



UNIVERSITY OF
CAMBRIDGE

Department of Engineering

A variational perspective on noise-robust speech recognition

R. C. van Dalen
rcv25@cam.ac.uk

M. J. F. Gales
mjfg@eng.cam.ac.uk

Technical Report CUED/F-INFENG/TR.670

28th July 2011

Abstract

Model compensation methods for noise-robust speech recognition have shown good performance. Predictive linear transformations can approximate these methods to balance computational complexity and compensation accuracy. Both of these approaches can be analysed as minimising the KL divergence to a predicted distribution. This paper examines this KL divergence from a variational perspective. Using a matched-pair approximation at the component level yields a number of standard forms of model compensation and predictive linear transformations. However, a tighter bound can be obtained by using variational approximations at the state level. Both model-based and predictive linear transform schemes can be implemented in this framework. Preliminary results show that the tighter bound resulting from the state-level variational approach can yield improved performance over standard schemes.

Contents

1	Introduction	2
2	The influence of noise on speech	3
	2.1 Sampling from the noise-corrupted speech	4
3	Predictive methods	4
	3.1 Estimating model parameters	6
	3.2 Estimating linear transformations	7
4	Predictive variational methods	9
	4.1 Sampling from a mixture model	10
	4.2 Estimating model parameters	11
	4.3 Estimating linear transformations	11
5	Noise estimation	13
	5.1 Monte Carlo noise estimation	15
	5.2 Refinements to the Monte Carlo scheme	17
6	Experiments	17
7	Conclusion	19

1 Introduction

One way of making speech recognisers more robust to noise is *model compensation*. Rather than enhancing the incoming observations, model compensation techniques modify a recogniser's state-conditional distributions so they model the speech in the target environment. Because the interaction between speech and noise is non-linear, the corrupted speech distribution has no closed form. In particular, even if the speech and noise distributions are Gaussian, the corrupted speech is not. Yet, the vast majority of schemes compensate each clean speech Gaussian with a Gaussian. Additionally, its covariance matrices are normally diagonalised. The possible improvements from computing the same form of distribution in a different way are limited (e.g. [19]).

Two approaches that remove the Gaussian assumption have been proposed. The "Algonquin" algorithm [16] still uses a Gaussian approximation, but tuned differently for each clean speech component and observation vector. Thus, its effective distribution over observation vectors is non-Gaussian. Another approach [27] uses a non-parametric distribution, by approximating the integral in the corrupted-speech likelihood expression with Monte Carlo. However, both these approaches require extensive computation for each incoming feature vector. Thus for most situations they are impractically slow.

The aim of this work is to find forms of fast likelihood computation that do model the non-Gaussian distributions. In this work the framework of *predictive methods* is applied to this problem. Predictive methods approximate a complicated model with a simpler form by minimising the KL divergence between them [25]. How this applies to noise-robust speech recognition can be seen in Fig. 1. The graphical model on the left represents noise-corrupted speech. The speech is governed by hidden Markov model, with states θ_t . Associated with each state is a mixture of Gaussians with mixture components k_t and Gaussian component distributions generating speech vectors \mathbf{x}_t . The

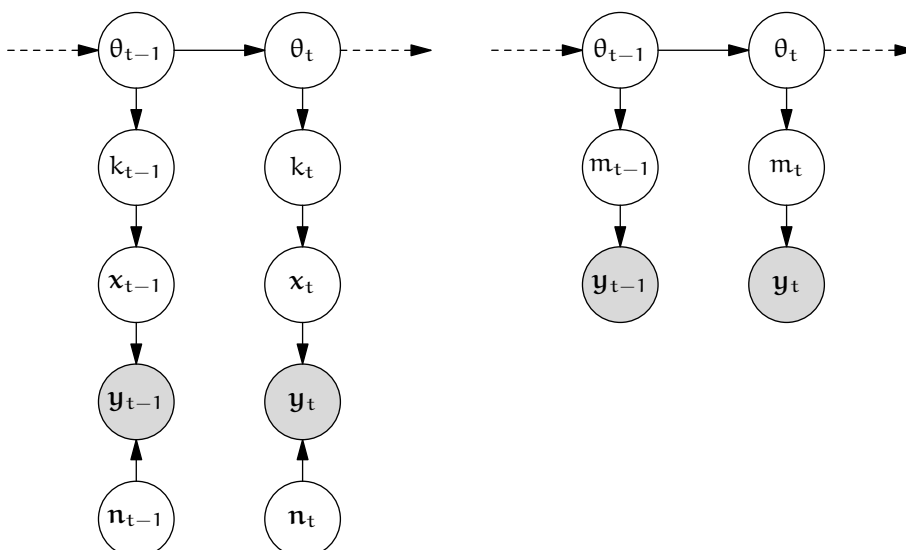


Figure 1: The noise-corrupted speech p (left), with components k , speech \mathbf{x} , noise \mathbf{n} , corrupted speech \mathbf{y} ; its approximation q (right) with components m .

independently and identically distributed noise \mathbf{n}_t corrupts the speech, resulting in noise-corrupted speech \mathbf{y}_t .

The right graphical model in Fig. 1 on the preceding page approximates the corrupted speech distribution. The Markov model $p(\theta_{t-1}|\theta_t)$, which governs the clean speech, is assumed unchanged. Many compensation methods map each component k of the predicted distribution onto one component m of the approximation and marginalise out only \mathbf{x} and \mathbf{n} . This is not a problem if the marginalisation is exact; however, usually each component is approximated with a Gaussian [2, 19, 24, 30]. This *matched-pair approximation* to the KL divergence neglects the potential for a mixture of Gaussians to approximate a mixture of non-Gaussian distributions. To approximate a whole state-conditional distribution at once, this paper will sever the hard link between components in the left- and right-hand models. Instead, a variational approach can assign part of the probability mass of component k in the left-hand model to any component m in the right-hand model. This tightens an upper bound on the KL divergence. Within this variational framework, other forms of mixture models can also be estimated. This paper will introduce a new form of predictive CMLLR, which transforms clean speech Gaussians with shared linear transformations. This yields optimal linear transformations for modelling the non-linear effects of the noise.

2 The influence of noise on speech

To make speech recognisers robust to noise, a model of how the noise influences the signal is required. Both noise and speech vectors are usually composed of Mel-frequency cepstral coefficients (MFCCs) and deltas and delta-deltas. The relation between the MFCC vectors of the speech \mathbf{x}^s , additive noise \mathbf{n}^s , and convolutional (channel) noise \mathbf{h}^s (which is usually assumed fixed, and not shown in Fig. 1), and the corrupted speech \mathbf{y}^s , called the mismatch function, is [1, 4]

$$\mathbf{y}^s = \mathbf{f}(\mathbf{x}^s, \mathbf{n}^s, \mathbf{h}^s, \boldsymbol{\alpha}) \triangleq \mathbf{C} \log \left(\exp(\mathbf{C}^{-1}(\mathbf{x}^s + \mathbf{h}^s)) + \exp(\mathbf{C}^{-1}\mathbf{n}^s) + 2\boldsymbol{\alpha} \circ \exp\left(\frac{1}{2}\mathbf{C}^{-1}(\mathbf{x}^s + \mathbf{h}^s + \mathbf{n}^s)\right) \right). \quad (1)$$

Here, $\log(\cdot)$, $\exp(\cdot)$, and \circ denote element-wise logarithm, exponentiation, and multiplication. \mathbf{C} is the truncated DCT matrix and \mathbf{C}^{-1} its pseudo-inverse. $\boldsymbol{\alpha}$ is the *phase factor*, which arises from the interaction between the phase of the speech and noise signal. Since the phase information is discarded to arrive at MFCCs, the effect of this interaction on \mathbf{y}^s is random even if \mathbf{x}^s , \mathbf{n}^s , \mathbf{h}^s are fixed. In this work, as in [27], $\boldsymbol{\alpha}$ is modelled with an independent and identically distributed truncated Gaussian with variances computed according to [17].

Speech recognisers append dynamic (delta and delta-delta) features to the MFCC feature vector. These are computed from a window of per-time slice, static, features. For exposition, assume a window ± 1 and only first-order dynamics. The observation vector \mathbf{y}_t is then computed as

$$\mathbf{y}_t = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ -\frac{1}{2}\mathbf{I} & \mathbf{0} & \frac{1}{2}\mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{y}_{t-1} \\ \mathbf{y}_t \\ \mathbf{y}_{t+1} \end{bmatrix} \triangleq \mathbf{D}\mathbf{y}_t^e, \quad (2)$$

where \mathbf{y}_t^e is the *extended* feature vector comprised of the statics in a window [30]. It is possible to compute the corrupted speech vector with statics and dynamics from a

window of speech and noise vectors by applying the mismatch function to each time slice:

$$\mathbf{y}_t = \mathbf{D} \begin{bmatrix} f(\mathbf{x}_{t-1}^s, \mathbf{n}_{t-1}^s, \mathbf{h}^s, \boldsymbol{\alpha}_{t-1}) \\ f(\mathbf{x}_t^s, \mathbf{n}_t^s, \mathbf{h}^s, \boldsymbol{\alpha}_t) \\ f(\mathbf{x}_{t+1}^s, \mathbf{n}_{t+1}^s, \mathbf{h}^s, \boldsymbol{\alpha}_{t+1}) \end{bmatrix} \triangleq \mathbf{D} f^e(\mathbf{x}_t^e, \mathbf{n}_t^e, \mathbf{h}^e, \boldsymbol{\alpha}_t^e), \quad (3)$$

where extended vectors \mathbf{x}^e , \mathbf{n}^e , \mathbf{h}^e , and $\boldsymbol{\alpha}^e$ are defined similarly to \mathbf{y}^e . Given distributions over \mathbf{x}^e , \mathbf{n}^e , $\boldsymbol{\alpha}^e$ and a setting for \mathbf{h}^e , it is now possible to write the distribution of \mathbf{y} as an integral with a Dirac delta δ :

$$p(\mathbf{y}) = \iiint \delta_{\mathbf{D} f^e(\mathbf{x}_t^e, \mathbf{n}_t^e, \mathbf{h}^e, \boldsymbol{\alpha}_t^e)}(\mathbf{y}) p(\boldsymbol{\alpha}^e) d\boldsymbol{\alpha}^e p(\mathbf{n}^e) d\mathbf{n}^e p(\mathbf{x}^e) d\mathbf{x}^e. \quad (4)$$

However, because of the non-linearity of the mismatch function, this integral has no closed form. In particular, even if the distributions over \mathbf{x}^e , \mathbf{n}^e , $\boldsymbol{\alpha}^e$ are Gaussian, $p(\mathbf{y})$ is not. It is a common approximation to linearise the mismatch function so $p(\mathbf{y})$ drops out as Gaussian [21, 2, 30], but this paper will avoid this. Given a value for \mathbf{y} and distributions for \mathbf{x}^e , \mathbf{n}^e , \mathbf{h}^e , and $\boldsymbol{\alpha}^e$, it is possible to approximate the value of the density in the limit exactly with a Monte Carlo approach [27], but this technique is too slow to be used in a speech recogniser.

2.1 Sampling from the noise-corrupted speech

The distribution $p(\mathbf{y})$ has no closed form, but it is straightforward to draw samples $\mathbf{y}^{(s)}$ from it. This first draws samples $\mathbf{x}^{e(s)}$, $\mathbf{n}^{e(s)}$, $\boldsymbol{\alpha}^{e(s)}$ from their respective distributions. (3) can then be applied to yield a corrupted speech sample:

$$\mathbf{y}^{(s)} = \mathbf{D} f^e(\mathbf{x}^{e(s)}, \mathbf{n}^{e(s)}, \mathbf{h}^e, \boldsymbol{\alpha}^{e(s)}). \quad (5)$$

The advantage of representing the distribution $p(\mathbf{y})$ by a set of samples is that it does not assume any properties of the distribution. Section 4 will estimate a form appropriate for decoding directly from this Monte Carlo approximation.

Drawing a corrupted-speech sample for a component k requires the component's extended clean speech distribution. This distribution can be found by modifying the last iteration of Baum–Welch so it does not apply the projection to statics and dynamics as it gathers statistics for training the speech recogniser's Gaussians [30]. The parameters of $p(\mathbf{n}_t)$ are often estimated on the data to be recognised using a linearisation of the mismatch function. However, see section 5 on page 13 for a strategy for estimating the noise model without using this linearisation. To remove the influence of the algorithm for noise model estimation, in the experiments the noise distribution will be assumed known.

3 Predictive methods

For practical speech recognition, it is often important to find approximations that are fast for decoding. Model compensation for noise robustness is a good example: it approximates the predicted corrupted speech with parametric distributions. This paper will present a framework for methods that minimise the Kullback-Leibler (KL) divergence between the predicted distribution p and its approximation q .

The KL divergence is suitable for two reasons. First, it is a well-understood measure of divergence between two distributions that is minimised when the distributions are equal. Second, minimising the KL divergence is equivalent to maximising the expected log-likelihood under the predicted distribution. Many known algorithms that maximise the log-likelihood on data can be turned into predictive variants.

The objective of predictive methods is to find the distribution q that minimises the KL divergence to predicted distribution p . An HMM speech recogniser consists of a distribution $p(\Theta)$ over both hidden variables $\Theta = \{\theta_t\}$, where θ_t are the HMM states in Fig. 1, and a distribution $p_\Theta(\mathcal{Y})$ over observed variables $\mathcal{Y} = \{\mathbf{y}_t\}$ given state sequence Θ . If both p and q are HMMs, they therefore factorise as

$$p(\Theta, \mathcal{Y}) = p(\Theta)p_\Theta(\mathcal{Y}); \quad q(\Theta, \mathcal{Y}) = q(\Theta)q_\Theta(\mathcal{Y}). \quad (6)$$

This paper will not change the state transition model, so that $q(\Theta) := p(\Theta)$. The optimal approximation q then is

$$\begin{aligned} q^* &:= \arg \min_q \mathcal{KL}(p\|q) \\ &= \arg \min_q \sum_{\Theta} \int p_\Theta(\mathcal{Y}) \log \left(\frac{p(\Theta)p_\Theta(\mathcal{Y})}{q(\Theta)q_\Theta(\mathcal{Y})} \right) d\mathcal{Y} \\ &= \arg \min_q \sum_{\Theta} p(\Theta) \int p_\Theta(\mathcal{Y}) \log \left(\frac{p_\Theta(\mathcal{Y})}{q_\Theta(\mathcal{Y})} \right) d\mathcal{Y} \\ &= \arg \min_q \sum_{\Theta} p(\Theta) \mathcal{KL}(p_\Theta\|q_\Theta). \end{aligned} \quad (7)$$

In HMMs, $p_\Theta(\mathcal{Y})$ and $q_\Theta(\mathcal{Y})$ represent the output distributions. Since these factorise per time, the optimal setting of q can be expressed in terms of the per-state KL divergence:

$$q^* := \arg \min_q \sum_{\Theta} p(\Theta) \mathcal{KL}(p_\Theta\|q_\Theta) \quad (8a)$$

Here, $p(\theta)$ is the prior distribution for state θ . The maximum-likelihood estimate of $p(\theta)$ is proportional to the occupancy of state θ on the training data.

The KL divergence can be defined in terms of the cross-entropy $\mathcal{H}(p_\Theta\|q_\Theta)$ and the entropy $\mathcal{H}(p_\Theta)$:

$$\mathcal{KL}(p_\Theta\|q_\Theta) \triangleq \mathcal{H}(p_\Theta\|q_\Theta) - \mathcal{H}(p_\Theta); \quad (8b)$$

$$\mathcal{H}(p_\Theta\|q_\Theta) = - \int p_\Theta(\mathbf{y}) \log q_\Theta(\mathbf{y}) d\mathbf{y}; \quad (8c)$$

$$\mathcal{H}(p_\Theta) = - \int p_\Theta(\mathbf{y}) \log p_\Theta(\mathbf{y}) d\mathbf{y}. \quad (8d)$$

When minimising the KL divergence with respect to q , the entropy $\mathcal{H}(p_\Theta)$ is constant, so that in (8a) only the cross-entropy needs to be minimised:

$$q^* := \arg \min_q \sum_{\Theta} p(\Theta) \mathcal{H}(p_\Theta\|q_\Theta). \quad (8e)$$

In speech recognition, the state-conditional distributions p_θ and q_θ are normally mixtures of Gaussians. Their weights will be written π_k and ω_m :

$$p_\theta(\mathbf{y}) = \sum_{k \in \Omega_\theta} \pi_k p_k(\mathbf{y}); \quad q_\theta(\mathbf{y}) = \sum_{m \in \Omega_\theta} \omega_m q_m(\mathbf{y}). \quad (9)$$

The key insight that motivates this paper is that unless it is possible to set q_m exactly equal to p_k , the KL divergence between these mixtures, in (8e), can usually not be analytically minimised exactly. Section 4 will introduce a variational upper bound to perform the minimisation. Initially, though, a simpler and commonly-used bound will be considered: the *matched-pair bound* [12], which relates each component k in p_θ to a component in q_θ , in this case $m = k$:

$$\begin{aligned} \sum_{\theta} p(\theta) \mathcal{H}(p_\theta \| q_\theta) &= - \sum_{\theta} p(\theta) \int p_\theta(\mathbf{y}) \log q_\theta(\mathbf{y}) d\mathbf{y} \\ &= - \sum_{\theta} \sum_{k \in \Omega_\theta} p(\theta) \pi_k \int p_k(\mathbf{y}) \log \left(\sum_{m \in \Omega_\theta} \omega_m q_m(\mathbf{y}) \right) d\mathbf{y} \\ &\leq - \sum_{\theta} \sum_{k \in \Omega_\theta} p(\theta) \pi_k \int p_k(\mathbf{y}) \log(\omega_k q_k(\mathbf{y})) d\mathbf{y} \\ &= \sum_{\theta} \sum_{k \in \Omega_\theta} p(k) (\mathcal{H}(p_k \| q_k) - \log(\omega_k)), \end{aligned} \quad (10a)$$

where the component priors $p(k) = p(\theta) \pi_k$. The minimisation in (8e) then becomes

$$q^* := \arg \min_q \sum_{\theta} \sum_{k \in \Omega_\theta} p(k) (\mathcal{H}(p_k \| q_k) - \log(\omega_k)). \quad (10b)$$

The optimal setting for the mixture weights is $\omega_k = \pi_k$. How to minimise the weighted cross-entropies $\mathcal{H}(p_k \| q_k)$ depends on the parametrisation of q_k . The following will consider two parametrisations: model compensation, which estimates each Gaussian parameter separately; and linear transformations that are shared between clean speech Gaussians.

3.1 Estimating model parameters

A straightforward method for optimising the cross-entropy in (10b) is to estimate the parameters of q_k directly. This can be done for each component k separately. Speech recogniser components are usually Gaussians, with parameters $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, which can be found with

$$\boldsymbol{\mu}_k := \mathcal{E}_{p_k}\{\mathbf{y}\}; \quad \boldsymbol{\Sigma}_k := \mathcal{E}_{p_k}\{\mathbf{y}\mathbf{y}^T\} - \boldsymbol{\mu}_k \boldsymbol{\mu}_k^T. \quad (11)$$

This is what model compensation methods approximate. A popular approach is to linearise the influence of the noise in (3), so that the noise-corrupted speech distribution drops out as Gaussian. This is called vector Taylor series (vts) compensation [21, 2, 4, 30].

To obtain a more accurate estimate, a Monte Carlo approach will be used, called extended DPMC [11, 26]. For each component, S samples $\mathbf{y}^{(s)}$ are drawn as discussed

in section 2.1. The expectations in (11) are then approximated with

$$\mathcal{E}_{p_k}\{\mathbf{y}\} \simeq \frac{1}{S} \sum_s \mathbf{y}^{(s)}; \quad \mathcal{E}_{p_k}\{\mathbf{y}\mathbf{y}^T\} \simeq \frac{1}{S} \sum_s \mathbf{y}^{(s)}\mathbf{y}^{(s)T}. \quad (12)$$

In the limit as the number of samples goes to infinity, extended DPMC yields the optimal Gaussian-for-Gaussian compensation. However, the corrupted speech distribution for one clean speech Gaussian is not Gaussian-distributed.

3.2 Estimating linear transformations

An approach that requires fewer parameters to be estimated is to constrain the approximate distribution q to a transformed clean speech model set. CMLLR [9] (constrained maximum-likelihood linear regression) is a well-known adaptation method that applies one affine transformation to all Gaussian components in a base class. CMLLR is normally estimated on adaptation data with expectation-maximisation. For its predictive variant, predictive CMLLR (PCMLLR) [10], the statistics from adaptation data are replaced by predicted statistics. Good results have been obtained by deriving these statistics from *joint uncertainty decoding* [10, 32] and from vocal tract length normalisation [3].

CMLLR applies the same linear transformation to each Gaussian in one base class. This is equivalent to applying the inverse transformation to the feature vector. That all Gaussians in one base class can then share the same transformed feature vector makes decoding with CMLLR fast. For component k with $q_k = \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$, in base class c , transformation $\{\mathbf{A}^{(c)}, \mathbf{b}^{(c)}\}$ is applied as

$$q_k(\mathbf{y}) \triangleq |\mathbf{A}^{(c)}| \cdot \mathcal{N}(\mathbf{A}^{(c)}\mathbf{y} + \mathbf{b}^{(c)}; \boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x). \quad (13)$$

Since $\mathbf{A}^{(c)}$ can be full, CMLLR allows modelling of feature correlations even when Gaussians' covariances are diagonal in a manner that hardly affects decoding speed.

To be able to convert existing methods that maximise a likelihood of training or adaptation data (such as CMLLR) into predictive methods, it is useful to describe the maximisation step of expectation-maximisation in terms of the KL divergence between the distribution of the training data and of the model [29]. The training data examples $\mathcal{Y}^{(i)}$ can be represented as Dirac deltas that form the *empirical distribution* $\tilde{p}(\mathcal{Y})$. The expectation step of expectation-maximisation then finds the optimal distribution $\rho_{\mathcal{Y}}(\Theta)$ of the hidden variables for training sequence \mathcal{Y} using the current model parameters. The distribution of the complete data can then be written as

$$\tilde{p} = \tilde{p}(\mathcal{Y})\rho_{\mathcal{Y}}(\Theta). \quad (14)$$

The maximisation step then sets the parameters of q to minimise $\mathcal{KL}(\tilde{p}||q)$. Once a method is expressed in these terms, as in [29], converting it to a predictive method is a matter of replacing empirical distribution \tilde{p} by predicted distribution p .

Once the original, adaptive, version of CMLLR is written as minimising a KL divergence to the complete data distribution, the only change required to convert from the adaptive to the predictive version is in the form of the statistics; the algorithm to estimate the transformations from these statistics remains the same. The statistics consists

of total occupancy $\gamma^{(c)}$ for each base class, and vectors $\mathbf{k}^{(ci)}$ and matrices $\mathbf{G}^{(ci)}$ for each base class c and each dimension i . Expressed in terms of distribution p ,

$$\gamma^{(c)} \triangleq \sum_{\mathbf{k} \in \Omega^{(c)}} p(\mathbf{k}); \quad (15a)$$

$$\mathbf{k}^{(ci)} \triangleq \sum_{\mathbf{k} \in \Omega^{(c)}} p(\mathbf{k}) \frac{\mu_{\mathbf{k},i}}{\sigma_{\mathbf{k},i}^2} \mathcal{E}_{p_{\mathbf{k}}} \left\{ \left[\mathbf{y}^T \ 1 \right] \right\}; \quad (15b)$$

$$\mathbf{G}^{(ci)} \triangleq \sum_{\mathbf{k} \in \Omega^{(c)}} p(\mathbf{k}) \frac{1}{\sigma_{\mathbf{k},i}^2} \mathcal{E}_{p_{\mathbf{k}}} \left\{ \left[\begin{array}{c} \mathbf{y} \mathbf{y}^T \ \mathbf{y} \\ \mathbf{y}^T \ 1 \end{array} \right] \right\}, \quad (15c)$$

where $p(\mathbf{k})$ is the component prior, and $\mathcal{E}_{p_{\mathbf{k}}}\{\cdot\}$ denotes the expectation under $p_{\mathbf{k}}$. It is straightforward to recover the original expression in [9] from this by substituting the distribution over the complete data for p .

$\mathbf{G}^{(ci)}$ can be computed with any method that yields component-dependent first- and second-order statistics in (11). This paper will use the Monte Carlo estimates in (12), but implement two improvements for efficiency. Substituting the Monte Carlo estimates for each component would mean that an equal number of samples is drawn for each component, whether they contribute much to (15c) or not. The contribution is determined mostly by the component prior $p(\mathbf{k})$. The first improvement is therefore to draw samples from both the component prior and the corrupted speech for the component. This yields joint samples $(\mathbf{k}^{(s)}, \mathbf{y}^{(s)})$. The sum and the expectation in (15c) are then approximated as (writing explicitly only the most complicated statistic),

$$\begin{aligned} \mathbf{G}^{(ci)} &\triangleq \sum_{\mathbf{k} \in \Omega^{(c)}} p(\mathbf{k}) \frac{1}{\sigma_{\mathbf{k},i}^2} \mathcal{E}_{p_{\mathbf{k}}} \left\{ \left[\begin{array}{c} \mathbf{y} \mathbf{y}^T \ \mathbf{y} \\ \mathbf{y}^T \ 1 \end{array} \right] \right\} \\ &\simeq \sum_s \frac{1}{\sigma_{\mathbf{k}^{(s)},i}^2} \left[\begin{array}{c} \mathbf{y}^{(s)} \mathbf{y}^{(s)T} \ \mathbf{y}^{(s)} \\ \mathbf{y}^{(s)T} \ 1 \end{array} \right]. \end{aligned} \quad (16)$$

However, it is now possible to draw no or very few samples from a component \mathbf{k} that has a high prior, which makes the variance across different draws of $\mathbf{G}^{(ci)}$ greater, so that more samples are necessary. To improve this, samples $\mathbf{k}^{(s)}$ can be generated using another scheme, as long as they yield an unbiased estimate of (15c). Note that samples $\mathbf{k}^{(s)}$ are not required to be independent. The requirements are very similar to those of *re-sampling* in sequential Monte Carlo methods (see, e.g., [5]). In particular, if S samples are drawn, it is straightforward to derive a scheme that makes sure that the number of samples for component \mathbf{k} is at least $\lfloor S \cdot p(\mathbf{k}) \rfloor$. Divide the probability mass of $p(\mathbf{k})$ into two parts, $p_1(\mathbf{k}) \geq 0$ and $p_2(\mathbf{k}) \geq 0$, so that $p_1(\mathbf{k}) + p_2(\mathbf{k}) = p(\mathbf{k})$. (15c) can then trivially be split into the corresponding two parts:

$$\mathbf{G}^{(ci)} = \sum_{\mathbf{k} \in \Omega^{(c)}} p_1(\mathbf{k}) \frac{1}{\sigma_{\mathbf{k},i}^2} \mathcal{E}_{p_{\mathbf{k}}} \left\{ \left[\begin{array}{c} \mathbf{y} \mathbf{y}^T \ \mathbf{y} \\ \mathbf{y}^T \ 1 \end{array} \right] \right\} + \sum_{\mathbf{k} \in \Omega^{(c)}} p_2(\mathbf{k}) \frac{1}{\sigma_{\mathbf{k},i}^2} \mathcal{E}_{p_{\mathbf{k}}} \left\{ \left[\begin{array}{c} \mathbf{y} \mathbf{y}^T \ \mathbf{y} \\ \mathbf{y}^T \ 1 \end{array} \right] \right\}. \quad (17a)$$

If the first part of the priors is now defined to yield the floor of the expected number of samples, and the second part as its residual,

$$p_1(\mathbf{k}) \triangleq \frac{\lfloor S \cdot p(\mathbf{k}) \rfloor}{p(\mathbf{k})}; \quad p_2(\mathbf{k}) \triangleq p(\mathbf{k}) - p_1(\mathbf{k}), \quad (17b)$$

the sum in first term in (17a) can be approximated exactly because for each component, $S \cdot p_1(k)$ is integer. Having thus approximated the first term, the second term in (17a) becomes 0 as the number of samples goes to infinity, so that any scheme for approximating it makes the estimate of $\mathbf{G}^{(ci)}$ unbiased. If samples are drawn independently from the second term in (17a), this is equivalent to *residual sampling*. This paper will instead apply *systematic sampling* [15].

However, the first- and second-order statistics that $\mathbf{G}^{(ci)}$ requires per component still make it impossible to model non-Gaussian effects in the distribution p_k . To do that, the next section will introduce an approach to estimating q per state.

4 Predictive variational methods

Minimising the KL divergences between states, instead of their components, makes it possible to approximate the non-Gaussian shape of p . In speech recognition, q_θ is normally a mixture of distributions q_m with weights π_m , as in (9). This section will first introduce a variational approach to estimating this mixture model. This requires p to be replaced with a Monte Carlo approximation. Then, two forms of parametrisation of q_θ will be discussed. The first, variational extended DPMC, estimates the components' weights, means, and covariances. The other, variational PCMLLR, only estimates the parameters of linear transformations shared between multiple clean speech Gaussians across states.

As in section 3, p is replaced with a Monte Carlo version by drawing joint samples $(\theta^{(s)}, \mathbf{y}^{(s)})$ from $p(\theta, \mathbf{y})$. Section 4.1 will discuss the sampling process in more detail. The joint samples are used to at the same time approximate the sum and the integral over the term in square brackets:

$$\begin{aligned} \sum_{\theta} p(\theta) \mathcal{H}(p_{\theta} \| q_{\theta}) &= - \sum_{\theta} p(\theta) \int p_{\theta}(\mathbf{y}) [\log q_{\theta}(\mathbf{y})] d\mathbf{y} \\ &\simeq - \sum_s \log q_{\theta^{(s)}}(\mathbf{y}^{(s)}) \end{aligned} \quad (18a)$$

Since the q_{θ} are mixture models, this expression cannot be minimised analytically. However, similarly to in expectation-maximisation, it is possible to minimise a variational upper bound as a proxy for optimising (18a) itself. This introduces an additional set of parameters whose optimisation is interleaved with the optimisation of q . These parameters $r_m^{(s)} \geq 0$ can be seen as assigning a fractional responsibility for sample s to component m . By insuring that for each sample s the responsibilities are normalised, $\sum_m r_m^{(s)} = 1$, Jensen's inequality can be used (in (18b)) to find an upper bound \mathcal{F} to the weighted cross-entropy in (8e). Substituting (9) into (18a), and inserting variational

parameters $\mathbf{r}_m^{(s)}$,

$$\begin{aligned}
 -\sum_s \log q_{\theta^{(s)}}(\mathbf{y}^{(s)}) &= -\sum_s \log \left(\sum_{m \in \Omega_{\theta^{(s)}}} \omega_m q_m(\mathbf{y}^{(s)}) \right) \\
 &= -\sum_s \log \left(\sum_{m \in \Omega_{\theta^{(s)}}} r_m^{(s)} \frac{\omega_m q_m(\mathbf{y}^{(s)})}{r_m^{(s)}} \right) \\
 &\leq -\sum_s \sum_{m \in \Omega_{\theta^{(s)}}} r_m^{(s)} \log \left(\frac{\omega_m q_m(\mathbf{y}^{(s)})}{r_m^{(s)}} \right) \quad (18b) \\
 &= -\sum_s \sum_{m \in \Omega_{\theta^{(s)}}} r_m^{(s)} \left(\log q_m(\mathbf{y}^{(s)}) + \log \frac{\omega_m}{r_m^{(s)}} \right) \\
 &\triangleq \mathcal{F}(\mathbf{p}, \mathbf{q}, \mathbf{r}). \quad (18c)
 \end{aligned}$$

The minimisation of \mathcal{F} interleaves optimising \mathbf{r} , the collection of $\mathbf{r}_m^{(s)}$, and \mathbf{q} . The optimal setting for the responsibilities for each sample can be found by introducing Lagrange multipliers $\lambda^{(s)}$ to make sure the responsibilities are normalised for one sample:

$$\begin{aligned}
 \frac{d(\mathcal{F}(\mathbf{p}, \mathbf{q}, \mathbf{r}) - \lambda^{(s)} (\sum_{m'} r_{m'}^{(s)} - 1))}{dr_m^{(s)}} &= 0; \\
 -\log(\pi_m q_m(\mathbf{y}^{(s)})) + 1 + \log r_m^{(s)} - \lambda^{(s)} &= 0.
 \end{aligned}$$

Re-arranging yields

$$\begin{aligned}
 \log r_m^{(s)} &= \log(\pi_m q_m(\mathbf{y}^{(s)})) - 1 + \lambda^{(s)}; \\
 r_m^{(s)} &= \pi_m q_m(\mathbf{y}^{(s)}) \exp(-1 + \lambda^{(s)}); \\
 r_m^{(s)} &:= \frac{\pi_m q_m(\mathbf{y}^{(s)})}{\sum_{m'} \pi_{m'} q_{m'}(\mathbf{y}^{(s)})}. \quad (19)
 \end{aligned}$$

How to optimise the distribution \mathbf{q} depends on its form. Sections 4.2 and 4.3 will discuss two forms.

4.1 Sampling from a mixture model

It is interesting to consider that the predicted distribution \mathbf{p}_θ is a mixture model. To draw a sample from \mathbf{p} , a sample $\theta^{(s)}$ from the state prior $\mathbf{p}(\theta)$ is drawn, and then a corresponding component $k^{(s)}$ with probability π_k . Section 2.1 has discussed how to draw sample $\mathbf{y}^{(s)}$ from the distribution $\mathbf{p}_{k^{(s)}}$.

The resulting joint sample $(\theta^{(s)}, \mathbf{y}^{(s)})$ can contribute to the estimation of any or all components of \mathbf{q}_θ : there is no explicit link to the component of \mathbf{p}_θ the sample was drawn from. An important consequence of this is that the components of \mathbf{q}_θ are free to move in space to represent samples drawn from other components than their counterparts in \mathbf{p}_θ . This allows the non-Gaussian shape of \mathbf{p}_k to be modelled.

The scheme that was discussed in section 3 found a matched-pair bound to the variational approximation discussed in this section. This assigns the responsibility for

each sample to the component of q_θ matching the component of p_θ . The variational approach becomes equal to the matched-pair bound when $r_m^{(s)}$ in (18b) is set to 1 for $m = k^{(s)}$ and to 0 otherwise. This upper bound will be used as an initialisation for the variational minimisation.

4.2 Estimating model parameters

As in section 3.1, it is possible to estimate the model parameters directly. This has been called “extended iterative DPMC” [11, 29], but here it will be called “variational eDPMC”. It is possible to optimise each state-conditional distribution q_θ separately. The update to the parameters of q_θ that optimises (18c) is the same as in expectation–maximisation for mixtures of Gaussians.

The component weights are found by introducing Lagrange multipliers $\lambda^{(s)}$ to make sure the weights are normalised for each state:

$$\begin{aligned} \frac{d(\mathcal{F}(p, q, \mathbf{r}) - \lambda^{(\theta)} (\sum_{m' \in \Omega_\theta} \omega_{m'} - 1))}{d\omega_m} &= 0; \\ d\left(\left(\sum_s r_m^{(s)} \log \omega_m\right) + \lambda^{(\theta)} (\sum_{m' \in \Omega_\theta} \omega_{m'} - 1)\right) &= 0. \end{aligned}$$

Re-arranging yields

$$\frac{\sum_s r_m^{(s)}}{\omega_m} = -\lambda^{(\theta)}; \quad \omega_m = \frac{\sum_s r_m^{(s)}}{-\lambda^{(\theta)}}; \quad \omega_m := \frac{\sum_s r_m^{(s)}}{\sum_{m' \in \Omega_\theta} \sum_s r_m^{(s)}}. \quad (20a)$$

Optimising the means and covariances of q_m is separate per component:

$$\frac{d\mathcal{F}(p, q, \mathbf{r})}{d\omega_m} = \frac{d \sum_s r_m^{(s)} \log q_m(\mathbf{y}^{(s)})}{d\omega_m} = 0, \quad (20b)$$

the well-known solution for which is

$$\boldsymbol{\mu}_m := \frac{1}{\sum_s r_m^{(s)}} \sum_s r_m^{(s)} \mathbf{y}^{(s)}; \quad \boldsymbol{\Sigma}_m := \left(\frac{1}{\sum_s r_m^{(s)}} \sum_s r_m^{(s)} \mathbf{y}^{(s)} \mathbf{y}^{(s)\top} \right) - \boldsymbol{\mu}_m \boldsymbol{\mu}_m^\top. \quad (20c)$$

Like in standard expectation–maximisation, this allows components to move in space to model the shape of the distribution, irrespective of which components the samples were originally drawn from.

4.3 Estimating linear transformations

It is also possible to apply predictive CMLLR in the variational Monte Carlo framework. The difference with variational eDPMC is that the only parameters that are estimated here are of a set of linear transformations. These transformations apply to the original clean speech Gaussians.

Note that, like in (13), different components for one state-conditional mixture can be transformed differently if they are in different base classes. However, because here

a variational approach is used, the optimisation can move Gaussians in the original model and re-use them to model parts of space that are generated by different Gaussians. This has the counter-intuitive effect that linear transformations of Gaussians allow modelling of the non-Gaussian shape of a distribution predicted from essentially the same Gaussians.

Here the predicted distribution p_k in (15c) is replaced by its Monte Carlo approximation. The derivation is not given here, but see [29] for details. Similarly to in (18a), joint samples at the same time approximate the sum and the integral in the expectation:

$$\mathbf{G}^{(ci)} \simeq \sum_s \sum_{m \in \Omega(c)} r_m^{(s)} \frac{1}{\sigma_{m,i}^2} \begin{bmatrix} \mathbf{y}^{(s)} \mathbf{y}^{(s)\top} & \mathbf{y}^{(s)} \\ \mathbf{y}^{(s)\top} & 1 \end{bmatrix}. \quad (21)$$

Given these statistics for one base class c , the CMLLR algorithm iteratively finds transformation $\{\mathbf{A}^{(c)}, \mathbf{b}^{(c)}\}$ that yields a local minimum for the variational upper bound in (18c).

However, gathering the statistics cannot be performed per base class, since a sample drawn from state $\theta^{(s)}$ can contribute to any of that state's components, which can be in any base class. This implies that it is impossible to guarantee that each $\mathbf{G}^{(ci)}$ is estimated on enough samples: the mass of samples drawn from a component in one base class can be assigned to components of another base class. However, this re-assignment is exactly the purpose of the variational scheme. Allowing components to be transformed to model part of the probability mass from other base classes removes the restriction that the noise-corrupted speech for one clean speech Gaussian is modelled by a Gaussian.

To partly mitigate the potential problem that some base classes are trained on too little data, an equal number of samples can be drawn from each base class, and they then be weighted. The weight is the base class prior, which is proportional to the sum of the priors of its components:

$$p(c) = \frac{\sum_{k \in \Omega(c)} p(k)}{\sum_k p(k)}. \quad (22a)$$

For each base class c , an equal number of samples are drawn. For each base class sample $c^{(s)}$, a component $k^{(s)}$ is drawn with probability proportional to $p(k)$. This component is in the mixture of a state that will be indicated with $\theta^{(s)}$. A corrupted-speech vector $\mathbf{y}^{(s)}$ is then drawn from the component distribution p_k . The joint samples $(c^{(s)}, \theta^{(s)}, k^{(s)}, \mathbf{y}^{(s)})$ now approximate the joint distribution of these with an empirical distribution (a distribution with delta spikes at the points corresponding to the samples):

$$p(c, \theta, k, \mathbf{y}) \simeq \tilde{p}(c, \theta, k, \mathbf{y}) \triangleq \sum_s p(c^{(s)}) \delta_{(c^{(s)}, \theta^{(s)}, k^{(s)}, \mathbf{y}^{(s)})}(c, \theta, k, \mathbf{y}). \quad (22b)$$

Using this empirical distribution introduces sample weights $p(c^{(s)})$ in the Monte Carlo approximation to the overall cross-entropy in (18a):

$$\begin{aligned} \sum_{\theta} p(\theta) \mathcal{H}(p_{\theta} \| q_{\theta}) &= - \sum_{\theta} p(\theta) \int p_{\theta}(\mathbf{y}) [\log q_{\theta}(\mathbf{y})] d\mathbf{y} \\ &\simeq - \sum_s p(c^{(s)}) \log q_{\theta^{(s)}}(\mathbf{y}^{(s)}) \end{aligned} \quad (22c)$$

The easiest way to introduce the weighting of the samples in the upper bound in (18c) is to adapt the responsibilities so they add up to the sample weight: $\sum_m r_m^{(s)} = p(c^{(s)})$. (19) then changes into

$$r_m^{(s)} := p(c^{(s)}) \frac{\pi_m q_m(\mathbf{y}^{(s)})}{\sum_{m'} \pi_{m'} q_{m'}(\mathbf{y}^{(s)})}. \quad (22d)$$

Using these weighted responsibilities, the statistics can be found as in (21). Additionally, as in section 3.2 on page 7, to reduce the variance of the statistics, systematic sampling can be used for drawing samples $k^{(s)}$ for each base class.

5 Noise estimation

The discussion has so far assumed that the distribution of the noise is known. In practice, however, this is seldom the case. The noise model, with parameters for the additive and convolutional noise, must therefore be estimated. The usual approach is to apply expectation–maximisation in an unsupervised-adaptation framework. The aim is then to optimise the noise model parameters to improve the likelihood using the form of model compensation that is used for decoding. Usually, noise model estimation applies a vector Taylor series approximation to the mismatch function. It is then possible to optimise the likelihood function directly with respect to the noise [21, 20, 18]. This allows the convolutional noise to be assumed constant. Alternatively, the noise can be modelled as a hidden variable [14, 8, 13]: using the vector Taylor series approximation, the posterior of this variable can be approximated with a Gaussian. For this approach there are issues with the diagonalisation of covariance matrices [7], and the convolutional noise must be assumed randomly distributed, or expectation–maximisation does not work. Either approach can end up oscillating if the new noise mean is used as a vector Taylor series expansion point as well. Even if the noise model estimation finds a minimum, it is optimised for the linearised mismatch function. In this paper, the linearisation is not applied, so a different approach should be used.

The underlying parameters for predictive transformations, the noise model in this case, is normally estimated for the predicted distribution and not for its approximation [26, 29, 32]. The reason for this is that many predictive methods, for example, full-transform PCMLLR, have no algebraic relation to the statistics it uses, but an algorithmic one, so it is not possible to perform inference from data. When PCMLLR is estimated from a Monte Carlo approximation of the exact corrupted speech, the obvious approach would be to estimate the noise model for the exact corrupted speech.

One approach for doing this is to find a numerical approximation to the posterior of the noise. [22] propose a method that uses a consistent approximation for estimation and compensation, which does not apply the vector Taylor series approximation. However, it assumes all dimensions independent, and it is not feasible to generalise it to multiple dimensions (see [29]).

Alternatively, it is possible to approximate the noise posterior with Monte Carlo [6]. However, [6] applies this to feature enhancement, which has a number of important simplifying consequences. It assumes that feature dimensions are independent, and only considers static features. Additionally, the approach approximates the clean speech model with a mixture of Gaussians, so that the speech model has only one state, and a small number of components. This makes importance sampling feasible. This

section will discuss this method and describe a way of overcoming these limitations for model compensation with dependent feature dimensions. For simplicity of notation, only additive noise will be assumed, but a distribution over the convolutional noise can be estimated in the same way.

Noise estimation based on expectation–maximisation is conceptually similar to training or adapting a speech recogniser. In those cases, the hidden variables are the components, and in the expectation step the posterior distribution over the component sequence $\rho_t(\{\mathbf{k}_t\})$ is found. Since given the marginal distribution over the components at each time instance, the speech is independent from between time instances, the posterior distribution can be represented with just the component–time posteriors γ_{tk} .

Here, on the other hand, the hidden variables per time frame are not only the component identities, but also the noise (the speech will be marginalised out). In iteration i , the expectation step finds distribution $\rho_t^{(i)}(\mathbf{k}_t, \mathbf{n}_t)$ using current noise distribution $p_n^{(i)}(\mathbf{n})$. Since, again, given the distribution over components, the time instances become independent, the marginal distributions per time instance, $\rho_t(\mathbf{k}_t, \mathbf{n}_t)$, will be of most interest. They can be factorised as

$$\rho_t(\mathbf{k}_t, \mathbf{n}_t) = \rho_t(\mathbf{k}_t)\rho_t(\mathbf{n}_t|\mathbf{k}_t). \quad (23)$$

The distribution $\rho_t(\mathbf{k}_t)$ is equivalent and equal to the normal component-time distribution and could be represented by γ_{tk} . However, $\rho_t(\mathbf{n}_t|\mathbf{k}_t)$ has no closed form, and it will be necessary to approximate it, e.g. with Monte Carlo. Here, the whole distribution $\rho_t(\mathbf{k}_t, \mathbf{n}_t)$ will be sampled from, so it will be written as a joint distribution.

The maximisation step finds the maximum-likelihood parameters of the new noise distribution $p_n^{(i+1)}(\mathbf{n})$.

$$p_n^{(i+1)} := \arg \max_{p_n} \sum_t \sum_k \int \rho_t(\mathbf{k}_t, \mathbf{n}_t) \log p(\mathbf{y}_t, \mathbf{n}_t|\mathbf{k}) d\mathbf{n}_t, \quad (24a)$$

where

$$p(\mathbf{y}_t, \mathbf{n}_t|\mathbf{k}) = p(\mathbf{y}_t|\mathbf{n}_t, \mathbf{k}) p_n(\mathbf{n}_t). \quad (24b)$$

The first term does not depend on the noise model p_n but on the environment model and the clean speech model, so it is not optimised here. The maximisation expression in (24a) thus becomes

$$p_n^{(i+1)} := \arg \max_{p_n} \sum_t \sum_k \int \rho_t(\mathbf{k}_t, \mathbf{n}_t) \log p_n(\mathbf{n}_t) d\mathbf{n}_t. \quad (24c)$$

A standard noise model is Gaussian with parameters $\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n$. It is well-known how to maximum-likelihood estimate a Gaussian (but [6] gives a complete derivation): the mean $\boldsymbol{\mu}_n$ is set to $\mathcal{E}\{\mathbf{n}\}$ and the covariance $\boldsymbol{\Sigma}_n$ to $\mathcal{E}\{\mathbf{n}\mathbf{n}^T\} - \boldsymbol{\mu}_n\boldsymbol{\mu}_n^T$. In this case, the expectation is under distribution ρ_t and over t, \mathbf{k}_t , and \mathbf{n}_t . The optimal setting for the mean (estimating the covariance is analogous) is

$$\boldsymbol{\mu}_n^{(i+1)} := \frac{1}{T} \sum_t \sum_k \int \rho_t(\mathbf{k}_t, \mathbf{n}_t) \mathbf{n}_t d\mathbf{n}_t. \quad (25)$$

As mentioned, the distribution ρ does not have a closed form. It is a standard strategy to use the same vector Taylor series approximation used for compensation

[14, 8, 13]. The joint distribution of the noise and observation then becomes Gaussian for every component.¹ However, the following will present a Monte Carlo approximation of the integral.

5.1 Monte Carlo noise estimation

A basic requirement for applying any Monte Carlo approximation to the integral under distribution ρ , which the expectation step of expectation–maximisation finds, is that ρ can be evaluated at any given point. It must be rewritten to make that possible. As in (23), the marginal of the posterior distribution for one time instance factorises as

$$\rho_t(k_t, \mathbf{n}_t) = \rho_t(k_t) \rho_t(\mathbf{n}_t | k_t). \quad (26)$$

$\rho_t(k_t)$ is acquired with the forward-backward algorithm on the current model. Since it is not feasible to decode with the real corrupted speech distribution, as an approximation, PCMLLR with the current noise model can be applied. The second part, $\rho_t(\mathbf{n}_t | k_t)$, is set to the posterior of \mathbf{n}_t given k_t and the observed variable \mathbf{y}_t , using the current noise model $p_n^{(i)}(\mathbf{n}_t)$:

$$\begin{aligned} \rho_t(\mathbf{n}_t | k_t) &= \frac{p(\mathbf{y}_t | k_t, \mathbf{n}_t) p_n^{(i)}(\mathbf{n}_t)}{p(\mathbf{y}_t | \mathbf{n}_t)} \\ &\propto p(\mathbf{y}_t | k_t, \mathbf{n}_t) p_n^{(i)}(\mathbf{n}_t). \end{aligned} \quad (27a)$$

Assuming α fixed, it is possible to evaluate this expression for given $\mathbf{n}_t, k_t, \mathbf{y}_t$. It is possible, though unnecessary, to use a Parzen window approximation, as in [6]. Using a variable transformation [22],

$$p(\mathbf{y}_t | k_t, \mathbf{n}_t) = \left| \frac{\partial \mathbf{x}(\mathbf{n}_t, \mathbf{y})}{\partial \mathbf{y}} \right|_{\mathbf{y}_t} \cdot p_k(\mathbf{x}(\mathbf{n}_t, \mathbf{y})), \quad (27b)$$

where $\mathbf{x}(\mathbf{n}, \mathbf{y})$ denotes the speech vector \mathbf{x} corresponding to setting the noise and observation vectors to \mathbf{n} and \mathbf{y} , and $p_k(\mathbf{x}(\mathbf{n}_t, \mathbf{y}))$ denotes the value of the clean speech distribution $p_k(\mathbf{x})$ at that point. This is straightforward to evaluate: [28] shows that the diagonal elements of the Jacobian are²

$$\left. \frac{\partial \mathbf{x}(\mathbf{n}_{t,i}, \mathbf{y})}{\partial \mathbf{y}} \right|_{\mathbf{y}_{t,i}} = \frac{1}{1 - \exp(\mathbf{n}_{t,i} - \mathbf{y}_{t,i})}. \quad (27c)$$

If a phase factor distribution is used, then $p_k(\mathbf{x}(\mathbf{n}_t, \mathbf{y}))$ in (27b) can be rewritten in a similar way, and an extra integral (over α) must be inserted.

[6] applies importance sampling to approximate (25). This draws samples $(k_t^{(s)}, \mathbf{n}_t^{(s)})$ from a proposal distribution \mathbf{v}_t , and weights the samples to make up for the difference

¹There are additional issues with diagonalisation [7].

²[28] derives this with \mathbf{x} and \mathbf{n} exchanged, but the mismatch function is symmetric.

between the proposal and actual distributions:

$$\begin{aligned}
\boldsymbol{\mu}_n^{(i+1)} &:= \frac{1}{T} \sum_t \sum_{\mathbf{k}_t} \int \rho_t(\mathbf{k}_t, \mathbf{n}_t) \mathbf{n}_t d\mathbf{n}_t \\
&= \frac{1}{T} \sum_t \sum_{\mathbf{k}_t} \int \frac{\rho_t(\mathbf{k}_t) \rho_t(\mathbf{n}_t | \mathbf{k}_t)}{v_t(\mathbf{k}_t, \mathbf{n}_t)} v_t(\mathbf{k}_t, \mathbf{n}_t) \mathbf{n}_t d\mathbf{n}_t \\
&\simeq \frac{1}{T} \sum_t \frac{1}{S} \sum_s \frac{\rho_t(\mathbf{k}_t^{(s)}) \rho_t(\mathbf{n}_t^{(s)} | \mathbf{k}_t^{(s)})}{v_t(\mathbf{k}_t^{(s)}, \mathbf{n}_t^{(s)})} \mathbf{n}_t^{(s)}, \tag{28a}
\end{aligned}$$

where $\mathbf{k}_t^{(s)}$ and $\mathbf{n}_t^{(s)}$ are drawn from the proposal distribution v_t . [6] sets it to

$$v_t(\mathbf{k}^{(s)}, \mathbf{n}^{(s)}) \triangleq \rho_t(\mathbf{k}^{(s)}) p_n^{(i)}(\mathbf{n}). \tag{28b}$$

(28a) can then be rewritten to examine the importance weight (the fraction ρ_t/v_t in (28a)):

$$\begin{aligned}
\boldsymbol{\mu}_n^{(i+1)} &:= \frac{1}{T} \sum_t \frac{1}{S} \sum_s \frac{\rho_t(\mathbf{k}_t^{(s)}) \rho_t(\mathbf{n}_t^{(s)} | \mathbf{k}_t^{(s)})}{\rho_t(\mathbf{k}_t^{(s)}) p_n^{(i)}(\mathbf{n}_t^{(s)})} \mathbf{n}_t^{(s)} \\
&= \frac{1}{T} \sum_t \frac{1}{S} \sum_s \frac{\rho_t(\mathbf{n}_t^{(s)} | \mathbf{k}_t^{(s)})}{p_n^{(i)}(\mathbf{n}_t^{(s)})} \mathbf{n}_t^{(s)}. \tag{28c}
\end{aligned}$$

The variance of the importance weights determines how many samples are required to find a good approximation.³ (28c) suggests that in the first iteration of noise estimation, where there is most discrepancy between the prior and posterior noise, the estimate will be least accurate. It also suggests that there are two factors that explain why [6] found this scheme to be practical, and those form a problem for extending this to model compensation in the cepstral domain. First: [6] applies this per dimension, whereas the probability of finding a weight that is too low increases exponentially with the dimensionality. For the cepstral domain, all dimensions will need to be considered. Second: the number of speech components [6] uses is small, because it is modelled with a mixture of Gaussians. This means that the speech variance for one component is relatively large, and that the posterior of \mathbf{n}_t is relatively wide. For one component from the back-end of a speech recogniser, which has a smaller variance, the noise vector that explains the observation has a smaller variance too. For these two reasons, it can be assumed that applying this scheme directly to model compensation and the cepstral domain is not feasible.

³ To compute the expression in (28c), note that since the distribution ρ_t can be rewritten (in (27a)) to include $p_n^{(i)}(\mathbf{n}_t)$:

$$\begin{aligned}
\boldsymbol{\mu}_n^{(i+1)} &:= \frac{1}{T} \sum_t \frac{1}{S} \sum_s \frac{p(\mathbf{y}_t | \mathbf{k}_t^{(s)}, \mathbf{n}_t^{(s)})}{p(\mathbf{y}_t | \mathbf{n}_t^{(s)})} \mathbf{n}_t^{(s)} \\
&\simeq \frac{1}{T} \sum_t \frac{1}{Z} \sum_s p(\mathbf{y}_t | \mathbf{k}_t^{(s)}, \mathbf{n}_t^{(s)}) \mathbf{n}_t^{(s)}, \tag{29}
\end{aligned}$$

where the $\mathbf{n}_t^{(s)}$ are multiplied by unnormalised weights, and Z can be approximated as the sum of those.

5.2 Refinements to the Monte Carlo scheme

The problems for estimating the noise for model compensation in the cepstral domain with [6] are similar to the problems with importance sampling in [28]. It therefore seems obvious that the same solution could be applied, which is to transform the space and to apply sequential importance re-sampling. However, this is probably not the best approach. The first problem is that as sequential importance re-sampling steps through the dimensions, the first dimensions become sparser, because re-sampling duplicates samples. For [28], where a multi-dimensional integral was sought, this did not matter: the normalisation constant for each dimension was approximated with dense, recently-drawn, samples, and the order in which dimensions were drawn was chosen to minimise the effect. Here, however, the first dimensions of the final samples determine the first dimensions of $\mu_n^{(i+1)}$.

However, other options are possible. The estimate of the noise mean in (28a) is

$$\mu_n^{(i+1)} := \frac{1}{T} \sum_t \sum_{k_t} \int \rho_t(k_t, \mathbf{n}_t) \mathbf{n}_t d\mathbf{n}_t. \quad (30)$$

This is essentially a weighted average over all values for (t, k_t, \mathbf{n}_t) of \mathbf{n}_t . It would be possible to sample from all three variables at once. Many Monte Carlo methods may be appropriate. In particular, the Metropolis–Hastings algorithm, which is a Markov chain Monte Carlo method, might be able to approximate the sum well. Markov chain Monte Carlo methods derive each new sample from the previous. This allows the algorithm to remain in high-probability regions. Compared with importance sampling, Metropolis–Hastings with a correctly-specified proposal distribution could obtain an advantage from sampling from t as well. The advantage is that information about the inferred noise at one time step would inform the samples for the next time step.

This paper will not apply this scheme for noise estimation in the experiments. However, it may be a viable alternative to noise estimation using the vector Taylor series approximation. The experiments will therefore aim to be invariant to the noise estimation algorithm, and estimate a noise model directly from the known noise.

6 Experiments

The variational predictive methods described in this paper were tested on the 1000-word-vocabulary Resource Management corpus [23] with Operations Room noise from the NOISEX-92 database [31] added at 20 and 14 dB. This task contains 109 training speakers reading 3990 sentences, a total of 3.8 hours of data. All results are averaged over three of the four available test sets, February 89, October 89, and February 91 (September 1992 is not used), a total of 30 test speakers and 900 utterances.

State-clustered triphone models with six components per mixture were built using the HTK RM recipe [33]. The number of components is about 9500. The extended speech statistics are striped as in [30]. The language model for recognition is a word-pair grammar. Since the additive background noise is known, the true, full-covariance, noise model is extracted.

Table 1 contains word error rates for model compensation. “vts” is a standard scheme that linearises the mismatch function. It makes an additional approximation

6. EXPERIMENTS

Method	Optimisation	Shape	20 dB	14 dB
vts	per component	diag	8.6	17.4
eDPMC	per component	diag	7.5	14.9
		full	7.4	13.3
	variational	full	6.9	12.0

Table 1: Word error rates for model compensation.

Statistics	Base classes			
	16		1024	
	20 dB	1024	16	14 dB
vts	10.9	10.0	23.8	20.9
eDPMC	9.2	8.1	19.3	16.4
Variational	8.7	7.9	19.6	15.1

Table 2: word error rates for predictive CMLLR.

that makes off-diagonal elements of the covariance matrix unreliable [30], so the covariance matrices are (as is usual) diagonalised.

Extended DPMC (eDPMC), discussed in section 3.1, estimates means and covariances for each component separately using Monte Carlo. As the number of samples goes to infinity (at 100 000 samples, performance has converged) it yields the optimal Gaussian-for-Gaussian compensation. That it outperforms vts when diagonalised speaks for the improvement that can be obtained by removing the linearisation of the mismatch function. Removing the diagonalisation for eDPMC is especially useful at lower signal-to-noise ratios, when the noise affects feature correlations most. At 14 dB, it yields another 15 % relative increase in performance.

However, this still assumes that one clean speech Gaussian generates Gaussian-distributed corrupted speech. The bottom row of Table 1 shows the effect of removing the Gaussian assumption. Variational eDPMC uses the variational approach from section 4 to allow corrupted-speech samples to be re-assigned to different components than they were drawn from. By moving probability mass to different components, this allows the state-conditional mixture to model the non-linear effects of the interaction of the speech and noise. This yields a 10 % relative reduction in word error rate compared to the optimal Gaussian-for-Gaussian compensation.

Table 2 shows the same contrasts, but on predictive CMLLR. As discussed in section 3.2, non-variational PCMLLR can be trained from different forms of statistics. The word error rates in the first row use statistics from a vts-compensated model. The double approximation (the linearisation of the mismatch function, and PCMLLR) leads to reduced performance. The numbers in the second row are from a form of CMLLR trained from full-covariance eDPMC statistics, but with the samples chosen more sparsely. 10 000 samples are drawn per base class. For the 16 base class system, the word error rate is the same with 5000 samples per base class, which makes the total number of samples 80 000. The complexity of this is invariant to the number of components, so for larger systems this operating point could yield a reasonable trade-off. At 1024 base classes performance does improve over vts performance. Though the full transformation matrix of PCMLLR implicitly performs some compensation for correlation changes, it does not model the non-Gaussian distributions.

The bottom row in Table 2 shows results for variational PCMLLR, which applies a variational approach to estimating linear transformations per base class. It is not a direct approximation of variational DPMC, since the optimal variational parameters at each iteration are different. It allows the transformations to move components in space to model the non-Gaussian distribution of the corrupted speech. At 20 dB, around 26 % of the probability mass of the samples goes to different components; at 14 dB, 37 %. At 16 base classes this does not consistently yield benefits, but with 128 or more, and especially at lower signal-to-noise ratios, word error rates improve compared to non-variational PCMLLR. That it is possible to model the non-Gaussian aspect of the corrupted-speech distribution with just linear transformations of clean speech Gaussian shows the power of the variational approach.

7 Conclusion

This paper has viewed model compensation for noise-robustness and predictive linear transformations from a variational perspective. These methods can be seen as minimising an upper bound on the divergence between the corrupted speech and the model for decoding. It is possible to find a tighter bound by considering the divergence between states rather than between components. When applied to DPMC and predictive CMLLR, this yields reductions in the word error rate. Even just linear transformations of Gaussians can model the non-linear impact of the noise. The variational predictive framework should enable a wide range of schemes, and allow for approaches to deal with the computational complexity.

Acknowledgment

The authors like to thank David Barber for suggesting the Monte Carlo approximation to predictive methods. This work was supported by EPSRC Project EP/1006583/1 (Generative Kernels and Score Spaces for Classification of Speech) within the Global Uncertainties Programme.

Bibliography

- [1] Alex Acero. *Acoustical and Environmental Robustness in Automatic Speech Recognition*. PhD thesis, Carnegie Mellon University, 1990.
- [2] Alex Acero, Li Deng, Trausti Kristjansson, and Jerry Zhang. HMM adaptation using vector Taylor series for noisy speech recognition. In *Proceedings of ICSLP*, volume 3, pages 229–232, 2000.
- [3] C. Breslin, K. K. Chin, M. J. F. Gales, K. Knill, and H. Xu. Prior information for rapid speaker adaptation. In *Proceedings of Interspeech*, pages 1644–1647, 2010.
- [4] Li Deng, Jasha Droppo, and Alex 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*, 12(2):133–143, 2004.
- [5] A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: fifteen years later. Technical report, Department of Statistics, University of British Columbia, December 2008.
- [6] Friedrich Faubel and Dietrich Klakow. Estimating noise from noisy speech features with a monte carlo variant of the expectation maximization algorithm. In *Proceedings of Interspeech*, pages 2046–2049, 2010.
- [7] F. Flego and M. J. F. Gales. Factor analysis based VTS and JUD noise estimation and compensation. Technical Report CUED/F-INFENG/TR.653, Cambridge University, 2011.
- [8] Brendan J. Frey, Trausti T. Kristjansson, Li Deng, and Alex Acero. ALGONQUIN: Learning dynamic noise models from noisy speech for robust speech recognition. In *Proceedings of NIPS*, 2001.
- [9] M. J. F. Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech and Language*, 12(2):75–98, 1998.
- [10] M. J. F. Gales and R. C. van Dalen. Predictive linear transforms for noise robust speech recognition. In *Proceedings of ASRU*, pages 59–64, Dec 2007.
- [11] Mark J. F. Gales. *Model-Based Techniques for Noise Robust Speech Recognition*. PhD thesis, Cambridge University, 1995.
- [12] John R. Hershey and Peder A. Olsen. Approximating the Kullback Leibler divergence between gaussian mixture models. In *Proceedings of ICASSP*, 2007.
- [13] Yu Hu and Qiang Huo. Irrelevant variability normalization based HMM training using VTS approximation of an explicit model of environmental distortions. In *Proceedings of Interspeech*, pages 1042–1045, 2007.
- [14] Do Yeong Kim, Chong Kwan Un, and Nam Soo Kim. Speech recognition in noisy environments using first-order vector Taylor series. *Speech Communication*, 24:39–49, 1998.
- [15] G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian non-linear state space models. *Journal of Computational and Graphical Statistics*, 5:1–25, 1996.

-
- [16] Trausti Thor Kristjansson. *Speech Recognition in Adverse Environments: a Probabilistic Approach*. PhD thesis, University of Waterloo, 2002.
- [17] Volker Leutnant and Reinhold Haeb-Umbach. An analytic derivation of a phase-sensitive observation model for noise robust speech recognition. In *Proceedings of Interspeech*, pages 2395–2398, 2009.
- [18] Jinyu Li, Li Deng, Dong Yu, Yifan Gong, and Alex Acero. High-performance HMM adaptation with joint compensation of additive and convolutive distortions via vector Taylor series. In *Proceedings of ASRU*, pages 65–70, 2007.
- [19] Jinyu Li, Dong Yu, Yifan Gong, and L. Deng. Unscented transform with online distortion estimation for HMM adaptation. In *Proceedings of Interspeech*, pages 1660–1663, 2010.
- [20] H. Liao and M. J. F. Gales. Joint uncertainty decoding for robust large vocabulary speech recognition. Technical Report CUED/F-INFENG/TR.552, Cambridge University Engineering Department, November 2006.
- [21] Pedro J. Moreno. *Speech Recognition in Noisy Environments*. PhD thesis, Carnegie Mellon University, 1996.
- [22] Tor André Myrvoll and Satoshi Nakamura. Optimal filtering of noisy cepstral [sic] coefficients for robust ASR. In *Proceedings of ASRU*, pages 381–386, 2003.
- [23] P. Price, W. M. Fisher, J. Bernstein, and D. S. Pallett. The DARPA 1000-word Resource Management database for continuous speech recognition. In *Proceedings of ICASSP*, volume 1, pages 651–654, 1988.
- [24] Mike Seltzer, Alex Acero, and Kaustubh Kalgaonkar. Acoustic model adaptation via linear spline interpolation for robust speech recognition. In *Proceedings of ICASSP*, pages 4550–4553, 2010.
- [25] R. C. van Dalen, F. Flego, and M. J. F. Gales. Transforming features to compensate speech recogniser models for noise. In *Proceedings of Interspeech*, pages 2499–2502, Sep 2009.
- [26] R. C. van Dalen and M. J. F. Gales. Covariance modelling for noise robust speech recognition. In *Proceedings of Interspeech*, pages 2000–2003, Sep 2008.
- [27] R. C. van Dalen and M. J. F. Gales. Asymptotically exact noise-corrupted speech likelihoods. In *Proceedings of Interspeech*, pages 709–712, Sep 2010.
- [28] R. C. van Dalen and M. J. F. Gales. A theoretical bound for noise-robust speech recognition. Technical Report CUED/F-INFENG/TR.648, Cambridge University Engineering Department, Apr 2010.
- [29] Rogier C. van Dalen. *Statistical Models for Noise-Robust Speech Recognition*. PhD thesis, University of Cambridge, Jan 2011.
- [30] Rogier C. van Dalen and Mark J. F. Gales. Extended VTS for noise-robust speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):733–743, Apr 2011.

BIBLIOGRAPHY

- [31] A. Varga and H. J. M. Steeneken. Assessment for automatic speech recognition II: NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Speech Communication*, 12(3):247–251, 1993.
- [32] Haitian Xu, M. J. F. Gales, and K. K. Chin. Joint uncertainty decoding with predictive methods for noise robust speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(6):1665–1676, 2011.
- [33] Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying (Andrew) Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, Valtcho Valtchev, and Phil Woodland. The HTK book (for HTK version 3.4), 2006.