Recursive Unsupervised Learning of Finite Mixture Models

Zoran Zivkovic, *Member*, *IEEE Computer Society*,
and Ferdinand van der Heijden, *Member*,
IEEE Computer Society

Abstract—There are two open problems when finite mixture densities are used to model multivariate data: the selection of the number of components and the initialization. In this paper, we propose an online (recursive) algorithm that estimates the parameters of the mixture and that simultaneously selects the number of components. The new algorithm starts with a large number of randomly initialized components. A prior is used as a bias for maximally structured models. A stochastic approximation recursive learning algorithm is proposed to search for the maximum a posteriori (MAP) solution and to discard the irrelevant components.

Index Terms—Online (recursive) estimation, unsupervised learning, finite mixtures, model selection, EM-algorithm.

1 Introduction

Finite mixture probability density models have been analyzed many times and used extensively for modeling multivariate data [16], [8]. In [3] and [6], an efficient heuristic was used to simultaneously estimate the parameters of a mixture and select the appropriate number of its components. The idea is to start with a large number of components and introduce a prior to express our preference for compact models. During some iterative search procedure for the MAP solution, the prior drives the irrelevant components to extinction. The "entropic-prior" from [3] leads to a MAP estimate that minimizes the entropy and, hence, leads to a compact model. The Dirichlet prior from [6] gives a solution that is related to model selection using the "Minimum Message Length" (MML) criterion [20].

This paper is inspired by the aforementioned papers [3], [6]. Our contribution is in developing an online version which is potentially very useful in many situations since it is highly memory and time efficient. We use a stochastic approximation procedure to estimate the parameters of the mixture recursively. More on the behavior of approximate recursive equations can be found in [13], [5], [15]. We propose a way to include the suggested prior from [6] in the recursive equations. This enables the online selection of the number of components of the mixture. We show that the new algorithm can reach solutions similar to those obtained by batch algorithms.

In Sections 2 and 3 of the paper, we introduce the notation and discuss some standard problems associated with finite mixture fitting. In Section 4, we describe the mentioned heuristic that enables us to estimate the parameters of the mixture and to simultaneously select the number of its components. Further, in Section 5, we develop an online version. The final practical algorithm we used in our experiments is described in Section 6. In Section 7, we demonstrate how the new algorithm performs for a number of standard problems and compare it to some batch algorithms.

2 Parameter Estimation

A mixture density with M components for a d-dimensional random variable $\vec{x}$ is given by:

$$p(\vec{x};\vec{\theta}) = \sum_{m=1}^{M} \pi_m p_m(\vec{x};\vec{\theta}_m), \quad \text{with} \quad \sum_{m=1}^{M} \pi_m = 1, \tag{1}$$

where $\vec{\theta} = \{\pi_1, .., \pi_M, \vec{\theta}_1, .., \vec{\theta}_M\}$ are the parameters. The number of parameter depends on the number of components M and the notation $\vec{\theta}(M)$ will be used to stress this when needed. The mth component of the mixture is denoted by $p_m(\vec{x};\vec{\theta}_m)$ and $\vec{\theta}_m$ are its parameters. The mixing weights denoted by π_m are nonnegative and add up to one.

Given a set of t data samples $\mathcal{X} = \{\vec{x}^{(1)}, \dots, \vec{x}^{(t)}\}$ the maximum likelihood (ML) estimate of the parameter values is:

$$\widehat{\vec{\theta}} = \arg\max_{\vec{\theta}}(\log p(\mathcal{X};\vec{\theta})).$$

The Expectation Maximization (EM) algorithm [4] is commonly used to search for the solution. The EM algorithm is an iterative procedure that searches for a local maximum of the log-likelihood function. In order to apply the EM algorithm, we need to introduce for each $\vec{x}$ a discrete unobserved indicator vector $\vec{y} = [y_1 \dots y_M]^T$. The indicator vector specifies (by means of position coding) the mixture component from which the observation $\vec{x}$ is drawn. The new joint density function can be written as a product:

$$p(\vec{x},\vec{y};\vec{\theta}) = p(\vec{y};\pi_1, .., \pi_M)p(\vec{x}|\vec{y};\vec{\theta}_1, .., \vec{\theta}_M) = \prod_{m=1}^{M} \pi_m^{y_m} p_m(\vec{x};\vec{\theta}_m)^{y_m},$$

where exactly one of the y_m from $\vec{y}$ can be equal to 1 and the others are zero. The indicators $\vec{y}$ have a multinomial distribution defined by the mixing weights $\pi_1, .., \pi_M$. The EM algorithm starts with some initial parameter estimate $\widehat{\vec{\theta}}_{(0)}$. If we denote the set of unobserved data by $\mathcal{Y} = \{\vec{y}^{(1)}, \dots, \vec{y}^{(t)}\}$ the estimate $\widehat{\vec{\theta}}_{(k)}$ from the kth iteration of the EM algorithm is obtained using the previous estimate $\widehat{\vec{\theta}}_{(k-1)}$:

$$\begin{aligned} \text{E step}: \quad & Q(\vec{\theta},\widehat{\vec{\theta}}_{(k-1)}) = E_{\mathcal{Y}}(\log p(\mathcal{X},\mathcal{Y};\vec{\theta})|\mathcal{X},\widehat{\vec{\theta}}_{(k-1)}) = \\ & \sum_{\text{all possible } \mathcal{Y}} p(\mathcal{Y}|\mathcal{X},\widehat{\vec{\theta}}_{(k-1)}) \log p(\mathcal{X},\mathcal{Y};\vec{\theta}) \\ \text{M step}: \quad & \widehat{\vec{\theta}}_{(k)} = \arg\max_{\vec{\theta}}(Q(\vec{\theta},\widehat{\vec{\theta}}_{(k-1)})). \end{aligned} \tag{2}$$

The attractiveness of the EM algorithm is that it is easy to implement and it converges to a local maximum of the log-likelihood function. However, one of the serious limitations of the EM algorithm is that it can end up in a poor local maximum if not properly initialized. The selection of the initial parameter values is still an open question that was studied many times. Some recent efforts were reported in [3], [6], [17], [18], [19].

3 Model Selection

Note that, in order to use the EM algorithm, we need to know the appropriate number of components M. Too many components lead to "overfitting" and too few to "underfitting." Choosing an appropriate number of components is important. Sometimes, for example, the appropriate number of components can reveal some important existing underlying structure that characterizes the data.

- *Z. Zivkovic is with the Informatics Institute, University of Amsterdam, Kruislaan 403, 1098SJ Amsterdam, The Netherlands. E-mail: zivkovic@science.uva.nl.*
- *F. van der Heijden is with the Laboratory for Measurement and Instrumentation, University of Twente, PO Box 217, 7500AE Enschede, The Netherlands. E-mail: f.vanderheijden@utwente.nl.*

Manuscript received 18 Nov. 2002; revised 24 June 2003; accepted 3 Nov. 2003.
Recommended for acceptance by Y. Amit.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 117789.

Full Bayesian approaches sample from the full a posteriori distribution with the number of components M considered unknown. This is possible using Markov chain Monte Carlo methods as reported in [11], [10]. However, these methods are still far too computationally demanding. Most of the practical model selection techniques are based on maximizing the following type of criteria:

$$J(M, \vec{\theta}(M)) = \log p(\mathcal{X}; \vec{\theta}(M)) - P(M). \tag{3}$$

Here, $\log p(\mathcal{X}; \vec{\theta}(M))$ is the log-likelihood for the available data. This part can be maximized using the EM. However, introducing more mixture components always increases the log-likelihood. The balance is achieved by introducing $P(M)$ that penalizes complex solutions. Some examples of such criteria are the Akaike Information Criterion [1], the Bayesian Inference Criterion [14], the Minimum Description Length [12], the Minimum Message Length (MML) [20], etc. For a detailed review, see, for example, [8].

4 Solution Using MAP Estimation

The standard procedure for selecting M is the following: Find the ML estimate for different M-s and choose the M that maximizes (3). Suppose that we introduce a prior $p(\vec{\theta}(M))$ for the mixture parameters that penalizes complex solutions in a similar way as $P(M)$ from (3). Instead of (3), we could use:

$$\log p(\mathcal{X}; \vec{\theta}(M)) + \log p(\vec{\theta}(M)). \tag{4}$$

As in [6] and [3], we use the simplest prior choice, the prior only on the mixing weights π_m-s. For example, the Dirichlet prior (see [7], chapter 16) for the mixing weights is given by:

$$p(\vec{\theta}(M)) \propto \exp \sum_{m=1}^{M} c_m \log \pi_m = \prod_{m=1}^{M} \pi_m^{c_m}. \tag{5}$$

The procedure is then as follows: We start with a large number of randomly initialized components M and search for the MAP solution using some iterative procedure, for example, the EM algorithm. The prior drives the irrelevant components to extinction. In this way, while searching for the MAP solution, the number of components M is reduced until the balance is achieved.

It can be shown that the standard MML model selection criterion can be approximated by the Dirichlet prior with the coefficients c_m equal to $-N/2$, where N presents the number of parameters per component of the mixture. See [6] for details. The parameters c_m have a meaningful interpretation. For a multinomial distribution, the c_m presents the prior evidence (in the MAP sense) for the class m (number of samples a priori belonging to that class). Negative prior evidence means that we will accept that the class m exists only if there is enough evidence from the data for the existence of this class. If there are many parameters per component, we will need many data samples to estimate them. In this sense, the presented linear connection between the c_m and N seems very logical. The procedure from [6] starts with all the π_m-s equal to $1/M$. Although there is no proof of optimality, it seems reasonable to discard the component m when its weight π_m becomes negative. This also ensures that the mixing weights stay nonnegative.

The "entropic prior" from [3] has a similar form: $p(\vec{\theta}(M)) \propto \exp(-\beta H(\pi_1, \ldots, \pi_M))$, where $H(\pi_1, \ldots, \pi_M) = -\sum_{m=1}^{M} \pi_m \log \pi_m$ is the entropy measure for the underlying multinomial distribution and β is a parameter. We use the mentioned Dirichlet prior because it leads to a closed form solution.

5 Recursive (Online) Solution

For the ML estimate, the following holds: $\frac{\partial}{\partial \vec{\theta}} \log p(\mathcal{X}; \widehat{\vec{\theta}}) = 0$. The mixing weights are constrained to sum up to 1. We take this into account by introducing the Lagrange multiplier λ and get: $\frac{\partial}{\partial \hat{\pi}_m} \left(\log p(\mathcal{X}; \widehat{\vec{\theta}}) + \lambda (\sum_{m=1}^{M} \hat{\pi}_m - 1) \right) = 0$. From here, after getting rid of λ, it follows that the ML estimate for t data samples should satisfy $\hat{\pi}_m^{(t)} = \frac{1}{t} \sum_{i=1}^{t} o_m^{(t)}(\vec{x}^{(i)})$ with the "ownerships" defined as:

$$o_m^{(t)}(\vec{x}) = \hat{\pi}_m^{(t)} p_m(\vec{x}; \widehat{\vec{\theta}}_m^{(t)}) / p(\vec{x}; \widehat{\vec{\theta}}^{(t)}). \tag{6}$$

Similarly, for the MAP solution, we have $\frac{\partial}{\partial \hat{\pi}_m} (\log p(\mathcal{X}; \widehat{\vec{\theta}}) + \log p(\widehat{\vec{\theta}}) + \lambda (\sum_{m=1}^{M} \hat{\pi}_m - 1)) = 0$, where $p(\widehat{\vec{\theta}})$ is the mentioned Dirichlet prior (5). For t data samples, we get:

$$\hat{\pi}_m^{(t)} = \frac{1}{K} \left(\sum_{i=1}^{t} o_m^{(t)}(\vec{x}^{(i)}) - c \right), \tag{7}$$

where $K = \sum_{m=1}^{M} (\sum_{i=1}^{t} o_m^{(t)}(\vec{x}^{(i)}) - c) = t - Mc$ (since $\sum_{m=1}^{M} o_m^{(t)} = 1$). The parameters of the prior are $c_m = -c$ (and $c = N/2$ as mentioned before). We rewrite (7) as:

$$\hat{\pi}_m^{(t)} = \frac{\hat{\Pi}_m - c/t}{1 - Mc/t}, \tag{8}$$

where $\hat{\Pi}_m = \frac{1}{t} \sum_{i=1}^{t} o_m^{(t)}(\vec{x}^{(i)})$ is the mentioned ML estimate and the bias from the prior is introduced through c/t. The bias decreases for larger data sets (larger t). However, if a small bias is acceptable we can keep it constant by fixing c/t to $c_T = c/T$ with some large T. This means that the bias will always be the same as if it would have been for a data set with T samples. If we assume that the parameter estimates do not change much when a new sample $\vec{x}^{(t+1)}$ is added and, therefore, $o_m^{(t+1)}(\vec{x}^{(i)})$ can be approximated by $o_m^{(t)}(\vec{x}^{(i)})$ that uses the previous parameter estimates, we get the following well behaved and easy to use recursive update equation:

$$\hat{\pi}_m^{(t+1)} = \hat{\pi}_m^{(t)} + (1+t)^{-1} \left(\frac{o_m^{(t)}(\vec{x}^{(t+1)})}{1 - Mc_T} - \hat{\pi}_m^{(t)} \right) - (1+t)^{-1} \frac{c_T}{1 - Mc_T}. \tag{9}$$

Here, T should be sufficiently large to make sure that $Mc_T < 1$. We start with initial $\hat{\pi}_m^{(0)} = 1/M$ and discard the $m\text{th}$ component when $\hat{\pi}_m^{(t+1)} < 0$. Note that the straightforward recursive version of (7) given by: $\hat{\pi}_m^{(t+1)} = \hat{\pi}_m^{(t)} + (1 + t - Mc)^{-1} (o_m^{(t)}(\vec{x}^{(t+1)}) - \hat{\pi}_m^{(t)})$, is not very useful. For small t, the update is negative and the weights for the components with high $o_m^{(t)}(\vec{x}^{(t+1)})$ are decreased instead of increased. In order to avoid the negative update, we could start with a larger value for t, but then we cancel out the influence of the prior. This motivates the important choice we made to fix the influence of the prior.

The most commonly used mixture is the Gaussian mixture. A mixture component $p_m(\vec{x}; \vec{\theta}_m) = \mathcal{N}(\vec{x}; \vec{\mu}_m, C_m)$ has its mean $\vec{\mu}_m$ and its covariance matrix C_m as the parameters. The prior has influence only on the mixing weights and we can use the recursive equations:

$$\widehat{\vec{\mu}}_m^{(t+1)} = \widehat{\vec{\mu}}_m^{(t)} + (t+1)^{-1} \frac{o_m^{(t)}(\vec{x}^{(t+1)})}{\hat{\pi}_m^{(t)}} (\vec{x}^{(t+1)} - \widehat{\vec{\mu}}_m^{(t)}) \tag{10}$$

$$\hat{C}_m^{(t+1)} = \hat{C}_m^{(t)} + (t+1)^{-1} \frac{o_m^{(t)}(\vec{x}^{(t+1)})}{\hat{\pi}_m^{(t)}} \Big((\vec{x}^{(t+1)} - \widehat{\vec{\mu}}_m^{(t)})(\vec{x}^{(t+1)} - \widehat{\vec{\mu}}_m^{(t)})^T - \hat{C}_m^{(t)} \Big) \tag{11}$$

from [15] for the rest of the parameters.

6 A Simple Practical Algorithm

For an online procedure, it is reasonable to fix the influence of the new samples by replacing the term $(1+t)^{-1}$ from the recursive update equations (9), (10), and (11) by $\alpha = 1/T$. There are also some practical reasons for using a fixed small constant α. It reduces the problems with instability of the equations for small t. Furthermore, a fixed α helps in forgetting the out-of-date statistics (random initialization and component deletion) more rapidly. It is equivalent to introducing an exponentially decaying envelope: $\alpha(1-\alpha)^{t-i}$ is applied to the influence of the sample $\vec{x}^{(t-i)}$.

For the sake of clarity, we present here the whole algorithm we used in our experiments. We start with a large number of components M and with a random initialization of the parameters (see next section for an example). We have $c_T = \alpha N/2$. Furthermore, we use Gaussian mixture components with full covariance matrices. Therefore, if the data is d-dimensional, we have $N = d + d(d+1)/2$ (the number of parameters for a Gaussian with a full covariance matrix). The online algorithm is then given by:

- **Input:** new data sample $\vec{x}^{(t+1)}$, current parameter estimates $\widehat{\vec{\theta}}^{(t)}$.
- Calculate "ownerships:" $o_m^{(t)}(\vec{x}^{(t+1)}) = \hat{\pi}_m^{(t)} p_m(\vec{x}^{(t+1)}; \widehat{\vec{\theta}}_m^{(t)}) / p(\vec{x}^{(t+1)}; \widehat{\vec{\theta}}^{(t)})$.
- Update mixture weights: $\hat{\pi}_m^{(t+1)} = \hat{\pi}_m^{(t)} + \alpha(\frac{o_m^{(t)}(\vec{x}^{(t+1)})}{1-Mc_T} - \hat{\pi}_m^{(t)}) - \alpha\frac{c_T}{1-Mc_T}$.
- Check if there are irrelevant components: if $\hat{\pi}_m^{(t+1)} < 0$, discard the component m, set $M = M-1$ and renormalize the remaining mixing weights.
- Update the rest of the parameters:
  - $\widehat{\vec{\mu}}_m^{(t+1)} = \widehat{\vec{\mu}}_m^{(t)} + w\vec{\delta}$ (where $w = \alpha\frac{o_m^{(t)}(\vec{x}^{(t+1)})}{\hat{\pi}_m^{(t)}}$ and $\vec{\delta} = \vec{x}^{(t+1)}$ - $\widehat{\vec{\mu}}_m^{(t)}$).
  - $\hat{C}_m^{(t+1)} = \hat{C}_m^{(t)} + w(\vec{\delta}\vec{\delta}^T - \hat{C}_m^{(t)})$ (tip: limit the update speed $w = \min(20\alpha, w)$).
- **Output:** new parameter estimates $\widehat{\vec{\theta}}^{(t+1)}$.

This simple algorithm can be implemented in only a few lines of code. The recommended upper limit 20α for w simply means that the updating speed is limited for the covariance matrices of the components representing less than 5 percent of the data. This was necessary since $\vec{\delta}\vec{\delta}^T$ is a singular matrix and the covariance matrices may become singular if updated too fast.

7 Experiments

In this section, we demonstrate the algorithm performance on a few standard problems. We show summary results from 100 trials for each data set. For the real-world data sets, we randomly sample from the data to generate longer sequences needed for our sequential algorithm. First, for each of the problems, we present in Fig. 1 how the selected number of components of the mixture was changing when new samples are sequentially added. The number of components that was finally selected is presented in the form of a histogram for the 100 trials. In Fig. 2, we present a comparison with some batch algorithms and study the influence of the parameter α.

The random initialization of the parameters is the same as in [6]. The means $\widehat{\vec{\mu}}_m^{(0)}$ of the mixture components are initialized by some randomly chosen data points. The initial covariance matrices are a fraction (1/10 here) of the mean global diagonal covariance matrix:

$$C_m^{(0)} = \frac{1}{10d} trace\left(\frac{1}{n}\sum_{i=1}^{n}(\vec{x}^{(i)} - \widehat{\vec{\mu}})(\vec{x}^{(i)} - \widehat{\vec{\mu}})^T\right)I,$$

where $\widehat{\vec{\mu}} = \frac{1}{n}\sum_{i=1}^{n}\vec{x}^{(i)}$ is the global mean of the data and I is the identity matrix with proper dimensions. We used the first $n = 100$ samples (it is also possible to estimate this initial covariance matrix recursively). Finally, we set the initial mixing weights to $\hat{\pi}_m^{(0)} = 1/M$. The initial number of components M should be large enough so that the initialization reasonably covers the data. We used here the same initial number of components as in [6].

7.1 The "Three Gaussians" Data Set

First, we analyze a Gaussian mixture with mixing weights $\pi_1 = \pi_2 = \pi_3 = 1/3$, means $\mu_1 = [0\ -2]^T$, $\mu_2 = [0\ 0]^T$, $\mu_3 = [02]^T$, and covariance matrices

$$C_1 = C_2 = C_3 = \begin{bmatrix} 2 & 0 \\ 0 & 0.2 \end{bmatrix}.$$

A modified version of the EM called "DAEM" from [17] was able to find the correct solution using a "bad" initialization. For a data set with 900 samples, they needed more than 200 iterations to get close to the solution. Here, we start with $M = 30$ mixture components. With random initialization, we performed 100 trials and the new algorithm was always able to find the correct solution while simultaneously estimating the parameters of the mixture and selecting the number of components. A similar batch algorithm from [6] needs about 200 iterations to identify the three components (on a data set with 900 samples). From the plot in Fig. 1, we see that already after 9,000 samples the new algorithm is usually able to identify the three components. The computation costs for 9,000 samples are approximately the same as for only 10 iterations of the EM algorithm on a data set with 900 samples. Consequently, the new algorithm for this data set is about 20 times faster in finding a similar solution (a typical solution is presented in Fig. 1 by the "$\sigma = 2$ contours" of the Guassian components). In [9], some approximate recursive versions of the EM algorithm were compared to the standard EM algorithm and it was shown that the recursive versions are usually faster. This is in correspondence with our results. Empirically, we decided that 50 samples per class are enough and used $\alpha = 1/150$.

7.2 The "Iris" Data Set

We disregard the class information from the well-known 3-class, 4-dimensional "Iris" data set [2]. From the 100 trials, the clusters were properly identified 81 times. This shows that the order in which the data is presented can influence the recursive solution. The data set had only 150 samples (50 per class) that were repeated many times. We expect that the algorithm would perform better with more data samples. We used $\alpha = 1/150$. The typical solution in Fig. 1 is presented by projecting the 4-dimensional data to the first two principal components.

7.3 The "Shrinking Spiral" Data Set

This data set presents a 1-dimensional manifold ("shrinking spiral") in the three dimensions with added noise: $\vec{x} = [(13 - 0.5t)\cos t\ \ (0.5t - 13)\sin t\ \ t] + \vec{n}$, with $t \sim Uniform[0, 4\pi]$ and the noise $\vec{n} \sim \mathcal{N}(0, I)$. The modified EM called "SMEM" from [18] was reported to be able to fit a 10 component mixture in about 350 iterations. The batch algorithm from [6] is fitting the mixture and selecting 11, 12, or 13 components using typically 300 to 400 iterations for a 900 samples data set. From the graph in Fig. 1, it is clear that we achieve similar results, but much faster. About 18,000 samples was enough to arrive at a similar solution. Consequently, again, the new algorithm is about 20 times faster. There are no clusters in this data set. The fixed α has as the effect that the influence of the old data is downweighted by the exponential decaying envelope $\alpha(1-\alpha)^{t-k}$ (for $k < t$). For comparison with the other algorithms that used 900 samples, we limited the influence of the older samples to 5 percent of the influence of the current sample by $\alpha = -\log(0.05)/900$. In Fig. 1, we present a typical solution by

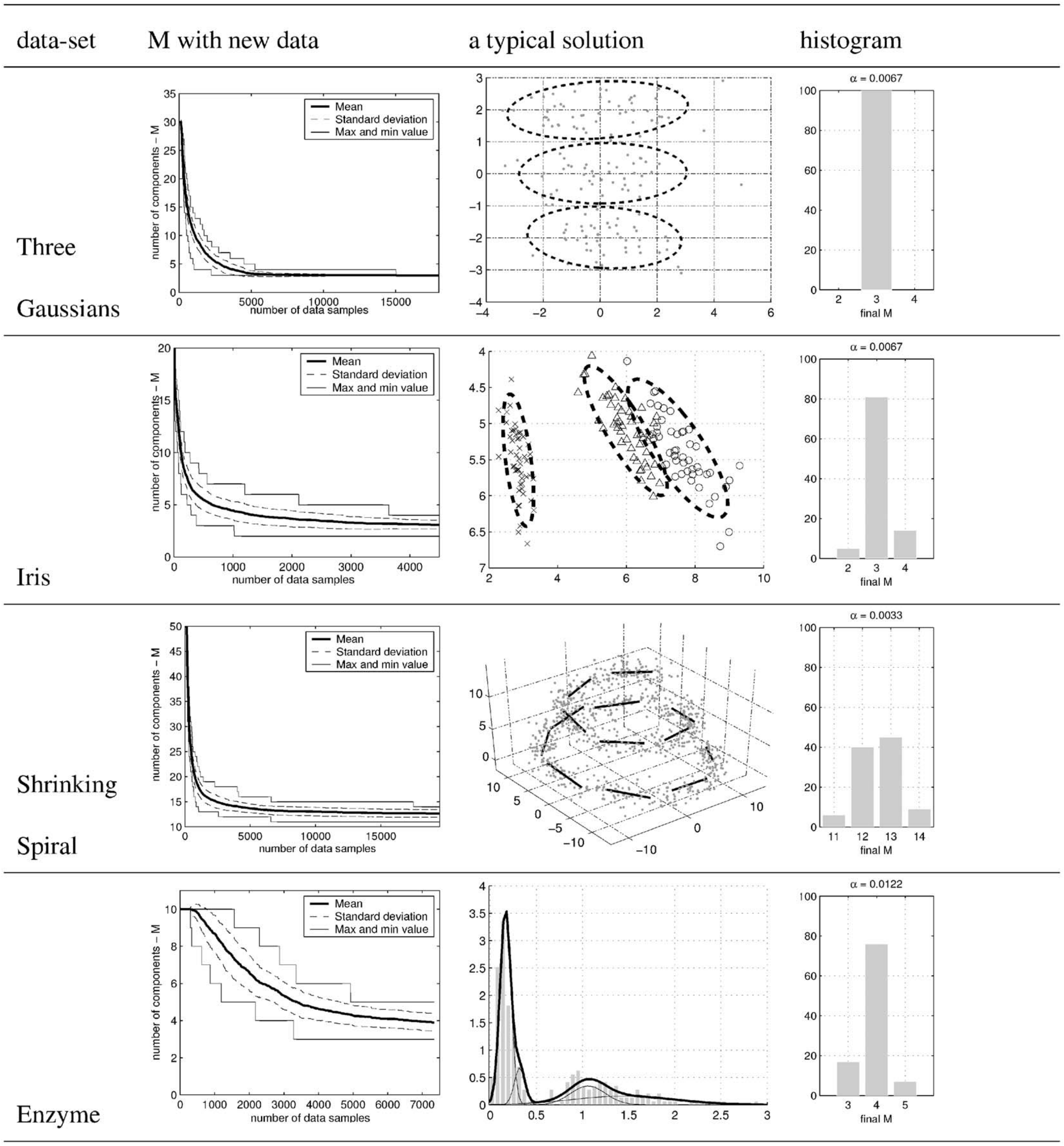


Fig. 1. Model selection results for a few standard problems (summary from 100 trials).

showing for each component the eigenvector corresponding to the largest eigenvalue of the covariance matrix.

7.4 The "Enzyme" Data Set

The 1-dimensional "Enzyme" data set has 245 data samples. It was shown in [11] using the MCMC that the number of components supported by the data is most likely four, but two and three are also good choices. Our algorithm arrived at similar solutions. In a similar way as before, we used $\alpha = -\log(0.05)/245$.

7.5 Comparison with Some Batch Algorithms

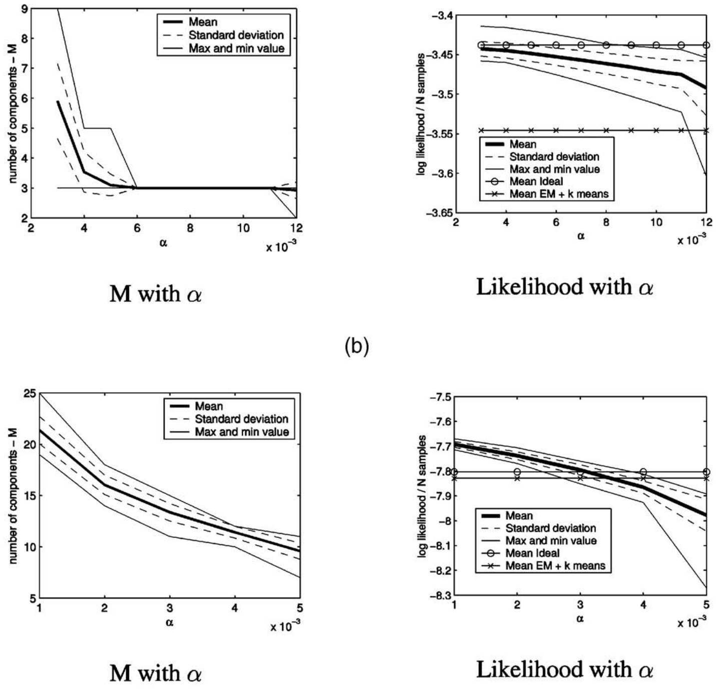
The following standard batch methods were considered for comparison: the EM algorithm initialized using the result from k-means clustering; the SMEM method [18]; the greedy EM method [19] that starts with a single component and adds new ones—reported to be faster than the elaborate SMEM. We used 900 samples for the "Three Gaussians" and the "Shrinking Spiral" data sets. The batch algorithms assume a known number of components: three for the "Three Gaussians" and the "Iris" data, 13 for the "Shrinking Spiral," and four for the "Enzyme" data set. Our new unsupervised recursive algorithm RUEM has selected on average approximately the same number of components for the chosen α. All the iterative batch algorithms in our experiments stop if the change in the log-likelihood is less than 10^{-5}. The results are presented in Fig. 2a. The best likelihood and the lowest standard deviation are reported in bold. We also added the ideal ML result obtained using a carefully initialized EM. For the "Iris" data, the EM was initialized using the means and the covariances of the three classes. However, the solution where the two close clusters are modeled using one component was better in terms of likelihood. This "wrong" solution was found occasionally by some of the algorithms. The results from the RUEM are biased. Furthermore, the parameter α is controlling the speed of updating the parameters and, therefore, also the effective amount of data that is considered. Therefore, we present also the results "polished" by additionally applying the EM algorithm and using the same sample size for the batch algorithms. The RUEM results and the "polished" results are better or similar to the batch results. We also observe that the greedy EM algorithm has problems with the "Iris" and the "Shrinking spiral" data.

7.6 The Influence of the Parameter α

In Figs. 2b and 2c, we show the influence of the parameter α on the selected number of components. We also plot the log-likelihood

	Three Gaussians	Iris	Shrinking spiral	Enzyme
Ideal	$-3.44(0.01)$	$-1.20(0.00)$	$-7.80(0.01)$	$-0.17(0.00)$
RUEM polished	$\mathbf{-3.44(0.01)}$	$\mathbf{-1.18}(0.12)$	$\mathbf{-7.80(0.02)}$	$-0.19(0.02)$
RUEM	$-3.46(0.01)$ $(\alpha = 0.0067)$	$-1.19(0.12)$ $(\alpha = 0.0067)$	$-7.81(0.02)$ $(\alpha = 0.0033)$	$-0.20(0.02)$ $(\alpha = 0.0122)$
SMEM	$-3.47(0.06)$	$-1.19(0.04)$	$-7.83(0.02)$	$-0.18(0.01)$
Greedy EM	$-3.46(0.05)$	$-1.24(\mathbf{0.02})$	$-7.88(0.04)$	$\mathbf{-0.17(0.00)}$
EM + k means	$-3.55(0.06)$	$-1.24(0.07)$	$-7.83(0.03)$	$-0.20(0.01)$

(a)



M with α Likelihood with α

(b)

M with α Likelihood with α

(c)

Fig. 2. Comparison with some standard batch methods and some experiments to study the influence of the parameter α (summary from 100 trials). (a) The mean and the standard deviation (between the brackets) of the log-likelihood over the number of samples—calculated on new test data for the synthetic data sets. (b) The "three Gaussians" data set—the influence of α. (c) The "shrinking spiral" data set—the influence of α.

per sample for different values of α. For the "Three Gaussians" data set, there is a range of values for α where the same number of components is finally selected. We can expect similar results for other data sets where the clusters are well described by the mixture components and the components are well separated. For the "Shrinking Spiral" data set, there are no clear clusters and the number of selected components slowly declines with larger α. Similarly, the log-likelihood also decreases with α. For comparison, we plotted also some log-likelihood values from some batch algorithms (see previous section). The new unsupervised procedure simultaneously estimates parameters and selects a compact model. We observe from the log-likelihood values that for a wide range of values for α, we get a good representation of the data with a compact model. The graphs for the real-world data "Iris" and "Enzyme" are not included since they look similar to the graphs for the "Shrinking Spiral" data.

8 Discussion and Conclusions

We have proposed an online method for fitting mixture models which relies on a "description-length reducing" prior and a MAP estimation procedure for selecting a compact model. The experimental results indicated that the recursive algorithm was able to solve difficult problems and to obtain similar solutions as other elaborate batch algorithms. However, the theoretical support for the finally selected number of components is questionable. Some arguments in favor of the "entropic prior" and its connections to other model selection criteria are given in [3]. The Dirichlet prior we used is related to the well founded MML principle, but it can be perhaps better viewed as an efficient heuristics. Therefore, if selecting the correct model is critical, we suggest, as in the much slower batch version [6], to perform an additional check with some standard model selection criterion (full MML for example). An additional problem when compared to the batch version [6] is the introduced parameter α that balances the influence of the data against the influence of the prior. This is similar to the parameter β from the "entropic prior" (λ in [3]). Some experiments were performed to show the influence of the parameter α. The parameter $\alpha = 1/T$ is related to the number of data samples T that are considered and some heuristic choices were used in the previous section. If selecting the correct number of components is not critical the new recursive procedure is highly time and memory efficient and potentially very useful to give a "quick" up-to-date compact description of the data.

Acknowledgments

This work was done while Z. Zivkovic was with the Laboratory for Measurement an Instrumentation, University of Twente, Enschede, The Netherlands.

References

[1] H. Akaike, "A New Look at the Statistical Model Identification," *IEEE Trans. Automatic Control,* vol. 19, no. 6, pp. 716-723, 1974.

[2] E. Anderson, "The Irises of the Gaspe Peninsula," *Bull. of the Am. Iris Soc.,* vol. 59, 1935.

[3] M.E. Brand, "Structure Learning in Conditional Probability Models via an Entropic Prior and Parameter Extinction," *Neural Computation J.,* vol. 11, no. 5, pp. 1155-1182, 1999.

[4] A.P. Dempster, N. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Royal Statistical Soc., Series B (Methodological),* vol. 1, no. 39, pp. 1-38, 1977.

[5] V. Fabian, "On Asymptotically Efficient Recursive Estimation," *Annals of Statistics,* vol. 6, pp. 854-866, 1978.

[6] M. Figueiredo and A.K. Jain, "Unsupervised Learning of Finite Mixture Models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 3, pp. 381-396, Mar. 2002.

[7] A. Gelman, J.B. Carlin, H.S. Stern, and D.B. Rubin, *Bayesian Data Analysis.* Chapman and Hall, 1995.

[8] G. McLachlan and D. Peel, *Finite Mixture Models.* John Wiley and Sons, 2000.

[9] R.M. Neal and G.E. Hinton, "A New View of the EM Algorithm that Justifies Incremental, Sparse and Other Variants," *Learning in Graphical Models,* pp. 355-368, M.I. Jordan ed., 1998.

[10] C. Rasmussen, "The Infinite Gaussian Mixture Model," *Advances in Neural Information Processing Systems,* vol. 12, pp. 554-560, 2000.

[11] S. Richardson and P. Green, "On Bayesian Analysis of Mixture Models with Unknown Number of Components," *J. Royal Statistical Soc., Series B (Methodological),* vol. 59, no. 4, pp. 731-792, 1997.

[12] J. Rissansen, "Stochastic Complexity," *J. Royal Statistical Soc., Series B (Methodological),* vol. 49, no. 3, pp. 223-239, 1987.

[13] J. Sacks, "Asymptotic Distribution of Stochastic Approximation Procedures," *Annals of Math. Statistics,* vol. 29, pp. 373-405, 1958.

[14] G. Schwarz, "Estimating the Dimension of a Model," *Annals of Statistics,* vol. 6, no. 2, pp. 461-464, 1978.

[15] D.M. Titterington, "Recursive Parameter Estimation Using Incomplete Data," *J. Royal Statistical Soc., Series B (Methodological),* vol. 2, no. 46, pp. 257-267, 1984.

[16] D.M. Titterington, A.F.M. Smith, and U.E. Makov, *Statistical Analysis of Finite Mixture Distributions.* John Wiley and Sons, 1985.

[17] N. Ueda and R. Nakano, "Deterministic Annealing EM Algorithm," *Neural Networks,* vol. 11, pp. 271-282, 1998.

[18] N. Ueda, R. Nakano, Z. Ghahramani, and G.E. Hinton, "SMEM Algorithm for Mixture Models," *Neural Computation,* vol. 12, no. 9, pp. 2109-2128, 2000.

[19] J.J. Verbeek, N. Vlassis, and B. Krose, "Efficient Greedy Learning of Gaussian Mixture Models," *Neural Computation,* vol. 15, no. 1, 2003.

[20] C. Wallace and P. Freeman, "Estimation and Inference by Compact Coding," *J. Royal Statistical Soc., Series B (Methodological),* vol. 49, no. 3, pp. 240-265, 1987.