A study of online and blockwise updating of the EM algorithm for Gaussian mixtures

Jochen Einbeck[1], Daniel Bonetti[2]

[1] Department of Mathematical Sciences, Durham University, UK
[2] Universidade de São Paulo, São Carlos, SP, Brazil

E-mail for correspondence: `daniel.bonetti@gmail.com`

Abstract: A variant of the EM algorithm for the estimation of multivariate Gaussian mixtures, which allows for online as well as blockwise updating of sequentially obtained parameter estimates, is investigated. Several different update schemes are considered and compared, and the benefits of artificially performing EM in batches, even though all data are available, are discussed.

Keywords: Multivariate Gaussian mixtures; Maximum Likelihood; Incremental EM.

1 Background

Consider multivariate data $y_i = (y_{i1}, \ldots, y_{ip})^T$, $i = 1, \ldots, n$ sampled independently from some population consisting of several latent groups or subpopulations. Data of this type are conveniently modelled through Gaussian mixture models

$$f(y_i|\theta) = \sum_{k=1}^{K} \pi_k f(y_i|\theta_k)$$

where $f(y_i|\theta_k) = (2\pi)^{-p/2}|\Sigma_k|^{-1/2} \exp\left\{-\frac{1}{2}(y_i - \mu_k)^T \Sigma_k^{-1}(y_i - \mu_k)\right\}$, $\theta_k = \{\pi_k, \mu_k, \Sigma_k\}$, and $\theta = \{\theta_1, \ldots, \theta_K\}$, with the restriction $\pi_K = 1 - \sum_{k=1}^{K-1} \pi_k$. Assume a given set of starting values, say $\hat{\theta}_{(0)}$. The EM algorithm iterates between the Expectation (E) –step, updating the posterior probabilities $w_{ik} = P(\text{obs. } i \text{ belongs to comp. } k)$ for given $\hat{\theta}_k$,

$$w_{ik} \equiv w(y_i, \hat{\theta}_k) = \frac{\pi_k f(y_i|\hat{\theta}_k)}{\sum_{k=1}^{K} \pi_k f(y_i|\hat{\theta}_k)}, \tag{1}$$

and the Maximization (M) –step, where the parameter estimates are updated for given w_{ik} as

$$\begin{aligned}\hat{\pi}_k &= \frac{1}{n}\sum_{i=1}^{n} w_{ik}; \qquad \hat{\mu}_k = \frac{\sum_{i=1}^{n} w_{ik} y_i}{\sum_{i=1}^{n} w_{ik}}; \\ \hat{\Sigma}_k &= \frac{\sum_{i=1}^{n} w_{ik}(y_i - \hat{\mu}_k)(y_i - \hat{\mu}_k)^T}{\sum_{i=1}^{n} w_{ik}}\end{aligned}$$

Let $\hat{\theta}_{(j)}$ denote the value of $\hat{\theta}$ after j–th iteration and $l_j = \sum_{i=1}^{n} \log f(y_i|\hat{\theta}_{(j)})$ the corresponding log-likelihood. Convergence is established at j–th iteration if the difference $l_j - l_{j-1}$ falls below a small threshold (which we take to be 0.001). An important feature of this methodology should be highlighted: After convergence of the EM algorithm we have not only obtained the estimate $\hat{\theta}$, but are also able to assign each observation i to a class $c_i \in \{1, \ldots, K\}$ via the classification rule

$$c_i = \arg\max_k w_{ik}.$$

For details on how the described routine relates to the original formulation of the EM algorithm, we refer to Aitkin et al. (2009), Section 7.

2 Updating the EM algorithm

The methodology described in the previous Section implies the assumption that the full set of responses $\boldsymbol{y} = (y_1^T, \ldots, y_n^T)^T$ has been observed before the EM algorithm is run. Unfortunately, for new data y_{n+j}, $j \geq 1$, the fitted model does not provide information on the class membership c_{n+j}, since the $w_{n+j,k}$ are unknown. Traditionally, if n' new data have been observed, the way forward would have been to re–fit the entire model, using data $y_1, \ldots, y_n, y_{n+1}, \ldots y_{n+n'}$. This is clearly inefficient, and may be computationally expensive especially if n and/or p are large. However, the EM algorithm can easily be adapted to become an update algorithm, which we outline as follows. Assume that, using a batch of size n, the EM algorithm as outlined in Section 1 has been executed, yielding estimates $\hat{\theta}$ and a weight matrix $W = (w_{ik})_{1\leq i\leq n, 1\leq k\leq K}$. Now, a new batch n' has been made available. Via (1), one can compute new membership probabilities

$$w'_{n+j,k} = w(y_{n+j}, \hat{\theta}_k)$$

using the already computed $\hat{\theta}$, which we again summarize in a matrix $W' = (w_{n+j,k})_{1\leq j\leq n', 1\leq k\leq K}$. Then we stack W' underneath W, which we denote as $W \cup W'$. At this stage we have several options.

(i) We use $W \cup W'$ to run a single M-step, using all $n + n'$ data, leading to an updated $\hat{\theta}$. We refer to this option as **one–step– update**. Note that, under this scenario, the matrix W has not got updated, so the posterior probabilities of the original data have not benefited from the new information.

(ii) After running (i), we can run one additional E-step (which then will update $W \cup W'$) and one additional M-step. We call this the **two–step update**.

(iii) One can now repeat step (ii) a further number of times, and if we do this until convergence, we call this a **converged update**.

Now, any of (i), (ii) or (iii) lead to an update of $\hat{\theta}$ for the single batch n'. After reception of the next batch, say n'', again any of (i) to (iii) needs to be executed, and this will be repeated until all batches have been received. Unlike other recently proposed approaches to 'incremental EM' (such as Zhang and Scordilis, 2009), where the focus is on approximative updating to gain computational speed, any combination of (i), (ii) and (iii) will give an 'exact' update, hence enabling to find the MLE of the full data at any stage, if only (iii) is executed after the last update.

3 Online updating

Options (i) and (ii) are especially attractive in an online scenario, where new observations i come in at certain timepoints t_i, and one wishes to continuously update the current parameter estimates and weight matrix. Clearly the initial batch needs to contain at least as many observations as parameters in θ, that is mathematically $n \geq K(1+3p/2+p^2/2)-1$ (but in practice larger values will be required). For all further batches one would have $1 = n' = n'' = \ldots$.
We illustrate the performance of the online update schemes (i) and (ii) through a simple simulation. We generated a random dataset with two Gaussian clusters containing a total of 100 points. The result of applying usual EM onto this data set is provided in Figure 1, with the log-likelihood provided by the dashed curve in the middle panel. Next, 60 from these 100 points were randomly selected to drop out the dataset, leaving us with a batch of size $n = 40$. After running EM until convergence for this reduced batch, which required 20 iterations, the previously removed points were returned one-by-one to the dataset, each followed by procedures (i) and (ii), respectively. After all data points have been included, we run step (iii) (this is the part after the vertical line in Figure 1 (middle)).
Note that the higher likelihoods (as compared to usual EM) obtained for small iteration numbers are an artefact of the smaller sample sizes used for these. The key observations from Fig. 1 (middle) are: (a) During the execution of the online–updating, scheme (i) leads to worse solutions than scheme (ii), and needs a longer time to recover once one goes into mode

(iii). (b) After final convergence, the log-likelihood $\ell(\hat{\theta})$ is the same for all approaches.
Conceptually, it is clear that (b) must be true: If we have achieved some estimate $\hat{\theta}$ through repeated online updating, and from this moment on we decide to fill in the remaining data, then we effectively carry out usual EM but with starting value $\hat{\theta}_{(0)} = \hat{\theta}$. The MLE of this problem is the same as for the full data set.

4 Blockwise updating

Even though the MLE is in principle the same, *there may be a difference in whether or not the maximum is actually found*, that is, whether or not the EM algorithm gets trapped in a local maximum on its way to the MLE. In other words, there is the intriguing possibility that, by *artificial blockwise splitting and updating* data sets which were in principle fully available, one is able to overcome local maxima (through the influx of fresh data), in which traditional EM gets trapped. Indeed, one can find scenarios where this is the case. An exemplary graphical illustration of the log-likelihoods obtained in a simulation involving 1200 samples (split in three batches) and $K = 8$ mixture components is provided in Figure 1 (right).

References

Aitkin, M, Hinde, J., Francis, B., and Darnell, R. (2009). *Statistical Modelling in R.* Oxford: Oxford University Press.

Zhang, Y., and Scordilis, M.S. (2007). Effective online unsupervised adaptation of Gaussian mixture models and its application to speech classification. *Pattern Recognition Letters*, **29**, 735–774.

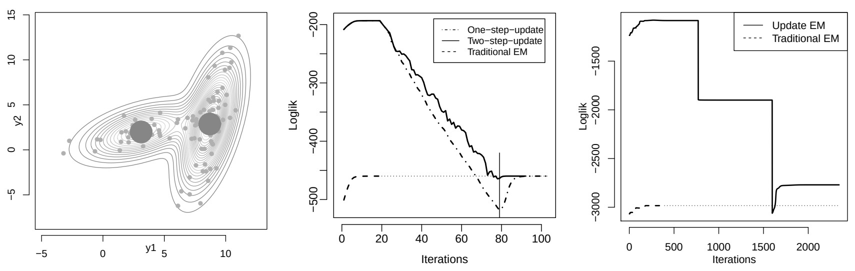

FIGURE 1. Left: Result of EM for bivariate data with two clusters; Middle: Comparing traditional EM with one– and two–step updates; Right: Comparing traditional EM with blockwise updating (for a more complex data set involving 8 clusters).