ELSEVIER

Pattern Recognition Letters 18 (1997) 1167–1178

Pattern Recognition Letters

Bayesian Ying–Yang machine, clustering and number of clusters [1]

Lei Xu [2]

Department of Computer Science and Engineering, Chinese University of Hong Kong, Shatin, NT, Hong Kong, China

Abstract

It is shown that a particular case of the Bayesian Ying–Yang learning system and theory reduces to the maximum likelihood learning of a finite mixture, from which we have obtained not only the EM algorithm for its parameter estimation and its various approximate but fast algorithms for clustering in general cases (including Mahalanobis distance clustering or elliptic clustering), but also criteria for the selection of the number of densities in a mixture, and the number k in the conventional Mean Square Error clustering. Moreover, a Re-weighted EM algorithm is also proposed and shown to be more robust in learning. Finally, experimental results are provided. © 1997 Elsevier Science B.V.

Keywords: Bayesian Ying–Yang machine; Number of clusters; Finite mixture; Cluster analysis

1. Introduction

Given a data set $D_x = \{x_i\}_{i=1}^N$, the task of partitioning D_x into k clusters is a classical problem in the literature of statistics and pattern recognition, usually called cluster analysis (Jain and Dubes, 1988). A well-known formulation of this task is to use k vectors $\{m_y^*\}_{y=1}^k$, called centers or code-vectors to represent the k clusters such that a sample x_i is classified into the yth cluster when $I(y|x_i) = 1$, according to

$$I(y|x_i) = \begin{cases} 1 & \text{if } y = \arg\min_y \|x_i - m_y\|^2, \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

with $\{m_y^*\}_{y=1}^k$ obtained by

$$\mathrm{Min}_{\{m_y\}_{y=1}^k} E_{\mathrm{MSE}},$$

$$E_{\mathrm{MSE}} = \frac{1}{N} \sum_{y=1}^{k} \sum_{i=1}^{N} I(y|x_i) \|x_i - m_y\|^2. \quad (2)$$

This formulation is called the *Mean Square Error* (MSE) clustering analysis or vector quantization. It is typically implemented by the well known k-means algorithm or the LBG algorithm, which is a two-step

[1] This work was supported by the HK RGC Earmarked Grants CUHK250/94E, CUHK484/95E, and Ho Sin-Hang Education Endowment Fund HSH 95/02.

[2] E-mail: lxu@cse.cuhk.edu.hk.

PII S0167-8655(97)00121-9

iterative procedure, starting from an initial guess on either $\{m_y\}_{y=1}^k$ or $\{I(y|x_i)\}_{i=1}^N$:

k-means Alg.

Step 1: update $\{I(y|x_i)\}_{i=1}^N$ by *Eq.* (1), then get

$$\alpha_y = \frac{1}{N}\sum_{i=1}^{N} I(y|x_i),$$

Step 2: update $m_y = \frac{1}{\alpha_y N}\sum_{i=1}^{N} I(y|x_i)\,x_i.$ (3)

Equivalently, many of the so-called competitive learning algorithms in the literature of neural networks can be regarded as adaptive variants of the above algorithm for MSE clustering (e.g., see the Reference List in (Xu et al., 1993)).

The algorithms in this formulation all have two serious limitations. The first one is that the number k of clusters must be pre-known and fixed. E_{MSE} monotonically decreases with increasing k, and thus cannot detect a correct k. However, a bad estimate of k can cause serious problems as stated in (Xu et al., 1993). To the best of our knowledge, the selection of a correct k remains an important open problem. There is no theoretical guide available for solving the problem, except for some heuristic techniques, e.g., ISODATA (Duda and Hart, 1972) (Jain and Dubes, 1988), and Rival Penalized Competitive Learning (Xu et al., 1993). The other limitation is that the formulation implies that samples come from a mixture of k Gaussian densities with equal proportion and equal variance $\sigma^2 I$, which can be clearly seen in Section 5. This special case deviates from many practical situations. In the literature, the so-called Mahalanobis distance clustering or elliptic clustering attempts to overcome this limitation.

A unified statistical learning approach called *Bayesian Ying–Yang* (BYY) *system and theory* has been developed by the present author in recent years (Xu, 1995, 1996, 1997a,b,c). This theory functions as a general theory for both unsupervised and supervised learning for parameter learning, regularization, structural scale or complexity selection, architecture design and data sampling. For unsupervised learning and its semi-unsupervised extension, as summarized recently in (Xu, 1997a), the general theory can provide new theories for unsupervised pattern recognition and clustering analysis, factorial encoding, data dimension reduction, and independent component analysis, such that not only several existing popular unsupervised learning approaches (e.g., finite mixture with the EM algorithm, *K*-means clustering algorithm, Helmholtz machine, principal component analysis (PCA) plus various extensions, Informax and minimum mutual information approaches for independent component analysis, ..., etc.), are unified as special cases with new insights and several new results, but also a number of new unsupervised learning models are obtained, and a number of hard model selection problems are solved, e.g., for subspace dimension in PCA related approaches, the number of clusters or number of Gaussians in clustering analysis, and the number of sources in ICA related blind separation. For supervised learning, as summarized recently in another paper (Xu, 1997b), the general theory can provide new theories for supervised classification and regression based on three-layer nets, mixtures-of-experts, and radial basis function nets such that not only the existing approaches are unified as special cases with new insights, but also new learning algorithms are obtained and new selection criteria for the number of hidden units and experts are developed.

In this paper, we show that a particular case of the BYY learning system and theory reduces to the maximum likelihood (ML) learning of a finite mixture, especially Gaussian mixture, from which we can get the EM algorithm and its variants for various extensions of the MSE clustering and the *k*-mean algorithm, with criteria for the selection of the number of densities or the number of clusters. Moreover, a Re-weighted EM (REM) algorithm is also proposed and shown to be more robust in learning.

2. BYY learning system and theory

The perception tasks can be summarized as the problem of estimating the joint distribution $p(x,y)$ of the observable pattern x in the observable space X and its representation pattern y in the representation space Y, as shown in Fig. 1. In the Bayesian framework, we have two complementary representations $p(x,y) = p(y|x)p(x)$ and $p(x,y) =$

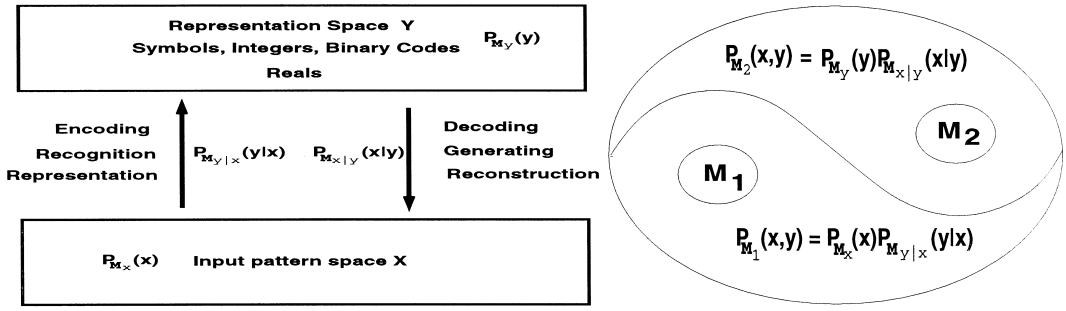

Fig. 1. The joint input-representation spaces X, Y and the YING-YANG machine.

$p(x|y)p(y)$. We use two sets of models $M_1 = \{M_{y|x}, M_x\}$ and $M_2 = \{M_{x|y}, M_y\}$ to implement each of the two representations:

$$p_{M_1}(x,y) = p_{M_{y|x}}(y|x)\, p_{M_x}(x),$$
$$p_{M_2}(x,y) = p_{M_{x|y}}(x|y)\, p_{M_y}(y). \quad (4)$$

We call M_x a Yang/(visible) model, which describes $p(x)$ in the visible domain X, and M_y a Ying [3]/(invisible) model which describes $p(y)$ in the invisible domain Y. Also, we call the passage $M_{y|x}$ for the flow $x \to y$ a *Yang/*(male) passage since it performs the task of transferring a pattern/(a real body) into a code/(a seed). We call a passage $M_{x|y}$ for the flow $y \to x$ a *Ying/*(female) passage since it performs the task of generating a pattern/(a real body) from a code/(a seed). Together, we have a *YANG* machine M_1 to implement $p_{M_1}(x,y)$ and a *YING* machine M_2 to implement $p_{M_2}(x,y)$. A pair of YING-YANG machines is called a YING-YANG pair or a Bayesian YING-YANG system. Such a formalization compliments to a famous Chinese ancient philosophy that *every entity in the universe involves the interaction between YING and YANG*.

The task of specifying a Ying–Yang system is called *learning* in a broad sense, which consists of the following four levels of specifications:

(a) Based on the nature of the perception task, the *Representation Domain Y and Its Complexity k* are designed. For example, we have $y \in [1,2,\ldots,k]$, with x mapped into one of k for the purpose of *clustering*.

(b) Based on the given set of training samples, some previous knowledge, assumption and heuristics, *Architecture Design* is made by specifying the architectures of four components $p_{M_x}(x)$, $p_{M_{y|x}}(y|x)$, $p_{M_{x|y}}(x|y)$ and $p_{M_y}(y)$. First, with a given set $D_x = \{x_i\}_{i=1}^N$ from an original density $p^o(x)$, $p_{M_x}(x)$ is fixed to some parametric or nonparametric empirical density estimation of $p^o(x)$, e.g., $p_{M_x}(x) = p_{h_x}(x)$ given by a kernel estimate (Devroye, 1987):

$$p_h(x) = \frac{1}{N}\sum_{i=1}^{N} K_h(x - x_i), \quad K_h(r) = \frac{1}{h^d}K\left(\frac{r}{h}\right), \quad (5)$$

with a prefixed kernel function $K(\cdot)$ and a prefixed smoothing parameter h. Next, for the other three components, each $p_{M_a}(a)$, $a \in \{x|y, y|x, y\}$ can be designed in two ways. One is called *Free*. It implies a totally unspecified density or probability function in the form $p(a)$ without any constraint. Thus, it is free to change such that it can be indirectly specified through other components. The other is called *Parameterized Architecture*. It means that $p_{M_a}(a)$, $a \in \{x|y, y|x, y\}$ is either a simple parametric density, e.g., a Gaussian $p_{M_{x|y}}(x|y) = G(x, m_{x|y}, \Sigma_{x|y})$ with mean $m_{x|y}$ and variance matrix $\Sigma_{x|y}$, or a compounded parametric density with some of its parameters defined by a complicated function with a given parametric architecture consisting of a number of elementary units that are organized in a given structure.

(c) We also need to select the above complexity k, as well as other scale or complexity parameters for a complicated architecture. This task is called *Structural Scale Selection* or *Model Selection*.

[3] It should be ''Yin'' in the Mainland Chinese spelling system. However, I prefer to use ''Ying'' for the beauty of symmetry.

(d) After the above three levels of specifications, the unspecified part for each component $p_{M_a}(a)$, $a \in \{x|y, y|x, y\}$ is a set θ_a of parameters in certain domains. Putting them together, we get the parameter set $\Theta = \{\theta_{x|y}, \theta_{y|x}, \theta_y\}$, which we call *Parameter Learning*.

Our basic theory is that the specifications of an entire Ying–Yang system in the above four levels best enhances the so-called *Ying–Yang Harmony or Marry*, through minimizing a harmony measure called *separation functional*:

$$F_s(M_1, M_2) =$$

$$F_s\big(p_{M_{y|x}}(y|x)\,p_{M_x}(x), p_{M_{x|y}}(x|y)\,p_{M_y}(y)\big) \geqslant 0,$$

$F_s(M_1, M_2) = 0$, if and only if

$$p_{M_{y|x}}(y|x)\,p_{M_x}(x) = p_{M_{x|y}}(x|y)\,p_{M_y}(y), \tag{6}$$

which describes the harmonic degree of the Ying–Yang pair. Such a learning theory is called *Bayesian Ying–Yang* (BYY) *Learning Theory*.

This $\min_{M_1, M_2} F_s$ can be implemented by an *Alternative Minimization* iterative procedure:

$$\begin{aligned} &\text{Step 1:} \quad \text{Fix } M_2 = M_2^{\text{old}}, \text{get } M_1^{\text{new}} = \arg\min_{M_1} F_s; \\ &\text{Step 2:} \quad \text{Fix } M_1 = M_1^{\text{old}}, \text{get } M_2^{\text{new}} = \arg\min_{M_2} F_s, \end{aligned} \tag{7}$$

which guarantees to reduce F_s until it converges to a local minimum.

Three categories of separation functionals, namely *Convex Divergence*, *L_p Divergence, and De-correlation Index*, have been suggested in (Xu, 1997c). Particularly, the *Convex Divergence* is defined as

$$F_s(M_1, M_2) = f(1) - \int_{x,y} p_{M_{y|x}}(y|x)\,p_{M_x}(x) \times f\left(\frac{p_{M_{x|y}}(x|y)\,p_{M_y}(y)}{p_{M_{y|x}}(y|x)\,p_{M_x}(x)}\right) \mathrm{d}x\,\mathrm{d}y, \tag{8}$$

where $f(u)$ is strictly convex on $(0, +\infty)$. The BYY learning is called Bayesian Convex YING-YANG (BCYY) learning. When $f(1) = 0$ and $f(u)$ is twice differentiable, Eq. (8) is equivalent to Csiszar general divergence. Particularly, when $f(u) = \ln u$, Eq. (8) becomes the well-known Kullback Divergence:

$$\mathrm{KL}(M_1, M_2) = \int_{x,y} p_{M_{y|x}}(y|x)\,p_{M_x}(x) \times \ln \frac{p_{M_{y|x}}(y|x)\,p_{M_x}(x)}{p_{M_{x|y}}(x|y)\,p_{M_y}(y)} \mathrm{d}x\,\mathrm{d}y. \tag{9}$$

In this special case, the BYY learning is called Bayesian-Kullback YING-YANG (BKYY) learning.

As shown in (Xu, 1997a), the theory given by Eq. (6) provides theoretical guides for *parameter learning*, *regularization*, *structural scale or complexity selection*, *architecture design and data sampling*. In this paper, we only consider the cases that the architecture has been pre-designed and a training set $D_x = \{x_i\}_{i=1}^N$ is given and that the remaining unspecified parts are the parameter set Θ and the structural scale k. In this case, we denote $F_s(M_1, M_2)$ simply by $F_s(\Theta_k, k)$. With k fixed, we determine

$$\Theta_k^* = \arg\min_{\Theta_k} F_s(\Theta_k, k), \tag{10}$$

which is called *parameter learning*. Then, we do *structural scale selection* by determining

$$k^* = \min_k \mathscr{K}, \quad \mathscr{K} = \{j \mid J_1(j) = \min_k J_1(k)\}, \quad J_1(k) = F_s(\Theta_k^*, k). \tag{11}$$

That is, to pick the smallest one among those values of k that makes $J_1(k)$ reach its smallest value. In other words, we select the simplest structural scale when we have multiple choices.

We also have an alternative way for selecting k^*,

$$k^* = \arg\min_k^{\mathrm{g}} J_2(k), \quad J_2(k) = -\int_{x,y} p_{M_1}(x,y)|_{\Theta_k^*} \ln p_{M_2}(x,y)|_{\Theta_k^*} \mathrm{d}x\,\mathrm{d}y, \tag{12}$$

where $p_{M_i}(x,y)|_{\Theta_k^*}$, $i = 1,2$, denote the learned joint densities given in Eq. (4) with the parameter Θ^* given by Eq. (10). This $J_2(k)$ is a kind of complexity measure of the BYY system and is expected to be the smallest for the least complicated system. Usually, $J_2(k)$ reaches its minimum for one value of k. In most cases, the results of Eq. (11) and (Eq. (12)) are the same. However, each way has a different feature, which will be discussed in the next section. In fact, $J_2(k)$ is just a part of $J_1(k)$.

3. BKYY learning, finite mixture and number of densities

3.1. BKYY learning, finite mixture and EM algorithm

Let $y = 1,\ldots,k$, $x \in R^d$ and $p_{M_x}(x) = p_h(x)$ given by Eq. (5), with other architectures being $p_{M_{x|y}}(x|y) = p(x|\theta_y)$ and [4]

$$p_{M_y}(y) = \alpha_y > 0, \quad \sum_{y=1}^{k} \alpha_y = 1,$$

$$p_{M_{y|x}}(y|x) = p(y|x) \geqslant 0, \quad \sum_{y=1}^{k} p(y|x) = 1. \tag{13}$$

That is, $p_{M_{x|y}}(x|y)$ is parametric, $p_{M_{y|x}}(y|x)$ and $p_{M_y}(y)$ are free probability functions.

Putting the above design into Eq. (9), we get

$$\begin{aligned}
&\mathrm{KL}(M_1, M_2) \\
&= \sum_{y=1}^{k} \int_x p(y|x)\, p_h(x) \ln \frac{p(y|x)\, p_h(x)}{p(x|\theta_y)\,\alpha_y}\, \mathrm{d}x \\
&= \mathrm{KL}(\Theta_k, p(y|x)) + \int_x p_h(x) \ln p_h(x)\, \mathrm{d}x,
\end{aligned}$$

$$\begin{aligned}
&\mathrm{KL}(\Theta_k, p(y|x)) \\
&= \int_x p_h(x) \left[\sum_{y=1}^{k} p(y|x) \ln p(y|x) \right] \mathrm{d}x \\
&\quad - \sum_{y=1}^{k} \int_x p(y|x)\, p_h(x) \ln p(x|\theta_y)\, \mathrm{d}x \\
&\quad - \sum_{y=1}^{k} \alpha_y \ln \alpha_y,
\end{aligned} \tag{14}$$

where $\Theta_k = \{\alpha_y, \theta_y\}_{y=1}^{k}$. We can equivalently just consider $\min_{\{p(y|x),\Theta_k\}} \mathrm{KL}(\Theta_k, p(y|x))$ since the second term in $\mathrm{KL}(M_1, M_2)$ is irrelevant to Θ_k. Moreover, by noticing that

$$\mathrm{KL}(\Theta_k, p(y|x)) = \int_x p_h(x) \mathrm{KL}_{y|x}\, \mathrm{d}x - L(\Theta_k),$$

$$\mathrm{KL}_{y|x} = \sum_{y=1}^{k} p(y|x) \ln \frac{p(y|x)}{p^*(y|x)},$$

$$L(\Theta_k) = \int_x p_h(x) \ln p(x, \Theta_k)\, \mathrm{d}x,$$

$$p^*(y|x) = \frac{p(x|\theta_y)\,\alpha_y}{p(x, \Theta_k)},$$

$$p(x, \Theta_k) = \sum_{y=1}^{k} \alpha_y p(x|\theta_y), \tag{15}$$

we further have Theorem 1.

Theorem 1. *BKYY learning under the above architecture design Eq. (13) is equivalent to obtaining $p^*(y|x)$ by Eq. (15) and simultaneously getting $k^*, \Theta_{k^*}^*$ by either maximizing $L(\Theta_k)$ or minimizing $\mathrm{KL}(\Theta_k, p(y|x))$ under the constraint $p(y|x) = p^*(y|x)$.*

That is, the result $k^*, \Theta_{k^*}^*$ obtained by BKYY learning in this special case is equivalent to the maximum log-likelihood (ML) solution of the finite mixture $p(x, \Theta_k)$ given by Eq. (15), based on $p_h(x)$ given by Eq. (5). In implementation, to avoid the difficulty due to the integral operations in Eqs. (14) and (15), when N is large enough, we approximate $\mathrm{KL}(\Theta_k, p(y|x))$ and $L(\Theta_k)$ by their limits as $h \to 0$:

$$\begin{aligned}
&\mathrm{KL}(\Theta_k, p(y|x)) \\
&\quad = -O_N(p(y|x)) \\
&\qquad - \frac{1}{N} \sum_{i=1}^{N} \sum_{y=1}^{k} p(y|x_i) \ln p(x_i|\theta_y) \\
&\qquad - \sum_{y=1}^{k} \alpha_y \ln \alpha_y,
\end{aligned}$$

$$L(\Theta_k) = \frac{1}{N} \sum_{i=1}^{N} \ln p(x_i, \Theta_k),$$

$$O_N(p(y|x)) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{y=1}^{k} p(y|x_i) \ln p(y|x_i). \tag{16}$$

[4] Strictly speaking, we can only use $p(\cdot)$ to denote a density of real a variable. For discrete y, we should use a probability $P(\cdot)$ to replace $p(\cdot)$. For convenience, we still use $p(\cdot)$ to denote a probability, but identified via y.

From which we can even more clearly see that $L(\Theta_k)$ is the likelihood function of $p(x,\Theta_k)$ by Eq. (15).

At a fixed k, the parameter learning Eq. (10) can be made by the ALTMIN, Eq. (7), as follows:

EM Algorithm

E step: get $p^*(y|x_i)$ by Eq. (15),

$$\text{and } \alpha_y = \frac{1}{N}\sum_{i=1}^{N} p^*(y|x_i);$$

M step: $\theta_y^{\text{new}} = \arg\max_{\theta_y} \sum_{i=1}^{N} p^*(y|x_i)\ln p(x_i|\theta_y).$ (17)

It is guaranteed to converge to a local maximum of $L(\Theta_k)$ since the ALTMIN, Eq. (7), is guaranteed to converge to one local minimum. Actually, it is exactly the well-known EM algorithm (Dempster et al., 1997). Here, we obtain it in a much simpler way, with its convergence proved easily.

3.2. The selection of scale k

Based on Eq. (16), we can use Eq. (11) or Eq. (12) for selecting the scale k – the number of densities in a mixture, with

$$J_1(k) = \mathrm{KL}(\Theta_k^*, p^*(y|x)),$$

$$J_2(k) = \frac{1}{N}\sum_{i=1}^{N}\sum_{y=1}^{k} p^*(y|x_i)\ln p(x_i|\theta_y^*) - \sum_{y=1}^{k}\alpha_y^*\ln\alpha_y^*, \quad (18)$$

where $\Theta_k^*, p^*(y|x)$ are the results of the parameter learning, e.g., by the EM algorithm Eq. (17). In the following, we provide some theorems for further theoretical justification on the selection criteria Eq. (18).

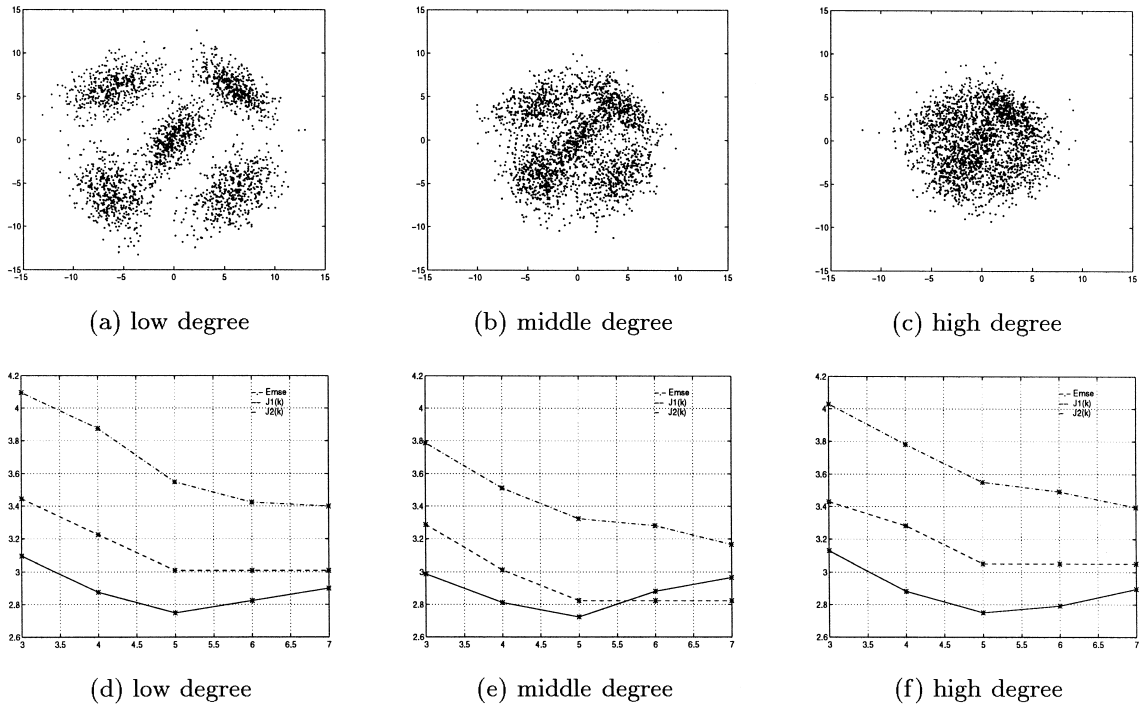

Fig. 2. The curves of $J_1^g(k)$ ‘‘- -’’, $J_2^g(k)$ ‘‘—’’ and E_{MSE} ‘‘-·-’’ with parameters estimated by the EM algorithm, on the data sets of five elliptic Gaussians with three different degrees of overlap.

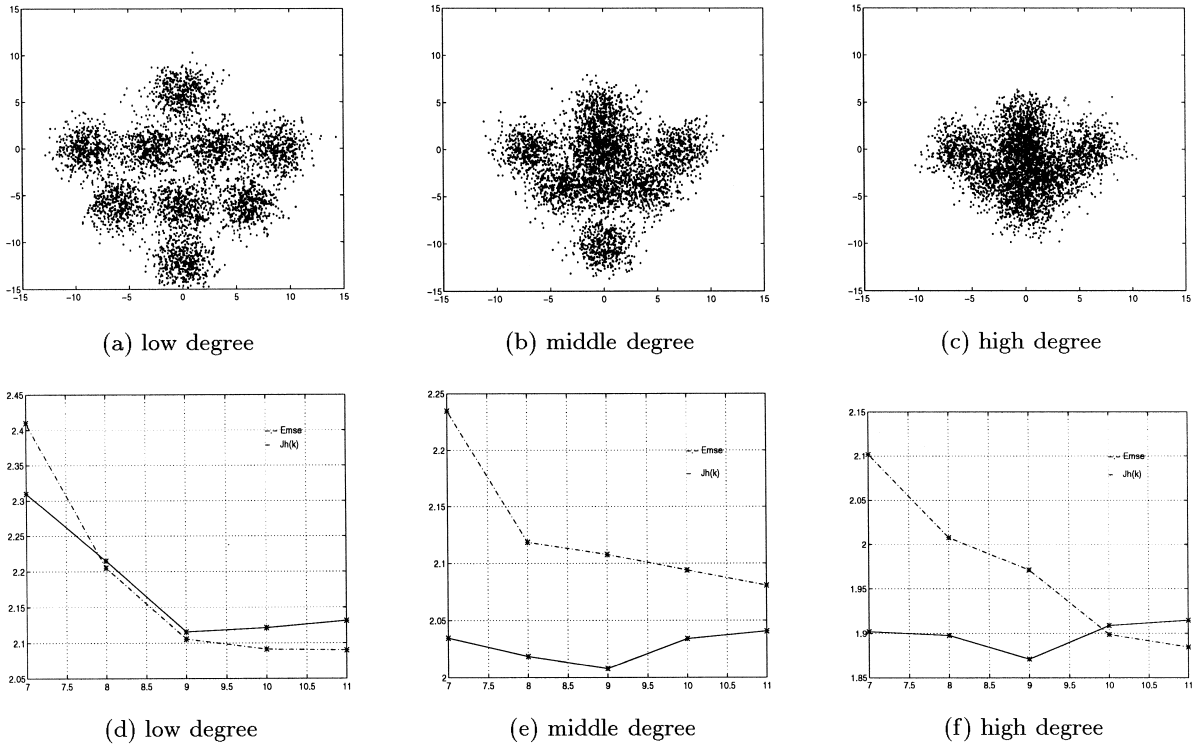

Fig. 3. The curves of $J_2^{\mathrm{gh}}(k)$ ''–'' and E_{MSE} ''-·-'' with parameters given by the k-means algorithm, on the data sets of nine spherical-shape Gaussians with three different degrees of overlap.

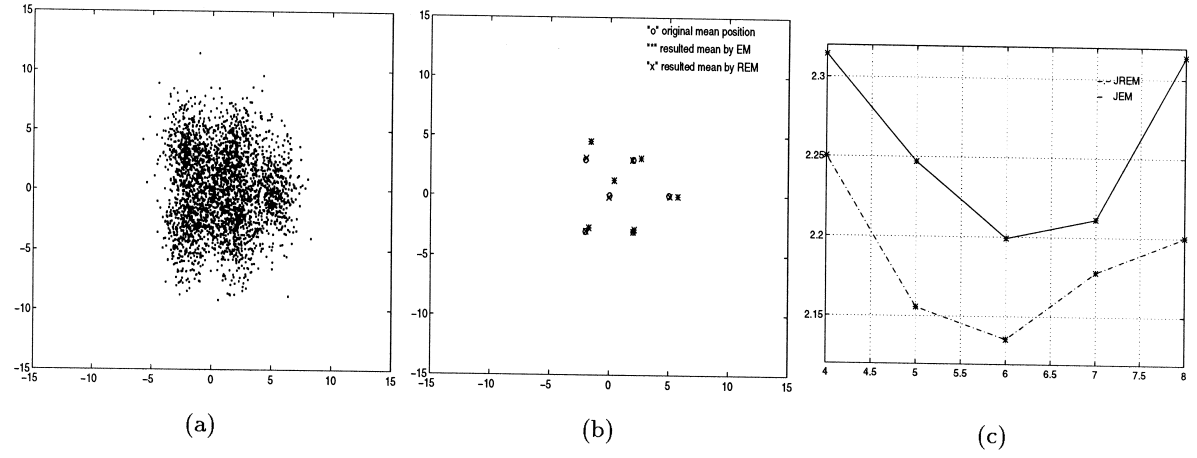

Fig. 4. Comparison of the REM and EM algorithms. (a) A mixture of six elliptic Gaussians with each long axis along y-direction; (b) the estimated means by REM ''x'', EM ''$*$'' versus the original ''o''; (c) The curves of $J_2^{\mathrm{gh}}(k)$ with parameters obtained by REM ''-·-'', EM ''–'', respectively.

Given a set $D_x = \{x_i\}_{i=1}^N$ from an original density $p^o(x)$,

$$p^o(x) = p^o(x, \Theta_{k^o}^o) = \sum_{y=1}^{k^o} \alpha_y^o p(x|\theta_y^o), \ \alpha_y^o > 0,$$

$$\sum_{y=1}^{k^o} \alpha_y^o = 1, \tag{19}$$

and $p_{M_x}(x) = p_{h_x}(x)$ by Eq. (5), we can get the following lemma.

Lemma 1. *When $p^o(x)$, $K(r)$ and $p(x|\theta_y)$ satisfy some mild regularity condition, as $Nh^d \to \infty$ and $h \to 0$, $\mathrm{KL}(\Theta_k, p(y|x))$ given by Eq. (14) or Eq. (16) tends to the following limit almost surely*:

$$\begin{aligned}\mathrm{KL}^o(\Theta_k, p(y|x)) &= \int_x p^o(x) \left[\sum_{y=1}^{k} p(y|x) \ln p(y|x) \right] \mathrm{d}x \\ &\quad - \sum_{y=1}^{k} \int_x p(y|x)\, p^o(x) \ln p(x|\theta_y)\, \mathrm{d}x \\ &\quad - \sum_{y=1}^{k} \alpha_y \ln \alpha_y, \end{aligned} \tag{20}$$

The condition is quite mild and can be given in several different forms. Here, we will not go into the details. This lemma ensures the consistence of the above approximation Eq. (16), and thus we are able to get some insight in Eq. (18) as N is large enough.

Theorem 2. *Given $J_1(k)$ by Eq. (18) and $p^o(x)$ by Eq. (19) with $p(x|\theta_y^o)$, $y = 1, \ldots, k^o$ being linearly independent. Also, for any Θ_k there is no θ_y that leads to a degenerate case $p(x|\theta_y) = \delta(x - c_y)$, with c_y being a constant. Then, as $Nh^d \to \infty$ and $h \to 0$ we have almost surely*:

(a) *$J_1(k^o) < J_1(k)$ for $k < k^o$ and $J_1(k^o) = J_1(k)$ for $k \geqslant k^o$;*

(b) *$J_1(k^o) < J_1(k)$ for any $k \neq k^o$ if and only if there is no Θ_k with $\theta_1 \neq \theta_2 \neq \cdots \neq \theta_k$ such that $p(x, \Theta_k) = p^o(x, \Theta_{k^o}^o)$.*

Here, we omit the proof. The condition of Theorem 2 is very mild. Particularly, it holds as long as $p(x|\theta_y)$, $y = 1, \ldots, k^o$ are linearly independent for any $\theta_1 \neq \theta_2 \neq \cdots \neq \theta_k$. Theorem 2 justifies the use of Eq. (18) as a criterion for the selection of k. *When N is large enough, as k increases we can calculate $J(k)$ until k^* with $J(k^*) - J(k^* + 1) = 0$.* However, if N is small, $J(k)$ may continue to decrease slowly even after $k \geqslant k^o$; in this case we can stop at k^* with $J(k^*) - J(k^* + 1) < \varepsilon$ with $\varepsilon > 0$ being a small threshold. Obviously, this ε should be chosen according to N. Theorem 2 also suggests an improvement on the EM algorithm Eq. (17), as will be discussed in Section 6.

Theorem 3. *Given $J_2(k)$ by Eq. (18), we define*

$$\begin{aligned} O_e(p^o(y|x)) &= -\int_x p^o(x) \left[\sum_{y=1}^{k^o} p^o(y|x) \ln p^o(y|x) \right] \mathrm{d}x, \\ p^o(y|x) &= \frac{\alpha_y^o p(x, \theta_y^o)}{p^o(x, \Theta_{k^o}^o)}. \end{aligned} \tag{21}$$

Under the same conditions as in Theorem 2, we have almost surely $J_2(k^o) < J_2(k)$ for any $k \neq k^o$ as long as $J_1(k) - J_1(k^o) > O_e(P^o(y|x))$ for $k < k^o$.

This $O_e(p^o(y|x))$ describes the degree of overlap between the component densities in $p^o(x)$ since $p^o(y|x)$ describes the degree that x belongs to the yth density. Theorem 3 says that as long as this overlap is not too high, we will have $J_2(k) < J_2(k^o)$ for any $k \neq k^o$. That is, in this case, $J_2(k)$ can be used, even when N is not large enough, as will be shown by the experimental results in Figs. 2–4.

4. Gaussian mixture, clustering and number of clusters [5]

4.1. Gaussian mixture, EM algorithm and number of Gaussians

Particularly, for Gaussian $p(x|\theta_y) = G(x, m_y, \Sigma_y)$ the finite mixture $p(x, \Theta_k)$ by Eq. (15) becomes

[5] The basic results in this section were first obtained in (Xu, 1995), and then some further extensions and variants were given in (Xu, 1996). In this section, those main results are systematically summarized into a concise form with certain modifications. Due to limited space, many details are still left to (Xu, 1995, 1996).

Gaussian mixture. The M-step of the EM algorithm Eq. (17) has a more detailed form as follows:

$$\text{M step:}\quad m_y^{\text{new}} = \frac{1}{\alpha_y N}\sum_{i=1}^{N} p^*(y|x_i)\,x_i,$$
$$\Sigma_y^{\text{new}} = \frac{1}{\alpha_y N}\sum_{i=1}^{N} p^*(y|x_i) \times (x_i - m_y^{\text{new}})(x_i - m_y^{\text{new}})^{\mathrm{T}}. \tag{22}$$

Moreover, for special cases of Σ_y (e.g., diagonal, spherical shape, etc), the above updating equation can be modified by incorporating the corresponding constraints. For example, for the spherical shape $\Sigma_y = \sigma_y^2 I$ and $\Sigma_y = \sigma^2 I$, we have (noticing that d is the dimension of x)

$$(\sigma_y^2)^{\text{new}} = \frac{1}{\alpha_y dN}\sum_{i=1}^{N} p^*(y|x_i)\|x_i - m_y^{\text{new}}\|^2,$$
$$(\sigma^2)^{\text{new}} = \frac{1}{dN}\sum_{y=1}^{k}\sum_{i=1}^{N} p^*(y|x_i)\|x_i - m_y^{\text{new}}\|^2. \tag{23}$$

Also, Theorems 1, 2 and 3 still hold. Here, the degenerate case $p(x|\theta_y) = \delta(x - m_y)$ happens when Σ_y becomes singular. Moreover, $G(x, m_y, \Sigma_y)$, $y = 1,\ldots,k^{\circ}$, are linearly independent when $m_1 \neq m_2 \neq \cdots \neq m_k$ only. This simplified property will be used in Section 6 to improve the EM algorithm.

Moreover, since

$$\mathrm{tr}\left[\frac{1}{\alpha_y N}\sum_{i=1}^{N} p(y|x_i)(x_i - m_y)(x_i - m_y)^{\mathrm{T}}\Sigma_y^{-1}\right] = \mathrm{tr}[I] = d,$$

by some derivation and ignoring some constant, $J_1(k)$ and $J_2(k)$ become

$$J_1^{\mathrm{g}}(k) = -O_N(p^*(y|x)) + J_2^{\mathrm{g}}(k),$$
$$J_2^{\mathrm{g}}(k) = \sum_{y=1}^{k}\alpha_y^* \ln\sqrt{|\Sigma_y^*|} - \sum_{y=1}^{k}\alpha_y^* \ln \alpha_y^*, \tag{24}$$

where $O_N(p(y|x))$ is still the same as given in Eq. (16). Also, $J_2^{\mathrm{g}}(k)$ can be further simplified for special cases of Σ_y. For example, we have

$$J_2^{\mathrm{g}}(k) = d\sum_{y=1}^{k}\alpha_y^* \ln\sigma_y^* - \sum_{y=1}^{k}\alpha_y^* \ln \alpha_y^*,$$
$$\text{for } \Sigma_y = \sigma_y^2 I,$$
$$J_2^{\mathrm{g}}(k) = d\ln\sigma^* + \ln k,$$
$$\text{for } \Sigma_y = \sigma^2 I \text{ and } \alpha_y = 1/k. \tag{25}$$

4.2. Hard-cut implementation, clustering algorithms and number of clusters

The purpose of cluster analysis is to partition a data set $\{x_i\}_{i=1}^N$ into non-overlapping regions R_y, $y = 1,\ldots,k$. It is equivalent to consider the ideal case that the data comes from a mixture of $p(x|\theta_y)$ with a priori α_y on the non-overlapping supports $R_y \subseteqq R^d$, $y = 1,\ldots,k$, such that $p(x|\theta_y) = 0$ if x is not in R_y, and then we can classify $x \in R_y$ with full certainty because we have $p(y|x) = 1$ and $p(j|x) = 0$ for $j \neq y$. In this case, the classification of x to a density and the partitioning of R^d into non-overlapping regions R_y, $y = 1,\ldots,k$, are equivalent in probability 1. All the previous results still apply to this special case. Moreover, we also simply have $J_1(k) = J_2(k)$ since $O_N(p^*(y|x)) = 0$.

However, in practice most of $p(x|\theta_y)$ are supported on overlapping regions. We often assign an x_i into one of non-overlapping regions by Bayesian Decision $y = \arg\max_y p(y|x_i)$, which is equivalent to hard-cut or quantize $p(y|x_i)$ into

$$I(y|x_i) = \begin{cases} 1 & \text{if } y = \arg\max_j \alpha_j p(x|\theta_j), \\ 0 & \text{otherwise.} \end{cases} \tag{26}$$

For a Gaussian mixture, Eq. (26) becomes

$$I(y|x_i) = \begin{cases} 1 & \text{if } y = \arg\min_y d(x_i|\alpha_y, m_y, \Sigma_y), \\ 0 & \text{otherwise,} \end{cases} \tag{27}$$

where $d(x_i|\alpha_y, m_y, \Sigma_y)$ is generally given by

$$d(x_i|\alpha_y, m_y, \Sigma_y) = -\ln\alpha_y\sqrt{|\Sigma_y|} + 0.5(x_i - m_y)^{\mathrm{T}}\Sigma_y^{-1}(x_i - m_y),$$
$$\alpha_y = \frac{1}{N}\sum_{i=1}^{N} I(y|x_i), \tag{28}$$

which can be further simplified for various special cases of Σ_y.

$I(y|x_i)$ partitions the whole domain into R_y, $y = 1,\ldots,k$, non-overlapping regions such that each

region R_y supports a hardcut density induced from $p(x,\Theta_k)$:

$$q(x,R_y,\Theta_k)=\begin{cases}\dfrac{p(x,\Theta_k)}{\alpha'_y} & \text{if } x\in R_y,\\ 0 & \text{otherwise,}\end{cases}$$

$$\text{with } \alpha'_y=\int_{x\in R_y} p(x,\Theta_k)\,\mathrm{d}x. \tag{29}$$

As a result, we have an induced finite mixture expression $p(x,\Theta_k)=\sum_{y=1}^{k}\alpha'_y q(x,R_y,\Theta_k)$ for clustering purpose. For this mixture, theoretically we can still get a $J_1(k)=J_2(k)$ for selecting k. However, in practice R_y is usually difficult to handle.

For simplicity, we directly use $I(y|x_i)$ to replace all the occurrences of $p^*(y|x)$ in $J_2(k)$ and the EM algorithm obtained previously, which results in their hardcut variants that can save the computing cost significantly. For a Gaussian mixture, with $I(y|x_i)$ by Eq. (27) to replace $p^*(y|x)$ we can have:

(a) *The hardcut EM Algorithm*: its E-step consists of getting $I(y|x_i)$ by Eq. (27) and α_y by Eq. (28); and its M-step consists of simply Eq. (22) and Eq. (23) with $p^*(y|x)$ replaced with $I(y|x_i)$ by Eq. (27).

(b) *The hardcut* $J_2^{g_h}(k)$, which is simply $J_2^{g}(k)$ given in Eqs. (24) and (25) with $p^*(y|x)$ replaced with $I(y|x_i)$ by Eq. (27).

In particular, for the special case of equal $\alpha_y=1/k$ and equal spherical shape $\Sigma_y=\sigma^2 I$, Eq. (27) simplifies into Eq. (1) exactly, and the hardcut EM Algorithm reduces into exactly the k-means algorithm Eq. (3), after ignoring the updating on σ^2 since it is not needed in the MSE clustering.

Moreover, from Eq. (25) we can get $J_2^{g_h}(k)$ for selecting the k that is used in the k-means algorithm Eq. (3):

$$J_2^{g_h}(k)=d\ln\sigma^*+\ln k,\quad \sigma^2=\frac{E_{\mathrm{MSE}}}{dN}, \tag{30}$$

with E_{MSE} given by Eq. (2). Moreover, we can even simplify it into

$$J_2^{g_h}(k)=E_{\mathrm{MSE}}^{(d/2)}k. \tag{31}$$

As shown in Fig. 3, although E_{MSE} decreases monotonically with increasing k, $J_2^{g_h}(k)$ has a U-shape with a clear minimum at the correct k^o due to the fact that $\ln k$ increases monotonically with k.

Furthermore, for various special cases of $\Sigma_y\neq\sigma^2 I$, the hardcut EM Algorithm will become various types of extensions of the k-means algorithm, including those so called Weighted MSE clustering, Mahalanobis distance clustering or elliptic clustering (Xu, 1996). Here we give them not only a unified form, but also we give criteria for detecting the correct number of clusters.

5. BCYY learning, finite mixture and REM algorithm

Instead of using the Kullback divergence Eq. (9) as $F_s(M_1,M_2)$, we can also use the *Convex Divergence* Eq. (8) and get the corresponding learning called *Bayesian Convex Ying–Yang* (BCYY) learning. In this case, we cannot get an expanded form as Eq. (14). However, we can still directly use ALTMIN, Eq. (7), for the minimization of $F_s(M_1,M_2)$, and its first step will result in the E-step in the EM algorithm Eq. (17). With $p^*(y|x_i)$ thus obtained put into Eq. (8), $L(\Theta_k)$ in Eq. (16) becomes

$$L_f(\Theta_k)=\frac{1}{N}\sum_{i=1}^{N} f(p(x_i,\Theta_k))=\frac{1}{N}\sum_{i=1}^{N} f(\mathrm{e}^{\ln p(x_i,\Theta_k)}). \tag{32}$$

That is, $\arg\min_{M_1,M_2} F_s(M_1,M_2)$ is equivalent to firstly obtaining $p^*(y|x)$ by Eq. (15) and then getting $k^*,\Theta_{k^*}^*$ by maximizing $L_f(\Theta_k)$. This is a generalized ML learning procedure for a finite mixture.

Moreover, the M-step in Eq. (17) or Eq. (22) will become, respectively,

M step: get θ_y^{new} by solving

$$\sum_{i=1}^{N} w(y,x_i)\frac{\mathrm{d}\ln p(x_i|\theta_y)}{\mathrm{d}\theta_y}=0;$$

M step:

$$m_y^{\mathrm{new}}=\frac{1}{\alpha_y N}\sum_{i=1}^{N} w(y,x_i)x_i,$$

$$\Sigma_y^{\mathrm{new}}=\frac{1}{\alpha_y N}\sum_{i=1}^{N} w(y,x_i)\times(x_i-m_y^{\mathrm{new}})(x_i-m_y^{\mathrm{new}})^{\mathrm{T}}, \tag{33}$$

where $w(y,x_i) = f'(p(x_i,\Theta_k))p(x_i,\Theta_k)p^*(y|x_i)$ and $f'(u) = \mathrm{d}f(u)/\mathrm{d}u$. Here, the original weight $p^*(y|x_i)$ is reweighted into $w(y,x_i)$. We call the corresponding EM algorithm the *Re-weighted EM* (REM) algorithm.

When $f(u)$ is monotonically increasing for positive u, e.g., $f(u) = u^\beta$, $0 < \beta < 1$. The effect is the maximization of $\mathrm{e}^{\beta\xi}$, $\xi = \ln p(x_i,\Theta_k)$. $\mathrm{e}^{\beta\xi}$ is also monotonically increasing with ξ, and gives a larger weight to larger values of ξ. Thus, the maximization of $L_f(\Theta_k)$ gives more weight to those samples with large $p(x_i,\Theta_k)$. In other words, the learning relies more on those samples around the modes of each density, while the boundary samples are discounted. Thus, *the learning will give more robust estimations in cases that data consists of multi-modes with outliers or high overlap between densities*. The more close the β is to 1, the more rapid $\mathrm{e}^{\beta\xi}$ changes, the more robust the learning will be.

6. Implementation and experiments

Several improvements can be made on the implementation of the EM algorithm, its hardcut variants and the selection of k. *First*, we can use RPCL (Xu et al., 1993) to find an initial estimation of k, and then finely search a best k^* via $J_1(k)$ or $J_2(k)$ around k. *Second*, for each fixed k, before running the EM algorithm or one of its hardcut variants, we can run some heuristic clustering algorithm to get an initialization. *Third*, during the running of the EM algorithm or one of its hardcut variants, according to Theorem 2 we can introduce in each iteration the following two enhancements:

(1) Once we find that Σ_y becomes singular or $\alpha_y = 0$, we can simply remove the corresponding $p(x|\theta_y)$.

(2) If there are two $y_1 \neq y_2$ such that $\theta_{y_1} = \theta_{y_2}$ or very close to each other, we can simply remove the corresponding $p(x|\theta_{y_1})$ and merge $\alpha_{y_1} + \alpha_{y_2} \to \alpha_{y_2}$.

Due to space limits, we only focus on demonstrating how the proposed criteria for selecting k work. In Fig. 2, all the parameters in the mixture are unknown, and the EM algorithm Eq. (22) was used for solving the parameters of each Gaussian. We can observe that E_{MSE} decreases monotonically as k increases, but both $J_1^{\mathrm{g}}(k)$, $J_2^{\mathrm{g}}(k)$ given by Eq. (24) can detect the correct $k^* = 5$. $J_2^{\mathrm{g}}(k)$ has its minimum at $k^* = 5$, while $J_1^{\mathrm{g}}(k)$ flattens out at $k^* = 5$. In Fig. 3, the k-means algorithm was used for getting the cluster centers. Again, E_{MSE} decreases monotonically as k increases, but $J_2^{\mathrm{g}_h}(k)$ given by Eq. (30) can detect the correct $k^* = 9$ at its minimum. Fig. 4 gives a comparison of the EM algorithm Eq. (22) and the REM algorithm Eq. (33). We can see that REM gives a more accurate estimate of the mean vectors. Moreover, from Fig. 4(c) we can successfully detect the correct $k^* = 6$ by the curves $J_2^{\mathrm{g}}(k)$ by Eq. (24) also.

7. Conclusions

We have obtained not only a unified form for various extensions of the MSE clustering and the k-means algorithm, but also criteria for selecting the number of densities in a mixture and the number k in the k-means algorithm. Moreover, a REM learning algorithm is given and shown to be more robust than the EM algorithm.

Acknowledgements

The author would like to thank Mr Wing-kai Lam for the help in preparing Figs. 2–4.

Discussion

Mao: I have a comment and a question. I agree with you that the Ying–Yang machine is a very general framework. I certainly see connections between the minimal description length model and the Ying–Yang machine. For example, the logarithm of the clustering criterion which you presented can be decomposed into two terms. One is the encoding length of the data, given the model, and the other is the description length of the model itself. So in that case, the minimal description length is a special case of the Ying–Yang machine.

Now my question: you claim that you can automatically determine the number of clusters for given data. I guess this is true if you are looking for Gaussian clusters. In general, for many data sets, the concept of clusters is not well-defined. For example, for many perceptual patterns you can have many different ways to form clusters. If you apply your Ying–Yang machine, you will end up with a certain

number of clusters. But, depending on how you view the data, you may have another number of clusters. Do you have any comments on that?

Xu: First, your comment is correct. The minimal description length is a special case of the Ying–Yang machine. Concerning your question: in general you are right, if you do not have a model for the clusters, then it is difficult to detect the number of clusters. It should be based on some probabilistic model. You can use a finite mixture of any probability densities, not necessarily Gaussian.

Nagy: How do you plan to generalize your theory?

Xu: I have already generalized. What I claim here has already been done. I have several conference papers in the last three years. I also have some Journal papers which will appear soon, for instance in a special issue on Computational Learning Theory of ''Algorithmica''. So, what I claim here has already been done.

Nagy: What I ask is, where do you move from here?

Kanal: In other words, what is your next step? Are you going to retire now?

Xu: What I presented here is just for perception and pattern recognition. It is a level lower than the graphical models that have been presented. Actually this is a graph with two nodes. Ying–Yang may generalize to some graphical model. In the modelling of time series, in the linear case, Ying–Yang is equivalent to a Kalman filter. But in the non-linear case, it can only be used for one-dimensional problems. If you have a time series in two dimensions in state space, that cannot be handled yet, but that should be even more useful. It is also successfully related to a hidden Markov model. I have a paper on that in a Chinese & IEEE joint conference, two years ago. The hidden Markov model is o.k. but for a more general case it is not successful yet.

Mardia: The quantity $J(k)$ which you use in your mixtures seems to be a particular case of the Akaki criterion.

Xu: Yes it looks like it. Because Akaki is also related to the Kullback divergence in the matching of data to model densities. But the difference is that I have two models. I have a Ying-model and a Yang-model. But Akaki is more limited. It is originally used for the linear model in time series.

Mardia: The Akaki criterion is again a family of goodness of fit criteria, so it is not a particular case.

Xu: Yes, it is also of use in a neural network. But I just mentioned that it is used for a model of a mixture density, but not for two models. Ying–Yang are two models which are marching together. It has some relationship, it looks quite similar, but details are different.

References

Dempster, A.P., Laird, N.M., Rubin, D.B., 1997. Maximum-likelihood from incomplete data via the EM algorithm. J. Roy. Statist. Soc. Ser. B 39, 1–38.

Devroye, L., 1987. A Course in Density Estimation. Birhhauser, Boston, MA.

Jain, A.K., Dubes, R.C., 1988. Algorithm for Clustering Data. Prentice-Hall, Englewood Cliffs, NJ.

Xu, L., 1995. YING-YANG machine: A Bayesian-Kullback scheme for unified learnings and new results on vector quantization, Keynote talk. In: Proc. Internat. Conf. on Neural Information Processing (ICONIP95), pp. 977–988.

Xu, L., 1996. How many clusters?: A YING-YANG machine based theory for a classical open problem in pattern recognition. Invited Talk. In: Proc. 1996 IEEE Internat. Conf. on Neural Networks, Vol. 3, pp. 1546–1551.

Xu, L., 1997a. Bayesian Ying–Yang system and theory as a unified statistical learning approach: (I) For unsupervised and semi-unsupervised learning. In: Amari, S., Kassabov, N. (Eds.), Brain-like Computing and Intelligent Information Systems. Springer, Berlin.

Xu, L., 1997b. Bayesian ying-yang system and theory as a unified statistical learning approach: (II) Supervised learning. In: Proc. Internat. Workshop on Theoretical Aspects of Neural Computation, Hong Kong, 26–28 May. Lecture Notes in Computer Science. Springer, Berlin.

Xu, L., 1997c. New advances on Bayesian ying-yang learning system with Kullback and non-Kullback separation functionals. In: Proc. 1997 IEEE Internat. Conf. on Neural Networks, Vol. III, pp. 1942–1947.

Xu, L., Krzyzak, A., Oja, E., 1993. Competitive learning for clustering analysis, RBF net and curve detection. IEEE Trans. Neural Networks 4 (4), 636–649.