Signal Processing 91 (2011) 801–820

Contents lists available at ScienceDirect

Signal Processing

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

Bayesian learning of finite generalized Gaussian mixture models on images

Tarek Elguebaly, Nizar Bouguila *

Concordia Institute for Information Systems Engineering, Faculty of Engineering and Computer Science, Concordia University, Montreal, QC, Canada H3G 2W1

ARTICLE INFO

Article history:
Received 28 October 2009
Received in revised form
14 May 2010
Accepted 30 August 2010
Available online 19 September 2010

Keywords:
Generalized Gaussian distribution
Mixture modeling
Bayesian analysis
Gibbs sampling
Metropolis-Hastings
Steerable model
Histogram
Texture classification
Image segmentation

ABSTRACT

This paper presents a fully Bayesian approach to analyze finite generalized Gaussian mixture models which incorporate several standard mixtures, widely used in signal and image processing applications, such as Laplace and Gaussian. Our work is motivated by the fact that the generalized Gaussian distribution (GGD) can be applied on a wide range of data due to its shape flexibility which justifies its usefulness to model the statistical behavior of multimedia signals [1]. We present a method to evaluate the posterior distribution and Bayes estimators using a Gibbs sampling algorithm. For the selection of number of components in the mixture, we use the integrated likelihood and Bayesian information criteria. We validate the proposed method by applying it to: synthetic data, real datasets, texture classification and retrieval, and image segmentation; while comparing it to different other approaches.

1. Introduction

Finite mixtures are a flexible and powerful probabilistic tool for modeling data [2]. Mixture models are very useful in areas where statistical modeling of data is needed such as in signal and image processing, pattern recognition, bioinformatics, computer vision, and machine learning. The three main problems in mixture modeling are the choice of the probability density function (pdf), the parameters estimation and the selection of the number of clusters. In most of the applications, the Gaussian density is used in the mixture modeling of data. However, many signal processing systems often operate in environments characterized by non-Gaussian and highly peaked sources (subband image and speech coefficients, for instance) [1,3–5]. An interesting approach involving very general parametric models (i.e. statistical distribution), based on Pearson's system, has been proposed in [6] to model non-Gaussian data. Moreover, many studies have shown that the GGD, can be a good alternative to the Gaussian thanks to its shape flexibility which allows the modeling of a large number of non-Gaussian signals [7–10]. The GGD contains the Laplacian, the Gaussian and asymptotically the uniform distribution as special cases [11] and has been used, for instance, in [12,4] to fit subband histograms, in [13] for multiresolution transmission of high-definition video, in [14] for subband decomposition of video, in [15] for buffer control, in [16–18] for texture classification and retrieval, in [19] for denoising applications, in [20,21] for data and image compression, in [22] for edge modeling, in [23,24] for image thresholding, in [25,26] for speech modeling, in [27–29] for video and image segmentation, in [30] for SAR images statistics modeling, and in [31] for multichannel audioresynthesis.

Several approaches have been considered in the past to estimate GGD's parameters such as moment estimation [32,14,33], entropy matching estimation [34,26], and

* Corresponding author.
E-mail addresses: t_elgue@encs.concordia.ca (T. Elguebaly), bouguila@ciise.concordia.ca (N. Bouguila).

0165-1684/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.sigpro.2010.08.014

maximum likelihood estimation [32,16,35,36,10]. It is noteworthy that these approaches consider a single distribution. Concerning finite mixture model parameters estimation, approaches can be classified into two categories: deterministic and Bayesian methods. In deterministic approaches, parameters are taken as fixed and unknown, and inference is based on the likelihood of the data. Some deterministic approaches have been proposed in the past for the estimation of finite generalized Gaussian mixture (GGM) model parameters (see, for instance [28,29]). Despite the fact that deterministic approaches, such as the expectation–maximization (EM) algorithm [37], have dominated mixture models estimation due to their small computational time, many works have proved that these methods have severe problems such as convergence to local maxima, and the tendency to complicate the resulted models (i.e overfitting) [38] especially when data are sparse or noisy. Several stochastic versions of the EM algorithm have been introduced to overcome these problems. Examples include the stochastic EM (SEM) [39], the stochastic approximation EM (SAEM) [40], the iterated conditional expectation (ICE) [41] and the Monte Carlo EM (MCEM) [42]. With the evolution of computational tools, signal and image processing researchers were encouraged also to develop and use pure Bayesian Markov chain Monte Carlo (MCMC) methods and techniques as an alternative approach. In Bayesian methods, parameters are considered random, and follow different probability distributions (prior distributions). These distributions describe our knowledge before considering the data, as for updating our prior beliefs the likelihood is used. For interesting and in depth discussions about the general Bayesian theory refer to [38,43].

To the best of our knowledge the learning techniques that have been proposed for the GGM are deterministic and then usually excessively sensitive to noise. Thus, we propose in this paper a novel Bayesian approach to evaluate the posterior distribution of GGM and then learn its parameters using Gibbs sampling [44] for the estimation and the integrated likelihood for the selection of the optimal number of components. To validate our learning algorithm, we compare it to four different stochastic techniques namely SEM, SAEM, MCEM, and ICE using synthetic data, real datasets, and real world applications involving texture classification and retrieval, and image segmentation.

The rest of this paper is organized as follows. The next section describes the GGM Bayesian estimation algorithm. In Section 3, we assess the performance of the new model on different applications. Our last section is devoted to the conclusion and some perspectives.

2. The finite GGM and Bayesian estimation

2.1. Finite GGM model

If the random variable $x \in \mathbb{R}$ follows a GGD with parameters μ, α and β, then the density function is given by [12,14]

$$P(x|\mu,\alpha,\beta) = \frac{\beta\alpha}{2\Gamma(1/\beta)} e^{-(\alpha|x-\mu|)^{\beta}} \quad (1)$$

where $\alpha = (1/\sigma)\sqrt{\Gamma(3/\beta)/\Gamma(1/\beta)}$, $-\infty < \mu < \infty$, $\beta > 0$, $\sigma > 0$, $\alpha > 0$, and $\Gamma(\cdot)$ is the Gamma function given by: $\Gamma(x) = \int_0^{\infty} t^{x-1} e^{-t}\, dt$, $x > 0$. μ, α, σ and β denote the distribution mean, the inverse scale parameter, the standard deviation, and the shape parameter, respectively. The parameter β controls the shape of the pdf. The larger the value, the flatter the pdf; and the smaller the value, the more picked the pdf. This means that β determines the decay rate of the density function (see Fig. 1). Note that for the two special cases, when $\beta = 2$ and 1, the GGD is reduced to the Gaussian and Laplacian distributions, respectively. If x follows a mixture of M GGDs, then

$$P(x|\Theta) = \sum_{j=1}^{M} P(x|\mu_j,\alpha_j,\beta_j)p_j \quad (2)$$

where p_j $(0 \le p_j \le 1$ and $\sum_{j=1}^{M} p_j = 1)$ are the mixing proportions and $p(x|\mu_j,\alpha_j,\beta_j)$ is the GGD describing component j. As for the symbol $\Theta = (\xi,p)$, it refers to the entire set of parameters to be estimated, knowing that $\xi = (\mu_1,\alpha_1,\beta_1,\ldots,\mu_M,\alpha_M,\beta_M)$, and $p=(p_1,\ldots,p_M)$.

Consider N observations, $\mathcal{X} = (x_1,\ldots,x_N)$, the well-known approach to estimate the parameters of a mixture model is to maximize the likelihood through the EM algorithm, supposing that the number of mixture components M is known. The likelihood corresponding to this case is

$$P(\mathcal{X}|\Theta) = \prod_{i=1}^{N}\sum_{j=1}^{M} P(x_i|\xi_j)p_j \quad (3)$$

where $\xi_j = (\mu_j,\alpha_j,\beta_j)$. For each variable x_i, let Z_i be an M-dimensional vector known by the unobserved or missing vector that indicates to which component x_i belongs, such that: Z_{ij} will be equal to 1 if x_i belongs to class j or 0, otherwise. The complete-data likelihood is then

$$P(\mathcal{X},Z|\Theta) = \prod_{i=1}^{N}\sum_{j=1}^{M} (P(x_i|\xi_j)p_j)^{Z_{ij}} \quad (4)$$

where $Z=\{Z_1,Z_2,\ldots,Z_N\}$. The EM algorithm consists of getting the mixture parameters that maximize the log-likelihood function given by

$$L(\Theta,Z,\mathcal{X}) = \sum_{j=1}^{M}\sum_{i=1}^{N} Z_{ij}\log(P(x_i|\xi_j)p_j) \quad (5)$$

by replacing each Z_{ij} by its expectation, defined as the posterior probability that the ith observation arises from the

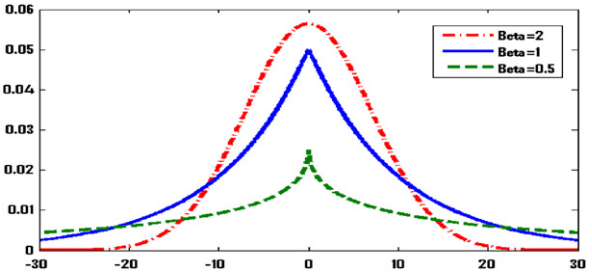

Fig. 1. Generalized Gaussian distributions with different values of the shape parameter.

jth component of the mixture as follows:

$$\hat{Z}_{ij}^{(t)} = \frac{P^{(t-1)}(x_i|\xi_j^{(t-1)})p_j^{(t-1)}}{\sum_{k=1}^{M} P^{(t-1)}(x_i|\xi_k^{(t-1)})p_k^{(t-1)}} \quad (6)$$

where t denotes the current iteration step and $\xi_j^{(t)}$ and $p_j^{(t)}$ are the current evaluations of the parameters. The EM produces a sequence of estimates to the mixture parameters Θ^t, for t=0,1,…, until a certain convergence criterion is satisfied through two different steps: the expectation and maximization. The EM algorithm consists of:

(1) Initialization of the mixture parameters.
(2) E-step: Compute $\hat{Z}_{ij}^{(t)}$ (Eq. (6)) using the initialized parameters.
(3) M-step: Update parameters estimates using: $\hat{\Theta}^{(t)} = \text{argmax}_{\Theta} L(\Theta, Z, \mathcal{X})$.

However, the EM has some drawbacks, like convergence to local maxima due to its dependence on the initialization step. For a detailed and interesting discussion about EM disadvantages refer to [37]. Several extensions have been proposed to address these disadvantages. For instance, a stochastic extension called SEM was elaborated by Celeux and Diebolt [39]. The SEM algorithm consists in fact of a modification of the EM algorithm in which a probabilistic teacher step (Stochastic step or S-Step) has been incorporated. It consists of the simulation of Z according to a posterior probability chosen to be Multinomial of order one with weights given by the $\hat{Z}_{ij}$ ($\mathcal{M}(1;\hat{Z}_{i1},\ldots,\hat{Z}_{iM})$). An adaptation of the SEM algorithm called SAEM has been proposed in [40] to reach almost sure convergence to a local maximum (more details and discussions can be found in [40,45]). An extension to the EM algorithm based on the execution of the E-step by Monte Carlo, called MCEM, has been proposed in [42]. Moreover, an iterative estimation method, called ICE and based on the conditional expectation rather than the notion of likelihood, was introduced by Pieczynski in [41] for the statistical segmentation of images and studied in [46]. An efficient alternative technique that we will propose in the following is the Bayesian approach which has received a lot of attention recently thanks to the evolution of MCMC computational tools.

2.2. Bayesian estimation of the GGM

Simulation methods like MCMC algorithms are chosen as a solution to overcome the problems of numerical methods. Generally these methods are related to the Bayesian theory, which means that they allow for probability statements to be made directly about the unknown parameters of the mixture, while taking into consideration prior or expert opinion. The goal is to get the posterior distribution $\pi(\Theta|\mathcal{X},Z)$, by combining the prior information about the parameters, $\pi(\Theta)$, with the observed value or realization of the complete data $P(\mathcal{X},Z|\Theta)$, which is derived from Bayes formula:

$$\pi(\Theta|\mathcal{X},Z) = \frac{\pi(\Theta)P(\mathcal{X},Z|\Theta)}{\int_{\Theta}\pi(\Theta)P(\mathcal{X},Z|\Theta)\,d\Theta} \propto \pi(\Theta)P(\mathcal{X},Z|\Theta) \quad (7)$$

where $(\mathcal{X},Z)$ is the complete data. Having the joint distribution, $\pi(\Theta)P(\mathcal{X},Z|\Theta)$, we can deduce the posterior distribution (Eq. (7)). With $\pi(\Theta|\mathcal{X},Z)$ in hand we can simulate our model parameters Θ, rather than computing them. The Gibbs sampler is a well known simulation technique [44] and it is based on the successive simulation of Z, p, and ξ conditional on each other and on the observations which offers an efficient way to explore the parameter space. The standard Gibbs sampler for mixture models consists of:

(1) Initialization: choose p^0 and ξ^0.
(2) Step t, for t=1,…
  (a) Generate $Z^{(t)}$ from $\pi(Z|\Theta^{(t-1)},\mathcal{X})$.
  (b) Generate $p^{(t)}$ from $\pi(p|Z^{(t)})$.
  (c) Generate $\xi^{(t)}$ from $\pi(\xi|Z^{(t)},\mathcal{X})$.

We simulate Z according to the posterior probability $\pi(Z|\Theta,\mathcal{X})$, chosen to be Multinomial of order one with a weight given by $\hat{Z}_{ij}(\mathcal{M}(1;\hat{Z}_{i1};\ldots;\hat{Z}_{iM}))$. This choice is due to two reasons, first, we know that each Z_i is a vector of zero–one indicator variables to define from which component j the x observation arises. Second, the probability that the ith observation, x_i, arises from the jth component of the mixture is given by $\hat{Z}_{ij}$. So, we can deduce that each vector Z_i is generated by a Multinomial distribution of order one with weight given by $\hat{Z}_{ij}$. Now to simulate p we need to get $\pi(p|Z^{(t)})$, using Bayes

Table 1
Parameters for the different generated datasets.

	j	μ_j	α_j	β_j	p_j	$\hat{\mu}_j$	$\hat{\alpha}_j$	$\hat{\beta}_j$	$\hat{p}_j$
Data 1 (N=262 144)	1	100.0000	0.0778	1.7000	0.7800	100.0086	0.0704	1.7300	0.7799
	2	200.0000	0.0406	2.3000	0.2200	200.0248	0.0434	2.2900	0.2201
Data 2 (N=65 536)	1	63.6283	0.0700	2.1000	0.1458	65.1100	0.0692	2.0000	0.1523
	2	104.6854	0.0603	1.9000	0.7370	106.0010	0.0658	1.9400	0.7222
	3	195.2122	0.0333	1.7000	0.1172	196.5765	0.0340	1.6600	0.1255
Data 3 (N=97 344)	1	33.1256	0.0500	2.0000	0.1721	33.7487	0.0509	1.9899	0.1768
	2	95.5550	0.0450	2.4000	0.2000	95.7038	0.0434	2.4368	0.1928
	3	150.5876	0.0650	3.5000	0.3700	150.0495	0.0636	3.7778	0.3751
	4	185.9900	0.0400	3.1000	0.2579	185.3104	0.0409	3.1177	0.2552

N represents the number of elements in each dataset. μ_j, α_j, β_j, and p_j are the real parameters. $\hat{\mu}_j$, $\hat{\alpha}_j$, $\hat{\beta}_j$, and $\hat{p}_j$ are the estimated parameters.

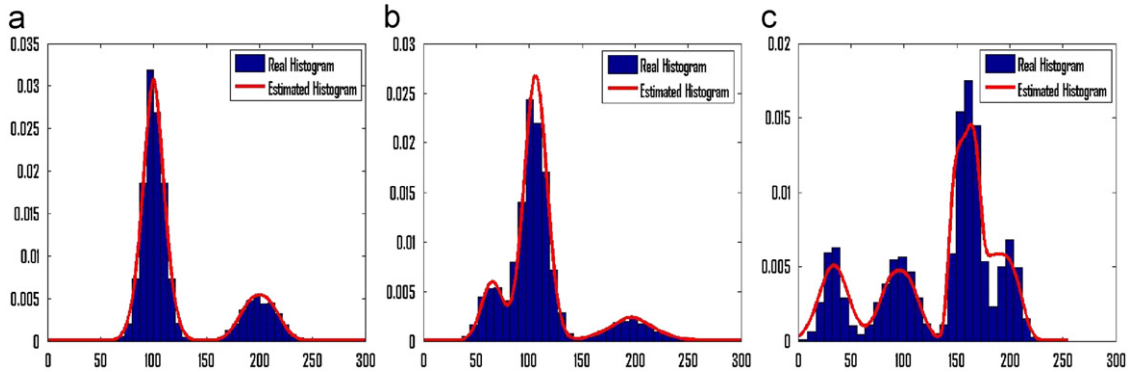

Fig. 2. Real and estimated histograms for the three datasets. (a) First dataset with $M=2$, (b) second dataset with $M=3$, (c) third dataset with $M=4$.

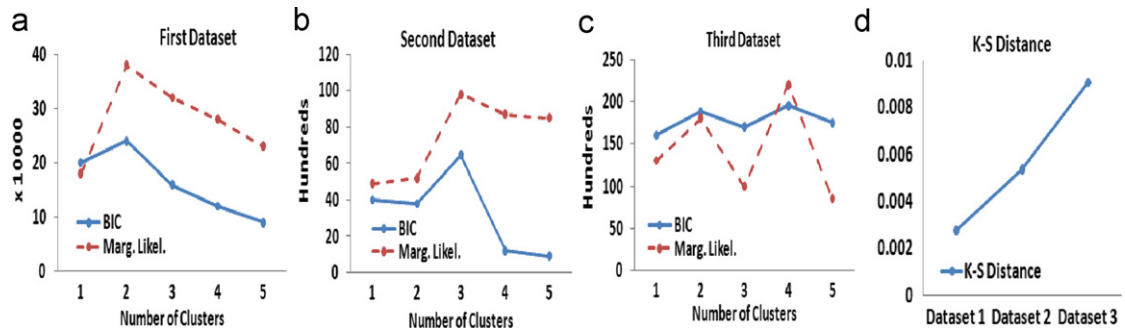

Fig. 3. Marginal likelihood and BIC values for the three datasets with different number of clusters and the Kolmogoroff–Smirnov distances. (a) First dataset, (b) second dataset, (c) third dataset, (d) Kolmogoroff–Smirnov distances for the three datasets.

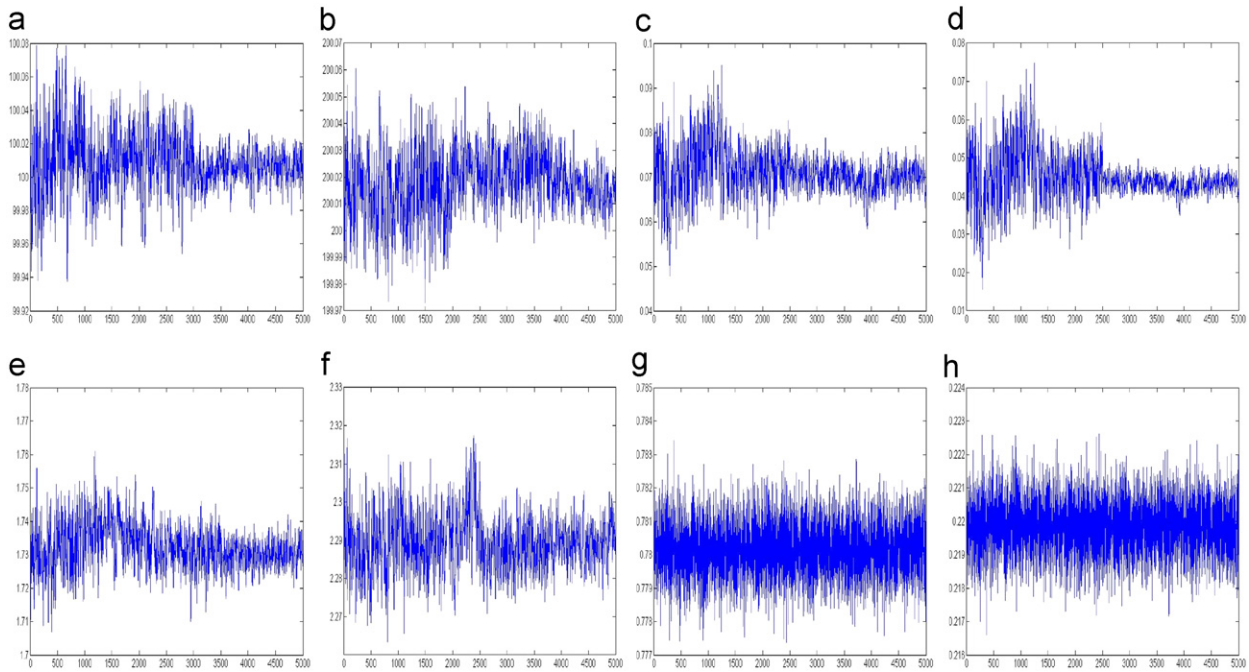

Fig. 4. Time series plot of Gibbs-within-Metropolis iterations for first dataset. (a) Iterations for $\hat{\mu}_1$, (b) iterations for $\hat{\mu}_2$, (c) iterations for $\hat{\alpha}_1$, (d) iterations for $\hat{\alpha}_2$, (e) iterations for $\hat{\beta}_1$, (f) iterations for $\hat{\beta}_2$, (g) iterations for $\hat{p_1}$, (h) iterations for $\hat{p_2}$.

rule: $\pi(p|Z)=\pi(Z|p)\pi(p)/\int_p \pi(Z|p)\pi(p)\,dp \propto \pi(Z|p)\pi(p)$. This means that we need to determine $\pi(Z|p)$, and $\pi(p)$. It is well known that the vector p is defined as ($\sum_{j=1}^{M} p_j = 1$, where $p_j \geq 0$), then the commonly considered choice as a prior is the Dirichlet distribution [38,43]:

$$\pi(p)=\frac{\Gamma\left(\sum_{j=1}^{M}\eta_j\right)}{\prod_{j=1}^{M}\Gamma(\eta_j)}\prod_{j=1}^{M}p_j^{\eta_j-1} \tag{8}$$

where $(\eta_1,\ldots,\eta_M)$ is the parameter vector of the Dirichlet distribution. As for $\pi(Z|p)$ we have

$$\pi(Z|p)=\prod_{j=1}^{M}\pi(Z_i|p)=\prod_{i=1}^{N}\prod_{j=1}^{M}p_j^{Z_{ij}}=\prod_{j=1}^{M}p_j^{n_j} \tag{9}$$

where $n_j=\sum_{i=1}^{N}\mathbb{I}_{Z_{ij}=1}$, $\mathbb{I}_{Z_{ij}=1}=1$ if Z_{ij}=1 and 0, otherwise, then we can conclude that

$$\pi(p|Z)\propto\pi(Z|P)\pi(p)$$

$$=\frac{\Gamma\left(\sum_{j=1}^{M}\eta_j\right)}{\prod_{j=1}^{M}\Gamma(\eta_j)}\prod_{j=1}^{M}p_j^{\eta_j+n_j-1}\propto\mathcal{D}(\eta_1+n_1,\ldots,\eta_M+n_M) \tag{10}$$

where $\mathcal{D}$ denotes the Dirichlet distribution with parameters $(\eta_1+n_1,\ldots,\eta_M+n_M)$. Thus, we can deduce that the Dirichlet distribution is a conjugate prior for the mixture proportions (i.e. the prior and the posterior have the same form). As for the parameters ξ, we assigned independent

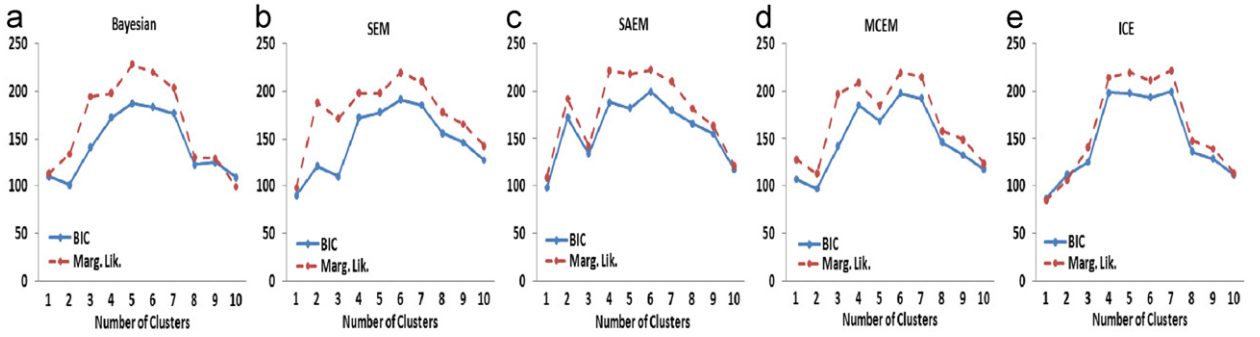

Fig. 5. Marginal likelihood and BIC values for the dataset with different number of clusters using the five algorithms. (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

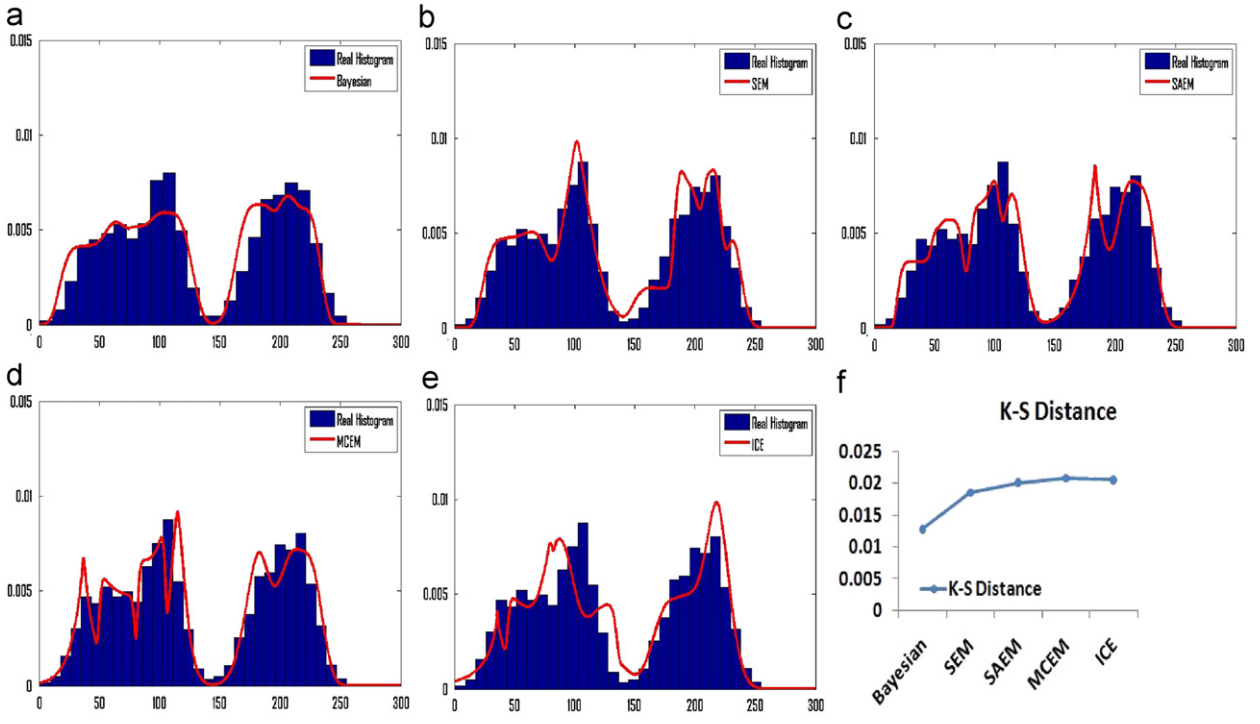

Fig. 6. Real and estimated histograms for the five components dataset using: (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm, (f) represents the Kolmogoroff–Smirnov distances of the five different approaches.

Normal priors for the distributions means, and Gamma priors for the inverse scale and shape parameters [47,48]: $\mu_j \sim \mathcal{N}(\mu_0,\sigma_0^2)$, $\beta_j \sim \mathcal{G}(\alpha_\beta,\beta_\beta)$, $\alpha_j \sim \mathcal{G}(\alpha_\alpha,\beta_\alpha)$, where $\mathcal{N}(\mu_0,\sigma_0^2)$ is the normal distribution with mean μ_0 and variance σ_0^2, $\mathcal{G}(\alpha_\beta,\beta_\beta)$ is the gamma distribution with shape parameter α_β and rate parameter β_β. $\mu_0,\sigma_0^2,\alpha_\beta,\beta_\beta,\alpha_\alpha,\beta_\alpha$ are called the hyperparameters of the model. Having these priors in hand, the posterior distributions for μ,α, and β are (see Appendix)

$$\pi(\mu_j|Z,\mathcal{X}) \propto \mathbf{e}^{-(\mu_j-\mu_0)^2/2\sigma_0^2 + \sum_{z_{ij}=1} -(\alpha_j|x_i-\mu_j|)^{\beta_j}} \tag{11}$$

$$\pi(\alpha_j|Z,\mathcal{X}) \propto \alpha_j^{\alpha_\alpha-1}\mathbf{e}^{-\beta_\alpha\alpha_j}(\alpha_j)^{n_j}\mathbf{e}^{\sum_{z_{ij}=1} -(\alpha_j|x_i-\mu_j|)^{\beta_j}} \tag{12}$$

$$\pi(\beta_j|Z,\mathcal{X}) \propto \beta_j^{\alpha_\beta-1}\mathbf{e}^{-\beta_\beta\beta_j}\left(\frac{\beta_j}{\Gamma(1/\beta_j)}\right)^{n_j}\mathbf{e}^{\sum_{z_{ij}=1} -(\alpha_j|x_i-\mu_j|)^{\beta_j}} \tag{13}$$

In this case we can notice that our posterior distributions are not in well known forms, so we cannot simulate directly from them. The Metropolis-Hastings (M-H) algorithm [47] offers a solution to this problem, and thus the complete algorithm is given by:

(1) Initialization: choose p^0 and Θ^0.
(2) Step t, for $t=1,\ldots$
  (a) Generate $Z_i^{(t)} \sim (\mathcal{M}(1;\hat{Z}_{i1}^{(t-1)},\ldots,\hat{Z}_{iM}^{(t-1)}))$.
  (b) Compute $n_j^{(t)} = \sum_{i=1}^{N} \mathbb{I}_{Z_{ij}^{(t)}}$.
  (c) Generate $p^{(t)}$ from Eq. (10).
  (d) Generate $(\mu_j,\alpha_j,\beta_j)^{(t)}$ for $(j=1,\ldots,M)$ from Eqs. (11) to (13) using M-H algorithm.

The M-H algorithm can be summarized in three steps:

(1) Generate $\left(\tilde{\mu}_j,\tilde{\alpha}_j,\tilde{\beta}_j\right) \sim q\left(\mu_j,\alpha_j,\beta_j|\mu_j^{(t-1)},\alpha_j^{(t-1)},\beta_j^{(t-1)}\right)$ and $\mathbf{U} \sim \mathcal{U}_{[0,1]}$.
(2) Compute

$$r = \pi(\tilde{\mu}_j,\tilde{\alpha}_j,\tilde{\beta}_j|Z,\mathcal{X})q(\mu_j^{(t-1)},\alpha_j^{(t-1)},\beta_j^{(t-1)}|\tilde{\mu}_j,\tilde{\alpha}_j,\tilde{\beta}_j)/\pi(\mu_j^{(t-1)},$$
$$\times\alpha_j^{(t-1)},\beta_j^{(t-1)}|Z,\mathcal{X})q(\tilde{\mu}_j,\tilde{\alpha}_j,\tilde{\beta}_j|\mu_j^{(t-1)},\alpha_j^{(t-1)},\beta_j^{(t-1)}).$$

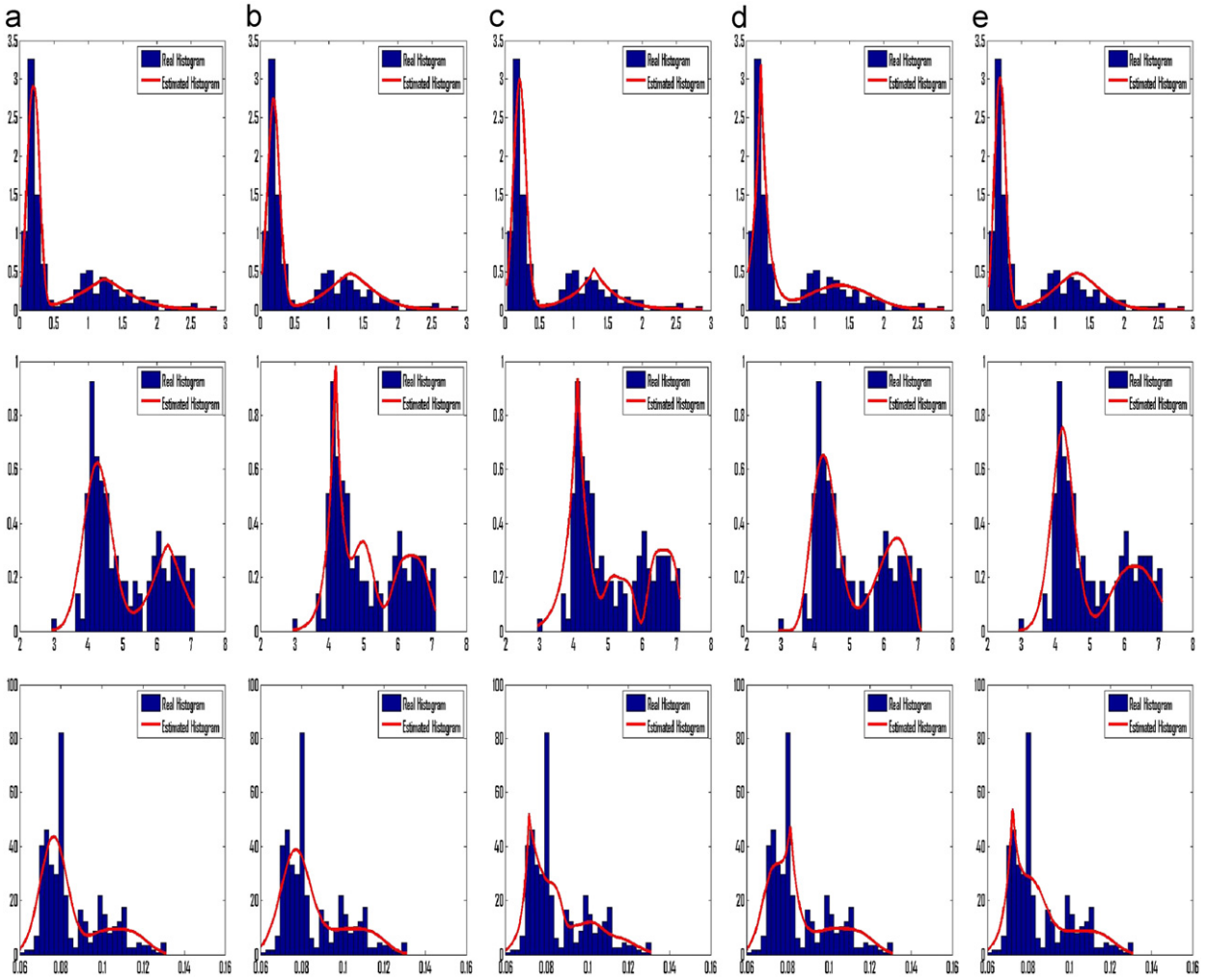

Fig. 7. Real and estimated histograms for the three real data sets (row 1: enzyme, row 2: acidity, row 3: stamp). (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

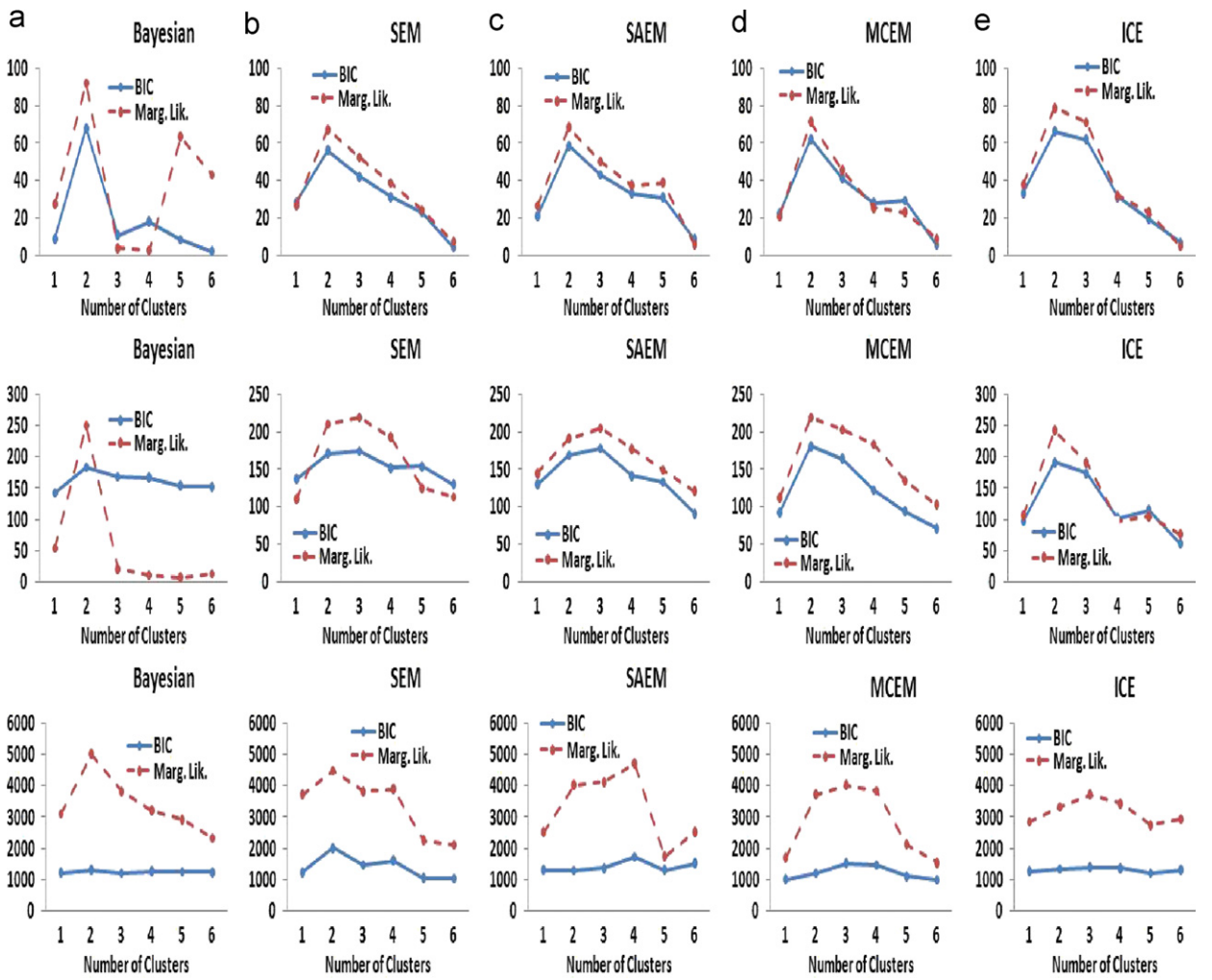

Fig. 8. The BIC and marginal likelihood of the five different algorithms for the three real datasets (row 1: enzyme, row 2: acidity, row 3: stamp). (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

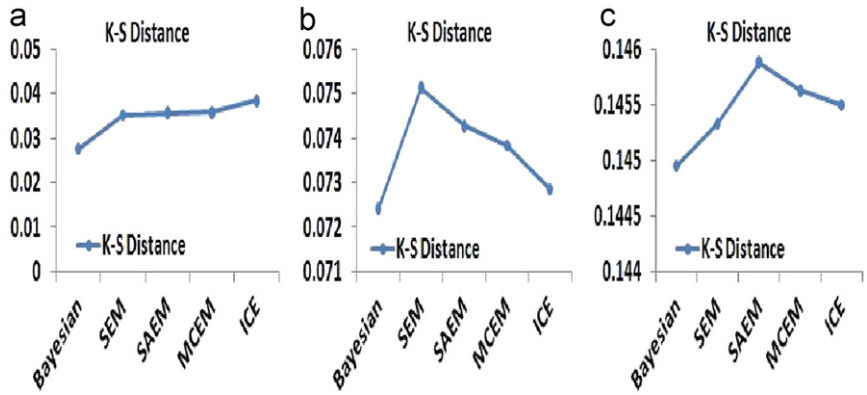

Fig. 9. The Kolmogoroff–Smirnov distances of the five different algorithms for the three real datasets. (a) Enzyme, (b) acidity, (c) stamp.

(3) If $r<\mathbf{U}$ then $(\mu_j^{(t)},\alpha_j^{(t)},\beta_j^{(t)})=(\tilde{\mu}_j,\tilde{\alpha}_j,\tilde{\beta}_j)$ else $(\mu_j^{(t)},\alpha_j^{(t)},\beta_j^{(t)})=(\mu_j^{(t-1)},\alpha_j^{(t-1)},\beta_j^{(t-1)})$.

The major problem in this algorithm is the need to choose the proposal distribution q. To solve this problem we used the most generic random walk M-H by considering the following proposals: $\tilde{\alpha}_j\sim\mathcal{LN}(\log(\alpha_j^{(t-1)}),\zeta^2)$, $\tilde{\beta}_j\sim\mathcal{LN}(\log(\beta_j^{(t-1)}),\zeta^2)$, where $\mathcal{LN}$ is the log-normal distribution, since, we know that $\tilde{\alpha}_j>0$ and $\tilde{\beta}_j>0$. As for $\tilde{\mu}_j$ we have $\tilde{\mu}_j\sim\mathcal{N}(\mu_j^{(t-1)},\zeta^2)$, where ζ^2 is the scale of the random walk. With these proposals the random walk M-H algorithm is composed of the following steps:

(1) Generate $\tilde{\mu}_j\sim\mathcal{N}(\mu_j^{(t-1)},\zeta^2)$, $\tilde{\alpha}_j\sim\mathcal{LN}(\log(\alpha_j^{(t-1)}),\zeta^2)$, $\tilde{\beta}_j\sim\mathcal{LN}(\log(\beta_j^{(t-1)}),\zeta^2)$, and $\mathbf{U}\sim\mathcal{U}_{[0,1]}$.
(2) Compute

$$r_\mu=\frac{\pi(\tilde{\mu}_j|Z,\mathcal{X})\mathcal{N}(\mu_j^{(t-1)}|\tilde{\mu}_j,\zeta^2)}{\pi(\mu_j^{(t-1)}|Z,\mathcal{X})\mathcal{N}(\tilde{\mu}_j|\mu_j^{(t-1)},\zeta^2)} \tag{14}$$

$$r_\alpha=\frac{\pi(\tilde{\alpha}_j|Z,\mathcal{X})\mathcal{LN}(\alpha_j^{(t-1)}|\log(\tilde{\alpha}_j),\zeta^2)}{\pi(\alpha_j^{(t-1)}|Z,\mathcal{X})\mathcal{LN}(\tilde{\alpha}_j|\log(\alpha_j^{(t-1)}),\zeta^2)} \tag{15}$$

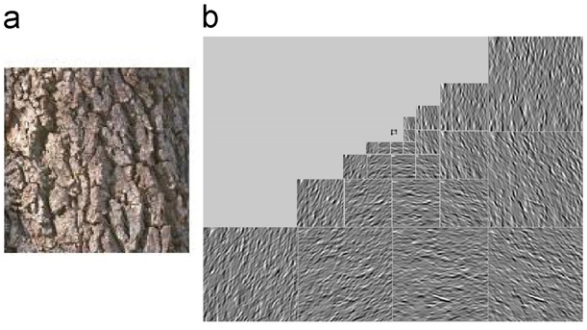

Fig. 10. Original image and its steerable pyramid decomposition. (a) Original image from Bark group in Vistex, (b) sub-images output using five level steerable pyramid.

$$r_\beta=\frac{\pi(\tilde{\beta}_j|Z,\mathcal{X})\mathcal{LN}(\beta_j^{(t-1)}|\log(\tilde{\beta}_j),\zeta^2)}{\pi(\beta_j^{(t-1)}|Z,\mathcal{X})\mathcal{LN}(\tilde{\beta}_j|\log(\beta_j^{(t-1)}),\zeta^2)} \tag{16}$$

(3)
- If $r_\mu>u$ then $\mu_j^{(t)}=\tilde{\mu}_j$, else $\mu_j^{(t)}=\mu_j^{(t-1)}$.
- If $r_\alpha>u$ then $\alpha_j^{(t)}=\tilde{\alpha}_j$, else $\alpha_j^{(t)}=\alpha_j^{(t-1)}$.
- If $r_\beta>u$ then $\beta_j^{(t)}=\tilde{\beta}_j$, else $\beta_j^{(t)}=\beta_j^{(t-1)}$.

3. Experimental results

3.1. Design of experiments

In this section, we apply our Bayesian GGM estimation algorithm for synthetic data, real datasets, and real applications involving texture classification and retrieval, and image segmentation. We validate our algorithm by comparing it to various stochastic versions of the EM like the SEM, SAEM, MCEM and ICE. In fact, choosing a relevant model consists of both choosing its form (GGM in our case) and the number of components M. We use two approaches in order to rate the ability of the tested models to fit the data or to determine the number of clusters M. The first criterion is one of the key quantities used for Bayesian hypothesis testing and model selection, the integrated or marginal likelihood defined by [49]

$$p(\mathcal{X}|M)=\int_\Theta\pi(\Theta,\mathcal{X}|M)\,d\Theta=\int_\Theta p(\mathcal{X}|\Theta,M)\pi(\Theta|M)\,d\Theta \tag{17}$$

where Θ is the vector of parameters of a finite mixture model, $\pi(\Theta|M)$ is its prior density, and $p(\mathcal{X}|\Theta,M)$ is the likelihood function taking into account that the number of clusters is M. Using the Laplace approximation as in [49] we get

$$\log(p(\mathcal{X}|M))=\log(p(\mathcal{X}|\hat{\Theta},M))+\log(\pi(\hat{\Theta}|M))+\frac{N_p}{2}\log(2\pi)+\frac{1}{2}\log(|H(\hat{\Theta})|) \tag{18}$$

where $|H(\hat{\Theta})|$ is the determinant of the Hessian matrix, and N_p is the number of parameters to be estimated which is equal to $(4M)$ for the GGM. We can use the Laplace-Metropolis estimator [49] which consists of finding the Metropolis estimates of $\hat{\Theta}$ and $H(\hat{\Theta})$. With samples of the posterior parameters simulated from the M-H in hand, we

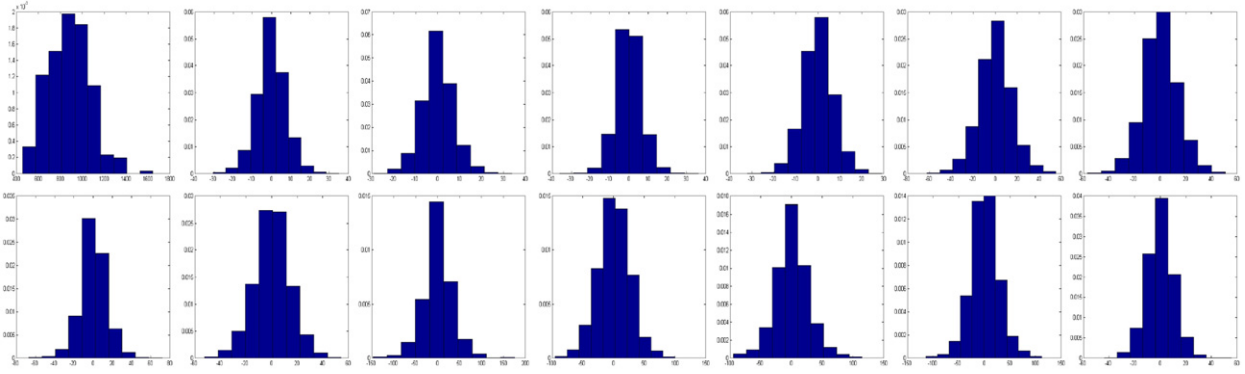

Fig. 11. Histograms of the 14 sub-images of the steerable pyramid.

estimate $\hat{\Theta}$ as the Θ in the sample at which the likelihood $p(\mathcal{X}|\hat{\Theta},M)$ achieves its maximum. For the other quantity needed $H(\hat{\Theta})$ it is asymptotically equal to the posterior covariance matrix, and could be estimated by the sample covariance matrix of the posterior simulation outputs. The second approach is the Bayesian information criterion (BIC) of Schwartz [50] which is actually an approximation for the integrated likelihood criterion [49]: $BIC = \log(\mathcal{X}|M,\hat{\Theta}_M) - (N_p/2)\log(N)$. In the following applications, we have used 5000 iteration for our Metropolis-within-Gibbs sampler (we discarded the first 800 iterations as "burn-in" and kept the rest), and our specific choices for the hyperparameters are $(\mu_0,\sigma_0^2,\alpha_\alpha,\beta_\alpha,\alpha_\beta,\beta_\beta,\eta_1,\ldots,\eta_M) = (0,1,0.2,2,0.2,2,1,\ldots,1)$. As for the scale of the random walk we use it as $\zeta^2 = 0.01$ to increase the sensitivity of the random walk sampler which is actually a common choice widely used. The choice of a symmetric Dirichlet with parameters set to 1 is also a common choice when dealing with Bayesian mixture modeling [47]. It is noteworthy, however, that a sensitivity analysis for the choice of the other hyperparameters revealed robustness of the posterior results. For instance, we took different values for α_α and α_β ranging from 0.05 to 1.9 (by considering a step of 0.05), and for β_α and β_β ranging from 0.1 to 2.9 (by considering a step of 0.1) and the results were the same (i.e. the differences between the final results were not statistically significant).

3.2. Synthetic data

This section has two main goals, first testing the effectiveness of the algorithm to estimate the mixture parameters and to select the number of clusters. Then to illustrate the higher performance of our algorithm compared to four stochastic versions of the EM (SEM, SAEM, MCEM, ICE). To reach our first goal we generated three datasets and applied our method to estimate the parameters and select the number of components of the associated mixture models. Table 1 contains the real and estimated parameters of the generated datasets. Fig. 2 shows the real and the estimated histograms of the three generated datasets. The integrated likelihood and the BIC calculated for different number of clusters (M=1, 2, 3, 4 and 5), and the Kolmogoroff–Smirnov distances related to the above datasets are given in Fig. 3.

Fig. 4 shows the time series plot of our Bayesian algorithm iterations by taking the first dataset as an example. We can see clearly that our algorithm is able to get the exact number of clusters and a very good approximation of the mixture parameters of the generated datasets.

For the second goal, we generated a dataset that has five components and applied our Bayesian algorithm and the four stochastic EM versions. In order to evaluate the accuracy of the different methods, we calculated the Kolmogoroff–Smirnov distances between the estimated distributions and the real histogram in each of the five cases. For our algorithm, it was able to recognize that the dataset is generated from five classes, and to estimate its parameters effectively. As for the other stochastic EM algorithms their selection of the number of components was wrong which forces them to a false estimation of the parameters. The integrated likelihood and the BIC calculated for different number of clusters are given in Fig. 5. Fig. 6 shows the real and the estimated histograms of the dataset using the five different methods, and the related Kolmogoroff–Smirnov distances. For our algorithm, it was able to recognize that the dataset is generated from five classes and to estimate its parameters effectively. As for the other stochastic EM algorithms their selection of the number of components was wrong which forces them to a false estimation of the parameters. Also the Kolmogoroff–Smirnov distances show that the Bayesian approach outperformed the various stochastic methods.

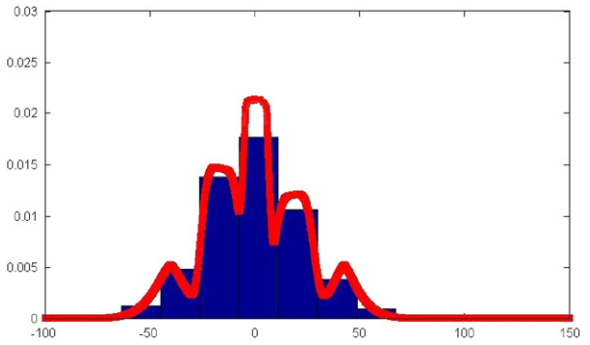

Fig. 12. An example of the sub-image real and estimated histogram using five components GGM.

3.3. Real datasets

We devote this section for real datasets. Our method is used to model three standard widely used datasets. The first one describes an enzymatic activity in the blood among a group of 245 unrelated individuals, and the second one is an acidity index measured in a sample of

Fig. 13. Sample images from each group. (a) Bark, (b) Fabric, (c) Food, (d) Metal, (e) Sand, (f) Water.

155 lakes in the Northeastern United States. The third and the last one consists of thickness of 485 postage stamps produced in Mexico. For these three datasets, a mixture of two distributions is generally identified [51]. Fig. 7 shows the real and the estimated histograms for the three datasets, respectively, when applying the Bayesian and different stochastic algorithms. In all cases, it is clear that all the algorithms can fit the data. The values of the BIC, and marginal likelihood criteria for the five methods for different values of M are given in Fig. 8. According to these figures the optimal number of components to fit the three datasets in all cases varies from $M=2$ to 4. Fig. 9 shows the Kolmogoroff–Smirnov distances of the five different algorithms for the three datasets. It is quite clear that for the three dataset the Bayesian approach yields a smaller distance than the other algorithms.

Table 2
The Average ($\pm$standard deviation) classification accuracy, over 10 trials, of the five different methods.

Method	Using three levels pyramid	Using five levels pyramid
Bayesian	$94.12\% \pm 1.62\%$	$95.62\% \pm 1.34\%$
SEM	$93.38\% \pm 1.64\%$	$94.46\% \pm 1.59\%$
SAEM	$92.82\% \pm 1.72\%$	$93.83\% \pm 1.91\%$
MCEM	$92.52\% \pm 1.84\%$	$93.75\% \pm 2.02\%$
ICE	$93.12\% \pm 1.67\%$	$93.88\% \pm 1.81\%$

Table 3
Average retrieval rates (%) for the five different methods.

Method	Using three levels pyramid (%)	Using five levels pyramid (%)
Bayesian	81.25	83.81
SEM	77.64	81.37
SAEM	76.19	80.99
MCEM	76.04	79.49
ICE	76.82	80.55

3.4. Classification and retrieval of texture images

3.4.1. Approach

Texture is one of the main characteristics used to describe natural images, which explain its important role in image processing, computer vision and pattern recognition applications. Texture analysis is a fundamental step in a variety of image processing applications such as industrial inspection, medical imaging, remote sensing, and content-based image classification and retrieval [52,53,16]. Texture analysis approaches can be divided into four categories: statistical,

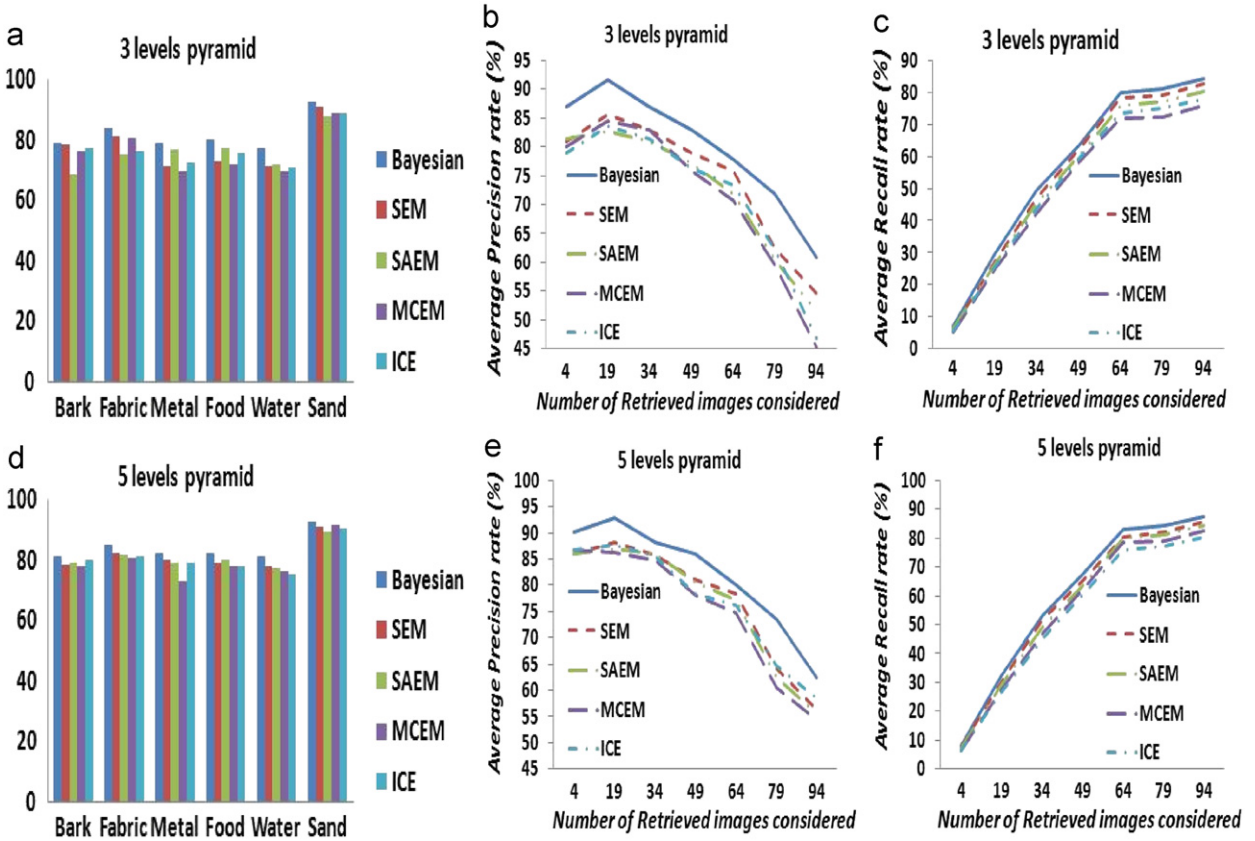

Fig. 14. Average retrieval rates. (a) Average precision rate using three levels pyramid, (b) overall precision using three levels pyramid, (c) overall recall rate using three levels pyramid, (d) average precision rate using five levels pyramid, (e) overall precision using five levels pyramid, (f) overall recall rate using five levels pyramid.

geometrical, model-based, and signal processing methods [54]. Many classification methods based on images frequency analysis have been proposed in the past. The basic assumption of these methods is that texture can be identified by the energy distribution in the frequency domain via the decomposition of the frequency spectrum into a sufficient number of sub-bands. Then, the statistics of the sub-band coefficients can be derived and modeled to distinguish different image textures. Indeed, texture information can be modeled using second or higher order statistics [55] and it is well-known that natural image textures generally give rise to non-Gaussian highly peaked sub-band densities [56]. In this section we propose an approach for texture images classification and retrieval based on our GGM Bayesian learning

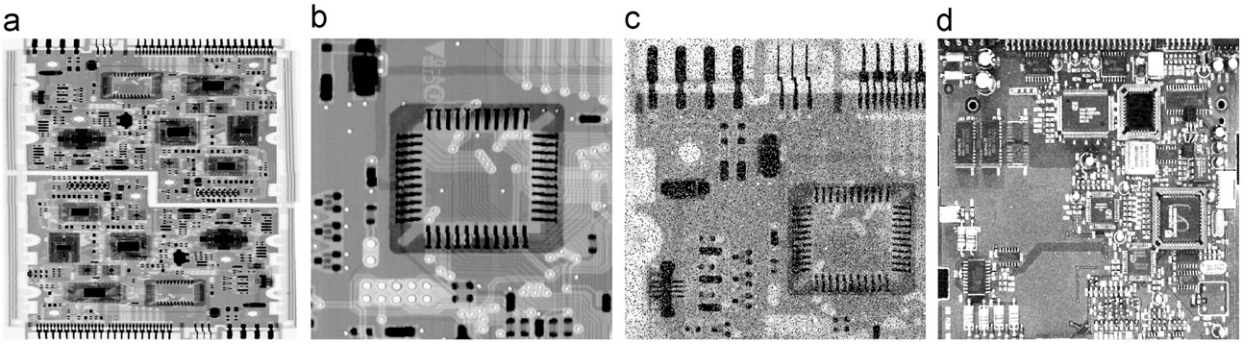

Fig. 15. Tested images. (a) Complex PCB, (b) PCB with text, (c) PCB with noise, (d) PCB with missing components.

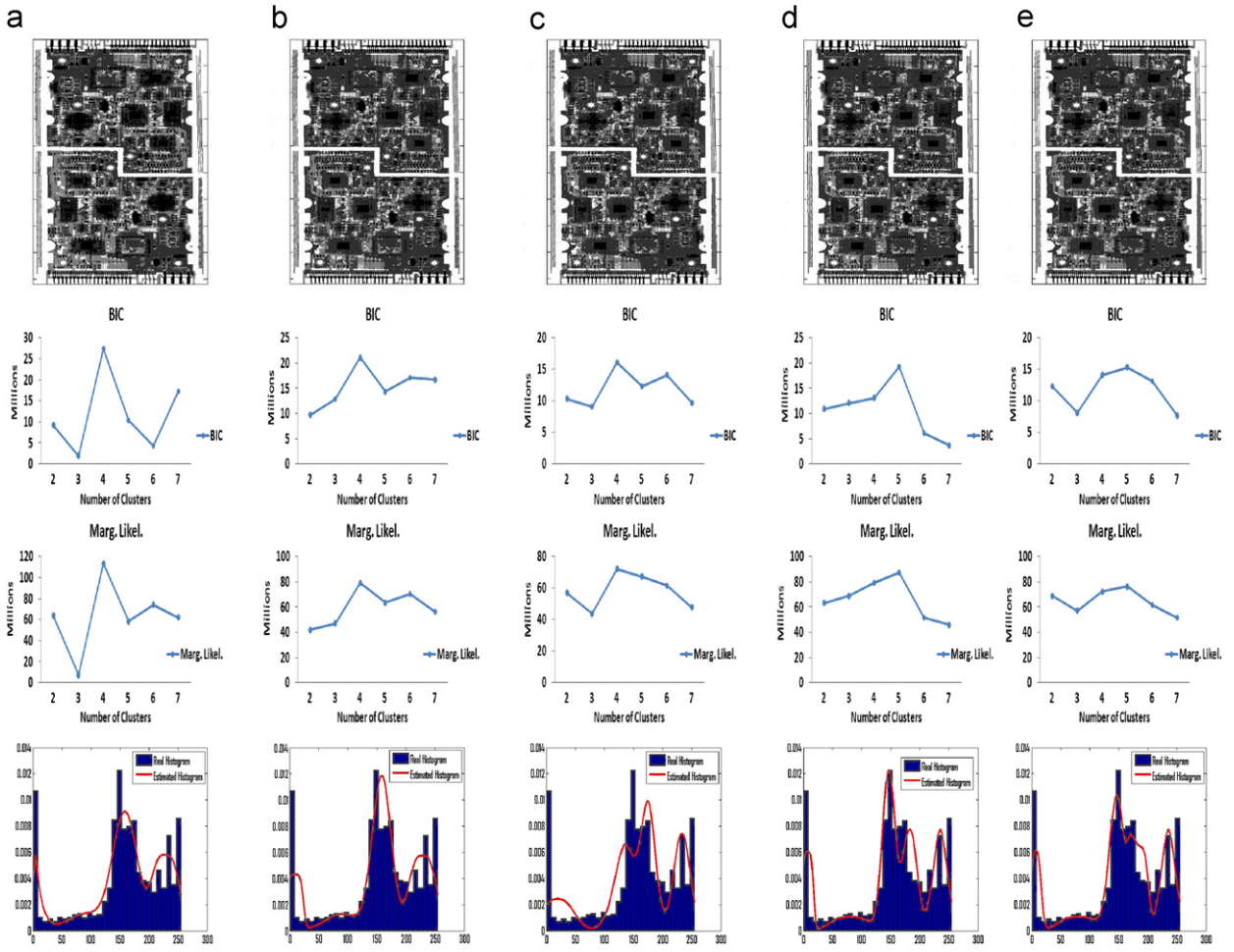

Fig. 16. Segmentations results for Fig. 15(a). (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

algorithm. The used classification methodology, previously adopted in [57] using GM, takes into consideration that signatures of different textures will differ when transformed to frequency domain.

In our classification framework, an image texture is first transformed to gray scale and decomposed into sub-bands using steerable filters [58,59]. Fig. 10 shows a texture image and its multiscale version in a pyramid hierarchy. The histograms of the resulted filtered images are shown in Fig. 11 which shows clearly that the Gaussian assumption would be inappropriate. Then, each sub-band's marginal density is approximated by a GGM model using our Bayesian estimation algorithm (see Fig. 12). Finally, the Earth Mover's Distance (EMD) [60] is used to measure the distribution similarity between a set of components representing an input image texture (i.e. test image) and sets of components representing texture classes (i.e training images). In our case, EMD can be viewed as the minimum cost of changing one mixture into another, when the cost of moving probability mass from components in the first mixture to components in the second mixture is calculated using Kullback–Leibler (KL) divergence given by

$$D(f_i\|g_j)=\int_x f_i(x)\log\left(\frac{f_i(x)}{g_j(x)}\right)dx \tag{19}$$

where f_i is the component i of the input sub-image mixture, g_j is the component j of the class sub-image mixture. The derivation for the KL divergence of the generalized Gaussian distribution is known to be [16]

$$D(f_i\|g_j)=\log\left(\frac{\beta_i\alpha_i\Gamma\left(\frac{1}{\beta_j}\right)}{\beta_j\alpha_j\Gamma\left(\frac{1}{\beta_i}\right)}\right)+\left(\frac{\alpha_j}{\alpha_i}\right)^{\beta_j}\frac{\Gamma\left(\beta_j+\frac{1}{\beta_i}\right)}{\Gamma\left(\frac{1}{\beta_i}\right)}-\frac{1}{\beta_i} \tag{20}$$

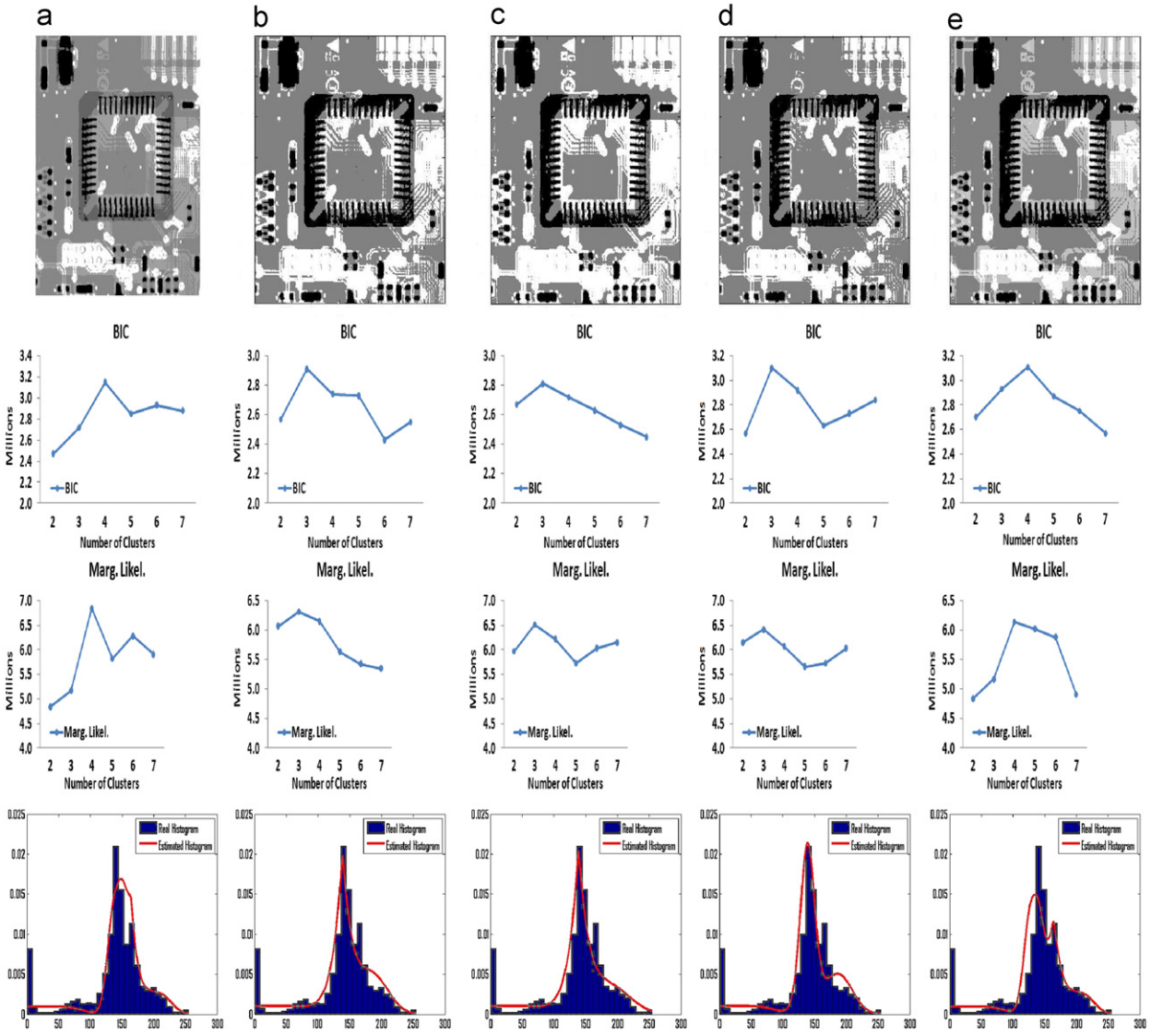

Fig. 17. Segmentations results for Fig. 15(b). (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

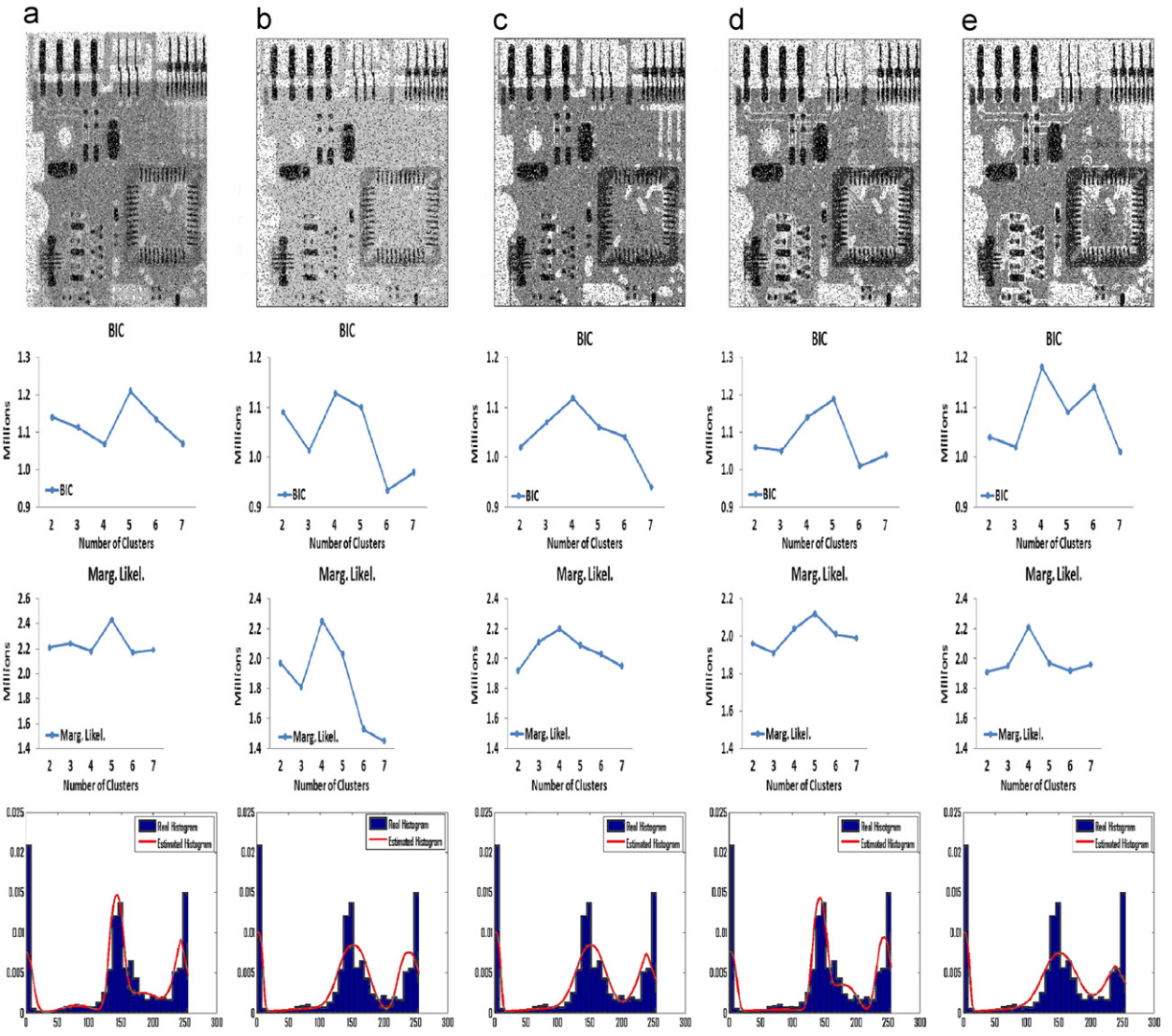

Fig. 18. Segmentations results for Fig. 15(c). (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

where (α_i,β_i) are the parameters of f_i, and (α_j,β_j) are the parameters of g_j. With KL in hand we have to start the minimization problem in which we need to get the $m \times n$ matrix F, where f_{ij} is the amount of weight w_{xi} matched to w_{yj} (w_{xi} and w_{yj} are the weights of the distribution), that will minimize the following equation:

$$EMD_{sub} = \sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} D(f_i||g_j) \tag{21}$$

and subjected to the following constraints: (1) $f_{ij} \geq 0$, where $1 \leq i \leq m$ and $1 \leq j \leq n$, (2) $\sum_{i=1}^{m} f_{ij} = w_{yj}$, where $1 \leq j \leq n$, (3) $\sum_{j=1}^{n} f_{ij} = w_{xi}$, where $1 \leq i \leq m$, (4) $\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} = \min(w_x, w_y)$, where $w_x = \sum_{i=1}^{m} w_{xi}$ and $w_y = \sum_{j=1}^{n} w_{yj}$. Note that when the image texture is decomposed of L sub-bands, then the total EMD is the sum of that of each sub-band, $EMD = \sum_{l=1}^{L} EMD_{sub_l}$. By computing the EMD between the input texture image and each texture class, each image is affected to the class for which the EMD is the smallest.

3.4.2. Results

The images that we have used in our experiments are from the MIT VisTex database.[1] Six homogeneous texture groups (Bark, Fabric, Food, Metal, Water, and Sand) were considered (Fig. 13). For each of the Bark, Fabric, and Metal texture groups, we used four 512×512 images each divided into sixty-four 64×64 subimages. And for the Food, Water, and Sand texture groups, we used six 512×512 images each divided into sixty-four 64×64 subimages as well. This gives us a total of 256 subimages

[1] MIT Vision and Modeling Group (http://vismod.www.media.mit.edu).

for each class in the first three groups, and 384 subimages for each class in the second three groups. We then applied our classification approach 10 times, each time using 24 subimages of each original texture image for training and the remaining 40 for testing. This brings us to a total of 720 images from all six groups as training samples for our algorithm, and 1200 as testing samples. We compared the accuracy of our algorithm to classify all 1200 images to those of the four other methods (SEM, SAEM, MCEM, ICE). We applied our algorithm twice, first using three levels pyramid and then using five levels pyramid to compare the two cases together (see Table 2). From these results we can observe two main points: our algorithm has the highest accuracy and as expected the five levels pyramid improves the performance over the three levels pyramids, however this improvement is very small compared to the enormous difference in computational time.

In the retrieval application, we have used each and every subimage as a query and verified if we are able to retrieve all the other 64 subimages coming from the same mother image. Our retrieval approach can be divided into two steps. First task, is the same as the classification approach, we classify the image into one of the six groups. For the second step, we compare the input image with the other images in the same group and retrieve the closest images to our query. We applied our retrieval process twice first using three levels pyramid and then using five levels pyramid.

To measure the retrieval rates (precision and recall), each image was used as a query and the number of relevant images among those that were retrieved was noted. Table 3 presents the retrieval rates obtained in terms of precision when 64 images are retrieved each time in response to a query. Note that in this case the precision and recall are the same because for a given

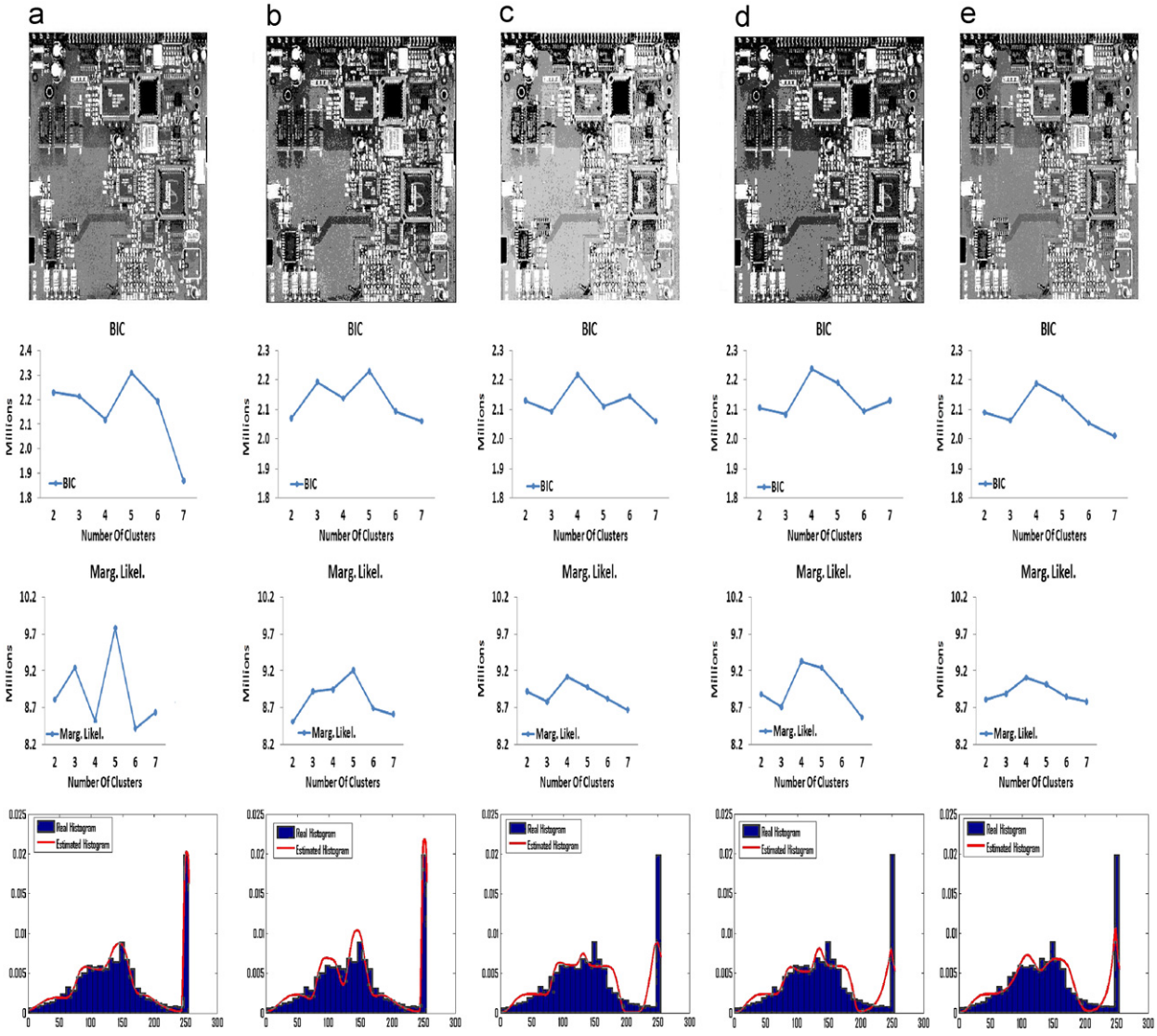

Fig. 19. Segmentations results for Fig. 15(d). (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

image we have at most 64 images which are similar to it. Figs. 14(a) and (d) display the average (averaged over all the queries) precision rate for different texture classes when we consider only the first 64 images retrieved in both cases (i.e. three and five levels pyramid). Figs. 14(b) and (e) show the overall precision of our retrieval methods when varying the total number of images retrieved in both cases. Figs. 14(c) and (f) show the overall recall of our retrieval methods when varying the total number of images retrieved in both cases. According

Fig. 20. Tested SAR images. (a) First image (courtesy of NASA), (b) second image (courtesy of NASA), (c) SAR image (courtesy of European Space Agency).

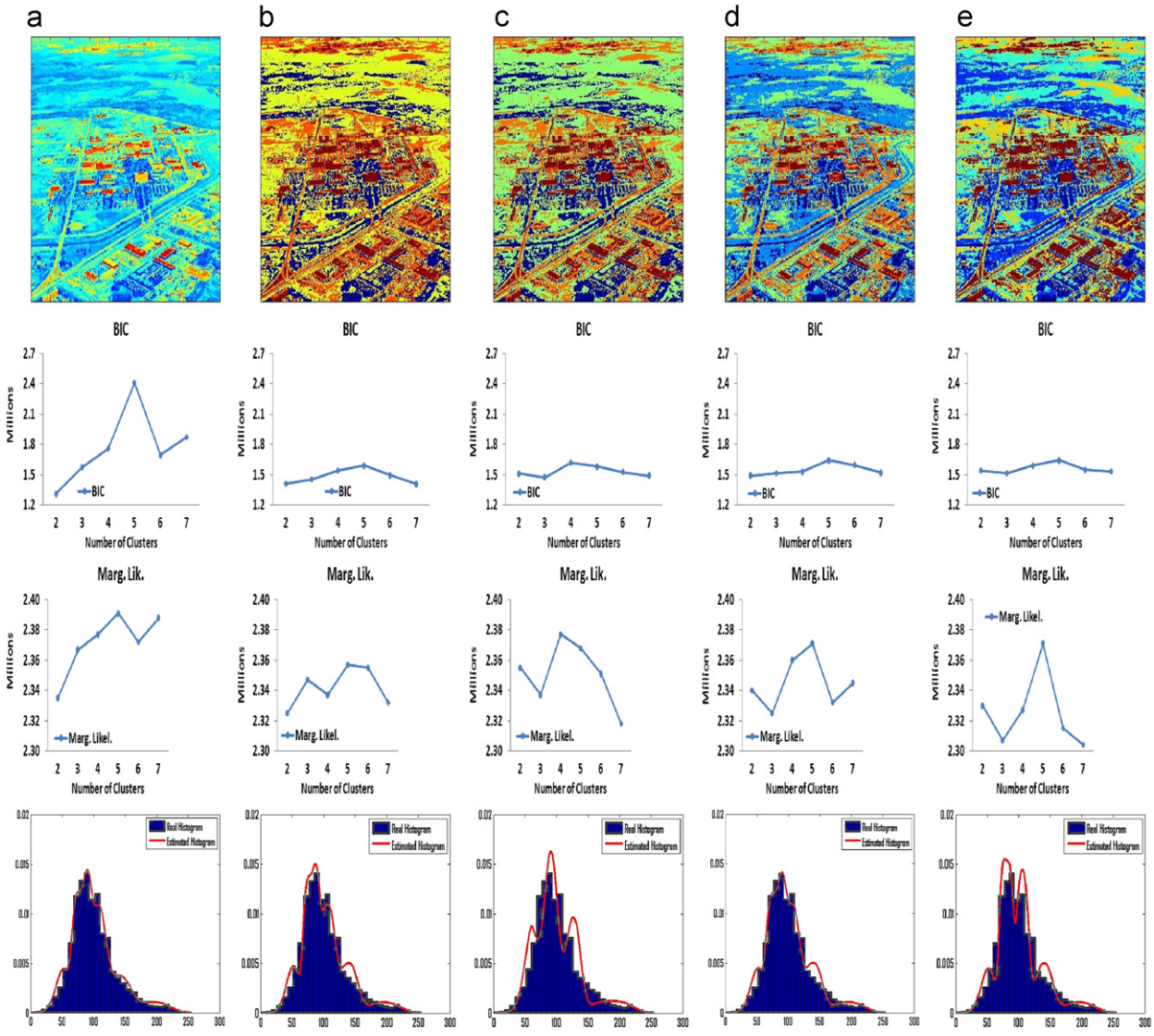

Fig. 21. Segmentations results for Fig. 20(a). (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

to the results in Table 3 we can reach the following conclusions. First the highest average retrieval rate is reached using our method. Second, it is enough to use three levels pyramid due to the large computational difference between it and the five levels pyramid, and the small difference in their effectiveness.

3.5. Image segmentation

Image segmentation is one of the most significant problems, due to the fact that it is fundamental for many tasks of pattern recognition, image processing, computer vision where the segmentation results generally govern the final quality of interpretation. Several approaches have been proposed in the past. Many techniques based on finite mixture Gaussians have also been developed, where the idea was to partition the image into regions (each associated with one mixture component). However, generally the pixel intensities inside the image regions are heavy-tailed which force the Gaussian mixture model to lose its accuracy. Often, image segmentation must be done in an unsupervised fashion in that training data is not available and the class conditioned feature vectors must be estimated directly from the data. In this section, we apply our estimation algorithm for the segmentation problem by formulating it as a classification problem with mixtures of generalized Gaussian distributions. It is noteworthy to mention that the main purpose of this application is to compare our Bayesian estimation with the different stochastic versions of the EM algorithm. Comparisons with several other segmentation approaches proposed in the past is beyond the scope of our work.

We tested the effectiveness of our method on two different types of images: printed circuit board (PCB) and optical images. We decided to use these images because of their non-Gaussian characteristics. We began

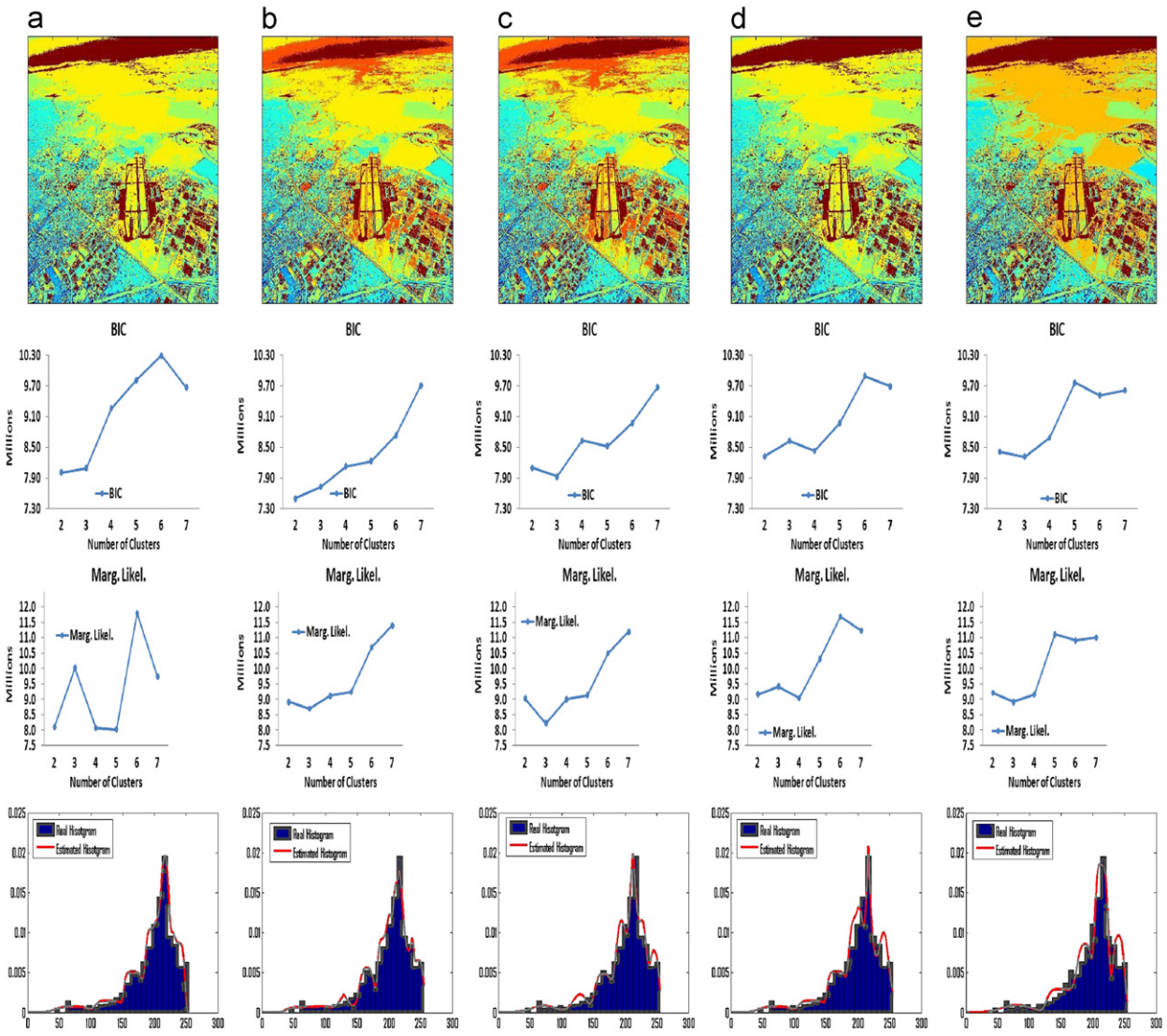

Fig. 22. Segmentations results for Fig. 20(b). (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

our experiment by applying our algorithm on four complex electronic printed circuit board (PCB) images, to check if the algorithm is able to recognize the background from the printed circuit, integrated circuits (IC), wiring, and the pinholes in images. We tested these images with the five different algorithms to compare their effectiveness for separating non-Gaussian data. The first image is an image of a complex PCB (Fig. 15(a)). Applying the five different estimation methods, we obtained three different outputs regarding the number of clusters. The Bayesian, SEM, and SAEM segmented the image into four groups, while the MCEM, ICE divided it into three classes, respectively (Fig. 16). We can notice two points: the Bayesian, SEM, and SAEM approaches were able to identify the right number of clusters, and the Bayesian algorithm performs a slightly better refinement in the electronic circuit wiring then the two other stochastic versions identified by its small Kolmogoroff–Smirnov distance (Fig. 24(a)). The second image is also an image of a PCB. We decided to use this image (Fig. 15(b)) as it has something written on the board. Applying our segmentation algorithm, it was able to identify the four different classes of the image and to segment it accurately. As for all the other algorithms, except the ICE, they were unable to identify the right number of classes which led them to wrong segmentations (Fig. 17).

The third image (Fig. 15(c)) is corrupted with salt and pepper noise. The goal is to check if the Bayesian algorithm will be able to segment a corrupted image correctly and to identify its components. We found that our method was able to identify all the small details of the PCB, while all the other methods fail to do so (Fig. 18). An important area, where image processing can be applied and indispensable, is the automated visual inspection of manufactured goods. Most of the electronic components manufactures use image processing to identify missing

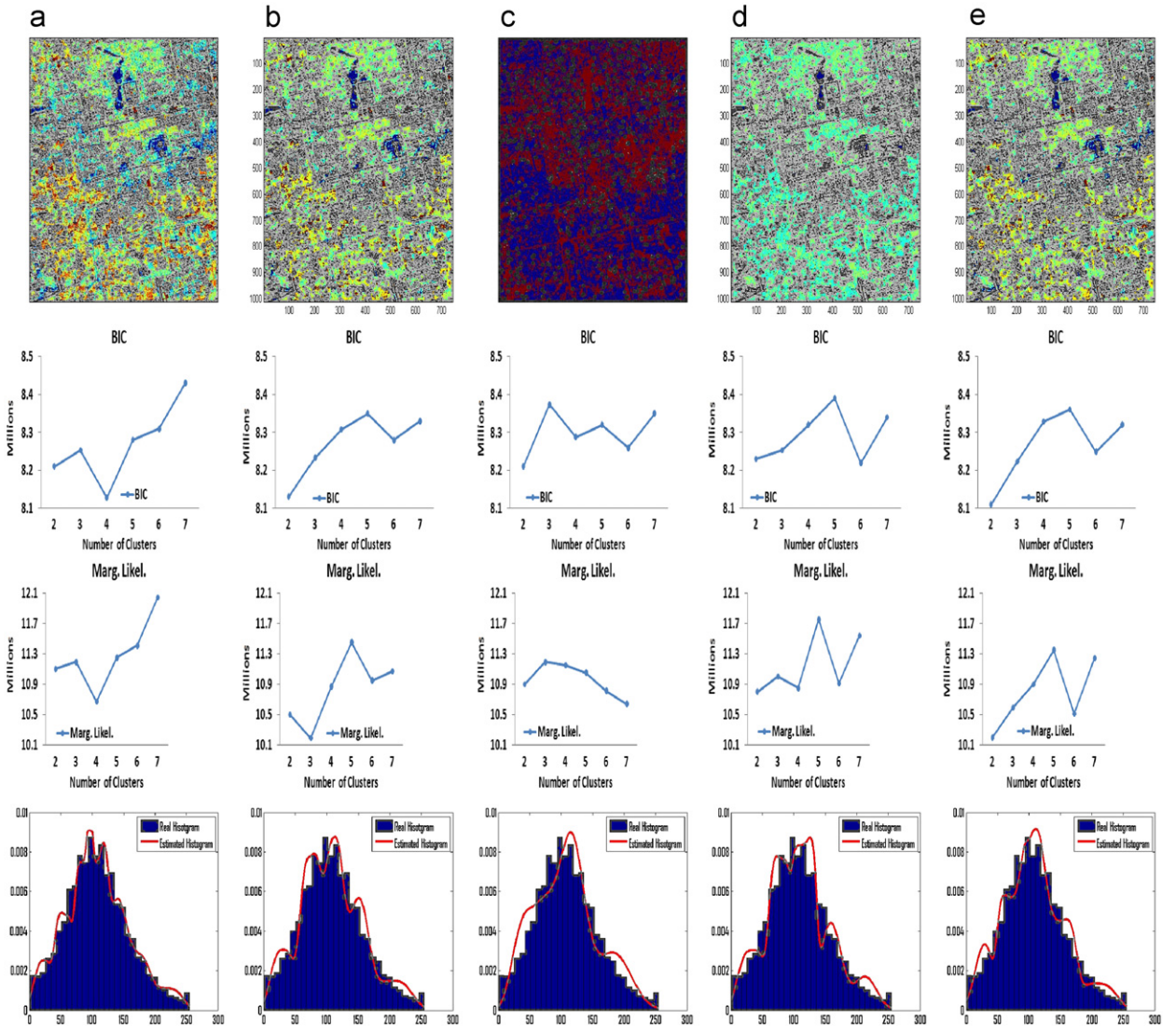

Fig. 23. Segmentations results for Fig. 20(c). (a) Bayesian algorithm, (b) SEM algorithm, (c) SAEM algorithm, (d) MCEM algorithm, (e) ICE algorithm.

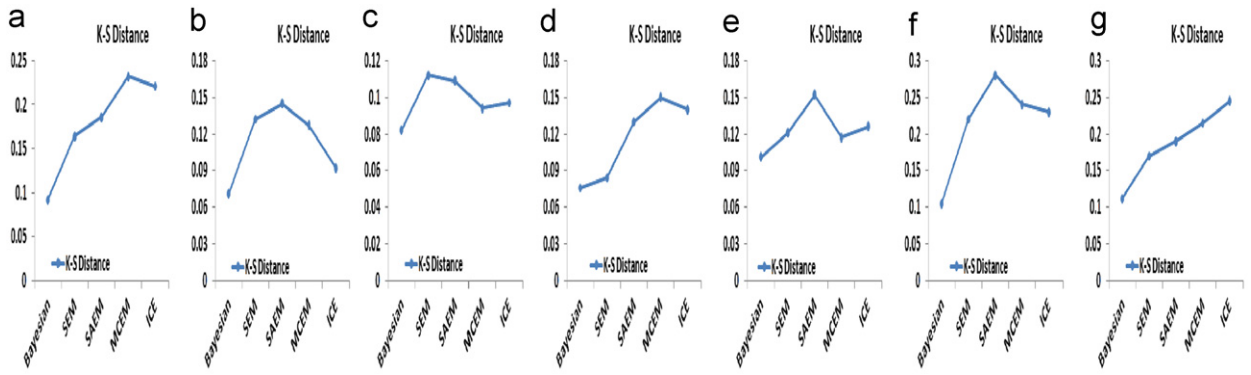

Fig. 24. Kolmogoroff–Smirnov distances. (a) Fig. 15(a), (b) Fig. 15(b), (c) Fig. 15(c), (d) Fig. 15(d), (e) Fig. 20(a), (f) Fig. 20(b), (g) Fig. 20(c).

components in the circuit before shipment. We used the five methods in Fig. 15(d) which contains some missing components. Our segmentation method was able to identify the missing components in the circuit and to separate it to a class that only contains these missing components. In other words, our method identifies that the image contains five classes exactly, where the fifth class contains only the places in the PCB where there are missing components. All other methods, except SEM, were unable to segment and identify the missing components in the image (Fig. 19).

We have also tested our method using optical images. Unlike natural images, airborne images characteristically are highly corrupted by different kinds of noise. This causes serious problems for optical image segmentation process. Hence, segmentation techniques that work successfully on natural images may not perform as well on optical images. We used three different images to investigate the effectiveness of our algorithm. The first and second images are used to illustrate the segmentation effectiveness of our method in segmenting images highly corrupted by atmospheric turbulence (Figs. 20(a) and (b)). For both images we can notice that the Bayesian algorithm is the most effective approach as it was able to approximate the data with the best estimated histogram compared to the four other methods (Figs. 21–23).

The third image (Fig. 20(c)) was taken for the Beijing area, China. We can notice that the Bayesian estimated histogram is the closest one to the real histogram. Also from the given Kolmogoroff–Smirnov distances in Fig. 24 we can see clearly that our method has the smallest distance which means the best approximation compared to the four other methods in all images under consideration.

4. Conclusion

We have presented a Bayesian analysis of finite generalized Gaussian mixtures. Our learning algorithm is based on the Monte Carlo simulation technique of Gibbs sampling mixed with a Metropolis-Hasting step. For the estimation of the number of clusters describing the mixture model, we used the marginal likelihood with Laplace approximation, and the BIC criterion. We have demonstrated clearly by different applications that Bayesian estimation and selection gives reliable estimates. The Bayesian approach provides a natural extension to deal with uncertainty and noise by incorporating prior information. Future works could be devoted, for instance, to the generalization of the proposed approach for the multidimensional case.

Acknowledgment

The completion of this research was made possible thanks to the Natural Sciences and Engineering Research Council of Canada (NSERC), a NATEQ Nouveaux Chercheurs Grant, and a start-up grant from Concordia University. The authors would like to thank the anonymous referees for their helpful comments.

Appendix A. Proof of Eqs. (11)–(13)

The derivation of Eq. (11)is as follows: $\pi(\mu_j|Z,\mathcal{X}) \propto \pi(\mu_j) \prod_{Z_{ij}=1} p(\mathcal{X}|\mu_j,\alpha_j,\beta_j)$, thus

$$\pi(\mu_j|Z,\mathcal{X}) \propto \frac{1}{\sqrt{2\pi}}\sigma_0 \mathbf{e}^{-(\mu_j-\mu_0)^2/2\sigma_0^2} \prod_{Z_{ij}=1} \left(\frac{\alpha_j\beta_j}{2\Gamma(1/\beta_j)} \mathbf{e}^{-(\alpha_j|\mathrm{x}_i-\mu_j|)^{\beta_j}} \right) \tag{22}$$

In this case we have σ_0 as a constant hyperparameter, also α_j,β_j are considered as constant parameters, which gives us

$$\pi(\mu_j|Z,\mathcal{X}) \propto \mathbf{e}^{-(\mu_j-\mu_0)^2/2\sigma_0^2} \mathbf{e}^{\sum_{Z_{ij}=1}(-\alpha_j|\mathrm{x}_i-\mu_j|)^{\beta_j}} \tag{23}$$

The derivation for Eq. (12)is as follows: $\pi(\alpha_j|Z,\mathcal{X}) \propto \pi(\alpha_j) \prod_{Z_{ij}=1} p(\mathrm{x}_i|\mu_j,\alpha_j,\beta_j)$, thus

$$\pi(\alpha_j|Z,\mathcal{X}) \propto \frac{\alpha_j^{\alpha_\alpha-1}\beta_\alpha^{\alpha_\alpha}\mathbf{e}^{-\beta_\alpha\alpha_j}}{\Gamma(\alpha_\alpha)} \prod_{Z_{ij}=1} \left(\frac{\alpha_j\beta_j}{2\Gamma(1/\beta_j)} \mathbf{e}^{-(\alpha_j|\mathrm{x}_i-\mu_j|)^{\beta_j}} \right) \tag{24}$$

In this case we have $(\alpha_\alpha,\ \beta_\alpha)$ are constant hyperparameters, also β_j is considered as a constant parameter, which gives us

$$\pi(\alpha_j|Z,\mathcal{X}) \propto \alpha_j^{\alpha_\alpha-1}\mathbf{e}^{-\beta_\alpha\alpha_j}(\alpha_j)^{n_j}\mathbf{e}^{\sum_{Z_{ij}=1}-(\alpha_j|\mathrm{x}_i-\mu_j|)^{\beta_j}} \tag{25}$$

The derivation for Eq. (13)is the same as for Eq. (12). (α_β, β_β) are constant hyperparameters, and α_j is considered as a constant parameter:

$$\pi(\beta_j|Z,\mathcal{X}) \propto \beta_j^{\alpha_\beta-1}\mathbf{e}^{-\beta_\beta\beta_j}\left(\frac{\beta_j}{\Gamma(1/\beta_j)}\right)^{n_j}\mathbf{e}^{\sum_{Z_{ij}=1}-(\alpha_j|x_i-\mu_j|)^{\beta_j}} \quad (26)$$

References

[1] J.R. Ohm, Multimedia Communication Technology, Representation, Transmission and Identification of Multimedia Signals, Springer, 2004.
[2] G.J. McLachlan, D. Peel, Finite Mixture Models, Wiley, New York, 2000.
[3] F. Chen, Z. Gao, J. Villasenor, Lattice vector quantization of generalized Gaussian sources, IEEE Transactions on Information Theory 43 (1) (1997) 92–103.
[4] R.L. Joshi, V.J. Crump, T.R. Fischer, Image subband coding using arithmetic coded trellis coded quantization, IEEE Transactions on Circuits and Systems for Video Technology 5 (6) (1995) 515–523.
[5] R. Laroia, N. Farvardin, A structured fixed-rate vector quantizer derived from a variable-length scalar quantizer: part I—memoryless sources, IEEE Transactions on Information Theory 39 (3) (1993) 851–867.
[6] Y. Delignon, A. Marzouki, W. Pieczynski, Estimation of generalized mixtures and its application in image segmentation, IEEE Transactions on Image Processing 6 (10) (1997) 1364–1375.
[7] J.H. Miller, J.B. Thomas, Detectors for discrete-time signals in non-Gaussian noise, IEEE Transactions on Information Theory 18 (2) (1972) 241–250.
[8] N. Farvardin, J.W. Modestino, Optimum quantizer performance for a class of non-Gaussian memoryless sources, IEEE Transactions on Information Theory 30 (3) (1984) 485–497.
[9] Z. Gao, B. Belzer, J. Villasenor, A comparison of the Z, E_8, and leech lattices for quantization of low-shape-parameter generalized Gaussian sources, IEEE Signal Processing Letters 2 (10) (1995) 197–199.
[10] S. Meignen, H. Meignen, On the modeling of small sample distributions with generalized Gaussian density in a maximum likelihood framework, IEEE Transactions on Image Processing 15 (6) (2006) 1647–1652.
[11] W. Mauersberger, Experimental results on the performance of mismatched quantizers, IEEE Transactions on Information Theory 25 (4) (1979) 381–386.
[12] S.G. Mallat, A theory for multiresolution signal decomposition: the wavelet representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (7) (1989) 674–693.
[13] T. Naveen, J.W. Woods, Motion compensated multiresolution transmission of high definition video, IEEE Transactions on Circuits and Systems for Video Technology 4 (1) (1994) 29–41.
[14] K. Sharifi, A. Leon-Garcia, Estimation of shape parameter for generalized Gaussian distributions in subband decomposition of video, IEEE Transactions on Circuits and Systems for Video Technology 5 (1) (1995) 52–56.
[15] G. Calvagno, C. Ghirardi, G.A. Mian, R. Rinaldo, Modeling of subband image data for buffer control, IEEE Transactions on Circuits and Systems for Video Technology 7 (2) (1997) 402–408.
[16] M.N. Do, M. Vetterli, Wavelet-based texture retrieval using generalized Gaussian density and Kullback–Leibler distance, IEEE Transactions on Image Processing 11 (2) (2002) 146–158.
[17] J-F. Aujol, G. Aubert, L. Blanc-Féraud, Wavelet-based level set evolution for classification of textured images, IEEE Transactions on Image Processing 12 (12) (2003) 1634–1641.
[18] S-K. Choi, C.-S. Tong, Supervised texture classification using characteristic generalized Gaussian density, Journal of Mathematical Imaging and Vision 29 (1) (2007) 35–47.
[19] P. Moulin, J. Liu, Analysis of multiresolution image denoising schemes using generalized Gaussian and complexity priors, IEEE Transactions on Information Theory 45 (3) (1999) 909–919.
[20] T.R. Fischer, A pyramid vector quantizer, IEEE Transactions on Information Theory 32 (4) (1986) 568–583.
[21] K.A. Birney, T.R. Fischer, On the modeling of DCT and subband image data for compression, IEEE Transactions on Image Processing 4 (2) (1995) 186–193.
[22] C. Bouman, K. Sauer, A generalized Gaussian image model for edge-preserving MAP estimation, IEEE Transactions on Image Processing 2 (3) (1993) 296–310.
[23] Y. Bazi, L. Bruzzone, F. Melgani, Image thresholding based on the EM algorithm and the generalized Gaussian distribution, Pattern Recognition 40 (2) (2007) 619–634.
[24] S.-K.S. Fan, Y. Lin, C.-C. Wu, Image thresholding using a novel estimation method in generalized Gaussian distribution mixture modeling, Neurocomputing 72 (1–3) (2008) 500–512.
[25] S. Gazor, W. Zhang, Speech probability distributions, IEEE Signal Processing Letters 10 (7) (2003) 204–207.
[26] K. Kokkinakis, A.K. Nandi, Exponent parameter estimation for generalized Gaussian probability density functions with application to speech modeling, Signal Processing 85 (9) (2005) 1852–1858.
[27] M.S. Allili, N. Bouguila, D. Ziou, A robust video foreground segmentation by using generalized Gaussian mixture modeling, in: Proceedings of the Canadian Conference on Robot and Vision (CRV), 2007, pp. 503–509.
[28] M.S. Allili, N. Bouguila, D. Ziou, Finite general Gaussian mixture modeling and application to image and video foreground segmentation, Journal of Electronic Imaging 17 (1) (2008) 1–13.
[29] S.-K.S. Fan, Y. Lin, A fast estimation method for the generalized Gaussian mixture distribution on complex images, Computer Vision and Image Understanding 113 (7) (2009) 839–853.
[30] G. Moser, J. Zerubia, S.B. Serpico, SAR amplitude probability density function estimation based on a generalized Gaussian model, IEEE Transactions on Image Processing 15 (6) (2006) 1429–1442.
[31] D. Cantzos, A. Mouchtari, C. Kyriakakis, Multichannel audio resynthesis based on a generalized Gaussian mixture model and cepstral smoothing, in: Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustic, 2005, pp. 215–218.
[32] M.K. Varanasi, B. Aazhang, Parametric generalized Gaussian density estimation, The Journal of the Acoustical Society of America 86 (4) (1989) 1404–1415.
[33] R. Krupiński, J. Purczyński, Approximated fast estimator for the shape parameter of generalized Gaussian distribution, Signal Processing 86 (2) (2006) 205–211.
[34] B. Aiazzi, L. Alpaone, S. Baronti, Estimation based on entropy matching for generalized Gaussian PDF modeling, IEEE Signal Processing Letters 6 (6) (1999) 138–140.
[35] M. Pi, Improve maximum likelihood estimation for subband GGD parameters, Pattern Recognition Letters 27 (14) (2006) 1710–1713.
[36] F. Müller, Distribution shape of two-dimensional DCT coefficients of natural images, Electronic Letters 29 (22) (1993) 1935–1936.
[37] G.J. McLachlan, T. Krishnan, The EM Algorithm and Extensions, Wiley-Interscience, New York, 1997.
[38] C.P. Robert, The Bayesian Choice from Decision-Theoretic Foundations to Computational Implementation, Springer, 2007.
[39] G. Celeux, J. Diebolt, The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem, Computational Statistics Quarterly 2 (1) (1985) 73–82.
[40] G. Celeux, J. Diebolt, A stochastic approximation type EM algorithm for the mixture problem, Stochastics and Stochastics Reports 41 (1992) 119–134.
[41] W. Pieczynski, Statistical image segmentation, Machine Graphics and Vision 1 (1–2) (1992) 261–268.
[42] G.C.G. Wei, M.A. Tanner, A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms, Journal of the American Statistical Association 85 (1990) 699–704.
[43] J.K. Ghosh, M. Delampady, T. Samanta, An Introduction to Bayesian Analysis Theory and Methods, Springer, 2006.
[44] S. Geman, D. Geman, Stochastic relaxation, gibbs distributions and the Bayesian restoration of images, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (6) (1984) 721–741.
[45] M. Lavielle, E. Moulines, A simulated annealing version of the EM algorithm for non-Gaussian deconvolution, Statistics and Computing 7 (4) (1997) 229–236.
[46] J.P. Delmas, An equivalence of the EM and ICE algorithm for exponential family, IEEE Transactions on Signal Processing 45 (10) (1997) 2613–2615.
[47] C.P. Robert, G. Casella, Monte Carlo Statistical Methods, Springer, 2004.
[48] J.E. Gentle, W. Härdle, Handbook of Computational Statistics. Concepts and Fundamentals, vol. 1, Springer, 2004.
[49] S.M. Lewis, A.E. Raftery, Estimating bayes factors via posterior simulation with the Laplace-metropolis estimator, Journal of the American Statistical Association 90 (1997) 648–655.
[50] G. Schwarz, Estimating the dimension of a model, Annals of Statistics 16 (1978) 461–464.
[51] S.L. Crawford, An application of the Laplace method to finite mixture distributions, Journal of the American Statistical Association 89 (1994) 259–267.

[52] R.M. Haralick, K. Shanmugam, I. Dinstein, Textural features for image classification, IEEE Transactions on Systems, Man, and Cybernetics 3 (1975) 610–622.
[53] J.R. Smith, S. Chang, Transform features for texture classification and discrimination in large image databases, in: Proceedings of the IEEE International Conference on Image Processing, 1994, pp. 407–411.
[54] M. Tuceryan, A.K. Jain, Texture analysis, in: The Handbook of Pattern Recognition and Computer Vision, 1998, pp. 207–248.
[55] D. Dunn, W.E. Higgins, J. Wakeley, Texture segmentation using 2-D gabor elementary functions, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (2) (1994) 130–149.
[56] E.P. Simoncelli, E.H. Adelson, Noise removal via Bayesian wavelet coring, in: Proceedings of the IEEE International Conference on Image Processing, 1996, pp. 379–382.
[57] Y. Wu, K.L. Chan, Y. Huang, Image texture classification based on finite Gaussian mixture models, in: Proceedings of the Third International workshop on texture analysis and synthesis, 9th International Conference on Computer Vision, 2003, pp. 107–112.
[58] W.T. Freeman, E.H. Adelson, The design and use of steerable filters, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (9) (1991) 891–906.
[59] E.P. Simoncelli, W.T. Freeman, The steerable pyramid: a flexible architecture for multi-scale derivative computation, in: Proceedings of the IEEE International Conference on Image Processing, 1995, pp. 444–447.
[60] Y. Rubner, C. Tomasi, L.J. Guibas, A metric for distributions with applications to image databases, in: Proceedings of the IEEE International Conference on Computer Vision, 1998, pp. 59–66.