Expectation-MiniMax Approach to Clustering Analysis*

Yiu-ming Cheung

Department of Computer Science
Hong Kong Baptist University, Hong Kong
ymc@comp.hkbu.edu.hk

Abstract. This paper proposes a general approach named *Expectation-MiniMax (EMM)* for clustering analysis without knowing the cluster number. It describes the contrast function of Expectation-Maximization (EM) algorithm by an approximate one with a designable error term. Through adaptively minimizing a specific error term meanwhile maximizing the approximate contrast function, the EMM automatically penalizes all rivals during the competitive learning. Subsequently, the EMM not only includes the *Rival Penalized Competitive Learning* algorithm (Xu et al. 1993) and its Type A form (Xu 1997) with the new variants developed, but also provides a better alternative way to optimize the EM contrast function with at least two advantages: (1) faster model parameter learning speed, and (2) automatic model-complexity selection capability. We present the general learning procedures of the EMM, and demonstrate its outstanding performance in comparison with the EM.

1 Introduction

In the literature, the conventional clustering algorithm k-means [4] has been widely used in a variety of applications, which however needs to pre-assign an appropriate cluster number. Otherwise, it often leads to a poor clustering result. Unfortunately, such a setting is an intractable problem from a practical viewpoint. Alternatively, clustering problem has been studied by formulating as a finite mixture of Gaussian densities, in which each density generally describes a cluster in any elliptic shape and in any portion of samples [8]. Subsequently, the Gaussian mixture with the parameters estimated by the Expectation-Maximization (EM) algorithm [3] provides a general solution for clustering in parallel [3, 5]. However, in analog with the k-means algorithm, it still needs to pre-assign the correct number of densities.

In the past decades, some works have been done towards determining the correct number of clusters or densities along two major lines. The first one is to formulate the cluster number selection as the choice of component number in a finite mixture model. Consequently, there have been some criteria proposed

* This work was supported by the Faculty Research Grant of Hong Kong Baptist University (Project No: FRG/02/03/I-06).

for model selection, such as AIC [1], SIC [6] and so forth. Often, these existing criteria may overestimate or underestimate the cluster number due to the difficulty of choosing an appropriate penalty function. In recent years, a number selection criterion developed from Ying-Yang Machine has been proposed and experimentally verified in [8], whose computing however is laborious.

In contrast, the other line aims to select an appropriate cluster number automatically by algorithms themselves during the competition learning without a large amount of extra computing. In the literature, the typical example is the Rival Penalized Competitive Learning (RPCL) algorithm [9] and its variant RPCL (Type A) [8]. Its basic idea is that for each input, not only the winner of the seed points is updated to adapt to the input, but also its nearest rival (the second winner) is de-learned by a smaller learning rate (also called *de-learning rate* hereafter). Many experiments have shown that the RPCL can perform well in clustering analysis without knowing the cluster number. However, such a penalization scheme is heuristically proposed without any theoretical guidance. In this paper, we propose a general learning approach named *Expectation-MiniMax (EMM)* that describes the contrast function of EM algorithm by an approximate one with a designable error term. Through adaptively minimizing a specific error term meanwhile maximizing the approximate contrast function, we will show that the EMM automatically possesses the penalization scheme to punish all rivals during the competitive learning. Subsequently, the EMM not only includes the RPCL and its Type A form with the new variants proposed, but also shows that such a rival penalized learning actually provides a better alternative way to optimize the EM contrast function with at least two advantages: (1) faster model parameter learning speed, and (2) automatic model-complexity selection capability. We will give out the general learning procedures of the EMM, and show its superior performance in comparison with the EM.

2 Expectation-MiniMax (EMM) Learning Approach

2.1 General Learning Framework

Suppose N observations $\mathbf{x}_1$, $\mathbf{x}_2$, …, $\mathbf{x}_N$ are independently and identically distributed from an identifiable finite-mixture density population:

$$p^*(\mathbf{x};\boldsymbol{\Theta}^*) = \sum_{j=1}^{k^*} \alpha_j^* p(\mathbf{x};\boldsymbol{\theta}_j^*), \qquad \sum_{j=1}^{k^*} \alpha_j^* = 1, \quad \text{and} \qquad \alpha_j^* > 0, \tag{1}$$

where k^* is the true mixture number of densities, $\boldsymbol{\Theta}^* = \{(\alpha_j^*, \boldsymbol{\theta}_j^*) | 1 \leq j \leq k^*\}$ is the unknown true parameter set. The paper [3] shows that the estimate of $\boldsymbol{\Theta}^*$, written as $\boldsymbol{\Theta}$, can be achieved by maximizing the following contrast function:

$$Q(\mathbf{X}_N;\boldsymbol{\Theta}) = \frac{1}{N}\sum_{t=1}^{N} q_t(\mathbf{x}_t;\boldsymbol{\Theta}), \tag{2}$$

with

$$q_t(\mathbf{x}_t;\boldsymbol{\Theta}) = \sum_{j=1}^{k} h(j|\mathbf{x}_t)\ln[\alpha_j p(\mathbf{x}_t;\boldsymbol{\theta}_j)], \tag{3}$$

where $\mathbf{X}_N = [\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_N^T]^T$, and the candidate mixture number k measures the model complexity. Furthermore, $h(j|\mathbf{x})$ is the posterior probability of the j^{th} density as given $\mathbf{x}$ with

$$h(j|\mathbf{x}) = \frac{\alpha_j p(\mathbf{x};\boldsymbol{\theta}_j)}{p(\mathbf{x};\boldsymbol{\Theta})} = \frac{\alpha_j p(\mathbf{x};\boldsymbol{\theta}_j)}{\sum_{r=1}^{k}\alpha_r p(\mathbf{x};\boldsymbol{\theta}_r)}, \tag{4}$$

and

$$p(\mathbf{x};\boldsymbol{\Theta}) = \sum_{j=1}^{k}\alpha_j p(\mathbf{x};\boldsymbol{\theta}_j), \quad \sum_{j=1}^{k}\alpha_j = 1, \quad \alpha_j > 0. \tag{5}$$

Hence, with a specific k, minimizing Eq.(2) can be implemented by an adaptive EM algorithm [7]. In Eq.(3), we further replace $h(j|\mathbf{x}_t)$ by

$$I(j|\mathbf{x}_t) = \begin{cases} 1, \text{ if } j = c = \arg\max_{1\le r\le k} h(r|\mathbf{x}_t); \\ 0, \text{ otherwise.} \end{cases} \tag{6}$$

Subsequently, the contrast function $Q(\mathbf{x};\boldsymbol{\Theta})$ in Eq.(2) is approximated by

$$R(\mathbf{X}_N;\boldsymbol{\Theta}) = \frac{1}{N}\sum_{t=1}^{N} J_t(\mathbf{x}_t;\boldsymbol{\Theta}), \quad J_t(\mathbf{x}_t;\boldsymbol{\Theta}) = \sum_{j=1}^{k} I(j|\mathbf{x}_t)\ln[\alpha_j p(\mathbf{x}_t;\boldsymbol{\theta}_j)]. \tag{7}$$

We therefore express $Q(\mathbf{X}_N;\boldsymbol{\Theta})$ by

$$Q(\mathbf{X}_N;\boldsymbol{\Theta}) = R(\mathbf{X}_N;\boldsymbol{\Theta}) - E(\mathbf{X}_N;\boldsymbol{\Theta}) \tag{8}$$

with $E(\mathbf{X}_N;\boldsymbol{\Theta}) = \frac{1}{N}\sum_{t=1}^{N} e(\mathbf{x}_t;\boldsymbol{\Theta})$, $e(\mathbf{x}_t;\boldsymbol{\Theta}) \ge 0$, where $E(\mathbf{X}_N;\boldsymbol{\Theta})$ is the average approximate error, and $e(\mathbf{x}_t;\boldsymbol{\Theta})$ measures an instantaneous error at time step t. In general, $E(\mathbf{X}_N;\boldsymbol{\Theta})$ varies with the change of $\mathbf{\Theta}$, thus maximizing $R(\mathbf{X}_N;\boldsymbol{\Theta})$ is not equivalent to maximize $Q(\mathbf{X}_N;\boldsymbol{\Theta})$. That is, we should minimize the approximate error $E(\mathbf{X}_N;\boldsymbol{\Theta})$, meanwhile maximizing $R(\mathbf{X}_N;\boldsymbol{\Theta})$. Subsequently, at each time step t, after calculating $h(j|\mathbf{x}_t)$s by Eq.(4) (also called the *Expectation*-step in the EM), we adjust the parameters with a small step towards minimizing $e(\mathbf{x}_t;\boldsymbol{\Theta})$ meanwhile maximizing $J_t(\mathbf{x}_t;\boldsymbol{\Theta})$. We name such a learning *Expectation-MiniMax (EMM)* approach. It can be seen that EMM degenerates to the EM learning [3] as the error $e(\mathbf{x}_t;\boldsymbol{\Theta}) = \bar{e}(\mathbf{x}_t;\boldsymbol{\Theta}) = J_t(\mathbf{x}_t;\boldsymbol{\Theta}) - q_t(\mathbf{x}_t;\boldsymbol{\Theta})$. In the EMM, we have noticed that replacing $h(j|\mathbf{x}_t)$s by $I(j|\mathbf{x}_t)$s at each time step t eventually brings about $E(\mathbf{X}_N;\boldsymbol{\Theta})$. Hence, we can generally describe the error $e(\mathbf{x}_t;\boldsymbol{\Theta})$ as a function of both $h(j|\mathbf{x})_t$s and $I(j|\mathbf{x}_t)$s so long as each $e(\mathbf{x}_t;\boldsymbol{\Theta}) \ge 0$ holds. Subsequently, we can describe the relationship between it and $\bar{E}(\mathbf{X}_N;\boldsymbol{\Theta})$ $= \frac{1}{N}\sum_{t=1}^{N}\bar{e}(\mathbf{x}_t;\boldsymbol{\Theta})$ by

$$\bar{E}(\mathbf{X}_N;\boldsymbol{\Theta}) = \lambda(\boldsymbol{\Theta})E(\mathbf{X}_N;\boldsymbol{\Theta}), \tag{9}$$

where $\lambda(\boldsymbol{\Theta})$ is a positive scale number, which is generally a function of $\boldsymbol{\Theta}$ only, irrelevant to N observations. Hence, estimation of $\boldsymbol{\Theta}$ by maximizing $Q(\mathbf{X}_N; \boldsymbol{\Theta})$ in Eq.(2) is equivalent to maximize

$$Q_m(\mathbf{X}_N; \boldsymbol{\Theta}) = R(\mathbf{X}_N; \boldsymbol{\Theta}) - \lambda_m E(\mathbf{X}_N; \boldsymbol{\Theta}) \tag{10}$$

where λ_m represents the scale number calculated via Eq.(9) with the value of $\boldsymbol{\Theta}$ being the optimal solution of maximizing $Q(\mathbf{X}_N; \boldsymbol{\Theta})$ in Eq.(2). Under the circumstances, it can be seen that maximizing Eq.(10) via the EMM generally leads to the same solution as the EM. In general, we however need not estimate λ_m. Instead, we simply regard λ_m as a constant during the learning. Subsequently, the EMM provides an approximate, but better way to maximize $Q(\mathbf{X}_N; \boldsymbol{\Theta})$ of Eq.(2). In the following subsection, we will give out a general EMM learning algorithm under a specific modeling of $E(\mathbf{X}_N; \boldsymbol{\Theta})$.

2.2 A General EMM Learning Algorithm

We let

$$E(\mathbf{X}_N; \boldsymbol{\Theta}) = \frac{1}{N}\sum_{t=1}^{N} e(\mathbf{x}_t; \boldsymbol{\Theta}) = \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{k}[I(j|\mathbf{x}_t) - h(j|\mathbf{x}_t)]^2. \tag{11}$$

Eq.(10) can then be specified as

$$Q_m(\mathbf{X}_N; \boldsymbol{\Theta}) = \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{k} J_t(\mathbf{x}_t; \boldsymbol{\Theta}) - \lambda_m \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{k} e(\mathbf{x}_t; \boldsymbol{\Theta}). \tag{12}$$

Hence, maximizing $Q_m(\mathbf{X}_N; \boldsymbol{\Theta})$ in Eq.(12) can be realized towards maximizing $R(\mathbf{X}_N; \boldsymbol{\Theta})$ meanwhile minimizing $E(\mathbf{X}_N; \boldsymbol{\Theta})$. In adaptive implementation, we have the following EMM learning algorithm:

Step 1 Initialize the parameter set $\boldsymbol{\Theta}$ as given a specific k.
Step 2 Given each input $\mathbf{x}_t$, calculate $I(j|\mathbf{x}_t)$ by Eq.(6) with $h(j|\mathbf{x}_t)$ given by Eq.(4) with $\boldsymbol{\Theta}$ fixed.
Step 3 Fix $I(j|\mathbf{x}_t)$, update $\boldsymbol{\Theta}$ with a small step towards the direction of maximizing $R(\mathbf{X}_N; \boldsymbol{\Theta})$. To avoid the constraint on α_js during the learning, we let α_js be the soft-max function of k totally-free new variables β_js with

$$\alpha_j = \frac{\exp(\beta_j)}{\sum_{r=1}^{k}\exp(\beta_r)}. \tag{13}$$

Subsequently, we update

$$\boldsymbol{\beta}_c^{(\text{new})} = \boldsymbol{\beta}_c^{(\text{old})} + \eta_1 \frac{\partial J_t(\mathbf{x}_t; \boldsymbol{\Theta})}{\partial \boldsymbol{\beta}_c}|_{\boldsymbol{\beta}_c^{(\text{old})}} \tag{14}$$

$$\boldsymbol{\theta}_c^{(\text{new})} = \boldsymbol{\theta}_c^{(\text{old})} + \eta_1 \frac{\partial J_t(\mathbf{x}_t; \boldsymbol{\Theta})}{\partial \boldsymbol{\theta}_c}|_{\boldsymbol{\theta}_c^{(\text{old})}}, \tag{15}$$

where η_1 is a small positive step size. We denote the updating results of **Step 3** as $\boldsymbol{\Theta}^a = \{\beta_j^a, \boldsymbol{\theta}_j^a\}_{j=1}^k$ with

$$\beta_j^a = \begin{cases} \beta_c^{(\text{new})}, \text{ if } \quad j = c, \\ \beta_j^{(\text{old})}, \text{ if } \quad j \neq c \end{cases} \qquad \boldsymbol{\theta}_j^a = \begin{cases} \boldsymbol{\theta}_c^{(\text{new})}, \text{ if } \quad j = c, \\ \boldsymbol{\theta}_j^{(\text{old})}, \text{ if } \quad j \neq c. \end{cases} \tag{16}$$

Step 4 Fix $\boldsymbol{\Theta}^a$, we let

$$h^a(j|\mathbf{x}_t) = \begin{cases} \frac{\alpha_c^{(\text{new})} p(\mathbf{x}_t; \boldsymbol{\theta}_c^{(\text{new})})}{p(\mathbf{x}_t; \boldsymbol{\Theta}^a)}, \text{ if } \quad j = c \\ \frac{\alpha_j^{(\text{old})} p(\mathbf{x}; \boldsymbol{\theta}_j^{(\text{old})})}{p(\mathbf{x}_t; \boldsymbol{\Theta}^a)}, \quad \text{otherwise,} \end{cases} \tag{17}$$

with $\alpha_c^{(\text{new})}$ calculated by Eq.(13) based on $\beta_c^{(\text{new})}$ and those $\beta_j^{(\text{old})}$s with $j \neq c$. Then, we adjust $\boldsymbol{\Theta}^{(\text{old})}$ with a small step towards the direction of minimizing $E(\mathbf{X}_N; \boldsymbol{\Theta})$, where $e(\mathbf{x}_t; \boldsymbol{\Theta})$ is explicitly given as

$$e(\mathbf{x}_t; \boldsymbol{\Theta}) = (\mathbf{I}_t^a - \mathbf{h}_t^a)^T (\mathbf{I}_t^a - \mathbf{h}_t^a) \tag{18}$$

with

$$\mathbf{I}_t^a = [I^a(1|\mathbf{x}_t), I^a(2|\mathbf{x}_t), \ldots, I^a(k|\mathbf{x}_t)]^T \tag{19}$$

$$\mathbf{h}_t^a = [h^a(1|\mathbf{x}_t), h^a(2|\mathbf{x}_t), \ldots, h^a(k|\mathbf{x}_t)]^T \tag{20}$$

$$I^a(j|\mathbf{x}_t) = \begin{cases} 1, \text{ if } j = \arg\max_{1 \leq r \leq k} h^a(r|\mathbf{x}_t); \\ 0, \text{ otherwise.} \end{cases} \tag{21}$$

It can be shown that $\mathbf{I}_t^a$ must be equal to $\mathbf{I}_t$. Since $\boldsymbol{\Theta}^a$ is fixed, it implies that $h^a(c|\mathbf{x}_t)$ is a constant in this step. Hence, we only need to adjust those remaining parameters in $\boldsymbol{\Theta}$ except for β_c and $\boldsymbol{\theta}_c$, denoted as $\tilde{\boldsymbol{\Theta}}$ hereafter. Furthermore, we notice that there is a summation constraint on α_js as shown in Eq.(5), the updating of β_c only in Eq.(14) is, in effect, to automatically update those α_js with $j \neq c$ with a small step towards the direction of minimizing $E(\mathbf{X}_N; \boldsymbol{\Theta})$. Hence, at this step, we need not update those α_js with $j \neq c$. Subsequently, we have, for $\forall 1 \leq j \leq k$ with $j \neq c$,

$$\boldsymbol{\theta}_j^{(\text{new})} = \boldsymbol{\theta}_j^{(\text{old})} - \eta_2 \frac{\partial e(\mathbf{x}_t; \boldsymbol{\Theta}^a)}{\partial \boldsymbol{\theta}_j}|_{\tilde{\boldsymbol{\Theta}}^{(\text{old})}} \tag{22}$$

$$= \boldsymbol{\theta}_j^{(\text{old})} - \eta_2 \frac{h^a(j|\mathbf{x}_t)}{p(\mathbf{x}_t; \boldsymbol{\Theta}^a)} \frac{\partial}{\partial \boldsymbol{\theta}_j} [\alpha_j p(\mathbf{x}_t; \boldsymbol{\theta}_j)]|_{\tilde{\boldsymbol{\Theta}}^{(\text{old})}}, \tag{23}$$

where $\eta_2 = \eta_1 \lambda_m$ is the learning step in **Step 4**, and $\mathbf{I}_t^a$ must be equal to $\mathbf{I}_t$ for each time step t. In Eq.(23), extra computing is required to calculate $p(\mathbf{x}_t; \boldsymbol{\Theta}^a)$ and $h^a(j|\mathbf{x}_t)$s with $j \neq c$. For simplicity, we hereafter further approximate $p(\mathbf{x}_t; \boldsymbol{\Theta}^a)$ by $p(\mathbf{x}_t; \boldsymbol{\Theta}^{(\text{old})})$), those $h^a(j|\mathbf{x}_t)$s with $j \neq c$ then become $h(j|\mathbf{x}_t)$s. That is, Eq.(23) is approximated by

$$\boldsymbol{\theta}_j^{(\text{new})} = \boldsymbol{\theta}_j^{(\text{old})} - \eta_2 \frac{h(j|\mathbf{x}_t)}{p(\mathbf{x}_t; \boldsymbol{\Theta}^{(\text{old})})} \frac{\partial}{\partial \boldsymbol{\theta}_j} [\alpha_j p(\mathbf{x}_t; \boldsymbol{\theta}_j)]|_{\tilde{\boldsymbol{\Theta}}^{(\text{old})}} \tag{24}$$

for $\forall 1 \leq j \leq k$ with $j \neq c$.

Step 5 **Step 1** – **Step 4** are repeated for each input until $\boldsymbol{\Theta}$ converges.

If we further let each mixture component $p(\mathbf{x};\boldsymbol{\theta}_j)$ be a Gaussian density, written as $G(\mathbf{x};\mathbf{m}_j,\boldsymbol{\Sigma}_j)$, where $\mathbf{m}_j$ and $\boldsymbol{\Sigma}_j$ are the mean and covariance matrix of $\mathbf{x}$, respectively, the previous **Step 3** and **Step 4** can then explicitly become

Step 3 Update $\boldsymbol{\theta}_c$ with $I(c|\mathbf{x}_t)=1$ only by

$$\beta_c^{(\text{new})} = \beta_c^{(\text{old})} + \eta_1(1-\alpha_c^{(\text{old})}) \tag{25}$$

$$\mathbf{m}_c^{(\text{new})} = \mathbf{m}_c^{(\text{old})} + \eta_1 \boldsymbol{\Sigma}_c^{-1^{(\text{old})}}(\mathbf{x}_t - \mathbf{m}_c^{(\text{old})}) \tag{26}$$

$$\boldsymbol{\Sigma}_c^{-1^{(\text{new})}} = (1+\eta_1)\boldsymbol{\Sigma}_c^{-1^{(\text{old})}} - \eta_1 \mathbf{U}_{t,c} \tag{27}$$

with $\mathbf{U}_{t,c} = [\boldsymbol{\Sigma}_c^{-1^{(\text{old})}}(\mathbf{x}_t - \mathbf{m}_c^{(\text{old})})(\mathbf{x}_t - \mathbf{m}_c^{(\text{old})})^T \boldsymbol{\Sigma}_c^{-1^{(\text{old})}}]$.

Step 4 Update those $\boldsymbol{\theta}_j$s with $j \neq c$ (we call them as *rivals* hereafter). That is,

$$\mathbf{m}_j^{(\text{new})} = \mathbf{m}_j^{(\text{old})} - \eta_2 h^2(j|\mathbf{x}_t)\boldsymbol{\Sigma}_j^{-1^{(\text{old})}}(\mathbf{x}_t - \mathbf{m}_j^{(\text{old})}) \tag{28}$$

$$\boldsymbol{\Sigma}_j^{-1^{(\text{new})}} = [1-\eta_2 h^2(j|\mathbf{x}_t)]\boldsymbol{\Sigma}_j^{-1^{(\text{old})}} + \eta_2 h^2(j|\mathbf{x}_t)\mathbf{U}_{t,j}. \tag{29}$$

If we fix η_2 at zero and initialize the seed points $\mathbf{m}_j$s such that each true cluster contains at least a seed point, it can be seen that the EMM actually degenerates to the k^*-means algorithm [2]. Otherwise, the EMM not only updates the winner of the component parameters β_c and $\boldsymbol{\theta}_c$ in **Step 3**, but also penalizes all rivals with a de-learning rate η_2 in **Step 4**. In consistence with the updating equations of $\mathbf{m}_c$ and $\boldsymbol{\Sigma}_c$ in Eq.(26) and Eq.(27) respectively, we therefore name $\eta_{j,t} = \eta_2 h^2(j|\mathbf{x}_t)$ with $j \neq c$ as the penalization force rate of the rival with the subscript j. Compared to the exiting RPCL (Type A), the EMM algorithm extends it with the two generalizations:

1. At each time step, the EMM penalizes all rivals rather than the nearest rival of the winner in the RPCL (Type A).
2. The penalization force rate in EMM is dynamically changed, while the RPCL (Type A) set the so-called *de-learning rate* constantly.

Hence, in analog with the RPCL (Type A), the EMM can automatically determine the correct cluster number so long as k in **Step 1** is not less than k^*. If we further simply penalize the nearest rival **only** each time in the same way as the RPCL and its Type A variant, and always fix its penalization force rate at a constant, written as $\bar{\eta}_\tau$, the EMM then degenerates to the RPCL (Type A). Furthermore, during the clustering, if we further fix all $\alpha_j = \frac{1}{k}$, and $\boldsymbol{\Sigma}_j = \mathbf{I}$ for $\forall 1 \leq j \leq k$, where $\mathbf{I}$ is the identity matrix, the learning rule of the seed points $\mathbf{m}_j$s in the EMM is then equivalent to that of the RPCL [9]. That is, either of RPCL or RPCL (Type A) is a special case of the EMM.

3 Experimental Demonstration

To show the performance of the EMM in comparison with the EM, we randomly generated the data points from a mixture of three bivariate Gaussian distributions:

$$p(\mathbf{x}) = 0.3G[\mathbf{x}|\begin{pmatrix}1\\1\end{pmatrix}, \begin{pmatrix}0.15, 0.05\\0.05, 0.25\end{pmatrix}] + 0.4G[\mathbf{x}|\begin{pmatrix}1.0\\2.5\end{pmatrix}, \begin{pmatrix}0.15, 0.0\\0.0, \ \ 0.14\end{pmatrix}]$$
$$+0.3G[\mathbf{x}|\begin{pmatrix}2.5\\2.5\end{pmatrix}, \begin{pmatrix}0.15, -0.1\\-0.1, \ \ 0.15\end{pmatrix}] \quad (30)$$

with the sample size $1,000$. We used six seed points, i.e., $k > k^* = 3$, and randomly set the learning rate $\eta_1 = \eta_2 = 0.001$.

Fig. 1 shows the parameter learning curves of the EMM and the EM, respectively. It can be seen that the EM learning speed is much slower than the proposed EMM. Actually, the parameters learned by the EMM have converged after 40 epoches, but the EM not despite 800 epoches. In the EMM, one snapshot at Epoch 40 found that $\{\alpha_i, \boldsymbol{\theta}_i\}$ with $i = 1, 2, 4$ all converged to the correct values, meanwhile $\alpha_3 = \alpha_5 = \alpha_6 = 0$. From the winning rule of Eq.(6), we know that the densities of $3, 5, 6$ have become dead because they have no chance any more to win in the competition learning process. In other words, the data set is recognized to be a mixture of the three densities: $1, 2, 4$. Hence, the EMM has the robust performance without knowing the exact cluster number. In contrast, it was found that the EM leaded six densities to compete each other without making extra densities die. Subsequently, all the parameters were converged to the wrong positions. That is, the EM cannot work in this case.

4 Conclusion

This paper have proposed a general *Expectation-MiniMax* learning approach that not only includes the RPCL and its Type A form with the new variants developed, but also shows that such a penalized learning actually provides an alternative way to optimize the EM contrast function. Compared to the EM, the EMM converges much faster with the robust performance in clustering analysis without knowing the cluster number. We have presented the general learning procedures of the EMM, and successfully demonstrated its superior performance in comparison with the EM.

References

1. H. Akaike, “A New Look at the Statistical Model Identfication”, *IEEE Transactions on Automatic Control AC-19*, pp. 716–723, 1974.
2. Y.M. Cheung, “k^*-means — A Generalized k-means Clustering Algorithm with Unknown Cluster Number”, *Proceedings of Third International Conference on Intelligent Data Engineering and Automated Learning* (IDEAL’02), pp. 307-317, 2002.
3. A.P. Dempster, N.M. Laird and D.B. Rubin, “Maximum Likelihood from Incomplete Data via The EM Algorithm”, *Journal of Royal Statistical Society*, Vol. 39, pp. 1–38, 1977.
4. J.B. MacQueen, “Some Methods for Classification and Analysis of Multivariate Observations”, *Proceedings of* 5^{nd} *Berkeley Symposium on Mathematical Statistics and Probability*, 1, Berkeley, University of California Press, pp. 281–297, 1967.

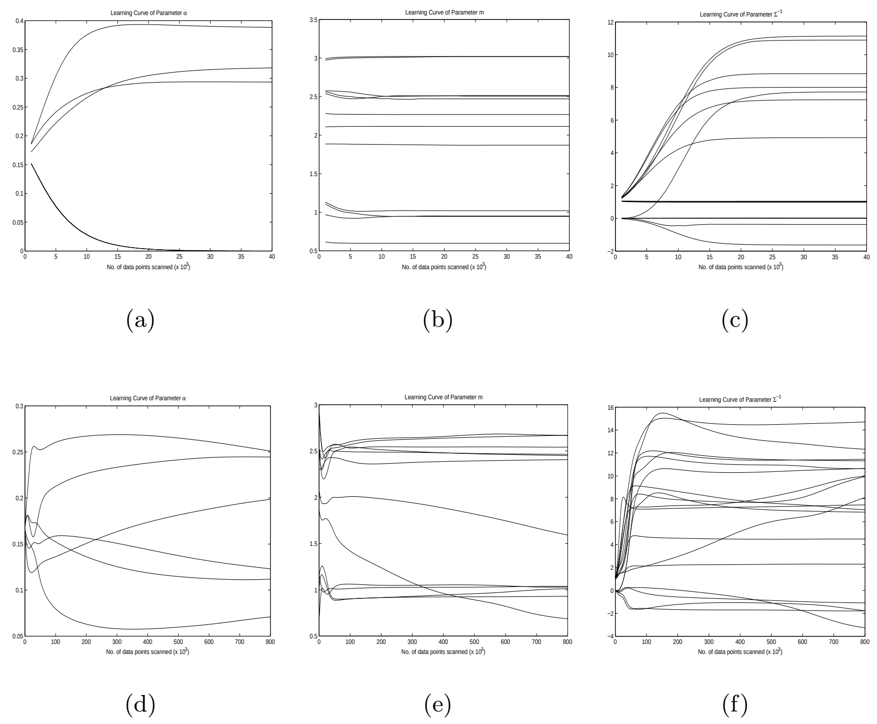

Fig. 1. Sub-figures (a)-(c) show the learning curves of α_js, $\mathbf{m}_j$s and $\boldsymbol{\Sigma}_j^{-1}$s by the EMM, while Sub-figures (d)-(f) show their learning curves by the EM.

5. G.J. McLachlan and K.E. Basford, "Mixture Models: Inference and Application to Clustering", Dekker, 1988.
6. G. Schwarz, "Estimating the Dimension of a Model", *The Annals of Statistics*, Vol. 6, No. 2, pp. 461–464, 1978.
7. L. Xu, "A Unified Learning Scheme: Bayesian-Kullback Ying-Yang Machine", *Advances in Neural Information Processing Systems*, Vol. 8, pp.444-450, 1996.
8. L. Xu, "Bayesian Ying-Yang Machine, Clustering and Number of Clusters", *Pattern Recognition Letters*, Vol. 18, No. 11-13, pp. 1167–1178, 1997.
9. L. Xu, A. Krzyżak and E. Oja, "Rival Penalized Competitive Learning for Clustering Analysis, RBF Net, and Curve Detection", *IEEE Transaction on Neural Networks*, Vol. 4, pp. 636–648, 1993.