Joint Uncertainty Decoding for Noise Robust Subspace Gaussian Mixture Models

Joint Uncertainty Decoding for Noise Robust Subspace Gaussian Mixture Models

Liang Lu *Student Member, IEEE*, KK Chin, Arnab Ghoshal *Member, IEEE*, and Steve Renals *Senior Member, IEEE*

***Abstract*—Joint uncertainty decoding (JUD) is a model-based noise compensation technique for conventional Gaussian Mixture Model (GMM) based speech recognition systems. Unlike vector Taylor series (VTS) compensation which operates on the individual Gaussian components in an acoustic model, JUD clusters the Gaussian components into a smaller number of classes, sharing the compensation parameters for the set of Gaussians in a given class. This significantly reduces the computational cost. In this paper, we investigate noise compensation for subspace Gaussian mixture model (SGMM) based speech recognition systems using JUD. The total number of Gaussian components in an SGMM is typically very large. Therefore direct compensation of the individual Gaussian components, as performed by VTS, is computationally expensive. In this paper we show that JUD-based noise compensation can be successfully applied to SGMMs in a computationally efficient way. We evaluate the JUD/SGMM technique on the standard Aurora 4 corpus. Our experimental results indicate that the JUD/SGMM system results in lower word error rates compared with a conventional GMM system with either VTS-based or JUD-based noise compensation.**

***Index Terms*—subspace Gaussian mixture model, vector Taylor series, joint uncertainty decoding, noise robust ASR, Aurora 4**

I. Introduction

Speech recognition accuracy is significantly degraded in the noisy environments that are characteristic of many real world applications. There is an extensive literature on methods to compensate for the mismatch between the speech recognition model and noise-corrupted data [1]. There are two broad categories of techniques for noise robust speech recognition: compensation in the feature domain, and compensation in the model domain. In the feature domain, feature enhancement or de-noising approaches aim to estimate the unobserved clean speech features given the observed noisy features. Many feature domain approaches have been proposed including spectral subtraction, cepstral mean and variance normalization (CMN/CVN), cepstral maximum mean square error estimation [2], SPLICE [3], ALGONQUIN [4] and feature space vector Taylor series (VTS) compensation [5]. Conventional feature domain methods provide a point estimate of the hidden clean speech features, which is then used as an observation vector for a speech recognition system. A number of approaches have moved beyond point estimation of clean speech features and have considered the observation uncertainties [6]–[9]. Such approaches have been shown to be more effective at improving recognition accuracy in noisy environments.

In contrast, model domain techniques adapt the model parameters in order to better explain the noisy observations. Purely data-driven model domain techniques include approaches in the maximum likelihood linear regression (MLLR) family [10], such as noisy constrained MLLR (NCMLLR) [11]. These approaches are not affected by the parameterisation of the acoustic features, since they use a generic compensation scheme—typically an affine transform—instead of an explicit model of the distortion caused by the noise. Hence, they may be combined with other feature-space compensation techniques. However, their performance is normally limited by the sparsity of adaptation data. Knowledge-based model domain approaches can overcome this limitation by estimating a mismatch function between the clean and noise-corrupted speech features in order to estimate the compensation parameters [12]. Examples of such techniques include model space VTS [13] and joint uncertainty decoding (JUD) [14], parallel model combination (PMC) [15] and a linear spline interpolation model [16]. These approaches can achieve good results without requiring a large amount of adaptation data, but are limited to only spectral or cepstral features, and combination with other feature space techniques is challenging.

In this paper, we present a model-based noise compensation scheme for subspace Gaussian mixture models (SGMMs) [17]. The SGMM is a recently proposed acoustic modeling technique that, similar to conventional models, uses a Gaussian mixture model (GMM) for the output density of a hidden Markov model (HMM). However, in an SGMM the parameters of each Gaussian component are derived from a low dimensional model subspace. This allows a much larger number of Gaussian components to be used by each HMM state while the total number of parameters to be estimated is typically smaller compared to conventional GMM-based acoustic models. Recent research has indicated that an SGMM acoustic model may result in more accurate speech recognition compared with its GMM counterpart, in both monolingual and multilingual settings [17]–[21]. However, in noisy environments uncompensated SGMMs suffer similar problems to conventional GMMs.

There are many more component Gaussians in a typical SGMM compared with a conventional GMM. Model-based compensation schemes which explicitly compensate the parameters of each component Gaussian, such as standard VTS

Manuscript received -; revised -

Liang Lu, Arnab Ghoshal and Steve Renals are with University of Edinburgh, UK; email: {liang.lu, a.ghoshal, s.renals}@ed.ac.uk

KK Chin was with Toshiba Research European Ltd, Cambridge, UK. He is now with Google Research, US. email: kk.chin@crl.toshiba.co.uk

This work was partly done during L Lu's research visit to Toshiba Research Europe Ltd, Cambridge, UK. The research was supported by EU FP7 Programme under grant agreement number 213850 (SCALE), and by EPSRC Programme Grant EP/I031022/1 (Natural Speech Technology).

compensation, will be computationally expensive if applied directly to SGMMs. Direct compensation of the individual Gaussian components in an SGMM is also inelegant, since it does not take account of the structure of the model. JUD can address this problem, since the entire set of Gaussian components in the model is clustered into a small number of classes, typically using a regression tree [22]. The mapping between a clean speech model and a noise-corrupted speech model is assumed to be common to all the Gaussians belonging to the same regression class. Moreover, JUD compensates the model using a feature space transformation (together with a bias term for the covariances), which is compatible with the compact model structure of an SGMM.

In this paper we have developed model domain noise compensation for SGMMs, and report on a number of experiments using the Aurora 4 corpus. These experiments indicate that, by using a smaller regression model, the computational cost is relatively low while the accuracy is significantly improved in noise mismatched conditions. In addition, the SGMM system is more accurate than similar GMM systems using either VTS-based or JUD-based noise compensation. We have recently presented preliminary results for an SGMM system using JUD noise compensation [23]. In this paper, we present further experiments, more detailed experimental analysis, and a derivation of JUD for SGMM acoustic models, including noise model estimation, the use of phase factors to model speech and noise correlations, and unsupervised noise model estimation to further reduce the computational cost.

The rest of the paper is organized as follows. Section II reviews the mismatch function used for many knowledge-based noise compensation approaches including JUD/SGMM. Section III and IV present an overview of JUD, and in Section V, we discuss the application of JUD to SGMMs including the configuration of the regression model, noise model estimation, and implementation details. Experiments on the Aurora 4 corpus are reported and analysed in Section VI, followed by a discussion and conclusions in Section VII.

II. Mismatch Function

In discrete time domain, the relationship between noise-corrupted speech $y(t)$, clean speech $x(t)$, additive noise $n(t)$ and the channel's impulse response $h(t)$ can be formulated as

$$y(t) = x(t) * h(t) + n(t). \tag{1}$$

where t is the time frame index. Applying the discrete Fourier transform (DFT) to both sides, the equivalent relationship in the frequency domain, for the k-th frequency bin is

$$\begin{aligned} y_{kt} &= x_{kt}h_{kt} + n_{kt} &(2)\\ &\approx x_{kt}h_k + n_{kt}. &(3)\end{aligned}$$

The channel distortion is assumed to be time invariant[1], so the subscript t may be dropped from h_{kt}. The power spectrum of the noisy speech can then be obtained as

$$\begin{aligned} |y_{kt}|^2 &\approx |x_{kt}h_k + n_{kt}|^2 \\ &= |x_{kt}|^2|h_k|^2 + |n_{kt}|^2 + 2|x_{kt}||h_k||n_{kt}|\cos\theta_{kt} \end{aligned} \tag{4}$$

[1]This is a safe assumption since the noise compensation is applied on a per-utterance basis.

where θ_{kt} denotes the (random) angle between the two complex variables $(x_{kt}h_k)$ and n_{kt}. By applying a set of Mel-scale filters, taking logarithm and multiplying by the truncated discrete cosine transform (DCT) matrix $\mathbf{C}$ on both sides, the distortion function in cepstral domain can be expressed as

$$\begin{aligned} \mathbf{y}_t &= f(\mathbf{x}_t, \mathbf{h}, \mathbf{n}_t, \boldsymbol{\alpha}_t) \\ &= \mathbf{x}_t + \mathbf{h} + \mathbf{C}\log\Big[\mathbf{1} + \exp\left(\mathbf{C}^{-1}\left(\mathbf{n}_t - \mathbf{x}_t - \mathbf{h}\right)\right) \\ &\quad + 2\boldsymbol{\alpha}_t \bullet \exp\left(\mathbf{C}^{-1}(\mathbf{n}_t - \mathbf{x}_t - \mathbf{h})/2\right)\Big], \end{aligned} \tag{5}$$

where $\mathbf{x}_t, \mathbf{y}_t, \mathbf{n}_t$, and $\mathbf{h}$ are the vector-valued clean and noise-corrupted speech, additive noise, and channel noise, respectively, at time frame t; $\mathbf{C}^{-1}$ is the pseudoinverse of the truncated DCT matrix $\mathbf{C}$ and $\mathbf{1}$ is the unit vector; $\log(\cdot)$, $\exp(\cdot)$, and $\bullet$ denote the element-wise logarithm, exponentiation, and multiplication, respectively. $\boldsymbol{\alpha}_t$ is a random variable that may be interpreted as a factor making the mismatch function sensitive to the phase between the clean speech and noise [24], [25]. The interpretation of $\boldsymbol{\alpha}_t$ as a phase factor suggests that the possible range of values for each dimension of $\boldsymbol{\alpha}_t$ is $[-1.0, 1.0]$ [24]. In many noise compensation applications, $\boldsymbol{\alpha}_t$ is often assumed to be zero. However, experimental results in [25] indicate that lower word error rates (WER) may be obtained by empirically tuning the value of $\boldsymbol{\alpha}_t$, and the lowest WER may be obtained for an $\boldsymbol{\alpha}_t$ that is outside the range suggested by the phase-sensitive theory. This is what we observe in our work as well, and a more detailed discussion of the phase factor is presented in the experimental section.

Following [25], we make a simplifying assumption that the value of $\boldsymbol{\alpha}_t$ does not depend on t, and rewrite the mismatch function as

$$\begin{aligned} \mathbf{y}_t &= f(\mathbf{x}_t, \mathbf{h}, \mathbf{n}_t, \boldsymbol{\alpha}) \\ &= \mathbf{x}_t + \mathbf{h} + \mathbf{C}\log\Big[\mathbf{1} + \exp\left(\mathbf{C}^{-1}\left(\mathbf{n}_t - \mathbf{x}_t - \mathbf{h}\right)\right) \\ &\quad + 2\boldsymbol{\alpha} \bullet \exp\left(\mathbf{C}^{-1}(\mathbf{n}_t - \mathbf{x}_t - \mathbf{h})/2\right)\Big]. \end{aligned} \tag{6}$$

Note that $\boldsymbol{\alpha} = \mathbf{0}$ corresponds to compensation in power domain, while $\boldsymbol{\alpha} = \mathbf{1}$ corresponds to magnitude domain compensation [26]. In subsequent sections, we drop the subscript t from the vectors $\mathbf{x}_t, \mathbf{y}_t$ and $\mathbf{n}_t$, in order to simplify the notation, wherever the dependence on the time index is obvious. Note that while the mismatch function is valid for Mel cepstra (static features), it is customary to append the first and second order temporal derivatives (delta and acceleration features) to obtain the complete observation vector. These dynamic coefficients are derived using a continuous-time approximation [27]. For example, the delta coefficients are given by:

$$\begin{aligned} \boldsymbol{\Delta}\mathbf{y}_t \approx \frac{\partial \mathbf{y}}{\partial t}|_t &= \frac{\partial \mathbf{y}}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial t}|_t + \frac{\partial \mathbf{y}}{\partial \mathbf{n}}\frac{\partial \mathbf{n}}{\partial t}|_t \\ &\approx \frac{\partial \mathbf{y}}{\partial \mathbf{x}}\boldsymbol{\Delta}\mathbf{x}_t + \frac{\partial \mathbf{y}}{\partial \mathbf{n}}\boldsymbol{\Delta}\mathbf{n}_t, \end{aligned} \tag{7}$$

and the acceleration coefficients, $\boldsymbol{\Delta}^2\mathbf{y}_t$, are derived similarly. By taking the expectation of these coefficients, the compensated dynamic mean and covariance parameters are obtained for model-based compensation (cf. Section IV).

In most noise compensation schemes, the additive noise is assumed to be Gaussian distributed, while the constant channel noise is represented by its "mean" for notational symmetry:

$$\mathbf{n}_t \sim \mathcal{N}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n), \qquad \mathbf{h} = \boldsymbol{\mu}_h. \tag{8}$$

The clean speech $\mathbf{x}_t$ is normally assumed to be Gaussian distributed, and the noise-corrupted speech $\mathbf{y}_t$ is still approximated by a Gaussian distribution in order to be compatible with GMM-based recognizers, although its "true" distribution may be very complex. Under this Gaussianity assumption, the aim of noise adaptation is to estimate the mean $\boldsymbol{\mu}_y$ and covariance $\boldsymbol{\Sigma}_y$ of the noise-corrupted speech. However, as the mismatch function (6) is highly nonlinear, no closed-form solution is available. A solution may be obtained either by using sampling techniques, such as data-driven parallel model combination (DPMC) [15], unscented transform (UT) (e.g. [28]–[31]), or by using a polynomial approximation such as vector Taylor series (VTS) [5]. Sampling techniques draw samples from a noise model and a clean speech model to synthesise the corresponding noisy speech samples using the mismatch function (6). They can achieve very good results with a sufficiently large number of samples, but this comes at a higher computational cost, thus limiting their applicability.

VTS approximates the nonlinear mismatch function by a truncated vector Taylor series expansion, by which a closed-form solution can be obtained for the noisy speech model. First order VTS is typically used, although recent results show that improvements can be obtained by a second or higher order VTS expansion [32], [33]. Compared to sampling, VTS compensation is relatively effective and efficient. However, since the parameters for each Gaussian component in the acoustic model are individually compensated in this approach, it is still computationally demanding, especially when the number of Gaussians is large. Joint uncertainty decoding (JUD) [9] provides a more efficient way of performing noise compensation, by clustering the Gaussian components into a relatively small number of classes, and sharing the compensation parameters among the Gaussians in each class. This significantly reduces the computational cost without a large sacrifice in accuracy.

III. Overview of Joint Uncertainty Decoding

In a standard acoustic model, the HMM observation vectors, used for computing HMM likelihoods and posterior probabilities for accumulating sufficient statistics, are obtained by appending the first- and second-order temporal derivatives (delta and acceleration features) to the static features. We denote the noise-corrupted observation by $\mathbf{o}$:

$$\mathbf{o} = \begin{bmatrix} \mathbf{y} \\ \boldsymbol{\Delta}\mathbf{y} \\ \boldsymbol{\Delta}^2\mathbf{y} \end{bmatrix}, \tag{9}$$

and the clean speech observation $\mathbf{c}$ is obtained similarly from $\mathbf{x}$ ($\mathbf{x}$ and $\mathbf{y}$ are 13-dimensional vectors, while $\mathbf{c}$ and $\mathbf{o}$ are 39-dimensional).

In the framework of joint uncertainty decoding (JUD) [9], the relationship between the observed noisy speech $\mathbf{o}$, the underlying clean speech vector $\mathbf{c}$, and Gaussian component m can be expressed as

$$p(\mathbf{o}|m) = \int p(\mathbf{c}, \mathbf{o}|m) d\mathbf{c} = \int p(\mathbf{o}|\mathbf{c}, m) p(\mathbf{c}|m) d\mathbf{c}, \tag{10}$$

where the conditional distribution $p(\mathbf{o}|\mathbf{c}, m)$ models the effect of noise on clean speech for Gaussian component m. If the dependency on m is removed from the conditional distribution, that is

$$p(\mathbf{o}|\mathbf{c}, m) \approx p(\mathbf{o}|\mathbf{c}), \tag{11}$$

then it results in a simplified uncertainty decoding rule, used for many feature domain approaches, such as SPLICE with uncertainty [7].

Although each individual Gaussian component could be compensated using (10), such an approach is not computationally feasible in practice. Instead, the Gaussians are grouped into a relatively small number of classes based on their acoustic similarities. One way of clustering the Gaussians is to use a regression tree [22], first proposed in the context of speaker adaptation. Equation (10) is approximated by replacing Gaussian component m with its regression class r_m:

$$p(\mathbf{o}|m) \approx \int p(\mathbf{o}|\mathbf{c}, r_m) p(\mathbf{c}|m) d\mathbf{c}. \tag{12}$$

The conditional distribution $p(\mathbf{o}|\mathbf{c}, r_m)$ is derived from the joint distribution of clean and noise-corrupted speech which is assumed to be Gaussian. For r^{th} regression class

$$p\left(\begin{bmatrix} \mathbf{c} \\ \mathbf{o} \end{bmatrix} \mid r\right) := \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_c^{(r)} \\ \boldsymbol{\mu}_o^{(r)} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_c^{(r)} & \boldsymbol{\Sigma}_{co}^{(r)} \\ \boldsymbol{\Sigma}_{oc}^{(r)} & \boldsymbol{\Sigma}_o^{(r)} \end{bmatrix}\right) \tag{13}$$

By marginalizing out the clean speech distribution $p(\mathbf{c}|m)$ using (12), the likelihood of corrupted speech for the m^{th} component may be expressed as a Gaussian with transformed features and an additive covariance bias, $\boldsymbol{\Sigma}_b^{(r_m)}$:

$$p(\mathbf{o}|m) \approx |\mathbf{A}^{(r_m)}| \mathcal{N}\left(\mathbf{A}^{(r_m)}\mathbf{o} + \mathbf{b}^{(r_m)}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m + \boldsymbol{\Sigma}_b^{(r_m)}\right), \tag{14}$$

where the JUD transform parameters are obtained as:

$$\mathbf{A}^{(r)} = \boldsymbol{\Sigma}_c^{(r)} {\boldsymbol{\Sigma}_{oc}^{(r)}}^{-1}, \tag{15}$$

$$\mathbf{b}^{(r)} = \boldsymbol{\mu}_c^{(r)} - \mathbf{A}^{(r)} \boldsymbol{\mu}_o^{(r)}, \tag{16}$$

$$\boldsymbol{\Sigma}_b^{(r)} = \mathbf{A}^{(r)} \boldsymbol{\Sigma}_o^{(r)} {\mathbf{A}^{(r)}}^T - \boldsymbol{\Sigma}_c^{(r)}. \tag{17}$$

The clean speech parameters, $\boldsymbol{\mu}_c^{(r)}$ and $\boldsymbol{\Sigma}_c^{(r)}$, may be derived from the clean speech model using a regression tree. The corresponding parameters for noise-corrupted speech, $\boldsymbol{\mu}_o^{(r)}$, $\boldsymbol{\Sigma}_o^{(r)}$, and the cross covariance $\boldsymbol{\Sigma}_{oc}^{(r)}$, are obtained from the mismatch function (6). In practise, only the parameters for the static cepstral coefficients, $\boldsymbol{\mu}_y^{(r)}$, $\boldsymbol{\Sigma}_y^{(r)}$, and $\boldsymbol{\Sigma}_{yx}^{(r)}$, are computed using (6), given an estimate of the noise parameters $\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n$, and $\boldsymbol{\mu}_h$. Certainly, the true noise parameters are unknown and they need to be estimated jointly with the transform parameters. Details of noise model estimation are provided in Section V-B.

The means and covariances of the dynamic coefficients are computed using the continuous time approximation (7), as

described in the following section. Cross-correlations between the static and dynamic components are assumed to be zero, which leads to a block-diagonal structure for the matrices appearing in equations (15)–(17). As an alternative to the continuous time approximation for dynamic features, the static coefficients $\mathbf{y}_t$ may be extended by appending the static coefficients of the preceding and succeeding frames, and calculating the dynamic coefficients as a linear transform of this extended vector. Although there is evidence that this approach improves upon the continuous time approximation [34], it has a much higher computational cost and hence not considered for this work.

IV. JUD Transformation Estimation

To estimate the JUD transform for regression class r we use a first-order VTS approximation [5] to linearise the mismatch function, around the expansion point $\{\boldsymbol{\mu}_x^{(r)}, \boldsymbol{\mu}_h, \boldsymbol{\mu}_n\}$ which results in:

$$\begin{aligned}\mathbf{y}|r \approx f(\boldsymbol{\mu}_x^r, \boldsymbol{\mu}_h, \boldsymbol{\mu}_n, \boldsymbol{\alpha}) + \mathbf{G}_x^{(r)}\left(\mathbf{x} - \boldsymbol{\mu}_x^{(r)}\right) \\ + \mathbf{G}_n^{(r)}\left(\mathbf{n} - \boldsymbol{\mu}_n\right),\end{aligned} \tag{18}$$

where $\mathbf{G}_x^{(r)}$ and $\mathbf{G}_n^{(r)}$ denote the Jacobian matrices

$$\mathbf{G}_x^{(r)} = \frac{\partial f(\cdot)}{\partial \mathbf{x}}\Big|_{\boldsymbol{\mu}_x^{(r)}, \boldsymbol{\mu}_h, \boldsymbol{\mu}_n}, \tag{19}$$

$$\mathbf{G}_n^{(r)} = \frac{\partial f(\cdot)}{\partial \mathbf{n}}\Big|_{\boldsymbol{\mu}_x^{(r)}, \boldsymbol{\mu}_h, \boldsymbol{\mu}_n} = \mathbf{I} - \mathbf{G}_x^{(r)}. \tag{20}$$

The mean and covariance of $\mathbf{y}$ can then be obtained by taking expectations:

$$\begin{aligned}\boldsymbol{\mu}_y^{(r)} &= \mathbb{E}\big[\mathbf{y}|r\big] \\ &= f\left(\boldsymbol{\mu}_x^{(r)}, \boldsymbol{\mu}_h, \boldsymbol{\mu}_n, \boldsymbol{\alpha}\right),\end{aligned} \tag{21}$$

$$\begin{aligned}\boldsymbol{\Sigma}_y^{(r)} &= \mathbb{E}\big[\mathbf{y}\mathbf{y}^T|r\big] - \boldsymbol{\mu}_y^{(r)}\boldsymbol{\mu}_y^{(r)T} \\ &= \mathbf{G}_x^{(r)}\boldsymbol{\Sigma}_x^{(r)}\mathbf{G}_x^{(r)T} + \mathbf{G}_n^{(r)}\boldsymbol{\Sigma}_n\mathbf{G}_n^{(r)T}.\end{aligned} \tag{22}$$

To obtain the delta parameters, we similarly take expectations on both sides of equation (7):

$$\boldsymbol{\mu}_{\Delta y}^{(r)} \approx \mathbf{G}_x^{(r)}\boldsymbol{\mu}_{\Delta x}^{(r)}, \tag{23}$$

$$\boldsymbol{\Sigma}_{\Delta y}^{(r)} \approx \mathbf{G}_x^{(r)}\boldsymbol{\Sigma}_{\Delta x}^{(r)}\mathbf{G}_x^{(r)T} + \mathbf{G}_n^{(r)}\boldsymbol{\Sigma}_{\Delta n}\mathbf{G}_n^{(r)T}. \tag{24}$$

Here we have assumed $\mathbb{E}\big[\boldsymbol{\Delta}\mathbf{n}\big] = \mathbf{0}$. This assumption was relaxed in [25], but the results showed no improvements when compensating the static or dynamic parts of the variances. Similar expressions can be obtained for the acceleration coefficients, where we assume $\mathbb{E}\big[\boldsymbol{\Delta}^2\mathbf{n}\big] = \mathbf{0}$ as well.

The cross covariance $\boldsymbol{\Sigma}_{yx}$ is calculated as:

$$\boldsymbol{\Sigma}_{yx}^{(r)} = \mathbb{E}\big[\mathbf{y}\mathbf{x}^T|r\big] - \boldsymbol{\mu}_y^{(r)}\boldsymbol{\mu}_x^{(r)T}. \tag{25}$$

By substituting the VTS approximation of $\mathbf{y}$ from equation (18) and $\boldsymbol{\mu}_y^{(r)}$ from equation (21) into (25), we obtain:

$$\boldsymbol{\Sigma}_{yx}^{(r)} \approx \mathbf{G}_x^{(r)}\boldsymbol{\Sigma}_x^{(r)}. \tag{26}$$

Again, a continuous time approximation can be used to derive the dynamic coefficients, which gives

$$\boldsymbol{\Sigma}_{\Delta y \Delta x}^{(r)} \approx \mathbf{G}_x^{(r)}\boldsymbol{\Sigma}_{\Delta x}^{(r)}, \qquad \boldsymbol{\Sigma}_{\Delta^2 y \Delta^2 x}^{(r)} \approx \mathbf{G}_x^{(r)}\boldsymbol{\Sigma}_{\Delta^2 x}^{(r)}. \tag{27}$$

Note that even if $\boldsymbol{\Sigma}_x^{(r)}$ and $\boldsymbol{\Sigma}_n$ are diagonal, $\boldsymbol{\Sigma}_y^{(r)}$ and $\boldsymbol{\Sigma}_{yx}^{(r)}$ are not, since the Jacobian matrices $\mathbf{G}_x^{(r)}$ and $\mathbf{G}_n^{(r)}$ are full. This makes the covariance bias term $\boldsymbol{\Sigma}_b^{(r)}$ block-diagonal, which is incompatible with standard HMM/GMM based speech recognizers that use diagonal covariance matrices. To obtain a final diagonal compensated covariance matrices, elements of the joint distribution are diagonalized for GMM based systems as in [35]. Diagonalising is expected to limit the compensation power of JUD. For SGMMs, however, diagonalising is not applied since the model can use full or block-diagonal covariance matrices.

V. Joint Uncertainty Decoding for SGMMs

In the SGMM acoustic model [17], the HMM state output density is modelled as:

$$p(\mathbf{o}_t|j) = \sum_{k=1}^{K_j} c_{jk} \sum_{i=1}^{I} w_{jki}\,\mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}_{jki}, \boldsymbol{\Sigma}_i), \tag{28}$$

$$\boldsymbol{\mu}_{jki} = \mathbf{M}_i\mathbf{v}_{jk}, \tag{29}$$

$$w_{jki} = \frac{\exp \mathbf{w}_i^T\mathbf{v}_{jk}}{\sum_{i'=1}^{I}\exp \mathbf{w}_{i'}^T\mathbf{v}_{jk}}, \tag{30}$$

where j is the HMM state index, k is a substate [17], I is the number of Gaussian components, and $\boldsymbol{\Sigma}_i$ is the i-th covariance matrix. $\mathbf{v}_{jk} \in \mathbb{R}^S$ is referred to as the substate vector, and S denotes the subspace dimension. The matrices $\mathbf{M}_i$ and the vectors $\mathbf{w}_i$ span the model subspaces for Gaussian means and weights, respectively, and are used to derive the GMM parameters given substate vectors (equations (29) and (30)). We refer to the Gaussian components that comprise the mixture models for each substate as the "surface Gaussians".

The SGMM parameters are initialised from a universal background model (UBM) (cf. [17], section 2), and the surface Gaussians in the region of the acoustic space corresponding to the i^{th} UBM component share the same global parameters $\mathbf{M}_i, \mathbf{w}_i$ and $\boldsymbol{\Sigma}_i$. The UBM is also used during likelihood evaluation for state j (equations (28)–(30)) to select a reduced set of the most likely regions for evaluation, rather than evaluating all $K_j \times I$ surface Gaussian components. In other words, the UBM itself provides a clustering of the surface Gaussians based on acoustic similarity. This obviates the use of a regression tree for clustering the surface Gaussians in JUD, and preserves the compact model structure of the SGMM. We estimate a JUD transform and a covariance bias for each of the I UBM components, and hence for SGMMs the JUD parameters are indexed by i instead of r.

A. Noise compensation with JUD

Since JUD noise compensation takes the form of a feature transform with an additive covariance bias, it is well suited to the SGMM framework. By contrast, VTS compensates each Gaussian individually, which is computationally infeasible (and inelegant) for an SGMM system which has a large number of surface Gaussians—for instance, in the experiments presented in this paper the models have 6.4 million surface

Gaussians. Using JUD with the UBM as the regression model, the likelihood of noise-corrupted speech becomes:

$$P(\mathbf{o}_t|j,\mathcal{M}_n) = \sum_{k=1}^{K_j} c_{jk} \sum_{i=1}^{I} w_{jki} \, |\mathbf{A}^{(i)}| \times \mathcal{N}\left(\mathbf{A}^{(i)}\mathbf{o}_t + \mathbf{b}^{(i)};\, \boldsymbol{\mu}_{jki}, \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_b^{(i)}\right) \quad (31)$$

where $\mathbf{A}^{(i)}, \mathbf{b}^{(i)}$ and $\boldsymbol{\Sigma}_b^{(i)}$ correspond to the i^{th} Gaussian in the UBM; and $\mathcal{M}_n = \{\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n, \boldsymbol{\mu}_h\}$ denotes the noise model.

Since the covariances are compensated using an additive bias, the data-independent normalisation terms in the SGMM likelihood computation (cf. [17], section 3) need to be re-computed on a per-utterance basis. This extra computation may be saved by using a predictive CMLLR method [36] that computes a set of feature transforms to minimise the Kullback-Leibler divergence between the CMLLR-adapted and JUD-compensated distributions [35]. The effect of the covariance bias terms is subsumed in the second-order statistics used for the estimation of the CMLLR transforms, thereby keeping the original covariances unchanged.

B. Noise model estimation

Noise compensation using the mismatch function (6) requires knowledge of the noise parameters $\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n$, and $\boldsymbol{\mu}_h$. Given the clean speech model, the noise parameters and the JUD transforms can be estimated alternately following the procedure outlined in Table I. The noise model may be estimated either using expectation-maximization (EM), which treats the noise parameters as latent variables [37]; or using a gradient based optimization approach [14], [25]. A comparison between the two approaches [38] showed the gradient-based approach to converge faster than EM, and to provide comparable or better recognition accuracy.

In this paper we use the gradient-based approach. The auxiliary function for noise model update is

$$\mathcal{Q}(\hat{\mathcal{M}}_n; \check{\mathcal{M}}_n) = \sum_{jkit} \gamma_{jki}(t) \Bigg[\log |\mathbf{A}^{(i)}| + \log \mathcal{N}\left(\mathbf{A}^{(i)}\mathbf{o}_t + \mathbf{b}^{(i)};\, \boldsymbol{\mu}_{jki},\, \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_b^{(i)}\right) \Bigg], \quad (32)$$

where $\hat{\mathcal{M}}_n$ and $\check{\mathcal{M}}_n$ are the 'new' and 'old' estimates of the noise model, respectively. $\gamma_{jki}(t)$ is the Gaussian component posterior, which is defined as:

$$\gamma_{jki}(t) = p\left(j,k,i|\mathbf{o}_t\right). \quad (33)$$

In [14], the derivatives of the objective function are computed numerically for JUD/GMM based noise model estimation. In this paper we derive explicitly the gradients and Hessian matrices for JUD/SGMM based noise model estimation. These derivations are similar to the VTS based noise model estimation in [14], [25], with a major difference being in the estimation of the additive noise variance $\boldsymbol{\Sigma}_n$ since we use the block-diagonal covariance matrices for SGMMs rather than diagonal covariance matrices typically used for GMMs [14], [25]. We present the overview of the estimation here with the details presented in the appendices.

TABLE I
PROCEDURE FOR JUD NOISE-COMPENSATION USING GRADIENT-BASED NOISE MODEL ESTIMATION. IN THIS PAPER, WE USED THE VITERBI ALIGNMENT FOR THE SGMM SYSTEM. STEP 3 IS REQUIRED FOR THE FIRST LOOP, BUT CAN BE SKIPPED AFTER THAT WHICH MEANS ONLY THE ALIGNMENT WILL BE UPDATED USING THE NEW NOISE MODEL.

1. Given a test utterance U, initialize the noise model $\mathcal{M}_n$.
2. Obtain the JUD transforms $\{\mathbf{A}^{(i)}, \mathbf{b}^{(i)}, \boldsymbol{\Sigma}^{(i)}\}$ using the current $\mathcal{M}_n$.
3. If required, decode U and generate the hypothesis $\mathcal{H}_u$ given the clean acoustic model $\mathcal{M}_s$, and the JUD transforms $\{\mathbf{A}^{(i)}, \mathbf{b}^{(i)}, \boldsymbol{\Sigma}^{(i)}\}$.
4. Given U, $\mathcal{M}_s$ and $\mathcal{H}_u$, accumulate the statistics λ_u by Viterbi alignment.
5. Update the noise model:
   for $i = 1$; $i \leq$ #iter1; i++
   1) Given λ_u, $\mathcal{M}_s$ and the 'old' noise model $\mathcal{M}_n$, update the noise model means $\boldsymbol{\mu}_n, \boldsymbol{\mu}_h$ (36).
   2) Compute the auxiliary function (32), and if its value decrease, back-off the noise model means (40, 41).
   for $j = 1$; $j \leq$ #iter2; j++
   3) Given λ_u, $\mathcal{M}_s$ and the 'old' noise model $\mathcal{M}_n$, update the noise model variance $\boldsymbol{\Sigma}_n$ (37);
   4) Compute the auxiliary function (32), and if its value decreases, back-off the noise model variance.
   end
   end
6. Go to step 2. if not converged.
7. Decode the utterance to obtain the final results.

1) Update the additive and channel noise mean: To update the additive and channel noise means, we first fix the VTS expansion point, and then the Jacobian matrices $\mathbf{G}_x^{(r)}$, $\mathbf{G}_n^{(r)}$ as well as the covariance bias terms $\boldsymbol{\Sigma}_b^{(r)}$ are also fixed. Taking the derivatives of $\mathcal{Q}(\cdot)$ with respect to $\hat{\boldsymbol{\mu}}_n$ and $\hat{\boldsymbol{\mu}}_h$, we obtain

$$\frac{\partial \mathcal{Q}(\cdot)}{\partial \hat{\boldsymbol{\mu}}_n} = \mathbf{d} - \mathbf{E}\hat{\boldsymbol{\mu}}_n - \mathbf{F}\hat{\boldsymbol{\mu}}_h, \quad (34)$$

$$\frac{\partial \mathcal{Q}(\cdot)}{\partial \hat{\boldsymbol{\mu}}_h} = \mathbf{u} - \mathbf{V}\hat{\boldsymbol{\mu}}_n - \mathbf{W}\hat{\boldsymbol{\mu}}_h, \quad (35)$$

where $\mathbf{d}, \mathbf{E}, \mathbf{F}$ and $\mathbf{u}, \mathbf{V}, \mathbf{W}$ are defined in equations (50–52) and (54–56) in Appendix A. By setting the two derivatives to zero, we obtain the additive and channel noise means as a solution to the following linear system:

$$\begin{bmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{V} & \mathbf{W} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\mu}}_n \\ \hat{\boldsymbol{\mu}}_h \end{bmatrix} = \begin{bmatrix} \mathbf{d} \\ \mathbf{u} \end{bmatrix}. \quad (36)$$

Here, we jointly estimate $\boldsymbol{\mu}_n$ and $\boldsymbol{\mu}_h$, which is similar to the VTS noise model estimation in [14] (Chapter 4). This approach is slightly different from that used in [25], in which $\boldsymbol{\mu}_h$ is updated first, and $\boldsymbol{\mu}_n$ is estimated using the updated $\boldsymbol{\mu}_h$. The detailed derivation can be found in Appendix A.

2) Update the additive noise variance: Unlike the additive and channel noise means, there is no closed-form solution for the additive noise variance $\boldsymbol{\Sigma}_n$. In this paper, we use Newton's algorithm to update it. If we denote σ_{nd}^2 as the d_{th} coefficient of $\boldsymbol{\Sigma}_n$, then

$$\hat{\sigma}_{nd}^2 = \sigma_{nd}^2 - \zeta \left(\frac{\partial^2 \mathcal{Q}(\cdot)}{\partial (\sigma_{nd}^2)^2}\right)^{-1} \left(\frac{\partial \mathcal{Q}(\cdot)}{\partial \sigma_{nd}^2}\right), \quad (37)$$

where ζ is the learning rate, and it was set to be 1 in this work. The gradient and Hessian are defined as:

$$\frac{\partial \mathcal{Q}(\cdot)}{\partial \sigma_{nd}^2} = -\frac{1}{2}\sum_{i=1}^{I}(\gamma_i \kappa_{id} - \beta_{id}), \tag{38}$$

$$\frac{\partial^2 \mathcal{Q}(\cdot)}{\partial (\sigma_{nd}^2)^2} = -\frac{1}{2}\sum_{i=1}^{I}(2\kappa_{id}\beta_{id} - \gamma_i \kappa_{id}^2), \tag{39}$$

where κ_{id}, β_{id} and $\mathbf{\Omega}_i$ are defined in equations (63), (66) and (65) in Appendix B, and $\gamma_i = \sum_{jkt} \gamma_{jki}(t)$. Note that in practice, the variance may be negative if (76) is applied directly. To enforce the positivity, the logarithm of variance is estimated as in [25], [39]. Details of the derivation are given in Appendix B. As observed in [38], the gradient-based optimization approach converges within a small number of iterations. For the experiments in this paper, we use `#iter = 3` and `#iter2 = 1` in the algorithm presented in Table I to update the noise model.

C. Implementation Details

Since the noise model only accounts for the static features and the Jacobian matrices are fixed during estimation, updating $\boldsymbol{\mu}_n$ and $\boldsymbol{\mu}_h$ according to equation (36) does not guarantee an increase in the auxiliary function (32). We used a simple back-off scheme [14] that interpolates between the 'old' and 'new' model parameters:

$$\hat{\boldsymbol{\mu}}_h = \eta\, \boldsymbol{\mu}_h^{old} + (1-\eta)\, \boldsymbol{\mu}_h^{new}, \tag{40}$$

$$\hat{\boldsymbol{\mu}}_n = \eta\, \boldsymbol{\mu}_n^{old} + (1-\eta)\, \boldsymbol{\mu}_n^{new}, \tag{41}$$

where $\eta \in [0, 1]$ is chosen by line search such that the auxiliary function does not decrease. A similar back-off scheme is also applied to the additive noise variance $\mathbf{\Sigma}_n$. The choice of η is conservative and makes the iterates stay at the starting value if the auxiliary function does not increase. In our experiments, we found that the back-off scheme is important for the noise model estimation (also reported in [14]). Finally, the auxiliary function (32) needs to be computed efficiently, since it is evaluated multiple times during the iterative update of the noise model. We do this by computing the sufficient statistics for each Gaussian component in the UBM over the entire utterance and caching them.

VI. Experimental Results

We performed experiments using the Aurora 4 corpus[2], which is derived from the Wall Street Journal (WSJ0) 5,000-word (5k) closed vocabulary transcription task. The clean training set contains about 15 hours of audio, and Aurora 4 provides a noisy version, which enables multi-condition training (MTR). The test set has 300 utterances from 8 speakers. The first test set, set A (`test01`), was recorded using a close talking microphone, similar to the clean training data. The data comprising set B (`test02` to `test07`) was obtained by adding six different types of noise, with randomly selected signal-to-noise ratios ranging from 5dB to 15dB, to set A. Set C (`test08`) was recording using a desk-mounted secondary microphone and the same type of noise used for set B was added to this test set to form set D (`test09` to `test14`).

In the following experiments, we used 39 dimensional feature vectors for both GMM- and SGMM-based systems: 12th order mel frequency cepstral coefficients, plus the zeroth order coefficient (C0), with delta and acceleration features. We used the standard WSJ0 5k bigram language model [40] and the CMU pronunciation dictionary[3].

TABLE II
WER of VTS and JUD based on GMM systems with $\boldsymbol{\alpha} = \mathbf{0}$.

Methods	A	B	C	D	Avg
Clean model	7.7	56.6	46.7	72.8	**59.3**
MTR model	12.7	18.6	31.7	36.8	**26.9**
VTS-init	8.7	22.4	43.0	48.0	33.9
+ 1st iter	7.1	15.8	17.3	28.6	20.8
+ 2nd iter	7.3	14.8	12.1	24.8	**18.3**
JUD-init	8.4	23.8	42.6	47.1	34.0
+1st iter	7.2	17.3	24.1	31.8	23.3
+2nd iter	7.0	16.6	16.3	28.7	**21.1**

TABLE III
WERs of noise compensation by JUD on SGMM systems with $\boldsymbol{\alpha} = \mathbf{0}$.

Methods	A	B	C	D	Avg
Clean model	5.2	58.2	50.7	72.1	**59.9**
MTR model	6.8	15.2	18.6	32.3	**22.2**
JUD-init	5.5	20.6	36.8	45.6	31.4
+1st iter	5.3	15.3	25.3	32.0	22.5
+2nd iter	5.3	14.7	20.7	28.4	**20.3**

A. Results of GMM based systems

The GMM systems were built using the HTK[4] software [41]. Table II shows the results of VTS and JUD noise compensation on a conventional GMM system, without the phase term (i.e. $\boldsymbol{\alpha} = \mathbf{0}$). Here, the clean and MTR models each have about 3,100 tied triphone states, with each speech state modelled using 16 Gaussian components and 32 Gaussian components for the silence state model. The language model scale factor and word insertion penalty were tuned on the development dataset `si_dt_05.odd` in the WSJ0 corpus and were fixed for all testing conditions. As expected, the clean model results in a high word error rate (WER) on the noisy test data, whereas the MTR model can alleviate the mismatch, resulting in significant reductions in WER, on average. For the JUD system, we used a regression model with 112 Gaussian clusters, in which 48 clusters were used for silence and the remaining 64 for speech. Two separate regression trees were used. For comparison, we carried out VTS-based noise compensation, which may be viewed as JUD when every Gaussian component corresponds to a regression class.

[2]http://aurora.hsnr.de/aurora-4.html; available from http://catalog.elra.info/

[3]Available from: http://www.speech.cs.cmu.edu/cgi-bin/cmudict

[4]http://htk.eng.cam.ac.uk

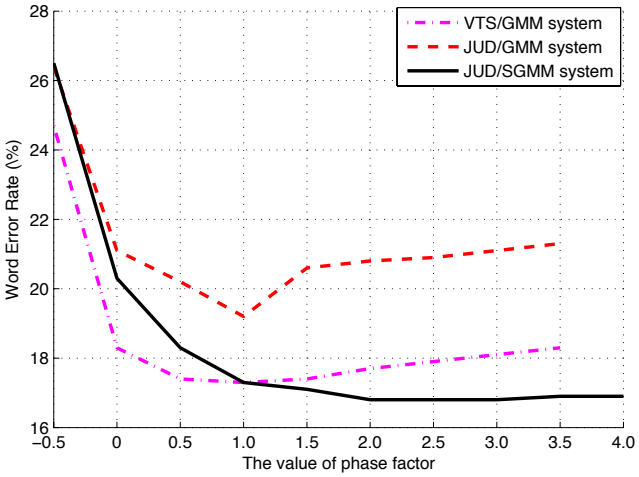

Fig. 1. Effect of phase term α for both GMM and SGMM system with VTS or JUD style noise compensation. The best result for VTS/GMM is 17.3% ($\boldsymbol{\alpha} = \mathbf{1.0}$), JUD/GMM is 19.2% ($\boldsymbol{\alpha} = \mathbf{1.0}$) and JUD/SGMM is 16.8% ($\boldsymbol{\alpha} = \mathbf{2.5}$).

TABLE IV
WERS OF VTS/GMM ($\boldsymbol{\alpha} = \mathbf{1.0}$) AND JUD/SGMM ($\boldsymbol{\alpha} = \mathbf{2.5}$) SYSTEMS.

Methods	A	B	C	D	Avg
VTS/GMM	7.0	14.8	10.6	22.7	17.3
JUD/SGMM	5.1	13.1	12.0	23.2	16.8

The noise model was initialized by the first and last 20 frames of each test utterance, corresponding to "VTS-init" and "JUD-init" in Table II. The hypotheses generated by the initial decoding were then used to update the noise model, and another decoding pass was conducted, giving results shown as "1st iter". The procedure was repeated to give the results "2nd iter". Table II indicates that updating the noise model leads to considerable gains in accuracy for both VTS and JUD. In addition, VTS-based systems consistently outperform their JUD counterparts as expected. However, the computation cost for JUD is much lower than that for VTS. The lowest WER given by VTS is 18.3% which is comparable to 17.8% reported in [42] with a similar system configuration, and that for JUD is 21.1% which is a little better than 22.2% in [43].

B. Results of SGMM based systems

The SGMM systems were built using the open-source Kaldi[5] speech recognition toolkit [44]. We used $I = 400$ components in the UBM and a subspace dimension $S = 40$. There were about 3,900 tied triphone states and a total of about 16,000 substates, resulting in 6.4 million surface Gaussians. Similar to the GMM-based systems, we separated speech and silence in the regression model, using 100 Gaussian components for silence and 300 for speech in the UBM. Table III gives the baseline results using clean and MTR models. The SGMM system has a lower WER than the GMM system on clean test data (A; 5.2% vs. 7.7%); however, the improvement disappears in noisy conditions. For the MTR model, where the mismatch is less serious, we observed that the SGMM system has a lower average WER compared with its GMM counterpart (22.2% vs. 26.9%). Just as we discussed in section I, an SGMM acoustic model can use larger number of Gaussian components and may results in more accurate speech recognition compared with its GMM counterpart due the larger modelling power. However, this is not observed in highly mismatched condition.

We then applied JUD noise compensation to a clean SGMM acoustic model. Table III shows the results without the phase term, i.e. $\boldsymbol{\alpha} = \mathbf{0}$. Again, the noise model is initialised by the first and last 20 frames of each utterance, and then updated by the algorithm described in section V-B. The results show that JUD compensation lead to lower WERs for SGMM systems in the presence of additive noise compared with the MTR model. Overall, using a three-pass decoding, we achieve 20.3% WER, which is about 2% absolute lower than that obtained using the MTR/SGMM.

We then investigated using a non-zero phase term. We did not optimize the value of $\boldsymbol{\alpha}$ (as in [24]) but set all the coefficients of $\boldsymbol{\alpha}$ to a fixed value [25]. As a comparison, we also investigated different values of the phase factor for the GMM-based VTS and JUD systems. Figure 1 graphs the average WERs. We find that the phase factor significantly affects both VTS and JUD compensation for GMM and for SGMM systems, consistent with previously reported results [24], [25]. The phase factor has a large effect on the JUD/SGMM system: tuning $\boldsymbol{\alpha}$ achieves 16.8% WER, significantly lower than the 20.3% WER with $\boldsymbol{\alpha} = \mathbf{0}$. It also outperforms the VTS/GMM system with optimal $\boldsymbol{\alpha}$ by 0.5% on average. Table IV reveals that the improvement is obtained on channel matched conditions (Set A and B), whereas for channel mismatched conditions (Set C and D), JUD/SGMM performs slightly worse than VTS/GMM system.

Possible reasons for the improvement obtained by optimizing $\boldsymbol{\alpha}$ may be the correlations between noise and speech captured by the phase factor, and the systematic bias introduced by the VTS linearisation error (equation (18)) [24], [25]. However, for SGMMs the best results are obtained by values of $\boldsymbol{\alpha}$ that are outside the range $[-1, 1]$ dictated by the phase-sensitive theory [24]. Similar results of better WER by using larger $\boldsymbol{\alpha}$ for GMM systems have also been reported in [25], in which the authors attribute the improvement to the fact that larger $\boldsymbol{\alpha}$ can compensate for the linearisation bias of VTS and perform domain combination since different values of $\boldsymbol{\alpha}$ corresponds to different feature domains. To explain the contradiction between the experimental results and the phase-sensitive theory, the authors in [26] argue that Equation (6) may be considered as a generalisation of the mismatch function, where $\boldsymbol{\alpha}$ is an additional parameter that should be optimized. We provide further analysis of the behavior of the JUD/SGMM system for different values of $\boldsymbol{\alpha}$ in the following section.

C. Analysis of the effect of phase factors

To gain further insight into the effect of phase term, we calculated the total variance of $\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_b^{(i)}$ and averaged it by I and the number of test utterances. The plot is shown in Figure

[5]http://kaldi.sf.net

TABLE V
WERS OF EACH TEST SET WITH REGARDS TO THE VALUE OF PHASE FACTOR FOR JUD/SGMM SYSTEM. "RESTAU." DENOTES RESTAURANT NOISE CONDITION.

	Set A	Set B						Set C	Set D						Average
	test01	test02	test03	test04	test05	test06	test07	test08	test09	test10	test11	test12	test13	test14	
$\boldsymbol{\alpha}$	clean	car	babble	restau.	street	airport	station	clean	car	babble	restau.	street	airport	station	
-0.5	5.2	9.0	20.9	26.3	21.9	16.6	23.1	23.3	29.1	36.4	41.6	41.1	36.2	40.1	26.5
0.0	5.3	7.6	14.6	20.1	15.8	13.4	16.6	20.7	18.5	28.0	32.6	32.4	28.4	30.7	20.3
0.5	5.3	**7.1**	13.0	18.6	**14.2**	12.4	15.5	17.8	14.1	25.6	30.7	29.2	24.9	27.9	18.3
1.0	5.3	7.2	**12.3**	17.5	14.2	11.6	**15.0**	16.2	12.7	24.0	30.0	27.1	23.6	26.2	17.3
1.5	5.3	7.1	12.4	17.5	14.5	**11.1**	15.2	14.2	12.5	23.9	28.8	26.3	23.5	26.3	17.1
2.0	5.2	7.1	12.5	**17.3**	14.4	11.2	15.3	13.1	12.1	**23.3**	28.6	**26.0**	23.2	**26.0**	16.8
2.5	5.1	7.3	12.5	17.5	14.4	11.5	15.5	12.0	**12.0**	23.4	28.2	26.1	23.1	26.2	16.8
3.0	**5.0**	7.4	12.5	17.4	14.8	12.0	15.7	10.8	12.1	24.0	**28.1**	26.0	**22.8**	26.5	16.8
3.5	5.3	7.6	12.8	17.5	14.6	11.8	15.8	10.7	12.1	24.5	28.3	26.3	23.0	26.5	16.9
4.0	5.1	7.6	13.2	17.8	14.9	11.7	16.0	**10.4**	12.2	23.8	28.3	26.4	23.2	26.7	16.9

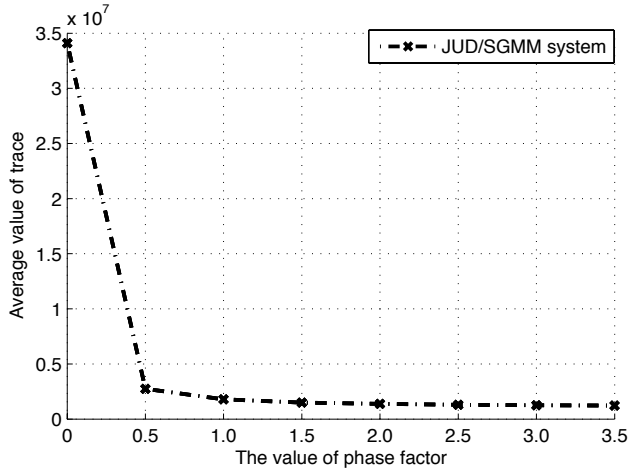

Fig. 2. Average trace of covariance matrix $\boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_b^{(i)}$ respect to the phase term $\boldsymbol{\alpha}$ for JUD/SGMM systems. $\boldsymbol{\Sigma}_b^{(i)}$ is large when $\boldsymbol{\alpha}$ is small (e.g. $\boldsymbol{\alpha} = \mathbf{0}$). The value for $\boldsymbol{\alpha} = -0.5$ is much larger, and it is not shown here for clarity.

2 for JUD/SGMM system, and shows a similar trend to that of the WER when using different values of phase factors. This is not unexpected if one interprets the value of covariance as indicating the degree of uncertainty of the model. A small covariance may indicate that the model is more confident in its explanation of the data; if this confidence is gained from more accurate model compensation, it is expected to result in lower WER. However, the absolute value in the figure is not intuitive as the features were first transformed into another feature space by the JUD transformation $(\mathbf{A}^{(i)}, \mathbf{b}^{(i)})$. As shown in Figure 2, we obtain large $\boldsymbol{\Sigma}_b^{(i)}$ when $\boldsymbol{\alpha}$ is small.

The reason may be that the phase term avoids over emphasising the noise parameters: it is clear from (6) that to explain the same mismatch between $\mathbf{x}$ and $\mathbf{y}$, the noise $\mathbf{n}$ or $\mathbf{h}$ should be larger if the phase term is removed, whereas with the positive phase term, $\mathbf{n}$ and $\mathbf{h}$ can be smaller given the same $\mathbf{x}$ and $\mathbf{y}$. The over estimation of noise will result in higher uncertainty and larger $\boldsymbol{\Sigma}_b^{(i)}$ which is in agreement with the uncertainty theory [7]. This argument also agrees with [26] that the mismatch function is generalized by using the phase term. Under this hypothesis, larger phase term may be beneficial if the mismatch between $\mathbf{x}$ and $\mathbf{y}$ is high, and this is confirmed by the results in Table V, which is explained in the next paragraph. Furthermore, note that the Jacobian matrix $\mathbf{G}_x^{(i)}$ (cf. equation (19)) for component i is a function of the phase factor $\boldsymbol{\alpha}$, and we observe that when $\boldsymbol{\alpha}$ is small, $\mathbf{G}_x^{(i)}$ has very small eigenvalues, leading to a large transformation matrix $\mathbf{A}^{(i)}$ since [14]:

$$\mathbf{A}^{(i)} \approx \begin{bmatrix} \mathbf{G}_x^{(i)-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_x^{(i)-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{G}_x^{(i)-1} \end{bmatrix} \tag{42}$$

and, consequently, $\boldsymbol{\Sigma}_b^{(i)}$ is also large (cf. equation (17)). A large value of the phase factor is able to smooth the Jacobian matrix $\mathbf{G}_x^{(i)}$, resulting in smaller transformation parameters $(\mathbf{A}^{(i)}, \mathbf{b}^{(i)}, \boldsymbol{\Sigma}_b^{(i)})$ and stabilizing the system.

To investigate the effect of phase factors in different noise conditions, we show the results of JUD/SGMM system on the 14 individual test sets in Table V. We can see that the optimal values of $\boldsymbol{\alpha}$ vary in different noise condition. Overall, the optimal $\boldsymbol{\alpha}$ is larger in highly mismatch condition (e.g. Set C and D) which confirms our claim in previous paragraph. For the clean test set A, introducing a non-zero phase term does not improve accuracy notably just as expected. For noisy set B, there is a consistent trend of lower word error rates when a positive values phase factor are used. The effect of the phase factor is most apparent for set C, which does not have added noise, but was recorded using a desk-mounted microphone. In this case, using a zero phase factor results in a high WER (20.7%, compared to 12.1% for VTS/GMM and 18.6% for MTR SGMM). Increasing the factor to a relatively large values (e.g. $\boldsymbol{\alpha} = \mathbf{4}$) reduces the WER by 50% relative to 10.4%. Set D (distant microphone with added noise) also demonstrated similar behaviour. The optimal phase factor for Set D is around $\boldsymbol{\alpha} = \mathbf{2.5}$. Further insight on the effect of phase factor $\boldsymbol{\alpha}$ may be obtained by using higher order VTS expansion [32], [33] in which the linearizaiton error will be less, or estimating the value of $\boldsymbol{\alpha}$ directly from the test data for either each utterance or each frame [24].

D. Analysis of speech and silence separation in UBM

In the regression model of the JUD/GMM system, clusters for speech and silence were defined using different regression

TABLE VI
CONFUSION MATRIX OF SPEECH AND SILENCE SEPARATION BY UBM MODEL.

	sil	speech
sil	64.8%	35.2%
speech	9.3%	90.7%

TABLE VII
COMPARISON OF UBM MODEL WITH ('YES/S') AND WITHOUT ('NO/S') SPEECH AND SILENCE SEPARATION FOR JUD/SGMM SYSTEM.

$\boldsymbol{\alpha}$	0.0	0.5	1.0	1.5	2.0	2.5	3.0	3.5
no/S	20.9	18.7	17.7	17.1	**16.8**	16.8	16.8	17.0
yes/S	20.3	18.3	17.3	17.1	**16.8**	16.8	16.8	16.9

trees. The reason for this is that speech and silence show different characteristics in the spectral domain, and their distortions by additive and channel noise are also different. This separation is expected to reduce the mismatch between the regression class model and its clusters. We also separate the speech and silence in the UBM in the JUD/SGMM system,. This was done by first identifying the speech and silence frames in the training data using a baseline system, and then building the two UBM models for speech and silence using 100 and 300 Gaussian components, respectively. They were combined to form the final UBM model, which was then used to classify the acoustic frame in the training data.

The results are shown in Table VI. We observe that by this approach, we achieve high accuracy to identify the speech frames, but not for silence (90.7% vs. 64.8%). The accuracy for noisy test data may decrease further even after noise compensation. This may undermine the gains achieved by separating speech and silence in the UBM. We compared the results of systems with and without speech and silence separation, which is shown in Table VII. Without the phase term, we achieved 0.6% gains relative by speech and silence separation in UBM, but the two system achieve the same accuracy after tuning the phase factor. Hence, separating the speech and silence in the UBM using this approach does not improve the accuracy of JUD/SGMM systems with phase factors. However, these results shed insight into the effect of phase factor with JUD/SGMM system. From Table VII, tuning $\boldsymbol{\alpha}$ achieves higher gain for the system without speech and silence separation in the UBM (from 20.9% to 16.8% vs. from 20.3% to 16.8%). This is because the approximation $p(\mathbf{o}|\mathbf{c}, r_m) \approx p(\mathbf{o}|\mathbf{c}, m)$ used in Equation (12) is poor since the effect of noise on speech and non-speech component is different. As stated before, the phase term can avoid the over estimation of noise model, and it may explain the gain here.

E. Unsupervised noise model estimation

Model-based noise compensation is normally computationally expensive, and not suitable for real time applications. For instance, in our experiments, we performed three decoding passes to obtain the final results, in which the first two passes were used to generate the hypotheses for noise model estimation. For applications with limited computational power, feature space noise compensation is normally preferred, but has lower accuracy compared to its model-based counterpart [45]. We have investigated reducing the computational cost of the JUD/SGMM by employing unsupervised noise model estimation. Instead of Equation (32), we used the UBM to update the noise model using the following auxiliary function:

$$\mathcal{Q}(\hat{\mathcal{M}}_n; \check{\mathcal{M}}_n) = \sum_{it} \gamma_i(t) \Bigg[\log |\mathbf{A}^{(i)}| + \log \mathcal{N}\left(\mathbf{A}^{(i)}\mathbf{y}_t + \mathbf{b}^{(i)}; \boldsymbol{\mu}_x^{(i)}, \boldsymbol{\Sigma}_x^{(i)} + \boldsymbol{\Sigma}_b^{(i)}\right) \Bigg], \quad (43)$$

where $\boldsymbol{\mu}_x^{(i)}$ and $\boldsymbol{\Sigma}_x^{(i)}$ are the mean and covariance the ith UBM component, and $\gamma_i(t)$ is the posterior of the ith component. In this case, the noise model can be estimated without a hypothesis. So decoding of test utterance is not required, leading to a significant reduction in computational cost. The motivation behind this is similar to the feature space VTS, in which a GMM is used to model the acoustic space, and to learn the mapping between a clean model and its noise-corrupted counterpart. However, we do not use the mapping to denoise the features, but to compensate for the noise in the model domain. This approach is also different from the front-end JUD (FE-JUD) [14], where a GMM is used to model the conditional distribution $p(\mathbf{y}|\mathbf{x})$ in equation (11) which is independent of the acoustic model. The transformation for each acoustic frame is globally shared by the all the Gaussian components in the acoustic model in FE-JUD. In contrast, our approach will have a regression class dependent transformation.

TABLE VIII
WERS (%) OF SUPERVISED ("SGMM-AUX") AND UNSUPERVISED ("UBM-AUX") AND HYBRID ("HYBRID") NOISE MODEL ESTIMATION FOR SGMM/JUD SYSTEM. "#PASS" DENOTES THE NUMBER OF DECODING PASSES.

$\boldsymbol{\alpha}$	1.0	1.5	2.0	2.5	3.0	3.5	4.0	#pass
UBM-aux	18.2	17.7	17.5	17.4	17.3	**17.2**	17.3	1
SGMM-aux	17.7	17.1	**16.8**	16.8	16.8	17.0	17.1	3
Hybrid	17.5	17.1	16.8	16.8	**16.7**	16.8	16.8	2

Table VIII shows that using unsupervised noise model estimation results in slightly increased WER compared with the supervised version (17.2% vs. 16.8%), while significantly reducing the computational cost using only single-pass decoding. In our system, the real-time factor for each decoding pass is 10.98 on average. We can also initialize the noise model in an unsupervised fashion, and then switch to supervised estimation to refine the noise model parameters. We denote such a system as "Hybrid" in Table VIII. The "Hybrid" system achieves a similar WER compared to "SGMM-aux" but with a significantly reduced computational cost.

VII. DISCUSSION AND CONCLUSION

This paper addresses robust speech recognition based on subspace Gaussian mixture models (SGMMs) using joint uncertainty decoding (JUD) noise compensation. Compared to VTS, JUD significantly reduces the computational cost by sharing the compensation parameters for Gaussian components within the same class. The major computational cost of such

TABLE IX
APPROXIMATION OF COMPUTATIONAL COST FOR VTS/GMM, JUD/GMM AND JUD/SGMM SYSTEM. M' AND R DENOTE THE TOTAL NUMBER GAUSSIANS AND REGRESSION CLASSES IN GMM SYSTEMS.

System	Model	Transform Estimation	Compensation
VTS/GMM	diag	$\mathcal{O}(M'D^3)$	$\mathcal{O}(M'D^2)$
JUD/GMM	diag	$\mathcal{O}(RD^3)$	$\mathcal{O}(RTD + M'D)$
JUD/SGMM	blck	$\mathcal{O}(ID^3)$	$\mathcal{O}(ITD^2 + ID^2)$

approaches is because of the multiple decoding passes that are required to estimate the noise model parameters and compensation parameters.

JUD is an attractive noise compensation approach for SGMMs, since the adaptation takes advantage of the SGMM structure, rather than adapting the surface Gaussians explicitly. For each decoding pass, the surface Gaussians are adapted using I sets of feature transforms and covariance biases, corresponding to the UBM components. For each test utterance with T frames, each frame is transformed by the I transformations with a computational cost of $\mathcal{O}(ITD^2)$. In this paper, $I = 400$, $D = 39$ and $100 \leq T \leq 1000$. Updating the variance is $\mathcal{O}(ID^2)$, with some additional computation required to update the normalization term [17]. The number of regression classes for the JUD/SGMM system does not need to be equal to the number of UBM components. A smaller number of regression classes may be used by clustering the UBM Gaussians and sharing the JUD transforms within each clustered set of UBM components.

However, the total computational cost is still significantly lower than direct VTS compensation of SGMMs, which would require $\mathcal{O}(MD^3)$ for the block-diagonal covariance matrices used in this paper, where M is the total number of surface Gaussians (6.4 million in this paper). As previously mentioned, further computational savings may be achieved by using by predictive CMLLR [36] to remove the covariance bias terms $\mathbf{\Sigma}_b^{(i)}$, so that the normalization terms of SGMMs can be left untouched.

In Table IX, we compare the computational cost in terms of transformation estimation and compensation for VTS/GMM, JUD/GMM and JUD/SGMM systems. For transform estimation, the main computational cost is to estimate the Jacobian matrices (e.g. equation (19)), which is linear in the number of Gaussians for VTS/GMM, while it is linear in the number of regression classes for JUD/GMM and JUD/SGMM. In this case, the cost of estimating transformations for JUD/SGMM system is lower than that of VTS/GMM system. For compensation, the computational cost lies in compensating the covariance (e.g. equation (22)) for VTS/GMM system, and as we used diagonal covariances, the cost was reduced to $\mathcal{O}(M'D^2)$. In this paper, the number of Gaussians components used in the VTS/GMM system is about 3,000, hence $M' < IT$. Thus, in this case, the overall computational cost of JUD/SGMM is greater than that of VTS/GMM. However, the gap will shrink when using more complex systems with a greater number of Gaussians, or using a smaller number of regression classes for the SGMM system.

To further reduce the computational cost of JUD/SGMM, we also investigated the unsupervised noise model estimation using UBM which removes the need for multiple decoding passes at the cost of a slightly increased WER, or which can be used to initialize the noise model, thus reducing the number of decoding passes for the supervised case.

To summarize our experimental results, by empirically tuning the phase factor, we achieved 16.8% WER for JUD/SGMM system on the Aurora 4 corpus, which is comparable to the state-of-the-art noise compensation results on this task [42], [46]. Further improvements have been observed by VTS-based noise adaptive training [39], joint speaker/noise compensation [42], and discriminative adaptive training [46]. The subspace GMM framework has considerable potential to explore adaptive approaches related to these, and in particular provides a highly promising structure to explicitly factorise causes of variability such as additive noise, channel noise, and speaker effects.

APPENDIX A
UPDATE THE ADDITIVE AND CHANNEL NOISE MEAN

The following derivations are for the static features; the delta and acceleration coefficients may be obtained using a continuous time approximation, as discussed in Section II, in which the static and dynamic coefficients are assumed to be independent. Likewise, the JUD transforms $(\mathbf{A}^{(i)}, \mathbf{b}^{(i)}, \mathbf{\Sigma}_b^{(i)})$ in these derivations correspond to the static coefficients only.

We denote the clean and noisy UBM models as $\{\boldsymbol{\mu}_x^{(i)}, \mathbf{\Sigma}_x^{(i)}; i = 1, \ldots, I\}$, and $\{\boldsymbol{\mu}_y^{(i)}, \mathbf{\Sigma}_y^{(i)}; i = 1, \ldots, I\}$, respectively. As stated before, the derivations here are similar to the VTS noise model estimation [14] (Chapter 4), but with a different accumulation of statistics for the SGMM. We use a similar notations to [14] in order to make clear the relations and difference between the two. We first rewrite the auxiliary function (32) for the static coefficients as

$$\begin{aligned}\mathcal{Q}(\cdot) &= \sum_{jkit} \gamma_{jki}(t) \\ &\quad \times \left[\log \mathcal{N}\left(\mathbf{y}_t;\ \mathbf{A}^{(i)-1}\left(\boldsymbol{\mu}_{jki} - \boldsymbol{\mu}_x^{(i)}\right) + \boldsymbol{\mu}_y^{(i)}, \tilde{\mathbf{\Sigma}}_y^{(i)}\right)\right] \\ &= -\frac{1}{2}\sum_{jkit} \gamma_{jki}(t)\left(\log|\tilde{\mathbf{\Sigma}}_y^{(i)}| + \tilde{\mathbf{y}}_{jkit}^T \tilde{\mathbf{\Sigma}}_y^{(i)-1} \tilde{\mathbf{y}}_{jkit}\right) \quad (44)\end{aligned}$$

where

$$\tilde{\mathbf{y}}_{jkit} = \left(\mathbf{y}_t - \mathbf{A}^{(i)-1}\left(\boldsymbol{\mu}_{jki} - \boldsymbol{\mu}_x^{(i)}\right) - \boldsymbol{\mu}_y^{(i)}\right), \quad (45)$$

$$\begin{aligned}\tilde{\mathbf{\Sigma}}_y^{(i)} &= \mathbf{A}^{(i)-1}\left(\mathbf{\Sigma}_i + \mathbf{\Sigma}_b^{(i)}\right)\mathbf{A}^{(i)-T} \\ &= \mathbf{\Sigma}_y^{(i)} + \mathbf{A}^{(i)-1}\left(\mathbf{\Sigma}_i - \mathbf{\Sigma}_x^{(i)}\right)\mathbf{A}^{(i)-T}. \quad (46)\end{aligned}$$

To update the noise model, we first fix the VTS expansion point, so that the Jacobian matrices are also fixed, and $\boldsymbol{\mu}_y^{(i)}$ is a function of the additive and channel noise means, $\boldsymbol{\mu}_n$ and $\boldsymbol{\mu}_h$, only. Using the first order VTS expansion around the old noise model parameters $(\check{\boldsymbol{\mu}}_n, \check{\boldsymbol{\mu}}_h)$, $\boldsymbol{\mu}_y^{(i)}$ can be expressed as

$$\begin{aligned}\boldsymbol{\mu}_y^{(i)} &\approx \mathbb{E}\Big\{f\left(\boldsymbol{\mu}_x^{(i)}, \check{\boldsymbol{\mu}}_n, \check{\boldsymbol{\mu}}_h, \boldsymbol{\alpha}\right) + \mathbf{G}_x^{(i)}\left(\mathbf{x} - \boldsymbol{\mu}_x^{(i)}\right) \\ &\quad + \mathbf{G}_n^{(i)}\left(\mathbf{n} - \check{\boldsymbol{\mu}}_n\right) + \mathbf{G}_h^{(i)}\left(\mathbf{h} - \check{\boldsymbol{\mu}}_h\right)\Big\} \\ &= \check{\boldsymbol{\mu}}_y^{(i)} + \mathbf{G}_n^{(i)}\left(\hat{\boldsymbol{\mu}}_n - \check{\boldsymbol{\mu}}_n\right) + \mathbf{G}_h^{(i)}\left(\hat{\boldsymbol{\mu}}_h - \check{\boldsymbol{\mu}}_h\right), \quad (47)\end{aligned}$$

where $\mathbf{G}_x^{(i)}$ and $\mathbf{G}_n^{(i)}$ are the Jacobian matrices (19) and (20). $\mathbf{G}_h^{(i)}$ is defined as

$$\mathbf{G}_h^{(i)} = \frac{\partial f(\cdot)}{\partial \check{\boldsymbol{\mu}}_h}|_{\boldsymbol{\mu}_x^{(i)}, \check{\boldsymbol{\mu}}_h, \check{\boldsymbol{\mu}}_n}. \tag{48}$$

Taking the derivative of $\mathcal{Q}(\cdot)$ w.r.t. $\hat{\boldsymbol{\mu}}_n$, we obtain

$$\begin{aligned}\frac{\partial \mathcal{Q}(\cdot)}{\partial \hat{\boldsymbol{\mu}}_n} &= \sum_{jkit} \gamma_{jki}(t) \mathbf{G}_n^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \tilde{\mathbf{y}}_{jkit} \\ &= \sum_{jkit} \gamma_{jki}(t) \mathbf{G}_n^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \Big(\mathbf{y}_t - \mathbf{A}^{(i)-1} \left(\boldsymbol{\mu}_{jki} - \boldsymbol{\mu}_x^{(i)} \right) \\ &\quad - \check{\boldsymbol{\mu}}_y^{(i)} - \mathbf{G}_n^{(i)} \left(\hat{\boldsymbol{\mu}}_n - \check{\boldsymbol{\mu}}_n \right) - \mathbf{G}_h^{(i)} \left(\hat{\boldsymbol{\mu}}_h - \check{\boldsymbol{\mu}}_h \right) \Big) \\ &= \mathbf{d} - \mathbf{E}\hat{\boldsymbol{\mu}}_n - \mathbf{F}\hat{\boldsymbol{\mu}}_h, \end{aligned} \tag{49}$$

where

$$\begin{aligned}\mathbf{d} = \sum_{jkit} \gamma_{jki}(t) \mathbf{G}_n^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \Big(& \mathbf{y}_t - \mathbf{A}^{(i)-1} \left(\boldsymbol{\mu}_{jki} - \boldsymbol{\mu}_x^{(i)} \right) \\ & - \check{\boldsymbol{\mu}}_y^{(i)} + \mathbf{G}_n^{(i)} \check{\boldsymbol{\mu}}_n + \mathbf{G}_h^{(i)} \check{\boldsymbol{\mu}}_h \Big), \end{aligned} \tag{50}$$

$$\mathbf{E} = \sum_i \gamma_i \mathbf{G}_n^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \mathbf{G}_n^{(i)}, \tag{51}$$

$$\mathbf{F} = \sum_i \gamma_i \mathbf{G}_n^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \mathbf{G}_h^{(i)}, \tag{52}$$

and $\gamma_i = \sum_{jkt} \gamma_{jki}(t)$. Similarly, taking the derivative of $\mathcal{Q}(\cdot)$ w.r.t. $\hat{\boldsymbol{\mu}}_h$ gives

$$\begin{aligned}\frac{\partial \mathcal{Q}(\cdot)}{\partial \hat{\boldsymbol{\mu}}_h} &= \sum_{jkit} \gamma_{jki}(t) \mathbf{G}_h^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \tilde{\mathbf{y}}_{jkit} \\ &= \mathbf{u} - \mathbf{V}\hat{\boldsymbol{\mu}}_n - \mathbf{W}\hat{\boldsymbol{\mu}}_h, \end{aligned} \tag{53}$$

where

$$\begin{aligned}\mathbf{u} = \sum_{jkit} \gamma_{jki}(t) \mathbf{G}_h^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \Big(& \mathbf{y}_t - \mathbf{A}^{(i)-1} \left(\boldsymbol{\mu}_{jki} - \boldsymbol{\mu}_x^{(i)} \right) \\ & - \check{\boldsymbol{\mu}}_y^{(i)} + \mathbf{G}_n^{(i)} \check{\boldsymbol{\mu}}_n - \mathbf{G}_h^{(i)} \check{\boldsymbol{\mu}}_h \Big), \end{aligned} \tag{54}$$

$$\mathbf{V} = \sum_i \gamma_i \mathbf{G}_h^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \mathbf{G}_n^{(i)}, \tag{55}$$

$$\mathbf{W} = \sum_i \gamma_i \mathbf{G}_h^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \mathbf{G}_h^{(i)}. \tag{56}$$

Setting the two derivatives to be zero we obtain

$$\begin{bmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{V} & \mathbf{W} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\mu}}_n \\ \hat{\boldsymbol{\mu}}_h \end{bmatrix} = \begin{bmatrix} \mathbf{d} \\ \mathbf{u} \end{bmatrix}, \tag{57}$$

which gives

$$\begin{bmatrix} \hat{\boldsymbol{\mu}}_n \\ \hat{\boldsymbol{\mu}}_h \end{bmatrix} = \begin{bmatrix} \mathbf{E} & \mathbf{F} \\ \mathbf{V} & \mathbf{W} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{d} \\ \mathbf{u} \end{bmatrix}. \tag{58}$$

In our implementation, we cached the statistics that are independent of the noise parameter so that they were not computed repeatedly. For instance, $\mathbf{d}$ can be decomposed as follows:

$$\begin{aligned}\mathbf{d} = & \sum_i \gamma_i \mathbf{G}_n^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \left(\mathbf{G}_n^{(i)} \check{\boldsymbol{\mu}}_n + \mathbf{G}_h^{(i)} \check{\boldsymbol{\mu}}_h - \check{\boldsymbol{\mu}}_y^{(i)} \right) \\ & + \sum_i \mathbf{G}_n^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \underbrace{\sum_{jkt} \gamma_{jki}(t) \mathbf{y}_t}_{\text{cached}} \\ & - \sum_i \mathbf{G}_n^{(i)T} \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \mathbf{A}^{(i)-1} \underbrace{\sum_{jkt} \gamma_{jki}(t) \left(\boldsymbol{\mu}_{jki} - \boldsymbol{\mu}_x^{(i)} \right)}_{\text{cached}} \end{aligned}$$

Caching was also used for the computation of $\mathbf{u}$ in (54).

Appendix B
Update the additive noise variance

The derivation here is similar to the estimation of the additive noise variance $\boldsymbol{\Sigma}_n$ for VTS [14] (App. C). To update $\boldsymbol{\Sigma}_n$, we first fix the value of $\boldsymbol{\mu}_n$ and $\boldsymbol{\mu}_h$. Again, the derivations are for static features only. For the dynamic coefficients of $\boldsymbol{\Sigma}_n$, the derivations are similar. We rewrite the auxiliary function (44):

$$\mathcal{Q}(\cdot) = -\frac{1}{2} \sum_{jkit} \gamma_{jki}(t) \left(\log |\tilde{\boldsymbol{\Sigma}}_y^{(i)}| + \tilde{\mathbf{y}}_{jkit}^T \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \tilde{\mathbf{y}}_{jkit} \right) \tag{59}$$

where $\tilde{\mathbf{y}}_{jkit}$ and $\tilde{\boldsymbol{\Sigma}}_y^{(i)}$ are defined in (45) and (46). Note that that $\tilde{\boldsymbol{\Sigma}}_y^{(i)}$ is full rather than diagonal (unlike [14]). Therefore, the derivations are slightly different. Since $\tilde{\mathbf{y}}_{jkit}$ does not depend on $\boldsymbol{\Sigma}_n$, by taking derivative $\mathcal{Q}(\cdot)$ w.r.t. to the d^{th} diagonal element of $\boldsymbol{\Sigma}_n$, we obtain:

$$\frac{\partial \mathcal{Q}(\cdot)}{\partial \sigma_{nd}^2} = -\frac{1}{2} \sum_{jkit} \gamma_{jki}(t) \left[\underbrace{\frac{\partial \log |\tilde{\boldsymbol{\Sigma}}_y^{(i)}|}{\partial \sigma_{nd}^2}}_{\text{first part}} + \underbrace{\tilde{\mathbf{y}}_{jkit}^T \frac{\partial \tilde{\boldsymbol{\Sigma}}_y^{(i)-1}}{\partial \sigma_{nd}^2} \tilde{\mathbf{y}}_{jkit}}_{\text{second part}} \right] \tag{60}$$

The first part of the derivative is

$$\frac{\partial \log |\tilde{\boldsymbol{\Sigma}}_y^{(i)}|}{\partial \sigma_{nd}^2} = \frac{1}{|\tilde{\boldsymbol{\Sigma}}_y^{(i)}|} \frac{\partial |\tilde{\boldsymbol{\Sigma}}_y^{(i)}|}{\partial \sigma_{nd}^2} = \text{Tr} \left(\tilde{\boldsymbol{\Sigma}}_y^{(i)-1} \frac{\partial \tilde{\boldsymbol{\Sigma}}_y^{(i)}}{\partial \sigma_{nd}^2} \right). \tag{61}$$

It can be seen from (46) that only $\boldsymbol{\Sigma}_y^{(i)}$ depends on σ_{nd}^2, and from (22)

$$\boldsymbol{\Sigma}_y^{(i)} = \mathbf{G}_x^{(i)} \boldsymbol{\Sigma}_x^{(i)} \mathbf{G}_x^{(i)T} + \mathbf{G}_n^{(i)} \boldsymbol{\Sigma}_n \mathbf{G}_n^{(i)T}.$$

Hence

$$\frac{\partial \tilde{\boldsymbol{\Sigma}}_y^{(i)}}{\partial \sigma_{nd}^2} = \frac{\partial \boldsymbol{\Sigma}_y^{(i)}}{\partial \sigma_{nd}^2} = [\mathbf{G}_n^{(i)}]_d [\mathbf{G}_n^{(i)}]_d^T, \tag{62}$$

where $[\mathbf{G}_n^{(i)}]_d$ denotes the d^{th} column of $\mathbf{G}_n^{(i)}$. Substituting (62) into (61), we obtain the first part of the derivative as

$$\begin{aligned}\kappa_{id} \equiv \frac{\partial \log |\tilde{\boldsymbol{\Sigma}}_y^{(i)}|}{\partial \sigma_{nd}^2} &= \text{Tr} \left(\tilde{\boldsymbol{\Sigma}}_y^{(i)-1} [\mathbf{G}_n^{(i)}]_d [\mathbf{G}_n^{(i)}]_d^T \right) \\ &= [\mathbf{G}_n^{(i)}]_d^T \tilde{\boldsymbol{\Sigma}}_y^{(i)-1} [\mathbf{G}_n^{(i)}]_d \end{aligned} \tag{63}$$

Similarly, for the second part of the derivative

$$\frac{\partial \tilde{\mathbf{\Sigma}}_y^{(i)-1}}{\partial \sigma_{nd}^2} = -\tilde{\mathbf{\Sigma}}_y^{(i)-1}\frac{\partial \tilde{\mathbf{\Sigma}}_y^{(i)}}{\partial \sigma_{nd}^2}\tilde{\mathbf{\Sigma}}_y^{(i)-1} = -\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1} \quad (64)$$

Hence, we can compute it as

$$\begin{aligned}
&\sum_{jkit}\gamma_{jki}(t)\tilde{\mathbf{y}}_{jkit}^T\frac{\partial \tilde{\mathbf{\Sigma}}_y^{(i)-1}}{\partial \sigma_{nd}^2}\tilde{\mathbf{y}}_{jkit}\\
&= -\sum_{jkit}\gamma_{jki}(t)\tilde{\mathbf{y}}_{jkit}^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}\tilde{\mathbf{y}}_{jkit}\\
&= -\sum_{jkit}\gamma_{jki}(t)[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}\tilde{\mathbf{y}}_{jkit}\tilde{\mathbf{y}}_{jkit}^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d\\
&= -\sum_i[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}\mathbf{\Omega}_i\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^i]_d,
\end{aligned}$$

where we have accumulated all the statistics indexed by j, k, t as

$$\mathbf{\Omega}_i = \sum_{jkt}\gamma_{jki}(t)\tilde{\mathbf{y}}_{jkit}\tilde{\mathbf{y}}_{jkit}^T. \quad (65)$$

Again, we decompose $\mathbf{\Omega}_i$ and cache the statistics that do not depend on the noise parameters in order to save the computation, but we omit the details here for brevity. If we denote

$$\beta_{id} \equiv [\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}\mathbf{\Omega}_i\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^i]_d, \quad (66)$$

then the gradient can be expressed as

$$\frac{\partial \mathcal{Q}(\cdot)}{\partial \sigma_{nd}^2} = -\frac{1}{2}\sum_{i=1}^{I}(\gamma_i\kappa_{id} - \beta_{id}). \quad (67)$$

As stated before, to enforce positivity, the logarithm of the variance is estimated [25]:

$$\tilde{\sigma}_{nd}^2 = \log(\sigma_{nd}^2), \quad (68)$$
$$\sigma_{nd}^2 = \exp(\tilde{\sigma}_{nd}^2). \quad (69)$$

Then we can obtain

$$\begin{aligned}
\frac{\partial \mathcal{Q}(\cdot)}{\partial \tilde{\sigma}_{nd}^2} &= \frac{\partial \mathcal{Q}(\cdot)}{\partial \sigma_{nd}^2}\frac{\partial \sigma_{nd}^2}{\partial \tilde{\sigma}_{nd}^2} = \frac{\partial \mathcal{Q}(\cdot)}{\partial \sigma_{nd}^2}\sigma_{nd}^2\\
&= -\frac{1}{2}\sum_{i=1}^{I}(\gamma_i\kappa_{id} - \beta_{id})\sigma_{nd}^2. \quad (70)
\end{aligned}$$

Next, we calculate the Hessian matrix of $\mathcal{Q}(\cdot)$ w.r.t. the noise variance $\partial^2\mathcal{Q}(\cdot)/\partial(\sigma_{nd}^2)^2$. From the gradient (67) we can obtain:

$$\begin{aligned}
\frac{\partial^2 \mathcal{Q}(\cdot)}{\partial(\sigma_{nd}^2)^2} &= \frac{\partial}{\partial \sigma_{nd}^2}\left(\frac{\partial \mathcal{Q}(\cdot)}{\partial \sigma_{nd}^2}\right)\\
&= \frac{\partial}{\partial \sigma_{nd}^2}\left(-\frac{1}{2}\sum_{i=1}^{I}(\gamma_i\kappa_{id} - \beta_{id})\right)\\
&= -\frac{1}{2}\sum_{i=1}^{I}(\gamma_i\frac{\partial \kappa_{id}}{\partial \sigma_{nd}^2} - \frac{\partial \beta_{id}}{\partial \sigma_{nd}^2}). \quad (71)
\end{aligned}$$

From (63) and (64), we can obtain the first part of the derivative as:

$$\begin{aligned}
\frac{\partial \kappa_{id}}{\partial \sigma_{nd}^2} &= [\mathbf{G}_n^{(i)}]_d^T\frac{\partial \tilde{\mathbf{\Sigma}}_y^{(i)-1}}{\partial \sigma_{nd}^2}[\mathbf{G}_n^{(i)}]_d\\
&= -\underbrace{[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d}_{\kappa_{id}}\underbrace{[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d}_{\kappa_{id}}\\
&= -\kappa_{id}^2. \quad (72)
\end{aligned}$$

Using (66) and (64), we can compute the second part of the derivative $\partial\beta_{id}/\partial\sigma_{nd}^2$:

$$\begin{aligned}
\frac{\partial \beta_{id}}{\partial \sigma_{nd}^2} &= \frac{\partial\left([\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}\mathbf{\Omega}_i\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d\right)}{\partial \sigma_{nd}^2}\\
&= [\mathbf{G}_n^{(i)}]_d^T\frac{\partial \tilde{\mathbf{\Sigma}}_y^{(i)-1}}{\partial \sigma_{nd}^2}\mathbf{\Omega}_i\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d\\
&\quad + [\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}\mathbf{\Omega}_i\frac{\partial \tilde{\mathbf{\Sigma}}_y^{(i)-1}}{\partial \sigma_{nd}^2}[\mathbf{G}_n^{(i)}]_d\\
&= -\underbrace{[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d}_{\kappa_{id}}\underbrace{[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}\mathbf{\Omega}_i\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d}_{\beta_{id}}\\
&\quad - \underbrace{[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}\mathbf{\Omega}_i\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d}_{\beta_{id}}\underbrace{[\mathbf{G}_n^{(i)}]_d^T\tilde{\mathbf{\Sigma}}_y^{(i)-1}[\mathbf{G}_n^{(i)}]_d}_{\kappa_{id}}\\
&= -2\kappa_{id}\beta_{id}. \quad (73)
\end{aligned}$$

By summing the two parts we obtain

$$\frac{\partial^2 \mathcal{Q}(\cdot)}{\partial(\sigma_{nd}^2)^2} = -\frac{1}{2}\sum_{i=1}^{I}(2\kappa_{id}\beta_{id} - \gamma_i\kappa_{id}^2), \quad (74)$$

and the Hessian of the logarithm of the variance can be estimated as

$$\begin{aligned}
\frac{\partial^2 \mathcal{Q}(\cdot)}{\partial(\tilde{\sigma}_{nd}^2)^2} &= \frac{\partial}{\partial \tilde{\sigma}_{nd}^2}\left(\frac{\partial \mathcal{Q}(\cdot)}{\partial \tilde{\sigma}_{nd}^2}\right)\\
&= \frac{\partial \mathcal{Q}(\cdot)}{\partial \sigma_{nd}^2}\sigma_{nd}^2 + \frac{\partial^2 \mathcal{Q}(\cdot)}{\partial(\sigma_{nd}^2)^2}\sigma_{nd}^2\sigma_{nd}^2\\
&= -\frac{1}{2}\sum_{i=1}^{I}(\gamma_i\kappa_{id} - \beta_{id})\sigma_{nd}^2\\
&\quad - \frac{1}{2}\sum_{i=1}^{I}(2\kappa_{id}\beta_{id} - \gamma_i\kappa_{id}^2)\sigma_{nd}^2\sigma_{nd}^2. \quad (75)
\end{aligned}$$

After obtaining the gradient (70) and Hessian (75) of the logarithm of the variance $\tilde{\sigma}_{nd}^2$ we can update the estimate similar to (76) as

$$\hat{\tilde{\sigma}}_{nd}^2 = \tilde{\sigma}_{nd}^2 - \zeta\left(\frac{\partial^2 \mathcal{Q}(\cdot)}{\partial(\tilde{\sigma}_{nd}^2)^2}\right)^{-1}\left(\frac{\partial \mathcal{Q}(\cdot)}{\partial \tilde{\sigma}_{nd}^2}\right). \quad (76)$$

Then we use equation (69) to compute the original variance.

REFERENCES

[1] J Droppo and A Acero, "Environmental robustness," in *Handbook of Speech Processing*, J Benesty, MM Sondhi, and Y Huang, Eds., chapter 33, pp. 653–679. Springer, 2008.

[2] D Yu, L Deng, J Droppo, J Wu, Y Gong, and A Acero, "Robust speech recognition using a cepstral minimum-mean-square-error-motivated noise suppressor," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 5, pp. 1061–1070, 2008.

[3] L Deng, A Acero, M Plumpe, and X Huang, "Large-vocabulary speech recognition under adverse acoustic environments," in *Proc. ICSLP*, 2000.

[4] BJ Frey, L Deng, A Acero, and T Kristjansson, "ALGONQUIN: Iterating Laplace's method to remove multiple types of acoustic distortion for robust speech recognition," in *Seventh European Conference on Speech Communication and Technology*, 2001.

[5] PJ Moreno, B Raj, and RM Stern, "A vector Taylor series approach for environment-independent speech recognition," in *Proc. ICASSP*. IEEE, 1996, vol. 2, pp. 733–736.

[6] JA Arrowood and MA Clements, "Using observation uncertainty in HMM decoding," in *Proc. ICSLP*, 2002.

[7] J Droppo, A Acero, and L Deng, "Uncertainty decoding with SPLICE for noise robust speech recognition," in *Proc. ICASSP*. IEEE, 2002.

[8] L Deng, J Droppo, and A Acero, "Dynamic compensation of HMM variances using the feature enhancement uncertainty computed from a parametric model of speech distortion," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 412–421, 2005.

[9] H Liao and MJF Gales, "Joint uncertainty decoding for noise robust speech recognition," in *Proc. INTERSPEECH*. Citeseer, 2005.

[10] PC Woodland, MJF Gales, and D Pye, "Improving environmental robustness in large vocabulary speech recognition," in *Proc. ICASSP*. IEEE, 1996, pp. 65–68.

[11] DK Kim and MJF Gales, "Noisy constrained maximum-likelihood linear regression for noise-robust speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 2, pp. 315–325, 2011.

[12] A Acero, *Acoustic and Enviromental Robustness in Automatic Speech Recognition*, Ph.D. thesis, Carnegie Mellon University, 1990.

[13] A Acero, L Deng, T Kristjansson, and J Zhang, "HMM adaptation using vector Taylor series for noisy speech recognition," in *Proc. ICSLP*, 2000.

[14] H Liao, *Uncertainty decoding for noise robust speech recognition*, Ph.D. thesis, University of Cambridge, 2007.

[15] MJF Gales, *Model-based techniques for noise robust speech recognition*, Ph.D. thesis, Cambridge University, 1995.

[16] K Kalgaonkar, ML Seltzer, and A Acero, "Noise robust model adaptation using linear spline interpolation," in *Proc. ASRU*, 2009, pp. 199–204.

[17] D Povey, L Burget, M Agarwal, P Akyazi, F Kai, A Ghoshal, O Glembek, N Goel, M Karafiát, A Rastrow, RC Rose, P Schwarz, and S Thomas, "The subspace Gaussian mixture model—A structured model for speech recognition," *Computer Speech & Language*, vol. 25, no. 2, pp. 404–439, 2011.

[18] D Povey, M Karafiát, A Ghoshal, and P Schwarz, "A symmetrization of the subspace Gaussian mixture model," in *Proc. ICASSP*. IEEE, 2011, pp. 4504–4507.

[19] L Lu, A Ghoshal, and S Renals, "Regularized subspace Gaussian mixture models for speech recognition," *IEEE Signal Processing Letters*, vol. 18, no. 7, pp. 419–422, 2011.

[20] L Burget, P Schwarz, M Agarwal, P Akyazi, K Feng, A Ghoshal, O Glembek, N Goel, M Karafiát, D Povey, A Rastrow, R Rose, and S Thomas, "Multilingual acoustic modeling for speech recognition based on subspace Gaussian mixture models," in *Proc. IEEE ICASSP*, 2010, pp. 4334–4337.

[21] L Lu, A Ghoshal, and S Renals, "Regularized subspace Gaussian mixture models for cross-lingual speech recognition," in *Proc. IEEE ASRU*, 2011.

[22] MJF Gales, "The generation and use of regression class trees for MLLR adaptation," Technical Report CUED/F-INFENG/TR.263, Cambridge University Engineering Department, August 1996.

[23] L Lu, KK Chin, A Ghoshal, and S Renals, "Noise compensation for subspace Gaussian mixture models," in *Proc. INTERSPEECH*, 2012, (to appear).

[24] L Deng, J Droppo, and A Acero, "Enhancement of log mel power spectra of speech using a phase-sensitive model of the acoustic environment and sequential estimation of the corrupting noise," *IEEE Transactions on Speech and Audio Processing*, vol. 12, no. 2, pp. 133–143, 2004.

[25] J Li, L Deng, D Yu, Y Gong, and A Acero, "A unified framework of HMM adaptation with joint compensation of additive and convolutive distortions," *Computer Speech & Language*, vol. 23, no. 3, pp. 389–405, 2009.

[26] MJF Gales and F Flego, "Discriminative classifiers with adaptive kernels for noise robust speech recognition," *Computer Speech & Language*, vol. 24, no. 4, pp. 648–662, 2010.

[27] RA Gopinath, MJF Gales, PS Gopalakrishnan, S Balakrishnan-Aiyer, and MA Picheny, "Robust speech recognition in noise—Performance of the IBM continuous speech recogniser on the ARPA noise spoke task," in *Proc. ARPA Workshops Spoken Lang. Syst. Technol.*, 1995, pp. 127–130.

[28] SJ Julier and JK Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[29] Y Hu and Q Huo, "An HMM compensation approach using unscented transformation for noisy speech recognition," *Chinese Spoken Language Processing*, pp. 346–357, 2006.

[30] H Xu and KK Chin, "Comparison of estimation techniques in joint uncertainty decoding for noise robust speech recognition," in *Proc. INTERSPEECH*, 2009, pp. 2403–2406.

[31] J Li, D Yu, Y Gong, and L Deng, "Unscented transform with online distortion estimation for HMM adaptation," in *Proc. INTERSPEECH*, 2010.

[32] H Xu and KK Chin, "Joint uncertainty decoding with the second order approximation for noise robust speech recognition," in *Proc. ICASSP*. IEEE, 2009, pp. 3841–3844.

[33] J Du and Q Huo, "A feature compensation approach using high-order vector Taylor series approximation of an explicit distortion model for noisy speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 8, pp. 2285–2293, 2011.

[34] RC van Dalen and MJF Gales, "Extended VTS for noise-robust speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 733–743, 2011.

[35] H Xu, MJF Gales, and KK Chin, "Joint uncertainty decoding with predictive methods for noise robust speech recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 19, no. 6, pp. 1665–1676, 2011.

[36] MJF Gales and RC Van Dalen, "Predictive linear transforms for noise robust speech recognition," in *Proc. ASRU*. IEEE, 2007, pp. 59–64.

[37] DY Kim, C Kwan Un, and NS Kim, "Speech recognition in noisy environments using first-order vector Taylor series," *Speech Communication*, vol. 24, no. 1, pp. 39–49, 1998.

[38] Y Zhao and BH Juang, "A comparative study of noise estimation algorithms for VTS-based robust speech recognition," in *Proc. INTERSPEECH*, 2010.

[39] O Kalinli, ML Seltzer, J Droppo, and A Acero, "Noise adaptive training for robust automatic speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 8, pp. 1889–1091, 2010.

[40] DB Paul and JM Baker, "The design for the Wall Street Journal-based CSR corpus," in *Proc. ICSLP*, 1992.

[41] S Young, G Evermann, MJF Gales, T Hain, D Kershaw, XA Liu, G Moore, J Odell, D Ollason, D Povey, V Valtchev, and P Woodland, "The HTK book (for HTK version 3.4)," *Cambridge University Engineering Department*, 2006.

[42] YQ Wang and MJF Gales, "Speaker and noise factorization for robust speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 7, pp. 2149–2158, 2012.

[43] F Flego and MJF Gales, "Factor analysis based VTS and JUD noise estimation and compensation," in *Proc. ICASSP*. IEEE, 2011, pp. 4792–4795.

[44] D Povey, A Ghoshal, G Boulianne, L Burget, O Glembek, N Goel, M Hannemann, P Motlıcek, Y Qian, P Schwarz, J Silovský, G Semmer, and K Veselý, "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[45] J Li, M Seltzer, and Y Gong, "Improvements to VTS feature enhanecement," in *Proc. ICASSP*. IEEE, 2012, pp. 4677–4680.

[46] A Ragni and MJF Gales, "Derivative kernels for noise robust ASR," in *Proc. ASRU*. IEEE, 2011, pp. 119–124.