

---

# Batch and On-line Parameter Estimation of Gaussian Mixtures Based on the Joint Entropy

---

**Yoram Singer**

AT&T Labs  
singer@research.att.com

**Manfred K. Warmuth**

University of California, Santa Cruz  
manfred@cse.ucsc.edu

## Abstract

We describe a new iterative method for parameter estimation of Gaussian mixtures. The new method is based on a framework developed by Kivinen and Warmuth for supervised on-line learning. In contrast to gradient descent and EM, which estimate the mixture's covariance matrices, the proposed method estimates the *inverses* of the covariance matrices. Furthermore, the new parameter estimation procedure can be applied in both on-line and batch settings. We show experimentally that it is typically faster than EM, and usually requires about half as many iterations as EM. We also describe experiments with digit recognition that demonstrate the merits of the on-line version.

## 1 Introduction

Mixture models, in particular mixtures of Gaussians, have been a popular tool for density estimation, clustering, and un-supervised learning with a wide range of applications (see for instance [7, 2] and the references therein). Mixture models are one of the most useful tools for handling incomplete data, in particular hidden variables. For Gaussian mixtures the hidden variables indicate for each data point the index of the Gaussian that generated it. Thus, the model is specified by a joint density between the observed and hidden variables. The common technique used for estimating the parameters of a stochastic source with hidden variables is the EM algorithm. In this paper we describe a new technique for estimating the parameters of Gaussian mixtures. The new parameter estimation method is based on a framework developed by Kivinen and Warmuth [12] for supervised on-line learning. This framework was successfully used in a large number of supervised and un-supervised problems (see for instance [9, 8, 13, 1]).

Our goal is to find a local minimum of a loss function which, in our case, is the negative log likelihood induced by a mixture of Gaussians. However, rather than minimizing the loss directly we add a term measuring the distance of the new parameters to the old ones. This distance is useful for iterative parameter estimation procedures. Its purpose is to keep the new parameters close to the old ones. The method for deriving iterative parameter estimation can be used in batch settings as well as on-line settings where the parameters are updated after each observation. The distance used for deriving the parameter estimation method in this paper is the relative entropy between the old and new joint density of the observed and hidden variables. For brevity we term the new iterative parameter estimation method the *joint-entropy* (JE) update.

The JE update shares a common characteristic with the Expectation Maximization [6, 17] algorithm as it first calculates the same expectations. However, it replaces

the maximization step with a different update of the parameters. For instance, it updates the inverse of the covariance matrix of each Gaussian in the mixture, rather than the covariance matrices themselves. We found in our experiments that the JE update often requires half as many iterations as EM. It is also straightforward to modify the proposed parameter estimation method for on-line setting where the parameters are updated after each new observation. As we demonstrate in our experiments with digit recognition, the on-line version of the JE update is especially useful in situations where the observations are generated by a non-stationary stochastic source.

## 2 Notation and preliminaries

Let  $S$  be a sequence of training examples  $\langle x_1, x_2, \dots, x_N \rangle$  where each  $x_i$  is a  $d$ -dimensional vector in  $\mathbb{R}^d$ . To model the distribution of the examples we use  $m$   $d$ -dimensional Gaussians. The parameters of the  $i$ -th Gaussian are denoted by  $\Theta_i$  and they include the mean-vector and the covariance matrix

$$\mu_i = E(\mathbf{x}|\Theta_i) \quad C_i = E((\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^T|\Theta_i).$$

The density function of the  $i$ th Gaussian, denoted  $P(\mathbf{x}|\Theta_i)$ , is

$$P(\mathbf{x}|\Theta_i) = (2\pi)^{-d/2} |C_i|^{-1/2} e^{-\frac{1}{2}(\mathbf{x} - \mu_i)^T C_i^{-1} (\mathbf{x} - \mu_i)}.$$

We denote the entire set of parameters of a Gaussian mixture by  $\Theta = \{\Theta_i\}_{i=1}^m = \{w_i, \mu_i, C_i\}_{i=1}^m$  where  $\mathbf{w} = (w_1, \dots, w_m)$  is a non-negative vector of mixture coefficients such that  $\sum_{i=1}^m w_i = 1$ . We denote by  $P(\mathbf{x}|\Theta) = \sum_{i=1}^m w_i P(\mathbf{x}|\Theta_i)$  the likelihood of an observation  $\mathbf{x}$  according to a Gaussian mixture with parameters  $\Theta$ . Let  $\Theta_i$  and  $\tilde{\Theta}_i$  be two Gaussian distributions. For brevity, we denote by  $E_i(Z)$  and  $\tilde{E}_i(Z)$  the expectation of a random variable  $Z$  with respect to  $\Theta_i$  and  $\tilde{\Theta}_i$ . Let  $f$  be a parametric function whose parameters constitute a matrix  $A = (a_{ij})$ . We denote by  $\partial f / \partial A$  the matrix of partial derivatives of  $f$  with respect to the elements in  $A$ . That is, the  $ij$  element of  $\partial f / \partial A$  is  $\partial f / \partial a_{ij}$ . Similarly, let  $B = (b_{ij}(x))$  a matrix whose elements are functions of a scalar  $x$ . Then, we denote by  $dB/dx$  the matrix of derivatives of the elements in  $B$  with respect to  $x$ , namely, the  $ij$  element of  $dB/dx$  is  $db_{ij}(x)/dx$ .

## 3 The framework for deriving updates

Kivinen and Warmuth [12] introduced a general framework for deriving on-line parameter updates. In this section we describe how to apply their framework for the problem of parameter estimation of Gaussian mixtures in a batch setting. We later discuss how a simple modification gives the on-line updates.

Given a set of data points  $S$  in  $\mathbb{R}^d$  and a number  $m$ , the goal is to find a set of  $m$  Gaussians that minimize the loss on the data, denoted as  $\text{loss}(S|\Theta)$ . For density estimation the natural loss function is the negative log-likelihood of the data  $\text{loss}(S|\Theta) = -(1/|S|) \ln P(S|\Theta) \stackrel{\text{def}}{=} -(1/|S|) \sum_{\mathbf{x} \in S} \ln P(\mathbf{x}|\Theta)$ . The best parameters which minimize the above loss cannot be found analytically. The common approach is to use iterative methods such as EM [6, 17] to find a local minimizer of the loss.

In an iterative parameter estimation framework we are given the old set of parameters  $\Theta^t$  and we need to find a set of new parameters  $\Theta^{t+1}$  that induce smaller loss. The framework introduced by Kivinen and Warmuth [12] deviates from the common approaches as it also requires to the new parameter vector to stay ‘‘close’’ to the old set of parameters which incorporates all that was learned in the previous iterations. The distance of the new parameter setting  $\Theta^{t+1}$  from the old setting  $\Theta^t$  is measured by a non-negative distance function  $\Delta(\Theta^{t+1}, \Theta^t)$ . We now search for a new set of parameters  $\Theta^{t+1}$  that minimizes the distance summed with the loss multiplied by  $\eta$ . Here  $\eta$  is a non-negative number measuring the relative importance of the distance versus the loss. This parameter  $\eta$  will become the learning

rate of the update. More formally, the update is found by setting  $\Theta^{t+1} = \arg \min_{\tilde{\Theta}} U^t(\tilde{\Theta})$  where  $U^t(\tilde{\Theta}) = \Delta(\tilde{\Theta}, \Theta^t) + \eta \text{loss}(S|\tilde{\Theta}) + \lambda(\sum_{i=1}^m \tilde{w}_i - 1)$ . (We use a Lagrange multiplier  $\lambda$  to enforce the constraint that the mixture coefficients sum to one.) By choosing the appropriate distance function and  $\eta = 1$  one can show that EM becomes the above update.

For most distance functions and learning rates the minimizer of the function  $U^t(\tilde{\Theta})$  cannot be found analytically as both the distance function and the log-likelihood are usually non-linear in  $\tilde{\Theta}$ . Instead, we expand the log-likelihood using a first order Taylor expansion around the old parameter setting. This approximation degrades the further the new parameter values are from the old ones, which further motivates the use of the distance function  $\Delta(\tilde{\Theta}, \Theta^t)$  (see also the discussion in [9]). We now seek a new set of parameters  $\Theta^{t+1} = \arg \min_{\tilde{\Theta}} V^t(\tilde{\Theta})$  where

$$V^t(\tilde{\Theta}) = \Delta(\tilde{\Theta}, \Theta^t) + \eta \left( \text{loss}(S|\Theta^t) + (\tilde{\Theta} - \Theta^t) \cdot \nabla_{\Theta} \text{loss}(S|\Theta^t) \right) + \lambda \left( \sum_{i=1}^m \tilde{w}_i - 1 \right). \quad (1)$$

Here  $\nabla_{\Theta} \text{loss}(S|\Theta^t)$  denotes the gradient of the loss at  $\Theta^t$ . We use the above method Eq. (1) to derive the updates of this paper. For density estimation, it is natural to use the relative entropy between the new and old density as a distance. In this paper we use the joint density between the observed (data points) and hidden variables (the indices of the Gaussians). This motivates the name joint-entropy update.

#### 4 Entropy based distance functions

We first consider the relative entropy between the new and old parameter parameters of a *single* Gaussian. Using the notation introduced in Sec. 2, the relative entropy between two Gaussian distributions denoted by  $\tilde{\Theta}_i, \Theta_i$  is

$$\begin{aligned} \Delta(\tilde{\Theta}_i, \Theta_i) &\stackrel{\text{def}}{=} \int_{\mathbf{x} \in \mathbb{R}^d} P(\mathbf{x}|\tilde{\Theta}_i) \ln \frac{P(\mathbf{x}|\tilde{\Theta}_i)}{P(\mathbf{x}|\Theta_i)} d\mathbf{x} \\ &= \frac{1}{2} \ln \frac{|\mathbf{C}_i|}{|\tilde{\mathbf{C}}_i|} - \frac{1}{2} \tilde{E}_i \left( (\mathbf{x} - \tilde{\mu}_i)^T \tilde{\mathbf{C}}_i^{-1} (\mathbf{x} - \tilde{\mu}_i) \right) + \frac{1}{2} \tilde{E}_i \left( (\mathbf{x} - \mu_i)^T \mathbf{C}_i^{-1} (\mathbf{x} - \mu_i) \right) \end{aligned}$$

Using standard (though tedious) algebra we can rewrite the expectations as follows:

$$\Delta(\tilde{\Theta}_i, \Theta_i) = \frac{1}{2} \ln \frac{|\mathbf{C}_i|}{|\tilde{\mathbf{C}}_i|} - \frac{d}{2} + \frac{1}{2} \text{tr}(\mathbf{C}_i^{-1} \tilde{\mathbf{C}}_i) + \frac{1}{2} (\tilde{\mu}_i - \mu)^T \mathbf{C}_i^{-1} (\tilde{\mu}_i - \mu). \quad (2)$$

The relative entropy between the new and the old *mixture* models is the following

$$\Delta(\tilde{\Theta}, \Theta) \stackrel{\text{def}}{=} \int_{\mathbf{x}} P(\mathbf{x}|\tilde{\Theta}) \ln \frac{P(\mathbf{x}|\tilde{\Theta})}{P(\mathbf{x}|\Theta)} d\mathbf{x} = \int_{\mathbf{x}} \sum_{i=1}^m \tilde{w}_i P(\mathbf{x}|\tilde{\Theta}_i) \ln \frac{\sum_{i=1}^m \tilde{w}_i P(\mathbf{x}|\tilde{\Theta}_i)}{\sum_{i=1}^m w_i P(\mathbf{x}|\Theta_i)} d\mathbf{x}. \quad (3)$$

Ideally, we would like to use the above distance function in  $V^t$  to give us an update of  $\tilde{\Theta}$  in terms of  $\Theta$ . However, there isn't a closed form expression for Eq. (3). Although the relative entropy between two Gaussians is a convex function in their parameters, the relative entropy between two Gaussian mixtures is non-convex. Thus, the loss function  $V^t(\tilde{\Theta})$  may have multiple minima, making the problem of finding  $\arg \min_{\tilde{\Theta}} V^t(\tilde{\Theta})$  difficult.

In order to sidestep this problem we use the log-sum inequality [5] to obtain an upper bound for the distance function  $\Delta(\tilde{\Theta}, \Theta)$ . We denote this upper bound as  $\hat{\Delta}(\tilde{\Theta}, \Theta)$ .

$$\begin{aligned} \Delta(\tilde{\Theta}, \Theta) &\leq \hat{\Delta}(\tilde{\Theta}, \Theta) \stackrel{\text{def}}{=} \int_{\mathbf{x}} \sum_{i=1}^m (\mathbf{x}, i|\tilde{\Theta}) \ln \frac{P(\mathbf{x}, i|\tilde{\Theta})}{P(\mathbf{x}, i|\Theta)} d\mathbf{x} = \int_{\mathbf{x}} \sum_{i=1}^m \tilde{w}_i P(\mathbf{x}|\tilde{\Theta}_i) \ln \frac{\tilde{w}_i P(\mathbf{x}|\tilde{\Theta}_i)}{w_i P(\mathbf{x}|\Theta_i)} d\mathbf{x} \\ &= \sum_{i=1}^m \tilde{w}_i \ln \frac{\tilde{w}_i}{w_i} + \sum_{i=1}^m \tilde{w}_i \int_{\mathbf{x}} P(\mathbf{x}|\tilde{\Theta}_i) \ln \frac{P(\mathbf{x}|\tilde{\Theta}_i)}{P(\mathbf{x}|\Theta_i)} d\mathbf{x} = \sum_{i=1}^m \tilde{w}_i \ln \frac{\tilde{w}_i}{w_i} + \sum_{i=1}^m \tilde{w}_i \Delta(\tilde{\Theta}_i, \Theta_i). \quad (4) \end{aligned}$$

We call the new distance function  $\widehat{\Delta}(\widetilde{\Theta}, \Theta)$  the *joint-entropy distance*. Note that in this distance the parameters of  $\widetilde{w}_i$  and  $w_i$  are “coupled” in the sense that it is a convex combination of the distances  $\Delta(\widetilde{\Theta}_i, \Theta_i)$ . In particular,  $\widehat{\Delta}(\widetilde{\Theta}, \Theta)$  as a function of the parameters  $\widetilde{w}_i, \widetilde{\mu}_i, \widetilde{\mathbf{C}}_i$  does not remain constant any more when the parameters of the individual Gaussians are permuted. Furthermore,  $\widehat{\Delta}(\widetilde{\Theta}, \Theta)$  is also sufficiently convex so that finding the minimizer of  $V^t$  is possible (see below).

## 5 The updates

We are now ready to derive the new parameter estimation scheme. This is done by setting the partial derivatives of  $V^t$ , with respect to  $\widetilde{\Theta}$ , to 0. That is, our problem consists of solving the following equations

$$\frac{\partial \widehat{\Delta}(\widetilde{\Theta}, \Theta)}{\partial \widetilde{w}_i} - \frac{\eta}{|S|} \frac{\partial \ln P(S|\Theta)}{\partial w_i} + \lambda = 0, \quad \frac{\partial \widehat{\Delta}(\widetilde{\Theta}, \Theta)}{\partial \widetilde{\mu}_i} - \frac{\eta}{|S|} \frac{\partial \ln P(S|\Theta)}{\partial \mu_i} = 0, \quad \frac{\partial \widehat{\Delta}(\widetilde{\Theta}, \Theta)}{\partial \widetilde{\mathbf{C}}_i} - \frac{\eta}{|S|} \frac{\partial \ln P(S|\Theta)}{\partial \mathbf{C}_i} = 0.$$

We now use the fact that  $\mathbf{C}_i$  and thus  $\mathbf{C}_i^{-1}$  is symmetric. The derivatives of  $\widehat{\Delta}(\widetilde{\Theta}, \Theta)$ , as defined by Eq. (4) and Eq. (2), with respect to  $\widetilde{w}_i, \widetilde{\mu}_i$  and  $\widetilde{\mathbf{C}}_i$ , are

$$\frac{\partial \widehat{\Delta}(\widetilde{\Theta}, \Theta)}{\partial \widetilde{w}_i} = \ln \frac{\widetilde{w}_i}{w_i} + 1 + \frac{1}{2} \ln \frac{|\mathbf{C}_i|}{|\widetilde{\mathbf{C}}_i|} - \frac{d}{2} + \frac{1}{2} \text{tr}(\mathbf{C}_i^{-1} \widetilde{\mathbf{C}}_i) + \frac{1}{2} (\widetilde{\mu}_i - \mu)^T \mathbf{C}_i^{-1} (\widetilde{\mu}_i - \mu) \quad (5)$$

$$\frac{\partial \widehat{\Delta}(\widetilde{\Theta}, \Theta)}{\partial \widetilde{\mu}_i} = \widetilde{w}_i \mathbf{C}_i^{-1} (\widetilde{\mu}_i - \mu) \quad (6)$$

$$\frac{\partial \widehat{\Delta}(\widetilde{\Theta}, \Theta)}{\partial \widetilde{\mathbf{C}}_i} = \frac{1}{2} \widetilde{w}_i (-\widetilde{\mathbf{C}}_i^{-1} + \mathbf{C}_i^{-1}). \quad (7)$$

To simplify the notation throughout the rest of the paper we define the following variables

$$\beta_i(\mathbf{x}) \stackrel{\text{def}}{=} \frac{P(\mathbf{x}|\Theta_i)}{P(\mathbf{x}|\Theta)} \quad \text{and} \quad \alpha_i(\mathbf{x}) \stackrel{\text{def}}{=} \frac{w_i P(\mathbf{x}|\Theta_i)}{P(\mathbf{x}|\Theta)} = P(i|\mathbf{x}, \Theta_i) = w_i \beta_i(\mathbf{x}).$$

The partial derivatives of the log-likelihood are computed similarly:

$$\frac{\partial \ln P(S|\Theta)}{\partial w_i} = \sum_{\mathbf{x} \in S} \frac{P(\mathbf{x}|\Theta_i)}{P(\mathbf{x}|\Theta)} = \sum_{\mathbf{x} \in S} \beta_i(\mathbf{x}) \quad (8)$$

$$\frac{\partial \ln P(S|\Theta)}{\partial \mu_i} = \sum_{\mathbf{x} \in S} \frac{w_i P(\mathbf{x}|\Theta_i)}{P(\mathbf{x}|\Theta)} \mathbf{C}_i^{-1} (\mathbf{x} - \mu_i) = \sum_{\mathbf{x} \in S} \alpha_i(\mathbf{x}) \mathbf{C}_i^{-1} (\mathbf{x} - \mu_i) \quad (9)$$

$$\begin{aligned} \frac{\partial \ln P(S|\Theta)}{\partial \mathbf{C}_i} &= -\frac{1}{2} \sum_{\mathbf{x} \in S} \frac{w_i P(\mathbf{x}|\Theta_i)}{P(\mathbf{x}|\Theta)} (\mathbf{C}_i^{-1} - \mathbf{C}_i^{-1} (\mathbf{x} - \mu_i) (\mathbf{x} - \mu_i)^T \mathbf{C}_i^{-1}) \\ &= -\frac{1}{2} \sum_{\mathbf{x} \in S} \alpha_i(\mathbf{x}) (\mathbf{C}_i^{-1} - \mathbf{C}_i^{-1} (\mathbf{x} - \mu_i) (\mathbf{x} - \mu_i)^T \mathbf{C}_i^{-1}). \end{aligned} \quad (10)$$

We now need to decide on an order for updating the parameter classes  $w_i, \mu_i$ , and  $\mathbf{C}_i$ . We use the same order that EM uses, namely,  $w_i$ , then  $\mu_i$ , and finally,  $\mathbf{C}_i$ . (After doing one pass over all three groups we start again using the same order.) Using this order results in a simplified set of equations as several terms in Eq. (5) cancel out. Denote the size of the sample by  $N = |S|$ . We now need to sum the derivatives from Eq. (5) and Eq. (8) while using the fact that the Lagrange multiplier  $\lambda$  simply assures that the new weight  $\widetilde{w}_i$  sum to one. By setting the result to zero, we get that

$$w_i \leftarrow \frac{w_i \exp\left(-\frac{\eta}{N} \sum_{\mathbf{x} \in S} \beta_i(\mathbf{x})\right)}{\sum_{j=1}^m w_j \exp\left(-\frac{\eta}{N} \sum_{\mathbf{x} \in S} \beta_j(\mathbf{x})\right)}. \quad (11)$$

Similarly, we sum Eq. (6) and Eq. (9), set the result to zero, and get that

$$\mu_i \leftarrow \mu_i + \frac{\eta}{N} \sum_{\mathbf{x} \in S} \beta_i(\mathbf{x}) (\mathbf{x} - \mu_i). \quad (12)$$

Finally, we do the same for  $\mathbf{C}_i$ . We sum Eq. (7) and Eq. (10) using the newly obtained  $\mu_i$ ,

$$\mathbf{C}_i^{-1} \leftarrow \mathbf{C}_i^{-1} + \frac{\eta}{N} \sum_{\mathbf{x} \in S} \beta_i(\mathbf{x}) (\mathbf{C}_i^{-1} - \mathbf{C}_i^{-1}(\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^T \mathbf{C}_i^{-1}). \quad (13)$$

We call the new iterative parameter estimation procedure the joint-entropy (JE) update. To summarize, the JE update is composed of the following alternating steps: We first calculate for each observation  $\mathbf{x}$  the value  $\beta_i(\mathbf{x}) = P(\mathbf{x}|\Theta_i)/P(\mathbf{x}|\Theta)$  and then update the parameters as given by Eq. (11), Eq. (12), and Eq. (13). The JE update and EM differ in several aspects. First, EM uses a simple update for the mixture weights  $\mathbf{w}$ . Second, EM uses the expectations (with respect to the current parameters) of the sufficient statistics [6] for  $\mu_i$  and  $\mathbf{C}_i$  to find new sets of mean vectors and covariance matrices. The JE uses a (slightly different) weighted average of the observation and, in addition, it adds the old parameters. The learning rate  $\eta$  determines the proportion to be used in summing the old parameters and the newly estimated parameters. Last, EM estimates the covariance matrices  $\mathbf{C}_i$  whereas the new update estimates the *inverses*,  $\mathbf{C}_i^{-1}$ , of these matrices. Thus, it is potentially be more stable numerically in cases where the covariance matrices have small condition number.

To obtain an on-line procedure we need to update the parameters after each new observation at a time. That is, rather than summing over all  $\mathbf{x} \in S$ , for a new observation  $\mathbf{x}_t$ , we update the parameters and get a new set of parameters  $\Theta^{t+1}$  using the current parameters  $\Theta^t$ . The new parameters are then used for inducing the likelihood of the next observation  $\mathbf{x}_{t+1}$ . The on-line parameter estimation procedure is composed of the following steps:

1. Set:  $\beta_i(\mathbf{x}_t) = \frac{P(\mathbf{x}_t|\Theta_i)}{P(\mathbf{x}_t|\Theta)}$ .
2. Parameter updates:
  - (a)  $w_i \leftarrow w_i \exp(-\eta_t \beta_i(\mathbf{x}_t)) / \sum_{j=1}^m w_j \exp(-\eta_t \beta_j(\mathbf{x}_t))$
  - (b)  $\mu_i \leftarrow \mu_i + \eta_t \beta_i(\mathbf{x}_t) (\mathbf{x}_t - \mu_i)$
  - (c)  $\mathbf{C}_i^{-1} \leftarrow \mathbf{C}_i^{-1} + \eta_t \beta_i(\mathbf{x}_t) (\mathbf{C}_i^{-1} - \mathbf{C}_i^{-1}(\mathbf{x}_t - \mu_i)(\mathbf{x}_t - \mu_i)^T \mathbf{C}_i^{-1})$ .

To guarantee convergence of the on-line update one should use a diminishing learning rate, that is  $\eta_t \rightarrow 0$  as  $t \rightarrow \infty$  (for further motivation and proof technique see [18]).

## 6 Experiments

We conducted numerous experiments with the new update. Due to the lack of space we describe here only a few. In the first experiment we compared the JE update and EM in batch settings. We generated data from Gaussian mixture distributions with varying number of components ( $m = 2$  to 100) and dimensions ( $d = 2$  to 20). Due to the lack of space we describe here results obtained from only one setting. In this setting the examples were generated by a mixture of 5 components with  $\mathbf{w} = (0.4, 0.3, 0.2, 0.05, 0.05)$ . The mean vectors were the 5 standard unit vectors in the Euclidean space  $\mathbb{R}^d$  and we set all of covariances matrices to the identity matrix. We generated 1000 examples. We then run EM and the JE update with different learning rates ( $\eta = 1.9, 1.5, 1.1, 1.05$ ). To make sure that all the runs will end in the same local maximum we fist performed three EM iterations. The results are shown on the left hand side of Figure 1. In this setting, the JE update with high learning rates achieves much faster convergence than EM. We would like to note that this behavior is by no means esoteric – most of our experiments data yielded similar results.

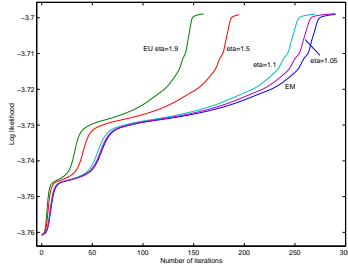


Figure 1: Comparison of EM and the JE update.

Classifier	Error, %
Decision tree, CART	17.0
Decision tree, C4.5	16.0
Best 2 layer neural network	6.6
Special architecture 5 layer network	5.1
Support Vector Machine with 6 degree polynomials	4.2
JE update (Batch mode)	7.8
JE update (On-line mode)	1.8

Figure 2: Performance of various classifiers on USPS data set.

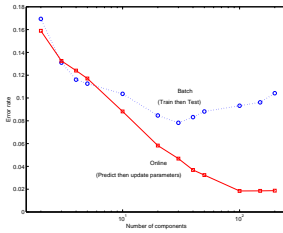


Figure 3: Left: Comparison of the performance of the on-line and batch versions of the JE update on the US Postal Service digit recognition problem. Right: The mean vectors of the first mixture component representing the digit 7 after 10, 100, 200, 500, 1000 and 8000 on-line updates.

**Experiments with US Postal Service data set.** The US Postal Service (USPS) data set is a collection of digits collected from actual handwritten mailings. The problem of automatic digit recognition using this data set was by several researchers (see [4] and the references therein). This data set contains 7,300 training digits and 2,000 digits for testing. Each digit is represented by a  $16 \times 16$  pixel image that can take 256 different values. Error rates of various classifiers for this data set are given in Figure 3 (taken from [4]). In our experiments with this data set we used the same pre-processing that was used in [4].

This data set is rather small and the error rates of the various learning algorithms tested on this data set are typically high compared to other digit data sets. One reason for the relatively poor performance of all classifiers is due to a significant disparity in the shape of the digits constituting the training and test sets [14]. While the training set was cleaned by removing digits that were “chopped” by a segmentation algorithm, the test set was kept untouched. Thus, there are shapes that occur rather frequently in the test data but do not resemble any of the images in the training set.

We used the JE update to estimate the parameters of Gaussian mixtures with varying number of components, from  $m = 2$  to 200. We separated the data into 10 subsets representing the different digits and built a mixture model for each subset. The digit were represented as real valued feature vectors in  $\mathbb{R}^{256}$ . We constrained the covariance matrices to be block diagonal where each block is of dimension  $5 \times 5$ . The mean vectors were initialized by setting them to randomly chosen images from the training data.

We first used the JE update in batch mode to estimates the parameters of the ten Gaussian mixtures using only the training data. We run the batch algorithm until it converged to a local maximum for all the digits. We then classified digits from the test set using the trained model. We assigned a digit to a class *iff* its probability with respect to the mixture model representing the class was the largest among all models. Formally, let  $\Theta^l = \{\Theta_i^l\}_{i=1}^m$  be the set of parameters for the  $l$ th digit. Then, we classified a test image  $\mathbf{x}$  as belonging to the  $l$ th class *iff* for  $l' \neq l$ ,  $P(\mathbf{x}|\Theta^l) > P(\mathbf{x}|\Theta^{l'})$ . Using the same initialization, we also

used the on-line parameter estimation algorithm to predict the class membership of each test digit. In this case, we updated the parameters after each prediction, both on the training *and* test data. (However, note that each digit was used only once for updating the parameters.) We used the diminishing learning rate  $\eta_t = 1/t$  for  $t > 100$  and a fixed learning rate  $\eta = 0.01$  for  $t \leq 100$ . We tested the performance by calculating the classification error on the test data. A comparison of the error rates for the batch and on-line modes is given on the left hand side of Figure 3.

It is clear from the figure that the on-line update is superior to the batch update when the latter is not trained on the test data. Furthermore, the on-line update is more robust in the sense that it does not overfit when mixture models with a large number of components are fitted to the data. We also would like to note that the best error rate achieved using a mixture model in an on-line mode is 1.8%, which is lower than the error rates of previously studied classifiers. Furthermore, it is actually better than human performance (in the sense defined in [3]). These results provide some empirical evidence that the JE update, when used in the on-line mode, is able to track and efficiently approximate the distribution of a time varying source. Note, however, that the classification results do not imply that the JE update is superior to other parameter estimation algorithms. The low error rates are mainly due to the fact that we allowed the algorithm to update the parameters on both the training and test data.

We conclude this section with an illustration of the behavior the JE update in an on-line mode on the USPS data set. On the right hand side of Figure 3 we show the mean vectors of the first component of a 30 component mixture model representing the digit 7. We show the mean vectors after 10, 100, 200, 500, 1000 and 8000 on-line updates. It is apparent from the figure that image representing the mean-vector becomes less noisy as more data is used to update the parameters. Indeed, the pixel image after 8000 on-line updates is very clear and easily recognizable as the digit 7.

**Acknowledgments** Thanks do Duncan Herring for careful proof reading.

## References

- [1] E. Bauer, D. Koller, and Y. Singer. Update rules for parameter estimation in Bayesian networks. In *Proceedings of the 13th Annual Conference on Uncertainty in AI*, pages 3–13, 1997.
- [2] C.M. Bishop. *Neural Networks and Pattern Recognition*. Oxford University Press, 1995.
- [3] J. Bromley and E. Sackinger. Neural-network and  $k$ -nearest neighbor classifiers. Technical Report Technical report 11359-010819-16TM, AT&T, 1991.
- [4] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
- [5] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley, 1991.
- [6] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B39:1–38, 1977.
- [7] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. Wiley, 1973.
- [8] D. P. Helmbold, J. Kivinen, and M.K. Warmuth. Worst-case loss bounds for sigmoided neurons. In *Advances in Neural Information Processing Systems 7*, pages 309–315, 1995.
- [9] D.P. Helmbold, R.E. Schapire, Y.Singer, and M.K. Warmuth. A comparison of new and old algorithms for a mixture estimation problem. *Machine Learning*, pages 97–124, 1997.
- [10] A. Jagota and M.K. Warmuth. Continuous and discrete time nonlinear gradient descent: relative loss bounds and convergenc. In *International Symposium on Artificial Intelligence and Mathematic*, 1998.
- [11] M.I. Jordan and L. Xu. Convergence results for the EM approach to mixtures of experts architectures. *Neural Networks*, 8:1409–1431, 1995.
- [12] J. Kivinen and M.K. Warmuth. Additive versus exponentiated gradient updates for linear prediction. *Information and Computation*, 132(1):1–64, January 1997.
- [13] J. Kivinen and M.K. Warmuth. Relative loss bounds for multidimensional regression problems. In *Advances in Neural Information Processing Systems 10*, 1997.
- [14] Y. LeCun, L. D. Jackel, L. Bottou, A. Brunot, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, U. A. Muller, E. Sackinger, P. Simard, and V Vapnik. Comparison of learning algorithms for handwritten digit recognition. In *International Conference on Artificial Neural Networks*, pages 53–60. World Scientific, 1995.
- [15] X. L. Meng and D. B. Rubin. Recent extensions of the EM algorithm (with discussion). In J.M. Bernardo, J.O. Berger, A.P. Dawid, and A.F.M. Smith, editors, *Bayesian Statistics, 4*. Oxford: Clarendon Press, 1992.
- [16] B.C. Peters and H.F. Walker. An iterative procedure for obtaining maximum-likelihood estimates of the parameters for a mixture of normal distributions. *SIAM Journal of Applied Mathematics*, 35:362–378, 1978.
- [17] R.A. Redner and H.F. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2), 1984.
- [18] D.M. Titterington, A.F.M. Smith, and U.E. Makov. *Statistical Analysis of Finite Mixture Distributions*. Wiley, 1985.
- [19] L. Xu and M.I. Jordan. On convergence properties of the EM algorithm for Gaussian mixtures. *Neural Computation*, 8:129–151, 1996.