Neurocomputing 151 (2015) 1015–1026

Contents lists available at ScienceDirect

Neurocomputing

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

BYY harmony learning of log-normal mixtures with automated model selection

Wenli Zheng, Zhijie Ren, Yifan Zhou, Jinwen Ma*

Department of Information Science, School of Mathematical Sciences and LMAM, Peking University, Beijing 100871, China

ARTICLE INFO

Article history:
Received 31 October 2013
Received in revised form
2 April 2014
Accepted 15 April 2014
Available online 19 October 2014

Keywords:
Bayesian Ying–Yang (BYY) harmony learning
Automated model selection
Parameter estimation
Logarithmic normal (log-normal) mixture

ABSTRACT

Bayesian Ying–Yang (BYY) harmony learning system is a powerful tool for statistical learning. Via the BYY harmony leaning of finite mixtures, model selection, i.e., the selection of an appropriate number of components for the mixture, can be made automatically during parameter learning on a given dataset. In this paper, an adaptive gradient BYY harmony learning algorithm is proposed for log-normal mixtures to implement parameter learning with automated model selection. It is demonstrated by the experiments on both synthetic and real-world datasets that the proposed BYY harmony learning algorithm not only has the ability of automated model selection, but also leads to a rather good estimation of the parameters in the original log-normal mixture.

1. Introduction

In data analysis and information processing, the finite mixture model is frequently used since there are so many practical cases in which the data can be regarded as arising from two or more populations linearly mixed in certain proportions (e.g., [1,2]). Although there have been already several statistical methods for finite mixture learning, such as the EM algorithm [3] and the self-organizing network with hyper-ellipsoidal clustering [4], the number k of components in the mixture should be known in advance. However, in many cases this critical information is not available so that an appropriate value of k must be selected or determined with the learning of other parameters in the mixture model, which is a rather difficult task [5]. Actually, since the number of components is just a scale of the finite mixture model, its selection is usually referred to as model selection in the literature. Our interest here focuses on this compound modeling problem of both parameter learning and model selection of finite mixtures only with a given set of sample data.

For solving this compound mixture modeling problem, a traditional statistical approach is to choose the optimal number k^* of components via one of the information, coding and statistical selection criteria such as Akaike's Information Criterion (AIC) [6] and Bayesian Inference Criterion (BIC) [7]. That is, given an interval of possible k and for each integer in it, we implement the EM or other parameter learning algorithm to estimate the parameters of the mixture. With all these obtained results, we can determine the optimal k^* according to the criterion, i.e., satisfying the optimal value of the criterion function with k as well as the corresponding estimated parameters. Along this direction, many kinds of selection criteria have been established according to different theories or principles. However, the process of evaluating a criterion incurs a large computational cost since we need to repeat the entire parameter estimation process at a number of different values of k. Moreover, all these existing model selection criteria have their limitations and often lead to a wrong result.

In the field of artificial neural networks, competitive learning (CL) can be also applied to the finite mixture modeling in the way of clustering analysis in which each component is considered just as a cluster. However, conventional competitive learning algorithms such as classical competitive learning algorithm [8] and frequency sensitive competitive learning algorithm [9] can only be considered as adaptive versions of the k-means algorithm and thus are still unable to solve the compound mixture modeling problem we are interested in. Fortunately, Xu et al. [10] proposed a new kind of competitive learning algorithm, called the rival penalized competitive learning (RPCL) algorithm, that can automatically determine the number of clusters in a dataset via a learning and de-learning mechanism. In some extended versions of the RPCL algorithm [11], clusters were generalized to be the probability distributions like Gaussians and the associated clustering problems became more similar to the compound mixture modeling problem. Furthermore, a cost function theory [12] was established for the Distance Sensitive RPCL (DSRPCL) algorithm as a generalized version of the original

* Corresponding author.
E-mail address: jwma@math.pku.edu.cn (J. Ma).

http://dx.doi.org/10.1016/j.neucom.2014.04.080
0925-2312/© 2014 Elsevier B.V. All rights reserved.

RPCL algorithm and the DSRPCL algorithm was further generalized to the new version with the Mahalanobis distance [13].

Although those RPCL algorithms have already made many successful practical applications with the new feature of automated selection of clusters or components for a dataset, they can only provide an incomplete parameter estimation for the finite mixture model since the mixing proportions of components are not involved in the learning process. More precisely, Figueiredo and Jain [14] proposed an unsupervised learning algorithm for learning a finite mixture model with the ability of selecting the number of components through the competitive learning on the mixing proportions instead of the means vectors in the RPCL algorithm. Actually, it was established under the framework of the EM algorithm with the MML (Minimum Message Length) criterion [15] being applied to the mixing proportions such that the extra components are discarded as soon as their mixing proportions become small enough during the learning process. On the other hand, some variational Bayes approaches [16,17] have been proposed to Gaussian mixture learning with adaptive model selection, but it is difficult to extend them to the cases of non-Gaussian mixtures.

With the development of statistical learning, Bayesian Ying–Yang (BYY) harmony learning system and theory [18,19] have provided a new tool for solving this compound mixture modeling problem. Actually, it was already shown in [20] that the compound modeling problem for Gaussian mixtures can be solved through the maximization of a harmony function on a specific BI-directional architecture (BI-architecture) of the BYY system for the Gaussian mixture model via a gradient learning rule such that an appropriate number of Gaussians can be automatically allocated for a dataset, with the mixing proportions of extra Gaussians attenuating to zero. Later on, the adaptive, conjugate, natural gradient and fixed-point learning algorithms [21–23] were further established to improve the efficiency of the harmony function maximization. Moreover, an annealing learning algorithm was also established through the maximization of the harmony function on the back-directional architecture (B-architecture) of the BYY system for Gaussian mixture with automated model selection [24]. As for the cases of non-Gaussian mixtures, the gradient BYY learning algorithms have been already established for Poisson and Weibull mixtures [25,26].

In the current paper, we extend the BYY harmony learning mechanism of parameter learning with automated model selection to the case of logarithmic normal (log-normal) mixture, which is another typical non-Gaussian mixture. Under a BI-architecture of the BYY learning system for log-normal mixtures, an adaptive gradient learning algorithm for maximizing the harmony function is proposed to implement the parameter learning of log-normal mixture with automated model selection. It is demonstrated well by the experiments on both synthetic and real-world datasets that the proposed BYY harmony learning algorithm not only automatically determines the number of actual components, i.e., log-normal distributions, in the sample data, but also leads to a rather good estimation of the parameters in the original or true log-normal mixture.

The rest of this paper is organized as follows. We begin with the description of log-normal distribution and mixture in Section 2. According to the BYY harmony function, we then derive and construct the adaptive gradient BYY learning algorithm for log-normal mixtures and discuss its implementation in Section 3. The proposed BYY learning algorithm is demonstrated and compared by the experiments on both synthetic and real-world datasets in Section 4. Finally, we make a brief conclusion in Section 5.

2. Log-normal distribution and mixture

We begin with a brief introduction of univariate log-normal distribution. Supposing that $U \sim N(\mu, \sigma^2)$, i.e, U is a random variable subject to a Normal or Gaussian (probability) distribution with mean μ and variance σ^2, the random variable (or function) $X = \exp(U)$ is called to be subject to a (univariate) log-normal distribution. Similarly, we denote it by $X \sim LN(\mu, \sigma^2)$. In fact, the univariate log-normal distribution is often used to describe a heavy-tailed distribution of a non-negative random variable such as the age distribution of an element or instrument and a stock index distribution.

Mathematically, the univariate log-normal probability density takes the following form:

$$p(x|\mu,\sigma) = \frac{1}{\sigma x\sqrt{2\pi}} \exp\left\{ -\frac{(\ln x - \mu)^2}{2\sigma^2} \right\}, \quad x > 0. \tag{1}$$

This density is derived from the normal probability density and takes a similar form of it. However, they are essentially different. In order to explain the differences between log-normal and normal densities, we plot the curves of the two probability densities. Typically, we consider their standard probability densities, i.e., the parameters are set as $\mu = 0$, $\sigma = 1$. The curves of the standard univariate log-normal and normal probability densities are given in Fig. 1(a) and (b), respectively.

As shown in Fig. 1, the value of the log-normal probability density is positive only when $x > 0$. So, a random variable subject to the log-normal distribution is limited to be positive, which can match the requirement of many practical problems in the fields

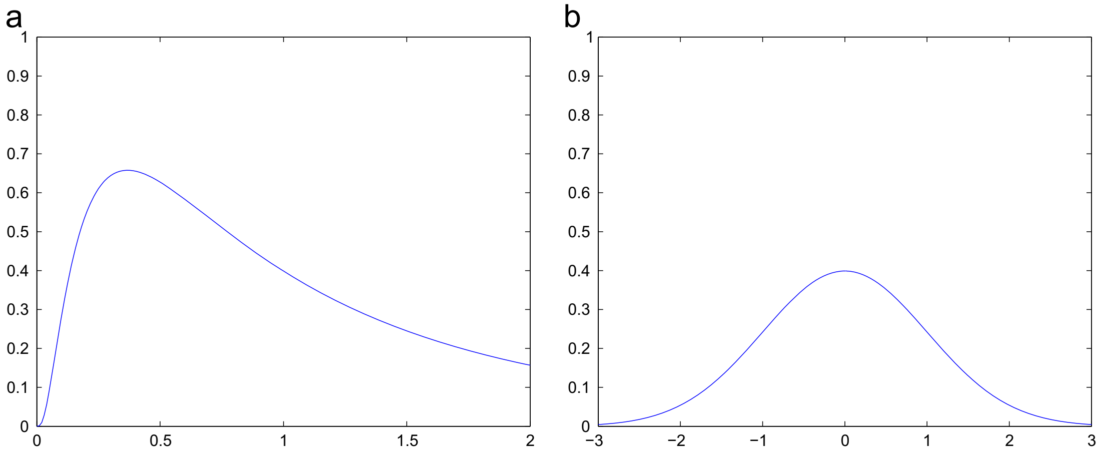


Fig. 1. The curves of univariate standard log-normal and normal probability densities. (a) The univariate standard log-normal probability density. (b) The univariate standard normal probability density.

such as economics, psychology and reliability analysis; while the value of the normal probability density is positive on the whole real field so that a random variable subject to the normal distribution is free. Moreover, the log-normal probability density is asymmetric. Actually, it increases rapidly and monotonously on $(0, e^{\mu-\sigma^2})$, reaches the maximum at $x = e^{\mu-\sigma^2}$, and then decreases gradually and monotonously on $(e^{\mu-\sigma^2}, +\infty)$, which makes the probability density be asymmetric. Actually, it increases rapidly and monotonously in the left interval (or region), but decreases gradually and monotonously in the right interval. Thus, the log-normal distribution complies with the actual properties of fatigue experiment and age distribution model to a large extent. On the other hand, the normal probability density is symmetric at its maximum point $u = \mu$ so that it decreases with $|u-\mu|$ at the same rate. Therefore, the log-normal and normal probability densities own different properties and correspond to different practical problems.

We further introduce the multivariate log-normal distribution. Let $U = [U_1, U_2, \ldots, U_n]$ be an n-dimensional (or n-D for short) random vector subject to the multivariate normal distribution with mean vector m and covariance matrix Σ. Through the transformation $X_i = \exp(U_i)$ for $i = 1, 2, \ldots, n$, we get the random vector (or function) $X = [X_1, X_2, \ldots, X_n]$. In the same way, the probability distribution of X is called the multivariate log-normal distribution. Similarly, we denote it by $X \sim LN(m, \Sigma)$. By mathematical derivation, we can get the multivariate log-normal probability density as follows:

$$p(x|m,\Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}\prod_{i=1}^{n} x_i} \exp\left\{ -\frac{1}{2}(\ln x - m)^T \Sigma^{-1} (\ln x - m) \right\}, \tag{2}$$

where $x_i > 0$, $i = 1, 2, \ldots, n$ and $\ln x = [\ln x_1, \ln x_2, \ldots, \ln x_n]^T$.

The multivariate log-normal distribution is an efficient tool for the studies of economics, psychology and reliability analysis where all the random variables are positive. In fact, the other multivariate distributions to describe a group of positive random variables, such as Gamma, Beta, Pareto and F distributions, are too complicated in their forms to be applied in these fields [27].

For comparison of the multivariate log-normal probability density with the multivariate normal probability density, we plot their curved surfaces in the standard forms in Fig. 2 for the 2-D case, i.e., $m = (0, 0)$, $\Sigma = \left[\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right]$.

As shown in Fig. 2, in each dimension, both the 2-D log-normal and normal probability distributions have the same characteristics as the univariate log-normal and normal probability distributions do, respectively. Specifically, the 2-D log-normal probability distribution takes the characteristics of the univariate log-normal probability distribution in each dimension. Therefore, it can be easily deduced that for the general multivariate case, the log-normal probability distribution owns the properties that each component of the corresponding random vector is positive and that the probability density is asymmetric and changes in the way of Fig. 1(a) in each dimension.

Finally, we turn to the log-normal mixture. In fact, it is a special type of the finite mixture model:

$$q(x|\Theta_k) = \sum_{j=1}^{k} \alpha_j q(x|\theta_j), \tag{3}$$

where $q(x|\theta_j)$ are the component probability densities or distributions with parameters θ_j, k is the number of components in the mixture, x denotes the variable or variable vector, and $\alpha_j \geq 0$ are mixing proportions of the components with the constraint that $\sum_{j=1}^{k} \alpha_j = 1$. For clarity, we let $\Theta_k = \{\alpha_j, \theta_j\}_{j=1}^{k}$ be the set of all parameters in the mixture model.

When $q(x|\theta_j)$ is just the log-normal probability density given by Eq. (2), i.e., $q(x|\theta_j) = p(x|m_j, \Sigma_j)$, the finite mixture model becomes the log-normal mixture as follows:

$$p(x|\Theta_k) = \sum_{j=1}^{k} \alpha_j p(x|\theta_j) = \sum_{j=1}^{k} \alpha_j p(x|m_j, \Sigma_j). \tag{4}$$

As the log-normal probability distribution is very common in practical applications, the log-normal mixture is often used in the fields of measurements, communications and finical analysis such as the distribution of red blood cell volume [28] and the modeling of the co-channel interference in wireless communication [29]. If the number k of log-normal components is known in advance, we can implement the EM algorithm [3] to estimate the parameters in the mixture with a dataset. As pointed in the previous section, the EM algorithm is constructed under a framework of maximum likelihood and thus unable to make model selection for log-normal mixture only with a set of sample data. In the following, based on the theory of BYY harmony learning, we will construct an adaptive gradient BYY learning algorithm for log-normal mixtures to make model selection automatically during parameter learning.

3. An adaptive gradient BYY learning algorithm for log-normal mixtures

For the compound finite mixture modeling, a BI-architecture of the BYY learning system has been established such that its BYY harmony learning is equivalent to the parameter learning with automated model selection on finite mixture [20,22,25]. Actually, given a sample dataset $D_x = \{x_t\}_{t=1}^{N}$ from the original finite mixture, the learning task on this architecture is to maximize the following harmony function on the finite mixture model Eq. (3) with the

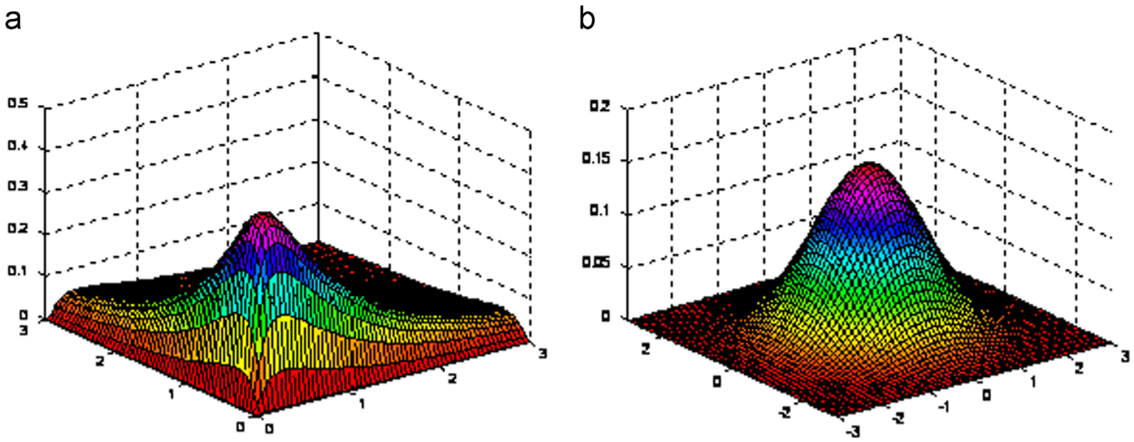


Fig. 2. The surfaces of the 2-D standard log-normal and normal probability densities. (a) The standard log-normal probability density. (b) The standard normal probability density.

parameter set $\Theta_k = \{\alpha_j, \theta_j\}_{j=1}^{k}$:

$$J(\Theta_k) = \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{k}\frac{\alpha_j q(x_t|\theta_j)}{\sum_{i=1}^{k}\alpha_i q(x_t|\theta_i)}\ln[\alpha_j q(x_t|\theta_j)]. \tag{5}$$

According to the experiments on the finite mixtures given in [20–26], the maximization of $J(\Theta_k)$ was able to make model selection automatically during parameter learning. In fact, as long as we set k to be larger than the number k^* of actual components (like Gaussians or Poissons) in the sample data, it can make k^* components from the estimated mixture match the actual components, and force the mixing proportions of the other $k-k^*$ extra components to attenuate to zero. In the same way, we now take $q(x|\theta_j)$ in Eq. (5) as a log-normal density $p(x|\theta_j) = p(x|m_j, \Sigma_j)$ given by Eq. (2) and construct an adaptive gradient BYY learning algorithm to maximize the harmony function $J(\Theta_k)$ for log-normal mixture.

3.1. Algorithm derivation

Let $U_j(x) = \alpha_j q(x|\theta_j)$ for $j = 1, 2, \ldots, k, J(\Theta_k)$ can be represented by the sum of the functions $J_t(\Theta_k), 1 \le t \le N$ as follows:

$$J(\Theta_k) = \frac{1}{N}\sum_{t=1}^{N} J_t(\Theta_k), \quad J_t(\Theta_k) = \sum_{j=1}^{k}\frac{U_j(x)}{\sum_{i=1}^{k}U_i(x_t)}\ln U_j(x_t). \tag{6}$$

For convenience of derivation, we utilize the following softmax representations of the mixing proportions α_j:

$$\alpha_j = \frac{\exp(\beta_j)}{\sum_{i=1}^{k}\exp(\beta_i)}, \quad j = 1, 2, \ldots k, \tag{7}$$

where $-\infty < \beta_j < +\infty, j = 1, 2, \ldots, k$. In such a way, α_j computed by Eq. (7) with any group β_j will certainly satisfy the conditions: $\alpha_j \ge 0$, $\sum_{j=1}^{k}\alpha_j = 1$.

With the above preparations, we can get the partial derivatives of $J_t(\Theta_k)$ per sample x_t with respect to β_j and θ_j as follows:

$$\begin{aligned}\frac{\partial J_t(\Theta_k)}{\partial \beta_j} &= \sum_{i=1}^{k}\frac{\partial J_t(\Theta_k)}{\partial U_i(x_t)}\frac{\partial U_i(x_t)}{\partial \beta_j}\\ &= \frac{1}{q(x_t|\Theta_k)}\sum_{i=1}^{k}\left[1-\sum_{l=1}^{k}(p(l|x_t)-\delta_{il})\ \ln U_l(x_t)\right](\delta_{ij}-\alpha_j)]U_i(x_t),\end{aligned} \tag{8}$$

$$\begin{aligned}\frac{\partial J_t(\Theta_k)}{\partial \theta_j} &= \sum_{i=1}^{k}\frac{\partial J_t(\Theta_k)}{\partial U_i(x_t)}\frac{\partial U_i(x_t)}{\partial \theta_j}\\ &= \frac{1}{q(x_t|\Theta_k)}\left[1-\sum_{l=1}^{k}(p(l|x_t)-\delta_{jl})\ \ln U_l(x_t)\right]\alpha_j\frac{\partial q(x_t|\theta_j)}{\partial \theta_j},\end{aligned} \tag{9}$$

where δ_{ij} is the Kronecker function and $p(l|x) = \alpha_l q(x|\theta_l)/q(x|\Theta_k)$.

Let $\lambda_i(t) = 1-\sum_{l=1}^{k}(p(l|x_t)-\delta_{il})\ln U_l(x_t)$ for $i = 1, \ldots, k$. According to Eqs. (8) and (9), we get the following adaptive gradient learning rule of β_j and θ_j:

$$\Delta\beta_j = \frac{\eta}{q(x_t|\Theta_k)}\sum_{i=1}^{k}\lambda_i(t)(\delta_{ij}-\alpha_j)U_i(x_t), \tag{10}$$

$$\Delta\theta_j = \frac{\eta\lambda_j(t)\alpha_j}{q(x_t|\Theta_k)}\frac{\partial q(x_t|\theta_j)}{\partial \theta_j} = \eta q(j|x_t)\lambda_j(t)\frac{\partial \ln q(x_t|\theta_j)}{\partial \theta_j}, \tag{11}$$

where $\eta > 0$ is the learning rate which starts from a reasonable initial value and decreases gradually to zero in the Robbin–Monro stochastic approximation manner [30].

For the log-normal mixture model in which $q(x|\theta_j) = p(x|m_j, \Sigma_j)$, $q(x|\Theta_k) = p(x|\Theta_k)$ are given by Eq. (4), we have the specific partial derivatives of the log-normal density function with respect to m_j and Σ_j as follows:

$$\frac{\partial p(x_t|m_j, \Sigma_j)}{\partial m_j} = p(x_t|m_j, \Sigma_j)\Sigma_j^{-1}(\ln x_t - m_j), \tag{12}$$

$$\frac{\partial p(x_t|m_j, \Sigma_j)}{\partial \Sigma_j} = \frac{1}{2}p(x_t|m_j, \Sigma_j)[\Sigma_j^{-1}(\ln x_t - m_j)(\ln x_t - m_j)^T - I_n]\Sigma_j^{-1}, \tag{13}$$

where I_n is an n-dimensional identity matrix. Substituting Eq. (12) into Eq. (11), we have the adaptive gradient learning rule of m_j:

$$\Delta m_j = \eta p(j|x_t)\lambda_j(t)\Sigma_j^{-1}(\ln x_t - m_j). \tag{14}$$

On the other hand, if we directly use the adaptive derivative with respect to Σ_j given in Eq. (13) for the gradient learning, Σ_j cannot be guaranteed to be always positive definite after each iteration. In order to overcome this difficulty, we can utilize the decomposition technique suggested in [21]: $\Sigma_j = B_j B_j^T$, where B_j is a nonsingular square matrix. Via this decomposition, we can get the following adaptive gradient learning rule of B_j:

$$\Delta vecB_j = \frac{\eta}{2}p(j|x_t)\lambda_j(t)vec\left\{\left[\Sigma_j^{-1}(\ln x_t - m_j)(\ln x_t - m_j)^T - I\right]\Sigma_j^{-1}\right\}\frac{\partial(B_j B_j^T)}{\partial B_j}, \tag{15}$$

where $vec(M)$ denotes the vector obtained by stacking the column vectors of the matrix M and the detailed expression of $\partial(B_j B_j^T)/\partial B_j$ can be found in [21].

Summing up the results of Eqs. (10), (14) and (15), we get the basic adaptive gradient learning algorithm for maximizing the harmony function on log-normal mixture. In each learning or updating iteration with sample x_t, β_j, m_j and B_j will be adaptively updated. Accordingly, α_j and Σ_j will be also updated. As long as the difference of the harmony functions at the two sequential steps is small enough, the gradient learning algorithm will stop and output the current values of the parameters in the log-normal mixture.

3.2. Algorithm implementation

We further improve our proposed adaptive gradient learning algorithm for log-normal mixtures and make it to be implemented efficiently. Firstly, we need to set an initial value k for the number of log-normal densities in the mixture which is greater than the true number k^* of actual log-normal densities in the original mixture or sample data. That is, we need to overestimate the number of log-normal densities in the dataset. Clearly, this is an easy task, but we should avoid making a large overestimate. The initial values of β_j can be set arbitrarily. However, the initial values of m_j can be selected from the logarithms of the samples in the dataset. More efficiently, they can be learned via a RPCL procedure [12]. As for the covariance matrix Σ_j, since the decomposition $\Sigma_j = B_j B_j^T$ is used, we need only to set the initial value of each B_j as an arbitrary nonsingular square matrix. In order to guarantee Σ_j being always nonsingular, we can add a check and revision procedure in the update of Σ_j such that, supposing that $\Sigma_j = (\sigma_{il}^j)_{d\times d}$, if $|\Sigma_j| < 10^{-14}$, we let $\sigma_{ii} = \sum_{l=1}^{d}|\sigma_{il}^k|$ directly.

In the above setting, there are $k-k^*$ extra components in the mixture model of the algorithm. In order to speed up the convergence of the algorithm, we can add a check and deleting procedure in the iteration such that some extra components can be immediately discarded if their mixing proportions are low enough and, on the other hand, we can also combine two similar components into a new component to eliminate one extra component. To realize the above idea, we start to set, by the prior experimental experiences, two threshold values: $Threshhold_\alpha$, and Γ_m, which respectively represent the threshold value of the component with too low mixing proportion and the threshold value of the difference between two mean vectors which are close enough to be combined. With these preparations, we can conduct the following two simple schemes at each iteration of the algorithm:

Scheme 1: Find out $\alpha_{j^*} = \arg\min_{1 \le j \le k} \alpha_j$ (the component with the lowest mixing proportion). If $\alpha_{j*} < Threshold_\alpha$, we delete this component and modify the other parameters and probability densities if necessary.

Scheme 2: Find out $\gamma_{i^*j^*} = \min_{i \ne j} \|m_i - m_j\|$. If $\gamma_{i^*j^*} < \Gamma_m^{-1}$, the components i^*&j^* are considered similar enough to be combined. The parameters of the new component, i.e., $\alpha_{new}, \beta_{new}, m_{new}, B_{new}$, can be computed as follows:

$$\alpha_{new} = \alpha_{i^*} + \alpha_{j^*}, \tag{16}$$

$$\beta_{new} = \ln\left[\frac{\alpha_{i^*} + \alpha_{j^*}}{\alpha_{i^*}} \exp(\beta_{i^*})\right], \tag{17}$$

$$m_{new} = \frac{\alpha_{i^*}}{\alpha_{i^*} + \alpha_{j^*}} m_{i^*} + \frac{\alpha_{j^*}}{\alpha_{i^*} + \alpha_{j^*}} m_{j^*}, \tag{18}$$

$$B_{new} = r\left(\frac{\alpha_{i^*}}{\alpha_{i^*} + \alpha_{j^*}} B_{i^*} + \frac{\alpha_{j^*}}{\alpha_{i^*} + \alpha_{j^*}} B_{j^*}\right), \tag{19}$$

where r represents an adjusting scale factor for the covariance matrix of the combined component, generally satisfying $1 < r \le 2$.

Combining these two schemes in the algorithm, we can obtain a flexible adjustment of k during the BYY harmony learning such that the algorithm is not only speeded up, but also insensitive to the initial setting of k even if it is any integer in the interval $[k^*, 3k^*]$. For the other parameters in the log-normal mixture, a better initialization is also necessary. We set the learning rate by $\eta(t) = \eta_0(1/t^p)$, where t is the time, η_0 can be set around 0.001, and the exponent p can be set around 0.05. Actually, if η_0 and p are set to be greater, the algorithm will converge too quickly with a wrong result. If they are set to be smaller, the algorithm will consume too much time. For the stop criterion: $|\Delta J| = |J(\Theta_k^{new}) - J(\Theta_k^{old})| < Threshold_J$, the threshold value $Threshold_J$ was generally set by 10^{-5} in the BYY learning algorithms for Gaussian mixtures [20,22]. But in our experiments for log-normal mixtures, such a value may be too large. In fact, as long as $|\Delta J| < 10^{-6}$, the change of the parameters is small enough to be ignored. Thus, we can set $Threshold_J$ to be 10^{-6}, which can guarantee the learning precision as well as the learning efficiency to our demand.

$Threshold_\alpha$ is an important threshold value which can considerably affect the learning efficiency of model selection. If it is too large, some components that should be kept will be wrongly discarded. Otherwise, if it is too small, the effect of the discarding extra components is not obvious. According to our experimental results, $Threshold_\alpha$ can be set around 0.05. For the imbalanced cases where mixing proportions of certain actual proportions are very small, $Threshold_\alpha$ can be set to be much smaller. Γ_m is another threshold value for model selection. If it is too small, some actual components will be combined. Otherwise, if it is too large, some extra components will not be combined and deleted. According to our experimental results, we have found that its reasonable range is 0.4–0.7. Generally, if Γ_m is set to be 0.6, the algorithm can effectively find those similar components to be combined and prevent the extra computation.

For clarity, we finally summarize our improved adaptive gradient learning algorithm for log-normal mixtures in Fig. 3.

4. Experimental results

In this section, the experiments on both various synthetic and real-world datasets are carried out to demonstrate the performance of our proposed adaptive gradient BYY learning algorithm for log-normal mixtures on both model selection and parameter estimation, with being compared with those of the other existing learning algorithms.

4.1. Simulation results

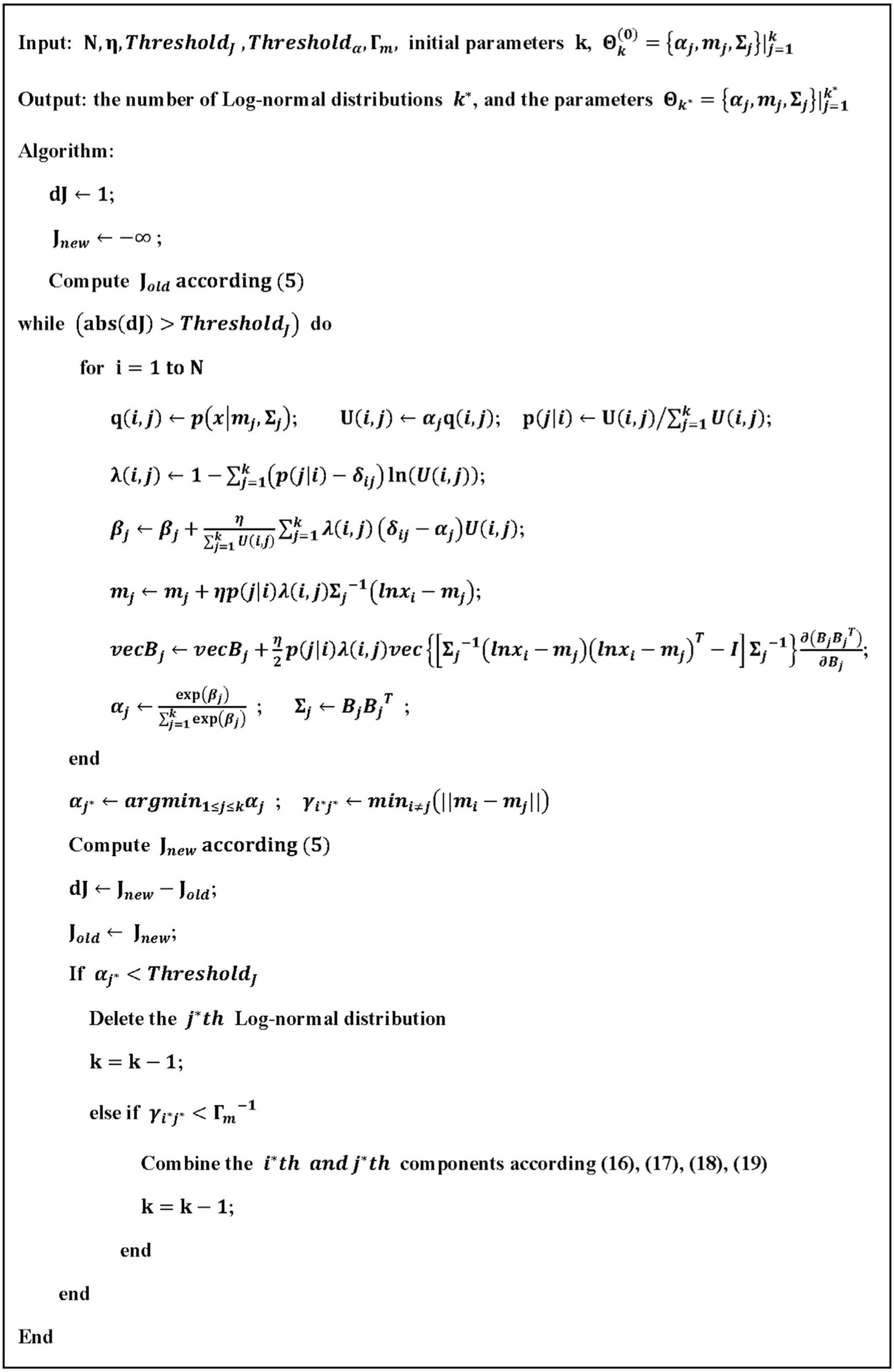
4.1.1. On 2-D synthetic datasets

We begin to test the efficiency of the proposed algorithm on four typical 2-D synthetic datasets, which are shown in Fig. 4. In fact, they are generated from four typical log-normal mixtures with different structures and overlaps among the components: (a) $\mathcal{S}_1$ is generated from a mixture of 4 log-normal distributions with the same covariance matrix and mixing proportions; (b) $\mathcal{S}_2$ is generated from a mixture of 4 log-normal distributions with different covariance matrix and mixing proportions; (c) $\mathcal{S}_3$ is generated from a mixture of 3 log-normal distributions with special shapes and different mixing proportions; (d) $\mathcal{S}_4$ is generated from a mixture being similar to that of $\mathcal{S}_2$, but only with a small number of samples. For visualization, two axes in Fig. 4 are both in log measure. The true parameters of these four datasets are listed in Table 1.

We implement our proposed algorithm on these four 2-D datasets with the initial parameters selected as discussed previously in Section 3.2. Specifically, k is set to be 7 or 6 and the initial mean vectors are set via a RPCL procedure of 200 iterations on the logarithms of the samples in the corresponding dataset. In order to test the ability of correct model selection, we implement the proposed algorithm on each of the datasets for 500 times and get the correct model selection percentage over 500 simulation results. Actually, the correct model selection percentages of the proposed algorithm on the four datasets are listed in the second row of Table 2. According to these data, we can find that the proposed algorithm has a rather high probability of making correct model selection automatically on these datasets. But it can be also found that the correct model selection percentage can only reach 100% at $\mathcal{S}_1$. As the shapes of components in the dataset become elliptical from $\mathcal{S}_1$ to $\mathcal{S}_3$, or the number of samples becomes small like $\mathcal{S}_4$, the correct model selection percentage decreases slowly but obviously.

On the other hand, we can further find that when model selection is made correctly, the actual log-normal densities as well as their mixing proportions are correctly located by the corresponding estimated or learned parameters. That is, the true parameters of the actual log-normal densities, as listed in Table 1, are estimated with a quite good accuracy, which can be seen from Table 3 where the average estimates of the true parameters with the standard deviation over 100 simulation results of correct model selection for those 2-D synthetic datasets are listed. Again, it can be still found that the estimation accuracy of some parameters may decrease as the components in the dataset become elliptical or the number of samples becomes small. The deviation between the estimated and true parameters will be further discussed later.

We further consider six heterogeneous 2-D or higher dimensional synthetic datasets of log-normal mixtures with a larger number of components of different elliptical shapes (i.e.,covariance matrixes) or a higher overlap among the components shown in Fig. 5. Specifically, $\mathcal{D}_1$–$\mathcal{D}_4$ are 2-D synthetic datasets with 6 or 9 components, being elliptical differently, imbalanced and overlapped. $\mathcal{D}_5$ and $\mathcal{D}_6$ are 3-D and 4-D synthetic datasets composed of incompact and imbalanced components, which can be seen clearly from Fig. 5. Both of these two higher dimensional datasets consist of four components, and their mixing proportion vector or distributions are (0.1, 0.7, 0.1, 0.1) and (0.08, 0.17, 0.5, 0.25), respectively. We implement our proposed algorithm on each of these six heterogeneous datasets for 100 times and get the correct model selection percentages over 100 experimental results, being listed in the second row of Table 4. According to these percentages,



```
Input: $N, \eta, Threshold_J, Threshold_\alpha, \Gamma_m$, initial parameters k, $\Theta_k^{(0)} = \{\alpha_j, m_j, \Sigma_j\}|_{j=1}^{k}$

Output: the number of Log-normal distributions $k^*$, and the parameters $\Theta_{k^*} = \{\alpha_j, m_j, \Sigma_j\}|_{j=1}^{k^*}$

Algorithm:
    $dJ \leftarrow 1$;
    $J_{new} \leftarrow -\infty$ ;
    Compute $J_{old}$ according (5)
while $(abs(dJ) > Threshold_J)$ do
        for i = 1 to N
            $q(i,j) \leftarrow p(x|m_j, \Sigma_j)$;    $U(i,j) \leftarrow \alpha_j q(i,j)$;    $p(j|i) \leftarrow U(i,j)/\sum_{j=1}^{k} U(i,j)$;
            $\lambda(i,j) \leftarrow 1 - \sum_{j=1}^{k}(p(j|i) - \delta_{ij})\ln(U(i,j))$;
            $\beta_j \leftarrow \beta_j + \frac{\eta}{\sum_{j=1}^{k} U(i,j)} \sum_{j=1}^{k} \lambda(i,j)\,(\delta_{ij} - \alpha_j)U(i,j)$;
            $m_j \leftarrow m_j + \eta p(j|i)\lambda(i,j)\Sigma_j^{-1}(lnx_i - m_j)$;
            $vecB_j \leftarrow vecB_j + \frac{\eta}{2} p(j|i)\lambda(i,j) vec\left\{\left[\Sigma_j^{-1}(lnx_i - m_j)(lnx_i - m_j)^T - I\right]\Sigma_j^{-1}\right\}\frac{\partial(B_j B_j^T)}{\partial B_j}$;
            $\alpha_j \leftarrow \frac{\exp(\beta_j)}{\sum_{j=1}^{k}\exp(\beta_j)}$ ;    $\Sigma_j \leftarrow B_j B_j^T$ ;
        end
        $\alpha_{j^*} \leftarrow argmin_{1 \le j \le k} \alpha_j$ ;    $\gamma_{i^* j^*} \leftarrow min_{i \ne j}(||m_i - m_j||)$
        Compute $J_{new}$ according (5)
        $dJ \leftarrow J_{new} - J_{old}$;
        $J_{old} \leftarrow J_{new}$;
        If $\alpha_{j^*} < Threshold_J$
            Delete the $j^*th$ Log-normal distribution
            $k = k - 1$;
            else if $\gamma_{i^* j^*} < \Gamma_m^{-1}$
                Combine the $i^*th$ and $j^*th$ components according (16), (17), (18), (19)
                $k = k - 1$;
            end
    end
End
```

Fig. 3. The pseudo-code of the adaptive gradient learning algorithm for log-normal mixtures.

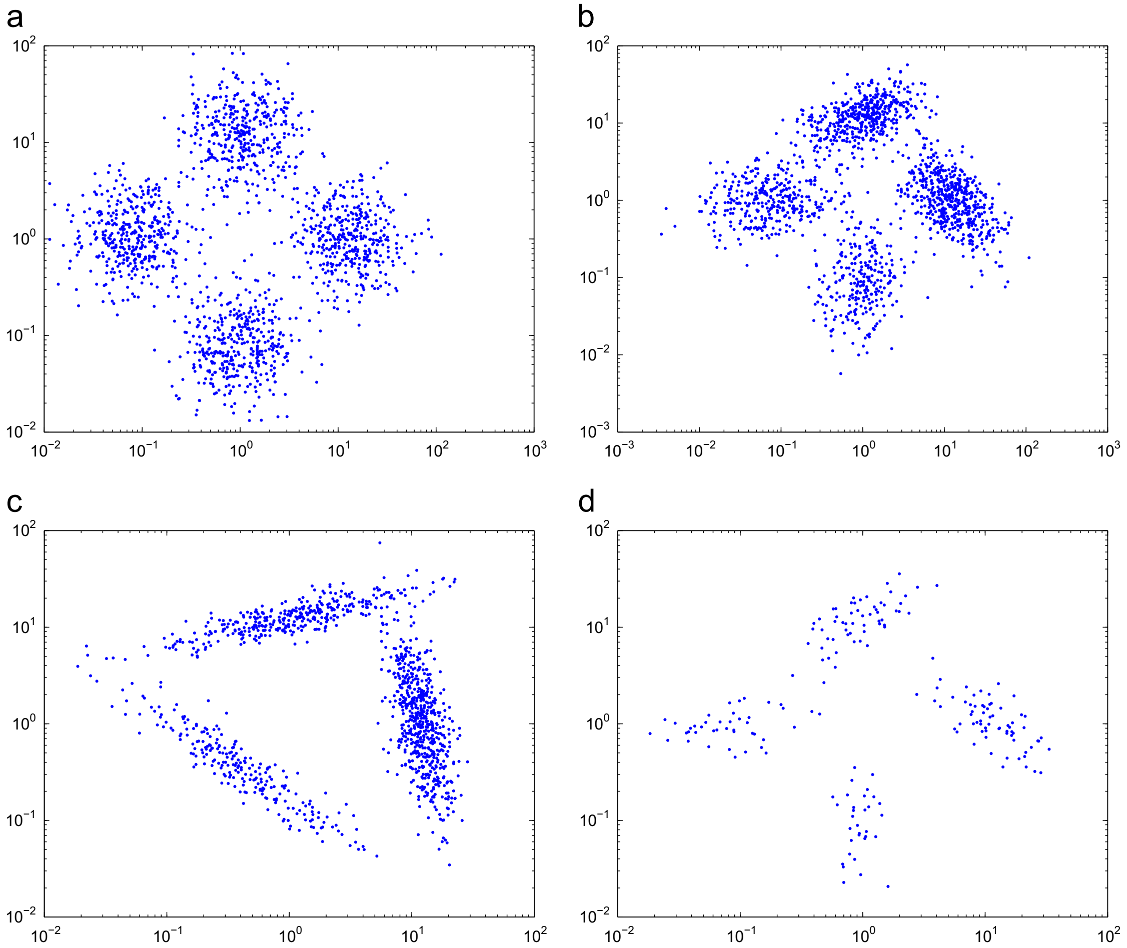


Fig. 4. The sketches of four typical 2-D synthetic datasets of log-normal mixtures. (a) Set $\mathcal{S}_1$. (b) Set $\mathcal{S}_2$. (c) Set $\mathcal{S}_3$. (d) Set $\mathcal{S}_4$.

Table 1
The true parameters of log-normal mixtures to generate four typical 2-D synthetic datasets.

Set	j	m_{j1}	m_{j2}	σ^j_{11}	σ^j_{12}	σ^j_{22}	α_j	N_j
$\mathcal{S}_1$ ($N=1600$)	1	2.50	0	0.50	0	0.50	0.25	400
	2	0	2.50	0.50	0	0.50	0.25	400
	3	−2.50	0	0.50	0	0.50	0.25	400
	4	0	−2.50	0.50	0	0.50	0.25	400
$\mathcal{S}_2$ (N=1600)	1	0.25	0	0.45	−0.25	0.55	0.34	544
	2	0	2.50	0.65	0.20	0.25	0.28	448
	3	−0.25	0	1	0.10	0.35	0.22	352
	4	0	−2.50	0.30	0.15	0.80	0.16	256
$\mathcal{S}_3$ ($N=1200$)	1	0.25	0	0.10	−0.20	1.25	0.50	600
	2	0	2.50	1.25	0.35	0.15	0.30	360
	3	−1	−1	1	−0.80	0.75	0.20	240
$\mathcal{S}_4$ ($N=200$)	1	0.25	0	0.28	−0.20	0.32	0.34	68
	2	0	2.50	0.34	0.20	0.22	0.28	56
	3	−0.25	0	0.50	0.04	0.12	0.22	44
	4	0	−2.50	0.10	0.05	0.50	0.16	32

Table 2
The correct model selection percentages of the AGL-BYY, UL-MML, EM-AIC and EM-BIC algorithms for log-normal mixtures on four 2-D synthetic datasets.

Algorithm	$\mathcal{S}_1$ (%)	$\mathcal{S}_2$ (%)	$\mathcal{S}_3$ (%)	$\mathcal{S}_4$ (%)
AGL-BYY	100	99.8	99.4	98
UL-MML	94.4	73.4	90.2	60.6
EM-AIC	74.6	50	91	56
EM-BIC	73.4	63.4	99.8	56.2

it can be found that there is still a high probability over or equal to 0.95 for the event that the actual log-normal densities in such a heterogeneous dataset can be located correctly, with the extra components discarded or combined. That is, model selection can be made automatically and correctly in each case with a rather high probability. As for parameter estimation, the average estimation accuracy for each parameter can be still maintained relatively high except for the case of the dataset (b) in Fig. 5. The major reason is that in such a dataset, there is a heavy overlap among the actual components. Particularly in this case, it is found by a simulation experiment that the resulted mixing proportion vector $\hat{\alpha}$ is (0.213631, 0.218454, 0.144483, 0.166157, 0.156863, 0.100411), while the true mixing proportion vector is (0.2, 0.2, 0.15, 0.15, 0.15, 0.15). It can be seen from this result that the parameter estimation is not as good as those on $\mathcal{S}_1$–$\mathcal{S}_4$, but we can still accept it in such a component-overlapping situation.

As a result, on each of above synthetic datasets which can be heterogeneous or higher dimensional, our proposed algorithm can make model selection automatically and lead to a rather good estimation of the parameters in the original log-normal mixture. Actually, the components in such an original log-normal mixture can be of various forms like the simple cases with the same covariance matrix and number of samples, or the complicated cases with special shapes, different covariance matrixes or mixing proportions, and the special case with a small number of samples.

Table 3
The average estimates of the true parameters over 100 simulation results on each of the four 2-D synthetic datasets.

Dataset	j	$\hat{m}_{j1}$	$\hat{m}_{j2}$	$\hat{\sigma}^j_{11}$	$\hat{\sigma}^j_{12}$	$\hat{\sigma}^j_{22}$	$\hat{\alpha}_j$
$\mathcal{S}_1$ $(k=7)$	1	2.57671 ± 0.0001	0.00147 ± 0.0002	0.38112 ± 0.0003	-0.02312 ± 0.0000	0.44812 ± 0.0002	0.25127 ± 0.0001
	2	0.07451 ± 0.0001	2.48950 ± 0.0000	0.44112 ± 0.0001	-0.01251 ± 0.0001	0.51156 ± 0.0001	0.25048 ± 0.0000
	3	-2.53604 ± 0.0001	0.07809 ± 0.0001	0.43553 ± 0.0001	-0.00046 ± 0.0000	0.45871 ± 0.0000	0.24568 ± 0.0000
	4	-0.03857 ± 0.0000	-2.54824 ± 0.0000	0.44492 ± 0.0001	0.02555 ± 0.0001	0.48871 ± 0.0002	0.25258 ± 0.0001
$\mathcal{S}_2$ $(k=7)$	1	2.51518 ± 0.0000	-0.03558 ± 0.0001	0.44813 ± 0.0003	-0.23685 ± 0.0001	0.54803 ± 0.0002	0.35193 ± 0.0003
	2	0.02054 ± 0.0000	2.50833 ± 0.0001	0.64644 ± 0.0002	0.18217 ± 0.0001	0.273334 ± 0.0002	0.28769 ± 0.0001
	3	-2.60646 ± 0.0000	-0.01583 ± 0.0001	0.92499 ± 0.0002	0.09664 ± 0.0006	0.34025 ± 0.0001	0.20519 ± 0.0002
	4	-0.03635 ± 0.0001	-2.46782 ± 0.0002	0.32555 ± 0.0002	0.08628 ± 0.0001	0.73543 ± 0.0001	0.15519 ± 0.0001
$\mathcal{S}_3$ $(k=6)$	1	2.46080 ± 0.0003	-0.00712 ± 0.0003	0.10111 ± 0.0003	-0.18439 ± 0.0003	1.20246 ± 0.0010	0.51718 ± 0.0004
	2	0.11040 ± 0.0007	2.53496 ± 0.0002	1.42775 ± 0.0008	0.39375 ± 0.0001	0.16328 ± 0.0001	0.29307 ± 0.0002
	3	-0.94521 ± 0.0012	-1.02946 ± 0.0011	1.17979 ± 0.0008	-0.96792 ± 0.0004	0.92859 ± 0.0009	0.18975 ± 0.0002
$\mathcal{S}_4$ $(k=7)$	1	2.42636 ± 0.0001	0.02801 ± 0.0000	0.30700 ± 0.0001	-0.20770 ± 0.0000	0.31812 ± 0.0001	0.34129 ± 0.0000
	2	-0.07591 ± 0.0000	2.44580 ± 0.0000	0.31994 ± 0.0000	0.20267 ± 0.0000	0.28346 ± 0.0001	0.29024 ± 0.0000
	3	-2.50147 ± 0.0000	-0.08329 ± 0.0001	0.50699 ± 0.0003	0.05034 ± 0.0001	0.11407 ± 0.0000	0.20926 ± 0.0000
	4	-0.07065 ± 0.0000	-2.38612 ± 0.0000	0.06242 ± 0.0000	0.00654 ± 0.0001	0.54708 ± 0.0000	0.15921 ± 0.0000

4.1.2. Comparison with the other possible approaches

Furthermore, we compare our proposed algorithm with the MML-based unsupervised learning algorithm [14] particularly for log-normal mixtures, which has been considered as a typical and powerful existing learning algorithm for the finite mixture modeling with automated model selection in a similar way. For convenience, our proposed algorithm is referred to as the AGL-BYY algorithm, while the unsupervised learning algorithm for log-normal mixtures based on the MML criterion is referred to as the UL-MML algorithm. We also compare our proposed algorithm with the EM algorithms for log-normal mixtures together with the AIC and BIC model selection criteria, which are referred to as the EM-AIC and EM-BIC algorithms, respectively.

As for the comparison on model selection, we also implement the UL-MML, EM-AIC and EM-BIC algorithms on each of the four 2-D typical synthetic datasets $\mathcal{S}_1$–$\mathcal{S}_4$ for 500 times and compute the correct model selection percentage over 500 experimental results, being listed in Table 2. Moreover, we implement the UL-MML, EM-AIC and EM-BIC algorithms on each of the six heterogeneous datasets $\mathcal{D}_1$–$\mathcal{D}_6$ for 100 times and compute the correct model selection percentage over 100 experimental results, being listed in Table 4. On the other hand, we further compute the average parameter estimation errors and running times of the AGL-BYY, UL-MML, EM-AIC and EM-BIC algorithms over 100 experimental results of correct model selection on each of the four typical synthetic datasets, being listed in Table 5, where for a parameter θ, $\Delta\theta = \|\theta^* - \hat{\theta}\|$, where θ^* is its true value, while $\hat{\theta}$ is its estimated value via the learning algorithm. The experiments are carried out on a server with HP Z820 (64G mem, 16 cores and 32 threads, CPU Intel Xeon E5-26500@2.00 GHZ).

According to those data listed in Tables 2, 4 and 5, we have found the following facts: (1) As for model selection, the AGL-BYY algorithm outperforms the UL-MML, EM-AIC and EM-BIC algorithms considerably on many aspects. Actually, as the dataset becomes more incompact, imbalaced and overlapped or has the less number of samples, the AGL-BYY algorithm performs much better than these three compared algorithms. In some synthetic datasets with elliptic components or a small number of samples, the AGL-BYY algorithm can determine the correct number of components, but the UL-MML algorithm often leads to a wrong number of components, which is consistent with the experimental results of our recently proposed competitive EM algorithm for Gaussian mixtures using the BYY harmony criterion instead of the MML criterion [31]. (2) For parameter estimation, the convergence results of the four learning algorithms are similar. That is, the deviations of any two resulted estimations are almost same. But the deviation of the AGL-BYY algorithm is slightly larger than those of the UL-MML, EM-AIC and EM-BIC algorithms, being equivalent to the EM algorithm at the final stage with the number of the components being correctly selected. This is reasonable since the AGL-BYY algorithm is involved in the competitive learning for automated model selection and thus produces certain deviation, while the EM algorithm leads to a maximum likelihood estimate which is consistent with the true parameter. (3) The convergence speed of the UL-MML algorithm is much faster than that of the AGL-BYY algorithm, which may be caused by the model selection mechanism in the AGL-BYY algorithm that incurs a larger computation. But the convergence speed of the EM-AIC and EM-BIC algorithms is much slower than that of the AGL-BYY algorithm. As model selection is so important here, we can think that our proposed algorithm is more efficient and powerful than the MML-based unsupervised learning algorithm for log-normal mixtures.

On the other hand, according to probability theory, if X is a normal random variable, by definition, $Y = \exp(X)$ becomes a log-normal random variable. Oppositely, if Y is a log-normal random variable, its log transformation, i.e., $\log(Y)$, becomes a normal random variable. Because each datum generated from a log-normal mixture is generated from one of the log-normal distributions, its log transformation can be considered being generated from a normal or Gaussian distribution. So, we can transform the sample data of a log-normal mixture to be those of a Gaussian mixture via the log transformation. In this way, we can implement a BYY learning algorithm for Gaussian mixtures on the transformed dataset to get the log-normal mixture learning, i.e., the model selection and parameter estimation for log-normal mixture. Is this normalization transformation approach better than our proposed adaptive gradient BYY learning algorithm? In order to answer this question, we compare our proposed algorithm with the fixed-point BYY learning algorithm [23], a typical BYY learning algorithm for Gaussian mixtures, on the transformed dataset. It is found by the experimental results that our proposed algorithm is much better than the normalization transformation approach. The reason can be explained as follows. As the normalization transformation, i.e., log transformation, makes the scale of a variable smaller, the overlap among the components in the transformed data becomes heavier than that in the original data. Since the BYY harmony learning becomes weaker on both model selection and parameter estimation as the overlap among the components in the

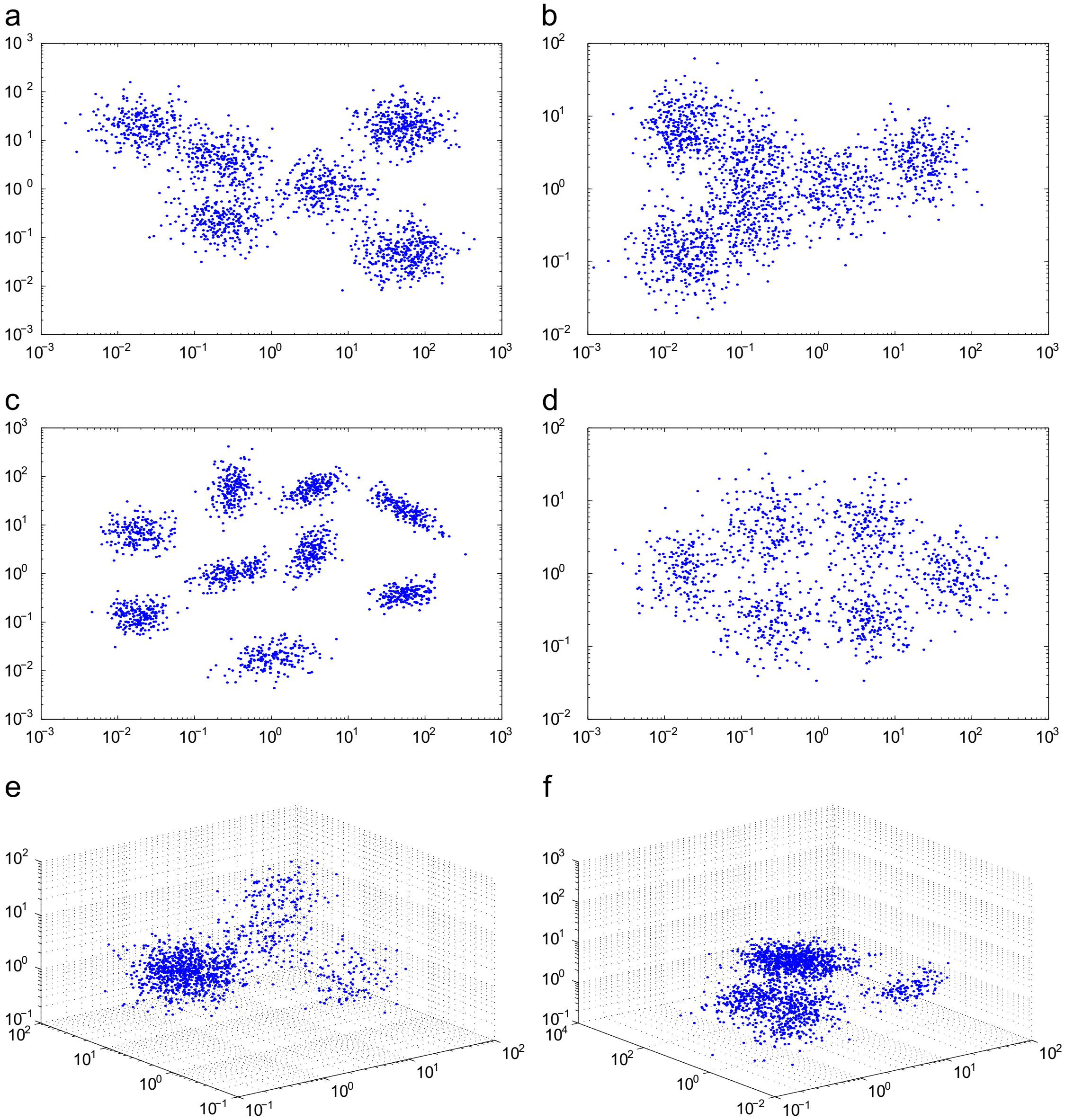


Fig. 5. The sketches of six heterogeneous datasets, where a–d are four 2-D synthetic datasets of log-normal mixtures with 6 or 9 components bing elliptical,imbalanced and overlapped, while e and f are 3-D and 4-D synthetic datasets with incompact and imbalanced components. (a) Set $\mathcal{D}_1$. (b) Set $\mathcal{D}_2$. (c) Set $\mathcal{D}_3$. (d) Set $\mathcal{D}_4$. (e) Set $\mathcal{D}_5$. (f) Set $\mathcal{D}_6$.

Table 4
The correct model selection percentages of the AGL-BYY, UL-MML, EM-AIC and EM-BIC algorithms for log-normal mixtures on six heterogeneous datasets.

Algorithm	$\mathcal{D}_1$ (%)	$\mathcal{D}_2$ (%)	$\mathcal{D}_3$ (%)	$\mathcal{D}_4$ (%)	$\mathcal{D}_5$ (%)	$\mathcal{D}_6$ (%)
AGL-BYY	99	98	99	100	95	100
UL-MML	54	80	62	77	40	79
EM-AIC	13	21	6	60	14	8
EM-BIC	40	27	10	70	27	35

sample data becomes heavier, the proposed algorithm is thus better than the normalization transformation approach.

4.1.3. Further discussions

We finally discuss the performance of the adaptive gradient BYY learning algorithm on the general datasets. For clarity, we begin to discuss the conditions by which the proposed algorithm can lead to the correct model selection. In fact, this problem was already investigated theoretically on the general finite mixture model in [32] and it was found that the correct convergence of a BYY harmony learning algorithm strongly depends on the overlap among the actual components in the original or true mixture. Specifically, when the overlap is low enough, the BYY harmony learning algorithm can lead to the correct model selection. So, as long as the overlap among the actual log-normal components in the original mixture keeps a low level, our proposed algorithm can converge with correct model selection. Oppositely, when any two actual log-normal distributions are strongly overlapped, our proposed algorithm may lead to a wrong result. This fact is demonstrated well by the above experiments as well as the other experiments on the different kinds of datasets. Moreover, it can be observed that our proposed algorithm is more robust than the BYY harmony learning algorithms for Gaussian mixtures. Actually, our proposed algorithm can converge correctly in the datasets with a heavy overlap like the dataset (b) given in Fig. 5, but, in a dataset with such a level of overlap, the fixed-point BYY learning

Table 5
The comparison of the AGL-BYY, UL-MML, EM-AIC and EM-BIC algorithms on the parameter estimation accuracy and the running time.

Dataset	Algorithm	$\Delta\mu$	$\Delta\Sigma$	$\Delta\alpha$	Running time (s)
$\mathcal{S}_1$ ($k=7$)	AGL-BYY	0.045516 ± 0.00004	0.033537 ± 0.00005	0.002160 ± 0.00000	24,932.3
	UL-MML	0.041604 ± 0.00006	0.025112 ± 0.00004	0.000643 ± 0.00003	6305.5
	EM-AIC	0.041604 ± 0.00000	0.025117 ± 0.00000	0.000644 ± 0.00000	81,037.1
	EM-BIC	0.041604 ± 0.00000	0.025117 ± 0.00000	0.000644 ± 0.00000	102,800.4
$\mathcal{S}_2$ ($k=7$)	AGL-BYY	0.051155 ± 0.00021	0.037598 ± 0.00009	0.005112 ± 0.00003	27,753.7
	UL-MML	0.052818 ± 0.03764	0.036023 ± 0.04092	0.003530 ± 0.01071	7290.5
	EM-AIC	0.048994 ± 0.00000	0.031898 ± 0.00000	0.002421 ± 0.00000	85,315.5
	EM-BIC	0.048994 ± 0.00000	0.031898 ± 0.00000	0.002421 ± 0.00000	104,300.1
$\mathcal{S}_3$ ($k=6$)	AGL-BYY	0.046059 ± 0.00052	0.087686 ± 0.000368	0.011488 ± 0.00036	24,976.2
	UL-MML	0.042694 ± 0.00004	0.066887 ± 0.00011	0.002236 ± 0.00008	4514.7
	EM-AIC	0.042680 ± 0.00000	0.066939 ± 0.00000	0.001549 ± 0.00000	41,184.8
	EM-BIC	0.042680 ± 0.00000	0.066939 ± 0.00000	0.001549 ± 0.00000	51,370.6
$\mathcal{S}_4$ ($k=7$)	AGL-BYY	0.062630 ± 0.00002	0.021148 ± 0.00004	0.005767 ± 0.00000	6331.3
	UL-MML	0.066265 ± 0.04116	0.033451 ± 0.05162	0.009980 ± 0.01350	178.1
	EM-AIC	0.060372 ± 0.00000	0.0026096 ± 0.00000	0.004653 ± 0.00000	4619.8
	EM-BIC	0.060372 ± 0.00000	0.0026096 ± 0.00000	0.004653 ± 0.00000	5236.6

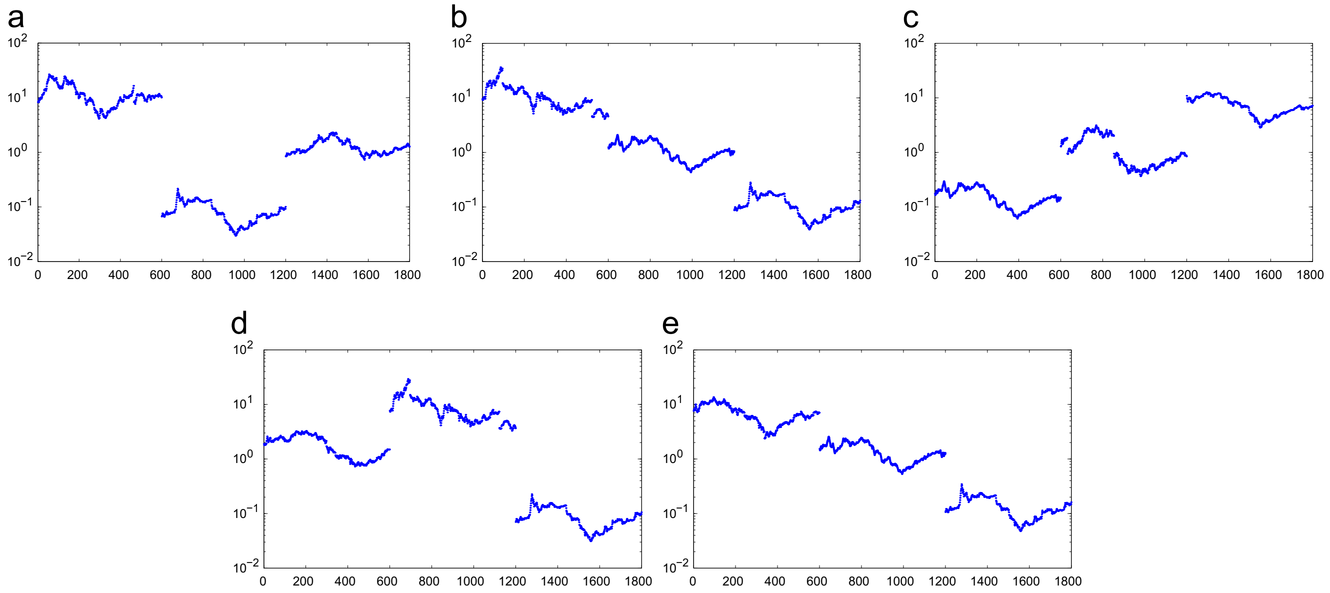


Fig. 6. The sketches of five stock index mixed datasets. (a) Dataset 1: $\mathcal{G}_1$, (b) Dataset 2: $\mathcal{G}_2$, (c) Dataset 3: $\mathcal{G}_3$, (d) Dataset 4: $\mathcal{G}_4$, (e) Dataset 5: $\mathcal{G}_5$.

algorithm [23] for Gaussian mixtures generally leads to a wrong result.

By the simulation experiments, we also find that the annihilation and combination mechanisms on the components not only speeds up the convergence of the algorithm, but also improve the convergent results of the algorithm, especially on the sample dataset with a relatively high overlap among the actual log-normal distributions. Indeed, there are many cases like the dataset (b) given in Fig. 5 on which the conventional adaptive gradient learning algorithm given only by Eqs. (10), (14) and (15) often leads to a wrong model selection, but our proposed adaptive gradient BYY learning algorithm with the annihilation and combination mechanisms can always get a good result. As for the threshold values for the annihilation and combination mechanisms, we can get their reasonable or optimal values by some searching techniques. It can be even found by simulation experiments that once they are optimally selected, they are applicable to a general dataset under the small overlap assumption and lead to a stable result.

By the other simulation experiments we further find that the proposed algorithm works well for both model selection and parameter estimation on the dataset with a large number components (e.g., 15 log-normal distributions) or a larger number of samples (e.g., $N=8000$), or being in a higher dimensional space (e.g., 15-dimensional space) as long as the actual log-normal distributions are separated at a degree as those in the above datasets. Furthermore, it can be found by the simulation experiments that the parameter estimation accuracy maintains a similar level with the increase of the space dimension or number of components in the mixture. As for the increase of number of samples in the dataset, the parameter estimation accuracy tends to be better.

In a summary, our proposed adaptive gradient BYY learning algorithm can be implemented efficiently for parameter estimation on log-normal mixture with automated model selection as long as the actual log-normal components in the sample data are separated in a certain degree. Moreover, it considerably outperforms the unsupervised learning algorithm [14] for log-normal mixtures on automated model selection. Thus, the BYY harmony learning can be applied to the log-normal mixtures just as it has been applied to the Gaussian mixtures.

4.2. On the classification of stock price indexes

For practical usage and test, we apply our proposed adaptive gradient BYY harmony learning algorithm for log-normal mixtures to the classification or recognition of stock price indexes and compare it with the adaptive gradient BYY harmony learning algorithm for Gaussian mixtures [22], which is considered as another typical Gaussian mixture modeling algorithm.

According to the general experiences in finance, the closing price indexes of a stock are probably subject to a log-normal distribution. Here, we use several stock's closing price indexes between 2007 and 2009 contained in Dazhihui Software as the real-world data subjecting to log-normal distributions and combine some of them together to form a dataset for a log-normal mixture. Fig. 6 shows five such datasets for our classification application. In each dataset, there are three components and each component consists of 600 closing price indexes of a stock. The statistics of each stock indexes in every normalized dataset are listed in Table 6.

To test the effectiveness of our proposed algorithm, we implement it on each of the five datasets. According to the converged or estimated log-normal mixture, we can classify each stock index x_t into a component or class via its maximum posterior, i.e., $\arg\max_j p(j|x_t)$. In such a way, the stock indexes in the mixed dataset are classified in an unsupervised mode. For comparison with the Gaussian mixture modeling approach, we also implement the adaptive gradient BYY learning algorithm for Gaussian mixtures on each of these five datasets and check the classification accuracies of the two algorithms. In our experiments on each dataset, from each stock indexes we typically choose 400 items randomly as training data and the other 200 items as testing data. So, together for every mixture, we have a training set of 1200 indexes and a testing set of 600 indexes.

Specifically, we implement the two algorithms on the training data of each dataset with $k=6$ (note that $k^*=3$). The initial parameters are set as discussed previously in Section 3.2. For this case, the initial mean values are selected via a short RPCL procedure on the logarithms of the training data. After the accomplishment of the training process for each algorithm, we obtain the estimated log-normal mixture and the estimated Gaussian mixture for fitting the dataset. According to the estimated mixture models over 100 experiments, we get the model selection percentages and classification accuracies of the two algorithms on the testing data, which are listed in Tables 7 and 8, respectively.

Table 6
The statistics of the three stock indexes in each of the five mixed datasets.

Dataset	j	Mean	Variance	Log Mean	Log Variance	α_j	N_j
$\mathcal{G}_1$	1	11.6875	27.2369	2.3634	0.1912	1/3	600
	2	0.0845	0.0013	−2.5623	0.1897	1/3	600
	3	1.2692	0.1417	0.1989	0.0754	1/3	600
$\mathcal{G}_2$	1	10.7743	34.7689	2.2539	0.2317	1/3	600
	2	1.1268	0.1847	0.0408	0.1648	1/3	600
	3	0.1105	0.0022	−2.2947	0.1897	1/3	600
$\mathcal{G}_3$	1	0.1590	0.0037	−1.9171	0.1648	1/3	600
	2	1.2010	0.5800	−0.0115	0.3835	1/3	600
	3	7.3920	7.3064	1.9286	0.1504	1/3	600
$\mathcal{G}_4$	1	1.8006	0.6366	0.4796	0.2285	1/3	600
	2	8.6521	22.4221	2.0345	0.2317	1/3	600
	3	0.0887	0.0014	−2.5141	0.1897	1/3	600
$\mathcal{G}_5$	1	6.9070	7.3857	1.8485	0.1786	1/3	600
	2	1.3799	0.2770	0.2435	0.1648	1/3	600
	3	0.1353	0.0033	−2.0920	0.1897	1/3	600

Table 7
The correct model selection percentages of the adaptive gradient BYY harmony learning algorithms (refer to AGL-BYYs) for log-normal and Gaussian mixtures on the five stock index mixed datasets.

Algorithm	$\mathcal{G}_1$ (%)	$\mathcal{G}_2$ (%)	$\mathcal{G}_3$ (%)	$\mathcal{G}_4$ (%)	$\mathcal{G}_5$ (%)
AGL-BYY for log-normal mixtures	100	99	98	98	98
AGL-BYY for Gaussian mixtures	91	98	96	97	96

Table 8
The average classification accuracies of the adaptive gradient BYY harmony learning algorithms (refer to AGL-BYYs) for log-normal and Gaussian mixtures on the five stock index mixed datasets.

Algorithm	$\mathcal{G}_1$ (%)	$\mathcal{G}_2$ (%)	$\mathcal{G}_3$ (%)	$\mathcal{G}_4$ (%)	$\mathcal{G}_5$ (%)
AGL-BYY for log-normal mixtures	100	100	94.33	99.67	94.67
AGL-BYY for Gaussian mixtures	100	100	85.47	98.67	94.33

It can be seen from Tables 7 and 8 that our proposed algorithm for log-normal mixtures not only makes a better model selection, but also gets a higher classification accuracy than the adaptive gradient BYY harmony learning algorithm for Gaussian mixtures does. These results indeed show that in certain practical usage and applications, a log-normal mixture modeling method can be more efficient than a Gaussian mixture modeling method.

5. Conclusions

We have applied the BYY automated model selection mechanism to the log-normal mixture modeling by constructing an adaptive gradient BYY learning algorithm for log-normal mixtures. Moreover, two simple strategies for deleting an extra component and combining two similar and close components are suggested to enhance the convergence efficiency of the algorithm. It is demonstrated by the experiments on both synthetic and real-world datasets that our proposed adaptive gradient BYY learning algorithm leads to automated model selection, i.e., the automated correct selection of number of actual Log-normal densities, as well as a rather good estimation of the parameters in the original Log-normal mixture to generate the dataset, and even outperforms the MML-based unsupervised learning algorithm for log-normal mixtures as well as the EM algorithm for log-normal mixtures together with the AIC or BIC criterion on model selection.

Acknowledgments

This work was supported by the Natural Science Foundation of China for Grant 61171138 and the Ph.D. Programs Foundation of Ministry of Education of China for Grant 20100001110006.

References

[1] B.G. Lindsay, Mixture models: theory, geometry, and applications, in: NSF-CBMS Regional Conference Series in Probability and Statistics, vol. 5, Institute for Mathematical Statistics, Hayward, CA, 1995.

[2] G.J. Mclachlan, D. Peel, Finite Mixture Models, John Wiley & Sons, New York, 2000.

[3] R.A. Render, H.F. Walker, Mixture densities, maximum likelihood and the EM algorithm, SLAM Rev. 26 (2) (1984) 195–239.

[4] J. Mao, A.K. Jain, A self-organizing network for hyperellipsoidal clustering, IEEE Trans. Neural Netw. 7 (1) (1996) 16–29.

[5] S. Fruhwirth-Schnatter, Finite Mixture and Markov Switching Models, Springer, Herdelberg, 2006.

[6] H. Akaike, A new look at the statistical model identification, IEEE Trans. Autom. Control AC-19 (1974) 716–723.

[7] G. Scharz, Estimating the dimension of a model, Ann Stat. 6 (1978) 461–464.

[8] R. Hecht-Nielsen, Neurocomputing, Addison-Wesley, Reading, MA, 1990.

[9] S.C. Ahalt, A.K. Krishnamurty, P. Chen, D.E. Melton, Competitive learning algorithm for vector quantization, Neural Netw. 3 (1990) 277–291.

[10] L. Xu, A. Krzyzak, E. Oja, Rival penalized competitive learning for clustering analysis, RBF net, and curve detection, IEEE Trans. Neural Netw. 4 (1993) 636–648.

[11] L. Xu, Rival penalized competitive learning, finite mixture, and multisets clustering, in: Proceedings of 1998 IEEE International Joint Conference on Neural Networks, May 4–9, 1998, Anchrorage, Alaska, vol. 3, pp. 251–2530.

[12] J. Ma, T. Wang, A cost-function approach to rival penalized Competitive learning (RPCL), IEEE Trans. Syst. Man Cybern. Part B: Cybern. 36 (4) (2006) 722–737.

[13] J. Ma, B. Cao, The Mahalanobis distance based rival penalized competitive learning algorithm, in: Lecture Notes in Computer Science, vol. 3971, 2006, pp. 442–447.

[14] M.A.T. Figueiredo, A.K. Jain, Unsupervised learning of finite mixture models, IEEE Trans. Pattern Anal. Mach. Intell. 24 (3) (2002) 381–395.

[15] C. Wallace, D. Dowe, Minimum message length and Kolmogorov Complexity, Comput. J. 42 (4) (1999) 270–283.

[16] N. Ueda, Z. Ghahramani, Bayesian model search for mixture models based on optimizing variational bounds, Neural Netw. 15 (10) (2002) 1123–1241.

[17] C. Constantinopoulos, A. Likas, Unsupervised learning of Gaussian mixtures based on variational component splitting, IEEE Trans. Neural Netw. 18 (3) (2007) 745–755.

[18] L. Xu, Best harmony, unified RPCL and automated model selection for unsupervised and supervised learning on Gaussian mixtures, three-layer nets and ME-RBF-SVM models, Int. J. Neural Syst. 11 (1) (2001) 43–69.

[19] L. Xu, BYY harmony learning, structural RPCL, and topological self-organizing on mixture modes, Neural Netw. 15 (2002) 1231–1237.

[20] J. Ma, T. Wang, L. Xu, A gradient BYY harmony learning rule on Gaussian mixture with automated model selection, Neurocomputing 56 (2004) 481–487.

[21] J. Ma, B. Gao, Y. Wang, Q. Cheng, Conjugate and natural gradient rules for BYY harmony learning on Gaussian mixture with automated model selection, Int. J. Pattern Recognit. Artif. Intell. 19 (2005) 701–713.

[22] J. Ma, L. Wang, BYY harmony learning on finite mixture: adaptive gradient implementation and a floating RPCL mechanism, Neural Process. Lett. 24 (2006) 19–40.

[23] J. Ma, X. He, A fast fixed-point BYY harmony learning algorithm on Gaussian mixture with automated model selection, Pattern Recognit. Lett. 29 (6) (2008) 701–711.

[24] J. Ma, J. Liu, The BYY annealing learning algorithm for Gaussian mixture with automated model selection, Pattern Recognit. 40 (7) (2007) 2029–2037.

[25] J. Ma, J. Liu, Z. Ren, Parameter estimation of Poisson mixture with automated model selection through BYY harmony learning, Pattern Recognit. 4211 (2009) 2570–2659.

[26] Z. Ren, J. Ma, BYY harmony learning on Weibull mixture with automated model selection, in: Lecture Notes in Computer Science, vol. 5263, 2008, pp. 589–599.

[27] G. Tarmast, Multivariate Log-normal Distribution, International Statistical Institute ⟨http://isi.cbs.nl/iamamember/CD2/pdf/329.PDF⟩.

[28] C.E. McLaren, M. Wagstaff, G.M. Brittenham, A. Jacobs, Detection of two-component mixtures of lognormal distributions in grouped, doubly truncated data: analysis of red blood cell volume distributions, Biometrics 47 (1991) 607–622.

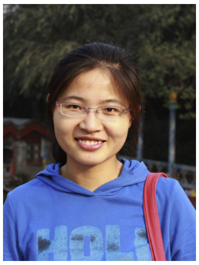
[29] Z. Liu, J. Almhana, F. Wang, R. McGorman, Mixture lognormal approximations to lognormal sum distributions, IEEE Commun. Lett. 11 (9) (2007) 711–713.

[30] H. Robbins, S. Monro, A stochastic approximation method, Ann. Math. Stat. 22 (1951) 400–407.

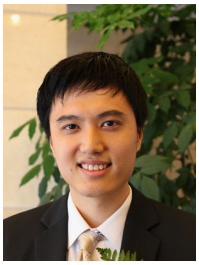
[31] H. Wang, L. Li, J. Ma, The competitive EM algorithm for Gaussian mixtures with BYY harmony criterion, in: Lecture Notes in Computer Science, vol. 5226, 2008, pp. 580–588.

[32] J. Ma, Automated Model Selection (AMS) on finite mixtures: a theoretical analysis, in: Proceedings of 2006 International Joint Conference on Neural Networks (IJCNN06), July 16–21, 2006, Vancouver, Canada, pp. 8255–8261.

Wenli Zheng received the B.S. degree, in 2011, from the School of Mathematical and Information Sciences at Shaanxi Normal University. Then, she has been a Ph.D. student in the Department of Information Science at the School of Mathematical Sciences, Peking University. Her main research interests include pattern recognition, learning theory and algorithm.



Zhijie Ren received her B.S. and M. S. degrees in Applied Mathematics from Peking University, in 2007 and 2010, respectively. She is now working at a software research institute. Her main research interests include pattern recognition, learning theory and algorithm.



Yifan Zhou received his B.S. in Information and Computing Sciences from Peking University, in 2007, M.S. degree in Petroleum Engineering from Stanford University, in 2009, and Ph.D degree in Energy Resources Engineering from Stanford University, in 2012. He is now a lead research scientist in Chevron Energy Technology Company (Houston TX, USA). His main research interests include numerical simulation of fluid flow in porous media and high performance computing.

Jinwen Ma received his M.S. degree in Applied Mathematics from Xi'an Jiaotong University, in 1988 and the Ph.D. degree in Probability Theory and Statistics from Nankai University, in 1992. From July 1992 to November 1999, he was a lecturer or associate professor at Department of Mathematics, Shantou University. From December 1999, he became a full professor at Institute of Mathematic, Shantou University. From September 2001, he has joined the Department of Information Science at the School of Mathematical Sciences, Peking University, where he is currently a full professor and Ph.D. tutor. During 1995 and 2003, he visited several times the Department of Computer Science and Engineering, the Chinese University of Hong Kong as a Research Associate or Fellow. From September 2005 to August 2006, he worked as a research scientist at Amari Research Unit, RIKEN Brain Science Institute, Japan. From September 2011 to February 2012, he visited as a scientist the Department of Systems Medicine and Bioengineering, Houston Methodist Hospital Research Institute. He has published over 100 academic papers on neural networks, pattern recognition, bioinformatics, and information theory.