MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY
and
CENTER FOR BIOLOGICAL AND COMPUTATIONAL LEARNING
DEPARTMENT OF BRAIN AND COGNITIVE SCIENCES

A.I. Memo No. 1458 November 18, 1993
C.B.C.L. Memo No. 87

Convergence Results for the EM Approach to Mixtures of Experts Architectures

Michael I. Jordan and Lei Xu

Abstract

The Expectation-Maximization (EM) algorithm is an iterative approach to maximum likelihood parameter estimation. Jordan and Jacobs (1993) recently proposed an EM algorithm for the mixture of experts architecture of Jacobs, Jordan, Nowlan and Hinton (1991) and the hierarchical mixture of experts architecture of Jordan and Jacobs (1992). They showed empirically that the EM algorithm for these architectures yields significantly faster convergence than gradient ascent. In the current paper we provide a theoretical analysis of this algorithm. We show that the algorithm can be regarded as a variable metric algorithm with its searching direction having a positive projection on the gradient of the log likelihood. We also analyze the convergence of the algorithm and provide an explicit expression for the convergence rate. In addition, we describe an acceleration technique that yields a significant speedup in simulation experiments.

Copyright © Massachusetts Institute of Technology, 1993

This report describes research done at the Dept. of Brain and Cognitive Sciences, the Center for Biological and Computational Learning, and the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology. Support for CBCL is provided in part by a grant from the NSF (ASC–9217041). Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Dept. of Defense. The authors were supported by a grant from the McDonnell-Pew Foundation, by a grant from ATR Human Information Processing Research Laboratories, by a grant from Siemens Corporation, by by grant IRI-9013991 from the National Science Foundation, by grant N00014-90-J-1942 from the Office of Naval Research, and by NSF grant ECS-9216531 to support an Initiative in Intelligent Control at MIT. Michael I. Jordan is a NSF Presidential Young Investigator.

Introduction

Although neural networks are capable in principle of representing complex nonlinear functions, the time required to train a complex network does not always scale well with problem size and the solution obtained does not always reveal the structure in the problem. Moreover, it is often difficult to express prior knowledge in the language of fully-connected neural networks. Achieving better scaling behavior, better interpretability of solutions and better ways of incorporating prior knowledge may require a more modular approach in which the learning problem is decomposed into sub-problems. Such an approach has been used with success in the statistics literature and the machine learning literature, where decision-tree algorithms such as CART and ID3 and multivariate spline algorithms such as MARS have running times that can be orders of magnitude faster than neural network algorithms and often yield simple, interpretable solutions (Breiman, Friedman, Olshen & Stone, 1984; Friedman, 1991; Quinlan, 1986).

A general strategy for designing modular learning systems is to treat the problem as one of combining multiple models, each of which is defined over a local region of the input space. Jacobs, Jordan, Nowlan and Hinton (1991) introduced such a strategy with their "mixture of experts" (ME) architecture for supervised learning. The architecture involves a set of function approximators ("expert networks") that are combined by a classifier ("gating network"). These networks are trained simultaneously so as to split the input space into regions where particular experts can specialize. Jordan and Jacobs (1992) extended this approach to a recursively-defined architecture in which a tree of gating networks combine the expert networks into successively larger groupings that are defined over nested regions of the input space. This "hierarchical mixture of experts" (HME) architecture is closely related to the decision tree and multivariate spline algorithms.

The problem of training a mixture of experts architecture can be treated as a maximum likelihood estimation problem. Both Jacobs et al. (1991) and Jordan and Jacobs (1992) derived learning algorithms by computing the gradient of the log likelihood for their respective architectures. Empirical tests revealed that although the gradient approach succeeded in finding reasonable parameter values in particular problems, the convergence rate was not significantly better than that obtained by using gradient methods in multi-layered neural network architectures. The gradient approach did not appear to take advantage of the modularity of the architecture. An alternative to the gradient approach was proposed by Jordan and Jacobs (in press), who introduced an Expectation-Maximization (EM) algorithm for mixture of experts architectures. EM is a general technique for maximum likelihood estimation that can often yield simple and elegant algorithms (Baum, Petrie, Soules & Weiss, 1970; Dempster, Laird & Rubin, 1977). For mixture of experts architectures, the EM algorithm decouples the estimation process in a manner that fits well with the modular structure of the architecture. Moreover, Jordan and Jacobs (in press) observed a significant speedup over gradient techniques.

In this paper, we provide further insight into the EM approach to mixtures of experts architectures via a set of convergence theorems. We study a particular variant of the EM algorithm proposed by Jordan and Jacobs (in press) and demonstrate a relationship between this algorithm and gradient ascent. We also provide theorems on the convergence rate of the algorithm and provide explicit formulas for the constants.

The remainder of the paper is organized as follows. Section 2 introduces the ME model. The EM algorithm for this architecture is derived and two convergence theorems are presented. Section 3 presents an analogous derivation and a set of convergence results for the HME model. Section 4 introduces two acceleration techniques for improving convergence and presents the results of numerical experiments. Section 5 presents our conclusions.

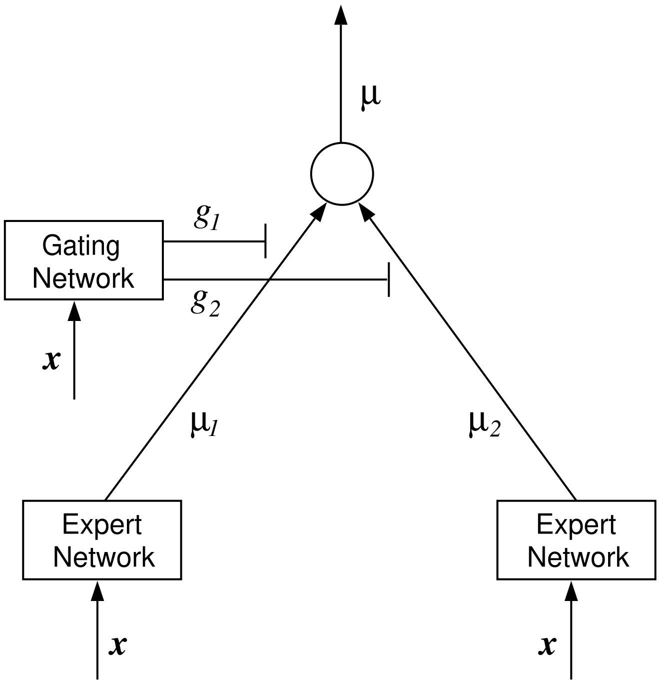

Figure 1: The mixture of experts architecture. The total output $\boldsymbol{\mu}$ is the weighted sum of the expert network outputs: $\boldsymbol{\mu} = g_1\boldsymbol{\mu}_1 + g_2\boldsymbol{\mu}_2$, where the weights are the gating network outputs g_1 and g_2.

Theoretical analysis of an EM algorithm for the mixture of experts architecture

Network learning based on maximum likelihood estimation

We begin by studying the non-hierarchical case. As shown in Figure 1, the mixture of experts (ME) architecture is comprised of K expert networks, each of which solves a function approximation problem over a local region of the input space. To each expert network we associate a probabilistic model that relates input vectors $\mathbf{x} \in R^n$ to output vectors $\mathbf{y} \in R^m$. We denote these probabilistic models as follows

$$P(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_j), j = 1, 2, \cdots, K,$$

where the $\boldsymbol{\theta}_j$ are parameter vectors. Each of these probability densities is assumed to belong to the exponential family of densities (Jordan & Jacobs, in press). The j^{th} expert network produces as output a parameter vector $\boldsymbol{\mu}_j$

$$\boldsymbol{\mu}_j = \mathbf{f}_j(\mathbf{x}, \boldsymbol{\theta}_j), \ j = 1, 2, \cdots, K,$$

which is the location parameter for the j^{th} probability density. In the current paper, as in Jordan and Jacobs (in press), we treat the case in which the functions $\mathbf{f}_j$ are linear in the parameters. We extend our results to the case of experts that are nonlinear in the parameters in the Appendix.

We also assume, for simplicity, that the probability densities $P(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_j)$ are gaussian, implying that the location parameter is simply the mean. We associate a covariance matrix Σ_j

with each expert network, yielding the following probabilistic model for expert j

$$P(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_j) = (2\pi \det \Sigma_j)^{-\frac{1}{2}} \exp\{-\frac{1}{2}[\mathbf{y} - \mathbf{f}_j(\mathbf{x}, \boldsymbol{\theta}_j)]^T \Sigma_j^{-1} [\mathbf{y} - \mathbf{f}_j(\mathbf{x}, \boldsymbol{\theta}_j)]\}. \tag{1}$$

The ME architecture also utilizes a auxiliary network known as a *gating network*, whose job it is to partition the input space into regions corresponding to the various expert networks. This is done by assigning a probability vector $[g_1, g_2, \cdots, g_K]^T$ to each point in the input space. In particular, the gating net implements a parameterized function $\mathbf{s} : R^n \rightarrow R^K$ and a normalizing function $\mathbf{g} : R^K \rightarrow R^K$ such that

$$g_j = g_j(\mathbf{x}, \boldsymbol{\theta}_g) = \frac{e^{s_j}}{\sum_{i=1}^K e^{s_i}}, \; j = 1, \cdots, K, \tag{2}$$

which satisfies

$$\sum_{j=1}^K g_j(\mathbf{x}, \boldsymbol{\theta}_g) = 1, \quad \text{for any } \mathbf{x}, \boldsymbol{\theta}_g. \tag{3}$$

In the current paper we focus on the case in which the function $\mathbf{s}$ is linear (cf. Jordan & Jacobs, in press). In this case the boundaries $g_i = g_{i'}$ are planar and the function $\mathbf{g}$ can be viewed as a smoothed piecewise-planar partitioning of the input space.

Training data are assumed to be generated according to the following probability model. We assume that for a given $\mathbf{x}$, a label j is selected with probability $P(j|\mathbf{x}) = g_j(\mathbf{x}, \boldsymbol{\theta}_g)$. An output $\mathbf{y}$ is then chosen with probability $P(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_j)$. Thus the total probability of observing $\mathbf{y}$ from $\mathbf{x}$ is given by the following finite mixture density

$$P(\mathbf{y}|\mathbf{x}) = \sum_{j=1}^K P(j|\mathbf{x}) P(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_j) = \sum_{j=1}^K g_j(\mathbf{x}, \boldsymbol{\theta}_g) P(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_j). \tag{4}$$

A training set $\mathcal{Y} = \{(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}), t = 1, \cdots, N\}$ is assumed to be generated as an independent set of draws from this mixture density. Thus the total probability of the training set, for a specified set of input vectors $\{\mathbf{x}^{(t)}\}_{t=1}^N$, is given by the following likelihood function

$$\begin{aligned} L = P(\{\mathbf{y}^{(t)}\}_1^N | \{\mathbf{x}^{(t)}\}_1^N) &= \prod_{t=1}^N P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}) && (5) \\ &= \prod_{t=1}^N \sum_{j=1}^K g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g) P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j). && (6) \end{aligned}$$

The learning algorithms that we discuss are all maximum likelihood estimators. That is, we treat learning as the problem of finding parameters $\boldsymbol{\theta}_g$, $\boldsymbol{\theta}_j$, and Σ_j to maximize L, or, more conveniently, to maximize the log likelihood $l = \ln L$

$$l(\boldsymbol{\Theta}, \mathcal{Y}) = \sum_{t=1}^N \ln \sum_{j=1}^K g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g) P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j),$$

where $\boldsymbol{\Theta} = [\boldsymbol{\theta}_g, \boldsymbol{\theta}_1, \boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_K, \Sigma_1, \Sigma_2, \cdots, \Sigma_K]^T$.

Given the probability model in Eq. 4, the expected value of the output is given as follows

$$\boldsymbol{\mu} \equiv E[\mathbf{y}|\mathbf{x}] = \sum_{j=1}^K g_j(\mathbf{x}, \boldsymbol{\theta}_g) \boldsymbol{\mu}_j.$$

This motivates using the weighted output of the expert networks as the total output of the ME architecture (cf. Figure 1).

The model in Eqs. (4) and (1) is a finite gaussian mixture model. It is interesting to compare this model to a related gaussian mixture model that is widely studied in statistics; i.e., the model

$$P(\mathbf{x}) = \sum_{j=1}^{K} \alpha_j P_j^{(k)}(\mathbf{x}|\boldsymbol{\theta}_j), \quad \alpha_j \geq 0, \quad \sum_{j=1}^{K} \alpha_j = 1. \tag{7}$$

The difference between these models is clear: the α_j's in Eq. (7) are independent of the input vectors, while the g_j's in Eq. (4) are conditional on $\mathbf{x}$ (they represent the probabilities $P(j|\mathbf{x})$). Thus model (7) represents a unconditional probability, appropriate for unsupervised learning, while model (4) represents a conditional probability, appropriate for supervised learning.

There is another model studied in statistics, the *switching regression* model (Quandt & Ramsey, 1972, 1978, De Veaux, 1986), that is intermediate between model (7) and model (4). The switching regression model is given as follows

$$P(y|\mathbf{x}) = \lambda P(y|\mathbf{x}, \boldsymbol{\theta}_1) + (1-\lambda) P(y|\mathbf{x}, \boldsymbol{\theta}_2), \tag{8}$$

where the $P(y|\mathbf{x}, \boldsymbol{\theta}_1)$ are univariate gaussians and the mean of each gaussian is assumed to be linear in $\mathbf{x}$. This model assumes that the data pair $\{y, \mathbf{x}\}$ is generated from a pair of linear regression models through a random switch which turns to one side with probability λ and to the other side with probability $1-\lambda$. This model can be generalized to allow for a multinomial switch

$$P(\mathbf{y}|\mathbf{x}) = \sum_{j=1}^{K} \alpha_j P(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_j), \tag{9}$$

where $\alpha_j \geq 0$, $\sum_{j=1}^{K} \alpha_j = 1$ and $P(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_j)$ is given by Eq. (1). The difference between switching regression and the ME model is that the switching regression model assumes that the setting of the switch is independent of the input vector. This assumption does not allow for piecewise variation in the form of the regression surface; all of the regression components contribute throughout the input space. Switching regression can be viewed as one end of a continuum in which the overlap in the regression components is total; decision tree models (e.g., Breiman et al., 1984) are the other end of the continuum in which the overlap is zero. The ME model interpolates smoothly between these extremes.

An EM algorithm for training the mixture of experts

In many estimation problems the likelihood is a complicated nonlinear function of the parameters. Parameter estimation in such cases usually involves some sort of numerical optimization technique, typically gradient ascent. An alternative to gradient techniques, applicable in many situations, is the "Expectation-Maximization" or "EM" algorithm (Baum, Petrie, Soules & Weiss, 1970; Dempster, Laird & Rubin, 1977). EM is based on the idea of solving a succession of simplified problems that are obtained by augmenting the original observed variables with a set of additional "hidden" variables. Unconditional mixture models are particularly amenable to the EM approach (Redner & Walker, 1984) and, as observed by Jordan and Jacobs (in press), the conditional mixture of experts model is also amenable to an EM approach.

Given an observed data set $\mathcal{Y}$, we augment $\mathcal{Y}$ with a set of additional variables $\mathcal{Y}_{mis}$, called "missing" or "hidden" variables, and consider a maximum likelihood problem for a "complete-data" set $\mathcal{Z} = \{\mathcal{Y}, \mathcal{Y}_{mis}\}$ (cf. Little & Rubin, 1987). We choose the missing variables in such a way that the resulting "complete-data log likelihood," given by $l_c(\boldsymbol{\Theta}, \mathcal{Z}) = \ln P(\mathcal{Y}, \mathcal{Y}_{mis}|\boldsymbol{\Theta})$, is easy to maximize with respect to $\boldsymbol{\Theta}$. The probability model $P(\mathcal{Y}, \mathcal{Y}_{mis}|\boldsymbol{\Theta})$ must be chosen so that its marginal distribution across $\mathcal{Y}$, referred to in this context as the "incomplete-data" likelihood, is the original likelihood

$$P(\mathcal{Y}|\boldsymbol{\Theta}) = \int P(\mathcal{Y}, \mathcal{Y}_{mis}|\boldsymbol{\Theta}) d\mathcal{Y}_{mis}. \tag{10}$$

In deriving an update to the parameters based on the complete-data log likelihood, we first note that we cannot work directly with the complete-data log likelihood, because this likelihood is a random function of the missing random variables $\mathcal{Y}_{mis}$. The idea is to average out $\mathcal{Y}_{mis}$, that is, to maximize the *expected* complete-data log likelihood $E_{\mathcal{Y}_{mis}}[\ln P(\mathcal{Y}, \mathcal{Y}_{mis}|\boldsymbol{\Theta})]$. This idea motivates the EM algorithm.

The EM algorithm is an iterative algorithm consisting of two steps:

- The Expectation (E) step, which computes the following conditional expectation of the log likelihood

$$\begin{aligned} Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) &= E_{\mathcal{Y}_{mis}}\{\ln P(\mathcal{Z}|\boldsymbol{\Theta})|\mathcal{Y}, \boldsymbol{\Theta}^{(k)}\} \\ &= \int P(\mathcal{Y}_{mis}|\mathcal{Y}, \boldsymbol{\Theta}^{(k)}) \ln P(\mathcal{Z}|\boldsymbol{\Theta}) d\mathcal{Y}_{mis} \end{aligned} \tag{11}$$

  where $\boldsymbol{\Theta}^{(k)}$ is the value of the parameter vector at iteration k.

- The Maximization (M) step, which computes

$$\boldsymbol{\Theta}^{(k+1)} = \arg\max_{\boldsymbol{\Theta}} \; Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}). \tag{12}$$

The M step chooses a parameter value that increases the Q function; the expected value of the complete-data log likelihood. Dempster, Laird and Rubin (1977) proved that an iteration of EM also increases the original log likelihood l. That is,

$$l(\boldsymbol{\theta}^{(k+1)}; \mathcal{Y}) \geq l(\boldsymbol{\theta}^{(k)}; \mathcal{Y}).$$

Thus the likelihood l increases monotonically along the sequence of parameter estimates generated by an EM algorithm.

Although in many cases the solution to the M step can be obtained analytically, in other cases an iterative inner loop is required to optimize Q. Another possibility is to simply increase the value of Q during the M step

$$Q(\boldsymbol{\Theta}^{(k+1)}|\boldsymbol{\Theta}^{(k)}) > Q(\boldsymbol{\Theta}^{(k)}|\boldsymbol{\Theta}^{(k)}) \tag{13}$$

by some means, for example by gradient ascent or by Newton's method. An algorithm with an M-step given by Eq. (13) is referred to as a *generalized EM (GEM)* algorithm (Dempster, Laird & Rubin, 1977).

For the ME architecture we choose the missing data to be a set of indicator random variables $\mathcal{Y}_{mis} = \{I_j^{(t)}, j = 1, \cdots, K, \; t = 1, \cdots, N\}$ with

$$I_j^{(t)} = \begin{cases} 1, & \text{if } \mathbf{y}^{(t)} \text{ is generated from the } j\text{-th model given by Eq. (1),} \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

and

$$\sum_{j=1}^{K} I_j^{(t)} = 1, \text{ for each } t.$$

We assume that the distribution of the complete data is given as follows

$$P(\mathcal{Z}|\boldsymbol{\Theta}) = \prod_{t=1}^{N}\prod_{j=1}^{K}[g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g)P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)]^{I_j^{(t)}}.$$

It is easy to verify that this distribution satisfies Eq. (10).

From Eq. (11), we also obtain

$$\begin{aligned}
Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) &= E_{\mathcal{Y}_{mis}}\{\ln P(Z|\boldsymbol{\Theta})|\mathcal{Y}, \boldsymbol{\Theta}^{(k)}\} \\
&= \sum_{t=1}^{N}\sum_{j=1}^{K} h_j^{(k)}(t)\ln[g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g)P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)] \\
&= \sum_{t=1}^{N}\sum_{j=1}^{K} h_j^{(k)}(t)\ln g_j(\mathbf{x}^{(t)}) + \sum_{t=1}^{N} h_1^{(k)}(t)\ln P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_1) \\
&+ \cdots + \sum_{t=1}^{N} h_K^{(k)}(t)\ln P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_K), \qquad (15)
\end{aligned}$$

where

$$\begin{aligned}
h_j^{(k)}(t) &= E[I_j^{(t)}|\mathcal{Y}, \boldsymbol{\Theta}^{(k)}] = P(j|\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \\
&= \frac{g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\sum_{i=1}^{K} g_i(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_i^{(k)})}, \qquad (16)
\end{aligned}$$

where $P(j|\mathbf{x}^{(t)}, \mathbf{y}^{(t)})$ denotes the probability that the pair $\{\mathbf{x}^{(t)}, \mathbf{y}^{(t)}\}$ comes from the j^{th} probability model. Note that we always have $h_j^{(k)}(t) > 0$.

With the Q function in hand, we now investigate the implementation of the M step. From Eq. (15), Eq. (1) and Eq. (2), we have

$$\begin{aligned}
\frac{\partial Q}{\partial \boldsymbol{\theta}_g} &= \sum_{t=1}^{N}\sum_{j=1}^{K} h_j^{(k)}(t)\frac{\partial g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g)}{\partial \boldsymbol{\theta}_g}/g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g) \\
&= \sum_{t=1}^{N}\sum_{j=1}^{K} h_j^{(k)}(t)[\frac{\partial s_j}{\partial \boldsymbol{\theta}_g} - \sum_{i=1}^{K} g_i(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g)\frac{\partial s_i}{\partial \boldsymbol{\theta}_g}] \\
&= \sum_{t=1}^{N}\sum_{j=1}^{K}[h_j^{(k)}(t) - g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g)]\frac{\partial s_j}{\partial \boldsymbol{\theta}_g}, \qquad (17)
\end{aligned}$$

$$\begin{aligned}
\frac{\partial Q}{\partial \boldsymbol{\theta}_j} &= \sum_{t=1}^{N} h_j^{(k)}(t)\frac{\partial P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}_j}/P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j) \\
&= \sum_{t=1}^{N} h_j^{(k)}(t)\frac{\partial \mathbf{f}_j{}^T(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}_j}\Sigma_j^{-1}[\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)], \quad j = 1, \cdots, K, \qquad (18)
\end{aligned}$$

and

$$\frac{\partial Q}{\partial \Sigma_j} = \sum_{t=1}^{N} h_j^{(k)}(t)\frac{\partial P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)}{\partial \Sigma_j}/P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)$$

$$
\begin{aligned}
&= -\frac{1}{2}\sum_{t=1}^{N} h_j^{(k)}(t)\Sigma_j^{-1}\{\Sigma_j - [\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)][\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)]^T\}\Sigma_j^{-1}, \\
j &= 1, \cdots, K. \qquad (19)
\end{aligned}
$$

By letting $\frac{\partial Q}{\partial \Sigma_j}|_{\Sigma_j = \Sigma_j^{(k+1)}} = 0$, we obtain the update for the covariance matrices

$$
\Sigma_j^{(k+1)} = \frac{1}{\sum_{t=1}^{N} h_j^{(k)}(t)} \sum_{t=1}^{N} h_j^{(k)}(t)[\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)][\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)]^T. \qquad (20)
$$

Assuming that the training set $\mathcal{Y}$ is generated by a mixture model, we note that when the the sample number N is sufficiently large (relative to the dimension of $\mathbf{y}$), the space spanned by the N vectors $[\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)], t = 1, \cdots, N$ will be of full dimension with probability one. Recalling that $h_j^{(k)}(t) > 0$ we observe that when the sample number N is sufficiently large the matrices $\Sigma_j^{(k+1)}$ are therefore positive definite with probability one.

Next, by letting $\frac{\partial Q}{\partial \boldsymbol{\theta}_j}|_{\boldsymbol{\theta}_j = \boldsymbol{\theta}_j^{(k+1)}} = 0$, we obtain

$$
\sum_{t=1}^{N} h_j^{(k)}(t) \frac{\partial \mathbf{f}_j^{\,T}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}_j} (\Sigma_j^{(k)})^{-1}[\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)] = 0, \qquad (21)
$$

which we can solve explicitly given our assumption that the expert networks are linear

$$
\boldsymbol{\theta}_j^{(k+1)} = (R_j^{(k)})^{-1}\mathbf{c}_j^{(k)} \qquad (22)
$$

where

$$
\mathbf{c}_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)}(t) X_t (\Sigma_j^{(k)})^{-1}\mathbf{y}^{(t)}, \qquad (23)
$$

$$
R_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)}(t) X_t (\Sigma_j^{(k)})^{-1} X_t^T, \qquad (24)
$$

and

$$
X_t^T = \left\{ \begin{array}{cccccc|cccccc}
(\mathbf{x}^{(t)})^T & 0 & \cdots & \cdots & & 0 & 1 & 0 & \cdots & \cdots & & 0 \\
0 & (\mathbf{x}^{(t)})^T & 0 & \cdots & & 0 & 0 & 1 & 0 & \cdots & & 0 \\
\vdots & \vdots & & & & \vdots & \vdots & & & & & \vdots \\
0 & \cdots & & \cdots & 0 & (\mathbf{x}^{(t)})^T & 0 & \cdots & \cdots & & 0 & 1
\end{array} \right\}. \qquad (25)
$$

Note that $R_j^{(k)}$ is invertible with probability one when the sample size N is sufficiently large.

Finally, let us consider the update for $\boldsymbol{\theta}_g$. Jordan and Jacobs (in press) observed that the gating network is a specific form of a generalized linear model, in particular a *multinomial logit* model (cf. McCullagh & Nelder, 1984). Multinomial logit models can be fit efficiently with a variant of Newton's method known as *iteratively reweighted least squares* (IRLS). For the purposes of the current paper, we simply write the generic form of a Newton update and refer the reader to Jordan and Jacobs (in press) for further details on IRLS. Note also that Jordan and Jacobs (in press) assume that the inner loop of IRLS fitting runs to completion. In the current paper we address only the case in which a single IRLS step is taken in the inner loop. The form of this IRLS step is generalized to allow a learning rate parameter.[1]

[1] Thus the algorithm that we analyze in this paper is, strictly speaking, a GEM algorithm.

The update for the gating network parameters is obtained as follows. Denote the gradient vector at iteration k as

$$\mathbf{e}_g^{(k)} = \sum_{t=1}^{N} \sum_{j=1}^{K} [h_j^{(k)}(t) - g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})] \frac{\partial s_j}{\partial \boldsymbol{\theta}_g^{(k)}}, \tag{26}$$

and the Hessian matrix at iteration k as

$$R_g^{(k)} = \sum_{t=1}^{N} \sum_{j=1}^{K} g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})[1 - g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})] \frac{\partial s_j}{\partial \boldsymbol{\theta}_g} \frac{\partial s_j}{\partial \boldsymbol{\theta}_g^T}. \tag{27}$$

Then the generalized IRLS update is given as follows

$$\boldsymbol{\theta}_g^{(k+1)} = \boldsymbol{\theta}_g^{(k)} + \gamma_g (R_g^{(k)})^{-1} \mathbf{e}_g^{(k)}, \tag{28}$$

where γ_g is a learning rate.

In summary, the parameter update for the model Eq. (4) is given as follows

---

Algorithm 1

1. (The E step): Compute the $h_j^{(k)}(t)$'s by Eq. (16).

2. (The M step): Compute $\Sigma_j^{(k+1)}$ by Eq. (20), compute $\boldsymbol{\theta}_g^{(k+1)}$ by Eq. (28), and also compute $\boldsymbol{\theta}_j^{(k+1)}, j = 1, \cdots, K$ by Eq. (22).

Before closing this section, let us return to the switching regression model (Eq. 9). Following the same procedure as above, we obtain the following EM algorithm for switching regression.

Algorithm 2

1. (The E step): Compute the $h_j^{(k)}(t)$'s by

$$h_j^{(k)}(t) = \frac{\alpha_j^{(k)}(\mathbf{x}^{(t)}) P(\mathbf{y}^{(t)} | \mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\sum_{i=1}^{K} \alpha_i^{(k)}(\mathbf{x}^{(t)}) P(\mathbf{y}^{(t)} | \mathbf{x}^{(t)}, \boldsymbol{\theta}_i^{(k)})}. \tag{29}$$

2. (The M step): Compute $\Sigma_j^{(k+1)}$ by Eq. (20), and let

$$\alpha_j^{(k+1)}(\mathbf{x}^{(t)}) = h_j^{(k)}(t). \tag{30}$$

Obtain $\boldsymbol{\theta}_j^{(k+1)}, j = 1, \cdots, K$ in the same manner as for model Eq. (4).

We see that the EM algorithm for switching regression is simpler, because the $\alpha_j^{(k)}(\mathbf{x}^{(t)})$'s are not constrained through a common parameter $\boldsymbol{\theta}_g$ as in the ME model.

Theoretical convergence results

In this section we provide a number of convergence results for the algorithm presented in the previous section. We study both the convergence and the convergence rate of the algorithm. In the Appendix we extend these results to a number of related algorithms.

We begin with a convergence theorem that establishes a relationship between the EM algorithm and gradient ascent.

Theorem 1 *For the model given by Eq. (4) and the learning algorithm given by Algorithm 1, we have:*

$$\begin{aligned}
\boldsymbol{\theta}_g^{(k+1)} - \boldsymbol{\theta}_g^{(k)} &= P_g^{(k)} \frac{\partial l}{\partial \boldsymbol{\theta}_g}|_{\boldsymbol{\theta}_g = \boldsymbol{\theta}_g^{(k)}}, \\
\boldsymbol{\theta}_j^{(k+1)} - \boldsymbol{\theta}_j^{(k)} &= P_j^{(k)} \frac{\partial l}{\partial \boldsymbol{\theta}_j}|_{\boldsymbol{\theta}_j = \boldsymbol{\theta}_j^{(k)}}, \ j = 1, \cdots, K, \\
vec[\Sigma_j^{(k+1)}] - vec[\Sigma_j^{(k)}] &= P_{\Sigma_j}^{(k)} \frac{\partial l}{\partial vec[\Sigma_j]}|_{\Sigma_j = \Sigma_j^{(k)}}, \ j = 1, \cdots, K,
\end{aligned} \tag{31}$$

where $l = \ln L$ is given by Eq. (6), Eq. (4) and Eq. (1), and "vec[A]" denotes the vector obtained by stacking the column vectors of the matrix A.

Moreover, assuming that the training set $\mathcal{Y}$ is generated by the mixture model of Eqs. (4) and (1) and assuming that the number N is sufficiently large, we have that $P_g^{(k)}$ is a positive definite matrix, and $P_j^{(k)}, P_{\Sigma_j}^{(k)}$, $j = 1, \cdots, K$ are positive definite matrices with probability one. Specifically, they take the following values:

(i) $P_g^{(k)} = \gamma_g (R_g^{(k)})^{-1}$ with $R_g^{(k)}$ given by Eq. (28).

(ii) For $j = 1, \cdots, K$, $P_j^{(k)} = (R_j^{(k)})^{-1}$ with $R_j^{(k)}$ given by Eq. (22).

(iii) For $j = 1, \cdots, K$,

$$P_{\Sigma_j}^{(k)} = \frac{2}{\sum_{t=1}^N h_j^{(k)}(t)} \Sigma_j^{(k)} \otimes \Sigma_j^{(k)} \tag{32}$$

where "$\otimes$" denotes the Kronecker product. For a $m \times n$ matrix A and $q \times m$ matrix B, the Kronecker product $A \otimes B$ is defined as

$$A \otimes B = \left\{ \begin{array}{cccc} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{array} \right\}$$

Proof. From Eqs. (6), (4) and (1), for $l = \ln L$, we obtain the following derivatives:

$$\begin{aligned}
\frac{\partial l}{\partial \boldsymbol{\theta}_g}|_{\theta_g = \theta_g^{(k)}} &= \sum_{t=1}^N \sum_{j=1}^K \{ \frac{g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)}) P(\mathbf{y}^{(t)} | \mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\sum_{i=1}^K g_i(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)}) P(\mathbf{y}^{(t)} | \mathbf{x}^{(t)}, \boldsymbol{\theta}_i^{(k)})} \} \frac{\partial g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g)/\partial \boldsymbol{\theta}_g|_{\theta_g = \theta_g^{(k)}}}{g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})} \\
&= \sum_{t=1}^N \sum_{j=1}^K [h_j^{(k)}(t) - g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})] \frac{\partial s_j}{\partial \boldsymbol{\theta}_g}|_{\theta_g = \theta_g^{(k)}},
\end{aligned} \tag{33}$$

$$\frac{\partial l}{\partial \boldsymbol{\theta}_j}|_{\theta_j = \theta_j^{(k)}} = \sum_{t=1}^N \{ \frac{g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)}) P(\mathbf{y}^{(t)} | \mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\sum_{i=1}^K g_i(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)}) P(\mathbf{y}^{(t)} | \mathbf{x}^{(t)}, \boldsymbol{\theta}_i^{(k)})} \} \frac{\partial P(\mathbf{y}^{(t)} | \mathbf{x}^{(t)}, \boldsymbol{\theta}_j)/\partial \boldsymbol{\theta}_j|_{\theta_j = \theta_j^{(k)}}}{P(\mathbf{y}^{(t)} | \mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}$$

$$
\begin{aligned}
&= \sum_{t=1}^{N} h_j^{(k)}(t) \frac{\partial \mathbf{f}_j^T(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}_j}|_{\theta_j=\theta_j^{(k)}} (\Sigma_j^{(k)})^{-1}[\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})], \\
j &= 1, \cdots, K,
\end{aligned} \tag{34}
$$

$$
\begin{aligned}
\frac{\partial l}{\partial \Sigma_j}|_{\Sigma_j=\Sigma_j^{(k)}} &= \sum_{t=1}^{N} \{ \frac{g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)}) P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\sum_{i=1}^{K} g_i(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)}) P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_i^{(k)})} \} \frac{\partial P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})/\partial \Sigma_j|_{\Sigma_j=\Sigma_j^{(k)}}}{P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})} \\
&= -\frac{1}{2} \sum_{t=1}^{N} h_j^{(k)}(t) (\Sigma_j^{(k)})^{-1} \{ \Sigma_j^{(k)} - [\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})][\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})]^T \} (\Sigma_j^{(k)})^{-1}, \\
j &= 1, \cdots, K,
\end{aligned} \tag{35}
$$

We now prove points (i), (ii) and (iii).

(i) Comparing Eq. (33) with Eq. (28) it follows that $P_g^{(k)} = \gamma_g (R_g^{(k)})^{-1}$. To show that $P_g^{(k)}$ is positive definite, we show that $R_g^{(k)}$ is positive definite.[2] For an arbitrary vector $\mathbf{u}$, from Eq. (28) we have

$$
\begin{aligned}
\mathbf{u}^T R_g^{(k)} \mathbf{u} &= \sum_{t=1}^{N} \sum_{j=1}^{K} g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})[1 - g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})] \mathbf{u}^T \frac{\partial s_j}{\partial \boldsymbol{\theta}_g} \frac{\partial s_j}{\partial \boldsymbol{\theta}_g^T} \mathbf{u} \\
&= \sum_{t=1}^{N} \sum_{j=1}^{K} g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})[1 - g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})] \mathbf{v}^T \mathbf{v} \geq 0,
\end{aligned}
$$

since $g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})[1 - g_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_g^{(k)})] > 0$. Equality holds in the above equation only when $\mathbf{v} = [\partial s_j / \partial \boldsymbol{\theta}_g^T]\mathbf{u} = 0$ for any $\mathbf{u}$, which is impossible. Thus we have established that $R_g^{(k)}$ (and thus also $(R_g^{(k)})^{-1}$) is positive definite.

(ii) Let $C_j^{(k)}$, $R_j^{(k)}$ be given by Eq. (22). From Eq. (21) and Eq. (34), we obtain

$$
[C_j^{(k)} - (R_j^{(k)}) \boldsymbol{\theta}_j^{(k)}] = \frac{\partial l}{\partial \boldsymbol{\theta}_j}|_{\theta_j=\theta_j^{(k)}}, \ j = 1, \cdots, K.
$$

Furthermore, it follows from Eq. (22) that

$$
\begin{aligned}
\boldsymbol{\theta}_j^{(k+1)} &= \boldsymbol{\theta}_j^{(k)} + (R_j^{(k)})^{-1} C_j^{(k)} - \boldsymbol{\theta}_j^{(k)} \\
&= \boldsymbol{\theta}_j^{(k)} + (R_j^{(k)})^{-1} [C_j^{(k)} - (R_j^{(k)}) \boldsymbol{\theta}_j^{(k)}].
\end{aligned} \tag{36}
$$

That is, we have

$$
\boldsymbol{\theta}_j^{(k+1)} = \boldsymbol{\theta}_j^{(k)} + (R_j^{(k)})^{-1} \frac{\partial l}{\partial \boldsymbol{\theta}_j}|_{\theta_j=\theta_j^{(k)}}, \ j = 1, \cdots, K.
$$

and $P_j^{(k)} = (R_j^{(k)})^{-1}$.

We now prove that $R_j^{(k)}$ is positive definite. For an arbitrary vector $\mathbf{u}$, from Eq. (22) we have

$$
\mathbf{u}^T R_j^{(k)} \mathbf{u} = \sum_{t=1}^{N} h_j^{(k)}(t) \mathbf{u}^T X_t (\Sigma_j^{(k)})^{-1} X_t^T \mathbf{u} = \sum_{t=1}^{N} h_j^{(k)}(t) \mathbf{v}^T (\Sigma_j^{(k)})^{-1} \mathbf{v} \geq 0.
$$

[2] A matrix A is positive definite if and only if A^{-1} is positive definite.

From the note immediately following Eq. (16)), we know that $\Sigma_j^{(k)}$ given by Eq. (20) is positive definite and invertible for each k with probability one. Thus, with probability one, the equality of the above equation holds only when $\mathbf{v} = X_t^T \mathbf{u} = 0$ for any $\mathbf{u}$, which is impossible. Thus we have established that $R_j^{(k)}$ (and thus also $(R_j^{(k)})^{-1}$) is positive definite with probability one.

(iii) We consider Eq. (20) for updating $\Sigma_j^{(k)}$. This equation can be expanded as follows

$$\begin{aligned}\Sigma_j^{(k+1)} &= \Sigma_j^{(k)} + \frac{1}{\sum_{t=1}^N h_j^{(k)}(t)} \sum_{t=1}^N h_j^{(k)}(t)[\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)][\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)]^T - \Sigma_j^{(k)} \\ &= \Sigma_j^{(k)} + \frac{2\Sigma_j^{(k)}}{\sum_{t=1}^N h_j^{(k)}(t)} V_{\Sigma_j} \Sigma_j^{(k)}, \end{aligned} \tag{37}$$

where

$$V_{\Sigma_j} = -\frac{1}{2}\sum_{t=1}^N h_j^{(k)}(t)(\Sigma_j^{(k)})^{-1}\{\Sigma_j^{(k)} - [\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})][\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})]^T\}(\Sigma_j^{(k)})^{-1}.$$

It follows from Eq. (35) that

$$V_{\Sigma_j} = \frac{\partial l}{\partial \Sigma_j}|_{\Sigma_j = \Sigma_j^{(k)}}.$$

That is, we have

$$\Sigma_j^{(k+1)} = \frac{2\Sigma_j^{(k)}}{\sum_{t=1}^N h_j^{(k)}(t)} \frac{\partial l}{\partial \Sigma_j}|_{\Sigma_j = \Sigma_j^{(k)}} \Sigma_j^{(k)}.$$

Utilizing the identity $vec[ABC] = (C^T \otimes A)vec[B]$, we obtain

$$vec[\Sigma_j^{(k+1)}] = \frac{2}{\sum_{t=1}^N h_j^{(k)}(t)} (\Sigma_j^{(k)} \otimes \Sigma_j^{(k)}) \frac{\partial l}{\partial \Sigma_j}|_{\Sigma_j = \Sigma_j^{(k)}}.$$

Thus $P_{\Sigma_j}^{(k)} = \frac{2}{\sum_{t=1}^N h_j^{(k)}(t)}(\Sigma_j^{(k)} \otimes \Sigma_j^{(k)})$. Moreover, for an arbitrary matrix U, we have

$$vec[U]^T(\Sigma_j^{(k)} \otimes \Sigma_j^{(k)})vec[U] = \mathrm{tr}(\Sigma_j^{(k)} U \Sigma_j^{(k)} U^T)$$

$$= \mathrm{tr}((\Sigma_j^{(k)} U)^T (\Sigma_j^{(k)} U)) = vec[\Sigma_j^{(k)} U]^T vec[\Sigma_j^{(k)} U] \geq 0$$

where the equality holds only when $\Sigma_j^{(k)} U = 0$, which is impossible with probability one since U is arbitrary, and $\Sigma_j^{(k)}$ is, as indicated above, positive definite with probability one. Thus we have established that $P_{\Sigma_j}^{(k)}$ is positive definite with probability one. □

Theorem 1 can be used to establish a relationship between the step taken by the EM algorithm and the direction of steepest ascent. Recall that for a positive matrix B, we have $\frac{\partial l}{\partial \boldsymbol{\theta}}^T B \frac{\partial l}{\partial \boldsymbol{\theta}} > 0$. This implies the following corollary.

Corollary 1 *Assume that the training set $\{\mathbf{y}^{(t)}, \mathbf{x}^{(t)}, t = 1, \cdots, N\}$ comes from the mixture model of Eqs. (4) and (1) and that N is sufficiently large. With probability one, the search direction of the EM algorithm has a positive projection on the gradient ascent searching direction of $l = \ln L$.*

That is, the EM algorithm can be viewed as a modified gradient ascent algorithm for maximizing $l = \ln L$. From Theorem 1, B changes with the iteration step k, thus, the EM algorithm can also be regarded as a variable metric gradient ascent algorithm. This algorithm searches in an uphill direction, so if the learning rate is appropriate, the searching process will converge to a local maximum or a saddle point of the likelihood $l = \ln L$.

Similar results have been obtained for unsupervised mixture models by Xu and Jordan (1993) and for Hidden Markov Models by Baum and Sell (1968). See Xu and Jordan (1993) for further discussion of the relationships between these theorems.

We now utilize a result from Xu and Jordan (1993) to establish the convergence of the parameters $\boldsymbol{\Theta}^{(k)}$. We also provide convergence rates for both $l(\boldsymbol{\Theta}^{(k)})$ and $\boldsymbol{\Theta}^{(k)}$.

Theorem 2 *Assume that the training set $\mathcal{Y}$ is generated by the mixture model of Eqs. (4)(1) and that N is sufficiently large. Assume further that $\Sigma_1, \cdots, \Sigma_K$ are diagonal and let $\mathbf{v}_{\Sigma_j}$ be a vector consisting of the diagonal elements of Σ_j.*

Let us denote

$$\boldsymbol{\Theta} = [\boldsymbol{\theta}_g^T, \boldsymbol{\theta}_1^T, \cdots, \boldsymbol{\theta}_K^T, \mathbf{v}_{\Sigma_1}, \cdots, \mathbf{v}_{\Sigma_K}]^T$$

$$P = \mathrm{diag}[P_g^{(k)}, P_1, \cdots, P_K, P_{\Sigma_1}, \cdots, P_{\Sigma_K}], \text{ and } \quad H(\boldsymbol{\Theta}) = \frac{\partial^2 l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta} \partial \boldsymbol{\Theta}^T}.$$

Furthermore, assume that on a given domain $D_{\boldsymbol{\Theta}}$

(i) $\frac{\partial^2 g_j(\boldsymbol{\theta}_q)}{\partial \boldsymbol{\theta}_q \partial \boldsymbol{\theta}_q^T}$, $\frac{\partial^2 f_{ji}(\boldsymbol{\theta}_j)}{\partial \boldsymbol{\theta}_j \partial \boldsymbol{\theta}_j^T}$, $j = 1, \cdots, K,\ i = 1, \cdots, m$ exist and are continuous;

(ii) the Hessian matrix $H(\boldsymbol{\Theta})$ is negative definite;

(iii) $\boldsymbol{\Theta}^*$ is a local maximum of $l(\boldsymbol{\Theta})$, and $\boldsymbol{\Theta}^* \in D_{\boldsymbol{\Theta}}$.

Then with probability one,

(1) Letting $-M, -m$ (here $M > m > 0$) be the minimum and maximum eigenvalues of the negative definite matrix $(P^{\frac{1}{2}})^T H(\boldsymbol{\Theta})(P^{\frac{1}{2}})$ (or equivalently the minimum and maximum eigenvalues of $PH(\boldsymbol{\Theta})$, since we have $PH\mathbf{e} = \lambda \mathbf{e}$ from $(P^{\frac{1}{2}})^T H P^{\frac{1}{2}} \mathbf{e} = \lambda \mathbf{e}$), we have

$$l(\boldsymbol{\Theta}^*) - l(\boldsymbol{\Theta}^{(k)}) \leq r^k [l(\boldsymbol{\Theta}^*) - l(\boldsymbol{\Theta}_0)], \tag{38}$$

$$\|P^{-\frac{1}{2}}(\boldsymbol{\Theta}^{(k)} - \boldsymbol{\Theta}^*)\| \leq |r|^{k/2} \sqrt{\frac{2}{m}[l(\boldsymbol{\Theta}^*) - l(\boldsymbol{\Theta}_0)]}, \tag{39}$$

where $r = 1 - (1 - \frac{M}{2})\frac{m^2}{M} < 1$. We also have $0 < |r| < 1$ when $M < 2$.

(2) For any initial point $\boldsymbol{\Theta}_0 \in D_{\boldsymbol{\Theta}}$, $\lim_{k \to \infty} \boldsymbol{\Theta}^{(k)} = \boldsymbol{\Theta}^*$ when $M < 2$.

Proof. As indicated earlier, when the training set $\{\mathbf{y}^{(t)}, \mathbf{x}^{(t)}, t = 1, \cdots, N\}$ is generated from the mixture model of Eqs. (4)(1) and N is sufficiently large, $\Sigma_j^{(k)}$ remains positive definite during the learning process. Thus, under the condition (i), it follows from Eqs. (6), (4) and (1) that $H(\boldsymbol{\Theta})$ exists and remains continuous on $D_{\boldsymbol{\Theta}}$. Expanding the log likelihood in a Taylor expansion, we have

$$l(\boldsymbol{\Theta}) - l(\boldsymbol{\Theta}^*) = (\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)^T \frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}|_{\boldsymbol{\Theta} = \boldsymbol{\Theta}^*} + \frac{1}{2}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)^T H(\boldsymbol{\Theta}^* + \xi(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*))(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)$$

with $0 < \xi < 1$. Since $\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}|_{\boldsymbol{\Theta} = \boldsymbol{\Theta}^*} = 0$, we have

$$l(\boldsymbol{\Theta}) - l(\boldsymbol{\Theta}^*) = \frac{1}{2}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)^T H(\boldsymbol{\Theta}^* + \xi(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*))(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*). \tag{40}$$

From Theorem 1 we know that P is positive definite. Furthermore, from condition (ii), $H(\boldsymbol{\Theta})$ is negative definite on $D_{\boldsymbol{\Theta}}$. This implies that $P^{\frac{1}{2}}$ exists and $(P^{\frac{1}{2}})^T H(\boldsymbol{\Theta})(P^{\frac{1}{2}})$ is negative definite on $D_{\boldsymbol{\Theta}}$. Utilizing the Rayleigh quotient we obtain that for any $\mathbf{u}$,

$$-M\|\mathbf{u}\|^2 \leq \mathbf{u}^T(P^{\frac{1}{2}})^T H(\boldsymbol{\Theta})(P^{\frac{1}{2}})\mathbf{u} \leq -m\|\mathbf{u}\|^2. \tag{41}$$

Substituting Eq. (41) into Eq. (40), we obtain

$$l(\boldsymbol{\Theta}) - l(\boldsymbol{\Theta}^*) = \frac{1}{2}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)^T(P^{-\frac{1}{2}})^T(P^{\frac{1}{2}})^T H(\boldsymbol{\Theta}^* + \xi(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*))P^{\frac{1}{2}}P^{-\frac{1}{2}}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*) \tag{42}$$

$$l(\boldsymbol{\Theta}) - l(\boldsymbol{\Theta}^*) \geq -\frac{M}{2}\|P^{-\frac{1}{2}}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)\|^2 \tag{43}$$

Moreover, we have

$$\begin{aligned} -m\|P^{-\frac{1}{2}}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)\|^2 &\geq |(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)^T[\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} - \frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}^*}]| \\ &\geq -\|P^{\frac{1}{2}}\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}\|\|P^{-\frac{1}{2}}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)\| \end{aligned}$$

Thus

$$\|P^{-\frac{1}{2}}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^*)\| \leq \frac{1}{m}\|P^{\frac{1}{2}}\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}\|.$$

Together with Eq. (43), we obtain

$$-\|P^{\frac{1}{2}}\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}\|^2 \leq \frac{2m^2}{M}[l(\boldsymbol{\Theta}) - l(\boldsymbol{\Theta}^*)]. \tag{44}$$

On the other hand, we also have

$$l(\boldsymbol{\Theta}) - l(\boldsymbol{\Theta}^{(k)}) = (\boldsymbol{\Theta} - \boldsymbol{\Theta}^{(k)})^T\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}^{(k)}} + \frac{1}{2}(\boldsymbol{\Theta} - \boldsymbol{\Theta}^{(k)})^T H(\boldsymbol{\Theta}^{(k)} + \xi'(\boldsymbol{\Theta} - \boldsymbol{\Theta}^{(k)}))(\boldsymbol{\Theta} - \boldsymbol{\Theta}^{(k)})$$

with $0 < \xi' < 1$. By Theorem 1, we know that for the EM algorithm, $\boldsymbol{\Theta}^{(k+1)} - \boldsymbol{\Theta}^{(k)} = P\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}^{(k)}}$. Utilizing this result in the above equation, we obtain

$$\begin{aligned} l(\boldsymbol{\Theta}^{(k+1)}) - l(\boldsymbol{\Theta}^{(k)}) &= \|P^{\frac{1}{2}}\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}\|^2_{\boldsymbol{\Theta}=\boldsymbol{\Theta}^{(k)}} \\ &+ \frac{1}{2}(P^{\frac{1}{2}}\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}^{(k)}})^T(P^{\frac{1}{2}})^T H(\boldsymbol{\Theta}^{(k)} + \xi'(\boldsymbol{\Theta} - \boldsymbol{\Theta}^{(k)}))P^{\frac{1}{2}}(P^{\frac{1}{2}}\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}|_{\boldsymbol{\Theta}=\boldsymbol{\Theta}^{(k)}}) \\ &\geq (1 - \frac{M}{2})\|P^{\frac{1}{2}}\frac{\partial l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}}\|^2_{\boldsymbol{\Theta}=\boldsymbol{\Theta}^{(k)}}. \end{aligned} \tag{45}$$

Combining Eq. (45) and Eq. (44), we obtain

$$l(\boldsymbol{\Theta}^{(k+1)}) - l(\boldsymbol{\Theta}^{(k)}) \geq -(1 - \frac{M}{2})\frac{2m^2}{M}[l(\boldsymbol{\Theta}^{(k)}) - l(\boldsymbol{\Theta}^*)]$$

and furthermore

$$\begin{aligned} l(\boldsymbol{\Theta}^{(k+1)}) - l(\boldsymbol{\Theta}^*) &\geq [1 - (1 - \frac{M}{2})\frac{2m^2}{M}][l(\boldsymbol{\Theta}^{(k)}) - l(\boldsymbol{\Theta}^*)] \\ &\geq [1 - (1 - \frac{M}{2})\frac{2m^2}{M}]^k[l(\boldsymbol{\Theta}_0) - l(\boldsymbol{\Theta}^*)]. \end{aligned} \tag{46}$$

Let $r = [1 - (1 - \frac{M}{2})\frac{2m^2}{M}]$. Multiplying both sides of the above equation by negative one, we obtain Eq. (38). In addition, it is easy to verify that $0 < |r| < 1$ when $M < 2$ (recall that $M > m$). Furthermore, it follows from Eq. (41) and Eq. (42) that we have

$$l(\boldsymbol{\Theta}^{(k)}) - l(\boldsymbol{\Theta}^*) \leq -\frac{m}{2}\|P^{-\frac{1}{2}}(\boldsymbol{\Theta}^{(k)} - \boldsymbol{\Theta}^*)\|^2$$

which, by Eq. (46), becomes

$$-\frac{m}{2}\|P^{-\frac{1}{2}}(\boldsymbol{\Theta}^{(k)} - \boldsymbol{\Theta}^*)\|^2 \geq r^k[l(\boldsymbol{\Theta}_0) - l(\boldsymbol{\Theta}^*)]$$

$$\|P^{-\frac{1}{2}}(\boldsymbol{\Theta}^{(k)} - \boldsymbol{\Theta}^*)\| \leq |r|^{k/2}\sqrt{\frac{2}{m}[l(\boldsymbol{\Theta}^*) - l(\boldsymbol{\Theta}_0)]}$$

which is just Eq. (39). In addition, when $M < 2$, $|r| < 1$, we have $\lim_{k\to\infty} \boldsymbol{\Theta}^{(k)} = \boldsymbol{\Theta}^*$ since P is positive definite. □

We see from this theorem that the EM algorithm converges linearly. Moreover, the speed of convergence depends on the difference between M and m: the smaller the difference, the faster the convergence.

Theoretical analysis of an EM algorithm for the hierarchical mixture of experts architecture

An EM algorithm for training the hierarchical architecture

The ME architecture can be viewed as an architecture for splitting the input space into regions in which different local functions are fit. The hierarchical mixture of experts (HME) architecture generalizes this idea to a nested model in which regions in the input space are split recursively into subregions (Jordan & Jacobs, 1992). The resulting tree-structured architecture can be viewed as a multi-resolution function approximator in which smoothed piecewise functions are fit at a variety of levels of resolution.

As shown in Figure 2, the HME architecture is a tree. In this tree, each terminal node is an expert network, and each nonterminal node is a root of a subtree which itself corresponds to an HME architecture. At every nonterminal node in the tree there is a gating network which is responsible for the topmost split of the HME architecture rooted at that node. All of the expert networks and the gating networks in the architecture have the same input vector $\mathbf{x} \in R^n$. In the remainder of this section, as in Jordan and Jacobs (in press), we consider the case in which the expert networks and the gating networks are generalized linear models. Furthermore, for simplicity, we consider only the case in which the probability model for the experts is gaussian.

Let us denote a node at depth r by $v_{i_0 i_1 \cdots i_r}$. This node is the i_r-th daughter of the node $v_{i_0 i_1 \cdots i_{r-1}}$. The root node of the tree is v_{i_0}. The number of branches emitted from $v_{i_0 i_1 \cdots i_r}$ is denoted by $K_{i_0 i_1 \cdots i_r}$. For simplicity, we can omit i_0 and write $v_{i_1 \cdots i_r}$ and $K_{i_1 \cdots i_r}$. In addition, the output of the subtree rooted at $v_{i_1 \cdots i_r}$ is denoted

$$\mathbf{y}_{i_1 \cdots i_r} = \sum_{i_{r+1}=1}^{K_{i_1 \cdots i_r}} g_{i_1 \cdots i_r i_{r+1}}(\mathbf{x}, \boldsymbol{\theta}^g_{i_1 \cdots i_r})\mathbf{y}_{i_1 \cdots i_r i_{r+1}}$$

where $g_{i_1 \cdots i_r i_{r+1}}$ is the gating coefficient generated by the gating network attached at $v_{i_1 \cdots i_r}$. This coefficient satisfies

$$\sum_{i_{r+1}=1}^{K_{i_1 \cdots i_r}} g_{i_1 \cdots i_r i_{r+1}}(\mathbf{x}, \boldsymbol{\theta}^g_{i_1 \cdots i_r}) = 1,$$

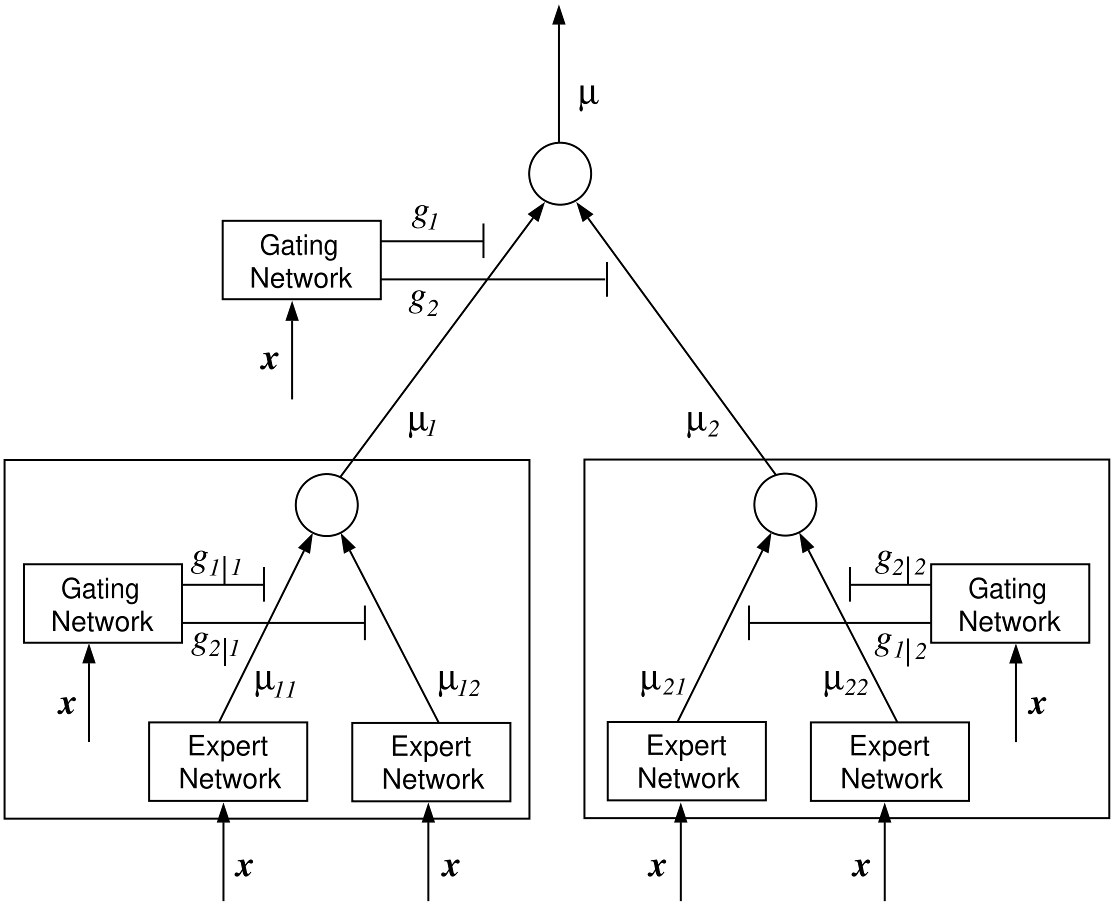

Figure 2: A two-level hierarchical mixture of experts. To form a deeper tree, each expert is expanded recursively into a gating network and a set of sub-experts.

for any $\mathbf{x}$, where $\boldsymbol{\theta}^g_{i_1\cdots i_r}$ is the parameter vector of the gating network.

Given a training set $\mathcal{Y} = \{(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}), t = 1, \cdots, N\}$, we want to maximize the likelihood function (cf. Eq. 6), that is,

$$L = P(\{\mathbf{y}^{(t)}\}_1^N | \{\mathbf{x}^{(t)}\}_1^N) = \prod_{t=1}^{N} P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}).$$

Expanding the probability model, we have

$$P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}) = \sum_{i_1=1}^{K_{i_0}} g_{i_1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^g_{i_0}) P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1})$$

where

$$P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1}) = \begin{cases} \sum_{i_2=1}^{K_{i_1}} g_{i_1 i_2}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^g_{i_1}) P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1 i_2}), \\ \qquad \text{if } v_{i_1} \text{ is a nonterminal node,} \\ (2\pi \det \Sigma_{i_1})^{-\frac{1}{2}} e^{\{-\frac{1}{2}[\mathbf{y}^{(t)} - \mathbf{f}_{i_1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1})]^T \Sigma_{i_1}^{-1} [\mathbf{y}^{(t)} - \mathbf{f}_{i_1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1})]\}}, \\ \qquad \text{if } v_{i_1} \text{ is a terminal node} \end{cases} \tag{47}$$

and recursively

$$P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1\cdots i_r}) = \begin{cases} \sum_{i_{r+1}=1}^{K_{i_1\cdots i_r}} g_{i_1\cdots i_r}(\mathbf{x}, \boldsymbol{\theta}^g_{i_1\cdots i_r}) P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1\cdots i_r i_{r+1}}), \\ \qquad \text{if } v_{i_1\cdots i_r} \text{ is a nonterminal node,} \\ (2\pi \det \Sigma_{i_1\cdots i_r})^{-\frac{1}{2}} e^{\{-\frac{1}{2}[\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1\cdots i_r})]^T \Sigma^{-1}_{i_1\cdots i_r} [\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1\cdots i_r})]\}}, \\ \qquad \text{if } v_{i_1\cdots i_r} \text{ is a terminal node} \end{cases} \tag{48}$$

where $\mathbf{f}_{i_1\cdots i_r}(.,.)$ is parameterized function implemented by the expert network at terminal node $v_{i_1\cdots i_r}$, and $\boldsymbol{\theta}_{i_1\cdots i_r}$, $\Sigma_{i_1\cdots i_r}$ are the parameters of this expert network; $P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1\cdots i_r})$ is the probability that $\mathbf{y}^{(t)}$ is generated from the probability model rooted at $v_{i_1\cdots i_r}$ when $\mathbf{x}^{(t)}$ is the input.

To derive an EM algorithm for the HME architecture, we attach a set of indicator random variables to each nonterminal node $v_{i_1\cdots i_r}$

$$I^{(t)}_{i_1\cdots i_r} = \begin{cases} 1, & \text{if } \mathbf{y}^{(t)} \text{ is generated by the subtree rooted at } v_{i_1\cdots i_r}, \\ 0, & \text{otherwise.} \end{cases}$$

The missing data $\mathcal{Y}_{mis}$ consists of all of the indicator variables attached to the nonterminal nodes throughout the tree. In addition, we denote by $\mathcal{Y}^{mis}_{i_1\cdots i_r}$ the set consisting of the indicator variables attached to the nonterminal nodes in the subtree rooted at $v_{i_1\cdots i_r}$.

We define the distribution of the complete data $\mathcal{Z} = \{\mathcal{Y}, \mathcal{Y}_{mis}\}$ as follows

$$P(\mathcal{Z}|\boldsymbol{\Theta}) = \prod_{t=1}^{N} \prod_{i_1=1}^{K_{i_0}} [g_{i_1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^g_{i_0}) P(\mathcal{Z}_{i_1}|\mathbf{x}^{(t)}, v_{i_1})]^{I^{(t)}_{i_1}} \tag{49}$$

where

$$P(\mathcal{Z}_{i_1}|\mathbf{x}^{(t)}, v_{i_1}) = \begin{cases} \prod_{i_2=1}^{K_{i_1}} [g_{i_1 i_2}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^g_{i_1}) P(\mathcal{Z}_{i_1 i_2}|\mathbf{x}^{(t)}, v_{i_1 i_2})^{I^{(t)}_{i_1 i_2}}, \\ \qquad \text{if } v_{i_1} \text{ is a nonterminal node,} \\ (2\pi det\Sigma_{i_1})^{-\frac{1}{2}} e^{\{-\frac{1}{2}[\mathbf{y}^{(t)} - \mathbf{f}_{i_1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1})]^T \Sigma^{-1}_{i_1} [\mathbf{y}^{(t)} - \mathbf{f}_{i_1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1})]\}}, \\ \qquad \text{if } v_{i_1} \text{ is a terminal node} \end{cases} \tag{50}$$

and recursively

$$P(\mathcal{Z}_{i_1\cdots i_r}|\mathbf{x}^{(t)}, v_{i_1\cdots i_r}) = \begin{cases} \prod_{i_{r+1}=1}^{K_{i_1\cdots i_r}} [g_{i_1\cdots i_r i_{r+1}}(\mathbf{x}, \boldsymbol{\theta}^g_{i_1\cdots i_r}) P(\mathcal{Z}_{i_1\cdots i_r i_{r+1}}|\mathbf{x}^{(t)}, v_{i_1\cdots i_r i_{r+1}})]^{I^{(t)}_{i_1\cdots i_r i_{r+1}}}, \\ \qquad \text{if } v_{i_1\cdots i_r} \text{ is a nonterminal node,} \\ (2\pi det\Sigma_{i_1\cdots i_r})^{-\frac{1}{2}} e^{\{-\frac{1}{2}[\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1\cdots i_r})]^T \Sigma^{-1}_{i_1\cdots i_r} [\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1\cdots i_r})]\}}, \\ \qquad \text{if } v_{i_1\cdots i_r} \text{ is a terminal node} \end{cases} \tag{51}$$

It is not difficult to verify that this distribution satisfies Eq. (10) as required.

We now compute the Q function as required by the E step of the EM algorithm. From Eq. (11), we obtain

$$Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) = \sum_{t=1}^{N} \sum_{i_1=1}^{K_{i_0}} h^{(k)}_{i_1}(t) \{\ln[g_{i_1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^g_{i_0})] + \ln F_{i_1}\}$$

where

$$h^{(k)}_{i_1}(t) = E[I^{(t)}_{i_1}|\mathcal{Y}, \boldsymbol{\Theta}^{(k)}] = \frac{g_{i_1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^{g(k)}_{i_0}) P^{(k)}(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1})}{\sum_{i_1=1}^{K_{i_0}} g_{i_1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^{g(k)}_{i_0}) P^{(k)}(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1})}, \tag{52}$$

$$\ln F_{i_1} = \begin{cases} \sum_{i_2=1}^{K_{i_1}} h_{i_1 i_2}^{(k)}(t)\{\ln[g_{i_1 i_2}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1}^g)] + \ln F_{i_1 i_2}\}, & \text{if } v_{i_1} \text{ is a nonterminal node,} \\ \ln P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1}), & \text{if } v_{i_1} \text{ is a terminal node} \end{cases}$$

$$h_{i_1 i_2}^{(k)}(t) = E[I_{i_1 i_2}^{(t)}|\mathcal{Y}, \boldsymbol{\Theta}^{(k)}, I_{i_1}^{(t)} = 1] = \frac{g_{i_1 i_2}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1}^{g(k)}) P^{(k)}(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1 i_2})}{\sum_{i_2=1}^{K_{i_1}} g_{i_1 i_2}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1}^{g(k)}) P^{(k)}(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1 i_2})}, \tag{53}$$

and recursively

$$\ln F_{i_1 \cdots i_r} = \begin{cases} \sum_{i_{r+1}=1}^{K_{i_1 \cdots i_r}} h_{i_1 \cdots i_r i_{r+1}}^{(k)}(t)\{\ln[g_{i_1 \cdots i_r i_{r+1}}(\mathbf{x}, \boldsymbol{\theta}_{i_1 \cdots i_r}^g)] + \ln F_{i_1 \cdots i_r i_{r+1}}\}, \\ \qquad \text{if } v_{i_1 \cdots i_r} \text{ is a nonterminal node,} \\ \ln P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1 \cdots i_r}), \\ \qquad \text{if } v_{i_1 \cdots i_r} \text{ is a terminal node} \end{cases}$$

$$\begin{aligned} h_{i_1 \cdots i_r i_{r+1}}^{(k)}(t) &= E[I_{i_1 \cdots i_r i_{r+1}}^{(t)}|\mathcal{Y}, \boldsymbol{\Theta}^{(k)}, I_{i_1}^{(t)} = 1, I_{i_1 i_2}^{(t)} = 1, \cdots, I_{i_1 \cdots i_r}^{(t)} = 1] \\ &= \frac{g_{i_1 \cdots i_r i_{r+1}}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1 \cdots i_r}^{g(k)}) P^{(k)}(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1 \cdots i_r i_{r+1}})}{\sum_{i_{r+1}=1}^{K_{i_1 \cdots i_r}} g_{i_1 \cdots i_r i_{r+1}}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1 \cdots i_r}^{g(k)}) P^{(k)}(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1 \cdots i_r i_{r+1}})}, \end{aligned} \tag{54}$$

where $\boldsymbol{\theta}_{i_1 \cdots i_r}^{g(k)}$ is the estimate of $\boldsymbol{\theta}_{i_1 \cdots i_r}^g$ at iteration k. $P(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1 \cdots i_r})$ is given by Eqs. (47) and (48) and $P^{(k)}(\mathbf{y}^{(t)}|\mathbf{x}^{(t)}, v_{i_1 \cdots i_r})$ means that the probability is determined with all the parameters in the subtree rooted at $v_{i_1 \cdots i_r}$ being fixed at the estimates obtained in iteration k.

Proceeding now to the M step of the EM algorithm, we obtain parameter updates by optimizing the Q function. If the node $v_{i_1 \cdots i_r}$ is a terminal node, by setting the partial derivative of Q with respect to $\Sigma_{i_1 \cdots i_r}$ equal to zero, we obtain an update for the covariance matrices

$$\Sigma_{i_1 \cdots i_r}^{(k+1)} = \frac{\sum_{t=1}^N h_{i_1}^{(k)}(t) h_{i_1 i_2}^{(k)}(t) \cdots h_{i_1 \cdots i_r}^{(k)}(t) [\mathbf{y}^{(t)} - \mathbf{f}_{i_1 \cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1 \cdots i_r})][\mathbf{y}^{(t)} - \mathbf{f}_{i_1 \cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1 \cdots i_r})]^T}{\sum_{t=1}^N h_{i_1}^{(k)}(t) h_{i_1 i_2}^{(k)}(t) \cdots h_{i_1 \cdots i_r}^{(k)}(t)}. \tag{55}$$

To obtain an update for the parameters of the expert networks we differentiate Q with respect to $\boldsymbol{\theta}_{i_1 \cdots i_r}$ and find that we must solve the following equation

$$\sum_{t=1}^N h_{i_1}^{(k)}(t) h_{i_1 i_2}^{(k)}(t) \cdots h_{i_1 \cdots i_r}^{(k)}(t) \frac{\partial \mathbf{f}_{i_1 \cdots i_r}^T(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1 \cdots i_r})}{\partial \boldsymbol{\theta}_{i_1 \cdots i_r}} \Sigma_{i_1 \cdots i_r}^{-1} [\mathbf{y}^{(t)} - \mathbf{f}_{i_1 \cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1 \cdots i_r})] = 0. \tag{56}$$

In the case of linear expert networks, this equation is a weighted least squares equation, which can be solved as follows

$$\boldsymbol{\theta}_{i_1 \cdots i_r}^{(k+1)} = (R_{i_1 \cdots i_r}^{(k)})^{-1} \mathbf{c}_{i_1 \cdots i_r}^{(k)},$$

where

$$\mathbf{c}_{i_1 \cdots i_r}^{(k)} = \sum_{t=1}^N h_{i_1}^{(k)}(t) h_{i_1 i_2}^{(k)}(t) \cdots h_{i_1 \cdots i_r}^{(k)}(t) X_t (\Sigma_{i_1 \cdots i_r}^{(k)})^{-1} \mathbf{y}^{(t)},$$

and

$$R_{i_1 \cdots i_r}^{(k)} = \sum_{t=1}^N h_{i_1}^{(k)}(t) h_{i_1 i_2}^{(k)}(t) \cdots h_{i_1 \cdots i_r}^{(k)}(t) X_t (\Sigma_{i_1 \cdots i_r}^{(k)})^{-1} X_t^T. \tag{57}$$

Finally, for any nonterminal node $v_{i_1 \cdots i_r}$, setting the partial derivative of Q with respect to $\boldsymbol{\theta}_{i_1 \cdots i_r}^g$ equal to zero, we have that $\boldsymbol{\theta}_{i_1 \cdots i_r}^{g(k+1)}$ is the solution of the following nonlinear system

$$\sum_{t=1}^N h_{i_1}^{(k)}(t) h_{i_1 i_2}^{(k)}(t) \cdots h_{i_1 \cdots i_r}^{(k)}(t) \sum_{i_{r+1}=1}^{K_{i_1 \cdots i_r}} [h_{i_1 \cdots i_{r+1}}^{(k)}(t) - g_{i_1 \cdots i_{r+1}}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1 \cdots i_r}^g)] \frac{\partial s_{i_1 \cdots i_{r+1}}}{\partial \boldsymbol{\theta}_{i_1 \cdots i_r}^g} = 0. \tag{58}$$

As in the case of the one-level ME architecture (cf. Eq. (26) and Eq. (28)), we obtain the following Newton step for updating the gating network parameters

$$\boldsymbol{\theta}^{g(k+1)}_{i_1\cdots i_r} = \boldsymbol{\theta}^{g(k)}_{i_1\cdots i_r} + \gamma_{i_1\cdots i_r}(R^{g(k)}_{i_1\cdots i_r})^{-1}\mathbf{e}^{g(k)}_{i_1\cdots i_r}, \tag{59}$$

where

$$\begin{aligned} R^{g(k)}_{i_1\cdots i_r} &= \sum_{t=1}^{N} h^{(k)}_{i_1}(t)h^{(k)}_{i_1 i_2}(t)\cdots h^{(k)}_{i_1\cdots i_r}(t)\sum_{i_{r+1}=1}^{K_{i_1\cdots i_r}} g_{i_1\cdots i_r}(\mathbf{x}^{(t)},\boldsymbol{\theta}^{g(k)}_{i_1\cdots i_r})\times \\ &\times\ [1-g_{i_1\cdots i_r}(\mathbf{x}^{(t)},\boldsymbol{\theta}^{g(k)}_{i_1\cdots i_r})]\frac{\partial s_{i_1\cdots i_{r+1}}}{\partial(\boldsymbol{\theta}^{g(k)}_{i_1\cdots i_r})}\frac{\partial s_{i_1\cdots i_{r+1}}}{\partial(\boldsymbol{\theta}^{g(k)}_{i_1\cdots i_r})^T} \end{aligned} \tag{60}$$

and

$$\begin{aligned} \mathbf{e}^{g(k)}_{i_1\cdots i_r} &= \sum_{t=1}^{N} h^{(k)}_{i_1}(t)h^{(k)}_{i_1 i_2}(t)\cdots h^{(k)}_{i_1\cdots i_r}(t)\times \\ &\times \sum_{i_{r+1}=1}^{K_{i_1\cdots i_r}}[h^{(k)}_{i_1\cdots i_{r+1}}(t) - g_{i_1\cdots i_{r+1}}(\mathbf{x}^{(t)},\boldsymbol{\theta}^{g(k)}_{i_1\cdots i_r})]\frac{\partial s_{i_1\cdots i_{r+1}}}{\partial(\boldsymbol{\theta}^{g(k)}_{i_1\cdots i_r})}. \end{aligned} \tag{61}$$

As in the case of the one-level architecture (cf. Eq. (28)) the algorithm in Eq. (59) is essentially the same as the IRLS algorithm suggested by Jordan and Jacobs (in press), although we have introduced a learning rate parameter and we restrict the update to a single step.

In summary, the EM algorithm for the HME architecture is given as follows.

Algorithm 3

1. (The E step): Compute the $h^{(k)}_{i_1}(t), h^{(k)}_{i_1 i_2}(t), \cdots, h^{(k)}_{i_1\cdots i_r}(t)$ by Eqs. (52), (53) and (54).
2. (The M step): Compute the $\Sigma^{(k+1)}_{i_1\cdots i_r}$ by Eq. (55), compute the $\boldsymbol{\theta}^{g(k+1)}_{i_1\cdots i_r}$ by Eq. (59), and compute the $\boldsymbol{\theta}^{(k+1)}_{i_1\cdots i_r}$ by Eq. (57).

We can think of the E step as assigning credit (posterior probability) to various branches of the tree for each data point and the M step as solving weighted least squares problems in which the weights are given by the posterior probabilities assigned in the E step. The updates for the gating networks simply cache away the posteriors.

Theoretical convergence results

Much of the analysis developed in Section 2.2 can be extended to cover the EM algorithm for the HME architecture. In this subsection we extend Theorem 1 and Theorem 2 to cover the hierarchical case. The results are given as Theorems 3 and 4, respectively.

We first compute the derivatives of the Q function

$$\frac{\partial Q}{\partial \boldsymbol{\theta}^{g}_{i_1\cdots i_r}} = \sum_{t=1}^{N} \quad h^{(k)}_{i_1}(t) \quad h^{(k)}_{i_1 i_2}(t)\cdots h^{(k)}_{i_1\cdots i_r}(t)\times$$

$$\times \sum_{i_{r+1}=1}^{K_{i_1\cdots i_r}} [h^{(k)}_{i_1\cdots i_{r+1}}(t) - g_{i_1\cdots i_{r+1}}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^g_{i_1\cdots i_r})] \frac{\partial s_{i_1\cdots i_{r+1}}}{\partial \boldsymbol{\theta}^g_{i_1\cdots i_r}}, \quad (62)$$

$$\frac{\partial Q}{\partial \boldsymbol{\theta}_{i_1\cdots i_r}} = \sum_{t=1}^{N} h^{(k)}_{i_1}(t) \quad h^{(k)}_{i_1 i_2}(t) \cdots h^{(k)}_{i_1\cdots i_r}(t) \times$$
$$\times \frac{\partial \mathbf{f}^T_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1\cdots i_r})}{\partial \boldsymbol{\theta}_{i_1\cdots i_r}} \Sigma^{-1}_{i_1\cdots i_r} [\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1\cdots i_r})], \quad (63)$$

$$\frac{\partial Q}{\partial \Sigma_{i_1\cdots i_r}} = -\frac{1}{2} \sum_{t=1}^{N} h^{(k)}_{i_1}(t) h^{(k)}_{i_1 i_2}(t) \cdots h^{(k)}_{i_1\cdots i_r}(t) \Sigma^{-1}_{i_1\cdots i_r} \times$$
$$\times \{\Sigma_{i_1\cdots i_r} - [\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1\cdots i_r})][\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1\cdots i_r})]^T\} \Sigma^{-1}_{i_1\cdots i_r}. \quad (64)$$

and the derivatives of the log likelihood

$$\frac{\partial l}{\partial \boldsymbol{\theta}^g_{i_1\cdots i_r}} |_{\boldsymbol{\theta}^g_{i_1\cdots i_r} = ^{(k)}\boldsymbol{\theta}^g_{i_1\cdots i_r}} = \sum_{t=1}^{N} h^{(k)}_{i_1}(t) h^{(k)}_{i_1 i_2}(t) \cdots h^{(k)}_{i_1\cdots i_r}(t) \times$$
$$\times \sum_{i_{r+1}=1}^{K_{i_1\cdots i_r}} [h^{(k)}_{i_1\cdots i_{r+1}}(t) - g_{i_1\cdots i_{r+1}}(\mathbf{x}^{(t)}, ^{(k)}\boldsymbol{\theta}^g_{i_1\cdots i_r})] \frac{\partial s_{i_1\cdots i_{r+1}}}{\partial \boldsymbol{\theta}^g_{i_1\cdots i_r}} |_{\boldsymbol{\theta}^g_{i_1\cdots i_r} = ^{(k)}\boldsymbol{\theta}^g_{i_1\cdots i_r}}, \quad (65)$$

$$\frac{\partial Q}{\partial \boldsymbol{\theta}_{i_1\cdots i_r}} |_{\boldsymbol{\theta}_{i_1\cdots i_r} = \boldsymbol{\theta}^{(k)}_{i_1\cdots i_r}} = \sum_{t=1}^{N} h^{(k)}_{i_1}(t) h^{(k)}_{i_1 i_2}(t) \cdots h^{(k)}_{i_1\cdots i_r}(t) \times$$
$$\times \frac{\partial \mathbf{f}^T_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_{i_1\cdots i_r})}{\partial \boldsymbol{\theta}_{i_1\cdots i_r}} |_{\boldsymbol{\theta}_{i_1\cdots i_r} = \boldsymbol{\theta}^{(k)}_{i_1\cdots i_r}} \Sigma^{-1}_{i_1\cdots i_r} [\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^{(k)}_{i_1\cdots i_r})], \quad (66)$$

$$\frac{\partial Q}{\partial \Sigma_{i_1\cdots i_r}} |_{\Sigma_{i_1\cdots i_r} = \Sigma^{(k)}_{i_1\cdots i_r}} = -\frac{1}{2} \sum_{t=1}^{N} h^{(k)}_{i_1}(t) h^{(k)}_{i_1 i_2}(t) \cdots h^{(k)}_{i_1\cdots i_r}(t) (\Sigma^{(k)}_{i_1\cdots i_r})^{-1} \times$$
$$\times \{\Sigma^{(k)}_{i_1\cdots i_r} - [\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^{(k)}_{i_1\cdots i_r})][\mathbf{y}^{(t)} - \mathbf{f}_{i_1\cdots i_r}(\mathbf{x}^{(t)}, \boldsymbol{\theta}^{(k)}_{i_1\cdots i_r})]^T\} (\Sigma^{(k)}_{i_1\cdots i_r})^{-1} \quad (67)$$

Based on these two sets of derivatives, we follow the same line of thought as in the proof of Theorem 1 to establish the following theorem for the HME architecture.

Theorem 3 *For the HME architecture of Eqs. (47) and (48), with the parameter updates given by Algorithm 3, we have that for every node* $v_{i_1\cdots i_r}$

$$\begin{aligned}
\boldsymbol{\theta}^{g(k+1)}_{i_1\cdots i_r} - \boldsymbol{\theta}^{g(k)}_{i_1\cdots i_r} &= P^{g(k)}_{i_1\cdots i_r} \frac{\partial l}{\partial \boldsymbol{\theta}^g_{i_1\cdots i_r}} |_{\boldsymbol{\theta}^g_{i_1\cdots i_r} = \boldsymbol{\theta}^{g(k)}_{i_1\cdots i_r}}, \\
\boldsymbol{\theta}^{(k+1)}_{i_1\cdots i_r} - \boldsymbol{\theta}^{(k)}_{i_1\cdots i_r} &= P^{(k)}_{i_1\cdots i_r} \frac{\partial l}{\partial \boldsymbol{\theta}_{i_1\cdots i_r}} |_{\boldsymbol{\theta}_{i_1\cdots i_r} = \boldsymbol{\theta}^{(k)}_{i_1\cdots i_r}}, \\
vec[\Sigma^{(k+1)}_{i_1\cdots i_r}] - vec[\Sigma^{(k)}_{i_1\cdots i_r}] &= P^{(k)}_{\Sigma_{i_1\cdots i_r}} \frac{\partial l}{\partial vec[\Sigma_{i_1\cdots i_r}]} |_{\Sigma_{i_1\cdots i_r} = \Sigma^{(k)}_{i_1\cdots i_r}},
\end{aligned} \quad (68)$$

where $l = \ln L$ *is defined by Eqs. (6), (47) and (48).*

Moreover, assuming that the training set $\mathcal{Y}$ is generated by the HME model of Eqs. (47) and (48), and that the number N is sufficiently large, then $P^{g(k)}_{i_1\cdots i_r}$ is a positive definite matrix, and $P^{(k)}_{i_1\cdots i_r}$, $P^{(k)}_{\Sigma_{i_1\cdots i_r}}$ are positive definite matrices with probability one. Specifically, they take the following values:

(i) $P^{g(k)}_{i_1\cdots i_r} = \gamma^g_{i_1\cdots i_r}(R^{g(k)}_{i_1\cdots i_r})^{-1}$ with $R^{g(k)}_{i_1\cdots i_r}$ given by Eq. (59).

(ii) $P^{(k)}_{i_1\cdots i_r} = (R^{(k)}_{i_1\cdots i_r})^{-1}$ with $R^{(k)}_{i_1\cdots i_r}$ given by Eq. (57).

(iii) For $P^{(k)}_{\Sigma_{i_1\cdots i_r}}$, we have

$$P^{(k)}_{\Sigma_{i_1\cdots i_r}} = \frac{2}{\sum_{t=1}^N h^{(k)}_{i_1}(t)h^{(k)}_{i_1 i_2}(t)\cdots h^{(k)}_{i_1\cdots i_r}(t)}\Sigma^{(k)}_{i_1\cdots i_r} \otimes \Sigma^{(k)}_{i_1\cdots i_r} \tag{69}$$

where "$\otimes$" denotes the Kronecker product as defined in Theorem 3.

From Theorem 5, we can again reach Corollary 1. Again, we see that the EM algorithm for training the HME architecture is a type of variable metric gradient ascent algorithm for maximizing $l = \ln L$.

Finally, we can also generalize Theorem 2 as follows.

Theorem 4 *Assume that the training set $\mathcal{Y}$ is generated by the HME model of Eqs. (47) and (48) and that the number N is sufficiently large. Assume that $\Sigma_{i_1\cdots i_r}$ is diagonal and $\mathbf{v}_{\Sigma_{i_1\cdots i_r}}$ is a vector consisting of the diagonal elements of $\Sigma_{i_1\cdots i_r}$.*

Let $\boldsymbol{\Theta}$ be a vector produced by cascading every vector $\mathbf{v}_{i_1\cdots i_r} = [(\boldsymbol{\theta}^g_{i_1\cdots i_r})^T, \boldsymbol{\theta}^T_{i_1\cdots i_r}, \mathbf{v}^T_{\Sigma_{i_1\cdots i_r}}]^T$ of every node $v_{i_1\cdots i_r}$ in the HME architecture. Let P be a diagonal block matrix with each diagonal item being a positive diagonal block matrix $D^b_{i_1\cdots i_r} = diag[P^q_{i_1\cdots i_r}, P_{i_1\cdots i_r}, P_{\Sigma_{i_1\cdots i_r}}]$. The items of $\boldsymbol{\Theta}$ and P are arranged in such a way that $D^b_{i_1\cdots i_r}$ in P corresponds to $\mathbf{v}_{i_1\cdots i_r}$ in $\boldsymbol{\Theta}$.

Furthermore, assume that on a given domain $D_{\boldsymbol{\Theta}}$

(i) The parameterized functions of all the expert networks and the gating networks have second order continuous derivatives.

(ii) The Hessian matrix $H(\boldsymbol{\Theta}) = \frac{\partial^2 l(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta} \partial \boldsymbol{\Theta}^T}$ is negative definite;

(iii) $\boldsymbol{\Theta}^*$ is a local maximum of $l(\boldsymbol{\Theta})$, and $\boldsymbol{\Theta}^* \in D_{\boldsymbol{\Theta}}$.

Then we have the same conclusion as given in Theorem 2. That is

(1) Letting $-M, -m$ (here $M > m > 0$) be the minimum and maximum eigenvalues of the negatively definite matrix $(P^{\frac{1}{2}})^T H(\boldsymbol{\Theta})(P^{\frac{1}{2}})$ (or equivalently the minimum and maximum eigenvalues of $PH(\boldsymbol{\Theta})$, since we have $PH\mathbf{e} = \lambda\mathbf{e}$ from $(P^{\frac{1}{2}})^T H P^{\frac{1}{2}}\mathbf{e} = \lambda\mathbf{e}$), we have

$$l(\boldsymbol{\Theta}^*) - l(\boldsymbol{\Theta}^{(k)}) \leq r^k[l(\boldsymbol{\Theta}^*) - l(\boldsymbol{\Theta}_0)],$$

$$\|P^{-\frac{1}{2}}(\boldsymbol{\Theta}^{(k)} - \boldsymbol{\Theta}^*)\| \leq |r|^{k/2}\sqrt{\frac{2}{m}[l(\boldsymbol{\Theta}^*) - l(\boldsymbol{\Theta}_0)]},$$

where $r = 1 - (1 - \frac{M}{2})\frac{m^2}{M} < 1$. We also have $0 < |r| < 1$ when $M < 2$.

(2) For any initial point $\boldsymbol{\Theta}_0 \in D_{\boldsymbol{\Theta}}$, $\lim_{k\to\infty} \boldsymbol{\Theta}^{(k)} = \boldsymbol{\Theta}^*$ when $M < 2$.

We should point out that the similarity in the conclusions of Theorem 2 and Theorem 4 does not mean that the EM algorithm for the hierarchical architecture has the same convergence rate as that for the one-level architecture. In the two cases the matrix P is different, and thus M, m, r are also different. This results in different convergence rates. Indeed, in practice, the hierarchical architecture is usually faster. The similarity in the conclusions of the two theorems does mean, however, that the convergence rates for both the algorithms are of the same (linear) order.

Variants of the EM Algorithm and Simulations

Variants of the EM algorithm

For convenience, we denote an EM update of the parameter vector as follows

$$\boldsymbol{\theta}^{(k+1)} = U_p(\boldsymbol{\theta}^{(k)}). \tag{70}$$

From Theorems 1 and 2 and Theorems 3 and 4, we see that this update is actually a line search method along an ascent direction of $l = \ln L$

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + P_{\boldsymbol{\theta}}^{(k)} \frac{\partial l}{\partial \boldsymbol{\theta}}|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(k)}}, \tag{71}$$

with $P_{\boldsymbol{\theta}}^{(k)}$ being a positive definite matrix evaluated at $\boldsymbol{\theta}^{(k)}$. Moreover this update has a linear convergence rate. This link between the EM algorithm and conventional gradient-based optimization techniques suggests the possibility of using acceleration techniques for improving the convergence. In the sequel we suggest two such acceleration techniques.

Modified line search

Eq. (71) can be replaced by a modified line search

$$\begin{aligned} \boldsymbol{\theta}^{(k+1)} &= \boldsymbol{\theta}^{(k)} + \lambda_k \mathbf{d}_k, \\ \mathbf{d}_k &= P_{\boldsymbol{\theta}}^{(k)} \frac{\partial l}{\partial \boldsymbol{\theta}}|_{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(k)} = U_p(\boldsymbol{\theta}^{(k)}) - \boldsymbol{\theta}^{(k)}, \end{aligned} \tag{72}$$

where λ_k is a stepsize which is optimized by maximizing $l(\boldsymbol{\theta}^{(k)} + \lambda_k \mathbf{d}_k)$ with respect to λ_k via a one-dimensional search (e.g., Fibonacci search).

The implementation of a one-dimensional optimization method at every parameter update is typically expensive. One often uses an inaccurate line search by decreasing (e.g., $\lambda_k \rightarrow r\lambda_k$) or increasing (e.g., $\lambda_k \rightarrow \frac{1}{r}\lambda_k$) the stepsize heuristically according to a stopping rule. One frequently used stopping rule is the so-called Goldstein test (Luenberger, 1984). The Goldstein test is implemented as follows

$$\begin{aligned} l(\lambda_k) &\leq l(0) + \varepsilon l'(0)\lambda_k, \\ l(\lambda_k) &> l(0) + (1-\varepsilon) l'(0)\lambda_k, \\ l'(0) &= \mathbf{d}_k^T \frac{\partial l}{\partial \boldsymbol{\theta}}|_{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(k)}, \end{aligned} \tag{73}$$

where $0 < \varepsilon < 1$ is a specified error bound.

Interestingly, if we rewrite the update as

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + \lambda_k [U_p(\boldsymbol{\theta}^{(k)}) - \boldsymbol{\theta}^{(k)}] = (1-\lambda_k)\boldsymbol{\theta}^{(k)} + \lambda_k U_p(\boldsymbol{\theta}^{(k)}),$$

we find that Eq. (72) is identical to the speedup technique for the EM algorithm studied by Peters & Walker (1978a,b), Meilijson(1989) and Redner & Walker (1984). These authors reported a significant speedup for an appropriately selected λ_k even without the Goldstein test.

A speeding up formula based on locally linearization

Using a first-order Taylor expansion of $U_p(\boldsymbol{\theta}^{(k)})$ around $\boldsymbol{\theta}^{(k-1)}$, we have, approximately,

$$U_p(\boldsymbol{\theta}^{(k)}) = U_p(\boldsymbol{\theta}^{(k-1)}) + B(\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^{(k-1)})$$

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + B(\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^{(k-1)})$$

or

$$\Delta\boldsymbol{\theta}_k = B\Delta\boldsymbol{\theta}_{k-1}, \tag{74}$$

where $B = \frac{\partial U_p}{\partial \boldsymbol{\theta}}|_{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(k)}$ and $\Delta\boldsymbol{\theta}_k = \boldsymbol{\theta}^{(k+1)} - \boldsymbol{\theta}^{(k)}$.

For the matrix B, we have the following characteristic equation

$$\det(\lambda I - B) = \lambda^n + \mu_1\lambda^{n-1} + \cdots + \mu_{n-1}\lambda + \mu_n = \lambda^n + \sum_{j=1}^{n} \mu_j \lambda^{n-j} = 0$$

$$\mu_j = (-1)^j \sum_{1 \le i_1 < i_2 < \cdots i_j \le n} \lambda_{i_1}\lambda_{i_1}\cdots\lambda_{i_j} \quad (j = 1, 2, \cdots, n)$$

where $\lambda_i, i = 1, \cdots, n$ are the eigenvalues of B. It follows from the Cayley-Hamilton theorem that

$$B^n + \sum_{j=1}^{n} \mu_j B^{n-j} = 0$$

Multiplying by $\Delta\boldsymbol{\theta}_{k-n}$ $(k \ge n)$, we have

$$B^n \Delta\boldsymbol{\theta}_{k-n} + \sum_{j=1}^{n} \mu_j B^{n-j} \Delta\boldsymbol{\theta}_{k-n} = 0. \tag{75}$$

From Eq. (74), we obtain

$$\Delta\boldsymbol{\theta}_{k-j} = B\Delta\boldsymbol{\theta}_{k-j-1} = \cdots = B^{n-j}\Delta\boldsymbol{\theta}_{k-n}, \ j = 0, 1, \cdots, n.$$

Substituting into Eq. (75), we have

$$\Delta\boldsymbol{\theta}_k + \sum_{j=1}^{n} \mu_j \Delta\boldsymbol{\theta}_{k-j} = 0. \tag{76}$$

Assuming that in comparison with the first l eigenvalues, the remaining eigenvalues can be neglected, we have approximately

$$\lambda^n + \sum_{j=1}^{l} \mu_j \lambda^{n-j} = 0, \quad B^n + \sum_{j=1}^{l} \mu_j B^{n-j} = 0$$

and correspondingly

$$\boldsymbol{\theta}_i + \sum_{j=1}^{l} \mu_j \Delta\boldsymbol{\theta}_{i-j} = 0, \quad i = k, k+1, \cdots. \tag{77}$$

The approximation becomes exact when the last $n - l$ eigenvalues are zero.

By minimizing $\|\boldsymbol{\theta}_k + \sum_{j=1}^{l} \mu_j \Delta\boldsymbol{\theta}_{k-j}\|^2$, we obtain the following linear equation for solving $\boldsymbol{\mu} = [\mu_1, \cdots, \mu_l]^T$

$$S\ \boldsymbol{\mu} = \mathbf{s}_0, \quad S = [s_{ij}]_{l\times l}, \quad \mathbf{s}_0 = [-s_{01}, -s_{02}, \cdots, -s_{0l}]^T,$$

$$s_{ij} = s_{ji} = (\Delta\boldsymbol{\theta}_{k-i})^T(\Delta\boldsymbol{\theta}_{k-j}), \tag{78}$$

Moreover, we have

$$\sum_{i=k}^{\infty}\sum_{j=0}^{l}\mu_j\Delta\boldsymbol{\theta}_{i-j} = \sum_{j=0}^{l}\mu_j\Delta\boldsymbol{\theta}_{k-j} + \sum_{i=k+1}^{\infty}\Delta\boldsymbol{\theta}_i + \sum_{j=1}^{l}\sum_{i=k+1}^{\infty}\mu_j\Delta\boldsymbol{\theta}_{i-j}.$$

where $\mu_0 = 1$. From Eq. (77) and

$$\sum_{i=k+1}^{\infty}\mu_j\Delta\boldsymbol{\theta}_{i-j} = -\Delta\boldsymbol{\theta}_{k+1} + \boldsymbol{\theta}^*$$

(where $\boldsymbol{\theta}^* = \lim_{k\to\infty}\Delta\boldsymbol{\theta}_k$), we have

$$-\Delta\boldsymbol{\theta}_{k+1} + \boldsymbol{\theta}^* + \sum_{j=1}^{l}\mu_j(-\Delta\boldsymbol{\theta}_{k+1-j} + \boldsymbol{\theta}^*) = 0.$$

Using this equation together with

$$\sum_{j=0}^{l}\mu_j\Delta\boldsymbol{\theta}_{k+1-j} = \Delta\boldsymbol{\theta}_{k+1}\sum_{j=0}^{l}\mu_j - \sum_{i=0}^{l-1}[\sum_{j=i+1}^{l}\mu_j]\Delta\boldsymbol{\theta}_{k-i}$$

we finally obtain

$$\boldsymbol{\theta}^*_{k+1} = \boldsymbol{\theta}_{k+1} - \frac{\sum_{i=0}^{l-1}[\sum_{j=i+1}^{l}\mu_j]\Delta\boldsymbol{\theta}_{k-i}}{\sum_{j=0}^{l}\mu_j}. \tag{79}$$

This formula can be used for speeding up the EM algorithm in two ways.

- Given an initial $\boldsymbol{\theta}_0$, we compute via the EM update $\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_{l+1}$, and then from $\boldsymbol{\theta}_j, j = 1, \cdots, l$, we solve $\mu_1, \cdots, \mu_l$ utilizing Eq. (78). This yields a new $\boldsymbol{\theta}^*_{l+1}$ via Eq. (79). We then let $\boldsymbol{\theta}_0 = \boldsymbol{\theta}^*_{l+1}$ and repeat the cycle.
- Instead of starting a new cycle after obtaining $\boldsymbol{\theta}^*_{k+1}$, we simply let $\boldsymbol{\theta}_{l+1} = \boldsymbol{\theta}^*_{l+1}$, and use the EM update (Eq. 70) to obtain a new $\boldsymbol{\theta}_{l+2}$, then we use $\boldsymbol{\theta}_1, \boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_{l+2}$ to get a $\boldsymbol{\theta}^*_{l+2}$. Similarly, after getting $\boldsymbol{\theta}^*_k, k > l$, we let $\boldsymbol{\theta}_k = \boldsymbol{\theta}^*_k$ and use the EM update to get a new $\boldsymbol{\theta}_{k+1}$, and then use $\boldsymbol{\theta}_{k-l}, \cdots, \boldsymbol{\theta}_{k+1}$ to get a $\boldsymbol{\theta}^*_{k+1}$.

Specifically, when $l = 1$, we have

$$\boldsymbol{\theta}^*_{k+1} = \boldsymbol{\theta}_k + \frac{\Delta\boldsymbol{\theta}_k}{1+\mu_1}, \quad \mu_1 = -\frac{(\Delta\boldsymbol{\theta}_k)^T(\Delta\boldsymbol{\theta}_{k-1})}{(\Delta\boldsymbol{\theta}_{k-1})^T(\Delta\boldsymbol{\theta}_{k-1})}. \tag{80}$$

In this case, the extra computation required by the acceleration technique is quite small, and we recommend the use of this approach in practice.

Simulations

We conducted two sets of computer simulations to compare the performance of the EM algorithm with the two variants described in the previous section. The training data for each simulation consisted of 1000 data points generated from the piecewise linear function $y = a_1x + a_2 + n_t,\ x \in [x_L, x_U]$ and $y = a_1'x + a_2' + n_t,\ x \in [x_L', x_U']$, where n_t is a gaussian random variable with zero mean and variance $\sigma = 0.3$. Training data were sampled from the first function with probability 0.4 and from the second function with probability 0.6.

We studied a modular architecture with $K = 2$ expert networks. The experts were linear; that is, $f_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)$ were linear functions $[x, 1]^T \boldsymbol{\theta}_j$. For the gating net, we have

$$s_j = [x, 1]^T \boldsymbol{\theta}_{gj},$$

where g_j is given by Eq. (2). For simplicity, we updated $\boldsymbol{\theta}_{gj}$ by gradient ascent

$$\boldsymbol{\theta}_{gj}^{(k+1)} = \boldsymbol{\theta}_{gj}^{(k)} + r_g \frac{\partial Q}{\partial \boldsymbol{\theta}_{gj}}. \tag{81}$$

The learning rate parameter was set to $r_g = 0.05$ for the first data set and $r_g = 0.002$ for the second data set. We used Eq. (20) and Eq. (22) to update the parameters $\boldsymbol{\theta}_j^{(k+1)}, j = 1, 2$ and $\sigma_j^{(k+1)}, j = 1, 2$, respectively.

The initial values of $\boldsymbol{\theta}_{gj}^{(0)}, \boldsymbol{\theta}_j^{(0)}, \sigma_j^{(0)}, j = 1, 2$ were picked randomly. To compare the performance of the algorithms, we let each algorithm start from the same set of initial values.

The first data set (see Figure 3(a)) was generated using the following parameter values

$$a1 = 0.8,\ a2 = 0.4,\ x_L = -1.0,\ x_U = 1.0,\ a_1' = -1.0,\ a_2' = 3.6,\ x_L' = 2.0,\ x_U' = 4.0.$$

The performance of the original algorithm, the modified line search variant with $\lambda_k = 1.1$, the modified line search variant with $\lambda_k = 0.5$, and the algorithm based on local linearization are shown in Figures 4, 5, 6, and 7, respectively. As seen in Figures 4(a) and 5(a), the log likelihood converged after 19 steps using both the original algorithm and the modified line search variant with $\lambda_k = 1.1$. When a smaller value was used ($\lambda_k = 0.5$), the algorithm converged after 24 steps (Figure 6(a)). Trying other values of λ_k, we verified that $\lambda_k < 1$ slows down the convergence, while $\lambda_k > 1$ may speed up the convergence (cf. Redner & Walker, 1984). We found, however, that the outcome was quite sensitive to the selection of the value of λ_k. For example, setting $\lambda_k = 1.2$ led the algorithm to diverge. Allowing λ_k to be determined by the Goldstein test (Eq. 73) yielded results similar to the original algorithm, but required more computer time. Finally, Figure 7(a) shows that the algorithm based on local linearization yielded substantially improved convergence—the log likelihood converged after only 8 steps.

Figures 4(b) and 4(c) show the evolution of the parameters for the first expert net and the second expert net, respectively. Comparison of these figures to Figures 5(b) and 5(c) shows that the original algorithm and the modified line search variant with $\lambda_k = 1.1$ behaved almost identically: $\boldsymbol{\theta}_1$ converged to the correct solution after about 18 steps in either case. Figures 6(b) and 6(c) show the slowdown obtained by using $\lambda_k = 0.5$. Figures 7(b) and 7(c) show the improved performance obtained using the local linearization algorithm. In this case, the weight vectors converged to the correct values within 7 steps.

Panel (d) in each of the figures show the evolution of the estimated variances σ_1^2, σ_2^2. The results were similar to those for the expert net parameters. Again, the algorithm based on local linearization yielded significantly faster convergence than the other algorithms.

A second simulation was run using the following parameter values (see Figure 3(b))

$$a1 = 0.8,\ a2 = 0.4,\ x_L = -1.0,\ x_U = 2.0,\ a_1' = -1.2,\ a_2' = 2.4,\ x_L' = 1.0,\ x_U' = 4.0.$$

The results obtained in this simulation were similar to those obtained in the first simulation. The EM algorithm converged in 11 steps and the local linearization algorithm converged in 6 steps.

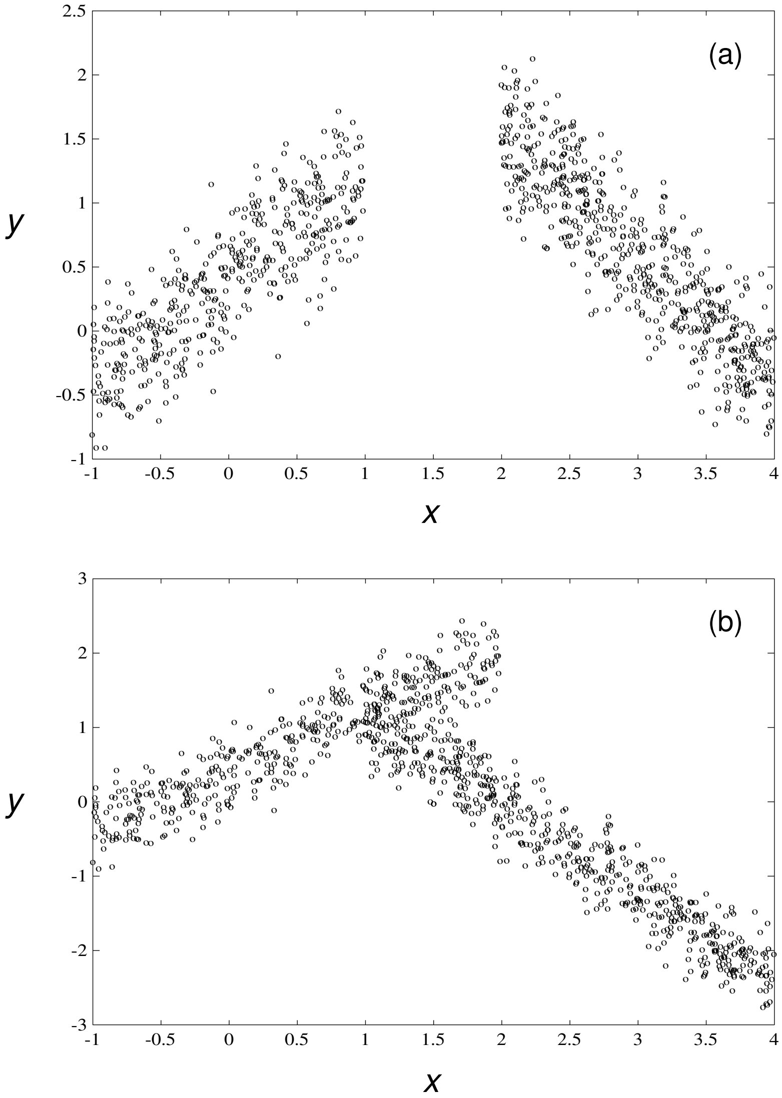

Figure 3: The data sets for the simulation experiments.

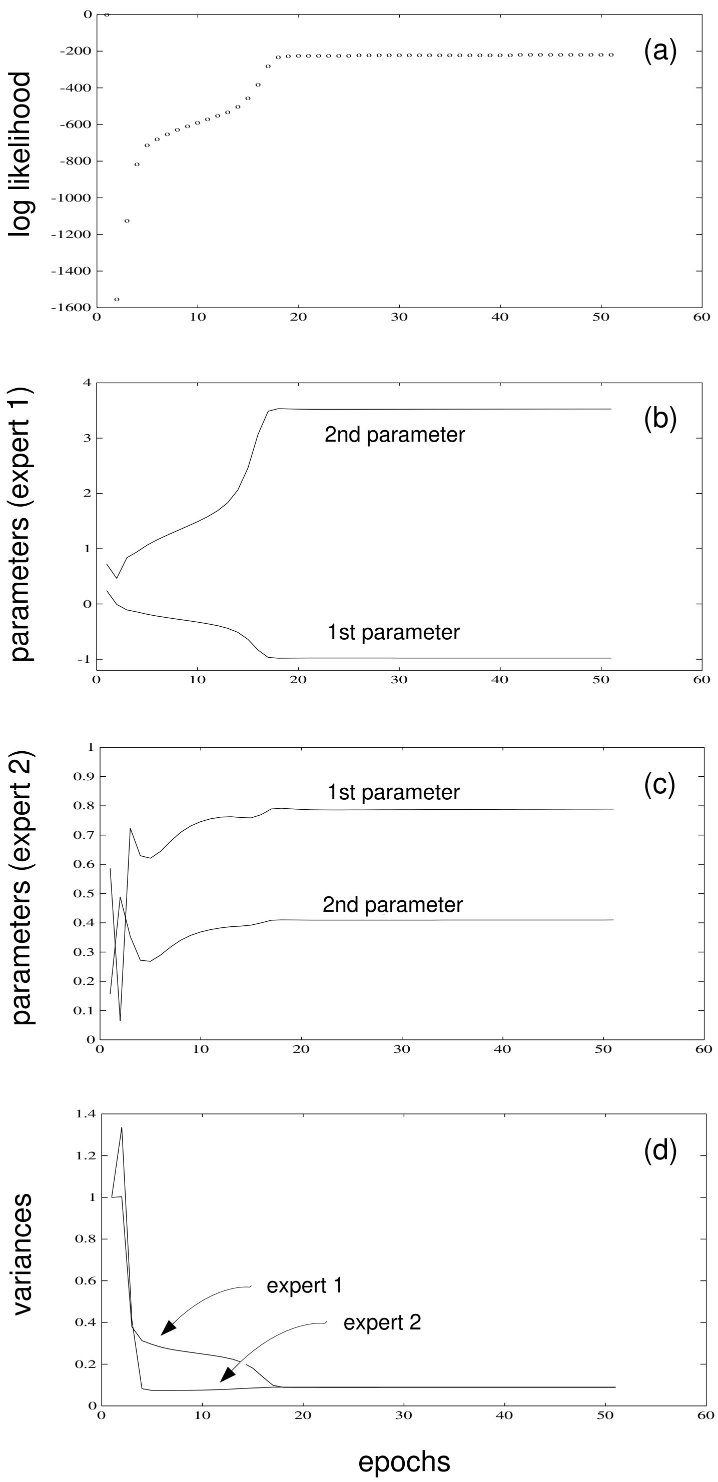

Figure 4: The performance of the original EM algorithm: (a) The evolution of the log likelihood; (b) the evolution of the parameters for expert network 1; (c) the evolution of the parameters for expert network 2; (d) the evolution of the variances.

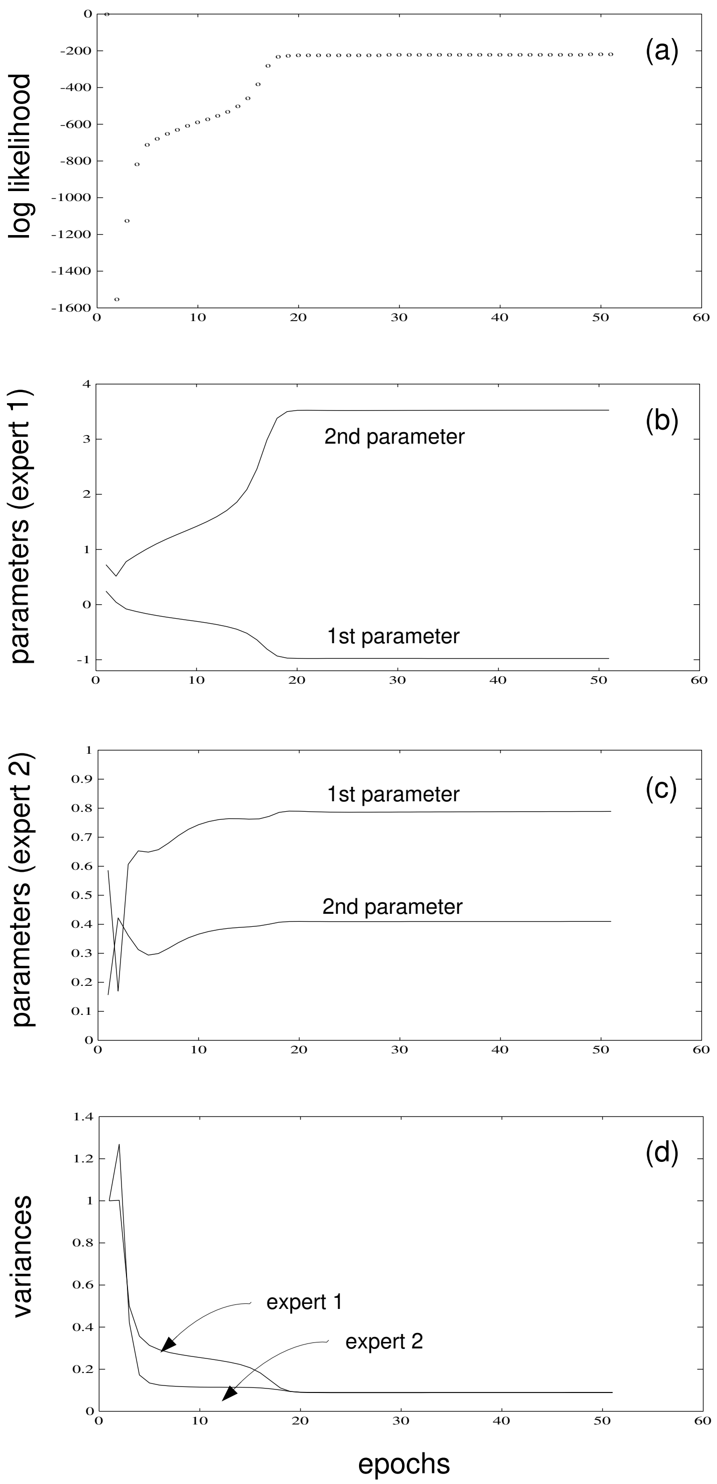

Figure 5: The performance of the linesearch variant with $\lambda_k = 1.1$.

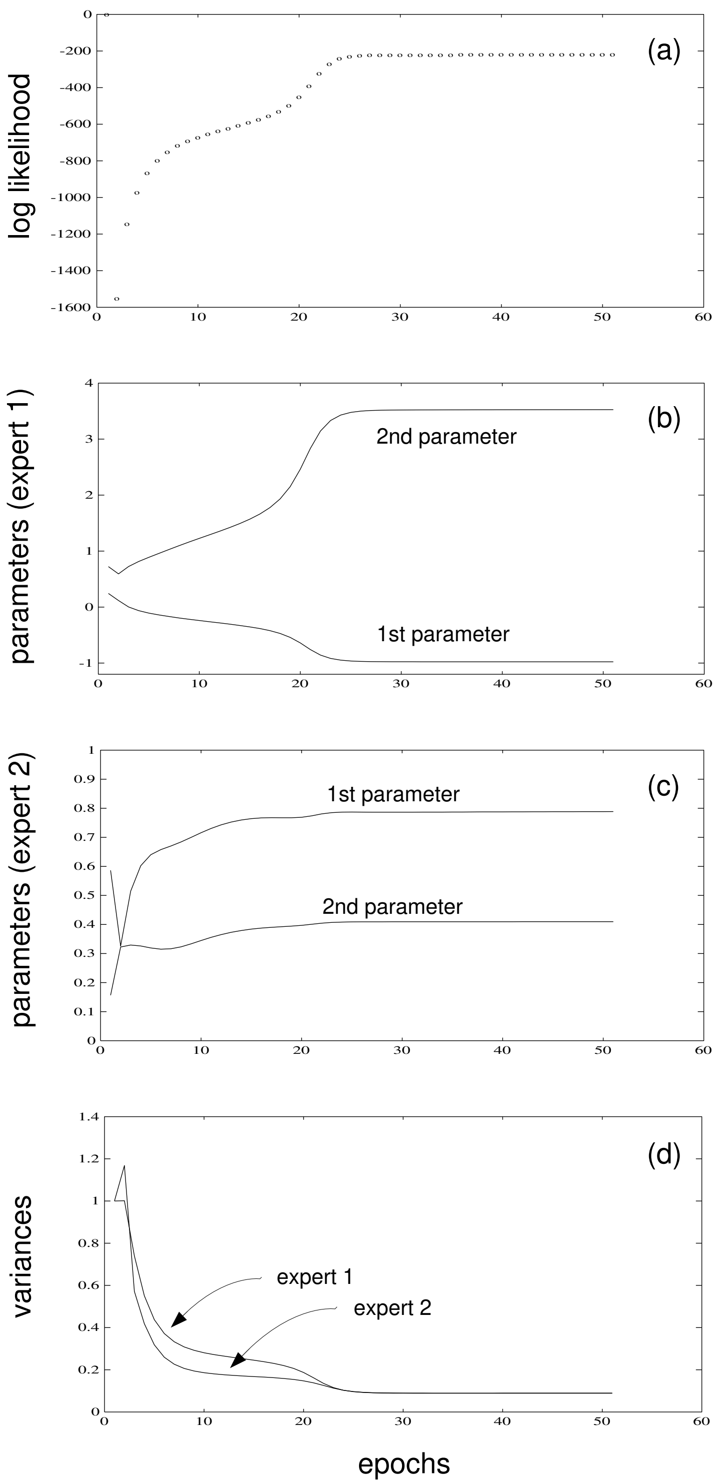

Figure 6: The performance of the linesearch variant with $\lambda_k = 0.5$.

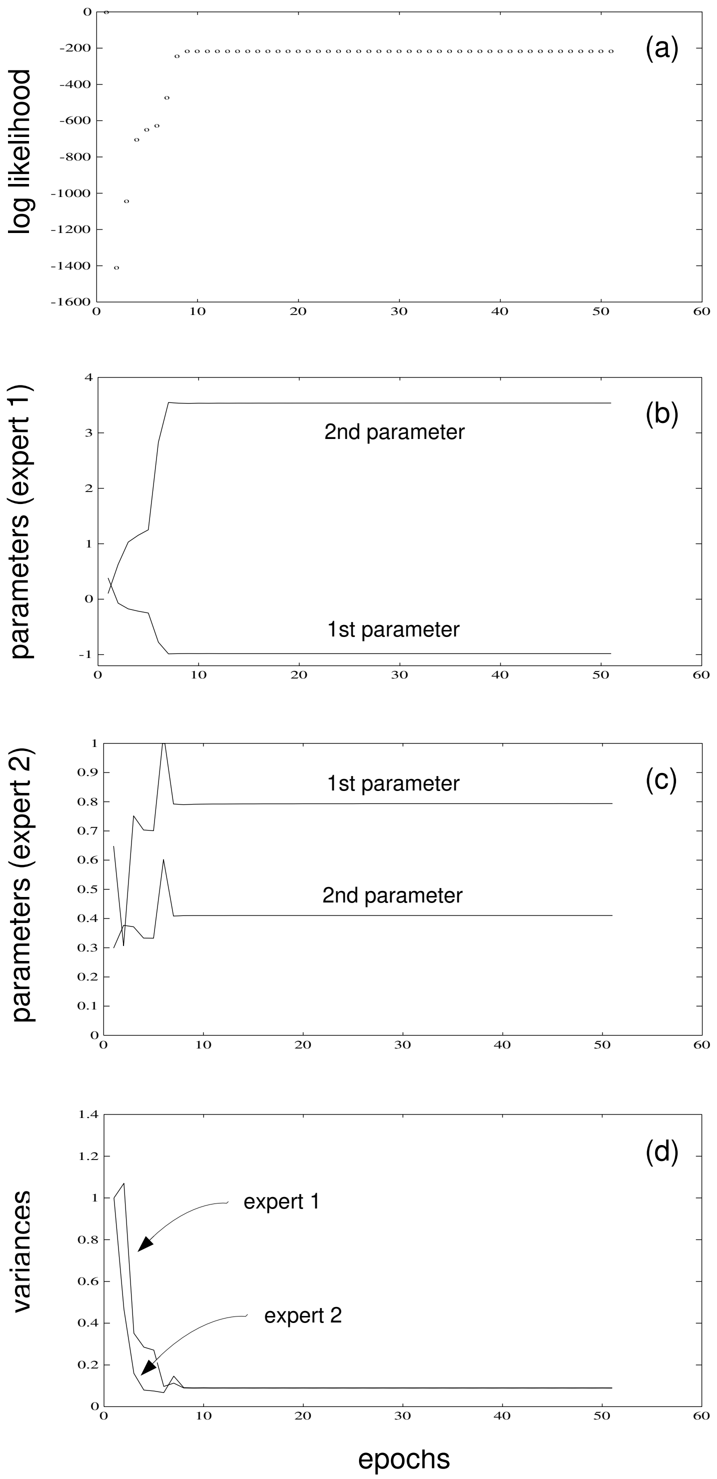

Figure 7: The performance of the algorithm based on local linearization.

The results from a number of other simulation experiments confirmed the results reported here. In general the algorithm based on local linearization provided significantly faster convergence than the original EM algorithm. The modified line search variant did not appear to converge faster (if the parameter λ_k was fixed). We also tested gradient ascent in these experiments and found that convergence was generally one to two orders of magnitude slower than convergence of the EM algorithm and its variants. Moreover, convergence of gradient ascent was rather sensitive to the learning rate and the initial values of the parameters.

Concluding remarks

Finite mixture models have become increasingly popular as models for unsupervised learning, partly because they occupy an interesting niche between parametric and nonparametric approaches to statistical estimation. Mixture-based approaches are parametric in that particular parametric forms must be chosen for the component densities, but they can also be regarded as nonparametric by allowing the number of components of the mixture to grow. The advantage of this niche in statistical theory is that these models have much of the flexibility of nonparametric approaches, but retain some of the analytical advantages of parametric approaches (McLachlan & Basford, 1988). Similar remarks can be made in the case of supervised learning: The ME architecture and the HME architecture provide flexible models for general nonlinear regression while retaining a strong flavor of parametric statistics. The latter model, in particular, compares favorably to decision tree models in this regard (Jordan & Jacobs, in press).

In the current paper we have contributed to the theory of mixture-based supervised learning. We have analyzed an EM algorithm for ME and HME architectures and provided theorems on the convergence of this algorithm. In particular, we have shown that learning algorithm can be regarded as a variable metric algorithm with its metric matrix P being positive definite, so that the searching direction of the algorithm always has a positive projection on the gradient of the log likelihood. We have shown that the algorithm converges linearly, with a rate determined by the difference between the minimal and maximal eigenvalues of a negative definite matrix.

Similar results to those obtained here can also be obtained for the case of the unsupervised learning of finite mixtures (Xu & Jordan, 1993).

References

Baum, L.E., & Sell, G.R. (1968). Growth transformation for functions on manifolds. *Pac. J. Math., 27*, 211-227.

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

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

De Veaux, R.D. (1986). *Parameter estimation for a mixture of linear regressions.* Ph.D. Dissertation and Tech. Rept. No. 247, Department of Statistics, Stanford University, Stanford, CA.

Dempster, A.P., Laird, N.M., & Rubin, D.B. (1977). Maximum-likelihood from incomplete data via the EM algorithm. *J. of Royal Statistical Society, B39*, 1-38.

Friedman, J.H. (1991). Multivariate adaptive regression splines. *The Annals of Statistics, 19*, 1-141.

Jacobs, R.A., Jordan, M.I., Nowlan, S.J., & Hinton, G.E. (1991). Adaptive mixtures of local experts. *Neural Computation, 3*, 79-87.

Jordan, M.I. & Jacobs, R.A. (1992). Hierarchies of adaptive experts. In J.E. Moody, S. Hanson & R.P. Lippmann, (Eds.), *Advances in Neural Information Processing System 4*. San Mateo: Morgan Kaufmann, 985-992.

Jordan, M.I. & Jacobs, R.A. (in press). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*.

Little, R.J.A., & Rubin, D.B. (1987). *Statistical Analysis with Missing Data*. New York: John Wiley.

McCullagh, P. & Nelder, J.A. (1983). *Generalized Linear Models*. London: Chapman and Hall.

McLachlan, G.J., & Basford, K.E. (1988). *Mixture Models: Inference and Application to Clustering*. New York: Dekker.

Meilijson, I. (1989). A fast improvement to the EM algorithm on its own terms. *J. of Royal Statistical Society, B51*, 127-138.

Peters, B.C. & Walker, H.F. (1978a). An iterative procedure for obtaining maximum-likelihood estimates of the parameters for a mixture of normal distributions. *SIAM J. Applied Mathematics, 35*, 362-378.

Peters, B.C., & Walker, H.F. (1978b). The numerical evaluation of the maximum-likelihood estimates of a subset of mixture proportions, *SIAM J. Applied Mathematics, 35*, 447-452.

Quandt, R.E. & Ramsey, J.B. (1972). A new approach to estimating switching regressions. *J. American Statistical Society, 67*, 306-310.

Quandt, R.E. & Ramsey, J.B. (1978). Estimating mixtures of normal distribution and switching regressions. *J. American Statistical Society, 73*, 730-738.

Redner, R.A., & Walker, H.F. (1984). Mixture densities, maximum likelihood, and the EM algorithm. *SIAM Review 26*, 195-239.

Wu, C.F.J. (1983). On the convergence properties of the EM algorithm. *The Annals of Statistics, 11*, 95-103.

Xu, L., & Jordan, M.I. (1993). Theoretical and experimental studies of the EM algorithm for unsupervised learning based on finite Gaussian mixtures. MIT Computational Cognitive Science Tech. Rep. 9302, MIT, Cambridge, MA.

Appendix

The theoretical results presented in the main text show that the EM algorithm for the ME and HME architectures converges linearly with a rate determined by the condition number of a particular matrix. These results were obtained for a special case in which the expert networks are linear with a gaussian probability model and the gating networks are multinomial logit models. In this section we discuss extensions of these results to other architectures.

We first note that Theorems 2 and 4 make no specific reference to the particular probability models utilized in specifying the architecture. The results on convergence rate in these theorems require only that the matrix P be positive definite. These theorems apply directly to other architectures if the corresponding P matrices can be shown to be positive definite. We therefore need only consider generalizations of Theorem 1, the theorem which established the positive definiteness of P for the generalized linear ME architectures. An analogous generalization of Theorem 3 for the HME architectures can also be obtained.

Let us consider the case in which the function implemented by each expert network ($\mathbf{f}_j(\mathbf{x}, \boldsymbol{\theta}_j)$) is nonlinear in the parameters. We consider two possible updates for the parameters: (1) a gradient algorithm

$$\boldsymbol{\theta}_j^{(k+1)} = \boldsymbol{\theta}_j^{(k)} + \gamma_j \mathbf{e}_j^{(k)}, \tag{82}$$

where

$$\mathbf{e}_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)}(t) \frac{\partial \mathbf{f}_j^T(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\partial \boldsymbol{\theta}_j^{(k)}} (\Sigma_j^{(k)})^{-1} [\mathbf{y}^{(t)} - \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})], \tag{83}$$

and (2) a Newton algorithm:

$$\boldsymbol{\theta}_j^{(k+1)} = \boldsymbol{\theta}_j^{(k)} + \gamma_j (R_j^{(k)})^{-1} \mathbf{e}_j^{(k)}, \tag{84}$$

where

$$R_j^{(k)} = \sum_{t=1}^{N} h_j^{(k)}(t) \frac{\partial {\mathbf{f}_j}^T(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\partial \boldsymbol{\theta}_j^{(k)}} (\Sigma_j^{(k)})^{-1} \frac{\partial \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\partial (\boldsymbol{\theta}_j^{(k)})^T}, \tag{85}$$

where $\gamma_j > 0$ is a learning rate.

These updates are covered by the following extension of Theorem 1.

Theorem 1A 1 *For the model given by Eq. (4) and the updates given by Eq. (83) or Eq. (85), we have:*

$$\boldsymbol{\theta}_j^{(k+1)} - \boldsymbol{\theta}_j^{(k)} = P_j^{(k)} \frac{\partial l}{\partial \boldsymbol{\theta}_j} |_{\boldsymbol{\theta}_j = \boldsymbol{\theta}_j^{(k)}}, \; j = 1, \cdots, K,$$

where $P_j^{(k)}$ is positive definite.

Proof. For the gradient descent algorithm, we have $P_j^{(k)} = \gamma_j I_K$, which is obviously positive definite because $\gamma_j > 0$. For the Newton algorithm, we have that $P_j^{(k)} = \gamma_j (R_j^{(k)})^{-1}$. We now show that this matrix is positive definite. For an arbitrary vector $\mathbf{u}$, we have

$$\begin{aligned} \mathbf{u}^T R_j^{(k)} \mathbf{u} &= \sum_{t=1}^{N} h_j^{(k)}(t) \mathbf{u}^T \frac{\partial \mathbf{f}_j^T(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\partial \boldsymbol{\theta}_j^{(k)}} (\Sigma_j^{(k)})^{-1} \frac{\partial \mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\partial (\boldsymbol{\theta}_j^{(k)})^T} \mathbf{u} \\ &= \sum_{t=1}^{N} h_j^{(k)}(t) \mathbf{v}^T (\Sigma_j^{(k)})^{-1} \mathbf{v} \\ &\geq 0. \end{aligned}$$

Equality holds only when $\mathbf{v} = \mathbf{u}^T \frac{\partial \mathbf{f}_j^T(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j^{(k)})}{\partial \boldsymbol{\theta}_j^{(k)}} = 0$, since $\Sigma_j^{(k)}$ is positive definite with probability one. This is impossible for any $\mathbf{u}$. So with probability one, $R_j^{(k)}$ (and thus $(R_j^{(k)})^{-1}$ also) is positive definite. $\square$

Note that the Newton update (Eq. 85) is particularly appropriate for the case in which the experts are generalized linear models (McCullagh & Nelder, 1983); that is, the case in which $\mathbf{f}_j(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j) = [f_{j1}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j), \cdots, f_{jd_y}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j)]$ (d_y is the dimension of $\mathbf{y}$) with

$$f_{ji}(\mathbf{x}^{(t)}, \boldsymbol{\theta}_j) = F_{ji}([\boldsymbol{\theta}_{j,1}, \cdots, \boldsymbol{\theta}_{j,m}]^T \mathbf{x}^{(t)} + \boldsymbol{\theta}_{j,m+1}),$$

where $F_{ji}(.)$ is a continuous univariate nonlinear function known as the *link* function. In this case the Newton algorithm reduces to the IRLS algorithm. The extension to generalized linear models also allows probability models from the generalized exponential family (cf. Jordan & Jacobs, in press) and Theorem 1A is applicable to this case as well.

We can also consider the case in which the gating network is nonlinear in the parameters. Both the Newton update (IRLS update) and the gradient update are applicable in this case. Theorem 1 already established that the Newton update for the gating network involves a positive definite P_g matrix. As in Theorem 1A, the result for the gradient update is immediate.