better than the MST systems, it was outperformed by the JAT and VAT systems. The latter VAT MPE yielded 21% WER relative improvement over MST-AFE MPE.

The same systems above were evaluated on the development S10 task. The objective of this experiment is to evaluate how these systems performed on mismatched training/testing conditions. Results are shown in Table 3.

Table 3: Maximum Likelihood (ML) and MPE discriminatively trained systems trained on the Aurora4 training data set and tested on the S10 task: unadapted multi-style (MST) system; JAT and VAT systems; and system trained on multi-style data pre-processed with the Advanced Aurora Front-End (MST-AFE).

<table>
<thead>
<tr>
<th>System</th>
<th>30dB</th>
<th>24dB</th>
<th>18dB</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>MST</td>
<td>ML</td>
<td>17.8</td>
<td>21.5</td>
<td>33.4</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>17.4</td>
<td>20.6</td>
<td>30.0</td>
</tr>
<tr>
<td>JAT</td>
<td>ML</td>
<td>14.7</td>
<td>16.1</td>
<td>21.4</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>14.7</td>
<td>16.9</td>
<td>21.4</td>
</tr>
<tr>
<td>VAT</td>
<td>ML</td>
<td>15.5</td>
<td>16.8</td>
<td>20.2</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>15.1</td>
<td>16.7</td>
<td>20.8</td>
</tr>
<tr>
<td>MST-AFE</td>
<td>ML</td>
<td>17.6</td>
<td>20.3</td>
<td>31.2</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>15.9</td>
<td>19.1</td>
<td>28.5</td>
</tr>
</tbody>
</table>

The first two rows of the table show the results obtained with the MST systems. For MPE the relative gain obtained over ML training is similar to that of Table 2, confirming the advantages of discriminative training. When adaptive training is applied a large gain over MST uncompensated systems is provided by both JAT-ML and VAT-ML. JAT-ML and VAT-ML achieved similar overall WER, though VAT-ML performed better on the lowest SNR. These adaptively trained systems however provided little improvement with respect to applying standard JUD and VTS compensa-

30The S10 development set was also used in preliminary experiments [not published] for NCMLLR compensation in [12]. In that work, an MST system trained on the SWJ S128i training data using 12 MFCCs plus energy and Cepstral Mean Subtraction (CMS) provided 13.2%, 14.5% and 21.4%, for the 30 dB, 24 dB, and 18 dB conditions, respectively. Using MST system trained on Aurora4 using the same coding provided 15.4%, 20.6% and 20.8% WER for the three noise conditions respectively.
tion on the clean trained systems, which resulted in a WER of 17.66% and 17.55% respectively. This can be explained by noting that model parameters in adaptive training are estimated on the residual obtained after JUD or VTS compensation is applied. In this experiment the channel and noise conditions of the S10 set are unseen during training, and as such the residual is only based on the noise conditions from Aurora4, which are different. This effect is even more evident for JAT-MPE and VAT-MPE, which performed slightly worse than JAT-ML and VAT-ML.

The MST-AFE system was also tested using the S10 task above and results are reported in the last two rows of the table. Though providing some gains over the MST baseline, the relative improvement was only about 5% and 6% WER, showing that this front-end does not generalise well in mismatched train/test conditions.

7.2 TREL

The TREL training data [3] comprises a total of about 486 hours of data, including artificially corrupted clean speech data with car noise added at a range of SNR levels and in-car collected data. Again both multi-style and adaptively trained systems were built using this data. Evaluation was carried out on an in-car recognition task consisting of four sub-tasks recorded using a microphone mounted on the rear-view mirror, with either the engine-on (ENON) or driving along a highway (HWAY). The SNR-levels were approximately 35 dB and 18 dB respectively. The four small-medium sized sub-tasks were: phone numbers (PH), four digits (4D), command & control (CC) and city names (CN). The first two contain digit sequences, the third contains in-car radio or navigator command and control sequences, and the fourth consists of single city name utterances. The test sets comprise about 30 speakers, each uttering 30 sentences (60 in the CC task). All test data was acquired in a causal fashion allowing decoding to be run in both batch and incremental modes. Since the task is simpler than the previous Aurora4 experiments, in batch mode only 2 iterations of noise estimation were performed, with the hypothesis updated after the first iteration. The same system types as those for Aurora4 were built on the TREL configuration and the results for each noise condition and task are shown in Table 4.

<table>
<thead>
<tr>
<th>System</th>
<th>ENON</th>
<th>HWAY</th>
<th>Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PH</td>
<td>4D</td>
<td>CC</td>
</tr>
<tr>
<td>MST</td>
<td>ML</td>
<td>1.1</td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>1.2</td>
<td>0.7</td>
</tr>
<tr>
<td>JAT</td>
<td>ML</td>
<td>0.7</td>
<td>0.4</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>VAT</td>
<td>ML</td>
<td>0.5</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td>MPE</td>
<td>0.3</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 4: Maximum Likelihood (ML) and MPE discriminatively trained systems: unadapted multi-style (MST) system; JAT and VAT systems.

Similar observations apply here as for the Aurora4 experiments above. The average WER for the MST-ML system was 2.43% WER and discriminative training gave a further WER relative reduction of more than 10% (this was also better than using a clean trained system with JUD and VTS compensation, which provided 2.40% and 2.27% WER respectively).

Using adaptive training yielded large improvements over the MST baseline for both the JAT and VAT systems, which reduced the WER by more than 40% relative and almost 50% relative respectively. MPE training further improved the performance and a more than 1% absolute WER reduction was given by JAT-MPE and VAT-MPE over MST-MPE. These relative improvements are larger than those observed using the Aurora4 dataset in the previous section. This effect was also observed in [5] and may be explained by the fact that, compared to the Aurora4 configuration, TREL has more training data and less complex models.
To evaluate the noise estimation approaches discussed in Section 6 incremental decoding was applied using the JAT-MPE system.

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>ENON WER back-off</th>
<th>ENON % JUD back-off</th>
<th>HWAY WER back-off</th>
<th>HWAY % JUD back-off</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.6</td>
<td>100</td>
<td>1.6</td>
<td>100</td>
<td>1.07</td>
</tr>
<tr>
<td>1</td>
<td>0.6</td>
<td>12</td>
<td>1.6</td>
<td>39</td>
<td>1.07</td>
</tr>
<tr>
<td>5</td>
<td>0.6</td>
<td>0</td>
<td>1.9</td>
<td>5</td>
<td>1.23</td>
</tr>
<tr>
<td>10</td>
<td>0.6</td>
<td>0</td>
<td>2.1</td>
<td>2</td>
<td>1.38</td>
</tr>
<tr>
<td>$\infty$</td>
<td>0.6</td>
<td>0</td>
<td>2.3</td>
<td>0</td>
<td>1.47</td>
</tr>
</tbody>
</table>

Table 5: FA-JUD/JUD incremental on TREL JAT-MPE system.

Results applying noise estimation using the standard second-order JUD approach and the combined FA-JUD scheme are shown in Table 5, where the parameter $\alpha$ described in Section 6.2.2 is varied. This allows to control how often the full standard JUD compensation and noise estimation is applied.

The baseline is obtained setting $\alpha = 0$, which results in the standard incremental JUD approach [9] and gives 1.07% WER\(^2\). Conversely setting $\alpha = \infty$ in the last row of the table yields pure FA-JUD noise estimation and compensation. This yielded 1.47% WER, confirming the sensitivity of FA-JUD to the noise estimate (obtained from the first utterance) used for the expansion point. Varying $\alpha$ controls the trade-off between speed and accuracy between the two extremes of FA-JUD and JUD. In this experiment setting $\alpha = 1$ yielded the same performance as standard JUD but the noise estimation stage was approximately twice as efficient. Note that also the model compensation stage is faster when FA-JUD is applied, compared to standard JUD.

8 Conclusions

This technical report has provided a review of the different existing approaches for JUD and VTS compensation applied to both noise estimation and adaptive training, as well as discriminative adaptive training. Both second-order and EM-based approaches can be used to maximise the auxiliary function related with the above schemes and different approximations are introduced depending on the form of compensation used and the maximisation approach employed. The effects of such approximations have been investigated in detail and a unified FA framework has been used to describe the EM-based approaches, allowing a more precise analysis of the constraints necessary for a correct derivation of the parameter update expressions.

Results measuring the training performance showed the advantages of JUD-EM canonical model estimation in terms of stability and convergence speed compared to the second-order JAT approach. This was found to be mainly related with the fact that fixed clean base-class parameters are used for JAT. However when VTS compensation is applied, this is no longer true, and results showed that the second-order based adaptive training schemes provided a higher training data likelihood compared to the EM related schemes.

JAT-EM and VAT-EM can be easily extended, using the FA framework, to obtain JUD and VTS discriminative adaptive training, which further improved the WER results obtained by these approaches.

Second-order approaches for JUD and VTS discriminative adaptive training were also discussed. However it was shown that these would introduce additional approximations, making the estimation of the stabilising parameters particularly complex given the nature of the weak-sense auxiliary function involved.

\(^2\)This gave better results than applying a single iteration of standard JUD batch adaptation, which gave an overall WER of 1.13\%. The reason for this is that a better initial noise estimate is used for each utterance in incremental mode, while in batch mode a noise estimate based on the first and last 20 frames is used for the initial noise parameters [3].
To assess the decoding performance of the above systems, second-order and EM-based noise estimation schemes were compared. Results showed that second-order approaches, though requiring more iterations and back-off procedures to converge, provided the best results in terms of WERs. In contrast EM-based noise estimation was faster. However, the constraints that need to be applied in this case to obtain valid update expressions limit the use of this approach to the case when the true noise parameters are not too different from those used to initialise the algorithm. It was shown that a combined second-order/EM/MUD-based approach can be used in an incremental fashion to combine the strengths of both schemes, that is, the better accuracy of the second-order approach and the speed of the EM scheme. This combined approach gave the same performance as the second-order approach alone while being twice as efficient.

Acknowledgement

This work was partly funded by Toshiba Research Europe Ltd (TREL). The authors would also like to thank TREL for providing data.

References


