Globally Normalized Model for Statistical Speech Synthesis

Heiga ZEN

Toshiba Research Euorpe Ltd.
Cambridge Research Lab.

Speech Synthesis Seminar Series @ CUED, Cambridge, UK
January 26th, 2011

Outline

Trajectory HMM

- Speech parameter generation
- Derivation of trajectory HMM
- Relationship between parameter generation & trajectory HMM
- Trajectory HMM as globally normalized model
- Parameter estimation

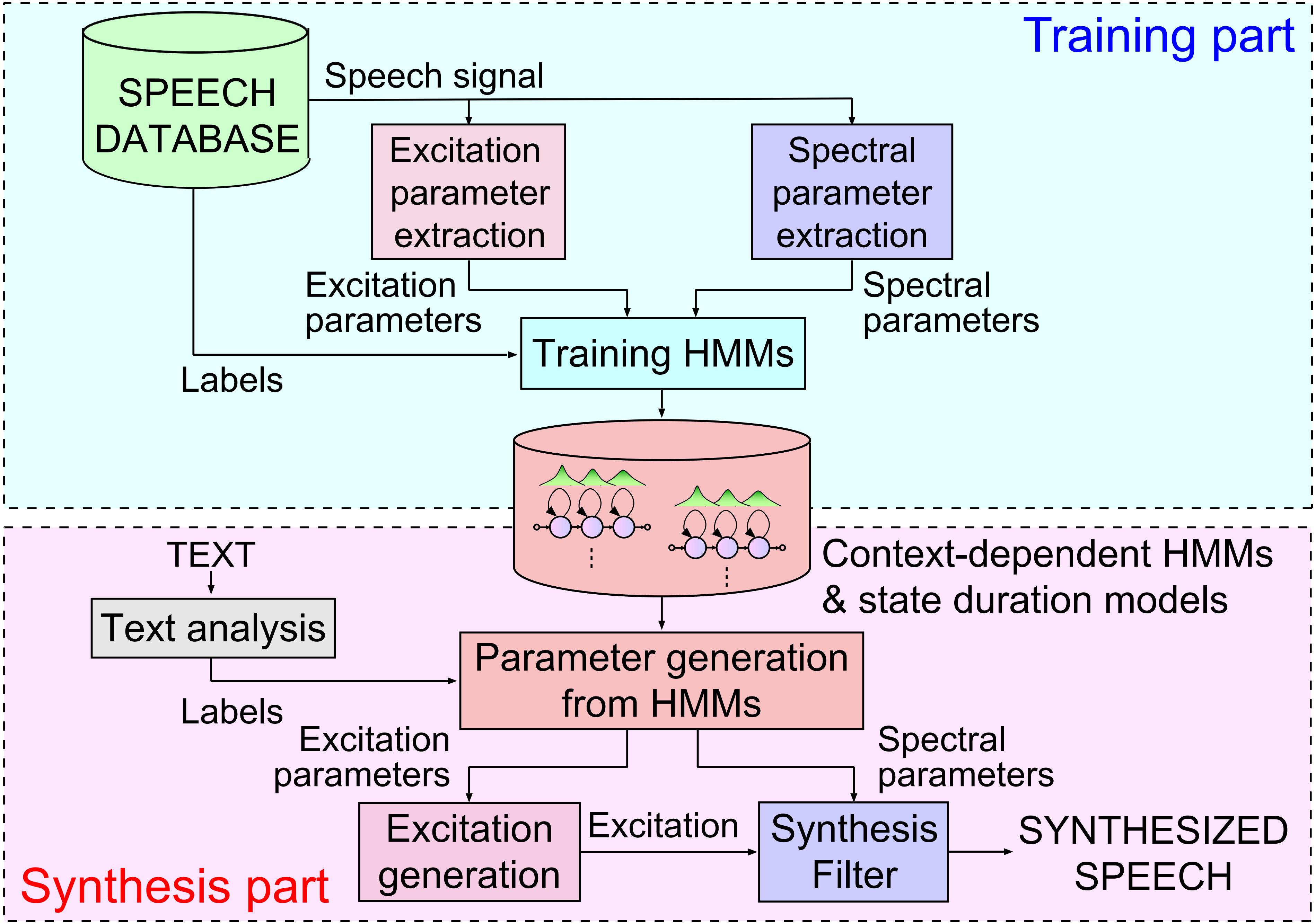
Min generation error (MGE) training & trajectory HMM

- Relationship
- Properties of MGE

(If time remains) Product of Experts (PoE)

- Combination of multiple AMs as PoE
- PoE & trajectory HMM

HMM-based speech synthesis system



Speech parameter generation algorithm

Determine a speech parameter vector sequence that maximizes its output probability given label l & HMM λ

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

$$= \arg\max_{\boldsymbol{o}} \sum_{\forall \boldsymbol{q}} p(\boldsymbol{o} \mid \boldsymbol{q}, \hat{\lambda}) p(\boldsymbol{q} \mid \boldsymbol{l}, \hat{\lambda})$$

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

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

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

Output prob of o given l & HMM λ

$$p(\boldsymbol{o} \mid \boldsymbol{l}, \lambda) = \sum_{\forall \boldsymbol{q}} \underbrace{p(\boldsymbol{o} \mid \boldsymbol{q}, \lambda)}_{\text{state-output}} \underbrace{P(\boldsymbol{q} \mid \boldsymbol{l}, \lambda)}_{\text{state-transition}}$$

$$p(\boldsymbol{o} \mid \boldsymbol{q}, \lambda) = \prod_{t=1}^{T} \mathcal{N}(\boldsymbol{o}_t \,;\, \boldsymbol{\mu}_{q_t}, \boldsymbol{\Sigma}_{q_t}) \quad \Leftarrow \text{single Gaussian}$$

$$= \mathcal{N}\left(\begin{bmatrix} \boldsymbol{o}_1 \\ \boldsymbol{o}_2 \\ \vdots \\ \boldsymbol{o}_T \end{bmatrix} ; \begin{bmatrix} \boldsymbol{\mu}_{q_1} \\ \boldsymbol{\mu}_{q_2} \\ \vdots \\ \boldsymbol{\mu}_{q_T} \end{bmatrix} , \underbrace{\begin{bmatrix} \boldsymbol{\Sigma}_{q_1} & & & 0 \\ & \boldsymbol{\Sigma}_{q_2} & & \\ & & \ddots & \\ 0 & & & \boldsymbol{\Sigma}_{q_T} \end{bmatrix}}_{\text{diagonal}} \right)$$

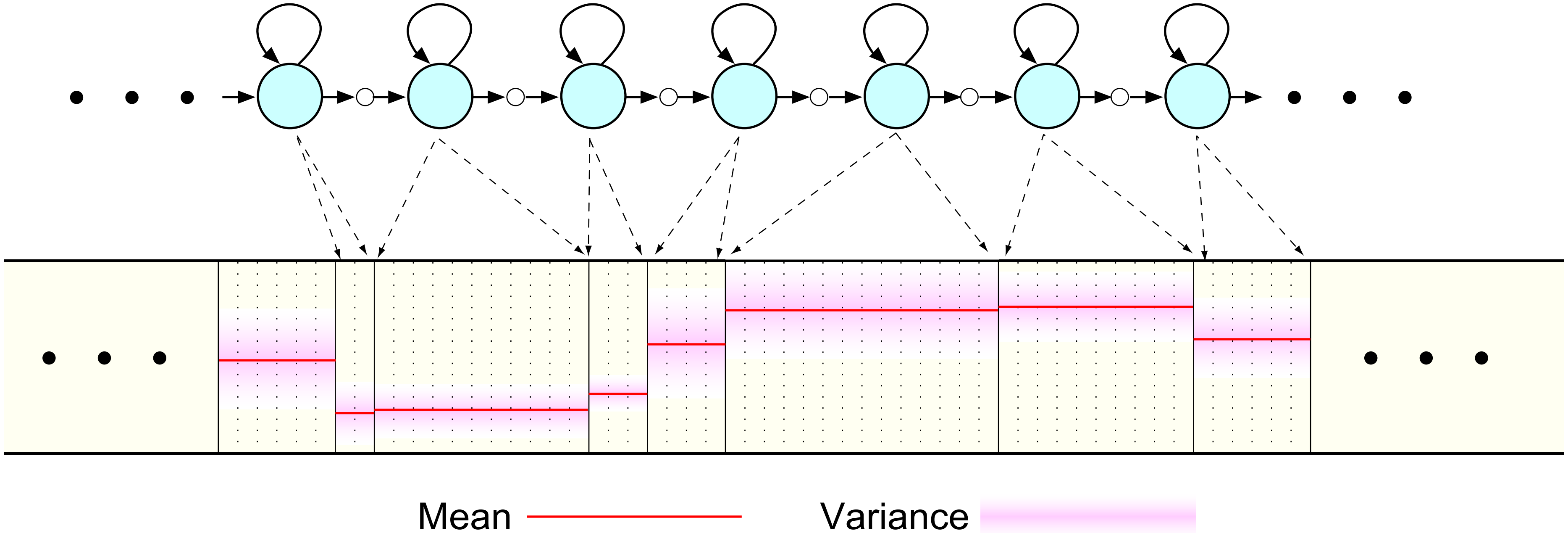
$$= \mathcal{N}(\boldsymbol{o} \,;\, \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}})$$

Generated trajectory

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

$$= \arg\max_{\boldsymbol{o}} \mathcal{N}(\boldsymbol{o}\,;\ \boldsymbol{\mu}_{\hat{\boldsymbol{q}}}, \boldsymbol{\Sigma}_{\hat{\boldsymbol{q}}})$$

$$= \boldsymbol{\mu}_{\hat{\boldsymbol{q}}} \quad \Leftarrow$$ **mean vector sequence**



Relationship between $\boldsymbol{o}$ and $\boldsymbol{c}$

$$\boldsymbol{o}_t = \begin{bmatrix} \boldsymbol{c}_t \\ \Delta \boldsymbol{c}_t \end{bmatrix} \begin{array}{l} \Leftarrow \text{static} \\ \Leftarrow \text{dynamic} \end{array} \qquad \Delta \boldsymbol{c}_t = \boldsymbol{c}_t - \boldsymbol{c}_{t-1}$$

$$
\begin{array}{rc}
 & \boldsymbol{o} \\
\begin{array}{r} \\ \boldsymbol{o}_{t-1} \\ \\ \boldsymbol{o}_t \\ \\ \boldsymbol{o}_{t+1} \\ \\ \\ \end{array} &
\left[\begin{array}{c} \vdots \\ \hline \boldsymbol{c}_{t-1} \\ \Delta\boldsymbol{c}_{t-1} \\ \hline \boldsymbol{c}_t \\ \Delta \boldsymbol{c}_t \\ \hline \boldsymbol{c}_{t+1} \\ \Delta\boldsymbol{c}_{t+1} \\ \hline \vdots \end{array}\right]
\end{array}
=
\begin{array}{c}
\boldsymbol{W} \\
\left[\begin{array}{cccccc}
\cdots & \vdots & \vdots & \vdots & \vdots & \cdots \\ \hline
\cdots & 0 & \boldsymbol{I} & 0 & 0 & \cdots \\
\cdots & -\boldsymbol{I} & \boldsymbol{I} & 0 & 0 & \cdots \\ \hline
\cdots & 0 & 0 & \boldsymbol{I} & 0 & \cdots \\
\cdots & 0 & -\boldsymbol{I} & \boldsymbol{I} & 0 & \cdots \\ \hline
\cdots & 0 & 0 & 0 & \boldsymbol{I} & \cdots \\
\cdots & 0 & 0 & -\boldsymbol{I} & \boldsymbol{I} & \cdots \\ \hline
\cdots & \vdots & \vdots & \vdots & \vdots & \cdots
\end{array}\right]
\end{array}
\begin{array}{c}
\boldsymbol{c} \\
\left[\begin{array}{c} \vdots \\ \hline \boldsymbol{c}_{t-2} \\ \hline \boldsymbol{c}_{t-1} \\ \hline \boldsymbol{c}_t \\ \hline \boldsymbol{c}_{t+1} \\ \hline \vdots \end{array}\right]
\end{array}
$$

Speech parameter generation algorithm

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

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

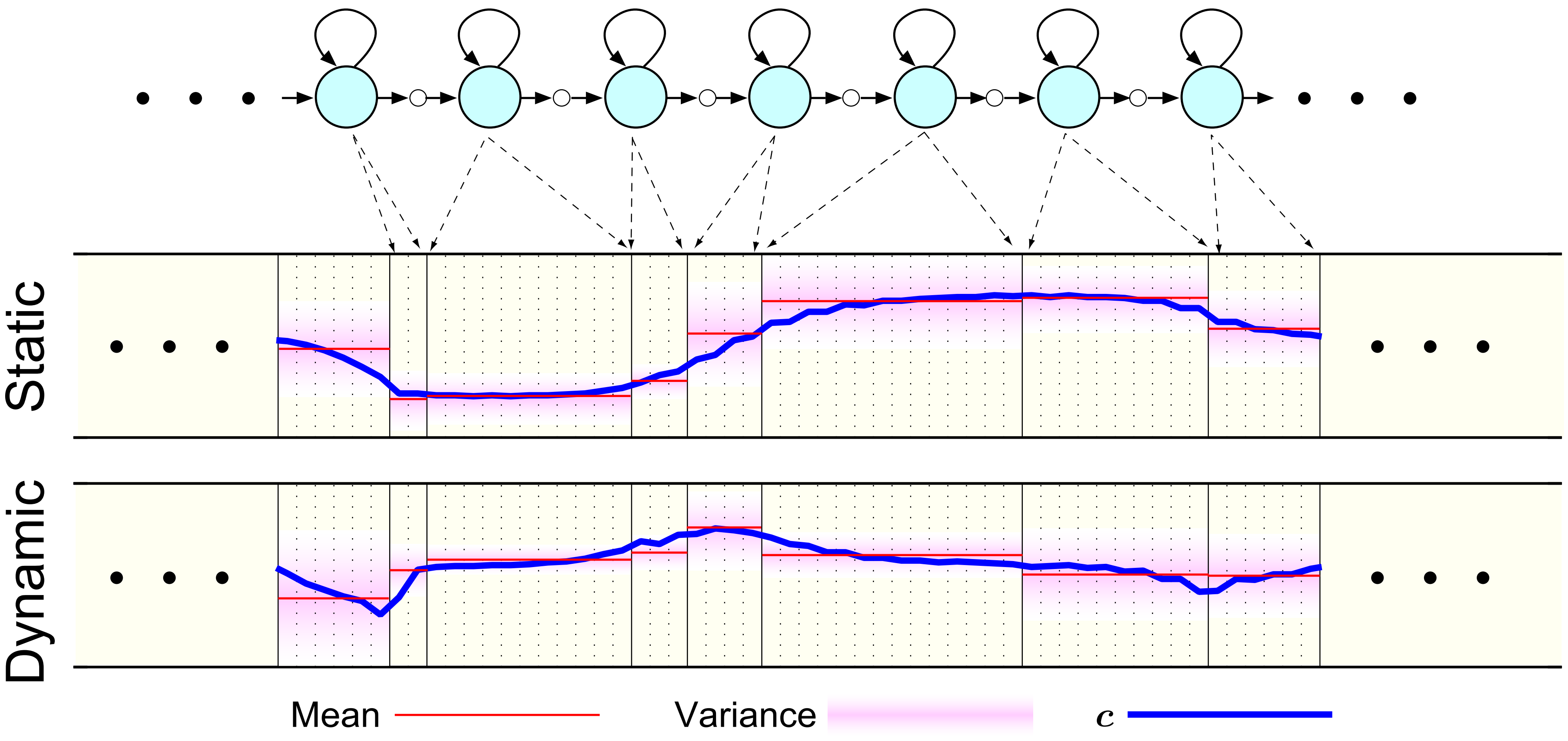
$$\Downarrow$$

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

$$\Downarrow$$

$$\boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{W} \hat{\boldsymbol{c}} = \boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{\mu}_{\boldsymbol{q}}$$

Generated trajectory



Inconsistency between training & synthesis

Training & synthesis parts are inconsistent

- **Training part**
  * Baum-Welch training
  * Labels are often given manually
  * Model training model w/o dynamic feature constraints
- **Synthesis part**
  * Viterbi (single-path) approximation
  * Labels are often given automatically (by text analysis)
  * Parameter generation w/ dynamic feature constraints

How about introducing dyn feature constraints to training?

Output prob of o given l & HMM λ

$$p(\boldsymbol{o} \mid \boldsymbol{l}, \lambda) = \sum_{\forall \boldsymbol{q}} \underbrace{p(\boldsymbol{o} \mid \boldsymbol{q}, \lambda)}_{\text{state-output}} \underbrace{P(\boldsymbol{q} \mid \boldsymbol{l}, \lambda)}_{\text{state-transition}}$$

$$p(\boldsymbol{o} \mid \boldsymbol{q}, \lambda) = \prod_{t=1}^{T} \mathcal{N}(\boldsymbol{o}_t \,;\, \boldsymbol{\mu}_{q_t}, \boldsymbol{\Sigma}_{q_t}) \quad \Leftarrow \text{single Gaussian}$$

$$= \mathcal{N}\left(\begin{bmatrix} \boldsymbol{o}_1 \\ \boldsymbol{o}_2 \\ \vdots \\ \boldsymbol{o}_T \end{bmatrix} ; \begin{bmatrix} \boldsymbol{\mu}_{q_1} \\ \boldsymbol{\mu}_{q_2} \\ \vdots \\ \boldsymbol{\mu}_{q_T} \end{bmatrix} , \underbrace{\begin{bmatrix} \boldsymbol{\Sigma}_{q_1} & & & 0 \\ & \boldsymbol{\Sigma}_{q_2} & & \\ & & \ddots & \\ 0 & & & \boldsymbol{\Sigma}_{q_T} \end{bmatrix}}_{\text{diagonal}} \right)$$

$$= \mathcal{N}(\boldsymbol{o} \,;\, \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}})$$

Inconsistency in HMM w/ dynamic features

Under $\boldsymbol{o} = \boldsymbol{W}\boldsymbol{c}$

$$p(\boldsymbol{o} \mid \boldsymbol{q}, \lambda) = \mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\ \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}})$$ $\Rightarrow$ **incorrect!**

Why?

$$\int_{\boldsymbol{c}} \mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\ \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}})d\boldsymbol{c} \neq 1$$ $\Rightarrow$ **integral over $\boldsymbol{c}$ must be 1 to be a *valid* PDF**

Why does this happen?

Static features $\Rightarrow$ random variables

Dynamic features $\Rightarrow$ Not random variables!!

Normalization

Normalized to achieve valid PDF

$$Z_{\boldsymbol{q}} = \int_{\boldsymbol{c}} \mathcal{N}\left(\boldsymbol{W}\boldsymbol{c}\,;\;\boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}\right) d\boldsymbol{c}$$

$$= \frac{\sqrt{(2\pi)^{MT}\left|\boldsymbol{P}_{\boldsymbol{q}}\right|}}{\sqrt{(2\pi)^{2MT}\left|\boldsymbol{\Sigma}_{\boldsymbol{q}}\right|}} \exp\left\{-\frac{1}{2}\left(\boldsymbol{\mu}_{\boldsymbol{q}}^{\top}\boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1}\boldsymbol{\mu}_{\boldsymbol{q}} - \boldsymbol{r}_{\boldsymbol{q}}^{\top}\boldsymbol{P}_{\boldsymbol{q}}\boldsymbol{r}_{\boldsymbol{q}}\right)\right\}$$

$\mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\;\boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}) \Rightarrow$ **invalid PDF!**

$\dfrac{1}{Z_{\boldsymbol{q}}}\mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\;\boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}) \Rightarrow$ **valid PDF!!**

Definition of trajectory HMM

Use normalized Gaussian ⇒ trajectory HMM is defined

$$p(\boldsymbol{c} \mid \boldsymbol{l}, \lambda) = \sum_{\forall \boldsymbol{q}} \underbrace{p(\boldsymbol{c} \mid \boldsymbol{q}, \lambda)}_{\text{state-output}} \underbrace{P(\boldsymbol{q} \mid \boldsymbol{l}, \lambda)}_{\text{state-transition}}$$

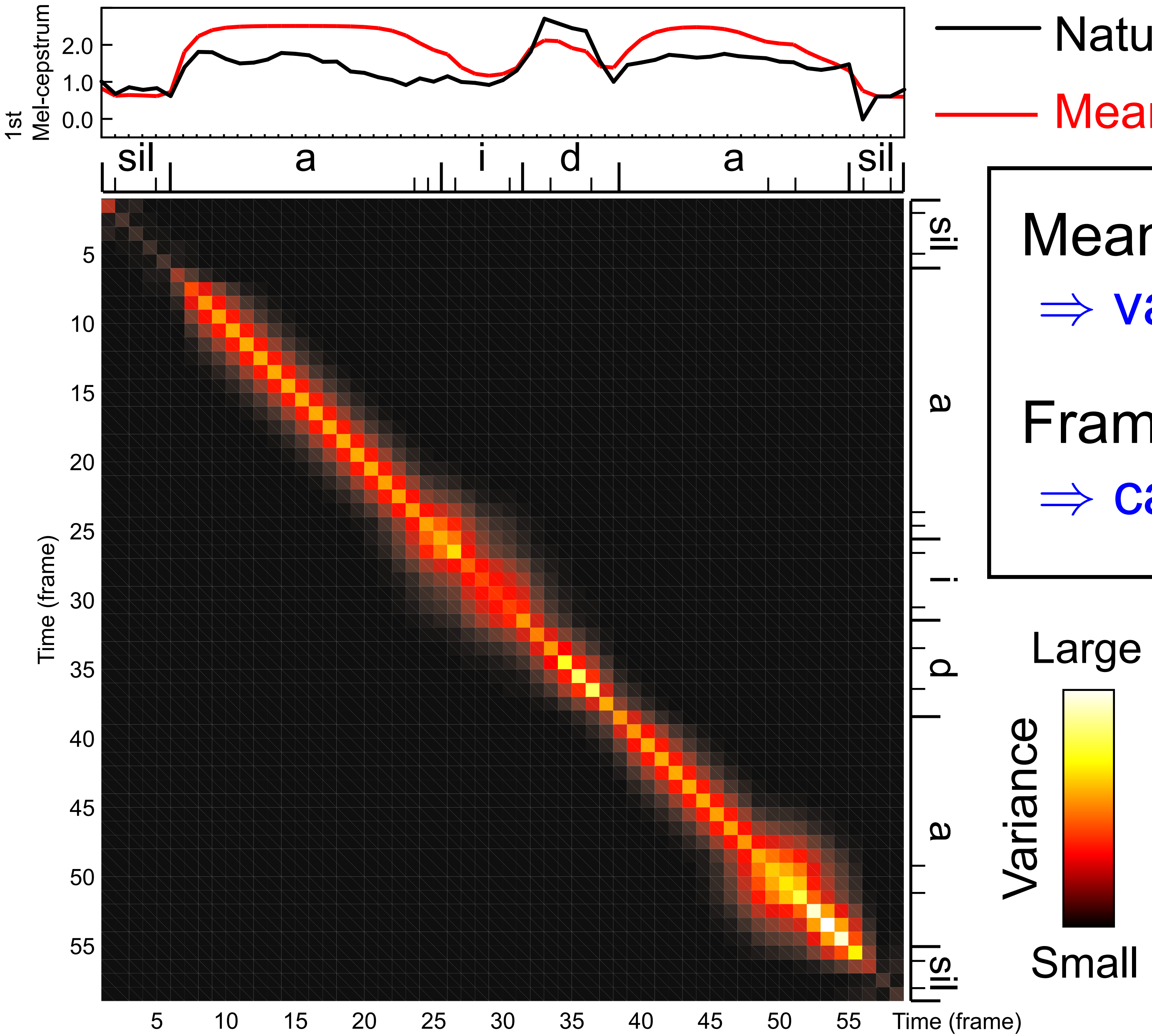
$$p(\boldsymbol{c} \mid \boldsymbol{q}, \lambda) = \frac{1}{Z_{\boldsymbol{q}}} \mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\ \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}) \Leftarrow$$ **normalized Gaussian**

$$= \mathcal{N}(\boldsymbol{c}\,;\ \underbrace{\bar{\boldsymbol{c}}_{\boldsymbol{q}}}_{\text{mean vec}}, \underbrace{\boldsymbol{P}_{\boldsymbol{q}}}_{\text{cov mat}}) \Leftarrow$$ **Gaussian over $\boldsymbol{c}$**

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

$$\boldsymbol{R}_{\boldsymbol{q}} = \boldsymbol{W}^{\top} \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{W} = \boldsymbol{P}_{\boldsymbol{q}}^{-1}$$

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



Mean & variance
$\Rightarrow$ varies in a state

Frame correlation
$\Rightarrow$ captured by P_q

Covariance matrix P_q

Trajectory HMM & speech parameter generation

Mean vector of trajectory HMM

$$\boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{W} \bar{\boldsymbol{c}}_{\boldsymbol{q}} = \boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{\mu}_{\boldsymbol{q}}$$

Trajectory by speech parameter generation algorithm

$$\boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{W} \hat{\boldsymbol{c}} = \boldsymbol{W}^\top \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{\mu}_{\boldsymbol{q}}$$

$\Rightarrow$ they are identical

Trajectory HMM as globally normalized model

HMM ⇒ locally (frame-level) normalized model

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

$$= \prod_{t=1}^{T} \mathcal{N}\left(\boldsymbol{o}_t \,;\, \boldsymbol{\mu}_{q_t}, \boldsymbol{\Sigma}_{q_t}\right)$$

Trajectory HMM ⇒ globally (utt-level) normalized model

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

$$= \frac{1}{Z_{\boldsymbol{q}}} \prod_{t=1}^{T} \mathcal{N}\left(\left[\boldsymbol{c}_t^{\top}, \Delta\, \boldsymbol{c}_t^{\top}\right]^{\top} \,;\, \boldsymbol{\mu}_{q_t}, \boldsymbol{\Sigma}_{q_t}\right)$$

Estimating trajectory HMM parameters

ML estimation of trajectory HMM

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

Locally normalized model

Parameter estimation for each state can be done separately

Globally normalized model

Parameter estimation of all states have to be done jointly

$$\boldsymbol{\mu} = \left[\boldsymbol{\mu}_1^\top, \boldsymbol{\mu}_2^\top, \ldots, \boldsymbol{\mu}_N^\top\right]^\top$$: all mean vectors

$$\boldsymbol{\phi} = \left[\boldsymbol{\Sigma}_1^{-1}, \boldsymbol{\Sigma}_2^{-1}, \ldots, \boldsymbol{\Sigma}_N^{-1}\right]$$: all precision matrices

Parameter update formulae

$$\sum_{\forall \boldsymbol{q}} p(\boldsymbol{q} \mid \boldsymbol{c}, \lambda') \boldsymbol{S}_{\boldsymbol{q}}^{\top} \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{W} \boldsymbol{P}_{\boldsymbol{q}} \boldsymbol{W}^{\top} \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{S}_{\boldsymbol{q}} \boldsymbol{\mu} = \sum_{\forall \boldsymbol{q}} p(\boldsymbol{q} \mid \boldsymbol{c}, \lambda') \boldsymbol{S}_{\boldsymbol{q}}^{\top} \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{W} \boldsymbol{c}$$

mean vectors ⇒ closed form

$$\frac{\partial \mathcal{Q}(\lambda, \lambda')}{\partial \boldsymbol{\phi}} = \sum_{\forall \boldsymbol{q}} p(\boldsymbol{q} \mid \boldsymbol{c}, \lambda') \Big\{ \frac{1}{2} \boldsymbol{S}_{\boldsymbol{q}}^{\top} \mathrm{diag}^{-1} \Big(\boldsymbol{W} \boldsymbol{P}_{\boldsymbol{q}} \boldsymbol{W}^{\top} - \boldsymbol{W} \boldsymbol{c} \boldsymbol{c}^{\top} \boldsymbol{W}^{\top} + \boldsymbol{W} \bar{\boldsymbol{c}}_{\boldsymbol{q}} \bar{\boldsymbol{c}}_{\boldsymbol{q}}^{\top} \boldsymbol{W}^{\top} + \boldsymbol{\mu}_{\boldsymbol{q}} \boldsymbol{c}^{\top} \boldsymbol{W}^{\top} + \boldsymbol{W} \boldsymbol{c} \boldsymbol{\mu}_{\boldsymbol{q}}^{\top} - \boldsymbol{\mu}_{\boldsymbol{q}} \bar{\boldsymbol{c}}_{\boldsymbol{q}}^{\top} \boldsymbol{W}^{\top} - \boldsymbol{W} \bar{\boldsymbol{c}}_{\boldsymbol{q}} \boldsymbol{\mu}_{\boldsymbol{q}}^{\top} \Big) \Big\}$$

covariance matrices ⇒ numerical optimization

Drawback of trajectory HMM training

Exact EM is intractable

- Computing posterior prob of q is intractable
- Single-path (Viterbi) or Monte Carlo approximation

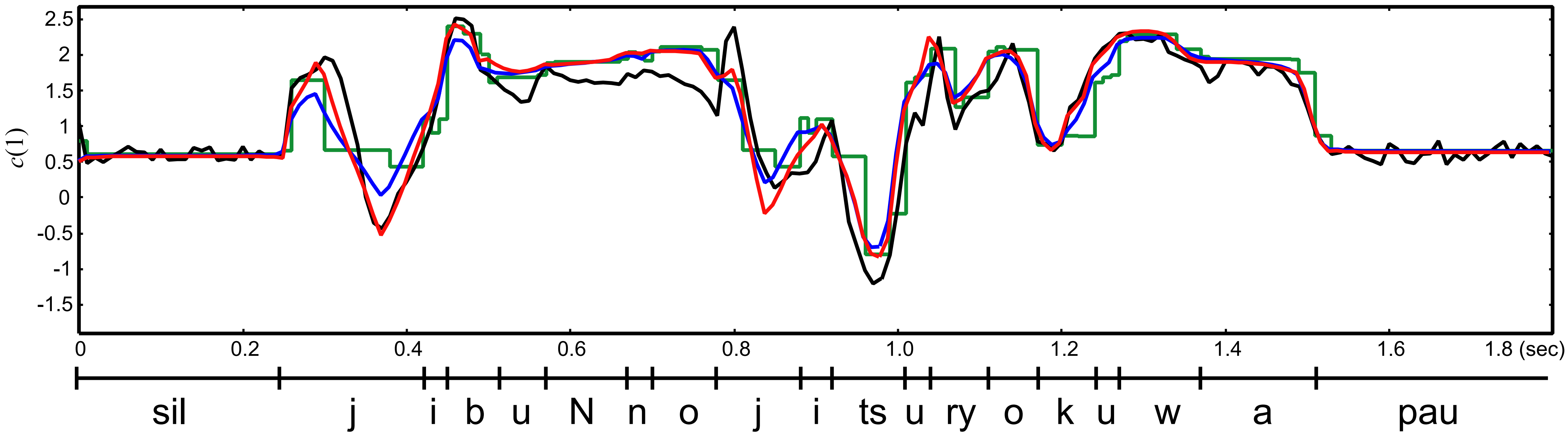
Exact tree-based clustering is also intractable

- Splitting one nodes affects the other nodes
- Trees built for HMMs are often used

Computationally & memory intensive

- High dimensional matrix operations
- Numerical optimization

Effect of parameter reestimation



— Training data

— Mean sequence of the HMM

— Mean sequence of the trajectory HMM (w/o update)

— Mean sequence of the trajectory HMM (with update)

Outline

Trajectory HMM

- Speech parameter generation
- Derivation of trajectory HMM
- Relationship between parameter generation & trajectory HMM
- Trajectory HMM as globally normalized model
- Parameter estimation

Min generation error (MGE) training & trajectory HMM

- Relationship
- Properties of MGE

(If time remains) Product of Experts (PoE)

- Combination of multiple AMs as PoE
- PoE & trajectory HMM

MGE training & trajectory HMM (1)

ML training & MGE training w/ Euclidean dist [Wu;'06]

$$\hat{\lambda}_{\mathrm{ML}} = \arg\max_{\lambda} p(\boldsymbol{c} \mid \boldsymbol{q}, \lambda)$$

$$= \arg\max_{\lambda} \mathcal{N}\left(\boldsymbol{c}\,;\ \bar{\boldsymbol{c}}_{\boldsymbol{q}}, \boldsymbol{P}_{\boldsymbol{q}}\right)$$

$$\hat{\lambda}_{\mathrm{MGE}} = \arg\min_{\lambda} \mathcal{E}(\boldsymbol{c}\,;\ \boldsymbol{q}, \lambda)$$

$$= \arg\min_{\lambda} \underbrace{||\boldsymbol{c} - \bar{\boldsymbol{c}}_{\boldsymbol{q}}||_2}_{\text{Euclidean distance}} \Leftarrow \textbf{MMSE estimation}$$

$$= \arg\max_{\lambda} \mathcal{N}\left(\boldsymbol{c}\,;\ \bar{\boldsymbol{c}}_{\boldsymbol{q}}, \boldsymbol{I}\right) \Leftarrow \textbf{Identity covariance matrix}$$

MGE training & trajectory HMM (3)

Performance of ML & MGE w/ Euc is similar, why?

$\Rightarrow$ Due to speech parameter generation algorithm

$$
\begin{aligned}
\hat{\boldsymbol{c}}_{\mathrm{ML}} &= \arg\max_{\boldsymbol{c}} p\left(\boldsymbol{c} \mid \hat{\boldsymbol{q}}, \hat{\lambda}_{\mathrm{ML}}\right) \\
&= \arg\max_{\boldsymbol{c}} \mathcal{N}\left(\boldsymbol{c} \,;\, \bar{\boldsymbol{c}}_{\hat{\boldsymbol{q}}}, \boldsymbol{P}_{\hat{\boldsymbol{q}}}\right) \\
&= \bar{\boldsymbol{c}}_{\hat{\boldsymbol{q}}}
\end{aligned}
$$

$$
\begin{aligned}
\hat{\boldsymbol{c}}_{\mathrm{MGE}} &= \arg\max_{\boldsymbol{c}} p\left(\boldsymbol{c} \mid \hat{\boldsymbol{q}}, \hat{\lambda}_{\mathrm{MGE}}\right) \\
&= \arg\max_{\boldsymbol{c}} \mathcal{N}\left(\boldsymbol{c} \,;\, \bar{\boldsymbol{c}}_{\hat{\boldsymbol{q}}}, \boldsymbol{I}\right) \\
&= \bar{\boldsymbol{c}}_{\hat{\boldsymbol{q}}}
\end{aligned}
$$

MGE training & trajectory HMM (4)

Random sampling from ML & MGE w/ Euc distance

ML

$$\tilde{\boldsymbol{c}}_{\mathrm{ML}} \sim \mathcal{N}\left(\bar{\boldsymbol{c}}_{\hat{\boldsymbol{q}}}, \boldsymbol{P}_{\hat{\boldsymbol{q}}}\right)$$

$\Rightarrow$ Temporal correlations will be kept

MGE

$$\tilde{\boldsymbol{c}}_{\mathrm{MGE}} \sim \mathcal{N}\left(\bar{\boldsymbol{c}}_{\hat{\boldsymbol{q}}}, \boldsymbol{I}\right)$$

$\Rightarrow$ Temporal correlations will be discarded

MGE training & trajectory HMM (5)

Which is better, ML or MGE?

- **w/ parameter generation, MGE is more reasonable**
  * MGE $\Rightarrow$ μ & Σ to represent mean trajectory
  * ML $\Rightarrow$ μ for mean trajectory, Σ for mean trj & temporal cov

  $\Rightarrow$ MGE can focus on modeling mean trajectory
- **w/ random sampling, ML is more reasonable**
  * MGE *ignores* temporal correlations
  * ML *models* temporal correlations

Summary

Trajectory HMM

- Derived from HMM w/ dynamic feature constraints
- Can be viewed as a globally normalized model
- All states need to be estimated jointly
- Generated params = mean vector of trajectory HMM

MGE training

- MGE w/ Euclid distance = MMSE estimation of trajectory HMM
- w/ speech parameter generation algorithm (ML parm gen)
  - $\Rightarrow$ ML & MGE work similarly
- w/ random sampling
  - $\Rightarrow$ MGE won't work well

Outline

Trajectory HMM

- Speech parameter generation
- Derivation of trajectory HMM
- Relationship between parameter generation & trajectory HMM
- Trajectory HMM as globally normalized model
- Parameter estimation

Min generation error (MGE) training & trajectory HMM

- Relationship
- Properties of MGE

(If time remains) Product of Experts (PoE)

- Combination of multiple AMs as PoE
- PoE & trajectory HMM

Combination of multiple acoustic models

Combine multiple AMs to reduce over-smoothing

* Training; estimate multiple-level AMs ***individually***

$$\hat{\lambda}_i = \arg\max_{\lambda_i} p\left(f_i(\boldsymbol{c}) \mid \lambda_i\right) \quad i = 1, \ldots, M$$

* Synthesis; generate $\boldsymbol{c}$ that ***jointly*** maximize output probs from AMs

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} \sum_{i=1}^{M} \alpha_i \log p(f_i(\boldsymbol{c}) \mid \hat{\lambda}_i)$$

* Feature function, $f_i(\boldsymbol{c})$, extracts acoustic feats for i-th AM from $\boldsymbol{c}$

- e.g., dynamic feats, DCT, average, summation, global variance

* Parameters of AMs, λ_i , are trained ***independently***

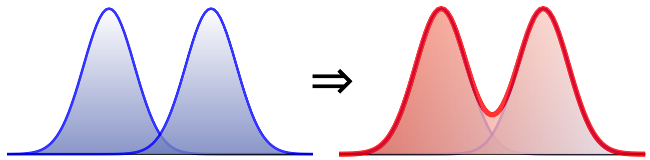
$\longrightarrow$ Use weights to control balance among AMs

* Weights, α_i , are determined by ***held-out data*** (or tuned manually)

Mixture model vs Product model

Mixture of experts

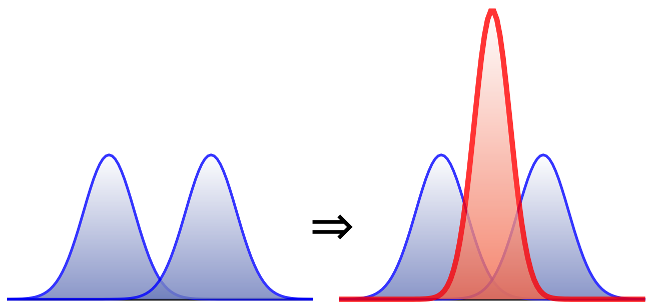
$$p\left(\boldsymbol{c} \mid \lambda_1, \ldots, \lambda_M\right) = \frac{1}{Z} \sum_{i=1}^{M} \alpha_i p\left(f_i(\boldsymbol{c}) \mid \lambda_i\right)$$



* Data is generated from ***union*** of experts
* Robust for modeling data with many variations
* GMM ⟶ Mixture of Gaussians

Product of experts [Hinton;'02]

$$p\left(\boldsymbol{c} \mid \lambda_1, \ldots, \lambda_M\right) = \frac{1}{Z} \prod_{i=1}^{M} \left\{p\left(f_i(\boldsymbol{c}) \mid \lambda_i\right)\right\}^{\alpha_i}$$



* Data is generated from ***intersection*** of experts
* Efficient for modeling data with many constraints
* PoG ⟶ Product of Gaussians

Combination of multiple AMs as PoE

Comibination of multiple AMs can be viewed as PoE

$$\hat{\boldsymbol{c}} = \arg\max_{\boldsymbol{c}} p\left(\boldsymbol{c} \mid \lambda_1, \ldots, \lambda_M\right) = \arg\max_{\boldsymbol{c}} \frac{1}{Z} \prod_{i=1}^{M} \left\{p\left(f_i(\boldsymbol{c}) \mid \lambda_i\right)\right\}^{\alpha_i}$$

$$= \arg\max_{\boldsymbol{c}} \prod_{i=1}^{M} \left\{p\left(f_i(\boldsymbol{c}) \mid \lambda_i\right)\right\}^{\alpha_i} = \arg\max_{\boldsymbol{c}} \sum_{i=1}^{M} \alpha_i \log p\left(f_i(\boldsymbol{c}) \mid \lambda_i\right)$$

* Generating $\boldsymbol{c}$ from combination of multiple AMs

  $\longrightarrow$ Equivalent to generating $\boldsymbol{c}$ from PoE consisting of AMs

* Regarding combination of multiple AMs as PoE

  $\longrightarrow$ ***Jointly*** estimate multiple AMs

$$\{\hat{\lambda}_1, \ldots, \hat{\lambda}_M\} = \arg\max_{\lambda_1, \ldots, \lambda_M} \frac{1}{Z} \prod_{i=1}^{M} \left\{p\left(f_i(\boldsymbol{c}) \mid \lambda_i\right)\right\}^{\alpha_i}$$

Product of Gaussians

Product of Gaussians (PoG)

$$p\left(\boldsymbol{c} \mid \lambda_1, \ldots, \lambda_M\right) = \frac{1}{Z} \prod_{i=1}^{M} \mathcal{N}\left(f_i(\boldsymbol{c})\ ;\ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)$$

* Special case of PoE; All experts are Gaussian

* If all feature functions are linear

- PoG also becomes Gaussian
- Normalization constant

$$Z = \int \prod_{i=1}^{M} \mathcal{N}\left(f_i(\boldsymbol{c})\ ;\ \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right) d\boldsymbol{c}$$

can be computed in ***closed form***

Trajectory HMM as product of Gaussians

Trajectory HMM can be viewed as PoG [Williams;'05, Zen;'07]

$$p(\boldsymbol{c} \mid \lambda) = \sum_{\forall \boldsymbol{q}} p(\boldsymbol{c} \mid \boldsymbol{q}, \lambda)\, p(\boldsymbol{q} \mid \lambda)$$

$$p(\boldsymbol{c} \mid \boldsymbol{q}, \lambda) = \mathcal{N}(\boldsymbol{c}\,;\, \bar{\boldsymbol{c}}_{\boldsymbol{q}}, \boldsymbol{P}_{\boldsymbol{q}}) = \frac{1}{Z_{\boldsymbol{q}}} \mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\, \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}})$$

$$\begin{array}{ccc}
\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}) & & \boldsymbol{W}\boldsymbol{c} \\
\vdots & & \vdots \\
\boldsymbol{\mu}^{(0)}_{q_{t\text{-}1}} \quad \boldsymbol{\Sigma}^{(0)}_{q_{t\text{-}1}} & \rightarrow & \boldsymbol{c}_{t-1} \\
\boldsymbol{\mu}^{(1)}_{q_{t\text{-}1}} \quad \boldsymbol{\Sigma}^{(1)}_{q_{t\text{-}1}} & \rightarrow & \Delta\, \boldsymbol{c}_{t-1} \\
\boldsymbol{\mu}^{(0)}_{q_t} \quad \boldsymbol{\Sigma}^{(0)}_{q_t} & \rightarrow & \boldsymbol{c}_{t} \\
\boldsymbol{\mu}^{(1)}_{q_t} \quad \boldsymbol{\Sigma}^{(1)}_{q_t} & \rightarrow & \Delta\, \boldsymbol{c}_{t} \\
\boldsymbol{\mu}^{(0)}_{q_{t+1}} \quad \boldsymbol{\Sigma}^{(0)}_{q_{t+1}} & \rightarrow & \boldsymbol{c}_{t+1} \\
\boldsymbol{\mu}^{(1)}_{q_{t+1}} \quad \boldsymbol{\Sigma}^{(1)}_{q_{t+1}} & \rightarrow & \Delta\, \boldsymbol{c}_{t+1} \\
\vdots & & \vdots
\end{array}$$

$$\underbrace{\begin{bmatrix} \vdots \\ \boldsymbol{c}_{t-1} \\ \Delta\, \boldsymbol{c}_{t-1} \\ \boldsymbol{c}_{t} \\ \Delta\, \boldsymbol{c}_{t} \\ \boldsymbol{c}_{t+1} \\ \Delta\, \boldsymbol{c}_{t+1} \\ \vdots \end{bmatrix}}_{\boldsymbol{W}\boldsymbol{c}} = \underbrace{\begin{bmatrix} \cdots & \vdots & \vdots & \vdots & \vdots & \cdots \\ \cdots & 0 & \boldsymbol{I} & 0 & 0 & \cdots \\ \cdots & -\boldsymbol{I} & \boldsymbol{I} & 0 & 0 & \cdots \\ \cdots & 0 & 0 & \boldsymbol{I} & 0 & \cdots \\ \cdots & 0 & -\boldsymbol{I} & \boldsymbol{I} & 0 & \cdots \\ \cdots & 0 & 0 & 0 & \boldsymbol{I} & \cdots \\ \cdots & 0 & 0 & -\boldsymbol{I} & \boldsymbol{I} & \cdots \\ \cdots & \vdots & \vdots & \vdots & \vdots & \cdots \end{bmatrix}}_{\boldsymbol{W}} \underbrace{\begin{bmatrix} \vdots \\ \boldsymbol{c}_{t-2} \\ \boldsymbol{c}_{t-1} \\ \boldsymbol{c}_{t} \\ \boldsymbol{c}_{t+1} \\ \vdots \end{bmatrix}}_{\boldsymbol{c}}$$

Trajectory HMM as product of Gaussians

Trajectory HMM can be viewed as PoG [Williams;'05, Zen;'07]

$$p(\boldsymbol{c} \mid \lambda) = \sum_{\forall \boldsymbol{q}} p(\boldsymbol{c} \mid \boldsymbol{q}, \lambda)\, p(\boldsymbol{q} \mid \lambda)$$

$$p(\boldsymbol{c} \mid \boldsymbol{q}, \lambda) = \mathcal{N}(\boldsymbol{c}\,;\ \bar{\boldsymbol{c}}_{\boldsymbol{q}}, \boldsymbol{P}_{\boldsymbol{q}}) = \frac{1}{Z_{\boldsymbol{q}}} \mathcal{N}(\boldsymbol{W}\boldsymbol{c}\,;\ \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}})$$

$$= \frac{1}{Z_{\boldsymbol{q}}} \prod_{t=1}^{T} \prod_{d=0}^{2} \mathcal{N}\left(f_t^{(d)}(\boldsymbol{c})\,;\ \boldsymbol{\mu}_{q_t}^{(d)}, \boldsymbol{\Sigma}_{q_t}^{(d)}\right)$$

$f_t^{(d)}(\boldsymbol{c})$: d-th dyn feat at frame t

$$Z_q = \int \prod_{t=1}^{T} \prod_{d=0}^{2} \mathcal{N}\left(f_t^{(d)}(\boldsymbol{c})\,;\ \boldsymbol{\mu}_{q_t}^{(d)}, \boldsymbol{\Sigma}_{q_t}^{(d)}\right) d\boldsymbol{c}$$

* Experts are Gaussians, feature functions are dynamic features
* Gaussian experts are multiplied over time

Linear feature function with Gaussian experts

Combining multiple AMs as PoE

* Multiple-level AMs often use linear feature functions w/ Gaussians
  - DCT [Latorre;'08, Qian;'09], average [Wang;'08], sum [Ling;'06, Gao;'08]
* PoEs become the same form as trajectory HMM
  - $\longrightarrow$ Training algorithm for trajectory HMM are applicable

Example: state & phoneme duration models [Ling;'06]

$$
\begin{aligned}
p\left(\boldsymbol{d} \mid \lambda\right) &= \frac{1}{Z}\mathcal{N}\left(\boldsymbol{W}\boldsymbol{d}\;;\;\boldsymbol{\mu},\boldsymbol{\Sigma}\right)\\
&= \frac{1}{Z}\prod_{i=1}^{P}\prod_{j=1}^{N_i}\mathcal{N}\left(d_{ij}\;;\;\xi_{ij},\sigma_{ij}\right)\\
&\quad\cdot\prod_{k=1}^{P}\mathcal{N}\left(p_k\;;\;\nu_k,\omega_k\right)
\end{aligned}
$$

$$
\begin{array}{cccc}
 & & \boldsymbol{W} & \boldsymbol{d}\\
\left[\begin{array}{c} d_{11}\\ d_{12}\\ d_{13}\\ p_1\\ \hline d_{21}\\ d_{22}\\ d_{23}\\ p_2\\ \hline \vdots \end{array}\right] & = & \left[\begin{array}{ccc|ccc|c} 1&0&0& & & &\cdots\\ 0&1&0& &\mathbf{0}& &\cdots\\ 0&0&1& & & &\cdots\\ 1&1&1& & & &\cdots\\ \hline & & &1&0&0&\cdots\\ &\mathbf{0}& &0&1&0&\cdots\\ & & &0&0&1&\cdots\\ & & &1&1&1&\cdots\\ \hline \vdots&\vdots&\vdots&\vdots&\vdots&\vdots&\cdots \end{array}\right] & \left[\begin{array}{c} d_{11}\\ d_{12}\\ d_{13}\\ \hline d_{21}\\ d_{22}\\ d_{23}\\ \hline \vdots \end{array}\right]
\end{array}
$$

d_{ij} : duration of state j in phoneme i

General PoE (non-linear feat or non-Gaussian)

General form of PoE

$$p\left(\boldsymbol{c} \mid \lambda_1, \ldots, \lambda_M\right) = \frac{1}{Z} \prod_{i=1}^{M} \left\{p\left(f_i(\boldsymbol{c}) \mid \lambda_i\right)\right\}^{\alpha_i}$$

* Feature functions can be non-linear, experts can be non-Gaussian

* Normalization term has no closed form

* Training is complex, usually normalization term is approximated

Example: global variance (GV) [Toda;'07]

$$p\left(\boldsymbol{c} \mid \boldsymbol{q}, \lambda, \lambda_{\mathrm{GV}}\right) = \frac{1}{Z_{\boldsymbol{q}}} \mathcal{N}\left(\boldsymbol{c}\,;\,\bar{\boldsymbol{c}}_{\boldsymbol{q}}, \boldsymbol{P}_{\boldsymbol{q}}\right)^{\alpha} \mathcal{N}\left(f_v(\boldsymbol{c})\,;\,\boldsymbol{\mu}_v, \boldsymbol{\Sigma}_v\right)$$

$$f_v(\boldsymbol{c}) = \frac{1}{T} \sum_{t=1}^{T} \mathrm{diag}\left[\left(\boldsymbol{c}_t - \bar{\boldsymbol{c}}\right)\left(\boldsymbol{c}_t - \bar{\boldsymbol{c}}\right)^{\top}\right]$$: intra-utt variance, ***quadratic***

Contrastive divergence leárining

Contrastive divergence learning [Hinton;'02]

* Training algorithm for general PoE
* Combination of sampling & gradient methods

1. Draw J samples from PoE

$$\boldsymbol{c}^{(j)} \sim p\left(\boldsymbol{c} \mid \boldsymbol{\lambda}\right) \quad j = 1, \ldots, J \qquad \boldsymbol{\lambda} = \{\lambda_1, \ldots, \lambda_M\}$$
: PoE model params

2. Compute approximated derivative of log likelihood w.r.t. $\boldsymbol{\lambda}$

$$\frac{\partial \log p\left(\boldsymbol{c} \mid \boldsymbol{\lambda}\right)}{\partial \boldsymbol{\lambda}} \approx \left\langle \frac{\partial \log p\left(\boldsymbol{c} \mid \boldsymbol{\lambda}\right)}{\partial \boldsymbol{\lambda}} \right\rangle_{p_0} - \left\langle \frac{\partial \log p\left(\boldsymbol{c} \mid \boldsymbol{\lambda}\right)}{\partial \boldsymbol{\lambda}} \right\rangle_{p_J}$$

expectation over data — expectation over samples

3. Update model params using gradient method

$$\boldsymbol{\lambda}' = \boldsymbol{\lambda} - \eta \cdot \left(\left\langle \frac{\partial \log p\left(\boldsymbol{c} \mid \boldsymbol{\lambda}\right)}{\partial \boldsymbol{\lambda}} \right\rangle_{p_0} - \left\langle \frac{\partial \log p\left(\boldsymbol{c} \mid \boldsymbol{\lambda}\right)}{\partial \boldsymbol{\lambda}} \right\rangle_{p_j} \right)$$

η : learning rate

$$\boldsymbol{\lambda} = \boldsymbol{\lambda}'$$

4. Iterate 1-3 until converge

Experiment - Multiple-Level Dur Models as PoE

Experimental conditions

* Training data; 2,469 utterances
* Development data; 137 utterances
  - Used to optimize weights in conventional method
  - Weights were optimized to minimize RMSE by grid search
  - Baseline & proposed method did not use development data
* Almost the same training setup as Nitech-HTS 2005 [Zen;'06]
* Test data; 137 utterances
* State, phone, & syllable-level models were clustered individually
  - # of leaf nodes
    * state; 607, phoneme; 1,364, syllable; 281

Experimental Results

Duration prediction results (RMSE in frame (rel imp))

Model	Phoneme	Syllable	Pause
Baseline (st)	5.08 (ref)	8.98 (ref)	35.0 (ref)
uPoE (st*ph)	4.62 (9.1%)	8.13 (9.5%)	**31.8 (9.1%)**
uPoE (st*ph*syl)	4.62 (9.1%)	8.11 (9.7%)	**31.8 (9.1%)**
PoE (st*ph)	4.60 (9.4%)	8.04 (10.5%)	31.9 (8.9%)
PoE (st*ph*syl)	**4.57 (10.0%)**	**8.02 (10.7%)**	31.9 (8.9%)

st; state only, st*ph; state & phoneme, st*ph*syl; state, phoneme, & syllable

uPoE; individually trained multiple-level duration models with optimized weights

PoE; jointly estimated multiple-level duration models

Experiment - Global Variance as PoE

Experimental conditions

* Training data; 2,469 utterances
  - Training data was split into mini-batch (250 utterances)
  - 10 MCMC sampling at each contrastive divergence learning
    * Hybrid Monte Carlo with 20 leap-frog steps
    * Leap-frog size was adjusted adaptively
  - Learning rate was annealed at every 2,000 iterations
  - Momentum method was used to accelerate learning
  - Context-dependent logarithmic GV w/o silence was used
* Test sentences; 70 sentences
  - Paired comparison test, # of subjects 7 (native English speaker)
  - 30 sentences per subject

Experimental Results

Paired comparison test result

Baseline	PoE	No preference
17.1	32.4	50.5

Baseline; conventional (not jointly estimated) GV

PoE; proposed (jointly estimated) GV

Difference was statistically significant at $p<0.05$ level

Summary

Statistical parametric synthesis based on PoE

- **Combination of multiple-level AMs is formulated as PoE**
- **Jointly estimate multiple-level AMs as PoE**
  - * Linear feature function with Gaussian experts
    - → Can be estimated in the same way as trajectory HMM
  - * Non-linear feature function and/or non-Gaussian experts
    - → Contrastive divergence learning
- **Experiments**
  - * Jointly estimating multiple AMs as PoE improved performance

References

[Hinton;'02] G. Hinton, "Training product of experts by minimizing contrastive divergence," Neural Computation, vol. 14, no. 8, pp. 1771-1800, 2002.

[Williams;'05] C. Williams, "How to pretend that correlated variables are independent by using difference observations," Neural Computation, vol. 17, no. 1, pp. 1--6, 2005.

[Zen;'07] H. Zen, et al., "Reformulating the HMM as a trajectory model by imposing explicit relationships between static and dynamic feature vector sequences," Computer Speech & Language, vol. 21, no. 1, pp.153-173, 2007.

[Latorre;'08] J. Latorre & M. Akamine, "Multilevel parametric-base F0 model for speech synthesis," Proc. Interspeech, pp. 2274--2277, 2008.

[Qian;'09] Y. Qian, et al., "Improved prosody generation by maximizing joint likelihood of state and longer units," Proc. ICASSP, pp. 1173--1176, 2009.

[Wang;'08] C.-C. Wang, et al., "Multi-layer F0 modeling for HMM-based speech synthesis," Proc. ISCSLP, pp. 129--132, 2008.

[Ling;'06] Z.-H. Ling, et al., "USTC system for Blizzard Challenge 2006 an improved HMM-based speech synthesis method," Proc. Blizzard Challenge Workshop, 2006.

[Gao;'08] B.-H. Gao, et al., "Duration refinement by jointly optimizing state and longer unit likelihood," Proc. Interspeech, 2266--2269, 2008.

[Toda;'07] T. Toda & 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.