Neurocomputing 188 (2016) 262–274

Contents lists available at ScienceDirect

Neurocomputing

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

BYY harmony learning of t-mixtures with the application to image segmentation based on contourlet texture features

Yunsheng Jiang, Chenglin Liu, Jinwen Ma*

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

ARTICLE INFO

Article history:
Received 28 August 2014
Received in revised form
14 January 2015
Accepted 18 January 2015
Available online 17 December 2015

Keywords:
Bayesian Ying-Yang (BYY) harmony learning
Multivariate t-mixture
Gradient learning
Model selection
Contourlet texture features

ABSTRACT

In this paper, we extend Bayesian Ying-Yang (BYY) harmony learning to the case of *multivariate t-mixtures* and propose a gradient BYY harmony learning algorithm that can automatically determine the number of actual t-distributions in a dataset during parameter learning. It is demonstrated by simulation experiments that this proposed algorithm for t-mixtures is both effective and stable on model selection and parameter estimation. Moreover, by mainly utilizing certain *contourlet texture features* from an image, the proposed algorithm is successfully applied to unsupervised image segmentation, showing considerable advantages for both general and multi-texture images.

1. Introduction

As a powerful tool for statistical learning and data analysis, finite mixture model has been widely used in pattern recognition, signal processing and image analysis. In the finite mixture modeling, data are viewed as arising from a linear mixture of two or more populations with certain proportions. Obviously, the number of components or populations is a very important parameter, but this information is not available in general. Actually, when the number of components is unknown, the finite mixture modeling becomes a rather challenging task.

Structurally, the number of components can be considered as a measure of complexity for the finite mixture model, so the determination of component number is usually referred to as *model selection*. In a conventional way, this model selection problem could be solved by optimizing certain criterion like Akaike's Information Criterion (AIC) [1], Bayesian Inference Criterion (BIC) [2] or Minimum Description Length (MDL) [3]. However, these optimization methods needed to loop the number of components, i.e., k, within a certain range, then make parameter estimation for each k, and finally choose the best k^* that could correspond to the optimal value of the criterion function. Clearly, this kind of methods would lead to a large computational cost due to the repetitions of k for many times. Besides, the stochastic simulated methods [4] were also used to solve this mixture modeling problem without knowing the number of components, but they generally required a large number of samples via different sampling rules. Furthermore, by projecting high-dimensional data into one dimension, the Probabilistic Validation (PV) approach [5] combined the cluster tendency estimation and cluster validation inside the clustering analysis to determine the number of clusters/components. However, this PV method was only suitable for linear-separable data with a few number of clusters and negligible overlaps.

With the help of competitive learning mechanism, Figueiredo and Jain [6] proposed an unsupervised learning framework for finite mixtures which could make model selection adaptively during the parameter learning with a simplified MML model selection criterion. In addition, Xu et al. [7] had already suggested the Rival Penalised Competitive Learning (RPCL) algorithm to achieve the automatic determination of cluster or component number by updating the winner unit and de-learning the rival. Unlike the RPCL approach, Cheung [8] constructed the k*-means algorithm which could carry out the rival penalization mechanism in a reasonable and implicit way, whereby circumventing the sensitive parameter (de-learning rate) in the original RPCL algorithm. Moreover, by combining the RPCL mechanism with the EM algorithm, the RPEM algorithm [9,10] was also developed to realize the automatic model selection of Gaussian mixtures and t-mixtures. Essentially, this kind of learning algorithms made automatic model selection using an effective rule that annihilates the components with small mixing proportions during the parameter learning. In fact, a similar rule is also adopted in our proposed algorithm.

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

http://dx.doi.org/10.1016/j.neucom.2015.01.112
0925-2312/

Bayesian Ying-Yang (BYY) harmony learning, firstly proposed in 1995 [11] and then systematically developed in the sequent years [12], can effectively tackle this finite mixture modeling problem in a new statistical learning way such that model selection can be made automatically during parameter learning. Specifically, BYY harmony learning can be implemented via maximizing a particular harmony function on finite mixtures, which is actually reduced from the harmony functional between Ying-machine and Yang-machine in the BYY system related to the finite mixture modeling problem. In fact, the Gaussian mixture modeling problem was already solved by the gradient BYY harmony learning on a specific BI-architecture of the BYY learning system [13], and then the conjugate gradient, adaptive gradient, and fixed-point BYY harmony learning algorithms [14–16] were further proposed to improve the efficiency of the harmony function maximization. All these methods can automatically select an appropriate number of components by attenuating the mixing proportions of extra components to zero. Methodically, this BYY harmony learning can be extended to any non-Gaussian mixture model, and it has been already realized on Poisson mixtures [17], Weibull mixtures [18] and log-normal mixtures [19].

In this paper, we extend this BYY harmony learning mechanism to the case of *multivariate t-mixtures* which are usually adopted for a set of continuous multivariate data that have a wider tail than Gaussian's or atypical observations. Under a BI-architecture of the BYY harmony learning system, a gradient BYY harmony learning algorithm is constructed for maximizing the harmony function so that model selection can be also made automatically during parameter learning for t-mixtures. The performance of the proposed algorithm is further demonstrated by various simulation experiments. Moreover, with the main utilization of *contourlet texture features*, this BYY harmony learning algorithm for t-mixtures is successfully applied to unsupervised image segmentation.

The rest of this paper is organized as follows. We begin with a brief introduction of t-distribution and t-mixture in Section 2. Then, the gradient BYY harmony learning algorithm for t-mixtures is derived and constructed in Section 3. The simulation experiments are carried out in Section 4. Section 5 contains a detailed description about the extraction of contourlet texture features and the application of the proposed gradient BYY harmony learning algorithm to unsupervised image segmentation. Finally, a brief conclusion is made in Section 6.

2. Multivariate t-distributions and mixture

In this section, we briefly introduce multivariate t-distributions as well as the t-mixture models.

2.1. Multivariate t-distribution

In statistics, a multivariate t-distribution is a multivariate generalization of Student's t-distribution. For the case of d-dimension, if y and U are independent and distributed as $\mathcal{N}(0,\Sigma)$ (Gaussian or normal distribution) and χ^2_ν (chi square distribution), respectively, where Σ is a $d\times d$ covariance matrix and $\nu>0$, $x=\frac{y}{\sqrt{\frac{U}{\nu}}}+\mu$ is then subject to a multivariate t-distribution with parameters $\{\mu,\Sigma,\nu\}$, and it takes the following density function:

$$q(x|\mu,\Sigma,\nu)=\frac{\Gamma\left(\frac{\nu+d}{2}\right)|\Sigma|^{-1/2}}{(\pi\nu)^{\frac{d}{2}}\Gamma(\frac{\nu}{2})[1+\nu^{-1}\delta(x,\mu,\Sigma)]^{(\nu+d)/2}}, \tag{1}$$

where $\delta(x,\mu,\Sigma)=(x-\mu)^T\Sigma^{-1}(x-\mu)$.

In fact, it can be proved that as $\nu\to\infty$, multivariate t distribution converges to the Gaussian distribution with mean μ and covariance Σ. However, multivariate t-distribution has a longer tail than Gaussian distribution, and is thus more robust in fitting the data with *atypical* observations.

2.2. Multivariate t-mixture model

A finite mixture is a probabilistic model with two or more components or populations linearly mixed with certain proportions. We consider the following finite mixture model:

$$q(x|\Theta_k)=\sum_{j=1}^{k}\alpha_j q(x|\theta_j), \tag{2}$$

where x denotes the variable and $\Theta_k=\{\alpha_j,\theta_j\}_{j=1}^k$ denotes all the parameters in the mixture. k is the number of components, $q(x|\theta_j)$ is the component distribution with parameter θ_j, and α_j is the mixing proportion with the constraint that $\sum_{j=1}^k\alpha_j=1$, $\alpha_j\geq 0$, $j=1,\ldots,k$. Two major tasks for the finite mixture modeling are model selection and parameter estimation,i.e., determining the true value of k and estimating the parameters in Θ_k.

If all the components $\{q(x|\theta_j)\}_{j=1}^k$ are Gaussians, the finite mixture model becomes the famous Gaussian Mixture Model (GMM). Due to its analytical tractability, asymptotic properties and computational convenience, GMM has been widely applied to model the data distributions of various random phenomena. However, GMM is sensitive to outliers, which can lead to instability and unreliability on small or noisy datasets. Alternatively, we can use multivariate t-mixture model to model this kind of datasets. Actually, multivariate t-mixture model just means that all the components in the above mixture model follow multivariate t-distribution. Since its components, i.e., t-distributions, have longer tail than Gaussians, multivariate t-mixture model is more robust than GMM in certain practical applications.

Mathematically, multivariate t-mixture model takes the following distribution:

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

where $q(x|\mu_j,\Sigma_j,\nu_j)$ is a multivariate t-distribution with parameters $\{\mu_j,\Sigma_j,\nu_j\}$ (as expressed in Eq. (1)) for $j=1,\ldots,k$. In fact, multivariate t-mixture model is often used in many practical application fields, such as clustering analysis [20] and image segmentation [21].

In order to solve the problem of t-mixture modeling, many efforts have been already made and the most frequently used approach might be the EM algorithm [22] and its extensions. However, due to the principle of the EM algorithm, the EM algorithm cannot make model selection on t-mixtures. In order to overcome this deficiency, certain penalized EM algorithms [9,23] were suggested for t-mixtures. But it is obvious that the EM-like algorithms consume a large amount of computation. In the next section, based on the BYY harmony learning framework, we will construct a gradient BYY harmony learning algorithm for t-mixtures, which can make model selection automatically during parameter estimation in a more effective and efficient way.

3. BYY Harmony Learning for t-Mixtures

In this section, we begin to introduce the BYY harmony learning system, and then derive the gradient BYY harmony learning algorithm for t-mixtures. The implementation details are finally discussed.

3.1. BYY harmony learning system and theory

In clustering analysis, data can be observed as the combination of two related parts, i.e., the observation $x \in \mathcal{X} \subset \mathcal{R}^n$ and its inner representation $y \in \mathcal{Y} \subset \mathcal{R}^m$. The BYY harmony learning system [11,12] is constructed to describe relationship between these two parts via the two types of Bayesian decomposition of the joint density: $p(x,y)=p(x)\,p(y|x)$ and $q(x,y)=q(y)\,q(x|y)$, which are called Yang machine and Ying machine, respectively. The goal of BYY harmony learning is to extract the hidden probabilistic structure of x with the help of y by specifying all aspects of $p(y|x)$, $p(x)$, $q(x|y)$ and $q(y)$. The harmony learning principle can be implemented by maximizing the functional:

$$H(p\|q)=\int p(y|x)p(x)\ln[q(x|y)q(y)]\,dx\,dy. \tag{4}$$

If both $p(y|x)$ and $q(x|y)$ are parametric,i.e., from a family of probability densities, then the BYY learning system is said to have a BI-directional architecture (BI-architecture for short).

For the multivariate t-mixture model with a given sample set $D_x=\{x_t\}_{t=1}^N$, we utilize the following specific BI-architecture: the inner representation y is discrete in $\mathcal{Y}=\{1,2,\ldots,k\}$, and the observation $x \in \mathcal{R}^d$ is generated from a multivariate t-mixture. In Ying space, we let $q(y=j)=\alpha_j \geq 0$ with $\sum_{j=1}^k \alpha_j = 1$. In Yang space, we assume that $p(x)$ is a blind d-dimensional t-mixture from which a set of samples $D_x=\{x_t\}$ is generated. Moreover, on the Ying path, we let $q(x|y=j)=q(x|\theta_j)$ be d-dimensional *t-distribution* with parameter θ_j. Thus, the Yang path can be constructed according to the Bayesian principle:

$$p(y=j|x)=\frac{\alpha_j q(x|\theta_j)}{q(x|\Theta_k)},\quad q(x|\Theta_k)=\sum_{j=1}^k \alpha_j q(x|\theta_j), \tag{5}$$

where $q(x|\Theta_k)$ tries to approximate the true t-mixture model $p(x)$ hidden in the data D_x via the harmony learning on the BYY system.

With all these component densities into Eq. (4) we get an estimate of $H(p\|q)$ on D_x as the following harmony function:

$$J(\Theta_k)=\frac{1}{N}\sum_{t=1}^N\sum_{j=1}^k \frac{\alpha_j q(x_t|\theta_j)}{\sum_{i=1}^k \alpha_i q(x_t|\theta_i)}\ln[\alpha_j q(x_t|\theta_j)]. \tag{6}$$

According to the BYY harmony learning principle, the maximization of harmony function $J(\Theta_k)$ can lead to automated model selection during parameter estimation as long as k is larger than the number of actual components in the given dataset.

3.2. Gradient BYY harmony learning algorithm

For convenience, we denote $U_j(x)=\alpha_j q(x|\theta_j)$ for $j=1,\ldots,k$, the harmony function $J(\Theta_k)$ can be then represented as

$$J(\Theta_k)=\frac{1}{N}\sum_{t=1}^N\sum_{j=1}^k \frac{U_j(x_t)}{\sum_{i=1}^k U_i(x_t)}\ln U_j(x_t). \tag{7}$$

As for the parameters in Θ_k, there are certain constraints such as: $\sum_{j=1}^k \alpha_j = 1, \alpha_j \geq 0$, $\nu_j > 0$, and Σ_j must be *positive definite*. To get rid of these constrains, we make the following transformations: $\alpha_j=\frac{e^{\beta_j}}{\sum_{i=1}^k e^{\beta_i}}$, $\nu_j=v_j^2$ and $\Sigma_j=B_jB_j^T$ for $j=1,2,\ldots,k$, where $\beta_j, v_j \in (-\infty,+\infty)$ and B_j is just a nonsingular square matrix. In this way, the partial derivatives of $J(\Theta_k)$ with respect to β_j, μ_j, B_j, v_j, respectively, can be given as follows:

$$\frac{\partial J(\Theta_k)}{\partial \beta_j}=\frac{1}{N}\sum_{t=1}^N \frac{1}{q(x_t|\Theta_k)}\sum_{i=1}^k \lambda_i(x_t)(\delta_{ij}-\alpha_j)U_i(x_t); \tag{8}$$

$$\frac{\partial J(\Theta_k)}{\partial \mu_j}=\frac{1}{N}\sum_{t=1}^N p(j|x_t)\;\lambda_j(x_t)\;\frac{(\nu_j+d)\Sigma_j^{-1}(x_t-\mu_j)}{\nu_j+\delta(x_t,\mu_j,\Sigma_j)}; \tag{9}$$

$$vec\left(\frac{\partial J(\Theta_k)}{\partial B_j}\right)=\frac{1}{2N}\sum_{t=1}^N\left[p(j|x_t)\lambda_j(x_t)\frac{\partial(B_jB_j^T)}{\partial B_j}\cdot vec\left\{\left[\frac{\Sigma_j^{-1}(x_t-\mu_j)(x_t-\mu_j)^T}{\nu_j+\delta(x_t,\mu_j,\Sigma_j)}-I_d\right]\Sigma_j^{-1}\right\}\right]; \tag{10}$$

$$\frac{\partial J(\Theta_k)}{\partial \nu_j}=\frac{1}{N}\sum_{t=1}^N p(j|x_t)\lambda_j(x_t)\left\{\Psi\left(\frac{\nu_j+d}{2}\right)-\ln[1+\nu_j^{-1}\delta(x_t,\mu_j,\Sigma_j)]-\Psi\left(\frac{\nu_j}{2}\right)-\frac{d}{\nu_j}+\frac{(\nu_j+d)\delta(x_t,\mu_j,\Sigma_j)}{\nu_j(\nu_j+\delta(x_t,\mu_j,\Sigma_j))}\right\}v_j \tag{11}$$

where δ_{ij} is the Kronecker function, $\lambda_i(x_t)=1-\sum_{l=1}^k (p(l|x_t)-\delta_{il})\ln U_l(x_t)$, $\Psi(z)=\frac{\partial \ln \Gamma(z)}{\partial z}$, I_d is the d-dimensional identity matrix, and $vec(M)$ denotes the vector obtained by stacking the column vector of the matrix M. The detailed expression of $\frac{\partial(B_jB_j^T)}{\partial B_j}$ can be found in [14].

After obtaining the partial derivatives in Eqs. (8)–(11), we can maximize the harmony function for t-mixtures by the following batch gradient BYY harmony learning algorithm:

$$\phi^{new}=\phi^{old}+\eta\cdot\frac{\partial J(\Theta_k)}{\partial \phi},\quad \phi \in \{\beta_j,\mu_j,B_j,v_j\}_{j=1}^k, \tag{12}$$

where $\eta(>0)$ denotes the learning rate. The iteration will stop when $\Delta J=|J(\Theta_k^{new})-J(\Theta_k^{old})|<T_{term}$, where $T_{term}(>0)$ is a threshold value for termination.

3.3. Implementation details

We finally turn to the implementation details of the proposed gradient BYY harmony learning algorithm that should be further discussed or explained, including the initialization of the parameters, the update of the learning rate and the deletion of extra components. For simplicity, our proposed algorithm is referred to as the BYY-t algorithm here and hereafter.

Initialization of the parameters: Before we start the iterations of the gradient BYY harmony learning with Eq. (12), we need to set some reasonable initial values for all the parameters. The number k of components should be (initially) set to be greater than the true number k^* of actual components. However, a too big k may increase not only the time of computation, but also the risk of selecting a wrong model. The mixing proportions $\{\alpha_j\}_{j=1}^k$ are initialized as $\alpha_j=\frac{1}{k}$, $j=1,\ldots,k$, that is, $\{\beta_j\}_{j=1}^k$ are initialized equally. The covariance matrix and degree of freedom in multivariate t-distribution are set as $\Sigma_j=B_j=I_d$ and $\nu_j=v_j=1$ for $j=1,\ldots,k$. As for $\{\mu_j\}_{j=1}^k$, they are initialized simply by k samples which are randomly selected from the dataset.

For the more accurate initialization, we can firstly divide the data points into k groups by a simple clustering analysis algorithm (like the k-means algorithm or the RPCL algorithm [7,24]), estimate the parameters of the j-th component with only the samples in the j-th group, and use the estimated values as the initialization settings of the parameters for the BYY-t algorithm. For example, α_j can be estimated as the proportion of the sample number of the j-th cluster over the whole sample number, and μ_j can be estimated as the cluster center.

Update of the learning rate: The learning rate η is an important parameter for a gradient-type learning algorithm. A small learning rate can make the algorithm converge slowly, while a large learning rate may lead to unstable iterations. To overcome this problem, we can update the learning rate dynamically during the iterations, instead of setting it to a fixed value. Actually, the

learning rate is updated as follows:

$$\eta^{(t+1)} = \eta^{(t)} \times \begin{cases} 1.05, & J(\Theta_k^{(t+1)}) - J(\Theta_k^{(t)}) > 0; \\ 0.7, & \text{else.} \end{cases}$$

And the initial value of the learning rate is set as $\eta^{(0)} = 0.5$.

Deletion of extra components: During the BYY harmony learning process, the mixing proportions of these extra components in the mixture will be forced to attenuate to zero eventually. In order to make the convergence efficiently, we can delete the extra components directly during the parameter learning. In fact, as the initial k is greater than the true value k^*, the BYY-t algorithm will use only k^* components to match the actual component distributions, and attenuate the mixing proportions of the other $(k-k^*)$ extra components to zero (approximately). So, in order to realize the automated model selection and accelerate the convergence of the BYY-t algorithm, we can check the mixing proportions of the existing components at each iteration. If $\alpha_j < T_\alpha$ (where T_α is a threshold value for component deletion and is fixed to $T_\alpha = 0.05$ in our following experiments), the j-th component will be deleted, and the number of components is set as $k \Leftarrow k-1$. By adding this *checking and deleting* mechanism to the original gradient algorithm, the BYY-t algorithm can achieve automated model selection during parameter learning more effectively and efficiently.

4. Simulation experiments

In this section, simulation experiments are carried out to demonstrate the performance of the BYY-t algorithm on synthetic datasets generated from various multivariate t-mixtures, in comparison with some competitive algorithms.

4.1. Performance of model selection

We firstly implement the BYY-t algorithm on 2D synthetic datasets $\mathcal{S}_1$–$\mathcal{S}_4$, which are sampled from four typical 2-dimensional t-mixtures whose actual parameters are shown in Table 1: (a) $\mathcal{S}_1$ is composed of four components which have the same mixing proportion and covariance. (b) The four components in $\mathcal{S}_2$ have different mixing proportions and covariances. (c) $\mathcal{S}_3$ has only three components, and their mixing proportions have a big difference. (d) $\mathcal{S}_4$ is similar to $\mathcal{S}_2$, but only with a small number of samples. The sketches of these four datasets are shown in Fig. 1, respectively.

To investigate the performance of model selection and parameter estimation, we implement the BYY-t algorithm on $\mathcal{S}_1-\mathcal{S}_4$ for different initial $k \in \{k^*, 2k^*, 3k^*, 4k^*\}$, where k^* denotes the number of actual components in the dataset. To make the algorithm converge faster, the mixing proportions and cluster centers in BYY-t are initialized via a RPCL procedure. For each dataset and each initial value of k, we run the algorithm for 100 *times* to get stable results. Table 2 shows the percentages of Correct Model Selection (CMS) for BYY-t algorithm, as well as comparisons with three other competitive algorithms. Here, "BYY-t" is our proposed algorithm, while "EM-AIC" and "EM-BIC" denote the EM algorithms for t-mixtures together with the AIC [1] and BIC [2] model selection criteria, respectively. Besides, "EM-MML" denotes the unsupervised learning algorithm based on EM algorithm and MML criterion [6], and similar to the BYY-t algorithm, it is a typical and competitive learning algorithm for the finite mixture models with automated model selection. Note that the cluster centers in these three EM-based algorithms are initialized with randomly selected samples. Actually, their performances become worse and unstable when the cluster centers are initialized with the RPCL procedure.

Table 1
The actual parameters of four typical 2D t-mixture datasets $\mathcal{S}_1-\mathcal{S}_4$.

Dataset	j	μ_1^j	μ_2^j	B_{11}^j	$B_{12}^j = B_{21}^j$	B_{22}^j	ν^j	α^j
	1	0	−3.0	0.50	0	0.50	3	0.25
$\mathcal{S}_1$	2	−3.0	0	0.50	0	0.50	3.5	0.25
$N=1600$	3	0	3.0	0.50	0	0.50	4	0.25
	4	3.0	0	0.50	0	0.50	4.5	0.25
	1	0	−3.0	0.45	−0.25	0.55	3	0.34
$\mathcal{S}_2$	2	−3.0	0	0.65	0.20	0.25	3.5	0.28
$N=1600$	3	0	3.0	1	0.10	0.35	4	0.22
	4	3.0	0	0.30	0.15	0.80	4.5	0.16
	1	2.5	0	0.20	−0.20	0.50	3	0.50
$\mathcal{S}_3$	2	0	2.5	0.40	0.10	0.20	3.5	0.30
$N=1200$	3	−1.5	−1.5	0.80	−0.20	0.30	4	0.20
	1	0	−3.0	0.28	−0.20	0.32	3	0.16
$\mathcal{S}_4$	2	−3.0	0	0.34	0.20	0.22	3.5	0.22
$N=200$	3	0	3.0	0.50	0.04	0.12	4	0.28
	4	3.0	0	0.10	0.05	0.50	4.5	0.34

According to the data listed in Table 2, we can find out the following facts: (1) The BYY-t algorithm gains high percentages of CMS for most cases (on different datasets with different initial k). (2) The BYY-t algorithm outperforms EM-AIC, EM-BIC and EM-MML algorithms considerably, by more than 10 percentage points for most cases, and this gap becomes much bigger in some cases with critical conditions (e.g., small data size and large initial k). (3) The BYY-t algorithm seems to keep high percentages of CMS when the initial value of k increases, which may benefit from the precise initialization of cluster centers via the RPCL procedure. On the other hand, if we compare the CMS% of the BYY-t algorithm on different datasets, we can further see that: (a) For the simple dataset $\mathcal{S}_1$ with round shaped components, the BYY-t algorithm can always obtain the correct model selection. (b) The CMS% on $\mathcal{S}_3$ are not as high as that on $\mathcal{S}_2$, which may be ascribed to its more unbalanced mixing proportions and more compact mixture components. (c) The CMS% on $\mathcal{S}_4$ are relatively lower than that on $\mathcal{S}_2$, due to its small data size.

4.2. Performance of parameter estimation

As for parameter estimation, we evaluate the performance by *relative square error*. For a set of parameters $w = \{w_j\}_{j=1}^{k^*}$, let their actual values be $\{w_j^*\}_{j=1}^{k^*}$ and the corresponding estimates be $\{\hat{w}_j\}_{j=1}^{k^*}$, then the relative square error of w is defined by

$$\Delta w = \sum_{j=1}^{k} \frac{\| \hat{w}_j - w_j^* \|^2}{\| w_j^* \|^2}. \tag{13}$$

For each dataset $\mathcal{S}_1$–$\mathcal{S}_4$, we fix the initial k as $k = 2k^*$ and repeat the experiments until the algorithms have obtained correct model selection for 100 *times*. We then average the relative errors for parameter α, μ, Σ, ν, respectively, as is list in Table 3. From this table we can find out that the relative errors of BYY-t algorithm are small in most cases, which means it can achieve relative accurate and stable parameter estimations in various conditions. However, compared with the EM-based algorithms, the errors of BYY-t algorithm is slightly larger. This is reasonable since the BYY-t algorithm achieves automated model selection by competitive learning, which causes certain deviation for parameter estimation, while the EM-based algorithms lead to a maximum likelihood estimate which is consistent with the true parameters.

Unlike α, μ, Σ, the degree of freedom ν just reflects the tail of t-distribution, which can not be simulated well in the synthetical datasets unless a very large number of samples are generated. Therefore, the estimation of ν is difficult and unstable for

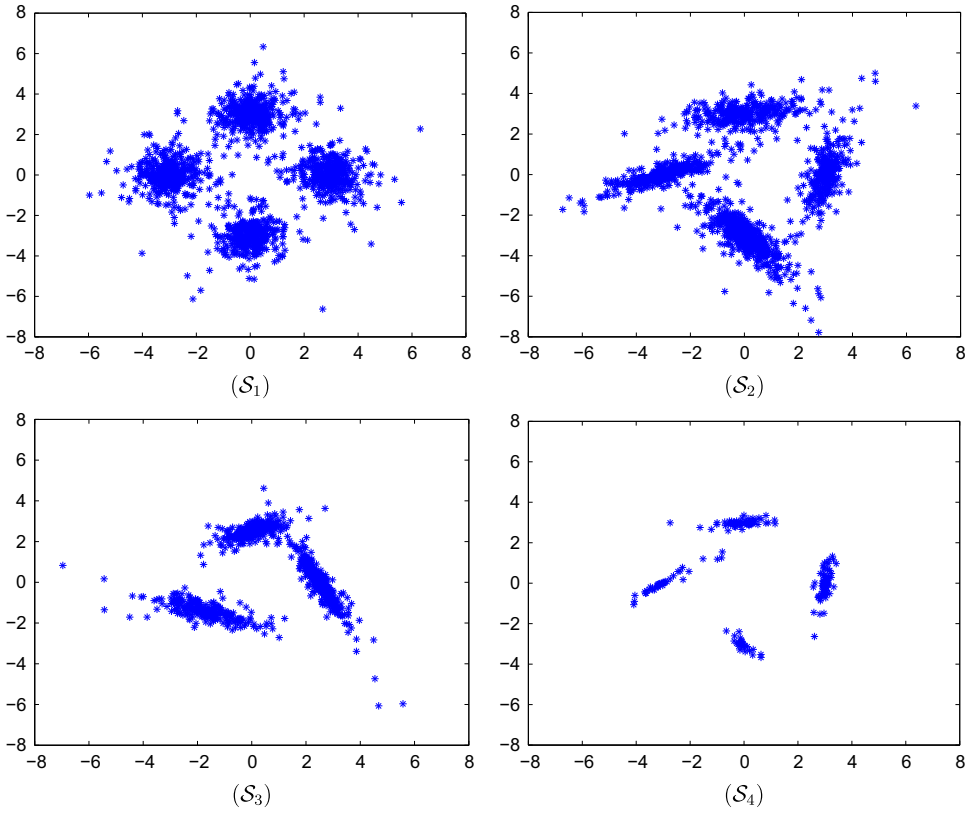

Fig. 1. The sketches of datasets $\mathcal{S}_1-\mathcal{S}_4$.

Table 2
The percentages of CMS for different algorithms (on datasets $\mathcal{S}_1$–$\mathcal{S}_4$, with initial $k\in\{k^*,2k^*,3k^*,4k^*\}$).

Datasets	Algorithm	$k=k^*$	$k=2k^*$	$k=3k^*$	$k=4k^*$
$\mathcal{S}_1$	BYY-t	100	100	100	100
	EM-AIC	80	61	64	64
	EM-BIC	71	65	65	71
	EM-MML	77	60	61	66
$\mathcal{S}_2$	BYY-t	100	96	97	96
	EM-AIC	84	42	47	56
	EM-BIC	84	67	69	68
	EM-MML	84	48	53	58
$\mathcal{S}_3$	BYY-t	100	90	86	91
	EM-AIC	86	75	73	63
	EM-BIC	85	88	82	74
	EM-MML	87	72	68	58
$\mathcal{S}_4$	BYY-t	100	93	85	82
	EM-AIC	86	11	9	10
	EM-BIC	80	66	62	55
	EM-MML	87	9	6	9

simulated datasets. Many algorithms have some "uncomfortable cases" in which the algorithms may return a bad estimation for ν, like the BYY-t algorithm on $\mathcal{S}_1$ and the EM-based algorithms on $\mathcal{S}_4$. These algorithm-specific uncomfortable cases may be caused by small data size, big component overlap, specific data shape, etc. Though they exist, the uncomfortable cases are not frequent, and in most cases the BYY-t algorithm or EM-based algorithms can obtain a good estimation for ν. Moreover, the influence of uncomfortable cases can be weaken by increasing the data size. In fact, further experiments have shown that, when the number of samples in $\mathcal{S}_1$ increases to 5000, the relative error of ν for the BYY-t algorithm is 0.9771 ± 0.2714, and if the size increases to 50,000, the relative error further decreases to 0.8244 ± 0.0744.

4.3. Influence of component overlap

To investigate the influence of dataset overlap to the performance of the BYY-t algorithm, we generate another two datasets $\mathcal{S}_5$ and $\mathcal{S}_6$, which have the same mixing proportions and covariances with $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively, but with different cluster centers. As is shown in Fig. 2, the component overlaps in $\mathcal{S}_5$ and $\mathcal{S}_6$ are much bigger than that in $\mathcal{S}_1$ and $\mathcal{S}_2$, respectively. For each of $\mathcal{S}_5$ and $\mathcal{S}_6$, we fix the initial k as $k=2k^*$ and repeat the experiments to get the percentages of CMS and relative estimation errors, which are listed in Table 4. From this table we can see that, though affected by the heavy component overlaps, the BYY-t algorithm still gains high percentages of CMS and good parameter estimation. Further experiments also show that the BYY-t algorithm can achieve correct model selection for most cases, provided that the component overlap is "weak" enough (i.e., below a certain threshold). The influence of component overlap on EM-based algorithms is not as large as that on BYY-t algorithm. However, the BYY-t algorithm still outperforms EM-AIC, EM-BIC and EM-MML algorithms on model selection.

4.4. Influence of high dimensionality and multiple classes

To test the effect of the BYY-t algorithm for higher-dimensional dataset, we further generate a dataset $\mathcal{S}_7$ which is a 3-dimensional t-mixture with 4 components (see Fig. 3(a)). In this case, the BYY-t algorithm can achieve correct model selection (with the initial $k=2k^*$), and the estimated proportions [0.2483, 0.2536, 0.2489, 0.2492] are very close to the actual values [0.25, 0.25, 0.25, 0.25]. Along this direction, we even implement the BYY-t algorithm on a 20-dimensional dataset of 4 components with the similar degree of overlap, and find out that the BYY-t algorithm works well for such a high-dimensional dataset, obtaining the correct model selection and good parameter estimation as well.

Table 3
The relative errors of parameter estimation for different algorithms (on datasets $\mathcal{S}_1-\mathcal{S}_4$, with initial k fixed as $k=2k^*$).

Datasets	Algorithm	α	μ	Σ	ν
$\mathcal{S}_1$	BYY-t	0.0043 ± 0.0000	0.0007 ± 0.0000	0.2314 ± 0.0046	1.2652 ± 0.0429
	EM-AIC	0.0031 ± 0.0000	0.0007 ± 0.0000	0.1754 ± 0.0036	0.1466 ± 0.0145
	EM-BIC	0.0031 ± 0.0000	0.0007 ± 0.0000	0.1754 ± 0.0035	0.1461 ± 0.0142
	EM-MML	0.0031 ± 0.0000	0.0007 ± 0.0000	0.1753 ± 0.0035	0.1465 ± 0.0144
$\mathcal{S}_2$	BYY-t	0.0068 ± 0.0017	0.0011 ± 0.0003	0.1058 ± 0.0228	0.1762 ± 0.0228
	EM-AIC	0.0056 ± 0.0003	0.0005 ± 0.0000	0.0383 ± 0.0027	0.0531 ± 0.0234
	EM-BIC	0.0055 ± 0.0007	0.0005 ± 0.0000	0.0419 ± 0.0233	0.5445 ± 3.6455
	EM-MML	0.0056 ± 0.0003	0.0005 ± 0.0000	0.0383 ± 0.0025	0.0526 ± 0.0225
$\mathcal{S}_3$	BYY-t	0.0125 ± 0.0019	0.0534 ± 0.1080	0.0001 ± 0.0005	0.3684 ± 0.3667
	EM-AIC	0.0067 ± 0.0000	0.0104 ± 0.0000	0.0622 ± 0.0025	0.1661 ± 0.0338
	EM-BIC	0.0067 ± 0.0000	0.0104 ± 0.0000	0.0622 ± 0.0024	0.1660 ± 0.0326
	EM-MML	0.0067 ± 0.0000	0.0104 ± 0.0000	0.0622 ± 0.0025	0.1656 ± 0.0341
$\mathcal{S}_4$	BYY-t	0.1694 ± 0.3388	0.1102 ± 0.5699	0.0001 ± 0.0000	0.1879 ± 0.1359
	EM-AIC	0.0735 ± 0.0000	0.0031 ± 0.0000	0.3052 ± 0.0139	8.7807 ± 7.4462
	EM-BIC	0.0735 ± 0.0000	0.0031 ± 0.0000	0.3070 ± 0.0098	7.7128 ± 5.2874
	EM-MML	0.0735 ± 0.0000	0.0031 ± 0.0000	0.3043 ± 0.0153	9.2829 ± 8.2308

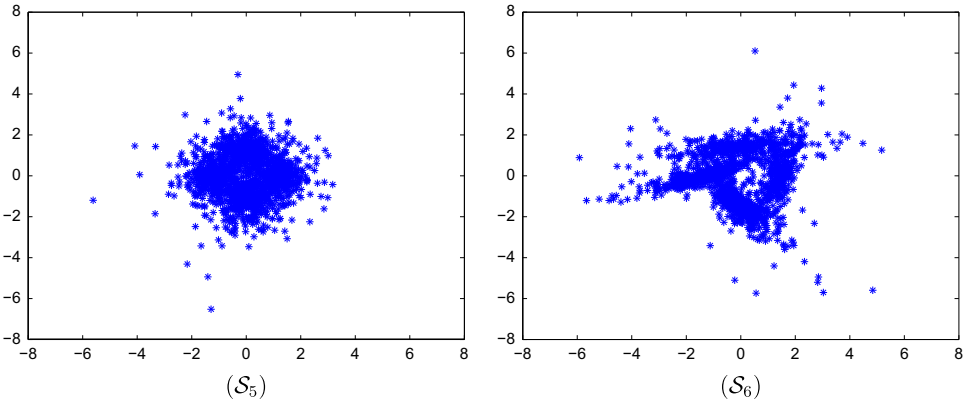

Fig. 2. The sketches of datasets $\mathcal{S}_5$ and $\mathcal{S}_6$.

Table 4
The percentages of CMS and relative estimation errors for different algorithms (on datasets $\mathcal{S}_5$ and $\mathcal{S}_6$, with initial k fixed as $k=2k^*$).

Datasets	Algorithm	CMS(%)	$\Delta\alpha$	$\Delta\mu$	$\Delta\Sigma$	$\Delta\nu$
$\mathcal{S}_5$	BYY-t	96	0.1569 ± 0.0125	0.0146 ± 0.0035	1.0207 ± 0.1359	8.1807 ± 1.5123
	EM-AIC	86	0.0695 ± 0.0030	0.0046 ± 0.0005	0.2351 ± 0.0239	0.1655 ± 0.0834
	EM-BIC	92	0.0696 ± 0.0030	0.0046 ± 0.0004	0.2348 ± 0.0231	0.1645 ± 0.0807
	EM-MML	86	0.0695 ± 0.0030	0.0046 ± 0.0005	0.2351 ± 0.0239	0.1655 ± 0.0834
$\mathcal{S}_6$	BYY-t	93	0.0522 ± 0.0114	0.0348 ± 0.0119	0.2340 ± 0.0447	0.3003 ± 0.0645
	EM-AIC	44	0.0284 ± 0.0001	0.0036 ± 0.0000	0.1744 ± 0.0023	0.1299 ± 0.0080
	EM-BIC	80	0.0284 ± 0.0001	0.0036 ± 0.0000	0.1737 ± 0.0021	0.1275 ± 0.0072
	EM-MML	80	0.0284 ± 0.0001	0.0036 ± 0.0000	0.1737 ± 0.0021	0.1275 ± 0.0072

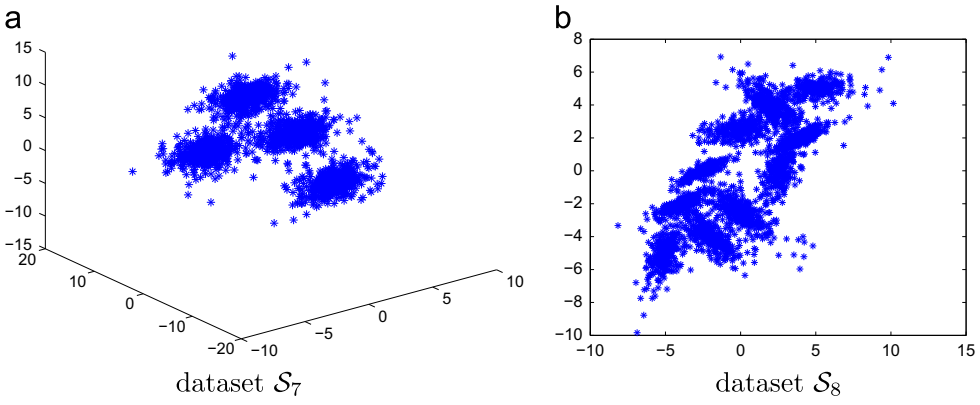

Fig. 3. The sketches of datasets $\mathcal{S}_7$ and $\mathcal{S}_8$. (a) $\mathcal{S}_7$ is a 3-dimensional t-mixture with 4 components; (b) $\mathcal{S}_8$ is a 2-dimensional t-mixture with 10 components.

Besides, we also generate a dataset $\mathcal{S}_8$ from a 2D t-mixture with 10 components (see Fig. 3) to test the effect of the BYY-t algorithm for multi-class datasets. The BYY-t algorithm can also get the correct model selection. However, when $k^* > 20$, the threshold value for deleting extra components should not be set as $T_\alpha = 0.05$, and need to be reduced in certain degree for the correct model selection.

In fact, in the next section, we will apply the BYY-t algorithm to the image segmentation task. In this case, a 8-dimensional feature vector is extracted for each pixel, and there may be about 10 segments for some images. Thus, it is really a high-dimensional and multi-class task, and the segmentation results will further demonstrate that the BYY-t algorithm works well for such a task.

4.5. Conclusions for simulation experiments

The simulation experiments in this section have demonstrated that the BYY-t algorithm outperforms EM-based algorithms on model selection by over 10 points of CMS percentage in general. Actually, this advantage becomes much bigger for some critical cases (e.g., small sample size and large initial k). However, the deviation of parameter estimation of the BYY-t algorithm is slightly larger than those of EM-based algorithms in some cases,

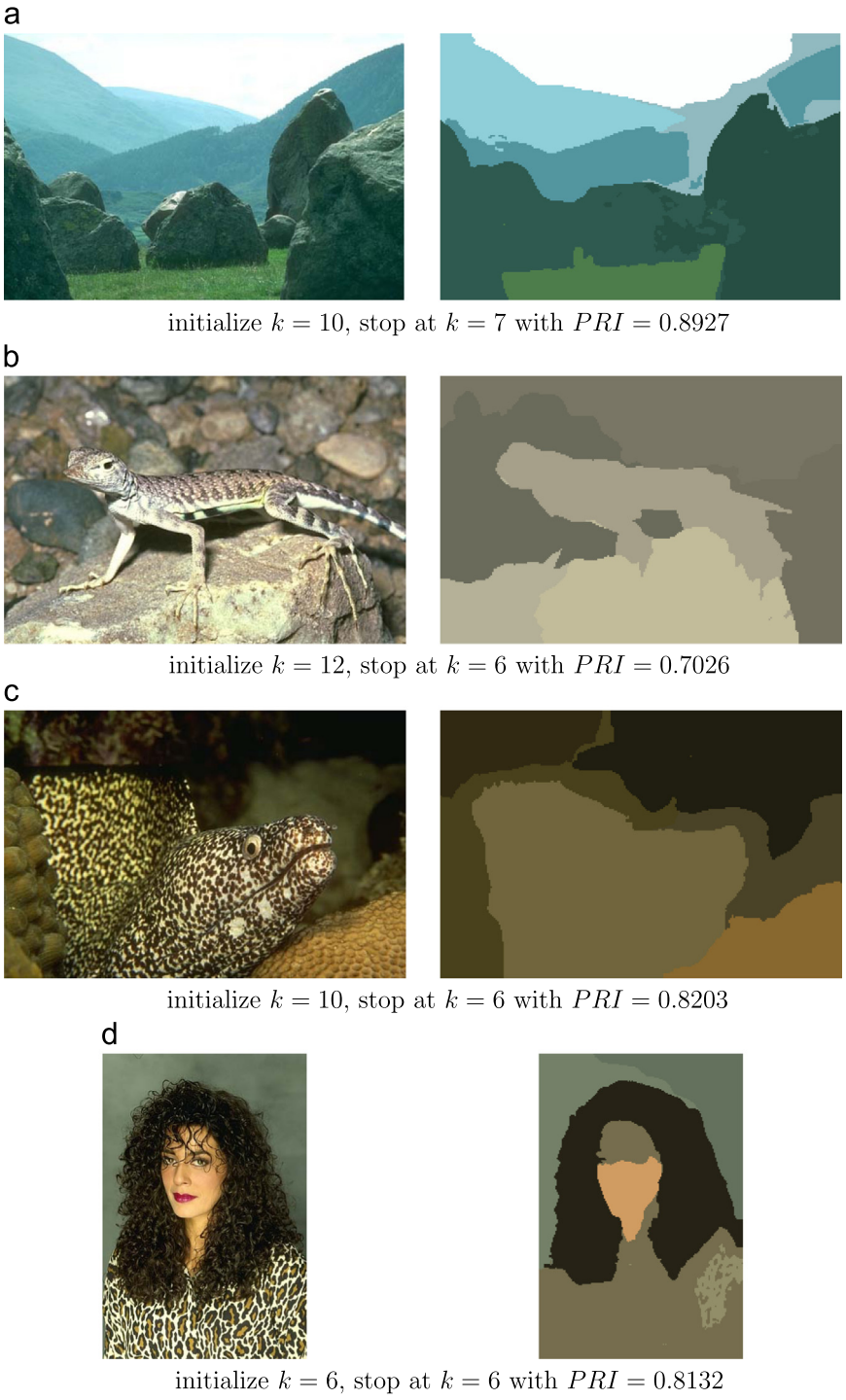

Fig. 4. Four segmentation examples of *BSDS300 general images*. Left column: original images. Right column: segmentation images.

which may be caused by the de-learning action in the BYY harmony learning process. As the model selection is so important, we can consider that the BYY-t algorithm is more effective and powerful than the EM-based algorithms for t-mixtures.

5. Unsupervised image segmentation via the BYY-t algorithm with contourlet texture features

Image segmentation is a fundamental research subject in image processing and computer vision. In fact, it tries to divide an image into some meaningful and homogeneous regions. If we can extract a feature vector for each pixel in the given image, the pixels can be considered as samples or points in the feature space. Then, image segmentation becomes a clustering analysis problem on the feature space, with each cluster corresponding to a segment. However, there are still two major challenges on this *segmentation-to-clustering* approach: (1) How to construct a proper feature space; (2) How to select an appropriate number of clusters. In this section, we utilize the contourlet texture features (plus the original position and color features) and the BYY-t algorithm to meet these two challenges.

5.1. Construction of the feature space

5.1.1. Contourlet texture features

To extract the texture features, we need to cut the image into small patches, and then analyze the textures on these patches. The extraction of contourlet texture features on each patch consists of two steps: contourlet subband clustering and spectral clustering.

Contourlet subband clustering: Texture classification algorithms based on wavelet transformation can be categorized into two types: the *model-based* approaches assume that the coefficients of wavelet subbands follow a specific distribution, while the *feature-based* approaches intent to extract statistical features from the wavelet subbands. However, the Contourlet Subband Clustering (CSC) method [25] combines the advantages of both these two categories. In particular, the CSC method applies the k-means algorithm to the coefficients of each contourlet subband, and then utilize the cluster centers (plus two other conventional features) to represent the corresponding subband (coefficients). By making such a clustering analysis on the contourlet subbands, the CSC method fully employs the information in subband coefficients without any hypothesis about their distributions, and has been demonstrated to have outstanding performance on texture image segmentation.

If we denote the number of levels for contourlet transformation as L, while the number of decomposition levels for each Directional Filter Bank (DFB) as l_i, $i=1,2,\ldots,L$, then, for each image patch, the contourlet transformation will generate $S=1+\sum_{i=1}^{L}2^{l_i}$ subbands, where the plus of one denotes the low-pass subband. After applying the k-means algorithm to each subband's coefficients with the cluster number set as M, we can obtain the CSC vector for the s-th *subband* as

$$f_s=[c_{s,1},c_{s,2},\ldots,c_{s,M},c_{s,M+1},c_{s,M+2}],\quad s=1,\ldots,S, \tag{14}$$

where $\{c_{s,j}\}_{j=1}^{M}$ are the cluster centers of the s-th subband and are sorted in the ascending order, while $c_{s,M+1}$ and $c_{s,M+2}$ are the variance and norm-2 energy of the s-th subband defined by

$$c_{s,M+1}=\frac{1}{N_s}\sum_{i=1}^{N_s}(x_{s,i}-\overline{x}_s)^2,\quad \overline{x}_s=\frac{1}{N_s}\sum_{i=1}^{N_s}x_{s,i};$$

$$c_{s,M+2}=\frac{1}{N_s}\sum_{i=1}^{N_s}x_{s,i}^2, \tag{15}$$

where $\{x_{s,i}\}_{i=1}^{N_s}$ are the coefficients of the s-th subband.

As for the CSC feature vector of an *image patch*, it is just the concatenation of the S subbands' CSC vectors, that is,

$$F=\{f_1,f_2,\ldots,f_S\}. \tag{16}$$

Spectral clustering: The CSC features are just a representation of texture information. However, for image segmentation task, what matters is the distinctiveness between different textures, not the textures themselves. Besides, the dimensionality of an image patch's CSC vector may be too high for the BYY-t algorithm. Hence, for the BYY-t algorithm based segmentation task, it is reasonable to re-extract the CSC features by emphasising the dissimilarity/similarity between different CSC vectors and reduce the feature dimensionality. This feature re-extraction or dimension reduction can be done by the spectral clustering analysis.

Before doing so, we need to define a similarity matrix for different image patches. Given two image patches A and B, and their corresponding CSC vectors $F^A=\{f_1^A,f_2^A,\ldots,f_S^A\}$ and $F^B=\{f_1^B,f_2^B,\ldots,f_S^B\}$, where $f_s^A=[c_{s,1}^A,\ldots,c_{s,M+2}^A]$ and $f_s^B=[c_{s,1}^B,\ldots,c_{s,M+2}^B]$, $s=1,\ldots,S$, are the subbands' CSC vectors, then the similarity between patches A and B is defined as

$$Sim(F^A,F^B)=\frac{1}{\tau+Dist(F^A,F^B)}, \tag{17}$$

where τ is a small value to avoid the zero value divisor, and the distance between F^A and F^B is just the sum of the *relative-L1* distances between their corresponding subbands' CSC vectors, that is,

$$Dist(F^A,F^B)=\sum_{s=1}^{S}RL_1(f_s^A,f_s^B)=\sum_{s=1}^{S}\sum_{j=1}^{M+2}\frac{|c_{s,j}^A-c_{s,j}^B|}{1+|c_{s,j}^A|+|c_{s,j}^B|}. \tag{18}$$

After obtaining the similarity matrix, we can utilize the *spectral clustering* algorithm to represent image patches with certain principal eigenvectors. Refer to [26] for detailed spectral clustering procedure and eigenvector selection method. Eventually, after the contourlet subband clustering and spectral clustering analysis, we get the final contourlet texture features for each image patch.

5.1.2. Position/color features

The position features of pixel (x,y) is simply its corresponding coordinate values, that is, its horizontal coordinate x and vertical

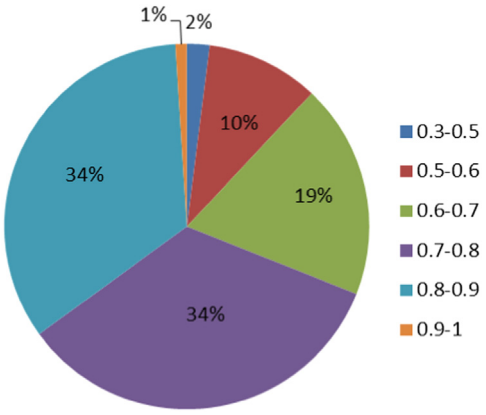

Fig. 5. The percentages of PRI levels for the 100 test images in BSDS300.

Table 5
The mean and standard deviation of PRI scores on 100 test images in BSDS300.

Method	Mean of PRI	Std of PRI
RPCL	0.727	0.132
CAC	0.732	0.154
BYY-t-DCT	0.737	0.108
BYY-t-CL	0.743	0.112

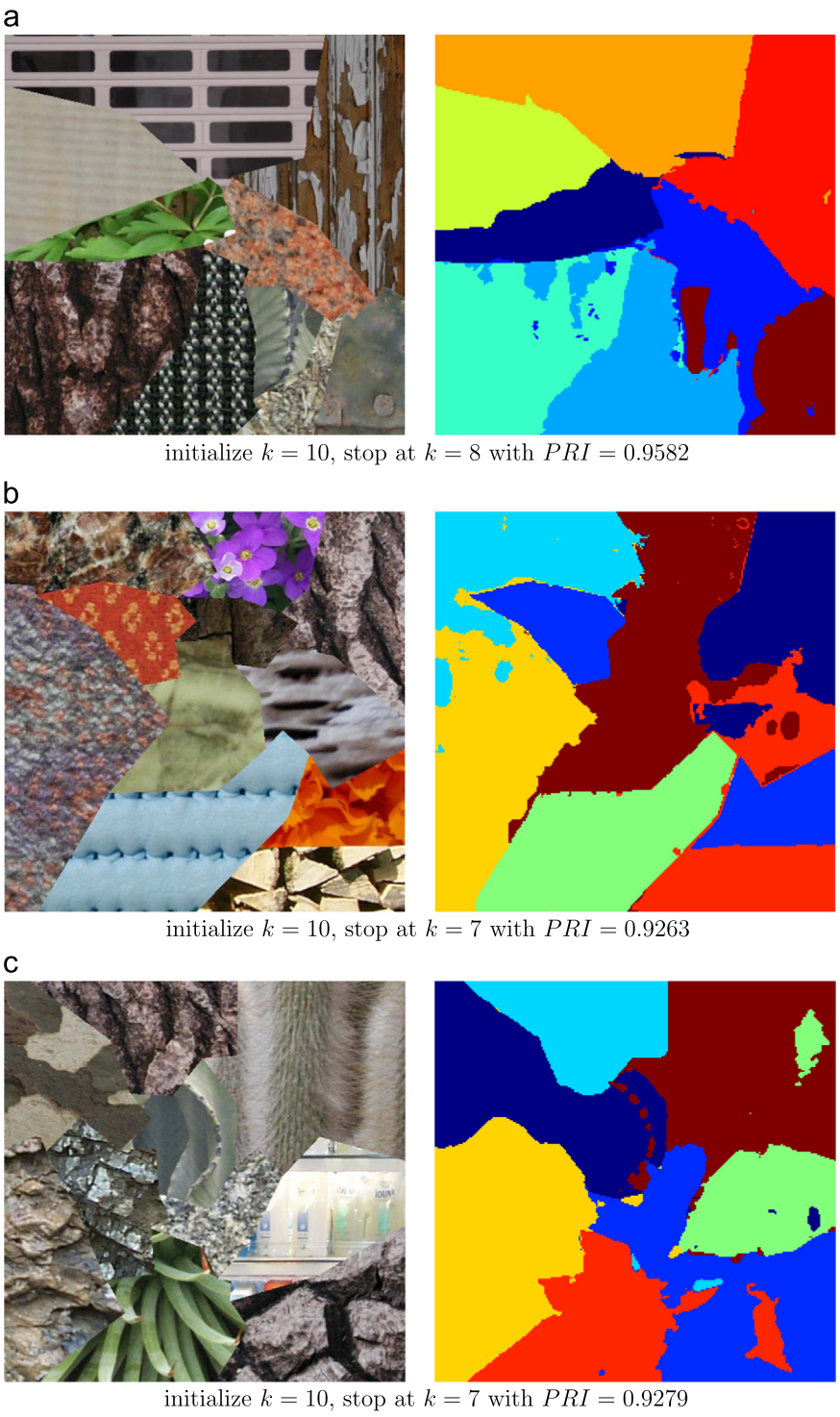

Fig. 6. Segmentation examples of *Prague-generated multi-texture images*: part I. Left column: original images. Right column: segmentation images.

coordinate y. In fact, these position features are also important to the overall segmentation of an object in an image.

As shown by Ooi and Lim [27], the CIE color representation could be the best for image segmentation. So, we adopt the CIE-LUV color space and each pixel is represented by a tri-tuple $\{L^*, u^*, v^*\}$, where L^* is the initialism of lightness, while u^* and v^* are color measurements. In consideration of the robustness to noises, a *mean filter* with 3×3 kernel size is applied to the original LUV color image.

5.2. Application of the BYY-t algorithm to image segmentation

Due to the ability of automated model selection and clustering analysis, the BYY-t algorithm is capable of unsupervised image segmentation. Firstly, we need to extract the contourlet texture features on each image patch. The patch size is set as 50×50 after we double-size the original image. The parameters in the subband clustering step are set as $L = 4$, $l_i = 3$, $i = 1, 2, 3, 4$ and $M = 3$, and we choose 3 principal eigenvectors in the spectral clustering step.

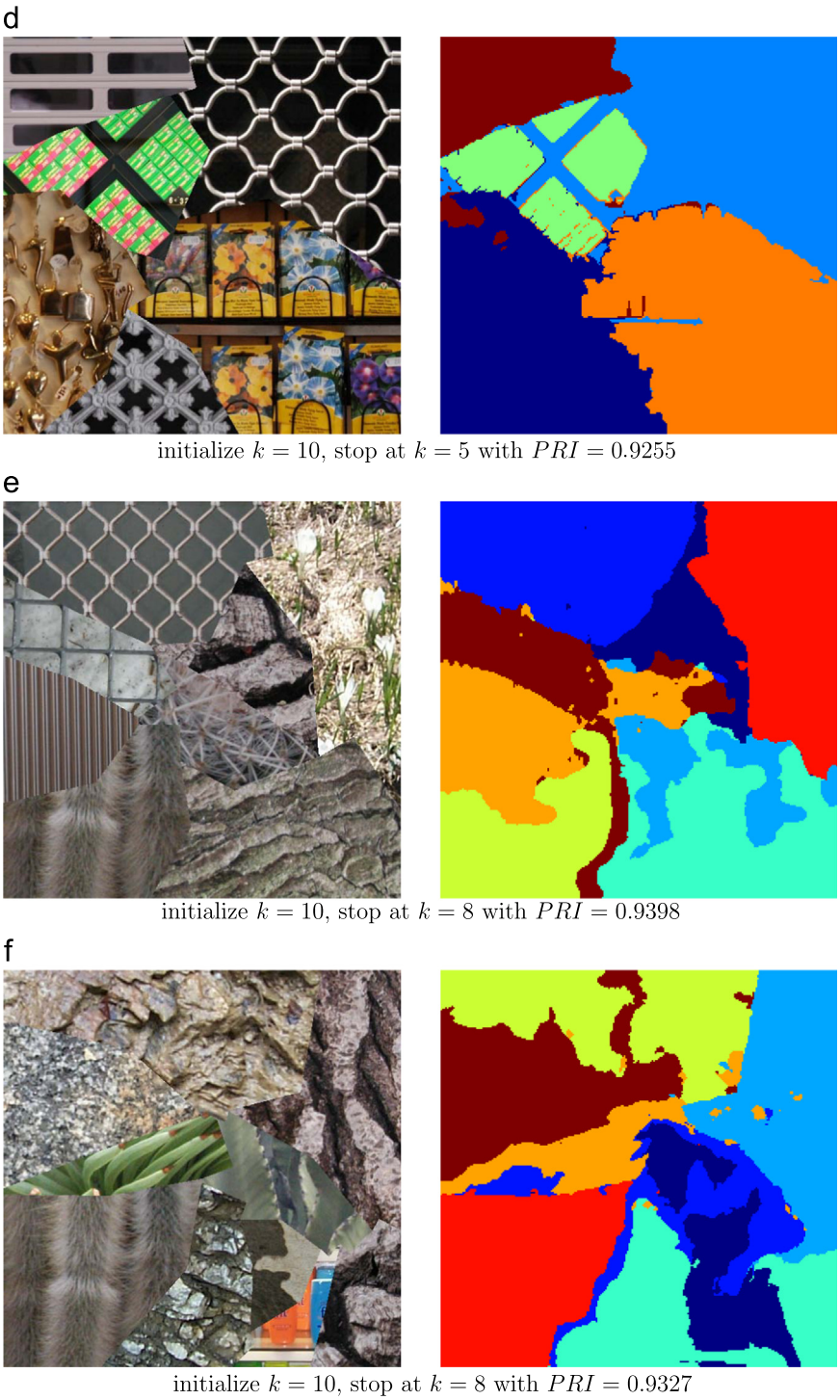

Fig. 7. Segmentation examples of *Prague-generated multi-texture images*: part II. Left column: original images. Right column: segmentation images.

Secondly, we extract 8-dimensional pixel-level feature vector for each pixel, including the contourlet texture features of its corresponding patch (3D), position features (2D) and LUV color features (3D). Finally we apply the BYY-t algorithm to the feature space of image pixels, assign each pixel to the component with the maximum posterior probability, and thus obtain the image segments. Usually, a post-process is also applied to delete "small pieces" in the segmentation results. For short, we refer to our segmentation method based on the contourlet texture features and the BYY-t algorithm as "BYY-t-CL".

5.2.1. Segmentation performance for the general images

To evaluate the performance of BYY-t-CL, we test it on the popular *Berkeley Segmentation Data Set* (*BSDS300*) [28], which covers a variety of images with complex scenarios and has manually produced ground-truth segmentation for each image. As for the evaluation of segmentation performance, we adopt the *Probability Rand Index* (PRI) [29], which is ranged in [0, 1], with the higher the better.

As in Fig. 4, there are four typical segmentation results on BSDS300 general images. In the first image, the grassland, stones and mountains have different colors, BYY-t-CL segments them

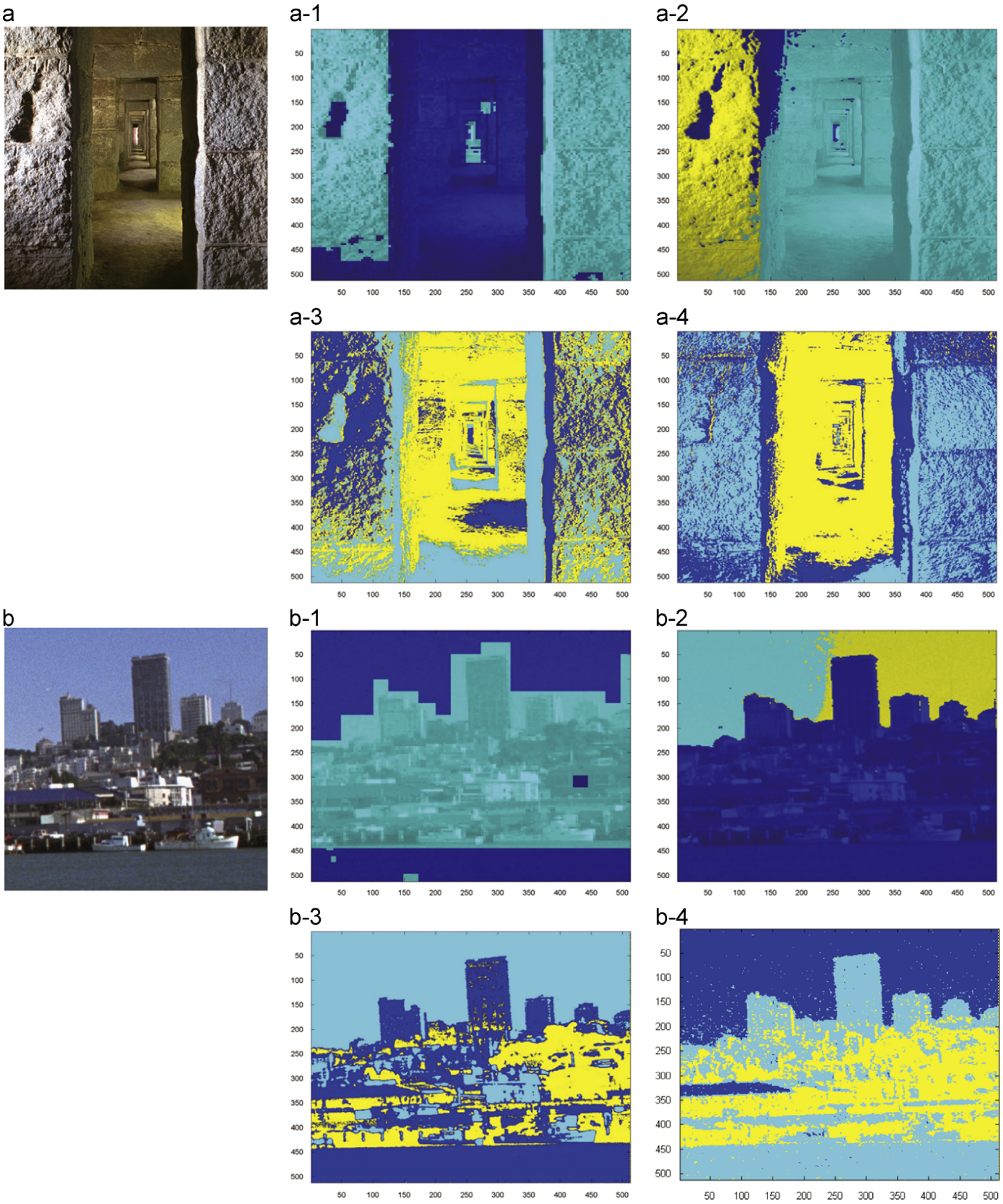

Fig. 8. Segmentation examples of *VisTex multi-texture images*. (a) Original corridor image. (b) Original city-sea image. (∗-1) segmentation by BYY-t-CL; (∗-2) segmentation by BYY-t-DCT; (∗-3) segmentation by Gaussian-EM algorithm; (∗-4) segmentation by the k-means algorithm, where ∗ refers to (a) or (b).

clearly. Moreover, even if the mountains with different distances have similar colors, but with different brightness, BYY-t-CL still distinguishes them correctly. This shows that BYY-t-CL is effective for the general images with different colors and brightness, which may be ascribed to the utilization of LUV color features.

In the second image, the skin of the gecko has similar color and brightness with its surrounding stones, but BYY-t-CL differentiates them almost correctly due to the small difference between their texture patterns. This indicates that BYY-t-CL can *grab* the texture information and distinguish them by the texture variations. It also shows the effectiveness of the contourlet texture features. The relatively low PRI score is mainly caused by the legs and tail of the gecko since they are in slender shapes and it is quite difficult to extract the patch-based texture features accurately for them.

As for the third image and the fourth image, both the skin of the fish and the clothes on the person have dazzling spots with much different colors, but they form obvious texture patterns as a whole. From the segmentation results we can see that BYY-t-CL clearly distinguishes them from the surroundings, without splitting them into small pieces. This indicates that BYY-t-CL is more sensitive to texture variation than to color variations.

Furthermore, we test BYY-t-CL on a set of 100 test images in the BSDS300. k is set as 12, 10, 8, or 6, differently in consideration of the complexity of the images. Table 5 lists the *mean* and *standard deviation* of the 100 PRI scores, as well as the comparisons with some relative segmentation methods, e.g., the Rival Penalized Competitive Learning (RPCL) algorithm [7,24], the Competitive Agglomeration Clustering (CAC) algorithm [30]. Besides, we also conduct a comparative experiment by replacing the contourlet texture features with the Discrete Cosine Transformation (DCT) texture features [31] in the BYY-t algorithm, which is denoted as "BYY-t-DCT" for short. From the Table 5 we can see that the two BYY-t approaches achieve better segmentation results than the RPCL and CAC algorithms (with a higher mean of PRI), and are more stable (with a lower standard deviation of PRI). Besides, in comparison with the DCT texture features, the contourlet texture features really improve the segmentation performance. Fig. 5 shows the percentages of different PRI levels. From this figure we can see that about 70% images obtain relatively good segmentation results with PRI larger than 0.7, and only 2% images have very bad segmentation results with PRI lower than 0.5.

5.2.2. Segmentation performance for multi-texture images

To further investigate the effects of the contourlet texture features in our BYY-t-CL approach, we test it on some multi-texture images, including synthetic images and real-world images. Multi-texture image means that the image is composed by several regions, with each region having evident texture pattern, and the textures are uniform within regions, but different among regions.

The synthetic multi-texture images are generated by the *Prague Texture Segmentation Datagenerator* [32]. The generator randomly selects several different texture patterns from a texture database, places them to a single image with random shapes, and thus obtains the multi-texture image. Some segmentation examples by BYY-t-CL are shown in Figs. 6 and 7, where the initial number of clusters is fixed to 10. From these two figures we can see that all the synthetic images are composed by regions with ruleless shapes, and each region corresponds a texture pattern. Most texture patterns are *in large scales*, with cluttered colors or objects, and thus very difficult for recognition. However, in most cases, our method recognizes the texture pattern accurately, and segments the images by texture variations. The PRI scores on these synthetic images are all greater than 0.9, which are much higher than the mean PRI on BSDD images. This indicates that BYY-t-CL is more suitable for multi-texture images than for general images, and it mainly benefits from the effective contourlet texture features we used in the BYY-t algorithm.

The real-world multi-texture images are come from the *Vision Texture Database* [33], which provides texture images that are representative of real world conditions. Two segmentation examples are shown in Fig. 8, as well as the comparisons with three other algorithms,i.e., BYY-t-DCT, Gaussian-EM algorithm [34] and the k-means algorithm. The initial number of clusters is set as 3.

As is shown in Fig. 8(a), the walls on both sides of the corridor have consistent texture patterns but varied colors and brightness. the Gaussian-EM and k-means algorithms are too sensitive to image color and brightness, and assign the walls to different clusters. In comparison, BYY-t-CL captures the texture features in the image, correctly obtains the actual number of clusters and segments the image into two parts,i.e., wall and corridor.

Fig. 8 (b) shows the scenery of a city around the sea. Since the color of sea is more similar with buildings than with sky, the other algorithms are more potential to group the sea and buildings together. In contrast, BYY-t-CL is more discriminating with texture variations, and correctly distinguishes the city from the sea. However, the segmentation boundaries via BYY-t-CL are still coarse (almost with square wave shape) due to the patch-based texture feature extraction strategy and need to be further improved.

In summary, the BYY-t algorithm is successfully applied to unsupervised image segmentation as the contourlet features are mainly utilized. The experimental results show that it leads to more reasonable and accurate segmentations for both general and multi-texture images, especially when the textures are involved.

6. Conclusions

We have extended the BYY harmony learning to the case of multivariate t-mixtures and derived a gradient BYY harmony learning algorithm for t-mixtures. In fact, our proposed algorithm can automatically determine the number of actual t-distributions in the dataset during the parameter learning. The simulation experiments have demonstrated its effectiveness for automated model selection and parameter estimation. Moreover, our proposed algorithm is successfully applied to unsupervised image segmentation as the contourlet features are mainly utilized. It is demonstrated by the experiments that this new segmentation approach is more discriminating with texture variations, and gains better segmentation performances for both general and multi-texture images. Since texture is often more consistent than color and brightness with image objects, this new approach is more suitable for the segmentation of images with complex and varied contents.

Acknowledgments

This work was supported by the Natural Science Foundation of China for Grant 61171138.

References

[1] H. Akaike., A new look at statistical model identification, IEEE Trans. Autom. Control 19 (1974) 716–723.
[2] G. Scharz, Estimating the dimension of a model, Ann. Stat. 6 (1978) 4612–4664.
[3] J. Rissanen, Modeling by shortest data description, Automatica 14 (1978) 465–471.
[4] M.D. Escobar, M. West, Bayesian density estimation and inference using mixtures, J. Am. Stat. Assoc. 90 (430) (1995) 577–588.
[5] M. Har-even, V.L. Brailovsky, Probabilistic validation approach for clustering, Pattern Recognit. Lett. 16 (1995) 1189–1196.
[6] M.A.T. Figueiredo, A.K. Jain, Unsupervised learning of finite mixture models, IEEE Trans. Pattern Anal. Mach. Intell. 24 (3) (2002) 381–396.
[7] L. Xu, A. Krzyzak, E. Oja, Rival penalized competitive learning for clustering analysis, RBF net and curve detection, IEEE Trans. Neural Netw. 4 (4) (1993) 636–648.
[8] Y.M. Cheung, k*-means: a new generalized k-means clustering algorithm, Pattern Recognit. Lett. 24 (2003) 2883–2893.
[9] Y.M. Cheung, Maximum weighted likelihood via rival penalized em for density mixture clustering with automatic model selection, IEEE Trans. Knowl. Data Eng. 17 (6) (2005) 705–761.

[10] C. Zhang, J. Tang, B. Luo, Automatic t-mixture model selection via rival penalized EM, in: Proceedings of the 6th International Conference on Hybrid Intelligent System, IEEE, 2006.
[11] L. Xu, Ying-yang machine: a Bayesian-Kullback scheme for unified learnings and new results on vector quantization, in: Proceedings of 1995 International Conference on Neural Information Processing, vol. 2, 1995, pp. 977–988.
[12] L. Xu, BYY harmony learning, structural RPCL, and topological self-organizing on mixture modes, Neural Netw. 15 (2002) 1231–1237.
[13] J. Ma, T. Wang, L. Xu., A gradient BYY harmony learning rule on gaussian mixture with automated model selection, Neurocomputing 56 (2004) 481–487.
[14] 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 (5) (2005) 701–713.
[15] J. Ma, L. Wang, BYY harmony learning on finite mixture: adaptive gradient implementation and a floating RPCL mechanism, Neural Process. Lett. 24 (1) (2006) 19–40.
[16] 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.
[17] J. Ma, J. Liu, Z. Ren, Parameter estimation of poisson mixture with automated model selection through BYY harmony learning, Pattern Recognit. 42 (2009) 2659–2670.
[18] Z. Ren, J. Ma, BYY harmony learning on weibull mixture with automated model selection, Lecture Notes in Computer Science 5263 (2008) 589–599.
[19] W. Zheng, Z. Ren, Y. Zhou, J. Ma, BYY harmony learning of log-normal mixtures with automated model selection, Neurocomputing 151 (Part 3) (2015) 1015–1026, http://dx.doi.org/10.1016/j.neucom.2014.04.080.
[20] S. Chatzis, T. Varvarigou, Robust fuzzy clustering using mixtures of student's-t distributions, Pattern Recognit. Lett. 29 (2008) 1901–1905.
[21] G. Sfikas, C. Nikou, N. Galatsanos, Robust image segmentation with mixtures of student's t-distribution, in: Proceedings of the 14th International Conference on Image Processing, vol. 1, San Antonio, TX, USA, 2007, pp. 273–276.
[22] D. Peel, G.J. Mclachlan, Robust mixture modelling using the t distribution, Stat. Comput. 10 (2000) 339–348.
[23] P.J. Green, Bayesian reconstruction from emission tomography data using a modified em algorithm, IEEE Trans. Med. Imaging 9 (1) (1990) 84–93.
[24] 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.
[25] Y. Dong, J. Ma, Feature extraction through contourlet subband clustering for texture classification, Neurocomputing 116 (0) (2013) 157–164.
[26] T. Xiang, S. Gong, Spectral clustering with eigenvector selection, Pattern Recognit. 41 (3) (2008) 1012–1029.
[27] W.S. Ooi, C.P. Lim, Fusion of colour and texture features in image segmentation: an empirical study, Imaging Sci. J. 57 (2009) 8–18.
[28] D. Martin, C. Fowlkes, D. Tal, J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, in: Proceedings of International Conference on Computer Vision, vol. 2, 2005, pp. 416–423.
[29] R. Unnikrishnan, C. Pantofaru, M. Hebert, Toward objective evaluation of image segmentation algorithms, IEEE Trans. Pattern Anal. Mach. Intell. 29 (6) (2007) 929–944.
[30] H. Frigui, R. Krishnapuram, Clustering by competitive agglomeration, Pattern Recognit. 30 (7) (1997) 1109–1119.
[31] W. Chen, M.J. Er, S. Wu, Illumination compensation and normalization for robust face recognition using discrete cosine transform in logarithm domain, IEEE Trans. Syst. Man Cybern.: Part B 36 (2) (2006) 458–466.
[32] M. Haindl, S. Mikes, The prague texture segmentation datagenerator and benchmark, ⟨http://www.mosaic.utia.cas.cz⟩.
[33] R. Picard, C. Graczyk, S. Mann, et al., Vision texture database, ⟨http://www.vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html⟩.
[34] Z. Fu, L. Wang, Multimedia and signal processing, Commun. Comput. Inf. Sci. 346 (2002) 61–66.

Yunsheng Jiang received his B.S. degree in information and computing sciences from the School of Mathematical Sciences at Dalian University of Technology in 2011, and he is now a Ph.D. student in Applied Mathematics at Peking University. His main research interests include object detection, computer vision, pattern recognition, learning theory and algorithm.

Chenglin Liu received her B.S. degree from the Department of Applied Mathematics at School of Science, China University of Petroleum (East China) in 2009, and Ph.D. degree in applied mathematics from Peking University in 2014. From March 2011 to August 2013, she visited as a research fellow in Houston Methodist Hospital Research Institute and Wake Forest Medical School (NC, USA). She is now a postdoctoral researcher in Shanghai Jiao Tong University. Her main research interests include bioinformatics, pattern recognition, learning theory and algorithm.

Jinwen Ma received the 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 an 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. advisor. During 1995 and 2003, he also visited several times the Department of Computer Science & Engineering, the Chinese University of Hong Kong as a research associate or fellow. He also worked as research scientist at Amari Research Unit, RIKEN Brain Science Institute, Japan from September 2005 to August 2006. He has published over 100 academic papers on neural networks, pattern recognition, artificial intelligence, and information theory.