Clustering of large-scale Gaussian Mixture Model (GMM) via accelerated model averaging methods

Amir-Salar Esteki, Hossein Moradian and Solmaz S. Kia, *Senior Member, IEEE*

I. Overview

In this project, we study the problem of accelerating reaching true estimates of a clustering problem when the dataset is distributed over a network of devices. Each device in the network has access to a limited portion of data, therefore, the devices need to communicate in order to achieve the solution of the clustering problem. By incorporating the expectation-maximization method, each device calculates the local GMM parameters and shares these parameters with its neighbors in the network at each iteration. After a pre-specified number of communications, agents acquire an estimate of the true parameters of the global model. We propose a novel algorithm that accelerates the convergence process and the devices of the network achieve a more accurate estimate of the GMM model within the pre-specified number of communications. A comparison between the proposed method and existing algorithms in the literature is conducted. For more details about the method used here, please refer to the main publication at Main Paper.

II. Distributed Clustering of a Gaussian Mixture Model (GMM)

Distributed clustering is a technique used in machine learning to divide a large dataset into smaller sub-datasets, which are then processed separately on different nodes in a cluster. The goal of this process is to improve the scalability, efficiency, and accuracy of the clustering algorithms. In distributed clustering, the data is partitioned and each node in the cluster is responsible for processing a portion of the data. This allows for parallel processing of the data, reducing the time required to complete the clustering algorithm. Moreover, it allows for handling of very large datasets, which would otherwise be too large to be processed on a single machine. Distributed clustering has numerous applications, including data analysis, pattern recognition, and customer segmentation. However, it also presents some challenges, such as dealing with heterogeneity of the nodes in the cluster, communication overhead, and coordination of the nodes. These challenges can be overcome by careful design and implementation of the algorithms.

In this project, we reformulate the distributed clustering problem into a Gaussian Mixture Model to be estimated. In order to solve this problem in a distributed manner, agents of the network communicate to share their estimations and obtain an average. In general, average consensus is an important tool to enable many distributed schemes in networked systems. To demonstrate the benefit of using our accelerated average consensus, we conduct a simulation study of a distributed expectation-maximization (EM) algorithm used in sensor networks to obtain a Gaussian Mixture Model (GMM) of a set of observed targets. In our case study, the setting consists of N agents observing the location $\mathbf{p} \in \mathbb{R}^2$ of M targets in a 2D plane. The agents want to collaboratively obtain the GMM model of the distribution of the targets, i.e., obtain the weight, mean and covariance of basis of the GMM model, i.e., $(\pi_l, \boldsymbol{\mu}_l, \Sigma_l)$ in $\hat{f}(\mathbf{p}) = \sum_{l=1}^{N_s} \pi_l \mathcal{N}(\mathbf{p}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)$, where N_s is the number of the bases of the GMM model which is known to all agents. A popular method to construct a GMM with a determined number of bases N_s from observed data is the EM algorithm which is an iterative method that alternates between an expectation (E) step and a maximization (M) step. In the E-step, the posterior probability is computed as

$$\zeta_{ln} := \Pr(z = l|\mathbf{p}_n) = \frac{\pi_l \mathcal{N}(\mathbf{p}_n|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}{\sum_{j=1}^{N_s} \pi_j \mathcal{N}(\mathbf{p}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{1}$$

using the target points $\mathbf{p}_n$, $n \in \{1, \cdots, M\}$ and the current values of $\{\pi_l, \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l\}_{l=1}^{N_s}$. Next, in the M-step, the parameters of each $l \in \{1, \cdots, N_s\}$ are updated by

$$\pi_l = \frac{\sum_{n=1}^{M} \zeta_{ln}}{M}, \tag{2a}$$

$$\boldsymbol{\mu}_l = \frac{\sum_{n=1}^{M} \zeta_{ln} \mathbf{p}_n}{\sum_{n=1}^{M} \zeta_{ln}}, \tag{2b}$$

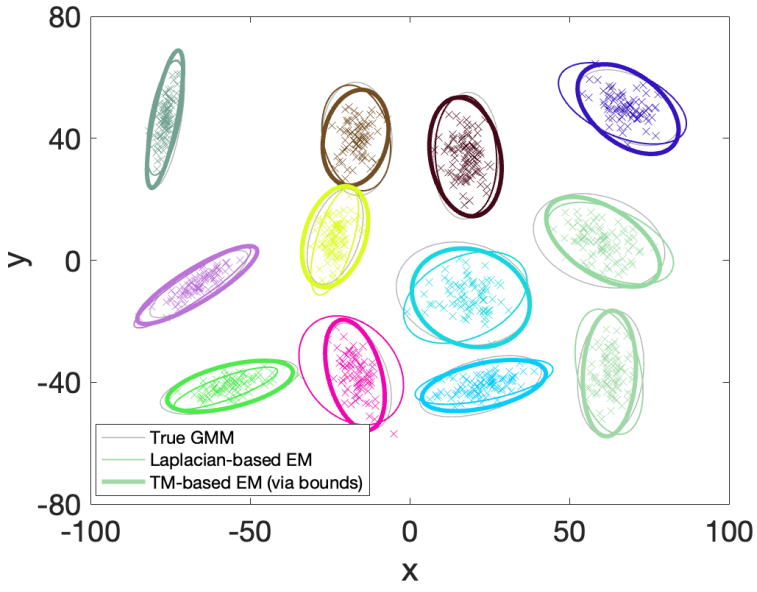
$$\boldsymbol{\Sigma}_l = \frac{\sum_{n=1}^{M} \zeta_{ln} (\mathbf{p}_n - \boldsymbol{\mu}_l)(\mathbf{p}_n - \boldsymbol{\mu}_l)^\top}{\sum_{n=1}^{M} \zeta_{ln}}, \tag{2c}$$

using the current values of ζ_{ln}. In the distributed EM algorithm, each agent only observes a M^i number of targets; the set is given by $\mathcal{M}^i \subset \{1, \cdots, M\}$ where $\cup_{j=1}^{N} \mathcal{M}^j = \{1, \cdots, M\}$, $\mathcal{M}^i \cap \mathcal{M}^j = \{\}$. The distributed EM algorithms assume that each agent has a local copy of the GMM parameters. Each agent $i \in \{1, \cdots, N\}$ performs the E-step in (1) locally using its

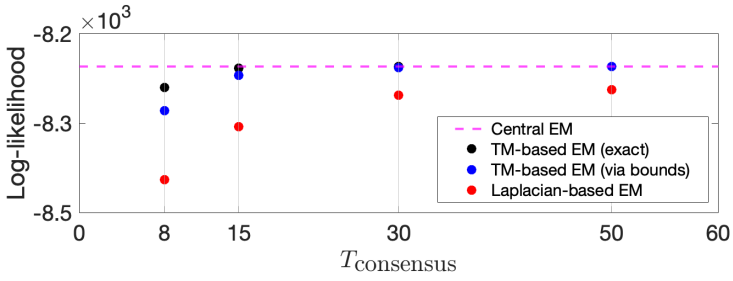
own GMM parameters for $n \in \mathcal{M}^i$. However, the summation terms in (2) are fragmented among the agents. Therefore, agents use a set of three average consensus algorithms to compute the summation terms that appear in the M-step (2).

In our numerical example, the number of agents is $N = 20$, and the agents communicate over a ring graph. These agents observe $M = 1000$ target points in a rectangle area of $(-80, 80) \times (-60, 60)$. The target points are drawn from a GMM model with $N_s = 12$ so that we can check the accuracy of the estimated GMM via distributed EM algorithms against this true model. Each agent initializes its $\{\pi_l, \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l\}_{l=1}^{N_s}$ locally. Let us denote $T_{\text{EM}} = 10$ as the number of iterations in the EM algorithm, and $T_{\text{consensus}}$ as the number of the consensus steps performed in each iteration of the M-step. First, we compare the performance of the EM algorithm when it uses the proposed Laplacian algorithm in (Laplacian-based EM) vs. when it uses our proposed TM-based algorithm. We conduct a set of four simulations for $T_{\text{consensus}} \in \{8, 15, 30, 50\}$. When using the TM algorithm, we consider two cases. In one, we assume that the agents know λ_2 and λ_N to compute the parameters of the TM algorithm (referred to as TM-based EM (exact)), and in the other case, we assume that agents replace $\lambda_2 = \frac{4}{N\text{diam}(\mathcal{G})}$ by its lower bound and λ_N with its upper bound $2\text{d}_{\max}$ (referred as TM-based EM (via bounds)).

Due to the limited space, we only show the results generated by agent 1 in all the simulations; the other agents have similar results. Fig. 1a depicts the 3σ-plot of the bases of the estimated GMM for $T_{\text{consensus}} = 8$. Here, the thin gray, the thin colored, and the thick colored ellipses represent, respectively, the true GMM model, the estimated GMM model using the Laplacian-based EM, and the estimated GMM model using the TM-based EM (via bounds). As seen in Fig. 1a, the results generated by the TM-based EM (via bounds) are closer to the true model, especially in some bases, such as the top right purple and the bottom center magenta one. The results for TM-based EM (exact) are not shown to reduce clutter in Fig. 1a. To better show the accuracy of each estimated GMM model, Fig. 1b depicts the maximum Log-likelihood of the distributed EM algorithms in comparison to the central EM. By using the same number of communications, the TM-based EM algorithms, even when we use the bounds instead of the exact values for λ_2 and λ_N achieve better results compared to the Laplacian algorithm in the sense that the maximum log-likelihood of the TM-based estimates are closer to the central solution. This difference is especially larger in cases where the number of communications is limited, e.g., $T_{\text{consensus}} = 8$.



(a)



(b)

Fig. 1: Plot (a) depicts the GMM estimates of Agent 1 after $T_{\text{EM}} = 10$ iterations of the EM algorithm. Plot (b) represents the maximum Log-likelihood of the estimated models of the central EM, TM-based EM (exact), TM-based EM (via bounds), and Laplacian-based EM.

In order to compare the convergence rates of the proposed algorithms against the existing ones in the literature, Fig. II illustrates the evolution of the estimates of π_l, for $l = 1$, over $T_{\text{consensus}} = 50$ steps in the first iteration of the EM algorithm. Here, the convergence error trajectories of the consensus algorithms is denoted by $e(k) = \log \sum_{i=1}^{N} (\pi_1^i(k) - \pi_1^\star)^2$, where $\pi_1^\star$ is the central solution obtained by (2a). It is shown that the TM algorithm achieves the fastest convergence rate and also, the buffered Laplacian algorithm with $d = 2$ is faster than the original Laplacian algorithm without using buffered feedback.

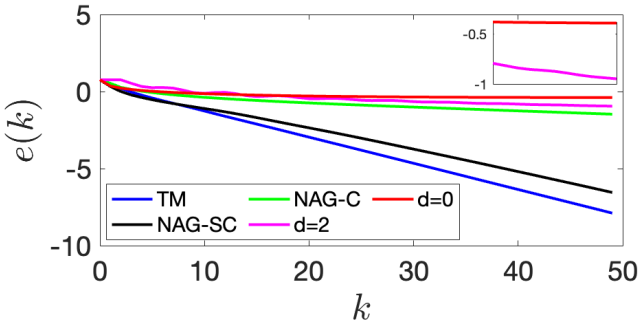


Fig. 2: This plot shows the convergence error of $e(k)$ for obtaining π_1^i with respect to the actual average computed as (2a). Five different average consensus algorithms have been implemented and compared as above. For better visual comparison, the convergence error trajectories of the Laplacian algorithm with $d = 0$ and $d = 2$ for $k = \{40, \cdots, 50\}$, is maximized in the top right corner.