Product of Experts for Statistical Parametric Speech Synthesis

Heiga Zen, *Member, IEEE*, Mark J. F. Gales, *Fellow, IEEE*, Yoshihiko Nankaku, and Keiichi Tokuda, *Senior Member, IEEE*

***Abstract*—Multiple acoustic models are often combined in statistical parametric speech synthesis. Both linear and non-linear functions of an observation sequence are used as features to be modeled. This article shows that this combination of multiple acoustic models can be expressed as a product of experts (PoE); the likelihoods from the models are scaled, multiplied together and then normalized. Normally these models are individually trained and only combined at the synthesis stage. This article discusses a more consistent PoE framework where the models are jointly trained. A training algorithm for PoEs based on linear feature functions and Gaussian experts is derived by generalizing the training algorithm for trajectory HMMs. However for non-linear feature functions or non-Gaussian experts this is not possible, so a scheme based on contrastive divergence learning is described. Experimental results show that the PoE framework provides both a mathematically elegant way to train multiple acoustic models jointly and significant improvements in the quality of the synthesized speech.**

***Index Terms*—statistical parametric speech synthesis, trajectory HMM, product of experts**

I. Introduction

STATISTICAL parametric speech synthesis based on hidden Markov models (HMMs) [38] has grown in popularity in recent years. This approach has various advantages over the concatenative speech synthesis approach, such as the flexibility to change its voice characteristics. However its major limitation is the quality of the synthesized speech. Zen *et al.* [43] highlighted three major factors that degrade the quality of the synthesized speech; vocoding, accuracy of acoustic models (AMs), and over-smoothing.[1] This article addresses the latter two factors, the accuracy of AMs and over-smoothing.

One way to improve the accuracy of the AMs is to use more sophisticated statistical models than HMMs to represent the speech parameter trajectories. There have been various attempts to use other AMs, such as trended HMMs [4], polynomial segment models [27], and autoregressive HMMs [24]. Although these alternative models have been successful to some extent, the dominant AMs in statistical parametric synthesis are still HMMs. Improvements from these alternative models are negligible and require additional model parameters. Furthermore, various essential algorithms such as decision tree-based context clustering [19] or speaker adaptation need to be re-derived for these models.

Zen *et al.* [45] showed that an HMM whose state-output vector included both static and dynamic features could be reformulated as a trajectory model by imposing explicit relationships between the static and dynamic features. This model, called a trajectory HMM, overcomes the conditional independence assumption of state-output probabilities and constant statistics within an HMM state, without the need for additional model parameters. The use of trajectory HMMs has been found to improve the quality of the synthesized speech over HMMs. One of its advantages over other models is that huge amounts of software resources or algorithms developed for HMMs can easily be reused [40], [41] as the parameterization of trajectory HMMs is equivalent to that of HMMs.

To achieve high quality synthesis, speech parameter trajectories generated from AMs should satisfy many constraints at different levels. For example, static/dynamic features and their distributions, which have been used in HMM-based statistical parametric speech synthesis [30], [38], can be viewed as frame-level "soft" constraints. However, they are local and not sufficient to fully describe the characteristics of speech. Other constraints at different levels should be added to achieve better synthesis. Based on this idea, combinations of multiple AMs have been investigated [14], [15], [21], [29], [32]. Here acoustic features of the training data at various levels (*e.g.*, phone, syllable, word, phrase, and utterance) are extracted and modelled *individually*. At the synthesis stage, speech parameters that *jointly* maximize the output probabilities from these multiple AMs are generated. Additionally, the output probabilities from the AMs are weighted to control the contribution of each AM. The weights are tuned manually or optimized using held-out data. The combination of multiple AMs provides extra flexibility to speech synthesis and can reduce the over-smoothing effect [21], [29], [32].

This article proposes a technique to *jointly* estimate these multiple AMs within the product of experts (PoE) framework [9]. The output probabilities from the individual models (experts) are multiplied together and then normalized, effectively forming an intersection of the distributions. This is an efficient way to model high-dimensional data which simultaneously satisfies many different low-dimensional constraints; each ex-

Manuscript received xx, 201x; revised xx, 201x. Part of this work has been presented in ICASSP (Dallas, TX, March 2010) [39]. H. Zen was with the Department of Computer Science and Engineering, Nagoya Institute of Technology, Nagoya, Japan. He is now with Toshiba Research Europe Ltd, Cambridge, UK. M. Gales is with Toshiba Research Europe Ltd., Cambridge, UK. Y. Nankaku, and K. Tokuda are with the Department of Computer Science and Engineering, Nagoya Institute of Technology, Nagoya, Japan. e-mail: heigazen@gmail.com, mark.gales@crl.toshiba.co.uk, {nankaku,tokuda}@sp.nitech.ac.jp.

[1]Over-smoothing appears when there is insufficient flexibility in the model to capture the precise structure of the data. The most significant impact of the over-smoothing is buzzy and muffled synthesized speech.

pert can focus on satisfying just one of these low-dimensional constraints. The use of the PoE framework allows general multiple AMs to be trained cooperatively, removing the need to tune weights.

The remainder of the article is organized as follows. Section II reviews statistical parametric speech synthesis. Section III shows the general PoE framework. Section IV describes the use of the PoE framework for statistical parametric speech synthesis. Experimental results are given in Section V. Concluding remarks are presented in the final section.

II. Statistical Parametric Speech Synthesis

A. HMM-based statistical parametric speech synthesis

A typical HMM-based statistical parametric speech synthesis system [38] consists of training and synthesis components. The training component is similar to that used for speech recognition. First a parametric representation of speech, including spectral parameters (*e.g.*, mel-cepstral coefficients [5] and their dynamic features [6]) and excitation parameters (*e.g.*, $\log F_0$ values, band aperiodicities [42], and their dynamic features), is extracted from the speech database. Second a speech parameter vector sequence $\boldsymbol{o} = \left[\boldsymbol{o}_1^\top, \ldots, \boldsymbol{o}_T^\top\right]^\top$ is formed from the extracted parameters, where $\boldsymbol{o}_t$ denotes a speech parameter vector at frame t and T is the total number of frames in the training data. This speech parameter vector typically consists of static, first- and second-order dynamic features[2] as

$$\boldsymbol{o}_t = \left[\Delta^{(0)} c_t, \Delta^{(1)} c_t, \Delta^{(2)} c_t\right]^\top, \tag{1}$$

where $\Delta^{(d)} c_t$ denotes the d-th order dynamic feature at frame t. They are typically calculated as

$$\Delta^{(0)} c_t = c_t, \tag{2}$$

$$\Delta^{(1)} c_t = \left(c_{t+1} - c_{t-1}\right)/2, \tag{3}$$

$$\Delta^{(2)} c_t = c_{t-1} - 2c_t + c_{t+1}. \tag{4}$$

Then speech parameter trajectories are modeled by a set of context-dependent sub-word (*e.g.*, phone) HMMs $\boldsymbol{\lambda}$ with single Gaussian state-output probability density functions (PDFs). The likelihood of $\boldsymbol{\lambda}$ given $\boldsymbol{o}$ and associated label sequence $\boldsymbol{l} = \{l_1, \ldots, l_L\}$ is given by

$$p(\boldsymbol{o} \mid \boldsymbol{l}, \boldsymbol{\lambda}) = \sum_{\forall \boldsymbol{q}} p(\boldsymbol{o} \mid \boldsymbol{q}, \boldsymbol{\lambda}) P(\boldsymbol{q} \mid \boldsymbol{l}, \boldsymbol{\lambda}), \tag{5}$$

$$p(\boldsymbol{o} \mid \boldsymbol{q}, \boldsymbol{\lambda}) = \prod_{t=1}^{T} p(\boldsymbol{o}_t \mid q_t, \boldsymbol{\lambda}), \tag{6}$$

$$p(\boldsymbol{o}_t \mid q_t, \boldsymbol{\lambda}) = \mathcal{N}\left(\boldsymbol{o}_t ;\, \boldsymbol{\mu}_{q_t}, \boldsymbol{\Sigma}_{q_t}\right), \tag{7}$$

where L is the total number of labels in $\boldsymbol{l}$, $\boldsymbol{q} = \{q_1, \ldots, q_T\}$ is a state sequence (latent variable), $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ correspond to the mean parameter vector and covariance matrix associated with state i defined as

$$\boldsymbol{\mu}_i = \left[\mu_{c(i,0)}, \mu_{c(i,1)}, \mu_{c(i,2)}\right]^\top, \tag{8}$$

$$\boldsymbol{\Sigma}_i = \operatorname{diag}\left[\sigma^2_{c(i,0)}, \sigma^2_{c(i,1)}, \sigma^2_{c(i,2)}\right]. \tag{9}$$

[2]For notational simplicity, static features are assumed to be scalar values. Extensions for vectors and higher-order dynamic features are straightforward.

$\left\{\mu_j, \sigma_j^2\right\}_{j=1}^N$ is the set of unique mean and variance parameters in the model set. $c(i,d) \in \{1, \ldots, N\}$ gives the index of the mean and variance parameter for the d-th dynamic feature at state i.[3] N is the total number of unique mean and variance parameters in the model set. The HMM parameters can be iteratively reestimated based on the maximum likelihood (ML) criterion

$$\hat{\boldsymbol{\lambda}} = \arg\max_{\boldsymbol{\lambda}} p(\boldsymbol{o} \mid \boldsymbol{l}, \boldsymbol{\lambda}), \tag{10}$$

using the Baum-Welch (EM) algorithm.

The synthesis component can be viewed as performing the inverse of speech recognition. First, the given text to be synthesized is converted to a context-dependent label sequence. A sentence HMM is then constructed by concatenating the context-dependent sub-word HMMs according to the label sequence. Second, the state durations of the sentence HMM are determined based on the state-duration PDFs. Third, the sequences of spectral and excitation parameters that maximize their output probabilities under the constraints between static and dynamic features [30] are generated as

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} p(\boldsymbol{o} \mid \hat{\boldsymbol{q}}, \hat{\boldsymbol{\lambda}})\Big|_{\boldsymbol{o}=\boldsymbol{W}\boldsymbol{c}} \tag{11}$$

$$= \arg\max_{\boldsymbol{c}} \mathcal{N}\left(\boldsymbol{o} ;\, \boldsymbol{\mu}_{\hat{\boldsymbol{q}}}, \boldsymbol{\Sigma}_{\hat{\boldsymbol{q}}}\right)\big|_{\boldsymbol{o}=\boldsymbol{W}\boldsymbol{c}} \tag{12}$$

$$= \arg\max_{\boldsymbol{c}} \mathcal{N}\left(\boldsymbol{W}\boldsymbol{c} ;\, \boldsymbol{\mu}_{\hat{\boldsymbol{q}}}, \boldsymbol{\Sigma}_{\hat{\boldsymbol{q}}}\right), \tag{13}$$

where $\hat{\boldsymbol{q}}$ is the state sequence determined by the state-duration PDFs,[4] and $\boldsymbol{\mu}_{\hat{\boldsymbol{q}}}$ and $\boldsymbol{\Sigma}_{\hat{\boldsymbol{q}}}$ correspond to a $3T \times 1$ mean parameter vector and a $3T \times 3T$ covariance parameter matrix defined as

$$\boldsymbol{\mu}_{\boldsymbol{q}} = \left[\boldsymbol{\mu}_{q_1}^\top, \ldots, \boldsymbol{\mu}_{q_T}^\top\right]^\top, \tag{14}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{q}} = \operatorname{diag}\left[\boldsymbol{\Sigma}_{q_1}, \ldots, \boldsymbol{\Sigma}_{q_T}\right]. \tag{15}$$

$\boldsymbol{W}$ is a $3T \times T$ window matrix which gives the relationship between the speech parameter vector sequence $\boldsymbol{o}$ and the static feature vector sequence $\boldsymbol{c} = [c_1, \ldots, c_T]^\top$ as

$$\begin{array}{c} \boldsymbol{o} \\ \left[\begin{array}{c} \vdots \\ \hline \Delta^{(0)} c_{t-1} \\ \Delta^{(1)} c_{t-1} \\ \Delta^{(2)} c_{t-1} \\ \hline \Delta^{(0)} c_t \\ \Delta^{(1)} c_t \\ \Delta^{(2)} c_t \\ \hline \Delta^{(0)} c_{t+1} \\ \Delta^{(1)} c_{t+1} \\ \Delta^{(2)} c_{t+1} \\ \hline \vdots \end{array}\right] \end{array} = \begin{array}{c} \boldsymbol{W} \\ \left[\begin{array}{cccccc} \cdots & \vdots & \vdots & \vdots & \vdots & \cdots \\ \hline \cdots & 0 & 1 & 0 & & \cdots \\ \cdots & -0.5 & 0 & 0.5 & & \cdots \\ \cdots & 1 & -2 & 1 & & \cdots \\ \hline \cdots & & 0 & 1 & 0 & \cdots \\ \cdots & & -0.5 & 0 & 0.5 & \cdots \\ \cdots & & 1 & -2 & 1 & \cdots \\ \hline \cdots & & & 0 & 1 & \cdots \\ \cdots & & & -0.5 & 0 & \cdots \\ \cdots & & & 1 & -2 & \cdots \\ \hline \cdots & \vdots & \vdots & \vdots & \vdots & \cdots \end{array}\right] \end{array} \begin{array}{c} \boldsymbol{c} \\ \left[\begin{array}{c} \vdots \\ \hline c_{t-2} \\ \hline c_{t-1} \\ \hline c_t \\ \hline c_{t+1} \\ \hline c_{t+2} \\ \hline \vdots \end{array}\right] \end{array}. \tag{16}$$

Note that empty elements of $\boldsymbol{W}$ in Eq. (16) are all 0. Setting the partial derivative of $\log \mathcal{N}\left(\boldsymbol{W}\boldsymbol{c} ;\, \boldsymbol{\mu}_{\hat{\boldsymbol{q}}}, \boldsymbol{\Sigma}_{\hat{\boldsymbol{q}}}\right)$ with respect to $\boldsymbol{c}$ to $\boldsymbol{0}$ yields a set of linear equations to determine $\hat{\boldsymbol{c}}$ as

$$\boldsymbol{R}_{\hat{\boldsymbol{q}}} \hat{\boldsymbol{c}} = \boldsymbol{r}_{\hat{\boldsymbol{q}}}, \tag{17}$$

[3]Usually they are defined by the results from the decision tree-based context clustering [19].

[4]If a left-to-right, no skip, structure is used as the HMM topology, determining the state durations is equivalent to determining the state sequence.

where $\boldsymbol{R}_{\hat{\boldsymbol{q}}}$ and $\boldsymbol{r}_{\hat{\boldsymbol{q}}}$ correspond to the $T \times T$ matrix and the $T \times 1$ vector given by

$$\boldsymbol{R}_{\hat{\boldsymbol{q}}} = \boldsymbol{W}^\top \boldsymbol{\Sigma}_{\hat{\boldsymbol{q}}}^{-1} \boldsymbol{W}, \tag{18}$$

$$\boldsymbol{r}_{\hat{\boldsymbol{q}}} = \boldsymbol{W}^\top \boldsymbol{\Sigma}_{\hat{\boldsymbol{q}}}^{-1} \boldsymbol{\mu}_{\hat{\boldsymbol{q}}}. \tag{19}$$

Equation (17) can be solved efficiently by the Cholesky decomposition as $\boldsymbol{R}_{\hat{\boldsymbol{q}}}$ becomes a positive definite symmetric band matrix [30]. Trajectories for both the spectral and excitation parameters are generated in this fashion. The speech waveform is synthesized directly from the generated spectral and excitation parameters using a speech synthesis filter.

B. Trajectory HMM

The previous section described how HMMs can be trained and the generated speech parameter trajectory can be used for synthesis. However, there exists an inconsistency; the relationships between the static and dynamic features are ignored in the HMM training but utilized in speech parameter generation. This inconsistency degrades the accuracy of the models and the quality of the synthesized speech.

To address this problem, Zen *et al.* [45] incorporated relationships between the static and dynamic features explicitly into training. Equations (6) and (7) can be rewritten as

$$p(\boldsymbol{o} \mid \boldsymbol{q}, \boldsymbol{\lambda}) = \mathcal{N}(\boldsymbol{o}\,;\, \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}). \tag{20}$$

If a distribution over the static feature vectors is considered, Eq. (20) is not a *valid* (properly normalized) PDF;

$$\tilde{p}(\boldsymbol{W}\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}) = \mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\, \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}), \tag{21}$$

$$\int_{\mathcal{R}^T} \mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\, \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}) d\boldsymbol{c} \neq 1, \tag{22}$$

where $\tilde{p}(\cdot)$ denotes an unnormalized PDF. It should be normalized to yield a valid (properly normalized) PDF. The normalization constant $Z_{\boldsymbol{q}}^{(\mathrm{trj})}$ can be computed in a closed form as

$$Z_{\boldsymbol{q}}^{(\mathrm{trj})} = \int_{\mathcal{R}^T} \mathcal{N}\left(\boldsymbol{W}\boldsymbol{c}\,;\, \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}\right) d\boldsymbol{c} \tag{23}$$

$$= \frac{\sqrt{(2\pi)^T \left|\boldsymbol{P}_{\boldsymbol{q}}\right|}}{\sqrt{(2\pi)^{3T} \left|\boldsymbol{\Sigma}_{\boldsymbol{q}}\right|}} \exp\left\{-\frac{1}{2}\left(\boldsymbol{\mu}_{\boldsymbol{q}}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{\mu}_{\boldsymbol{q}} - \boldsymbol{r}_{\boldsymbol{q}}^\top \boldsymbol{R}_{\boldsymbol{q}}^{-1} \boldsymbol{r}_{\boldsymbol{q}}\right)\right\}. \tag{24}$$

Thus the output probability of $\boldsymbol{c}$ rather than $\boldsymbol{o}$ given $\boldsymbol{q}$ and $\boldsymbol{\lambda}$ can be defined as

$$p(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}) = \frac{1}{Z_{\boldsymbol{q}}^{(\mathrm{trj})}} \mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\, \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}) \tag{25}$$

$$= \mathcal{N}(\boldsymbol{c}\,;\, \bar{\boldsymbol{c}}_{\boldsymbol{q}}, \boldsymbol{P}_{\boldsymbol{q}}), \tag{26}$$

where $\bar{\boldsymbol{c}}_{\boldsymbol{q}}$ and $\boldsymbol{P}_{\boldsymbol{q}}$ correspond to the $T \times 1$ mean vector and the $T \times T$ covariance matrix for $\boldsymbol{q}$ given as

$$\boldsymbol{R}_{\boldsymbol{q}} \bar{\boldsymbol{c}}_{\boldsymbol{q}} = \boldsymbol{r}_{\boldsymbol{q}}, \tag{27}$$

$$\boldsymbol{P}_{\boldsymbol{q}} = \boldsymbol{R}_{\boldsymbol{q}}^{-1}. \tag{28}$$

It can be seen from Eqs. (17) and (27) that $\bar{\boldsymbol{c}}_{\boldsymbol{q}}$ is exactly the same as the speech parameter trajectory generated by the speech parameter generation algorithm. By replacing Eq. (6) by Eq. (25), a trajectory HMM is defined as

$$p\left(\boldsymbol{c} \mid \boldsymbol{l}, \boldsymbol{\lambda}\right) = \sum_{\forall \boldsymbol{q}} p\left(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}\right) P\left(\boldsymbol{q} \mid \boldsymbol{l}, \boldsymbol{\lambda}\right), \tag{29}$$

$$p\left(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}\right) = \mathcal{N}\left(\boldsymbol{c}\,;\, \bar{\boldsymbol{c}}_{\boldsymbol{q}}, \boldsymbol{P}_{\boldsymbol{q}}\right). \tag{30}$$

It should be noted that the mean vector $\bar{\boldsymbol{c}}_{\boldsymbol{q}}$ yields a smooth trajectory and the inter-frame covariance matrix $\boldsymbol{P}_{\boldsymbol{q}}$ is generally full. Therefore, the trajectory HMM overcomes two fundamental limitations of HMMs; constant statistics within an HMM state; and the conditional independence assumptions of state-output probabilities.

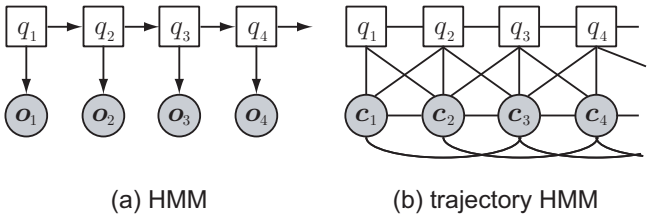

Fig. 1. Graphical model representation of (a) HMM and (b) trajectory HMM with window matrix given by Eqs. (2)–(4).

It is interesting to note that the trajectory HMM is related to a Markov random field (MRF) [13], whose cliques are defined by $\boldsymbol{W}$ and clique potential functions are given by Gaussian distributions. As a latent variable (state sequence) exists and potential functions are Gaussian distributions, a trajectory HMM is actually a hidden Gaussian Markov random field (HGMRF) [22] over time. It is known that MRFs can be represented as undirected graphical models [2]. The graphical model representations of an HMM and trajectory HMM whose window matrix is specified by Eqs. (2)–(4) are shown in Fig. 1. Note that edges in Fig. 1(b) depends on cliques that are specified by the window coefficients. Therefore, if different windows are used to compute dynamic features, the graphical model structure of the trajectory HMM will change.

ML estimation of trajectory HMMs can be carried out using the EM algorithm.[5] Here, the auxiliary function is defined as

$$\mathcal{Q}\left(\boldsymbol{\lambda}; \boldsymbol{\lambda}'\right) = \sum_{\forall \boldsymbol{q}} \gamma_{\boldsymbol{q}} \log p(\boldsymbol{c}, \boldsymbol{q} \mid \boldsymbol{l}, \boldsymbol{\lambda}), \tag{31}$$

where $\boldsymbol{\lambda}'$ and $\boldsymbol{\lambda}$ are the current and new sets of model parameters, respectively. $\gamma_{\boldsymbol{q}}$ is the posterior probability of $\boldsymbol{q}$ given $\boldsymbol{c}$, $\boldsymbol{l}$, and $\boldsymbol{\lambda}'$. The reestimation formula of all mean parameters is derived [45] as

$$\hat{\boldsymbol{\mu}} = \boldsymbol{G}^{-1} \boldsymbol{k}, \tag{32}$$

where $\boldsymbol{\mu} = \left[\mu_1, \ldots, \mu_N\right]^\top$ is a vector consisting of all unique mean parameters in the model set. $\boldsymbol{G}$ and $\boldsymbol{k}$ are accumulated statistics computed as

$$\boldsymbol{G} = \sum_{\forall \boldsymbol{q}} \gamma_{\boldsymbol{q}} \boldsymbol{S}_{\boldsymbol{q}}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{W} \boldsymbol{P}_{\boldsymbol{q}} \boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{S}_{\boldsymbol{q}}, \tag{33}$$

$$\boldsymbol{k} = \sum_{\forall \boldsymbol{q}} \gamma_{\boldsymbol{q}} \boldsymbol{S}_{\boldsymbol{q}}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{W} \boldsymbol{c}, \tag{34}$$

[5]The single path (Viterbi) [45] or Monte Carlo [44] approximation is often employed, as it is intractable to marginalize over all possible state sequences.

where $\boldsymbol{S_q}$ is a matrix representing the parameter sharing structure and the index function $c(i,j)$. The relationship between $\boldsymbol{\mu_q}$ of Eq. (14) and $\boldsymbol{\mu}$ can be written using $\boldsymbol{S_q}$ as

$$\boldsymbol{\mu_q} = \boldsymbol{S_q}\boldsymbol{\mu}. \tag{35}$$

For example, if $c(q_1,0) = c(q_2,0) = 1$, $c(q_1,1) = c(q_2,1) = 2$, $c(q_1,2) = c(q_2,2) = 3$, $c(q_3,0) = 4$, $c(q_3,1) = 5$, $c(q_3,2) = 6$, $T = 3$, and $N = 6$ then Eq. (35) is illustrated as

$$\begin{bmatrix} \boldsymbol{\mu}_{q_1} \\ \hline \boldsymbol{\mu}_{q_2} \\ \hline \boldsymbol{\mu}_{q_3} \end{bmatrix} = \left[\begin{array}{cccccc} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ \hline 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ \hline & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{array}\right] \begin{bmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \\ \mu_5 \\ \mu_6 \end{bmatrix}. \tag{36}$$

Note that the empty elements of $\boldsymbol{S_q}$ in Eq. (36) are all 0. There is no closed form solution to reestimate variance parameters. Therefore, a gradient method is employed. The partial derivative of Eq. (31) with respect to all variance parameters in the model set can be expressed as

$$\begin{aligned} \frac{\partial \mathcal{Q}(\boldsymbol{\lambda};\boldsymbol{\lambda}')}{\partial \boldsymbol{\phi}} = \sum_{\forall \boldsymbol{q}} \frac{\gamma_{\boldsymbol{q}}}{2} \boldsymbol{S_q}^\top \operatorname{diag}^{-1} \Big[& \boldsymbol{W}\boldsymbol{P_q}\boldsymbol{W}^\top \\ & - \boldsymbol{W}\boldsymbol{c}\boldsymbol{c}^\top\boldsymbol{W}^\top + \boldsymbol{W}\bar{\boldsymbol{c}}_{\boldsymbol{q}}\bar{\boldsymbol{c}}_{\boldsymbol{q}}^\top\boldsymbol{W}^\top \\ & + \boldsymbol{W}(\boldsymbol{c} - \bar{\boldsymbol{c}}_{\boldsymbol{q}})\boldsymbol{\mu}_{\boldsymbol{q}}^\top + \boldsymbol{\mu}_{\boldsymbol{q}}(\boldsymbol{c} - \bar{\boldsymbol{c}}_{\boldsymbol{q}})^\top \boldsymbol{W}^\top \Big], \end{aligned} \tag{37}$$

where $\boldsymbol{\phi} = \left[1/\sigma_1^2, \ldots, 1/\sigma_N^2\right]$ is a vector consisting of all unique precision (inverse variance) parameters in the model set. Note that relationship between $\boldsymbol{\Sigma_q}$ of Eq. (15) and $\boldsymbol{\phi}$ can also be written using $\boldsymbol{S_q}$ as

$$\boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} = \operatorname{diag}\left[\boldsymbol{S_q}\boldsymbol{\phi}\right]. \tag{38}$$

The ML estimation of the trajectory HMM can improve the quality of the synthesized speech [45].

C. Model combinations for speech synthesis

A major limitation of statistical parametric speech synthesis is the quality of the synthesized speech. This often sounds buzzy and muffled [43]. There exist three major factors that degrade the quality of the synthesized speech; vocoding, accuracy of the AMs, and over-smoothing. To provide additional flexibility to model constraints at various levels, so that the over-smoothing effect is reduced, the combining of multiple AMs has recently been proposed. These techniques extract acoustic features of the training data at various levels (*e.g.*, phone, syllable, word, phrase, and utterance). Both linear (*e.g.*, summation [15], [21], average [32], and DCT [14], [21]) and non-linear (*e.g.*, quadratic [29]) functions of the observation sequence are used. Then the extracted features at each level are modeled by an AM consisting of a set of context-dependent Gaussian (*e.g.*, [14], [15], [21], [29], [32]) and/or non-Gaussian (*e.g.* gamma [17], [21] and log Gaussian [35]) distributions. Typically the AM at each level is trained *individually* based on the ML criterion as

$$\hat{\boldsymbol{\lambda}}_j = \arg\max_{\boldsymbol{\lambda}_j} p(f_j(\boldsymbol{c}) \mid \boldsymbol{\lambda}_j), \qquad j = 1, \ldots, M \tag{39}$$

where M is the number of AMs and $\boldsymbol{\lambda}_j$ and $f_j(\boldsymbol{c})$ correspond to the set of parameters and an arbitrary function to extract features from observation $\boldsymbol{c}$ for the j-th AM. At the synthesis stage, speech parameters that *jointly* maximize the output probabilities from these multiple AMs are generated as

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} \prod_{j=1}^{M} \left\{ p(f_j(\boldsymbol{c}) \mid \hat{\boldsymbol{\lambda}}_j) \right\}^{\alpha_j} \tag{40}$$

$$= \arg\max_{\boldsymbol{c}} \sum_{j=1}^{M} \alpha_j \log p(f_j(\boldsymbol{c}) \mid \hat{\boldsymbol{\lambda}}_j), \tag{41}$$

where α_j is the weight for the j-th AM. Thus the output probabilities from the AMs are weighted to control the contribution of each AM. These weights are tuned manually or optimized using held-out data. A closed-form solution of Eq. (40) can be found if all AMs are Gaussian and all feature functions are linear [14], [15], [21], [32]. Otherwise, a gradient method is often used [29]. This technique can generate the speech parameter trajectory that *jointly* satisfies constraints in multiple feature spaces and gives better synthesized speech.

Although this framework allows multiple AMs to be used for synthesis, there exists an inconsistency again; the AMs are trained individually but combined at the synthesis stage. The next section will propose a technique to *jointly* estimate multiple AMs within the product of experts (PoE) framework [9]. The use of the PoE framework allows general multiple AMs to be trained cooperatively, removing the need to tune weights.

III. Product of Experts

This section reviews the general framework of product of experts (PoE) and its training algorithms. Its application to statistical parametric speech synthesis will be described in the next section.

A. PoE framework

A product of experts (PoE) [9], [33] combines multiple models (experts) by taking their product and normalizing the result. Each expert can be an unnormalized model[6] $\tilde{p}(\boldsymbol{x} \mid \boldsymbol{\lambda}_j)$ over the input space. A PoE is expressed as

$$p(\boldsymbol{x} \mid \{\boldsymbol{\lambda}_j\}_{j=1}^M) = \frac{1}{Z} \prod_{j=1}^{M} \tilde{p}(\boldsymbol{x} \mid \boldsymbol{\lambda}_j), \tag{42}$$

where $\boldsymbol{x}$ is a K-dimensional input vector.[7] Z is a normalization constant computed as

$$Z = \int_{\mathcal{R}^K} \prod_{j=1}^{M} \tilde{p}(\boldsymbol{x} \mid \boldsymbol{\lambda}_j) d\boldsymbol{x}. \tag{43}$$

[6]In unnormalized models, $\int_{\mathcal{R}^K} \tilde{p}(\boldsymbol{x} \mid \boldsymbol{\lambda}_i) d\boldsymbol{x} \neq 1$.

[7]In this section feature functions, $\{f_j(\cdot)\}$, are omitted for notational simplicity.

The PoE can be contrasted with a mixture of experts (MoE) [12], which combines expert models additively,

$$p(\boldsymbol{x} \mid \{\boldsymbol{\lambda}_j\}_{j=1}^M) = \sum_{j=1}^M w_j p(\boldsymbol{x} \mid \boldsymbol{\lambda}_j), \tag{44}$$

where each model $p(\boldsymbol{x} \mid \boldsymbol{\lambda}_j)$ is normalized over $\boldsymbol{x}$ as

$$\int_{\mathcal{R}^K} p(\boldsymbol{x} \mid \boldsymbol{\lambda}_j) d\boldsymbol{x} = 1, \qquad j = 1, \ldots, M \tag{45}$$

and the weights must satisfy

$$\sum_{j=1}^M w_j = 1. \tag{46}$$

The MoE can have a high probability for the input where one or more models assign high probability, and thus the MoE tends to be broader than the individual models alone. It can be thought as a *union* of all models. On the other hand, the PoE can have a high probability for the input only where all the models assign high probability. Thus, the PoE tends to be sharper than its individual models. It can be thought of as an *intersection* of all models. The PoE is an efficient way to represent high-dimensional data which simultaneously satisfies many different low-dimensional constraints; each model can focus on satisfying just one of these low-dimensional constraints. As AMs for speech synthesis requires many constraints at different levels, the PoE is more suitable than the MoE.

B. Training PoE

1) Gaussian case: Training MoEs by the EM algorithm is usually straight-forward. However, training PoEs is significantly more complicated, due to the normalization constant. This issue has motivated various approximate training schemes for PoEs. One way to address this problem is to use tractable distributions for experts. If the individual experts are Gaussian or Gaussian mixtures, the resultant PoEs are also Gaussian or Gaussian mixtures. Its normalization constant can be found in a closed form [7] thus the training is dramatically simplified compared to the general PoEs.

The product of Gaussian distributions (PoG) can be written as

$$p(\boldsymbol{x} \mid \{\boldsymbol{\lambda}_j\}_{j=1}^M) = \frac{1}{Z} \prod_{j=1}^M \mathcal{N}(\boldsymbol{x} \,;\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{47}$$

$$= \mathcal{N}(\boldsymbol{x} \,;\, \boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*), \tag{48}$$

where $\boldsymbol{\mu}_*$ and $\boldsymbol{\Sigma}_*$ correspond to the mean vector and the covariance matrix of the resulting distribution given by

$$\boldsymbol{\mu}_* = \boldsymbol{\Sigma}_* \left(\sum_{j=1}^M \boldsymbol{\Sigma}_j^{-1} \boldsymbol{\mu}_j \right), \tag{49}$$

$$\boldsymbol{\Sigma}_* = \left(\sum_{j=1}^M \boldsymbol{\Sigma}_j^{-1} \right)^{-1}. \tag{50}$$

Unlike many other PoEs, there exists a closed form expression for Z as

$$Z = \frac{\sqrt{(2\pi)^K |\boldsymbol{\Sigma}_*|}}{\prod_{j=1}^M \sqrt{(2\pi)^K |\boldsymbol{\Sigma}_j|}} \exp\left\{ -\frac{1}{2} \left(\sum_{j=1}^M \boldsymbol{\mu}_j^\top \boldsymbol{\Sigma}_j^{-1} \boldsymbol{\mu}_j - \boldsymbol{\mu}_*^\top \boldsymbol{\Sigma}_*^{-1} \boldsymbol{\mu}_* \right) \right\}. \tag{51}$$

Parameter reestimation formulae of PoGs [1] and extension to mixture of Gaussians have been derived [7].

2) General case: Training general PoEs is complicated because of the normalization constant Z. However, there exists a simple and effective technique to approximate ML estimation of general PoEs [9]. By taking the partial derivative of the log likelihood with respect to $\boldsymbol{\lambda}_k$, the parameter update based on the ML criterion by the steepest ascent method is written as[8]

$$\boldsymbol{\lambda}_k' = \boldsymbol{\lambda}_k + \eta \nabla \boldsymbol{\lambda}_k, \quad k = 1, \ldots, M \tag{52}$$

$$\nabla \boldsymbol{\lambda}_k = \frac{1}{D} \sum_{i=1}^D \frac{\partial}{\partial \boldsymbol{\lambda}_k} \mathcal{L}(\boldsymbol{x}_i \,;\, \{\boldsymbol{\lambda}_j\}_{j=1}^M), \tag{53}$$

where η is a user-defined learning rate, $\boldsymbol{x}_i$ is the i-th training sample, and D is the total number of training samples. $\mathcal{L}(\boldsymbol{x} \,;\, \{\boldsymbol{\lambda}_j\}_{j=1}^M)$ denotes the log likelihood of the PoE as

$$\mathcal{L}(\boldsymbol{x} \,;\, \{\boldsymbol{\lambda}_j\}_{j=1}^M) = \log p(\boldsymbol{x} \mid \{\boldsymbol{\lambda}_j\}_{j=1}^M) \tag{54}$$

$$= \sum_{j=1}^M \log \tilde{p}(\boldsymbol{x} \mid \boldsymbol{\lambda}_j) - \log Z \tag{55}$$

$$= \tilde{\mathcal{L}}(\boldsymbol{x} \,;\, \{\boldsymbol{\lambda}_j\}_{j=1}^M) - \log Z, \tag{56}$$

where $\tilde{\mathcal{L}}(\boldsymbol{x} \,;\, \{\boldsymbol{\lambda}_j\}_{j=1}^M)$ denotes the unnormalized log likelihood of the PoE. Equation (56) reproduces

$$\frac{1}{D} \sum_{i=1}^D \frac{\partial}{\partial \boldsymbol{\lambda}_k} \mathcal{L}(\boldsymbol{x}_i \,;\, \{\boldsymbol{\lambda}_j\}_{j=1}^M) = \frac{1}{D} \sum_{i=1}^D \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\boldsymbol{x}_i \,;\, \{\boldsymbol{\lambda}_j\}_{j=1}^M) - \frac{\partial}{\partial \boldsymbol{\lambda}_k} \log Z. \tag{57}$$

The first term of Eq. (57) can be expressed as

$$\frac{1}{D} \sum_{i=1}^D \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\boldsymbol{x}_i \,;\, \{\boldsymbol{\lambda}_j\}_{j=1}^M) = \left\langle \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\boldsymbol{x} \,;\, \{\boldsymbol{\lambda}_j\}_{j=1}^M) \right\rangle_{p^0(\boldsymbol{x})}, \tag{58}$$

where $p^0(\boldsymbol{x})$ denotes the empirical (data) distribution, $\frac{1}{D} \sum_{i=1}^D \delta(\boldsymbol{x} - \boldsymbol{x}_i)$, and $\langle \cdot \rangle_{p^0(\boldsymbol{x})}$ denotes the expectation over the empirical distribution. The second term of Eq. (57) results

[8]Here contrastive divergence learning is derived as an approximation to the derivative of the log likelihood with respect to model parameters. Alternatively, it is possible to formulate it as a minimization of the Kullback-Leibler (KL) divergence between empirical (data) and model distribution [10].

in

$$\frac{\partial}{\partial \boldsymbol{\lambda}_k} \log Z = \frac{1}{Z} \frac{\partial}{\partial \boldsymbol{\lambda}_k} Z \tag{59}$$

$$= \frac{1}{Z} \frac{\partial}{\partial \boldsymbol{\lambda}_k} \int \exp\left\{\tilde{\mathcal{L}}(\boldsymbol{x}\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M)\right\} d\boldsymbol{x} \tag{60}$$

$$= \int \frac{1}{Z} \frac{\partial}{\partial \boldsymbol{\lambda}_k} \exp\left\{\tilde{\mathcal{L}}(\boldsymbol{x}\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M)\right\} d\boldsymbol{x} \tag{61}$$

$$= \int \frac{1}{Z} \exp\left\{\tilde{\mathcal{L}}(\boldsymbol{x}\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M)\right\} \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\boldsymbol{x}\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M) d\boldsymbol{x} \tag{62}$$

$$= \left\langle \frac{\partial \tilde{\mathcal{L}}(\boldsymbol{x}\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M)}{\partial \boldsymbol{\lambda}_k} \right\rangle_{p^\infty(\boldsymbol{x})}, \tag{63}$$

where $p^\infty(\boldsymbol{x})$ denotes the model distribution and $\langle\cdot\rangle_{p^\infty(\boldsymbol{x})}$ denotes the expectation over the model distribution. Thus Eq. (53) can be rewritten as

$$\nabla \boldsymbol{\lambda}_k = \left\langle \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\boldsymbol{x}\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M) \right\rangle_{p^0(\boldsymbol{x})} - \left\langle \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\boldsymbol{x}\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M) \right\rangle_{p^\infty(\boldsymbol{x})}. \tag{64}$$

It can be seen from the above equation that the normalization constant Z is not required. While the expectation over the training data is easy to compute, the expectation over the model distribution is computationally expensive. The expectation over the model distribution is typically computed by running MCMC sampling [2] but it may take a very long time until the Markov chain converges. Alternatively, *contrastive divergence* learning [9] approximates the expectation over the model distribution as

$$\nabla \boldsymbol{\lambda}_k \approx \left\langle \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\boldsymbol{x}\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M) \right\rangle_{p^0(\boldsymbol{x})} - \left\langle \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\boldsymbol{x}\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M) \right\rangle_{p^V(\boldsymbol{x})} \tag{65}$$

$$= \frac{1}{D} \sum_{i=1}^{D} \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\boldsymbol{x}_i\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M) - \frac{1}{V} \sum_{v=1}^{V} \frac{\partial}{\partial \boldsymbol{\lambda}_k} \tilde{\mathcal{L}}(\tilde{\boldsymbol{x}}_v\ ;\ \{\boldsymbol{\lambda}_j\}_{j=1}^M) \tag{66}$$

where $\langle\cdot\rangle_{p^V(\boldsymbol{x})}$ denotes the expectation over the model distribution after V MCMC sampling iterations and $\tilde{\boldsymbol{x}}_v$ is the sample drawn from the model distribution at the v-th MCMC sampling iteration.

The key idea of contrastive divergence learning is to initialize the sampler at the data points rather than random values, and run MCMC iterations for a small, fixed number of steps (typically $V = 1$ or $V = 10$) rather than very long iterations until the Markov chain converges to equilibrium ($V = \infty$). The intuition here is that by sampling for just a few iterations starting from the data points will draw samples close to a mode of the model distribution, which should be sufficient to estimate the parameter updates. Contrastive divergence learning has been applied for training various models in machine learning including a restricted Boltzmann machine (RBM) [10], which is one of the simplest forms of PoEs, and a deep belief net (DBN) [8], which is a multi-layered composition of RBMs. Refer to [10] for further details about contrastive divergence learning.

IV. PoEs for Statistical Parametric Speech Synthesis

A. Trajectory HMM as PoE

If a feature function to compute d-th order dynamic features at frame t given $\boldsymbol{c}$ is defined as

$$f_t^{(d)}(\boldsymbol{c}) = \Delta^{(d)} c_t, \tag{67}$$

Eq. (30) can be reformulated as

$$p\left(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}\right) = \frac{1}{Z_{\boldsymbol{q}}^{(\mathrm{trj})}} \mathcal{N}\left(\boldsymbol{W}\boldsymbol{c}\ ;\ \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}\right) \tag{68}$$

$$= \frac{1}{Z_{\boldsymbol{q}}^{(\mathrm{trj})}} \prod_{t=1}^{T} \prod_{d=0}^{2} \mathcal{N}\left(f_t^{(d)}(\boldsymbol{c})\ ;\ \mu_{c(q_t,d)}, \sigma^2_{c(q_t,d)}\right). \tag{69}$$

As discussed in [34], [45], Eq. (69) can be viewed as a PoG; local constraints (static and dynamic characteristics of speech parameter trajectory) are modeled by unnormalized Gaussian experts. They are multiplied over time and then normalized to yield a valid PDF. Here, the number of experts is three times larger than the input dimension. This type of PoE is called an over-complete PoE [28].

B. Combining multiple AMs as PoE

Considering each AM as an "expert", the combination of multiple AMs described in Section II-C can be reformulated as a PoE

$$p(\boldsymbol{c} \mid \{\boldsymbol{\lambda}_j\}_{j=1}^M) = \frac{1}{Z} \prod_{j=1}^{M} \{p(f_j(\boldsymbol{c}) \mid \boldsymbol{\lambda}_j)\}^{\alpha_j}. \tag{70}$$

It allows us to estimate all AMs jointly based on the ML criterion

$$\{\hat{\boldsymbol{\lambda}}_j\}_{j=1}^M = \arg\max_{\{\boldsymbol{\lambda}_j\}_{j=1}^M} \frac{1}{Z} \prod_{j=1}^{M} \{p(f_j(\boldsymbol{c}) \mid \boldsymbol{\lambda}_j)\}^{\alpha_j}. \tag{71}$$

This training framework is consistent with the synthesis framework of Eq. (40); the combined AMs are considered both at the training and synthesis stages.

This section shows how to estimate multiple AMs simultaneously based on the PoE framework. This removes the need to tune weights as the variances of the individual expert will subsume their role.

1) Linear and Gaussian case: Here the linear and Gaussian case is discussed with multiple-level duration models [15] as an example. In a typical HMM-based statistical parametric speech synthesis system, the state durations are modeled explicitly by context-dependent single Gaussian distributions clustered by decision trees [38]. At the synthesis stage, these state-duration models are used to determine the most probable state sequence. Durations can be predicted more accurately

if the information of states and higher-level speech units are incorporated [15], [21]. In this work the state and phone durations are modeled by Gaussians, thus

$$p(d_{ij} \mid l_i, \boldsymbol{\lambda}_{\rm st}) = \mathcal{N}\left(d_{ij} \,;\, \nu_{q(i,j)}, \xi_{q(i,j)}\right), \tag{72}$$

$$p(h_i \mid l_i, \boldsymbol{\lambda}_{\rm phn}) = \mathcal{N}\left(h_i \,;\, \nu_{r(i)}, \xi_{r(i)}\right), \tag{73}$$

where d_{ij} denotes the state duration of state j of phone i, h_i is the phone duration of phone i, which can be computed from the state durations by

$$h_i = \sum_{j=1}^{S} d_{ij}, \tag{74}$$

and S is the number of states in a sub-word HMM. ν_i and ξ_i correspond to the i-th mean and variance parameters. $q(i,j) \in \{1, \ldots, N_{\rm st}\}$ gives the index of the mean and variance parameter for the duration of j-th state of i-th phone. $r(i) \in \{1, \ldots, N_{\rm ph}\}$ gives the index of the mean and variance parameter for the duration of i-th phone. $N_{\rm st}$ and $N_{\rm ph}$ correspond to the numbers of unique Gaussian distributions in the state and phone duration model sets. The combination of state and phone duration models can be reformulated as a PoE by

$$p(\boldsymbol{d} \mid \boldsymbol{l}, \boldsymbol{\lambda}_{\rm st}, \boldsymbol{\lambda}_{\rm phn}) = \frac{1}{Z} \prod_{i=1}^{L} \left\{ p(h_i \mid l_i, \boldsymbol{\lambda}_{\rm phn}) \prod_{j=1}^{S} p(d_{ij} \mid l_i, \boldsymbol{\lambda}_{\rm st}) \right\} \tag{75}$$

$$= \frac{1}{Z} \mathcal{N}(\boldsymbol{W}\boldsymbol{d} \,;\, \boldsymbol{\mu}_{\boldsymbol{l}}, \boldsymbol{\Sigma}_{\boldsymbol{l}}), \tag{76}$$

where $\boldsymbol{d} = [d_{11}, \ldots, d_{1S}, \ldots, d_{L1}, \ldots, d_{LS}]^\top$ is the sequence of the state durations and L is the number of phones in the sentence HMM. For example, if $S = 3$, $L = 2$, $N = 5$, $q(1,1) = q(2,1) = 1$, $q(1,2) = q(2,2) = 2$, $q(1,3) = q(2,3) = 3$, $r(1) = 4$, and $r(2) = 5$, then $\boldsymbol{W}$ and $\boldsymbol{S}_{\boldsymbol{l}}$ can be illustrated as

$$\underset{\boldsymbol{o}}{\begin{bmatrix} d_{11} \\ d_{11} \\ d_{13} \\ \hline d_{21} \\ d_{21} \\ d_{23} \\ \hline h_1 \\ h_2 \end{bmatrix}} = \underset{\boldsymbol{W}}{\left[\begin{array}{ccc|ccc} 1 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ \hline & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \\ \hline 1 & 1 & 1 & & & \\ & & & 1 & 1 & 1 \end{array}\right]} \underset{\boldsymbol{d}}{\begin{bmatrix} d_{11} \\ d_{12} \\ d_{13} \\ \hline d_{21} \\ d_{22} \\ d_{23} \end{bmatrix}}, \tag{77}$$

$$\underset{\boldsymbol{\mu}_{\boldsymbol{l}}}{\begin{bmatrix} \nu_{q(1,1)} \\ \nu_{q(1,2)} \\ \nu_{q(1,3)} \\ \hline \nu_{q(2,1)} \\ \nu_{q(2,2)} \\ \nu_{q(2,3)} \\ \hline \nu_{r(1)} \\ \nu_{r(2)} \end{bmatrix}} = \underset{\boldsymbol{S}_{\boldsymbol{l}}}{\left[\begin{array}{ccc|cc} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ \hline 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ \hline & & & 1 & \\ & & & & 1 \end{array}\right]} \underset{\boldsymbol{\mu}}{\begin{bmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \\ \nu_4 \\ \nu_5 \end{bmatrix}}. \tag{78}$$

Again, the empty elements of $\boldsymbol{W}$ and $\boldsymbol{S}_{\boldsymbol{l}}$ in Eqs. (77) and (78) are all 0. It can be seen from Eqs. (16), (36), (68)–(69), and (75)–(78) that this PoE has exactly the same form as the trajectory HMM; only the structures of $\boldsymbol{W}$ and $\boldsymbol{S}_{\boldsymbol{q}}$ are different. Most techniques for combining multiple AMs adopt linear functions of the observation sequence, such as summation [15], [21], average [32], DCT [14], [21], and use Gaussian distributions to model the extracted features. Any combinations of linear feature functions and Gaussian experts, can be reformulated as a trajectory HMM. Thus the parameter update formulae derived for trajectory HMMs can directly be applied to jointly estimate the multiple AMs.

It is known that PoEs with linear feature functions and Gaussian experts can be viewed as a basis superposition precision matrix model [26]; its precision matrix, $\boldsymbol{R}_{\boldsymbol{q}}$, is formed by superimposing multiple basis matrices, each of which is defined by an expert.

2) Non-linear and non-Gaussian cases: With non-linear feature functions, or non-Gaussian experts, it is not possible to use the trajectory HMM's parameter update formulae. One example is speech parameter generation including a global variance (GV) term [29]. The GV is defined as the intra-utterance variance of a speech parameter trajectory and typically modeled by a Gaussian distribution. The PoE for speech parameter generation including the GV term is written as

$$p(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\lambda}_{\rm gv}) = \frac{1}{Z_{\boldsymbol{q}}^{(\rm trj \cdot gv)}} \tilde{p}(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}) \tilde{p}\left(f_{\rm gv}(\boldsymbol{c}) \mid \boldsymbol{\lambda}_{\rm gv}\right), \tag{79}$$

$$\tilde{p}(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}) = \left\{\mathcal{N}\left(\boldsymbol{c} \,;\, \bar{\boldsymbol{c}}_{\boldsymbol{q}}, \boldsymbol{P}_{\boldsymbol{q}}\right)\right\}^{\alpha_{\rm gv}}, \tag{80}$$

$$\tilde{p}\left(f_{\rm gv}(\boldsymbol{c}) \mid \boldsymbol{\lambda}_{\rm gv}\right) = \mathcal{N}\left(f_{\rm gv}(\boldsymbol{c}) \,;\, \mu_{\rm gv}, \sigma_{\rm gv}^2\right), \tag{81}$$

where $\boldsymbol{\lambda}_{\rm gv}$ is the set of parameters for GV, $\alpha_{\rm gv}$ is an utterance-length adaptive weight (typically $\alpha_{\rm gv} = 1/3T$),[9] and $\mu_{\rm gv}$ and $\sigma_{\rm gv}^2$ correspond to the mean and variance of the GV Gaussian distribution. $f_{\rm gv}(\boldsymbol{c})$ is a function to compute the GV given $\boldsymbol{c}$, which is defined as

$$f_{\rm gv}(\boldsymbol{c}) = \frac{1}{T} \sum_{t=1}^{T} (c_t - \bar{c})^2, \quad \bar{c} = \frac{1}{T} \sum_{t=1}^{T} c_t. \tag{82}$$

$Z_{\boldsymbol{q}}^{(\rm trj \cdot GV)}$ is a normalization constant given as

$$Z_{\boldsymbol{q}}^{(\rm trj \cdot gv)} = \int_{\mathcal{R}^T} \tilde{p}(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}) \tilde{p}\left(f_{\rm gv}(\boldsymbol{c}) \mid \boldsymbol{\lambda}_{\rm gv}\right) d\boldsymbol{c}. \tag{83}$$

As the feature function $f_{\rm gv}(\boldsymbol{c})$ is non-linear (quadratic) and the experts are Gaussian, it is not possible to use the training algorithm for trajectory HMMs shown in Section II-B and no closed-form solution exists to calculate $Z_{\boldsymbol{q}}^{(\rm trj \cdot gv)}$. However, model parameters can be updated iteratively by contrastive divergence learning. The first partial derivatives of the unnormalized log likelihood with respect to the model parameters

[9] $\alpha_{\rm gv}$ is not required for the PoE framework, but was used for better initialization at the training stage in the experiment reported in Section V-C.

are given as

$$\frac{\partial \tilde{\mathcal{L}}(\boldsymbol{c}\,;\,\boldsymbol{q},\boldsymbol{\lambda},\boldsymbol{\lambda}_{\rm gv})}{\partial \boldsymbol{\mu}} = -\alpha_{\rm gv}\boldsymbol{S}_{\boldsymbol{q}}^{\top}\boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1}\boldsymbol{W}\left(\boldsymbol{c}-\bar{\boldsymbol{c}}_{\boldsymbol{q}}\right), \tag{84}$$

$$\begin{aligned}\frac{\partial \tilde{\mathcal{L}}(\boldsymbol{c}\,;\,\boldsymbol{q},\boldsymbol{\lambda},\boldsymbol{\lambda}_{\rm gv})}{\partial \boldsymbol{\phi}} &= \frac{\alpha_{\rm gv}}{2}\,\boldsymbol{S}_{\boldsymbol{q}}^{\top}\,\mathrm{diag}^{-1}\Big[\boldsymbol{W}\boldsymbol{P}_{\boldsymbol{q}}\boldsymbol{W}^{\top} \\ &\quad -\boldsymbol{W}\boldsymbol{c}\boldsymbol{c}^{\top}\boldsymbol{W}^{\top}+\boldsymbol{W}\bar{\boldsymbol{c}}_{\boldsymbol{q}}\bar{\boldsymbol{c}}_{\boldsymbol{q}}^{\top}\boldsymbol{W}^{\top} \\ &\quad +\boldsymbol{W}\left(\boldsymbol{c}-\bar{\boldsymbol{c}}_{\boldsymbol{q}}\right)\boldsymbol{\mu}_{\boldsymbol{q}}^{\top}+\boldsymbol{\mu}_{\boldsymbol{q}}\left(\boldsymbol{c}-\bar{\boldsymbol{c}}_{\boldsymbol{q}}\right)^{\top}\boldsymbol{W}^{\top}\Big],\end{aligned} \tag{85}$$

$$\frac{\partial \tilde{\mathcal{L}}(\boldsymbol{c}\,;\,\boldsymbol{q},\boldsymbol{\lambda},\boldsymbol{\lambda}_{\rm gv})}{\partial \mu_{\rm gv}} = -\frac{f_{\rm gv}(\boldsymbol{c})-\mu_{\rm gv}}{\sigma_{\rm gv}^2}, \tag{86}$$

$$\frac{\partial \tilde{\mathcal{L}}(\boldsymbol{c}\,;\,\boldsymbol{q},\boldsymbol{\lambda},\boldsymbol{\lambda}_{\rm gv})}{\partial\, 1/\sigma_{\rm gv}^2} = -\frac{1}{2\sigma_{\rm gv}^2}\left\{1-\frac{\left(f_{\rm gv}(\boldsymbol{c})-\mu_{\rm gv}\right)^2}{\sigma_{\rm gv}^2}\right\}, \tag{87}$$

where

$$\tilde{\mathcal{L}}(\boldsymbol{c}\,;\,\boldsymbol{q},\boldsymbol{\lambda},\boldsymbol{\lambda}_{\rm gv}) = \log\left\{\tilde{p}(\boldsymbol{c}\mid\boldsymbol{q},\boldsymbol{\lambda})\tilde{p}\left(f_{\rm gv}\left(\boldsymbol{c}\right)\mid\boldsymbol{\lambda}_{\rm gv}\right)\right\}. \tag{88}$$

Calculating the contrastive divergence (Eq. (66)) requires samples from the model distribution. However, sampling $\boldsymbol{c}$ from $p(\boldsymbol{c}\mid\boldsymbol{q},\boldsymbol{\lambda},\boldsymbol{\lambda}_{\rm gv})$ directly is difficult. Alternatively, the Metropolis-Hastings algorithm with a reasonable proposal distribution (*e.g.*, Gaussian approximation) or Hamiltonian Monte Carlo (also known as hybrid Monte Carlo, HMC) sampling [2], [18] can be used. Multiple AMs with non-Gaussian distributions [17], [21], [35] can also be estimated jointly with contrastive divergence learning.

C. Synthesis from PoEs

No modifications are required to generate speech parameters from estimated PoEs. As maximization is independent of the normalization constant Z,

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} p(\boldsymbol{c}\mid\{\boldsymbol{\lambda}_j\}_{j=1}^{M}) \tag{89}$$

$$= \arg\max_{\boldsymbol{c}} \frac{1}{Z}\prod_{j=1}^{M}\{p(f_j(\boldsymbol{c})\mid\boldsymbol{\lambda}_j)\}^{\alpha_j} \tag{90}$$

$$= \arg\max_{\boldsymbol{c}} \prod_{j=1}^{M}\{p(f_j(\boldsymbol{c})\mid\boldsymbol{\lambda}_j)\}^{\alpha_j} \tag{91}$$

$$= \arg\max_{\boldsymbol{c}} \sum_{j=1}^{M}\alpha_j \log p(f_j(\boldsymbol{c})\mid\boldsymbol{\lambda}_j). \tag{92}$$

It can be seen from Eqs. (40) and (91) that speech parameter generation from PoEs is identical to generating the speech parameter trajectory from the multiple AMs.

Training multiple AMs as a PoE has a greater computational load compared with the conventional independent training of multiple AMs with optimized weights. However, the computational cost for synthesis from PoEs is identical to that of the conventional approach. This property nicely fits the real scenario of speech synthesis as the training part can use large computational resources which may be limited at the synthesis stage.

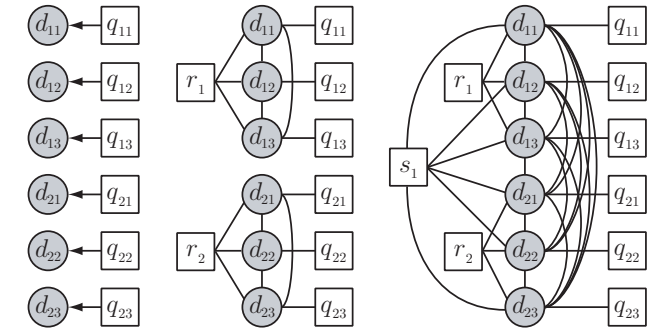

Fig. 2. Graphical model representation of (a) baseline, (b) PoE (state*phone), and (c) PoE (state*phone*syllable) duration models. In this figure, d_{ij}, q_{ij}, r_i, and s_k correspond to the state duration of state i of phone j, state-duration distribution of state i of phone j, phone-duration distribution of phone j, and syllable-duration distribution of syllable k. Note that the phones and the syllable here are assumed to consist of 3 states and 2 phones, respectively.

V. Experiments

A. Experimental conditions

Speech data from a female and a male professional speakers were used for training two speaker-dependent statistical parametric speech synthesizers. The training data consisted of 1 100 US English sentences per each speaker. The speech analysis conditions and model topologies were similar to those used for the Nitech-HTS 2005 [42] system. The speech data was downsampled to 16 kHz sampling then 39-order mel-cepstral coefficients [5], fundamental frequency (F_0) values, and 23 Bark-scale band aperiodicities [36] were extracted at every 5 ms. The F_0 values of the recordings were automatically extracted using the voting method [37]. Five-state, left-to-right, no-skip hidden semi-Markov models (HSMMs) [46] were used.[10] After training the baseline systems, PoEs were estimated using the baseline systems as the initial models.

B. Multiple-level duration models as PoE

The first experiment investigated the effect of joint estimation for multiple-level duration models (state and phone [15] and state, phone, and syllable [21]). State, phone, and syllable durations were modeled by 1-dimensional Gaussian distributions. They were derived from the manually corrected phone boundaries. These were clustered by decision trees based on the minimum description length (MDL) criterion [25] in the same way as the state duration models [38]. Table I shows the numbers of distributions (leaf nodes) for the state, phone, and syllable duration models. These duration

TABLE I
Numbers of distributions in the state, phone, and syllable duration models.

Speaker	Number of distributions		
	State	Phone	Syllable
Female	2 280	602	619
Male	2 075	538	405

[10]The sub-word unit used here was phone.

models were then jointly estimated in the PoE framework. The graphical model representations of the baseline state duration model and multiple-level duration models are illustrated in Fig. 2. It can be seen from the figure that the multiple duration models have a more complex dependency structure than the baseline system.

TABLE II
ROOT MEAN SQUARE ERRORS (RMSES) OF DURATION PREDICTION BY BASELINE, CONVENTIONAL UNNORMALIZED POE AND PROPOSED NORMALIZED POE DURATION MODELS. st*ph AND st*ph*sy CORRESPOND TO THE PRODUCT OF STATE AND PHONE DURATION MODELS AND THE PRODUCT OF STATE, PHONE, AND SYLLABLE DURATION MODELS. "UPOE" DENOTES THE CONVENTIONAL UNNORMALIZED POE DURATION MODEL. THE SYSTEMS WHICH ACHIEVED STATISTICALLY SIGNIFICANT IMPROVEMENTS OVER THE BASELINE SYSTEM ARE IN THE BOLD FONT.

Speaker	Duration models	RMSE in ms (rel. imp. in %)		
		phone	syllable	pause
Female	baseline	28.2	48.0	161
	uPoE (st*ph)	**25.9** (8.16)	**45.1** (6.04)	**151** (6.21)
	uPoE (st*ph*sy)	**25.7** (8.86)	**44.5** (7.29)	**151** (6.21)
	PoE (st*ph)	**25.6** (9.22)	**43.8** (8.75)	**150** (6.83)
	PoE (st*ph*sy)	**25.3** (10.3)	**43.8** (8.75)	**150** (6.83)
Male	baseline	31.2	52.0	156
	uPoE (st*ph)	**28.6** (8.33)	**48.3** (7.12)	153 (1.92)
	uPoE (st*ph*sy)	**28.6** (8.33)	**48.1** (7.50)	153 (1.92)
	PoE (st*ph)	**28.3** (9.29)	**47.7** (8.27)	156 (0.0)
	PoE (st*ph*sy)	**28.1** (9.94)	**47.6** (8.46)	156 (0.0)

Table II shows the duration prediction results. The duration prediction accuracy was evaluated on an evaluation set (100 sentences) which were not contained in the training set. Note that uPoE and PoE in the table correspond to the conventional unnormalized and the proposed normalized PoE duration models.[11] These uPoE systems use the standard independent training of the "experts" with the weights optimized to minimize RMSEs (of phone) over the development set (100 sentences) which were contained in neither the training nor test sets. The weight of the phone duration models for uPoE (st*ph) was 1.3 for the female speaker and 1.1 for the male speaker. The weights of phone and syllable duration models for uPoE (st*ph*sy) were 1.3 and 0.4 for the female speaker, respectively, and 1.1 and 0.4 for the male speaker, respectively. It can be seen from the table that the proposed PoE systems achieved significant error reductions over the baseline systems and comparable performance to the conventional uPoE systems, without requiring the use of the development set for weight tuning.

A paired-comparison preference listening test was also conducted. This test compared the naturalness of the synthesized speech generated from the baseline, conventional unnormalized PoE, and proposed normalized PoE duration models for the 100 evaluation sentences. The uPoE and PoE duration models were combinations of state, phone, and syllable duration models. The same model was used for generating spectra, F_0 values, and aperiodicities with these duration models. To see the effect of changing the speaking rate of the synthesized speech, normal (the most likely durations predicted by these state duration models), fast (0.75 $\times$ total number of frames in normal speech), and slow (1.25 $\times$ total number of frames in normal speech) speech samples were synthesized with these duration models. The technique to predict state durations given the total number of frames with full covariance duration models [16] was used to control the speaking rate for the PoE duration models. The listening tests were carried out using Amazon Mechanical Turk (http://www.mturk.com/). To ensure that pairs of speech samples were played equally often in AB as in BA order, both orders were regarded as different pairs. Thus there were 2×100 evaluation pairs in the test. One subject could evaluate a maximum of 40 pairs, they were randomly chosen and presented for each subject. Each pair was evaluated by two subjects. After listening to each pair of samples, the subjects were asked to choose their preferred one. Note that the subjects could select "No preference" if they had no preference.

[11]The normalized PoEs were estimated so as to maximize the normalized log likelihood given training data. On the other hand, the unnormalized PoEs were trained so as to maximize the unnormalized log likelihood given data.

TABLE III
PREFERENCE SCORES (%) AMONG SPEECH SAMPLES SYNTHESIZED FROM THE BASELINE, UNNORMALIZED POE (UPOE), AND NORMALIZE POE (POE). NOTE THAT "N/P" DENOTES "NO PREFERENCE". THE SYSTEMS WHICH ACHIEVED SIGNIFICANTLY BETTER PREFERENCE AT $p < 0.05$ LEVEL ARE IN THE BOLD FONT.

Speaker	Speaking rate	Preference score				p (t-test)
		baseline	uPoE	PoE	N/P	
Female	fast	30.0	**39.5**	–	30.5	0.032
		25.3	–	**41.9**	32.8	0.001
		–	29.4	34.0	36.6	0.190
	normal	35.8	35.0	–	29.2	0.438
		31.1	–	38.8	30.1	0.068
		–	31.1	33.0	35.9	0.356
	slow	32.4	**41.1**	–	26.5	0.045
		28.2	–	**45.4**	26.4	$<$0.001
		–	32.6	39.8	27.6	0.086
Male	fast	27.6	**42.4**	–	30.0	0.002
		31.0	–	**41.5**	27.5	0.021
		–	33.0	34.0	33.0	0.418
	normal	31.4	**42.4**	–	26.2	0.018
		28.6	–	**43.7**	27.7	0.002
		–	33.2	34.5	32.3	0.398
	slow	23.8	**47.6**	–	28.6	$<$0.001
		28.5	–	**46.3**	25.2	$<$0.001
		–	30.6	35.6	33.8	0.164

Table III shows the preference test results. Note that uPoE and PoE in the table correspond to the conventional unnormalized and the proposed normalized PoE duration models. It can be seen from the table that both the unnormalized and normalized PoE duration models were preferred to the baseline state duration models if the speaking rate was modified. However, the differences between the baseline and combined duration models were not significant without speaking-rate modification for the female speaker. The use of phone and syllable-level duration models can provide information about correlations of the state durations across states within phones and syllables, respectively. This information can help to predict more realistic state durations when the speaking rate is modified, because this information is incorporated while determining the state durations [16]. On the other hand, this information is not required to predict state durations if the speaking rate is not modified. Thus the use of the higher-level duration models did not give statistically significant improvements over the baseline system. Although there was no statistically significant

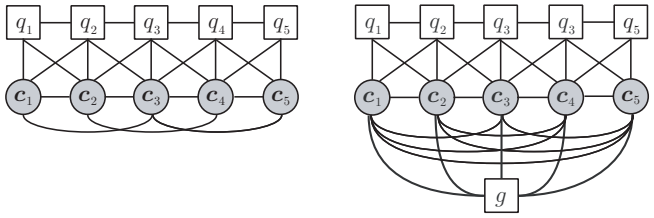

Fig. 3. Graphical model representation of (a) trajectory HMM and (b) PoE (trajectory HMM*GV) models ($T = 5$).

difference between the unnormalized and normalized PoE duration models, there was a slight consistent preference to the normalized PoE.

C. Speech parameter generation including global variance term as PoE

The second experiment investigated the effect of joint estimation of trajectory HMMs and GV Gaussian distributions. The graphical model representations of (a) a trajectory HMM and (b) a PoE with a trajectory HMM and GV distribution are illustrated in Fig. 3. Contrastive divergence learning was applied to update the PoE of speech parameter generation including the GV term. Instead of using the entire database at each iteration of contrastive divergence learning, the data was split into two batches of 550 utterances each, and only the data from one batch used at each iteration. In this experiment, 10 000 stochastic gradient iterations were used, each performing a contrastive divergence learning step with 10 MCMC iterations ($V = 10$). The learning rate was started from $\eta = 0.01$ and annealed (halved) at every 2 000 iterations. To improve the learning speed, the momentum method was used [23]. The parameter updates at iteration j, $\{\nabla \boldsymbol{\lambda}_k^{(j)}\}$, were supplemented by its momentum term, $\{0.9 \nabla \boldsymbol{\lambda}_k^{(j-1)}\}$. The inclusion of a momentum term has been found to increase the rate of convergence dramatically [23]. To draw samples from $p(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\lambda}_{\mathrm{gv}})$, HMC sampling [2], [18] with 20 leap-frog steps was used. The leap step was fixed to 0.001. The context-dependent logarithmic GV without silence [37] rather than standard, context-independent linear GV [29] was used in this experiment. α_{gv} was set utterance-length adaptively ($\alpha_{\mathrm{gv}} = 1/3T$) as suggested in [29]. Contrastive divergence learning was applied to the spectral part of the model parameters, *i.e.*, the model parameters for $\log F_0$ and band aperiodicities were not updated as the effect of GV was small for these speech parameters.

Initializing the MCMC sampler at the data point, which is a typical setting used in contrastive divergence learning, may not always work well for training the PoE for speech parameter generation including the GV term. This is because the feature function of this PoE is highly non-linear and its model distribution may have multiple modes. It is known that contrastive divergence learning does not work well if the model distribution has multiple modes and these modes are separated by low-probability regions. One way to address this problem is to give advance knowledge of the location of these modes to the MCMC sampler [11]. Based on a similar idea, here the MCMC sampler was initialized at the trajectory determined by speech parameter generation including the GV term. This trajectory is in a local optimum (mode) of $p(\boldsymbol{c} \mid \boldsymbol{q}, \boldsymbol{\lambda}, \boldsymbol{\lambda}_{\mathrm{gv}})$ and is the particular mode of interest. Thus initializing the MCMC sampler by this trajectory sounds feasible for training this PoE.[12]

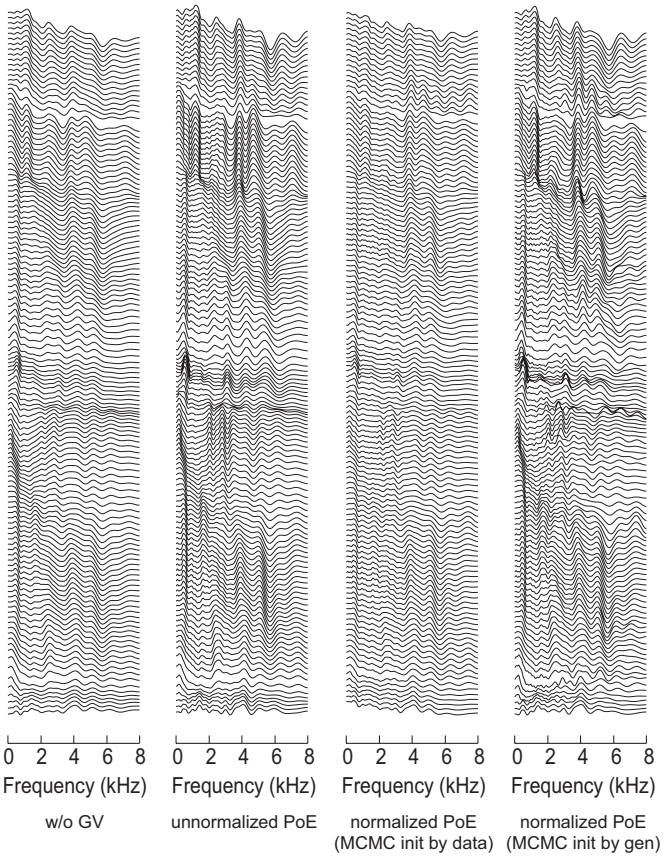

Fig. 4. Generated spectra from trajectory HMMs without GV, unnormalized PoE for speech parameter generation including the GV term, normalized PoE estimated by contrastive divergence learning with MCMC initialized by data points, and normalized PoE estimated by contrastive divergence learning with MCMC initialized by generated trajectory.

Figure 4 plots the generated spectra from the estimated PoEs with different initialization of the MCMC sampler. It can be seen from the figure that initializing the MCMC sampler by data points removed the effect of GV and the generated spectra became flatter than the conventional unnormalized PoE for speech parameter generation including the GV term. On the other hand, the formant structure of spectra generated from the estimated PoE with generated trajectory-based initialization looks clearer than that of the conventional unnormalized PoE one. From this result, the PoE with generated trajectory-based initialization was used in the following experiment.

A paired-comparison preference listening test was conducted. This test compared the naturalness of the synthesized speech generated from the conventional unnormalized PoE and proposed normalized PoE for speech parameter generation

[12] A similar idea to use the most probable samples for training MRFs with contrastive learning has been proposed in the machine learning area [31].

including the GV term over 100 evaluation sentences. The listening test conditions were the same as those in the previous section except that each pair was evaluated by three subjects rather than two subjects.

TABLE IV
PREFERENCE SCORES (%) BETWEEN THE CONVENTIONAL UNNORMALIZED PoE AND PROPOSED NORMALIZED PoEs FOR SPEECH PARAMETER GENERATION INCLUDING THE GV TERM. N/P DENOTES "NO PREFERENCE". THE SYSTEMS WHICH ACHIEVED SIGNIFICANTLY BETTER PREFERENCE AT $p < 0.05$ LEVEL ARE IN THE BOLD FONT.

	Preference score			p
Speaker	uPoE	PoE	N/P	(t-test)
Female	25.9	**44.8**	29.3	$<$0.001
Male	29.2	**49.2**	21.6	$<$0.001

Table IV shows the preference test result. Note that uPoE and PoE in the table correspond to the conventional unnormalized and the proposed normalized PoEs for speech parameter generation including the GV term. It can be seen from the table that the proposed normalized PoEs for speech parameter generation including the GV term achieved a significantly better preference score than the conventional unnormalized PoE one.

VI. CONCLUSIONS

To achieve high quality speech synthesis multiple statistical models, trained at different levels, are often combined together. Each of these models is normally trained individually. At synthesis time, the likelihood contribution from each of the models is weighted, and the most likely trajectory from this combined distribution used for synthesis. This article has shown that this process can be described within a product of experts framework. For Gaussian experts trained on linear transforms of the underlying features, closed-form solutions for the estimation of the mean parameters, and a gradient ascent based approach for estimation of the variance parameters are detailed. For more general experts, either using non-Gaussian distributions, or non-linear transformations of the features, a contrastive divergence based training scheme is described. Training all the experts together allows the contribution of each expert to be derived within the training process (via the model variance) rather than relying on a separately tuned set of weights.

Training multi-level models in this product of experts framework was evaluated for both linear Gaussian experts, duration modelling, and non-linear experts, the incorporation of a global variance model. The joint training of a global variance expert and trajectory model yielded statistically significant preference scores over the standard individual training of the models. For duration modelling, a slight preference, not significant, for the jointly trained models was observed.

This article has described a general approach for training and combining multiple models for statistical parametric speech synthesis within the product of experts framework. The scheme can be applied to a wide-range of experts within the statistical parametric speech synthesis domain. Using the consistent joint training of multiple experts will be more important as the diversities in the experts increases.

Future work includes investigation of other feature functions (*e.g.*, segmental features [20]) and/or distributions (*e.g.*, Student's t distribution or "unigauss" distribution [9]) for experts and updating window coefficients of each experts [3], within the proposed framework.

REFERENCES

[1] S. Airey, "Products of Gaussians," MPhil thesis, University of Cambridge, 2002.
[2] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
[3] L. Chen, Y. Nankaku, H. Zen, K. Tokuda, Z. Ling, and L. Dai, "Estimation of window coefficients for dynamic feature extraction for HMM-based speech synthesis," in *Proc. Interspeech*, 2011, (to appear).
[4] J. Dines and S. Sridharan, "Trainable speech synthesis with trended hidden Markov models," in *Proc. ICASSP*, 2001, pp. 833–837.
[5] T. Fukada, K. Tokuda, T. Kobayashi, and S. Imai, "An adaptive algorithm for mel-cepstral analysis of speech," in *Proc. ICASSP*, 1992, pp. 137–140.
[6] S. Furui, "Speaker independent isolated word recognition using dynamic features of speech spectrum," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 34, pp. 52–59, 1986.
[7] M. Gales and S. Airey, "Product of Gaussians for speech recognition," *Comput. Speech Lang.*, vol. 20, no. 1, pp. 22–40, 2006.
[8] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
[9] G. Hinton, "Product of experts," in *Proc. ICANN*, vol. 1, 1999, pp. 1–6.
[10] ——, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.
[11] G. Hinton, M. Welling, and A. Mnih, "Wormholes improve contrastive divergence," in *Proc. NIPS*, 2003, pp. 417–424.
[12] R. Jacobs, M. Jordan, S. Nowlan, and G. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.
[13] R. Kindermann and J. Snell, *Markov Random Fields and Their Applications*. Americal Mathematical Society, 1980.
[14] J. Latorre and M. Akamine, "Multilevel parametric-base F0 model for speech synthesis," in *Proc. Interspeech*, 2008, pp. 2274–2277.
[15] Z. Ling, Y. Wu, Y. Wang, L. Qin, and R. Wang, "USTC system for Blizzard Challenge 2006 an improved HMM-based speech synthesis method," in *Proc. Blizzard Challenge Workshop*, 2006.
[16] H. Lu, Y. Wu, K. Tokuda, L. Dai, and R. Wang, "Full covariance state duration modeling for HMM-based speech synthesis," in *Proc. ICASSP*, 2009, pp. 4033–4036.
[17] K. Nagao, H. Zen, Y. Nankaku, and K. Tokuda, "Investigation of global variance modeling for HMM-based speech synthesis," in *Proc. Sprint Meeting of ASJ*, 2009, pp. 427–428.
[18] R. Neal, "Probabilistic inference using Markov chain Monte Carlo methods," University of Tronto, Tech. Rep. CRG-TR-93-1, 1993.
[19] J. Odell, "The use of context in large vocabulary speech recognition," Ph.D. dissertation, Cambridge University, 1995.
[20] M. Ostendorf, V. Digalakis, and O. Kimball, "From HMMs to segment models," *IEEE Trans. Speech Audio Process.*, vol. 4, no. 5, pp. 360–378, 1996.
[21] Y. Qian, Z. Wu, B. Gao, and F. Soong, "Improved prosody generation by maximizing joint probability of state and longer units," *IEEE Trans. Acoust. Speech Lang. Process.*, vol. 19, no. 6, pp. 1702–1710, 2011.
[22] H. Rue and L. Held, *Gaussian Markov Random Fields: Theory and Applications*. Chapman & Hall/CRC, 2005.
[23] D. Rumelhart and J. McCelland, *Parallel distributed processing*. MIT Press, 1986.
[24] M. Shannon and W. Byrne, "Autoregressive HMMs for speech synthesis," in *Proc. Interspeech*, 2009, pp. 400–403.
[25] K. Shinoda and T. Watanabe, "Acoustic modeling based on the MDL criterion for speech recognition," in *Proc. Eurospeech*, 1997, pp. 99–102.
[26] K. Sim and M. Gales, "Basis superposition precision matrix modelling for large vocabulary continuous speech recognition," in *Proc. ICASSP*, 2004, pp. 801–804.
[27] J. Sun, F. Ding, and Y. Wu, "Polynomial segment model based statistical parametric speech synthesis system," in *Proc. ICASSP*, 2009, pp. 4021–4024.
[28] Y. Teh, M. Welling, S. Osindero, and G. Hinton, "Energy-based models for sparse overcomplete representations," *The Journal of Machine Learning Research*, vol. 4, 2004.
[29] T. Toda and K. Tokuda, "A speech parameter generation algorithm considering global variance for HMM-based speech synthesis," *IEICE Trans. Inf. Syst.*, vol. E90-D, no. 5, pp. 816–824, 2007.

[30] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," in *Proc. ICASSP*, 2000, pp. 1315–1318.
[31] D. Vickrey, C. Lin, and D. Koller, "Non-local contrastive objectives," in *Proc. ICML*, 2010, pp. 1103–1110.
[32] C. Wang, Z. Ling, B. Zhang, and L. Dai, "Multi-layer F0 modeling for HMM-based speech synthesis," in *Proc. ISCSLP*, 2008, pp. 129–132.
[33] M. Welling, "Products of experts," ScholarPedia http://www.scholarpedia.org/article/Product_of_experts, 2007.
[34] C. Williams, "How to pretend that correlated variables are independent by using difference observations," *Neural Computation*, vol. 17, no. 1, pp. 1–6, 2005.
[35] J. Yamagishi, T. Nose, H. Zen, T. Toda, and K. Tokuda, "Performance evaluation of the speaker-independent HMM-based speech synthesis system 'HTS-2007' for the Blizzard Challenge 2007," in *Proc. ICASSP*, 2008, pp. 3957–3960.
[36] J. Yamagishi and O. Watts, "The CSTR/EMIME HTS system for Blizzard Challenge," in *Proc. Blizzard Challenge Workshop*, 2010.
[37] J. Yamagishi, H. Zen, Y. Wu, T. Toda, and K. Tokuda, "The HTS-2008 system: Yet another evaluation of the speaker-adaptive HMM-based speech synthesis system in the 2008 Blizzard Challenge," in *Proc. Blizzard Challenge Workshop*, 2008.
[38] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis," in *Proc. Eurospeech*, 1999, pp. 2347–2350.
[39] H. Zen, M. Gales, Y. Nankaku, and K. Tokuda, "Statistical parametric speech synthesis based on product of experts," in *Proc. ICASSP*, 2010, pp. 4242–4245.
[40] H. Zen, Y. Nankaku, and K. Tokuda, "Model-space MLLR for trajectory HMMs," in *Proc. Interspeech*, 2007, pp. 2065–2068.
[41] H. Zen, Y. Nankaku, K. Tokuda, and T. Kitamura, "Speaker adaptation of trajectory HMMs using feature-space MLLR," in *Proc. Interspeech*, 2006, pp. 2274–2277.
[42] H. Zen, T. Toda, M. Nakamura, and T. Tokuda, "Details of the Nitech HMM-based speech synthesis system for the Blizzard Challenge 2005," *IEICE Trans. Inf. Syst.*, vol. E90-D, no. 1, pp. 325–333, 2007.
[43] H. Zen, K. Tokuda, and A. Black, "Statistical parametric speech synthesis," *Speech Commun.*, vol. 51, no. 11, pp. 1039–1064, 2009.
[44] H. Zen, K. Tokuda, and T. Kitamura, "Estimating trajectory HMM parameters by Monte Carlo EM with Gibbs sampler," in *Proc. ICASSP*, 2006, pp. 1173–1176.
[45] ——, "Reformulating the HMM as a trajectory model by imposing explicit relationships between static and dynamic features," *Comput. Speech Lang.*, vol. 21, no. 1, pp. 153–173, 2007.
[46] H. Zen, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "A hidden semi-Markov model-based speech synthesis system," *IEICE Trans. Inf. Syst.*, vol. E90-D, no. 5, pp. 825–834, 2007.

PLACE PHOTO HERE

Heiga Zen received the A.E. degree from the Suzuka National College of Technology, Japan, in 1999, and the B.E., M.E., and Ph.D. degrees from Nagoya Institute of Technology, Japan, in 2001, 2003, and 2006, respectively. From 2004–2005, he was an intern/co-op researcher at the IBM T. J. Watson Research Center, NY. From 2006–2008, he was a Research Associate at Nagoya Institute of Technology. He has been a Research Engineer at Toshiba Research Europe Ltd. Cambridge Research Lab, UK, since 2008. His research interests include statistical speech recognition and synthesis. Dr. Zen was awarded a 2006 ASJ Awaya Award, a 2008 ASJ Itakura Award, a 2008 TAF TELECOM System Technology Award, a 2008 IEICE Information and Systems Society Best Paper Award, and a 2009 IPSJ Yamashita SIG Research Award.

PLACE PHOTO HERE

Mark J. F. Gales studied for the B.A. in Electrical and Information Sciences at the University of Cambridge from 1985–88. Following graduation he worked as a consultant at Roke Manor Research Ltd. In 1991 he took up a position as a Research Associate in the Speech Vision and Robotics group in the Engineering Department at Cambridge University. In 1995 he completed his doctoral thesis: Model-Based Techniques for Robust Speech Recognition supervised by Professor Steve Young. From 1995–1997 he was a Research Fellow at Emmanuel College Cambridge. He was then a Research Staff Member in the Speech group at the IBM T. J. Watson Research Center until 1999 when he returned to Cambridge University Engineering Department as a University Lecturer. He is currently a Reader in Information Engineering and a Fellow of Emmanuel College. Mark Gales is a Fellow of the IEEE and was a member of the Speech Technical Committee from 2001–2004. He is currently an associate editor for IEEE Signal Processing Letters and IEEE Transactions on Audio Speech and Language Processing. He is also on the Editorial Board of Computer Speech and Language. Mark Gales was awarded a 1997 IEEE Young Author Paper Award for his paper on Parallel Model Combination and a 2002 IEEE Paper Award for his paper on Semi-Tied Covariance Matrices.

PLACE PHOTO HERE

Yoshihiko Nankaku received the B.E. degree in Computer Science, and the M.E., and Ph.D. degrees in the Department of Electrical and Electronic Engineering from the Nagoya Institute of Technology, Japan, in 1999, 2001, and 2004 respectively. After a year as a postdoctoral fellow at the Nagoya Institute of Technology, he is currently an Assistant Professor at the same Institute. His research interests include statistical machine learning, speech recognition, speech synthesis, image recognition, and multimodal interface.

PLACE PHOTO HERE

Keiichi Tokuda received the B.E. degree in electrical and electronic engineering from the Nagoya Institute of Technology, Japan, in 1984 and the M.E. and Dr.Eng. degrees in information processing from the Tokyo Institute of Technology, Japan, in 1986 and 1989, respectively. From 1989–1996, he was a Research Associate in the Department of Electronic and Electric Engineering, Tokyo Institute of Technology. From 1996–2004, he was an Associate Professor in the Department of Computer Science, Nagoya Institute of Technology, where he is currently a Professor. He is also an Invited Researcher at the National Institute of Information and Communications Technology (NICT), formally known as the ATR Spoken Language Communication Research Laboratories, Japan from 2000, and was a Visiting Researcher at Carnegie Mellon University, PA, from 2001–2002. In 2005, Dr. Alan Black (CMU) and Keiichi Tokuda organized the largest ever evaluation of corpus-based speech synthesis techniques, the Blizzard Challenge, which has progressed to an annual event. He was a member of the Speech Technical Committee of the IEEE Signal Processing Society from 2000–2003, and acts as organizer and reviewer for many major speech conferences, workshops and journals. He published over 70 journal papers and over 160 conference papers, and received 5 paper awards.