Mini-batch learning of exponential family finite mixture models

Hien D Nguyen, Florence Forbes, Geoffrey J Mclachlan

▶ **To cite this version:**

Hien D Nguyen, Florence Forbes, Geoffrey J Mclachlan. Mini-batch learning of exponential family finite mixture models. Statistics and Computing, 2020, 30, pp.731-748. 10.1007/s11222-019-09919-4 . hal-02415068v2

HAL Id: hal-02415068
https://hal.science/hal-02415068v2

Submitted on 26 Mar 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Mini-batch learning of exponential family finite mixture models

Hien D. Nguyen[1*], Florence Forbes[2], and Geoffrey J. McLachlan[3]

September 6, 2019

[1]Department of Mathematics and Statistics, La Trobe University, Melbourne, Victoria, Australia.

[2]Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP[†], LJK, 38000 Grenoble, France. [†]Institute of Engineering Univ. Grenoble Alpes.

[3]School of Mathematics and Physics, University of Queensland, St. Lucia, Brisbane, Australia.

[*]Corresponding author: Hien Nguyen (Email: h.nguyen5@latrobe.edu.edu.au).

Abstract

Mini-batch algorithms have become increasingly popular due to the requirement for solving optimization problems, based on large-scale data sets. Using an existing online expectation--maximization (EM) algorithm framework, we demonstrate how mini-batch (MB) algorithms may be constructed, and propose a scheme for the stochastic stabilization of the constructed mini-batch algorithms. Theoretical results regarding the convergence of the mini-batch EM algorithms are presented. We then demonstrate how the mini-batch framework may be applied

to conduct maximum likelihood (ML) estimation of mixtures of exponential family distributions, with emphasis on ML estimation for mixtures of normal distributions. Via a simulation study, we demonstrate that the mini-batch algorithm for mixtures of normal distributions can outperform the standard EM algorithm. Further evidence of the performance of the mini-batch framework is provided via an application to the famous MNIST data set.

Key words: expectation--maximization algorithm, exponential family distributions, finite mixture models, mini-batch algorithm, normal mixture models, online algorithm

1 Introduction

The exponential family of distributions is an important class of probabilistic models with numerous applications in statistics and machine learning. The exponential family contains many of the most commonly used univariate distributions, including the Bernoulli, binomial, gamma, geometric, inverse Gaussian, logarithmic normal, Poisson, and Rayleigh distributions, as well as multivariate distributions such as the Dirichlet, multinomial, multivariate normal, von Mises, and Wishart distributions. See Forbes et al. (2011, Ch. 18), DasGupta (2011, Ch. 18), and Amari (2016, Ch. 2).

Let $\boldsymbol{Y}^\top = (Y_1, \ldots, Y_d)$ be a random variable (with realization $\boldsymbol{y}$) on the support $\mathbb{Y} \subseteq \mathbb{R}^d$ ($d \in \mathbb{N}$) , arising from a data generating process (DGP) with probability density/mass function (PDF/PMF) $f(\boldsymbol{y}; \boldsymbol{\theta})$ that is characterized by some parameter vector $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p$ ($p \in \mathbb{N}$). We say that the distribution that characterizes the DGP of $\boldsymbol{Y}$ is in the exponential family class, if the PDF/PMF can be written in the form

$$f(\boldsymbol{y};\boldsymbol{\theta}) = h(\boldsymbol{y}) \exp\left\{ [\boldsymbol{s}(\boldsymbol{y})]^\top \boldsymbol{\phi}(\boldsymbol{\theta}) - \psi(\boldsymbol{\theta}) \right\}, \tag{1}$$

where $\boldsymbol{s}(\cdot)$ and $\boldsymbol{\phi}(\cdot)$ are p-dimensional vector functions, and $h(\cdot)$ and $\psi(\cdot)$ are 1-dimensional functions of $\boldsymbol{y}$ and $\boldsymbol{\theta}$, respectively. If the dimensionality of $\boldsymbol{s}(\cdot)$ and $\boldsymbol{\phi}(\cdot)$ is less than p, then we say that the distribution that characterizes the DGP of $\boldsymbol{Y}$ is in the curved exponential class.

Let $Z \in [g]$ ($[g] = \{1, \ldots, g\}$; $g \in \mathbb{N}$) be a latent random variable, and write $\boldsymbol{X}^\top = \left(\boldsymbol{Y}^\top, Z\right)$. Suppose that the PDF/PMF of $\{\boldsymbol{Y} = \boldsymbol{y} | Z = z\}$ can be written as $f(\boldsymbol{y};\boldsymbol{\omega}_z)$, for each $z \in [g]$. If we assume that $\mathbb{P}(Z = z) = \pi_z > 0$, such that $\sum_{z=1}^{g} \pi_z = 1$, then we can write the marginal PDF/PMF of $\boldsymbol{Y}$ in the form

$$f(\boldsymbol{y};\boldsymbol{\theta}) = \sum_{z=1}^{g} \pi_z f(\boldsymbol{y};\boldsymbol{\omega}_z), \tag{2}$$

where we put the unique elements of π_z and $\boldsymbol{\omega}_z$ into $\boldsymbol{\theta}$. We call $f(\boldsymbol{y};\boldsymbol{\theta})$ the g-component finite mixture PDF, and we call $f(\boldsymbol{y};\boldsymbol{\omega}_z)$ the zth component PDF, characterized by the parameter vector $\boldsymbol{\omega}_z \in \Omega$, where Ω is some subset of a real product space. We also say that the elements π_z are prior probabilities, corresponding to the respective component.

The most common finite mixtures models are mixtures of normal distributions, which were popularization by Pearson (1894), and have been prolifically used by numerous prior authors (cf. McLachlan et al., 2019). The g-component d-dimensional normal mixture model has PDF of the form

$$f(\boldsymbol{y};\boldsymbol{\theta}) = \sum_{z=1}^{g} \pi_z \varphi(\boldsymbol{y};\boldsymbol{\mu}_z,\boldsymbol{\Sigma}_z), \tag{3}$$

where the normal PDFs

$$\varphi\left(\boldsymbol{y};\boldsymbol{\mu}_z,\boldsymbol{\Sigma}_z\right)=\left|2\pi\boldsymbol{\Sigma}_z\right|^{-1/2}\exp\left[-\frac{1}{2}\left(\boldsymbol{y}-\boldsymbol{\mu}_z\right)^\top\boldsymbol{\Sigma}_z^{-1}\left(\boldsymbol{y}-\boldsymbol{\mu}_z\right)\right], \tag{4}$$

replace the component densities $f\left(\boldsymbol{y};\boldsymbol{\omega}_z\right)$, in (2). Each component PDF (4) is parameterized by a mean vector $\boldsymbol{\mu}_z\in\mathbb{R}^d$ and a positive-definite symmetric covariance matrix $\boldsymbol{\Sigma}_z\in\mathbb{R}^{d\times d}$. We then put each π_z, $\boldsymbol{\mu}_z$, and $\boldsymbol{\Sigma}_z$ into the vector $\boldsymbol{\theta}$.

As earlier noted, the normal distribution is a member of the exponential family, and thus (4) can be written in form (1). This can be observed by putting the unique elements of $\boldsymbol{\mu}_z$ and $\boldsymbol{\Sigma}_z$ into $\boldsymbol{\omega}_z$, and writing $\varphi\left(\boldsymbol{y};\boldsymbol{\mu}_z,\boldsymbol{\Sigma}_z\right)=f\left(\boldsymbol{y};\boldsymbol{\omega}_z\right)$ in form (1), with mappings

$$h\left(\boldsymbol{y}\right)=\left(2\pi\right)^{-d/2},\ \boldsymbol{s}\left(\boldsymbol{y}\right)=\begin{bmatrix}\boldsymbol{y}\\ \operatorname{vec}(\boldsymbol{y}\boldsymbol{y}^\top)\end{bmatrix},\ \boldsymbol{\phi}\left(\boldsymbol{\omega}_z\right)=\begin{bmatrix}\boldsymbol{\Sigma}_z^{-1}\boldsymbol{\mu}_z\\ -\frac{1}{2}\operatorname{vec}(\boldsymbol{\Sigma}_z^{-1})\end{bmatrix},\text{ and} \tag{5}$$

$$\psi\left(\boldsymbol{\omega}_z\right)=\frac{1}{2}\boldsymbol{\mu}_z^\top\boldsymbol{\Sigma}_z^{-1}\boldsymbol{\mu}_z+\frac{1}{2}\log\left|\boldsymbol{\Sigma}_z\right|. \tag{6}$$

When conducting data analysis using a normal mixture model, one generally observes an independent and identically (IID) sequence of $n\in\mathbb{N}$ observations $\{\boldsymbol{Y}_i\}_{i=1}^n$, arising from a DGP that is hypothesized to be characterized by a PDF of the form (3), with unknown parameter vector $\boldsymbol{\theta}=\boldsymbol{\theta}_0$. The inferential task is to estimate $\boldsymbol{\theta}_0$ via some estimator that is computed from $\{\boldsymbol{Y}_i\}_{i=1}^n$. The most common computational approach to obtaining an estimator of $\boldsymbol{\theta}_0$ is via maximum likelihood (ML) estimation, using the expectation--maximization algorithm (EM; Dempster et al., 1977). See McLachlan & Peel (2000, Ch. 3.2) for a description of the normal mixture EM algorithm. Generally, when g, d, and n are of small to moderate size, the conventional EM approach is feasible, and is able to perform the task of ML estimation in a timely manner. Unfortunately, due to its high memory demands, costly matrix operations (Nguyen & McLachlan, 2015), and slow

convergence rates (McLachlan & Krishnan, 2008, Sec. 3.9), the conventional EM algorithm is not suited for the computational demands of analyzing increasingly large data sets, such as those that could be considered as *big data* in volumes such as Buhlmann et al. (2016), Han et al. (2017), and Hardle et al. (2018).

Over the years, numerous algorithms have been proposed as means to alleviate the computational demands of the EM algorithm for normal mixture models. Some of such approaches include the component-wise algorithm of Celeux et al. (2001), the greedy algorithm of Vlassis & Likas (2002), the sparse and incremental *kd*-tree algorithm of Ng & McLachlan (2004), the subspace projection algorithm of Bouveyron et al. (2007), and the matrix operations-free algorithm of Nguyen & McLachlan (2015).

There has been a recent resurgence in stochastic approximation algorithms, of the Robbins & Monro (1951) and Kiefer & Wolfowitz (1952) type, developed for the purpose of solving computationally challenging optimization problems, such as the ML estimation of normal mixture models. A good review of the current literature can be found in Chau & Fu (2015). NaÃ¯ve and direct applications of the stochastic approximation approach to mixture model estimation can be found in Liang & Zhang (2008), Zhang & Liang (2008), and Nguyen & Jones (2018).

Following a remark from Cappé & Moulines (2009) regarding the possible extensions of the online EM algorithm, we propose mini-batch EM algorithms for the ML estimation of exponential family mixture models. These algorithms include a number of variants, among which are update truncation variants that had not been made explicit, before. Using the theorems from Cappé & Moulines (2009), we state results regarding the convergence of our algorithms. We then specialize our attention to the important case of normal mixture models, and demonstrate that the required assumptions for convergence are met in such a scenario.

A thorough numerical study is conducted in order to assess the performance of our normal mixture mini-batch algorithms. Comparisons are drawn between our algorithms and the usual batch EM algorithm for ML estimation of normal mixture models. We show that our mini-batch algorithms can be applied to very large data sets by demonstrating its applicability to the ML estimation of normal mixture models on the famous MNIST data of LeCun et al. (1998).

References regarding mixtures of exponential family distributions and EM-type stochastic approximation algorithms, and comments regarding some recent related literature are relegated to the Supplementary Materials, in the interest of brevity. Additional remarks, numerical results, and derivations are also included in these Supplementary Materials in order to provide extra context and further demonstrate the capabilities of the described framework. These demonstrations include the derivation of mini-batch EM algorithms for mixtures of exponential and Poisson distributions. The Supplementary Materials can be found at `https://github.com/hiendn/StoEMMIX/blob/master/Manuscript_files/SupplementaryMaterials.pdf`.

The remainder of the paper is organized as follows. In Section 2, we present the general results of Cappé & Moulines (2009) and demonstrate how they can be used for mini-batch ML estimation of exponential family mixture models. In Section 3, we derive the mini-batch EM algorithms for the ML estimation of normal mixtures, as well as verify the convergence of the algorithms using the results of Cappé & Moulines (2009). Via numerical simulations, we compare the performance of our mini-batch algorithms to the usual EM algorithm for ML estimation of normal mixture models, in Section 4. A set of real data study on a very large data set is presented in Section 5. Conclusions are drawn in Section 6. Additional material, such as mini-batch EM algorithms for exponential and Poisson mixture models, can be found in the Supplementary Materials.

2 The mini-batch EM algorithm

Suppose that we observe a single pair of random variables $\boldsymbol{X}^\top = \left(\boldsymbol{Y}^\top, \boldsymbol{Z}^\top\right)$, where $\boldsymbol{Y} \in \mathbb{Y}$ is observed but $\boldsymbol{Z} \in \mathbb{L}$ is latent, where $\mathbb{Y}$ and $\mathbb{L}$ are subsets of multivariate real-valued spaces. Furthermore, suppose that the marginal PDF/PMF of $\boldsymbol{Y}$ is hypothesized to be of the form $f\left(\boldsymbol{y}; \boldsymbol{\theta}_0\right)$, for some unknown parameter vector $\boldsymbol{\theta}_0 \in \Theta \subseteq \mathbb{R}^p$. A good estimator for $\boldsymbol{\theta}_0$ is the ML estimator $\hat{\boldsymbol{\theta}}$ that can be defined as:

$$\hat{\boldsymbol{\theta}} \in \left\{ \hat{\boldsymbol{\theta}} : \log f\left(\boldsymbol{Y}; \hat{\boldsymbol{\theta}}\right) = \max_{\boldsymbol{\theta} \in \Theta} \log f\left(\boldsymbol{Y}; \boldsymbol{\theta}\right) \right\}. \tag{7}$$

When the problem (7) cannot be solved in a simple manner (e.g. when the solution does not exist in closed form), one may seek to employ an iterative scheme in order to obtain an ML estimator. If the joint PDF/PMF of $\boldsymbol{X}$ is known, then one can often construct an EM algorithm in order to solve the problem in the bracket of (7).

Start with some initial guess for $\boldsymbol{\theta}_0$ and call it the zeroth iterate of the EM algorithm $\boldsymbol{\theta}^{(0)}$ and suppose that we can write the point PDF/PMF of $\boldsymbol{X}$ as $f\left(\boldsymbol{y}, \boldsymbol{z}; \boldsymbol{\theta}\right)$, for any $\boldsymbol{\theta}$. At the rth iterate of the EM algorithm, we perform an expectation (E-) step, followed by a maximization (M-) step. The rth E-step consists of obtaining the conditional expectation of the complete-data log-likelihood (i.e. $\log f\left(\boldsymbol{y}, \boldsymbol{z}; \boldsymbol{\theta}\right)$) given the observed data, using the current estimate of the parameter vector

$$Q\left(\boldsymbol{\theta}; \boldsymbol{\theta}^{(r-1)}\right) = \mathbb{E}_{\boldsymbol{\theta}^{(r-1)}}\left[\log f\left(\boldsymbol{y}, \boldsymbol{Z}; \boldsymbol{\theta}\right) | \boldsymbol{Y} = \boldsymbol{y}\right],$$

which we will call the conditional expected complete-data log-likelihood.

Upon obtaining the conditional expectation of the complete-data log-likelihood, one then con-

ducts the rth M-step by solving the problem

$$\boldsymbol{\theta}^{(r)} = \arg\max_{\boldsymbol{\theta}\in\Theta} Q\left(\boldsymbol{\theta}; \boldsymbol{\theta}^{(r-1)}\right).$$

The E- and M-steps are repeated until some stopping criterion is met. Upon termination, the final iterate of the algorithm is taken as a solution for problem (7). See McLachlan & Krishnan (2008) for a thorough exposition regarding the EM algorithm.

2.1 The online EM algorithm

Suppose that we observe a sequence of n IID replicates of the variable $\boldsymbol{Y}$, $\{\boldsymbol{Y}_i\}_{i=1}^n$, where each $\boldsymbol{Y}_i$ is the visible component of the pair $\boldsymbol{X}_i = \left(\boldsymbol{Y}_i^\top, \boldsymbol{Z}_i^\top\right)$ $(i \in [n])$. In the online learning context, each of the observations from $\{\boldsymbol{Y}_i\}_{i=1}^n$ is observed one at a time, in sequential order.

Using the sequentially obtained sequence $\{\boldsymbol{Y}_i\}_{i=1}^n$, we wish to obtain an ML estimator for the parameter vector $\boldsymbol{\theta}_0$, in the same sense as in (7). In order to construct an online EM algorithm framework with provable convergence, Cappé & Moulines (2009) assume the following restrictions regarding the nature of the hypothesized DGP of $\{\boldsymbol{Y}_i\}_{i=1}^n$.

A1 The complete-data likelihood corresponding to the pair $\boldsymbol{X}$ is of exponential family form. That is,

$$f\left(\boldsymbol{x}; \boldsymbol{\theta}\right) = h\left(\boldsymbol{x}\right) \exp\left\{\left[\boldsymbol{s}\left(\boldsymbol{x}\right)\right]^\top \boldsymbol{\phi}\left(\boldsymbol{\theta}\right) - \psi\left(\boldsymbol{\theta}\right)\right\}, \tag{8}$$

where $h\left(\cdot\right)$, $\psi\left(\cdot\right)$, $\boldsymbol{s}\left(\cdot\right)$, and $\boldsymbol{\phi}\left(\cdot\right)$ are as defined for (1).

A2 The function

$$\bar{\boldsymbol{s}}\left(\boldsymbol{y}; \boldsymbol{\theta}\right) = \mathbb{E}_{\boldsymbol{\theta}}\left[\boldsymbol{s}\left(\boldsymbol{X}\right) | \boldsymbol{Y} = \boldsymbol{y}\right] \tag{9}$$

is well defined for all $\boldsymbol{y} \in \mathbb{Y}$ and $\boldsymbol{\theta} \in \Theta$.

A3 There exists a convex open subset $\mathbb{S} \subseteq \mathbb{R}^p$, which satisfies the properties that:

(i) for all $\boldsymbol{s} \in \mathbb{S}$, $\boldsymbol{y} \in \mathbb{Y}$, $\boldsymbol{\theta} \in \Theta$, $(1-\gamma)\,\boldsymbol{s} + \gamma \bar{\boldsymbol{s}}\,(\boldsymbol{y}; \boldsymbol{\theta}) \in \mathbb{S}$ for any $\gamma \in (0,1)$, and

(ii) for any $\boldsymbol{s} \in \mathbb{S}$, the function

$$q\,(\boldsymbol{s}; \boldsymbol{\theta}) = \boldsymbol{s}^\top \boldsymbol{\phi}\,(\boldsymbol{\theta}) - \psi\,(\boldsymbol{\theta})$$

has a unique global maximum over Θ, which will be denoted by

$$\bar{\boldsymbol{\theta}}\,(\boldsymbol{s}) = \arg\max_{\boldsymbol{\theta} \in \Theta} q\,(\boldsymbol{s}; \boldsymbol{\theta})\,.$$

Let $Q_n\left(\boldsymbol{\theta}; \boldsymbol{\theta}^{(r-1)}\right)$ be the expected complete-data log-likelihood over data $\{\boldsymbol{Y}_i\}_{i=1}^n$, at the rth E-step of an EM algorithm for solving the problem:

$$\hat{\boldsymbol{\theta}}_n \in \left\{ \hat{\boldsymbol{\theta}} : n^{-1} \sum_{i=1}^n \log f\left(\boldsymbol{Y}_i; \hat{\boldsymbol{\theta}}\right) = \max_{\boldsymbol{\theta} \in \Theta} n^{-1} \sum_{i=1}^n \log f\left(\boldsymbol{Y}_i; \boldsymbol{\theta}\right) \right\},$$

where we say that $\hat{\boldsymbol{\theta}}_n$ is the ML estimator, based on the data $\{\boldsymbol{Y}_i\}_{i=1}^n$. When, A1–A3 are satisfied, we can write

$$Q_n\left(\boldsymbol{\theta}; \boldsymbol{\theta}^{(r-1)}\right) = nq\left(n^{-1} \sum_{i=1}^n \bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i; \boldsymbol{\theta}^{(r-1)}\right); \boldsymbol{\theta} \right) + \text{Constant},$$

which can then be maximized, with respect to $\boldsymbol{\theta}$, in order to yield an M-step update of the form:

$$\boldsymbol{\theta}^{(r)} = \bar{\boldsymbol{\theta}}\left(n^{-1} \sum_{i=1}^n \bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i; \boldsymbol{\theta}^{(r-1)}\right) \right), \tag{10}$$

where $\boldsymbol{\theta}^{(r)}$ is a function that depends only on the average $n^{-1}\sum_{i=1}^{n}\bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(r-1)}\right)$.

Now we suppose that we sample the individual observations of $\{\boldsymbol{Y}_i\}_{i=1}^{n}$, one at a time and sequentially. Furthermore, upon observation of $\boldsymbol{Y}_i$, we wish to compute an online estimate of $\boldsymbol{\theta}_0$, which we denote as $\boldsymbol{\theta}^{(i)}$. Based on the simplification of the EM algorithm under A1–A3, as described above, Cappé & Moulines (2009) proposed the following online EM algorithm.

Upon observation of $\boldsymbol{Y}_i$, compute the intermediate updated sufficient statistic

$$\boldsymbol{s}^{(i)} = \boldsymbol{s}^{(i-1)} + \gamma_i\left[\bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(i-1)}\right) - \boldsymbol{s}^{(i-1)}\right], \tag{11}$$

with $\boldsymbol{s}^{(0)} = \bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(0)}\right)$. Here, γ_i is the ith term of the learning rate sequence that we will discuss in further details in the sequel. Observe that we can also write

$$\boldsymbol{s}^{(i)} = \gamma_i\bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(i-1)}\right) + (1-\gamma_i)\,\boldsymbol{s}^{(i-1)},$$

which makes it clear that for $\gamma_i \in (0,1)$, $\boldsymbol{s}^{(i)}$ is a weighted average between $\bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(i-1)}\right)$ and $\boldsymbol{s}^{(i-1)}$. Using $\boldsymbol{s}^{(i)}$ and the function $\bar{\boldsymbol{\theta}}$, we can then express the ith iteration online EM estimate of $\boldsymbol{\theta}_0$ as

$$\boldsymbol{\theta}^{(i)} = \bar{\boldsymbol{\theta}}\left(\boldsymbol{s}^{(i)}\right). \tag{12}$$

Next, we state a consistency theorem that strongly motivates the use of the online EM algorithm, defined by (11) and (12). Suppose that the true DGP that generates each $\boldsymbol{Y}_i$ of $\{\boldsymbol{Y}_i\}_{i=1}^{n}$ is characterized by the probability measure F_0. Write the expectation operator with respect to this measure as $\mathbb{E}_{F_0}$. In order to state the consistency result of Cappé & Moulines (2009), we require the following additional set of assumptions.

B1 The parameter space Θ is a convex and open subset of a real product space, and the functions $\boldsymbol{\phi}$ and ψ, in (8), are both twice continuously differentiable with respect to $\boldsymbol{\theta} \in \Theta$.

B2 The function $\bar{\boldsymbol{\theta}}$, as defined in (10), is a continuously differentiable function with respect to $\boldsymbol{s} \in \mathbb{S}$, where $\mathbb{S}$ is as defined in A3.

B3 For some $p > 2$, and all compact $\mathbb{K} \subset \mathbb{S}$,

$$\sup_{\boldsymbol{s} \in \mathbb{K}} \mathbb{E}_{F_0} \left[\left| \bar{\boldsymbol{s}} \left(\boldsymbol{Y}; \bar{\boldsymbol{\theta}} \left(\boldsymbol{s} \right) \right) \right|^p \right] < \infty. \tag{13}$$

As the algorithm defined by (11) and (12) is of the Robbins-Monro type, establishment of convergence of the algorithm requires the definition of a mean field (see Chen, 2003 and Kushner & Yin, 2003 for comprehensive treatments regarding such algorithms). In the case of the online EM algorithm, we write the mean field as

$$\boldsymbol{h} \left(\boldsymbol{s} \right) = \mathbb{E}_{F_0} \left[\bar{\boldsymbol{s}} \left(\boldsymbol{Y}; \bar{\boldsymbol{\theta}} \left(\boldsymbol{s} \right) \right) \right] - \boldsymbol{s}$$

and define the set of its roots as $\Gamma = \left\{ \boldsymbol{s} \in \mathbb{S} : \boldsymbol{h} \left(\boldsymbol{s} \right) = \boldsymbol{0} \right\}$.

Define the log-likelihood of the hypothesized PDF $f \left(\cdot; \boldsymbol{\theta} \right)$ with respect to the measure F_0, as

$$\ell \left(f \left(\cdot; \boldsymbol{\theta} \right) \right) = \mathbb{E}_{F_0} \left[\log f \left(\boldsymbol{Y}; \boldsymbol{\theta} \right) \right].$$

Let $\nabla_{\boldsymbol{\theta}}$ denote the gradient with respect to $\boldsymbol{\theta}$, and define the sets

$$\mathbb{W}_{\Gamma} = \left\{ \ell \left(f \left(\cdot; \boldsymbol{\theta} \right) \right) : \boldsymbol{\theta} = \bar{\boldsymbol{\theta}} \left(\boldsymbol{s} \right), \, \boldsymbol{s} \in \Gamma \right\}$$

and

$$\mathbb{M}_{\Theta} = \left\{ \hat{\boldsymbol{\theta}} \in \Theta : \nabla_{\boldsymbol{\theta}} \, \ell \left(f \left(\cdot ; \boldsymbol{\theta} \right) \right) |_{\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}} = \mathbf{0} \right\}.$$

Note that $\mathbb{M}_{\Theta}$ is the set of stationary points of the log-likelihood function. Further, define the distance between a real vector $\boldsymbol{a}$ and a set $\mathbb{B}$ by

$$\operatorname{dist} \left(\boldsymbol{a}, \mathbb{B} \right) = \inf_{\boldsymbol{b} \in \mathbb{B}} \left\| \boldsymbol{a} - \boldsymbol{b} \right\|,$$

where $\| \cdot \|$ is the usual Euclidean metric, and denote the complement of a subset $\mathbb{A}$ of a real product space by $\mathbb{A}^c$. Finally, make the following assumptions.

C1 The sequence of learning rates $\{\gamma_i\}_{i=1}^{\infty}$ fulfills the conditions that $0 < \gamma_i < 1$, for each i,

$$\sum_{i=1}^{\infty} \gamma_i = \infty, \text{ and } \sum_{i=1}^{\infty} \gamma_i^2 < \infty.$$

C2 At initialization $\boldsymbol{s}^{(0)} \in \mathbb{S}$ and, with probability 1,

$$\limsup_{i \to \infty} \left| \boldsymbol{s}^{(i)} \right| < \infty, \text{ and } \liminf_{i \to \infty} \operatorname{dist} \left(\boldsymbol{s}^{(i)}, \mathbb{S}^c \right) = 0.$$

C3 The set $\mathbb{W}_{\Gamma}$ is nowhere dense.

Theorem 1 (Cappe and Moulines, 2009)**.** *Assume that A1–A3, B1–B3, and C1–C3 are satisfied, and let* $\{\boldsymbol{Y}_i\}_{i=1}^{\infty}$ *be an IID sample with DGP characterized by the PDF* f_0*, which is hypothesized to have the form* $f \left(\cdot ; \boldsymbol{\theta} \right)$*, as in (8). Further, let* $\left\{ \boldsymbol{s}^{(i)} \right\}_{i=1}^{\infty}$ *and* $\left\{ \boldsymbol{\theta}^{(i)} \right\}_{i=1}^{\infty}$ *be sequences generated by the*

online EM algorithm, defined by (11) and (12). Then, with probability 1,

$$\lim_{i\to\infty} \; dist\left(\boldsymbol{s}^{(i)}, \Gamma\right) = 0, \; and \; \lim_{i\to\infty} \; dist\left(\boldsymbol{\theta}^{(i)}, \mathbb{M}_\Theta\right) = 0.$$

Notice that this result allows for a mismatch between the true probability measure F_0 and the assumed pseudo-true family $f(\cdot;\boldsymbol{\theta})$ from which $\{\boldsymbol{Y}_i\}_{i=1}^{\infty}$ is hypothesized to arise. This therefore allows for misspecification, in the sense of White (1982), which is almost certain to occur in the modeling of any sufficiently complex data. In any case, the online EM algorithm will converge towards an estimate of the parameter vector $\boldsymbol{\theta}$, which is in the set $\mathbb{M}_\Theta$. When the DGP can be characterized by a density in the family of the form $f(\cdot;\boldsymbol{\theta})$, we observe that $\mathbb{M}_\Theta$ contains not only the global maximizer of the log-likelihood function, but also local maximizers, minimizers, and saddle points. Thus, the online algorithm suffers from the same lack of strong convergence guarantees, as the batch EM algorithm (cf. Wu, 1983).

In the case of misspecification the set $\mathbb{M}_\Theta$ will include the parameter vector $\boldsymbol{\theta}_0$ that maximizes the log-likelihood function, with respect to the true probability measure F_0. However, as with the well-specified case, it will also include stationary points of other types, as well. We further provide characterizations of the sets $\mathbb{W}_\Gamma$ and $\mathbb{M}_\Theta$ in terms of the Kullback-Leibler divergence (KL; Kullback & Leibler, 1951) in the Supplementary Materials.

Assumption C1 can be fulfilled by taking sequences $\{\gamma_i\}_{i=1}^{\infty}$ of form $\gamma_i = \gamma_0 i^{\alpha}$, for some $\alpha \in (0, 1]$ and $\gamma_0 \in (0, 1)$. We shall discuss this point further, in the sequel. Although the majority of the assumptions can be verified or are fulfilled by construction, the two limits in C2 stand out as being particularly difficult to verify. In Cappé & Moulines (2009), the authors suggest that one method for enforcing C2 is to use the method of update truncation, but they did not provide an explicit scheme for conducting such truncation.

A truncation version of the algorithm defined by (11) and (12) can be specified via the method of Delyon et al. (1999). That is, let $\{\mathbb{K}_m\}_{m=0}^{\infty}$ be a sequence of compact sets, such that

$$\mathbb{K}_m \subset \text{interior}\left(\mathbb{K}_{m+1}\right), \text{ and } \bigcup_{m=0}^{\infty} \mathbb{K}_m = \mathbb{S}. \tag{14}$$

We then replace (11) and (12) by the following scheme. At the ith iteration, firstly compute

$$\tilde{\boldsymbol{s}}^{(i)} = \boldsymbol{s}^{(i-1)} + \gamma_i \left[\bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i; \boldsymbol{\theta}^{(i-1)}\right) - \boldsymbol{s}^{(i-1)}\right]. \tag{15}$$

Secondly,

$$\text{if } \tilde{\boldsymbol{s}}^{(i)} \in \mathbb{K}_{m_{i-1}}, \text{ then set } \boldsymbol{s}^{(i)} = \tilde{\boldsymbol{s}}^{(i)}, \boldsymbol{\theta}^{(i)} = \bar{\boldsymbol{\theta}}\left(\boldsymbol{s}^{(i)}\right), \text{ and } m_i = m_{i-1}, \tag{16}$$

else

$$\text{if } \tilde{\boldsymbol{s}}^{(i)} \notin \mathbb{K}_{m_{i-1}}, \text{ then set } \boldsymbol{s}^{(i)} = \boldsymbol{S}_i, \boldsymbol{\theta}^{(i)} = \bar{\boldsymbol{\theta}}\left(\boldsymbol{S}_i\right), \text{ and } m_i = m_{i-1} + 1, \tag{17}$$

where $\{\boldsymbol{S}_i\}_{i=1}^{\infty}$ is an arbitrary random sequence, such that $\boldsymbol{S}_i \in \mathbb{K}_0$, for each $i \in \mathbb{N}$. We have the following result regarding the algorithm defined by (15)–(17).

Proposition 1. *Assume that A1–A3, B1–B3, C1 and C3 are satisfied, and let* $\{\boldsymbol{Y}_i\}_{i=1}^{\infty}$ *be an IID sample with DGP characterized by the PDF* f_0*, which is hypothesized to have the form* $f\left(\cdot; \boldsymbol{\theta}\right)$*, as in (8). Further, let* $\left\{\boldsymbol{s}^{(i)}\right\}_{i=1}^{\infty}$ *and* $\left\{\boldsymbol{\theta}^{(i)}\right\}_{i=1}^{\infty}$ *be sequences generated by the truncated online EM algorithm, defined by (15)–(17). Then, with probability 1,*

$$\lim_{i\to\infty} \; dist\left(\boldsymbol{s}^{(i)}, \Gamma\right) = 0, \; and \; \lim_{i\to\infty} \; dist\left(\boldsymbol{\theta}^{(i)}, \mathbb{M}_{\Theta}\right) = 0.$$

The proof of Proposition 1 requires the establishment of equivalence between A1–A3, B1–B3,

C1, and C3, and the many assumptions of Theorem 3 and 6 of Delyon et al. (1999). Thus the proof is simple and mechanical, but long and tedious. We omit it for the sake of brevity.

2.2 The mini-batch algorithm

At the most elementary level, a mini-batch algorithm for computation of a sequence of estimators $\left\{\boldsymbol{\theta}^{(r)}\right\}_{r=1}^{R}$ for some parameter $\boldsymbol{\theta}_0$, from some sample $\{\boldsymbol{Y}_i\}_{i=1}^{n}$, where $R \in \mathbb{N}$, has the following property. The algorithm is iterative, and at the rth iteration of the algorithm, the estimator $\boldsymbol{\theta}^{(r)}$ only depends on the previous iterate $\boldsymbol{\theta}^{(r-1)}$ and some subsample, possibly with replacement, of $\{\boldsymbol{Y}_i\}_{i=1}^{n}$. Typical examples of mini-batch algorithms include the many variants of the stochastic gradient descent-class of algorithms; see, for example, Cotter et al. (2011), Li et al. (2014), Zhao et al. (2014), and Ghadimi et al. (2016).

Suppose that we observe a fixed size realization $\{\boldsymbol{y}_i\}_{i=1}^{n}$ of some IID random sample $\{\boldsymbol{Y}\}_{i=1}^{n}$. Furthermore, fix a so-called batch size $N \leq n$ and a learning rate sequence $\{\gamma_r\}_{r=1}^{R}$, and select some appropriate initial values $\boldsymbol{s}^{(0)}$ and $\boldsymbol{\theta}^{(0)}$ from which the sequences $\left\{\boldsymbol{s}^{(r)}\right\}_{r=1}^{R}$ and $\left\{\boldsymbol{\theta}^{(r)}\right\}_{r=1}^{R}$ can be constructed. A mini-batch version of the online EM algorithm, specified by (11) and (12) can be specified as follows. For each $r \in [R]$, sample N observations from $\{\boldsymbol{y}_i\}_{i=1}^{n}$ uniformly, with replacement, and denote the subsample by $\{\boldsymbol{Y}_i^r\}_{i=1}^{N}$. Then, using $\{\boldsymbol{Y}_i^r\}_{i=1}^{N}$, compute

$$\boldsymbol{s}^{(r)} = \boldsymbol{s}^{(r-1)} + \gamma_r \left[N^{-1} \sum_{i=1}^{N} \bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i^r; \boldsymbol{\theta}^{(r-1)}\right) - \boldsymbol{s}^{(r-1)} \right], \text{ and } \boldsymbol{\theta}^{(r)} = \bar{\boldsymbol{\theta}}\left(\boldsymbol{s}^{(r)}\right). \tag{18}$$

In order to justify the mini-batch algorithm, we make the following observation. The online EM algorithm, defined by (11) and (12), is designed to obtain a root in the set $\mathbb{M}_\Theta$, which is a vector $\hat{\boldsymbol{\theta}} \in \Theta$ such that

$$\nabla_{\boldsymbol{\theta}} \, \ell\left(f\left(\cdot; \boldsymbol{\theta}\right)\right)\big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}} = \mathbf{0}.$$

If $N = 1$ (i.e., the case proposed in Cappé & Moulines, 2009, Sec. 2.5), then the DGP for generating subsamples is simply a single draw from the empirical measure:

$$F_{\text{Emp}}(\boldsymbol{y}) = \sum_{i=1}^{n} \frac{1}{n} \delta(\boldsymbol{y} - \boldsymbol{y}_i),$$

where δ is the Dirac delta function (see, for details, Prosperetti, 2011, Ch. 2). We can write

$$\begin{aligned} \ell(f(\cdot;\boldsymbol{\theta})) &= \mathbb{E}_{F_0}[\log f(\boldsymbol{Y};\boldsymbol{\theta})] \\ &= \mathbb{E}_{F_{\text{Emp}}}[\log f(\boldsymbol{Y};\boldsymbol{\theta})] \\ &= \frac{1}{n} \sum_{i=1}^{n} \log f(\boldsymbol{y}_i;\boldsymbol{\theta}), \end{aligned} \tag{19}$$

which is the log-likelihood function, with respect to the realization $\{\boldsymbol{y}_i\}_{i=1}^n$, under the density function of form $f(\cdot;\boldsymbol{\theta})$. Thus, in the $N = 1$ case, the algorithm defined by (18) solves for log-likelihood roots $\hat{\boldsymbol{\theta}}$ of the form

$$\frac{1}{n} \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{y}_i;\boldsymbol{\theta})|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}} = \mathbf{0},$$

or equivalently, solving for an element in the set

$$\mathbb{M}_{\Theta}^{\text{Emp}} = \left\{ \hat{\boldsymbol{\theta}} \in \Theta : \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \log f(\boldsymbol{y}_i;\boldsymbol{\theta})|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}} = \mathbf{0} \right\}.$$

The $N > 1$ case follows the same argument, and is described in the Supplementary Materials (Section 2.2). Let F_{Emp}^N denote the probability measure corresponding to the DGP of N independent random samples from F_{Emp}. We have the following result, based on Theorem 1.

Corollary 1. *For any $N \in \mathbb{N}$, assume that A1–A3, B1–B3, and C1–C3 are satisfied (replacing i by r, and F_0 by F_{Emp}^N, where appropriate), and let $\{\boldsymbol{y}_i\}_{i=1}^n$ be a realization of some IID random sequence $\{\boldsymbol{Y}_i\}_{i=1}^n$, where each $\boldsymbol{Y}_i$ is hypothesized to arise from a DGP having PDF of the form $f(\cdot;\boldsymbol{\theta})$, as in (8). Let $\left\{\boldsymbol{s}^{(r)}\right\}_{i=1}^{\infty}$ and $\left\{\boldsymbol{\theta}^{(r)}\right\}_{i=1}^{\infty}$ be sequences generated by the mini-batch EM algorithm, defined by (11) and (12). Then, with probability 1,*

$$\lim_{r\to\infty} \; dist\left(\boldsymbol{s}^{(r)}, \Gamma\right) = 0, \; and \; \lim_{r\to\infty} \; dist\left(\boldsymbol{\theta}^{(r)}, \mathbb{M}_{\Theta}^{Emp}\right) = 0.$$

That is, as we take $R \to \infty$, the algorithm defined by (11) and (12) will identify elements in the sets Γ and $\mathbb{M}_{\Theta}^{\text{Emp}}$, with probability 1. As with the case of Theorem 1, C2 is again difficult to verify. Let $\{\mathbb{K}_m\}_{m=0}^{\infty}$ be as per (14). Then, we replace the algorithm defined via (18), by the following truncated version.

Again, suppose that we observe a fixed size realization $\{\boldsymbol{y}_i\}_{i=1}^n$ of some IID random sample $\{\boldsymbol{Y}\}_{i=1}^n$. Furthermore, fix a so-called batch size $N \leq n$ and a learning rate sequence $\{\gamma_r\}_{r=1}^R$, and select some appropriate initial values $\boldsymbol{s}^{(0)}$ and $\boldsymbol{\theta}^{(0)}$ from which the sequences $\left\{\boldsymbol{s}^{(r)}\right\}_{r=1}^R$ and $\left\{\boldsymbol{\theta}^{(r)}\right\}_{r=1}^R$ can be constructed. For each $r \in [R]$, sample N observations from $\{\boldsymbol{y}_i\}_{i=1}^n$ uniformly, with replacement, and denote the subsample by $\{\boldsymbol{Y}_i^r\}_{i=1}^N$. Using $\{\boldsymbol{Y}_i^r\}_{i=1}^N$, compute

$$\tilde{\boldsymbol{s}}^{(r)} = \boldsymbol{s}^{(r-1)} + \gamma_r \left[N^{-1} \sum_{i=1}^{N} \bar{\boldsymbol{s}}\left(\boldsymbol{Y}_i^r; \boldsymbol{\theta}^{(r-1)}\right) - \boldsymbol{s}^{(r-1)} \right]. \tag{20}$$

Then, with i being appropriately replaced by r, use (16) and (17) to compute $\boldsymbol{s}^{(r)}$ and $\boldsymbol{\theta}^{(r)}$. We obtain the following result via an application of Proposition 1.

Corollary 2. *For any $N \in \mathbb{N}$, assume that A1–A3, B1–B3, and C1–C3 are satisfied (replacing i by r, and F_0 by F_{Emp}^N, where appropriate), and let $\{\boldsymbol{y}_i\}_{i=1}^n$ be a realization of some IID random*

sequence $\{\boldsymbol{Y}_i\}_{i=1}^n$*, where each* $\boldsymbol{Y}_i$ *is hypothesized to arise from a DGP having PDF of the form* $f(\cdot;\boldsymbol{\theta})$*, as in (8). Let* $\left\{\boldsymbol{s}^{(r)}\right\}_{i=1}^{\infty}$ *and* $\left\{\boldsymbol{\theta}^{(r)}\right\}_{i=1}^{\infty}$ *be sequences generated by the truncated mini-batch EM algorithm, defined by (20), (16), and (17). Then, with probability 1,*

$$\lim_{r\to\infty} \; dist\left(\boldsymbol{s}^{(r)}, \Gamma\right) = 0, \; and \; \lim_{r\to\infty} \; dist\left(\boldsymbol{\theta}^{(r)}, \mathbb{M}_{\Theta}^{Emp}\right) = 0.$$

2.3 The learning rate sequence

As previously stated, a good choice for the learning rate sequence $\{\gamma_i\}_{i=1}^{\infty}$ is to take $\gamma_i = \gamma_0 i^{\alpha}$, for each $i \in \mathbb{N}$, such that $\alpha \in (1/2, 1]$ and $\gamma_0 \in (0, 1)$. Under the assumptions of Theorem 1, Cappé & Moulines (2009, Thm. 2) showed that the learning rate choice leads to the convergence of the sequence $\gamma_0^{1/2} i^{\alpha/2}\left(\boldsymbol{\theta}^{(i)} - \boldsymbol{\theta}_0\right)$, in distribution, to a normal distribution with mean $\mathbf{0}$ and covariance matrix depending on $\boldsymbol{\theta}_0$, for some $\boldsymbol{\theta}_0 \in \mathbb{M}_{\Theta}$. Here $\left\{\boldsymbol{\theta}^{(i)}\right\}_{i=1}^{\infty}$ is a sequence of online EM algorithm iterates, generated by (11) and (12). A similar result can be stated for the truncated online EM, mini-batch EM, and truncated mini-batch EM algorithms, by replacing the relevant indices and quantities in the previous statements by their respective counterparts.

The result above implies that the convergence rate is $\boldsymbol{\theta}^{(i)} - \boldsymbol{\theta}_0 = o_{\mathrm{p}}\left(i^{\alpha/2}\right)$, for any valid α, where o_{p} is the usual order in probability notation (see White, 2001, Defn. 2.33). Thus, it would be tempting to take $\alpha = 1$ in order to obtain a rate with optimal order of $n^{1/2}$. However, as shown in Cappé & Moulines (2009, Thm. 2), the $\alpha = 1$ case requires constraints on γ_0 in order to fulfill a stability assumption that is impossible to validate, in practice.

It is, however, still possible to obtain a sequence of estimators that converges to some $\boldsymbol{\theta}_0$ at a rate with optimal order $n^{1/2}$. We can do this via the famous so-called Polyak averaging scheme of Polyak (1990) and Polyak & Juditsky (1992). In the current context, one takes as an input

the sequence of online EM iterates $\left\{\boldsymbol{\theta}^{(i)}\right\}_{i=1}^{\infty}$, and output the running average sequence $\left\{\boldsymbol{\theta}_{\mathrm{A}}^{(i)}\right\}_{i=1}^{\infty}$, where

$$\boldsymbol{\theta}_{\mathrm{A}}^{(i)} = i^{-1} \sum_{j=1}^{i} \boldsymbol{\theta}^{(j)}, \tag{21}$$

for each $i \in \mathbb{N}$. For any $\alpha \in (1/2, 1)$, it is provable that $\boldsymbol{\theta}_{\mathrm{A}}^{(i)} - \boldsymbol{\theta}_0 = o_{\mathrm{p}}\left(n^{1/2}\right)$. As before, this result generalizes to the cases of the truncated online EM, mini-batch EM, and truncated mini-batch EM algorithms, also.

We note that the computation of the ith running average term (21) does not require the storage of the entire sequence of iterates $\left\{\boldsymbol{\theta}^{(i)}\right\}_{i=1}^{\infty}$, as one would anticipate by applying (21) naÃ¯vely. One can instead write (21) in the iterative form

$$\boldsymbol{\theta}_{\mathrm{A}}^{(i)} = i^{-1}\left[(i-1)\,\boldsymbol{\theta}_{\mathrm{A}}^{(i-1)} + \boldsymbol{\theta}^{(i)}\right].$$

3 Normal mixture models

3.1 Finite mixtures of exponential family distributions

We recall from Section 1, that the random variable $\boldsymbol{Y}$ is said to arise from a DGP characterized by a g component finite mixture of component PDFs of form $f\left(\boldsymbol{y}; \boldsymbol{\omega}_z\right)$, if it has a PDF of the form (2). Furthermore, if the component PDFs are of the exponential family form (1), then we further write the PDF of $\boldsymbol{Y}$ as

$$f\left(\boldsymbol{y}; \boldsymbol{\theta}\right) = \sum_{z=1}^{g} \pi_z h\left(\boldsymbol{y}\right) \exp\left\{\left[\boldsymbol{s}\left(\boldsymbol{y}\right)\right]^{\top} \boldsymbol{\phi}\left(\boldsymbol{\omega}_z\right) - \psi\left(\boldsymbol{\omega}_z\right)\right\}. \tag{22}$$

From the construction of the finite mixture model, we have the fact that (22) is the marginal-

ization of the joint density of the random variable $\boldsymbol{X}^\top = \left(\boldsymbol{Y}^\top, Z\right)$:

$$f\left(\boldsymbol{x};\boldsymbol{\theta}\right) = \prod_{\zeta=1}^{g}\left[\pi_\zeta h\left(\boldsymbol{y}\right)\exp\left\{\left[\boldsymbol{s}\left(\boldsymbol{y}\right)\right]^\top \boldsymbol{\phi}\left(\boldsymbol{\omega}_\zeta\right) - \psi\left(\boldsymbol{\omega}_\zeta\right)\right\}\right]^{[\![z=\zeta]\!]} \tag{23}$$

over the random variable $Z \in [g]$, recalling that Z is a categorical random variable with g categories (cf. McLachlan & Peel, 2000, Ch. 2). Here, $[\![c]\!]$ is the Iverson bracket notation, that takes value 1 if condition c is true, and 0 otherwise (Iverson, 1967, Ch. 1). We rewrite (23) as follows:

$$\begin{aligned} f\left(\boldsymbol{x};\boldsymbol{\theta}\right) &= h\left(\boldsymbol{y}\right)\exp\left\{\sum_{\zeta=1}^{g}[\![z=\zeta]\!]\left[\log\pi_\zeta + \left[\boldsymbol{s}\left(\boldsymbol{y}\right)\right]^\top \boldsymbol{\phi}\left(\boldsymbol{\omega}_\zeta\right) - \psi\left(\boldsymbol{\omega}_\zeta\right)\right]\right\} \\ &= h\left(\boldsymbol{x}\right)\exp\left\{\left[\boldsymbol{s}\left(\boldsymbol{x}\right)\right]^\top \boldsymbol{\phi}\left(\boldsymbol{\theta}\right) - \psi\left(\boldsymbol{\theta}\right)\right\}, \end{aligned}$$

where $h\left(\boldsymbol{x}\right) = h\left(\boldsymbol{y}\right)$, $\psi\left(\boldsymbol{\theta}\right) = 0$,

$$\boldsymbol{s}\left(\boldsymbol{x}\right) = \begin{bmatrix} [\![z=1]\!] \\ [\![z=1]\!]\,\boldsymbol{s}\left(\boldsymbol{y}\right) \\ \vdots \\ [\![z=g]\!] \\ [\![z=g]\!]\,\boldsymbol{s}\left(\boldsymbol{y}\right) \end{bmatrix}, \text{ and } \boldsymbol{\phi}\left(\boldsymbol{\theta}\right) = \begin{bmatrix} \log\pi_1 - \psi\left(\boldsymbol{\omega}_1\right) \\ \boldsymbol{\phi}\left(\boldsymbol{\omega}_1\right) \\ \vdots \\ \log\pi_g - \psi\left(\boldsymbol{\omega}_g\right) \\ \boldsymbol{\phi}\left(\boldsymbol{\omega}_g\right) \end{bmatrix},$$

and thus obtain the following general result regarding finite mixtures of exponential family distributions.

Proposition 2. *The complete-data likelihood of any finite mixture of exponential family distributions with PDF of the form (22) can also be written in the exponential family form (8).*

With Proposition 2, we have proved that when applying the online EM or the mini-batch EM algorithm to the problem of conducting ML estimation for any finite mixture model of exponential family distributions, A1 is automatically satisfied.

3.2 Finite mixtures of normal distributions

Recall from Section 1 that the random variable $\boldsymbol{Y}$ is said to be distributed according to a g-component finite mixture of normal distributions, if it characterized by a PDF of the form (3). Using the exponential family decomposition from (5) and (6), we write the complete-data likelihood of $\boldsymbol{X}^\top = \left(\boldsymbol{Y}^\top, Z\right)$ in the form (8) by setting $h\left(\boldsymbol{x}\right) = \left(2\pi\right)^{-d/2}$, $\psi\left(\boldsymbol{\theta}\right) = 0$,

$$\boldsymbol{s}\left(\boldsymbol{x}\right) = \begin{bmatrix} [\![z=1]\!] \\ [\![z=1]\!]\,\boldsymbol{y} \\ [\![z=1]\!]\,\mathrm{vec}(\boldsymbol{y}\boldsymbol{y}^\top) \\ \vdots \\ [\![z=g]\!] \\ [\![z=g]\!]\,\boldsymbol{y} \\ [\![z=g]\!]\,\mathrm{vec}(\boldsymbol{y}\boldsymbol{y}^\top) \end{bmatrix}, \text{ and } \boldsymbol{\phi}\left(\boldsymbol{\theta}\right) = \begin{bmatrix} \log\pi_1 - \frac{1}{2}\boldsymbol{\mu}_1^\top\boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\log|\boldsymbol{\Sigma}_1| \\ \boldsymbol{\Sigma}_1^{-1}\boldsymbol{\mu}_1 \\ -\frac{1}{2}\mathrm{vec}(\boldsymbol{\Sigma}_1^{-1}) \\ \vdots \\ \log\pi_g - \frac{1}{2}\boldsymbol{\mu}_g^\top\boldsymbol{\Sigma}_g^{-1}\boldsymbol{\mu}_g + \frac{1}{2}\log|\boldsymbol{\Sigma}_g| \\ \boldsymbol{\Sigma}_g^{-1}\boldsymbol{\mu}_g \\ -\frac{1}{2}\mathrm{vec}(\boldsymbol{\Sigma}_g^{-1}) \end{bmatrix}, \tag{24}$$

where $\mathrm{vec}(\cdot)$ is the matrix vectorization operator.

Using the results from McLachlan & Peel (2000, Ch. 3), we write the conditional expectation (9) in the form

$$\left[\bar{\boldsymbol{s}}\left(\boldsymbol{y};\boldsymbol{\theta}\right)\right]^\top = \big(\tau_1\left(\boldsymbol{y};\boldsymbol{\theta}\right), \tau_1\left(\boldsymbol{y};\boldsymbol{\theta}\right)\boldsymbol{y}, \tau_1\left(\boldsymbol{y};\boldsymbol{\theta}\right)\mathrm{vec}(\boldsymbol{y}\boldsymbol{y}^\top), \ldots, \tau_g\left(\boldsymbol{y};\boldsymbol{\theta}\right), \tau_g\left(\boldsymbol{y};\boldsymbol{\theta}\right)\boldsymbol{y}, \tau_g\left(\boldsymbol{y};\boldsymbol{\theta}\right)\mathrm{vec}(\boldsymbol{y}\boldsymbol{y}^\top)\big),$$

where

$$\tau_z\left(\boldsymbol{y};\boldsymbol{\theta}\right)=\frac{\pi_z\varphi\left(\boldsymbol{y};\boldsymbol{\mu}_z,\boldsymbol{\Sigma}_z\right)}{\sum_{\zeta=1}^{g}\pi_\zeta\varphi\left(\boldsymbol{y};\boldsymbol{\mu}_\zeta,\boldsymbol{\Sigma}_\zeta\right)},$$

is the usual *a posteriori* probability that $Z=z$ ($z\in[g]$), given observation of $\boldsymbol{Y}=\boldsymbol{y}$. Again, via the results from McLachlan & Peel (2000, Ch. 3), we write the update function $\bar{\boldsymbol{\theta}}$ in the following form. Define $\bar{\boldsymbol{\theta}}$ to have the elements $\bar{\pi}_z$ and $\bar{\boldsymbol{\omega}}_z$, for each $z\in[g]$, where each $\bar{\boldsymbol{\omega}}_z$ subsequently has elements $\bar{\boldsymbol{\mu}}_z$ and $\bar{\boldsymbol{\Sigma}}_z$. Furthermore, for convenience, we define for $\boldsymbol{s}$ the following notation

$$\boldsymbol{s}^\top=(s_{11},\boldsymbol{s}_{21},\mathrm{vec}(\mathbf{S}_{31}),\ldots,s_{1g},\boldsymbol{s}_{2g},\mathrm{vec}(\mathbf{S}_{3g})),$$

and

$$\left[\bar{\boldsymbol{s}}\left(\boldsymbol{y};\boldsymbol{\theta}\right)\right]^\top=\left(\bar{s}_{11}\left(\boldsymbol{y};\boldsymbol{\theta}\right),\bar{\boldsymbol{s}}_{21}\left(\boldsymbol{y};\boldsymbol{\theta}\right),\mathrm{vec}(\bar{\mathbf{S}}_{31}\left(\boldsymbol{y};\boldsymbol{\theta}\right)),\ldots,\bar{s}_{1g}\left(\boldsymbol{y};\boldsymbol{\theta}\right),\bar{\boldsymbol{s}}_{2g}\left(\boldsymbol{y};\boldsymbol{\theta}\right),\mathrm{vec}(\bar{\mathbf{S}}_{3g}\left(\boldsymbol{y};\boldsymbol{\theta}\right))),$$

with

$$\bar{s}_{1z}\left(\boldsymbol{y};\boldsymbol{\theta}\right)=\tau_z\left(\boldsymbol{y};\boldsymbol{\theta}\right),\ \bar{\boldsymbol{s}}_{2z}\left(\boldsymbol{y};\boldsymbol{\theta}\right)=\tau_z\left(\boldsymbol{y};\boldsymbol{\theta}\right)\boldsymbol{y},\text{ and }\bar{\mathbf{S}}_{3z}\left(\boldsymbol{y};\boldsymbol{\theta}\right)=\tau_z\left(\boldsymbol{y};\boldsymbol{\theta}\right)\boldsymbol{y}\boldsymbol{y}^\top.$$

Then the application of the M-step is equivalent to apply function $\bar{\boldsymbol{\theta}}$ as a function of $\boldsymbol{s}$, countaining the unique elements of $\bar{\pi}_z$, $\bar{\boldsymbol{\mu}}_z$, and $\bar{\boldsymbol{\Sigma}}_z$, for $z\in[g]$, defined by

$$\bar{\pi}_z(\boldsymbol{s})=\frac{s_{1z}}{\sum_{j=1}^{g}s_{1j}},\quad\bar{\boldsymbol{\mu}}_z(\boldsymbol{s})=\frac{\boldsymbol{s}_{2z}}{s_{1z}},\quad\text{and }\bar{\boldsymbol{\Sigma}}_z(\boldsymbol{s})=\frac{\mathbf{S}_{3z}}{s_{1z}}-\frac{\boldsymbol{s}_{2z}\boldsymbol{s}_{2z}^\top}{s_{1z}^2}.\tag{25}$$

This implies that the mini-batch EM and truncated mini-batch EM algorithms proceed via update rule $\bar{\boldsymbol{\theta}}\left(\boldsymbol{s}^{(r)}\right)$, where $\bar{\boldsymbol{\theta}}$ and $\boldsymbol{s}^{(r)}$ are as given in (18). We start from $\boldsymbol{\theta}^{(0)}$ and $\boldsymbol{s}^{(1)}=$

$N^{-1}\sum_{i=1}^{N}\bar{\boldsymbol{s}}(\boldsymbol{Y}_i,\boldsymbol{\theta}^{(0)})$. Then, $\boldsymbol{\theta}^{(1)\top}=[\bar{\boldsymbol{\theta}}(\boldsymbol{s}^{(1)})]^\top$, which has elements

$$\bar{\pi}_z\left(\boldsymbol{s}^{(1)}\right)=N^{-1}\sum_{i=1}^{N}\tau_z\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(0)}\right),\quad \bar{\boldsymbol{\mu}}_z\left(\boldsymbol{s}^{(1)}\right)=\frac{\sum_{i=1}^{N}\tau_z\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(0)}\right)\boldsymbol{Y}_i}{\sum_{i=1}^{N}\tau_z\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(0)}\right)},\tag{26}$$

and

$$\bar{\boldsymbol{\Sigma}}_z\left(\boldsymbol{s}^{(1)}\right)=\frac{\sum_{i=1}^{N}\tau_z\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(0)}\right)\boldsymbol{Y}_i\boldsymbol{Y}_i^\top}{\sum_{i=1}^{N}\tau_z\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(0)}\right)}-\frac{\left[\sum_{i=1}^{N}\tau_z\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(0)}\right)\boldsymbol{Y}_i\right]\left[\sum_{i=1}^{N}\tau_z\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(0)}\right)\boldsymbol{Y}_i\right]^\top}{\left[\sum_{i=1}^{N}\tau_z\left(\boldsymbol{Y}_i;\boldsymbol{\theta}^{(0)}\right)\right]^2}.\tag{27}$$

3.3 Convergence analysis of the mini-batch algorithm

In addition to Assumptions A1–A3, B1–B3, and C1–C3, make the additional assumption

D1 The Hessian matrix of $\sum_{i=1}^{n}\log f\left(\boldsymbol{y}_i;\boldsymbol{\theta}\right)$, evaluated at any $\boldsymbol{\theta}_0\in\mathbb{M}_\Theta^{\text{Emp}}$, is non-singular with respect to $\boldsymbol{\theta}\in\Theta$.

Assumption D1 is generally satisfied for all but pathological samples $\{\boldsymbol{y}_i\}_{i=1}^n$. The following result is proved in the Supplementary Materials.

Proposition 3. *Let $\{\boldsymbol{y}_i\}_{i=1}^n$ be a realization of some IID random sequence $\{\boldsymbol{Y}_i\}_{i=1}^n$, where each $\boldsymbol{Y}_i$ is hypothesized to arise from a DGP having PDF of the form (3). If $\left\{\boldsymbol{s}^{(r)}\right\}_{i=1}^\infty$ and $\left\{\boldsymbol{\theta}^{(r)}\right\}_{i=1}^\infty$ are sequences generated by the mini-batch EM algorithm, defined by (18) and (25), then for any $N\in\mathbb{N}$, if C1, C2, and D1 are satisfied (replacing i by r, and F_0 by $\prod_{j=1}^N F_{Emp}$, where appropriate), then, with probability 1,*

$$\lim_{r\to\infty}\ dist\left(\boldsymbol{s}^{(r)},\Gamma\right)=0,\ and\ \lim_{r\to\infty}\ dist\left(\boldsymbol{\theta}^{(r)},\mathbb{M}_\Theta^{Emp}\right)=0.$$

Alternatively, if $\left\{\boldsymbol{s}^{(r)}\right\}_{i=1}^\infty$ and $\left\{\boldsymbol{\theta}^{(r)}\right\}_{i=1}^\infty$ are sequences generated by the truncated mini-batch EM

algorithm, defined by (20), (16), (17) and (25), then for any $N \in \mathbb{N}$, if C1 and D1 are satisfied (replacing i by r, and F_0 by $\prod_{j=1}^{N} F_{Emp}$, where appropriate), then, with probability 1,

$$\lim_{r\to\infty} \; dist\left(\boldsymbol{s}^{(r)}, \Gamma\right) = 0, \; and \; \lim_{r\to\infty} \; dist\left(\boldsymbol{\theta}^{(r)}, \mathbb{M}_{\Theta}^{Emp}\right) = 0.$$

3.4 A truncation sequence

In order to apply the truncated version of the mini-batch EM algorithm, we require an appropriate sequence $\{\mathbb{K}_m\}_{m=0}^{\infty}$ that satisfies condition (14). This can be constructed in parts. Let us write

$$\mathbb{K}_m = \mathbb{D}_{g-1}^{m} \times \prod_{i=1}^{g} \left(\mathbb{B}_d^m \times \mathbb{H}_d^m\right), \tag{28}$$

where we shall let $c_1, c_2, c_3 \geq 1$,

$$\mathbb{D}_{g-1}^{m} = \left\{ (\pi_1, \ldots, \pi_g) \in \mathbb{R}^g : \sum_{z=1}^{g} \pi_z = 1, \text{ and } \pi_z \geq \frac{1}{c_1 + m}, \text{ for each } z \in [g] \right\},$$

$$\mathbb{B}_d^m = \left[-\left(c_2 + m\right), c_2 + m \right]^d,$$

and

$$\mathbb{H}_d^m = \left\{ \mathbf{H} \in \mathbb{H}_d : \lambda_1\left(\mathbf{H}\right) \geq \frac{1}{c_3 + m}, \, \lambda_d\left(\mathbf{H}\right) \leq c_3 + m \right\},$$

using the notation $\lambda_1\left(\mathbf{H}\right)$ and $\lambda_d\left(\mathbf{H}\right)$ to denote the smallest and largest eigenvalues of the matrix $\mathbf{H}$. A justification regarding this truncation scheme can be found in the Supplementary Materials.

We make a final note that the construction (28) is not a unique method for satisfying the conditions of (14). One can instead, for example, replace $c_j + m$, by $c_j\left(1 + m\right)$ $(j \in [3])$ in the definitions of the sets that constitute (28).

4 Simulation studies

We present a pair of simulation studies, based upon the famous Iris data set of Fisher (1936) and the Wreath data of Fraley et al. (2005), in the main text. A further four simulation scenarios are presented in the Supplementary Materials. In each case, we utilize the initial small data sets, obtained from the base `R` package (R Core Team, 2018) and the `mclust` package for `R` (Scrucca et al., 2016), respectively, and use them as templates to generate much larger data sets. All computations are conducted in the `R` programming environment, although much of the bespoke programs are programmed in `C` and integrated in `R` via the `Rcpp` and `RcppArmadillo` packages of (Eddelbuettel, 2013). Furthermore, timings of programs were conducted on a MacBook Pro with a 2.2 GHz Intel Core i7 processor, 16 GB of 1600 MHz DDR3 RAM, and a 500 GB SSD hard drive. We note that all of the code used to conduct the simulations and computations for this manuscript can be accessed from `https://github.com/hiendn/StoEMMIX`.

In the sequel, in all instances, we shall use the learning rate sequence $\{\gamma_r\}_{r=1}^{\infty}$, where $\gamma_r = (1 - 10^{-10}) \times r^{6/10}$, which follows from the choice made by Cappé & Moulines (2009) in their experiments. In all computations, a fixed number of epochs (or epoch equivalence) of 10 is allotted to each algorithm. Here, recall that the number of epochs is equal to the number of sweeps through the data set $\{\boldsymbol{y}_i\}_{i=1}^{n}$ that an algorithm is allowed. Thus, drawing $10n$ observations from the data $\{\boldsymbol{y}_i\}_{i=1}^{n}$, with replacement, is equivalent to 10 epochs. Thus, each iteration of the standard EM algorithm counts as a single epoch, whereas, for a mini-batch algorithm with batch size N, every n/N iterations counts as an epoch.

Next, in both of our studies, we consider batch sizes of $N = n/10$ and $N = n/5$, we further consider Polyak averaging as well as truncation. Thus, for each study, a total of eight variants of the mini-batch EM algorithm is considered. In the truncate case, we set $c_1, c_2, c_3 = 1000$. Finally,

the variants of the mini-batch EM algorithm are compared to the standard (batch) EM algorithm for fitting finite mixtures of normal distributions. In the interest of fairness, each of the algorithms is initialized at the same starting value of $\boldsymbol{\theta}^{(0)}$, using the randomized initialization scheme suggested in McLachlan & Peel (2000, Sec. 3.9.3). That is, the same randomized starting instance is used for the EM algorithm and each of the mini-batch variants.

To the best of our knowledge, the most efficient and reliable implementation of the EM algorithm for finite mixtures of normal distributions, in `R`, is the `em` function from the `mclust` package. Thus, this will be used for all of our comparisons. Data generation from the template data sets is handled using the `simdataset` function from the `MixSim` package (Melnykov et al., 2012), in the Iris study, and the `simVVV` function from `mclust` in the Wreath study. Timing was conducted using the `proc.time` function.

4.1 Iris data

The Iris data (accessed in `R` via the `data(iris)` command) contain measurements of $d = 4$ dimensions from 150 iris flowers, 50 of each are of the species *Setosa*, *Versicolor*, and *Virginica*, respectively. The 4 dimensions of each flower that are measured are *petal length*, *petal width*, *sepal length*, and *sepal width*. To each of the subpopulations of species, we fit a single multivariate normal distribution to the 50 observations (i.e., we estimate a mean vector and covariance matrix, for each species). Then, using the three mean vectors and covariance matrices, we construct a template $g = 3$ component normal mixture model with equal mixing proportions $\pi_z = 1/3$ ($z \in [3]$), of form (3). This template distribution is then used to generate synthetic data sets of any size n.

Two experiments are performed using this simulation scheme. In the first experiment, we generate $n = 10^6$ observations $\{\boldsymbol{y}_i\}_{i=1}^n$ from the template. We then utilize $\{\boldsymbol{y}_i\}_{i=1}^n$ and each of the

truncated EM algorithm variants as well as the batch EM algorithm to compute ML estimates. We use a number of measures of performance for each algorithm variant. These include the computation time, the log-likelihood, the squared error of the parameter estimates (SE; the Euclidean distance as compared to the generative parameter vector), and the adjusted-Rand index (ARI; Hubert & Arabie, 1985) between the maximum *a posteriori* clustering labels obtained from the fitted mixture model and the true generative data labels.

The ARI measures whether or not two sets of labels are in concordance or not. Here a value of 1 indicates perfect similarity, and 0 indicates discordance. Since the ARI allows for randomness in the labelling process, it is possible to have negative ARI values, which are rare and also indicates discordance in the data. Each variant is repeated Rep = 100 times, and each performance measurement is recorded in order to obtain a measure of the overall performance of each algorithm. For future reference, we name this study Iris1. In the second study, which we name Iris2, we repeat the setup of Iris1 but with the number of observations increased to $n = 10^7$.

4.2 Wreath data

The Wreath data (accessed in `R` via the `data(wreath)` command) contain 1000 observations of $d = 2$ dimensional vectors, each belonging to one of $g = 14$ distinct but unlabelled subpopulations. We use the `Mclust` function from `mclust` to fit a 14 component mixture normal distributions to the data. The data, along with the means of the subpopulation normal distributions, are plotted in Figure 1. Here, each observation is colored based upon the subpopulation that maximizes its *a posteriori* probability.

As with the Iris data, using the fitted mixture model as a template, we can then simulate synthetic data sets of any size n. We perform two experiments using this scheme. In the first

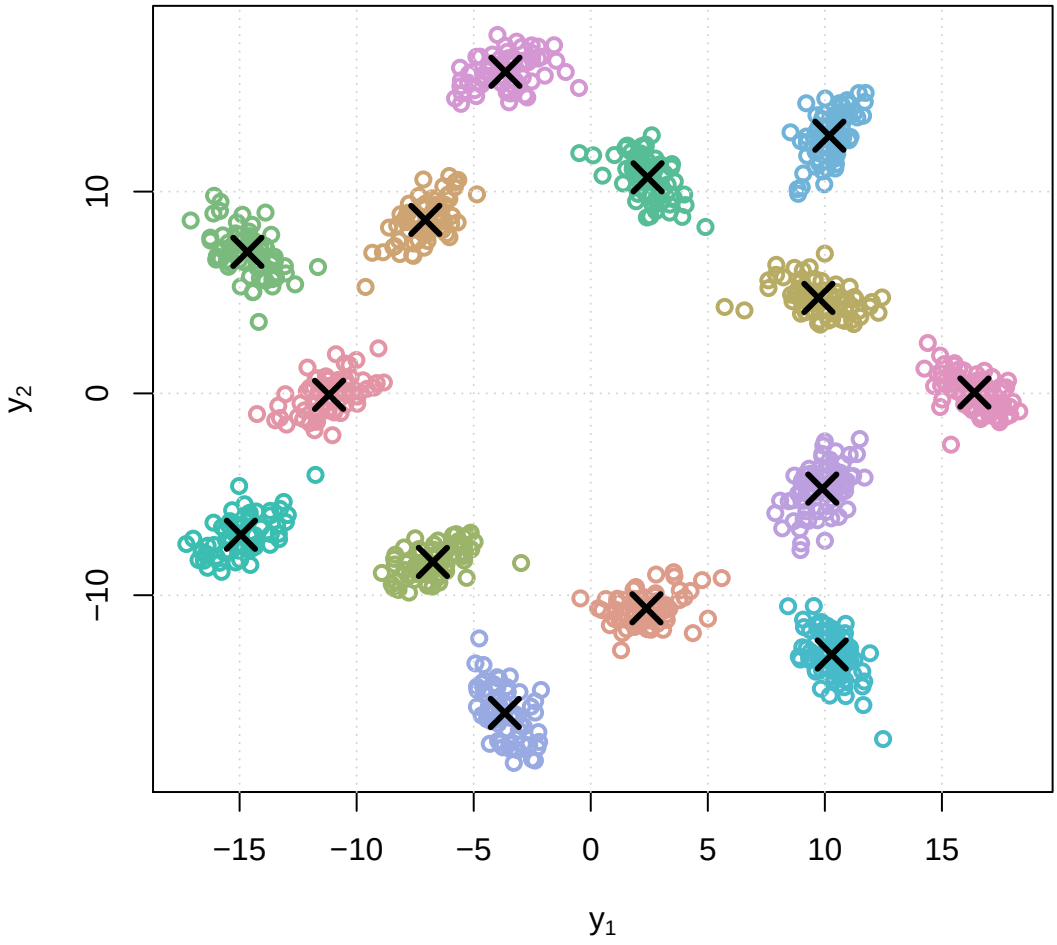


Figure 1: Plot of the 1000 observations of the Wreath data set, colored by subpopulation with subpopulation means indicated by crosses.

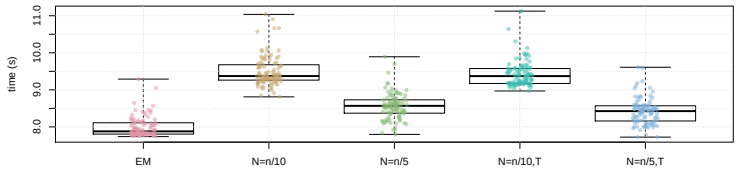
experiment, we simulate $n = 10^6$ observations and assess the different algorithms, based on the computation time, the log-likelihood, the SE, and the ARI over Rep=100 repetitions, as per Iris1. We refer to this experiment as Wreath1. In the second experiment, we repeat the setup of Wreath1, but with $n = 10^7$, instead. We refer to this case as Wreath2.

4.3 Results

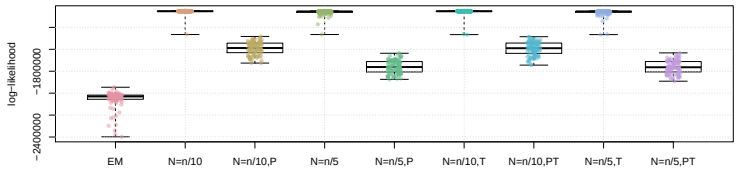
Figures 2 and 3 contain box plots that summarize the results of Iris1 and Iris2, respectively. Similarly, Figures 4 and 5 contain box plots that summarize the results of Wreath1 and Wreath2, respectively.

Firstly, we note that Polyak averaging requires no additional computational effort, for a given value of N. Thus, we do not require separate timing data for Polyak averaging variants of the mini-batch EM algorithms in each of Figures 2–5. From the timing results, we observe that the standard EM algorithm is faster than the mini-batch versions, in all scenarios, regardless of the fact that all of the algorithms were computed using 10 epochs worth of data access. This is because the mini-batch algorithms require more additional intermediate steps in each algorithm loop (e.g., random sampling from the empirical distribution), as well as a multiplicative factor of n/N more loops. From the plots, we observe that the larger value of N tends to result in smaller computing times. There appears to be no difference in timing between the use of truncation or not.

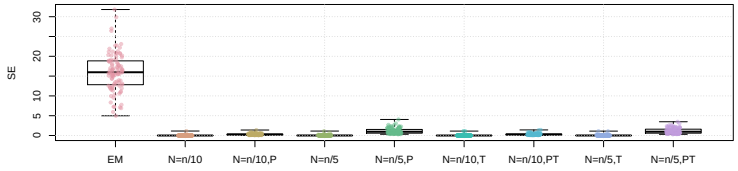
In the Iris1 and Iris2 studies, we observe that the mini-batch EM algorithms uniformly outperform the standard EM algorithm in terms of the log-likelihood, SE, and ARI. In all three measurements, we observe that $N = n/10$ performed better than $N = n/5$, and also that there were no differences between truncated versions of the mini-batch algorithms and equivalent variants without truncation. Polyak averaging appears to reduce the performance of the mini-batch



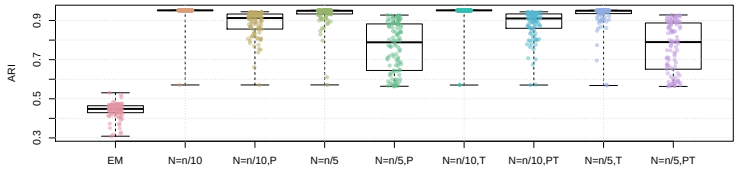
(a) Timing results, in seconds.



(b) Log-likelihood results.

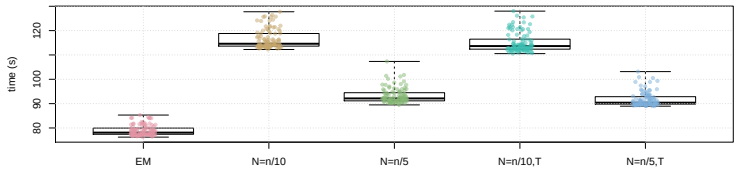


(c) Standard error results.

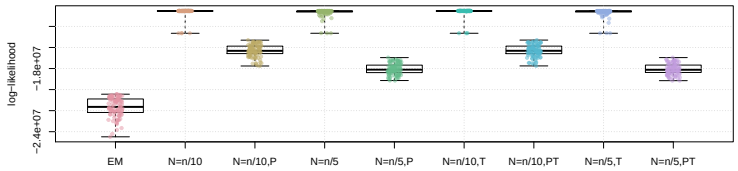


(d) Adjusted-Rand index results.

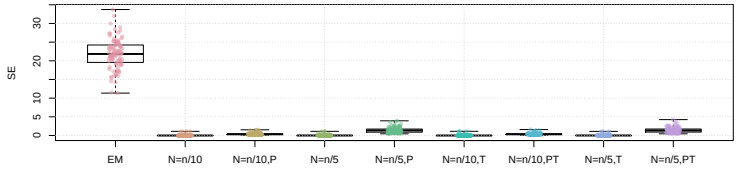
Figure 2: Results from Rep = 100 replications of the Iris1 simulation experiment. The 'EM' box plot summarizes the performance of the standard EM algorithm. The other plots are labelled by which variant of the mini-batch EM algorithm is summarized. The value of the batch size N is indicated (either $N = n/10$ or $N = n/5$), and a 'P' or a 'T' designates that Polyak averaging or truncation was used, respectively.



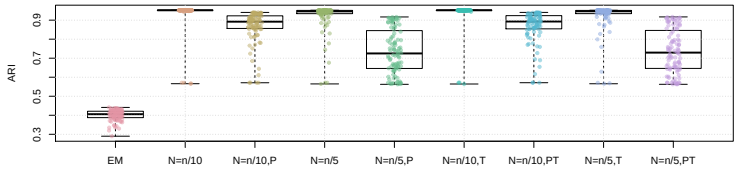
(a) Timing results, in seconds.



(b) Log-likelihood results.

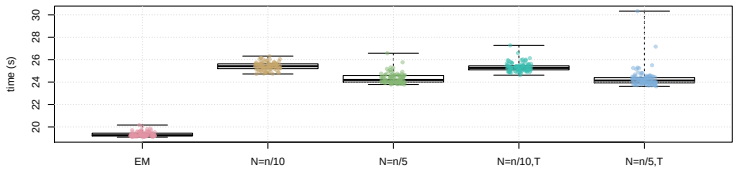


(c) Standard error results.

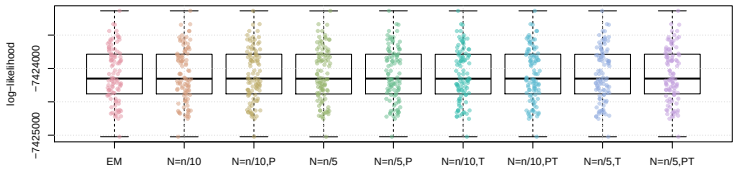


(d) Adjusted-Rand index results.

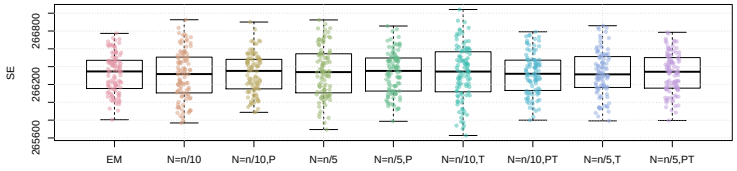
Figure 3: Results from Rep = 100 replications of the Iris2 simulation experiment. The 'EM' box plot summarizes the performance of the standard EM algorithm. The other plots are labelled by which variant of the mini-batch EM algorithm is summarized. The value of the batch size N is indicated (either $N = n/10$ or $N = n/5$), and a 'P' or a 'T' designates that Polyak averaging or truncation was used, respectively.



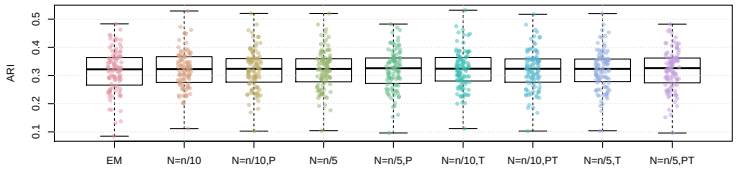
(a) Timing results, in seconds.



(b) Log-likelihood results.

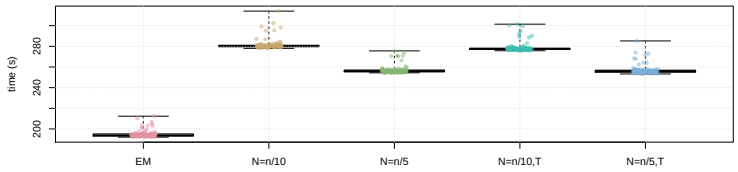


(c) Standard error results.

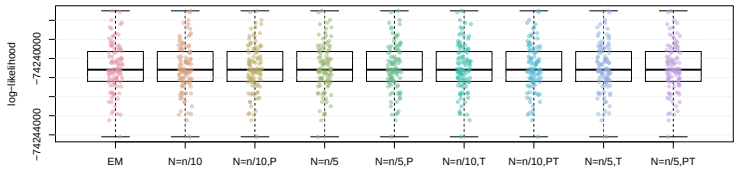


(d) Adjusted-Rand index results.

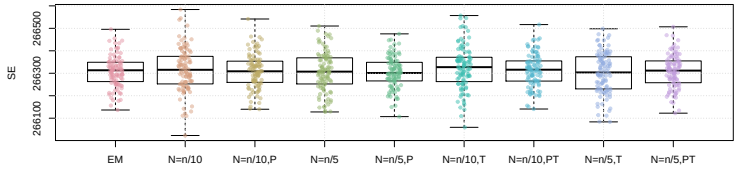
Figure 4: Results from Rep = 100 replications of the Wreath1 simulation experiment. The 'EM' box plot summarizes the performance of the standard EM algorithm. The other plots are labelled by which variant of the mini-batch EM algorithm is summarized. The value of the batch size N is indicated (either $N = n/10$ or $N = n/5$), and a 'P' or a 'T' designates that Polyak averaging or truncation was used, respectively.



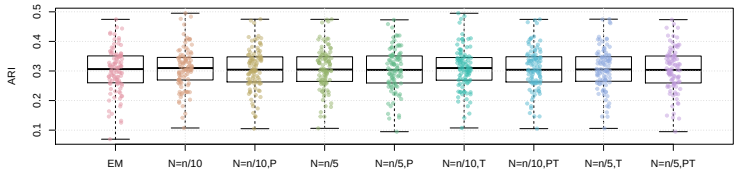
(a) Timing results, in seconds.



(b) Log-likelihood results.



(c) Standard error results.



(d) Adjusted-Rand index results.

Figure 5: Results from Rep = 100 replications of the Wreath2 simulation experiment. The 'EM' box plot summarizes the performance of the standard EM algorithm. The other plots are labelled by which variant of the mini-batch EM algorithm is summarized. The value of the batch size N is indicated (either $N = n/10$ or $N = n/5$), and a 'P' or a 'T' designates that Polyak averaging or truncation was used, respectively.

algorithms, for a given level of N, with respect to each of the three measurements.

In the Wreath1 and Wreath2 studies, we observe that the standard and mini-batch EM algorithms perform virtually the same across the log-likelihood, SE, and ARI metrics. This is likely due to the high degree of separability of each of the $g = 12$ mixture components of the Wreath data, in comparison to the overlapping components of the Iris data. Our observation is true for all of the mini-batch EM variants, with or without truncation or Polyak averaging. Upon first impression, this may appear as a weakness of the mini-batch EM algorithm, since it produces the same performance while requiring more computational time. However, we must also remember that the mini-batch EM algorithm does not require all of the data to be stored in memory at each iteration of algorithm, whereas the EM algorithm does. Thus, the mini-batch algorithm is feasible in very large data situations, since it only requires a fixed memory size, of order N, regardless of sample size n, whereas the standard EM algorithm has a memory requirement that increases with n.

To expand upon our currently presented results, we have also included a further two simulation studies regarding the fitting of finite mixtures of normal distributions using mini-batch EM algorithms. Our two studies are based on the Flea data of Wickham et al. (2011), a test scenario from the ELKI project of Schubert et al. (2015), and an original data generating process. The ELKI scenario is chosen due to its separability, in order to assess whether our conclusion regarding Wreath1 and Wreath2 are correct. The Flea data shares similarities with the Iris data but is higher dimensional. In all cases, we found that the mini-batch algorithms tended to outperform the EM algorithm in all but timing, on average. Detailed assessments of these studies can be found in the Supplementary Materials.

In addition, we have also investigated the use of the mini-batch EM algorithm for estimation

of non-normal mixture models. Namely, we present a pair of algorithms for the estimation of exponential and Poisson mixture models. We demonstrate their performance via an additional pair of simulation studies.

To conclude, we make the following recommendations. Firstly, smaller batch sizes appear to yield higher likelihood values. Secondly, averaging appears to slow convergence of the algorithm to the higher likelihood value and is thus not recommended. Thirdly, truncation appears to have no effect on the performance. This is likely due to the fact that truncation may not have been needed in any of the experiments. In any case, it is always useful to use the truncated version of the algorithm, in case there are unforeseen instabilities in the optimization process. And finally, the standard EM algorithm may be preferred to the mini-batch EM algorithm when sample sizes are small and when the data are highly separable. However, even in the face of high separability, for large n, it may not be feasible to conduct estimation by the standard EM algorithm and thus the mini-batch algorithms may be preferred due to feasibility.

It is interesting to observe that Polyak averaging tended to diminish the performance of the algorithms, in our studied scenarios. This is in contradiction to the theory that suggests that Polyak averaging should in fact increase the convergence rate to stationary solutions. We note, however, that the theory is asymptotic and the number of epochs that were used may be too short for the advantages of Polyak averaging to manifest, in practice.

5 Real data study

5.1 MNIST data

The MNIST data of LeCun et al. (1998) consists of $n = 70,000$ observations of $d = 28 \times 28 = 784$ pixel images of handwritten digits. These handwritten digits were sampled nearly uniformly. That is there were 6903, 7877, 6990, 7141, 6824, 6313, 6876, 7293, 6825, and 6958 observations of the digits 0–9, respectively.

Next, it is notable that not all d pixels are particularly informative. In fact, there is a great amount of redundancy in the d dimensions. Out of the d pixels, 65 are always zero, for every observation. Thus, the dimensions of the data are approximately 8.3% sparse.

We eliminate the spare pixels across all images to obtain a dense dimensionality of $d_{\text{dense}} = 719$. Using the d_{dense} dimensions of the data, we conduct a principal component analysis (PCA) in order to further reduce the data dimensionality; see Jolliffe, 2002 for a comprehensive treatment on PCA. Using the PCA, we extract the principal components (PCs) of each observation, and for various number of PCs $d_{\text{PC}} \in [d_{\text{dense}}]$. We can then use the data sets of n observations and dimension d_{PC}, to estimate mixture of normal distributions for various values of g.

5.2 Experimental setup

In the following study, we utilize only the truncated version of the mini-batch algorithm, having drawn the conclusions, from Section 4, that there appeared to be no penalty in performance due to truncation in practice. Again, drawing upon our experience from Section 4, we set $N = n/10 = 7000$ as the batch size in all applications. The same learning rate sequence of $\{\gamma_r\}_{r=1}^{\infty}$, where $\gamma_r = (1 - 10^{-10}) \times r^{6/10}$ is also used, and $c_1, c_2, c_3 = 1000$.

We apply the mini-batch algorithm to data with $d_{\text{PC}} = 10, 20, 50, 100$. Initialization of the parameter vector $\boldsymbol{\theta}^{(0)}$ was conducted via the randomization scheme of McLachlan & Peel (2000, Sec. 3.9.3). The mini-batch algorithm was run 100 times for each d_{PC} and the log-likelihood values were recorded for both the fitted models using the Polyak averaging and no averaging versions of the algorithm. The standard EM algorithm, as applied via the `em` function of the `mclust` package is again used for comparison. Each of the algorithms, including the k-means algorithm, were initialized from the same initial randomization, in the interest of fairness, for each of the 100 runs. That is, a random partition of the data is generated once for each of the 100 runs, and the initial parameters for the EM, mini-batch and k-means algorithms are all computed from the same initialization. The log-likelihood values of the standard and mini-batch EM algorithms are compared along with the ARI values. Algorithms are run for 10 epochs.

We compute the ARI values obtained when comparing the maximum *a posteriori* clustering labels, obtained from each of the algorithms (cf. McLachlan & Peel, 2000, Sec. 1.15), and the true digit classes of each of the images. For a benchmark, we also compare the performance of the three EM algorithms with the k-means algorithm, as applied via the `kmeans` function in `R`, which implements the algorithm of Hartigan & Wong (1979). For fairness of comparison, we also allow the k-means algorithm 10 epochs in each of 100 runs. As in Section 4, we note that all codes are available at `https://github.com/hiendn/StoEMMIX`, for the sake of reproducibility and transparency.

5.3 Results

The results from the MNIST experiment are presented in Table 1. We observe that for $d_{\text{PC}} \in \{10, 20, 50\}$, all three EM variants provided better ARI than the k-means algorithm. The best

ARI values for all three EM algorithms occur when $d_{\text{PC}} = 20$. When $d_{\text{PC}} = 100$, the k-means algorithm provided a better ARI, which appeared to be somewhat uniform across the four values of d_{PC}.

Among the EM algorithms, the mini-batch algorithm provided better ARI values, with the two variants not appearing to be significantly different from one another, when considering the standard errors of the ARI values, when $d_{\text{PC}} \in \{20, 50\}$. When $d_{\text{PC}} = 10$, we observe that no averaging yielded a better ARI, whereas, when $d_{\text{PC}} = 100$, averaging appeared to be better, on average.

Regarding the log-likelihoods, the mini-batch EM algorithm, when applied without averaging, uniformly and significantly outperformed the standard EM algorithm. On the contrary, when applied with averaging, the EM algorithm uniformly and significantly outperformed the mini-batch algorithm. This is also in contrary with what was observed in Section 4. This is an interesting result considering that the ARI of the mini-batch algorithm, with averaging, is still better than that of the EM algorithm. As in Section 4, we can recommend the use of the mini-batch EM algorithm without averaging, as it tends to outperform the standard EM algorithm for fit and is also yields better clustering outcomes, when measured via the ARI.

6 Conclusions

In Section 2, we reviewed the online EM algorithm framework of Cappé & Moulines (2009), and stated the key theorems that guarantee the convergence of algorithms that are constructed under the online EM framework. We then presented a novel interpretation of the online EM algorithm that yielded our framework for constructing mini-batch EM algorithms. We then utilized the

Table 1: Tabulation of results from the 100 runs of the EM algorithms and the k-means algorithm, for each value of $d_{\text{PC}} \in \{10, 20, 50, 100\}$. The columns EM, Mini, and Mini Pol refer to the standard EM, the mini-batch EM, and the mini-batch EM algorithm with Polyak averaging, respectively. The SE rows contain the standard error over each of the 100 runs (i.e. the standard deviation over 10). Boldface text highlight the best results.

		ARI				log-likelihood		
d_{PC}		EM	Mini	Mini Pol	k-means	EM	Mini	Mini Pol
10	Mean	0.401	**0.443**	0.432	0.352	-4.98E+06	**-4.96E+06**	-5.01E+06
	SE	0.004	0.004	0.004	0.002	1.19E+03	6.43E+02	5.90E+02
20	Mean	0.436	0.475	**0.480**	0.367	-9.46E+06	**-9.44E+06**	-9.52E+06
	SE	0.005	0.005	0.005	0.002	2.20E+03	1.37E+03	1.67E+03
50	Mean	0.394	0.434	**0.438**	0.369	-2.18E+07	**-2.17E+07**	-2.20E+07
	SE	0.005	0.005	0.006	0.002	8.47E+03	7.01E+03	4.85E+03
100	Mean	0.326	0.356	**0.377**	0.372	-3.99E+07	**-3.97E+07**	-4.05E+07
	SE	0.004	0.004	0.005	0.002	1.83E+04	1.68E+04	8.72E+03

theorems of Cappé & Moulines (2009) in order to produce convergence results for this new mini-batch EM algorithm framework. Extending upon some remarks of Cappé & Moulines (2009), we also made rigorous the use of truncation in combination with both the online EM and mini-batch EM algorithm frameworks, using the construction and theory of Delyon et al. (1999).

In Section 3, we demonstrated how the mini-batch EM algorithm framework could be applied to construct algorithms for conducting ML estimation of finite mixtures of exponential family distributions. A specific analysis is made of the particularly interesting case of the normal mixture models. Here, we validate the conditions that permit the use of the Theorems from Section 2 in order to guarantee the convergence of the mini-batch EM algorithms for ML estimation of normal mixture models.

In Section 4, we conducted a set of four simulation studies in order to study the performance of the mini-batch EM algorithms, implemented in eight different variants, as compared to the standard EM algorithm for ML estimation of normal mixture models. There, we found that

regardless of implementation, in many cases, the mini-batch EM algorithms were able to obtain log-likelihood values that were better on average than the standard EM algorithm. We also found that the use of larger batch sizes and Polyak averaging tended to diminish performance of the mini-batch algorithms, but the use of truncation tended to have no effect. Although the mini-batch algorithms is generally slower than the standard EM algorithm, we note that in many cases, the fixed memory requirement of the mini-batch algorithms make them feasible where the standard EM algorithm is not.

A real data study was conducted in Section 5. There, we explored the use of the standard EM algorithm and the truncated mini-batch EM algorithm for cluster analysis of the famous MNIST data of LeCun et al. (1998). From our study, we found that the mini-batch EM algorithm was able to obtain better log-likelihood values than the standard EM algorithm, when applied without Polyak averaging. However, with averaging, the mini-batch EM algorithm was worse than the standard EM algorithm, on average. However, regardless of whether averaging was used, or not, the mini-batch EM algorithm appeared to yield better clustering outcomes, when measured via the ARI of Hubert & Arabie (1985).

This research poses numerous interesting directions for the future. First, we may extend the results to other exponential family distributions that permit the satisfaction of theorem assumptions from Section 2. We make initial steps in this direction via a pair of mini-batch algorithms for exponential and Poisson distribution mixtures. Secondly, we may use the framework to construct mini-batch algorithms for large-scale mixture of regression models (cf. Jones & McLachlan, 1992), following the arguments made by Cappé & Moulines (2009) that permitted them to construct an online EM algorithm for their mixture of regressions example analysis. Thirdly, this research theme can be extended further to the construction of mini-batch algorithms for mixture of experts models

(cf. Nguyen & Chamroukhi, 2018), which may be facilitated via the Gaussian gating construction of Xu et al. (1995).

In addition to the three previous research questions, we may also ask questions regarding the practical application of the mini-batch algorithms. For instance, we may consider the question of optimizing learning rates and batch sizes for particular application settings. Furthermore, we may consider whether the theoretical framework still applies to algorithms where we may have adaptive batch sizes and learning rate regimes. As these directions fall vastly outside the scope of the current paper, we shall leave them for future exploration.

Acknowledgements

The authors are indebted to the Co-ordinating Editor and two Reviewers for their insightful comments that have improved the exposition of the manuscript. HDN is personally funded by Australian Research Council (ARC) grant DE170101134. GJM and HDN are also funded under ARC grant DP180101192. The work is supported by Inria project LANDER.

References

Amari, S. (2016). *Information Geometry and Its Applications*. Japan: Springer.

Bouveyron, C., Girard, S., & Schmid, C. (2007). High-dimensional data clustering. *Computational Statistics and Data Analysis*, 52, 502–519.

Buhlmann, P., Drineas, P., Kane, M., & van der Laan, M., Eds. (2016). *Handbook of Big Data*. Boca Raton: CRC Press.

Cappé, O. & Moulines, E. (2009). On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society B*, 71, 593–613.

Celeux, G., Chretien, S., Forbes, F., & Mkhadri, A. (2001). A component-wise EM algorithm for mixtures. *Journal of Computational and Graphical Statistics*, 10, 697–712.

Chau, M. & Fu, M. C. (2015). An overview of stochastic approximation. In M. C. Fu (Ed.), *Handbook of Simulation Optimization* (pp. 149–178). New York: Springer.

Chen, H.-F. (2003). *Stochastic Approximiation and Its Applications*. New York: Kluwer.

Cotter, A., Shamir, O., Srebro, N., & Sridharan, K. (2011). Better mini-batch algorithms via accelerated gradient methods. In *Adavances in Neural Information Processing Systems* (pp. 1647–1655).

DasGupta, A. (2011). *Probability for Statistics and Machine Learning*. New York: Springer.

Delyon, B., Lavielle, M., & Moulines, E. (1999). Counvergence of a stochastic approximation version of the EM algorithm. *Annals of Statistics*, 27, 94–128.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 39, 1–38.

Eddelbuettel, D. (2013). *Seamless R and C++ Integration with Rcpp*. New York: Springer.

Fisher, R. A. (1936). The use of multiple measurments in taxonomic problems. *Annals of Eugenics*, (pp. 179–188).

Forbes, C., Evans, M., Hastings, N., & Peacock, B. (2011). *Statistical Distributions*. New York: Wiley.

Fraley, C., Raftery, A., & Wehrens, R. (2005). Incremental model-based clustering for large datasets with small clusters. *Journal of Computation and Graphical Statistics*, 14, 529–546.

Ghadimi, S., Lan, G., & Zhang, H. (2016). Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming Series A*, 155, 267–305.

Han, Z., Hong, M., & Wang, D. (2017). *Signal Processing and Networking for Big Data Applications*. Cambridge: Cambridge University Press.

Hardle, W. K., Lu, H. H.-S., & Shen, X., Eds. (2018). *Handbook of Big Data Analytics*. Cham: Springer.

Hartigan, J. A. & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society Series C*, 28, 100–108.

Hubert, L. & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2, 193–218.

Iverson, K. E. (1967). *A Programming Language*. New York: Wiley.

Jolliffe, I. T. (2002). *Principal Component Analysis*. New York: Springer.

Jones, P. N. & McLachlan, G. J. (1992). Fitting finite mixture models in a regression context. *Australian Journal of Statistics*, 34, 233–240.

Kiefer, J. & Wolfowitz, J. (1952). Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics*, 23, 462–466.

Kullback, S. & Leibler, R. A. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, 22, 79–86.

Kushner, H. J. & Yin, G. G. (2003). *Stochastic Approximiation and Recursive Algorithms and Applications*. New York: Springer.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86, 2278–2324.

Li, M., Zhang, T., Chen, Y., & Smola, A. J. (2014). Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 661–670).

Liang, F. & Zhang, J. (2008). Estimating the false discovery rate using the stochastic approximation algorithm. *Biometrika*, 95, 961–977.

McLachlan, G. J. & Krishnan, T. (2008). *The EM Algorithm And Extensions*. New York: Wiley.

McLachlan, G. J., Lee, S. X., & Rathnayake, S. I. (2019). Finite mixture models. *Annual Review of Statistis and Its Application*. In press.

McLachlan, G. J. & Peel, D. (2000). *Finite Mixture Models*. New York: Wiley.

Melnykov, V., Chen, W.-C., & Maitra, R. (2012). MixSim: an R package for simulating data to study performance of clustering algorithms. *Journal of Statistical Software*, 51, 1–25.

Ng, S.-K. & McLachlan, G. J. (2004). Speeding up the EM algorithm for mixture model-based segmentation of magnetic resonance images. *Pattern Recognition*, 37, 1573–1589.

Nguyen, H. D. & Chamroukhi, F. (2018). Practical and theoretical aspects of mixture-of-experts modeling: an overview. *WIREs Data Mining and Knowledge Discovery*, (pp. e1246).

Nguyen, H. D. & Jones, A. T. (2018). Big Data-appropriate clustering via stochastic approximation and Gaussian mixture models. In M. Ahmed & A.-S. K. Pathan (Eds.), *Data Analytics: Concepts, Techniques, and Applications*. Boca Raton: CRC Press.

Nguyen, H. D. & McLachlan, G. J. (2015). Maximum likelihood estimation of Gaussian mixture models without matrix operations. *Advances in Data Analysis and Classification*, 9, 371–394.

Pearson, K. (1894). Contributions to the theory of mathematical evolution. *Philosophical Transactions of the Royal Society of London A*, 185, 71–110.

Polyak, B. T. (1990). A new method of stochastic approximation type. *Automatic and Remote Control*, 51, 98–107.

Polyak, B. T. & Juditsky, A. B. (1992). Acceleration of stochastic approximation by averaging. *SIAM Journal of Control and Optimization*, 30, 838–855.

Prosperetti, A. (2011). *Advanced Mathematics for Applications*. Cambridge: Cambridge University Press.

R Core Team (2018). *R: a language and environment for statistical computing*. R Foundation for Statistical Computing.

Robbins, H. & Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, 22, 400–407.

Schubert, E., Koos, A., Emrich, T., Zufle, A., Schmid, K. A., & Zimek, A. (2015). A framework for clustering uncertain data. *Proceedings of the VLDB Endowment*, 8, 1976–1979.

Scrucca, L., Fop, M., Murphy, T. B., & Raftery, A. E. (2016). mclust: clustering, classification and density estimation using Gaussian finite mixture models. *R Journal*, 8, 289–317.

Vlassis, N. & Likas, A. (2002). A greedy EM algorithm for Gaussian mixture learning. *Neural Processing Letters*, 15, 77–87.

White, H. (1982). Maximum likelihood estimation of misspecified models. *Econometrica*, 50, 1–25.

White, H. (2001). *Asymptotic Theory For Econometricians*. San Diego: Academic Press.

Wickham, H., Cook, D., Hofmann, H., & Buja, A. (2011). tourr: an R package for exploring multivariate data with projections. *Journal of Statistical Software*, 40, 1–18.

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

Xu, L., Jordan, M. I., & Hinton, G. E. (1995). An alternative model for mixtures of experts. In *Advances in Neural Information Processing Systems* (pp. 633–640).

Zhang, J. & Liang, F. (2008). Convergence of stochastic approximation algorithms under irregular conditions. *Statistica Neerlandica*, 62, 393–403.

Zhao, T., Yu, M., Wang, Y., Arora, R., & Liu, H. (2014). Accelerated mini-batch randomized block coordinate descent method. In *Advances in Neural Information Processing Systems* (pp. 3329–3337).