Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Hierarchical Bayes based Adaptive Sparsity in Gaussian Mixture Model ☆

Binghui Wang [a], Chuang Lin [a,b,*], Xin Fan [a], Ning Jiang [b], Dario Farina [b]

[a] School of Software, Dalian University of Technology, Dalian, China
[b] Department of Neurorehabilitation Engineering, University Medical Center Goettingen, Georg-August University, Goettingen, Germany

A R T I C L E   I N F O

Article history:
Received 5 September 2013
Available online 22 July 2014

Keywords:
High-dimensional parameter estimation
Hierarchical Bayes
Adaptive sparsity
GMM

A B S T R A C T

Gaussian Mixture Model (GMM) has been widely used in statistics for its great flexibility. However, parameter estimation for GMM with high dimensionality is a challenge because of the large number of parameters and the lack of observation data. In this paper, we propose an effective method named hierarchical Bayes based *Adaptive Sparsity in Gaussian Mixture Model* (*ASGMM*) to estimate the parameters in a GMM by incorporating a two-layer hierarchical Bayes based adaptive sparsity prior. The prior we impose on the precision matrices can encourage sparsity and hence reduce the dimensionality of the parameters to be estimated. In contrast to the l_1-norm penalty or Laplace prior, our approach does not involve any hyperparameters that must be tuned, and the sparsity adapts to the observation data. The proposed method is achieved by three steps: first, we formulate an adaptive hierarchical Bayes model of the precision matrices in the GMM with a Jeffrey's noninformative hyperprior, which expresses scale-invariance and, more importantly, is *hyperparameter-free* and *unbiased*. Second, we perform a Cholesky decomposition on the precision matrices to impose the positive definite property. Finally, we exploit the expectation maximization (EM) algorithm to obtain the final estimated parameters in the GMM. Experimental results on synthetic and real-world datasets demonstrate that ASGMM cannot only adapt the sparsity of high-dimensional data with small estimated error, but also achieve better clustering performance comparing with several classical methods.

1. Introduction

Gaussian Mixture Model (GMM) is among the most popular statistical modeling tools used for density estimation, clustering, and discriminant analysis [12]. As a generative approach to clustering, GMM models each cluster with a Gaussian graphical model (GGM), and each example is drawn from its cluster-specific distribution. Unfortunately, despite its great flexibility, GMM experiences severe performance degradation when modeling high-dimensional data. The failure occurs because the number of estimated parameters grows quickly with the dimensionality of the covariance matrices. This is fundamentally an over-fitting issue, and traditional solutions are unstable and unsuitable.

Previous studies attempt to solve this issue by virtue of structure learning, that is, they attempt to find the conditional independencies between the parameters to be estimated in the observed data [13]. Taking GGM, the special case of GMM, as an example, the inference of the graph structure is equivalent to the estimation of the zero entries of its precision matrix. Specifically, if the *ij*th component is zero, then variables *i* and *j* are conditionally independent in the graphical model. The aim of such models is to reduce dependencies between variables, however, in essence, it cannot solve the over-fitting problem of high-dimensional parameters.

Recently, researchers have paid much attention to overcoming the difficulty of estimating high-dimensional parameters in GGM. Generally, these methods can be categorized into two types. One type is to directly work with the sample covariance matrix of the GGM [4,8,15,10,16,18]. The key advantage is that this is readily computable regardless of dimensionality. However, the method cannot capture the sparse graphical structure of the parameters in GGM. The second type of method is aimed at discovering the sparse structure that naturally occurs. Instead of handling the covariance matrix in GGM, these techniques deal with the precision matrix by imposing *sparsity* on its entries. Among them, several techniques have been utilized. An approximation method [19] estimates a sparse graphical model by fitting a LASSO model to each variable, using the others as predictors. Exact maximization methods of the MLE (Maximus Likelihood Evaluation) with an l_1-norm penalty for encouraging sparsity have been proposed by

☆ This paper has been recommended for acceptance by G. Borgefors.

* Corresponding author. Address: School of Software, Dalian University of Technology and Department of Neurorehabilitation Engineering, University Medical Center Goettingen, Georg-August University, Goettingen, Germany. Tel.: +49 15784361050.

E-mail address: lin.chuang@bccn.uni-goettingen.de (C. Lin).

http://dx.doi.org/10.1016/j.patrec.2014.07.008
0167-8655/© 2014 Elsevier B.V. All rights reserved.

[2,13,28,9,22]. Benefit from the success in estimating high-dimensional precision matrix in GGM, Ruan et al. [23] proposes a Gaussian mixture LASSO (GMLASSO) method by employing l_1-norm penalty on the entries of each precision matrix in GMM. GMLASSO is actually an extension of GLASSO [13] which utilize mixture Gaussians instead of a single Gaussian.

However, all aforementioned l_1 regularization methods have two major problems. On one hand, they are hyperparametric-based, that is, they need a regularization hyperparameter which controls the amount of sparsity to be adjusted. One should note that the selection of hyperparameter (model) is one of the most crucial problems in machine learning. In some cases, the learning performance may change greatly with different choices of hyperparameters [14]. On the other hand, it is undesirable that the l_1-norm based LASSO is a biased estimator [20]. Therefore, taking these two issues into account, we are motivated to find a hyperparameter-free way to capture the sparse structure of precision matrix adaptively, and to seek an unbiased estimator.

We propose a two-layer hierarchical Bayes model for adaptive sparse estimation of precision matrices in GMM. For simplicity, we name it as hierarchical Bayes based Adaptive Sparsity in Gaussian Mixture Model, abbreviated ASGMM. It is worthwhile to highlight several attractive characteristics of ASGMM. First, in contrast to l_1-norm based biased methods and hyperparametric regularization based methods, the main advantage of ASGMM is that it does not involve estimation of hyperparameters, that is, the sparsity obtained reflects the observed data. Note that this adaptive sparsity is non-trivial. It is achieved by modeling the precision matrices in GMM with a Jeffrey's noninformative hyperprior [11], a modified two-layer Bayes–Laplace prior. More importantly, it is hyperparameter-free and unbiased [17,25]. Second, the solution of ASGMM is a matrix-based optimization problem, thus, we can perform a Cholesky decomposition to gain easily solved element-based equations. Further, ASGMM can be naturally incorporated into the famous EM [7] framework in order to obtain the estimated parameters.

2. Related work

With recent advances in science and technology, we are often faced with the problem of high dimensional parameter estimation where the dimensionality of data is (much) larger than the sample size. A common way to solve the problem is to exploit the sparse structure, and a number of sparse models have been introduced in recent years.

Bickel and Levina [4] pioneered the theoretical analysis of sparse covariance matrices in high dimensional space. They considered the case where the magnitude of entries in the covariance matrix decayed at a polynomial rate with their distance from the diagonal, and showed that banding the sample covariance matrix resulted in well-behaved estimation. Further, Cai et al. [5] established min–max convergence rates for estimating this type of covariance matrix. A more general model was investigated in [3], where the rows or columns of the covariance matrix were assumed to come from an $l_p(0 < p < 1)$ ball.

Additionally, instead of focusing the sparsity of a covariance matrix, other methods focus on the sparsity of the precision matrix. This type of sparsity naturally relates to the problem of covariance selection, making it appealing in many applications. Yuan and Lin [28] proposed to impose an l_1-norm penalty on the entries of the precision matrix when maximizing the log-likelihood, and therefore encouraged some of the estimated entries to be exactly zero. A similar approach was taken by d'Aspremont et al. [6]. However, one main disadvantage of this method is the large computation time, which has been addressed by [6,13,22]. Moreover, theoretical properties related to computation time have been developed by [28,21,16], etc. In particular, the results of [21,22] suggested that, the extent to which the sparsity of a precision matrix requires well-behaved covariance matrix estimates remains unclear. In very recent years, Yuan [27] showed that the estimability of a high dimensional precision matrix was related to how well it could be approximated by a graphical model, the model is with a relatively low degree. Taking advantage of the connection between multivariate linear regression and calculation of the precision matrix, he found that the use of linear programming was adaptive to different types of sparsity. From another view, Aune et al. [1] mentioned that evaluating the log-likelihood target function, which contains large, sparse precision matrices, is infeasible. He presented an approach for evaluating likelihoods that utilized matrix functions and Krylov subspaces. The technique required only the computation of matrix–vector products. In our work, we focus on estimating sparse high-dimensional precision matrices in GMM from an adaptive angle by using a two-layer hierarchical Bayes model.

3. Background

In this section, we review two existing methods, the Gaussian Mixture Model and the hierarchical Bayes view of adaptive sparse prior. The two methods motivated the ASGMM method.

3.1. Gaussian Mixture Model

GMM is a generative approach to clustering. Each cluster corresponds to a Gaussian distribution, and each example is drawn from its cluster-specific distribution. Given the observed data $\mathbf{X}$ with N independent samples, the goal of GMM is to maximize the log-likelihood of the observed data, when the number of clusters K is known. The formula can be expressed as follows:

$$\begin{aligned}\ln p(\mathbf{X}|\Theta) &= \ln \prod_{n=1}^{N} p(\mathbf{x}_n|\Theta) = \sum_{n=1}^{N} \ln p(\mathbf{x}_n|\Theta) \\ &= \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k) \end{aligned} \tag{1}$$

where $\Theta = \{(\pi_k, \mu_k, \Sigma_k) : k = 1, \ldots, K\}$ is the collection of all unknown parameters, and π_ks are nonnegative coefficients such that $\sum_{k=1}^{K} \pi_k = 1$. The log of the sum component makes optimization difficult. Fortunately, one can use the EM method to learn these parameters by defining a Q-Function.

3.2. Hierarchical Bayes view of adaptive sparse prior

Assume that each β_i has a zero-mean Gaussian prior, with its own variance τ_i, and each τ_i has an exponential hyperprior

$$p(\beta_i|\tau_i) = \mathcal{N}(\beta_i|0, \tau_i), \quad p(\tau_i|\eta) = \frac{\eta}{2} \exp\left\{-\frac{\eta}{2}\tau_i\right\}, \quad \text{for } \tau_i \geqslant 0. \tag{2}$$

The marginal distribution of β_i with respect to η is to the integral of τ_i,

$$p(\beta_i|\eta) = \int_0^{\infty} p(\beta_i|\tau_i) p(\tau_i|\eta) \mathrm{d}\tau_i = \frac{\sqrt{\eta}}{2} \exp\{-\sqrt{\eta}|\beta_i|\}, \tag{3}$$

which obtains a *Laplaceprior* [24]. The result indicates that the Laplace prior is equivalent to a two-level hierarchical Bayes model. This equivalence has been previously exploited to derive EM algorithms for robust regression under Laplacian noise models [17].

The degree of sparsity in GLASSO and GMLASSO depends on the parameter λ on the l_1 regularization term (see [13,23] for the detail). However, in the hierarchical Bayes model, it is controlled by η. Figueiredo [11] suggested removing η from the model by replacing the exponential hyperprior on each τ_i. This can be accomplished through the Jeffrey's noninformative hyperprior:

$$p(\tau_i) \propto \frac{1}{\tau_i} \tag{4}$$

This prior expresses ignorance to scale, and, more importantly, it is *hyperparameter-free*. Although this hyperprior is improper (its integral is not finite), it can yield sparse solutions and lead to good performance in regression and classification problems [11].

4. Adaptive Sparsity in Gaussian Mixture Model (ASGMM)

In this section, we introduce our hierarchical Bayes model for adaptive sparse precision matrices estimation in GMM. In contrast to [11] which utilizes a hierarchical Bayes prior for supervised sparse regression and classification, our model is constructed for unsupervised parameter estimation. In what follows, we give detailed explanations of the method, which involves three major steps.

4.1. Hierarchical Bayes prior in GMM

For GMM of high dimensionality, the number of parameters to be estimated in each precision matrix is very large. However, in real-world applications, the observed data is often not sufficient for estimating all these parameters. Fortunately, in structure learning view, the graph structure with respect to each precision matrix is sparse. That is, most entries of each precision matrix are zeros, and there is a one-to-one correspondence between zero entries and absent edges in the graph. Therefore, our task of parameter estimation is transformed to identifying as many zeros in each precision matrix as possible.

As mentioned above in Eq. (2) and (3), the two-layer hierarchical Bayes interpretation of the Laplace prior, as modified by the Jeffrey's noninformative hyperprior in Eq. (4), exhibits the strong ability to achieve sparsity adaptively. Therefore, we construct a sparse-induced hierarchical Bayes prior which is then imposed on the entries of each precision matrix. To be specific, we rewrite $\mathbf{X}$ as $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_K]$, where $\mathbf{X}_k$ contains the data generated from the kth Gaussian: $\mathbf{X}_k \sim \mathcal{N}(\mathbf{X}_k|\mu_k, \Sigma_k), \Omega_k = \Sigma_k^{-1}$. We then have the following hierarchical model:

$$\begin{aligned} &P(\Omega_{k,ij}|\tau_k, ij) = \mathcal{N}(\Omega_{k,ij}|0, \tau_{k,ij}), \quad \text{for } i > j \\ &P(\tau_{k,ij}) \propto \frac{1}{\tau_{k,ij}} \end{aligned} \tag{5}$$

Note that the parameters to be estimated are the entries of precision matrices, i.e., $\Omega_{k,ij}$. To better correspond with the first layer in Eq. (2), we adopt the prior that each $\Omega_{k,ij}$ follows the normal distribution. In fact, this is a very common way to model prior that satisfies the normal distribution for its simplicity, analyticity, and commonality. What is more, the assumption about the precision matrices are independent and normal distribution is based on the GMM's assumption. In GMM each cluster corresponds to an independent normal distribution. Similarly, the second layer of Eq. (5) that adopts the Jeffreys noninformative hyperprior aims at satisfying Eq. (4), the Jeffreys noninformative hyperprior is constrained on each latent variable $\tau_{k,ij}$.

One thing to be emphasized is that we only put the prior on off-diagonal entries of each Ω_k, that is, we do not seek to shrink the diagonal entries since they are nonzeros in essence. In the following, we will treat each $\tau_{k,ij}$ as a latent variable to be integrated out, leaving us with a log posterior $\ln p(\Omega_k|\mathbf{X})$ that has no hyperparameter to be tuned. This is the main advantage over the parametric l_1-norm regularization and Laplace prior.

4.2. Adaptive sparse precision matrices estimation

In this subsection, we construct the objective function of our ASGMM method by incorporating sparsity into each precision matrix while removing the need for hyperparameter tuning. Then, we use our hierarchical Bayes–Laplace prior to develop a MAP estimate procedure solved by the EM method.

Our objective function of ASGMM is defined as

$$\begin{aligned} l(\Theta|\mathbf{X}) &= \sum_{n=1}^{N} \ln \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k) + \sum_{k=1}^{K} f(\Omega_k) \\ &\geqslant \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_k(\mathbf{x}_n) \ln \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)}{\gamma_k(\mathbf{x}_n)} + \sum_{k=1}^{K} f(\Omega_k) \\ &= \sum_{k=1}^{K} \left\{ \sum_{n=1}^{N} \gamma_k(\mathbf{x}_n) \ln \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)}{\gamma_k(\mathbf{x}_n)} + f(\Omega_k) \right\} \end{aligned} \tag{6}$$

where $\gamma_k(\mathbf{x}_n) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\mu_k,\Sigma_k)}{\sum_l \pi_l \mathcal{N}(\mathbf{x}_l|\mu_l,\Sigma_l)}$ is the responsibility that cluster k takes for $\mathbf{x}_n$. Note that the inequality in the second line holds because ln-function is concave and because $\sum_{k=1}^{K} \gamma_k(\mathbf{x}_n) = 1$ is regarded as a probability distribution. Thus, it satisfies the condition of *Jensen's Inequality*. The term $f(\Omega_k)$ is a function defined over each precision matrix Ω_k and determines the adaptive sparsity of parameters to be estimated. Its exact form will be given later.

Maximizing $l(\Theta|\mathbf{X})$ explicitly is quite difficult. The common strategy is to construct a tight lower-bound on $l(\Theta|\mathbf{X})$ by defining a Q-Function used in the EM method. Specifically, in **E-Step**, we define the following Q-Function:

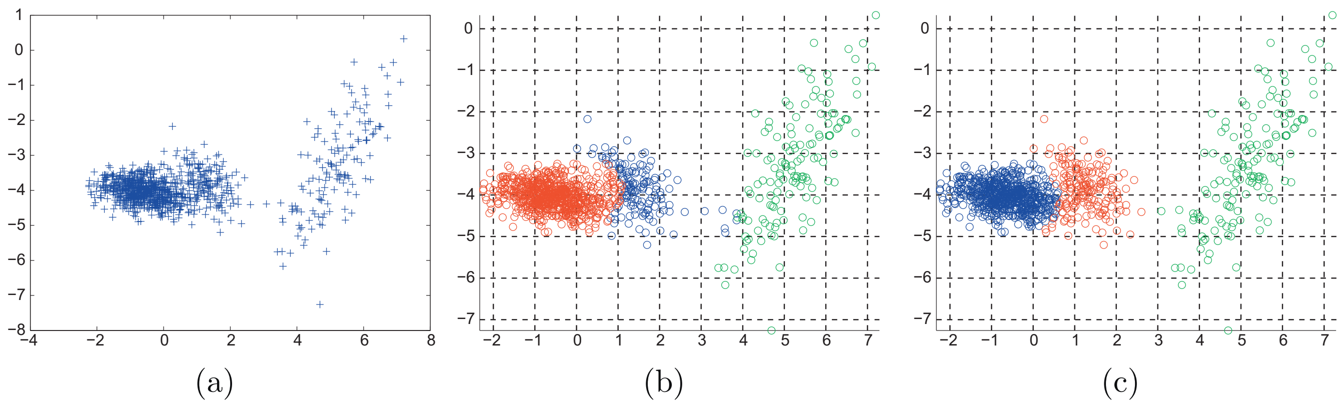

Fig. 1. (a) Original synthetic data; (b) GMLASSO clustering result; (c) ASGMM clustering result.

$$\begin{aligned} Q'(\Theta,\Theta^{(t)}) &= \sum_{k=1}^{K}\left\{\sum_{n=1}^{N}\gamma_k(\mathbf{x}_n)\ln\frac{\pi_k\mathcal{N}(\mathbf{x}_n|\mu_k,\Sigma_k)}{\gamma_k(\mathbf{x}_n)} + f\left(\Omega_k|\Omega_k^{(t)}\right)\right\} \\ &= \sum_{k=1}^{K}\left\{\ln p(\mathbf{X}|\Theta_k) + f\left(\Omega_k|\Omega_k^{(t)}\right)\right\} = \sum_{k=1}^{K}Q'_k\left(\Theta_k,\Theta_k^{(t)}\right)\end{aligned}$$

where $\ln p(\mathbf{X}|\Theta_k) = \sum_{n=1}^{N}\gamma_k(\mathbf{x}_n)(\ln(\pi_k\mathcal{N}(\mathbf{x}_n|\mu_k,\Sigma_k)) - \ln\gamma_k(\mathbf{x}_n))$ is the data log-likelihood.

For simplicity, as the estimates for π_k and μ_k are actually the same as GMM, we just allow for maximizing Q'_k with respect to Ω_k. Hence, for each Q'_k, it reduces to

$$Q'_k\left(\Omega_k,\Omega_k^{(t)}\right) = \ln p(\mathbf{X}|\Omega_k) + f\left(\Omega_k|\Omega_k^{(t)}\right) \tag{7}$$

As mentioned above, the Q-Function is a lower-bound on the log-posterior objective function $l(\Omega|\mathbf{X})$ ($l(\Theta|\mathbf{X})$ reduces to $l(\Omega|\mathbf{X})$) we wish to maximize.It is defined as the expectation of the log-posterior over latent variables τ_k, when given the estimated parameters $\Omega_k^{(t)}$ at iteration t. Thus, we have

$$\begin{aligned} Q'_k\left(\Omega_k,\Omega_k^{(t)}\right) &= \mathbb{E}\left[\ln p(\Omega_k|\tau_k,\mathbf{X})\Big|\mathbf{X},\Omega_k^{(t)}\right] \\ &= \int \ln p(\Omega_k|\tau_k,\mathbf{X})p\left(\tau_k|\mathbf{X},\Omega_k^{(t)}\right)\mathrm{d}\tau_k\end{aligned}$$

Note that $p(\Omega_k|\tau_k,\mathbf{X}) \propto p(\Omega_k|\tau_k)p(\mathbf{X}|\Omega_k,\tau_k) = p(\Omega_k|\tau_k)p(\mathbf{X}|\Omega_k)$, and similar strategy has been used in [11]. We can then split the log-posterior inside the integral into two parts, i.e., the data log-likelihood that does not depend on τ_k, and the parameter log-prior that depends on τ_k. Specifically,

$$\begin{aligned} Q'_k(\Omega_k,\Omega_k^{(t)}) &= \sum_{i=2}^{D}\sum_{j=1}^{i-1}\int\{\ln p(\mathbf{X}|\Omega_{k,ij}) \\ &\quad + \ln p(\Omega_{k,ij}|\tau_{k,ij})\}p(\tau_{k,ij}|\mathbf{X},\Omega_{k,ij}^{(t)})\mathrm{d}\tau_{k,ij} \\ &= \sum_{i=2}^{D}\sum_{j=1}^{i-1}\ln p(\mathbf{X}|\Omega_{k,ij}) + \sum_{i=1}^{D}\sum_{j=1}^{i-1}\int \ln p(\Omega_{k,ij}|\tau_{k,ij})p(\tau_{k,ij}|\mathbf{X},\Omega_{k,ij}^{(t)})\mathrm{d}\tau_{k,ij} \\ &= \ln p(\mathbf{X}|\Omega_k) + \sum_{i=2}^{D}\sum_{j=1}^{i-1}\int \ln p(\Omega_{k,ij}|\tau_{k,ij})p(\tau_{k,ij}|\mathbf{X},\Omega_{k,ij}^{(t)})\mathrm{d}\tau_{k,ij}\end{aligned}$$

Here, when we compare Eq. (7) with the last line of the above equation, we naturally have the exact form of our pre-given $f(\Omega_k|\Omega_k^{(t)})$, that is

$$\begin{aligned} f\left(\Omega_k|\Omega_k^{(t)}\right) &= \sum_{i=2}^{D}\sum_{j=1}^{i-1}f\left(\Omega_{k,ij}|\Omega_{k,ij}^{(t)}\right) \\ &= \sum_{i=2}^{D}\sum_{j=1}^{i-1}\int \ln p(\Omega_{k,ij}|\tau_{k,ij})p\left(\tau_{k,ij}|\mathbf{X},\Omega_{k,ij}^{(t)}\right)\mathrm{d}\tau_{k,ij} \\ &= \sum_{i=2}^{D}\sum_{j=1}^{i-1}\frac{\frac{1}{2}\tau_{k,ij}^{-2}\Omega_{k,ij}^{2}\mathcal{N}\left(\Omega_{k,ij}^{(t)}|0,\tau_{k,ij}^{-1}\right)\mathrm{d}\tau_{k,ij}}{\tau_{k,ij}^{-1}\mathcal{N}\left(\Omega_{k,ij}^{(t)}|0,\tau_{k,ij}^{-1}\right)\mathrm{d}\tau_{k,ij}} \\ &= \sum_{i=2}^{D}\sum_{j=1}^{i-1}\left(\Omega_{k,ij}^{(t)}\right)^{-2}(\Omega_{k,ij})^2 \\ &= \mathrm{tr}\left((\Gamma_k^{(t)}\circ\Omega_k)(\Gamma_k^{(t)}\circ\Omega_k)^T\right)\end{aligned} \tag{8}$$

where each element in Γ_k is the inverse of that in Ω_k, i.e., $\Gamma_{k,ij} = \frac{1}{\Omega_{k,ij}}$. $A\circ B$ refers to the *Hadamard (element-wise) product* of matrix A and B. The integral result is very close to what is in [11].

In **M-Step**, we maximize Eq. (7) with respect to each Ω_k. By substituting $\ln p(\mathbf{X}|\Omega_k)$ at iteration t into Eq. (7) and removing terms that are irrelevant to Ω_k, we have

$$\begin{aligned} \Omega_k^{(t+1)} &= arg\,max_{\Omega_k}\frac{1}{2}\sum_{n=1}^{N}\gamma_k^{(t)}(\mathbf{x}_n)\left\{\ln|\Omega_k| - \left(\mathbf{x}_n-\mu_k^{(t)}\right)^T\Omega_k(\mathbf{x}_n-\mu_k^{(t)})\right\} \\ &\quad + \mathrm{tr}\left((\Gamma_k^{(t)}\circ\Omega_k)\left(\Gamma_k^{(t)}\circ\Omega_k\right)^T\right) \Longleftrightarrow arg\,max_{\Omega_k}\left\{\ln|\Omega_k| - \mathrm{tr}\left(\Omega_k\mathbf{S}_k^{(t)}\right)\right. \\ &\quad \left. + c^{(t)}\cdot\mathrm{tr}\left(\left(\Gamma_k^{(t)}\circ\Omega_k\right)\left(\Gamma_k^{(t)}\circ\Omega_k\right)^T\right)\right\}\end{aligned} \tag{9}$$

where $\mathbf{S}_k^{(t)} = \frac{1}{\gamma_k^{(t)}}\sum_{n=1}^{N}\gamma_k^{(t)}(\mathbf{x}_n)\left(\mathbf{x}_n-\mu_k^{(t)}\right)\left(\mathbf{x}_n-\mu_k^{(t)}\right)^T, c^{(t)} = \frac{1}{\gamma_k^{(t)}}$, and $\gamma_k^{(t)} = \sum_n\gamma_k^t(\mathbf{x}_n)$. Note that terms in the last line of the above equation are divided by the constant $\gamma_k^{(t)}$.

So, using a gradient-based method, we write the optimal condition for the solution Ω_k to Eq. (9) as

$$\frac{\mathrm{d}Q'_k\left(\Omega_k,\Omega_k^{(t)}\right)}{\mathrm{d}\Omega_k} = \Omega_k^{-1} + \mathbf{S}_k^{(t)} + 2c^{(t)}\cdot\Gamma_k^{(t)}\circ\Omega_k\circ\Gamma_k^{(t)} = \mathbf{0}$$

By multiplying Ω_k, since a Hadamard product satisfies the commutative law, we finally obtain:

$$\mathbf{I}_k + \mathbf{S}_k^{(t)}\Omega_k + 2c^{(t)}\cdot\left(\Gamma_k^{(t)}\circ\Gamma_k^{(t)}\circ\Omega_k\right)\Omega_k = \mathbf{0} \tag{10}$$

where $\mathbf{I}_k$ is the identity matrix with size $D\times D$.

The solution to matrix-based optimization problem in Eq. (10) is difficult to achieve, and more importantly, the *positive-definite* property of Ω_k is not directly used, so in the next subsection, we rewrite the matrix form into its corresponding element form through Cholesky decomposition.

4.3. Incorporate Cholesky decomposition into ASGMM

We denote the Cholesky decomposition of Ω_k as $\Omega_k = L_kL_k^T$, where L_k is a lower triangular. Then, the lower triangular entries of Ω_k are easily calculated by

$$\Omega_{k,ij} = \sum_{s=1}^{j}L_{k,is}L_{k,js}, \quad \text{for } j\leqslant i \tag{11}$$

Note that this formulation naturally enforces the *symmetric* and *positive-definite* properties of Ω_k: $\Omega_k = L_kL_k^T$ is without doubt symmetric in essence and Ω_k being positive-definite is equivalent to the diagonal entries of L_k being strictly positive.

In what follows, we rewrite Eq. (9) into the element form by substituting Eq. (11) into it. For the first two terms, we have

$$\begin{aligned} \ln|\Omega_k| - \mathrm{tr}\left(\Omega_k\mathbf{S}_k^{(t)}\right) &= 2\sum_{j=1}^{D}\ln L_{k,jj} - \sum_{i=1}^{D}\sum_{j=1}^{D}\Omega_{k,ij}S_{k,ij}^{(t)} \\ &= 2\sum_{j=1}^{D}\ln L_{k,jj} - \sum_{i=1}^{D}\left(2\sum_{j=1}^{i-1}\sum_{s=1}^{j}S_{k,ij}^{(t)}L_{k,is}L_{k,js} + \sum_{s=1}^{i}S_{k,ii}^{(t)}L_{k,is}^2\right)\end{aligned} \tag{12}$$

The two terms inside the parentheses of Eq. (12) correspond to the off-diagonal and diagonal elements of Ω_k respectively.

Similarly, for the last term, we have

$$\begin{aligned} & c^{(t)}\cdot\mathrm{tr}\left(\left(\Gamma_k^{(t)}\circ\Omega_k\right)\left(\Gamma_k^{(t)}\circ\Omega_k\right)^T\right) \\ &= c^{(t)}\cdot\sum_{i=2}^{D}\sum_{j=1}^{i-1}\left(\Omega_{k,ij}^{(t)}\right)^{-2}(\Omega_{k,ij})^2 \\ &= c^{(t)}\cdot\sum_{i=2}^{D}\sum_{j=1}^{i-1}\left(\Omega_{k,ij}^{(t)}\right)^{-2}\left(\sum_{s=1}^{j}L_{k,is}L_{k,js}\right)^2\end{aligned} \tag{13}$$

Thus, the **M-Step** is solved by taking the derivative of Q'_k with respect to the off-diagonal elements $L_{k,pq}(p>q)$ and diagonal elements $L_{k,pp}$, and then setting them to zero. Specifically

Table 1
Estimated errors measured by two criteria on Model 1 (M1) and Model 2 (M2) and consumed time with $N = 100$ and $D = 30, 50, 100$.

	Method	GMM		GMLASSO			ASGMM		
Model	D	ASN	AFN	ASN	AFN	Time (s)	ASN	AFN	Time (s)
M1	30	38.57	55.27	10.18	29.62	0.24	10.76	31.43	0.41
M1	50	53.25	67.44	10.39	38.75	0.85	12.88	40.66	1.67
M1	100			20.11	62.26	13.88	20.56	61.58	20.33
M2	30	5.78	9.08	0.97	1.36	0.21	1.08	2.25	0.36
M2	50	5.30	16.65	1.08	1.65	1.31	1.17	2.31	1.62
M2	100			1.40	3.03	15.32	1.49	3.59	19.85

Notes: Empty cells mean not applicable. The same below.

$$\frac{dQ'_k}{dL_{k,pq}} = -2\left(\sum_{j=q}^{p-1} S^{(t)}_{k,pj} L_{k,jq} + S^{(t)}_{k,pp} L_{k,pq}\right) + 2c^{(t)} \cdot \left(\sum_{j=q}^{p-1} \left(\Omega^{(t)}_{k,pj}\right)^{-2} L_{k,jq} \sum_{s=1}^{j} L_{k,ps} L_{k,js} + \sum_{i=p+1}^{D} \left(\Omega^{(t)}_{k,ip}\right)^{-2} L_{k,iq} \sum_{s=1}^{p} L_{k,is} L_{k,ps}\right) \tag{14}$$

and

$$\frac{dQ'_k}{dL_{k,pp}} = \frac{2}{L_{k,pp}} - 2c^{(t)} \cdot S^{(t)}_{k,pp} L_{k,pp} \tag{15}$$

Notice that Eq. (14) is a linear function with $L_{k,pq}$, so the maximization of Q'_k with respect to $L_{k,pq}$ is equivalent to solving a linear equation $aL_{k,pq} + b = 0$ with

$$a = -2S^{(t)}_{k,pp} + 2c^{(t)} \cdot \left(\sum_{j=q}^{p-1} \left(\Omega^{(t)}_{k,pj}\right)^{-2} L^2_{k,jq} + \sum_{i=p+1}^{D} \left(\Omega^{(t)}_{k,ip}\right)^{-2} L^2_{k,iq}\right)$$

$$b = -2\sum_{j=q}^{p-1} S^{(t)}_{k,pj} L_{k,jq} + 2c^{(t)} \cdot \left(\sum_{j=q}^{p-1} \frac{L_{k,jq}}{\left(\Omega^{(t)}_{k,pj}\right)^2} \sum_{\substack{s=1\\ s\neq q}}^{j} L_{k,ps} L_{k,js} + \sum_{i=p+1}^{D} \frac{L_{k,iq}}{\left(\Omega^{(t)}_{k,ip}\right)^2} \sum_{\substack{s=1\\ s\neq q}}^{p} L_{k,is} L_{k,ps}\right)$$

Analogously, from Eq. (15), we can simply get $L_{k,pp} = (c^{(t)} \cdot S^{(t)}_{k,pp})^{-\frac{1}{2}}$. Returning to the Cholesky decomposition, one should recall that the positive-definite property requires that

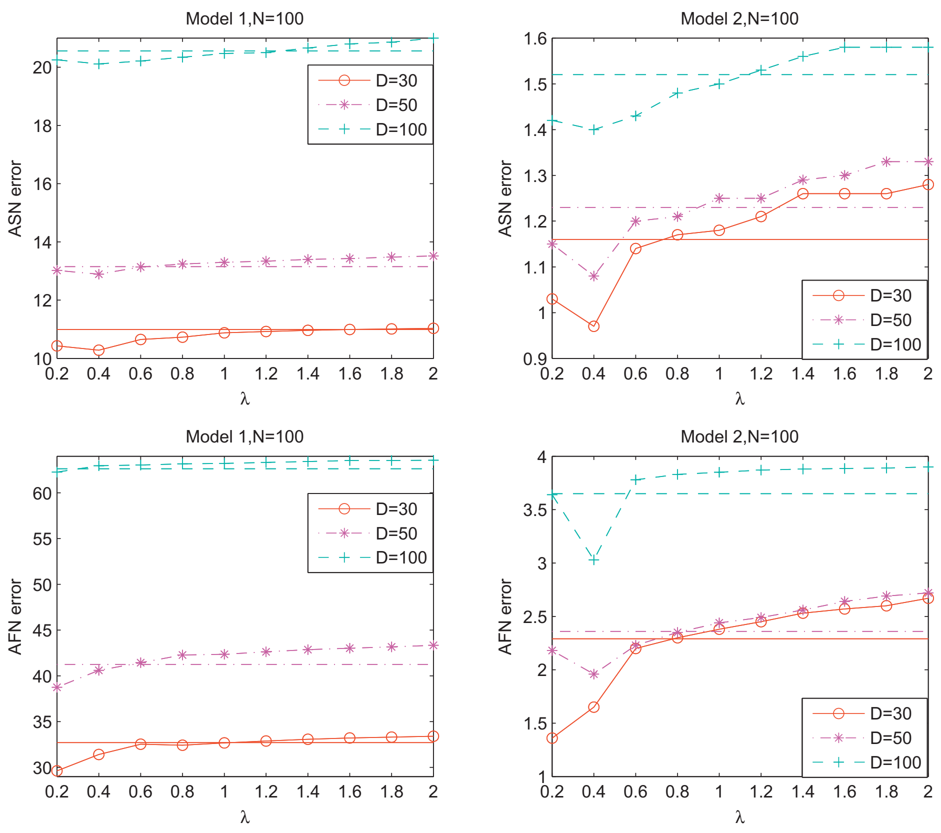

Fig. 2. Estimated errors of GMLASSO vs. the hyperparameter λ measured by two criteria on Model 1 and Model 2. The horizon line in each figure is the corresponding result of ASGMM.

Table 2
Estimated errors measured by two criteria on Model 1 (M1) and Model 2 (M2) with $D = 100$ and $N = 200, 400$.

	Method	GMM		GMLASSO		ASGMM	
Model	N	ASN	AFN	ASN	AFN	ASN	AFN
M1	200			15.23	57.91	16.55	57.75
M1	400	21.54	83.64	9.40	56.56	9.67	56.61
M2	200			1.15	2.73	1.24	2.73
M2	400	2.61	13.45	1.07	2.27	1.17	2.42

Table 3
Estimated degree of sparsity on Model1(M1) with $D = 30, 50, 100$ and $N = 100, 200, 400$.

D	N						
	100	200	400	100	200	400	
M1	ASGMM			GMLASSO			STD
30	0.888	0.889	0.902	0.902	0.902	0.902	0.902
50	0.926	0.937	0.941	0.941	0.941	0.941	0.941
100	0.927	0.966	0.970	0.970	0.970	0.970	0.970
M2	ASGMM			GMLASSO			STD
30	0.944	0.967	0.967	0.967	0.967	0.967	0.967
50	0.958	0.980	0.980	0.980	0.980	0.980	0.980
100	0.966	0.985	0.988	0.988	0.988	0.988	0.988

Notes: "STD" item means the standard degree of sparsity of the precision matrices on the given Model.

$L_{k,pp}$ is strictly positive. Here, from its exact form, it is indeed the case.

One important thing to be noted in the implementation is that many entries of Ω_k are forced to zero and we need to divide by them, so we add a fixed small epsilon to each $\Omega_{k,ij}$ to ensure numerical stability. We summarize our ASGMM algorithm in "Algorithm 1":

Algorithm 1 Hierarchical Bayes Gaussian Mixture Model

```
Input
    N observed data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{D\times N}$, the number
  of clusters K.
Output
    $\{\pi_k\}_{k=1}^K, \{\mu_k\}_{k=1}^K, \{\Omega_k\}_{k=1}^K$.
    Step 1: initialize $\{\pi_k^{(0)}\}_{k=1}^K, \{\mu_k^{(0)}\}_{k=1}^K$, and $\{L_k^{(0)}\}_{k=1}^K$ by
  using K-Centroid method.
    Step 2: for each iteration, update the estimate for each
  mixture component k individually.
        E – Step: calculate the unknowns
            $\gamma_k^{(t+1)}(\mathbf{x}_n) = \frac{\pi_k^{(t)}\mathcal{N}(\mathbf{x}_n|\mu_k^{(t)},\Sigma_k^{(t)})}{\sum_l \pi_k^{(t)}\mathcal{N}(\mathbf{x}_l|\mu_l^{(t)},\Sigma_l^{(t)})}$.
        M – Step: update the parameters
            $\pi_k^{(t+1)} = \frac{1}{N}\sum_{n=1}^N \gamma_k^{(t+1)}(\mathbf{x}_n)$.
            $\mu_k^{(t+1)} = \frac{\sum_{n=1}^N \gamma_k^{(t+1)}(\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^N \gamma_k^{(t+1)}(\mathbf{x}_n)}$.
            $\mathbf{S}_k^{(t+1)} = \sum_{n=1}^N \left(\frac{\gamma_k^{(t+1)}(\mathbf{x}_n)}{\sum_{j=1}^N \gamma_k^{(t+1)}(\mathbf{x}_j)}\right)\left(\mathbf{x}_n - \mu_k^{(t+1)}\right)\left(\mathbf{x}_n - \mu_k^{(t+1)}\right)^T$.
            $L_{k,ij(i\neq j)}^{(t+1)} = -b/a$.
            $L_{k,ii}^{(t+1)} = (c^{(t+1)} \cdot S_{k,ii}^{(t+1)})^{-\frac{1}{2}}$.
            $\Omega_{k,ij}^{(t+1)} = \sum_{s=1}^j L_{k,is}^{(t+1)} L_{k,js}^{(t+1)}$,   for   $j \leqslant i$
    Step 3: go back to step 2 until reaching the maximal
  iteration or the predefined threshold.
```

5. Experimental results

In this section, several experiments are carried out to show the performance of our ASGMM method. All experiments are conducted in Matlab R2010b with the platform Intel(R) Core(TM) i3-2100, CPU 3.10 GHz, and RAM 16.0 GB. In each experiment, we initialize the parameters of GMM, GMLASSO, and ASGMM by using K-Centroid method, setting the maximal iteration as 100, and running each method 20 times. Through this coarse clustering, the results are stable in each run with specific iterations. As for ASGMM, in order to avoid zero values in the denominator, we add a small epsilon. Here, the magnitude of epsilon is 1e-4 in all experiments.

5.1. Synthetic data clustering validation

We begin with a simple visualized clustering experiment to verify the correctness of our ASGMM method. The original data points are generated from a GMM formed by three Gaussian distributions. Clearly, there are three natural clusters. The original synthetic data and ASGMM clustering result are shown in Fig. 1(a) and (c). As a comparison, we also present the GMLASSO clustering result where $\lambda = 0.2$ in Fig. 1(b). Intuitively, ASGMM yields the ideal clustering result and is a little better than that of GMLASSO.

5.2. Estimated error and degree of sparsity

In this part, we respectively carry out two experiments to test the estimated error and degree of sparsity of precision matrices. We compare ASGMM, GMM, and GMLASSO.

In the first experiment, we fix the number of clusters $K = 2$. The sample size $N = 100$ and each covariance matrix size $D^2 = 30^2, 50^2$, or 100^2. For convenience sake, the mean vector of each mixture is set to be $\mathbf{0}$. Here, we allow for two covariance structures:

Model 1: the covariance matrix for both clusters follows an Autoregressive(AR(1)) model (the precision matrices are tridiagonal with $D^2 - 3D + 2$ zero entries and $3D - 2$ non-zero entries):

$$\Sigma_1(i,j) = 0.4^{|i-j|}, \quad \Sigma_2(i,j) = 0.5 \times 0.8^{|i-j|}.$$

Model 2: both covariance matrices are diagonal (the precision matrices are diagonal with $D^2 - D$ zero entries and D non-zero entries):

$$\Sigma_1(j,j) = \log(j+1), \quad \Sigma_2(j,j) = \log(D+2-j).$$

We examine the estimated error through two criteria: the averaged spectral norm (ASN) of the difference between the estimating precision matrix and the truth,

$$ASN = \frac{1}{K}\sum_{k=1}^K \left\|\hat{\Sigma}_k^{-1} - \Sigma_k^{-1}\right\|$$

where $||A||$ is the largest singular value of A, and the average Frobenius norm (AFN) error

$$AFN = \frac{1}{K}\sum_{k=1}^K \sqrt{\sum_{i,j}\left(\hat{\Sigma}_k^{-1}(i,j) - \Sigma_k^{-1}(i,j)\right)^2}$$

The results, averaged over 20 runs for each model are reported in Table 1. It is clear from the results that, sparse based ASGMM and GMLASSO methods outperform the dense based GMM method to a large extent for both models. When the size of the precision matrix increases, the superiority becomes more evident. Note that, for GMLASSO, we select the optimal result in each case with hyper-parameter λ ranging from 0.2 to 2 with a gap 0.2. In order to see the detailed estimated errors of GMLASSO versus λ on M1 and M2, we

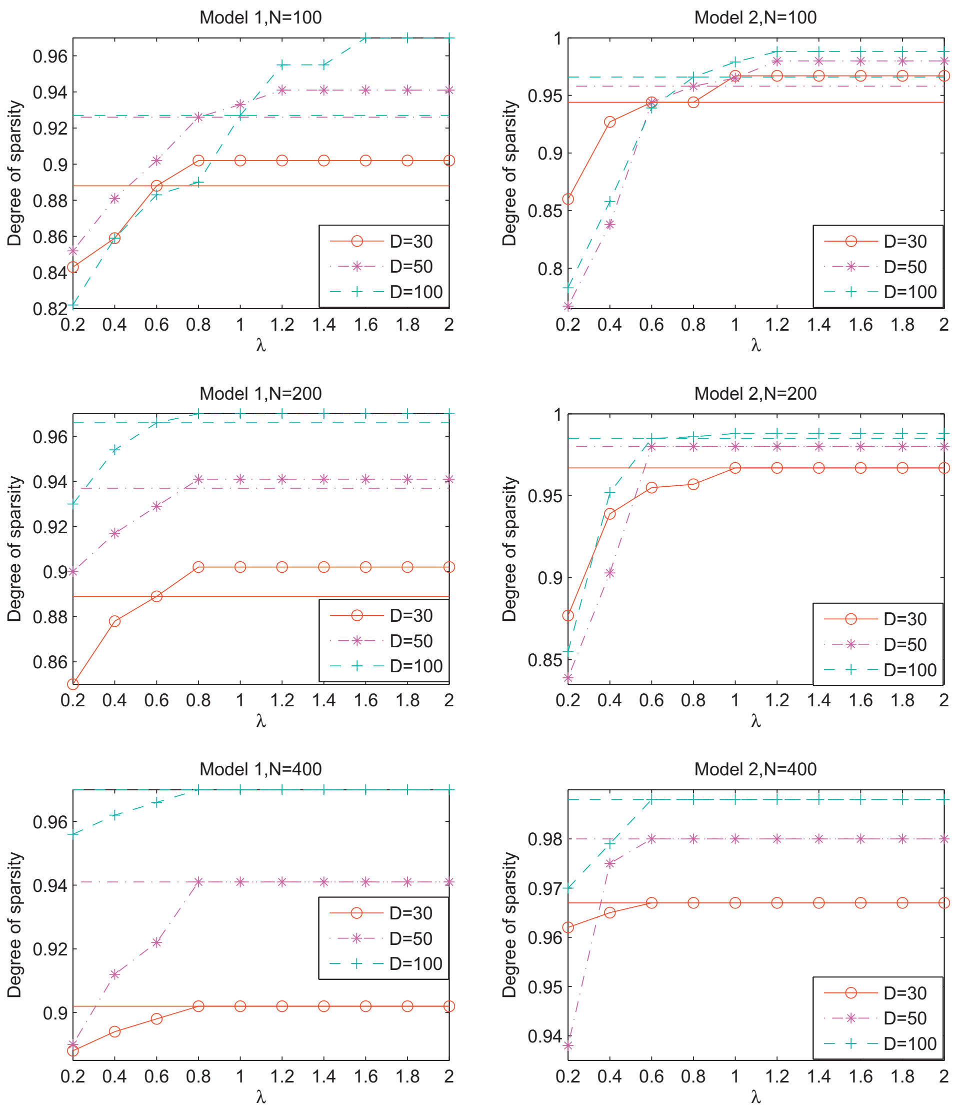

Fig. 3. Estimated degree of sparsity of GMLASSO vs. the hyperparameter λ measured on Model 1 and Model 2. The horizon line in each figure is the corresponding result of ASGMM.

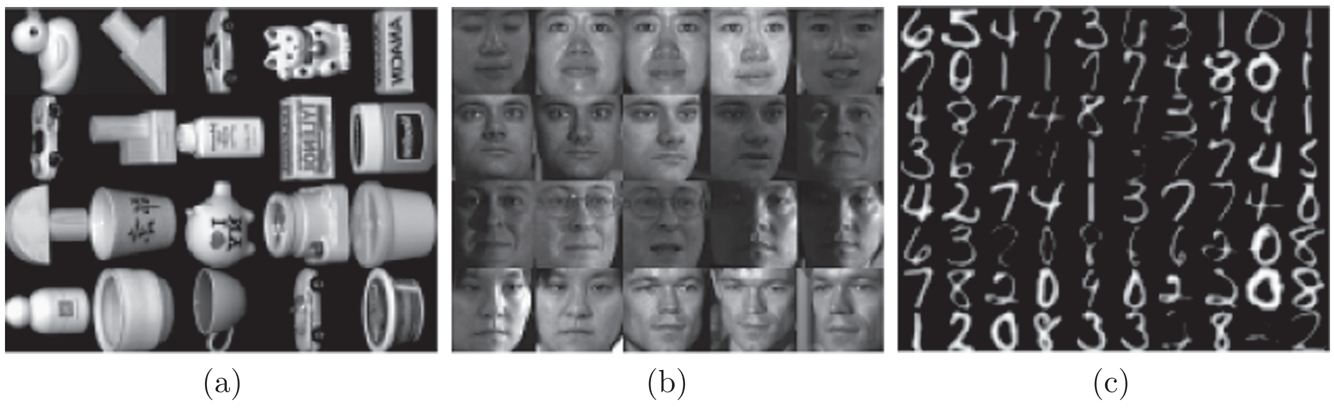

Fig. 4. Sample images from the (a) COIL20 image library, (b) CMU PIE face dataset, and (c) USPS handwritten digits.

plot them in Fig. 2. As a comparison, we also show the estimated errors of ASGMM, that is, the horizontal line in each subfigure. From it we notice that, when λ is small, both the AFN and ASL errors of GMLASSO of each model are also small, and the optimal results are in cases where $\lambda = 0.2$ or $\lambda = 0.4$. As λ further increases, the AFN and ASL errors of both models increase accordingly, and then are larger than those of ASGMM. The reason is that, since λ controls the degree of sparsity of GMLASSO, and the l_1-norm

Table 4
Clustering performance measured by NMI on COIL20.

K	K-Means	NMF	GMM	GMLASSO	ASGMM
4	**0.36**	0.35	0.36	0.33	0.34
6	0.45	**0.47**	0.42	0.39	0.46
8	0.54	0.52	0.50	0.53	**0.54**
10	0.57	0.59	0.54	0.57	**0.60**
12	0.63	0.64	0.61	0.61	**0.65**
14	0.65	0.66	0.64	0.66	**0.69**
16	0.69	0.70	0.66	0.69	**0.72**
18	0.74	0.74	0.70	0.74	**0.75**
20	0.76	0.76	0.73	0.76	**0.77**

Table 5
Clustering performance measured by NMI on CMU PIE.

K	K-Means	NMF	GMM	GMLASSO	ASGMM
5	0.41	**0.45**	**0.45**	0.42	0.43
10	0.47	**0.60**	0.49	0.58	**0.60**
15	0.57	0.61	0.53	0.61	**0.63**
20	0.57	0.62	0.56	0.63	**0.64**
25	0.60	0.64	0.59	**0.65**	0.64
30	0.59	**0.66**	0.61	0.64	**0.66**
35	0.58	0.67	0.63	0.68	**0.69**
40	0.60	**0.67**	0.64	**0.67**	**0.67**

penalty leads to the biasedness of estimated values of elements in precision matrices, when λ increases, GMLASSO focus more on the degree of sparsity, and thus causes a bigger biasedness. In contrast, our ASGMM achieves the sparse precision matrices in an unbiased way, so it is not only stable but also has estimated errors comparable with the optimal results of GMLASSO. More importantly, GMLASSO needs many iterations to adjust the hyperparameter and obtain each optimal result. However, our method is hyperparameter-free, it is adaptive and executes just once in each run. For comparison, we also list the consumed time of GMLASSO and ASGMM in Table 1. Note that the time-consuming evaluation of GMLASSO covers just 1 time(λ ranges from 0.2 to 2 with a gap 0.2). From the result, we see if it is computed only one time, ASGMM implements slightly slower than GMLASSO. However, if we want to find the optimal result that λ ranges from 0.2 to 2 with a gap 0.2, our ASGMM obviously costs far less time than GMLASSO.

Next, we investigate the effect of sample size given a fixed covariance matrix size. Here, we set $N = 200$ and $N = 400$ with $D^2 = 100^2$. The estimated errors are reported in Table 2. From 2, we can arrive at the following conclusions: on one hand, when the sample size increases, all methods can improve the performance under both criteria. The result is coincident with the fact that the more data the much better. On the other hand, our ASGMM has comparable results with the optimal results obtained by GMLASSO, and the difference between them is analogous to the case aforementioned where sample size $N = 100$.

In the second experiment, our aim is to evaluate the ability of ASGMM and GMLASSO to deal with sparsity. We first report in Table 3 the estimated degree of sparsity of the precision matrices on Model 1 and Model 2, with different sample sizes $N = 100, 200, 400$ and different precision matrix sizes $D^2 = 30^2, 50^2, 100^2$. Note that: first, the threshold for defining sparsity is the same as the epsilon value. I.e., it has the magnitude 1e-4; Second, the standard degrees of sparsity in Model 1 are 0.902, 0.941, 0.970, and in Model 2 are 0.967, 0.980, 0.988 when $D = 30, 50$, and 100 respectively. The results show that both ASGMM and GMLASSO are capable of deeply exploiting sparsity. The high performance they possess comes to the fact that they are sparse based methods. For GMLASSO, in all cases, it can discover 100% degree of sparsity, controlled by the hyperparameter λ. For ASGMM, as the sample size increases, it can also fully discover sparsity. Furthermore, to find the relationship between λ and the estimated degree of sparsity of GMLASSO, we explored the results and showed them in Fig. 3. From it, we see that when λ increases, the sparsity adds correspondingly. When it reaches to a certain value, GMLASSO exploits 100% degree of sparsity.

Returning to previous experimental results, we know that when λ exceeds a specific threshold, both the estimated AFN and ASN errors increase. On the other hand, the increase of sparsity also keeps pace with λ. In this sense, we can safely draw a conclusion that, the acquisition of high sparsity comes at the cost of estimated errors. This leads to the difficulty of balancing the selection of high sparsity and small estimated error. However, our ASGMM method adaptively obtains the unique sparsity and estimated error simultaneously, thus avoiding the problems.

5.3. Real-world data clustering

In this part, we carry out several clustering experiments on three famous real-world datasets, the COIL20 image library dataset,[1] CMU PIE face dataset,[2] and USPS handwritten digits dataset.[3] Notice that each dataset can be regarded as a type of image.

COIL20 image library is from Columbia. It contains 20 objects, overall, and each having 72 images. The images are taken 5 degrees apart as the object is rotated on a turntable and each object has 72 images. The image size is 32 by 32 pixels, with 256 gray levels per pixel. So, each image is a 1024-dimensional vector.

CMU PIE face dataset consists of 41,368 images of 68 people, each person under 13 different poses, 43 different illumination conditions, and with 4 different expressions. In this experiment, we select a subset with 40 people, and each has 42 face images under different lighting and illumination conditions. The image size is 32 by 32 pixels. So, each image is a 1024-dimensional vector.

USPS handwritten digits dataset contains 9298 images of handwritten digits. The digits 0 to 9 have 1553, 1269, 929, 824, 852, 716, 834, 792, 708, and 821 samples, respectively. The digits data were gathered at the Center of Excellence in Document Analysis and Recognition (CEDAR) at SUNY Buffalo, as part of a project sponsored by the US postal Service. The image size is 16 by 16 pixels, with 256 gray levels per pixel. Thus, each image is represented as a 256-dimensional vector. Some sample images from the three datasets are shown in Fig. (4).

In this paper, we choose normalized mutual information (NMI) [26] as the metric to measure the clustering performance. Let C and C' respectively denote the set of clusters obtained from the ground truth and obtained from our algorithm. $H(C)$ and $H(C')$ are the entropies of C and C'. Their mutual information metric $MI(C, C')$ is defined as

$$MI(C, C') = \sum_{c_i \in C, c'_j \in C'} p(c_i, c'_j) \cdot \log_2 \frac{p(c_i, c'_j)}{p(c_i) \cdot p(c'_j)}$$

where $p(c_i), p(c'_j)$ are the probabilities that an arbitrarily selected data point belongs to the clusters $c_i, c'_j)$ and $p(c_i, c'_j)$ is the joint probability that the data point belongs to the clusters c_i and c'_j at the same time. Then, the NMI is

$$NMI(C, C') = \frac{MI(C, C')}{\max(H(C), H(C'))}$$

It is easy to check $NMI(C, C')$ ranges from 0 to 1. $NMI = 1$ if the two sets of clusters are identical, and 0 if they are independent.

[1] http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php
[2] http://www.ri.cmu.edu/projects/project_418.html
[3] http://www.csie.ntu.edu.tw/cjlin/libsvmtools/datasets/multiclass.html#usps

Table 6
Clustering performance measured by NMI on USPS.

K	K-Means	NMF	GMM	GMLASSO	ASGMM
2	0.84	0.79	0.84	0.88	**0.90**
3	0.75	0.78	0.84	**0.87**	0.86
4	0.71	0.74	0.80	0.80	**0.84**
5	0.67	0.71	0.75	0.79	**0.81**
6	0.66	0.71	**0.76**	0.74	**0.76**
7	0.65	0.67	0.70	0.68	**0.73**
8	0.67	0.61	0.64	**0.71**	0.68
9	**0.64**	0.63	0.60	**0.64**	**0.64**
10	0.61	0.63	0.61	**0.65**	0.63

We compare our ASGMM with classical K-Means, GMM, Non-negative Matrix Factorization (NMF) [26], and the GMLASSO methods. To save running time, we apply a PCA transformation to all methods and reduce the dimension to 100. Evaluations are conducted on different number of clusters K. For COIL20, K ranges from 4 to 20; for CMU PIE, it ranges from 5 to 40, and for USPS, it ranges from 2 to 10.

Table 4–6 respectively show the average clustering results of all methods on the three datasets with 20 runs. As a whole, our ASGMM performs the best and GMLASSO takes the second place. On COIL20 and CMU PIE, NMF achieves the results next to ASGMM and GMLASSO. While on USPS, the result obtained by GMM ranks the third. These experimental results uncover several significant and interesting points:

(1) Regardless of the type of dataset, our ASGMM can always achieve the best performance. It demonstrates that the adaptive sparsity with respect to the precision matrices obtained by ASGMM indeed makes data clustering meaningful.
(2) Compared with dense based GMM, both sparse based GMLASSO and ASGMM perform high dimensional data clustering from the viewpoint of structure learning. They use the fact that most entries of the precision matrices are zero, and try to discover these zero entries. The better performance of ASGMM and GMLASSO than that of GMM demonstrates the great importance of identifying the sparse structure of the precision matrices.
(3) All methods obtain relative better results on USPS than those on COIL20 and CMU PIE. This is because USPS contains more samples, and induces two positive effects: on one hand, it validates more samples give the better results, meaning sample size is an important factor for clustering; On the other hand, it proves again that as sample size increases, both ASGMM and GMLASSO can reduce the estimated errors of precision matrices, and thus can improve the clustering performance.
(4) The high performance of NMF on COIL20 and CMU PIE indicates the strength of part-based additive representation in discovering the latent factors hidden in positive value images.

6. Conclusions and future work

In this paper, we have presented a two-layer hierarchical Bayes model which adaptively estimates the sparse precision matrices in GMM from the viewpoint of structure learning. Compared with commonly used hyperparametric and biased l_1 regularization methods, the key advantage of our approach is the absence of biasedness and hyperparameters, which are tuned via the Jeffrey's noninformative prior. ASGMM proved to have comparable performance with the famous LASSO based GMLASSO method when estimating precision matrices and discovering the degree of sparsity in high-dimensional parameter space. Since GMLASSO faces the model selection problem, it needs a lot of cross-validation to achieve the optimal regularization hyperparameter. However, our ASGMM can avoid this problem and achieve the optimal result by executing just one time. Moreover, sparse based ASGMM and GMLASSO perform better than dense based GMM. Finally, when performing real-world data clustering, ASGMM can also obtain good results compared with several classical methods.

Despite these results, we face several problems that remain to be investigated in future work. First, since theoretical guarantees exists in l_1-norm or LASSO based methods, it would be better if we can also get theoretical guarantees derived from our hierarchical model and discover the relationship between ASGMM and GMLASSO. Second, ASGMM is applied to finite number Gaussians, so the extension to infinite number will also be investigated. Finally, by seeking adaptive sparse estimation of precision matrices, our ASGMM consumes much more time when performing high dimensional data clustering. This is because the gradient optimization of ASGMM is calculated on elements of each precision matrix, while other methods use matrix-based optimization techniques. In this respect, a more effective and efficient optimizing method should be pursued.

Acknowledgments

This work is supported by the Natural Science Foundation of China under Grant Nos. 61033012, 611003177, and 61272371. Fundamental Research Funds for the Central Universities, No. DUT14QY16, Specialized Research Fund for the Doctoral Program of Higher Education, No. 20120041120046, and the Advanced Research Grant DEMOVE of European Research Council.

References

[1] E. Aune, D.P. Simpson, J. Eidsvik, Parameter estimation in high dimensional gaussian distributions, Stat. Comput. (2013) 1–17.
[2] O. Banerjee, L.E. Ghaoui, A. d'Aspremont, Convex optimization techniques for fitting sparse gaussian graphical models, in: Proc of the 23rd Int'l Conf. on Machine learning, ACM, 2006, pp. 89–96.
[3] P.J. Bickel, E. Levina, Covariance regularization by thresholding, Ann. Stat. (2008) 2577–2604.
[4] P.J. Bickel, E. Levina, Regularized estimation of large covariance matrices, Ann. Stat. (2008) 199–227.
[5] T.T. Cai, C.-H. Zhang, H.H. Zhou, et al., Optimal rates of convergence for covariance matrix estimation, Ann. Stat. 38 (4) (2010) 2118–2144.
[6] A. d'Aspremont, O. Banerjee, L. El Ghaoui, First-order methods for sparse covariance selection, SIAM J. Matrix Anal. Appl. 30 (1) (2008) 56–66.
[7] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the em algorithm, J. R. Stat. Soc. Ser. B (Methodological) (1977) 1–38.
[8] X. Deng, M. Yuan, Large gaussian covariance matrix estimation with markov structures, J. Comput. Graph. Stat. 18 (3) (2009).
[9] J. Duchi, S. Gould, D. Koller, Projected subgradient methods for learning sparse gaussians, 2012. Available from: <arXiv:1206.3249>.
[10] J. Fan, Y. Fan, J. Lv, High dimensional covariance matrix estimation using a factor model, J. Econ. 147 (1) (2008) 186–197.
[11] M.A. Figueiredo, Adaptive sparseness for supervised learning, IEEE Trans. Pattern Anal. Mach. Intell. 25 (9) (2003) 1150–1159.
[12] C. Fraley, E. Raftery, Model-based methods of classification: using the mclust software in chemometrics, J. Stat. Softw. 18 (6) (2007) 1–13.
[13] J. Friedman, T. Hastie, R. Tibshirani, Sparse inverse covariance estimation with the graphical lasso, Biostatistics 9 (3) (2008) 432–441.
[14] X. He, D. Cai, Y. Shao, H. Bao, J. Han, Laplacian regularized gaussian mixture model for data clustering, IEEE Trans. Knowledge Data Eng. 23 (9) (2011) 1406–1418.
[15] E. Karoui, Operator norm consistent estimation of large dimensional sparse covariance matrices, Ann. Stat. (2008) 2717–2756.
[16] C. Lam, J. Fan, Sparsistency and rates of convergence in large covariance matrix estimation, Ann. Stat. 37 (6B) (2009) 4254.
[17] K. Lange, J.S. Sinsheimer, Normal/independent distributions and their applications in robust regression, J. Comput. Graph. Stat. 2 (2) (1993) 175–198.
[18] E. Levina, A. Rothman, Sparse estimation of large covariance matrices via a nested lasso penalty, Ann. Appl. Stat. (2008) 245–263.
[19] N. Meinshausen, P. Bühlmann, High-dimensional graphs and variable selection with the lasso, Ann. Stat. 34 (3) (2006) 1436–1462.
[20] K.P. Murphy, Machine Learning: A Probabilistic Perspective, The MIT Press, 2012.

[21] P. Ravikumar, M.J. Wainwright, G. Raskutti, B. Yu, et al., High-dimensional covariance estimation by minimizing l1-penalized log-determinant divergence, Electron. J. Stat. 5 (2011) 935–980.
[22] A.J. Rothman, P.J. Bickel, E. Levina, J. Zhu, et al., Sparse permutation invariant covariance estimation, Electron. J. Stat. 2 (2008) 494–515.
[23] L. Ruan, M. Yuan, H. Zou, Regularized parameter estimation in high-dimensional gaussian mixture models, Neural Comput. 23 (6) (2011) 1605–1622.
[24] M. West, On scale mixtures of normal distributions, Biometrika 74 (3) (1986) 646–648.
[25] E. Wong, S. Awate, P.T. Fletcher, Adaptive sparsity in gaussian graphical models, in: Proceedings of the 30th International Conference on Machine Learning, vol. 28, 2013, pp. 311–319.
[26] W. Xu, X. Liu, Y. Gong, Document clustering based on non-negative matrix factorization, in: Proceedings of the 26th Annual Int'l SIGIR Conf. ACM, 2003, pp. 267–273.
[27] M. Yuan, High dimensional inverse covariance matrix estimation via linear programming, J. Mach. Learn. Res. 11 (2010) 2261–2286.
[28] M. Yuan, Y. Lin, Model selection and estimation in the gaussian graphical model, Biometrika 94 (1) (2007) 19–35.