
PERGAMON

Neural Networks 15 (2002) 1125–1151

Neural Networks

www.elsevier.com/locate/neunet

2002 Special Issue

BYY harmony learning, structural RPCL, and topological self-organizing on mixture models☆

Lei Xu*

Department of Computer Science and Engineering, Chinese University of Hong Kong, Shatin, NT, Hong Kong, People's Republic of China

Abstract

The Bayesian Ying-Yang (BYY) harmony learning acts as a general statistical learning framework, featured by not only new regularization techniques for parameter learning but also a new mechanism that implements model selection either automatically during parameter learning or via a new class of model selection criteria used after parameter learning. In this paper, further advances on BYY harmony learning by considering modular inner representations are presented in three parts. One consists of results on unsupervised mixture models, ranging from Gaussian mixture based Mean Square Error (MSE) clustering, elliptic clustering, subspace clustering to NonGaussian mixture based clustering not only with each cluster represented via either Bernoulli–Gaussian mixtures or independent real factor models, but also with independent component analysis implicitly made on each cluster. The second consists of results on supervised mixture-of-experts (ME) models, including Gaussian ME, Radial Basis Function nets, and Kernel regressions. The third consists of two strategies for extending the above structural mixtures into self-organized topological maps. All these advances are introduced with details on three issues, namely, (a) adaptive learning algorithms, especially elliptic, subspace, and structural rival penalized competitive learning algorithms, with model selection made automatically during learning; (b) model selection criteria for being used after parameter learning, and (c) how these learning algorithms and criteria are obtained from typical special cases of BYY harmony learning.

Keywords: BYY system; Harmony learning; Normalization; Data-smoothing; Model selection; Clustering; Elliptic and structural RPCL; Gaussian mixture; NonGaussain mixture; ICA; Factor analysis; LMSER; Topological map; Mixture-of-experts; RBF net; Kernel regression

1. Introduction

Statistical learning is a process that an intelligent system estimates or learns the underlying distribution together with dependence structures among the world **X** it observes. Such a learning consists of two key subtasks. First, we need an appropriate structure for the intelligent system to accommodate the dependence structures among the world **X**. Second, based on a set $\{x_t\}_{t=1}^N$ of samples from the world **X**, we need to decide not only an appropriate scale for this structure but also all the unknown parameters in this structure.

Though the observation world in reality may have various complicated dependence structures, our attention can be focused on certain simplified specific structures in performing different learning tasks. Moreover, we can also decompose a complicated task into a number of tasks on much simplified small worlds. A number of typical structures have been reviewed and discussed recently (Xu, 2002b). Among them, a widely used structure is finite mixture that consists a finite number of individual models such that an observation x comes from a specific individual model $q(x|\ell)$ with a prior probability α_ℓ (McLachlan & Basford, 1988; Redner & Walker, 1984). Particularly, it becomes the widely used Gaussian mixture when $q(x|\ell)$ is simply a Gaussian. Moreover, finite mixture has also been extended to supervised learning. Typical examples include the popular mixture expert (ME) model (Jacobs, Jordan, Nowlan, & Hinton, 1991; Jordan & Jacobs, 1994; Jordan & Xu, 1995) and its alternative model (Xu, 1998a; Xu, Jordan, & Hinton, 1995) as well as the normalized radial basis function (RBF) nets (Moody & Darken, 1989; Nowlan, 1990; Xu, 1998a; Xu, Krzyzak, & Yuille, 1994).

The task of learning on a finite mixture is usually made via Maximum Likelihood (ML) that can be effectively implemented by the Expectation–Maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977; Redner & Walker, 1984). The ML learning works well on a large size set $\{x_t\}_{t=1}^N$ of samples. However, a key challenge to all the learning tasks is that learning is usually made on a finite size N of samples but our ambition is to get the underlying

☆ The work described in this paper was fully supported by a grant from the Research Grant Council of the Hong Kong SAR (Project No.: CUHK4336/02E).

* Tel.: +852-2609-8423; fax: +852-2603-5024.
E-mail address: lxu@cse.cuhk.edu.hk (L. Xu).

0893-6080/02/$ - see front matter
PII: S0893-6080(02)00084-9

distribution such that we can apply it to all or as many as possible new coming samples from **X**. That is, we expect that a learned system has its generalization ability as good as possible.

In past decades, many efforts have been made towards this critical challenge. On one hand, a number of model selection criteria have been developed to evaluate a family of structures with different scales such that a best one is selected. Typical examples include the VC dimension based learning theory (Vapnik, 1995), AIC (Akaike, 1974) as well as its extensions (Bozdogan, 1987; Bozdogan & Ramirez, 1988; Cavanaugh, 1997; Sugiura, 1978), cross-validation (Rivals & Personnaz, 1999; Stone, 1974, 1978). On the other hand, a number of theories for regularization have been proposed to impose constraints on parameters in a given structure of a large scale such that it becomes effectively equivalent to reducing the scale of the structure to an appropriate level. Typical examples include Tikhonov-type regularization theory (Girosi, Jones, & Poggio, 1995; Tikhonov & Arsenin, 1977), Bayesian theory (Schwarz, 1978; Neath & Cavanaugh, 1997; Mackey, 1992), the Minimum Message Length (MML) theory (Wallace & Boulton, 1968; Wallace & Dowe, 1999), and the Minimum Description Length (MDL) theory (Hinton & Zemel, 1994; Rissanen, 1986, 1999).

Firstly proposed in 1995 (Xu, 1995; Xu, 1996) and systematically developed in past years, the BYY harmony learning acts as a general statistical learning framework not only for understanding several existing typical learning models, but also for tackling the above key challenge with a new learning mechanism that makes model selection implemented either *in parallel automatically* during parameter learning or *subsequently after* parameter learning via a new class of model selection criteria obtained from this mechanism. Also, this BYY harmony learning has motivated three types of regularization, namely a data smoothing technique that provides a new solution on the hyper-parameter in a Tikinov-like regularization (Tikhonov & Arsenin, 1977), a normalization with a new conscience de-learning mechanism that closely relates to the Rival Penalized Competitive Learning (RPCL) (Xu, Krzyzak, & Oja, 1993), and a structural regularization by imposing certain structural constraints via designing a specific forward structure in a BYY system.

Specifically, the BYY harmony learning in different specific cases results in various specific learning algorithms as well as the detailed forms for implementing regularization and model selection, covering three main statistical learning paradigms.

First, new results are obtained on several major unsupervised learning tasks, including: (a) new criteria for selecting the number of clusters in the k-means clustering and its various extensions that base on Gaussian mixture, together with adaptive EM-like algorithms (Xu, 1997, 2001b); (b) an adaptive algorithm for Gaussian factor analysis and a criterion for determining the dimension of principal subspace (Xu, 1998b); (c) a learned parametric model for Independent Component Analysis (ICA) that works on sources mixed with super-Gaussian and sub-Gaussian (Xu, Cheung, & Amari, 1998); (d) adaptive algorithms for independent binary factor analysis and independent real factor analysis (or noisy ICA), together with the corresponding criteria for the number of independent factors (Xu, 1998b, 2000, 2001a); as well as (e) several extensions of LMSER learning and the corresponding criteria for the number of hidden units (Xu, 1993, 1998b, 2000, 2001a).

Second, we obtain not only new understanding on three major supervised learning models, namely three layer forward net with back-propagation learning, Mixture-of-Experts (ME) model (Jacobs et al., 1991; Jordan & Jacobs, 1994; Jordan & Xu, 1995) and its alternative model (Xu, 1998a; Xu et al., 1995), as well as normalized RBF nets (Moody & Darken, 1989; Nowlan, 1990; Xu et al., 1994) and its extensions (Xu, 1998a), but also new adaptive algorithms for learning and new criteria for deciding the number of hidden units, experts, and basis functions (Xu, 1998a, 2001b).

Third, the BYY harmony learning has also been extended to act as a general state space approach for modeling data that has temporal relationship among samples, which provides not only a unified point of view on Kalman filter, Hidden Markov model (HMM), ICA and Blind Source Separation (BSS) with extensions, but also several new results such as higher order HMM, independent HMM for binary BSS, temporal ICA and temporal factor analysis for noisy real BSS, with adaptive algorithms for implementation and criteria for selecting the number of states or sources (Xu, 2000).

In this paper, further advances on the BYY harmony learning are obtained by considering modular inner representations, which leads to mixture models with different structures. Specifically, we start at Gaussian mixture and Gaussian ME in Section 3, including not only unsupervised tasks of Mean Square Error (MSE) clustering, elliptic clustering, subspace clustering but also supervised tasks of Gaussian ME, RBF nets, and Kernel regressions. Not only previous results are reviewed from a unified perspective, but also elliptic RPCL and structural RPCL algorithms are presented as special cases of the BYY harmony learning. Model selection on both the number of clusters and the dimensions of subspaces is made either automatically during learning or alternatively by the corresponding criteria obtained from the BYY harmony principle.

Then, Section 4 further goes beyond Gaussians with new results from three aspects. *First*, a nonGaussian structural RPCL clustering is made via Bernoulli–Gaussian mixtures, with the number of clusters and the polyhedral structure of each cluster determined either automatically during learning or after learning by a given criterion. Moreover, a local LMSER is obtained for not only acting as a fast

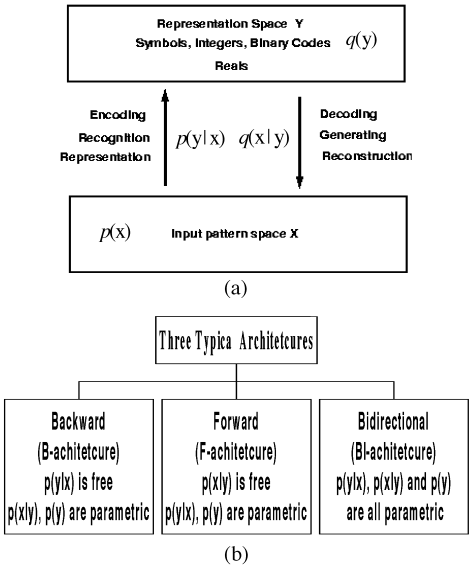

Fig. 1. Bayesian Ying-Yang system and three architectures.

implementation of such a structural clustering but also implementing a competitive *Principal ICA* (P-ICA) on these clusters. *Second*, all the results are further extended to clusters of nonGaussian represented via independent real factor models. *Third*, typical algorithms for ICA and competitive ICA are summarized from the perspective of the BYY harmony learning on a F-architecture, in comparison with P-ICA as well as competitive P-ICA.

Finally, before concluding in Section 6, two strategies are discussed in Section 5 such that the above structural mixtures are extended into self-organized topological maps for tasks of both supervised learning and unsupervised learning.

2. BYY system and harmony learning

2.1. BYY system, harmony learning, and regularization techniques

2.1.1. BYY system and harmony principle

As shown in Fig. 1(a), we consider a world $\mathbf{X}$ with each object in observation represented by a stochastic notation $\mathbf{x} \in \mathbf{X}$. Corresponding to each $\mathbf{x}$, there is an inner representation $\mathbf{y} \in \mathbf{Y}$ in the representation domain $\mathbf{Y}$ of a learning system. We consider the joint distribution of $\mathbf{x}, \mathbf{y}$, which can be understood from two complement perspectives.

On one hand, we can interpret that each $\mathbf{x}$ is generated (or reconstructed/decoded) from an invisible inner representation $\mathbf{y}$, that may in a form of symbol, integral, binary code, and real vectors according to the natures of learning tasks, via a backward path distribution $q(\mathbf{x}|\mathbf{y})$ or a generative model by

$$q(\mathbf{x}) = \int q(\mathbf{x}|\mathbf{y})q(\mathbf{y})\mathrm{d}\mathbf{y}. \tag{1}$$

That is, $\mathbf{x}$ is generated from an inner distribution $q(\mathbf{y})$ in a structure that is designed according to the learning tasks.

On the other hand, we can interpret that each $\mathbf{x}$ is mapped (or encoded/recognized) into an invisible inner representation $\mathbf{y}$ via a forward path distribution $p(\mathbf{y}|\mathbf{x})$ or a representative model by

$$p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{x})p(\mathbf{x})\mathrm{d}\mathbf{x}, \tag{2}$$

which matches the inner density $q(\mathbf{y})$ in a pre-specified structure.

The two perspectives reflect the two types of Bayesian decomposition of the joint density $q(\mathbf{x}|\mathbf{y})q(\mathbf{y}) = q(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y}|\mathbf{x})$ on $X \times Y$. Without any constraints, the two decompositions should be theoretically identical. However, in a real consideration, the four components $p(\mathbf{y}|\mathbf{x})$, $p(\mathbf{x})$, $q(\mathbf{x}|\mathbf{y})$, $q(\mathbf{y})$ should be subject to certain structural constraints. Thus, we usually have two different but complementary Bayesian representations:

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x}), \qquad q(\mathbf{x}, \mathbf{y}) = q(\mathbf{x}|\mathbf{y})q(\mathbf{y}), \tag{3}$$

which, as discussed in the original paper (Xu, 1996), compliments to the famous Chinese ancient Ying-Yang philosophy with $p(\mathbf{x}, \mathbf{y})$ called Yang machine that consists of the observation space (or called Yang space) by $p(\mathbf{x})$ and the forward pathway (or called Yang pathway) by $p(\mathbf{y}|\mathbf{x})$, and with $q(\mathbf{x}, \mathbf{y})$ called Ying machine that consists of the invisible state space (or Ying space) by $q(\mathbf{y})$ and the Ying (or backward) pathway by $q(\mathbf{x}|\mathbf{y})$. Such a pair of Ying-Yang models is called *Bayesian Ying-Yang* (BYY) system.

The task of learning on a BYY system consists of specifying all the aspects of the system.

First, on a set $\{x_t\}_{t=1}^N$ of samples from the observed world $\mathbf{X}$, the distribution $p(\mathbf{x})$ can be obtained by a nonparametric Parzen window PDF estimate (Devroye, Györfi, & Lugosi, 1996):

$$p_{h_x}(x) = \frac{1}{N}\sum_{t=1}^{N} G(x|x_t, h_x^2 I), \tag{4}$$

where and throughout this paper, $G(x|m, \Sigma)$ denotes a Gaussian density with mean vector m and covariance matrix Σ. Particularly, when $h_x = 0$, $p_{h_x}(x)$ becomes the empirical density:

$$p_0(x) = \frac{1}{N}\sum_{t=1}^{N} \delta(x - x_t),$$

$$\delta(u) = \begin{cases} \lim_{\delta V_u \to 0} 1/\delta V_u, & \text{for } u = 0, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

where δV_u is the infinitesimal volume at $u = 0$ in the space of u. This $\delta(u)$ is usually referred as the Kronecker function.

Second, we need to design the structure of each of other three components $p(\mathbf{y}|\mathbf{x})$, $q(\mathbf{x}|\mathbf{y})$, $q(\mathbf{y})$.

Specifically, the structure of $q(\mathbf{y})$ depends on the considered learning tasks, which is closely related to the complexity of the world $\mathbf{X}$ that we observe. In this paper, we consider a modular world $\mathbf{X} = \{X, L\}$ that consists of a number of individual objects to observe, with L denoting a set of labels and each $\ell \in L$ denoting an object. In this case, each $\mathbf{x} = \{x, \ell\}$ contains a feature vector $x = [x^{(1)}, ..., x^{(d)}]^{\mathrm{T}}$ observed from the object ℓ, subject to a joint underlying distribution $p(\mathbf{x}) = p(x, \ell)$. Correspondingly, we consider a representation domain $\mathbf{Y} = \{Y, L\}$, subject to a parametric structure of $p(\mathbf{y}) = p(y, \ell)$ that describes the vector $\mathbf{y}$ and the label ℓ jointly. This $p(\mathbf{y})$ is specified by three ingredients. One is a set $\mathbf{k}$ that consists of the scales k, $\{m_\ell\}$ of the representation domain $\mathbf{Y}$, where k is the number of labels in L, and m_ℓ is the dimension of either a binary or real vector y. The other ingredient is the function form of $p(\mathbf{y})$ which is usually pre-specified according to a specific learning task. Another ingredient consists of a set θ_y of parameters in this given function form.

Moreover, we need to design the structures of $p(\mathbf{y}|\mathbf{x})$, $q(\mathbf{x}|\mathbf{y})$ that specify the mapping capacity of $x \rightarrow y$ and $y \rightarrow x$, respectively. Each of the two can be either parametric or structure free. We say $p(u|v)$ is structural free in the sense that $p(u|v)$ can be any function that satisfies $\int p(u|v) = 1$, $p(u|v) \geq 0$. A structure-free distribution is actually specified via learning. Given its function form, a parametric $p(u|v, \theta_{u|v})$ is featured by a set $\theta_{u|v}$ of unknown parameters.

Coordinately, the nature of a BYY system is featured by the structure of $q(\mathbf{y})$ for describing the representation domain $\mathbf{Y}$, and the architecture of a BYY system is featured by a combination of the specific structures of $p(\mathbf{y}|\mathbf{x})$, $q(\mathbf{x}|\mathbf{y})$. Discarding a useless architecture that both $p(\mathbf{y}|\mathbf{x})$, $q(\mathbf{x}|\mathbf{y})$ are structure-free, a meaningful BYY architecture can be one of the three choices given in Fig. 1(b) namely, the B-, F-, and BI-architecture.

In a narrow sense, our learning task includes two subtasks. One is parameter learning for determining the value of θ that consists of all the unknown parameters in $p(\mathbf{y}|\mathbf{x})$, $q(\mathbf{x}|\mathbf{y})$, $q(\mathbf{y})$ as well as h_x (if any). The other subtask is selecting the representation scales $\mathbf{k} = \{k, \{m_\ell\}\}$, which is called *model selection* since a collection of specific BYY systems with different such scales corresponds to a family of specific models that share a same system configuration but in different scales.

The *fundamental learning principle* is to make the Ying machine and Yang machine be best harmony in a twofold sense:

- The difference between the two Bayesian representations in Eq. (3) should be minimized.
- The resulting BYY system should be of the least complexity.

2.1.2. Kullback divergence, harmony measure, and regularization techniques

To implement the above harmony learning principle, we need to formalize it mathematically. One possible measure is the well known Kullback divergence

$$D_K(p\|q) = \int p(u) \ln \frac{p(u)}{q(u)} \mathrm{d}u \geq 0, \quad \text{with } D_K(p\|q) = 0, \text{ iff } p(u) = q(u), \tag{6}$$

which is applicable to both the cases that p, q are discrete and continuous densities. The minimization of Kullback divergence does implement the above first point well, and thus this is why it has been used in the early stage of the BYY system (Xu, 1996, 1997, 1998b). However, it is not able to implement the least complexity nature. In other words, the Kullback divergence can only be used for a partial implementation. We need a measure that implements both of the above two points.

We consider the following cross-entropy

$$H(p\|q) = \sum_{t=1}^{N} p_t \ln q_t, \tag{7}$$

where both $p(u)$, $q(u)$ are discrete densities in the form

$$q(u) = \sum_{t=1}^{N} q_t \delta(u - u_t), \qquad \sum_{t=1}^{N} q_t = 1. \tag{8}$$

The maximization of $H(p\|q)$ has two interesting natures:

- *Matching nature* with p fixed, $\max_q H(p\|q)$ pushes q towards

$$q_t = p_t, \text{ for all } t. \tag{9}$$

- *Least complexity nature* $\max_p H(p\|q)$ with q fixed pushes p towards its simplest form

$$p(u) = \delta(u - u_\tau), \text{ or } p_t = \bar{\delta}_{t,\tau}, \text{ with } \tau = \arg\max_t q_t, \tag{10}$$

where and throughout this paper, $\bar{\delta}_{j,j^*}$ denotes

$$\bar{\delta}_{j,j^*} = \begin{cases} 1, & j = j^*, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

As discussed in Xu (2001a), Eq. (10) is a kind of the least complexity from the statistical perspective. In other words, the maximization of the functional $H(p\|q)$ indeed implements the above harmony learning principle mathematically.

Moreover, as shown in Xu (2001a), either a discrete or continuous density $q(u)$ can be represented in the form of Eq. (8) via the following normalization:

$$\hat{q}_t = q(u_t)/z_q, \qquad z_q = \sum_{t=1}^{N} q(u_t) \tag{12}$$

based on a given set $\{u_t\}_{t=1}^N$ of samples.

Putting Eq. (12) into Eq. (7), it follows that we can

further get a general form of the harmony measure (Xu, 2001a):

$$H(p\|q) = \int p(u)\ln q(u)\mathrm{d}u - \ln z_q, \tag{13}$$

which degenerates to Eq. (7) when $q(u)$, $p(u)$ are discrete as in Eq. (8). Particularly, when $p(u)$ is given by its empirical density in the form of Eq. (5), a crude approximation $z_q = 1$ will make $H(p\|q)$ become the likelihood

$$L(\theta) = \sum_{t=1}^{N} \ln q(u_t). \tag{14}$$

That is, the harmony learning and the ML learning are equivalent in this case.

We further consider Eq. (13) with the normalization term z_q in Eq. (12). It follows from $p(u)$ given by Eq. (5) and $q(u)$ by Eq. (12) with u in the place of x that

$$H(p\|q) = L(\theta) - \ln \sum_{t=1}^{N} q(u_t). \tag{15}$$

By comparing the gradients:

$$\begin{aligned}
&\nabla_\theta L(\theta) = Gd(\gamma_t)|_{\gamma_t=1/N},\\
&\nabla_\theta H(p\|q) = Gd(\gamma_t)|_{\gamma_t=(1/N)-\tilde{q}(u_t|\theta)},\\
&Gd(\gamma_t) = \sum_t \gamma_t \nabla_\theta \ln q(u_t|\theta),\\
&\tilde{q}(u_t|\theta) = q(u_t|\theta)/\sum_\tau q(u_\tau|\theta),
\end{aligned} \tag{16}$$

we see that the log-normalization term $\ln z_q$ causes a *conscience* de-learning made on the ML learning, which is thus referred as *normalization learning*, in a sense that the degree of de-learning on learning u_t is proportional to the likelihood that $q(u|\theta)$ fits u_t. That is, more better it is fitted, more conscience it makes during learning, which actually provides a new regularization technique that prevents $q(u|\theta)$ to over-fit a data set of a finite size.

Alternatively, considering $p(u)$ given by a Parzen window estimate Eq. (4) with u in the place of x, we can also approximate z_q under a weak constraint $\sum_{t=1}^{N} p(u_t) \approx \sum_{t=1}^{N} q(u_t)$, and thus leads to a regularized learning as shown by Eqs. (28) and (33) in (Xu, 2001a). That is, Eq. (13) becomes

$$\begin{aligned}
&H(p\|q) = \tilde{L}_S(\theta_h) + 0.5k\ln(2\pi h^2) + \ln N - J_H(h,k),\\
&J_H(h,k) = \ln\left[\sum_{\tau=1}^{N}\sum_{t=1}^{N}\exp\left(-0.5\frac{\|u_t-u_\tau\|^2}{h^2}\right)\right],\\
&\tilde{L}_S(\theta_h) = \int p_h(u)\ln q(u)\mathrm{d}u \approx L(\theta) + 0.5h^2\pi_q,\\
&\pi_q = \frac{1}{N}\sum_t \mathrm{Tr}\left[\frac{\partial^2 \ln q(u|\theta)}{\partial u\,\partial u^{\mathrm{T}}}\right]_{u=u_t}.
\end{aligned} \tag{17}$$

$\tilde{L}_S(\theta_h)$ regularizes the ML learning by smoothing each likelihood $\ln q(u_t)$ in the near-neighbor of u_t, which is referred as *data smoothing*. It can also be observed that the role h^2 is equivalent to the hyper-parameter in Tikhonov-type regularization (Bishop, 1995). What is new here is that the other terms in $H(p\|q)$ balance $\tilde{L}_S(\theta_h)$ such that an appropriate h can be learned together with θ.

We return to the Kullback divergence equation (6). When both $p(u)$, $q(u)$ are discrete densities in the form of Eq. (8), from Eq. (6) we can directly get

$$\begin{aligned}
&D_K(p\|q) = \sum_{t=1}^{N} p_t \ln\frac{p_t}{q_t} = -E_p - H(p\|q),\\
&E_p = -\sum_{t=1}^{N} p_t \ln p_t.
\end{aligned} \tag{18}$$

In help of the form in Eq. (12) for both p, q, similar to Eq. (13) we can get

$$D_K(p\|q) = \sum_{t=1}^{N} \frac{p(u_t)}{z_p}\ln\frac{p(u_t)/z_p}{q(u_t)/z_q} \approx \int p(u)\ln\frac{p(u)/z_p}{q(u)/z_q}\mathrm{d}u,$$

or

$$D_K(p\|q) = -H(p\|q) - E_p,\quad E_p = -\int p(u)\ln p(u)\mathrm{d}u + \ln z_p. \tag{19}$$

Obviously, $D_K(p\|q)$ returns back to Eq. (6) when $z_q = z_p$ or we can at least approximately have $z_q = z_p$.

Moreover, we have two observations. First, Eq. (6) is directly applicable to the cases that both $p(u)$, $q(u)$ are discrete densities and that both $p(u)$, $q(u)$ are continuous densities. Second, $\min D_K(p\|q)$ is different from $\max H(p\|q)$ in that it also maximizes the entropy E_p of $p(u)$, which prevents $p(u)$ towards the form by Eq. (10). This explains why Eq. (18) does not have the least complexity nature.

However, Eq. (6) is not directly applicable to the cases that $p(u)$ contains δ densities. Thus, we cannot assume with $p(u) = p_0(u)$ given by the empirical estimate Eq. (5) and $q(u)$ by a continuous parametric model, since it will involve an infinite term $\ln \delta_u(0)$ that makes $\min D_K(p\|q)$ meaningless, where and throughout the paper, we denote $\delta_u(0) = \delta(u)|_{u=0}$.

In contrast, it follows from $p(u_t)/z_p = 1/N$ that $D_K(p\|q)$ in Eq. (19) becomes

$$D_K(p\|q) = -H(p\|q) + \ln N, \tag{20}$$

with $H(p\|q)$ given by Eq. (15). In this case, $\min D_K(p\|q)$ becomes equivalent to $\max H(p\|q)$ in its normalization implementation or equivalently the regularized ML learning. Moreover, considering $z_q = 1$, we will again lead to the ML learning Eq. (14).

Generally, it follows from Eqs. (18) and (19) that $\min D_K(p\|q)$ and $\max H(p\|q)$ become equivalent when E_p is

a constant that does not relate to learning, which happens in the following typical situations:

(a) $p(u) = \delta(u - u^*)$ with u^* unknown. In this case, $E_p = 0$ and thus u^* is decided via learning with $u^* = \arg\max_u q(u)$.
(b) $p(u)$ is completely known and thus E_p is a fixed constant, e.g., the above discussed $p(u) = p_0(u)$ by Eq. (5) is such a case.
(c) Either $p(u) = G(u|u^*, h^2 I)$ with u^* known but $h^2 > 0$ unknown or $p(u)$ given by a Parzen window estimate Eq. (4). In this case, considering $z_p = z_q$ approximately, we can rewrite $D_K(p\|q)$ into

$$D_K(p\|q) = \int p(u) \ln q(u) \mathrm{d}u - Z_q(h), \qquad Z_q(h) = -\int p(u) \ln p(u) \mathrm{d}u, \tag{21}$$

which has a same format as in Eq. (13), with $Z_q(h)$ taking a similar role as $\ln z_q$. It further leads to the same form as in Eq. (17) when $p(u)$ is given by Eq. (4), except that the term $J_H(h,k)$ is replaced by

$$J_K(h,k) = \frac{1}{N} \sum_{\tau=1}^{N} \ln\left[\sum_{t=1}^{N} \exp\left(-0.5 \frac{\|u_t - u_\tau\|^2}{h^2}\right)\right], \tag{22}$$

it can be observed that the difference $J_H(h,k) - J_K(h,k)$ approaches zero when h is small enough. So, acting as two types of the regularized ML, learning by the harmony principle and Kullback principle are not exactly the same but becomes approximately equivalent when the smoothing parameter h is small enough.
(d) A combination of all or any two of the above items (a), (b) and (c).

The fact that $\max_\theta \int p_0(u) \ln q(u) \mathrm{d}u$ leads to the ML learning is well known in the literature. The above discussed relations between $\min D_K(p\|q)$ and $\max H(p\|q)$ are not surprising too. What is new here is that a general z_q introduces a regularization to the ML learning via either a de-learning or a Tikhonov-type. But there is no need to make model selection since $p(u)$ is usually fixed by either Eqs. (4) or (5). This explains why the least complexity nature Eq. (10) is regarded as being useless in the conventional literature. In the sequel, however, we will show that this nature does make model selection possible in learning on the BYY system.

2.1.3. BYY harmony learning

By putting $p(u) = p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x})$, $q(u) = q(\mathbf{x}, \mathbf{y}) = q(\mathbf{x}|\mathbf{y})q(\mathbf{y})$ into Eq. (13), we have

$$H(p\|q) = \int p(\mathbf{y}|\mathbf{x})p(\mathbf{x}) \ln[q(\mathbf{x}|\mathbf{y})q(\mathbf{y})] \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{y} - \ln z_q, \tag{23}$$

where only $p(\mathbf{x})$ is fixed at a nonparametric estimate by Eqs. (4) or (5), but $p(\mathbf{y}|\mathbf{x})$ is not. In this case, the least complexity nature by Eq. (10) will push $p(\mathbf{y}|\mathbf{x})$ into its least complexity during learning, e.g. in a B-architecture, $p(\mathbf{y}|\mathbf{x})$ is free and thus determined by $\max_{p(\mathbf{y}|\mathbf{x})} H(p\|q)$, resulting in

$$p(\mathbf{y}|\mathbf{x}) = \delta(\mathbf{y} - \hat{\mathbf{y}}), \qquad \hat{\mathbf{y}} = \arg\max_{\mathbf{y}} [q(\mathbf{x}|\mathbf{y})q(\mathbf{y})] \tag{24}$$

In turn, the matching nature of harmony learning will further push $q(\mathbf{x}|\mathbf{y})$ and $q(\mathbf{y})$ towards their corresponding least complexity forms. In a BI-architecture, the learning will similarly push $p(\mathbf{y}|\mathbf{x})$ into its least complexity form, e.g. $p(\mathbf{y}|\mathbf{x}) = \delta(\mathbf{y} - f(\mathbf{x}, W_{y|x}))$ (Xu, 2001a), where $f(\mathbf{x}, W_{y|x})$ is a deterministic forward mapping.

This salient feature can be further understood from an information transfer perspective. As discussed (Xu, 2002a), the BYY harmony learning shares the common sprit of the MML (Wallace & Boulton, 1968; Wallace & Dowe, 1999) or the MDL (Hinton & Zemel, 1994; Rissanen, 1986, 1999), but with two key differences. First, the BYY harmony learning maps $\mathbf{x}$ to $\mathbf{y}$ and then code $\mathbf{y}$ for transmission while the MML/MDL approach uses a model $p(\mathbf{x}|\theta)$ to directly code $\mathbf{x}$. Second, the BYY harmony learning uses $\ln z_q^{-1}$ to avoid the implementing difficulty of the MML/MDL approach on requiring a known prior density $p(\theta)$ for encoding parameter set θ.

Mathematically, the implementation of the harmony learning is the following optimization

$$\max_{\theta,\mathbf{k}} H(\theta, \mathbf{k}), \qquad H(\theta, \mathbf{k}) = H(p\|q), \tag{25}$$

which is a combined task of continuous optimization for parameter learning and discrete optimization for model selection, both under a same cost function $H(\theta, \mathbf{k})$.

In literature, existing model selection criteria are only used for selecting models after parameter learning, but not applicable to parameter learning. Instead, parameter learning has to be made under a different criterion, usually under the ML criterion. However, it is well known that ML is not good on model selection, especially for a small size of training samples.

Of course, problem (25) can also be implemented in such a two-phase style as follows:

- In the first phase, we enumerate $\mathbf{k} = \{k, \{m_\ell\}\}$ from small values incrementally. At each specific k, $\{m_\ell\}$, we perform parameter learning for a best value θ^* by

$$\max_{\theta} H(\theta), \qquad H(\theta) = H(\theta, \mathbf{k}). \tag{26}$$

For simplicity, we can make Eq. (26) under the following

constraint

$$\alpha_\ell = 1/k, \qquad \int (y-\mu_{y,\ell})(y-\mu_{y,\ell})^{\mathrm{T}} q(y|\ell)\mathrm{d}y = b_0 I,$$
$$\mu_{y,\ell} = \int y q(y|\ell)\mathrm{d}y. \tag{27}$$

where $b_0 > 0$ is a given constant. For example, $b_0 = 0.25$ when y comes from Bernoulli and $b_0 = 1$ when y comes from Gaussian.

- In the second phase, with the result θ^* obtained from Eq. (26), we select a best k^*, $\{m_\ell^*\}$ by

$$\min_{k,\{m_\ell\}} J(k,\{m_\ell\}), \qquad J(k,\{m_\ell\}) = -H(\theta^*, \mathbf{k}). \tag{28}$$

If there are more than one solutions so that $J(k,\{m_\ell\})$ gets a same minimum, we take the one with the smallest value on k. If still more than one solutions, we take ones with smallest values of $\{m_\ell\}$.

Being different from the conventional approaches, Eqs. (26) and (28) share the same cost function. This feature makes it possible to simultaneously implement parameter learning and model selection in parallel. Actually, the least complexity nature equation (10) makes us possible to implement parameter learning with automatic model selection, e.g. in a B-architecture, Eq. (10) will lead to Eq. (24) that makes not only the integrals over y disappear but also θ take a specific value such that $\mathbf{k} = \{k, \{m_\ell\}\}$ are effectively reduced to appropriate scales.

To get a further insight, we consider

$$q(\mathbf{y}) = q(y,\ell) = q(y|\ell)q(\ell),$$
$$q(\ell) = \sum_{j=1}^{k} \alpha \delta(\ell - j), \quad \alpha_\ell \geq 0, \quad \sum_{\ell=1}^{k} \alpha_\ell = 1. \tag{29}$$

It is obvious that $\alpha_\ell = 0$ for some ℓ implies that k is reduced by one. Also, we observe that the form of

$$q(y|\ell) = \delta(y^{(j)} - c_0) q(y^{-}|\ell),$$
$$\text{with } y = [y^{-}, y^{(j)}]^{\mathrm{T}} \text{ and } c_0 \text{ being a constant,} \tag{30}$$

implies that the dimension for $y^{(j)}$ can be removed and thus m_ℓ is reduced by one. Therefore, a value of θ with these two types of settings is equivalent to forcing $k, \{m_\ell\}$ effectively to be reduced to appropriate scales.

With Eq. (24) put into Eq. (23), we can further observe that the on-line implementation of max $H(p\|q)$ is equivalent to maximizing $\ln q(x|y,\ell)$, $\ln q(y|\ell)$ and $\ln \alpha_\ell$. Specifically, maximizing $\ln \alpha_\ell$ leads to that

$$\text{each extra } \alpha_\ell \text{ is pushed towards zero.} \tag{31}$$

and maximizing $\ln q(y|\ell)$ leads to that

$$q(y^{(j)}|\ell) \text{ on each extra dimension is pushed towards } \delta(y^{(j)} - c_0). \tag{32}$$

Therefore, fixing the scales of $\mathbf{k}$ large enough, we implement Eq. (26) with the least complexity nature by Eq. (10) automatically implying model selection during learning.

The detailed implementation depends on the detailed forms of $H(p\|q)$ that are different for different architectures of a BYY system.

2.2. Three typical architectures

2.2.1. B-architecture

It consists of a Ying machine $q(\mathbf{x},\mathbf{y}) = p(y,\ell|x)p(x)$ and a Yang machine $q(\mathbf{x},\mathbf{y}) = q(x|y,\ell)q(y|\ell)q(\ell)$. The feature is that $p(\mathbf{y}|\mathbf{x}) = p(y,\ell|x) = p(y|x,\ell)p(\ell|x)$ is structure-free and thus indirectly specified via learning in help of the structures of $q(y|\ell)$, $q(\ell)$ and $q(x|y,\ell)$.

From the harmony learning Eq. (25), we find that Eq. (24) takes the following detailed form

$$p(y|x,\ell) = \delta(y - y_\ell(x)), \qquad p(\ell|x) = \delta(\ell - \ell(x)),$$
$$y_\ell(x) = \arg\min_y d(\ell, x, y),$$
$$\ell(x) = \arg\min_\ell d(\ell, x, y_\ell(x)), \tag{33}$$
$$d(\ell, x, y) = -\ln[q(x|y,\ell)q(y|\ell)q(\ell)] + \ln z_q.$$

Further substituting it into Eq. (23) with $p(x)$ given by Eq. (4) we get

$$H(p\|q) = \frac{1}{N}\sum_{t=1}^{N}\sum_{\ell} p_t(\ell) H(x_t, y_{t,\ell}) - \ln z_q, \quad y_{t,\ell} = y_\ell(x_t),$$
$$p_t(\ell) = \bar{\delta}_{\ell,\ell_t}, \qquad \ell_t = \ell(x_t). \tag{34}$$

Specifically, with $p(x) = p_0(x)$ given by Eq. (5), it is similar to Eq. (15) that $H(x_t, y_{t,\ell})$ and z_q have the following detailed form

$$H(x_t, y_{t,\ell}) = \ln[q(x_t|y_{t,\ell})q(y_{t,\ell}|\ell)q(\ell)],$$
$$z_q = z_q^N = \sum_{t,\ell\in L_t} q(x_t|y_{t,\ell},\ell)q(y_{t,\ell}|\ell)q(\ell), \tag{35}$$

with the normalization term z_q introducing a de-learning, which is hereafter referred as *normalization learning*. Particularly, this regularization disappears by setting $z_q = 1$, which is referred as *empirical learning*.

Usually, L_t takes a set in the following four typical

choices:

$$L_t = \begin{cases} \text{a unique } \ell_t = \ell(x_t) \text{ or equivalently } \ell_t = \arg\min_{\ell} d(\ell, x_t, y_{t,\ell}), & \text{(a)}, \\ \ell_t \text{ and } \ell_t^r = \arg\min_{\ell \neq \ell_t} d(\ell, x_t, y_{t,\ell}), & \text{(b)}, \\ \{1, 2, \ldots, k\}, & \text{(c)}, \\ n_t \text{ values of } \ell \text{ that correspond the first } n_t \text{ smallest values of } d(\ell, x_t, y_{t,\ell}), & \text{(d)}. \end{cases} \tag{36}$$

Actually, choices (a), (b), and (c) are, respectively, equal to the special cases of choice (d) with $n_t = 1, 2, k$, respectively.

Moreover, with $p(x) = p_{h_x}(x)$ by Eq. (4) and a mild assumption that $\sum_{t,\ell \in L_t} q(x_t|y_{t,\ell}, \ell) q(y_{t,\ell}|\ell) q(\ell) = \sum_{t,\ell \in L_t} p_{h_x}(x_t) q(y_{t,\ell}|\ell) q(\ell)$, it is similar to Eq. (17) that $H(x_t, y_{t,\ell})$ and z_q have the following detailed form

$$H(x_t, y_{t,\ell}) = \int G(x|x_t, h_x^2 I) \ln[q(x|y_{t,\ell}, \ell) q(y_{t,\ell}|\ell) q(\ell)] \mathrm{d}x$$

$$\approx \ln[q(x_t|y_{t,\ell}) q(y_{t,\ell}|\ell) q(\ell)] + 0.5 h_x^2 \operatorname{Tr}[\Pi^x_{x|y,\ell}],$$

$$z_q = z_{h_x^N} = \sum_{t,\ell \in L_t} p_{h_x}(x_t) q(y_{t,\ell}|\ell) q(\ell),$$

$$\Pi^x_{x|y,\ell} = \frac{\partial^2 \ln q(x|y,\ell)}{\partial x \partial x^{\mathrm{T}}}. \tag{37}$$

In general, the Hessian matrix $\Pi^x_{x|y,\ell}$ maybe a function of either or both of x, y. In the rest of this paper, we consider a typical case that

$$q(x|y,\ell) = G(x|A_\ell y + c_\ell, \Sigma_\ell), \text{ and thus } \Pi^x_{x|y,\ell} = -\Sigma_\ell^{-1}. \tag{38}$$

It follows from Eqs. (33), (35) and (37) that z_q in $\min_y d(\ell, x, y)$ and $\min_\ell d(\ell, x, y_\ell(x))$ has no actual effect on a particular outcome of the minimization, since z_q does not vary with ℓ, x, y. However, keeping $\ln z_q$ as a part of $d(\ell, x, y)$ is conceptually useful. Since $q(\ell)$ is a δ density and either or both of $q(x|y,\ell)$, $q(y|\ell)$ may be δ density, the first term of $d(\ell, x, y)$ will contain an infinite term $-\ln \delta(0)$. Correspondingly, $\ln z_q$ will also contain a term $\ln \delta(0)$ to cancel the infinite term.

Based on Eqs. (34), (35) and (37), the harmony parameter learning equation (26) can be implemented by an adaptive algorithm as shown Table 1, which can be used for implementing either parameter learning with automatic model selection via Step 3 or only parameter learning by skipping Step 3 with model selection made after learning by Eq. (28). Also, instead of Step 6, we can alternatively let h_x start at an initial value and then gradually reduce to zero, in analog to the procedure of simulated annealing (Kirkpatrick, Gelatt, & Vecchi, 1983).

2.2.2. BI-architecture

In implementing the learning by Table 1, the task of getting $y_{t,\ell} = y_\ell(x_t)$ by Eq. (33) is still quite computational expensive since it is an optimization problem that should be made as each sample x_t comes. This problem, together with the problem of local minimum by Eq. (24), can be solved by considering $p(y, \ell|x) = p(y|x,\ell) p(\ell|x)$ in an appropriate parametric structure. Several typical cases are discussed as follows.

(a) *Deterministic $p(y|x,\ell)$ model*. A structure $p(y|x,\ell)$, that shares the least complexity nature of Eq. (24) but is easy to compute y_t for each x_t, is a deterministic model

$$p(y|x,\ell) = \delta(y - y_\ell(x)), \qquad y_\ell(x) = f_\ell(x|\theta_{y|x,\ell}). \tag{39}$$

Substituting it in Eq. (23), with $p(\ell|x)$ being structure-free, we can get the rest of Eq. (33) again.

Thus, we have $H(p\|q)$ in the same format as in Eqs. (34), (35) and (37), except two modifications in Table 1. First, $y_{t,\ell} = y_\ell(x_t)$ is now given by Eq. (39). Second, the parameter $\theta_{y|x,\ell}$ should also be updated, for which Step 5(c) of Table 1 is modified into:

$$\text{Step 5(c)}: \qquad \tau^{\text{new}} = \tau^{\text{old}} + 1,$$

$$\theta^{\text{new}}_{y|x,\ell} = \theta^{\text{old}}_{y|x,\ell} + \gamma_{t,\ell} \nabla_{\theta_{y|x,\ell}} [\ln q(y_l(x_t)|\ell) + \ln q(x_t|y_l(x_t), \ell)],$$

$$\text{subject to } y_l(x_t) = f_\ell(x|\theta_{y|x,\ell_t}). \tag{40}$$

(b) *Noisy $p(y|x,\ell)$ model*. The deterministic equation (39) does solving the problem of computing integrals because of the direct mapping $y_\ell(x) = f_\ell(x|\theta_{y|x,\ell})$. But it has no consideration on regularizing the WTA competition Eq. (24). When y is from a continuous density, it is similar to Eq. (4) that another measure for regularizing the WTA competition would be

$$p(y|x,\ell) = G(y|y_\ell(x), h_y^2 I), \qquad y_\ell(x) = f_\ell(x|\theta_{y|x,\ell}), \tag{41}$$

with h_y^2 acting as another smoothing parameter. In this

Table 1
An adaptive learning procedure for B-architecture

	Set $z_q(t)=0$, $\alpha_\ell=1/k$, $\tau=1$. $\theta_{y\|\ell}$ denotes parameters in $q(y\|\ell)$. Also $q(x\|y,\ell)=G(x\|A_{\ell y}+c_\ell,\Sigma_\ell)$
Step 1	$y_{t,\ell}=y_\ell(x_t)$ and $\ell_t=\ell(x_t)$ by Eq. (33), $e_{t,\ell}=x_t-A_\ell y_{t,\ell}-c_\ell$
Step 2	$z_q(t)=z_q(t-1)+\begin{cases}\sum_{\ell\in L_t} G(e_{t,\ell}\|0,\Sigma_\ell)q(y_{t,\ell}\|\ell)\alpha_\ell, & \text{(a) for normalization learning,}\\ \sum_{\ell\in L_t} p_{h_x}(x_t)q(y_{t,\ell}\|\ell)\alpha_\ell, & \text{(b) for data smoothing learning}\end{cases}$
	$p_t(\ell)=\bar{\delta}_{\ell,\ell_t},\ \gamma_{t,\ell}=\gamma_0\left[\frac{p_t(\ell)}{\tau}-I(l\in L_t)\gamma^d_{t,\ell}\right]$
	$I(\ell\in L_t)=\begin{cases}1 & \text{for } \ell\in L_t,\\ 0, & \text{otherwise}\end{cases}$ $\quad \gamma^d_{t,\ell}=\begin{cases}\frac{G(e_{t,\ell}\|0,\Sigma_\ell)q(y_{t,\ell}\|\ell)\alpha_\ell}{z_q(t)}, & \text{(a) for normalization learning,}\\ \frac{p_{h_x}(x_t)q(y_{t,\ell}\|\ell)\alpha_\ell}{z_q(t)}, & \text{(b) for data smoothing learning,}\\ 0, & \text{(c) for empirical learning}\end{cases}$
Note	$\gamma_0>0$ is a constant step size that may be different for updating different sets of parameters.
Step 3	Skip this step for the two-phase style implementation, otherwise update.
	$\alpha^{\text{new}}_\ell=\frac{e^{\alpha^{\text{new}}_\ell}}{\sum_{j=1}^k e^{\alpha^{\text{new}}_j}}$, $\quad \alpha^{\text{new}}_\ell=\alpha^{\text{old}}_\ell+\gamma_{t,\ell}\begin{cases}1-\alpha^{\text{old}}_\ell, & \ell=\ell_t,\\ -\alpha^{\text{old}}_\ell & \text{otherwise}\end{cases}$
	If $\alpha^{\text{new}}_\ell\to 0$, we discard the corresponding cluster ℓ according to Eq. (31).
Step 4	$\theta^{\text{new}}_{y\|\ell}=\theta^{\text{old}}_{y\|\ell}+\gamma_{t,\ell}\nabla_{\theta_{y\|\ell}}\ln q(y_{t,\ell}\|\ell)$ (which is subject to the constraint Eq. (27) for the two-phase style implementation)
Step 5(a)	$c^{\text{new}}_\ell=c^{\text{old}}_\ell+\gamma_{t,\ell}e_{t,\ell}$, $\quad A^{\text{new}}_\ell=A^{\text{old}}_\ell+\gamma_{t,\ell}e_{t,\ell}y^{\text{T}}_{t,\ell}$, $e_{t,\ell}$ is given in Step 1.
Step 5(b)	$\Sigma^{\text{new}}_\ell=S^{\text{new}}_\ell S^{\text{new T}}_\ell$, $S^{\text{new}}_\ell=S^{\text{old}}_\ell+\gamma_{t,\ell}G^{\text{old}}_{\Sigma_\ell}S^{\text{old}}_\ell$, $G_{\Sigma_\ell}=\Sigma_\ell^{-1}[e_{t,\ell}e^{\text{T}}_{t,\ell}+h_x^2 I]\Sigma_\ell^{-1}-\Sigma_\ell^{-1}$
Step 5(c)	$\tau^{\text{new}}=\tau^{\text{old}}+1$
Note	(a) The updating direction for A_ℓ, c_ℓ comes from the gradient direction multiplied by a positive definite matrix Σ_ℓ. (b) The updating on Σ_ℓ guarantees that it keeps a positive definite matrix even for the case $\gamma_{t,\ell}<0$
Step 6	Set $h_x=0$, unless for data-smoothing learning, we update $h^{\text{new}}_x=h^{\text{old}}_x+\gamma_0 g_x(h_x)$
	$g_x(h_x)=\frac{d}{h_x}-h_x\pi^{x\ \text{new}}_q-\frac{dh^{2\ \text{new}}_{x,0}}{h^3_x}$, with $\pi^x_q=\sum_{\ell=1}^k\alpha_\ell\text{Tr}[\Sigma_\ell^{-1}]$
	$h^2_{x,0}=\frac{1}{d}\sum_{t,t'}p_{t,t'}\|x_t-x_{t'}\|^2$, $p_{t,t'}=\frac{\sum_{\ell\in L_t}\exp\left(-0.5\frac{\|x_t-x_{t'}\|^2}{h^2_x}\right)q(y_{t,\ell}\|\ell)q(\ell)}{z_q(t)}$

Table 2
Modifications on adaptive learning procedure for BI-architecture

Table 1 is implemented with the following modifications:

(a) Step 2 is simplified into $\gamma_{t,\ell} = \gamma_0(p_t(\ell)/\tau)$

(b) In Step 5(a), update $A_\ell^{\text{new}} = A_\ell^{\text{old}} + \gamma_{t,\ell}(e_{t,\ell}y_{t,\ell}^{\mathrm{T}} - A_\ell^{\text{old}})$
In Step 5(b), G_{Σ_ℓ} is given by $G_{\Sigma_\ell} = \Sigma_\ell^{-1}[e_{t,\ell}e_{t,\ell}^{\mathrm{T}} + h_x^2 I + h_y^2 A_\ell^{\mathrm{T}} A_\ell]\Sigma_\ell^{-1} - \Sigma_\ell^{-1}$

(c) Step 5(c) is modified consisting of $\tau^{\text{new}} = \tau^{\text{old}} + 1$ and $\theta_{y|x,\ell}^{\text{new}} = \theta_{y|x,\ell}^{\text{old}} + \gamma_{t,\ell}\nabla_{\theta_{y|x,\ell}}[\ln q(y_\ell(x_t)|\ell) + \ln q(x_t|y_\ell(x_t),\ell) + 0.5h_y^2\mathrm{Tr}[\Pi_{y,\ell}^y]]$, subject to $y_\ell(x_t) = f_{\ell_t}(x|\theta_{y|x,\ell_t})$

(d) Step 6 is replaced by $h_x^{\text{new}} = h_x^{\text{old}} + \gamma_0 g_x(h_x)$, $g_x(h_x) = (d/h_x) - h_x\pi_q^{x\ \text{new}} - (dh_{x,0}^{2\ \text{new}}/h_x^3)$

$$\pi_q^x = \sum_{\ell=1}^{k}\alpha_\ell\mathrm{Tr}[\Sigma_\ell^{-1}],\ h_{x,0}^2 = \frac{1}{d}\sum_{t,t'}p_{t,t'}\|x_t - x_{t'}\|^2,\ p_{t,t'} = \frac{\exp\left(-0.5\frac{\|x_t - x_{t'}\|^2}{h_x^2}\right)\sum_{\ell\in L_t}\frac{q(\ell)}{(2\pi h_y^2)^{m_\ell}}}{z_q}$$

$$h_y^{\text{new}} = h_y^{\text{old}} + \gamma_0 g_y(h_y),\ g_y(h_y) = -h_y\pi_q^{y\ \text{new}} - \frac{\sum_{t,\ell\in L_t}p_{h_x}(x_t)\frac{m_\ell q(\ell)}{(2\pi h_y^2)^{m_\ell}}}{h_y z_q},\ \pi_q^y = \sum_{\ell=1}^{k}\alpha_\ell\mathrm{Tr}[A_\ell^{\mathrm{T}}\Sigma_\ell^{-1}A_\ell]$$

case, $H(x_t, y_{t,\ell})$ in Eq. (37) should be modified into

$$H(x_t, y_{t,\ell}) \approx \ln[q(x_t|y_{t,\ell},\ell)q(y_{t,\ell}|\ell)q(\ell)] + 0.5h_x^2\ \mathrm{Tr}[\Pi_{x|y,\ell}^x] + 0.5h_y^2\ \mathrm{Tr}[\Pi_{x|y,\ell}^y + \Pi_{y,\ell}^y],$$

$$\Pi_{x|y,\ell}^y = \frac{\partial^2\ln q(x|y,\ell)}{\partial y\partial y^{\mathrm{T}}} = -A_\ell^{\mathrm{T}}\Sigma_\ell^{-1}A_\ell,$$

$$\Pi_{y,\ell}^y = \frac{\partial^2\ln q(y|\ell)}{\partial y\partial y^{\mathrm{T}}},$$

$$z_q = z_p = \sum_{t,\ell\in L_t}p_{h_x}(x_t)G(y_{t,\ell}|y_{t,\ell},h_y^2 I)q(\ell) = \sum_t p_{h_x}(x_t)\sum_{t,\ell\in L_t}\frac{q(\ell)}{(2\pi h_y^2)^{0.5m_\ell}}. \quad (42)$$

That is, we got an extra term $0.5h_y^2\ \mathrm{Tr}[\Pi_{x|y,\ell}^y + \Pi_{y,\ell}^y]$ and have a new z_q.

With Eq. (42) taking the place of Eq. (37), Table 1 should be modified as described in Table 2. Again, instead of Step 6, we can alternatively let h_x, h_y starting at an initial value and then gradually reducing to zero in analog to the simulated annealing procedure.

(c) *Bayesian $p(\ell|x)$ model*. Instead of being free and then decided by Eq. (33), $p(\ell|x)$ can be alternatively a parametric structure in the following Bayesian inverse

$$p(\ell|x_t) = \sum_{j=1}^{k}p_t(j)\delta(\ell - j),$$
$$p_t(\ell) = \frac{q(x_t|y_{t,\ell},\ell)q(y_{t,\ell}|\ell)\alpha_\ell}{\sum_\ell q(x_t|y_{t,\ell},\ell)q(y_{t,\ell}|\ell)\alpha_\ell}. \quad (43)$$

Again, we have $H(p\|q)$ in the same format as in Eqs. (34), (35) and (37), and we can still implement learning as in Table 1. The only exception is that $p_t(\ell)$ is now given by Eq. (43) such that the WTA competition on ℓ_t is now relaxed into the soft weighting by this $p_t(\ell)$.

2.2.3. F-architecture

Instead of modeling how x_t is generated by an appropriate model in either a B-architecture or a BI-architecture, the task of a F-architecture is mapping x_t to a pre-specified representation form with the scales $\mathbf{k} = \{\{m_\ell\}, k\}$ fixed. Thus, model selection is no longer needed, and we only consider parameter learning.

Putting $p(x) = p_0(x)$ and $p(y|x,\ell)$ given by Eq. (39) into Eq. (23), we get

$$H(p,q) = \frac{1}{N}\sum_{t=1}^{N}\sum_\ell\int\delta(y - f_\ell(x_t|\theta_{y|x,\ell}))p_t(\ell) \times\ln[q(x_t|y,\ell)q(y|\ell)\alpha_\ell]\mathrm{d}y - \ln z_q$$
$$= \int\sum_\ell p(y,\ell)\sum_{t=1}^{N}\delta(y - f_\ell(x_t|\theta_{y|x,\ell}))\frac{p_t(\ell)}{N}/p(y,\ell) \times\ln[q(x_t|y,\ell)q(y|\ell)\alpha_\ell]\mathrm{d}y - \ln z_q. \quad (44)$$

When z_q is irrelevant to $q(x_t|y,\ell)$, making max $H(p,q)$ with

respect to $q(x_t|y,\ell)$ results in

$$q(x_t|y,\ell) = \delta(y - f_\ell(x_t|\theta_{y|x,\ell}))\frac{p_t(\ell)}{N}/p(y,\ell),$$

$$p(y,\ell) = \frac{1}{N}\sum_{t=1}^{N}\delta(y - f_\ell(x_t|\theta_{y|x,\ell}))p_t(\ell),$$

$$H(p,q) = \bar{H}(p,q) + D - H_\ell - \ln z_q - \ln N,$$

$$H_\ell = -\frac{1}{N}\sum_{t=1}^{N}\sum_{\ell} p_t(\ell)\ln p_t(\ell),$$

$$\bar{H}(p,q) = \int\sum_{\ell} p(y,\ell)\ln\frac{q(y|\ell)\alpha_\ell}{p(y,\ell)}\,\mathrm{d}y$$

$$= \frac{1}{N}\sum_{t=1}^{N}\sum_{\ell} p_t(\ell)\int\delta(y - f_\ell(x_t|\theta_{y|x,\ell}))\ln\frac{q(y|\ell)\alpha_\ell}{p(y|\ell)|p(\ell)}\,\mathrm{d}y,$$

$$D = \frac{1}{N}\sum_{t=1}^{N}\sum_{\ell}\int\delta(y - f_\ell(x_t|\theta_{y|x,\ell}))p_t(\ell)\ln\,\delta(y - f_\ell(x_t|\theta_{y|x,\ell}))\mathrm{d}y$$

$$= \ln\,\delta_y(0). \tag{45}$$

Where we have $p(y,\ell) = p(y|\ell)p(\ell)$ with

$$p(\ell) = \int p(y,\ell)\mathrm{d}y = \sum_j \beta_j\delta(\ell - j), \qquad \beta_\ell = \frac{1}{N}\sum_{t=1}^{N} p_t(\ell) \tag{46}$$

Thus, we get that y comes from a Gaussian $p_t(y|\ell) = \lim_{h_x\to 0} G(y|y_{t,\ell}, W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)h_x^2 I)$ and

$$p(y|\ell) = \lim_{h_x\to 0}\frac{1}{N}\sum_{t=1}^{N} G\big(y|y_{t,\ell}, W_\ell(x_t)W_\ell^{T}(x_t)h_x^2 I\big). \tag{48}$$

Also, when $y_{t,\ell} = f_\ell(x_t|\theta_{y|x,\ell})$ is one-to-one for $t = 1,\ldots,N$, it follows from $y' = [W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)]^{-0.5}y$ that

$$p(y_{t,\ell}|\ell) = \lim_{h_x\to 0}\frac{G(y'_{t,\ell}|y'_{t,\ell}, h_x^2 I)}{N|W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)|^{0.5}} = \frac{\delta_y(0)}{N|W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)|^{0.5}}. \tag{49}$$

Putting Eqs. (48) and (49) into $\bar{H}(p,q)$ in Eq. (45), we get

$$\bar{H}(p,q) = \frac{1}{N}\sum_{t=1}^{N}\sum_{\ell} p_t(\ell)\{0.5\,\ln|W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)| + \ln q(y_{t,\ell}|\ell)\} - \ln\,\delta_y(0) + \ln N - \sum_{\ell}\beta_\ell\,\ln\frac{\beta_\ell}{\alpha_\ell},$$

$$H(p,q) = \frac{1}{N}\sum_{t=1}^{N}\sum_{\ell} p_t(\ell)\{0.5\,\ln|W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)| + \ln q(y_{t,\ell}|\ell)\} + \frac{1}{N}\sum_{t=1}^{N}\sum_{\ell} p_t(\ell)\ln p_t(\ell) - \sum_{\ell}\beta_\ell\,\ln\frac{\beta_\ell}{\alpha_\ell} - \ln z_q. \tag{50}$$

When z_q is irrelevant to $q(\ell)$, making $\max H(p,q)$ with respect to α_ℓ and $p_t(\ell)$, we have

$$\alpha_\ell = \beta_\ell = \frac{1}{N}\sum_{t=1}^{N} p_t(\ell), \qquad p_t(\ell) = \bar{\delta}_{\ell,\ell_t}, \qquad \ell_t = \arg\max_{\ell}\big[|W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)|^{0.5}q(y_{t,\ell}|\ell)\big],$$

$$H(p,q) = \frac{1}{N}\sum_{t=1}^{N}\sum_{\ell} p_t(\ell)\ln\big[|W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)|^{0.5}q(y_{t,\ell}|\ell)\big] - \ln z_q,$$

$$z_q = \begin{cases} 1, & \text{for empirical learning,} \\ \sum_{t,\ell\in L_t}|W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)|^{0.5}q(y_{t,\ell}|\ell), & \text{for normalization learning.}\end{cases} \tag{51}$$

and $p(y|\ell) = p(y,\ell)/p(\ell)$. More concisely, by definition we also have $p(y|\ell) = \int\delta(y - f_\ell(x|\theta_{y|x,\ell}))p_0(x)\mathrm{d}x = (1/N)\times\sum_{t=1}^{N} p_t(y|\ell)$ and $p_t(y|\ell) = \lim_{h_x\to 0}\int\delta(y - f_\ell(x|\theta_{y|x,\ell}))G(x|x_t, h_x^2 I)\mathrm{d}x$. In other words, $p_t(y|\ell)$ can be regarded as obtained from $\lim_{h_x\to 0} G(x|x_t, h_x^2 I)$ via a mapping $y = f_\ell(x|\theta_{y|x,\ell})$ in a neighbor of each x_t. Approximately, we consider the linear part $y - y_{t,\ell} = W_\ell(x_t)(x - x_t)$ of the mapping with

$$W_\ell(x) = \frac{\partial f_\ell(x|\theta_{y|x,\ell})}{\partial x^{\mathrm{T}}}, \qquad y_{t,\ell} = f_\ell(x_t|\theta_{y|x,\ell}). \tag{47}$$

Instead of the above WTA competition on ℓ_t, $p_t(\ell)$ can also be given by the Bayesian inverse as follows

$$p_t(\ell) = \frac{|W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)|^{0.5}q(y_{t,\ell}|\ell)}{\sum_\ell|W_\ell(x_t)W_\ell^{\mathrm{T}}(x_t)|^{0.5}q(y_{t,\ell}|\ell)}. \tag{52}$$

In addition to Eq. (51), we can also get an alternative formulation via simply setting $q(x|y,\ell) = p_0(x)$ such that the harmony learning equation (23) in normalization

implementation leads to

$$\max H(p,q),$$

$$H(p,q)=\frac{1}{N}\sum_{t=1}^{N}\sum_{\ell}p_t(\ell)\ln[q(y_{t,\ell}|\ell)\alpha_\ell]-\ln z_q,$$
$$p_t(\ell)=\bar{\delta}_{\ell,\ell_t},\qquad \ell_t=\arg\max_{\ell}[q(y_{t,\ell}|\ell)\alpha_\ell], \tag{53}$$
$$z_q=\sum_{t=1}^{N}\sum_{\ell\in L_t}q(y_{t,\ell}|\ell)\alpha_\ell.$$

Here, we cannot simply set $z_q=1$ otherwise $H(p,q)$ can be arbitrarily large by rescaling the components of $y_{t,\ell}=f_\ell(x_t|\theta_{y|x,\ell})$ when y is real. With z_q given in Eq. (53), a rescaling effect of the first term of $H(p,q)$ is balanced by the corresponding rescaling effect of $-\ln z_q$ such that the value of $H(p,q)$ remains invariant to a rescaling of y. However, we can set $z_q=1$ by making max $H(p,q)$ subject to the following constraint

$y_{t,\ell}$ has a unit variance on each of its components.

Finally, instead of the WTA competition in Eq. (53), $p_t(\ell)$ can again be given by the Bayesian inverse equation (43). In this case, it is simplified into

$$p_t(\ell)=q(y_{t,\ell}|\ell)\alpha_\ell/\sum_{\ell}q(y_{t,\ell}|\ell)\alpha_\ell. \tag{55}$$

2.3. Least divergence versus best harmony

2.3.1. Best harmony vs. least Kullback divergence

Putting $p(u)=p(y|x,\ell)p(\ell|x)p(x)$, $q(u)=q(x|y,\ell)q(y|\ell)q(\ell)$ into Eq. (6), we have

$$\min_{\theta} D_K(\theta),$$

$$D_K(\theta)=D_K(p\|q)=\sum_{\ell}\int p(y|x,\ell)p(\ell|x)p(x)\ln\frac{p(y|x,\ell)p(\ell|x)p(x)}{q(x|y,\ell)q(y|\ell)q(\ell)}\,\mathrm{d}x\,\mathrm{d}y. \tag{56}$$

As discussed in Section 2.1.2, the relation between min $D_K(p\|q)$ and max $H(p\|q)$ closely depends on the structure of the Yang machine $p(u)=p(y|x,\ell)p(\ell|x)p(x)$, which further depends on the combination of the structure of each of $p(y|x,\ell)$, $p(\ell|x)$, and $p(x)$. Specifically, min $D_K(p\|q)$ and max $H(p\|q)$ become equivalent or very close on special situations that are featured by the following structures of $p(y|x,\ell)$, $p(\ell|x)$, and $p(x)$:

- Similar to the combination of cases (a) and (b) discussed at the end of Section 2.1.2, we have $E_p=-\int p(u)\ln p(u)\mathrm{d}u+\ln z_p=\ln N$, when $p(x)=p_0(x)$ by Eq. (5), $p(y|x,\ell)$ by Eq. (39), and $p(\ell|x)=\delta(\ell-\ell_t)$ with unknown ℓ_t. In this case, we get Eq. (20) again. That is, min $D_K(p\|q)$ becomes equivalent to max $H(p\|q)$.
- With $p(x)=p_{h_x}(x)$ by Eq. (4) in the above case, we encounter a combination of cases (a), (b), and (c) as discussed at the end of Section 2.1.2 and get Eq. (21) with

$$Z_q(h)=-\int p_{h_x}(x)\ln p_{h_x}(x)\mathrm{d}x=0.5\ln h_x^2-\frac{1}{N}\sum_{\tau=1}^{N}\ln\left[\sum_{t=1}^{N}\exp\left(-0.5\frac{\|x_t-x_\tau\|^2}{h_x^2}\right)\right]. \tag{57}$$

Moreover, if $p(y|x,\ell)$ is also replaced by Eq. (41), we encounter another combination of cases (a), (b), and (c) and get Eq. (21) again but with

$$Z_q(h)=-\sum_{\ell}\int p(y|x,\ell)p(\ell|x)p(x)\ln p(y|x,\ell)\mathrm{d}x\,\mathrm{d}y-\int p_{h_x}(x)\ln p_{h_x}(x)\mathrm{d}x=0.5\ln(2\pi h_y^2)\left(\sum_{\ell}m_\ell\right)+0.5\ln h_x^2-\frac{1}{N}\sum_{\tau=1}^{N}\ln\left[\sum_{t=1}^{N}\exp\left(-0.5\frac{\|x_t-x_\tau\|^2}{h_x^2}\right)\right]. \tag{58}$$

In both the cases, min $D_K(p\|q)$ and max $H(p\|q)$ are very close, with $Z_q(h)$ taking a similar role as $\ln z_q$.

The above discussed situation should not be a surprise to us because the design of Eq. (39) or (41) has already enforced the least complexity nature equation (24) of the harmony learning.

However, such types of equivalence do not apply generally. Recalling the previous discussion made after Eq. (6), the minimization of the Kullback divergence does have the matching nature by Eq. (9) but have not the least complexity nature by Eq. (10). In order to have both the natures, we can combine the use of minimizing the Kullback divergence for parameter learning and the use of maximizing the harmony measure for model selection. That is, we implement learning in a two-phase style that consists of making parameter learning by Eq. (56) and model selection by Eq. (28).

Further insights can be obtained on a B-architecture with $p(x)$ given by the empirical density equation (5). Making min $D_K(\theta)$ with respect to a structure free $p(y,\ell|x)$ results in

$$p(y,\ell|x)=\frac{q(x|y,\ell)q(y|\ell)\alpha_\ell}{\sum_\ell\int q(x|y,\ell)q(y|\ell)\alpha_\ell dy}, \tag{59}$$

which, instead of the WTA-δ density in Eq. (33), can compensate the local minimum effect due to the WTA competition by Eq. (24). This acts as a regularization to its counterpart harmony parameter learning.

Substituting Eq. (59) in Eq. (56), it follows that $\min D_K(p\|q)$ becomes equivalent to the ML learning on the marginal density in Eq. (1). Even on a Bi-architecture with $p(y|x,\ell)$ given by Eq. (39) but $p(\ell|x)$ being free, $\min D_K(p\|q)$ becomes equivalent to the ML learning on $\sum_\ell q(x|f_\ell(x|\theta_{y|x,\ell}),\ell)q(f_\ell(x|\theta_{y|x,\ell})|\ell)\alpha_\ell$ since making $\min D_K(\theta)$ with respect to $p(\ell|x)$ results in $p(\ell|x)$ given by Eq. (43).

As studied in (Xu, 1996, 1997, 1998b, 2000), from Eq. (56) we are easily lead to the EM algorithm for implementing the above ML learning. Here in this paper, one new thing that comes from the learning Eq. (56) with $p(x)$ by Eq. (4) is a Tikhonov-like regularization due to the role of smoothing parameter h_x. Another new thing is that model selection can be made by Eq. (28) in help of the harmony principle, which is a key feature that is not shared by only using the Kullback divergence for parameter learning.

Sharing the conventional two-phase style of "firstly ML parameter learning and subsequently model selection by a given criterion", we also encounter the problem of extensive computations on implementing Eq. (56) repeatedly by enumerating a large number of different scales $\mathbf{k}=\{k,\{m_\ell\}\}$. Moreover, in computing $p(y,\ell|x)$ by Eq. (59) we cannot avoid to compute the integral over y, which makes the EM algorithm usually difficult to implement, except for three computable special cases, namely, (a) $q(y|\ell)$ is a discrete density, (b) $q(y|\ell)$ is Bernoulli for a binary vector y, and (c) $q(y|\ell)$ is Gaussian.

In contrast, the best harmony learning by Table 1 has no such difficulty since the integral over y has been removed for Eqs. (34), (35) and (37), and the local minimum effect of Eq. (24) is compensated by either normalization learning or data-smoothing learning in the cases of a small size of samples. More interestingly, model selection is made automatically during the learning by Step 3 in Table 1.

In addition to the above discussed two phase style of combing the use of best harmony and least divergence, for those cases that $p(y,\ell|x)$ is computable we can also combine the two in parallel by replacing $H(\theta)$ in Eq. (26) with

$$H_K(\theta)=H(\theta)-\lambda D_K(\theta) \tag{60}$$

with $\lambda>0$ gradually reducing towards zero from a given initial value.

2.3.2. f-divergence and f-harmony

The Kullback divergence is a special case of the f-divergence

$$D_f(p,q)=\sum_{t=1}^{N}p_t f\left(\frac{q_t}{p_t}\right),\ \text{with}\ f(1)=0,\quad \frac{d^2f(u)}{d^2u}>0\ \text{on}\ [0,\infty], \tag{61}$$

at $f(x)=-\ln x$. The f-divergence was firstly studied by Csiszar in 1967 and a nice introduction can be found in Devroye et al. (1996). Similar to Eq. (18), it has the matching nature that $D_f(p,q)\geq 0$ and $D_f(p,q)=0$ if and only if $p_t=q_t$. However, it does not have the least complexity nature by Eq. (10). As discussed in Xu (1997), an appropriate function f will make learning become more robust on a set of samples with some outliers.

Moreover, it is similar to Eq. (19) that we have

$$D_f(p\|q)=\int p(u)f\left(\frac{q(u)/z_q}{p(u)/z_p}\right)du,\qquad z_q=\sum_{t=1}^{N}q(u_t),\quad z_p=\sum_{t=1}^{N}p(u_t). \tag{62}$$

Correspondingly, we can also extend the harmony measure equation (7) into

$$H_f(p\|q)=-\sum_{t=1}^{N}p_t f(q_t),\quad H_f(p\|q)=-\int p(u)f\left(\frac{q(u)}{z_q}\right)du. \tag{63}$$

Similar to Eq. (7), maximizing $H_f(p\|q)$ has the least complexity nature by Eq. (10) but unfortunately with the matching nature by Eq. (9) lost.

To implement both the two natures for harmony learning, we can combine the use of minimizing the f-divergence for parameter learning and maximizing the f-harmony measure for model selection. Specifically, putting $p(u)=p(y|x,\ell)p(\ell|x)p(x)$, $q(u)=q(x|y,\ell)q(y|\ell)q(\ell)$ into Eqs. (62) and (63), we implement

$$\min_\theta D_f(\theta),\qquad D_f(\theta)=D_f(p\|q). \tag{64}$$

With the obtained result θ^*, we select a best k^*, $\{m_\ell^*\}$ by

$$\min_{k,\{m_\ell\}} J(k,\{m_\ell\}),\qquad J(k,\{m_\ell\})=-H_f(\theta^*,\mathbf{k}),\quad H_f(\theta^*,\mathbf{k})=H_f(p\|q). \tag{65}$$

3. Gaussian mixture and Gaussian mixture-of-experts

In this section, we focus on the B-architecture with Gaussian components, i.e.

$$q(x|y,\ell)=G(x|A_\ell y+c_\ell,\Sigma_\ell)\ \text{as in Eq. (38)},\quad q(y|\ell)=G(y|\mu_\ell,\Lambda_\ell). \tag{66}$$

In this case, the B-architecture with $p(y,\ell|x)$ being structure

free or the BI-architecture with $p(y,\ell|x)$ being Gaussian are actually equivalent since they lead to the same solution via either the harmony learning or the Kullback learning. In the following, we consider a number of typical special cases.

3.1. MSE clustering, cluster number selection, and RPCL learning

When $q(y|\ell)=\delta(y)$, $\Sigma_\ell=\sigma^2 I$, we can get several interesting special results from Eqs. (34) and (35).

First, max $H(p,q)$ with $H(x_t,y_{t,\ell})$ by Eq. (37) becomes equivalent to making

$$\min \sigma^2, \quad \sigma^2=s^2+h_x^2, \qquad s^2=\frac{1}{N}\sum_{t=1}^{N}\sum_{\ell} p_t(\ell)\|e_{t,\ell}\|^2,$$

$$e_{t,\ell}=x_t-c_\ell, \qquad p_t(\ell)=\bar{\delta}_{\ell,\ell_t}, \tag{67}$$

$$\ell_t=\arg\min_{\ell}\left[0.5\frac{\|e_{t,\ell}\|^2}{\sigma^2}-\ln\alpha_\ell\right].$$

Particularly, when $\alpha_\ell=1/k$, ℓ_t is given by the minimum distance WTA competition

$$\ell_t=\arg\min_{\ell}\|x_t-c_\ell\|^2, \tag{68}$$

and Eq. (67) becomes equivalent to exactly the classical MSE clustering or VQ problem (Makhoul, Rpucos, & Gish, 1985). In this special case, the adaptive learning procedure in Table 1 will degenerate to the well known KMEAN algorithm (Xu, 1997, 2001b). However, this is not simply just a revisit of an old story, but provides the following two new results.

(a) *Smoothed k-selection criterion*. We can estimate h_x by Step 6 in Table 1 and then get a smoothed k-selection criterion $J(k)$, given by Item (1,2) in Table 3 for selecting k together with the use of the KMEAN algorithm. When $h_x=0$, it returns to Item (1,1) that was obtained firstly in Xu (1997) from max $H(p,q)$ with $z_q=1$, i.e. empirical learning. The smoothed one in Item (1,2) provides an improvement on Item (1,1), especially in the cases of a small size of samples, in help of a term h_x^2/σ^2 that prevents σ^2 becoming too small, which would cause over-fitting. As the number of samples increases, the role of the extra term reduces as h_x reduces.

(b) *Nonuniform KMEAN clustering and k-selection criteria*. When $\alpha_\ell\neq 1/k$, i.e. the population of each cluster is not equal, the WTA competition in Eq. (67) becomes different from Eq. (68) via biasing to a larger α_ℓ. Such a biasing is scaled by the value of σ^2. Thus, we get a nonuniform KMEAN algorithm by which σ^2 is computed as in Eq. (67). Moreover, α_ℓ is initialized at $1/k$ and then updated as follows

$$\alpha_\ell^{\text{new}}=\begin{cases}\dfrac{\alpha_\ell^{\text{old}}+\gamma_0}{1+\gamma_0}, & \ell=\ell_t,\\ \dfrac{\alpha_\ell^{\text{old}}}{1+\gamma_0}, & \text{otherwise,}\end{cases} \tag{69}$$

where $\gamma_0>0$ is a small constant step size.

Correspondingly, we also get its k-selection criterion in Table 3.

The biasing role of α_ℓ enhances the competition in favor to large clusters such that clusters can be automatically selected by discarding those α_ℓ being zero or too small. However, this enhanced competition may also cause the 'dead unit' problem (Xu et al., 1993), especially in the case of a small size of samples. Interestingly, the role of h_x^2 prevents σ^2 becoming too small and thus the biasing role of $q(\ell)$ is balanced to avoid the 'dead unit' problem.

Second, the normalization learning with max $H(p,q)$ via Eq. (35) can be implemented by the adaptive learning procedure in Table 1, which is now simplified into:

Step 1 : $p_t(\ell)$ by Eq. (67), update

$$z_q(t)=z_q(t-1)+\sum_{\ell\in L_t}\alpha_\ell G(x_t|c_\ell,\sigma_\ell^2 I),$$

Step 2 : $$\gamma_{t,\ell}=\gamma_0\left[\frac{p_t(\ell)}{\tau^{\text{new}}}-I(\ell\in L_t)\gamma_{t,\ell}^d\right],$$

$$\gamma_{t,\ell}^d=\alpha_\ell G(x_t|c_\ell,\sigma_\ell^2 I)/z_q(t),$$

Step 3 : the same as Step 3 in Table 1,

Step 4 : $$c_\ell^{\text{new}}=c_\ell^{\text{old}}+\gamma_{t,\ell}(x_t-c_\ell^{\text{old}}),$$

$$\sigma_\ell^{\text{new}}=\sigma_\ell^{\text{old}}+\gamma_{t,\ell}\sigma_\ell^{\text{old}}(\|x_t-c_\ell^{\text{old}}\|^2-\sigma_\ell^{2\text{old}}). \tag{70}$$

Corresponding to the four special cases of L_t in Eq. (36), we can get new results as follows:

(a) *Conscience competitive learning*. For L_t of case (a) in Eq. (36), we have

$$\gamma_{t,\ell}=\begin{cases}\gamma_0\left(\dfrac{p_t(\ell)}{\tau}-\dfrac{\gamma_f\gamma_d}{\sum_{t=1}^{\tau}\sum_{\ell\in L_t}\alpha_\ell G(x_t|c_\ell,\Sigma)}\right), & \text{for } \ell=\ell_t,\\ \gamma_f=\alpha_{\ell_t}, \qquad \gamma_d=G(x_t|c_{\ell_t},\Sigma), & \\ 0, & \text{otherwise,}\end{cases} \tag{71}$$

For each x_t, only the parameters of the winner ℓ_t will be updated. Particularly, the updating on c_ℓ would return to a simple competitive learning if $\gamma_{t,\ell}$ would become irrelevant

Table 3
Model selection criteria $\mathrm{Min}_k\ J(k)$ or $\mathrm{Min}_{k,m}\ J(k,m)$ on Gaussian mixture and Gaussian mixture of experts

	Empirical learning	Data smoothing learning	Normalization learning
MSE Clustering (a) k-means	$J(k) = 0.5d \ln \sigma^2 + \ln k$	$J(k) = 0.5d \ln \sigma^2 + \frac{0.5dh_x^2}{\sigma^2} + \ln k$	$J(k) = \ln z_p,\ z_p = \sum_{t=1}^{N}\sum_{l=1}^{k} \exp\left(-0.5\frac{\|x_t - m_l\|^2}{\sigma^2}\right)$
(b) Nonuniform k-means	$J(k) = 0.5d \ln \sigma^2 - \sum_{l=1}^{k} \alpha_l \ln \alpha_l$	$J(k) = 0.5d \ln \sigma^2 + \frac{0.5dh_x^2}{\sigma^2} - \sum_{l=1}^{k} \alpha_l \ln \alpha_l$	$J(k) = \ln\left\{\sum_{t=1}^{N}\sum_{l=1}^{k} \alpha_l \exp\left(-0.5\frac{\|x_t - m_l\|^2}{\sigma^2}\right)\right\} - \sum_{l=1}^{k} \alpha_l \ln \alpha_l$
Elliptic clustering and Gaussian mixture	$J(k) = 0.5\sum_{l=1}^{k} \alpha_l \ln\|\Sigma_l\| - \sum_{l=1}^{k} \alpha_l \ln \alpha_l$	$J(k) = 0.5\sum_{l=1}^{k} \alpha_l \ln\|\Sigma_l\| - \sum_{l=1}^{k} \alpha_l[\ln\alpha_l + 0.5h_x^2 \mathrm{Tr}[\Sigma_l^{-1}]]$	$J(k) = 0.5\sum_{l=1}^{k} \alpha_l \ln\|\Sigma_l\| - \sum_{l=1}^{k} \alpha_l \ln \alpha_l + \ln z_q,$ $z_q = \sum_{t=1}^{N}\sum_{l=1}^{k} \alpha_l G(x_t\|c_l, \Sigma_l)$
Subspace clustering	$J(k) = 0.5\sum_{l=1}^{k} \alpha_l\{d \ln \sigma_l^2 - 2 \ln \alpha_l + m_l[1 + \ln(2\pi)]\}$	$J(k, \{m_l\}) = 0.5\sum_{l=1}^{k} \alpha_l\left\{d\left(\ln \sigma_l^2 + \frac{h_x^2}{\sigma_l^2}\right) + m_l[1 + \ln(2\pi)] - 2 \ln \alpha_l\right\}$	$J(k, \{m_l\}) = \ln z_q + 0.5d\sum_{l=1}^{k} \alpha_l \ln \sigma_l^2 + 0.5\sum_{l=1}^{k} \alpha_l\{m_l[1 + \ln(2\pi)] - 2 \ln \alpha_l\},$ $z_q = \sum_{t=1}^{N}\sum_{l=1}^{k} \alpha_l \exp\left(-0.5\frac{\|x_t - A_l y_t - c_l\|^2}{\sigma^2}\right)/(2\pi)^{0.5m_l}(\sigma_l^2)^{0.5d}$
Gaussian ME	$J(k) = -\sum_{l=1}^{k} \alpha_l \ln \alpha_l + 0.5\sum_{l=1}^{k} \alpha_l(d \ln \sigma_l^2 + \ln\|\Lambda_l\|)$	$J(k) = 0.5\sum_{l=1}^{k} \alpha_l(d \ln \sigma_l^2 + \ln\|\Lambda_l\|) - \sum_{l=1}^{k} \alpha_l[\ln \alpha_l + 0.5h_y^2 \mathrm{Tr}[\Lambda_l^{-1}]]$	$J(k) = 0.5\sum_{l=1}^{k} \alpha_l(d \ln \sigma_l^2 + \ln\|\Lambda_l\|) - \sum_{l=1}^{k} \alpha_1 \ln \alpha_l + \ln z_q,$ $z_q = \sum_{t=1}^{N}\sum_{l=1}^{k} \alpha_l G(x_t\|A_l y_t + c_l, \Sigma_l) G(y_t\|\mu_l, \Lambda_l)$
RBF nets	Directly use the above $J(k)$ of Gaussian ME but with $\alpha_l = \|\Lambda_l\|^{0.5}/\sum_{l=1}^{k}\|\Lambda_l\|^{0.5}$		

Note: $J(k)$ in a specific block at the ith row and jth column is referred by Item (i,j).

to ℓ. However, $\gamma_{t,\ell}$ does relate to ℓ via γ_f, γ_d, which leads to two types of conscience competitive learning. Specifically, when $\alpha_\ell = 1/k$, $\gamma_{t,\ell}$ relates to ℓ only via γ_d in a way that the smaller is the distance $\|x_t - c_{\ell_t}\|^2$, the larger is the γ_d, and thus the weaker is the learning strength $\gamma_{t,\ell}$. That is, a de-learning is introduced by the winner's conscience, based on its winning degree (i.e. the likelihood or equivalently the distance $\|x_t - c_{\ell_t}\|^2$). In other words, we get a likelihood sensitive competitive learning (LSCL), which includes the distance sensitive competitive learning (DSCL) as a special case. Moreover, when $\alpha_\ell \neq 1/k$, $\gamma_{t,\ell}$ relates to ℓ via both γ_d and γ_f such that the de-learning rate also depends on the proportion or frequency $\gamma_f = \alpha_{\ell_t}$ of winning by ℓ_t. That is, the more it gets winning, and the more conscience will be introduced, which provides an alternative to the frequency sensitive competitive learning (FSCL) (Ahalt, Krishnamurty, Chen, & Melton, 1990), where the conscience is imposed during the competition for ℓ_t, instead of during learning. Therefore, via updating $\gamma_{t,\ell}$ as in Eq. (70), the updating $c_\ell^{\text{new}} = c_\ell^{\text{old}} + \gamma_{t,\ell}(x_t - c_\ell^{\text{old}})$ acts as a conscience competitive learning that includes FSCL and DSCL as special cases.

(b) *Generalized RPCL*. For L_t of case (b) in Eq. (36), we have

$$\gamma_{t,\ell} = \begin{cases} \gamma_0\left(\dfrac{1}{\tau} - \dfrac{\gamma_f\gamma_d}{z_q}\right), & \text{for } \ell = \ell_t, \\ -\gamma_0\dfrac{\gamma_f\gamma_d}{z_q}, & \text{for } \ell = \ell_r, \\ 0, & \text{otherwise.} \end{cases} \tag{72}$$

In this case, the updating $c_\ell^{\text{new}} = c_\ell^{\text{old}} + \gamma_{t,\ell}(x_t - c_\ell^{\text{old}})$ becomes conceptually equivalent to the RPCL that was previously proposed in Xu et al. (1993). As reviewed in Xu (2001a,b), RPCL has been shown to be an effective clustering algorithm in many applications, featured by that

The number of clusters is determined with extra units driven far away automatically. (73)

Here we see that this feature can be explained by the least complexity nature of the harmony learning, with the following new results:

- Eq. (72) provides a reasonable solution for the learning rate and de-learning rate that are used in Eq. (70). However, they were set heuristically in the original RPCL (Xu et al., 1993).
- A cluster can also be discarded when $\alpha_\ell \to 0$, which is easier to check than checking whether its corresponding mean vector has been driven far away as in Xu et al. (1993).
- Attempting to explain RPCL via heuristically modifying the EM algorithm into adaptive algorithms, it was found in Xu (1995) that the main behavior of RPCL is shared by a spectrum of RPCL variants via various combinations of competing and penalizing. Similarly, we can get from Eq. (70) a wide spectrum of RPCL algorithms in the case (d) of L_t, with solutions for the learning rates and de-learning rates.

In companion with the harmony learning, we can also get the k-selection criterion in Item (1,3) of Table 3, which can be used not only after various cases of RPCL learning, but also as an alternative to Item (1,2).

3.2. Elliptic clustering, cluster number selection, and elliptic RPCL

The MSE clustering is limited to data that consists of spherical clusters with a same $\Sigma_\ell = \sigma^2 I$. It can be further extended to clusters with more complicated shapes by considering different case of Σ_ℓ :

- *Spherical clustering*. Spherical clusters of different radius are considered with $\Sigma_\ell = \sigma_\ell^2 I$ that may be different for different ℓ.
- *Elliptic clustering*. Various elliptic clusters are considered with a general covariance matrix Σ_ℓ that may be different for different ℓ.

Clearly, a spherical clustering is a special case of elliptic clustering. So, we only consider elliptic clustering from both the data smoothing and normalization perspectives.

From the data smoothing perspective, $\max H(p,q)$ with $H(x_t, y_{t,\ell})$ by Eq. (37) leads to

$$p_t(\ell) = \bar{\delta}_{\ell,\ell_t}, \ell_t = \arg\min_\ell \big[0.5(x_t - c_\ell)^{\mathrm{T}}\Sigma_\ell^{-1}(x_t - c_\ell) + 0.5\ln|\Sigma_\ell| - \ln\alpha_\ell\big]. \tag{74}$$

Alternatively, making Kullback learning by Eq. (56), we get

$$p_t(\ell) = \alpha_\ell G(x_t|c_\ell, \Sigma_\ell)/\sum_{\ell=1}^{k}\alpha_\ell G(x_t|c_\ell, \Sigma_\ell). \tag{75}$$

Subsequently, we update parameters either in batch by

$$\alpha_\ell = \frac{1}{N}\sum_{t=1}^{N} p_t(\ell), \qquad c_\ell = \frac{1}{N\alpha_\ell}\sum_{t=1}^{N} p_t(\ell)x_t,$$
$$\Sigma_\ell = h_x^2 I + \frac{1}{N\alpha_\ell}\sum_{t=1}^{N} p_t(\ell)(x_t - c_\ell)(x_t - c_\ell)^{\mathrm{T}} \tag{76}$$

or adaptively by

$$\alpha_\ell \text{ updated by Eq. (69)}, \qquad e_{t,\ell} = x_t - c_\ell^{\text{old}},$$
$$c_\ell^{\text{new}} = c_\ell^{\text{old}} + \gamma_0 p_t(\ell)e_{t,\ell}, \tag{77}$$
$$\Sigma_\ell^{\text{new}} = [1 - \gamma_0 p_t(\ell)]\Sigma_\ell^{\text{old}} + \gamma_0 p_t(\ell)\big[h_x^2 I + e_{t,\ell}e_{t,\ell}^{\mathrm{T}}\big].$$

In both the case, empirical learning is included as a special

case with $h_x = 0$. Generally, we can estimate h_x by Step 6 in Table 1.

Particularly, we can get the following new results for elliptic clustering:

(a) *The smoothed EM algorithm, the hard-cut EM algorithm, and the smoothed hard-cut EM algorithm.* When $h_x = 0$, we get the popular EM algorithm on Gaussian mixture with Eq. (75) as its E-step and Eq. (76) as its M-step, and also get the hard-cut EM algorithm with its E-step given not by Eq. (75) but by Eq. (74) (Xu, 1997). When $h_x > 0$ is estimated at Step 6 in Table 1, we are further lead to either a smoothed EM algorithm or a smoothed hard-cut EM algorithm, which are more suitable to the cases of a small size of high dimensional samples. Moreover, for the smoothed hard-cut EM algorithm, the nature by Eq. (31) will be in action during learning with automatic model selection.

(b) *The k-selection criterion and its smoothed version.* Also, we can make the above learning with $\alpha_\ell = 1/k$ fixed, and then select k by using the criterion given in Item (3,2) of Table 3, which is an improvement on the criterion Item (3,1) that was previously obtained in Xu (1997).

From the normalization perspective, it follows from Table 1 that the adaptive algorithm in Eq. (70) is modified with not only $G(x_t|c_\ell, \sigma_\ell^2 I)$ in Steps 1 and 2 replaced by its general case $G(x_t|c_\ell, \Sigma_\ell I)$ but also the updating on σ_ℓ^2 in Step 4 replaced by Step (b) with $e_{t,\ell} = x_t - c_\ell^{\text{old}}$. Now, $h_x^2 > 0$ takes a role that avoids Σ_ℓ becoming singular.

Similar to the discussion made in Section 3.1, we can get further insights by considering the special cases of L_t in Eq. (36) as follows:

(a) *Conscience competitive learning*. For L_t of the case (a), an elliptic clustering is made by a conscience competitive learning that includes FSCL, LSCL, and DSCL as special cases.
(b) *Elliptic RPCL and variants*. For L_t of the case (b), we get an elliptic RPCL algorithm with a solution for the learning rate and de-learning rate by Eq. (70), and with automatic model selection featured by Eqs. (31) and (73). Moreover, we can get a wide spectrum of elliptic RPCL variants from L_t of the case (c) and the case (d).
(c) We can also get the k-selection criterion for elliptic clustering in Item (3,3) of Table 3.

3.3. Subspace clustering, structural RPCL, and dimension determination

In the above elliptic clustering, it needs to compute each covariance matrix Σ_ℓ, which not only is expensive in computational cost, but also does not work well in the cases of a small size of high dimensional samples since Σ_ℓ may become singular. This motivates to represent each cluster via a linear subspace instead of directly using Σ_ℓ. With this consideration, the local PCA learning has been proposed in Tipping and Bishop (1999) and Xu (1995). In this subsection, we re-elaborate it from the perspective of BYY system and harmony learning.

One way is to consider the decomposition (Xu, 2001b)

$$\Sigma_\ell = \phi_\ell^{\mathrm{T}} D_\ell^2 \phi_\ell + \sigma_\ell^2 I, \qquad \phi_\ell \phi_\ell^{\mathrm{T}} = I, \tag{78}$$

D_ℓ is a $m_\ell \times m_\ell$ diagonal matrix.

We modify Step 5 (b) in Table 1 such that Σ_ℓ is updated within the constraint equation (78).

First, considering to maximize $\ln G(x|c_\ell, \Sigma)$ with respect to D_ℓ, σ_ℓ via the variational analysis, it follows from $2\mathrm{Tr}[G_{\Sigma_\ell} \phi_\ell^{\mathrm{T}} \delta D_\ell D_\ell \phi_\ell]$ and $2\mathrm{Tr}[G_{\Sigma_\ell} \sigma_\ell \delta\sigma_\ell]$ that $\delta D_\ell = \mathrm{diag}[\phi_\ell G_{\Sigma_\ell} \phi_\ell^{\mathrm{T}}] D_\ell$ and $\delta\sigma_\ell = \sigma_\ell \mathrm{Tr}[G_{\Sigma_\ell}]$, with

$$G_{\Sigma_\ell} = \Sigma_\ell^{-1}(x_t - c_\ell)(x_t - c_\ell)^{\mathrm{T}} \Sigma_\ell^{-1} - \Sigma_\ell^{-1} \tag{79}$$

Second, it follows from $\phi_\ell \phi_\ell^{\mathrm{T}} = I$ that the solution of $\delta\phi_\ell \phi_\ell^{\mathrm{T}} + \phi_\ell \delta\phi_\ell^{\mathrm{T}} = 0$ must satisfy

$$\delta\phi_\ell = Z\phi_\ell + W(I - \phi_\ell^{\mathrm{T}} \phi_\ell),$$

W is any $m \times d$ matrix and $Z = -Z$ is an asymmetric matrix. (80)

Similarly, to maximize $\ln G(x|c_\ell, \Sigma)$ with respect to ϕ_ℓ, it follows from Eq. (80) and $\mathrm{Tr}[G_{\Sigma_\ell}(\delta\phi_\ell^{\mathrm{T}} D_\ell \phi_\ell + \phi_\ell^{\mathrm{T}} D_\ell \delta\phi_\ell)] = 2\mathrm{Tr}[G_{\Sigma_\ell} \phi_\ell^{\mathrm{T}} D_\ell \delta\phi_\ell]$ that maximizing $\mathrm{Tr}[G_{\Sigma_\ell} \phi_\ell^{\mathrm{T}} D_\ell Z \phi_\ell]$ and $\mathrm{Tr}[G_{\Sigma_\ell} \phi_\ell^{\mathrm{T}} D_\ell W(I - \phi_\ell^{\mathrm{T}} \phi_\ell)]$ results in

$$Z = D_\ell \phi_\ell G_{\Sigma_\ell} \phi_\ell^{\mathrm{T}} - \phi_\ell G_{\Sigma_\ell} \phi_\ell^{\mathrm{T}} D_\ell,$$

$$W = D_\ell \phi_\ell G_{\Sigma_\ell} (I - \phi_\ell^{\mathrm{T}} \phi_\ell),$$

$$\delta\phi_\ell = \begin{cases} W(I - \phi_\ell^{\mathrm{T}} \phi_\ell) = W, & \text{(a)}, \\ Z\phi_\ell, & \text{(b)}, \\ Z\phi_\ell + W, & \text{(c)}. \end{cases} \tag{81}$$

That is, under the constraint $\phi_\ell \phi_\ell^{\mathrm{T}} = I$, we can use anyone of the above three choices of $\delta\phi_\ell$ as the updating direction of ϕ_ℓ. Therefore, the updating on Σ_ℓ is replaced by

$$\phi_\ell^{\text{new}} = \phi_\ell^{\text{old}} + \gamma_{t,\ell} \delta\phi_\ell, \qquad \sigma_\ell^{\text{new}} = \sigma_\ell^{\text{old}} + \gamma_{t,\ell} \delta\sigma_\ell,$$

$$\delta\sigma_\ell = \sigma_\ell \mathrm{Tr}[G_{\Sigma_\ell}], \qquad D_\ell^{\text{new}} = D_\ell^{\text{old}} + \gamma_{t,\ell} \delta D_\ell, \tag{82}$$

$$\delta D_\ell = \mathrm{diag}[\phi_\ell G_{\Sigma_\ell} \phi_\ell^{\mathrm{T}}] D_\ell,$$

which seeks the subspace spanned by ϕ_ℓ and the corresponding variance structure by D_ℓ^2 and σ_ℓ.

During learning, automatic selection on clusters is in action, with not only the features of Eq. (73) and of Eq. (31), but also a new feature that σ_ℓ of an extra cluster will also be pushed towards zero due to the nature by Eq. (32) because $\sigma_\ell = 0$ if and only if the ℓ-th cluster becomes a δ density, which equivalently implies the corresponding $\alpha_\ell = 0$ except for a rare case that there are certain samples exactly located at c_ℓ. Therefore, at Step 3 in Table 1 we may also

use either of the following two strategies:

(a) If both $\sigma_\ell^{new} \to 0$, $\alpha_\ell^{new} \to 0$, we discard the corresponding cluster ℓ.
(b) If either of $\sigma_\ell^{new} \to 0$, $\alpha_\ell^{new} \to 0$ happens, we discard the corresponding cluster ℓ.

In the above discussion, the dimension m_ℓ of the subspace (equivalently the dimension of D_ℓ) are assumed to be given. If it is unknown, we can also solve it. Specifically, considering a B-architecture with Eq. (66) in the following special case

$$q(x|y,\ell) = G(x|A_\ell y + c_\ell, \sigma_\ell^2 I),$$
$$q(y|\ell) = G(y|\mu_\ell, I), \quad \text{subject to} \tag{83}$$
$$A_\ell = \phi_\ell^T D_\ell, \quad \phi_\ell, D_\ell \text{ as in Eq. (78)},$$

we can obtain criteria given by the 4th row of Table 3 for deciding both the number of clusters and the dimension of subspaces.

Moreover, for each ℓ we have $q(x|\ell) = \int G(x|A_\ell y + c_\ell, \sigma_\ell^2 I) G(y|\mu_\ell, I)\mathrm{d}y = G(x|c_\ell, \Sigma_\ell)$ with Σ_ℓ being same as in Eq. (78). That is, a Gaussian $G(x|c_\ell, \Sigma_\ell)$ is represented via a factor analysis model. From this perspective, we can implement learning by the adaptive learning procedure in Table 1 at the special case of Eq. (83), which leads to the following simplified form:

Step 1 : $y_{t,\ell} = (\sigma_\ell^2 I + A_\ell^T A_\ell)^{-1} A_\ell^T x_t + \mu_\ell$,

$\tau^{new} = \tau^{old} + 1$, $e_{t,\ell} = x_t - A_\ell y_{t,\ell} - c_\ell$,

$e_{t,\ell}^y = y_t - \mu_\ell^{old}$,

$$\ell_t = \arg\min_\ell \left[0.5\left(\ln \sigma_\ell^2 + \frac{\|e_{t,\ell}\|^2}{\sigma_\ell^2} + \|e_{t,\ell}^y\|^2 \right) - \ln \alpha_\ell \right],$$

Step 2 :

Same as Step 2 in Table 1 with Eq. (83) inserted,

Step 3 : Same as Step 3 in Table 1,

Step 4 : $\mu_\ell^{new} = \mu_\ell^{old} + \gamma_{t,\ell} e_{t,\ell}^y$,

Step 5 : $c_\ell^{new} = c_\ell^{old} + \gamma_{t,\ell} e_{t,\ell}$,

$D_\ell^{new} = D_\ell^{old} + \gamma_{t,\ell}\, \mathrm{diag}[\phi_\ell \Sigma_\ell^{-1} e_{t,\ell} y_{t,\ell}^T]$,

$\phi_\ell^{new} = \phi_\ell^{old} + \gamma_{t,\ell} \delta\phi_\ell$,

$\sigma_\ell^{new} = \sigma_\ell^{old} + \gamma_{t,\ell} \sigma_\ell^{old} (\|e_{t,\ell}\|^2 + h_x^2 - \sigma_\ell^{2old})$,

Step 6 : same as in Table 1. (84)

Specifically, $\delta\phi_\ell$ is obtained in a way similar to Eq. (80). That is, we can get

$$\delta\phi_\ell = \begin{cases} W, & \text{(a)}, \\ Z\phi_\ell, & \text{(b)}, \\ Z\phi_\ell + W, & \text{(c)}, \end{cases} \tag{85}$$
$$Z = D_\ell y_{t,\ell} e_{t,\ell}^T \Sigma_\ell^{-1} \phi_\ell^T - \phi_\ell \Sigma_\ell^{-1} e_{t,\ell} y_{t,\ell}^T D_\ell,$$
$$W = D_\ell y_{t,\ell} e_{t,\ell}^T \Sigma_\ell^{-1} (I - \phi_\ell^T \phi_\ell).$$

3.4. Gaussian ME and RBF nets

Next, we further make learning on a B-architecture with a set of paired samples $\{y_t, x_t\}$, instead of a set of samples for x only, i.e. we consider supervised learning.

Specifically, from Eq. (66) we can get

$$q(x|y) = \sum_{\ell=1}^k G(x|A_\ell y + c_\ell, \sigma_\ell^2) p(\ell|y),$$
$$p(\ell|y) = \frac{G(y|\mu_\ell, \Lambda_\ell)\alpha_\ell}{\sum_{\ell=1}^k G(y|\mu_\ell, \Lambda_\ell)\alpha_\ell}, \tag{86}$$

which is a typical case of the alternative mixture-of-expert model (Xu, 1998a; Xu et al., 1995) for mapping $y \to x$. On the average, we have the regression $E(x|y) = \sum_{\ell=1}^k p(\ell|y) \times (A_\ell y + c_\ell)$ weighted by the gate $p(\ell|y)$.

From Eq. (66) and noticing that y_t is already known for each x_t, it follows from Table 1 that we can get the following adaptive algorithm:

Step 1 :

$$\ell_t = \arg\min_\ell \left[0.5\left(\ln \sigma_\ell^2 + \frac{\|e_{t,\ell}\|^2}{\sigma_\ell^2} + \ln|\Lambda_\ell| + e_t^{yT} \Lambda_\ell^{-1} e_t^y \right) - \ln \alpha_\ell \right],$$

$e_{t,\ell} = x_t - A_\ell y_t - c_\ell$, $e_t^y = y_t - \mu_\ell^{old}$,

Step 2 : Same as in Table 1 with Eq. (86) inserted,

Step 3 : Same as in Table 1,

Step 4 : $\mu_\ell^{new} = \mu_\ell^{old} + \gamma_{t,\ell} e_t^y$, $\Lambda_\ell^{new} = \Gamma_\ell^{new} \Gamma_\ell^{new\,T}$,

$\Gamma_\ell^{new} = \Gamma_\ell^{old} + \gamma_{t,\ell} G_{\Sigma_\ell}^{old} S_\ell^{old}$,

$G_{\Sigma_\ell} = \Lambda_\ell^{-1} e_t^{y\,old} e_t^{y\,old\,T} \Lambda_\ell^{-1} - \Lambda_\ell^{-1}$,

$$\text{Step 5}: \quad A_\ell^{\text{new}} = A_\ell^{\text{old}} + \gamma_{t,\ell} e_{t,\ell}^{\text{old}} y_t^{\mathrm{T}},$$

$$c_\ell^{\text{new}} = c_\ell^{\text{old}} + \gamma_{t,\ell} e_{t,\ell}^{\text{old}},$$

$$\sigma_\ell^{\text{new}} = \sigma_\ell^{\text{old}} + \gamma_{t,\ell}\sigma_\ell^{\text{old}}\left(\|e_{t,\ell}^{\text{old}}\|^2 + h_x^2 - \sigma_\ell^{2\text{old}}\right),$$

$$\text{Step 6}: \quad \text{same as in Table 1,} \tag{87}$$

which actually acts as a general RPCL learning algorithm.

Specifically, Step 1 implements the coordinated competition (Xu, 1998a), which has different simplified forms for specific $q(x|y,\ell)$ and $q(y,\ell)$. For the case of Eq. (83), Step 1 becomes equivalent to a class of the shortest distance competition in various specific forms (Xu, 1998a).

Again, we can get different variants of the supervised learning for different choices of L_t in Eq. (36). We can also further modify the updating on Λ_ℓ via the orthogonal subspaces in a way similar to Eqs. (81) and (82).

Moreover, similar to RPCL clustering, model selection on an appropriate number of experts/basis-functions is made automatically during learning with k initialized at a large enough value. Furthermore, we can get criteria from Eq. (28) for selecting k, as given by Items (5,2) in Table 3, which are improved variants of Item (5,1), previously proposed for RBF net with empirical learning (Xu, 1998a).

Next, we consider a special case of Eq. (86) with the following constraint

$$\alpha_\ell = \sqrt{|\Lambda_\ell|}/\sum_{r=1}^{k}\sqrt{|\Lambda_r|}. \tag{88}$$

It follows from Eq. (86) that

$$E(x|y) = \frac{\sum_{\ell=1}^{k}(A_\ell y + c_\ell)e^{-0.5(y-\mu_\ell)^{\mathrm{T}}\Lambda_\ell^{-1}(y-\mu_\ell)}}{\sum_{\ell} e^{-0.5(y-\mu_\ell)^{\mathrm{T}}\Lambda_\ell^{-1}(y-\mu_\ell)}}, \tag{89}$$

which is exactly the Extended Normalized Gaussian RBF net (Xu, 1998a). Particularly, when $A_\ell = 0$ it reduces into the normalized RBF nets (Moody & Darken, 1989; Nowlan, 1990; Xu, 1998a; Xu et al., 1994).

In this case, the learning can be implemented again by Eq. (87) with Step 3 replaced with Eq. (88).

3.5. Parzen window density, kernel regression, and support vectors

Given a set of samples $\{x_t\}_{t=1}^N$, we consider a very special case of Eq. (66) with

$$k = N, \quad A_\ell = 0, \quad c_\ell = x_\ell, \qquad q(y|\ell) = \delta(y), \quad \Sigma_\ell = \lambda_\ell^2 I. \tag{90}$$

In this case, learning is made only on α_ℓ and λ_ℓ, which can be implemented either from the normalization learning perspective directly by Eq. (70) under the constraints (90) or from the data smoothing perspective by the following algorithm

$$\text{Step 1}: \quad p_t(\ell) \text{ given by Eq. (67) with } \sigma_\ell^2 = \lambda_\ell^2,$$

$$\text{Step 2}: \quad \text{update } \alpha_\ell \text{ by Eq. (69),}$$

$$\text{Step 3}: \quad \lambda^{\text{new}} = \lambda^{\text{old}} + \gamma_{t,\ell}\lambda^{\text{old}}\left(\|x_t - c_\ell\|^2 + h_x^2 - \lambda^{2\text{old}}\right),$$

$$\text{Step 4}: \quad \text{update } h_x \text{ by Step 6 in Table 1.} \tag{91}$$

Substituting Eq. (90) in Eq. (66), it follows that $q(x) = \sum_{t=1}^N \alpha_t G(x|x_t, \lambda^2 I)$. Particularly, we have

- When $\alpha_\ell = 1/N$, $q(x) = (1/N)\sum_{t=1}^N G(x|x_t, \lambda^2 I)$ becomes exactly the Parzen window density equation (4). But what is new here is that the smoothing parameter λ is also estimated during learning.
- When α_ℓ is unknown and estimated during learning by either Eq. (70) or (91), we can get a subset $L_{SV} = \{\tau:$ with each element $\alpha_\tau \neq 0$ or $\alpha_\tau > \varepsilon_0$ with ε_0 being a pre-specified small number$\}$, and approximately we have $q(x) = \sum_{t\in L_{SV}} \alpha_t G(x|x_t, \lambda^2 I)$, i.e. $q(x)$ can be estimated on a set of support vectors $\{x_t : t \in L_{SV}\}$.

Similarly, we also consider a special case of the RBF network equation (89) by adding a set of constraints that include (a) $k = N$ and $\mu_\ell = \mu_t = y_t$, (b) $\Lambda_\ell = \lambda^2 I$, and (c) $A_\ell = 0$ and $c_\ell = c_t = x_t$, resulting in

$$E(x|y) = \frac{\sum_{\ell\in L} x_\ell \alpha_\ell e^{-0.5\|y-y_\ell\|^2/\lambda^2}}{\sum_{\tau\in L}\alpha_\tau e^{-0.5\|y-y_\tau\|^2/\lambda^2}}, \tag{92}$$

which also brings us to three interesting cases:

- When $\alpha_\ell = 1/N$, as previously pointed out in Xu et al. (1994), it is actually Gaussian kernel regression that has been widely studied in the literature of statistics (Devroye et al., 1996). In contrast to RBF nets, the most salient feature of kernel regression is that there is no unknown parameters except that the smoothing parameter λ^2 needs to be pre-specified. Though many studies have been made in literature on getting the smoothing parameter λ^2, there are only theoretical upper bounds and how to estimating a best λ^2 still remains a challenge problem. Considering Eq. (87) in this special case, we have $x_\ell = x_t$ and $q(x_t|y_t,\ell) = G(x_t|x_t, \sigma^2 I) = 1$ with the role of σ^2 simply ignorable. Thus, the problem degenerates to estimate the smoothing parameter λ of the parzen window density $q(y)$ on the set $\{y_t\}_{t=1}^N$, which can be solved in the way same as the case of $q(x) = (1/N)\times \sum_{t=1}^N G(x|x_t, \lambda^2 I)$.
- When α_ℓ is unknown, again we will get a support vector

set L_{SV} via learning in the way same as the case of $q(x) = \sum_{t\in L_{SV}} \alpha_t G(x|x_t, \lambda^2 I)$. Then, we have

$$E(x|y) = \frac{\sum_{\ell\in L_{SVM}} x_\ell \alpha_\ell e^{-0.5\|y-y_\ell\|^2/\lambda^2}}{\sum_{\tau\in L_{SVM}} \alpha_\tau e^{-0.5\|y-y_\tau\|^2/\lambda^2}}. \quad (93)$$

- We can also take the role of σ^2 in consideration for an given appropriate σ^2. Setting $\tau = 1$, we can update α_t and λ^2 by

$$\text{Step 1}: \quad \tau^{\text{new}} = \tau^{\text{old}} + 1, \quad (94)$$

randomly taking a sample ε_t from $G(\varepsilon_t|0, \sigma^2 I)$,

$$\text{Step 2}: \quad p_t(\ell) = \bar{\delta}_{\ell,\ell_t},$$

$$\ell_t = \arg\min_\ell \left[\frac{\|\varepsilon_t\|^2}{\sigma^2} + \frac{\|y_t - y_\ell\|^2}{\lambda^2} - \ln\alpha_\ell\right],$$

$$\text{Step 3}: \quad \text{update } \alpha_\ell \text{ by Eq. (69)},$$

$$\text{Step 4}:$$

$$\lambda^{\text{new}} = \lambda^{\text{old}} + \frac{\gamma_0}{\tau^{\text{new}}}\lambda^{\text{old}}\left(\|y_t - y_\ell\|^2 + h_y^2 - \lambda^{2\text{old}}\right),$$

$$\text{Step 5}: \quad \text{update } h_y \text{ in the way same as updating } h_x$$

by Step 6 in Table 1.

After learning, we again get a regression by either Eq. (92) or (93). The variance σ^2 takes a role similar to the smoothing parameter h_y. However, the value of σ^2 needs to be pre-given but not able to be learned since we only have one output sample x_t that corresponds to an input sample y_t. In contrast, h_y can be learned via the way similar to Step 6 in Table 1. Also, we can simply set $h_y = 0$ and discard Step 5 to make an empirical implementation.

4. NonGaussian mixture and mixture of independent mapping

We further go beyond Gaussian mixture, with either or both of $q(x|y,\ell)$ and $q(y|\ell)$ being nonGaussian.

4.1. Bernoulli–Gaussian mixtures and structural clustering

We start to consider a B-architecture with

$$q(x|y,\ell) = G(x|A_\ell y + c_\ell, \sigma_\ell^2 I),$$
$$q(y|\ell) = \prod_{j=1}^{m_\ell}[q_{\ell,j}\delta(y^{(j)}) + (1 - q_{\ell,j})\delta(1 - y^{(j)})]. \quad (95)$$

From which we get a finite mixture $q(x) = \sum_\ell q(x|\ell)q(\ell)$ with each $q(x|\ell) = \sum_y q(x|y,\ell)q(y|\ell)$ itself consisting of 2^{m_ℓ} Gaussians mixed by $q(y|\ell)$.

To get an insight on the organization of such a $q(x|\ell)$, we consider the geometry of $A_\ell y + c_\ell$. We start at a m_ℓ dimensional hypercubic in the R^d space of x, with 2^{m_ℓ} vertices and each straight edge being a unit length. We can rotate it arbitrarily and locate it anywhere. Moreover, we also allow any deformation caused by extending or shrinking the lengths of the straight edges. Thus, by relocations, rotations, and such deformations, we can get a family of polyhedra and each polyhedra retains the same topology of the original m_ℓ dimensional hypercubic. We call it a hypercubic-induced polyhedra. Thus, $q(x|\ell)$ actually represents an organized cluster that consists of 2^{m_ℓ} Gaussians of zero mean and covariance $\sigma_\ell^2 I$, located at each vertex of such a hypercubic-induced polyhedra. So, the task of learning is a structural clustering problem that seeks clusters in every hypercubic-induced polyhedra.

Learning on the B-architecture Eq. (95) can be implemented as a special case of the adaptive learning procedure in Table 1. Specifically, a structure-free $p(y,\ell|x)$ is still determined by Eq. (33) with

$$d(\ell,x,y) = 0.5\frac{\|x - A_\ell y - c_\ell\|^2}{\sigma_\ell^2} + 0.5\ln\sigma_\ell^2 - \ln\alpha_\ell - \sum_{j=1}^{m_\ell}\left[y^{(j)}\ln q_{\ell,j} + (1 - y^{(j)})\ln(1 - q_{\ell,j})\right]. \quad (96)$$

With Eqs. (95) and (96) in Table 1, Step 4 takes the following detail form:

$$\text{Step 4}: \quad q_{\ell,j}^{\text{new}} = b_{\ell,j}^{\text{new}\ 2},$$

$$b_{\ell,j}^{\text{new}} = b_{\ell,j}^{\text{old}} + \gamma_{t,\ell} b_{\ell,j}^{\text{old}}(y_t^{(j)} - q_{\ell,j}^{\text{old}});$$

if either $q_{\ell,j}^{\text{new}} \to 0$ or $q_{\ell,j}^{\text{new}} \to 1$, discard the jth dimension

according to Eq. (32). (97)

Moreover, it follows from Eq. (38) that $\pi^x_{x|y,\ell} = -0.5\sigma_\ell^{-2} I$. Thus, the updating on Σ in Step 5 (b) is now simplified into $\sigma_\ell^{\text{new}} = \sigma_\ell^{\text{old}} + \gamma_{t,\ell}\sigma_\ell^{\text{old}}(\|e_{t,\ell}\|^2 + h_x^2 - \sigma_\ell^{2\text{old}})$. During learning, not only automatic selection on clusters is in action via Step 3, but also automatic determination on each dimension m_ℓ will be made via the above Step 4.

Alternatively, it follows from Eq. (28) that we can make model selection by the criterion given in Item 2.2 or Item 2.3 in Table 4, even simply by Item 2.1 for a large size of samples.

4.2. Structural clustering, local LMSER, and competitive P-ICA

As discussed in Section 2.2.2, the task of getting $y_{t,\ell}$ by

$\min_y\ d(\ell, x, y)$ is a quadratic discrete optimization problem that should be made for each sample x_t. This computing cost can be very expensive. In a BI-architecture, this cost can be avoided when $p(y|x, \ell)$ is given by a special case of Eq. (39) as follows

$$y_\ell(x) = s(\hat{y}_\ell), \quad \hat{y}_\ell = W_\ell x + d_\ell, \quad s(r) = 1/(1 + \mathrm{e}^{-\beta r}). \tag{98}$$

Here and thereafter, for a vector $u = [u^{(1)}, \ldots, u^{(m)}]^{\mathrm{T}}$ and any scalar function $s(r)$, we use the notation

$$s(u) = [s(u^{(1)}), \ldots, s(u^{(m)})]^{\mathrm{T}}. \tag{99}$$

In this case, we get $y_{t,\ell} = y_\ell(x_t)$ and put it into Table 1, which is implemented with Eq. (97) added in. Particularly, Step 5(c) in Eq. (40) takes the following detailed form

$$\text{Step 5(c)}: \qquad \tau^{\text{new}} = \tau^{\text{old}} + 1,$$

$$e_\ell = A_\ell^{\text{old T}} e_{t,\ell} + \sigma_\ell^{\text{old 2}} \phi_y, \qquad e_{t,\ell} = x_t - A_\ell y_t - c_\ell,$$

$$W_\ell^{\text{new}} = W_\ell^{\text{old}} + \gamma_{t,\ell} D_s(\hat{y}_\ell) e_\ell x_t^{\mathrm{T}},$$

$$d_\ell^{\text{new}} = d_\ell^{\text{old}} + \gamma_{t,\ell} D_s(\hat{y}_\ell) e_\ell,$$

$$D_s(\hat{y}_\ell) = \mathrm{diag}[s'(\hat{y}_\ell^{(1)}), \ldots, s'(\hat{y}_\ell^{(m)})], \qquad s'(r) = \frac{\mathrm{d}s(r)}{\mathrm{d}r},$$

$$\hat{y}_\ell = W_\ell x + d_\ell, \tag{100}$$

where ϕ^y is the correcting term that comes from the part of $q(y|\ell)$, given as follows

$$\phi^y = \left[\ln \frac{q_{\ell,1}}{1 - q_{\ell,1}}, \ldots, \ln \frac{q_{\ell,m}}{1 - q_{\ell,m}} \right]^{\mathrm{T}}. \tag{101}$$

To get a further insight, we consider Eq. (96) at the special case that $k = 1$ and $q_j = q_{\ell,j} = 0.5$, $c = c_\ell = 0$, $d = d_\ell = 0$, $h_x = 0$. It follows that empirical harmony learning becomes equivalent to minimize

$$\sigma^2 = \frac{1}{N} \sum_{t=1}^{N} \|x_t - As(Wx_t)\|^2, \tag{102}$$

which is referred as auto-association learning via three layer net and can be trained in the same way of training a three layer net by the Back-propagation technique (Rumelhart, Hinton, & Williams, 1986). Under the constraint that $A = W^{\mathrm{T}}$, it further becomes the *Least Mean Square Error Reconstruction* (LMSER) learning. The LMSER learning was firstly proposed in Xu (1993) with not only both a batch and an adaptive gradient algorithm provided, but also a finding that a sigmoid nonlinearity $s(r)$ leads to an automatic breaking on the symmetry of the components in the subspace. Three years later, the LMSER learning and its adaptive algorithm given in Xu (1993) have been directly adopted to implement ICA with promising results under the name of nonlinear PCA (Karhunen & Joutsensalo, 1994).

Why LMSER performs such an ICA task can also been understood from the perspective of Eqs. (95) and (98). From Eq. (95), each bit of y is expected to take value 1 with probability q_j, independent from other bits. Moreover, it follows from Eq. (98) that each mapping from $s(\hat{y}_\ell)$ to y has no cross-talk among components. Thus, the independence of y among components means the independence of $\hat{y} = Wx$, i.e. it attempts to implement ICA. One key feature of this ICA is that the observation noise is considered via minimizing σ^2, so it provides a solution to the noisy ICA problem.

From the BI-architecture Eqs. (95) and (98) with the learning algorithm in Eq. (100), the original LMSER learning is further extended with the following new results:

(a) *Criterion for hidden unit number and automatic selection*. From Eq. (28) that we get model selection criterion given by the choice (b) of Item 2.2 or of Item 2.3 in Table 4, even simply by the choice (a) of Item 2.1 for a large size of samples.

(b) *ICA that works on both super-Gaussian and sub-Gaussian sources*. Instead of fixing $q_j = q_{\ell,j} = 0.5$ as the case in Eq. (102), making learning with q_j updated via Eq. (97) will let q_j to adapt the distribution of y such that the above discussed ICA works on the observation x that not only contains noise but also is generated from y of either or both of super-Gaussian and sub-Gaussian sources. Thus, it not only acts as an alternative of the LPM-ICA (Xu et al., 1998) with a much simplified computation, but also makes sources selected automatically during learning.

(c) *Local LMSER for structural clustering and competitive ICA*. With $k > 1$, on one hand, we get a local LMSER learning that approximately implements the above Bernoulli–Gaussian mixtures with a structural clustering. On the other hand, the above discussed ICA is made locally on each cluster of data via competition equation (33), i.e. $\ell_t = \arg\min_\ell d(\ell, x_t, y_\ell(x_t))$. In other words, it implements a competitive ICA, with the number k selected either automatically during learning or via the criteria given by the second row of Table 4.

4.3. Independent factorized NonGaussians and local nonGaussian LMSER

Next, we consider that y consists of real independent components as follows:

$$q(y|\ell) = \prod_{j=1}^{m_\ell} q(y^{(j)}|\theta_\ell^{(j)}), \tag{103}$$

where each $q(y^{(j)}|\theta_\ell^{(j)})$ is a nonGaussian scalar density, e.g. it can be modeled either by a finite mixture as in (Xu et al.,

Table 4
Model selection criteria $\mathrm{Min}_{k,m} J(k,m)$ on nonGaussian mixture

	Empirical learning	Data smoothing learning	Normalization learning
General form	$J_0(k,\{m_l\}) = 0.5d\sum_{l=1}^{k}\alpha_l \ln \sigma_l^2 - \sum_{l=1}^{k}\alpha_l(\ln \alpha_l + \vartheta_l)$	$J(k,\{m_l\}) = J_0(k,\{m_l\}) + 0.5dh_x^2\sum_{l=1}^{k}\frac{\alpha_l}{\sigma_l^2}$	$J(k,\{m_l\}) = \ln z_q + J_0(k,\{m_l\}),$ $z_q = \sum_{t=1}^{N}\sum_{l=1}^{k}\alpha_l G(x_t\|A_l y_t + c_l, \sigma_l^2 I) q(y_t\|\theta_l^y)$
Gaussian–Bernoulli mixture	$\vartheta_l = \sum_{j=1}^{m_l} q_{l,j}\ln q_{l,j} + \sum_{j=1}^{m_l}(1-q_{l,j})\ln(1-q_{l,j})$	$\vartheta_l = \sum_{j=1}^{m_l} q_{1,j}\ln q_{l,j} + \sum_{j=1}^{m_l}(1-q_{l,j})\ln(1-q_{l,j})$	$q(y_t\|\theta_l^y) = \prod_{j=1}^{m_l} q_{l,j}^{y_t^{(j)}}(1-q_{l,j})^{y_t^{(j)}}$
Special cases (a) Uniform	$J_0(k,\{m_l\}) = \ln k + \frac{0.5}{k}\sum_{l=1}^{k}(d\ln\sigma_l^2 + 2m_l\ln 2)$	$J(k,\{m_l\}) = J_0(k,\{m_l\}) + \frac{0.5dh_x^2}{k}\sum_{l=1}^{k}1/\sigma_l^2$	$J(k,\{m_l\}) = J_0(k,\{m_l\}) - \ln k + \ln z_q,\ z_q = \sum_{t=1}^{N}\sum_{l=1}^{k}\exp\left(-0.5\frac{\|x_t - A_l y_t - c_l\|^2}{\sigma^2}\right)/2^{m_l}\sigma_l^d$
(b) $k=1$	$J_0(m) = 0.5d\ln\sigma^2 + m\ln 2$	$J(m) = J_0(m) + \frac{0.5dh_x^2}{\sigma^2}$	$J(m) = \ln\left\{\sum_{t=1}^{N}\exp\left(-0.5\frac{\|x_t - Ay_t - c\|^2}{\sigma^2}\right)\right\}$
Mixture of independent factor models	$\vartheta_l = \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{m_l}\ln q(y_t^{(j)}\|\theta_l^{(j)})$	$\vartheta_l = \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{m_l}\ln q(y_t^{(j)}\|\theta_l^{(j)})$	$q(y_t\|\theta_l^y) = \prod_{j=1}^{m_l} q(y_t^{(j)}\|\theta_l^{(j)})$
Special cases (a) Uniform	$J_0(k,\{m_l\}) = \frac{0.5d}{k}\sum_{l=1}^{k}\ln\sigma_l^2 + \ln k - \frac{1}{k}\sum_{l=1}^{k}\vartheta_l$	$J(k,\{m_l\}) = J_0(k,\{m_l\}) + \frac{0.5}{k}\sum_{l=1}^{k}\frac{dh_x^2}{\sigma_l^2}$	$J(k,\{m_l\}) = \ln z_q + \frac{0.5d}{k}\sum_{l=1}^{k}\ln\sigma_l^2 - \frac{1}{k}\sum_{l=1}^{k}\vartheta_l,\ z_q = \sum_{t=1}^{N}\sum_{l=1}^{k}G(x_t\|A_l y_t + c_l, \sigma_l^2 I)\prod_{j=1}^{m_l} q(y_t^{(j)}\|\theta_l^{(j)})$
(b) $k=1$	$J_0(m) = 0.5d\ln\sigma^2 - \frac{1}{N}\sum_{t=1}^{N}\ln q(y_t^{(j)}\|\theta^{(j)})$	$J(m) = J_0(m) + \frac{0.5dh_x^2}{\sigma^2}$	$J(m) = -\frac{1}{N}\sum_{t=1}^{N}\ln q(y_t^{(j)}\|\theta^{(j)}) + \ln\left[\sum_{t=1}^{N}\exp\left(-0.5\frac{\|x_t - Ay_t - c\|^2}{\sigma^2}\right)\prod_{j=1}^{m_l} q(y_t^{(j)}\|\theta^{(j)})\right]$

Note: *J(k)* in a specific block at the *i*th row and *j*th column is referred by Item (i,j).

1998) or via higher order statistics by an expansion

$$q(y^{(j)}|\theta_\ell^{(j)}) = G(y^{(j)}|0,1)[1+\beta_3 h_3(y^{(j)})/6+\beta_4 h_4(y^{(j)})/24], \tag{104}$$

where β_3, β_4 are unknown coefficients, and h_3, h_4 are Hermite polynomials.

With $q(x|y,\ell)$ still given in Eq. (95), we have that $q(x|\ell) = \int G(x|A_\ell y + c_\ell, \sigma_\ell^2 I) q(y|\ell)\mathrm{d}y$ represents a non-Gaussian density via independent factors via a linear model $A_\ell y + c_\ell$. Then, $q(x) = \sum_\ell \alpha_\ell q(x|\ell)$ models all the samples via a mixture of nonGaussian densities with each represented via an independent factor model.

Again, such a mixture can be implemented as a special case of the adaptive learning procedure in Table 1, with $p(y|\ell,x)$ and $p(\ell|x)$ still given by Eq. (33) but with

$$d(\ell,x,y) = 0.5\frac{\|x-A_\ell y - c_\ell\|^2}{\sigma_\ell^2} + 0.5\ln\sigma_\ell^2 - \ln\alpha_\ell - \sum_{j=1}^{m_\ell}\ln q(y^{(j)}|\theta_\ell^{(j)}). \tag{105}$$

In this case, the problem of $y_\ell(x) = \arg\min_y d(\ell,x,y)$ is usually a nonlinear continuous optimization problem. We can solve it by anyone of the classic continuous optimization iterative algorithms. When $q(y^{(j)}|\theta_\ell^{(j)})$ is given by a Gaussian mixture, two types of iterative techniques are given in Table 2 of Xu (2001a) for solving $y_\ell(x)$ by approximately regarding $\nabla_y d(\ell,x,y) = 0$ as a linear equation. Generally, we can denote all these iterative algorithms by the following iterative operator

$$y_\ell^{\text{new}}(x_t) = ITER(y_\ell^{\text{old}}(x_t)). \tag{106}$$

Then, learning is implemented by the adaptive learning procedure in Table 1, with $y_{t,\ell}$ obtained in Step 1 in help of repeating Eq. (106) for a number of iterations, and with Step 4 made in a way similar to that given in Table 3(C) of Xu (2001a).

Again, the computing cost can be reduced in a BI-architecture with $p(y|x,\ell)$ given by a deterministic mapping $y_\ell(x) = s(W_\ell x + d_\ell)$. Similar to Section 4.2, we can update $y_\ell(x) = s(W_\ell x + d_\ell)$ by Eq. (100) but with Eq. (101) replaced by

$$\phi^y = \left[\frac{\partial \ln q(y^{(1)}|\theta_\ell^{(1)})}{\partial y^{(1)}}, \ldots, \frac{\partial \ln q(y^{(m_\ell)}|\theta_\ell^{(m_\ell)})}{\partial y^{(m_\ell)}}\right]^{\mathrm{T}}. \tag{107}$$

Particularly, at the special case that $k=1$ and $c = c_\ell = 0$, $d = d_\ell = 0$, $h_x = 0$, we are lead to an extension of LMSER that not only minimizes the reconstruction error in Eq. (102), but also maximizes the following regularized likelihood

$$\frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{m}\ln q(y_t^{(j)}|\theta^{(j)}) - \ln z_q,$$
$$z_q = \sum_{t=1}^{N} G(x_t|Ay_t,\sigma^2 I)\prod_{j=1}^{m} q(y_t^{(j)}|\theta^{(j)}), \tag{108}$$

for y to fit the product of independent densities in Eq. (103).

On one hand, we can interpret that such an extension implements a nonGaussian independent LMSER via minimizing the reconstruction error in Eq. (102) with a regularization by Eq. (108) that enforces the inner representation y coming from a product of independent nonGaussian densities in Eq. (103). On the other hand, from the reason similar to what discussed in Section 4.2, $\hat{y} = Wx$ implements a ICA task, since the maximization of Eq. (108) enforces y to be independent among its components and $f(\hat{y})$ in the form of Eq. (99) will not introduce any cross-talk among components.

In comparison with that in Section 4.2, this ICA works on observation x that is generated from real independent sources. This point can be further understood at the special case $f(r) = r$. On one hand, the reconstruction error in Eq. (102) is minimized when W spans a m-dimensional principal subspace in the space of x, in a sense that the error in Eq. (102) is the average residual between x and its orthogonal projection on this principal subspace. On the other hand, the maximization of Eq. (108) searches W that make $\hat{y} = Wx$ to become independent. As a result, we are lead to a W that either satisfies or trades off both the purposes.

In the cases of $k > 1$, implemented by Table 1 with Step 4 by Eq. (97) and Step 5 by Eq. (100), a general BI-architecture by Eq. (39) with $y_\ell(x) = f(W_\ell x + d_\ell)$ acts as both a localized nonGaussian LMSER for modeling a mixture of nonGaussian densities and a competitive ICA on x generated from real independent sources.

Again, from Eq. (28) we can make model selection by the criteria given by the last three rows of Table 4.

4.4. Mixture of independent mappings: competitive ICA versus P-ICA

The role of the above discussed competitive ICA can also be understood as mapping samples of x into k clusters with each cluster in a representation of least redundant, i.e. in the form of independent components. Moreover, these clusters are described by a mixture of independent densities $q(y|\ell)$ in Eq. (103).

Similarly, a F-architecture in Section 2.2.3 also performs such a role. Putting $q(y|\ell)$ from Eq. (103) in Eq. (51), we start at a special case $k=1$ and $z_q = 1$. In this case, Eq. (51) is simplified into

$$H(p,q) = \frac{1}{N}\sum_{t=1}^{N}[0.5\ln|W(x_t)W^{\mathrm{T}}(x_t)| + \ln q(y_t)], \tag{109}$$

which is exactly the cost function that leads to the nonlinear LPM-ICA algorithm (or particularly the LPM-ICA algorithm at the special case $f(r) = r$) (Xu, Yang, & Amari, 1996; Xu, 1998b; Xu et al., 1998). Moreover, when W is full rank and the function form of $q(y)$ is pre-specified, it becomes simply $J(W) = (1/N)\sum_{t=1}^{N}[0.5\ln|WW^{\mathrm{T}}| + \ln q(Wx_t)]$, which is the well known cost obtained from several perspectives (Amari, Cichocki, & Yang, 1996; Bell & Sejnowski, 1995; Gaeta & Lacounme, 1990).

Particularly, with $f(r)$ by Eq. (98) and $q(y)$ by Eq. (95), we have

$$W(x) = |WW^{\mathrm{T}}|^{0.5} \prod_{j=1}^{k} q_s(y^{(j)})$$

with $q_s(r) = ds(r)/dr$.

Then, it follows from Eq. (109) that

$$H(p,q) = 0.5 \ln|WW^{\mathrm{T}}| + \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{k}[\ln q_s(y_t^{(j)}) + \bar{y}_t^{(j)} \ln q_j + (1-\bar{y}_t^{(j)}) \ln (1-q_j)],$$

where $y = Wx$ and $\bar{y} = s(y)$. Then, we make max $H(p,q)$ with not only W updated by the following natural gradient ascent:

$$W^{\mathrm{new}} = W^{\mathrm{old}} + \gamma_t (I + \phi^y y^{\mathrm{T}}) W^{\mathrm{old}},$$

$$q_s'(r) = \frac{dq_s(r)}{dr} = \frac{d^2 s(r)}{dr^2},$$

$$\phi^y = \left[q_s'(y^{(1)}) + q_s(y^{(1)}) \ln \frac{q_1^{\mathrm{new}}}{1-q_1^{\mathrm{new}}}, \ldots, q_s'(y^{(k)}) + q_s(y^{(k)}) \ln \frac{q_k^{\mathrm{new}}}{1-q_k^{\mathrm{new}}} \right]$$

but also q_j updated as in Eq. (97) in order to be applicable to the cases that consist of both super-Gaussian and sub-Gaussian sources, since

$$\kappa_j = E(\bar{y}^{(j)} - q_j)^4 - 3[E(\bar{y}^{(j)} - q_j)^2]^2 \geq 0$$

if

$$q_j > \frac{3+\sqrt{3}}{6} \quad \text{or} \quad q_j < \frac{3-\sqrt{3}}{6},$$

and $\kappa_j < 0$

if

$$\frac{3+\sqrt{3}}{6} \geq q_j \geq \frac{3-\sqrt{3}}{6}.$$

In a general case with $k > 1$, as shortly discussed in Xu (2001a), Eq. (51) leads to a competitive ICA algorithm that implements a nonlinear ICA (or simply ICA for $f(r) = r$) at different locations, which map samples separately to different clusters in a mixture of independent densities $q(y|\ell)$. Moreover, $p_t(\ell)$ can also be given by the soft weighting via Eq. (52), instead of giving by a WTA competition equation (51).

All these types of ICA learning can be implemented by simplifying the adaptive learning procedure in Table 1. For implementing Eq. (51), we get

Step 1 : $\hat{y}_\ell = W_\ell x + d_\ell, \quad y_t = f(\hat{y}_{\ell_t}),$

$\tau^{\mathrm{new}} = \tau^{\mathrm{old}} + 1,$

Step 2(a) : Get $p_t(\ell)$ by either Eq .(51) for best harmony or Eq. (52) for least Kullback divergence;

Step 2(b) : $z_q(t) = z_q(t-1) + \sum_{\ell \in L_t} |W_\ell(x_t) W_\ell^{\mathrm{T}}(x_t)|^{0.5} q(y_{t,\ell}|\ell),$

$$\gamma_{t,\ell} = \gamma_0 \left[\frac{p_t(\ell)}{\tau^{\mathrm{new}}} - I(\ell \in L_t) \gamma_{t,\ell}^d \right],$$

$$\gamma_{t,\ell}^d = \frac{|W_\ell(x_t) W_\ell^{\mathrm{T}}(x_t)|^{0.5} q(y_{t,\ell}|\ell)}{z_q(t)},$$

Step 3 : Same as in Table 1,

Step 4 : Same as in Table 1 with a detailed example given in Table 3(C) of Xu (2001a),

Step 5 : $d_{\ell_t}^{\mathrm{new}} = d_{\ell_t}^{\mathrm{old}} + \gamma_{t,\ell} D_s(\hat{y}_\ell) \phi_y,$

$$W_{\ell_t}^{\mathrm{new}} = W_{\ell_t}^{\mathrm{old}} + \gamma_{t,\ell} \begin{cases} [I + D_s(\hat{y}_\ell) \phi^y (W_{\ell_t}^{\mathrm{old}} x_t)^{\mathrm{T}}] W_{\ell_t}^{\mathrm{old}}, & \text{(a)} \\ D_s(\hat{y}_\ell) \phi^y x_t^{\mathrm{T}}, & \text{(b);} \end{cases} \quad (110)$$

where $D_s(\hat{y}_\ell)$ and ϕ^y are same as in Eq. (100). On updating W, the choice (a) comes from implementing Eq. (51), while the choice (b) comes from implementing Eq. (53). The former is an extension of Eq. (74) in Xu (2001a) and the latter provides another type of ICA learning algorithm that actually implementing a nonlinear Hebbian rule in help of a conscience control via $\gamma_{t,\ell}$.

We further compare the above F-architecture based competitive ICA with those competitive ICA implementations by the local LMSER in Sections 4.2 and 4.3. On one hand, one advantage here is saving computing cost on the Ying passage $q(x|y,\ell)$. This can be observed by considering a BI-architecture at the special case $\sigma^2 = 0$ and $q(x|y,\ell) = \delta_x(x - x(y))$. In this case, there is no need to consider the reconstruction error. We only need to consider Eq. (108) where $z_q = \delta_x(0) z_q^y$, $z_q^y = \sum_{t=1}^{N} \prod_{j=1}^{m} q(y_t^{(j)}|\theta^{(j)})$ and $\ln \delta_x(0)$ cancels its counterpart $-\ln \delta_x(0)$ that comes from the reconstruction error. As a result, Eq. (108) becomes equivalent to the harmony learning on the F-architecture given in Eq. (53).

On the other hand, such a connection also reveals one key advantage of the competitive ICA implementations by the local LMSER in Sections 4.2 and 4.3, where a reconstruction is generated from y_t via the Ying passage to best fit the current

observation x_t. For this best reconstruction, the mapping from x_t to y_t must retain the main information about x_t. This is a feature similar to that of PCA. Actually, LMSER by Eq. (102) indeed implements PCA when $s(r) = r$. It is why LMSER is also called nonlinear PCA for a nonlinear $s(r)$. Alternatively, it may be better to be called principal ICA (P-ICA) (Xu, 2001a) since it retains the principal information about x_t, in an analog to PCA. This nature makes it not only possible to consider noise in observation and but also make the concept of model selection meaningful.

Unfortunately, the two advantages disappear in the above F-architecture based competitive ICA. Without the Ying passage, there is no concept on which components of y or what kind of y contains the main information of x since it is only required that $q(y|\ell)$ is independent among components. As a result, not only noise was not taken into consideration, but also we are not able to determine what is an appropriate dimension for y.

However, during learning by Eq. (110), automatic selection on the cluster number k is still workable via its learning rule on α_ℓ. Also, we can simply set $\alpha_\ell = 1/k$ and select k by Eq. (28) with $H(p, q)$ given by Eq. (51).

5. BYY harmony topological self-organizing

5.1. Two computational strategies for topological self-organizing

Up to now, we have discussed BYY harmony learning on several types of finite mixture models $q(x|\ell)q(\ell)$, $x \in X$, $\ell \in L$, with $L = \{1, ..., k\}$ being a nonstructural set. That is, topological relation among individual models has been not considered yet.

We further consider the cases that L has a given regular d-dimensional lattice topology (e.g. 2D or 3D lattice). Since the label ℓ associated with the observed x is invisible, the dependence structures across different objects are not recoverable. Alternatively, we re-establish a dependence structure according to a general belief that objects locating topologically in a small neighborhood N_ℓ should be same or similar to each other. For a knot ℓ on this lattice, its neighborhood N_ℓ consists of 2^d knots that are directly connected to ℓ.

Given a criterion or measure to judge whether two objects are same or similar, a direct placement of all the objects on such a d-dimensional lattice topology is computationally a NP hard problem. Interestingly, a good approximate solution for this problem is provided by biological brain dynamics of self-organization (von der Malsburg, 1973), featured by a Mexican hat type interaction, namely, neurons in near neighborhood excite each other with learning, while neurons far away inhibit each other with de-learning.

Computationally, such a dynamic process can be simplified by certain heuristic strategies. In the following, we consider two typical ones.

(1) *One member wins, a family gains*. That is, as long as one member wins in the WTA competition, all the members of a family gain regardless whether other members are strong or not. This direction is initialized by a simple and clever technique, i.e., the well known Kohonen self-organizing map (Kohonen, 1982, 1995), that is

$$\ell_t = \arg\min_{\ell} \|x_t - c_\ell\|^2,$$

$$c_\ell^{\text{new}} = c_\ell^{\text{old}} + \gamma_0(x_t - c_\ell^{\text{old}}), \qquad \forall \ell \in N_{\ell_t}, \tag{111}$$

where N_{ℓ_t} is a given neighborhood of ℓ_t. In the literatures, a great number of studies have been made on extending Eq. (111). For examples, the Euclidean distance $\|x_t - c_\ell\|^2$ has been extended to various other distances, ranging from subspace distance to symbolic distance (Kohonen, 2001).

Here, we attempt to summarize all the existing studies as special cases of the following general formulation:

$$\begin{aligned} &\text{Step 1}: && \ell_t = \max_{\ell} [q(x|\ell)q(\ell)], \\ &\text{Step 2}: && \text{to increase } q(x|\ell)q(\ell),\ \forall \ell \in N_{\ell_t} \end{aligned} \tag{112}$$

via updating their parameters.

E.g. when $q(x|\ell) = G(x|c_\ell, \sigma^2 I)$ and $q(\ell) = \sum_{j \in L} \frac{1}{k} \delta(\ell - j)$, Eq. (112) returns back to Eq. (111).

(2) *Strongers gain and then teaming together*. That is, a number of strongers are picked as winners who not only gain learning but also are teamed together such that they become the neighbors to each other. More specifically, we have

Step 1 : Let N_w to consist of every $\ell \in L$ that

corresponds to each of the first $2^m + 1$ largest ones

(e.g. $4 + 1$ or $8 + 1$) of $q(x|\ell)q(\ell)$,

Step 2 : Increase $q(x|\ell)$, $q(\ell)$, $\forall \ell \in N_w$

via updating their parameters,

Step 3 : Locate $\forall \ell \in N_w$ according to their values of

q(x|ℓ)q(ℓ) in such a way that the largest locates at ℓ_t,

and every other $\ell \in N_w$ that was not in the

neighborhood of ℓ_t is exchanged with a neighbor of ℓ_t

that does not belong to N_w. (113)

The second strategy is new in the literature. It can speed up self-organization, especially in the early stage of learning. Also, we can combine the first and the

second strategies by using the second in the early stage and subsequently switching to the first.

5.2. BYY harmony topological self-organizing

Following a deriving line similar to that used in developing temporal BYY harmony learning in Xu (2000), we can obtain a more theoretically justified approach for implementing BYY harmony topological self-organizing, in help of considering $\mathbf{x} = \{x, \ell\}$, $\ell \in L$ on a lattice L and introduce a Markovian property that x located at ℓ_t relates only to those knots in a small neighborhood of ℓ_t.

However, for simplicity and easy implementation, we would rather adopt the above two strategies to make BYY harmony topological self-organization. Particularly, the first strategy is easy to be added to those learning algorithms introduced in the previous sections.

The key point is to make the following modification, i.e.

$$p_t(\ell) = \delta_{\ell,\ell_t} = \begin{cases} 1, & \ell = \ell_t, \\ 0, & \text{otherwise}; \end{cases}$$

$$\text{is replaced with } p_t(\ell) = \begin{cases} 1, & \forall \ell \in N_{\ell_t}, \\ 0, & \text{otherwise}; \end{cases} \tag{114}$$

where N_{ℓ_t} is a given small neighbor of ℓ_t. For a lattice L, N_{ℓ_t} usually consists of 2^m neighbors (e.g. four or eight neighbors) of ℓ_t. All the other parts in the previous learning algorithms remain unchanged, with topological self-organizing implemented during learning.

Several typical topics are listed as follows:

- *Elliptic RPCL map*. With Eq. (114) used in Section 3.2, the elliptic RPCL learning will also organize those elliptic clusters topologically in a map;
- *Structural RPCL based subspace map*. With Eq. (114) used in Section 3.3, by using Eq. (84) we can get learned subspaces to be organized topologically in a map;
- *Gaussian ME map and RBF net map*. With Eq. (114) used in Section 3.4 and especially in Eq. (87);
- *LMSER map*. With Eq. (114) used in Section 4.2;
- *NonGaussian LMSER map*. With Eq. (114) used in Section 4.3;
- *ICA map*. With Eq. (114) used in Section 4.4.

The features of these topological self-organizing maps can be understood from two perspectives. On one hand, inheriting the advantages of the classical Kohonen map, the establishment of topological structure not only further regularizes learning via interaction among neighbors but also makes a fast retrieval of the desired object feasible since the similarity among neighbors acts as an associative key for searching, which are much preferred when applying these learnings for data-mining. Moreover, the performance is more robust in a sense that some damage part are recoverable from its neighbors. On the other hand, being different from the Kohonen map, not only it has been extended to using various types of models as a unit on the map, but also it combines the advantages of BYY harmony learning, particularly the mechanism of conscience learning and automatic model selection, as described in the previous sections, into topological self-organization.

6. Conclusions

Fundamentals of BYY learning have been systemically elaborated, and new advances are obtained on BYY systems with modular inner representations. New results are obtained on not only Gaussian mixture based MSE clustering, elliptic clustering, subspace clustering but also nonGaussian mixture based clustering with each cluster represented via either Bernoulli–Gaussian mixtures or independent real factor models. Moreover, typical algorithms for ICA and competitive ICA are summarized from the perspective of BYY harmony learning, with a comparison made on the P-ICA and competitive P-ICA. Furthermore, new results are obtained on Gaussian ME, RBF net, and Kernel regression. Even further, two strategies are presented for extending all these mixture models into self-organized topological maps. All these results include not only adaptive learning algorithms with model selection automatically made during learning but also model selection criteria for being used in a two phase style learning.

Acknowledgments

The author would like to express thanks to the reviewers for their comments that improve the original manuscript.

References

Ahalt, S. C., Krishnamurty, A. K., Chen, P., & Melton, D. E. (1990). Competitive learning algorithms for vector quantization. *Neural Networks*, *3*, 277–291.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *19*, 714–723.

Amari, S.-I., Cichocki, A., & Yang, H. H. (1996). A new learning algorithm for blind separation of sources. In D. S. Touretzky, et al. (Eds.), *Advances in neural information processing*, (*Vol. 8*) (pp. 757–763). Cambridge: MIT Press.

Bell, A. J., & Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind de-convolution. *Neural Computation*, *7*, 1129–1159.

Bishop, C. M. (1995). Training with noise is equivalent to Tikhonov regularization. *Neural Computation*, *7*, 108–116.

Bozdogan, H. (1987). Model selection and Akaike's information criterion: The general theory and its analytical extension. *Psychometrika*, *52*, 345–370.

Bozdogan, H., & Ramirez, D. E. (1988). FACAIC: Model selection algorithm for the orthogonal factor model using AIC and FACAIC. *Psychometrika*, *53*(3), 407–415.

Cavanaugh, J. E. (1997). Unifying the derivations for the Akaike and corrected Akaike information criteria. *Statistics and Probability Letters*, *33*, 201–208.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum-likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society B*, *39*, 1–38.

Devroye, L., Györfi, L., & Lugosi, G. (1996). *A probability theory of pattern recognition*. Berlin: Springer.

Gaeta, M., & Lacounme, J.-L. (1990). Source separation without a priori knowledge: The maximum likelihood solution. *Proceedings of EUSIPCO90*, 621–624.

Girosi, F., Jones, M., & Poggio, T. (1995). Regularization theory and neural architectures. *Neural Computation*, *7*, 219–269.

Hinton, G. E., & Zemel, R. S. (1994). Autoencoders, minimum description length and Helmholtz free energy. *Advances in NIPS*, *6*, 3–10.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, *3*, 79–87.

Jordan, M. I., & Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, *6*, 181–214.

Jordan, M. I., & Xu, L. (1995). Convergence results for the EM approach to mixtures of experts. *Neural Networks*, *8*, 1409–1431.

Karhunen, J., & Joutsensalo, J. (1994). Representation and separation of signals using nonlinear PCA type Learning. *Neural Networks*, *7*, 113–127.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*, 671–680.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, *43*, 59–69.

Kohonen, T. (1995). *Self-organizing maps*. Berlin: Springer.

Kohonen, T. (2001). Mining and mapping biochemical data from the internet. *Plenary talk, IJCNN01, Washington DC, 1–19 July*, 2001.

Mackey, D. J. C. (1992). A practical Bayesian framework for back-propagation. *Neural Computation*, *4*, 448–472.

Makhoul, J., Rpucos, S., & Gish, H. (1985). Vector quantization in speech coding. *Proceedings of IEEE*, *73*, 1551–1558.

von der Malsburg, Ch. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, *14*, 85–100.

McLachlan, G. J., & Basford, K. E. (1988). *Mixture models: Inference and application to clustering*. New York: Dekker.

Moody, J., & Darken, J. (1989). Fast learning in networks of locally-tuned processing units. *Neural Computation*, *1*, 281–294.

Neath, A. A., & Cavanaugh, J. E. (1997). Regression and time series model selection using variants of the Schwarz information criterion. *Communications in Statistics A*, *26*, 559–580.

Nowlan, S. J. (1990). *Max likelihood competition in RBF networks.* Technical Report CRG-Tr-90-2, Department of Computer Science, University of Toronto.

Redner, R. A., & Walker, H. F. (1984). Mixture densities, maximum likelihood, and the EM algorithm. *SIAM Review*, *26*, 195–239.

Rivals, I., & Personnaz, L. (1999). On cross validation for model selection. *Neural Computation*, *11*, 863–870.

Rissanen, J. (1986). Stochastic complexity and modeling. *Annals of Statistics*, *14*(3), 1080–1100.

Rissanen, J. (1999). Hypothesis selection and testing by the MDL principle. *Computer Journal*, *42*(4), 260–269.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). *Learning internal representations by error propagation* (*Vol. 1*). *Parallel distributed processing*, Cambridge, MA: MIT Press.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, *6*, 461–464.

Stone, M. (1974). Cross-validatory choice and assessment of statistical prediction. *Journal of the Royal Statistical Society B*, *36*, 111–147.

Stone, M. (1978). Cross-validation: A review. *Mathematics, Operations and Statistics*, *9*, 127–140.

Sugiura, N. (1978). Further analysis of data by Akaike's information criterion and the finite corrections. *Communications in Statistics A*, *7*, 12–26.

Tikhonov, A. N., & Arsenin, V. Y. (1977). *Solutions of ill-posed problems*. Baltimore, MD: Winston & Sons.

Tipping, M. E., & Bishop, C. M. (1999). Mixtures of probabilistic principal component analysis. *Neural Computation*, *11*, 443–482.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Berlin: Springer.

Wallace, C. S., & Boulton, D. M. (1968). An information measure for classification. *Computer Journal*, *11*, 185–194.

Wallace, C. S., & Dowe, D. R. (1999). Minimum message length and Kolmogorov complexity. *Computer Journal*, *42*(4), 270–280.

Xu, L. (1993). Least mean square error reconstruction for self-organizing neural-nets. *Neural Networks*, *6*, 627–648. Its early version on Proceedings of IJCNN91'Singapore, 1991, pp. 2363–2373.

Xu, L. (1995). A unified learning framework: multisets modeling learning. *Proceedings of 1995 World Congress on Neural Networks*, *1*, 35–42.

Xu, L. (1996). A unified learning scheme: Bayesian–Kullback YING-YANG machine. *Advances in Neural Information Processing Systems*, *8*, 444–450. A part of its preliminary version on Proceedings of ICONIP95, 1995, pp. 977–988.

Xu, L. (1997). Bayesian Ying-Yang machine, clustering and number of clusters. *Pattern Recognition Letters*, *18*(11–13), 1167–1178.

Xu, L. (1998a). RBF nets, mixture experts, and Bayesian Ying-Yang learning. *Neurocomputing*, *19*(1–3), 223–257.

Xu, L. (1998b). Bayesian Kullback Ying-Yang dependence reduction theory. *Neurocomputing*, *22*(1–3), 81–112.

Xu, L. (2000). Temporal BYY learning for state space approach, hidden Markov model and blind source separation. *IEEE Transactions on Signal Processing*, *48*, 2132–2144.

Xu, L. (2001a). BYY harmony learning, independent state space and generalized APT financial analyses. *IEEE Transactions on Neural Networks*, *12*(4), 822–849.

Xu, L. (2001b). Best harmony, unified RPCL and automated model selection for unsupervised and supervised learning on Gaussian mixtures, three-layer nets and ME-RBF-SVM models. *International Journal of Neural Systems*, *11*(1), 43–69.

Xu, L. (2002a). Bayesian Ying Yang harmony learning. In M. A. Arbib (Ed.), *The handbook of brain theory and neural networks* (second edition). Cambridge, MA: The MIT Press, in press.

Xu, L. (2002b). *Mining dependence structures from statistical learning perspective*. *Lecture Notes in Computer Science: Proceedings of the Third International Conference on IDEAL*, Berlin: Springer.

Xu, L., Cheung, C. C., & Amari, S.-I. (1998). Learned parametric mixture based ICA algorithm. *Neurocomputing*, *22*(1–3), 69–80. A part of its preliminary version on Proceedings on ESANN97, pp. 291–296.

Xu, L., Jordan, M. I., & Hinton, G. E. (1995). An alternative model for mixtures of experts. In J. D. Cowan, et al. (Eds.), *Advances in neural information processing systems*, (*Vol. 7*) (pp. 633–640). Cambridge, MA: MIT Press, Its preliminary version on Proceedings of WCNN'94, San Diego, vol. 2, 1994, pp. 405–410.

Xu, L., Krzyzak, A., & Oja, E. (1993). Rival penalized competitive learning for clustering analysis, RBF net and curve detection. *IEEE Transactions on Neural Networks*, *4*, 636–649.

Xu, L., Krzyzak, A., & Yuille, A. L. (1994). On radial basis function nets and kernel regression: Statistical consistency, convergence rates and receptive field size. *Neural Networks*, *7*, 609–628.

Xu, L., Yang, H. H., & Amari, S.-I. (1996). *Signal source separation by mixtures accumulative distribution functions or mixture of bell-shape density distribution functions.* Research proposal, presented at FRONTIER FORUM (speakers: D. Sherrington, S. Tanaka, L. Xu & J. F. Cardoso), organised by S. Amari, S. Tanaka, & A. Cichocki, RIKEN, Japan.