Time-line Hidden Markov Experts and Its Application in Time Series Prediction

Xin Wang **Peter Whigham** **Da Deng**

Department of Information Science
University of Otago
Dunedin, New Zealand

{*xinw, pwhigham, ddeng*}*@infoscience.otago.ac.nz*

Abstract

A modularised connectionist model, based on the Mixture of Experts (ME) algorithm for time series prediction, is introduced. A set of connectionist modules learn to be local experts over some commonly appearing states of a time series. The dynamics for mixing the experts is a Markov process, in which the states of a time series are regarded as states of a HMM. Hence, there is a Markov chain along the time series and each state associates to a local expert. The state transition on the Markov chain is the process of activating a different local expert or activating some of them simultaneously by different probabilities generated from the HMM. The state transition property in the HMM is designed to be time-variant and conditional on the first order dynamics of the time series. A modified Baum–Welch algorithm is introduced for the training of the time-variant HMM and it has been proved that by EM process the likelihood function will converge to a local minimum. Experiments, with two time series, show this approach achieves significant improvement in the generalisation performance over global models.

Key Words: Time Series prediction; Mixture of Experts; HMM; Connectionist Model; Expectation and Maximization; Gauss Probability Density Distribution;

$\boldsymbol{Y}$:	A time series, also a series of observations for HMM. $\boldsymbol{Y}=\{y_1, y_2, ..., y_T\}$.
y_t:	Value of a time series at time t.
T:	Length of a time series.
$\boldsymbol{X}_t$:	Model input at time t, it is a vector by embedding L previous values of a time series. $\boldsymbol{X}_t=[y_{t-1}, y_{t-2}, ..., y_{t-L}]$.
$\boldsymbol{D}_t$:	Dynamic situation defined for a time series at time t: $\boldsymbol{D}_t = \Delta\boldsymbol{X}_t = \boldsymbol{X}_t - \boldsymbol{X}_{t-1}$.
$\hat{y}_j(\boldsymbol{X}_t)$:	Output from expert j at time t.
M:	The number of states in HMM, also the number of local experts, and the number of clusters extracted by fuzzy clustering technique.
$b(t)$:	Distribution function of probability density in HMM. It is assumed Gauss in the paper.
σ^2:	Variance of Gauss distribution.
$\boldsymbol{S}$:	State series of a Markov chain along a time series. $\boldsymbol{S}_t=\{s_1, s_2, ..., s_T\}$.
s_t:	The value of $\boldsymbol{S}$ at time t.
ζ:	Space of state values, s, can be taken.
π_i:	Probability of being in state i at $t=0$.
$a_{ij}(t)$:	State transition probability from state i at time $t-1$ to state j at time t.
$\boldsymbol{\lambda}'$:	Value of HMM parameters used for E-step in the modified Baum-Welch algorithm.
$\boldsymbol{\lambda}$:	Variable for HMM parameters or the parameters to be estimated in M-step in the modified Baum-Welch algorithm.
ξ:	Space of HMM parameter, $\boldsymbol{\lambda}$, can be taken.
$Q(\boldsymbol{\lambda},\boldsymbol{\lambda}')$:	Likelihood function of observing $\boldsymbol{Y}$.
$p(\boldsymbol{Y} \mid \boldsymbol{\lambda}')$:	Probability of observing $\boldsymbol{Y}$ with $\boldsymbol{\lambda}'$.
$\alpha_i(t)$:	Probability of observing $\boldsymbol{Y}_t=\{y_1, y_2, ..., y_t\}$ ($t \le T$) and ending up in state i at time t.
$\beta_i(t)$:	Probability of observing $\boldsymbol{Y}_t'=\{y_{t+1}, y_{t+2},..., y_T\}$ from time t and starting with state $s_t=i$ at time t.
$\gamma_j(t)$:	Probability of being in state j at time t when observing $\boldsymbol{Y}$. ($t \le T$).
$\eta_{ij}(t)$:	Probability of being in state i at time t and being in state j at time $t+1$ when observing $\boldsymbol{Y}$.

Figure 1. List of the symbols used in the paper.

1 Introduction

In the field of time series analysis, the modelling techniques can be divided generally into two categories: local modelling or nonparametric modelling and global modelling or parametric modelling. Local models, such as nearest neighbour algorithm, are formed in each step relied on amount of data. The philosophy is finding segments of the time series that closely resemble the segment of the current point. Global models, such as autoregression models, and connectionist models, are usually constructed to fit the whole process of a time series by minimizing the squared error. One problem of local models is that they cannot give a global description of the time series. However, it is also not easy to construct a single global model to represent a time series precisely especially when the time series show some complex features, such as chaos. To deal with these problems, a type of model, called Mixture of Experts (ME), appearing. The ME was developed on divide-and-conquer principle with the idea that dividing a complex problem into some simple ones and dealing with each of them separately.

1.1 Connectionist ME models

The ME was introduced to connectionist society by Jacobs (Jordan *et al.*, 1991) in 1991. The main point is training some " sub-models" in local environments to make them become "experts" over the local environments, and combining the experts by some algorithms to generate final output. Generally there are three main model structures developed based on ME: GE (Gated Experts and Hierarchical Mixture of Experts) (Jordan *et al.*, 1991; Jordan *et al.*, 1992; Jordan *et al.*, 1994; Weigend *et al.*, 1996), HME (Hidden Markov Experts) (Weigend *et al.*, 2000), and IOHMM (Input/Output Hidden Markov Model) (Bengio *et al.*, 1995; Bengio, 1996). GE combines the experts by a gating network, which is usually a liner (Jordan *et al.*, 1991; Jordan *et al.*, 1992; Jordan *et al.*, 1994) or a non-linear (Weigend *et al.*, 1996) feed forward network. In both HME and IOHMM, the experts are hosted by a HMM, but the state transition probabilities in the IOHMM are generated from a set of recurrent networks called state transition network.

1.2 ME model in time series prediction

For time series modelling, the benefits of ME include that, on one side it could be used to extract regimes or states from complex time series, on the other side taking a sub-model to fit each state leads the localised modelling more efficient and precise. Just as Weigend said, "Extracting regime information does not sacrifice prediction accuracy. In contrary, we can obtain better predictions since the experts can truly be experts in their region, as opposed to covering everything poorly" (Weigend *et al.*, 1996).

Some early works for time series modeling include TAR (Threshold Autoregression) (Tong *et al.*, 1980), CART (Classification and Regression Trees) (Breiman *et al.*, 1984), and MARS (Multivariate Adaptive Regression Splines) (Friedman, 1991). These models simply split the input space into some regions and fit them locally with regression models. In connectionist society, all the GE, HME, and IOHME models have been used for time series prediction (Weigend *et al.*, 1996; Weigend *et al.*, 2000; Bengio, 2001). These models have firmly statistical background as the model-training is a process of maximising the probability of observing the time series instead of minimizing the squared error. In addition to these models, there are also some other models based on ME: HMME (Liehr *et al.*, 1999) and the model introduced in (Kohlmorgen *et al.*, 2000). Both models are based on HMM and trying to describe the state transition process more precisely. Another model, which should

be classified to GM model, is the model introduced in (Cao, 2003), where a SOM is used as gating network rather than a feed forward network.

For dynamical time series, it is commonly to define them in terms of a state space description:

$$\boldsymbol{s}_t = g(\boldsymbol{s}_{t-1}, \boldsymbol{w}_t) \tag{1}$$

$$y_t = f(\boldsymbol{s}_t, v_t) \tag{2}$$

where $\boldsymbol{s}_t \in R^s$ is the state of the time series, $\boldsymbol{w}_t$, v_t are white noise. For such time series, hidden Markov model is appropriate for representing the underlying dynamical process. So HMMs have been popularly adopted for time series analysis.

Some significant work about HMM for time series analysis include the follows: Hamilton applied the idea of switching regimes to model conditional variance of economical time series, where autoregressive conditional heteroskedasticity (ARCH) is used to model the variance, but its parameters are regime-related and learned by EM algorithm (Hamilton, 1989; 1990; 1996; Hamilton *et al.*, 1994). Weigend combined HMM and nonlinear feed forward networks to predict probability density distribution for a financial time series: S&P500 (Weigend *et al.*, 2000). Bengio employed different IOHMEs, which take linear and linear networks as experts, for financial return series prediction (Bengio, 2001). In addition to the applications in economical field, Liehr, (Liehr *et al.*, 1999) and (Kohlmorgen, 2000) used their models for chaotic time series segmentation.

Usually HMM works in a ME structured model, where it moderate the experts to represent the state transition. When the model is applied for prediction, a problem it face to is that it is unable to describe the transitions at each time point precisely since the state transition property is defined over the whole process. Hence the difficulty is that on one hand people intend to take the advantage of HMM to model time series more accurately, on the other hand the global property makes the transitions in same probability and results in inferior predictions. Here we introduce a model in ME structure named "THME" (Time-line Hidden Markov Experts) for point prediction. It has a similar process to a HME (Weigend *et al.*, 2000), but the state transition property is time-variant. That means rather than holding a transition probability for the whole process, the THME localizes the transition property at each time point and models it from the dynamic situation of the time series. Hence, the state transition at a time point is available when the dynamic situation of the time series is known. The training process for THME includes decomposing a complex time series into some simple and commonly appeared states, learning each of them locally by an expert, and constructing a first order HMM to moderate the experts for observing the whole series with maximum probability. Finally a connectionist model is employed to learning the time-variant transition probability. In the process of prediction, all experts with the same inputs take part in prediction but the relative contributions of them are determined by the HMM.

The paper is organised as follows: in section 2 we give a description about the THME and compare it with other ME models. The algorithms for the model training are provided in section 3, and the prediction process is given in section 4. In section 5 we test the model with two time series: Laser data, and Leuven data, followed by discussion and conclusion in section 6. In appendix we present the details about a modified Baum-Welch algorithm for model training and give the explanation for its convergence.

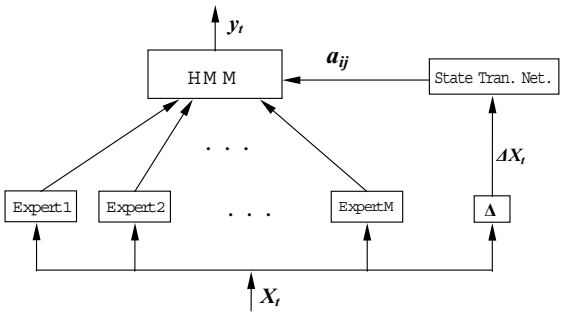

Figure 2. THME with *M* local experts moderated by a HMM. "State Tran. Net." Means "state Transition Network". Δ is differentiating operation.

2 THME Model

The architecture of THME is shown in Figure 2, the experts in THME have similar meanings as that in ME, but they are not limited to particular structures such as linear or non-linear feed forward network, which is often the case in GE, HME, and IOHMM, but not always appropriate for time series modelling. The experts may be any type of connectionist models or regression models depending on the suitability for a particular problem. Each expert responds to a time series state extracted by a Fuzzy clustering technique according to the dynamical situation of the time series $\Delta \boldsymbol{X}_t = \boldsymbol{X}_t - \boldsymbol{X}_{t-1}$, so that the data samples that have similar features are clustered into the same groups. This allows the experts to be relatively simple to learn, and therefore to have good generalization properties. The State Transition Network is a connectionist model used to map localized state transitions. It takes $\Delta \boldsymbol{X}$ as input to estimate state transition so as to trace the experts defined by some changing patterns of a time series. The HMM combines experts by the prior probability of being in each state, which is generated by the state transition probabilities and previous state status. Therefore the mixing process is determined by both interior information and exterior information. A processing diagram is given in Figure 3. In the THME it assumes that for each state the probability density for observations is Gaussian. It takes the output of the expert as the conditional mean and adjusts the variance in training process to fit the noise level on the state. This not only allows a distribution explanation for the expert's output at each time point, but also paves the way for using Bayes law to calculate posterior probability in one-step-ahead prediction. So the THME has a closed-loop re-correction process.

The advantages of THME over GE include that the THME has dynamics in experts-mixing process, it allows state status re-estimating by Bayes law, and no limitation for expert structure. Comparing with HME, both of them take HMM as the dynamics in experts-mixing, however THME is able to describe the state transition probabilities at each point. To IOHMM, THME has Gaussian explanation for output of each expert, whereas the state network in IOHMM just generates conditional mean and its experts must be MLPs. Although the models in (Liehr *et al.*, 1999) and (Kohlmorgen, 2000) have Gauss assumption, they have a restriction that all experts share a same variance value, and the latter one takes the value as the variance of mixed output. For time series prediction, as experts fit their regions with different noise levels, they cannot have same variance value. Even though the noise levels of all experts are same, the variance of the mixed output must be smaller than that of any single expert. Otherwise the mixture would be a failed one. Another problem with the models is that, for state estimation, they either use *softmax*-

function (McCullagh *et al.*, 1989) $P(s_t=i|y_t)=\exp\{-[y_t-\hat{y}_i(\boldsymbol{X}_t)]^2\}\Big/\sum_{j=1}^{M}\exp\{-[y_t-\hat{y}_j(\boldsymbol{X}_t)]^2\}$ (Liehr *et al.*, 1999) or Bayes law but covering just a *temporal neighbourhood* (Kohlmorgen, 2000) without consider of the probability derived from the process: $P(s_t=i|\boldsymbol{Y}_t)$.

There is a significant difference between THME and above models that all the models, except HME, take input $\boldsymbol{X}$ to search experts or state transitions. That means the regions of the experts or the state transition are determined by the input position, whereas THME uses the differentiation of input $\Delta\boldsymbol{X}$. Since THME defines the experts' domains by some changing patterns, taking $\Delta\boldsymbol{X}$ to trace state transition would be more efficient and precise. Previously the authors have experimented with defining experts and modelling state transitions by $\boldsymbol{X}$, but generalization performance, accuracy and convergence speed were inferior to the current model.

3 Model Training

The training process and the productions in each step is shown in Figure 4.

3.1 States extraction

The dynamical situation of a time series at time point *t* is defined as follows:

$$\boldsymbol{D}_t = \Delta\boldsymbol{X}_t = \boldsymbol{X}_t - \boldsymbol{X}_{t-1} = \left[(y_{t-1} - y_{t-2}),\ (y_{t-2} - y_{t-3}),\ldots,(y_{t-L} - y_{t-(L+1)})\right] \tag{3}$$

Fuzzy C-means clustering (Bezdek, 1981; Bezdek *et al.*, 1992) is applied to cluster all time points into groups according to the feature defined in equation 3. The process calculates the cluster membership degree μ_j, which is defined as the degree to which vector $\boldsymbol{X}$ belongs to cluster *j*, and updates cluster centres $\boldsymbol{V}_j$ iteratively to make the following objective function reach a minimum:

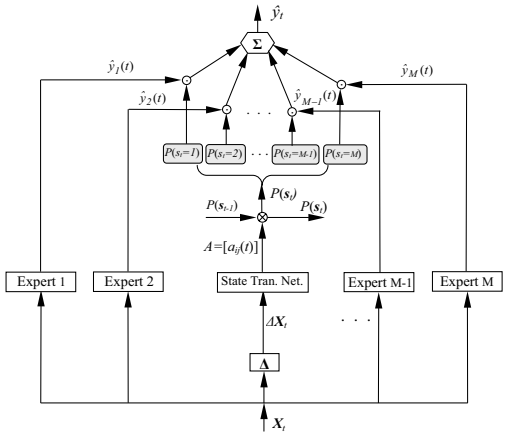

Figure 3. Diagram of TMME. $\boldsymbol{s}_t$ is the state status at time t. $P(\boldsymbol{s}_t)=[P(s_t=1), P(s_t=2), \ldots, P(s_t=M)]$. $P(s_t=i)$ is the probability of being in *i* at time *t*. $\hat{y}_i(t)$ is the output from expert *i*. "Σ" denotes summating operation. "⊙ " denotes multiplication between two values. "⊗" denotes multiplication between matrixes. Others are same as Figure 2.

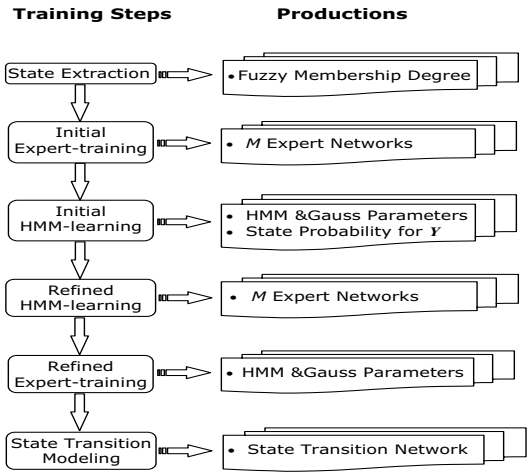

Figure 4. Training process and the Productions.

$$O = \sum_{j=1}^{M} \sum_{t=1}^{T} [\mu_j(t)]^2 \left\| \boldsymbol{D}_t - \boldsymbol{V}_j \right\| \tag{4}$$

The clustering process classifies the time points into M clusters. These clusters are called states of the time series and are also regarded as states of the HMM. Hence, $\mu_j(t)$ may be interpreted as the degree to which the point t belongs to a specified state j.

3.2 Expert training

In both initial expert-training and refined expert-training, local experts are trained by corresponding data samples clustered by state extraction. In initial expert-training, a relatively high threshold K is applied to the fuzzy membership degrees to extract the time points that are strongly featured with a particular state. These time points are then used to train a local expert so as to link the expert to the state. The philosophy behind the process is that some time points that belong to a state of the time series to a high degree should be extracted to train a module to make it a local expert for the state. In the refined expert-training following initial HMM-learning, a relatively lower threshold K' is applied for the probabilities a time point belongs to each state. This allows the model to assign the point into a state or two states if probabilities for the two states are all over the threshold. This process classifies all time points into states and uses them to train the corresponding experts. Comparing with initial expert-training, the refined expert-training re-trains the experts with relatively wider range of samples to make them cover their domains totally.

3.3 HMM learning

Along with the expert training process, the HMM learning also splits into two processes: an initial one and refined one. The difference between them is that the first one is based on the performance of the initial-trained experts and the latter one is on the refined-trained experts. The performance is determined by the output from each expert, which provides the conditional mean for the assumed Gauss distribution on the corresponding HMM state. The HMM with time-variant transition property is learnt by a modified Baum-Welch algorithm based on EM principle (Baum *et al.*, 1970; Dempster *et al.*, 1977; Rabiner, 1989).

Supposing the probability distribution of the HMM is Gaussian. For state j

$$b_j(y_t) = p(y_t \mid s_t = j, \boldsymbol{X}_t, \boldsymbol{\lambda}') = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{[y_t - \hat{y}_j(\boldsymbol{x}_t)]^2}{2\sigma_j^2}} \tag{5}$$

As each state is associated with a local expert, the number of cluster, M, is also the number of states and the number of expects. $j \in 1,2,...,M$; $t \in 1,2,...,T$; $s_t \in \boldsymbol{\zeta}$. Similar to (Baum *et al.*, 1970; Liporace, 1982), the likelihood function for getting observations $\boldsymbol{Y}$ with current parameters $\boldsymbol{\lambda}'$ and to-be-optimised parameters $\boldsymbol{\lambda}$ ($\boldsymbol{\lambda}', \boldsymbol{\lambda} \in \boldsymbol{\xi}$) is defined as follows:

$$Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}') = \sum_{\boldsymbol{S} \in \boldsymbol{\zeta}} \log p(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}) P(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}') \tag{6}$$

Given a particular state sequence $\boldsymbol{S}$, the probability of getting $\boldsymbol{Y}$ is:

$$P(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}') = \boldsymbol{\pi}_{s_0} \prod_{t=1}^{T} a_{s_{t-1}, s_t}(t) b_{s_t}(y_t) \tag{7}$$

where $\boldsymbol{\pi}_{s_0} = \{\pi_i\}$ is the probability of being in state s_0 ($s_0 \in \boldsymbol{\zeta}$) at $t=0$. As in THME the state transition probability is time-variant, at time t it is defined as the probability of transiting from state s_{t-1} at time $t-1$ to state s_t at time t: $a_{s_{t-1}, s_t}(t) = p(s_t \mid \boldsymbol{\lambda}', \boldsymbol{Y}, s_{t-1})$. Hence the likelihood function becomes:

$$Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}') = \sum_{S \in \boldsymbol{\zeta}} \log \boldsymbol{\pi}_{s_0} p(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}') + \sum_{S \in \boldsymbol{\zeta}} \left(\sum_{t=1}^{T} \log a_{s_{t-1}, s_t}(t) \right) p(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}') + \sum_{S \in \boldsymbol{\zeta}} \left(\sum_{t=1}^{T} \log b_{s_t}(y_t) \right) p(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}') \tag{8}$$

In order to estimate the parameters for obtaining the maximum of the likelihood function, the following two steps should be repeated.

- **Expectation** (E-step)

In E-step a forward and a backward process are performed. In the forward process, a probability of observing the partial sequence $\boldsymbol{Y}_t = \{y_1, y_2, ..., y_t\}$ and ending up in state i ($i \in 1,2,...,M$) at time t is defined as: $\alpha_i(t) = p(y_1, y_2, ..., y_t, s_t = i \mid \boldsymbol{\lambda}')$. In the backward process, we define $\beta_i(t) = p(\boldsymbol{Y}_t' \mid s_t = i, \boldsymbol{\lambda}')$ for the probability of starting with state $s_t=i$ at time t to observe $\boldsymbol{Y}_t' = \{y_{t+1}, y_{t+2}, ..., y_T\}$. The probability of being in state i at time t for observing the whole sequence of observations is defined as $\gamma_i(t) = p(s_t = i \mid Y, \boldsymbol{\lambda}')$. Then the three parts in Q function become (see Appendix **A**):

$$Q_1 = \sum_{i=1}^{M} \log \pi_i \, p(\boldsymbol{Y}, s_0 = i \mid \boldsymbol{\lambda}') \tag{9}$$

$$Q_2 = \sum_{i=1}^{M} \sum_{j=1}^{M} \left[\sum_{t=1}^{T} \log a_{ij}(t) \right] p(\boldsymbol{Y}, s_{t-1} = i, s_t = j \mid \boldsymbol{\lambda}') \tag{10}$$

$$Q_3 = \sum_{i=1}^{M} \left[\sum_{t=1}^{T} \log \, b_i(y_t) \right] p(\boldsymbol{Y}, s_t = i \mid \boldsymbol{\lambda}') \tag{11}$$

- **Maximisation** (M-step)

By maximising the *Q* function, updating formulas are derived as follows:

$$\tilde{\pi}_i = \frac{p(\boldsymbol{Y}, s_0 = i \mid \boldsymbol{\lambda}')}{p(\boldsymbol{Y} \mid \boldsymbol{\lambda}')} \tag{12}$$

$$\tilde{a}_{ij}(t) = \frac{p(\boldsymbol{Y}, s_{t-1} = i, s_t = j \mid \boldsymbol{\lambda}')}{p(\boldsymbol{Y}, s_{t-1} = i \mid \boldsymbol{\lambda}')} \tag{13}$$

$$\tilde{\sigma}_i^2 = \frac{\sum_{t=1}^{T} [\gamma_i(t)(y_t - \hat{y}(\boldsymbol{X}_t))^2]}{\sum_{t=1}^{T} \gamma_i(t)} \tag{14}$$

By the modified Baum-Welch algorithm, function *Q* and *P* can converge to local maximums (Appendix **B**).

3.4 State transition modelling

The state transition property is a series of matrix entries corresponding to the hidden state sequence for each point of the time series. In THME, when the time series' dynamical situation changes, there will be a change of the expert that is best for modelling current situation, or a change of the proportion that each expert contributes to the output. That means the expert that was best for the preceding situation may no longer be best for the current situation, or that the degree of fitness for each expert to the current situation changes. Consequently the HMM will experience a transition on state, which may be either a transition from one state to another or a partial transition from being a state in some probability to a new value. Hence we can map the state transition probabilities from the dynamical situation of the time series. Here RBF structured State Transition Network performs the modelling (Figure 2,3).

4 Prediction

The prior probability for each state is taken as the combining coefficient for each expert, and therefore one-step-ahead prediction is available by the following steps:

- At time *t*, with estimated state transition probabilities, the prior probability for each state is determined as:

$$P(s_t = j \mid \boldsymbol{Y}_{t-1}, \boldsymbol{\lambda}') = \sum_{i=1}^{M} a_{ij}(t) P(s_{t-1} = i \mid \boldsymbol{Y}_{t-1}, \boldsymbol{\lambda}') \tag{15}$$

- Combine expert by the prior probabilities to make prediction:

$$\hat{y}_t = \sum_{i=1}^{M} P(s_t = i \mid \boldsymbol{Y}_{t-1}, \boldsymbol{\lambda}') \hat{y}_i(\boldsymbol{X}_t) \tag{16}$$

- The posterior probability for each state can be obtained by Bayes law, and it will be taken as the status of the preceding state for next step prediction.

$$P(s_t = i|\boldsymbol{Y}_t, \boldsymbol{\lambda}') = \frac{P(\boldsymbol{Y}_t|s_t = i, \boldsymbol{\lambda}')}{\sum_{j=1}^{M} P(\boldsymbol{Y}_{t-1}|s_t = j, \boldsymbol{\lambda}')}$$

$$= \frac{p(y_t|s_t = i, \boldsymbol{\lambda}')P(s_t = i|\boldsymbol{Y}_{t-1}, \boldsymbol{\lambda}')}{\sum_{i=1}^{M} p(y_t|s_t = i, \boldsymbol{\lambda}')P(s_t = i|\boldsymbol{Y}_{t-1}, \boldsymbol{\lambda}')} \quad (17)$$

5 Experiments

The THME model has been tested using two chaotic time series: Laser data (Santa Fe Time Series Prediction and Analysis Competition) and Leuven data (K. U. Leuven Time Series Prediction Competition). Laser data is a low dimension (dimension=2.0~2.2), chaotic, low noise data. Chaotic pulsations more or less follow the theoretical Lorenz model of a two level system (Hübner *et al.*, 1994). In the experiments we take 1000 data points for model training and following 500 data points for testing. The embedding dimension is 5 and the time delay is 1. Leuven Competition data is generated from the following computer generalized Chua's circuit (Suykens *et al.*, 1997):

$$\begin{cases} \dot{x}_1 = \alpha[x_2 - h(x_1)] \\ \dot{x}_2 = x_1 - x_2 + x_3 \\ \dot{x}_3 = -\beta x_2 \end{cases}$$

For the K. U. Leuven data, we take first 1000 data samples as training data, next 500 samples for test. Embedding dimension is 12 and time delay is 1.

For each data set, an RBF network and one-hidden-layer MLP are used separately as experts. The RBF network has exponential transfer function. The MLP takes "log-Sigmoid" transfer function, and trained by Back-propagation algorithm. All experiments are conducted based on the "same structure" and "same-scale" principle. It means that in the experiments a global model is compared with a THME model whose experts have the same network structure, same number of hidden layers, and same number of neurons with the global model. In other words both models have the same number of degrees of freedom. For example, if the THME model uses 2 experts in RBF structure and each of them has 20 hidden neurons, it will compare to a single RBF network that has 40 hidden neurons. In the experiments with the Laser data, THME model been tested with 2 experts in RBF and MLP structure separately. For the Leuven data, the number of experts in THME is three for both expert structures.

The quality of prediction in the experiments was evaluated by the *RMSE* (Rooted Mean Squared Error) and *NMSE* (Normalized Mean Squared Error).

$$RMSE = \sqrt{1/T \sum_{t=1}^{T} (y_t - \hat{y}_t)^2} \tag{18}$$

$$NMSE = \frac{\sum_{t=1}^{T} (y_t - \hat{y}_t)^2}{\sum_{t=1}^{T} (y_t - 1/T \sum_{t=1}^{T} y_t)^2} \tag{19}$$

For the training process, in Figure 5 and Figure 6, we gave the state statues of both time series measured in probabilities Hence, by the end of step 3----"refined HMM-learning", the probability of each point being in each state was determined. Here we just give point 101 to 200 for Laser data in Figure 5 and 201 to 400 for Leuven data in Figure 6.

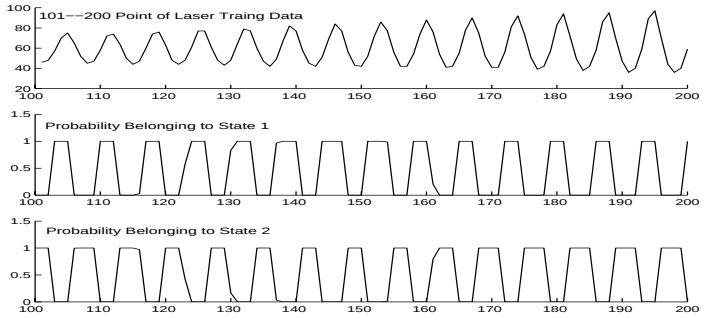

Figure 5. Probability for each state of Laser training data from point 101 to 200.

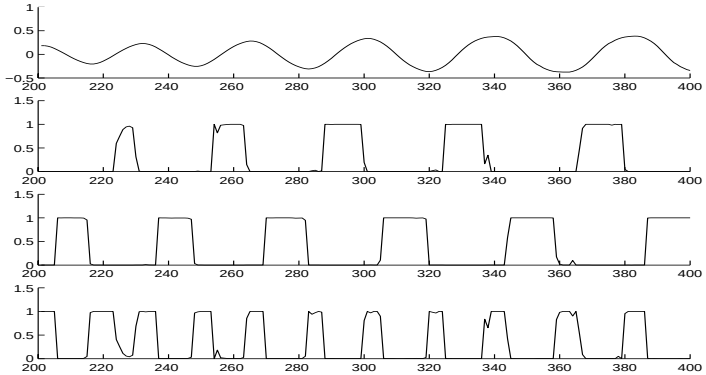

Figure 6. Probability for each state of Leuven training data from point 201 to 400..

For the prediction performance, Table 1 and Table 2 show the prediction results from the THME models with both RBF and MLP as experts and their corresponding global models. There are consistent improvements on the prediction quality over the corresponding single global models. In terms of state prediction, Figure 7 and Figure 8 show the prior probabilities of being in each state for both data sets. From the view of Mixture of Experts, they are the gating coefficients for each expert in the process of prediction. Here we show only the probabilities for the RBF structured experts. For the behaviour of each expert, we display the prediction errors from each expert in the THME models in Figure 9 and Figure 10. Here we just give the prediction errors for both time series when THME has RBF experts.

	Single Model		THME Model	
	N. H. N.	RMSE (NMSE)	N. H. N	RMSE (NMSE)
RBF	20	8.531 (0.039)	10/Expert	4.521 (0.011)
	50	8.387 (0.038)	25/ Expert	4.332 (0.010)
MLP	20	21.24 (0.242)	10/ Expert	16.15 (0.139)
	40	18.34 (0.181)	20/ Expert	13.34 (0.095)

Table 1. Prediction RMSE and NMSE on Laser data from global model and THME model (with RBF network and MLP as expert separately). "N. H. N." Means "Number of Hidden Neurons".

	Single Model		THME Model	
	N. H. N.	RMSE (NMSE)	N. H. N.	RMSE (NMSE)
RBF	30	0.0120 (0.0054)	10/ Expert	0.0091(0.0031)
	45	0.0137 (0.0071)	25/ Expert	0.0121(0.0054)
MLP	30	0.0491 (0.0910)	10/ Expert	0.0257(0.0248)
	45	0.0372 (0.0519)	25/ Expert	0.0281(0.0296)

Table 2. Prediction RMSE and NMSE on Leuven data from global model and THME model (with RBF network and MLP as expert separately). "N. H. N." Means "Number of Hidden Neurons".

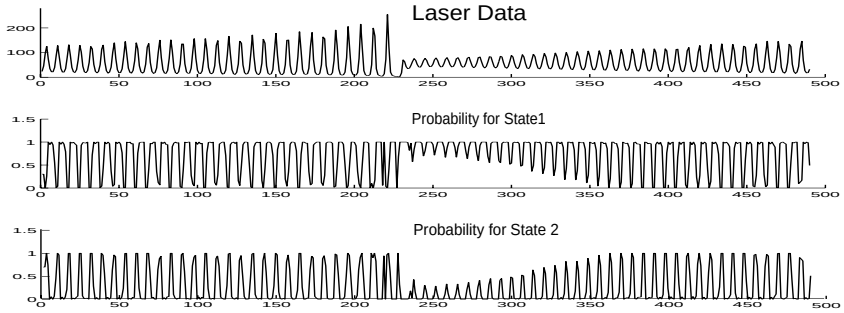

Figure 7. Prior Probability for each state on Laser data. The experts are in RBF structure

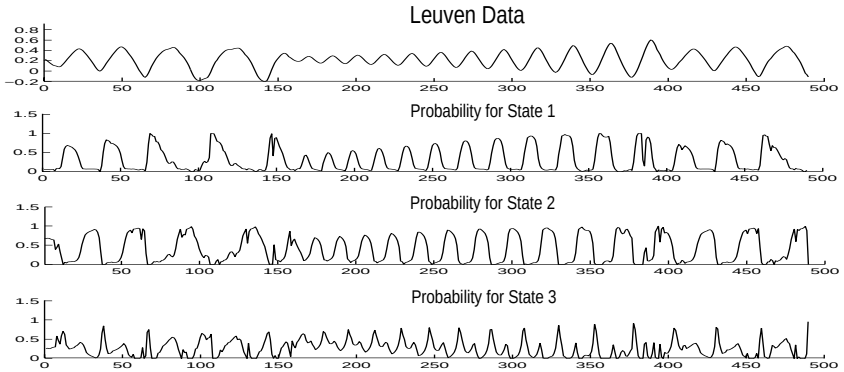

Figure 8. Prior Probability for each state on Leuven data. The experts are in RBF structure.

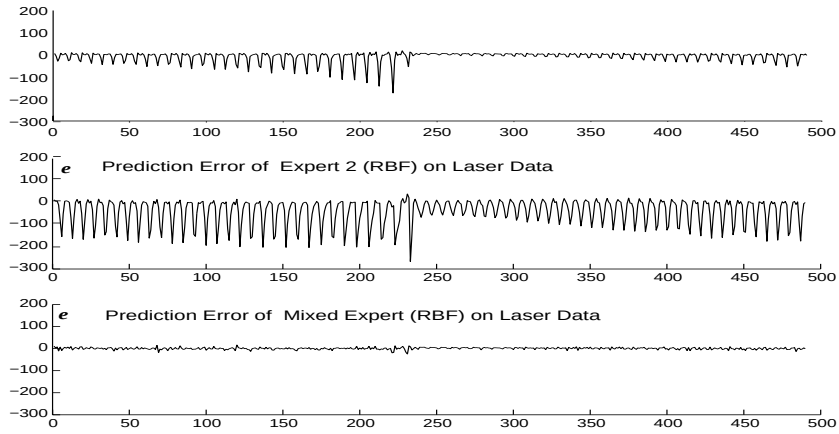

Figure 9. Prediction errors of all experts and mixed expert for Laser data

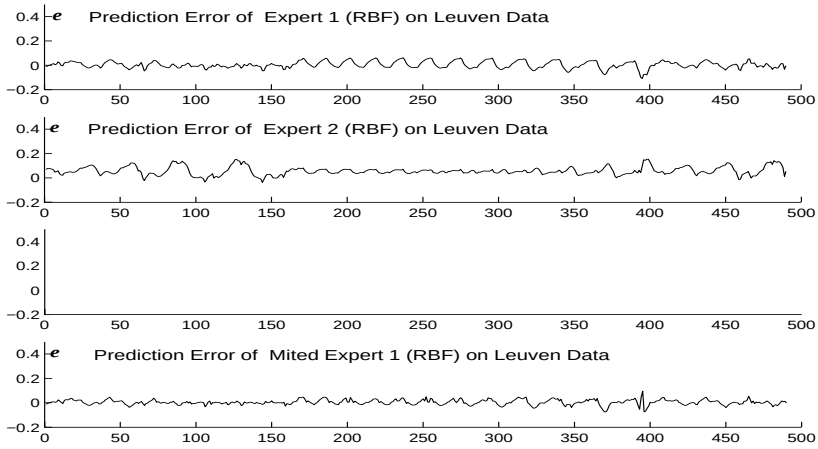

Figure 10. Prediction errors of all experts and mixed expert for Leuvan data

6 Conclusion and Discussion

The experiments show the ME model hosted by a HMM with time-variant transition property can be applied to enhance the quality of one-step-ahead time series predictions. Using the same network scale and structure, such as the number of hidden neurons and number of hidden layers, or the same number of degrees of freedom, the THME model can generate better predictions than a global model for the shown data sets. Additionally, it has been demonstrated that a connectionist network can be used to model the state transitions along a time series.

One question with the model is the computing cost and convergence property of training a series of input-related transition probability matrixes. As the updating of the transition probabilities is simultaneously happening at every time point, the time cost is tolerable and convergence speed is expeditious. For example, by the modified Baum-Welch algorithm, the HMM learning with 1000 points Laser data takes about 5 seconds in Matlab platform on a

1MHz PC, and the learning process needs about 10--20 iterations to get to convergence (Figure 11).

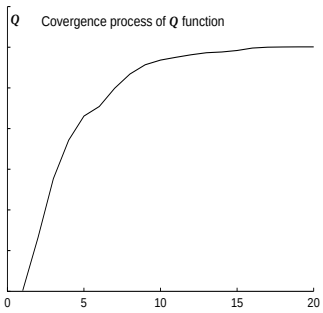

Figure 11. Converging of *Q* Function for Laser data when THME has RBF experts.

Another significant question with THME is how to choose the number of experts (and therefore the number of states). From our experience the number has a direct impact on prediction quality and the complexity of the models. More local experts allow a time series to be modelled more precisely (i.e. the training data can be more accurately described), however larger errors appear on the state-transition modelling. This trade-off has to be achieved manually by trial and error. Future work will consider how this trade-off can be quantified and tuned without manual intervention to achieve an appropriate level of generalization.

References

Baum, L.E., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains, *Annals of Mathematical Statistics* 41, 164-171.

Bengio, Y., and Frasconi, P. (1995). An input output HMM architecture. In Advances in Neural Information Processing Systems, G. Tesauro, Touretzky, M.D.S. and Leen, T.K., ed. Cambridge, MA, MIT Press, 427-434.

Bengio, Y., Frasconi, Paolo (1996). Input/Output HMMs for sequence processing, *IEEE Transactions on Neural Networks* 7(5), 1231-1249.

Bengio, Y., Lauzon, V., Ducharme. R., (2001). Experiments on the application of IOHMMS to model financial return series., *IEEE Transactions on Neural Networks* 12, 113-123.

Bezdek, J. (1981). *Pattern Recognition with Fuzzy Objective Function Algorithms*, New York, Plenum Press.

Bezdek, J., and Pal. S. (1992). *Fuzzy Models for Pattern Recognition*, IEEE Press.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, P. J. (1984). *Classification and Regression Trees*, CA, Wadsworth International Group.

Cao, L. (2003). Support vector machines experts for time series forecasting, *Neurocomputing* 51, 321-339.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society* Series B, 1-38.

Friedman, J. (1991). Multivariate Adaptive Regression Splines, *Annals of Statistics* 19, 1-142.

Hamilton, J. D. (1989). A new approach to the economic analysis of nonstationary time series and the business cycle, *Econometrica* 57, 357-384.

Hamilton, J. D. (1990). Analysis of time series subject to changes in regime, *Journal of Econometrics* 45, 39-70.

Hamilton, J. D., and Susmel, R. (1994). Autoregressive Conditional Heteroskedasticity and Changes in Regime, *Journal of Econometrics* 64, 307-333.

Hamilton, J. D. (1996). Specification testing in Markov-switching time-series models, *Journal of Econometrics* 70, 127-157.

Hübner, U., Weiss, C. O., Abraham, N. B., and Tang, D. (1994). Lorenz-like Chaos in NH 3-FIR Lasers. In Time Series Prediction: Forecasting the Future and Understanding the Past, A. S. Weigend, Gershenfeld, N. A., ed. MA, Addison-Wesley, 73-104.

Jordan, M. I., and Jacobs, R. A. (1991). Adaptive mixtures of local experts, *Neural Computation* 3, 79-87.

Jordan, M. I., and Jacobs, R. A. (1992). Hierarchies of adaptive experts. In Advances in Neural Information Processing Systems, J. Moody, Hanson, S., & Lipmann, R., ed. 985-992.

Jordan, M. I., and Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm, *Neural Computation* 6, 181-214.

Liehr, S., Pawelzik, K., Kohlmorgen, J., and Muller, K.-R. (1999). Hidden Markov Mixtures of Experts with an Application to EEG Recordings from Sleep, *Theory in Biosciences* 118, 246-260.

Liporace, L. A. (1982). Maximum likelihood estimation for multivariate observations of Markov Source, *IEEE Transactions on Information Theory* 28(5), 729-734.

Kohlmorgen, J., Müller, K.-R., Rittweger, J., Pawelzik, K. (2000). Identification of Nonstationary Dynamics in Physiological Recordings, *Biological Cybernetics* 83, 73-84.

McCullagh, P., and Nelder, J. A. (1989). *Generalised linear Models, Monographs on Statistics and Applied Probability*, Second edn London, Chapman and Hall.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition, *Proceedings of the IEEE* 77, 257-286.

Suykens, J. A. K., Huang, A., and Chua, L. O. (1997). A family of n-scroll attractors from a generalized Chua's circuit, *International Journal of Electronics and Communications* 51, 131-138.

Tong, H., and Lim, K. S. (1980). Threshold autoregression, Limit Cycles and cyclical data, *Journal of the Royal Statistical Society* B, 245-292.

Weigend, A. S., Mangeas, M., and Srivastava, A. N. (1996). Nonlinear gated experts for time series: discovering regimes and avoiding overfitting, *International Journal of Neural Systems* 6, 373-399.

Weigend, A. S., and Shi, S. (2000). Predicting Daily Probability Distributions of S&P500 Returns, *Journal of Forecasting* 19, 375-392.

Wolfgang, H. (1990). *Applied Nonparametric Regression*, Cambridge, Cambridge University Press.

Appendix A

Parameter updating in the modified Baum-Welth algorithm.

We define:

$$\gamma_i(t) = p(s_t = i \mid \boldsymbol{Y}, \boldsymbol{\lambda}')$$

$$= \frac{p(s_t = i, \boldsymbol{Y} \mid \boldsymbol{\lambda}')}{p(\boldsymbol{Y} \mid \boldsymbol{\lambda}')}$$

$$= \frac{\alpha_i(t)\beta_i(t)}{\sum_{i=1}^{M} p(s_t = i, \boldsymbol{Y} \mid \boldsymbol{\lambda}')}$$

$$= \frac{\alpha_i(t)\beta_i(t)}{\sum_{i=1}^{M} \alpha_i(t)\beta_i(t)} \qquad \text{(A-1)}$$

We define $\eta_{ij}(t)$ as the probability of being in state i at time t and being in state j at time $t+1$ for observing the whole sequence of observations with parameters $\boldsymbol{\lambda}'$.

$$\eta_{ij}(t) = p(s_t = i, s_{t+1} = j \mid \boldsymbol{Y}, \boldsymbol{\lambda}')$$

$$= \frac{p(s_t = i, s_{t+1} = j, \boldsymbol{Y} \mid \boldsymbol{\lambda}')}{p(\boldsymbol{Y} \mid \boldsymbol{\lambda}')}$$

$$= \frac{\alpha_i(t)a_{ij}(t+1)b_j(y_{t+1})\beta_j(t+1)}{\sum_{i=1}^{M}\sum_{j=1}^{M} \alpha_i(t)a_{ij}(t+1)b_j(y_{t+1})\beta_j(t+1)} \qquad \text{(A-2)}$$

Now return to Q function for Maximisation. As the parameters π_i, a_{ij}, and σ_i^2 are independently split into three terms in equation 9,10,11, we can optimise each term individual. The first term in Q function is:

$$Q_1 = \sum_{S \in \zeta} log\ \pi_{s_0} p(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}') = \sum_{i=1}^{M} log\ \pi_i p(\boldsymbol{Y}, s_0 = i \mid \boldsymbol{\lambda}') \qquad \text{(A-3)}$$

Adding the Lagrange multiplier δ and using the constraint that $\sum_{i=1}^{M} \pi_i = 1$ to maximise Q_1.

$$\frac{\partial}{\partial \pi_i}\left[\sum_{i=1}^{M} log\, \pi_i p(\boldsymbol{Y}, s_0 = i \mid \boldsymbol{\lambda}') + \delta\left(\sum_{i=1}^{M}(\pi_i - 1)\right)\right] = 0 \qquad \text{(A-4)}$$

We can get the updating formula for π_i:

$$\bar{\pi}_i = \frac{p(\boldsymbol{Y}, s_0 = i \mid \boldsymbol{\lambda}')}{p(\boldsymbol{Y} \mid \boldsymbol{\lambda}')} \qquad \text{(A-5)}$$

The second term in Q function becomes:

$$Q_2 = \sum_{\boldsymbol{S} \in \boldsymbol{\zeta}}\left(\sum_{t=1}^{T} log\, a_{s_{t-1}, s_t}\right) p(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}')$$

$$= \sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{t=1}^{T} log\ a_{ij}(t)\, p(\boldsymbol{Y}, s_{t-1} = i, s_t = j \mid \boldsymbol{\lambda}')$$

$$= \left\{ \begin{array}{l} [\log a_{11}(1)\, p(\boldsymbol{Y}, s_0=1, s_1=1|\boldsymbol{\lambda}') + \log a_{11}(2)\, p(\boldsymbol{Y}, s_1=1, s_2=1|\boldsymbol{\lambda}') + \ldots + \log a_{11}(T)\, p(\boldsymbol{Y}, s_{T-1}=1, s_T=1|\boldsymbol{\lambda}')] + \\ [\log a_{12}(1)\, p(\boldsymbol{Y}, s_0=1, s_1=2|\boldsymbol{\lambda}') + \log a_{12}(2)\, p(\boldsymbol{Y}, s_1=1, s_2=2|\boldsymbol{\lambda}') + \ldots + \log a_{12}(T)\, p(\boldsymbol{Y}, s_{T-1}=1, s_T=2|\boldsymbol{\lambda}')] + \\ \ldots + \\ [\log a_{1M}(1)\, p(\boldsymbol{Y}, s_0=1, s_1=M|\boldsymbol{\lambda}') + \log a_{1M}(2)\, p(\boldsymbol{Y}, s_1=1, s_2=M|\boldsymbol{\lambda}') + \ldots + \log a_{1M}(T)\, p(\boldsymbol{Y}, s_{T-1}=1, s_T=M|\boldsymbol{\lambda}')] \end{array} \right\} +$$

*(*for $i=1, j=1,2,\ldots,M, t=1,2,\ldots,T$)

$$\left\{ \begin{array}{l} [\log a_{21}(1)\, p(\boldsymbol{Y}, s_0=2, s_1=1|\boldsymbol{\lambda}') + \log a_{21}(2)\, p(\boldsymbol{Y}, s_1=2, s_2=1|\boldsymbol{\lambda}') + \ldots + \log a_{21}(T)\, p(\boldsymbol{Y}, s_{T-1}=2, s_T=1|\boldsymbol{\lambda}')] + \\ [\log a_{22}(1)\, p(\boldsymbol{Y}, s_0=2, s_1=2|\boldsymbol{\lambda}') + \log a_{22}(2)\, p(\boldsymbol{Y}, s_1=2, s_2=2|\boldsymbol{\lambda}') + \ldots + \log a_{22}(T)\, p(\boldsymbol{Y}, s_{T-1}=2, s_T=2|\boldsymbol{\lambda}')] + \\ \ldots + \\ [\log a_{2M}(1)\, p(\boldsymbol{Y}, s_0=2, s_1=M|\boldsymbol{\lambda}') + \log a_{2M}(2)\, p(\boldsymbol{Y}, s_1=2, s_2=M|\boldsymbol{\lambda}') + \ldots + \log a_{2M}(T)\, p(\boldsymbol{Y}, s_{T-1}=2, s_T=M|\boldsymbol{\lambda}')] \end{array} \right\} +$$

*(*for $i=2, j=1,2,\ldots,M, t=1,2,\ldots,T$)

$+, \ldots, +$

$$\left\{ \begin{array}{l} [\log a_{M1}(1)\, p(\boldsymbol{Y}, s_0=M, s_1=1|\boldsymbol{\lambda}') + \log a_{M1}(2)\, p(\boldsymbol{Y}, s_1=M, s_2=1|\boldsymbol{\lambda}') + \ldots + \log a_{M1}(T)\, p(\boldsymbol{Y}, s_{T-1}=M, s_T=1|\boldsymbol{\lambda}')] + \\ [\log a_{M2}(1)\, p(\boldsymbol{Y}, s_0=M, s_1=2|\boldsymbol{\lambda}') + \log a_{M2}(2)\, p(\boldsymbol{Y}, s_1=M, s_2=2|\boldsymbol{\lambda}') + \ldots + \log a_{M2}(T)\, p(\boldsymbol{Y}, s_{T-1}=M, s_T=2|\boldsymbol{\lambda}')] + \\ \ldots + \\ [\log a_{MM}(1)\, p(\boldsymbol{Y}, s_0=M, s_1=M|\boldsymbol{\lambda}') + \log a_{MM}(2)\, p(\boldsymbol{Y}, s_1=M, s_2=M|\boldsymbol{\lambda}') + \ldots + \log a_{MM}(T)\, p(\boldsymbol{Y}, s_{T-1}=M, s_T=M|\boldsymbol{\lambda}')] \end{array} \right\} +$$

(for $i=M, j=1,2,\ldots,M, t=1,2,\ldots,T$)

(A-6)

In a similar way, we use the Lagrange multiplier δ and the constraint that

$\sum_{j=1}^{M} a_{ij}(t) = 1$, then get:

$$\frac{\partial}{\partial a_{ij}(t)} \left[Q_2 + \delta \left(\sum_{j=1}^{M} a_{ij}(t) - 1 \right) \right] = 0 \qquad \text{(A-7)}$$

Then we can get following formula:

$$\frac{1}{a_{ij}(t)} p(\boldsymbol{Y}, s_{t-1}=i, s_t=j|\boldsymbol{\lambda}') + \delta = 0 \qquad \text{(A-8)}$$

So the updating function for $a_{ij}(t)$ can be gotten as follows:

$$\tilde{a}_{ij}(t) = \frac{p(\boldsymbol{Y}, s_{t-1}=i, s_t=j|\boldsymbol{\lambda}')}{p(\boldsymbol{Y}, s_{t-1}=i|\boldsymbol{\lambda}')}$$

$$= \frac{\eta_{ij}(t-1)}{\gamma_i(t)} \qquad \text{(A-9)}$$

The third term in Q function becomes:

$$Q_3 = \sum_{\boldsymbol{S} \in \zeta} \left[\sum_{t=1}^{T} \log\ b_{s_t}(y_t) \right] p(\boldsymbol{Y}, \boldsymbol{S}|\boldsymbol{\lambda}') = \sum_{i=1}^{M} \sum_{t=1}^{T} \log\ b_i(y_t)\, p(\boldsymbol{Y}, s_t=i|\boldsymbol{\lambda}')$$

To maximise it, we get:

$$\tilde{\sigma}_i^2 = \frac{\sum_{t=1}^{T} p(\boldsymbol{Y}, s_t = i \mid \boldsymbol{\lambda}')(y_t - \hat{y}(\boldsymbol{X}_t))^2}{\sum_{t=1}^{T} p(\boldsymbol{Y}, s_t = i \mid \boldsymbol{\lambda}')}$$

$$= \frac{\sum_{t=1}^{T} [\gamma_i(t)(y_t - \hat{y}(\boldsymbol{X}_t))^2]}{\sum_{t=1}^{T} \gamma_i(t)} \qquad \text{(A-10)}$$

Appendix B

Convergence property of the modified Baum-Welch algorithm

Convergence character of modified Baum-Welch algorithm is similar to that of classical Baum-Welch algorithm. For the issue of estimating parameters $\boldsymbol{\lambda}$ to get a set of observations $\boldsymbol{Y}$ with maximum possibility, the Modified Baum-Welch algorithm, in each EM cycle, maximises the likelihood function $Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}')$ and estimates corresponding parameters $\boldsymbol{\lambda}$ in the M-step. This is a process of finding critical point of Q function based on parameters value $\boldsymbol{\lambda}'$. For Gauss probability density *b*, Liparace (Liporace, 1982) has proven that $Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}')$ has a unique global maximum as a function of $\boldsymbol{\lambda}'$, and this maximum is the one and only one critical point. Hence there is $Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}') \geq Q(\boldsymbol{\lambda}', \boldsymbol{\lambda}')$ with the equality of $\boldsymbol{\lambda} = \boldsymbol{\lambda}'$. With this conclusion, we can prove that there are: $P(\boldsymbol{Y} \mid \boldsymbol{\lambda}) \geq P(\boldsymbol{Y} \mid \boldsymbol{\lambda}')$ with equality of $\boldsymbol{\lambda} = \boldsymbol{\lambda}'$ (see Theorem 1). By EM iteration both functions monotonic increase with updating $\boldsymbol{\lambda}$ and converge to a local maximum (see Theorem 2). So by the modified Baum-Welch algorithm, the likelihood function and the probability function could converge to (local) maximum.

- **Theorem 1**. By each EM iteration, for $\boldsymbol{\lambda}, \boldsymbol{\lambda}' \in \boldsymbol{\xi}$, there is:

  $p(\boldsymbol{Y} \mid \boldsymbol{\lambda}) \geq p(\boldsymbol{Y} \mid \boldsymbol{\lambda}')$

  with equality of $\boldsymbol{\lambda} = \boldsymbol{\lambda}'$.

  Proof:

  We define $\boldsymbol{\lambda}_l$ as the value of $\boldsymbol{\lambda}$ that gets the maximum of Q in l^{th} iteration, and $\boldsymbol{\lambda}_l'$ as the current value of the parameters used in the E step.

  As defined in equation 6,

  $$Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}') = \sum_{\boldsymbol{S} \in \boldsymbol{\zeta}} log\ p(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}) p(\boldsymbol{Y}, \boldsymbol{S} \mid \boldsymbol{\lambda}')$$

  In the l^{th} iteration, by *M* step there is:

  $$Q(\boldsymbol{\lambda}_l, \boldsymbol{\lambda}_l') \geq Q(\boldsymbol{\lambda}_l', \boldsymbol{\lambda}_l') \qquad \text{(B-1)}$$

  For any positive scalar *a*, there is: $\log(a) \leq (a - 1)$, so we can get:

  $$Q(\boldsymbol{\lambda}_l, \boldsymbol{\lambda}_l') - Q(\boldsymbol{\lambda}_l', \boldsymbol{\lambda}_l')$$

$$= \sum_{\boldsymbol{S}\in\zeta} \left\{ log\left[\frac{p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}_l)}{p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}_l^{'})} \right] \right\} p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}_l^{'})$$

$$\leq \sum_{\boldsymbol{S}\in\zeta} \left\{ \left[\frac{p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}_l)}{p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}_l^{'})} \right] - 1 \right\} p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}_l^{'})$$

$$= \sum_{\boldsymbol{S}\in\zeta} \left[p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}_l) - p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}_l^{'}) \right]$$

$$= p(\boldsymbol{Y} \mid \boldsymbol{\lambda}_l) - p(\boldsymbol{Y} \mid \boldsymbol{\lambda}_l^{'}) \qquad \text{(B-2)}$$

As $Q(\boldsymbol{\lambda}_l, \boldsymbol{\lambda}_l^{'}) - Q(\boldsymbol{\lambda}_l^{'}, \boldsymbol{\lambda}_l^{'}) \geq 0$, there is: $p(\boldsymbol{Y} \mid \boldsymbol{\lambda}) \geq p(\boldsymbol{Y} \mid \boldsymbol{\lambda}')$ and when $\boldsymbol{\lambda} = \boldsymbol{\lambda}'$, $p(\boldsymbol{Y} \mid \boldsymbol{\lambda}) = p(\boldsymbol{Y} \mid \boldsymbol{\lambda}')$

- **Theorem 2.** Q function and P function converge to local maximum. In other words: if $\boldsymbol{\lambda} = \boldsymbol{\lambda}'$ is a critical point of function Q, it is also a critical point of $p(\boldsymbol{Y} \mid \boldsymbol{\lambda})$. That is:

  If: $$\left. \frac{\partial Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}')}{\partial \boldsymbol{\lambda}} \right|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}'} = 0$$

  Then: $$\left. \frac{\partial p(\boldsymbol{Y} \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} \right|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}'} = 0$$

  Proof:

  $$Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}') = \sum_{\boldsymbol{S}\in\zeta} log\ p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}) p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}')$$

  $$\left. \frac{\partial Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}')}{\partial \boldsymbol{\lambda}} \right|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}'}$$

  $$= \sum_{\boldsymbol{S}\in\zeta} \left[\frac{1}{p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda})} \cdot \frac{\partial p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} \right]_{\boldsymbol{\lambda}=\boldsymbol{\lambda}'} \cdot p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda}')$$

  (B-3)

  $$= \sum_{\boldsymbol{S}\in\zeta} \left[\frac{\partial p(\boldsymbol{Y},\boldsymbol{S} \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} \right]_{\boldsymbol{\lambda}=\boldsymbol{\lambda}'}$$

  $$= \left. \frac{\partial p(\boldsymbol{Y} \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} \right|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}'}$$

  So if: $\left. \frac{\partial Q(\boldsymbol{\lambda}, \boldsymbol{\lambda}')}{\partial \boldsymbol{\lambda}} \right|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}'} = 0$, there is: $\left. \frac{\partial p(\boldsymbol{Y} \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} \right|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}'} = 0$.