A NOVEL STATIC PARAMETER CALCULATION METHOD FOR MODEL COMPENSATION

Suliang Bu[1] *Yunxin Zhao*[1] *Yanmin Qian*[2] *Kai Yu*[2]

[1] Department of Computer Science, University of Missouri-Columbia, USA
[2] Department of Computer Science and Engineering, Shanghai Jiao Tong University
sbkc6@mail.missouri.edu, ZhaoY@missouri.edu, {yanminqian,kai.yu}@sjtu.edu.cn

ABSTRACT

Vector Taylor Series (VTS) based model compensation approach has been successfully applied to various robust speech recognition tasks. In this paper, we propose a novel method of variable transformation to calculate the static statistics. In addition, we provide a detailed explanation of VTS and random variable transformations adopted in some recent papers. Experiments on Aurora 4 showed that the proposed approach obtained 22.8% relative WER reduction over the traditional first-order VTS methods.

Index Terms— robust speech recognition, model compensation approach, Vector Taylor Series

1. INTRODUCTION

It is known that the performance of an automatic speech recognition (ASR) system degrades greatly when noise is present and the system is trained with only clean speech. Two approaches are commonly used to deal with noise under the GMM-HMM framework. One is called feature enhancement, which aims to remove the noise effect in test utterances so that the processed data could better match the clean models. The other is called model compensation, which adapts the original models to the noisy conditions in test utterances.

In both approaches, the first-order vector Taylor series (fVTS) based methods [1, 2, 3] are widely used because they are simple and effectiveness. However, relatively large residual errors might be caused by such a simple approximation. For the second-order VTS (sVTS) or even higher orders [4], the resulting formulae would be very complicated if the conventioanl way is followed. In fact, high-order Taylor series of a function with more than one variable would become complex, let alone the fact that here the number of variables is way more than one. To solve this difficulty, recently [5] proposed a transformation of random variables, which could greatly reduce the complexity of sVTS. Though the method is quite effective, further experiments suggested that this method might reach a bottleneck. Here we adopt a new way for variable transformation to improve the performance. To help readers gain a better understanding of VTS and variable transformations adopted here and in [5, 6], we also provide a detailed explanation on these issues in this paper.

This paper is organized as follows. Section 2 describes the formulation for static and dynamic statistics. A detailed explanation of VTS and the random variable transformations involved is provided in section 3. Experiment results on aurora 4 are analyzed in section 4, and finally we conclude in section 5.

2. MODEL COMPENSATION APPROACH

In this study, only the effect of additive noise is considered, and the channel distortion is ignored.

2.1. Static Statistics

2.1.1. A Novel Formula for Calculating Static Statistics

In the static cepstral domain, the nonlinear effect of additive noise can be expressed as:

$$\mathbf{y_s} = \mathbf{x_s} + \mathbf{C} \cdot \ln\left(\mathbf{1} + \mathrm{e}^{\mathbf{C^{-1}}(\mathbf{n_s}-\mathbf{x_s})}\right) \tag{1}$$

where $\mathbf{y_s}$, $\mathbf{x_s}$ and $\mathbf{n_s}$ are static features corresponding to noisy speech, clean speech and additive noise, respectively; $\mathbf{C}$ is the truncated discrete cosine transform (DCT) matrix and $\mathbf{C^{-1}}$ is its pseudo inverse. The subscript "s" indicates static parameters. Here $\mathbf{x_s}$ and $\mathbf{n_s}$ are assumed to have independent Gaussian distributions with mean $\boldsymbol{\mu_{xs}}$, $\boldsymbol{\mu_{ns}}$ and covariance $\mathbf{\Sigma_{xs}}$, $\mathbf{\Sigma_{ns}}$, respectively.

Next, we use $\mathbf{z_s}$ to denote $\mathbf{C^{-1}}(\mathbf{n_s} - \mathbf{x_s})$, since 1) each component of $\mathbf{x_s}$ and $\mathbf{n_s}$ is Gaussian distributed, 2) all these vector components are assumed to be mutually independent and 3) $\mathbf{C^{-1}}$ is a linear transformation, and so $\mathbf{z_s}$ is also Gaussian distributed. Please note that components in $\mathbf{z_s}$ are not mutually independent. The mean and covariance for $\mathbf{z_s}$ are given by

$$\boldsymbol{\mu_{zs}} = \mathbf{C^{-1}}(\boldsymbol{\mu_{ns}} - \boldsymbol{\mu_{xs}}) \tag{2}$$

$$\mathbf{\Sigma_{zs}} = \mathbf{C^{-1}}(\mathbf{\Sigma_{ns}} + \mathbf{\Sigma_{xs}})(\mathbf{C^{-1}})^{\mathbf{T}} \tag{3}$$

Further, we view $e^{\mathbf{z_s}}$ as a new random variable $\mathbf{w}$, whose mean and covariance are given by

$$\boldsymbol{\mu_w} = exp\left(\boldsymbol{\mu_{zs}} + diag^{-1}(\mathbf{\Sigma_{zs}})/2\right) \tag{4}$$

$$\mathbf{\Sigma_w} = \left[\boldsymbol{\mu_w}\left(\boldsymbol{\mu_w}\right)^T\right] \odot \left[exp\left(\mathbf{\Sigma_{zs}}\right) - \mathbf{1}\right] \tag{5}$$

where $diag^{-1}(\cdot)$ extracts the diagonal elements of a matrix as a column vector and $\odot$ denotes element-wise multiplication. Now taking expectation of the noisy speech, we have:

$$E\left[\mathbf{y_s}\right] = E\left[\mathbf{x_s}\right] + \mathbf{C} \cdot E\left[\ln\left(\mathbf{1} + \mathbf{w}\right)\right] \tag{6}$$

The first-order VTS expansion of $\ln(\mathbf{1} + \mathbf{w})$ around $\boldsymbol{\mu_w}$ is

$$\ln(\mathbf{1} + \mathbf{w}) \approx \mathbf{f}^{(0)} + \mathbf{f}^{(1)} \odot (\mathbf{w} - \boldsymbol{\mu}_{\mathrm{w}}) \tag{7}$$

where $\mathbf{f}^{(0)} = \ln(\mathbf{1} + \boldsymbol{\mu_w})$ and $\mathbf{f}^{(1)} = \frac{1}{\mathbf{1}+\boldsymbol{\mu_w}}$. By computing the expectation of Eq. (6), we have

$$\boldsymbol{\mu_{ys}} \approx \boldsymbol{\mu_{xs}} + \mathbf{C} \cdot \mathbf{f}^{(0)} \tag{8}$$

As for the static covariance, it can be calculated by

$$\begin{aligned}\boldsymbol{\Sigma}_{\mathbf{ys}} &\approx E(\mathbf{y_s} - \boldsymbol{\mu_{ys}})(\mathbf{y_s} - \boldsymbol{\mu_{ys}})^T \\ &= \boldsymbol{\Sigma}_{\mathbf{xs}} - \mathbf{K_2} - \mathbf{K_2}^T + \mathbf{C} \cdot [\boldsymbol{\Sigma}_{\mathbf{w}} \odot \mathbf{F_1}] \cdot \mathbf{C}^T\end{aligned} \tag{9}$$

where

$$\mathbf{K_1} = \left[diag^{-1}(\boldsymbol{\Sigma}_{\mathbf{xs}}) \cdot \boldsymbol{\mu}_{\boldsymbol{w}}^T\right] \odot (\mathbf{C^{-1}})^T \tag{10}$$

$$\mathbf{K_2} = \mathbf{K_1} \cdot diag(\mathbf{f}^{(1)}) \cdot \mathbf{C}^T \tag{11}$$

$$\mathbf{F_1} = \mathbf{f}^{(1)} \left(\mathbf{f}^{(1)}\right)^T \tag{12}$$

and $diag(\cdot)$ gets a diagonal matrix from a column vector.

2.2. Dynamic Statistics

2.2.1. Theoretical Analysis

We use the subscript "Δ" and "$\Delta\Delta$" to denote delta and delta delta statistics, respectively. To compute dynamic statistics, continuous time (CT) approximation [7] was used. Different from previous methods [1, 2, 8], here we directly take derivative of $\mathbf{y_s}$ with respect to time to derive dynamic mismatch functions without any further approximation:

$$\frac{\partial \mathbf{y_s}}{\partial t} = \frac{\partial \mathbf{x_s}}{\partial t} + \mathbf{C} \cdot \left(\frac{\mathbf{e}^{\mathbf{z_s}}}{\mathbf{1} + \mathbf{e}^{\mathbf{z_s}}} \odot \frac{\partial \mathbf{z_s}}{\partial \mathbf{t}}\right) \tag{13}$$

For the sake of convenience, let's first define some notations:

$$\mathbf{L_1} = E\left[\frac{\mathbf{e}^{\mathbf{z_s}}}{\mathbf{1} + \mathbf{e}^{\mathbf{z_s}}}\right] = E\left[\frac{\mathbf{1}}{\mathbf{1} + \mathbf{e}^{-\mathbf{z_s}}}\right] \tag{14}$$

$$\mathbf{L_2} = E\left[\frac{\mathbf{e}^{\mathbf{z_s}}}{(\mathbf{1} + \mathbf{e}^{\mathbf{z_s}})^{\mathbf{2}}}\right] = E\left[\frac{\mathbf{1}}{\mathbf{e}^{\mathbf{z_s}} + \mathbf{2} + \mathbf{e}^{-\mathbf{z_s}}}\right] \tag{15}$$

$$\mathbf{L_{11}} = E\left[\frac{\mathbf{e}^{\mathbf{z_s}}}{\mathbf{1} + \mathbf{e}^{\mathbf{z_s}}}\right]\left[\frac{\mathbf{e}^{\mathbf{z_s}}}{\mathbf{1} + \mathbf{e}^{\mathbf{z_s}}}\right]^T \tag{16}$$

$$\mathbf{L_{22}} = E\left[\frac{\mathbf{e}^{\mathbf{z_s}}}{(\mathbf{1} + \mathbf{e}^{\mathbf{z_s}})^{\mathbf{2}}}\right]\left[\frac{\mathbf{e}^{\mathbf{z_s}}}{(\mathbf{1} + \mathbf{e}^{\mathbf{z_s}})^{\mathbf{2}}}\right]^T \tag{17}$$

$$\mathbf{L_{21}} = E\left[\frac{\mathbf{e}^{\mathbf{z_s}}}{(\mathbf{1} + \mathbf{e}^{\mathbf{z_s}})^{\mathbf{2}}}\right]\left[\frac{\mathbf{e}^{\mathbf{z_s}}}{\mathbf{1} + \mathbf{e}^{\mathbf{z_s}}}\right]^T \tag{18}$$

Assume $\frac{\partial \mathbf{n_s}}{\partial t}$ and $\mathbf{n_s}$ are independent (similarly for speech). Since $\mathbf{C^{-1}}$ is a linear transformation, it is easy to prove $\frac{\partial \mathbf{z_s}}{\partial t}$ and $\mathbf{z_s}$ are independent. Taking expectation of Eq.(13) gives:

$$\begin{aligned}\boldsymbol{\mu_{y\Delta}} &= \boldsymbol{\mu_{x\Delta}} + \mathbf{C} \cdot \left(E\left[\frac{\mathbf{e}^{\mathbf{z_s}}}{\mathbf{1} + \mathbf{e}^{\mathbf{z_s}}}\right] \odot E\left[\frac{\partial \mathbf{z_s}}{\partial t}\right]\right) \\ &= \boldsymbol{\mu_{x\Delta}} + \mathbf{C} \cdot (\mathbf{L_1} \odot \boldsymbol{\mu_{z\Delta}})\end{aligned} \tag{19}$$

As for the delta covariance, it can be computed by:

$$\begin{aligned}\boldsymbol{\Sigma}_{\mathbf{y\Delta}} &= \boldsymbol{\Sigma}_{\mathbf{x\Delta}} - \mathbf{K_3} - \mathbf{K_3}^T + \mathbf{C} \cdot [\boldsymbol{\Sigma}_{\mathbf{z\Delta}} \odot \mathbf{L_{11}}] \cdot \mathbf{C}^T \\ &+ \mathbf{C} \cdot \left[\left(\mathbf{L_{11}} - \mathbf{L_1}\left(\mathbf{L_1}\right)^T\right) \odot \left(\boldsymbol{\mu_{z\Delta}}(\boldsymbol{\mu_{z\Delta}})^T\right)\right] \cdot \mathbf{C}^T\end{aligned} \tag{20}$$

where $\boldsymbol{\mu_{z\Delta}}$, $\boldsymbol{\Sigma_{z\Delta}}$ can be computed like Eq. (2), (3), and

$$\mathbf{K_3} = \boldsymbol{\Sigma}_{\mathbf{x\Delta}}(\mathbf{C^{-1}})^{\mathbf{T}} \cdot diag\left(\mathbf{L_1}\right) \cdot \mathbf{C}^T \tag{21}$$

Similarly, the delta delta statistics can be computed by:

$$\boldsymbol{\mu}_{\mathbf{y\Delta\Delta}} = \boldsymbol{\mu}_{\mathbf{x\Delta\Delta}} + \mathbf{C} \cdot (\mathbf{L_1} \odot \boldsymbol{\mu}_{\mathbf{z\Delta\Delta}} + \mathbf{L_2} \odot \mathbf{K_4}) \tag{22}$$

$$\begin{aligned}\boldsymbol{\Sigma}_{\mathbf{y\Delta\Delta}} =& \boldsymbol{\Sigma}_{\mathbf{x\Delta\Delta}} - \mathbf{K_5} - \mathbf{K_5}^T + \mathbf{C}\left(\mathbf{K_6} + \mathbf{K_7} + \mathbf{K_7}^T\right)\mathbf{C}^T \\ &+ \mathbf{C}\left[\boldsymbol{\Sigma}_{\mathbf{z\Delta\Delta}} \odot \mathbf{L_{11}} + \mathbf{K_8}\right]\mathbf{C}^T\end{aligned} \tag{23}$$

where

$$\mathbf{K_4} = diag^{-1}(\boldsymbol{\Sigma}_{\mathbf{z\Delta}}) + \boldsymbol{\mu}_{\mathbf{z\Delta}} \odot \boldsymbol{\mu}_{\mathbf{z\Delta}} \tag{24}$$

$$\mathbf{K_5} = \boldsymbol{\Sigma}_{\mathbf{x\Delta\Delta}}\left(\mathbf{C^{-1}}\right)^{\mathbf{T}} \cdot diag\left(\mathbf{L_1}\right) \cdot \mathbf{C}^T \tag{25}$$

$$\begin{aligned}\mathbf{K_6} = \boldsymbol{\Sigma}_{\mathbf{z\Delta}} \odot &\left(2\boldsymbol{\Sigma}_{\mathbf{z\Delta}} + 4\boldsymbol{\mu}_{\mathbf{z\Delta}}(\boldsymbol{\mu}_{\mathbf{z\Delta}})^T\right) \odot \mathbf{L_{22}} \\ &+ \left[\mathbf{L_{22}} - \mathbf{L_2}(\mathbf{L_2})^T\right] \odot \left[\mathbf{K_4}(\mathbf{K_4})^T\right]\end{aligned} \tag{26}$$

$$\mathbf{K_7} = \left[\mathbf{L_{21}} - \mathbf{L_2}(\mathbf{L_1})^T\right] \odot \left[\mathbf{K_4}(\boldsymbol{\mu}_{\mathbf{z\Delta\Delta}})^T\right] \tag{27}$$

$$\mathbf{K_8} = \left[\mathbf{L_{11}} - \mathbf{L_1}(\mathbf{L_1})^T\right] \odot \left[\boldsymbol{\mu}_{\mathbf{z\Delta\Delta}}(\boldsymbol{\mu}_{\mathbf{z\Delta\Delta}})^T\right] \tag{28}$$

2.2.2. Formulae for Calculating Dynamic Statistics

To calculate the dynamic statistics, we need to know the values of $\mathbf{L_1}$, $\mathbf{L_2}$, $\mathbf{L_{11}}$, $\mathbf{L_{21}}$, $\mathbf{L_{22}}$. In order to get a compact formula, we make the same approximation as in [6]:

$$\mathbf{L_1} \approx \frac{\mathbf{1}}{\mathbf{E}\left[\mathbf{1} + \mathbf{e}^{-\mathbf{z_s}}\right]} \quad \mathbf{L_2} \approx \frac{\mathbf{1}}{\mathbf{E}\left[\mathbf{e}^{\mathbf{z_s}} + \mathbf{2} + \mathbf{e}^{-\mathbf{z_s}}\right]} \tag{29}$$

$$\mathbf{L_{11}} \approx \mathbf{L_1}\left(\mathbf{L_1}\right)^T \ \mathbf{L_{22}} \approx \mathbf{L_2}\left(\mathbf{L_2}\right)^T \ \mathbf{L_{21}} \approx \mathbf{L_2}\left(\mathbf{L_1}\right)^T \tag{30}$$

For a Gaussian variable x with mean μ and variance σ^2, $E\left[e^x\right] = exp(\mu + \sigma^2/2)$, which makes it easy to derive the formulae for $\mathbf{L_1}$ and $\mathbf{L_2}$. With all these above, the formulae for calculating the dynamic statistics become available:

$$\boldsymbol{\Sigma}_{\mathbf{y\Delta}} \approx \boldsymbol{\Sigma}_{\mathbf{x\Delta}} - \mathbf{K_3} - \mathbf{K_3}^T + \mathbf{C}\left[\boldsymbol{\Sigma}_{\mathbf{z\Delta}} \odot \mathbf{L_{11}}\right]\mathbf{C}^T \tag{31}$$

$$\begin{aligned}\boldsymbol{\Sigma}_{\mathbf{y\Delta\Delta}} \approx& \boldsymbol{\Sigma}_{\mathbf{x\Delta\Delta}} - \mathbf{K_5} - \mathbf{K_5}^T + \mathbf{C}\left[\boldsymbol{\Sigma}_{\mathbf{z\Delta\Delta}} \odot \mathbf{L_{11}}\right]\mathbf{C}^T \\ &+ \mathbf{C}\left[\boldsymbol{\Sigma}_{\mathbf{z\Delta}} \odot \left(2\boldsymbol{\Sigma}_{\mathbf{z\Delta}} + 4\boldsymbol{\mu}_{\mathbf{z\Delta}}(\boldsymbol{\mu}_{\mathbf{z\Delta}})^T\right) \odot \mathbf{L_{22}}\right]\mathbf{C}^T\end{aligned} \tag{32}$$

To do model compensation, noise parameters are needed. In this paper, noise is modeled by a single Gaussian. Usually, the noise parameters are iteratively estimated using EM-like

algorithms. However, in this study noise parameters are estimated by using the first and the last 20 frames of each test utterance [3]. When a high-order VTS is used, it is not easy to directly derive the formula for re-estimation, even for the mean. We plan to provide a noise re-estimation algorithm for high order VTS as well as the new method of this paper in a future work.

In practice, when noise is relatively stationary, $\boldsymbol{\mu_{n\Delta}}$ and $\boldsymbol{\mu_{n\Delta\Delta}}$ are set to be zero, and covariance matrices are usually diagonalized for computational convenience.

3. FURTHER UNDERSTANDING OF VTS AND VARIABLE TRANSFORMATIONS

Here we discuss several issues of VTS and the random variable transformations involved in our method to help readers better understand both VTS and our method.

3.1 Variable transformations in [5, 6] and this paper

There are following advantages using the transformation $\mathbf{z_s} = \mathbf{C^{-1}}(\mathbf{n_s} - \mathbf{x_s})$:

1) Avoid huge memory overhead and complicated formulae when sVTS (or even higher order) is used

Traditionally, we expand function (1) around two vector variables. In this way, we need a Hessian matrix for each component function if sVTS is used to calculate static statistics. Since static feature is often 13 in dimensionality, the size of each Hessian matrix would be (13+13)*(13+13). If the DCT matrix is 13*d by size, then we need d*26*26 storage space for the whole vector function. But if the transformation $\mathbf{z_s} = \mathbf{C^{-1}}(\mathbf{n_s} - \mathbf{x_s})$ is used, a mere "d" sized vector is enough, which is only 1/(26*26) of the previous space usage. Besides, with some tricks, the resulting formulae become rather compact.

Please note that the major difficulty and computation reduction actually do not happen in static part, but in dynamic part. If we derive dynamic formulae using CT approximation in the traditional way, the resulting formulae would be very complex, which would also cause troubles in implementation.

2) Extend easily to higher order VTS

Using the traditional method, it is very difficult to derive the analytic formulae for both static and dynamic parts based on the third order VTS (tVTS). To get the third order accuracy for the static part, previously unscented transform (UT) [9, 10, 11] was used. In fact, UT will introduce some additional terms, thus it does not return the same value as tVTS. Besides, traditional tVTS dynamic formulae are too complex to derive. In contrast, this transformation $\mathbf{z_s} = \mathbf{C^{-1}}(\mathbf{n_s} - \mathbf{x_s})$ would make it much easier to derive both static and dynamic formulae for tVTS, as well as for other order VTS (though not provided in this paper due to our space constraint).

3) Provide a way to observe covariance structure

The covariance of $\mathbf{z_s}$ is calculated by Eq.(3). Considering that the pseudo inverse on C is not accurate, we might want to know how this covariance differs from the one if real inverse is used. In [6] we have an interesting finding about the difference, which could be used to largely improve performance.

3.2 Why sVTS is better than fVTS

The reason is usually attributed to the reduced residual error when sVTS is used. Strictly speaking, it is not entirely true. For example, let's calculate the expectation $E[ln(1+x)]$, where "x" is a positive random variable with mean 1 and variance 8. Since "x" is positive, the expectation should be positive. But if we use sVTS, we would have $E\left[ln\left(1+x\right)\right] = ln\left(1+\mu\right) - \frac{1}{2}\frac{\sigma^2}{(1+\mu)^2} = ln2 - 1 < 0$. Thus the result would be worse than fVTS : $ln(1+\mu)$. That a Taylor series does not converge to its original function is sometimes used to explain this kind of situation. But even if the series converges, higher order VTS could still be worse than lower order unless sufficiently many terms are included. Therefore sVTS does not always get better results than fVTS.

Then why sVTS is better than fVTS in model compensation as shown in [5]? Strictly speaking, the reason is that given a certain dataset where noise is relatively stationary and SNR is relatively high, for most dimensions of most multivariate Gaussian components of GMM, sVTS returns more accurate value than fVTS does (In fact, as SNR tends to 0, the noisy speech tends to have two modes, which makes the assumption of model compensation — the noisy speech is still Gaussian distributed — rather problematic, hence both fVTS and sVTS would suffer. Non-stationary noise would cause other problems.)

3.3 Why not use a second order expansion in this paper?

If a second order expansion is used, then terms such as $E[\mathbf{x_s}(\mathbf{w^2})^\mathbf{T}]$ and $E[(\mathbf{w}-\boldsymbol{\mu_w})^\mathbf{2}((\mathbf{w}-\boldsymbol{\mu_w})^\mathbf{2})^\mathbf{T}]$ are needed, which are more difficult to derive. (We used second order in [5] because the Gaussian variable "$\mathbf{z_s}$" is much easier to handle even if a high order is used. But here "$\mathbf{w}$" is not Gaussian distributed.) Besides, second order is not necessarily better.

3.4 How about viewing $\mathbf{1} + \mathbf{w}$ as a new variable $\mathbf{v}$?

When $\mathbf{z_s}$ is used, mathematical convenience and better results are got, and here we propose using $e^{\mathbf{z_s}}$ as a variable $\mathbf{w}$. It is meaningless to push for $\mathbf{1} + \mathbf{w}$ because 1) it does not offer any new insights on some otherwise hidden structure, 2) we still need to expand $\mathbf{v}^n$ with the binomial formula $(\mathbf{1} + \mathbf{w})^n$ to compute terms like $E[\mathbf{x_s}(\mathbf{v^n})^\mathbf{T}]$. Such a transform therefore does not offer any advantage over the"$\mathbf{w}$" here.

3.5 VTS order beyond the second order is not preferred

Firstly, formulae of higher order VTS would be difficult to derive and check, and the resulting complex formulae will make implementation harder. (Overly complex formulae are not welcome since ASR is a practical problem, not totally mathematics. There could always be unseen factors unattended, making complex formulae not that accurate.) Secondly, as we said before, mathematically, a higher order is not necessarily better than a lower order. Even if sVTS makes an improvement over fVTS, it does not necessarily mean tVTS

(or higher) would be better than sVTS. Thirdly, more time will be needed on higher-order model compensations.

4. EXPERIMENTS AND RESULTS

4.1. Experiment Setup

To evaluate the proposed method, we conducted a series of experiments on Aurora 4, which was based on the Wall Street Journal 5k-vocabulary database. In this study, speech models were trained on clean data, which comprised 7138 utterances, and decision tree state clustering was used to get about 3000 tied triphone states. Since we only considers additive noise, the experiments were conducted on the test set B of Aurora 4 corpus, which was recorded using the same microphone as did in the training data. Therefore, channel distortion could be omitted. Six different noises at various SNRs were artificially added to turn original clean data into the noisy database. Each noise condition had 330 test utterances from 8 speakers, and only 16kHz testing data were used for evaluation. We used 12 MFCCs and C0 as well as the delta and delta-delta features. HTK [12] was used to build the system, in which bigram language model was adopted. Each speech state was represented by 16 Gaussian components while 32 Gaussian components were used for the silence state model. All experiment settings were the same for all noise conditions.

4.2. Experiment Results

In the following experiments, we first construct the GMM-HMM baseline and present the results in Table 1.

clean	car	babble	rest.	street	airport	train	avg
6.8	37.0	55.2	54.4	64.2	48.7	64.0	53.9

Table 1. WER (%) of the baseline on test B using clean model

We next compare the following four methods in Table 2: 1, 2) both static and dynamic statistics are computed using fVTS [2] or sVTS [5], respectively; 3) the method proposed in [6], denoted as "SIMP"; 4) the method proposed in this paper, denoted as "logN" since $\mathbf{w}$ is log-normal distributed.

	car	babble	rest.	street	airport	train	avg
fVTS	14.9	21.4	28.3	24.2	22.2	25.6	22.8
sVTS	11.5	18.4	23.9	20.7	18.6	21.4	19.1
SIMP	11.7	17.8	22.4	20.2	18.0	**20.9**	18.5
logN	**11.1**	**17.5**	**20.8**	**19.3**	**15.5**	21.1	**17.6**

Table 2. WER (%) of four methods on test B

Comparing fVTS with sVTS, we can see sVTS indeed improves the performance. The only difference between sVTS and SIMP lies in the calculation of dynamic statistics. Although both are based on CT approximation, the former's dynamic mismatch functions are derived with an additional sVTS approximation while the latter does not. The above results seem to prefer SIMP, which differs from logN in calculating the static part: SIMP uses the same way as sVTS, while logN adopts a new transformation. Since logN behaves the best, this new way of calculating the static part seems better.

Next, we consider the method of modifying $\mathbf{\Sigma_{zs}}$, which is introduced in section 3 of [6] to see how would the pseudo inverse of the unsquare DCT matrix influence the covariance $\mathbf{\Sigma_{zs}}$. After all, if a square DCT matrix is really used, there is no need to fuss about $\mathbf{\Sigma_{zs}}$. Relying on MATLAB simulations, we find that the diagonal elements of $\mathbf{\Sigma_{zs}}$ tend to be smaller than the ones obtained from using a full DCT inverse. To compensate for the smaller diagonal elements, a variable α [1] is introduced in this method. In the following experiment, α is assigned 0.1 to calculate new $\mathbf{\Sigma_{zs}}$.

	car	babble	rest.	street	airport	train	avg
fVTS	14.3	20.7	26.5	23.2	20.9	24.6	21.7
sVTS	11.4	18.2	23.2	20.5	17.5	21.8	18.8
SIMP	11.5	17.4	21.7	19.5	16.7	**20.1**	17.8
logN	**11.1**	**17.2**	**20.3**	**19.5**	**15.4**	21.7	**17.5**

Table 3. WER (%) of four methods on test B: $\alpha = 0.1$ was used in modifying $\mathbf{\Sigma_{zs}}$

Comparing Table 3 with Table 2, we can see that introducing α indeed improves the average performance for each of the four methods. However, for some conditions, performance might decrease. On the other hand, it seems that such modification does not work for logN: only a minor improvement is observed. In fact, if we set α to be 0.2, the performance of all the other three methods gets further improved whereas logN decreases. One possible reason is that the way logN calculates the static part somehow overlaps the effects of the modification on $\mathbf{\Sigma_{zs}}$. A further analysis is needed here.

5. CONCLUSION

We have introduced a novel method for random variable transformation to calculated the static statistics. We have given a detailed clarification on the random variable transformations used in this paper and [5, 6], which explains why such transformations are preferred. We have also explained why sVTS is better than fVTS, which might be easily misunderstood. In our experiment, we have unexpectedly found that the modification on $\mathbf{\Sigma_{zs}}$ does not seem to benefit the logN method, which requires a futher analysis. Finally, experiments on Aurora 4 have showed that our proposed method obtained a 22.8% relative WER reduction over the widely used first-order VTS approach.

[1] When α is considered, we need formulae which explicitly contain $\mathbf{\Sigma_{zs}}$ otherwise we can not use this method. The new fVTS formulae can be easily inferred from sections 2 and 3 in [5].

6. REFERENCES

[1] Pedro J Moreno, Bhiksha Raj, and Richard M Stern, "A vector taylor series approach for environment-independent speech recognition," in *ICASSP*, 1996, pp. 733–736.

[2] Alex Acero, Li Deng, Trausti T Kristjansson, and Jerry Zhang, "Hmm adaptation using vector taylor series for noisy speech recognition.," in *INTERSPEECH*, 2000, pp. 869–872.

[3] Jinyu Li, Li Deng, Dong Yu, Yifan Gong, and Alex Acero, "A unified framework of hmm adaptation with joint compensation of additive and convolutive distortions," *Computer Speech & Language*, pp. 389–405, 2009.

[4] Jun Du and Qiang 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*, pp. 2285–2293, 2011.

[5] Suliang Bu, Yanmin Qian, Khe Chai Sim, Yongbin You, and Kai Yu, "Second order vector taylor series based robust speech recognition," in *ICASSP*, 2014, pp. 1769–1773.

[6] Suliang Bu, Yanmin Qian, and Kai Yu, "A novel dynamic parameters calculation approach for model compensation," in *INTERSPEECH*, 2014.

[7] P. S. Gopalakrishnan S. Balakrishnan-Aiyer R. A. Gopinath, M. J. F. Gales and M. A. Picheny, "Robust speech recognition in noise - performance of the IBM continuous speech recognizer on the ARPA noise spoke task," in *ARPA Workshop on Spoken Language System Technology*, 1995, pp. 127–130.

[8] Ozlem Kalinli, Michael L Seltzer, and Alex Acero, "Noise adaptive training using a vector taylor series approach for noise robust automatic speech recognition," in *ICASSP*, 2009, pp. 3825–3828.

[9] Simon J Julier and Jeffrey K Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, pp. 401–422, 2004.

[10] Jinyu Li, Dong Yu, Yifan Gong, and Li Deng, "Unscented transform with online distortion estimation for hmm adaptation.," in *INTERSPEECH*, 2010, pp. 1660–1663.

[11] Yu Hu and Qiang Huo, "An hmm compensation approach using unscented transformation for noisy speech recognition," in *Chinese Spoken Language Processing*, pp. 346–357. 2006.

[12] Steve Young, Gunnar Evermann, Dan Kershaw, Gareth Moore, Julian Odell, Dave Ollason, Valtcho Valtchev, and Phil Woodland, "The htk book," 2002.