Learning a Multivariate Gaussian Mixture Model with the Reversible Jump MCMC Algorithm

Zhihua Zhang

Department of Computer Science, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. Email: zhzhang@cs.ust.hk

Kap Luk Chan
Yiming Wu
Chibiao Chen

School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798

Summary. This paper is a contribution to the methodology of fully Bayesian inference in a multivariate Gaussian mixture model using the reversible jump Markov chain Monte Carlo algorithm. To follow the constraints of preserving the £rst two moments before and after the split or combine moves, we concentrate on a simpli£ed multivariate Gaussian mixture model, in which the covariance matrices of all components share a common eigenvector matrix. We then propose an approach to the construction of the reversible jump Markov chain Monte Carlo algorithm for this model. Experimental results on various data sets demonstrate the ef£cacy of our algorithm.

Keywords: Gaussian mixture model; Markov chain Monte Carlo; Bayesian inference; Model selection; Reversible jump methodology; Split and combine moves.

1. Introduction

The Gaussian (or normal) mixture model (GMM) (McLachlan and Peel, 2000; Titterington et al., 1985) is a ubiquitous statistical tool for density estimation due to its analytical tractability, asymptotic properties, and universal approximation ability to any continuous density function. It has also been used in pattern recognition widely (Banfield and Raftery, 1993; Bensmail and Celeux, 1996). Roughly speaking, each component in the GMM stands for a class that consists of patterns with some similarity. So, each class is implicitly assumed to follow a Gaussian distribution. For cases where this assumption is not fully met, a natural extension is to assume that each class is also a mixture of normally distributed subclasses (Hastie and Tibshirani, 1996). In this paper, we concentrate on the learning problem of GMM. In most settings, not only the parameters of a GMM, but also its structure, i.e., the number of components, are unknown *a priori*. So, we need to adopt some statistical procedures to infer the number of components. This may be treated as a model selection problem. Therefore, learning the GMM consists of two aspects: parameter estimation and model selection. A class of convenient learning algorithms has been proposed, such as the traditional expectation maximization (EM) algorithm (Dempster et al., 1977) and its variants (McLachlan and Krishnan, 1997), and the Markov chain Monte Carlo (MCMC) methods(Gilks et al., 1996). Recently, a so-called variational Bayes (VB) method has been proposed (Attial, 2000).

The main attraction of the EM algorithm is its simplicity. It estimates the parameters through the maximum likelihood (ML) criterion or in a maximum *a posteriori* (MAP) setting. The EM algorithm provides an alternative iterative procedure that converges to a local maximum in the space of GMM ranked by likelihood. A fundamental problem of the ML criterion is overfitting. In other words, the likelihood is a non-decreasing function of the number of components, so the ML cannot be used as a criterion for model selection. To cope with this problem, several model selection criteria have been proposed based on Bayesian approximation, such as the Laplace-empirical criterion and Schwarz's Bayesian inference criterion (BIC), and based on information theory, such as the AIC of Akaike's, the Rissanen's minimum description length (MDL), the minimum message length (MML), and the maximum entropy criterion. Generally, the EM algorithm and one of the model selection methods mentioned above are used separately for parameter estimation and model selection. On the other hand, the VB method can be itself used for model selection. However, in the VB framework, parameter estimation and model selection also work in two separate procedures. Although, Sato (2001) presented an on-line model selection method in the VB framework, this method is based on a data-driven technique.

The MCMC method provides a powerful scope for realistic statistical modelling and play a central role in modern Bayesian computation. Grenander and Miller (1994) proposed the jump-diffusion MCMC random sampling algorithm for varying-dimension problem. A novel alternative algorithm, termed the reversible jump MCMC (RJMCMC), has been elaborated in Green (1994, 1995). The RJMCMC algorithm is essentially a random sweep Metropolis-Hastings method. It constructs the dimension matching transform with a reversible jump methodology. The RJMCMC algorithm is attractive because it can deal with parameter estimation and model selection jointly in a single paradigm. The RJMCMC algorithm has been widely applied to statistical models (Richardson and Green, 1997; Robert et al., 2000), neural networks (Andrieu et al., 2001; Holmes and Mallick, 1998) and signal processing(Andrieu and Doucet, 1999; Larocque and Reilly, 2002).

In the case where the number of components is fixed, the MCMC method for parameter estimation of univariate GMM has been presented by Diebolt and Robert (1994) (for multivariate settings, see Robert (1996)). In the case where the number of components are unknown, a fully Bayesian methodology for the univariate GMM using the RJMCMC algorithm has been proposed by Richardson and Green (1997). However, applying the RJMCMC algorithm to fully Bayesian inference in the multivariate GMM is still an open problem. In the univariate setting, Richardson and Green (1997) developed the split-combine moves and the birth-death moves for the reversible jump. Compared to the birth-death moves, the split-combine moves are at the core of implementing the reversible jump methodology. One could also say that they are the main obstacle to the successful application of reversible jump methodology. The split or combine methods of Richardson and Green's (1997) are based on the constraints of preserving the first two moments before and after the split or combine moves. With these constraints, we can see that the combine move is a well-posed problem while the split move is an ill-posed one. In the multivariate setting, it is very difficult to resolve this ill-posed problem due to the need for introducing many free parameters and keeping the positive definiteness of the covariance matrix.

Richardson and Green (1997) felt that the constraints of preserving the first two moments before and after the combine or split moves might be too restrictive for the application of the reversible jump to the multivariate GMM, and they suggested to extend the birth-and-death moves to include non-empty components and then to dispense with the split-and-combine moves, or to define combine-and-split moves with respect to the underlying partition driven by the data rather than the parameters. At the same time, Richardson and Green (1997) thought these moves less efficient. Stephens (2000a) used the birth-death process instead of the reversible-jump methodology, and described an alternative MCMC, the birth-death MCMC (BDMCMC). However, unlike the reversible jump, the birth-death process did not utilize the information from the data or the parameters. So the convergence of BDMCMC is far slower than that of RJMCMC (Cappé et al., 2003).

In this paper, following the framework of Richardson and Green's (1997), we successfully develop a methodology of fully Bayesian inference in a multivariate GMM through the RJMCMC algorithm. The paper is organized as follows. In Section 2, we give the basic problem of learning the multivariate GMM with the RJMCMC algorithm. In Section 3, the Bayesian framework for a simplified multivariate GMM is defined. In Section 4, we present the detailed process of implementing the RJMCMC algorithm for the simplified GMM and focus our main attention on the design of the split-combine moves. In Section 5, we present some experiments to demonstrate the efficacy of our algorithm. Some concluding remarks are given in the final section.

2. Basic Formulation

Let $\mathcal{X} = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_K}\} \subset \mathbb{R}^p (p \geq 1)$ be K independent observations from a mixture model with $M > 1$ components:

$$p(\mathbf{x}) = \sum_{m=1}^{M} \pi_m p(\mathbf{x} \,|\boldsymbol{\theta}_m), \tag{1}$$

where $\pi_m \in (0,1)(m = 1,2,\ldots,M)$ are the mixing weights subject to the constraint $\sum\limits_{m=1}^{M} \pi_m = 1$ and the density $p(\cdot \,|\boldsymbol{\theta}_m)$ of the mth component is a given parametric family of densities indexed by a scalar or vector $\boldsymbol{\theta}_m$. For the GMM, the component density $p(\mathbf{x} \,|\boldsymbol{\theta}_m)$ is a normal probability distribution:

$$p(\mathbf{x} \,|\boldsymbol{\theta}_m) = \frac{1}{(2\pi)^{p/2} \det(\boldsymbol{\Sigma}_m)^{1/2}} \exp\left\{ -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_m)^T \boldsymbol{\Sigma}_m^{-1} (\mathbf{x} - \boldsymbol{\mu}_m) \right\},$$

where T denotes the transpose operation and $\boldsymbol{\theta}_m = (\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$ represent the parameters, the mean vector and the covariance matrix, of the mth component. We encapsulate these parameters into a parameter vector $\Theta = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_M\}$. Similarly, we use the notations $\Pi = \{\pi_1, \ldots, \pi_M\}$ and $\mathcal{Z} = \{z_1, z_2, \ldots, z_K\}$, where $z_k = m \in \{1, \ldots, M\}$ is missing data that indicates the observation of $\mathbf{x}_k$ arising from the mth component.

In a fully Bayesian method, the unknowns M, Π, and Θ are drawn from appropriate prior distributions. Along the lines similar to those of Richardson and Green (1997), we assume that the joint density of all variables mentioned takes the form

$$p(M, \Pi, \mathcal{Z}, \Theta, \mathcal{X}) = p(M) p(\Pi|M) p(\mathcal{Z} \,|\Pi, M)\, p(\Theta \,|M)\, p(\mathcal{X} \,|\Theta, \mathcal{Z}), \tag{2}$$

where the conditional independencies $p(\Theta \,|\mathcal{Z}, \Pi, M) = p(\Theta \,|M)$ and $p(\mathcal{X} \,|\Theta, \mathcal{Z}, \Pi, M) = p(\mathcal{X} \,|\Theta, \mathcal{Z})$ are imposed. Now, the goal of Bayesian inference is to generate realizations from the conditional joint density $p(M, \Pi, \Theta, \mathcal{Z} \,|\mathcal{X})$. Richardson and Green (1997) developed a methodology for the univariate GMM by using the RJMCMC algorithm, one sweep of which consists of six types of moves (Richardson and Green, 1997):

(a) update the mixing weights Π;
(b) update the parameters Θ;
(c) update the allocation $\mathcal{Z}$;
(d) update the hyperparameters;
(e) split one component into two, or combine two into one;
(f) the birth or death of an empty component.

Move type (c) is used for missing data, while move types (a), (b) and (d) are used for parameter estimation via the Gibbs sampling. Since types (e) and (f) involve changing the number of component, M, by 1, they constitute the reversible jump and are used for model selection via the Metropolis-Hastings algorithm. Assume that a move of type m, from $\mathbf{s}$ to a point $\mathbf{s}'$ in a higher dimensional space, is presented. Usually, it is implemented by drawing a vector of continuous random variables $\mathbf{v}$, independent of $\mathbf{s}$, and obtaining $\mathbf{s}'$ with an invertible deterministic function $f(\mathbf{s}, \mathbf{v})$. Then the acceptance probabilities from $\mathbf{s}$ to $\mathbf{s}'$ and from $\mathbf{s}'$ to $\mathbf{s}$ are $\min\{1, R\}$ and $\min\{1, R^{-1}\}$, where

$$R = \frac{p\left(\mathbf{s}'|\mathbf{x}\right) r_m(\mathbf{s}')}{p\left(\mathbf{s}|\mathbf{x}\right) r_m(\mathbf{s}) q(\mathbf{v})} \left|\frac{\partial \mathbf{s}'}{\partial(\mathbf{s}, \mathbf{v})}\right|, \tag{3}$$

where $r_m(\mathbf{s})$ is the probability of choosing move m in state $\mathbf{s}$, $q(\mathbf{v})$ is the density function of $\mathbf{v}$, and $\left|\frac{\partial \mathbf{s}'}{\partial(\mathbf{s}, \mathbf{v})}\right|$ is the Jacobian arising from the change of variables from $(\mathbf{s}, \mathbf{v})$ to $\mathbf{s}'$.

In the reversible jump methodology proposed by Richardson and Green (1997), the birth-and-death moves were used for empty components contained in the mixture, while the split and combine moves were used for non-empty components. Therefore the birth and death moves are supplements to the split and combine moves.

Here and later, we assume that the ith and jth components are merged as the i'th component, and the kth component is split into the j'th and k'th components. In the univariate setting, Richardson and Green (1997) proposed the constraints of preserving the first two moments before and after the combine and split moves. In the multivariate setting, the constraints show that the mean vector and the covariance matrix of the i'th component satisfy

$$\pi_i + \pi_j = \pi_{i'}, \tag{4}$$

$$\pi_i \boldsymbol{\mu}_i + \pi_j \boldsymbol{\mu}_j = \pi_{i'} \boldsymbol{\mu}_{i'}, \tag{5}$$

$$\pi_i(\boldsymbol{\Sigma}_i + \boldsymbol{\mu}_i \boldsymbol{\mu}_i^T) + \pi_j(\boldsymbol{\Sigma}_j + \boldsymbol{\mu}_j \boldsymbol{\mu}_j^T) = \pi_{i'}(\boldsymbol{\Sigma}_{i'} + \boldsymbol{\mu}_{i'} \boldsymbol{\mu}_{i'}^T). \tag{6}$$

On the other hand, because the split is the exact inverse procedure of the combine, the corresponding mixing weights, the mean vectors and the covariance matrices should satisfy the following split equations:

$$\pi_k = \pi_{j'} + \pi_{k'}, \tag{7}$$

$$\pi_k \boldsymbol{\mu}_k = \pi_{j'} \boldsymbol{\mu}_{j'} + \pi_{k'} \boldsymbol{\mu}_{k'}, \tag{8}$$

$$\pi_k(\boldsymbol{\Sigma}_k + \boldsymbol{\mu}_k \boldsymbol{\mu}_k^T) = \pi_{j'}(\boldsymbol{\Sigma}_{j'} + \boldsymbol{\mu}_{j'} \boldsymbol{\mu}_{j'}{}^T) + \pi_{k'}(\boldsymbol{\Sigma}_{k'} + \boldsymbol{\mu}_{k'} \boldsymbol{\mu}_{k'}{}^T). \tag{9}$$

Clearly, the combine move is a well-posed problem. However, the split move is an ill-posed problem because the number of equations is less than the number of unknowns. In the multivariate setting, it is very difficult to resolve this ill-posed problem due to the need for constructing many free parameters and keeping the positive definiteness of the covariance matrix. In this paper, we develop split-and-combine moves following the above constraints and extend the RJMCMC method to a simplified multivariate GMM.

3. Bayesian Framework for a Simpli£ed Multivariate GMM

By using the singular value decomposition (Golub and Loan, 1996), the covariance matrix, $\mathbf{\Sigma}_m$, of the components in (1) can be decomposed into

$$\mathbf{\Sigma}_m = \mathbf{A}_m \mathbf{\Lambda}_m \mathbf{A}_m^T \qquad \text{for } m = 1, \ldots, M,$$

where $\mathbf{A}_m = \left(\mathbf{a}_1^{(m)}, \ldots, \mathbf{a}_p^{(m)}\right)$ is an orthogonal matrix and $\mathbf{\Lambda}_m = \operatorname{diag}\{\lambda_{m1}, \ldots, \lambda_{mp}\}$ a is diagonal matrix with positive elements. For convenience, we refer to the matrices $\mathbf{A}_m$ and $\mathbf{\Lambda}_m$ as eigenvector and eigenvalue matrices, respectively. We shall use the notation λ_{mi} to denote the ith largest eigenvalue of $\mathbf{\Sigma}_m$. Banfield and Raftery (1993) proposed the model-based clustering methods in terms of the above decomposition of the covariance matrices. Having reformulated $\mathbf{\Lambda}_m = \lambda_{m1}\mathbf{D}_m$, Bensmail and Celeux (1996) developed eight schemes according to different combinations of λ_{m1}, $\mathbf{A}_m$, and $\mathbf{D}_m$, and Bensmail et al. (1997) then presented the Bayesian parameter estimation via Gibbs sampling for each scheme.

In this paper, we consider a GMM in which all the covariance matrices have the same eigenvector matrix. That is, $\mathbf{A}_1 = \mathbf{A}_2 = \cdots = \mathbf{A}_M = \mathbf{A}$, where $\mathbf{A} = (\mathbf{a}_1, \ldots, \mathbf{a}_p) \in \mathbb{R}^{p \times p}$ is orthogonal. Employing the notations by Bensmail and Celeux (1996), we denote this model as $[\mathbf{A}\mathbf{\Lambda}_m\mathbf{A}^T]$. This simplified GMM is one of the most representative GMMs. This point has been also mentioned by Bensmail and Celeux (1996), and Bensmail et al. (1997). Our goal here is to reduce the free parameters arisen during the split move because we know that an $p \times p$ symmetric positive definite matrix has $\frac{p(p+1)}{2}$ degrees of freedom, whereas its eigenvalue matrix has only p degrees of freedom. Our paper addresses the fully Bayesian analysis of this simplified GMM through the RJMCMC algorithm.

It is necessary to choose a proper prior distribution for each parameter in Bayesian inference. For the number of components, M, we assume that M is uniform on $\{1, 2, \ldots, M_{\max}\}$, where $M_{\max}$ is pre-specified. The conjugate prior on Π will always be taken as symmetric Dirichlet distribution

$$(\pi_1, \pi_2, \ldots, \pi_M) \sim \mathcal{D}(\delta, \delta, \ldots, \delta),$$

and the conjugate prior distributions of $\boldsymbol{\mu}_m$ and $\mathbf{\Sigma}_m^{-1}$ are normal and the Wishart distributions, respectively,

$$\boldsymbol{\mu}_m | \mathbf{\Sigma}_m \sim \mathcal{N}(\boldsymbol{\xi}_m, \tau_m^{-1}\mathbf{\Sigma}_m) \quad \text{and} \quad \mathbf{\Sigma}_m^{-1} \sim \mathcal{W}(r_m, \mathbf{W}_m),$$

where the symbol "|" denotes generic conditional dependence. For our model $[\mathbf{A}\mathbf{\Lambda}_m\mathbf{A}^T]$, we define the prior distribution of the parameter λ_{mn}^{-1} as Gamma distribution

$$\lambda_{mn}^{-1} \sim \mathcal{G}(r_m/2, l_{mn}^{-1}/2) \quad (m = 1, \ldots, M)(n = 1, \ldots, p).$$

We fix the value of the hyperparameter r_m, and the other hyperparameters $\boldsymbol{\xi}_m, \tau_m$ and l_{mn} are drawn from normal or gamma priors:

$$\begin{aligned} p(\boldsymbol{\xi}_m) &\sim \mathcal{N}(\boldsymbol{\nu}, \rho^2\mathbf{I}), \\ p(\tau_m) &\sim \mathcal{G}(1/2, \rho^{-2}/2), \\ p(l_{mn}^{-1}) &\sim \mathcal{G}(1/2, \zeta_n/2) \quad (m = 1, \ldots, M)(n = 1, \ldots, p). \end{aligned}$$

The prior distribution of the eigenvector matrix $\mathbf{A}$ is very delicate to specify due to its orthogonality. Bensmail et al. (1997) defined its prior as $\mathbf{A} \sim \mathcal{W}(r, \mathbf{I})$. However, it cannot be proven that a matrix distributed according to $\mathcal{W}(r, \mathbf{I})$ is orthogonal, so it cannot ensure that $\mathbf{A}$ obtained from its full condition is orthogonal. For simplicity, we use a nonparametric estimation method for $\mathbf{A}$ instead of a random sampling. Specifically, we first calculate the sample mean vector and covariance matrix of K observations $\mathbf{x}_1, \ldots, \mathbf{x}_K$:

$$\bar{\mathbf{x}} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{x}_k, \qquad \mathbf{S} = \frac{1}{K}\sum_{k=1}^{K}(\mathbf{x}_k - \bar{\mathbf{x}})(\mathbf{x}_k - \bar{\mathbf{x}})^{\mathrm{T}},$$

respectively, and then let the eigenvector matrix of $\mathbf{S}$ be $\mathbf{A}$.

4. The RJMCMC Algorithm for the Simpli£ed Multivariate GMM

One sweep of our reversible jump MCMC algorithm for the mixture $[\mathbf{A}\boldsymbol{\Lambda}_m\mathbf{A}^T]$ also consists of six move types as described in Section 2. We present the details of these move types.

4.1. Gibbs Moves

The move types (a)-(d) are Gibbs moves. Following Robert (1996), or Diebolt and Robert (1994), we present the details of move types (a)-(d).

Move type (a): Simulate the mixing weights Π from the full conditional

$$(\pi_1, \ldots, \pi_M)| \cdots \sim \mathcal{D}(\delta + n_1, \delta + n_2, \ldots, \delta + n_M),$$

where n_m is the number of observations allocated to the mth component. Here and later we use '$|\cdots$' to denote conditioning on all other variables.

Move type (b): Simulate $\boldsymbol{\mu}_m$ and λ_{mn} from the full conditionals

$$\begin{aligned}\boldsymbol{\mu}_m|\cdots &\sim \mathcal{N}\Big(\bar{\boldsymbol{\xi}}_m, \frac{1}{n_m+\tau_m}\boldsymbol{\Sigma}_m\Big),\\ \lambda_{mn}^{-1}|\cdots &\sim \mathcal{G}\Big(\frac{r_m+n_m+1}{2}, \frac{l_{mj}^{-1} + \mathbf{a}_n^T\{\tau_m(\boldsymbol{\mu}_m - \boldsymbol{\xi}_m)(\boldsymbol{\mu}_m - \boldsymbol{\xi}_m)^T + \mathbf{S}_m\}\mathbf{a}_n}{2}\Big),\end{aligned}$$

where

$$\bar{\boldsymbol{\xi}}_m = \frac{\tau_m\boldsymbol{\xi}_m + n_m\bar{\mathbf{x}}_m}{\tau_m + n_m}, \quad \bar{\mathbf{x}}_m = \frac{1}{n_m}\sum_{k;z_k=m}\mathbf{x}_k, \quad \mathbf{S}_m = \sum_{k;z_k=m}(\mathbf{x}_k - \boldsymbol{\mu}_m)(\mathbf{x}_k - \boldsymbol{\mu}_m)^T.$$

Move type (c): Derive the missing data z_k $(1 \le k \le K)$ from a standard uniform random variable u_k, $z_k = i$ if $p_{k1} + \cdots + p_{k(i-1)} < u_k \le p_{k1} + \cdots + p_{ki}$, where

$$p_{ki} \propto \pi_i|\boldsymbol{\Sigma}_i|^{-\frac{1}{2}}\exp\Big\{-\frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}_k - \boldsymbol{\mu}_i)\Big\}.$$

Move type (d): Update the hyperparameters $\boldsymbol{\xi}_m$, τ_m and l_{mj};

$$\begin{aligned}\boldsymbol{\xi}_m|\cdots &\sim \mathcal{N}\Big(\widehat{\boldsymbol{\Sigma}}_m(\rho^{-2}\boldsymbol{\nu} + \tau_m\boldsymbol{\Sigma}_m^{-1}\boldsymbol{\mu}_m), \widehat{\boldsymbol{\Sigma}}_m\Big),\\ \tau_m|\cdots &\sim \mathcal{G}\Big(\frac{p+1}{2}, \frac{\rho^{-2} + (\boldsymbol{\mu}_m - \boldsymbol{\xi}_m)^T\boldsymbol{\Sigma}_m^{-1}(\boldsymbol{\mu}_m - \boldsymbol{\xi}_m)}{2}\Big),\\ l_{mj}^{-1}|\cdots &\sim \mathcal{G}\Big(\frac{1+r_m}{2}, \frac{\lambda_{mj}^{-1} + \zeta_j}{2}\Big),\end{aligned}$$

where $\widehat{\boldsymbol{\Sigma}}_m^{-1} = \rho^{-2}\mathbf{I} + \tau_m\boldsymbol{\Sigma}_m^{-1}$.

4.2. Split and Combine Moves

In this section, we discuss the split-and-combine moves for the mixture $[\mathbf{A}\mathbf{\Lambda}_m\mathbf{A}^T]$. The split-combine moves consist of constructing the split-combine methods and calculating the Jacobian determinant. We first construct the split and combine methods by solving the split and combine equations defined in Section 2, and then calculate the value of the Jacobian determinant of the transformation involved in the split-combine moves.

4.2.1. Combine and Split Methods

Intuitively, it is straightforward to obtain the mixing weight $\pi_{i'}$, the mean vector $\boldsymbol{\mu}_{i'}$ and the covariance matrix $\mathbf{\Sigma}_{i'}$ from (4), (5) and (6). For our GMM in question, since the covariance matrices of all components share a common eigenvector matrix $\mathbf{A}$, this implies that the eigenvector matrix of the covariance matrix $\mathbf{\Sigma}_{i'}$ must be $\mathbf{A}$. From (5) and (6), we have

$$\mathbf{\Sigma}_{i'} = \frac{1}{\pi_{i'}}\mathbf{A}^T\left(\pi_i\mathbf{\Lambda}_i + \pi_j\mathbf{\Lambda}_j\right)\mathbf{A} + \frac{\pi_i\pi_j}{\pi_{i'}^2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T. \tag{10}$$

This equation shows that the eigenvector matrix of $\mathbf{\Sigma}_{i'}$ does not necessarily equal $\mathbf{A}$. So, this prevents us from directly obtaining $\mathbf{\Sigma}_{i'}$ by (10). However, we only need to know the eigenvalues $\lambda_{i'n}(n = 1, \ldots, p)$ of $\mathbf{\Sigma}_{i'}$, other than $\mathbf{\Sigma}_{i'}$ itself. From (10), we have:

$$\mathrm{tr}(\mathbf{\Sigma}_{i'}) = \frac{\pi_i}{\pi_{i'}}\mathrm{tr}(\mathbf{\Sigma}_i) + \frac{\pi_j}{\pi_{i'}}\mathrm{tr}(\mathbf{\Sigma}_j) + \frac{\pi_i\pi_j}{\pi_{i'}^2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j),$$

where the symbol “tr” represents the trace of a matrix. Thus,

$$\sum_{n=1}^{p}\lambda_{i'n} = \sum_{n=1}^{p}\left(\frac{\pi_i}{\pi_{i'}}\lambda_{in} + \frac{\pi_j}{\pi_{i'}}\lambda_{jn}\right) + \frac{\pi_i\pi_j}{\pi_{i'}^2}\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|^2. \tag{11}$$

Here we refer to (4), (5) and (11) as a new set of the combine equations. Applying the theorem 8.1.8 in Golub and Loan (1996) to equation (10), we have

$$\lambda_{i'n} = \frac{\pi_i}{\pi_{i'}}\lambda_{in} + \frac{\pi_j}{\pi_{i'}}\lambda_{jn} + q_n\frac{\pi_i\pi_j}{\pi_{i'}^2}\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|^2, \quad n = 1, \ldots, p$$

with $q_1 + \cdots + q_p = 1$. An intuitive approach to choosing q_n's is to set

$$q_n = \frac{(\mu_{in} - \mu_{jn})^2}{\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|^2}, \quad n = 1, \ldots, p.$$

The resultant $\lambda_{i'n}$'s obviously satisfy the combine equation (11).

On the other hand, since the split moves is the exact inverse procedure of the combine moves, the corresponding mixing weights, the mean vectors and the covariance matrices should satisfy (7), (8) and the following equation:

$$\sum_{n=1}^{p}\lambda_{kn} = \sum_{n=1}^{p}\left(\frac{\pi_{j'}}{\pi_k}\lambda_{j'n} + \frac{\pi_{k'}}{\pi_k}\lambda_{k'n}\right) + \frac{\pi_{j'}\pi_{k'}}{\pi_k^2}\|\boldsymbol{\mu}_{j'} - \boldsymbol{\mu}_{k'}\|^2. \tag{12}$$

Clearly, resolving the split equations (7), (8) and (12) is an ill-posed problem. Here we construct a set of solutions for this ill-posed problem as follows:

$$\begin{aligned}
\pi_{j'} &= \pi_k \alpha, \quad \pi_{k'} = \pi_k(1-\alpha), && (13)\\
\boldsymbol{\mu}_{j'} &= \boldsymbol{\mu}_k - \sqrt{\frac{\pi_{k'}}{\pi_{j'}}} \sum_{n=1}^{p} \lambda_{kn}^{\frac{1}{2}} u_n \mathbf{a}_n, \\
\boldsymbol{\mu}_{k'} &= \boldsymbol{\mu}_k + \sqrt{\frac{\pi_{j'}}{\pi_{k'}}} \sum_{n=1}^{p} \lambda_{kn}^{\frac{1}{2}} u_n \mathbf{a}_n, && (14)\\
\lambda_{j'n} &= \beta_n (1-u_n^2) \frac{\pi_k}{\pi_{j'}} \lambda_{kn}, \\
\lambda_{k'n} &= (1-\beta_n)(1-u_n^2) \frac{\pi_k}{\pi_{k'}} \lambda_{kn} \quad \text{for} \quad n = 1, 2, \ldots, p, && (15)
\end{aligned}$$

where α is randomly sampled from the Beta distribution $Be(1,1)$, β_n's are randomly sampled from $Be(1,1)$, and the value of u_n is sampled from $Be(2,2)$ and the its sign is equally likely to be either positive or negative. It is easy to find that the $\pi_{j'}$, $\boldsymbol{\mu}_{j'}$, $\lambda_{j'i}$, $\pi_{k'}$, $\boldsymbol{\mu}_{k'}$ and $\lambda_{k'i}$, determined above, satisfy (7), (8) and (12).

4.2.2. *The Acceptance Probabilities for Split and Combine Moves*

Now, we calculate the acceptance probability of split and combine moves: $\min\{1, R\}$ and $\min\{1, R^{-1}\}$. From (3), we have

$$R = \frac{p(M+1, \Pi, \mathcal{Z}, \Theta | \mathcal{X}) d_{M+1}}{p(M, \Pi, \mathcal{Z}, \Theta | \mathcal{X}) b_M P_{\text{alloc}} q(\cdot)} \left| \det(\mathbf{J}) \right|,$$

where $q(\cdot) = p(\alpha)p(\beta_1, \ldots, \beta_p)p(u_1, \ldots, u_p)$, P_{alloc} is the probability of making this particular allocation, b_M and d_M are respectively the probabilities of choosing split and combine moves, and

$$\begin{aligned}
\frac{p(M+1, \Pi, \mathcal{Z}, \Theta | \mathcal{X})}{p(M, \Pi, \mathcal{Z}, \Theta | \mathcal{X})} &= (\text{likelihood ratio}) \frac{p(M+1)}{p(M)} (M+1) \frac{\pi_{j'}^{\delta-1+n_{j'}} \pi_{k'}^{\delta-1+n_{k'}}}{\pi_k^{\delta-1+n_{j'}+n_{k'}} B(\delta, M\delta)} \\
&\times \frac{(2\pi)^{-p/2} \left| \tau_k^{-1} \boldsymbol{\Sigma}_k \right|^{1/2}}{\left| \tau_{j'}^{-1} \tau_{k'}^{-1} \boldsymbol{\Sigma}_{j'} \boldsymbol{\Sigma}_{k'} \right|^{1/2}} \exp\Big[-\frac{1}{2} \Big\{ (\boldsymbol{\mu}_{j'} - \boldsymbol{\xi}_{j'})^T \tau_{j'} \boldsymbol{\Sigma}_{j'}^{-1} (\boldsymbol{\mu}_{j'} - \boldsymbol{\xi}_{j'}) \\
&\quad + (\boldsymbol{\mu}_{k'} - \boldsymbol{\xi}_{k'})^T \tau_{k'} \boldsymbol{\Sigma}_{k'}^{-1} (\boldsymbol{\mu}_{k'} - \boldsymbol{\xi}_{k'}) - (\boldsymbol{\mu}_k - \boldsymbol{\xi}_k)^T \tau_k \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{\mu}_k - \boldsymbol{\xi}_k) \Big\} \Big] \\
&\times \left(\frac{\Gamma(r_k/2)}{\Gamma(r_{j'}/2)\Gamma(r_{k'}/2)} \right)^p \prod_{n=1}^{p} \frac{\lambda_{kn} \left(\frac{\lambda_{j'n} l_{j'n}}{2} \right)^{-r_{j'}/2} \left(\frac{\lambda_{k'n} l_{k'n}}{2} \right)^{-r_{k'}/2}}{\lambda_{j'n} \lambda_{k'n} \left(\frac{\lambda_{kn} l_{kn}}{2} \right)^{-r_k/2}} \\
&\quad \exp\left\{ -\frac{1}{2} \sum_{n=1}^{p} \left[\lambda_{j'n}^{-1} l_{j'n}^{-1} + \lambda_{k'n}^{-1} l_{k'n}^{-1} - \lambda_{kn}^{-1} l_{kn}^{-1} \right] \right\},
\end{aligned}$$

where $B(\cdot, \cdot)$ is the Beta function.

From the transformation defined by (13), (14) and (15), we can obtain the Jacobian matrix J as

$$\begin{bmatrix} \alpha & \pi_k & \mathbf{0}_{1\times p} & \mathbf{0}_{1\times p} & \mathbf{0}_{1\times p} & \mathbf{0}_{1\times p} \\ (1-\alpha) & -\pi_k & \mathbf{0}_{1\times p} & \mathbf{0}_{1\times p} & \mathbf{0}_{1\times p} & \mathbf{0}_{1\times p} \\ \mathbf{0}_{p\times 1} & \mathbf{0}_{p\times 1} & \mathbf{I} & -\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{\frac{1}{2}} & -\frac{1}{2}\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{-\frac{1}{2}}\mathbf{U} & \mathbf{0}_{p\times p} \\ \mathbf{0}_{p\times 1} & \mathbf{0}_{p\times 1} & I & \sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{\frac{1}{2}} & \frac{1}{2}\sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{-\frac{1}{2}}\mathbf{U} & \mathbf{0}_{p\times p} \\ \mathbf{0}_{p\times 1} & \mathbf{0}_{p\times 1} & \mathbf{0}_{p\times p} & -2\frac{\pi_k}{\pi_{j'}}\boldsymbol{\Lambda}_k\mathbf{B}\mathbf{U} & \frac{\pi_k}{\pi_{j'}}\mathbf{B}(\mathbf{I}-\mathbf{U}^2) & \frac{\pi_k}{\pi_{j'}}\boldsymbol{\Lambda}_k(\mathbf{I}-\mathbf{U}^2) \\ \mathbf{0}_{p\times 1} & \mathbf{0}_{p\times 1} & \mathbf{0}_{p\times p} & 2\frac{\pi_k}{\pi_{k'}}\boldsymbol{\Lambda}_k(\mathbf{B}-\mathbf{I})\mathbf{U} & \frac{\pi_k}{\pi_{k'}}(\mathbf{I}-\mathbf{B})(\mathbf{I}-\mathbf{U}^2) & \frac{\pi_k}{\pi_{k'}}\boldsymbol{\Lambda}_k(\mathbf{U}^2-\mathbf{I}) \end{bmatrix}, \tag{16}$$

where $\mathbf{0}_{p\times 1}$ is the $p \times 1$ zero vector, $\mathbf{0}_{1\times p}$ is the $1 \times p$ zero vector, $\mathbf{0}_{p\times p}$ is the $p \times p$ zero matrix, and $\mathbf{U} = \text{diag}\{u_1, u_2, \ldots, u_p\}$ and $\mathbf{B} = \text{diag}\{\beta_1, \beta_2, \ldots, \beta_p\}$ are diagonal. The absolute value of the determinant $\det(\mathbf{J})$ is

$$|\det(\mathbf{J})| = \frac{\pi_k^{3p+1}}{(\pi_{j'}\pi_{k'})^{\frac{3p}{2}}} \prod_{n=1}^{p} \lambda_{kn}^{\frac{3}{2}}(1-u_n^2). \tag{17}$$

The detailed derivations of the Jacobian matrix J and its determinant are given in Appendices A and B, respectively.

Note that in the case of one-dimensional setting, the mean vector and covariance matrix degenerate to scalars, respectively, i.e., $p = 1$, $\lambda_k = \sigma_k^2$, and $u \sim Be(2, 2)$. It is easy to see that (14) and (15) degenerate to the split method of Richardson and Green's (1997). Moreover, (17) reduces to the equation given by Richardson and Green's (1997).

4.3. Birth and Death Moves

Our birth and death moves can be straightforwardly obtained from the ones in one dimensional setting of Richardson and Green (1997). We first make a random choice between birth and death with the same probabilities b_M and d_M as above. For a birth, a mixing weight and parameters of the new component proposed are drawn using

$$\begin{aligned} \pi_{m*} &\sim Be(1, M), \quad \boldsymbol{\mu}_{m*}|\boldsymbol{\Sigma}_{m*} \sim \mathcal{N}\left(\boldsymbol{\xi}_{m*}, \tau_{m*}^{-1}\boldsymbol{\Sigma}_{m*}\right), \\ \lambda_{m*n}^{-1} &\sim \mathcal{G}\left(\frac{r_{m*}}{2}, \frac{l_{m*n}^{-1}}{2}\right) \quad (n = 1, \ldots, p). \end{aligned}$$

The acceptance probabilities for birth and death are $\min\{1, R\}$ and $\min\{1, R^{-1}\}$ respectively (Richardson and Green, 1998), where

$$\begin{aligned} R &= \frac{p(M+1)}{p(M)} \frac{1}{B(\delta, M\delta)} \pi_{m*}^{\delta-1}(1-\pi_{m*})^{K+M\delta-M}(M+1) \\ &\quad \times \frac{d_{M+1}}{(M_0+1)b_M} \frac{1}{g_{1,M}(\pi_{m*})}(1-\pi_{m*})^{M-1}. \end{aligned}$$

Here M is the number of components and M_0 the number of empty components, before the birth.

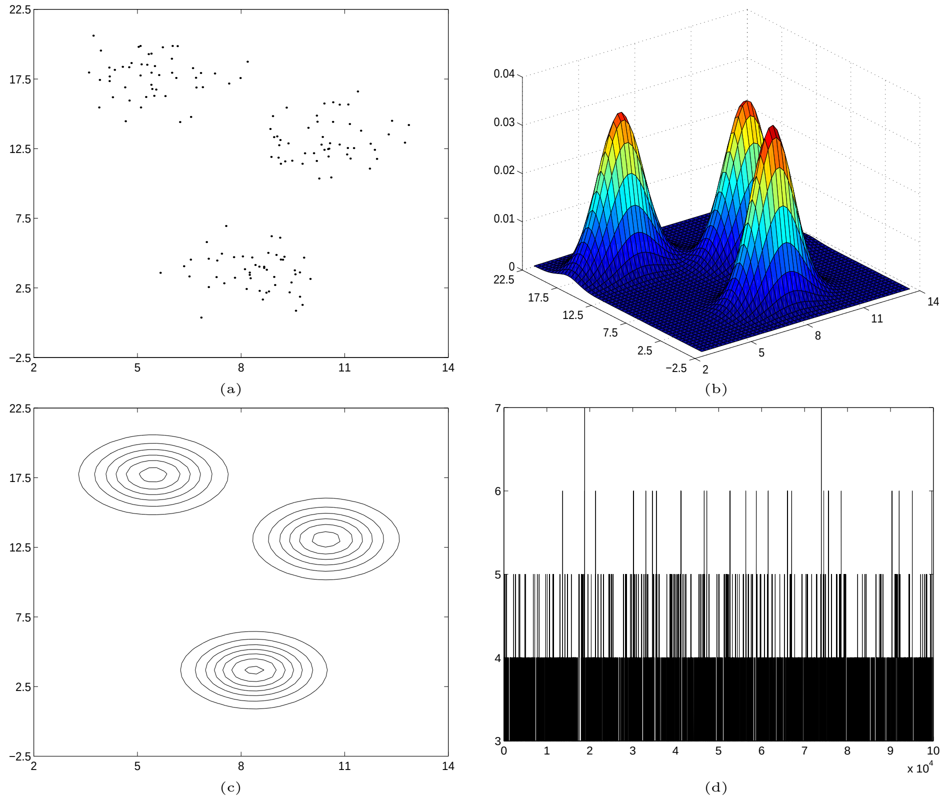

Fig. 1. Dataset 1: (a) original data points; (b) overall predictive density; (c) the contour of overall predictive density; (d) the number of components vs. the number of sweeps.

5. Experimental Results

In this section, we present our experiments on several data sets to demonstrate the efficacy of our proposed algorithm. These data sets have been analyzed with the hierarchical mixtures defined in Section 3, which slightly differ from those of Richardson and Green (1997). That is, we add an extra layer of priors to infer the mean vectors and use a different prior hyperparameter for the covariance matrix of each component. Referring to the sensitivity analysis of posterior distribution with respect to prior assumptions and the suggestions of selecting hyperparameters by Richardson and Green (1997), we set the hyperparameters: $r_m = 4$, $\delta = 1$, $\boldsymbol{\nu} = \frac{1}{K}\sum_{k=1}^{K}\mathbf{x}_k$, $\rho^2 = \frac{1}{K}\sum_{k=1}^{K}\|\mathbf{x}_k - \boldsymbol{\nu}\|^2$, $\zeta_n = \frac{1}{K}\sum_{k=1}^{K}(x_{kn} - \nu_n)^2$, and $M_{\max} = 32$, in the following experiments.

For convenience of interpretation and visualization, the first experiment uses three sets of two-dimensional data, each of which is generated from a bivariate GMM (Table 1). These three datasets and their corresponding experimental results are shown in Figures 1-3, respectively. For each of the three data sets, we ran our algorithms for 200,000 sweeps. The first 100,000 sweeps are considered as burn-in, and will be discarded. All the inferences are based on the next 100,000 sweeps. From Figures 1- 3, we can see that all the runs are mixing quite well, therefore 200,000 sweeps should be enough.

Table 1. The £ve multivariate GMMs used in the experiments.

GMMs	$\boldsymbol{\mu}$	$\boldsymbol{\Sigma}$	K
1	$\boldsymbol{\mu}_1 = (17.5,\ 5.5)^T$	$\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}_3 = \begin{bmatrix} 1.2 & 0 \\ 0 & 1.5 \end{bmatrix}$	50
	$\boldsymbol{\mu}_2 = (12.5,\ 11)^T$		50
	$\boldsymbol{\mu}_3 = (\ 3,\ \ 8)^T$		50
2	$\boldsymbol{\mu}_1 = (22,\ 17)^T$	$\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}_3 = \begin{bmatrix} 1.2 & 0 \\ 0 & 1.5 \end{bmatrix}$	50
	$\boldsymbol{\mu}_2 = (8.5,\ 11.5)^T$		50
	$\boldsymbol{\mu}_3 = (5.5,\ 13)^T$		50
3	$\boldsymbol{\mu}_1 = (12,\ 11)^T$	$\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}_3 = \begin{bmatrix} 1.2 & 0 \\ 0 & 1.5 \end{bmatrix}$	50
	$\boldsymbol{\mu}_2 = (11,\ 13.5)^T$		50
	$\boldsymbol{\mu}_3 = (\ 8,\ 11.5)^T$		50
4	$\boldsymbol{\mu}_1 = (-20,\ 3,\ -20,\ 20,\ -10,\ 5,\ 11,\ 15)^T$	$\boldsymbol{\Sigma}_1 = \mathrm{diag}(4,\ 7,\ 2,\ 3,\ 6,\ 2,\ 2,\ 3)$	100
	$\boldsymbol{\mu}_2 = (-15,\ -2,\ 10,\ -15,\ 0,\ -8,\ 8,\ 12)^T$	$\boldsymbol{\Sigma}_2 = \mathrm{diag}(10,\ 4,\ 2,\ 1,\ 4,\ 6,\ 2,\ 2)$	50
	$\boldsymbol{\mu}_3 = (-14,\ 14,\ -10,\ 10,\ -6,\ -10,\ 9,\ 5)^T$	$\boldsymbol{\Sigma}_3 = \mathrm{diag}(7,\ 3,\ 5,\ 3,\ 5,\ 6,\ 7,\ 2)$	50
	$\boldsymbol{\mu}_4 = (-10,\ -7,\ 0,\ -10,\ 3,\ -5,\ 3,\ 5)^T$	$\boldsymbol{\Sigma}_4 = \mathrm{diag}(3,\ 7,\ 4,\ 2,\ 2,\ 1,\ 5,\ 3)$	100
	$\boldsymbol{\mu}_5 = (10,\ -12,\ 5,\ 0,\ 6,\ 8,\ -3,\ 5)^T$	$\boldsymbol{\Sigma}_5 = \mathrm{diag}(10,\ 3,\ 3,\ 2,\ 3,\ 2,\ 9,\ 1)$	50
	$\boldsymbol{\mu}_6 = (15,\ -4,\ -15,\ 15,\ 8,\ 20,\ 1,\ 12)^T$	$\boldsymbol{\Sigma}_6 = \mathrm{diag}(5,\ 6,\ 1,\ 8,\ 1,\ 5,\ 6,\ 2)$	100
	$\boldsymbol{\mu}_7 = (24,\ 21,\ -5,\ 5,\ -3,\ 10,\ -9,\ -3)^T$	$\boldsymbol{\Sigma}_7 = \mathrm{diag}(5,\ 2,\ 4,\ 3,\ 1,\ 4,\ 3,\ 2)$	100
	$\boldsymbol{\mu}_8 = (35,\ -22,\ 30,\ -25,\ 10,\ -8,\ 18,\ 3)^T$	$\boldsymbol{\Sigma}_8 = \mathrm{diag}(10,\ 4,\ 2,\ 1,\ 14,\ 6,\ 1,\ 2)$	50
	$\boldsymbol{\mu}_9 = (40,\ -30,\ 35,\ 5,\ -22,\ 10,\ 22,\ -1)^T$	$\boldsymbol{\Sigma}_9 = \mathrm{diag}(2,\ 1,\ 2,\ 13,\ 4,\ 13,\ 5,\ 1)$	100
	$\boldsymbol{\mu}_{10} = (60,\ -9,\ 15,\ -5,\ -12,\ 0,\ 12,\ -2)^T$	$\boldsymbol{\Sigma}_{10} = \mathrm{diag}(2,\ 8,\ 2,\ 3,\ 4,\ 3,\ 5,\ 12)$	100
5	$\boldsymbol{\mu}_1^T = (-1,\ 5,\ -3,\ 4)$	$\boldsymbol{\Sigma}_1 = \begin{bmatrix} 1.0 & 0.2 & 0.5 & 0.8 \\ 0.2 & 1.4 & 0.3 & 0.6 \\ 0.5 & 0.3 & 1.2 & 0.3 \\ 0.8 & 0.6 & 0.3 & 1.8 \end{bmatrix}$	100
	$\boldsymbol{\mu}_2^T = (4,\ 10,\ 6,\ 7)$	$\boldsymbol{\Sigma}_2 = \begin{bmatrix} 1.4 & 0.6 & 0.4 & 0.2 \\ 0.6 & 0.7 & 0.6 & 0.6 \\ 0.4 & 0.6 & 2.3 & 0.6 \\ 0.2 & 0.6 & 0.6 & 0.8 \end{bmatrix}$	75
	$\boldsymbol{\mu}_3^T = (7,\ 15,\ 11,\ 10)$	$\boldsymbol{\Sigma}_3 = \begin{bmatrix} 1.3 & 0.2 & 0.3 & 0.8 \\ 0.2 & 1.2 & 0.3 & 0.5 \\ 0.3 & 0.3 & 2.1 & 0.3 \\ 0.8 & 0.5 & 0.3 & 1.6 \end{bmatrix}$	75

Table 2. Posterior probability of M for the £ve datasets.

Data	$p(M \vert \mathcal{X})$
1	$p(3 \vert \mathcal{X}) = \mathbf{0.8561}$, $p(4 \vert \mathcal{X}) = 0.1310$, $p(5 \vert \mathcal{X}) = 0.0120$, $p(6 \vert \mathcal{X}) = 0.0009$
2	$p(2 \vert \mathcal{X}) = 0.1660$, $p(3 \vert \mathcal{X}) = \mathbf{0.6390}$, $p(4 \vert \mathcal{X}) = 0.1652$, $p(5 \vert \mathcal{X}) = 0.0272$, others $= 0.0026$
3	$p(2 \vert \mathcal{X}) = \mathbf{0.6442}$, $p(3 \vert \mathcal{X}) = 0.2421$, $p(4 \vert \mathcal{X}) = 0.0712$, $p(5 \vert \mathcal{X}) = 0.0177$, others $= 0.0248$
4	$p(10 \vert \mathcal{X}) = \mathbf{0.5810}$, $p(11 \vert \mathcal{X}) = 0.3215$, $p(9 \vert \mathcal{X}) = 0.0841$, others $= 0.0134$
5	$p(3 \vert \mathcal{X}) = \mathbf{0.9321}$, $p(4 \vert \mathcal{X}) = 0.0588$, $p(5 \vert \mathcal{X}) = 0.0091$

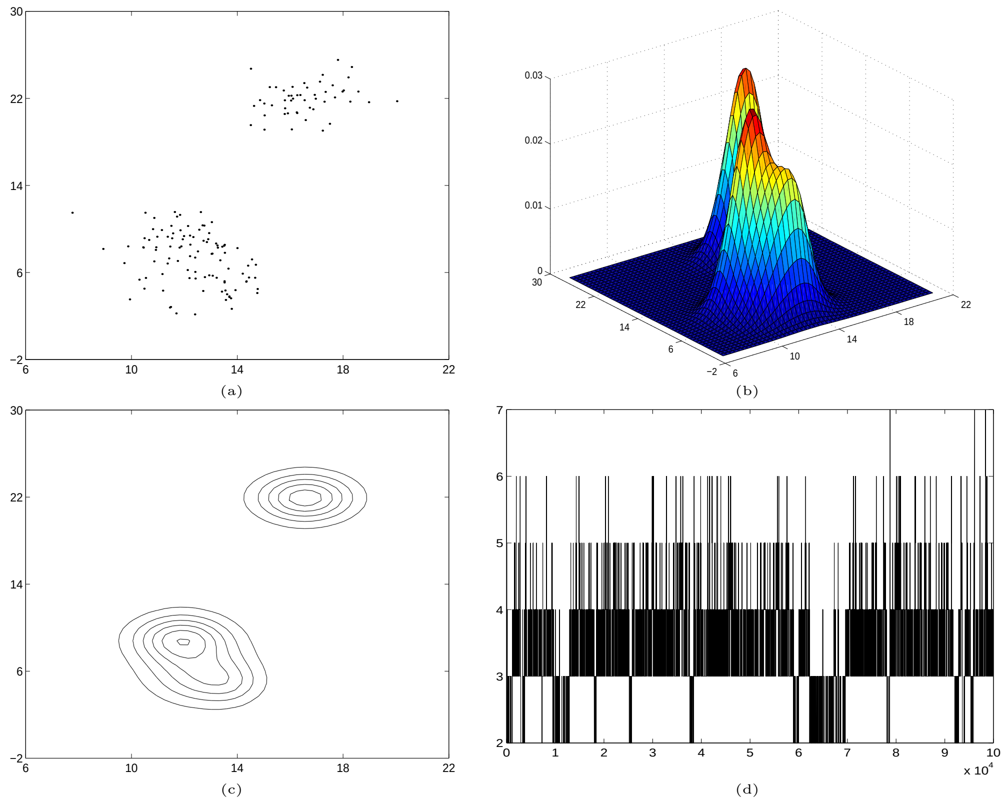

Fig. 2. Dataset 2: (a) original data points; (b) overall predictive density; (c) the contour of overall predictive density; (d) the number of components vs. the number of sweeps.

At each sweep of the algorithm, the values for the mixing weights and parameters (Π, Θ) are produced, with which the densities $p(\cdot|M, \Pi, \Theta) = \sum_{m=1}^{M} \pi_m p(\cdot|\boldsymbol{\theta}_m)$ can be computed. Averaging $p(\cdot|M, \Pi, \Theta)$ across the MCMC run, conditioned on fixed value of M, gives an estimate of $E\left[p(\cdot|M, \Pi, \Theta)|M, \mathcal{X}\right]$, a Bayesian predictive density estimate of the GMM with M components. An essential attribute of the RJMCMC sampler is its ability to jump between different values of M. A plot of the changes in M against the number of sweeps is depicted in Figures 1- 3(d).

The first dataset is very easy to analyze because it consists of three well-separated classes. From Figure 1 and Table 2, we can ascertain that the number of classes, i.e., the number of components, is 3. The second dataset is more difficult to analyze because two of three classes are severely overlapped, and the other one is also moderately overlapped with these two. Note that the posterior $p(M = 3|\mathcal{X}) = 0.6390$, $p(M = 2|\mathcal{X}) = 0.1660$ and $p(M = 4|\mathcal{X}) = 0.1652$, so the probability of clustering the observations into three classes is the largest. The third dataset is the most difficult to analyze. From Figure 3(a), it is seen that all classes are severely overlapped and it cannot tell us how many classes there are by visual inspection. Our algorithm considers the most possible number of classes as 2, and the second possible number as 3. This result is understandable, because among the three components of the GMM from which the observations are generated, there are

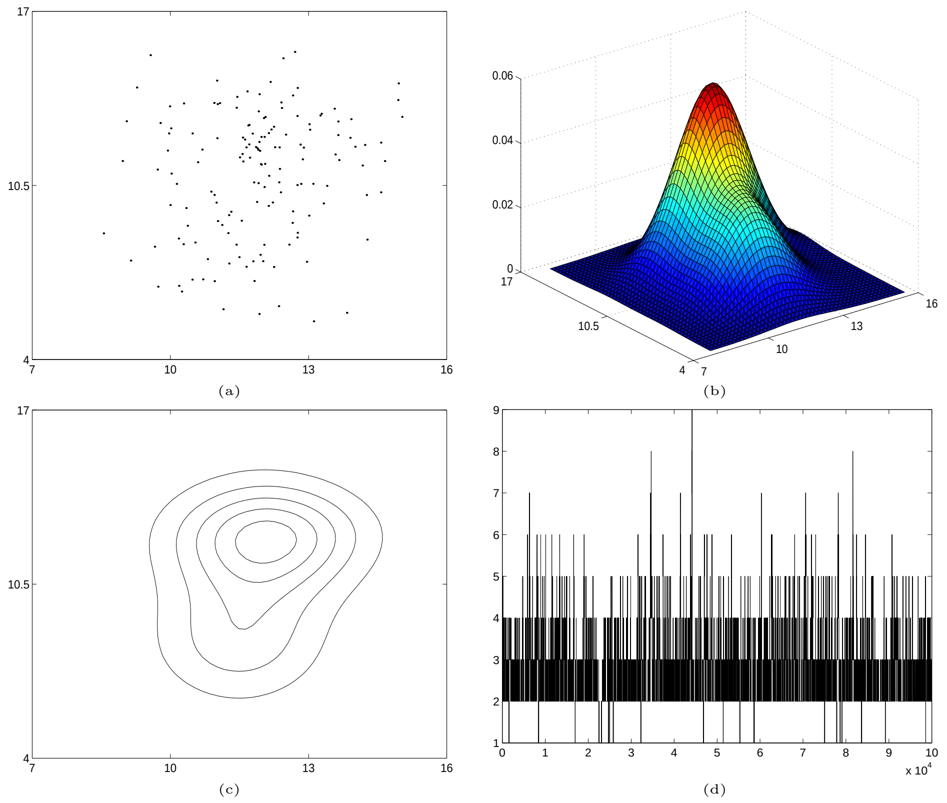

Fig. 3. Dataset 3: (a) original data points; (b) overall predictive density; (c) the contour of overall predictive density; (d) the number of components vs. the number of sweeps.

two components, whose mean vectors are almost the same, and the observations that are generated from these two components should be clustered into one large class.

In the second experiment, we ran our algorithms for 20,000 sweeps on it. The first 10,000 sweeps are considered as burn-in, and will be discarded. All our inferences are based on the next 10,000 sweeps. This experiment uses two datasets, i.e., dataset 4 and dataset 5, of which one is from a 8-dimensional GMM and another is from a 4-dimensional GMM (Table 1). In the dataset 4, the covariance matrix of each component is diagonal but has the different diagonal elements, and in the dataset 5, the covariance matrix of each component is a general positive definite matrix.

The estimated posterior probabilities of M are given in Table 2. We can see that our simplified GMM is able to find the expected number of components for each of the two sets. In Tables 3 and 4, we give the estimated mixing weights π, mean vectors $\boldsymbol{\mu}$'s and eigenvalues, $\boldsymbol{\lambda}_m = (\lambda_{m1}, \ldots, \lambda_{mp})^T$, of the covariance matrices, corresponding to the expected number of components for each mixture. Since each component in a GMM represents one class and the mixing weight of component represents the prior probability of the corresponding class, the mixing weight implies the percentage of the number of observations belonging to the class out of the total number of observations. Therefore the mixing weights can tell us the accuracy of clustering or classification. For these two GMMs, the number of the

Table 3. The estimated mixing weights π, means $\boldsymbol{\mu}$ and eigenvalues $\boldsymbol{\lambda}$ of the covariance matrices for our used dataset 4 when $M = 10$.

$\pi_1 = 0.1248$, $\pi_2 = 0.0628$, $\pi_3 = 0.0629$, $\pi_4 = 0.1250$, $\pi_5 = 0.0630$								
$\pi_6 = 0.1248$, $\pi_7 = 0.1247$, $\pi_8 = 0.0625$, $\pi_9 = 0.1250$, $\pi_{10} = 0.1245$								
$\boldsymbol{\mu}_1^T =$	(-20.12	2.84	-20.27	20.02	-10.24	4.79	11.14	14.97)
$\boldsymbol{\mu}_2^T =$	(-15.02	-1.94	9.66	-15.01	-0.25	-8.46	8.48	11.56)
$\boldsymbol{\mu}_3^T =$	(-14.08	8.38	-2.86	1.21	-3.76	-9.02	8.87	7.73)
$\boldsymbol{\mu}_4^T =$	(-9.85	-7.07	-0.23	-9.91	3.09	-5.12	3.24	5.32)
$\boldsymbol{\mu}_5^T =$	(9.75	-11.59	4.57	0.36	6.32	7.47	-3.02	5.03)
$\boldsymbol{\mu}_6^T =$	(14.71	-3.94	-15.13	14.78	7.92	20.04	1.50	11.84)
$\boldsymbol{\mu}_7^T =$	(24.23	21.10	-4.99	5.25	-3.02	9.76	-8.66	-2.90)
$\boldsymbol{\mu}_8^T =$	(35.33	-22.30	30.28	-25.08	9.30	-8.00	17.94	3.08)
$\boldsymbol{\mu}_9^T =$	(39.98	-29.88	35.13	5.17	-22.04	9.89	21.90	-0.98)
$\boldsymbol{\mu}_{10}^T =$	(60.01	-9.32	15.34	-5.15	-12.05	0.25	12.17	-2.24)
$\boldsymbol{\lambda}_1^T =$	(3.23	6.50	1.96	3.13	5.60	2.08	1.61	3.13)
$\boldsymbol{\lambda}_2^T =$	(8.31	3.63	3.16	2.40	4.35	8.27	5.71	1.86)
$\boldsymbol{\lambda}_3^T =$	(7.44	3.66	4.22	3.41	4.60	6.86	7.85	1.70)
$\boldsymbol{\lambda}_4^T =$	(3.20	7.90	3.76	2.35	2.54	1.21	4.72	4.10)
$\boldsymbol{\lambda}_5^T =$	(8.78	2.87	5.22	1.72	3.21	2.66	9.69	0.80)
$\boldsymbol{\lambda}_6^T =$	(5.17	5.98	0.96	7.86	1.10	4.97	7.74	1.89)
$\boldsymbol{\lambda}_7^T =$	(4.48	1.87	3.98	3.78	1.02	4.50	3.53	2.46)
$\boldsymbol{\lambda}_8^T =$	(12.54	3.96	2.16	1.32	11.80	7.36	0.99	2.08)
$\boldsymbol{\lambda}_9^T =$	(2.30	1.48	2.18	12.56	4.42	14.54	5.87	1.32)
$\boldsymbol{\lambda}_{10}^T =$	(2.05	8.58	2.19	3.30	3.76	2.95	5.62	8.79)

Table 4. The estimated mixing weights π, means $\boldsymbol{\mu}$ and eigenvalues $\boldsymbol{\lambda}$ of the covariance matrices for our used dataset 5 when $M = 3$..

$\pi_1 = 0.3992$	$\pi_2 = 0.3007$	$\pi_3 = 0.3001$
$\boldsymbol{\mu}_1^T =$ (-0.96, 5.04, -3.09, 3.96)	$\boldsymbol{\mu}_2^T =$ (3.98, 9.90, 5.83, 6.59)	$\boldsymbol{\mu}_3^T =$ (6.99, 14.98, 10.98, 9.96)
$\boldsymbol{\lambda}_1^T =$ (1.12, 1.32, 1.39, 2.25)	$\boldsymbol{\lambda}_2^T =$(1.68, 2.17, 1.16, 1.61)	$\boldsymbol{\lambda}_3^T =$ (1.44, 0.94, 1.12, 1.79)

generated observations from each class is somewhat different, we still obtain the matched mixing weights. This shows that our model works well for clustering with unknown number of classes.

For the dataset 4, the covariance matrices are all diagonal, so the corresponding eigenvector matrices are the same (i.e., equal to the identity matrix). Thus this dataset is generated from our considered mixture model. Clearly, the dataset 5 is generated from the general mixture model. Our results demonstrate the our considered model for it still works well in both parameter estimation and model selection.

It is well-known that Gibbs sampler simulations could encounter *trapping states* and the so-called *label switching* (Redner and Walker, 1984) could occur because of symmetry in the likelihood of the model parameters, when dealing with mixture models. Since our algorithm is a generalization of the algorithm by Richardson and Green (1997) to multivariate setting, the methods proposed by Richardson and Green (1997) that deal with this problems can be also used for our algorithm. Similarly to the method suggested by Richardson and Green (1997), here we impose an identifiability constraint on the parameter space such as the orderings of the mixing weights and the mean vectors. In the above experiments, we order the mean vectors alphabetically to handle the label switching problem. Here the alphabetical order between two $n-$dimensional vectors $\mathbf{x} = [x_1, \ldots, x_n]^T$ and $\mathbf{y} = [y_1, \ldots, y_n]^T$ is defined as $\mathbf{x} > \mathbf{y}$ iff there exists a $k < n$ such that $x_i = y_i (i = 1, \ldots, k)$ and $x_{k+1} > y_{k+1}$. It is worthy to note that for the label switching problem, some sophisticated methods, based on the decision theory, are proposed by Celeux et al. (2000) and Stephens (2000b).

6. Concluding Remarks

This paper shows that it is possible to use the reversible jump MCMC methodology for fully Bayesian analysis of multivariate GMMs. Following the constraints of preserving the first second moments before and after the split or combine moves, we applied the reversible jump MCMC algorithm to a simplified multivariate GMM, where the eigenvalue matrices of all covariance matrices are the same. This simplified GMM is one of the most representative model among GMMs, because firstly a GMM with diagonal covariance matrices, which is widely used in classification, clustering or density function estimation, is a special case of our simplified model. Secondly, using our simplified model can still obtain very good estimates on general GMMs, especially on their model selection. Our experiments demonstrated this point.

As discussed in fully Bayesian analysis for univariate hidden Markov models by Robert *et al.* (2000), we can consider the split move as two independent subsplit moves, which respectively involve transition probability matrices and component parameters. Moreover, the Jacobian determinant of the total split moves is the product of the subdeterminants corresponding to these two subsplit moves. Making use of the subsplit move for transition matrices by Robert *et al.*(2000) and the subsplit move for component parameters in this paper, we can immediately present a methodology of fully Bayesian inference in multivariate hidden Markov models. In a recent article (Zhang et al., 2003), a split method that satisfies the split equations (7), (8) and (9) for an arbitrary-structure covariance matrix has been presented. However, since the numbers of free parameters before and after the split are different, it is hard to use this method to devise a reversible jump for general GMMs. Nevertheless, we think that this parameter-driven method can be introduced into the VB framework, giving rise to an efficient on-line model selection method.

A. The Calculation of the Jacobian Matrix

Denote $\mathbf{s}' = \left\{\pi_{j'}, \pi_{k'}, \boldsymbol{\mu}_{j'}, \boldsymbol{\mu}_{k'}, \mathbf{g}_{j'}, \mathbf{g}_{k'}\right\}$, and $\mathbf{s} = \{\pi_k, \alpha, \boldsymbol{\mu}_k, \mathbf{u}, \mathbf{g}_k, \mathbf{b}\}$, where $\mathbf{u} = (u_1, \ldots, u_p)^T$, $\mathbf{g}_{j'} = (\lambda_{j'1}, \ldots, \lambda_{j'p})^T$, $\mathbf{g}_{k'} = (\lambda_{k'1}, \ldots, \lambda_{k'p})^T$, $\mathbf{g}_k = (\lambda_{k1}, \ldots, \lambda_{kp})^T$ and $\mathbf{b} = (\beta_1, \ldots, \beta_p)^T$. Then the transformation defined by (13), (14) and (15) is from $\{\pi_k, \alpha, \boldsymbol{\mu}_k, \mathbf{u}, \mathbf{g}_k, \mathbf{b}\}$ to $\left\{\pi_{j'}, \pi_{k'}, \boldsymbol{\mu}_{j'}, \boldsymbol{\mu}_{k'}, \mathbf{g}_{j'}, \mathbf{g}_{k'}\right\}$, and the corresponding Jacobian matrix J is:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \pi_{j'}}{\partial \pi_k} & \frac{\partial \pi_{j'}}{\partial \alpha} & \frac{\partial \pi_{j'}}{\partial \boldsymbol{\mu}_k} & \frac{\partial \pi_{j'}}{\partial \mathbf{u}} & \frac{\partial \pi_{j'}}{\partial \mathbf{g}_k} & \frac{\partial \pi_{j'}}{\partial \mathbf{b}} \\ \frac{\partial \pi_{k'}}{\partial \pi_k} & \frac{\partial \pi_{k'}}{\partial \alpha} & \frac{\partial \pi_{k'}}{\partial \boldsymbol{\mu}_k} & \frac{\partial \pi_{k'}}{\partial \mathbf{u}} & \frac{\partial \pi_{k'}}{\partial \mathbf{g}_k} & \frac{\partial \pi_{k'}}{\partial \mathbf{b}} \\ \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \pi_k} & \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \alpha} & \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \boldsymbol{\mu}_k} & \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \mathbf{u}} & \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \mathbf{g}_k} & \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \mathbf{b}} \\ \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \pi_k} & \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \alpha} & \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \boldsymbol{\mu}_k} & \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \mathbf{u}} & \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \mathbf{g}_k} & \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \mathbf{b}} \\ \frac{\partial \mathbf{g}_{j'}}{\partial \pi_k} & \frac{\partial \mathbf{g}_{j'}}{\partial \alpha} & \frac{\partial \mathbf{g}_{j'}}{\partial \boldsymbol{\mu}_k} & \frac{\partial \mathbf{g}_{j'}}{\partial \mathbf{u}} & \frac{\partial \mathbf{g}_{j'}}{\partial \mathbf{g}_k} & \frac{\partial \mathbf{g}_{j'}}{\partial \mathbf{b}} \\ \frac{\partial \mathbf{g}_{k'}}{\partial \pi_k} & \frac{\partial \mathbf{g}_{k'}}{\partial \alpha} & \frac{\partial \mathbf{g}_{k'}}{\partial \boldsymbol{\mu}_k} & \frac{\partial \mathbf{g}_{k'}}{\partial \mathbf{u}} & \frac{\partial \mathbf{g}_{k'}}{\partial \mathbf{g}_k} & \frac{\partial \mathbf{g}_{k'}}{\partial \mathbf{b}} \end{bmatrix}. \qquad \text{(A1)}$$

From (13), (14) and (15), we can calculate the partial derivatives:

$$\frac{\partial \pi_{j'}}{\partial \pi_k} = \alpha, \frac{\partial \pi_{j'}}{\partial \alpha} = \pi_k, \frac{\partial \pi_{j'}}{\partial \boldsymbol{\mu}_k} = \mathbf{0}_{1\times p}, \frac{\partial \pi_{j'}}{\partial \mathbf{u}} = \mathbf{0}_{1\times p}, \frac{\partial \pi_{j'}}{\partial \mathbf{g}_k} = \mathbf{0}_{1\times p}, \frac{\partial \pi_{j'}}{\partial \mathbf{u}} = \mathbf{0}_{1\times p}, \tag{A2}$$

$$\frac{\partial \pi_{k'}}{\partial \pi_k} = 1-\alpha, \frac{\partial \pi_{k'}}{\partial \alpha} = -\pi_k, \frac{\partial \pi_{k'}}{\partial \boldsymbol{\mu}_k} = \mathbf{0}_{1\times p}, \frac{\partial \pi_{k'}}{\partial \mathbf{u}} = \mathbf{0}_{1\times p}, \frac{\partial \pi_{k'}}{\partial \mathbf{g}_k} = \mathbf{0}_{1\times p}, \frac{\partial \pi_{kj'}}{\partial \mathbf{u}} = \mathbf{0}_{1\times p}, \tag{A3}$$

$$\frac{\partial \boldsymbol{\mu}_{j'}}{\partial \pi_k} = \mathbf{0}_{p\times 1}, \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \alpha} = \mathbf{0}_{p\times 1}, \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \mathbf{b}} = \mathbf{0}_{p\times p}; \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \pi_k} = \mathbf{0}_{p\times 1}, \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \alpha} = \mathbf{0}_{p\times 1}, \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \mathbf{b}} = \mathbf{0}_{p\times p}, \tag{A4}$$

$$\frac{\partial \mathbf{g}_{j'}}{\partial \pi_k} = \mathbf{0}_{p\times 1}, \frac{\partial \mathbf{g}_{j'}}{\partial \alpha} = \mathbf{0}_{p\times 1}, \frac{\partial \mathbf{g}_{j'}}{\partial \boldsymbol{\mu}_k} = \mathbf{0}_{p\times p}; \frac{\partial \mathbf{g}_{k'}}{\partial \pi_k} = \mathbf{0}_{p\times 1}, \frac{\partial \mathbf{g}_{k'}}{\partial \alpha} = \mathbf{0}_{p\times 1}, \frac{\partial \mathbf{g}_{k'}}{\partial \boldsymbol{\mu}_k} = \mathbf{0}_{p\times p}, \tag{A5}$$

and, for $(n = 1, \ldots, p)$

$$\begin{array}{ll} \frac{\partial \boldsymbol{\mu}_{j'}}{\partial u_n} = -\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\lambda_{kn}^{\frac{1}{2}}\mathbf{a}_n, & \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \lambda_{kn}} = -\frac{1}{2}\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\lambda_{kn}^{-\frac{1}{2}}u_n\mathbf{a}_n, \\ \frac{\partial \boldsymbol{\mu}_{k'}}{\partial u_n} = \sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\lambda_{kn}^{\frac{1}{2}}\mathbf{a}_n, & \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \lambda_{kn}} = \sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\lambda_{kn}^{-\frac{1}{2}}u_n\mathbf{a}_n, \end{array}$$

$$\frac{\partial \lambda_{j'n}}{\partial u_l} = \begin{cases} -2\beta_n\lambda_{kn}u_n\frac{\pi_k}{\pi_{j'}} & l = n \\ 0 & l \neq n \end{cases}, \quad \frac{\partial \lambda_{k'n}}{\partial u_l} = \begin{cases} -2(1-\beta_n)\lambda_{kn}u_n\frac{\pi_k}{\pi_{k'}} & l = n \\ 0 & j \neq n \end{cases},$$

$$\frac{\partial \lambda_{j'n}}{\partial \lambda_{kl}} = \begin{cases} \beta_n(1-u_n^2)\frac{\pi_k}{\pi_{j'}} & l = n \\ 0 & l \neq n \end{cases}, \quad \frac{\partial \lambda_{k'n}}{\partial \lambda_{kl}} = \begin{cases} (1-\beta_n)(1-u_n^2)\frac{\pi_k}{\pi_{k'}} & l = n \\ 0 & l \neq n \end{cases},$$

$$\frac{\partial \lambda_{j'n}}{\partial \beta_l} = \begin{cases} \lambda_{kn}(1-u_n^2)\frac{\pi_k}{\pi_{j'}} & l = n \\ 0 & l \neq n \end{cases}, \quad \frac{\partial \lambda_{k'n}}{\partial \beta_l} = \begin{cases} -\lambda_{kn}(1-u_n^2)\frac{\pi_k}{\pi_{k'}} & l = n \\ 0 & l \neq n \end{cases}.$$

So, we have

$$\begin{array}{lll} \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \boldsymbol{\mu}_k} = \mathbf{I}, & \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \mathbf{u}} = -\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{\frac{1}{2}}, & \frac{\partial \boldsymbol{\mu}_{j'}}{\partial \mathbf{g}_k} = -\frac{1}{2}\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{-\frac{1}{2}}\mathbf{U}, \\ \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \boldsymbol{\mu}_k} = \mathbf{I}, & \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \mathbf{u}} = \sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{\frac{1}{2}}, & \frac{\partial \boldsymbol{\mu}_{k'}}{\partial \mathbf{g}_k} = \frac{1}{2}\sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{-\frac{1}{2}}\mathbf{U} \end{array}. \tag{A6}$$

$$\begin{array}{lll} \frac{\partial \mathbf{g}_{j'}}{\partial \mathbf{u}} = -2\frac{\pi_k}{\pi_{j'}}\boldsymbol{\Lambda}_k\mathbf{B}\mathbf{U}, & \frac{\partial \mathbf{g}_{j'}}{\partial \mathbf{g}_k} = \frac{\pi_k}{\pi_{j'}}\mathbf{B}\left(\mathbf{I}-\mathbf{U}^2\right), & \frac{\partial \mathbf{g}_{j'}}{\partial \mathbf{b}} = \frac{\pi_k}{\pi_{j'}}\boldsymbol{\Lambda}_k\left(\mathbf{I}-\mathbf{U}^2\right), \\ \frac{\partial \mathbf{g}_{k'}}{\partial \mathbf{u}} = -2\frac{\pi_k}{\pi_{k'}}\boldsymbol{\Lambda}_k\left(\mathbf{I}-\mathbf{B}\right)\mathbf{U}, & \frac{\partial \mathbf{g}_{k'}}{\partial \mathbf{g}_k} = \frac{\pi_k}{\pi_{k'}}\left(\mathbf{I}-\mathbf{B}\right)\left(\mathbf{I}-\mathbf{U}^2\right), & \frac{\partial \mathbf{g}_{k'}}{\partial \mathbf{b}} = -\frac{\pi_k}{\pi_{k'}}\boldsymbol{\Lambda}_k\left(\mathbf{I}-\mathbf{U}^2\right) \end{array}. \tag{A7}$$

Substituting (A2)-(A7) into (A1), we obtain (16).

B. The Calculation of the Jacobian Determinant

From the Jacobian matrix defined by (16), we immediately have

$$\det(\mathbf{J}) = \pi_k \cdot \det(\mathbf{J}_1), \tag{B1}$$

where

$$\mathbf{J}_1 = \begin{bmatrix} \mathbf{I} & -\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{\frac{1}{2}} & -\frac{1}{2}\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{-\frac{1}{2}}\mathbf{U} & \mathbf{0}_{p\times p} \\ \mathbf{I} & \sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{\frac{1}{2}} & \frac{1}{2}\sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\mathbf{A}\boldsymbol{\Lambda}_k^{-\frac{1}{2}}\mathbf{U} & \mathbf{0}_{p\times p} \\ \mathbf{0}_{p\times p} & -2\frac{\pi_k}{\pi_{j'}}\boldsymbol{\Lambda}_k\mathbf{B}\mathbf{U} & \frac{\pi_k}{\pi_{j'}}\mathbf{B}(\mathbf{I}-\mathbf{U}^2) & \frac{\pi_k}{\pi_{j'}}\boldsymbol{\Lambda}_k(\mathbf{I}-\mathbf{U}^2) \\ \mathbf{0}_{p\times p} & 2\frac{\pi_k}{\pi_{k'}}\boldsymbol{\Lambda}_k(\mathbf{B}-\mathbf{I})\mathbf{U} & \frac{\pi_k}{\pi_{k'}}(\mathbf{I}-\mathbf{B})(\mathbf{I}-\mathbf{U}^2) & \frac{\pi_k}{\pi_{k'}}\boldsymbol{\Lambda}_k(\mathbf{U}^2-\mathbf{I}) \end{bmatrix}. \tag{B2}$$

We partition $\mathbf{J}_1$ into $\mathbf{J}_1 = \begin{bmatrix} \mathbf{J}_{11} & \mathbf{J}_{12} \\ \mathbf{J}_{21} & \mathbf{J}_{22} \end{bmatrix}$, where

$$\mathbf{J}_{11} = \begin{bmatrix} \mathbf{I} & -\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\mathbf{A}\mathbf{\Lambda}_k^{\frac{1}{2}} \\ \mathbf{I} & \sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\mathbf{A}\mathbf{\Lambda}_k^{\frac{1}{2}} \end{bmatrix}, \quad \mathbf{J}_{12} = \begin{bmatrix} -\frac{1}{2}\sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\mathbf{A}\mathbf{\Lambda}_k^{-\frac{1}{2}}\mathbf{U} & \mathbf{0}_{p\times p} \\ \frac{1}{2}\sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\mathbf{A}\mathbf{\Lambda}_k^{-\frac{1}{2}}\mathbf{U} & \mathbf{0}_{p\times p} \end{bmatrix},$$

$$\mathbf{J}_{21} = \begin{bmatrix} \mathbf{0}_{p\times p} & -2\frac{\pi_k}{\pi_{j'}}\mathbf{\Lambda}_k\mathbf{B}\mathbf{U} \\ \mathbf{0}_{p\times p} & 2\frac{\pi_k}{\pi_{k'}}\mathbf{\Lambda}_k(\mathbf{B}-\mathbf{I})\mathbf{U} \end{bmatrix}, \mathbf{J}_{22} = \begin{bmatrix} \frac{\pi_k}{\pi_{j'}}\mathbf{B}(\mathbf{I}-\mathbf{U}^2) & \frac{\pi_k}{\pi_{j'}}\mathbf{\Lambda}_k(\mathbf{I}-\mathbf{U}^2) \\ \frac{\pi_k}{\pi_{k'}}(\mathbf{I}-\mathbf{B})(\mathbf{I}-\mathbf{U}^2) & \frac{\pi_k}{\pi_{k'}}\mathbf{\Lambda}_k(\mathbf{U}^2-\mathbf{I}) \end{bmatrix}.$$

According to theorem in (Golub and Van Loan, 1996; Anderson, 1984), we can have:

$$\det(\mathbf{J}_1) = \det(\mathbf{J}_{11}) \cdot \det(\mathbf{J}_{22} - \mathbf{J}_{21}\mathbf{J}_{11}^{-1}\mathbf{J}_{12}). \tag{B3}$$

Through some arithmetic calculations, it is easily to obtain

$$\mathbf{J}_{11}^{-1} = \begin{bmatrix} \frac{\pi_{j'}}{\pi_k}\mathbf{I} & \frac{\pi_{k'}}{\pi_k}\mathbf{I} \\ -\frac{\sqrt{\pi_{k'}\pi_{j'}}}{\pi_k}\mathbf{\Lambda}_k^{-\frac{1}{2}}\mathbf{A}^T & \frac{\sqrt{\pi_{k'}\pi_{j'}}}{\pi_k}\mathbf{\Lambda}_k^{-\frac{1}{2}}\mathbf{A}^T \end{bmatrix}$$

and

$$\det(\mathbf{J}_{11}) = \det\left(\sqrt{\frac{\pi_{j'}}{\pi_{k'}}}\mathbf{A}\mathbf{\Lambda}_k^{\frac{1}{2}} + \sqrt{\frac{\pi_{k'}}{\pi_{j'}}}\mathbf{A}\mathbf{\Lambda}_k^{\frac{1}{2}}\right) = \left(\frac{\pi_k}{\sqrt{\pi_{j'}\pi_{k'}}}\right)^p \cdot \prod_{n=1}^{p} \lambda_{kn}^{\frac{1}{2}}. \tag{B4}$$

Here we use the property that the eigenvector matrix $\mathbf{A}$ is orthogonal matrix and its determinant is one. Using simple matrix operations, we have

$$\mathbf{J}_{21}\mathbf{J}_{11}^{-1}\mathbf{J}_{12} = \begin{bmatrix} -\frac{\pi_k}{\pi_{j'}}\mathbf{B}\mathbf{U}^2 & \mathbf{0}_{p\times p} \\ -\frac{\pi_k}{\pi_{k'}}(\mathbf{I}-\mathbf{B})\mathbf{U}^2 & \mathbf{0}_{p\times p} \end{bmatrix},$$

$$\mathbf{J}_{22} - \mathbf{J}_{21}\mathbf{J}_{11}^{-1}\mathbf{J}_{12} = \begin{bmatrix} \frac{\pi_k}{\pi_{j'}}\mathbf{B} & \frac{\pi_k}{\pi_{j'}}\mathbf{\Lambda}_k(\mathbf{I}-\mathbf{U}^2) \\ \frac{\pi_k}{\pi_{k'}}(\mathbf{I}-\mathbf{B}) & -\frac{\pi_k}{\pi_{k'}}\mathbf{\Lambda}_k(\mathbf{I}-\mathbf{U}^2) \end{bmatrix}.$$

Thus,

$$\begin{aligned} &\det\left(\mathbf{J}_{22} - \mathbf{J}_{21}\mathbf{J}_{11}^{-1}\mathbf{J}_{12}\right) \\ = \;& \det(\frac{\pi_k}{\pi_{j'}}\mathbf{B})\det\left(-\frac{\pi_k}{\pi_{k'}}\mathbf{\Lambda}_k(\mathbf{I}-\mathbf{U}^2) - \frac{\pi_k}{\pi_{k'}}(\mathbf{I}-\mathbf{B})\frac{\pi_{j'}}{\pi_k}\mathbf{B}^{-1}\frac{\pi_k}{\pi_{j'}}\mathbf{\Lambda}_k(\mathbf{I}-\mathbf{U}^2)\right) \\ = \;& (-1)^p\left(\frac{\pi_k^2}{\pi_{j'}\pi_{k'}}\right)^p \prod_{n=1}^{p}(1-u_n^2)\lambda_{kn}. \end{aligned}$$

Substituting the above equation and (B4) into (B3), and then into (B1), we obtain (17).

References

Andrieu, C., N. de Freitas, and A. Doucet (2001). Robust full Bayesian learning for radial basis networks. *Neural Computation 13*, 2359–2407.

Andrieu, C. and A. Doucet (1999). Joint Bayesian model selection and estimation of noisy sinusoids via reversible jump MCMC. *IEEE Transactions Signal Processing 47*, 2667–2676.

Attial, H. (2000). A variational Bayesian framework for graphical models. *Advances in Neural Information Processing Systems 12*, 21–30.

Banfield, J. D. and A. E. Raftery (1993). Model-based Gaussian and non-Gaussian clustering. *Biometrics 49*, 803–821.

Bensmail, H. and G. Celeux (1996). Regularized Gaussian discriminant analysis through eigenvalue decomposition. *Joutnal of the American Statistical Association 91*(436), 1743–1748.

Bensmail, H., G. Celeux, A. E. Raftery, and C. P. Robert (1997). Inference in model-based cluster analysis. *Statistics and Computing 7*, 1–10.

Cappé, O., C. P. Robert, and T. Rydén (2003). Reversible jump, birth-and-death and more general continuous time Markov chain Monte Carlo samplers. *Journal of the Royal Statistical Society, B 65*, 679–700.

Celeux, G., M. Hurn, and C. P. Robert (2000). Computational and inferential difficulties with mixture posterior distributions. *Journal of the American Statistical Association 95*(451), 957–970.

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

Diebolt, J. and C. P. Robert (1994). Estimation of finite mixture distribution through Bayesian sampling. *Journal of the Royal Statistical Society Series B 56*(2), 363–375.

Gilks, W. R., S. Richardson, and D. J. Spiegelhalter (1996). *Markov Chain Monte Carlo in Practice*. London: Chapman and Hall.

Golub, G. H. and C. F. V. Loan (1996). *Matrix Computations* (Third ed.). Baltimore: The Johns Hopkins University Press.

Green, P. J. (1994). Discussion on representations of knowledge in complex systems (by U. Grenander). *Journal of the Royal Statistial Society Series B 56*, 589–590.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika 82*, 711–732.

Grenander, U. and M. I. Miller (1994). Representations of knowledge in complex systems (with discussion). *Journal of the Royal Statistical Society Series B 56*, 549–603.

Hastie, T. and R. Tibshirani (1996). Discriminant analysis by Gaussian mixtures. *Journal of the Royal Statistical Society Series B 58*, 155–176.

Holmes, C. C. and B. K. Mallick (1998). Bayesian radial basis functions of variable dimension. *Neural Computation 10*, 1217–1233.

Larocque, J. R. and J. P. Reilly (2002). Reversible jump MCMC for joint detection and estimation of sources in colored noise. *IEEE Transactions on Signal Processing 50*(2), 231–240.

McLachlan, G. and T. Krishnan (1997). *The EM Algorithm and Extensions*. New York: John Wiley and Sons.

McLachlan, G. and D. Peel (2000). *Finite Mixture Models*. New York: John Wiley and Sons.

Redner, R. A. and H. F. Walker (1984). Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review 26*(2), 195–239.

Richardson, S. and P. J. Green (1997). On Bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society Series B 59*, 731–792.

Richardson, S. and P. J. Green (1998). Corrigendum: On Bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society Series B 60*, 661.

Robert, C. P. (1996). Mixtures of distributions : inference and estimation. In W. R. Gilks, S. Richardson, and D. J. Spiegelhalter (Eds.), *Markov Chain Monte Carlo in Practice*, Chapter 24, pp. 441–464. London: Chapman and Hall.

Robert, C. P., T. Rydén, and D. M. Titterington (2000). Bayesian inference in hidden markov models through the reversible jump Markov chain Monte Carlo method. *Journal of the Royal Statistical Society Series B 62*, 57–75.

Sato, M. (2001). On-line model selection based on the variational Bayes. *Neural Computation 13*(7), 1649–1681.

Stephens, M. (2000a). Bayesian analysis of mixtures with an unknown number of components — an alternative to reversible jump methods. *Annals of Statistics 28*, 40–74.

Stephens, M. (2000b). Dealing with label switching in mixture models. *Journal of the Royal Statistical Society Series B 62*, 795–809.

Titterington, D. M., A. F. M. Smith, and U. E. Makov (1985). *Statistical Analysis of Finite Mixture Distributions*. Wiley Series in Probability and Mathematical Statistics. New York: Wiley.

Zhang, Z., C. Chen, J. Sun, and K. L. Chan (2003). EM algorithms for Gaussian mixtures with split-and-merge operation. *Pattern Recognition 36*, 1973–1983.