Model Selection and High-Dimensional Clustering Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian Data

Student Name: Shouvik Ganguly PID: A53043486
Student Name: Pinar Sen PID:A53074382

***Abstract*—In this work, we look at an efficient algorithm for computing sparse maximum likelihood estimates for the covariance matrix of high dimensional Gaussian data. The sparsity is enforced through adding an ℓ_1-norm penalty to the maximum likelihood estimation problem. This, in addition to regularizing the problem, makes the underlying graphical model, representing the inter-dependence of the data, sparse, so that the relationships among the various features can be represented by a simple graph. As the second part of our work, we study on the clustering problem of high dimensional Gaussian data. In this part, the mentioned sparse model selection algorithm is used as part of two different modified "Expectation Maximization (EM)-like" algorithms for computing regularized maximum-likelihood estimates for the parameters of a Gaussian mixture model (GMM). Performance of the different algorithms are compared (via experiments) in terms of different metrics like relative mean-squared error, sparsity of the estimated covariance matrices and CPU times.**

Keywords*—*sparse covariance matrix, maximum likelihood, high dimensional Gaussian, block coordinate descent, EM algorithm, clustering, Gaussian mixture model.

I. Introduction

Representing a bunch of data samples with a simple graph can be important for a variety of applications, like radar or communication theory. Therefore, in the first part of our work, we focus on estimating the parameters of Gaussian distribution in a way that the resulting graphical model is sparse, so that the data is represented adequately with a simple graph [1]. For this purpose, we first look at the maximum-likelihood estimation of the inverse covariance matrix of the multivariate Gaussian distribution with known mean under an added sparsity constraint (manifested in terms of the ℓ_1-norm of the vectorized matrix) [2]. Absence of the sparsity requirement might lead to overfitting of the data, as well as a highly complicated model of inter-dependence among different dimensions (which might be spurious if the number of data points is not large compared to the number of dimensions). In other words, the sparsity requirement injected in the problem ensures that we choose the simplest possible graphical model, while still being able to estimate the relationship reasonably well.

Therefore, we concentrate on the primal and dual problem formulation with the added sparsity constraint which has high computational complexity, particularly for larger dimensions [2], [3]. Considering the higher dimensions, we study on the block coordinate descent algorithm in [2], [4] which is an efficient suboptimal solution for the dual problem since it expresses the optimization problem by quadratic programming. We observe the recovery performance and simulation time of both primal-dual problem and block coordinate descent algorithm. Moreover, we modify this sparse model selection problem for a multivariate Gaussian whose actual mean vector is not known. Using our proposed method in the block coordinate descent algorithm, we achieve very similar recovery performance to the known mean scenario.

In the second part of our work, we focus on the high dimensional clustering problem. Lately, there is variety of applications where the clustering has an important role, such as image processing, graphics, multimedia and etc. A popular approach to clustering is using Gaussian mixture models (GMMs), where each cluster corresponds to a Gaussian distribution. For low dimension, EM algorithm is proposed to cluster the data samples in the literature [5]. However, its performance degrades dramatically. Following the idea in [6], we modify two different version of the EM algorithm where the sparse model estimation described in the first part of our study is used for estimating the parameters of a GMM. Also, we compare the clustering performance of all these algorithms under different scenarios through simulations and reach that our modified algorithms give better results than regular EM algorithm.

The report is organized as follows: Section II introduces the model selection problem for multivariate Gaussian data, including the primal and dual optimization problem formulations and the block coordinate descent algorithm to solve the dual problem efficiently. In Section III, the problem of high dimensional clustering for GMM is explained together with the algorithms in the literature as well as our proposed method. The simulation results for each part are presented within each section. Finally, Section IV concludes the report.

The notations used in the report are as follows. Lower case letters (e.g., x) denote scalars, lower case bold letters (e.g., $\mathbf{x}$) denote vectors (visualized as column vectors), and upper case letters (e.g., X) denote matrices. For a given vector random variable $\mathbf{x}$, $\boldsymbol{\mu}$ denotes its mean vector and $\boldsymbol{\Sigma}$ denotes its covariance matrix. The indicator $(\cdot)^T$ denotes transpose operation.

II. Model Selection for Multivariate Gaussian Data

This section presents the first part of our study which aims to estimate the parameters of Gaussian distribution in a way that the resulting graphical model is sparse so that

the data is represented adequately with a simple graph [1]. For this purpose, we would like to look at the maximum-likelihood estimation of the inverse covariance matrix of multivariate Gaussian distribution with known mean $\boldsymbol{\mu}$, under an added sparsity constraint (manifested in terms of the ℓ_1-norm of the vectorized matrix) [2]. Absence of the sparsity requirement might lead to overfitting of the data, as well as a highly complicated model of inter-dependence among different dimensions (which might be spurious if the number of data points is not large compared to the number of dimensions). Therefore, the sparsity requirement injected in the problem ensures that we choose the simplest possible graphical model, while still being able to estimate the relationship among data samples reasonably well.

A. Primal and Dual Problem Formulation

Our problem is to estimate the inverse of the covariance matrix of a multivariate Gaussian distribution in a sparse form which can represent the data samples adequately. If we denote n data samples (each in p dimensions) independently drawn from a p-variate Gaussian distribution by $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \sim N(\boldsymbol{\mu}, \Sigma)$, then the data log-likelihood will be

$$\mathfrak{L}(\mu, \Sigma, \mathbf{x}_1, \ldots \mathbf{x}_n) = \sum_{i=1}^{n} \log \Big((2\pi)^{-p/2} \det(\Sigma)^{-1/2} \exp\left(-\frac{1}{2} (\mathbf{x}_i - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_i - \boldsymbol{\mu}) \right) \Big). \tag{1}$$

Let $\mathbf{S}$ denote the second moment matrix about the mean (similar to an empirical covariance matrix), which is expressed as

$$\mathbf{S} = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T. \tag{2}$$

Using equations (1)-(2), the modified maximum likelihood problem under the sparsity constraint for the inverse covariance matrix, $\boldsymbol{\Sigma}^{-1}$, can be stated as

$$\begin{aligned} \widehat{\boldsymbol{\Sigma}}^{-1} = \arg\max_{\mathbf{X}} \quad & \log\det(\mathbf{X}) - \mathrm{tr}(\mathbf{S}\mathbf{X}) - \lambda ||\mathbf{X}||_1 \\ subject\ to \quad & \mathbf{X} \succ \mathbf{0}, \end{aligned} \tag{3}$$

where $||\mathbf{X}||_1$ denotes the sum of the absolute values of the elements of $\mathbf{X}$. We call the optimization problem given in (3) as primal problem. We know that it is a convex optimization problem since it tries to maximize a concave objective function over a convex set.

Primal problem in (3) can be rewritten as

$$\max_{\mathbf{X} \succ \mathbf{0}} \min_{||\mathbf{Y}||_\infty \leq \lambda} \log\det(\mathbf{X}) - \mathrm{tr}(\mathbf{X}(\mathbf{S} + \mathbf{Y})), \tag{4}$$

using the equivalent expression for $\lambda ||\mathbf{X}||_1$

$$\lambda ||\mathbf{X}||_1 = \max_{||\mathbf{Y}||_\infty \leq \lambda} \mathrm{tr}(\mathbf{X}\mathbf{Y}) \tag{5}$$

where $||\mathbf{Y}||_\infty$ denotes the element of matrix $\mathbf{Y}$ with the maximum absolute value.

The equivalent of the primal optimization problem in (4) can be easily dualized to give rise to the dual problem which estimates the covariance matrix (instead of the inverse). So, the dual of (4) is written as

$$\begin{aligned} \min_{||\mathbf{Y}||_\infty \leq \lambda} \quad & -\log\det(\mathbf{S} + \mathbf{Y}) - p, \\ subject\ to \quad & \mathbf{S} + \mathbf{Y} \succ \mathbf{0}, \end{aligned} \tag{6}$$

where the primal and dual variables are related as $\mathbf{X} = (\mathbf{S} + \mathbf{Y})^{-1}$. To simplify further, let $\mathbf{W} \triangleq \mathbf{S} + \mathbf{Y}$. Then, the equivalent of the dual problem can be written as

$$\begin{aligned} \hat{\Sigma} = \arg\max_{\mathbf{W}} \quad & \log\det(\mathbf{W}) \\ subject\ to \quad & ||\mathbf{W} - \mathbf{S}||_\infty \leq \lambda, \\ & \mathbf{W} \succ \mathbf{0}, \end{aligned} \tag{7}$$

where the primal and dual variables are related as $\mathbf{X} = \mathbf{W}^{-1}$. Since the primal problem is a convex optimization problem and Slater's condition holds, there is no duality gap and the dual problem gives the same results as the primal problem [4].

To visualize the recovery problem described so far, a representative scenario is depicted in Fig. 1 where a positive definite $p \times p$ sparse matrix with $p = 30$ is generated as the covariance matrix of a zero mean Gaussian multivariate distribution and $n = 60$ samples are drawn from this distribution. Fig. 1 shows the estimation of the inverse of the covariance matrix ($\hat{\Sigma}^{-1}$) using both primal and dual optimization problems as well as the emprical estimate $\mathbf{S}^{-1}$.

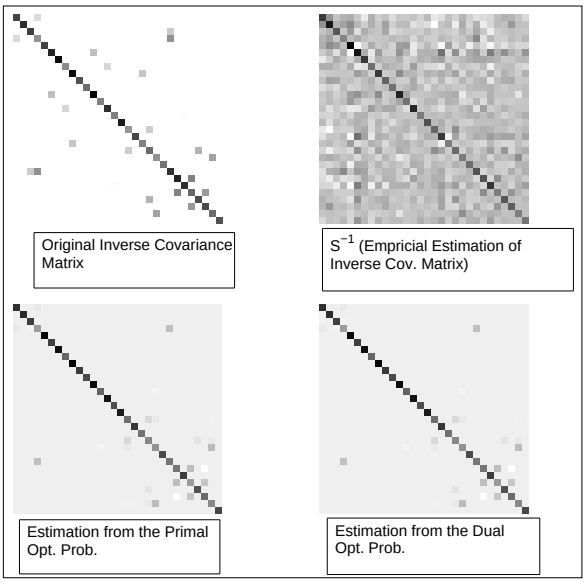

Fig. 1. Recovering the Sparsity Pattern

As shown by Figure 1, primal and dual optimization solvers (using CVX toolbox in Matlab) give the same result for $widehat\boldsymbol{\Sigma}^{-1}$ and this estimation is much more close to the original sparse inverse covariance matrix of $\boldsymbol{\Sigma}^{-1}$ than $\mathbf{S}^{-1}$.

Although this is a convex optimization problem, it is hard to solve it using well-known programs. On the other hand, it is shown that this problem can be solved by existing interior point methods with $O(p^6)$ complexity [7]. However, it is still high order complexity for larger values of p. Therefore, in the preceding section, we will introduce the block coordinate

Initialize $\mathbf{W}^0 = \widehat{\mathbf{W}}^0 = \mathbf{S} + \lambda\mathbf{I}$
while $\mathrm{tr}((\widehat{\mathbf{W}}^0)^{-1}\mathbf{S}) - p + \lambda||(\widehat{\mathbf{W}}^0)^{-1}||_1 \leq \epsilon$ **do**
  for $j = 1, 2, \ldots, p$ **do**
    For iterate W^{j-1}, solve the quadratic program (QP):

$$\hat{\mathbf{v}} = \arg\min_{\mathbf{v}} \mathbf{v}^T \left(\mathbf{W}^{j-1}_{\setminus j \setminus j}\right)^{-1} \mathbf{v}$$
$$\text{subject to } ||\mathbf{v} - \mathbf{S}_j||_\infty \leq \lambda$$

    Update: $\mathbf{W}^j$ is $\mathbf{W}^{j-1}$ with its column/row W^{j-1}_j replaced by $\hat{\mathbf{v}}$
  end for
  Update: $\widehat{\mathbf{W}}^0 = \mathbf{W}^p$
end while

Fig. 2. Block Coordinate Descent Algorithm

descent algorithm studied in [2] to solve the dual problem efficiently. Since strong duality holds in this case, this solves the original problem as well.

B. Block Coordinate Descent Algorithm

In this section, we present the block coordinate descent algorithm to solve the dual problem in (7) efficiently. The algorithm steps are given in Fig. 2, where, for a symmetric matrix $\mathbf{W}$, $\mathbf{W}_{\setminus k \setminus j}$ denotes the matrix produced by removing row k and column j, $\mathbf{W}_j$ denotes column j with the diagonal element W_{jj} removed. The value of ϵ is the gap between the objective function of the dual problem in (6) to that of the primal problem in (3) using the block coordinate descent algorithm. Hence, ϵ can be interpreted as a target duality gap of block coordinate descent algorithm that the user can specify.

The block coordinate descent algorithm tries to solve the dual optimization problem by optimizing over one row and one column at each iteration using the quadratic program in Fig. 2. Therefore, the complexity caused by the quadratic programming at each inner iteration is $O(p^3)$ which makes $O(p^4)$ over p inner iteration. If there are K sweeps of those inner iterations until convergence, the total complexity can be written as $O(Kp^4)$ as given in [2] which is much less than that of primal and dual optimization. We compare the running times of these algorithms in the next section.

Although the convergence of this algorithm is shown in [2], we would like to shortly go over the proof so as to have a better insight on why the block coordinate descent algorithm gives ϵ-suboptimal solution to (7). In the optimization problem in 7, log-determinant of a symmetric matrix $\mathbf{W}$ is to be maximized. $\mathbf{W}$ can be written as in the following form

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}_{\setminus p \setminus p} & \mathbf{w}_p \\ \mathbf{w}_p^T W_{pp} & W_{pp} \end{bmatrix}, \tag{8}$$

where $\mathbf{w}_p$ is the last column except the last entry W_{pp}. Then, determinant of $\mathbf{W}$ can be expressed using the Schur complements as follows

$$\det \mathbf{W} = \det(\mathbf{W}_{\setminus p \setminus p}) \cdot \left(W_{pp} - \mathbf{w}_p^T (\mathbf{W}_{\setminus p \setminus p})^{-1} \mathbf{w}_p\right). \tag{9}$$

Since log is an increasing function, maximizing log-determinant is equivalent to maximizing $\det \mathbf{W}$. Suppose we are trying to maximize $\det \mathbf{W}$ by optimizing its last row and last column as in the last iteration of block coordinate descent algorithm. Then, all but the last row and column of $\mathbf{W}$ is fixed in equation (9). Therefore, maximizing its determinant is equivalent to the quadratic program in the algorithm in Fig. 2.

The block coordinate descent algorithm is converged if this quadratic program has unique solution at each inner iteration [2], [3]. It is enough to show that $\mathbf{W}^j \succ 0$ at each inner iteration which can be proved by induction [2]. At initialization, $\mathbf{W}^0 = \mathbf{S} + \lambda\mathbf{I} \succ \mathbf{0}$ since $\mathbf{S} \succcurlyeq \mathbf{0}$ and $\lambda > 0$. Assume $\mathbf{W}^{j-1} \succ \mathbf{0}$, then the following Schur complement is positive [2]:

$$W_{jj} - (\mathbf{W}_j^{j-1})^T \left(\mathbf{W}^{j-1}_{\setminus j \setminus j}\right)^{-1} (\mathbf{W}_j^{j-1}) > 0. \tag{10}$$

The minimization in the quadratic program in the algorithm results in

$$\begin{aligned} W_{jj} - (W_j^j)^T \left(W^j_{\setminus j \setminus j}\right)^{-1} (W_j^j) &> W_{jj} - (W_j^{j-1})^T \left(W^{j-1}_{\setminus j \setminus j}\right)^{-1} (W_j^{j-1}) \\ &> 0, \end{aligned} \tag{11}$$

which means $W^j \succ 0$.

Selection of λ: Using a similar way to [2], we choose parameter λ as

$$\lambda = 0.1 \cdot (\max_{i>j} S_{ii} S_{jj}) \frac{t_{n-2}(\alpha/2p^2)}{\sqrt{n - 2 + t^2_{n-2}(\alpha/2p^2)}} \tag{12}$$

where $t_{n-2}(\alpha)$ denotes the $(100-\alpha)\%$ point of the Student's t-distribution for $n-2$ degrees of freedom, α is a chosen level from $[0\ 1]$, and S is the empirical estimation of the inverse covariance matrix given in (2).

Thresholding: A thresholding is needed for the estimated inverse covariance matrices at output of both coordinate block descent algorithm and the solvers of primal and dual optimization problems while preserving the positive definiteness of the estimated inverse covariance matrices. We have chosen 10^{-4} as the threshold level by trial and error, and assign 0 for every element of the estimated inverse covariance matrix which is less than 10^{-4} in our simulations.

Decreasig Mean Square Error of the Estimate: We have observed that sparsity is accurately obtained by the defined optimization problems in (3),(7). However, mean square error between the original inverse covariance matrix and the estimated one needs to be improved. Therefore, we propose to scale the estimate of the covariance matrix, $\hat{\Sigma}$ as follows

$$\widehat{\mathbf{\Sigma}} \leftarrow \widehat{\mathbf{\Sigma}} \cdot \frac{\sum_{i=1}^{p} \gamma_i(\mathbf{S})}{\sum_{i=1}^{p} \gamma_i(\widehat{\mathbf{\Sigma}})}, \tag{13}$$

where $\gamma_i(\mathbf{X})$ denotes the i^{th} eigenvalue of matrix $\mathbf{X}$.

C. Unknown Mean Scenario

So far, we have mentioned the problem of estimating the inverse covariance matrix of a multivariate Gaussian distribution with a known mean vector in a sparse form. In this section, we will present how to generalize the existing solutions for unknown mean scenario.

The problem description followed by primal and dual optimization problems and block coordinate descent algorithm can be used with the proposed exchange of μ with $\hat{\mu}$ where

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i. \tag{14}$$

So, we propose to solve the convex optimization problems in (3),(7) by updating S in (2) using $\hat{\boldsymbol{\mu}}$ as follows

$$S = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^T. \tag{15}$$

The performance difference between known and unknown mean scenarios is given in the preceding section.

D. Simulation Results

In this section, we present the simulation results related to sparse model selection for multivariate Gaussian data using the methods described in Section II so far. In simulations, we generate actual sparse covariance matrices, Σ, whose entries (both location and value) are randomly chosen so as to satisfy the predefined density $\delta = 0.1$, i.e., the ratio of the number of nonzero elements to p^2 is 0.1. $p = 30$ is chosen in our simulations. Then, n number of data sample vectors are independently drawn from zero-mean Gaussian multivariate distribution with the generated actual covariance matrix Σ.

The solvers of primal and dual problem and the block coordinate descent algorithm in Fig. 2 are implemented using CVX toolbox in Matlab under the scenarios where the mean of the distribution is known and unknown. We observe 3 different measure of performance. Relative Frobenious norm square of estimation error, called relative MSE in plots, is computed as

$$\text{Relative MSE} = \frac{||\Sigma^{-1} - \hat{\Sigma}^{-1}||_F^{\,2}}{||\Sigma^{-1}||_F^{\,2}}. \tag{16}$$

Another performance measure is the recovery of the zero entries which is computed by the similarity between the support of 0s for estimated covariance matrix and that for actual covariance matrix. We call this measure as probability of detection of zeros, $P_D(0)$, which is expressed by

$$P_D(0) = \frac{|\hat{A} \cap A|}{|A|}, \tag{17}$$

where A and $\hat{A}$ are the support of 0s for the actual and the estimated covariance matrix respectively.

The third notion is the sparsity of the estimated solution, $\hat{\delta}$. We also compare the sparsity, meaning the ratio of the number of non-zero entries to p^2, of the estimated covariance matrices for each algorithm as well as the actual sparsity $\delta = 0.1$.

Our simulations agree with the 0 duality gap between the primal and dual optimization problems (we have observe that there is no difference between two estimates obtained from solving primal and dual problem). Therefore, we give the performance results of dual problem only among those two, considering the crowd in the plots.

From Figure 3, we see that for all algorithms, the relative MSE decreases as the number of samples increases,

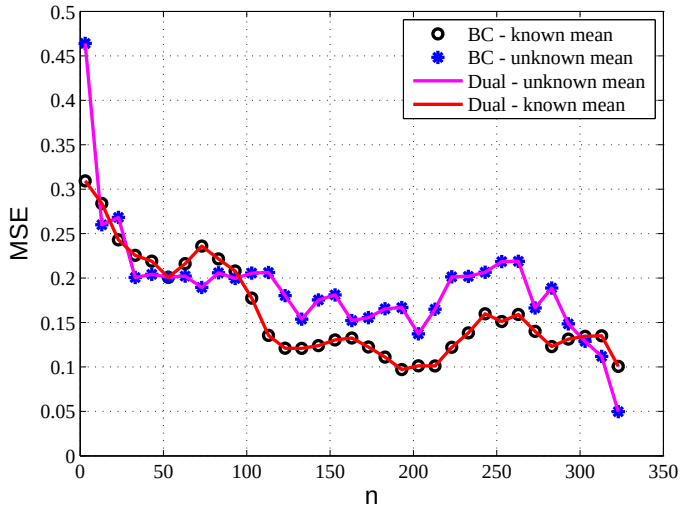

Fig. 3. Relative MSE for different algorithms with known and unknown mean

as expected. Moreover, when the mean is unknown, there is a penalty in the MSE, due to estimating the expectations using the sample mean. Also, the dual algorithm and the block coordinate descent algorithm give the same MSE for each case (since the estimates obtained by the two methods are essentially the same). Thus, the block coordinate descent algorithm gives the same MSE performance as the naive algorithm.

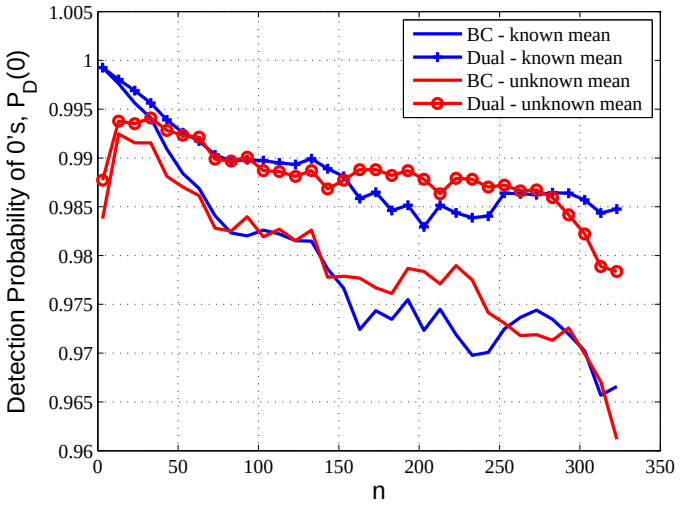

Fig. 4. Detection Probability of 0's for different algorithms with known and unknown mean

Figure 4 shows that the dual algorithm gives in general slightly better detection rates compared to the block coordinate descent algorithm, for both the known mean and unknown mean case. Moreover, knowledge of the distribution mean does not appear to have a substantial effect on the detection rates. The same trend is observed in Figure 5, which shows the sparsities of our estimates for the inverse covariance matrix. The similarity in the trends in Figures 4 and 5 makes intuitive sense, since both these metrics are related to the number and arrangements of nonzero entries in our estimates of the inverse

covariance matrix. An additional point to be noted in Figure 5 is that the number of nonzero entries in our estimates of the inverse covariance matrix are typically lower than the actual number. One possible way to increase the number of nonzero entries would be to lower the threshold used for making the estimate sparse.

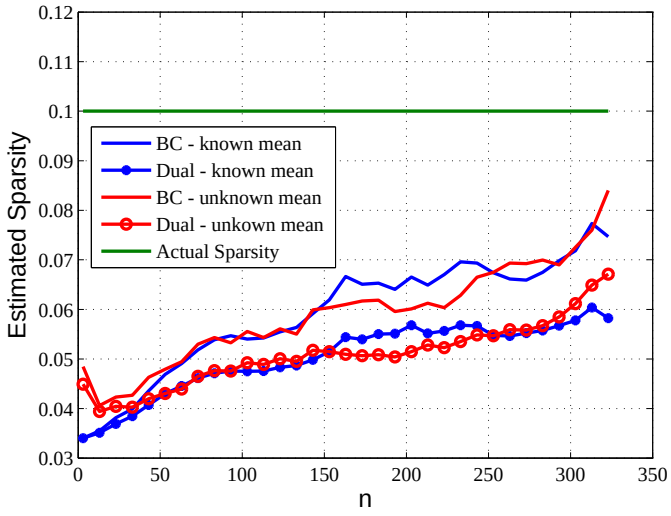

Fig. 5. Estimated Sparsity for different algorithms with known and unknown mean

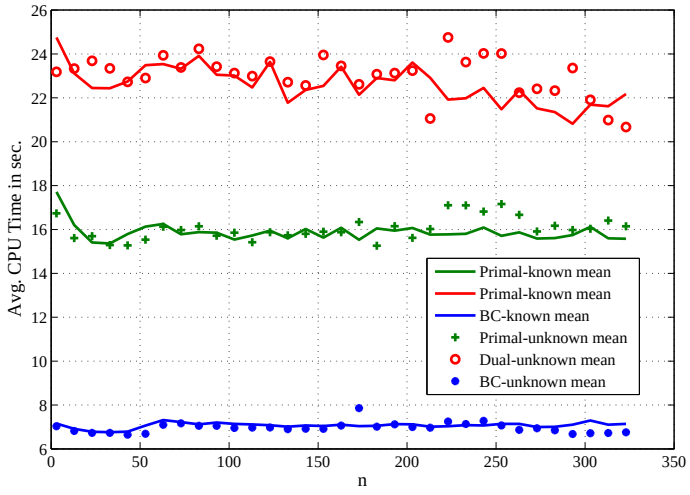

Fig. 6. CPU Times

Figure 6 shows that the block coordinate descent algorithm takes substantially less time than both the naive primal and dual algorithms. Moreover, the dual takes much more time to run than even the primal. We also notice that the runtime is not substantially affected by whether or not we know the mean of the distribution, since the unknown mean problem only involves one additional step of estimating the distribution mean by the sample mean.

The simulation results for this section show that the block coordinate descent algorithm gives almost similar performance to the primal and dual algorithms, while taking much smaller time to run. Thus, it is a more efficient algorithm for sparse estimation of covariance matrices. We will apply this algorithm to a more general class of distributions in the next section.

III. High Dimensional Clustering with Sparse Gaussian Mixture Models

Clustering refers to partitioning multi-dimensional data into classes according to certain parameters, and the determination of conditions which would enable one to decide which class new data should go into. Clustering is usually based on some kind of distance notion, where points that are "close" to one another are put into the same class. This classification based on distance can also be performed after applying some transformation on the raw data (such as principal component analysis (PCA) [8]). The most common distance notion used is Euclidean distance, though other kinds of Mahalanobis distances can also be used.

In recent years, clustering has found lots of usage in image classification, image recognition, graphics, multimedia and so on. This has spawned a lot of research in this area. A popular approach to clustering is using Gaussian mixture models (GMMs), where each cluster corresponds to a Gaussian distribution.

A. Background for Gaussian Mixture Models

In a Gaussian mixture model (GMM), each example is drawn from its cluster-specific distribution, which is a Gaussian with certain mean and covariance matrix. Mathematically, the probability density function (pdf) for a Gaussian mixture model with k clusters (for each data point $\mathbf{x}$) can be expressed as

$$f(\mathbf{x}) := \sum_{i=1}^{k} \pi_i \phi(\mathbf{x}|\boldsymbol{\mu}_i, \Sigma_i), \tag{18}$$

where the π_i's are the weights (which lie on the k-dimensional simplex), and $\phi(\mathbf{x}|\boldsymbol{\mu}_i, \Sigma_i)$ is a Gaussian density with mean vector $\boldsymbol{\mu}_i$ and covariance matrix Σ_i, given by

$$\phi(\mathbf{x}|\boldsymbol{\mu}_i, \Sigma_i) = \frac{1}{(2\pi)^p |\Sigma_i|^{1/2}} \exp\Big[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\mathbf{x}-\boldsymbol{\mu}_i)\Big]. \tag{19}$$

A GMM can also be interpreted in the following way. First, one of the clusters 1, 2, ... k is chosen (with corresponding probabilities π_i), and given cluster i, $\mathbf{x}$ is chosen with pdf $\phi(\mathbf{x}|\boldsymbol{\mu}_i, \Sigma_i)$.

B. GMM EM Algorithm

Given a set of samples, one can use the expectation maximization (EM) algorithm to find maximum likelihood estimates for the parameters of the GMM [5]. Learning the parameters leads to a natural clustering of the data, since we can assign each example to the Gaussian distribution it is most likely to originate from. This method of parameter estimation leads to maximum likelihood estimates for the parameters, and therefore, clustering benefits from the statistical properties of MLEs. It is well known that GMMs work quite well in practice, provided that the data is low-dimensional [5], [6].

For the sake of completeness, we present the algorithm steps of EM algorithm [5] in Fig. 7 where Δ is the predefined

Initialize $\hat{\pi}, \widehat{\boldsymbol{\mu}_i}, \widehat{\Sigma_i}$ for $i = 1, \ldots, k$ randomly, $q = 0$
while $\sum_{i=1}^{k} ||\widehat{\boldsymbol{\mu}_i}^{q} - \widehat{\boldsymbol{\mu}_i}^{q-1}||_2^2 > \Delta$ **do**
$q = q + 1$,
E-step:
for $j = 1, 2, \ldots, n$ **do**
for $i = 1, 2, \ldots, k$ **do**

$$t_{ji}^{q} = \frac{\hat{\pi}_i^{q-1} \phi(\mathbf{x}_j, \widehat{\boldsymbol{\mu}_i}^{q-1}, \widehat{\Sigma_i}^{q-1})}{\sum_{m=1}^{K} \hat{\pi}_m^{q-1} \phi(\mathbf{x}_j, \widehat{\boldsymbol{\mu}_m}^{q-1}, \widehat{\Sigma_m}^{q-1})} \tag{20}$$

end for
end for
M-step:

$$n_i^q = \sum_{j=1}^{n} t_{ji}^q \tag{21}$$

$$\hat{\pi}_i^q = n_i^q / n \tag{22}$$

$$\widehat{\boldsymbol{\mu}_i}^q = \frac{\sum_{j=1}^{n} t_{ji}^q \mathbf{x}_j}{n_i^q} \tag{23}$$

$$\widehat{\Sigma_i}^q = \frac{\sum_{j=1}^{n} t_{ji}^q \left(\mathbf{x}_j - \widehat{\boldsymbol{\mu}_i}^q\right)\left(\mathbf{x}_j - \widehat{\boldsymbol{\mu}_i}^q\right)^T}{n_i^q} \tag{24}$$

end while

Fig. 7. EM Algorithm for GMM (GMM EM)

threshold on the convergence of the estimated mean vectors of all cluster.

It is known that the performance of the EM algorithm degrades when the dimension of clustering $k > 1$ [5], [6].Therefore, a joint method, in which the sparse model selection technique described in Section II is included into the M-step, is introduced in the preceding section. Also, we use the EM algorithm to make a performance comparison with the following joint method.

C. Joint EM Algorithm with Sparse ML Estimation (sGMM EM)

We use the idea of using sparse model selection method in the estimation of the cluster parameters in [6] and modify according to the GMM EM algorithm given in Fig. 7 to get a joint method of clustering high dimensional data. The modified algorithm, which is called sGMM EM, is given in Fig. 8.

Maximization problem in the sGMM EM algorithm in Fig. 8 is the same as the primal problem described in (3). Therefore, we used the block coordinate descent algorithm explained in Seciton II-B to solve this convex optimization problem efficiently in every M-step of the modified clustering algorithm.

It is observed that the performance of the sGMM EM is extremely good for $k = 2$ but degrades for $k > 2$. We come up with another algorithm for larger dimensions which is explained in the following section. The performance comparison of all these algorithms under different cases will be given in Section III-E.

Initialize $\hat{\pi}, \widehat{\boldsymbol{\mu}_i}, \widehat{\Sigma_i}$ for $i = 1, \ldots, k$ randomly, $q = 0$
while $\sum_{i=1}^{k} ||\widehat{\boldsymbol{\mu}_i}^{q} - \widehat{\boldsymbol{\mu}_i}^{q-1}||_2^2 > \Delta$ **do**
$q = q + 1$,
E-step:
for $j = 1, 2, \ldots, n$ **do**
for $i = 1, 2, \ldots, k$ **do**

$$t_{ji}^{q} = \frac{\hat{\pi}_i^{q-1} \phi(\mathbf{x}_j, \widehat{\boldsymbol{\mu}_i}^{q-1}, \widehat{\Sigma_i}^{q-1})}{\sum_{m=1}^{K} \hat{\pi}_m^{q-1} \phi(\mathbf{x}_j, \widehat{\boldsymbol{\mu}_m}^{q-1}, \widehat{\Sigma_m}^{q-1})} \tag{25}$$

end for
end for
M-step:

$$n_i^q = \sum_{j=1}^{n} t_{ji}^q \tag{26}$$

$$\hat{\pi}_i^q = n_i^q / n \tag{27}$$

$$\widehat{\boldsymbol{\mu}_i}^q = \frac{\sum_{j=1}^{n} t_{ji}^q \mathbf{x}_j}{n_i^q} \tag{28}$$

$$S_i = \frac{\sum_{j=1}^{n} t_{ji}^q \left(\mathbf{x}_j - \widehat{\boldsymbol{\mu}_i}^q\right)\left(\mathbf{x}_j - \widehat{\boldsymbol{\mu}_i}^q\right)^T}{n_i^q} \tag{29}$$

$$\left(\widehat{\Sigma_i}^q\right)^{-1} = \arg\max_{X} \ \log det(X) - tr(S_i X) - \lambda ||X||_1 \quad subject\ to \quad X \succ 0 \tag{30}$$

end while

Fig. 8. EM Algorithm with Sparse Model Selection for GMM (sGMM EM)

D. Joint k-means Algorithm with Sparse ML Estimation

A different approach to clustering can be adopted, to that described in the previous section. The performance of this method is slightly worse than sGMM EM for $k = 2$, but for $k > 2$, this algorithm still gives reasonable performance, while that of sGMM EM degrades severely.

This method relies on the well-known k-means clustering [9] for dividing the data into Voronoi clusters. The central idea is to divide the data into clusters, such that all the points belonging to a particular cluster are closer to the centroid of that cluster than to that of any other cluster. This is achieved iteratively via the algorithm described in Fig. 9. In the algorithm, $\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n$ represent the data points, k represents the number of clusters, $\widehat{\boldsymbol{\mu}_i}^q$ represents the centroid of the i^{th} cluster after the q^{th} iteration, and the norm is Euclidean. Once the clustering is done, the estimates for the $\boldsymbol{\mu}_i$'s are taken as simply the centroids of the clusters, and the estimates for the inverse covariance matrices, i.e. the $\widehat{\Sigma_i}^{-1}$'s, are computed as ℓ_1-norm constrained maximum likelihood estimates using the block coordinate descent algorithm [2].

E. Simulation Results

In this section, the simulation results for the clustering methods (GMM EM, sGMM EM, k-means) described in Section III under different scenarios are presented.

Initialize $\widehat{\boldsymbol{\mu}_i}$, for $i = 1$ through k, as k randomly chosen distinct data points.

while $\max_{i=1,2,\ldots k} ||\widehat{\boldsymbol{\mu}_i}^q - \widehat{\boldsymbol{\mu}_i}^{q-1}||_2^2 > \Delta$ **do**

for $j = 1, 2, \ldots n$ **do**

$a_j = \arg\min_{i=1,2,\ldots k} ||\mathbf{x}_j - \widehat{\boldsymbol{\mu}_i}^q||$

end for

$q = q + 1$.

for $i = 1, 2, \ldots k$ **do**

$$\widehat{\boldsymbol{\mu}_i}^q = \frac{\sum_{1 \le j \le n: a_j = i} \mathbf{x}_j}{\#\{j : 1 \le j \le n, a_j = i\}} \tag{31}$$

end for

end while

for $i = 1, 2, \ldots k$ **do**

$$n_i = \#\{j : 1 \le j \le n, a_j = i\}, \tag{32}$$

$$S_i = \frac{1}{n_i} \sum_{1 \le j \le n: a_j = i} (\mathbf{x}_j - \widehat{\boldsymbol{\mu}_i}^q)(\mathbf{x}_i - \widehat{\boldsymbol{\mu}_i}^q)^T, \tag{33}$$

$$\widehat{\Sigma_i}^{-1} = \arg\max_X \log det(X) - tr(S_i X) - \lambda ||X||_1 \quad subject\ to \quad X \succ 0 \tag{34}$$

end for

Fig. 9. k-Means Clustering Algorithm

In the first set of experiments, we take $k = 2$ and generate data for 2 Gaussian distributions with means $\boldsymbol{\mu}_1 = [0\ 0 \ldots 0]$ and $\boldsymbol{\mu}_2[3/\sqrt{p}\ \ 3/\sqrt{p} \ldots 3/\sqrt{p}]$. The covariance matrices C_1 and C_2 are randomly chosen having sparsity values of 0.4 and 0.05 respectively. The sparsity values and covariance matrices are purposely chosen different in order to observe the performance of the sparse method based EM (sGMM EM) algorithm. There are total of $n = 200$ data samples, 100 of which are drawn from each distribution. Without knowing neither any information about the covariance matrices nor the number of samples belonging to each cluster, the GMM EM algorithm in Fig. 7 and sGMM EM in Fig. 8 are performed to cluster the data samples for varying number of dimension p. Fig. 10 shows the training error, which is the ratio of the incorrectly assigned data samples, for both algorithms. As seen from Fig. 10, even for $k = 2$, the performance of the GMM EM algorithm degrades. Moreover, it has a large variations meaning that it depends on the initialization [6]. On the other hand, the performance of the sGMM EM algorithm is quite good even for larger p values.

We make another experiment which is exactly the same as above except the covariance matrices C_1 and C_2. This time, we choose full covariance matrices $C_1 = C_2$ taken from the

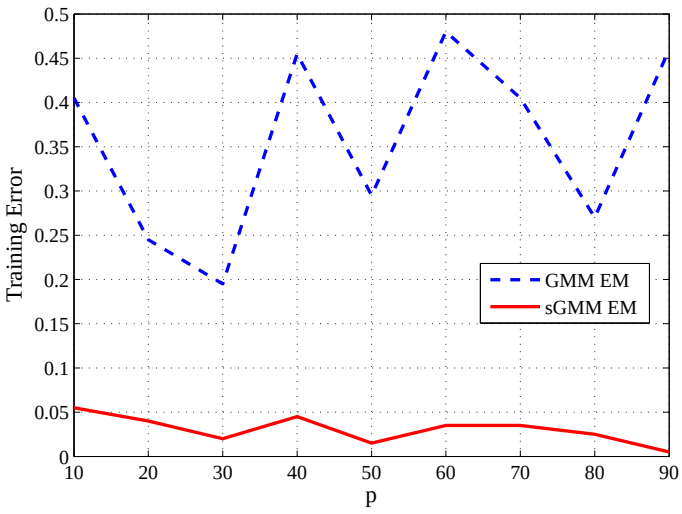

Fig. 10. Performance of GMM algorithms for $k = 2$ with different covariance matrices

experiment in [6] such that

$$C_1^{-1} = C_2^{-1} = \begin{bmatrix} 2 & 1 & \ldots & 1 \\ 1 & 2 & \ddots & \vdots \\ \vdots & \ddots & 2 & 1 \\ 1 & \ldots & 1 & 2 \end{bmatrix}_{p \times p}. \tag{35}$$

The training error performance of all the algorithms described in Section III for this experiment with full covariance matrices is given in Fig 11. This time we also observe the performance of the k-means algorithm. From Figure 11, we see that for the case of $k = 2$, with full (and equal) covariance matrices, the sGMM EM algorithm performs the best, with very small training error, while the training error for the k-means clustering algorithm (described Figure 9) is around 5% for almost all values of p. However, as in the previous case, the performance of the GMM EM algorithm is not so good, and even reaches 50% error for some values of p.

Fig. 12 shows the result for $k = 4$ with $n = 1000$ sample data. For this case, we chose the mean vectors for each cluster randomly, and generated different sparse covariance matrices with 10% nonzero elements. From the plot, we see that the k-means clustering algorithm works even for $k = 4$, though the training error performance is much degraded in this case. However, this should be put into perspective against the fact that the sGMM EM algorithm does not even seem to converge for this case, which is why we do not have a plot for sGMM EM.

Thus, to conclude this section, sGMM EM is the algorithm of choice for a GMM EM with two clusters, however its performance severely degrades as the number of clusters increases, and the k-means clustering algorithm steps in to save the day. As a future study, the effects of the initialization can be investigated to further improve the performance of k-means algorithm.

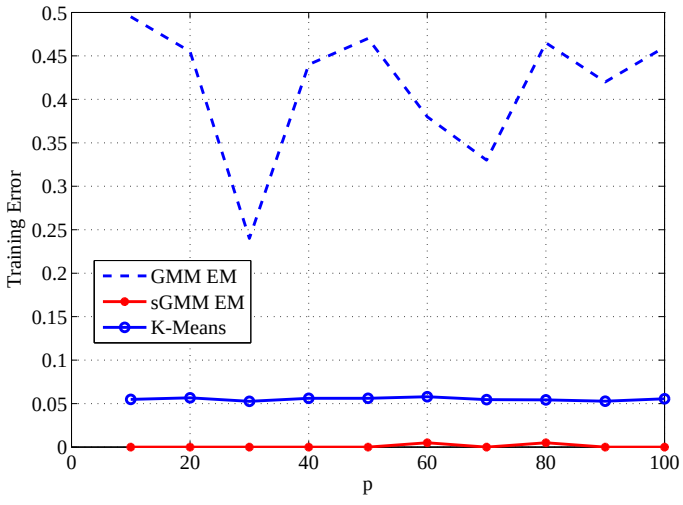

Fig. 11. Performance of GMM algorithms for $k = 2$ with full covariance matrices described in (35)

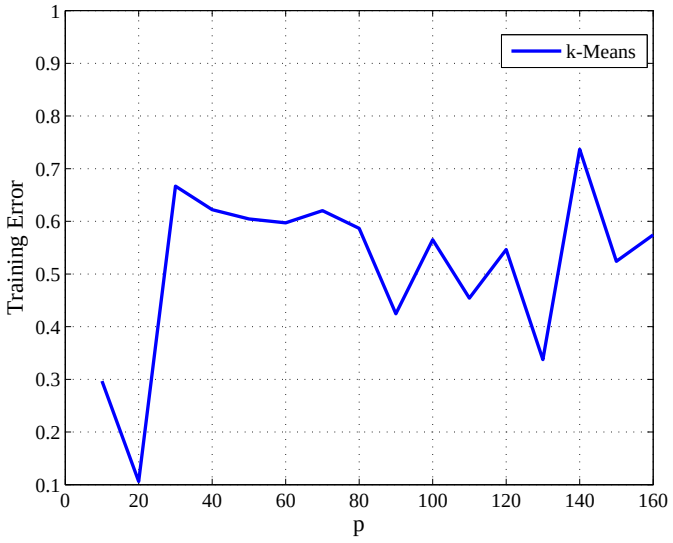

Fig. 12. Performance of k-means algorithm for $k = 4$

IV. Conclusion

We studied an algorithm to efficiently compute maximum likelihood estimate for covariance matrix of high-dimensional Gaussian data, with added sparsity constraints. The sparsity constraints (incorporated in the usual way, by constraining the ℓ_1 norm of the estimate) ensure that the estimated graphical model for the data is sparse, and that the solution is regularized by preventing overfitting, which might lead to problems such as badly conditioned estimates for the covariance matrix. Comparing the performance of this algorithm (called the block coordinate descent algorithm) with the naive approach, we found that it gives similar performance but needs much smaller time that also scales better with the problem size.

We also applied the block coordinate descent algorithm for estimating the parameters of Gaussian mixture models (GMM) with high dimensions. We studied three algorithms for this problem. The first one was the naive EM algorithm applied to GMMs, where the performance was not satisfactory, and also not consistent. The second one, abbreviated as sGMM EM, was a modified EM approach together with the block coordinate descent algorithm, which gives very good results for $k = 2$ clusters. However, the performance of this algorithm degrades severely for $k > 2$. The third algorithm we studied uses the k-means approach to partition the data into clusters, and then estimated the mean and covariance for each cluster using the block coordinate descent algorithm, similar to the approach adopted for a pure multivariate Gaussian with unknown mean. This algorithm does not perform as well as sGMM EM for $k = 2$, but it still works when the number of clusters is bigger than 2. Thus, this algorithm holds promise as the algorithm of choice for higher number of clusters.

As future work, the effect of initialization of the parameters on the performance of the algorithms needs to be studied. Moreover, a slight generalization of the GMM problem can be thought of, whereby the number of clusters in the data is unknown. These algorithms need to be generalized to take care of this case as well.

References

[1] J. Dahl, L. Vandenberghe, and V. Roychowdhury, "Covariance selection for nonchordal graphs via chordal embedding," *Optimization Methods Software*, vol. 23, no. 4, pp. 501–520, Aug. 2008. [Online]. Available: http://dx.doi.org/10.1080/10556780802102693

[2] O. Banerjee, L. El Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data," *J. Mach. Learn. Res.*, vol. 9, pp. 485–516, Jun. 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1390681.1390696

[3] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 1999.

[4] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[5] C. Bouveyron and C. Brunet-Saumard, "Model-based clustering of high-dimensional data: A review," *Comput. Stat. Data Anal.*, vol. 71, pp. 52–78, Mar. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.csda.2012.12.008

[6] A. Krishnamurthy, "Clustering with sparse gaussian mixture models," 2014.

[7] L. Vandenberghe, S. Boyd, and S.-P. Wu, "Determinant maximization with linear matrix inequality constraints," *SIAM J. Matrix Anal. Appl.*, vol. 19, no. 2, pp. 499–533, Apr. 1998. [Online]. Available: http://dx.doi.org/10.1137/S0895479896303430

[8] I. Jolliffe, *Principal Component Analysis*. Springer Verlag, 1986.

[9] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, L. M. L. Cam and J. Neyman, Eds., vol. 1. University of California Press, 1967, pp. 281–297.