Applied Informatics
a SpringerOpen Journal

RESEARCH Open Access

Projection-embedded BYY learning algorithm for Gaussian mixture-based clustering

Guangyong Chen[1], Pheng-Ann Heng[1] and Lei Xu[1,2*]

*Correspondence:
lxu@cse.cuhk.edu.hk
[1] Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, NT, Hong Kong, China
[2] Department of Computer Science and Engineering, Shanghai Jiao Tong University, SEIEE Building 3, 800 Dongchuan Road, Minhang District, 200240 Shanghai, China

Abstract

On learning the Gaussian mixture model, existing BYY learning algorithms are featured by a gradient-based line search with an appropriate stepsize. Learning becomes either unstable if the stepsize is too large or slow and gets stuck in a local optimal solution if the stepsize is too small. An algorithm without a learning stepsize has been proposed with expectation-maximization (EM) like two alternative steps. However, its learning process may still be unstable. This paper tackles this problem of unreliability by a modified algorithm called projection-embedded Bayesian Ying-Yang learning algorithm (pBYY). Experiments have shown that pBYY outperforms learning algorithms developed from not only minimum message length with Jeffreys prior (MML-Jef) and Variational Bayesian with Dirichlet-Normal-Wishart (VB-DNW) prior but also BYY with these priors (BYY-Jef and BYY-DNW). pBYY obtains the superiority with an easy implementation, while DNW prior-based learning algorithms suffer a complicated and tedious computation load. The performance of pBYY has also been demonstrated on the Berkeley Segmentation Dataset for the topic of unsupervised image segmentation. The resulted performances of semantic image segmentation have shown that pBYY outperforms not only MML-Jef, VB-DNW, BYY-Jef, and BYY-DNW but also three leading image segmentation algorithms, namely gPb-owt-ucm, MN-Cut, and mean shift.

Keywords: Gaussian mixture model; Model selection; BYY harmony learning; Nearest projection; Minimum description length; Variational Bayes; Unsupervised image segmentation

Background

Introduction

Gaussian mixture model (GMM) has been widely used in different areas, e.g., clustering, image segmentation (Zhang et al. 2001), speaker identification (Reynolds 1995), document classification (Nigam et al. 2000), market analysis (Chiu and Xu 2001), etc. Learning a GMM consists of parameter learning for estimating all unknown parameters and model selection for determining the number of Gaussian components k. Parameter learning is usually implemented under the maximum likelihood principle by an expectation-maximization (EM) algorithm (Redner and Walker 1984). A conventional model selection approach is featured by a two-stage implementation, which suffers from a huge computation because it requires parameter learning for each candidate GMM. Moreover, parameter learning will become less reliable as k becomes larger, which implies more free parameters.

One road to tackle these problems is referred as automatic model selection that automatically determines k during parameter learning. An early effort is rival penalized competitive learning (RPCL) (Xu et al. 1992; Xu 1998) with the number k automatically determined during learning. Automatic model selection may also be approached via appropriate priors on unknown parameters by Bayesian approaches. Two examples are minimum message length (MML) (Figueiredo and Jain 2002) and variational Bayesian (VB) (Corduneanu and Bishop 2001). Firstly proposed in (Xu 1995) and systematically developed in the past two decades, Bayesian Ying-Yang (BYY) learning provides not only a new model selection criteria but also a family of learning algorithms that is capable of automatic model selection during parameter learning, with details referred to recent tutorial and survey by (Xu 2010, 2012).

A systematic comparison has been recently made by (Shi et al. 2011) among MML, VB, and BYY with two types of priors. One is the Jeffreys prior and another is a parametric conjugate prior that imposes a Dirichlet prior on mixing weights and a joint normal-Wishart prior on mean vectors and covariance matrices, shortly denoted as DNW. Automatic model selection performances of these approaches are evaluated through extensive experiments, with several interesting empirical findings. Among them, it has been shown that BYY considerably outperforms both VB and MML. Different from VB and MML that rely on appropriate priors to perform model selection, BYY is capable of selecting model automatically even without imposing any priors on parameters, while its performance can be further improved with appropriate priors incorporated. Similar findings have also been obtained (Zhu et al. 2013), where a simplified BYY learning algorithm with DNW priors is shown to outperform or at least be competitive to the existing state-of-the-art image segmentation methods.

The algorithms by (Shi et al. 2011) for implementing BYY are featured by a gradient-based line search with an appropriate stepsize. Learning becomes either unstable if this stepsize is too large or slow and gets stuck in a local optimal solution if the stepsize is too small. Given in Algorithm two (Xu 2009) and Equation (11) (Xu 2010), there is a Ying-Yang two-step alternation algorithm that is similar to the EM algorithm without a learning stepsize for the learning procedure. However, the Ying step (Xu 2010) ignores the constrain that the covariance matrix of each Gaussian component must be positive definite matrix, so the learning procedure may become unstable.

To constrain the covariance matrix as a positive definite matrix, this paper proposes a projection operation into the Yang step, which results in a modified algorithm called projection-embedded BYY learning algorithm or shortly denoted as pBYY. To facilitate its implementation, we also add a Kullback Leibler divergence-based indicator into the algorithm to improve the detection of redundant Gaussian components. Experiments have shown that pBYY significantly outperforms not only the Jeffreys-based MML (Figueiredo and Jain 2002) and the DNW-based VB but also the BYY learning algorithms with these two types of priors (Shi et al. 2011), and it further avoids the cost of complicated and tedious computation brought by the DNW prior.

Gaussian mixture model and four learning principles

GMM assumes that an observation $x \in R^d$ is drawn from the following mixture of k Gaussian distributions:

$$\begin{aligned} q(x|\theta) = \textstyle\sum_{i=1}^{k} \alpha_i G\left(x|\mu_i, \Sigma_i\right), \\ \theta = \{\alpha, \mu_i, \Sigma_i\}_{i=1}^{k}, \ \alpha_i \geq 0, \ \textstyle\sum_{i=1}^{k} \alpha_i = 1, \end{aligned} \tag{1}$$

where $G(x|\mu, \Sigma)$ denotes a Gaussian density with a mean μ and a covariance matrix Σ.

GMM can be also regarded as a latent variable model by introducing a binary latent vector $y = \left[y_1, y_2, \ldots, y_k\right]^T$, subject to $y_i \in \{0, 1\}, \forall i$, and $\sum_{i=1}^{k} y_i = 1$, the latent variable $y_i = 1$ means that the random variable x is drawn from ith Gaussian component. The generative process of an observation x is interpreted as that y is sampled from a multinomial distribution with probabilities α and then x is randomly generated by the ith Gaussian component with $y_i = 1$. Let $X \in R^{d\times n}$ denote the set of n i.i.d. d-dimension observation samples, $Y \in R^{k\times n}$ denote the set of latent vectors for the observable set X, we have the following:

$$\begin{aligned} q(X, Y|\theta) &= q(X|Y, \theta) q(Y|\theta), \\ q(X|Y, \theta) &= \prod_{t=1}^{n} \prod_{i=1}^{k} G(x_t|\mu_i, \Sigma_i)^{y_{it}}, \\ q(Y|\theta) &= \prod_{t=1}^{n} \prod_{i=1}^{k} \alpha_i^{y_{it}}. \end{aligned} \tag{2}$$

Learning a GMM consists of parameter learning for estimating all the unknown parameters in θ and model selection for determining the number of Gaussian components k, which can be implemented differently under different learning principles.

The most widely used principle is called the maximum likelihood (ML), that is, we estimate θ by

$$\begin{aligned} \max_{\theta} \quad & q(X|\theta), \\ q(X|\theta) \quad = \quad & \sum_{Y} q(X|Y, \theta) q(Y|\theta) = \prod_{t=1}^{n} q(x_t|\theta). \end{aligned} \tag{3}$$

The ML learning with a known k is typically made by the well known EM algorithm (Redner and Walker 1984). However, an unknown k is poorly estimated by Equation (3) when the sample number n is not large enough. The task of determining an appropriate k is called model selection, which is usually made in a two-stage implementation with the help of a model selection criterion. However, such a two-stage implementation suffers from a huge computation and an unreliable estimation. The problems are tackled by automatic model selection that automatically determines k during learning θ without such a two-stage implementation.

There are three Bayesian related learning principles that can be implemented with such the property of automatic model selection.

One is called minimum message length (MML) (Wallace and Dowe 1999), which is actually an information theoretic restatement of Occam's Razor. The MML was introduced to learn GMM with the property of automatic model selection (Figueiredo and Jain 2002). Learning is made by the following maximization:

$$\begin{aligned} \max_{\theta} \quad & J_{MML}(X|\theta), \\ J_{MML}(X|\theta) \quad = \quad & \ln q(X|\theta) + \ln q(\theta) - \frac{1}{2} \ln |\mathbf{I}(\theta)|, \end{aligned} \tag{4}$$

where $|\mathbf{I}(\theta)|$ represents the determinant of Fisher information matrix with respect to (w.r.t) Θ. Equation (4) is mathematically equivalent to a maximum a posteriori (MAP) approach with modifying a proper prior $q(\theta)$ into being proportional to $q(\theta)/|\mathbf{I}(\theta)|^{1/2}$.

Using the Jeffreys prior $q(\theta) \propto |\mathbf{I}(\theta)|^{1/2}$ directly, Equation (4) degenerates to be ML learning principle. To avoid this situation, Figueiredo and Jain (2002) considered the following:

$$\ln \frac{q(\theta)}{|\mathbf{I}(\theta)|^{1/2}} \approx -\frac{\rho}{2}\sum_{i=1}^{k} \ln \alpha_i - \frac{k(\rho+1)}{2} \ln N, \tag{5}$$

where $\rho = d+0.5d(d+1)$ is the number of free parameters in each Gaussian component. In (Shi et al. 2011), it has shown that some improvement can be obtained by an algorithm that implements the MML principle with the help of a Dirichlet prior and a *joint* normal-Wishart prior (shortly DNW prior).

The other Bayesian related learning principle is called variational Bayesian (Corduneanu and Bishop 2001). The naive Bayes considers $q(X|\theta)q(\theta)$ with a prior $q(\theta)$ which takes a strong role. Unfortunately, a poor $q(\theta)$ may affect the learning performance seriously. Such a bad influence can be smoothed out by considering the following marginal distribution:

$$q(X) = \int q(X|\theta)q(\theta)d\theta. \tag{6}$$

However, it is difficult in computation with integral. The VB tackles this difficulty via constructing a lower bound J_{VB} with the help of Jensen's inequality as follows:

$$\begin{gathered} \max J_{VB}, \\ J_{VB} = \int p(\theta, Y|X) \ln\left[\tfrac{q(X,Y|\theta)q(\theta)}{p(\theta,Y|X)}\right] dY\, d\theta. \\ \ln q(X) \geq J_{VB}. \end{gathered} \tag{7}$$

The goal is to choose a suitable posterior distribution $p(\theta, Y|X)$ from a distribution family $\mathcal{P}$, so that the lower bound J_{VB} can readily be evaluated and yet sufficiently flexible. One challenge is to provide a suitable distribution family $\mathcal{P}$. In (Corduneanu and Bishop 2001), the family of prior distribution $\mathcal{P}$ can be approximately factorized as follows:

$$p(\theta, Y|X) = p(Y|X)\prod_i p(\theta_i|X). \tag{8}$$

With $q(X, Y|\theta)$ by Equation (2) and a DNW prior $q(\theta)$, the above $p(\theta_i|X)$ can be obtained with $p(Y|X)$ and $p(\theta_j|X)$ $\forall j \neq i$ given by the following equation (Bishop and Nasrabadi 2006):

$$p(\theta_i|X) = \frac{\int_{j\neq i} p(Y|X)p(\theta_j|X) \ln q(X, Y, \theta)d\theta dY}{\int_{j\neq i} p(Y|X)p(\theta_j|X) \ln q(X, Y, \theta)d\theta dY\, dX}.$$

A tight bound is unavailable to be obtained by Equation (8), which affects the learning performances. Also, DNW is quite tedious and has hyperparameters $\left\{\lambda, \xi, m_i, \frac{\Sigma_i^{-1}}{\beta}, \Phi, \gamma\right\}$ to be updated, which is time-consuming and may fall into local optimal. To avoid the tedious computation of the DNW prior-based VB, an algorithm for implementing VB principle is developed (Shi et al. 2011) with the help of the Jeffreys prior via approximately using a block-diagonal complete data Fisher information (Figueiredo and Jain 2002).

The last Bayesian related principle is BYY harmony learning. Firstly proposed by (Xu 1995) and systematically developed in the past two decades, BYY harmony learning on typical structures leads to new model selection criteria, new techniques for implementing learning regularization, and a class of algorithms that approach automatic model selection during parameter learning. Readers are referred to (Xu 2010, 2012, 2014) for latest systematical introductions about BYY harmony learning.

Briefly, a BYY system consists of Yang machine and Ying machine, corresponding to two types of decomposition, namely, Yang $p(R|X)p(X)$ and Ying $q(X|R)q(R)$ respectively, where the data X is regarded to be generated from its inner representation $R = \{Y, \theta\}$ that consists of latent variables Y and parameters θ, supported by a hyperparameter set Ξ. The harmony measure is mathematically expressed as follows:

$$H\left(p||q\right) = \int p\left(R|X\right)p(X)\ln\left[q(X|R)q(R)\right]dXdR. \tag{9}$$

Maximizing this $H(p||q)$ leads to not only a best matching between the Ying-Yang pair but also a compact model with a least complexity. Such an ability can be observed from several perspectives (see Section 4 in (Xu 2010)).

Applied to GMM by Equation (2), we have $R = \{Y, \theta\}$ and $q(R) = q(Y|\theta)q(\theta|\Xi)$. Comparing Equation (9) and Equation (7), the key difference is that there is only $q(X, Y|\theta)q(\theta)$ inside the basket $\ln[*]$ for the BYY harmony learning while there is also a denominator $p(\theta, Y|X)$ for the VB learning. Maximizing J_{VB} leads to a best match between $q(X, Y|\theta)q(\theta)$ and $p(\theta, Y|X)$, while maximizing $H(p||q)$ leads to not only such a best match but also a modeling of $q(X, Y|\theta)q(\theta)$ in a least complexity. Readers are referred to Section 4 and its figure five in (Xu 2012) for various aspects of this key difference, as well as how they relate and differ from MML and minimum description length (MDL) (Barron et al. 1998; Rissanen 1978).

Maximizing $H(p||q)$ leads to specific algorithms according to not only what types of $q(\theta|\Xi)$ are chosen for the Ying machine but also how the structure of $p(\theta, Y|X)$ is designed for the Yang machine. Details are referred to Section 4.2 in (Xu 2010) and Section 3.2 in (Xu 2012). For the GMM by Equation (2), we introduce two typical examples here.

One example is $p(\theta, Y|X)$ given by Equation (8) together with a DNW prior. Putting them into Equation (9), the DNW prior-based BYY harmony learning algorithm has been developed for maximizing $H(p||q)$ in (Shi et al. 2011). Extensive experiments have shown that the DNW prior-based BYY considerably outperforms both VB and MML for any type of priors and with whether or not hyper-parameters optimized. As the hyper-parameters of DNW prior are optimized by its corresponding learning principle, BYY further improves its performance and outperforms the others significantly, because learning hyper-parameters is a part of the entire BYY harmony learning. However, both VB and MML deteriorate when there are too many free hyper-parameters, especially the performance of VB drops drastically. The reason is that VB and MML maximize the marginal likelihood via variational approximation and Laplace approximation, respectively, where maximizing the marginal likelihood with respect to a free priori $q(\theta|\Xi)$ makes it tend to the maximum likelihood.

Another example is the following structure:

$$\begin{aligned} p(\theta, Y|X) &= p(Y|X, \theta)p(\theta|X), \\ p(Y|X, \theta) &= \frac{q(X, Y|\theta)}{\int q(X, Y|\theta)dY}, \\ p(\theta|X) &\quad \text{is free of structure.} \end{aligned} \tag{10}$$

Maximizing $H(p||q, \Xi)$ with respect to $p(\theta|X)$ makes Equation (9) simplified into

$$\begin{aligned} \max_\theta H(\theta),\ H(\theta) = H_0(\theta) + \ln q(\theta), \\ H_0(\theta) = \textstyle\sum_{t=1}^n \sum_{i=1}^k p(i|x_t, \theta) \ln \left[G(x_t|\mu_i, \Sigma_i)\alpha_i \right], \\ p(i|x_t, \theta) = \tfrac{\alpha_i G(x_t|\mu_i, \Sigma_i)}{\sum_{i=1}^k \alpha_i G(x_t|\mu_i, \Sigma_i)}. \end{aligned} \tag{11}$$

Automatic model selection and two-step alternation

Given a known k, learning the unknown parameters θ on a GMM is usually implemented under the maximum likelihood principle by an EM algorithm (Redner and Walker 1984), which is one typical instance of Algorithm 1 featured by a two-step alternation. As remarked at the bottom of the table, we get the EM algorithm after simply removing the lines of **trimming** with

$$p_{it} = p\big(i|x_t, \theta^{new}\big),\ \eta_i = 0,\ \rho_i = 0, i = 1, \ldots, k, \tag{12}$$

where $p(i|x_t, \theta)$ is the Bayes posteriori probability as follows:

$$\begin{aligned} p(i|x_t, \theta) &= \frac{\alpha_i G\left(x_t|\mu_i, \Sigma_i\right)}{\sum_{i=1}^k \alpha_i^p G(x_t|\mu_i, \Sigma_i)}, \\ \theta &= \{\theta_i\}_{i=1}^k,\ \theta_i = \{\alpha_i, \mu_i, \Sigma_i\}. \end{aligned} \tag{13}$$

Generally, η_i, ρ_i come from a priori distribution that takes a regularization role. This role is shut off by simply setting them to zero. When $\eta_i = 0, \rho_i > 0$, the EM algorithm is extended to the smoothed EM algorithm that was firstly proposed in 1997 (Xu 2010). Also, we get the EM algorithm for naive Bayes with Jeffreys priori on α_i, Σ_i with

$$\eta_i = \frac{d + 0.5d(d+1) - 1}{2n}, \rho_i = \frac{d}{2n}. \tag{14}$$

Algorithm 1 Two-step alternation

```
Require: $X = \{x_1, x_2, \ldots, x_t, \ldots, x_n\}, \eta_i,\ \rho_i$
  initialize all $p_{i,t}$ simply by $1/k$.
  Repeat the following two steps until convergence reached
  Ying step:
  for $i = 1$ to $k$ do
      $n_i = \sum_{t=1}^n p_{i,t} + \eta_i,\ \alpha_i^{new} = \frac{n_i}{\sum_{i=1}^k n_i}$,
      $\mu_i^{new} = \frac{\sum_{t=1}^n p_{i,t} x_t}{n_i}$,
      $\Sigma_i^{new} = \frac{\sum_{i=1}^n p_{i,t}(x_t - \mu_i^{new})(x_t - \mu_i^{new})^T}{n_i} + \rho_i I.$
  end for
  trimming: for $i = 1, 2 \ldots, k$, discard the $i$th Gaussian
  component if $\Psi_j(\theta^{new}) \to 0$, and let $k = k - 1$
  Yang step:
  for $t = 1, \ldots, n$, allocate $x_t$ to the $i$th Gaussian component
  via a weight $p_{i,t}$ according to a learning principle.
```

Remark:

The algorithm is a modification of the one in table one of (Xu 2012) with semi-supervised BYY omitted.

An unknown k is poorly estimated via the ML learning by Equation (3), especially when the sample number n is not large enough. The task of determining an appropriate k is made by model selection, which is usually made in a two-stage implementation. The first stage enumerates k to get a set of candidate models $\mathcal{M}$ with unknown parameters of each candidate estimated by the EM algorithm. In the second stage, the best candidate is selected by a model selection criterion. Examples of such criteria include Akaike's information criterion (AIC) (Akaike 1974), Bayesian inference criterion (BIC), minimum description length (MDL) criterion (which stems from another viewpoint but coincides with BIC when it is simplified to an analytically computable criterion), etc (Barron et al. 1998; Rissanen 1978). However, this two-stage implementation suffers from a huge computation because it requires parameter learning for each $k \in \mathcal{M}$. Moreover, a larger k often implies more unknown parameters, thus parameter estimation becomes less reliable and the criterion evaluation reduces its accuracy (see Section 2.1 in (Xu 2010) for a detailed discussion).

One road to tackle the problems is referred to automatic model selection that means to automatically determine an appropriate k during parameter learning. An early effort is RPCL (Xu et al. 1992; Xu 1998). The key idea is that not only the winning Gaussian component moves a little bit to adapt the current sample but also the rival (i.e., the second winner) Gaussian component is repelled a little bit from this sample to reduce a duplicated information allocation. As a result, an extra Gaussian component is driven far away from data.

A batch learning version of RPCL learning may be also obtained as one instance of Algorithm 1, simply with

$$p_{\ell t} = \begin{cases} 1, & \ell^* = \arg\max_j p\left(j|x_t, \theta^{new}\right), \\ -\gamma, & \ell = \arg\max_{\ell \neq \ell^*} p\left(j|x_t, \theta^{new}\right), \\ 0, & \text{otherwise}, \end{cases} \tag{15}$$

by which learning is made on a cluster when $p_{\ell t} = 1$ and penalizing or de-learning is made on a cluster when $p_{\ell t} = -\gamma$. Usually, the penalizing strength is set $\gamma \approx 0.005 \sim 0.05$. When $\gamma = 0$, it degenerates to the so called hard-cut EM algorithm, see Equations (19) and (20) in (Xu 1995).

According to its general formulation (e.g., see the last part of Section 2.1 in (Xu 2010)), automatic model selection is a nature of learning a mixture of k individual substructures with the following two features:

- There is an indicator $\Psi_j(\theta)$ on θ or its subset, based on which a particular structural component j can be effectively discarded if its corresponding $\Psi_j(\theta) \to 0$. Taking the GMM as an example, we may consider

$$\Psi_j(\theta) = \alpha_j, \ or \ \Psi_j(\theta) = \alpha_j Tr\left[\Sigma_j\right]. \tag{16}$$

- With initial k large enough, there is an intrinsic mechanism that drives such an indicator $\Psi_j(\theta)$ towards zero if the corresponding structure is redundant and thus can be effectively discarded.

Three Bayesian-related approaches introduced in the previous subsection can all be implemented with such a nature of automatic model selection. For both MML and VB, this nature comes from an appropriate prior $q(\theta|\Xi)$. Favorably, BYY is capable of automatic model selection even without imposing any priors on the parameters, and its performance can be further improved as appropriate priors are incorporated. Actually, the BYY harmony learning by maximizing $H(p||q)$ bases on $q(R) = q(Y|\theta)q(\theta|\Xi)$ to make model selection, with $q(Y|\theta)$ in a role that is not only equally important to $q(\theta|\Xi)$ but also easy computing, while $q(\theta|\Xi)$ is still handled in a way similar to MML and VB.

The BYY harmony learning by Equation (11) can be implemented by Algorithm 1, with the Yang step given as follows:

$$\begin{aligned} p_{it} &= p_{it}(\theta^{new}), \\ p_{it}(\theta) &= p(i|x_t, \theta)\left[1 + \delta_{i,t}(\theta)\right], \\ \delta_{i,t}(\theta) &= \pi_t(\theta_i) - \sum_i p(i|x_t, \theta_i)\pi_t(\theta_i), \\ \pi_t(\theta_i) &= \ln\left[G(x_t|\mu_i, \Sigma_i)\alpha_i\right]. \end{aligned} \tag{17}$$

The algorithm implements a BYY harmony learning without a priori $\ln q(\theta)$ in Algorithm (1) by simply setting $\eta_i = 0, \rho_i = 0$ or a data smoothing based BYY harmony learning when $\eta_i = 0, \rho_i > 0$. Readers are referred to Section 3.1 of (Xu 2010) for further details. Also, we may implement the Jeffreys priori based on BYY harmony learning by using Equation (14), see table one in (Shi et al. 2011).

Methods

Learning unreliability and convex combination

As introduced in Section 3.1 of (Xu 2010), the existing algorithms for implementing BYY principle come from taking a gradient of $H(\theta)$ by Equation (11) w.r.t a subset ϕ of parameters. That is, we consider

$$\begin{aligned} \nabla_\phi H(\theta) &= \nabla_\phi H_0(\theta) + \nabla_\phi \ln q(\theta), \\ \nabla_\phi H_0(\theta) &= \textstyle\sum_{t=1}^n \sum_{i=1}^k p_{i,t}(\theta)\nabla_\phi \pi_t(\theta_i), \end{aligned} \tag{18}$$

with $p_{i,t}(\theta)$ and $\pi_t(\theta_i)$ given in Equation (17).

Based on this gradient, one attempt to update parameters is gradient-based local search. The parameter ϕ can be updated iteratively as below:

$$\phi^{new} = \phi^{old} + \eta\nabla_\phi H(\theta), \tag{19}$$

where $\eta > 0$ is a small learning stepsize. Both the BYY learning algorithm given in figure seven of (Xu 2010) and the BYY-Jef algorithm given in table one of (Shi et al. 2011) are derived from Equation (19) with the help of some computing tricks and simplification. However, the performance of such algorithms all depend on an appropriate stepsize. Learning becomes either unstable if η is too large or slow and gets stuck in a local optimal if η is too small. No such a learning stepsize is required for EM algorithms.

Another typical implementation attempts to make the BYY harmony learning by Equation (11) also in a Ying-Yang two-step alternation, as previously suggested in Section 2.1 and table one of (Xu 2012). This two-step alternation algorithm is actually derived from approximately letting $p_{i,t}(\theta)$ in Equation (18) to be fixed at its value $p_{it} = p_{it}(\theta^{new})$ such that we can solve the root of $\nabla_\phi H(\theta) = 0$ subject to this fixation to get the Ying step in Algorithm 1.

Still, there lacks theoretical analyses that either guarantee the learning convergence or provide the convergence conditions. Oppositely, we find empirically that the learning process of this BYY two-step alternation may become unstable.

Actually, the root of $\nabla_{\phi} H(\theta) = 0$ subject to $p_{it} = p_{it}(\theta^{new})$ can be considerably deviated from the true root of $\nabla_{\phi} H(\theta) = 0$ since this true root is coupled with $p_{it}(\theta)$ that varies with θ. Not only correctly solving the root of $\nabla_{\phi} H(\theta) = 0$ is a challenging task but also it is unclear whether fixing $p_{it} = p_{it}(\theta^{new})$ makes the learning procedure become unstable.

From the likelihood by Equation (3) and Equation (1), it can be observed that

$$\nabla_{\phi} \ln q(X|\theta) = \sum_{t=1}^{n} \sum_{i=1}^{k} p(i|x_t, \theta) \nabla_{\phi} \pi_t(\theta_i), \tag{20}$$

with $p(i|x_t, \theta)$ given in Equation (13). Fixing $p(i|x_t, \theta) = p_{it} = p(i|x_t, \theta^{new})$, solving the root of $\nabla_{\phi} \ln q(X|\theta) = 0$ leads to the Ying step in Algorithm 1, or precisely the M step of the EM algorithm while letting $p_{it} = p(i|x_t, \theta^{new})$ is just the E step of the EM algorithm. As well known, the convergence of the EM algroithm has been theoretically proved. That is, though the root of $\nabla_{\phi} \ln q(X|\theta) = 0$ is also coupled with $p(i|x_t, \theta)$ that varies with θ, this deviation actually does not affect the convergence.

The difference between $p_{it} = p(i|x_t, \theta) = p(i|x_t, \theta^{new})$ and $p_{it} = p_{it}(\theta^{new})$ is that $p(i|x_t, \theta), i = 1, \ldots, k$ remains to be probability with θ_i, while $p_{it}(\theta_i), i = 1, \ldots, k$ given in Equation (17) are no longer the probabilities and even take negative values sometimes. Thus $p_{it}(\theta^{new})$ is more sparse than $p(i|x_t, \theta^{new})$, and Yang step in the BYY theory introduces a nature of automatical model selection into the iteration procedure.

To further investigate the influence of replacing $p(i|x_t, \theta^{new})$ by $p_{it}(\theta^{new})$, we now focus on Ying step, which can be reformulated as below:

$$\begin{aligned}
\alpha_i &= \frac{\sum_{t=1}^{n} p_{it}}{N}, \\
\mu_i &= \sum_{t=1}^{N} \frac{p_{it}}{\sum_{t=1}^{n} p_{it}} x_t, \\
\Sigma_i &= \sum_{t=1}^{N} \frac{p_{it}}{\sum_{t=1}^{n} p_{it}} (x_t - \mu_i)(x_t - \mu_i)^T.
\end{aligned} \tag{21}$$

For EM algorithm, both μ_i and Σ_i are constrained in the convex hulls spanned by x_t and $(x_t - \mu_i)(x_t - \mu_i)^T$, respectively, because its p_{it} still remains in the probability space. However, in BYY algorithm, p_{it} is no longer the probabilities and even take negative values sometimes. Thus, μ_i and Σ_i may break through their corresponding convex hulls. For GMM, the model parameters θ must satisfy following constrains:

$$\begin{aligned}
\sum_{i=1}^{k} \alpha_i = 1, \alpha_i \geq 0, \quad \forall i \in \{1, 2, \ldots, k\}, \\
\Sigma_i \in \mathbb{R}_{+}^{d \times d} \quad \forall i \in \{1, 2, \ldots, k\},
\end{aligned} \tag{22}$$

where $\mathbb{R}_{+}^{d \times d}$ denotes the set of positive semidefinite matrix of size $d \times d$. Thus the updated α_i and Σ_i in BYY may no longer exist in their feasible regions sometimes. Instead of projecting α_i and Σ_i to the set of positive semidefinite matrix directly, we are motivated to project $\nabla_{\phi} H_0(\theta)$ back to the convex hull of local gradients $\nabla_{\phi} \pi_t(\theta_i)$, $t = 1, \ldots, n$, via projecting $p_{it}(\theta^{new})$ onto the following set of probabilities to preserve more information of α_i and Σ_i:

$$\mathcal{P} = \left\{ p_1, \ldots, p_k : p_i \geq 0, \sum_{i=1}^{k} p_i = 1 \right\}. \tag{23}$$

For updating each mean vector μ_i, we are encouraged to use $p_{it}(\theta^{new})$, because the updating equation of μ_i is no longer a convex combination of all observable samples, and the redundant components can be pushed outside the convex hull; thus, this operation accelerates the speed of model selection.

The relative structure among the original $\{p_{it}(\theta^{new})\}$ is encoded by the position of the vector $p_t^H = \left[p_{1t}(\theta^{new}), \ldots, p_{kt}(\theta^{new})\right]^T$ in R^k. Projecting p_t^H from R^k to $\mathcal{P}$ in Equation (23) means to find a vector $p_t = \left[p_{1t}, \ldots, p_{kt}\right]^T \in \mathcal{P}$ that is the nearest one to p_t^H and thus best keeps the relative structure within elements of p_t^H. To be specific, we choose the nearest one in a sense of the least square distance, that is, we consider the following optimization problem:

$$p_t^* = arg \min_{p \in \mathcal{P}} \left|\left| p - p_t^H \right|\right|^2. \tag{24}$$

The above implementation maybe regarded as a two-step approach of making the BYY harmony learning by Equation (17) under a principle of *multiple convex combination* preservation (Xu 2014).

Fast approximation and pBYY-Jef algorithm

The problem Equation (24) is often encountered in the literature of applied mathematics and scientific computing and tackled by several algorithms such as variants of the method of alternating projections (Bauschke and Borwein 1993) and variants of Dykstra's algorithm (Bauschke and Borwein 1994). However, these algorithms suffer from a huge computing cost, especially on a large-size data set.

Alternatively, we propose a fast approximation algorithm with two steps, motivated by the Kolmogorov's criterion (see Chapter of 1 (Escalante and Raydan 2011)). Let $\prod_{\mathcal{S}}(x)$ denote the projection point of an arbitrary point $x \in \mathbb{R}^k$ onto a non-empty closed convex set $\mathcal{S} \subset \mathbb{R}^n$; Kolmogorov's criterion states that $z^* = \prod_{\mathcal{S}}(x)$ if and only if $z^* \in \mathcal{S}$ and $(z - z^*)^T(x - z^*) \leq 0$ for all $z \in \mathcal{S}$, from which we can get the following:

Theorem 1. Let $\mathcal{F}_p = \{p_1, \ldots, p_k : \sum_{i=1}^{k} p_i = 1\}$ with $\mathcal{P} \subset \mathcal{F}_p$, we have $\prod_{\mathcal{P}}(x) = \prod_{\mathcal{P}} \prod_{\mathcal{F}_p}(x)$ for an arbitrary point $x \in \mathbb{R}^k$.

Proof. Let $z' = \prod_{\mathcal{F}_p}(x)$, $z^* = \prod_{\mathcal{P}}(x)$ and $z'^* = \prod_{\mathcal{P}}(z')$. From $(z - z'^*)^T(z' - z'^*) \leq 0$ for all $z \in \mathcal{P}$, we have $(z - z'^*)^T(z' - x + x - z'^*) \leq 0$ or $(z - z'^*)^T(z' - x) + (z - z'^*)^T(x - z'^*) \leq 0$. It follows $(z - z'^*)^T(z' - x) = 0$ since $z' = \prod_{\mathcal{F}_p}(x)$ is the projection point of x to the hyperplane $\mathcal{F}_p$ and thus orthogonal to the vector $z - z'^*$ that lies in this hyperplane $\mathcal{F}_p$. Therefore, we get the inequality $(z - z'^*)^T(x - z'^*) \leq 0$, which holds for all $z \in \mathcal{P}$ and thus $z^* = z'^*$ according to Kolmogorov's criterion. **End.** □

Based on this theorem, we split the projection into two steps. First, we consider the following orthogonal projection of p_t^H onto the hyperplane $\mathcal{F}_p$:

$$f_t = \left(I - \mathbf{n}\mathbf{n}^T\right)\left(p_t^H - f_0\right) + f_0, f_0 = \frac{1}{k}\mathbf{1}, \tag{25}$$

where $\mathbf{n} = \frac{1}{\sqrt{k}}\mathbf{1}$ is the normal vector of the hyperplane $\sum_{i=1}^{k} p_i = 1, f_0$ is the center point of the closed convex set $\mathcal{P}$, and all elements in $\mathbf{1} \in R^{k\times 1}$ are equal to 1.

Second, we further project f_t onto $\mathcal{P}$. However, accurately calculating the projecting point is still very time-consuming. Instead, we consider a fast approximation along the line between f_t and f_0 as follows:

$$p_t = \lambda f_0 + (1 - \lambda) f_t \tag{26}$$

with a minimum λ that make p_t locate within $\mathcal{P}$.

In a summary, we get a modified algorithm as one new instance of Algorithm 1. Its Ying step remains unchanged but its Yang step gets $\{p_{it}(\theta^{new})\}$ by Equation (17) and then makes the nearest projection onto $\mathcal{P}$ by Equation (25) and Equation (26). For clarity, we rewrite Algorithm 1 into a detailed form in **Algorithm**-2 that is dedicated to implementing this projection-embedded BYY learning (shortly named pBYY).

Algorithm 2 pBYY algorithm

```
Require: $X = \{x_1, x_2, \ldots, x_t, \ldots, x_n\}$, and thresholds $\epsilon_0, \epsilon_1$,
  $k$ is initially set a large enough value
  $\boldsymbol{n} = \frac{1}{\sqrt{k}}\mathbf{1}$, $\mathbf{1} = [1, \ldots, 1]^T$ , $f_0 = \frac{1}{k}\mathbf{1}$, and a large $T_b$,
  Initialization: set $T = 0$ and $p_t = \left[p_{1t}, \ldots, p_{kt}\right]^T = f_0$ for all $t$.
  Repeat the following two steps until convergence reached
  $T = T + 1$
  Ying step:
  for $i = 1$ to $k$ do
      $n_i = \sum_{t=1}^{n} p_{i,t}$, $\alpha_i^{new} = \frac{n_i}{\sum_{i=1}^{k} n_i}$, $\mu_i^{new} = \frac{\sum_{t=1}^{n} p_{i,t} x_t}{n_i}$,
      $\Sigma_i^{new} = \frac{\sum_{i=1}^{n} p_{i,t}(x_t - \mu_i^{new})(x_t - \mu_i^{new})^T}{n_i}$.
  end for
  trimming:
  for $i = 1$ to $k$ do
      if $\alpha_i^{new} Tr\left[\Sigma_i^{new}\right] < \epsilon_0$, then discard $i$, let $k = k - 1$
      if $T > T_b$ and $KL_{ij} < \epsilon_1$ $j \neq i$, then discard $i$, let $k = k - 1$
      go to Initialization
  end for
  Yang step:
  for $t = 1$ to $n$ do
      for $i = 1$ to $k$ do
                                  $\delta_{i,t}(\theta^{new}) =$
                  $\ln p(i|x_t, \theta^{new}) - \sum_j p\left(j|x_t, \theta^{new}\right) \ln p\left(j|x_t, \theta^{new}\right)$
          $p_{it}(\theta^{new}) = p(i|x_t, \theta^{new})\left[1 + \delta_{i,t}(\theta^{new})\right]$ where $p(i|x_t, \theta) = \frac{\alpha_i G(x_t|\mu_i, \Sigma_i)}{\sum_{i=1}^{k} \alpha_i G(x_t|\mu_i, \Sigma_i)}$.
      end for
      $p_t^H = \left[p_{1t}(\theta^{new}), \ldots, p_{kt}(\theta^{new})\right]^T$
      Projection:
      $f_t = (I - \mathbf{n}\mathbf{n}^T)\left(p_t^H - f_0\right) + f_0$, $p_t = \lambda f_0 + (1 - \lambda) f_t$.
      by a minimum $\lambda$ that makes each element of $p_t$ nonnegative.
  end for
  End repeat
```

The pBYY implementation repeats the Ying step and the Yang step alternatively. It gets out of the repeating circle in two cases. One is that learning is finally completed as the repeating circle converges with an unchanged k. The other is after trimming one Gaussian component with k reducing by 1, after which it goes to the line of **initialization** and start a new repeating circle. This re-initialization is helpful to avoid accumulation of estimating bias, though it requires extra computing costs. Whether we need this depending on a trade-off of computing cost versus estimating accuracy. We may remove this re-initialization by simply deleting the line 'go to **Initialization**'.

Trimming a Gaussian component bases on an indicator $\Psi_j(\theta)$ as given in Equation (16). Empirically, we find that there are scenarios and add the following new indicator for detection:

$$\begin{aligned} KL_{ij} &= \int G(x|\mu_i, \Sigma_i) \ln \frac{G(x|\mu_i, \Sigma_i)}{G(x|\mu_j, \Sigma_j)} dx \\ &= \frac{1}{2}\left[\ln \frac{|\Sigma_j|}{|\Sigma_i|} - d + \mathrm{Tr}(\Sigma_j^{-1}\Sigma_i) + d_{i,j}^M \right], \\ d_{i,j}^M &= (\mu_i - \mu_j)^T \Sigma_j^{-1} (\mu_i - \mu_j). \end{aligned} \tag{27}$$

That is, we use the Kullback–Leibler (KL) divergence to measure the similarity between two Gaussian components. When KL_{ij} becomes more close to 0 for any $j \neq i$, we may regard that the ith Gaussian component is redundant and thus discarded.

Results and discussion

Performance measures and algorithms

When samples locate in a space with its dimension less than 3, we can visualize and judge the clustering performance manually. However, samples are usually located in a high dimensional space for practical problems. Also, human evaluation is too subjective. In this paper, we consider four typical measures for clustering performance and model selection on number of clusters.

First, a traditional criterion to measure the performances of model selection could be named as the correct selection rate (CSR), namely how many times the algorithm gets the accurate number of clusters among a large number of trials. Sometimes, this criterion is argued to be too strict. For example, there exists four clusters in the set of observation samples. If an algorithm splits one cluster into two but gets the other three clusters correctly, this trial gets a zero count in computing CSR, though the clustering result still has some reasonable interpretation.

Second, one popular measure in the current literature is called variational information (VI), which evaluates the distance between one clustering result $\mathcal{C}'$ and the ground-truth $\mathcal{C}$ as follows:

$$\begin{aligned} \mathrm{VI}(\mathcal{C},\mathcal{C}') &= H(\mathcal{C}) + H(\mathcal{C}') - \mathrm{MI}(\mathcal{C},\mathcal{C}'), \\ H(\mathcal{C}) &= -\sum_{i=1}^{k} P(i) \log P(i), \\ \mathrm{MI}(\mathcal{C},\mathcal{C}') &= \sum_{i=1}^{k} \sum_{j=1}^{m} P(i,j) \log_2 \frac{P(i,j)}{P(i)P(j)}, \\ P(i) &= \frac{|C_i|}{N},\ P(i,j) = \frac{|C_i \cap C_j'|}{N}, \end{aligned} \tag{28}$$

with $|C_i|$ denoting the size of cluster C_i, where we get k clusters $\{C_i\}$ in clustering $\mathcal{C}$ and m clusters $\{C_j\}$ in clustering $\mathcal{C}'$. This MI denotes the mutual information that describes how much we can reduce the uncertainty about the cluster of a random sample when knowing its cluster in another clustering of the same set of observation samples (Wagner and Wagner 2007). The smaller the VI value is, the better the performance is.

The last popular measure is called probabilistic Rand index (PRI). It further considers to partition the set of all (unordered) pairs of observation samples in $\mathcal{S}$ into the disjoint union of the following sets:

$\mathcal{R}_{11}$ = {pairs that are in the same cluster under $\mathcal{C}$ and $\mathcal{C}'$}
$\mathcal{R}_{00}$ = {pairs that are in the different clusters under $\mathcal{C}$ and $\mathcal{C}'$}
$\mathcal{R}_{10}$ = {pairs that are in the same cluster under $\mathcal{C}$ but in different ones under $\mathcal{C}'$}
$\mathcal{R}_{01}$ = {pairs that are in the different clusters under $\mathcal{C}$ but in the same under $\mathcal{C}'$}.

Assume that each sample is randomly assigned to one cluster. The probability that two samples are in the same cluster in both partitions is $p_{11} = \frac{1}{k} \cdot \frac{1}{m}$. Corresponding to the $\mathcal{R}_{10}$, $\mathcal{R}_{01,}$ and $\mathcal{R}_{00}$, we get $p_{10} = \frac{1}{k} \cdot \left(1 - \frac{1}{m}\right)$, $p_{01} = \left(1 - \frac{1}{k}\right) \cdot \frac{1}{m}$, and $p_{00} = \left(1 - \frac{1}{k}\right) \cdot \left(1 - \frac{1}{m}\right)$. Then, PRI can be expressed as follows (Carpineto and Romano 2012):

$$\mathrm{PRI}(\mathcal{C}, \mathcal{C}') = \frac{w_{11}n_{11} + w_{00}n_{00}}{w_{11}n_{11} + w_{10}n_{10} + w_{01}n_{01} + w_{00}n_{00}}, \tag{29}$$

where $n_{ab} = |R_{ab}|$ and $w_{ab} = -\log_2 (p_{ab})$ for $a, b \in \{0, 1\}$. Simple analysis show that PRI vary between 0 (no agreement on any pair of samples in clusterings $\mathcal{C}$ and $\mathcal{C}'$) and 1 (when two clusterings are equal).

Moreover, one popular application of clustering algorithms is image segmentation. To evaluate the performances of semantic image segmentation, one widely used measure is the covering rate (CR) (Richardson and Green 1997), by whcih a larger CR value indicates a better performance.

We aim at comparisons of the proposed **Algorithm**-2 with those typical algorithms investigated in (Shi et al. 2011). For clarification, we summarize as follows:

BYY-Jef and BYY-DNW: both come from table one and table six in (Shi et al. 2011).
MML-Jef: this was taken from table two in (Shi et al. 2011), same as the one given in (Figueiredo and Jain 2002).
VB-DNW: this was taken from table six in (Shi et al. 2011), same as the one given in (Bishop and Nasrabadi 2006; Corduneanu and Bishop 2001).

All algorithms are programmed in MATLAB R2010b on a 32-bit PC with 3.1 GHz Intel Core i5-2400 CPU and 4 GB memory.

All data sets and source codes used in this paper can be downloaded from the website http://www.cse.cuhk.edu.hk/~gychen/pBYY.

Empirical comparison

We start at three types of synthetic data sets illustrated in Figure 1. Each type of data set is processed 500 independent trails with random initializations. In the algorithm implementations, the mean vector of each Gaussian component is initialized randomly, and the initial mixing weight and initial covariance matrix of each Gaussian component are computed with help of the k mean algorithm.

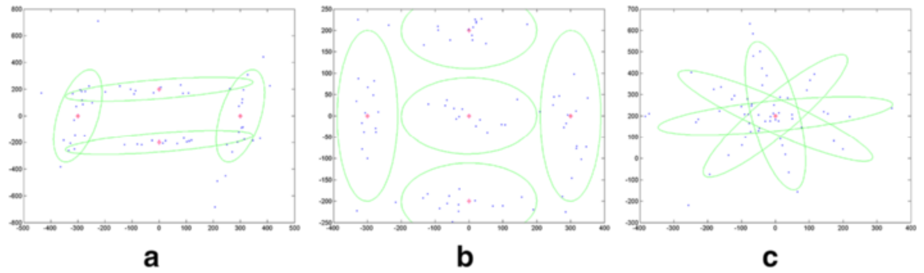

Figure 1 Three synthetic data sets and their ground-truth clusters. For both **(a)** and **(c)**, data sets of 60 samples are generated from a 2-dimensional 4-component GMM, with equal mixing weights of $\frac{1}{4}$; **(b)** 75 samples are generated from a 2-dimensional 5-component GMM, with equal mixing weight of $\frac{1}{5}$. (Each red curve indicates to a contour of equal probability density per component, and the read diamonds indicate the Gaussian means).

The comparisons of performance of each algorithm are shown in Table 1. We observe that **pBYY** significantly outperforms all the other algorithms almost in all the cases, without using any priori. The only exception occurs on the data set **GMM-b**, where BYY-DNW scored the best VI value though pBYY also got a value that is very close to the VI score. We also observe how the choice of an appropriate learning stepsize affects the performance of BYY-Jef and BYY-DNW. Closely related to the configurations of data sets, this choice is a difficult task. On the configuration type of **GMM-b** similar to the datasets studied in (Shi et al. 2011), experiments reconfirm the statement that BYY outperforms its counterparts of VB and MML (Shi et al. 2011). However, the statement seemly no longer holds for the configuration types of **GMM-a** and **GMM-c**, probably due to inappropriate learning stepsizes. Favorably, this statement has been reconfirmed by pBYY on the data sets of **GMM-a** and **GMM-c** with re-initialization period T_b being set as 5, namely, pBYY still significantly outperforms not only VB-DNW and MML-Jef but also BYY-Jef and BYY-DNW.

Table 2 presents a set of real-world data, where acidity, enzyme, and galaxy data sets come from (Richardson and Green 1997). On these data sets, it is difficult to use CSR, PRI, and VI because the information about the correct clustering result is unavailable. Following (Bishop and Nasrabadi 2006), we compare the performances of these algorithms on modeling the distributions of acidity, enzyme, and galaxy data sets visually. As demonstrated in Figure 2, BYY-Jef, BYY-DNW, and pBYY all obviously outperform VB-DNW and MML-Jef on the acidity and enzime, with pBYY performing best and

Table 1 Performance of each algorithm on three synthetic data sets after 500 trials, with the initial number of Gaussian components is set as $k = 20$, where 'a' indicates the best within its column

Data set	**GMM-a**			**GMM-b**			**GMM-c**		
Algorithms	**CSR**	**VI**	**PRI**	**CSR**	**VI**	**PRI**	**CSR**	**VI**	**PRI**
VB-DNW	0.4660	1.0243	0.7730	0.5160	0.6264	0.8599	0.1060	1.3337	0.6469
MML-Jef	0.1700	3.2637	0.7345	0.1600	4.8235	0.7573	0.4140	58.0039	0.6388
BYY-Jef	0.2167	1.1135	0.7006	0.5533	0.6650	0.8257	0.0100	1.6889	0.4732
BYY-DNW	0.1433	1.1947	0.7039	0.0700	0.5373^a	0.8760	0	1.7948	0.4622
pBYY	0.7260^a	0.5852^a	0.8692^a	0.8840^a	0.5482	0.8779^a	0.6100^a	1.1328^a	0.7451^a

For a good performance, we expect that the values of CSR and PRI are big and that the VI value is small.

Table 2 Details of 1D real data sets

Data set	Instances	Input feature
Acidity	155	1
Enzyme	245	1
Galaxy	82	1

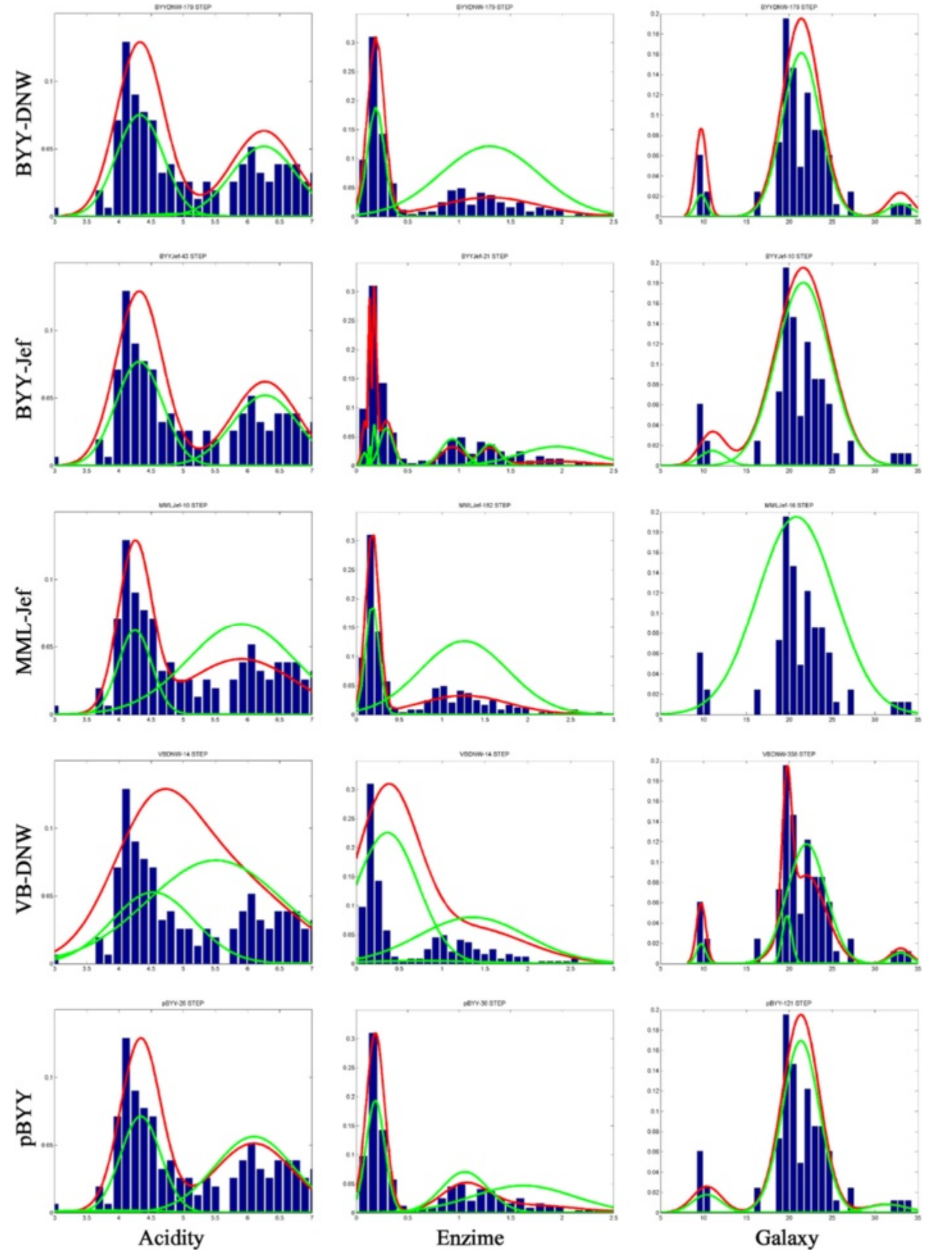

Figure 2 Density fitting on 1D real-world data sets. From left to right: acidity, enzyme, and galaxy data sets; while from top to bottom: BYY-DNW, BYY-Jef, MML-Jef, VB-DNW, and pBYY algorithms. (Red curve indicates the overall density function and green curve the density function per component).

MML-Jef outperforming. VB-DNW. On the galaxy data set, pBYY and BYY-DNW perform similarly and both outperform BYY-Jef, VB-DNW, and MML-Jef. In summary, these experiments confirm the previous findings obtained on synthetic data sets. In other words, pBYY outperforms not only VB and MML but also BYY-Jef and BYY-DNW.

To further evaluate the performance of pBYY algorithm, we apply the proposed algorithm to unsupervised image segmentation on 100 testing images from Berkeley Segmentation Data Set (BSDS) www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html, where each image has five ground-truth segmentations hand-drawn by persons, as illustrated in Figure 3. For a clustering-based image segmentation algorithm, an important issue is how to get features as input vectors. In this paper, we use the features proposed by Varma and Zisserman (2003), which has been used with promising image segmentation results (Nikou et al. 2010; Shi et al. 2011; Zhu et al. 2013). To concentrate on the performance of clustering algorithms, we do not conduct post-processing operations, such as region merging and graph cut, although they may further improve the segmentation results.

We compare the performance of pBYY algorithm with several leading segmentation algorithms, including gPb-owt-ucm (Arbelaez et al. 2011), multiscale graph decomposition (MN-Cut) (Cour et al. 2005), and mean shift (Comaniciu and Meer 2002). To make a fair comparison, these algorithms are implemented under the same prespecified configuration. For MML-Jef, VB-DNW, MN-Cut, and pBYY, the initial cluster number is set to be 20. For mean shift, the minimum region area is set at 5,000 pixels. For gPb-owt-ucm, we use the segmentation results posted by (Arbelaez et al. 2011), and set the threshold to be 0.5. These settings are fixed throughout all the evaluations. To simplify the computation, we also ignore the re-initiation step in Algorithm 2 to accelerate the speed of pBYY algorithm.

Following the existing convention (Arbelaez et al. 2011), we use PRI, VI, and CR to measure the comparison performance. The result of PRI, VI, and CR scores are shown in Table 3. Moreover, pairwise comparisons of pBYY with each competing algorithm are illustrated in Figure 4. By the PRI and CR measures, pBYY outperforms almost all the algorithms.

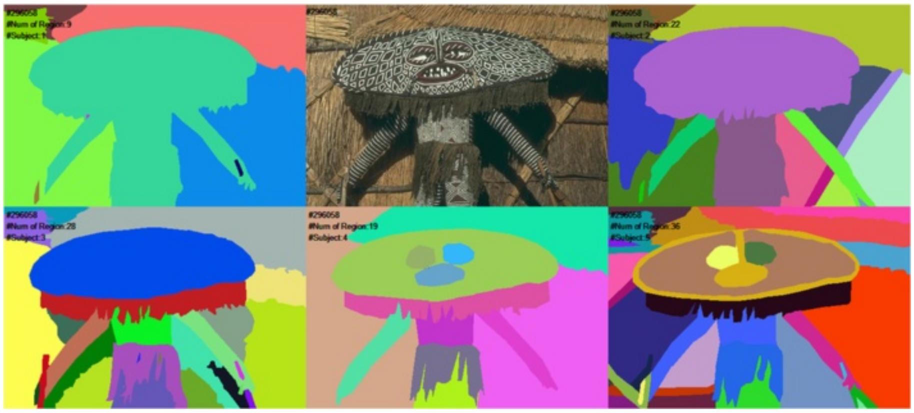

Figure 3 Ground-truth segmentation results hand-drawn by five different human objects on the image ♯296058.

Table 3 Performance scores on the BSDS

	BSDS500						
	Human	**Mean shift**	**MN-Cut**	**gPb-owt-ucm**	**MML-Jef**	**VB-DNW**	**pBYY**
PRI	0.88	0.8157	0.8066	0.7489	0.7851	0.7866	0.8196
VI	1.17	2.2912	2.5163	1.7539	3.4966	3.5589	2.8140
CR	0.72	0.439	0.393	0.439	0.325	0.325	0.487

The performance of each algorithm is evaluated separately against each of five human-drawn ground-truth segmentations per image, and then their average is obtained as the score on this image. For the covering rate metrics, a larger value indicates a better performance.

There is one exception at the center of the first row. The mean shift performs better than pBYY on 53 pieces of images according to PRI. Figure 5 shows the comparisons on four images randomly picked from the BSDS. Human judgement may clearly identify that the segmentations by pBYY look much bettter than the counterparts by mean shift. By the VI criterion, pBYY outperforms MML-Jef and VB-DNW but fails to win gPb-owt-ucm, MN-Cut, and mean shift. Observed from Figure 5, a human judgement may identify that the segmentations by pBYY are much better than the counterparts by gPb-owt-ucm, MN-Cut, and mean shift. Seemingly, the VI is more suitable to measure the clustering-based segmentations for a purpose of getting superpixels, while pBYY outperforms all the algorithms for semantic image segmentation but not be so for segmentations towards superpixels.

Conclusions

On learning the Gaussian mixture model, the existing BYY learning algorithms are featured by either a gradient-based local search that needs an appropriate stepsize to be prespecified or a EM-like two-step alternation that does not request a learning stepsize but may lead to a unstable learning. The proposed pBYY still implements such a

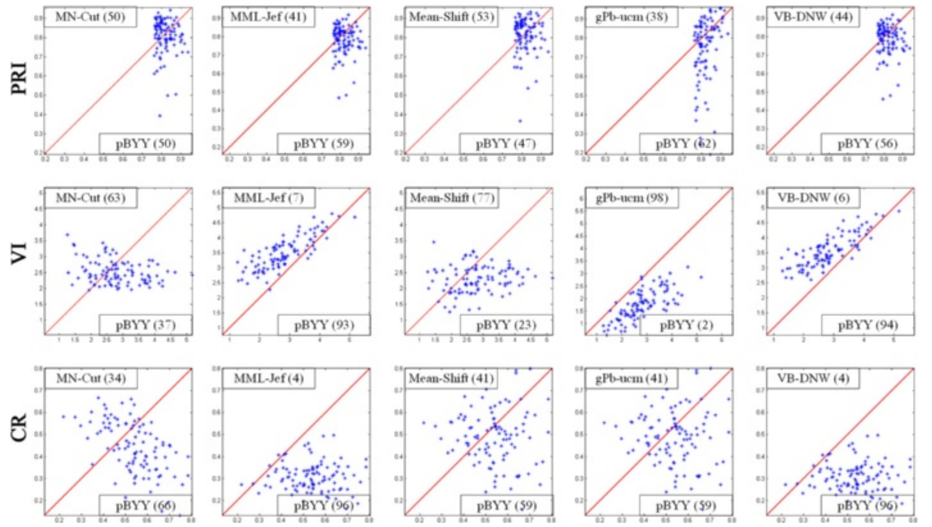

Figure 4 Pairwise comparison of segmentation algorithms on the BSDS500. The coordinates of the blue dots are the PRI, VI, and CR scores pBYY and its competing algorithms obtained per image. Red line represents the boundary of equal performance by the two algorithms, and the boxed digits indicate the number of images where one algorithm is better. For example, the last one shows MN-Cut outscores pBYY merely on 4/100 images.

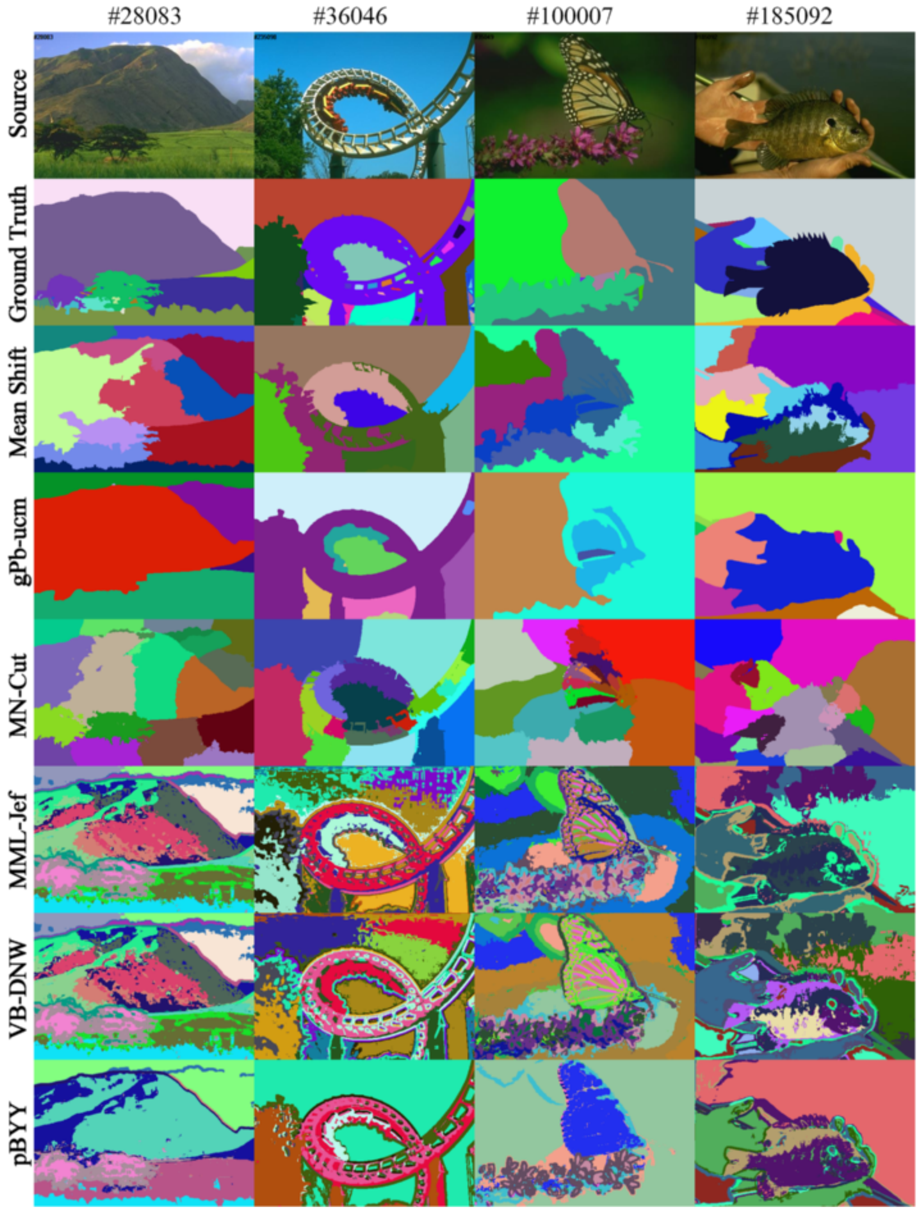

Figure 5 Comparisons on four images from the BSDS500.

two-step alternation but removes the learning unreliability by an embedded projection, outperforming the existing BYY learning algorithms significantly. In the machine learning literature, Bayesian approach with appropriate priori provides a standard direction of developing learning algorithms for model selection, with VB and MML being two typical instances. In (Shi et al. 2011), BYY outperforms MML and VB with the help of the same types of priories, but still fail to prevail with no priori. It has been shown in this paper that pBYY without any priori has outperformed MML-Jef, VB-DNW, BYY-Jef, and BYY-DNW, which confirms that the BYY best harmony learning provides a new perspective for automatic model selection even without a prior. Especially, this pBYY uses an easy computation to prevail the tedious computation required for using the DNW prior. More interestingly, the semantic image segmentation performance on the Berkeley Segmentation Data Set of 100 testing images have shown that pBYY outperforms not only MML-Jef, VB-DNW, BYY-Jef and BYY-DNW but also gPb-owt-ucm, MN-Cut, and mean shift.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

GYC proposed the idea of pBYY algorithm and designed the experiment part with PAH, and LX improved the original idea of pBYY algorithm and refined the presentation of this method. All authors read and approved the final manuscript.

Acknowledgements

Lei Xu was supported by a starting-up grant for the Zhi-Yuan chair professorship by Shanghai Jiao Tong University. Pheng-Ann Heng was partly supported by Hong Kong Research Grants Council General Research Fund (Project No. 412513).

Received: 23 July 2014 Accepted: 15 October 2014
Published online: 27 November 2014

References

Akaike H (1974) A new look at the statistical model identification. Automatic Control IEEE Trans 19(6):716–723

Arbelaez P, Maire M, Fowlkes C, Malik J (2011) Contour detection and hierarchical image segmentation. Pattern Anal Mach Intell IEEE Trans 33(5):898–916

Bauschke H, Borwein JM (1993) On the convergence of von Neumann's alternating projection algorithm for two sets. Set-Valued Anal 1(2):185–212

Bauschke H, Borwein JM (1994) Dykstra's alternating projection algorithm for two sets. J Approximation Theory 79(3):418–443

Barron A, Rissanen J, Yu B (1998) The minimum description length principle in coding and modeling. Inf Theory IEEE Trans 44(6):2743–2760

Bishop CM, Nasrabadi NM (2006) Pattern recognition and machine learning vol 1. Springer, New York

Carpineto C, Romano G (2012) Consensus clustering based on a new probabilistic rand index with application to subtopic retrieval. Pattern Analysis and Machine Intelligence, IEEE Transactions on 34(12):2315–2326

Chiu KC, Xu L (2001) Tests of Gaussian temporal factor loadings in financial APT. In: Proc. of 3rd International Conference on Independent Component Analysis and Blind Signal Separation, December 9-12, San Diego, California, USA. pp 313-318

Corduneanu A, Bishop CM (2001) Variational Bayesian model selection for mixture distributions. In: Artificial Intelligence and Statistics, vol 2001. Morgan Kaufmann, Waltham, MA. pp 27–34

Cour T, Benezit F, Shi J (2005) Spectral segmentation with multiscale graph decomposition. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference On, vol 2. IEEE. pp 1124–1131

Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. Pattern Anal Mach Intell IEEE Trans 24(5):603–619

Escalante R, Raydan M (2011) Alternating projection methods. vol 8. SIAM

Figueiredo MAT, Jain AK (2002) Unsupervised learning of finite mixture models. Pattern Anal Mach Intell IEEE Trans 24(3):381–396

Nigam K, McCallum AK, Thrun S, Mitchell T (2000) Text classification from labeled and unlabeled documents using EM. Mach Learn 39(2-3):103–134

Nikou C, Likas C, Galatsanos NP (2010) A Bayesian framework for image segmentation with spatially varying mixtures. Image Process IEEE Trans 19(9):2278–2289

Reynolds DA (1995) Speaker identification and verification using Gaussian mixture speaker models. Speech Commun 17(1):91–108

Redner RA, Walker HF (1984) Mixture densities, maximum likelihood and the EM algorithm. SIAM Rev 26(2):195–239

Richardson S, Green PJ (1997) On Bayesian analysis of mixtures with an unknown number of components (with discussion). J R Stat Soc: Series B (Statistical Methodology) 59(4):731–792

Rissanen J (1978) Modeling by shortest data description. Automatica 14(5):465–471

Shi L, Tu S, Xu L (2011) Learning Gaussian mixture with automatic model selection: a comparative study on three Bayesian related approaches. Frontiers Electrical Electron Eng China 6(2):215–244

Varma M, Zisserman A (2003) Texture classification: are filter banks necessary? In: Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference On, vol 2. IEEE. pp 691–698

Wallace CS, Dowe DL (1999) Minimum message length and Kolmogorov complexity. Comput J 42(4):270–283

Wagner S, Wagner D (2007) Comparing clusterings: an overview. Universität Karlsruhe, Fakultät für Informatik

Xu L, Krzyzak A, Oja E (1992) Unsupervised and supervised classifications by rival penalized competitive learning. In: Pattern Recognition, 1992. Vol. II. Conference B: Pattern Recognition Methodology and Systems, Proceedings. 11th IAPR International Conference On. IEEE. pp 496–499

Xu L (1995) Bayesian-kullback coupled Ying-Yang machines: unified learnings and new results on vector quantization. In: Proceedings of International Conference on Neural Information Processing, Oct 30–Nov.3, Beijing, China. pp 977–988

Xu L (1998) Rival penalized competitive learning, finite mixture, and multisets clustering. In: Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference On, vol 3. IEEE. pp 2525–2530

Xu L (2009) Learning algorithms for RBF functions and subspace based functions. In: Olivas E, et al. (eds). Handbook of Research on Machine Learning, Applications and Trends: Algorithms, Methods and Techniques. IGI Global, Hershey, PA. pp 60–94

Xu L (2010) Bayesian Ying-Yang system, best harmony learning, and five action circling. Frontiers Electrical Electron Eng China 5(3):281–328

Xu L (2012) On essential topics of BYY harmony learning: current status, challenging issues, and gene analysis applications. Frontiers Electrical Electron Eng 7(1):147–196

Xu L (2014) Further advances on Bayesian Ying-Yang harmony learning. Appl Inform, to appear.

Zhang Y, Brady M, Smith S (2001) Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm. Med Imaging IEEE Trans 20(1):45–57

Zhu S, Zhao J, Guo L, Zhang Y (2013) Unsupervised natural image segmentation via Bayesian Ying–Yang harmony learning theory. Neurocomputing 121:532–539

doi:10.1186/s40535-014-0002-2
Cite this article as: Chen *et al.*: Projection-embedded BYY learning algorithm for Gaussian mixture-based clustering. *Applied Informatics* 2014 **1**:2.