Journal of Machine Learning Research 23 (2022) 1-40 Submitted 1/21; Revised 2/22; Published 2/22

Distributed Learning of Finite Gaussian Mixtures

Qiong Zhang QIONG.ZHANG@STAT.UBC.CA
Department of Statistics
University of British Columbia
Vancouver, BC V6T 1Z4, Canada

Jiahua Chen JHCHEN@STAT.UBC.CA
Department of Statistics
University of British Columbia
Vancouver, BC V6T 1Z4, Canada

Editor: Sathiya Keerthi

Abstract

Advances in information technology have led to extremely large datasets that are often kept in different storage centers. Existing statistical methods must be adapted to overcome the resulting computational obstacles while retaining statistical validity and efficiency. In this situation, the split-and-conquer strategy is among the most effective solutions to many statistical problems, including quantile processes, regression analysis, principal eigenspaces, and exponential families. This paper applies this strategy to develop a distributed learning procedure of finite Gaussian mixtures. We recommend a reduction strategy and invent an effective majorization-minimization algorithm. The new estimator is consistent and retains root-n consistency under some general conditions. Experiments based on simulated and real-world datasets show that the proposed estimator has comparable statistical performance with the global estimator based on the full dataset, if the latter is feasible. It can even outperform the global estimator for the purpose of clustering if the model assumption does not fully match the real-world data. It also has better statistical and computational performance than some existing split-and-conquer approaches.

Keywords: Barycenter, Gaussian mixture, Global convergence, Mixture reduction, MM algorithm, Model-based clustering, Split-and-conquer.

1. Introduction

In the era of big data, the sizes of the datasets for various applications may be so large that they cannot be stored on a single machine. According to Corbett et al. (2013), Google distributes its huge database around the world. Even if the dataset is stored on a single machine, it can be difficult to load the whole dataset into the computer memory. Distributed data storage is also natural when the datasets are collected and managed by independent agencies. Examples include patient information collected from different hospitals and data collected by different government agencies (Agrawal et al., 2003). Privacy considerations may make it difficult or even impossible to pool the separate collections of data into a single dataset. Data analysis methods in such applications should therefore be designed so that they can work with subsets of the dataset, in parallel or sequentially. The information extracted from the subsets may then be combined to draw conclusions about the whole

population. Under distributed data storage, a two-step split-and-conquer procedure is often used for inference:

(i) Local inference: standard inference is carried out on local machines;

(ii) Aggregation: the local results are transmitted to a central machine and aggregated.

The split-and-conquer approach addresses privacy concerns by sharing only summary statistics across machines. It is also communication efficient and avoids the potentially high transmission cost that may exceed the computational cost (Jaggi et al., 2014).

A variety of split-and-conquer approaches have been developed. For example, the split-and-conquer learning of the generalized linear models (Chen and Xie, 2014), kernel ridge regression models (Zhang et al., 2015), ordinary linear models (Chang et al., 2017), and the split-and-conquer estimation of principal eigen-spaces (Fan et al., 2019). Also see the split-and-conquer version of the Wald and score tests (Battey et al., 2015). Most existing approaches first obtain one local estimate of the model parameters based on per local dataset. These local estimates are then pooled and aggregated through a linear averaging operation.

This paper focuses on developing split-and-conquer approaches for finite Gaussian mixture models (GMMs), which is widely used for model-based clustering (Murphy, 2012). Learning of mixture models is a well studied problem (McLachlan and Peel, 2004; Frühwirth-Schnatter, 2006; Mengersen et al., 2011; Balakrishnan et al., 2017; Chen et al., 2020; Heinrich and Kahn, 2018; Dwivedi et al., 2020) while the relatively new distributed learning of finite mixtures remains to be investigated. We usually learn finite mixture models via the maximum likelihood estimate (MLE) through a well-established EM algorithm (Dempster et al., 1977; Wu, 1983). Distributed versions of the EM algorithm such as Nowak (2003), Safarinejadian et al. (2010), and Chen et al. (2013) are also developed when datasets are stored in a distributed fashion. However, these approaches require the communication of summary statistics and the coordination across local machines at each iteration. Another approach is to first condense the data on local machines into much smaller coresets (Lucic et al., 2017) and then refit the mixture to the pooled coresets at a central machine. The coreset approach achieves high computational efficient at the cost of statistical efficiency, and privacy due to the transmission of raw data. The split-and-conquer based approaches also have their challenges under finite mixtures. The parameter of finite mixture, the mixing distribution, is a discrete distribution with a finite number of support points. The popular linear averaging operation of the local estimates results in a mixing distribution with an inflated number of support points. The aggregated mixture hence has redundant and spurious subpopulations. The KL divergence based aggregation approach of Liu and Ihler (2014) achieves the best efficiency under models from exponential family, but it is less efficient under GMMs. We therefore address the important problem of *developing an aggregation procedure with high computational and statistical efficiencies under GMMs.*

In this paper, we learn the finite Gaussian mixtures via the penalized MLE during the local inference. We explore two potential aggregation approaches and find that a specific reduction approach has satisfactory statistical and computational efficiencies. During the aggregation step, the reduction method first pools the local estimates to form a mixture model with clearly an excessive number of subpopulations. We then search for a mixture with a designated order that is optimal according to some divergence criteria. We identify a

divergence metric based on the transportation distance so that the corresponding estimator retains statistical efficiency and computational simplicity. We also develop an majorization-minimization algorithm for the numerical computation.

The remainder of the paper is structured as follows. In Section 2, we introduce finite Gaussian mixture models and the transportation distance. In Section 3, we study two split-and-conquer methods and recommend a specific one. In Section 4, we describe an algorithm for computing the recommended estimator. In Section 5, we show that the proposed estimator has the best possible statistical convergence rate under some generic conditions. In Section 6, we review existing approaches for distributed learning of finite Gaussian mixtures and their pros and cons compared with our proposed method. Numerical experiments on simulated, public, and real datasets are presented in Section 7. In all cases, the proposed approach performs well. Section 8 contains some discussion and concluding remarks. The technical details are given in the appendix.

2. Preliminaries on Finite Gaussian Mixtures and Divergences

In this section, we introduce finite Gaussian mixture models, EM algorithm, and the transportation distances between mixing distributions or between mixtures.

2.1 Finite Gaussian Mixtures

A statistical model is a family of distributions. In many applications, a dataset can be regarded as a random sample from a population, and the population is a member of some parametric distribution family. For instance, a Gaussian distribution family is often assumed in applications. When any single distribution in a parametric family fails to properly model the data, it may be caused by the heterogeneity in the population. For instance, the population is made of several subpopulations so that each subpopulation has a distinct distribution belonging to the same parametric family. Then each observation in the dataset is a random outcome from a subpopulation. The subpopulation, or the membership of the observation, is itself random. Without the knowledge of subpopulation memberships of the individual units, the data form a random sample from a mixture distribution. Mathematically, a finite mixture model is defined as follows.

Definition 1 (Finite Mixture Model) *Let $\mathcal{F} = \{f(\cdot;\theta) : \theta \in \Theta\}$ be a parametric family of density functions with respect to some σ-finite measure, where Θ is some parameter space. Let $G = \sum_{k=1}^{K} w_k \delta_{\theta_k}$ be a discrete probability measure, assigning probability w_k to parameter value θ_k on Θ for some integer $K > 0$. The finite mixture distribution of $\mathcal{F}$ with mixing distribution G has density function*

$$f(x;G) := \int f(x;\theta)\, dG(\theta) = \sum_{k=1}^{K} w_k f(x;\theta_k).$$

The finite mixture model of $\mathcal{F}$ is the distribution family with densities in the form of (1).

The parameter space Θ is usually a subset of d-dimensional Euclidean space $\mathbb{R}^d$ for some integer $d \geq 1$. In addition, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)^\top$ is a vector of subpopulation parameters. We

call $f(\cdot;\theta_k)$s subpopulation density functions. Let the $K-1$ dimensional simplex be

$$\Delta_{K-1} = \left\{ (w_1, \ldots, w_K) : w_i \geq 0, \sum_{i=1}^{K} w_i = 1 \right\}.$$

The components of the vector $\boldsymbol{w} = (w_1, w_2, \ldots, w_K)^\top \in \Delta_{K-1}$ are called the mixing weights. We use $F(x;\theta)$ and $F(x;G)$ respectively for the cumulative distribution functions (CDFs) of $f(x;\theta)$ and $f(x;G)$. The space of mixing distributions of order up to K is denoted

$$\mathbb{G}_K = \left\{ G = \sum_{k=1}^{K} w_k \delta_{\theta_k}, \boldsymbol{w} \in \Delta_{K-1}, \boldsymbol{\theta} \in \Theta^K \right\}. \tag{1}$$

An order K mixture has its mixing distribution in $\mathbb{G}_K - \mathbb{G}_{K-1}$. When the subpopulation distribution is a multivariate Gaussian, we have a finite Gaussian mixture and the corresponding Gaussian mixture model (GMM).

Example 1 (Finite Gaussian Mixtures) *Let the density function of a d-dimensional Gaussian random variable with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ be*

$$\phi(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \det\{2\pi\boldsymbol{\Sigma}\}^{-1/2} \exp\left\{ -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \right\},$$

and let $\Phi(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma})$ be its CDF. A Gaussian mixture of order K is a distribution with density and CDF given by

$$\phi(\boldsymbol{x};G) = \sum_{k=1}^{K} w_k \phi(\boldsymbol{x};\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k); \quad \Phi(\boldsymbol{x};G) = \sum_{k=1}^{K} w_k \Phi(\boldsymbol{x};\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k).$$

Note the mixing distribution G is a discrete distribution on Θ, which is the space of the d-dimensional vector $\boldsymbol{\mu}$ and $d \times d$ positive definite matrix $\boldsymbol{\Sigma}$.

The earliest example of GMM in Pearson (1894) suggests that the skewness displayed in the histogram of the biometrics of $1,000$ crabs is due to the presence of two non-identified species. There are also applications of mixture in finance. Suppose the stock prices in the hidden "normal" or "extreme" periods have different Gaussian distributions. The marginal distribution is then a Gaussian mixture of order 2 (Liesenfeld, 2001). For general theory and applications of the finite mixture models, see McLachlan and Peel (2004) and Frühwirth-Schnatter (2006).

The most popular learning approach for finite mixture models is MLE. Under finite GMMs, the likelihood is however unbounded, and the MLE is not well defined. Let $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ be observed values on a set of independent and identically distributed (IID) random vectors with a finite Gaussian mixture distribution $\phi(\boldsymbol{x};G)$ of order K. The log-likelihood function of G is given by

$$\ell_n(G|\mathcal{X}) = \sum_{i=1}^{n} \log \phi(\boldsymbol{x}_i;G) = \sum_{i=1}^{n} \log \Big\{ \sum_{k=1}^{K} w_k \phi\big(\boldsymbol{x}_i;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k\big) \Big\}.$$

This log-likelihood function is unbounded: its value goes to infinity for a specific combination of $\boldsymbol{\mu}_k$ and some degenerating $\boldsymbol{\Sigma}_k$ (Chen and Tan, 2009). To solve this issue, Chen and Tan (2009) shows the log-likelihood function can be regularized by adding a penalty function on the subpopulation covariance matrices. The maximizer of the penalized likelihood is well-defined and is a consistent estimator. Let S_x be the sample covariance matrix. Chen and Tan (2009) recommends the following penalized log-likelihood function

$$p\ell_n(G|\mathcal{X}) = \ell_n(G|\mathcal{X}) - a_n \sum_k \big\{\mathrm{tr}(S_x\boldsymbol{\Sigma}_k^{-1}) + \log\det(\boldsymbol{\Sigma}_k)\big\}$$

for some positive a_n that is allowed to depend on n, and they learn the mixing distribution G via

$$\widehat{G}_n := \arg\sup p\ell_n(G|\mathcal{X}). \tag{2}$$

The size of a_n controls the strength of the penalty, and a recommended value is $n^{-1/2}$. Regularizing the likelihood function via a penalty function fixes the problem caused by degenerated $\boldsymbol{\Sigma}_k$. We call $\widehat{G}_n$ in (2) pMLE and it is strongly consistent when the order has a known upper bound.

The specific form of the penalty function mimics a Wishart prior on the subpopulation precision matrices. It allows an easy generalization of the EM algorithm for the numerical computation and a consistency proof (Chen and Tan, 2009). Other forms of the penalty function should be possible but unlikely to markedly outperform the current one. We describe the EM algorithm for numerical computation in the next section.

2.2 EM Algorithm for Computing pMLE

Let $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \overset{\text{i.i.d.}}{\sim} \phi(\cdot; G)$ and the sample covariance matrix is S_x. Recall that each observation in the dataset can be regarded as a sample from a subpopulation $\phi(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ with probability w_k. We hence introduce a membership vector $\boldsymbol{z}_i = (z_{i1}, \ldots, z_{iK})$ for the ith unit. The kth entry of $\boldsymbol{z}_i$ is 1 when the response value $\boldsymbol{x}_i$ is an observation from the kth subpopulation, and 0 otherwise. When the complete data $\{(\boldsymbol{z}_i, \boldsymbol{x}_i), i = 1, 2, \ldots, n\}$ are available, we can construct a penalized complete likelihood

$$p\ell_n^{\mathrm{c}}(G) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}\log\{w_k\phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\} - a_n\sum_k \big\{\mathrm{tr}(S_x\boldsymbol{\Sigma}_k^{-1}) + \log\det(\boldsymbol{\Sigma}_k)\big\}.$$

Given the full dataset $\mathcal{X}$ and a proposed mixing distribution $G^{(t)}$, we may compute the conditional expectation

$$w_{ik}^{(t)} = \mathbb{E}(z_{ik}|\mathcal{X}, G^{(t)}) = \frac{w_k^{(t)}\phi(\boldsymbol{x}_i; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{j=1}^{K} w_j^{(t)}\phi(\boldsymbol{x}_i; \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)})}. \tag{3}$$

With this, we define

$$\begin{aligned} Q(G; G^{(t)}) \;=\; & \sum_{k=1}^{K}\big\{\sum_{i=1}^{n} w_{ik}^{(t)}\big\}\log w_k - \frac{1}{2}\sum_{k=1}^{K}\big(2a_n + \sum_{i=1}^{n} w_{ik}^{(t)}\big)\log\det(\boldsymbol{\Sigma}_k) \\ & -\frac{1}{2}\sum_{k=1}^{K}\big\{2a_n\mathrm{tr}\big\{\boldsymbol{\Sigma}_k^{-1}S\big\} + \sum_{i=1}^{n} w_{ik}^{(t)}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^\top\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)\big\}. \end{aligned} \tag{4}$$

Note that the subpopulation parameters are separated in $Q(\cdot;\cdot)$. We refer to the procedure starting from $G^{(t)}$ to the definition of $Q(G;G^{(t)})$ as the **E-step** of the algorithm. The **M-step** of the algorithm maximizes $Q(G;G^{(t)})$ with respect to G. Let

$$S_k^{(t+1)} = \sum_{i=1}^{n} w_{ik}^{(t)} (\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(t+1)})(\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(t+1)})^\top.$$

The solution of the M-step has an explicit expression with the mixing distribution $G^{(t+1)}$ given by

$$\begin{cases} w_k^{(t+1)} & = n^{-1} \sum_{i=1}^n w_{ik}^{(t)}, \\ \boldsymbol{\mu}_k^{(t+1)} & = \{n w_k^{(t+1)}\}^{-1} \sum_{i=1}^n w_{ik}^{(t)} \boldsymbol{x}_i, \\ \boldsymbol{\Sigma}_k^{(t+1)} & = \{2a_n + n w_k^{(t+1)}\}^{-1} \{2a_n S_x + S_k^{(t+1)}\}. \end{cases}$$

The adapted EM algorithm iterates between E and M steps until some convergence criterion is satisfied and $G^{(t)}$ in the latest iteration is regarded as the pMLE. Like its MLE counterpart, $p\ell_n(G^{(t)})$ is an increasing sequence in t. We say $A \geq B$ if $A - B$ is semi-positive definite for any two square matrices A and B. In this sense, $\boldsymbol{\Sigma}_k^{(t+1)} \geq \{2a_n/(n+2a_n)\} S_x$ and the latter one does not dependent on t. Hence, the increasing sequence $\{p\ell_n(G^{(t)}), t = 1, 2, \ldots\}$ also has a finite upper bound. Therefore, $p\ell_n(G^{(t)})$ is guaranteed to converge and the limiting points of $G^{(t)}$ are non-degenerate local maxima as $t \to \infty$, following additional mathematical justifications of Wu (1983).

2.3 Divergence and Distance between Probability Measures

To develop split-and-conquer methods for learning finite mixture models, we need a measure of the closeness between mixing distributions or between mixtures. We first introduce the divergence and distance.

Definition 2 (Divergence and Distance) *Let Θ be a space. A bivariate function $\rho(\cdot,\cdot)$ defined on Θ is a divergence if $\rho(\theta_1,\theta_2) \geq 0$, with equality holds if and only if $\theta_1 = \theta_2$.*

Suppose that in addition $\rho(\cdot,\cdot)$ satisfies (i) symmetry, i.e., $\rho(\theta_1,\theta_2) = \rho(\theta_2,\theta_1)$, and (ii) the triangle inequality, i.e., $\rho(\theta_1,\theta_2) \leq \rho(\theta_1,\theta_3) + \rho(\theta_3,\theta_2)$, for all $\theta_1,\theta_2,\theta_3 \in \Theta$. Then $\rho(\cdot,\cdot)$ is a distance on Θ. When $\rho(\cdot,\cdot)$ is a distance, we call (Θ,ρ) a metric space.

We need divergences or distances specifically defined on the space of probability measures. The transportation divergence (Villani, 2003) is particularly useful. Let $\mathcal{P}(\Theta)$ and $\mathcal{P}(\Theta^2)$ be the spaces of probability measures on Θ and $\Theta \times \Theta$, respectively, equipped with some compatible σ-algebras. For any $\boldsymbol{\pi} \in \mathcal{P}(\Theta^2)$, let its marginal measures be $\boldsymbol{\pi}_{1,\cdot}$ and $\boldsymbol{\pi}_{\cdot,2}$. For any $\eta, \nu \in \mathcal{P}(\Theta)$, we define a space of distributions on $\Theta \times \Theta$ as follows:

$$\Pi(\eta,\nu) = \{\boldsymbol{\pi} \in \mathcal{P}(\Theta^2) : \boldsymbol{\pi}_{1,\cdot} = \eta,\ \boldsymbol{\pi}_{\cdot,2} = \nu\}.$$

The space $\Pi(\eta,\nu)$ is usually referred to as the coupling between η and ν, and it consists of bivariate distributions for which the marginal distributions are η and ν. For convenience, we use $\Pi(\eta,\cdot)$ for the space of distributions with first marginal distribution being η, and similarly for $\Pi(\cdot,\nu)$. We regard the mixing weights $\boldsymbol{w}$ as a distribution on Θ when appropriate.

Definition 3 (Transportation Divergence and r-Wasserstein distance) *Let $c(\cdot,\cdot)$ be a non-negative valued bivariate function on $\Theta \times \Theta$ satisfying $c(\theta,\theta) = 0$ for all $\theta \in \Theta$. The transportation divergence from η to ν is defined to be*

$$\mathcal{T}_c(\eta,\nu) = \inf\{\mathbb{E}_{\boldsymbol{\pi}}[c(X,Y)] : \boldsymbol{\pi} \in \Pi(\eta,\nu)\}.$$

where $\mathbb{E}_{\boldsymbol{\pi}}\{c(X,Y)\}$ is the expectation calculated when the joint distribution of X, Y is $\boldsymbol{\pi}$.

If $c(\cdot,\cdot) = D^r(\cdot,\cdot)$ for some $r \geq 1$ and distance $D(\cdot,\cdot)$ on Θ, then $\mathbb{W}_D = \mathcal{T}_c^{1/r}(\cdot,\cdot)$ is called the r-Wasserstein distance with the ground distance $D(\cdot,\cdot)$.

The infimum in the above definition is with respect to $\boldsymbol{\pi}$ over $\Pi(\eta,\nu)$. The same convention may be used subsequently. The joint distribution $\boldsymbol{\pi}$ that minimizes $\mathbb{E}_{\boldsymbol{\pi}}\{c(X,Y)\}$ is called the optimal transportation plan and is usually denoted $\boldsymbol{\pi}^*$. It is the most efficient plan that transports mass distributed according to η to mass distributed according to ν.

Definition 4 (Barycenter) *Let $(\mathcal{P}(\Theta),\rho)$ be a space of probability measures on Θ that is endowed with the divergence $\rho(\cdot,\cdot)$. Given positive constants $(\lambda_1,\lambda_2,\ldots,\lambda_M) \in \Delta_{M-1}$, the (weighted) barycenter of $\nu_1,\ldots,\nu_M \in \mathcal{P}(\Theta)$ is a minimum point of $\sum_{m=1}^M \lambda_m \rho(\nu_m,\eta)$.*

We allow $\rho(\cdot,\cdot)$ to be any divergence function. When $\rho(\cdot,\cdot) = D^r(\cdot,\cdot)$, it becomes the usual r-Wasserstein barycenter (Cuturi and Doucet, 2014). Conceptually, a barycenter should always be referred to together with information on ρ and the weights $(\lambda_1,\lambda_2,\ldots,\lambda_M)$. However, the formal description can be tedious. We provide the full information only when necessary.

3. Proposed Split-and-Conquer Approach

Suppose we have an IID random sample $\mathcal{X} = \{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_N\}$ from a parametric distribution $f(\boldsymbol{x};\theta)$. Suppose that it is partitioned into M subsets $\mathcal{X}_1,\ldots,\mathcal{X}_M$ completely at random and stored on M local machines. Let N_m denote the sample size on the mth local machine. Clearly, $\sum_{m=1}^M N_m = N$. Let $\widehat{\theta}_m$ be the local estimate of θ based on $\mathcal{X}_m$. For example, under finite GMM, the parameter θ becomes the mixing distribution G and the local estimate is the pMLE

$$\widehat{G}_m = \arg\max \left\{ \sum_{i \in \mathcal{X}_m} \log \phi(\boldsymbol{x}_i; G) - a_m \sum_{k=1}^K \left\{ \operatorname{tr}(S_m \boldsymbol{\Sigma}_k^{-1}) + \log\det(\boldsymbol{\Sigma}_k) \right\} \right\}$$

where S_m is the sample covariance matrix based on the observations in $\mathcal{X}_m$ and $a_m = N_m^{-1/2}$ as recommended. *How should we aggregate the local estimates, $\widehat{\theta}_m$ in general and $\widehat{G}_m$ in particular, in the split-and-conquer framework?*

One approach is to combine the local estimates by their linear average (Zhang et al., 2015; Chang et al., 2017). The aggregated estimator is then the weighted average $\overline{\theta} = \sum_{m=1}^M \lambda_m \widehat{\theta}_m$ with λ_m set to the sample proportion N_m/N. This simple approach is appropriate for parameters in a vector space, but it is nonsensical if the average of the parameters is not

well defined. Under finite mixtures, let $\widehat{G}_1, \ldots, \widehat{G}_M$ be the local estimators. Then the linear average is

$$\overline{G} = \sum_{m=1}^{M} \lambda_m \widehat{G}_m. \tag{5}$$

Its corresponding mixture has density function $f(x; \overline{G}) = \sum_{m=1}^{M} \lambda_m f(x; \widehat{G}_m)$. While $f(x; \overline{G})$ is a good estimate, it can be unsatisfactory in terms of revealing the latent structure of the mixture. For instance, this estimator gives a mixture with MK subpopulations rather than the assumed K, which is useless for clustering the dataset into K clusters.

Approach 1 One way to adapt the linear averaging is to define a sensible "average" in the space of the mixing distributions. Recall that the linear average of vectors in the Euclidean space is the centroid (barycenter) of these vectors from a geometric point of view. Similarly, the local estimators $\widehat{G}_1, \ldots, \widehat{G}_m$ may be "averaged" through their barycenter:

$$\overline{G}^C = \operatorname*{arg\,inf}_{G \in \mathbb{G}_K} \sum_{m=1}^{M} \lambda_m \rho(\widehat{G}_m, G) \tag{6}$$

for some choice of the divergence $\rho(\cdot, \cdot)$. The parameter space $\mathbb{G}_K$ is the space of mixing distributions with K support points and is defined in (1).

Approach 2 The mixing distribution $\overline{G}$ is likely close to the true mixing distribution G^*, except for the incorrect number of support points. This problem can be solved by approximating $\overline{G}$ by some $G \in \mathbb{G}_K$. This suggests another aggregation approach. Let $\rho(\cdot, \cdot)$ be a divergence in the space of mixing distributions. We can aggregate the local estimates via the reduction estimator, given by

$$\overline{G}^R = \operatorname*{arg\,inf}_{G \in \mathbb{G}_K} \rho(\overline{G}, G). \tag{7}$$

These two aggregation approaches, barycenter and reduction, are connected when specific divergences are used. For example, let the divergence $\rho(\cdot, \cdot)$ in the barycenter estimator (6) or in the reduction estimator (7) be the well-known KL-divergence:

$$\rho(G_1, G_2) = D_{\mathrm{KL}}(\Phi(\cdot; G_1) \| \Phi(\cdot; G_2)) = \int \phi(x; G_1) \log\{\phi(x; G_1)/\phi(x; G_2)\} dx.$$

In this case, we have

$$\begin{aligned} D_{\mathrm{KL}}(\Phi(\cdot; \overline{G}) \| \Phi(\cdot; G)) &= \int \phi(x; \overline{G}) \log\left\{\phi(x; \overline{G})/\phi(x; G)\right\} dx \\ &= C_1 - \int \phi(x; \overline{G}) \log \phi(x; G) dx \\ &= C_1 - \sum_{m=1}^{M} \lambda_m \int \phi(x; \widehat{G}_m) \log \phi(x; G) dx \\ &= C_2 + \sum_{m=1}^{M} \lambda_m D_{\mathrm{KL}}(\Phi(\cdot; \widehat{G}_m) \| \Phi(\cdot; G)) \end{aligned}$$

where C_1 and C_2 are constants not dependent on G. This relationship implies that

$$\overline{G}^R = \underset{G\in\mathbb{G}_K}{\arg\inf}\, \rho(\overline{G}, G) = \underset{G\in\mathbb{G}_K}{\arg\inf} \Big\{ \sum_{m=1}^{M} \lambda_m \rho(\widehat{G}_m, G) \Big\} = \overline{G}^C. \tag{8}$$

Thus, the two aggregation methods give identical aggregated estimators. The following example shows that the two methods do not always have the same outcome and the barycenter may not give ideal aggregation result.

Example 2 (Barycenter of Mixtures with Identical Subpopulations) *Suppose we wish to aggregate two local estimates given by*

$$\begin{aligned} \Phi(x; G_1) &= 0.4\Phi(x; -1, 1) + 0.6\Phi(x; 1, 1) := 0.4\Phi_{-1} + 0.6\Phi_1, \\ \Phi(x; G_2) &= 0.6\Phi(x; -1, 1) + 0.4\Phi(x; 1, 1) := 0.6\Phi_{-1} + 0.4\Phi_1 \end{aligned}$$

with $\lambda_1 = \lambda_2 = 0.5$. We anticipate that whatever distance or divergence we choose, the barycenter will be given by

$$\Phi(x; \overline{G}) = 0.5\Phi(x; -1, 1) + 0.5\Phi(x; 1, 1) = 0.5\Phi_{-1} + 0.5\Phi_1.$$

Consider the 2-Wasserstein distance with Euclidean ground distance between two univariate Gaussian distributions that is given by

$$D^2\big(\Phi(\cdot; \mu_1, \sigma_1^2), \Phi(\cdot; \mu_2, \sigma_2^2)\big) = (\mu_1 - \mu_2)^2 + (\sigma_1 - \sigma_2)^2.$$

Surprisingly, when the divergence is chosen to be $\rho = \mathbb{W}_D^2(\cdot, \cdot)$, the barycenter is given by

$$\Phi(x; \overline{G}^C) = 0.4\Phi(x; -1, 1) + 0.6\Phi(x; 2/3, 1) := 0.4\Phi_{-1} + 0.6\Phi_{2/3}.$$

We defer the technical details to Appendix 8.1.1. This example is extremely sharp and best illustrates this issue. The distortion of barycenter to the level in this example is unlikely in real-world applications. The issue here is rooted in the divergence ρ employed in defining the barycenter. Other choices of ρ may lead to a solution not far from our intuition. In comparison, for the reduction estimator, whatever divergence ρ is employed, we always have $\overline{G}^R = \overline{G}$. This example urges us to explore various choices of the divergence ρ in (6) and (7). We must take into account both statistical efficiency and computational complexity. For statistical efficiency, the properties of the divergence in (6) must ensure that the corresponding barycenter matches our intuition. Besides, the computation of (6) is usually more expensive than that of (7) given the same ρ. This paper hence focuses more on reduction estimator (7) for split-and-conquer learning of finite mixtures.

In the machine learning community, approximate one Gaussian mixture by another with a lower order is called Gaussian mixture reduction (GMR) (Crouse et al., 2011). These approaches are either ad hoc or computationally challenging. For example, Williams and Maybeck (2006) proposes to perform GMR by minimizing $\rho(\overline{G}, G) = L_2(\Phi(\cdot; \overline{G}), \Phi(\cdot; G))$. Although the L_2 distance between two Gaussian mixtures has an analytical form, the optimization can be expensive. Our key observation is that it is usually difficult to compute the divergence between two mixtures, but easy to compute that between two Gaussian distributions. This leads to the effective reduction algorithm that we present in the next section. For simplicity, we refer to the proposed estimator as the GMR estimator.

4. Numerical Algorithm for Proposed Reduction Approach

Let $\overline{G}$ be defined as in (5). Let the subpopulations in $\overline{G}$ be $\overline{\Phi}_i = \Phi(x; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ and the mixing weights be w_i for $i \in [MK] = \{1, 2, \ldots, MK\}$. Let G be any mixing distribution of order K with the K subpopulations $\Phi_\gamma = \Phi(x; \boldsymbol{\mu}_\gamma, \boldsymbol{\Sigma}_\gamma)$ and the mixing weights v_γ for $\gamma \in [K]$. In vector format, the weights are $\boldsymbol{w}$ and $\boldsymbol{v}$.

Let $c(\cdot, \cdot)$ be a cost function based on a divergence in the space of d-dimensional Gaussian distributions for which the computational cost is low. Then the transportation divergence (Nguyen, 2013) between mixing distributions $\overline{G}$ and $G \in \mathbb{G}_K$ with cost function c becomes

$$\mathcal{T}_c(\overline{G}, G) = \inf \Big\{ \sum_{i,\gamma} \pi_{i\gamma} c(\overline{\Phi}_i, \Phi_\gamma) : \boldsymbol{\pi} \in \Pi(\boldsymbol{w}, \boldsymbol{v}) \Big\}.$$

The corresponding GMR estimator is

$$\overline{G}^R = \arg\inf \big\{ \mathcal{T}_c(\overline{G}, G) : G \in \mathbb{G}_K \big\}. \tag{9}$$

For (9), it may appear that calculating our estimator involves two optimizations: computing $\mathcal{T}_c(\overline{G}, G)$ for each pair of $\overline{G}$ and G, and then searching for $\arg\inf_G \mathcal{T}_c(\overline{G}, G)$. We are able to design a more efficient optimization algorithm based on the following observation. The optimization problem in $\mathcal{T}_c(\overline{G}, G)$ involves searching for transportation plans $\boldsymbol{\pi}$ under two marginal constraints specified by $\boldsymbol{w}$ and $\boldsymbol{v}$. While constraint $\boldsymbol{w}$ is strict, $\boldsymbol{v}$ is a moving constraint. Instead of searching for $\boldsymbol{\pi}$ satisfying constraint $\boldsymbol{v}$, we move $\boldsymbol{v}$ to meet $\boldsymbol{\pi}$. This makes the marginal distribution constraint $\boldsymbol{v}$ on $\boldsymbol{\pi}$ redundant.

Let us define two functions of G, with $\overline{G}$ hidden in the background:

$$\mathcal{J}_c(G) = \inf \big\{ \sum_{i,\gamma} \pi_{i\gamma} c(\overline{\Phi}_i, \Phi_\gamma) : \boldsymbol{\pi} \in \Pi(\boldsymbol{w}, \cdot) \big\}, \tag{10}$$

$$\boldsymbol{\pi}(G) = \arg\inf \big\{ \sum_{i,\gamma} \pi_{i\gamma} c(\overline{\Phi}_i, \Phi_\gamma) : \boldsymbol{\pi} \in \Pi(\boldsymbol{w}, \cdot) \big\}. \tag{11}$$

Note that both functions depend on G through its subpopulations Φ_γ but are free of its mixing weights $\boldsymbol{v}$. The optimizations in (10) and (11) involve only the linear constraint in terms of $\boldsymbol{w}$. Hence, the optimal transportation plan $\boldsymbol{\pi}(G)$ for a given G has an analytical form:

$$\pi_{i\gamma}(G) = \begin{cases} w_i & \text{if } \gamma = \arg\min_{\gamma'} c(\overline{\Phi}_i, \Phi_{\gamma'}) \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

When $c(\Phi_i, \Phi)$ has multiple minima, we transport Φ_i evenly to every minimum Φ. For example, if $c(\Phi_1, \Phi)$ have two minima $\Phi_{\gamma'}$ and $\Phi_{\gamma''}$, we let $\pi_{1\gamma'}(G) = \pi_{1\gamma''}(G) = w_1/2$.

Theorem 5 *Let $\overline{G}$, $\mathcal{T}_c(\cdot)$, $\mathcal{J}_c(\cdot)$, $\boldsymbol{\pi}(\cdot)$, and other notations be the same as earlier. We have*

$$\inf\{\mathcal{T}_c(G) : G \in \mathbb{G}_K\} = \inf\{\mathcal{J}_c(G) : G \in \mathbb{G}_K\}. \tag{13}$$

The subpopulations of the GMR estimator are hence given by

$$\overline{G}^R = \arg\inf\{\mathcal{J}_c(G) : G \in \mathbb{G}_K\} \tag{14}$$

and the mixing weights are given by $\boldsymbol{v}$ *with*

$$v_\gamma = \sum_i \pi_{i\gamma}(\overline{G}^R). \tag{15}$$

The existence of a solution to (14) is guaranteed under a simple condition on cost function $c(\cdot,\cdot)$, see Theorem 7. The proof of Theorem 5 is in Appendix 8.1.2. Based on this theorem, the optimization reduces to searching for K subpopulations Φ_γ for $\gamma \in [K]$ to make up G. The mixing proportions are then determined by (15). An iterative algorithm quickly emerges following the well-known majorization-minimization (MM) idea (Hunter and Lange, 2004).

Algorithm 1 MM algorithm for GMR estimator with KL-divergence cost function.

Initialization: Φ_γ, $\gamma \in [K]$
repeat
  for $\gamma \in [K]$ **do**
    for $i \in [MK]$ **do**
      Let

$$\pi_{i\gamma} = \begin{cases} w_i & \text{if } \gamma = \arg\min_{\gamma'} D_{\mathrm{KL}}(\overline{\Phi}_i, \Phi_{\gamma'}) \\ 0 & \text{otherwise} \end{cases}$$

    end for
    Let

$$\pi_{\cdot\gamma} = \sum_{i=1}^{MK} \pi_{i\gamma}, \boldsymbol{\mu}_\gamma = \sum_{i=1}^{MK} \{\pi_{i\gamma}/\pi_{\cdot\gamma}\}\boldsymbol{\mu}_i$$

$$\boldsymbol{\Sigma}_\gamma = \sum_{i=1}^{MK} \{\pi_{i\gamma}/\pi_{\cdot\gamma}\}\{\boldsymbol{\Sigma}_i + (\boldsymbol{\mu}_i - \boldsymbol{\mu}_\gamma)(\boldsymbol{\mu}_i - \boldsymbol{\mu}_\gamma)^\top\}$$

  end for
until the change in the value of the objective function $\sum_{i,\gamma} \pi_{i\gamma} D_{\mathrm{KL}}(\overline{\Phi}_i, \Phi_\gamma)$ is below some threshold $\epsilon > 0$
Let $v_\gamma = \sum_i \pi_{i\gamma}$ for $\gamma \in [K]$
Output: $\{(v_\gamma, \boldsymbol{\mu}_\gamma, \boldsymbol{\Sigma}_\gamma) : \gamma \in [K]\}$

The algorithm starts with some $G^{(0)}$ with K subpopulations specified. Let $G^{(t)}$ be the mixing distribution after t MM iterations. Define a majorization function of $\mathcal{J}_c$ at $G^{(t)}$ to be

$$\mathcal{K}_c(G|G^{(t)}) = \sum_{i,\gamma} \pi_{i\gamma}(G^{(t)}) c(\overline{\Phi}_i, \Phi_\gamma) \tag{16}$$

where $\pi_{i\gamma}(G^{(t)})$ is computed according to (12). Once $\boldsymbol{\pi}(G^{(t)})$ has been obtained, we update the mixing proportion vector of $G^{(t)}$ easily via

$$v_\gamma^{(t+1)} = \sum_i \pi_{i\gamma}(G^{(t)}).$$

In fact, $\boldsymbol{v}^{(t)}$ is not needed until the algorithm converges.

The subpopulations Φ_γ are separated in the majorization function (16). This allows us to update the subpopulation parameters, one Φ_γ at a time and possibly in parallel, as the solutions to

$$\Phi_\gamma^{(t+1)} = \arg\inf_{\Phi} \sum_i \pi_{i\gamma}(G^{(t)}) c(\overline{\Phi}_i, \Phi). \tag{17}$$

The MM algorithm then iterates between the majorization step (16) and the minimization step (17) until some user-selected convergence criterion is met.

The most expensive step in the MM algorithm is the optimization in (17). If we choose the cost function $c(\cdot,\cdot) = \rho^r(\cdot,\cdot)$ with $\rho(\cdot,\cdot)$ being a divergence in the space of probability measures, the solution to (17) is a barycenter as given in Definition 4. The following lemma shows that the KL-based barycenter of Gaussian distributions has an analytical form and is therefore computationally simple.

Lemma 6 (KL-Barycenter of Gaussian Measures) *Let $\nu_m = \Phi(\cdot; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m), m \in [M]$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_M) \in \Delta_{M-1}$. Then $\sum_{m=1}^M \lambda_m D_{KL}(\nu_m \| \eta)$ is minimized uniquely in the space of Gaussian measures at $\eta = \Phi(\cdot; \overline{\mu}, \overline{\Sigma})$ with*

$$\overline{\mu} = \sum_{m=1}^M \lambda_m \boldsymbol{\mu}_m \quad \textit{and} \quad \overline{\boldsymbol{\Sigma}} = \sum_{m=1}^M \lambda_m \{\boldsymbol{\Sigma}_m + (\boldsymbol{\mu}_m - \overline{\mu})(\boldsymbol{\mu}_m - \overline{\mu})^\top\}.$$

Due to the ease of computing the barycenter as shown in this lemma, we recommend $c(\overline{\Phi}_i, \Phi_\gamma) = D_{\mathrm{KL}}(\overline{\Phi}_i \| \Phi_\gamma)$ in (9). Cost functions define the geometries on the Gaussian distribution space (Peyré and Cuturi, 2019), leading to slightly different outputs. We do not rule out the possibility of better choices. The pseudo-code for the MM algorithm with KL divergence as the cost function is given in Algorithm 1.

For notational convenience, in the following theorem, we use Φ for both the parameter $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and the distribution, and similarly for Φ^*.

Theorem 7 *Suppose the cost function $c(\cdot,\cdot)$ is continuous in both arguments. For some distance in the parameter space of Φ, assume that for any constant $\Delta > 0$ and Φ^* the following set is compact:*

$$\{\Phi : c(\Phi^*, \Phi) \le \Delta\}. \tag{18}$$

Let $\{G^{(t)}\}$ be the sequence generated by $G^{(t+1)} = \arg\min \mathcal{K}_c(G|G^{(t)})$ with some initial mixing distribution $G^{(0)}$. Then

(i) $\mathcal{J}_c(G^{(t+1)}) \le \mathcal{J}_c(G^{(t)})$ for any t;

(ii) if G^* is a limiting point of $G^{(t)}$, then $G^{(t)} = G^*$ implies $\mathcal{J}_c(G^{(t+1)}) = \mathcal{J}_c(G^*)$.

These two properties imply that $\mathcal{J}_c(G^{(t)})$ converges monotonically to some constant $\mathcal{J}^*$. All the limiting points $G^{(t)}$ are stationary points of $\mathcal{J}_c(\cdot)$: iterations from G^* do not further reduce the value of the objective function $\mathcal{J}_c(\cdot)$. We have practically cloned the global convergence theorem (Zangwill, 1969). We do not see a way to directly apply it and therefore provide a proof of the theorem in Appendix 8.1.3.

We have all the ingredients for the split-and-conquer learning of a finite GMM. We then consider the statistical properties of the GMR estimator and the experimental evidence for the efficiency of our method.

5. Statistical Properties of the Reduction Estimator

We show that the proposed GMR estimator $\overline{G}^R$ is consistent and retains the optimal rate of convergence in a statistical sense. We first state some conditions on the data and the estimation methods.

C1 The data $\mathcal{X}$ are IID observations from the finite Gaussian mixture $\Phi(x; G^*)$ with K distinct subpopulations, that is the order of G^* is known to be K. The subpopulations have distinct parameters and positive definite covariance matrices.

C2 The dataset $\mathcal{X}$ is partitioned into M subsets $\mathcal{X}_1, \ldots, \mathcal{X}_M$ of sizes $N_1, \ldots, N_M$, where each set contains IID observations from the same finite Gaussian mixture distribution. The number of local machines M does not increase with $N = \sum_m N_m$.

C3 The local machine sample ratios N_m/N have a nonzero limit as $N \to \infty$.

C4 The cost function $c(\Phi_k, \Phi_0) \to 0$ or $c(\Phi_0, \Phi_k) \to 0$ if and only if $\Phi_k \to \Phi_0$ in distribution, and $c(\Phi_1, \Phi_2)$ is continuous in both Φ_1 and Φ_2.

Condition C4 is necessary to ensure consistency. It further rules out the case that $\mathcal{T}_c(G, G^*) = \infty$ for any G with different mixing weights from that of G^*.

The consistency and asymptotic normality of the pMLE under the finite multivariate GMM are established in Chen and Tan (2009) under the standard IID setting. We do not repeat the details here but state the conclusion in a simplified fashion. We use Φ_k^* and $(w_k^*, \boldsymbol{\mu}_k^*, \boldsymbol{\Sigma}_k^*)$ to respectively denote true subpopulation, the mixing weight and the true subpopulation parameters.

Lemma 8 (Consistency of pMLE) *Given n IID observations from a finite multivariate Gaussian mixture with known order K, the pMLE $\widehat{G}$ as defined by (2) with $a_n = n^{-1/2}$ is asymptotically normal with rate $n^{-1/2}$.*

Specifically, it is possible to line up the subpopulation parameters of the true mixing distribution G^* and of the pMLE $\widehat{G}$ such that

$$(\widehat{w}_k, \widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k) = (w_k^*, \boldsymbol{\mu}_k^*, \boldsymbol{\Sigma}_k^*) + o(1)$$

and

$$(\widehat{w}_k, \widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k) = (w_k^*, \boldsymbol{\mu}_k^*, \boldsymbol{\Sigma}_k^*) + O_p(n^{-1/2})$$

as $n \to \infty$ in obvious notation.

Here $o(1)$ is a quantity that converges to 0 almost surely. A quantity is $O_p(n^{-1/2})$ if it is bounded by $Cn^{-1/2}$ for sufficiently large C with a probability arbitrarily close to 1. Chen and Tan (2009) proves the root-n-consistency for a non-random penalty term, the asymptotic normality remains valid when the sample covariance matrix is part of the penalty. This is because the sample covariance matrix converges to a positive definite matrix. The assumption of known K is crucial for the claimed rate of convergence. If K is unknown, then the convergence rate of $\widehat{G}$ is far below $n^{-1/2}$: see Chen (1995) and the recent developments in Rousseau and Mengersen (2011), Nguyen (2013), Heinrich and Kahn (2018), and Dwivedi et al. (2020). Our proposed reduction estimator is aggregated from the pMLEs learned

at the local machines. Under the condition that $\min N_m \to \infty$ stated above, all the local estimators are consistent by Lemma 8. Hence, the consistency of the linear average estimator $\overline{G}$ is taken as granted for a constant M.

Theorem 9 (Consistency of $\overline{G}^R$) *Let $\overline{G}$ be the linear average estimator defined by (5) and $\overline{G}^R$ be the aggregated estimator by reduction defined by (7) with $\rho = \mathcal{T}_c$. Assume conditions C1–C4 are satisfied. Then $\overline{G}^R$ is strongly consistent. Specifically, $\mathcal{T}_c(\overline{G}^R, G^*) \to 0$ almost surely as $N \to \infty$.*

The proof of the theorem is given in Appendix 8.1.4. The following theorem shows that under one additional mild condition on the cost function $c(\cdot, \cdot)$, the reduction estimator $\overline{G}^R$ has the standard $N^{-1/2}$ convergence rate. We denote by $\|\Phi_1 - \Phi_2\|$ the Euclidean norm in $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ in the sense of (20). We use $\overline{\Phi}_k^R$ for the kth subpopulation of $\overline{G}^R$ and $\overline{w}_k$ for its mixing weight for $k \in [K]$.

Theorem 10 (Convergence Rate of $\overline{G}^R$) *Let $\overline{G}$ be the aggregate estimator defined by (5) and $\overline{G}^R$ be the aggregate estimator by reduction defined by (7). Assume conditions C1–C4 are satisfied and further assume that*

C5 *For any Φ, there exists a small neighborhood Ω of Φ and a positive constant A, such that for any $\Phi_1, \Phi_2 \in \Omega$, we have*

$$A^{-1}\|\Phi_1 - \Phi_2\|^2 \le c(\Phi_1, \Phi_2) \le A\|\Phi_1 - \Phi_2\|^2.$$

Then with proper labelling of subpopulations, we have

$$\overline{\Phi}_k^R - \Phi_k^* = O_p(N^{-1/2}), \quad \overline{w}_k^R - \pi_k^* = O_p(N^{-1/2}).$$

Condition C5 requires the cost function $c(\cdot, \cdot)$ behaves locally as a quadratic loss function. This is a most natural property for a cost function. In Appendix 8.1.6, we show this condition holds for the KL divergence. The conclusion should hold with any other reasonable choices. The proof of the theorem is given in Appendix 8.1.5. Our proof remains valid, for instance, if we replace $\|\Phi_1 - \Phi_2\|^2$ by $\|\Phi_1 - \Phi_2\|^r$ for any $r > 0$ in C5.

6. Related Work

In this section, we describe several related approaches. We compare some of these approaches with our proposed approach in Section 7 in the experiments.

6.1 KL Averaging

Liu and Ihler (2014) considers the distributed learning of models from an exponential family by the split-and-conquer approach. The parameter θ is a real vector in this case. It proposes to perform local inference by finding the local MLEs and aggregate them by their KL barycenter. This estimator is referred to as the KL averaging (KLA). When the model belongs to the exponential family, the aggregated estimator is as efficient as the global MLE based on the full dataset. For models not in the exponential family, such as GMM, the KLA

estimator is less efficient. Moreover, the exact computation of the KL barycenter of local estimators under GMM is difficult. Liu and Ihler (2014) suggests to find an approximate solution instead. It first generates random samples $\widehat{\mathcal{X}}_m$ of size 1000 from the local estimates $\widehat{G}_m$ at the central machine. Then a GMM of order K is fitted on the pooled sample $\cup_m \widehat{\mathcal{X}}_m$ which has a moderate size of $1000M$. This approach does not need to transmit the raw data but requires refitting of the mixture on the central machine.

6.2 Distributed EM Algorithm

The distributed learning of GMMs can also be tackled by developing a distributed version of the EM algorithm (DEM) (Nowak, 2003). We briefly describe DEM and provide a conceptual comparison to our approach in this section.

Under distributed learning setting, let $\mathcal{X}_m$ be the dataset stored at the mth local machine $m \in [M]$ and N be the total sample size. Note that the quantities required in defining $Q(G; G^{(t)})$ based on full dataset $\mathcal{X}$ have the following decomposition: for $k \in [K]$,

$$\begin{aligned}
\Omega_k^{(t)} &:= \sum_{i=1}^{N} w_{ik}^{(t)} = \sum_{m=1}^{M} \Big\{ \sum_{i \in \mathcal{X}_m} w_{ik}^{(t)} \Big\} := \sum_{m=1}^{M} \Omega_{m,k}^{(t)}, \\
\mathbf{A}_k^{(t)} &:= \sum_{i=1}^{N} w_{ik}^{(t)} \boldsymbol{x}_i = \sum_{m=1}^{M} \Big\{ \sum_{i \in \mathcal{X}_m} w_{ik}^{(t)} \boldsymbol{x}_i \Big\} := \sum_{m=1}^{M} \mathbf{A}_{m,k}^{(t)}, \\
\mathbf{B}_k^{(t)} &:= \sum_{i=1}^{N} w_{ik}^{(t)} \boldsymbol{x}_i \boldsymbol{x}_i^\top = \sum_{m=1}^{M} \Big\{ \sum_{i \in \mathcal{X}_m} w_{ik}^{(t)} \boldsymbol{x}_i \boldsymbol{x}_i^\top \Big\} := \sum_{m=1}^{M} \mathbf{B}_{m,k}^{(t)}.
\end{aligned}$$

Given $G^{(t)}$, one can compute $w_{ik}^{(t)}$ defined by (3) for ith observation on local machine m for all $m \in [M]$ and $k \in [K]$. Hence, one can obtain $\cup_{k=1}^{K} \{\Omega_{m,k}^{(t)}, \mathbf{A}_{m,k}^{(t)}, \mathbf{B}_{m,k}^{(t)}\}$ at the mth local machine and have them transmitted to a central machine. One can then construct $Q(G; G^{(t)})$ on the central machine and carry out the M-step to get $G^{(t+1)}$, which reproduces the EM-iteration based on the full dataset.

Nowak (2003) considers the situation where the local machines form a sensor network and the transmission cost cannot be ignored. The paper suggests the mth machine transmits $\cup_{j=1}^{m} \{\Omega_{j,k}^{(t)}, \mathbf{A}_{j,k}^{(t)}, \mathbf{B}_{j,k}^{(t)}\}$ to the next machine in the queue. Furthermore, it adopts the incremental E and M steps of Neal and Hinton (1998) to speed up the convergence of the algorithm. Nowak (2003) further shows that the DEM has a local linear convergence rate.

The DEM and proposed GMR approaches are designed for distributed learning with different communication schemes. DEM requires a high level of coordination between local machines and repeated access of the local data. Our proposed method allows local machines to complete the learning on its own. Moreover, our method only requires one round of communication across all machines and is communication efficient. If successful, DEM leads to a solution to the original learning problem retaining full statistical efficiency. We do not include DEM in the experiment because the conclusions are already known. Our proposed method is superior in terms of communication cost. The statistical efficienty of DEM is same as the global estimator which is compared with our proposed method in the experiment.

6.3 Learning at Scale via Coresets

Most machine learning problems can be formulated as an optimization problem that minimizes a cost function $\text{cost}(\mathcal{X}, \theta)$ over the parameter space. For our problem, the cost function could be the negative log likelihood, $\mathcal{X}$ is the full dataset, and the parameter space $\mathbb{G}_K$ is given by (1).

When $\mathcal{X}$ is very large, the computational burden can be extremely heavy. Feldman et al. (2011) suggests to replace $\mathcal{X}$ by a much smaller weighted subset $\mathcal{C}$, called coreset hereafter, such that

$$\frac{|\text{cost}(\mathcal{X}, G) - \text{cost}(\mathcal{C}, G)|}{\text{cost}(\mathcal{X}, G)} \leq \epsilon \tag{19}$$

for some given small $\epsilon > 0$. Minimize the $\text{cost}(\mathcal{C}, G)$ based on the coreset can be much faster than the original cost based on the full dataset. The construction of coreset is to assure the minimizer of $\text{cost}(\mathcal{C}, G)$ approximates that of $\text{cost}(\mathcal{X}, G)$. Lucic et al. (2017) provides theoretical analysis and techniques for constructing coresets under GMM. For GMM of dimension d, order K, it gives a scheme to obtain coresets of size $|\mathcal{C}| = O(d^4 K^6 \epsilon^{-2})$ satisfying (19) uniformly over some compact subset of $\mathbb{G}_K$. This is a surprising result as the size of $\mathcal{C}$ does not depend on the size of $\mathcal{X}$.

When the datasets are stored in distributed fashion, one may first reduce the dataset in each machine into the first generation coresets. Then these coresets are paired up to create a second generation coreset from each pair. This procedure is repeated if needed until we get a final coreset. Due to the composition properties of coreset (Lucic et al., 2017, Section 5), the quality of the final coreset can be maintained. We refer to this approach as the Coreset approach hereafter.

The Coreset approach is computationally very efficient. Unlike DEM, it looks for approximate solutions leading to inevitable loss in statistical efficiency. Moreover, the Coreset approach requires the transmission of the raw data unlike other approaches for distributed learning that only requires the communication of summary statistics. We include the Coreset approach in our experiment for efficiency comparison.

We wish to remark that the log-likelihood function in statistics is defined up to an additive constant. The precision specification (19) can be affected by how it is normalized. We adopt the normalization convention of (Lucic et al., 2017).

6.4 Bayesian Moment Matching (BMM)

Direct Bayesian inference under GMM is challening due to the well-known label switching problem. See Murphy (2012, Chapter 11) for explanation and possible solutions. Jaini and Poupart (2016) proposes an approximate inference procedure that does not suffer from the issue of label switching. Under GMM with known order K, given a prior $\boldsymbol{\pi}_0$ with Dirichlet for weights and Gaussian-Gamma for subpopulation parameters, the posterior distribution $\tilde{\boldsymbol{\pi}}_1$ with **a single observation** x_1 is a complex mixture of Dirichlet and Gaussian-Gamma combination. Repeating this operation given another observation would lead to an even more complex posterior. Instead, Jaini and Poupart (2016) suggests to approximate $\tilde{\boldsymbol{\pi}}_1$ posterior with a simple Dirichlet and Gaussian-Gamma combination $\boldsymbol{\pi}_1$ so that $\tilde{\boldsymbol{\pi}}_1$ and $\boldsymbol{\pi}_1$ have the same lower order moments. Let $\boldsymbol{\pi}_1$ be the prior distribution, with the next single observation x_2, we obtain $\boldsymbol{\pi}_2$ in the same way. Repeat sequentially until we have exhausted

all data to get $\boldsymbol{\pi}_N$. Under multivariate GMMs, one replaces Gaussian-Gamma prior by Gaussian-Wishart. The end product $\boldsymbol{\pi}_N$ is regarded as an approximate posterior and it serves well for Bayes inferences.

Being sequential in nature, BMM is computationally efficient but apparently loses statistical efficiency due to approximation. Based on Table 2 in Jaini and Poupart (2016) and our Table 1, the per observation log-likelihood value of BMM when applied to Magic04 is -32.1, which is much lower than that of our proposed GMR -26.6. Jaini and Poupart (2016) shows that BMM has higher statistical efficiency than the online EM approach of Cappé and Moulines (2009, Theorem 2), whose convergence rate is lower than $N^{-1/2}$.

We do not include BMM in the experiment not only because of the comparison above but also because we do not know how they handle the redundant moment equations. Under the multivariate GMM, there are $K+dK+d(d+1)K/2$ parameters to be estimated at each step but there are $dK+2K-1+3d(d+1)K/2$ moment equations. The redudant equations make it difficult for us to replicate their approach.

6.5 ADMM for Distributed Optimization

The distributed learning of GMM is essentially a distributed optimization problem. We may therefore directly use the Alternating Direction Method of Multipliers (ADMM) of Boyd et al. (2011). Consider the optimization problem defined as

$$\min\Big\{\sum_{m=1}^{M} f(\theta_m|\mathcal{X}_m) : \theta_m-\theta=0,\ m\in[M],\ \theta\in\Theta\Big\}$$

for some function $f(\cdot|\mathcal{X}_m)$ and a Euclidean parameter space Θ. The parameters θ_m are called local variables and θ is called a global variable. The ADMM for this optimization problem is based on the augmented Lagrangian

$$L_\rho(\theta_m,\eta_m,\theta)=\sum_{m=1}^{M}\{f(\theta_m|\mathcal{X}_m)+\eta_m^\top(\theta_m-\theta)+(\rho/2)\|\theta_m-\theta\|_2^2\}$$

for some regularization parameter $\rho>0$. The ADMM then iterates according to

$$\begin{aligned}
\theta_m^{(t+1)} &= \arg\min_{\theta_m}\big\{f(\theta_m|\mathcal{X}_m)+(\eta_m^{(t)})^\top(\theta_m-\bar\theta^{(t)})+(\rho/2)\|\theta_m-\bar\theta^{(t)}\|_2^2\big\},\\
\bar\theta^{(t+1)} &= M^{-1}\sum_{m=1}^{M}\theta_m^{(t+1)},\\
\eta_m^{(t+1)} &= \eta_m^{(t)}+\rho(\theta_m^{(t+1)}-\bar\theta^{(t+1)}).
\end{aligned}$$

Similar to DEM, the ADMM requires a high level of coordination between local machines at each iteration. If successful, it gives the solution to the original optimization problem, void the statistical efficiency comparison. In the context of GMM, one must look for suitable substitutes for the term $(\eta_m^{(t)})^\top(\theta_m-\bar\theta^{(t)})$ in the augmented Lagrangian since θ_m is a discrete distribution. We therefore do not include ADMM in the experiment.

7. Experiments

We conduct experiments on both simulated and real data to illustrate the effectiveness of the proposed GMR estimator in (9) with $c(\overline{\Phi}_i, \Phi_\gamma) = D_{\mathrm{KL}}(\overline{\Phi}_i, \Phi_\gamma)$. We compare its performance with some existing approaches in terms of their statistical efficiency and computational costs. Our experiments include the following estimators:

1. *Global*. The pMLE based on the full dataset. The Global estimator is statistically most efficient and therefore used as the baseline for comparison.

2. *Median*. An off-the-shelf aggregation approach is the median

$$\overline{G}^{\mathrm{M}} = \underset{G \in \{\widehat{G}_1, \widehat{G}_2, \ldots, \widehat{G}_M\}}{\arg\min} \sum_{m=1}^{M} \lambda_m \mathcal{T}_{D_{\mathrm{KL}}}(\widehat{G}_m, G).$$

   The sample median is intuitively a robust alternative with minor efficiency loss.

3. *KLA*. The KL-Averaging in Liu and Ihler (2014) with $n_m = 1000$ observations generated from local estimate $\widehat{G}_m$. The real datasets have different dimensions and sample sizes, the size n_m for real data experiments is specified if different.

4. *Coreset*. The Coreset approach with $|\mathcal{C}_m| = 1000$ on each local machine. They are repeatedly merged as in Lucic et al. (2017) to arrive at the final coreset $\mathcal{C}$ of size 1000, the coreset sizes for real data experiment is specified if different.

For the ease of comparison, we use the pMLE defined in (2) with penalty size $N_m^{-1/2}$ as local estimates when applicable. The pMLE is also used in the KLA estimator of Liu and Ihler (2014) on the central machine with penalty size $(1000M)^{-1/2}$, and for the Coreset method with penalty size $|\mathcal{C}|^{-1/2}$. We use the EM algorithm to compute pMLE and declare convergence when the per observation penalized log-likelihood function is less than 10^{-6}. With very large sample sizes of the simulated data, the maxima of the penalized likelihood should be attained at a mixing distribution close to the true mixing distribution. We therefore use the true mixing distribution as the initial value and regard the output of the EM algorithm as the global maximum of the penalized likelihood. This strategy does not work for the real-world data in the absence of a true mixing distribution. For real-world data, we use *kmeans++* with default arguments in *scikit-learn* package (Pedregosa et al., 2011) to generate 10 initial values for the EM algorithm. Ideally, we run the EM algorithm with these initial values until convergence and regard the output of the EM algorithm with the highest penalized log-likelihood function as the pMLE. To save time on the real dataset, we use a warmup strategy. We run the EM algorithm with these 10 initial values for 20 iterations and pick the one with the highest penalized log-likelihood value. We use the output of this one as the initial value to run the EM algorithm further until convergence and this output is treated as the pMLE.

The choice of the initial value in the aggregation step is also important. When the sample size is large, we have good reason to believe that the optimal solution is close to the true value. Also, by the principle of majority rules, the median of the local estimates is likely the closest to the optimal solution. Thus, in simulation studies, we initialize the algorithm

with the true mixing distribution and the median estimate. For real-world data, we use the local estimators as multiple initial values and output of the MM algorithm with the lowest objective function value is regarded as the GMR estimator. We declare the convergence of the MM algorithm for the GMR estimator when the change in the objective function is less than 10^{-6}.

All experiments are conducted on the Compute Canada (Baldwin, 2012) Cedar cluster with Intel E5 Broadwell CPUs with 64G memory. The codes are written in Python and are publicly available at `https://github.com/SarahQiong/SCGMM`. The code for Coreset method is provided by the author of Lucic et al. (2017).

7.1 Performance Measure

In each simulation experiment, we generate $R = 100$ random samples $\mathcal{X}^{(r)}$ from a finite GMM with mixing distribution $G^{(r)}$ and obtain its estimate $\widehat{G}^{(r)}$, for $r \in [R]$. We use the following performance metrics.

1. **Transportation divergence** (W_1). We define the ground distance between Φ_i and Φ_γ with mean vectors $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_\gamma$ and covariance matrices $\boldsymbol{\Sigma}_i$ and $\boldsymbol{\Sigma}_\gamma$ to be

$$D(\Phi_i, \Phi_\gamma) = \|\boldsymbol{\mu}_i - \boldsymbol{\mu}_\gamma\|_2 + \|\boldsymbol{\Sigma}_i^{1/2} - \boldsymbol{\Sigma}_\gamma^{1/2}\|_F \tag{20}$$

   where $\|A\|_F = \sqrt{\mathrm{tr}(A^\top A)}$ is the Frobenius norm of a matrix. The corresponding transportation distance is $W_1(\widehat{G}^{(r)}, G^{(r)}) = \mathcal{T}_D(\widehat{G}^{(r)}, G^{(r)})$ given in Definition 3.

2. **Adjusted Rand Index** (ARI). The well known adjusted Rand index (ARI) measures the degree of similarity between two clustering outcomes. Suppose the observations in a dataset are divided into K clusters $A_1, A_2, \ldots, A_K$ by one method, and K' clusters $B_1, B_2, \ldots, B_{K'}$ by another method. Let $N_i = \#(A_i)$, $M_j = \#(B_j)$, $N_{ij} = \#(A_i B_j)$ for $i \in [K]$ and $j \in [K']$, where $\#(A)$ is the number of units in set A. The ARI of these two clustering outcomes is defined to be

$$\mathrm{ARI} = \frac{\sum_{i,j}\binom{N_{ij}}{2} - \binom{N}{2}^{-1}\sum_{i,j}\binom{N_i}{2}\binom{M_j}{2}}{\frac{1}{2}\sum_i\binom{N_i}{2} + \frac{1}{2}\sum_j\binom{M_j}{2} - \binom{N}{2}^{-1}\sum_{i,j}\binom{N_i}{2}\binom{M_j}{2}}$$

   where $\binom{n}{k}$ is the number of k combinations from a given set of n elements. An ARI value close to 1 indicates a high degree of agreement. The ARI is 1 when the two clustering methods completely agree.

   The mixture models are commonly used for model-based clustering. Based on a given G, we classify a unit with observed value $\boldsymbol{x}$ into cluster

$$k(\boldsymbol{x}) = \arg\max_k\{w_k\phi(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\} \tag{21}$$

   in obvious notation. We obtain the ARI between the clustering outcomes based on $\widehat{G}^{(r)}$ and $G^{(r)}$ and use it as a performance metric.

3. **Log-likelihood** (LL). The per observation log-likelihood value at the estimated mixing distribution $\widehat{G}^{(r)}$ based on the full dataset is often regarded as a goodness metric. See Liu and Ihler (2014), Jaini and Poupart (2016), and Lucic et al. (2017).

The split-and-conquer approaches have some advantages in computation time by performing local inference on multiple machines. Since we can compute the local estimates in parallel, the time it takes to obtain the final estimate is the maximum local time plus the aggregation time. We keep record of this time for each experiment. Similarly, we record the total time for constructing the coreset and fitting the mixture on the central machine. We do not keep track of the transmitting time.

7.2 Experiments Based on Simulated Data

Distributed learning methods are designed for learning at scale where the observations have high dimensions and large sample size. To reduce the potential influence of human bias, we simulate data from finite Gaussian mixtures with randomly generated parameter values. We use the `R` package `MixSim` (Melnykov et al., 2012; Maitra and Melnykov, 2010). An important quantity of a finite mixture is pairwise overlap

$$o_{j|i} = \mathbb{P}(w_i f(X; \boldsymbol{\theta}_i) < w_j f(X; \boldsymbol{\theta}_j) | X \sim f(\cdot; \boldsymbol{\theta}_i)).$$

The maximum overlap of a finite mixture is defined to be

$$\texttt{MaxOmega} = \max_{i,j \in [K]} \{o_{j|i} + o_{i|j}\}.$$

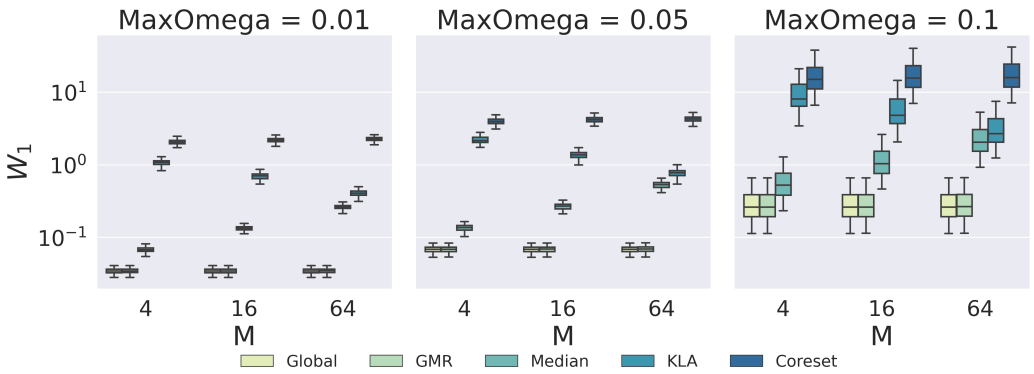
We use `MixSim` to generate 100 finite Gaussian mixtures with $d = 50$ and $K = 5$. We let `MaxOmega` be 1%, 5%, and 10%, $N = 2^l$ for $l = 17, 19, 21$ and $M = 2^l$ for $l = 2, 4, 6$. The simulated data are divided evenly over the local machines. We combine 100 outcomes from each combination of dimension, order, `MaxOmega`, sample size, and number of local machines to form boxplots for each estimation method. Figure 1 and Figure 2 show the results.

Figure 1 reports the result when the total sample size is $N = 2^{21}$. Within each subfigure, the `MaxOmega` increases from the left panel to the right panel. Within each panel, the x-axis gives the number of local machines: 4, 16, or 64. The plots in Figure 1 contain boxplots of W_1, ARI, LL, and the computation time.

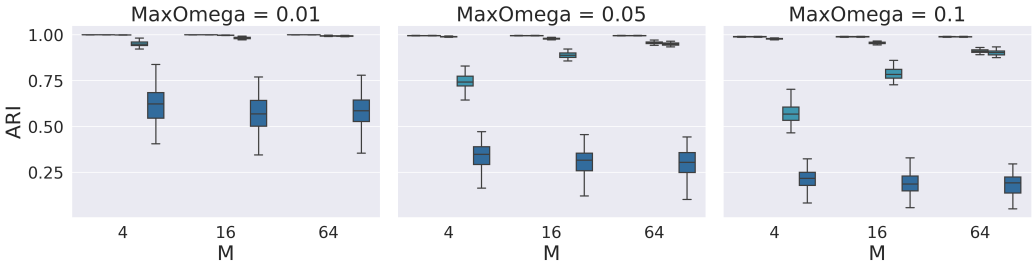
Based on Figure 1, all methods have better performance in terms of W_1, ARI, and LL when `MaxOmega` is lower. This is consistent with our intuition and the experiment survives the sanity check.

The proposed GMR has comparable performance to the gold-standard global estimator in all three metrics. It is arguably the best aggregation approach. The number of local machines has little influence on its performance. KLA has relatively poor performance though it improves moderately when the number of local machines increases. Recall that KLA generates a fixed number of observations from each local estimator. More local machines lead to a larger total sample size in its aggregation step. This helps its statistical performance but not computational efficiency. The median estimator does not perform with a large number of local machines. The Coreset estimator does not perform in all cases.

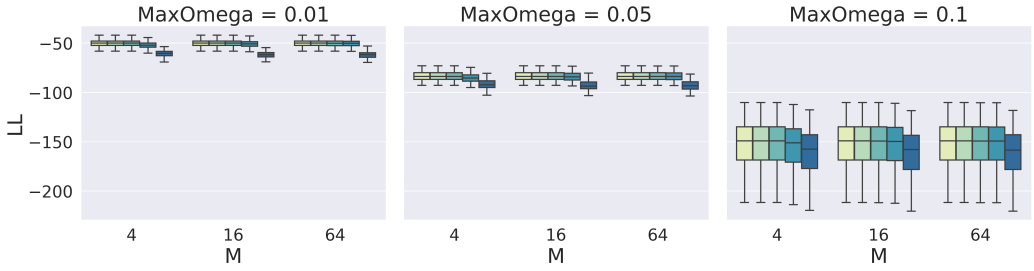
All aggregation approaches take less computation time than the global estimator. The Coreset estimator takes the least amount of computation time. However, the computation time does not taken the transmission cost into account. Moreover, the computation time does not make up the poor statistical performance. By far a large chunk of computation time of the split-and-conquer approaches is spent on learning the local estimators. Because



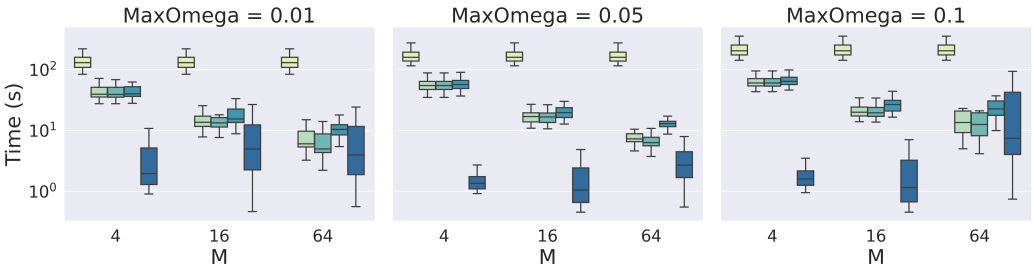
(a) Transportation distance W_1



(b) ARI



(c) Log-likelihood

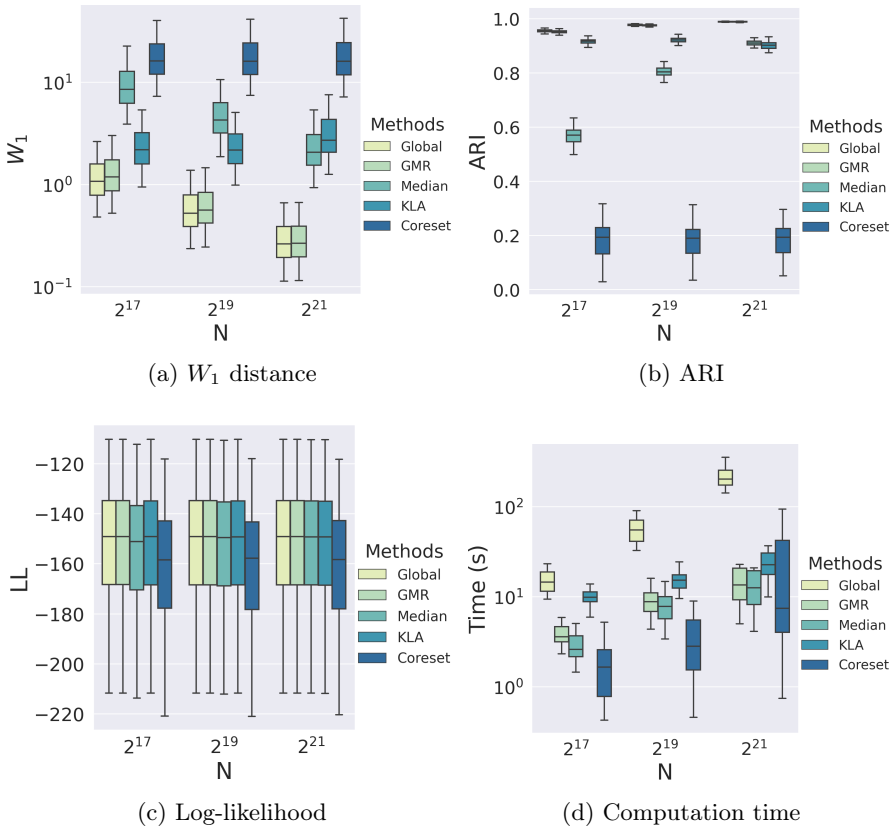


(d) Computation time

Figure 1: Performance of five estimators for learning 50-dimensional order 5 Gaussian mixtures with sample size $N = 2^{21}$. All subplots share the same color coding.

GMR and the median estimators spend negligible time on aggregation, they use about $1/M$ of the global estimator computational time. When the data generating mixture has a high degree of overlapping, the EM algorithm needs more iterations to converge leading to higher computation time. KLA must re-learn the GMM based on the pooled data generated from the local estimates, therefore generally takes longer time than the GMR approach.

Figure 2 presents results when `MaxOmega` is 10% and $M = 64$. It is seen the proposed



(a) W_1 distance

(b) ARI

(c) Log-likelihood

(d) Computation time

Figure 2: Performance of five estimators when `MaxOmega` $= 0.1$ and $M = 64$ for learning 50-dimensional order 5 Gaussian mixtures.

GMR has very good performance, comparable to the global estimator. The relative performance of the median estimator improves when the sample size increases, but still not good enough to be recommended. The performance of KLA and Coreset approaches do not improve with increased sample size. They are far from competitive against the proposed GMR approach.

Coreset approach does not benefit from larger sample size likely because the coreset size is fixed at 1000. KLA approach does not because the larger sample size improves only the precision of the local estimates. Its aggregation step is heavily influenced by the built-in randomness when we generate samples from the local estimates. The other aspects of the simulation results are as expected.

We have more simulation results when the dimensions $d = 10, 50$ combined with orders $K = 5$, 10, and 50. They are presented in Appendix 8.2.1. These results are consistent with what we find so far in terms of the statistical efficiency. Since the Coreset estimator is found not competitive, it is not included in these experiments.

7.3 Public and Real-World Datasets

We now examine the performance of the proposed approach on large scale public datasets in Section 7.3.1, for clustering the handwritten digits in Section 7.3.2, and for clustering a large scale spatio-temporal data in Section 7.3.3.

7.3.1 Public Datasets

We experiment on the public datasets that are widely used for learning Gaussian mixtures at scale in this section. The following datasets are used in Lucic et al. (2017) and Jaini and Poupart (2016).

1. MAGIC04. This is a simulated dataset for classifying gamma particles in the upper atmosphere. It contains $19,020$ observations with 10 real valued features and is publicly available at UCI machine learning repository.

2. MiniBooNE. The dataset is taken from the MiniBooNE experiment that is used to distinguish electron neutrinos from muon neutrinos. It contains $130,065$ observations with 50 real valued features and is publicly available at UCI machine learning repository.

3. KDD. This dataset is used in Lucic et al. (2017). It contains $145,751$ observations with 74 real valued features for predicting the protein types. It is available at `https://kdd.org/kdd-cup/view/kdd-cup-2004/Data`.

4. MSYP. The dataset is used to predict the release year of a song from audio features. It contains $515,345$ observations with 90 real valued features. The dataset is publicly available at UCI machine learning repository. Following Lucic et al. (2017), we reduce the dataset to its top 25 principal components and fit the mixture model with these 25 features.

For the first three datasets, we divide the dataset onto $M = 4$ local machines completely at random. Since MSYP is very big and the order of the mixture to be fitted is high, we divide the dataset onto $M = 16$ local machines. The random partition of the dataset is repeated $R = 100$ times. The size of the generated sample and coreset size are set to be 1000 for MAGIC04, and $10,000$ for other datasets. The order of the fitted mixture on each dataset follows the setting in Lucic et al. (2017) and is specified in Table 1.

For each method, due to repetition, we have 100 LL values based on the full dataset at the learned mixing distributions. We summarize these values by its median and inter quartile range (IQR). We also obtain the total computation time for each method. These results are given in Table 1.

Based on Table 1, it is clear that the proposed GMR has the best performance among all split-and-conquer based approaches in terms of the LL value. For the MiniBooNE dataset,

Table 1: Performance of five learning approaches on public datasets.

Dataset	N	d	K	M	Global	GMR	Median	KLA	Coreset
Median (IQR) LL values (the larger the better)									
MIGIC04	19020	10	10	4	-24.15	-24.30(0.07)	-26.60(0.05)	-26.73(0.07)	-27.16(0.55)
MiniBooNE	130065	50	10	4	-19.46	-22.00(0.53)	-24.60(0.32)	$-6.41(1.95) \times 10^3$	$-8.6(2.56) \times 10^9$
KDD	145751	74	10	4	-221.80	-223.25(0.42)	-232.93(8.02)	-235.00(8.96)	-374.43(193.58)
MSYP	515345	25	50	16	-166.56	-167.05(0.04)	-171.10(0.04)	-170.72(0.01)	-181.64(1.78)
Median (IQR) computation times in seconds									
MIGIC04	19020	10	10	4	19.3	7.0(3.2)	6.7(3.2)	10.2(3.1)	2.2(0.6)
MiniBooNE	130065	50	10	4	346.9	313.1(162.6)	313.2(162.6)	511.3(213.2)	26.6(64.3)
KDD	145751	74	10	4	1033.9	544.4(309.5)	543.0(310.0)	706.0(290.3)	4.3(64.0)
MSYP	515345	25	50	16	67048.8	2611.6(474.0)	1777.5(511.2)	5515.9(1629.7)	67.4(12.6)

the LL values of KLA and Coreset approaches are very small. This is likely because the total sample size to refit the GMM on the central machine is relatively small in 50-dimensional space. The fitted order 10 mixture may not be able to cover the entire space properly and the log-likelihood contribution of some observations is practically negative infinity. A single near zero likelihood value could lead to a very small LL value. To evaluate the improvement of the GMR approach over other approaches, we consider how many extra subpopulations are needed to achieve the same gain in the LL value. Note that the famous BIC (Schwarz, 1978) would favor a model with an extra parameter if the gain in LL is more than $\log(N)/(2N)$. The $\log(N)/(2N)$ values for these datasets are $(26, 4.5, 4.1, 1.3) \times 10^{-5}$. A subpopulation in GMM with $d = 74$ needs $1 + 74 + 74 * 75/2 = 2854$ parameters. This translates into a difference of 0.116 in LL. For the KDD data, the gain in GMR compared to KLA would allow another 17 subpopulations.

All the split-and-conquer learning methods are much faster than the global method. For the MSYP dataset, the split-and-conquer methods can be 10 times as fast compared to the global estimator. The Coreset method takes the shortest time. The proposed GMR approach takes comparable computation time with the KLA approach.

7.3.2 NIST Handwritten-Digit Dataset

The finite GMM is often used for model-based clustering (Friedman et al., 2001; Fraley and Raftery, 2002, Chapter 14.3). When the dataset is large and/or distributed over many local machines, split-and-conquer approaches such as the proposed GMR become useful. In this section, we demonstrate the use of the GMR method on the famous NIST dataset for character recognition (Grother and Hanaoka, 2016). We use the second edition of the dataset, named *by_ class.zip* [1]. It consists of approximately 4M images of handwritten digits and characters (0–9, A–Z, and a–z) by different writers. Our experiment focuses on the digits and we still refer to it as the NIST dataset. The images of the digits are in directories 30–39. According to the user guide [2], the images in the *train_ 30* to *train_ 39* and *hsf_ 4* folders are used as the training and test sets respectively. The numbers of training images for each digit are listed in the following table:

1. Available at `https://www.nist.gov/srd/nist-special-database-19`.
2. Available at `https://s3.amazonaws.com/nist-srd/SD19/sd19_users_guide_edition_2.pdf`.

Table 2: The numbers of training images for each digit in NIST dataset.

Digits	0	1	2	3	4	5	6	7	8	9
Training	34803	38049	34184	35293	33432	31067	34079	35796	33884	33720
Test	5560	6655	5888	5819	5722	5539	5858	6097	5695	5813

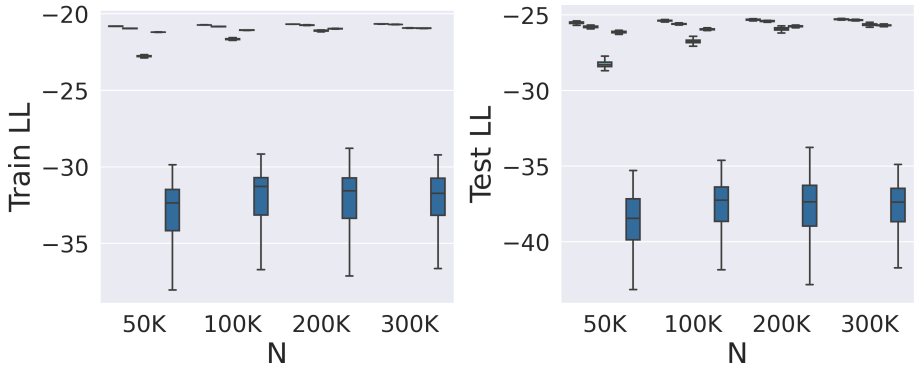
Each image is a 128×128 pixel grayscale matrix whose entries are real values between 0 and 1 that record the darkness of the corresponding pixels. A darker pixel has a value closer to 1. Following the common practice, we first train a 5-layer convolutional neural network and reduce each image to a $d = 50$ feature vector of real values. The details of the neural network for the dimension reduction are given in Appendix 8.2.2. A naïve approach to build a classifier is to regard the features of each digit as a random sample from a distinct Gaussian distribution. The pooled data is therefore a sample from a finite Gaussian mixture of order $K = 10$. We may learn this model based on the whole dataset or through split-and-conquer approaches.

We randomly select $R = 100$ datasets of size $N = 50K$ from the training set. Each dataset is then randomly partitioned into $M = 10$ subsets. We obtain global, GMR, Median, KLA, and Coreset estimates for a Gaussian mixture of order 10 on each dataset. The size of the generated sample for the KLA method and the coreset size in the Coreset method are both set to be 3000. This experiment is also carried out with the sample sizes $N = 100K$, $200K$, and $300K$. These mixture estimates are then used to cluster images of handwritten digits in the training and test sets. We show the boxplots of the LL values based on the training dataset and the test dataset in Figure 3(a) and Figure 3(b) respectively. The ARI between the true label of the image and the predicted label based on (21)is respectively given in Figure 3(c) and Figure 3(d).

In terms of the LL value, the proposed GMR approach attains the highest log-likelihood among all split-and-conquer approaches. The performance of Coreset estimator is far behind. In this experiment, the number of local machine is fixed at $M = 10$ and the sample sizes are from $50K$ to $300K$. Increasing the sample size benefits median estimator most notably. This is because the local sample size increases as the total sample size increases. When the total sample size is $300K$, the number of samples used to fit the local models and the samples generated to fit the aggregated model in the KLA approach are the same. The LL value of median estimator and the KLA estimator are about the same.

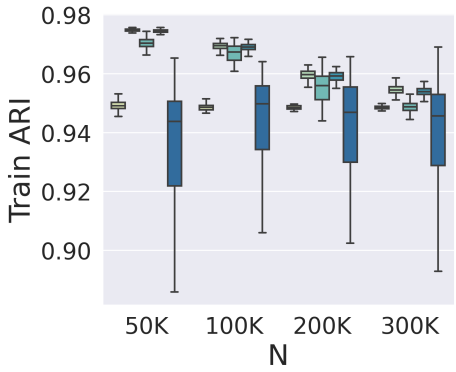
In terms of clustering performance, the global estimator surprisingly performs noticeably worse than the split-and-conquer approaches. The high LL value of the global estimator does not help. A likely explanation is that a GMM of order 10 is merely a **working** model rather than the **true** model, whereas true models are used in simulated data. This eliminates the advantage of the global estimator. This can also be seen that an increased total sample size N does not lead to an improved fit in general. The ARIs of all approaches get worse when the sample size increases. We think that the damage of the model-misspecification is more severe when the sample size is large. Nevertheless, the proposed GMR method has the best performance in all cases. It has the highest average ARI values and smaller variations.

Figure 3(e) gives the computation time. All split-and-conquer approaches save computation time. The Coreset estimator is most time efficient, the GMR and median estimators takes slightly longer and the KLA takes the longest.

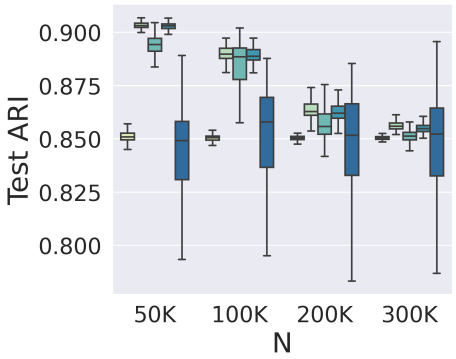


(a) Training LL

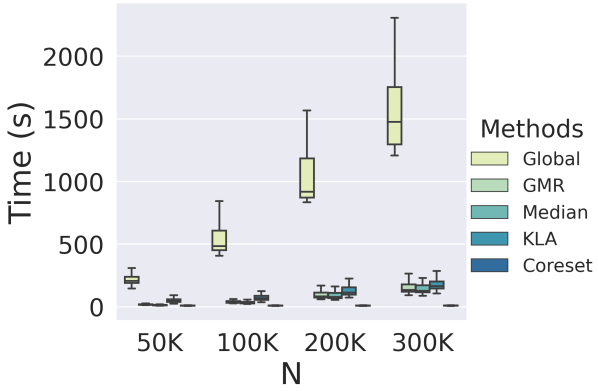
(b) Test LL



(c) Training ARI



(d) Test ARI



(e) Computation time

Figure 3: Performance of five approaches for learning of 50-dimension order 10 Gaussian mixture for NIST digit classification. All subplots share the same color coding.

7.3.3 Atmospheric Data Analysis

We apply the proposed GMR approach to fit a finite GMM to an atmospheric dataset [3] named *CCSM run cam5.1.amip.2d.001* following Chen et al. (2013). These data are computer simulated based on Community Atmosphere Model version 5 (CAM5). The dataset contains daily observations of multiple atmospheric variables between years 1979 and 2005 over 192 longitudes (lon), 288 latitudes (lat), and 30 vertical altitude levels (lev). There variables are included: the moisture content (Q), temperature (T), and vertical velocity (Ω, OMEGA) of the air.

For ease of comparison, we analyze only observations in December, January, and February, i.e., winter in the northern hemisphere. The number of days is thus $2,430$, and the restriction reduces the variation in the dataset. At each surface location, we filter out non-wet days (less than 1 mm of daily precipitation) and focus on days with precipitation above the 95th wet-day percentile. This step reduces the number of observations at each location, not necessarily evenly. The analysis aims to cluster the locations according to the multivariate variable of dimension $d = 91$: 30 lev $\times$ $\{$Q, T, $\Omega\}$ plus the daily precipitation (PRECL) at the surface. Following Chen et al. (2013), we fit a finite GMM of order $K = 4$. They suggest that this model is helpful in identifying modes of extreme precipitation in 3D atmospheric space over a few atmospheric variables.

After this preprocessing, the dataset still takes about 3 GB of memory, so we cannot learn a global mixture in a reasonable time. We partition the dataset evenly into $M = 128$ subsets and apply the proposed GMR approach with the same numerical strategies as in the NIST experiments. For comparison, we also aggregate the local estimates by the KLA with 500 observations generated from each local estimate.

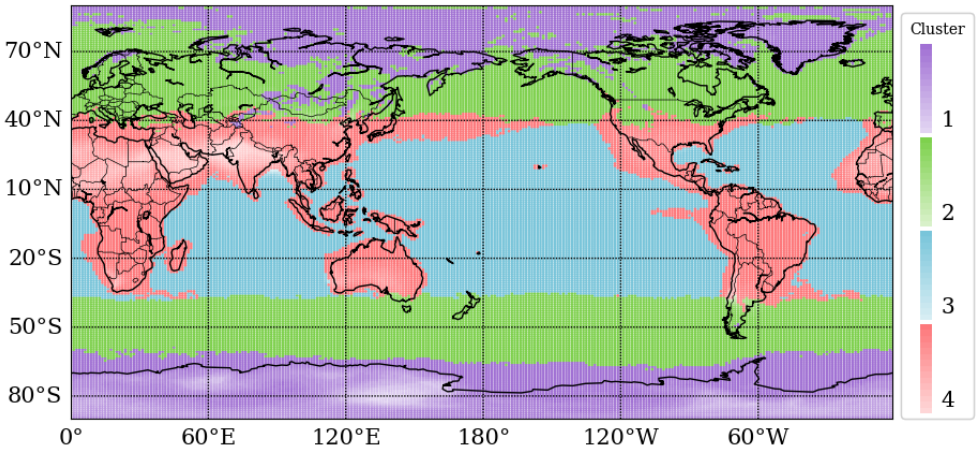


Figure 4: Surface locations colored by clusters. Within each cluster, the darker the color, the more wet days at that location.

Once a finite GMM is learned, we cluster the observations based on (21). Each combination of day and surface location is clustered into one of four subpopulations. To visualize

3. Available at `https://www.earthsystemgrid.org/dataset`.

the clusters, we further allocate each surface location to the cluster in which it belongs on most days. Figure 4 shows the geographical distribution of these four clusters represented by different colors. Similar to Chen et al. (2013), the GMR clusters reveal a strong latitudinal structure, they clearly separate the frigid, temperate, and tropical zones. The KLA results in similar clusters. Unlike Chen et al. (2013), the proposed GMR clusters are able to separate the continental and oceanic areas in the temperate zone.

8. Discussion and Concluding Remarks

We develop an effective split-and-conquer approach for learning finite GMMs. Our experiments show that it has good performance both statistically and computationally. We focus on finite GMMs, but with some adjustment our approach could be applied to learning mixtures with other subpopulation distributions such as Gamma and Poisson.

We have ignored many potential issues in this paper. In particular, we assume that the order of the mixture is correctly specified and that datasets on different local machines are IID and have the same underlying distributions. Our method is a preliminary attempt toward more satisfactory solutions to these real-world problems.

Acknowledgement

This research was enabled in part by support provided by WestGrid (`www.westgrid.ca`) and Compute Canada Calcul Canada (`www.computecanada.ca`). The authors would like to thank Mario Lučić for providing the code in Lucic et al. (2017). We would like thank the reviewer and action editor for their help and advice.

References

Rakesh Agrawal, Alexandre Evfimievski, and Ramakrishnan Srikant. Information sharing across private databases. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 86–97. ACM, 2003.

Sivaraman Balakrishnan, Martin J Wainwright, Bin Yu, et al. Statistical guarantees for the EM algorithm: From population to sample-based analysis. *The Annals of Statistics*, 45 (1):77–120, 2017.

Susan Baldwin. Compute canada: advancing computational research. In *Journal of Physics: Conference Series*, volume 341, page 012001. IOP Publishing, 2012.

Heather Battey, Jianqing Fan, Han Liu, Junwei Lu, and Ziwei Zhu. Distributed estimation and inference with statistical guarantees, 2015. Unpublished manuscript, arXiv:1509.05457.

Stephen Boyd, Neal Parikh, and Eric Chu. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

Olivier Cappé and Eric Moulines. On-line expectation–maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 71(3): 593–613, 2009.

Xiangyu Chang, Shao-Bo Lin, and Yao Wang. Divide and conquer local average regression. *Electronic Journal of Statistics*, 11(1):1326–1350, 2017.

Jiahua Chen. Optimal rate of convergence for finite mixture models. *The Annals of Statistics*, 23(1):221–233, 1995.

Jiahua Chen and Xianming Tan. Inference for multivariate normal mixtures. *Journal of Multivariate Analysis*, 100(7):1367–1383, 2009.

Sitan Chen, Jerry Li, and Zhao Song. Learning mixtures of linear regressions in subexponential time via fourier moments. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 587–600, 2020.

Wei-Chen Chen, George Ostrouchov, David Pugmire, Prabhat, and Michael Wehner. A parallel EM algorithm for model-based clustering applied to the exploration of large spatio-temporal data. *Technometrics*, 55(4):513–523, 2013.

Xueying Chen and M. Xie. A split-and-conquer approach for analysis of extraordinarily large data. *Statistica Sinica*, 24(4):1655–1684, 2014.

James C Corbett, Jeffrey Dean, Michael Epstein, Andrew Fikes, Christopher Frost, Jeffrey John Furman, Sanjay Ghemawat, Andrey Gubarev, Christopher Heiser, Peter Hochschild, et al. Spanner: Google's globally distributed database. *ACM Transactions on Computer Systems (TOCS)*, 31(3):Article 8, 2013.

David F Crouse, Peter Willett, Krishna Pattipati, and Lennart Svensson. A look at Gaussian mixture reduction algorithms. In *14th International Conference on Information Fusion*, pages 1–8. IEEE, 2011.

Marco Cuturi and Arnaud Doucet. Fast computation of Wasserstein barycenters. In *International Conference on Machine Learning*, pages 685–693, 2014.

Suresh Dara and Priyanka Tumma. Feature extraction by using deep learning: A survey. In *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, pages 1795–1801. IEEE, 2018.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B*, 39:1–38, 1977.

Raaz Dwivedi, Nhat Ho, Koulik Khamaru, Martin Wainwright, Michael Jordan, and Bin Yu. Sharp analysis of expectation-maximization for weakly identifiable models. In *International Conference on Artificial Intelligence and Statistics*, pages 1866–1876, 2020.

Jianqing Fan, Dong Wang, Kaizheng Wang, and Ziwei Zhu. Distributed estimation of principal eigenspaces. *Annals of Statistics*, 47(6):3009–3031, 2019.

Dan Feldman, Matthew Faulkner, and Andreas Krause. Scalable training of mixture models via coresets. In *Advances in neural information processing systems*, pages 2142–2150, 2011.

Chris Fraley and Adrian E Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97(458):611–631, 2002.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The Elements of Statistical Learning*, volume 1. Springer Series in Statistics New York, 2001.

Sylvia Frühwirth-Schnatter. *Finite Mixture and Markov Switching Models*. Springer Science & Business Media, 2006.

Patrick Grother and Kayee Hanaoka. NIST special database 19 handprinted forms and characters 2nd edition. Technical report, National Institute of Standards and Technology, 2016.

Philippe Heinrich and Jonas Kahn. Strong identifiability and optimal minimax rates for finite mixture estimation. *The Annals of Statistics*, 46(6A):2844–2870, 2018.

David R Hunter and Kenneth Lange. A tutorial on MM algorithms. *The American Statistician*, 58(1):30–37, 2004.

Martin Jaggi, Virginia Smith, Martin Takac, Jonathan Terhorst, Sanjay Krishnan, Thomas Hofmann, and Michael I Jordan. Communication-efficient distributed dual coordinate ascent. In *Advances in Neural Information Processing Systems 27*, pages 3068–3076, 2014.

Priyank Jaini and Pascal Poupart. Online and distributed learning of Gaussian mixture models by Bayesian moment matching. *arXiv preprint arXiv:1609.05881*, 2016.

Roman Liesenfeld. A generalized bivariate mixture model for stock price volatility and trading volume. *Journal of Econometrics*, 104(1):141–178, 2001.

Qiang Liu and Alexander T Ihler. Distributed estimation, information loss and exponential families. In *Advances in Neural Information Processing Systems 27*, pages 1098–1106, 2014.

Mario Lucic, Matthew Faulkner, Andreas Krause, and Dan Feldman. Training gaussian mixture models at scale via coresets. *The Journal of Machine Learning Research*, 18(1): 5885–5909, 2017.

Ranjan Maitra and Volodymyr Melnykov. Simulating data to study performance of finite mixture modeling and clustering algorithms. *Journal of Computational and Graphical Statistics*, 19(2):354–376, 2010.

Geoffrey McLachlan and David Peel. *Finite Mixture Models*. John Wiley & Sons, 2004.

Volodymyr Melnykov, Wei-Chen Chen, and Ranjan Maitra. MixSim: An R package for simulating data to study performance of clustering algorithms. *Journal of Statistical Software*, 51(12):1–25, 2012.

Kerrie L Mengersen, Christian Robert, and Mike Titterington. *Mixtures: estimation and applications*, volume 896. John Wiley & Sons, 2011.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Radford M Neal and Geoffrey E Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. Springer, 1998.

XuanLong Nguyen. Convergence of latent mixing measures in finite and infinite mixture models. *The Annals of Statistics*, 41(1):370–400, 2013.

Robert D Nowak. Distributed EM algorithms for density estimation and clustering in sensor networks. *IEEE Transactions on Signal Processing*, 51(8):2245–2253, 2003.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8026–8037, 2019.

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

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Gabriel Peyré and Marco Cuturi. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5–6):355–607, 2019.

Judith Rousseau and Kerrie Mengersen. Asymptotic behaviour of the posterior distribution in overfitted mixture models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 73(5):689–710, 2011.

Behrooz Safarinejadian, Mohammad B Menhaj, and Mehdi Karrari. A distributed EM algorithm to estimate the parameters of a finite mixture of components. *Knowledge and Information Systems*, 23(3):267–292, 2010.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2): 461–464, 1978.

Aad W Van der Vaart. *Asymptotic Statistics*, volume 3. Cambridge University Press, 2000.

Cédric Villani. *Topics in Optimal Transportation*, volume 58. American Mathematical Society, 2003.

Jason L Williams and Peter S Maybeck. Cost-function-based hypothesis control techniques for multiple hypothesis tracking. *Mathematical and Computer Modelling*, 43(9–10):976–989, 2006.

CF Jeff Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.

Willard I Zangwill. *Nonlinear Programming: A Unified Approach*, volume 52. Prentice-Hall Englewood Cliffs, NJ, 1969.

Yuchen Zhang, John Duchi, and Martin Wainwright. Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates. *The Journal of Machine Learning Research*, 16(1):3299–3340, 2015.

Appendices

The appendix is organized as follows. Section 8.1 contains all left over technical details and proofs. Section 8.2 provides additional details in the experiment.

8.1 Proofs

8.1.1 Example 2: Technical Details.

Let

$$\mathbb{D}(G) = 0.5\mathbb{W}_D^2(G_1, G) + 0.5\mathbb{W}_D^2(G_2, G).$$

By definition, the barycenter minimizes $\mathbb{D}(G)$. Recall we suggest that the barycenter is given by

$$\Phi(x; \overline{G}^C) = 0.4\Phi_{-1} + 0.6\Phi_{2/3}$$

instead of

$$\Phi(x; \overline{G}) = 0.5\Phi_{-1} + 0.5\Phi_1.$$

The full proof is overly tedious. Instead, we verify that $\mathbb{D}(\overline{G}^C) < \mathbb{D}(\overline{G})$ so that $\overline{G}$ is not a barycenter but the suggested $\overline{G}^C$ is likely the one.

Note that all transportation plans from G_1 and G_2 to the presumed barycenter $\overline{G}^C$ have the form

$$\begin{pmatrix} p & 0.4-p \\ 0.4-p & 0.2+p \end{pmatrix} \text{ and } \begin{pmatrix} p & 0.6-p \\ 0.4-p & p \end{pmatrix},$$

respectively, for some p between 0 and 0.4. These two matrices are bivariate probability mass functions with the marginal probability masses $(0.4, 0.6)$ and $(0.6, 0.4)$ as required. The cost functions may be presented as

$$\begin{pmatrix} c(-1,-1) & c(-1,2/3) \\ c(1,-1) & c(1,2/3) \end{pmatrix} = \begin{pmatrix} 0 & 25/9 \\ 4 & 1/9 \end{pmatrix}.$$

It is clear that $p = 0.4$ gives the optimal plans for transporting G_1 to $\overline{G}^C$ and G_2 to $\overline{G}^C$. With these plans in place, we can see that

$$\mathbb{D}(\overline{G}^C) = 0.5\mathbb{W}_D^2(G_1, \overline{G}^C) + 0.5\mathbb{W}_D^2(G_2, \overline{G}^C) = 1/3.$$

In comparison, the optimal transportation plan from G_1 to $\overline{G}$ is to move 0.1 mass from Φ_1 to Φ_{-1} with a total cost of $0.1 \times 4 = 0.4$. Hence,

$$\mathbb{D}(\overline{G}) = 0.5\mathbb{W}_D^2(G_1, \overline{G}) + 0.5\mathbb{W}_D^2(G_2, \overline{G}) = 0.4 > 1/3.$$

That is, $\overline{G}$ is not a barycenter.

8.1.2 Proof of Theorem 5

The key conclusion of this theorem is

$$\inf\{\mathcal{T}_c(G) : G \in \mathbb{G}_K\} = \inf\{\mathcal{J}_c(G) : G \in \mathbb{G}_K\}.$$

We prove this result by showing both

$$\inf\{\mathcal{T}_c(G) : G \in \mathbb{G}_K\} \geq \inf\{\mathcal{J}_c(G) : G \in \mathbb{G}_K\} \tag{22}$$

and

$$\inf\{\mathcal{T}_c(G) : G \in \mathbb{G}_K\} \leq \inf\{\mathcal{J}_c(G) : G \in \mathbb{G}_K\}. \tag{23}$$

We first prove (22). Let $G^* = \arg\inf\{\mathcal{J}_c(G) : G \in \mathbb{G}_K\}$. Denote the mixing weights of any $G \in \mathbb{G}_K$ as $\boldsymbol{v}(G)$, and the subpopulations prescribed by G as Φ_γ. According to (15), we have

$$\boldsymbol{v}(G^*) = \sum_i \pi_{i,\gamma}(G^\star),$$

which implies that $\boldsymbol{\pi}(G^*) \in \Pi(\cdot, \boldsymbol{v}(G^*))$. Since $\boldsymbol{\pi}(G^*) \in \Pi(\boldsymbol{w}, \cdot)$ by (10), we also have that $\boldsymbol{\pi}(G^*) \in \Pi(\boldsymbol{w}, \boldsymbol{v}(G^*))$. In other words, $\boldsymbol{\pi}(G^*)$ is a valid transportation plan from $\overline{G}$ to G^*. Consequently,

$$\inf\{\mathcal{T}_c(G) : G \in \mathbb{G}_K\} \leq \sum_{i,\gamma} \pi_{i\gamma}(G^*) c(\Phi_i, \Phi^*_\gamma) = \mathcal{J}_c(G^*) = \inf\{\mathcal{J}_c(G) : G \in \mathbb{G}_K\},$$

with the last equality implied by the definition of G^*. This proves (22).

Next, we prove (23). Let $G^\dagger = \inf\{\mathcal{T}_c(G) : G \in \mathbb{G}_K\}$, the solution to the optimization on the left-hand side of (23). We denote the subpopulations prescribed by $G^\dagger$ as $\Phi^\dagger_\gamma$. Let

$$\boldsymbol{\pi}^\dagger = \arg\inf\big\{\sum_{i,\gamma} \boldsymbol{\pi}_{i\gamma} c(\overline{\Phi}_i, \Phi^\dagger_\gamma) : \boldsymbol{\pi} \in \Pi(\boldsymbol{w}, \boldsymbol{v}(G^\dagger))\big\},$$

which is the optimal transportation plan from $\overline{G}$ to this $G^\dagger$. Because of this, we have

$$\inf\{\mathcal{T}_c(G) : G \in \mathbb{G}_K\} = \mathcal{T}_c(G^\dagger) = \sum_{i,\gamma} \boldsymbol{\pi}^\dagger_{i\gamma} c(\overline{\Phi}_i, \Phi^\dagger_\gamma) \geq \mathcal{J}_c(G^\dagger) \geq \inf\{\mathcal{J}_c(G) : G \in \mathbb{G}_K\}.$$

The last step holds because $\boldsymbol{\pi}^\dagger \in \Pi(\boldsymbol{w}, \cdot)$. This proves (23).

8.1.3 Proof of Theorem 7

Theorem 7 claims that the MM-iteration leads to (i) decreasing sequence $\mathcal{J}_c(G^{(t)})$ in t; and (ii) stationary limiting points G^* of $\{G^{(t)})\}$.

Proof of **(i)**: Clearly, we have $\mathcal{K}_c(G|G^{(t)}) \geq \mathcal{J}_c(G)$ for all G with equality holds at $G = G^{(t)}$. Hence,

$$\begin{aligned}
\mathcal{J}_c(G^{(t)}) &\geq \mathcal{J}_c(G^{(t)}) - \{\mathcal{K}_c(G^{(t+1)}|G^{(t)}) - \mathcal{J}_c(G^{(t+1)})\} \\
&= \mathcal{J}_c(G^{(t+1)}) - \{\mathcal{K}_c(G^{(t+1)}|G^{(t)}) - \mathcal{J}_c(G^{(t)})\} \\
&\geq \mathcal{J}_c(G^{(t+1)}) - \{\mathcal{K}_c(G^{(t)}|G^{(t)}) - \mathcal{J}_c(G^{(t)})\} \\
&= \mathcal{J}_c(G^{(t+1)}).
\end{aligned}$$

This is the property that an MM-algorithm must have.

Proof of **(ii)**. Suppose $G^{(t)}$ has a convergent subsequence leading to a limit G^*. Let this subsequence be $G^{(t_k)}$. By Helly's selection theorem (Van der Vaart, 2000), there is a subsequence s_k of t_k such that $G^{(s_k+1)}$ has a limit, say G^{**}. These limits, however, could be subprobability distributions. That is, we cannot rule out the possibility that the total probability in the limit is below 1 by Helly's theorem.

This is not the case under the theorem conditions. Let $\Delta > 0$ be large enough such that

$$A_1 = \{\Phi : c(\overline{\Phi}_i, \Phi) \leq \Delta, \text{for all subpopulations } \overline{\Phi}_i \text{ of } \overline{G}\}$$

is not empty. With this Δ, we define

$$A_2 = \{\Phi : c(\overline{\Phi}_i, \Phi) > \Delta, \text{for all subpopulations } \overline{\Phi}_i \text{ of } \overline{G}\}.$$

Suppose $G^\dagger$ has a subpopulation $\Phi^\dagger$ such that $c(\overline{\Phi}_i, \Phi^\dagger) > \Delta$ for all i. Replacing this subpopulation in $G^\dagger$ by any $\Phi^{\dagger\dagger} \in A_1$ to form $G^{\dagger\dagger}$, we can see that for any t,

$$\mathcal{K}_c(G^\dagger | G^{(t-1)}) > \mathcal{K}_c(G^{\dagger\dagger} | G^{(t-1)}).$$

Because $G^{(t)}$ minimizes $\mathcal{K}_c(G|G^{(t-1)})$ in G, the above result shows that none of the subpopulations of $G^{(t)}$ are members of A_2.

Note that the complement of A_2 is compact by condition (18). Consequently, the subpopulations of $G^{(t)}$ are confined to a compact subset. Hence, all limit points of $G^{(t)}$, including both G^* and G^{**}, are proper distributions. By the monotonicity of the iteration:

$$\mathcal{J}_c(G^{(s_{k+1})}) \leq \mathcal{J}_c(G^{(s_k+1)}) \leq \mathcal{J}_c(G^{(s_k)}).$$

Let $k \to \infty$, we get

$$\mathcal{J}_c(G^{**}) = \mathcal{J}_c(G^*). \tag{24}$$

By the definition of the MM iteration, we have

$$\mathcal{K}_c(G^{(s_k+1)} | G^{(s_k)}) \leq \mathcal{K}_c(G | G^{(s_k)}).$$

Let $k \to \infty$ and by the continuity of $\mathcal{K}_c(\cdot|\cdot)$, we get

$$\mathcal{K}_c(G^{**} | G^*) \leq \mathcal{K}_c(G | G^*).$$

Hence, G^{**} is a solution to $\min \mathcal{K}_c(G|G^{t)})$ when $G^{(t)} = G^*$. Namely, we have $\mathcal{K}_c(G^{**}|G^*) = \mathcal{K}_c(G^{(t+1)}|G^*)$. With the help of (24), it further implies

$$\mathcal{J}_c(G^{**}) = \mathcal{J}_c(G^{(t+1)}) = \mathcal{J}_c(G^*)$$

when $G^{(t)} = G^*$. This shows that iteration from $G^{(t)} = G^*$ does not make $\mathcal{J}_c(G^{(t+1)})$ smaller than $\mathcal{J}_c(G^{(t)})$. Hence, G^* is a stationary point of the MM iteration. This is conclusion (ii) and we have completed the proof.

8.1.4 Proof of Theorem 9

Recall that the local estimators $\widehat{G}_m, m \in [M]$ are strongly consistent for G^* when the order K of G^* is known. This implies that the simple average $\overline{G} \to G^*$ and $\mathcal{T}_c(\overline{G}, G^*) \to 0$ almost surely. Because $\mathcal{T}_c(\cdot, \cdot)$ is not necessarily a mathematical distance, we do not have an easy-to-use triangle inequality. The subsequent steps may appear tedious but unavoidable.

By "almost surely", it implies other than a probability 0 event in the probability space Ω on which the random variables are defined, $\mathcal{T}_c(\overline{G}, G^*) \to 0$ holds. Without loss of generality, assume $\mathcal{T}_c(\overline{G}, G^*) \to 0$ holds at all $\omega \in \Omega$ without a zero-probability exception. With this, $\mathcal{T}_c(\overline{G}, G^*) \to 0$ holds only if each support point of $\overline{G}$ converges to one of those of G^*. The total weights of the support points of $\overline{G}$ converging to the same support of G^* must converge to the corresponding weight of G^*.

By definition, $\overline{G}^R$ has K support points. We also notice that

$$\mathcal{T}_c(\overline{G}, \overline{G}^R) \leq \mathcal{T}_c(\overline{G}, G^*) \to 0. \tag{25}$$

Suppose that $\overline{G}^R$ does not converge to G^* at some $\omega \in \Omega$. One possibility is that the smallest mixing weight of $\overline{G}^R$ (or a subsequence thereof) goes to zero as $N \to \infty$. In this case, $\overline{G}^R$ has $K - 1$ or fewer meaningful support points. Since the support points of $\overline{G}$ are in an infinitesimal neighborhood of those of G^*, one of them must be a distance away from any of the support points of $\overline{G}^R$. Therefore, by Condition 4, the transportation cost of this support point is larger than a positive constant not depending on N. The positive transportation cost implies that $\mathcal{T}_c(\overline{G}, \overline{G}^R) \not\to 0$, which contradicts (25).

The next possibility is that the smallest mixing weight of $\overline{G}^R$ does not go to zero. In this case, there is a subsequence such that all the mixing weights converge to positive constants. Without loss of generality, all the mixing weights simply converge to positive constants as $N \to \infty$. If there is a subsequence of support points of $\overline{G}^R$ that is at least ϵ-distance away from any of the support points of G^*, then the transportation cost from $\overline{G}$ to this support point will be larger than a positive constant not depending on N. This again leads to a contradiction to (25).

The final possibility is that $\overline{G}^R$ (or a subsequence thereof) has a proper limit, say $G^{**} \neq G^*$. If so, $\mathcal{T}_c(\overline{G}, \overline{G}^R) \to \mathcal{T}_c(G^*, G^{**}) \neq 0$, contradicting (25).

We have exhausted all the possibilities. Hence, the consistency claim is true.

8.1.5 Proof of Theorem 10

Theorem 10 states that the GMR estimator $\overline{G}^R$ retains the convergence rate $N^{-1/2}$, under the key condition C1 that the order K of the finite mixture model is known.

Let Φ_{mk} be the kth subpopulation learned at local machine m and w_{mk} be its mixing weight. Note that we do not put a "hat" on them for notation simplicity. According to Lemma 8 on the rate of convergence of the pMLE at local machines, these subpopulations can be arranged so that for all $m \in [M]$ and $k \in [K]$, we have

$$\|\Phi_{mk} - \Phi_k^*\| = O_p(N^{-1/2}), \quad w_{mk} - w_k^* = O_p(N^{-1/2}).$$

By C5, the first rate conclusion above implies

$$\max\{c(\Phi_{mk}, \Phi_k^*) : k \in [K]\} = O_p(N^{-1}).$$

For each k, let $\widetilde{w}_k = \sum_{m=1}^{M} \lambda_m w_{mk}$ and $\widetilde{\Phi}_k$ be the local barycenter of Φ_{mk}, $m \in [M]$:

$$\widetilde{\Phi}_k = \arg\min\{\Phi : \sum_{m=1}^{M} \lambda_m w_{mk} c(\Phi_{mk}, \Phi)\}.$$

Let $\tilde{G}$ be the mixing distribution with weights $\widetilde{w}_k$ and subpopulations $\widetilde{\Phi}_k$. By the rate conclusions given earlier, we have $\widetilde{w}_k = w_k^* + O_p(N^{-1/2})$ for $k \in [K]$. By C5, we must also have

$$\|\widetilde{\Phi}_k - \Phi_k^*\| = O_p(N^{-1/2})$$

and $\mathcal{T}_c(\overline{G}, \widetilde{G}) = O_p(N^{-1})$. If the GMR $\overline{G}^R = \widetilde{G}$, then the rate conclusion of the theorem would have been proved.

Next, we show $\overline{G}^R = \widetilde{G}$ asymptotically. By theorem conditions, the true subpopulations Φ_k^* are all distinct. Hence, by condition C4, we have

$$\min\{c(\Phi_k^*, \Phi_{k'}^*) : k \neq k' \in [K]\} > 0.$$

Thus, if the subpopulations of $\overline{G}^R$ is not in an $o_p(1)$ neighbourhood of one of Φ_k^* even though everyone of $\overline{G}$ is, the transport cost to this subpopulation from any subpopulation of $\overline{G}$ exceeds a positive constant in probability. This contradicts

$$\mathcal{T}_c(\overline{G}, \overline{G}^R) \leq \mathcal{T}_c(\overline{G}, \widetilde{G}) = O_p(N^{-1}). \tag{26}$$

This implies, all subpopulations of $\overline{G}^R$ are within $o_p(1)$ neighbourhood of one of Φ_k^*. Denote these subpopulations as $\overline{\Phi}_k^R$. The optimal plan must transport Φ_{mk} to $\overline{\Phi}_k^R$, otherwise the total transport cost exceeds a positive constant in probability which again contradicts (26). Since $\overline{G}^R$ minimizes the transport cost, we must have $\overline{\Phi}_k^R = \widetilde{\Phi}_k$, the local barycenter. These conclusions imply that GMR $\overline{G}^R = \widetilde{G}$ with probability approaching to 1. Consequently, the rates of convergence of $\widetilde{w}_k, \widetilde{\Phi}_k$ extend to those of $\overline{G}^R$ and this completes the proof.

8.1.6 KL-divergence satisfies C5

Let $\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1$ and $\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2$ be the parameters of Φ_1 and Φ_2. It is known that

$$2D_{\mathrm{KL}}(\Phi_1\|\Phi_2) = \log\{\det(\boldsymbol{\Sigma}_2)/\det(\boldsymbol{\Sigma}_1)\} + \mathrm{tr}(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1 - \mathbf{I}_d) + (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^\tau \boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1).$$

Assume both Φ_1 and Φ_2 are in a small neighborhood of Φ whose covariance matrix $\boldsymbol{\Sigma}$ is positive definite. Hence, eigenvalues of $\boldsymbol{\Sigma}_2$ are in small neighborhood of these of $\boldsymbol{\Sigma}$. Thus, there exists a positive constant A_1 such that the third term in $2D_{\mathrm{KL}}(\Phi_1\|\Phi_2)$ satisfies

$$A_1^{-1}\|\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\|^2 \leq (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^\tau \boldsymbol{\Sigma}_2^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) \leq A_1\|\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1\|^2 \tag{27}$$

Let $\lambda_1, \ldots, \lambda_d$ be eigenvalues of $\boldsymbol{\Sigma}_2^{-1/2}\boldsymbol{\Sigma}_1\boldsymbol{\Sigma}_2^{-1/2}$. Since both $\boldsymbol{\Sigma}_1$ and $\boldsymbol{\Sigma}_2$ are in a small neighborhood of $\boldsymbol{\Sigma}$, we have $\lambda_1, \ldots, \lambda_d$ all close to 1. Note that

$$\log\{\det(\boldsymbol{\Sigma}_2)/\det(\boldsymbol{\Sigma}_1)\} + \mathrm{tr}(\boldsymbol{\Sigma}_2^{-1}\boldsymbol{\Sigma}_1 - \mathbf{I}_d) = \sum_{j=1}^{d}\{(\lambda_j - 1) - \log\lambda_j\}.$$

Note that $(\lambda-1)-\log\lambda$ is a convex function with its minimum attained at $\lambda=1$ at which point its second derivative equals 1. Hence, there exists an $A_2>0$ such that

$$A_2^{-1}(\lambda-1)^2 \le (\lambda-1)-\log\lambda \le A_2(\lambda-1)^2.$$

We have therefore shown that

$$A_2^{-1}\sum_{j=1}^{d}(\lambda_j-1)^2 \le \log\{\det(\mathbf{\Sigma}_2)/\det(\mathbf{\Sigma}_1)\} + \mathrm{tr}(\mathbf{\Sigma}_2^{-1}\mathbf{\Sigma}_1-\mathbf{I}_d) \le A_2\sum_{j=1}^{d}(\lambda_j-1)^2. \quad (28)$$

We now connect this bound with Frobenius norm.

For a positive definite matrix $\mathbf{\Sigma}$, it is easy to see that $\|\mathbf{\Sigma}-\mathbf{I}_d\|_F^2=\sum_{j=1}^{d}(\sigma_j-1)^2$, where $\sigma_1,\sigma_2,\ldots,\sigma_d$ are eigenvalues of $\mathbf{\Sigma}$. Frobenius norm also has sub-multiplicative property

$$\|\mathbf{\Sigma}_1\mathbf{\Sigma}_2\|_F \le \|\mathbf{\Sigma}_1\|_F\|\mathbf{\Sigma}_2\|_F.$$

Applying the sub-multiplicative property in our context, we get

$$\begin{aligned}\|\mathbf{\Sigma}_1-\mathbf{\Sigma}_2\|_F^2 &\le \|\mathbf{\Sigma}_2^{1/2}\|_F^4\ \|\mathbf{\Sigma}_2^{-1/2}\mathbf{\Sigma}_1\mathbf{\Sigma}_2^{-1/2}-\mathbf{I}_d\|_F^2\\ &\le A_3\ \|\mathbf{\Sigma}_2^{-1/2}\mathbf{\Sigma}_1\mathbf{\Sigma}_2^{-1/2}-\mathbf{I}_d\|_F^2 = A_3\ \sum_{j=1}^{d}(\lambda_j-1)^2\end{aligned}$$

for some local positive constant $A_3>\|\mathbf{\Sigma}^{1/2}\|_F^4$, as both matrices are in a small neighborhood of Σ. Similarly, we have

$$\begin{aligned}\sum_{j=1}^{d}(\lambda_j-1)^2 &= \|\mathbf{\Sigma}_2^{-1/2}\mathbf{\Sigma}_1\mathbf{\Sigma}_2^{-1/2}-\mathbf{I}_d\|_F^2 = \|\mathbf{\Sigma}_2^{-1/2}\{\mathbf{\Sigma}_1-\mathbf{\Sigma}_2\}\mathbf{\Sigma}_2^{-1/2}\|_F^2\\ &\le \|\mathbf{\Sigma}_2^{-1/2}\|_F^4\ \|\mathbf{\Sigma}_1-\mathbf{\Sigma}_2\|_F^2 \le A_4\|\mathbf{\Sigma}_1-\mathbf{\Sigma}_2\|_F^2.\end{aligned}$$

for some positive constant A_4. This leads to

$$\|\mathbf{\Sigma}_1-\mathbf{\Sigma}_2\|_F^2 \ge A_4^{-1}\sum_{j=1}^{d}(\lambda_j-1)^2.$$

Let $A=2\max\{A_1,A_2,A_3,A_4\}$. Applying (27) and (28), we have

$$A^{-1}\{\|\boldsymbol{\mu}_1-\boldsymbol{\mu}_2\|^2+\|\mathbf{\Sigma}_1-\mathbf{\Sigma}_2\|_F^2\} \le D_{\mathrm{KL}}(\Phi_1\|\Phi_2) \le A\{\|\boldsymbol{\mu}_1-\boldsymbol{\mu}_2\|^2+\|\mathbf{\Sigma}_1-\mathbf{\Sigma}_2\|_F^2\}$$

when Φ_1,Φ_2 are in a small neighborhood of Φ, with A being a positive constant depends on Φ. This shows that the KL-divergence has property C5.

8.2 Other Details

8.2.1 Additional Simulation Results

We present simulation results for $K=5,10,50$ and $d=10,50$ as supplementary to Section 7.2. Figures 5–6 show the results for $N=2^{19}$ and $M=4$ with various combinations of

K and d. The panels in each figure are arranged so that the order of the mixture increases from left to right, and the dimension of the mixture increases from top to bottom.

Comparing panels within the same row in Figure 5, we note that the performance of each the estimators becomes worse as the order of the mixture increases in terms of W_1 distance. The panels within the same column in Figures 5 show that all the estimators become worse as the dimension of the mixture increases in terms of both performance measures.

Regardless, our estimator has performance comparable to that of the global estimator. In terms of the misclassification error, for the same degree of overlapping, the superiority of our estimator becomes even clearer compared to the KL-averaging as the number of components and the dimension increase.

The computational costs of the local estimators are typically low, and this gives our method an added computational advantage. However, this advantage is not guaranteed: see the bottom right panel in Figure 6, where $d = 50$, $K = 50$, and the degree of overlapping is above 1%. There are many other factors at play. A more skillful implementation may lead to different conclusions on the computational time.

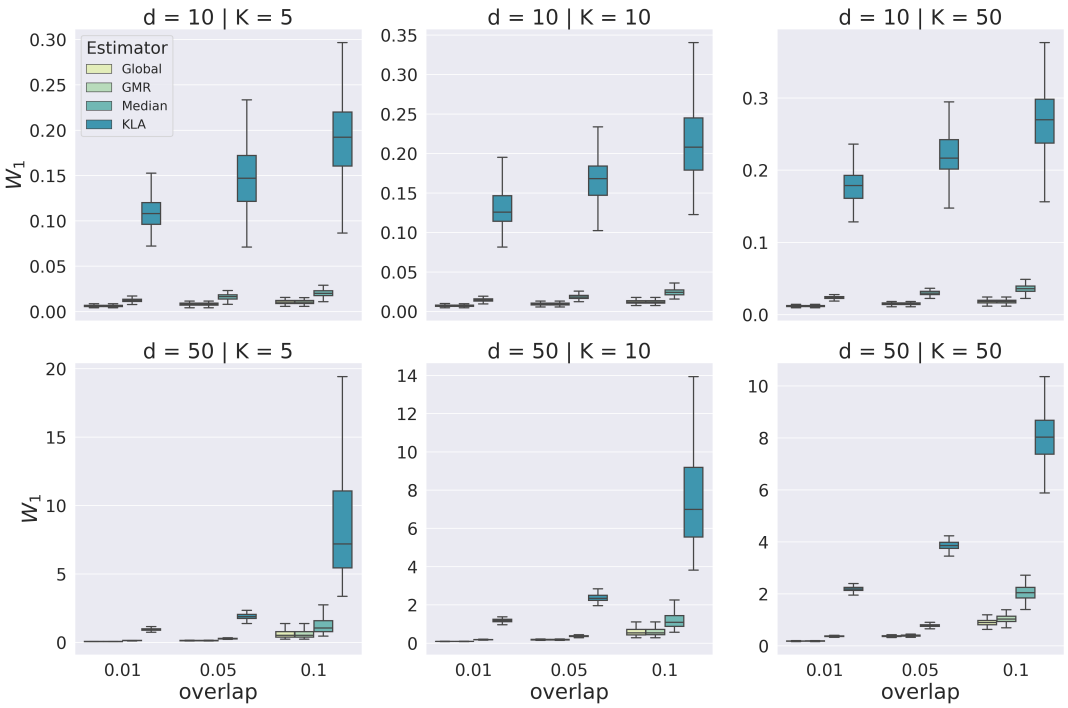


Figure 5: W_1 distances of estimators for learning mixtures.

8.2.2 Convolutional Neural Network in NIST Example

Deep convolutional neural networks (CNNs) are commonly used to reduce the complex structure of a dataset to informative rectangle data. CNNs effectively perform dimension reduction and classification in an end-to-end fashion (Dara and Tumma, 2018). The final soft-max layer in a CNN can be viewed as fitting a multinomial logistic regression model on

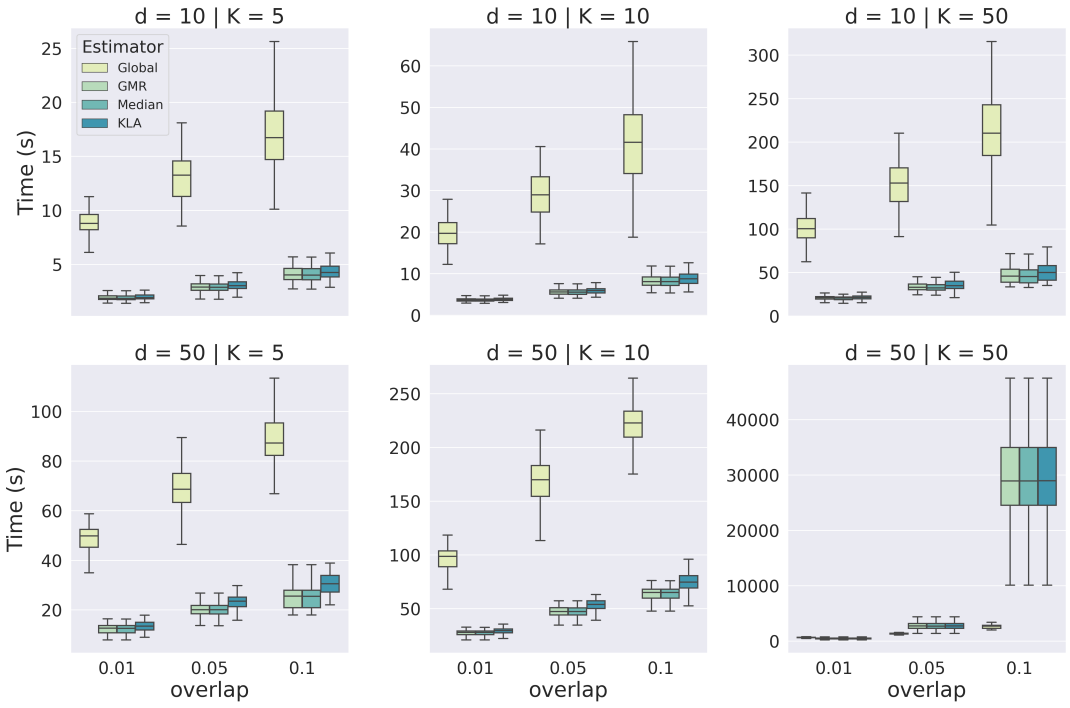


Figure 6: Computation times for learning mixtures.

the reduced feature space. We use a CNN for dimension reduction in the NIST experiment; its architecture is specified in Table 3. We implement the CNN in *pytorch 1.5.0* (Paszke et al., 2019) and train it for 10 epochs on the NIST training dataset. We use the SGD optimizer with learning rate 0.01, momentum 0.9, and batch size 64. After training, we drop the final layer and use the resulting CNN to reduce the dimension to 50 for both the training and test sets.

Table 3: Architecture and layer specifications of CNN for dimension reduction in NIST example.

Layer	**Layer specification**	**Activation function**
Conv2d	$C_{\text{in}} = 1$, $C_{\text{out}} = 20$, $H = W = 5$	Relu
MaxPool2d	$k = 2$	–
Conv2d	$C_{\text{in}} = 20$, $C_{\text{out}} = 50$, $H = W = 5$	Relu
MaxPool2d	$k = 2$	–
Flatten	–	–
Linear	$H_{\text{in}} = 800$, $H_{\text{out}} = 50$	Relu
Linear	$H_{\text{in}} = 50$, $H_{\text{out}} = 10$	Softmax