Independent Component Analysis and Extensions with Noise and Time: A Bayesian Ying-Yang Learning Perspective

Lei Xu
Department of Computer Science and Engineering
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, P.R. China
Email: lxu@cse.cuhk.edu.hk

Submitted on July 28, 2003; Accepted on October 24, 2003

Abstract— After summarizing typical approaches for solving independent component analysis (ICA) problems, advances on the ICA studies that consider hybrid sources of both subGaussians and super-Gaussians and the ICA extensions that consider noise and temporal dependence among observations have been overviewed from the perspective of Bayesian Ying-Yang independence learning. Not only new insights are provided on existing results in literature, but also a number of further results are presented.

Keywords— BYY harmony learning, ICA, PCA, MCA, nonlinear Hebbian, regularization, model selection, factor analysis, non-Gaussian factor analysis (NFA), binary factor analysis (BFA), LMSER learning, three layer net, temporal extensions, Kalman filter, independent state space

1. Introduction

Independent component analysis (ICA) becomes popular in the recent literature of neural networks and signal processing, which aims at blindly solving a linear system from its observation. Such a linear system can be either simply an instantaneous system

$$x = Ay, x = [x^{(1)}, \cdots, x^{(d)}]^T, y = [y^{(1)}, \cdots, y^{(m)}]^T, \tag{1}$$

or a convolution system

$$x_t = \sum_{\tau=-\infty}^{+\infty} A_\tau y_{\tau-t}, \tag{2}$$

where $\{x_t\}, \{y_t\}$ are both time series. However, both the system A and source y are unknown, and thus solution on both types of the linear systems can not be determined.

One early effort on the problem eq.(2) can be backtracked, under the name of blind deconvolution, to the period from 60's to the 80's in the literature of geophysical exploration on oil and gas, where y_t is called seismic wavelet, A_t describes the reflection nature of underground layers, and x_t represents seismic waves received by a sensor array laid on the ground [69, 42, 49, 86]. To tackle the indeterminacy problem, efforts have been made by regarding eq.(2) as a system that is either deterministic or stochastic. Regarding it as a deterministic system, a typical strategy is to assume that y_t is of the minimum phase or that the phase spectrum of y_t is obtainable from a priori knowledge. Alternatively, when x_t, A_t are assumed to be finite length, the problem can be solved in certain cases by only knowing the endpoint or partly samples of y_t [125, 126, 127, 128]. While regarding eq.(2) as a stochastic system, the problem was found to be solvable when y_t is assumed to be a Bernoulli white noise series [49].

Backtracked to the 70's [75], another typical early effort on the problem eq.(2) has been made widely in the field of communication under the name of blind channel equalization/ identification/estimation. Readers are referred to the survey paper [84].

One typical early effort on the problem eq.(1) is made in 1986 by [29] for solving the blind source separation by assuming that the components of y are mutually independent, i.e., the density of of y takes the following form:

$$q(y) = \prod_{j=1}^m q(y^{(j)}),\ y = [y^{(1)}, \cdots, y^{(m)}]^T. \tag{3}$$

This effort is supported by the result of (Tong, Inouye, & Liu, 1993). As long as W makes $\hat{y}_t = Wx_t$ become component-wise independent, they showed that $\hat{y}_t = Wx_t$ recovers y_t up to constant scales and a permutation of components when the components of y_t are mutually independent and at most one of them is Gaussian. The problem is further formularized by Comon [22] under the name independent component analysis (ICA). Moreover, the mapping $\hat{y}_t = Wx_t$ can recover from $x_t = Ay_t$ the waveform of each $\{y_t^{(j)}\}$ series subject to unknown scales, which is also said to blindly separate sources [83].

Studies on the ICA problem have become quite popularized in the literature of neural networks, with many extensions. One direction is extending the ICA problem to noisy environments. One typical example is adding noise to the observation:

$$x = Ay + e, \tag{4}$$

subject to satisfying eq.(3), where e usually consists of i.i.d. samples that is independent of y. Particularly, when

$$e \sim G(e|0, \Sigma),\ y \sim G(y|0, I), \tag{5}$$

with $x \sim p(x)$ denoting that x comes from $p(x)$, eq.(4) becomes to the classic factor analysis (FA) model [7] that has been widely applied to various data analyses.

The other direction is to further extend the ICA problem as well as its noisy extensions by considering temporal relation among data. Another direction is going beyond a linear system by considering certain nonlinear systems under the name nonlinear ICA.

There have been several survey papers on the ICA studies [4, 15, 34]. Instead of including all the studies, the present paper will focus on the first two directions, with a systematic sketch from a unified perspective called Bayesian Ying-Yang (BYY) harmony learning. The BYY harmony learning was firstly proposed in [119] and then developed in past several years as a 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 after parameter learning [87, 89, 90, 92, 93, 94]. Main focus of this paper will be put on the problem of eq.(1) and its extensions that consider noise observation and temporal dependence. The main contents of this paper are listed as follows:

- Typical Solving Approaches
  - Solving equations: Higher-order-statistics vs Hebbian learning
  - Cost function approaches vs equation based approaches
  - Cost functions for PCA and nonlinear extensions
  - From PCA vs MCA to P-ICA vs M-ICA
- BYY Independence Learning
  - Bayesian Ying-Yang system
  - BYY harmony learning
  - Regularization vs model selection: (I) ML, HL, and z-regularization
  - Regularization vs. model selection: (II) KL-λ- HL Spectrum
  - Ying-Yang alternative procedure for parameter learning
  - Regularization vs Model Selection: (III) From $\ln(r)$ to Convex Function
- $L(W)$ based ICA studies
  - Marginal densities: prefixed vs learned
  - Loosely matching, 1-bit-conjecture, and open issues
  - Natural gradient, non-invertible matrix, and other extensions
- Extensions of ICA to Noisy Environment
  - Non-Gaussian factor analysis, algebraic equation, and ML learning
  - BYY harmony learning based nonGaussian FA

  - Model selection vs automatic model selection
  - Special cases: LMSER and auto-association
  - Approximate ML, exact ML, and EM algorithm
  - Non-negative data analyses and supervised learning on three layer net
  - Minimizing fitting error vs enforcing factor independence

- Extensions of ICA with temporal dependence
  - Typical approaches for ICA with temporal dependence
  - Two types of temporal BYY systems
  - TFA, temporal NFA, and space dimension selection
  - Temporal ICA, Kalman filter, and identifiable state spaces
  - Temporal BFA, temporal LMSER, and supervised recurrent net

Not only new insights are provided on existing results in a systematic manner, but also a number of new results are presented, which are introduced in appropriate places of the above context, and them summarized in the concluding remark section. Moreover, due to a close relation between the two linear systems eq.(1) and eq.(2), further discussions on the problem of eq.(2) may also be made similarly but are omitted in this paper.

2. Typical Solving Approaches

Different approaches have been attempted on solving the ICA problems. These approaches can be classified into two families, namely solving equations and optimizing costs.

2.1. Solving Equations: Higher-Order-Statistics vs Hebbian Learning

Higher-order-statistics based algebraic equations Taking the higher order statistics (e.g., moments) of x, y subject to the constraints by eq.(1) or equivalently $y = Wx$ as well as eq.(3), algebraic equations can be obtained with the matrix W and these higher order statistics in consideration [18, 16]. Though being nonlinear, these equations are algebraic equations and the problems become solving the roots of these equation. Generally speaking, those algorithms developed in the literature of numerical analysis for solving the roots of algebraic equations can be adopted for this purpose.

The approaches of this type have the following features:

(1) Equations must be obtained from a batch of samples because statistics need to be computed collectively. That is, this type is difficult to work adaptively per each sample comes.

(2) Intuitively, we can obtain an enough number of equations by increasing the order of statistics, such that W can be finally solved with eq.(3) satisfied. Usually the statistics up to 4th order are considered [53]. However, there lack theoretical guides on up to which order to be considered, since statistics of different orders usually have certain dependence and thus the obtained joint equations may involve a complicated dependent relations that are difficult to be described and understood explicitly.

(3) Nonlinear joint equations usually have many roots. Of course, we desire that any one of the roots corresponds a W with eq.(3) satisfied. However, we are not clearly under what conditions such a situation occurs. If we encounter a case that some roots corresponds a W with eq.(3) satisfied, while other roots are not. How can we find an algorithm to tackle this problem ?

(4) The number of statistics to be considered increases exponentially as the order increases. As a result, we have to consider a large set of joint nonlinear equations and thus suffer a high expense on computation, even when the above problems solved.

Hebbian learning based stochastic equations Inspired by the fact that implementing an anti-Hebbian learning on the recurrent weights will remove second order correlations among the outputs of a linear recurrent neural network, higher order dependences among the outputs are expected to be removed by two modifications. One is implementing an anti-Hebbian learning on the recurrent weights with each linear neuron is replaced by a sigmoid neuron (i.e., a linear neuron followed by a sigmoid scalar function). The other is implementing a nonlinear anti-Hebbian learning on the recurrent weights of a linear recurrent neural network. The two modifications are equivalent, which have been shown, both via many experiments and also partially via theoretical analyses that

higher order dependences between the outputs can indeed be removed. It is this type of results that originally motivated the early studies on ICA [29, 35].

Similarly, inspired by the fact that implementing a Hebbian learning on the forward weights will make a linear forward net perform PCA with second order correlations among the outputs removed, studies have been also made on removing higher order dependences among the outputs of a linear forward net via either implementing a Hebbian learning with each linear neuron replaced by a sigmoid neuron or implementing a nonlinear Hebbian learning on the weights of a linear forward neural network [57, 59].

Both the two types of Hebbian learning extensions are closely related. E.g., the former type on a linear recurrent net can be equivalently turned into the latter type on a linear forward net, when the dimension of the inputs and outputs are the same. Let W to denote the matrix of either recurrent weights or forward weights to be learned, the processes of both types of Hebbian learning extensions can be summarized into the following stochastic equation:

$$\frac{dW_t}{dt} = f(W_t, x_t, y_t, H_t), H_t = \phi(y_t)x_t^T, \tag{6}$$

where $x = [x^{(1)}, \cdots, x^{(d)}]^T$ is an input random vector and $y = [y^{(1)}, \cdots, y^{(m)}]^T$ is an output vector. Moreover, $\phi(y)$ denotes $\phi(y) = [\phi(y^{(1)}), \cdots, \phi(y^{(m)})]^T$ and $\phi(r)$ is a nonlinear scalar function that closely relates to the sigmoid non-linearity after a linear neuron.

In eq.(6), the key term is $H_t = \phi(y_t)x_t^T$ that implements a nonlinear Hebbian learning, which is expected to make W_t tend to let eq.(3) satisfied. However, a dynamic stochastic equations contains only H_t may not work stably. To avoid this problem, an appropriate function $f(W_t, x_t, y_t, H_t)$ must be designed to make the dynamic system stabilized.

This type of approaches is usually referred as being heuristic, with the following features:

(1) Though the motivation of using nonlinear Hebbian learning to make W satisfy eq.(3) is intuitively sound, whether the learning by eq.(6) works is not guaranteed for a given function $f(W_t, x_t, y_t, H_t)$, it should be tested either experimentally or via a convergence analysis, which is usually not a quite easy task [79]. Moreover, even when it is conducted, a convergence analysis is usually able to be made in the simple mean convergence sense:

$$E[f(W_t, x_t, y_t, H_t)] = 0, \ as \ t \to \infty. \tag{7}$$

(2) Except some hints, there lacks a systematic way to guide selecting an appropriate $f(W_t, x_t, y_t, H_t)$. Also, there lacks a general guideline on how to control the convergence performance.

(3) Learning equation eq.(6) is usually concise and regular. Thus, it is easy to be implemented. Especially, the learning may be made in parallel at the level of each weight coefficient, which is favorable for hardware implementation.

2.2. Cost Function Approaches vs Equation Based Approaches

The cost function approaches are featured by optimizing a cost (or called contrast) function. That is, to maximize or minimize a cost function with respect to W such that an optimal point is searched at the point that eq.(3) is satisfied. A cost function can be obtained from four typical aspects, discussed as follows.

(1) Solving a joint equations $g(W) = 0$ can be directly turned into a cost $\|g(W)\|^2$ or $C(g(W)) \geq 0$ that is minimized when $g(W) = 0$. One advantage of considering the problem in such a way is providing a control on errors of a solution that can not ensure $g(W) = 0$ exactly.

(2) A dynamic stochastic equations by eq.(6) that converges to a stable solution actually corresponds to a descent process of a stochastic cost function. Getting such a cost function will facilitate convergence analysis on eq.(6) in help of the optimization theory. Though a task of finding such a cost is usually not straight forward, we may still have some hints, as to be discussed in the next subsection. In the literature of PCA/MCA learning, we have already known a number of cost functions by which their optimizations lead to performing PCA/MCA [120, 121]. We may extend these costs to implement ICA simply by replacing a linear neuron with a sigmoid neuron.

(3) The independence eq.(3) implies that no cross dependence exists among statistics of the components of y up to any order. This motivates to build a cost that bases on higher order statistics of $y = Wx$ (e.g., the higher order cumulants) [77, 32, 17, 39, 78].

(4) A cost function also comes from typical statistical learning principles that can be used to estimate the product of independent densities by eq.(3). Typical examples include maximizing likelihood (ML) [26], minimizing mutual information (MMI) [6, 22], maximum information transfer (INFOMAX) [10], max-negentropy [34], etc.

Though formularized from different statistical learning principles, the resulted cost functions may relate closely and even are equivalent. E.g., in the case that W is invertible, MMI and INFOMAX are both equivalent to ML that maximizes the following cost

$$L = \ln|W| + \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{m}\ln q(y_t^{(j)})|_{y_t=Wx_t}, \tag{8}$$

with $q(y^{(j)})$ being an estimate on the density of $y^{(j)}$.

Any positive cost function $\sum_t C(y_t)$ with $C(y_t) \geq 0$ can be equivalently turned into a likelihood $\sum_t \ln p(y_t)$ via the so called Gibbs distribution $p(y) = Z^{-1}exp(-C(y))$ as long as $Z = \int_{-\infty}^{\infty} exp(-C(y))dy < \infty$.

The cost L can be maximized via a gradient ascent related algorithm either on a batch of samples or adaptively as each sample comes. Actually, these algorithms relate closely to these nonlinear Hebbian learning based approaches. This point can be clearly observed from the fact that

$$g_W(t) = \nabla_W \ln q(y_t) = \phi(y_t)x_t^T = H_t, \quad \phi(y) = \frac{d\ln p(y)}{dy}, \tag{9}$$

i.e., H_t is exactly the nonlinear Hebbian term in eq.(6).

A more clear link can be observed in the case that a pre-whitening processing is made such $Ex = 0, Exx^T = I$. In this case, the ICA mapping $y = Wx$ is subject to the orthogonal constraint $WW^T = I$. Thus, the stochastic gradient ascent algorithm for $\max L(W)$ is given as follows

$$\frac{dW_t}{dt} = g_W(t) = \phi(y_t)x_t^T,\ y_t = W_t x_t,\ s.t.\ W_t W_t^T = I. \tag{10}$$

That is, it is exactly a nonlinear Hebbian learning on the Stiefel manifold $W_t W_t^T = I$. Specifically, it can be implemented by either of two approaches in Tab.1. E.g., It follows from the choice (a) of the approach I that we can get one Stiefel manifold based Hebbian learning rule as follows:

$$\frac{dW_t}{dt} = g_W(t) - W_t g_W^T(t) W_t, \tag{11}$$

which reduces to the known formula in [15] when we neglect B (i.e., simply setting $B = 0$) and to Eq.(4.13) in [3] when we neglect A (i.e., simply setting $A = 0$), respectively.

Furthermore, a pre-whitening process can also be implemented adaptively on a general input ξ_t, e.g., by

$$\frac{dU_t}{dt} = (I - x_t x_t^T)U_t,\ x_t = U\xi_t, \tag{12}$$

which can be implemented jointly with eq.(11).

2.3. Cost Functions for PCA and Nonlinear Extensions

In [120, 121], typical cost functions for PCA and nonlinear extensions are systematically discussed. Here we make a brief summary from an ICA perspective. Without losing generality, we focus on the following three families of cost functions:

Maximum Output's Variation of Bounded Linear System This family is featured by a principle that consists of two key points. One is choosing W that maximizes the variation of the output $y = Wx$ for a maximum information transfer. Without any constraint, this maximization however will tend to infinite and thus become undefined. Thus, the other key point is to avoid this by imposing that this system is bounded, i.e., $\|W\| \leq c$, where $\|A\|$ denotes the matrix norm of A, and $c > 0$ is a given constant. Mathematically, the family of cost functions share the following general form:

$$\max_W V_y(W),\ subject\ to\ \|W\| \leq c. \tag{13}$$

This $V_y(W)$ denotes a measure that describes the variation of y. Since the 2nd order statistics usually describes a major part of variation, several measures in the literature are based on the 2nd order statistics. They are summarized into the following general form

$$V_y(W) = f(\sum_{j=1}^{m} g_j(\lambda_j^y)), \tag{14}$$

Tab.1 Two Approaches for $\max_W L(W)$ within Stiefelf Manifold $WW^T = I$

Approach I
As shown in (Xu, 2002a), we differentiate $W_t W_t^T = I$ and get $\delta W_t W_t^T + W_t \delta W_t^T = 0$.
The solution δW_t has a form $\delta W_t = 0.5(A - A^T)W_t + B(I - W_t^T W_t)$.
$(A - A^T)W_t$ is orthogonal to $B(I - W_t^T W_t)$ since $Tr[((A - A^T)W_t)^T B(I - W_t^T W_t)] = 0$.
Further denoting the gradient of $L(W)$ with respect to W by $g_W(t) = \nabla_W L(W)$,
we consider its maximum projection on $W_t W_t^T = I$ via
$Tr[g_W(t)(0.5(A - A^T)W_t)^T]$, $Tr[g_W(t)(B(I - W_t^T W_t))^T]$, which are maximized by
$A = g_W(t)W_t^T - W_t g_W^T(t)$, $B = g_W(t)(I - W_t^T W_t)$, and then we update
$W_{t+1} = W_t + \eta \delta W_t$ with $\delta W_t = \begin{cases} g_W(t) - W_t g_W^T(t) W_t, & \text{choice (a)}, \\ g_W(t)(I - W_t^T W_t), & \text{choice (b)}, \\ g_W(t) - W_t g_W^T(t) W_t + g_W(t)(I - W_t^T W_t), & \text{choice (c)}. \end{cases}$
Approach II
Learning is made within Stiefelf Manifold $WW^T = I$ along its geodesic
$W_{t+1} = M_t^T W_t + N_t^T R$
with Q, M_t, N_t obtained in help of the QR decomposition of
$(g_W(t) - W_t g_W^T(t) W_t)(I - W_t^T W_t)$. The details are referred to (Edelman, 1998).

where $\lambda_j^y, j = 1, \cdots, m$ are the eigenvalues of the covariance matrix $\Sigma_y = E(yy^T)$, and $f(r), g_j(r), j = 1, \cdots, m$ are monotonically increasing as a scalar r increases. E.g., two special examples are

$$\begin{aligned} &V_y(W) = Tr[\Sigma_y] = \textstyle\sum_{j=1}^m \lambda_j^y,\ f(r) = r,\ g_j(r) = r, \\ &V_y(W) = |\Sigma_y| = \textstyle\prod_{j=1}^m \lambda_j^y,\ f(r) = e^r,\ g_j(r) = \ln r. \end{aligned} \tag{15}$$

Beyond the 2nd order statistics, a more accurate way to describe the variation of y should also include higher order statistics. A general measure is the entropy [10]

$$V_y(W) = -\int p(y) \ln p(y) dy,\ subject\ to\ \|W\| \leq c. \tag{16}$$

that bases on the entire density $p(y)$ and thus covers statistics of all the orders.

It can be further observed, e.g., from eq.(14), that the maximization of $V_y(W)$ subject to $\|W\| \leq c$ is reached at the boundary $\|W\| = c$. One widely considered case is $W^T W = I$, i.e., an orthogonal matrix W. Equivalently we can also get its unconstrained version $W = (UU^T)^{-0.5}U$ with a nonsingular U. Subject to such systems, $\max_W V_y(W)$, either by eq.(14) or by eq.(16) with y being Gaussian, results in a W that spans a principal subspace that is spanned by the m eigenvectors of $\Sigma_x = E[(x - Ex)(x - Ex)^T]$ and these eigenvectors correspond to the m largest eigen-values of Σ_x. The maximization can be implemented adaptively per sample, either by updating W within $W^T W = I$ (e.g., by Oja subspace rule [56]) or by updating U along the gradient direction $\nabla_U V_y(W)|_{W=(UU^T)^{-0.5}U}$.

More generally, we can consider

$$W^T W = D^2\ or\ W = D(UU^T)^{-0.5}U, \tag{17}$$

where $D = diag[d_1, \cdots, d_m]$ is a diagonal matrix. When $d_1 \neq d_2 \neq \cdots \neq d_m$, it can be shown, similar to the analysis in [120] and [121], that the maximum of $V_y(W)$ is reached only when the m row vectors of W become the m eigenvectors of Σ_x that correspond to the m largest eigen-values, respectively. That is, $y = Wx$ performs exactly the m-PCA. When $D = cI$, it degenerates to being equivalent to the case of $W^T W = I$ that performs a Principal Subspace Analysis (PSA) made by the Oja subspace rule [56].

Best Reconstruction of x by $\hat{x} = W^T W x$ This family is featured by a principle that a reconstruction $\hat{x} = W^T y$ from $y = Wx$ should be in a best fit to x. Again, the fitting is usually measured in the second order. That is, we have

$$\min_W E_2(W),\ E_2(W) = E\|x - \hat{x}\|^2 = E\|x - W^T y\|^2 = E\|x - W^T W x\|^2,\ y = Wx, \tag{18}$$

which has been widely studied in the field of signal and image processing. Here, an explicit constraint of $WW^T = I$ is not needed. As shown in [123], $\min_W E_2(W)$ results in a W that performs PSA. In help of $\min_W E_2(W)$, the global convergence of the Oja subspace rule [56] was firstly proved mathematically in [123]. With W given via U in eq.(17), we can further know that the minimum of $E_2(W)$ is reached at a W that $y = Wx$ performs exactly the m-PCA.

Similarly, we can go beyond the 2nd order by considering a maximum likelihood fitting as follows:

$$L(W) = -\frac{1}{N}\sum_{t=1}^{N} \ln p(x_t|W^T y, \theta),\ y = Wx. \tag{19}$$

Maximum Relative Variation This family can be regarded as an extension of the first family. Still, the variation of the output $y = Wx$ is maximized for a maximum information transfer. However, to avoid W becoming unbounded, a reference output of the same system $\zeta = W\xi$ is considered, where ξ denotes a given input with zero mean and uncorrelated elements, i.e., $E\xi = 0$ and $E(\xi\xi^T) = \Lambda_\xi$ being a known diagonal matrix. Instead of imposing constraint on W directly, we minimize the variation of the reference $\zeta = W\xi$. In other words, we maximize the variation that is purely contributed from x by excluding out the contribution from W via a standard reference ξ. A measure for such a relative variation can be obtained by combing the variation $V_y(W)$ for y and the variation $V_\zeta(W)$ for ζ. A general form is given as follows

$$J(W) = f(g_y(V_y(W)) - g_\zeta(V_\zeta(W))), \tag{20}$$

where $f(r), g_y(r), g_\zeta(r)$ are monotonically increasing as the scalar increases. E.g., two special examples are

$$\begin{aligned} J(W) &= V_y(W) - \lambda V_\zeta(W),\ with\ f(r) = r,\ g_y(r) = r,\ g_\zeta(r) = \lambda r, \\ J(W) &= \frac{V_y(W)}{V_\zeta^\lambda(W)},\ with\ f(r) = e^r,\ g_y(r) = \ln r,\ g_\zeta(r) = \lambda \ln r. \end{aligned} \tag{21}$$

Particularly, we simply have $\Sigma_\zeta = W\Lambda_\xi W^T$ in the case of eq.(15).

As firstly proposed in [119], there are two general directions to extend a PCA cost function to its nonlinear cases and thus to implement a sort of ICA.

The first is simply replace the linear system $y = Wx$ by a semi-linear system

$$y = s(\bar{y}),\ \bar{y} = Wx, with\ s(u) = [s(u^{(1)}), \cdots, s(u^{(m)})]^T, \tag{22}$$

where $-b < s(r) < a$ is a sigmoid function that monotonically increases as r increases.

A typical example is the LMSER learning, which is obtained by using eq.(22) to replace $y = Wx$ in eq.(18), resulting in

$$\sigma^2 = \tfrac{1}{N}\textstyle\sum_{t=1}^{N} \|x_t - W^T s(Wx_t)\|^2, \tag{23}$$

which was firstly studied in [123], with not only both a batch algorithm and an adaptive gradient algorithm provided, but also an experimental finding that the replacement of a linear neuron by a sigmoid neuron leads to an automatic breaking on the symmetry of the components in the subspace. Three years later, the LMSER learning and the adaptive algorithm given in [123] have been directly adopted to implement ICA with promising results under the name of nonlinear PCA [38]. Moreover, the relations of this LMSER learning to several other ICA approaches have been further explored by [38].

Similarly, corresponding to eq.(15) we have $\Sigma_y = E[s(Wx)s^T(Wx)]$ which considers higher order statistics indirectly via $s(.)$ for implementing ICA. Also, the constraints on W can be removed since $s(.)$ is already bounded.

The second general direction is to replace the second order error by other error measures. E.g., both in $E_2(W)$ and $V_y(W) = Tr[\Sigma_y] = E\|y\|^2$, the L_2 norm $\|u\|^2$ can be replaced either by a general $L_p, 0 < p < \infty$ norm or by a robust piecewise error measure as discussed in [121]. Alternatively, we can also consider the entire density as in eq.(16) and eq.(19).

Various extensions can also be obtained by combining the above two directions in different ways.

2.4. From PCA vs MCA to P-ICA vs M-ICA

In the literature, ICA is regarded as the extension of PCA [22]. Strictly speaking, this is inexact. ICA is the extension of a whitening process such as the above eq.(12) or called de-correlated component analyses (DCA).

De-correlated components require independence among the components of $y = Wx$ only in the second order of statistics, which is a special case of independent components that require independence among components $y = Wx$ in all the orders of statistics. PCA is one extreme case of DCA that chooses those de-correlated components with the first k largest variances [7, 55, 56, 123]. In contrast, we have the so called *Minor component analysis (MCA)* [124, 122, 58] that chooses those de-correlated components with the first k smallest variances. That is, MCA is another extreme case of DCA. Actually, there are a lots of combinations for various types of DCA.

PCA has a wide application in pattern recognition, signal processing, neural networks and various data analyses. PCA can be implemented adaptively per sample via a constrained Hebbian learning [55, 56, 123]. The components resulted from MCA spans a complement subspace that is orthogonal to the subspace spanned by the components resulted from PCA. In the area of signal processing, it has also been used for spectrum estimation under the name of SVD (singular value decomposition). It got the current name in [124] as being used together with PCA for representing pattern class in a so called dual pattern recognition approach. It has been further studied in [122] via an anti-Hebbian learning for adaptive implementation with application for the total least square fitting and extension to higher order Hebbian learning. Hereafter, MCA has been widely studied in the literature of neural networks.

PCA and MCA are a pair of dual representations. The adapting direction is either Hebbian xy^T for PCA or anti-Hebbian $-xy^T$, subject to $WW^T = I$. Though it looks only a sign difference in the learning direction. An appropriate implementation, that ensures W eventually satisfying $WW^T = I$ and the whole learning process converged, is different for PCA and MCA. Thus, in the level of updating rules, several existing rules for PCA can not be turned into ones for MCA simply via changing the sign of adapting direction. Recently, efforts have been made on searching such a rule that can implement PCA and MCA by simply changing the sign of learning [19].

We would like to argue that this apparent difficulty is actually not difficult to solve. E.g., $\min_W E_2(W)$ in eq.(18) results in a principal subspace with $WW^T = I$ satisfied automatically. In contrast, we can not get a minor subspace directly by $\max_W E_2(W)$ since E_2 will tend to infinity. However, the apparent difficulty comes from a simplification that discards a type of constraints by eq.(17). Restoring this constraint, $\max_W E_2(W)$ will become bounded and W does stably reach a solution that spans the minor subspace. Similarly, it follows from eq.(13) that $\min_{W,\ \|W\| \leq c} V_y(W)$ will lead to the minor subspace. In other words, the dual representation does apply under the constraint by eq.(17), though this constraint may sometimes be ignored for PCA but not ignorable for MCA.

Without losing generality, we take the following general form as an example:

$$\min J(W), W = D(UU^T)^{-0.5}U \tag{24}$$

that leads to PCA via a unconstrained minimization. In this case, $WW^T = D^2$ is always satisfied for any matrix U and the learning process on W will be stabilized though the learning process on D remains not converged. Given the gradient $G_W = \nabla_W J(W)$, it further follows from $dJ(W) = G_W dW$ and $dW = G_{\hat{W}}^T d(D(UU^T)^{-0.5}U)$ that we get the gradient direction

$$\nabla_U J(W) = (UU^T)^{-0.5}[(UU^T)^{0.5}DG_W - 0.5(UG_W^T D + DG_W U^T)](UU^T)^{-0.5}. \tag{25}$$

Then we have the updating

$$U^{new} = U^{old} + \eta \begin{cases} -\nabla_U J(W), & \text{for PCA,} \\ \nabla_U J(W), & \text{for MCA;} \end{cases} \ \eta \text{ is a learning stepsize.} \tag{26}$$

That is, there is only a difference in a sign for PCA versus MCA. Though U^{new} may not stabilize, we can terminate the learning process once W becomes stabilized or $J(W)$ becomes maximized or minimized.

Furthermore, extending the de-correlation among the components of y to the independence among components y in all the orders of statistics, we correspondingly have

(a) For PCA to principal ICA (P-ICA) that chooses those independent components with the first k largest variances. For an example, the LMSER learning by eq.(23) actually implements a P-ICA.

(b) For MCA to minor ICA (M-ICA) that chooses those independent components with the first k smallest variances. For an example, when $WW^T = I$ and $q(y) = G(y|0, I)$, it follows from eq.(9) that $g_W(t) = -y_t x_t^T$. That is, it is an anti-Hebbian learning for MCA. Thus, eq.(8) actually implements a M-ICA.

3. BYY Independence Learning

3.1. Bayesian Ying-Yang System

Another type of cost function comes from the so called BYY independence learning, a special case of Bayesian Ying-Yang (BYY) harmony learning that was firstly proposed in [119] and then developed in past several

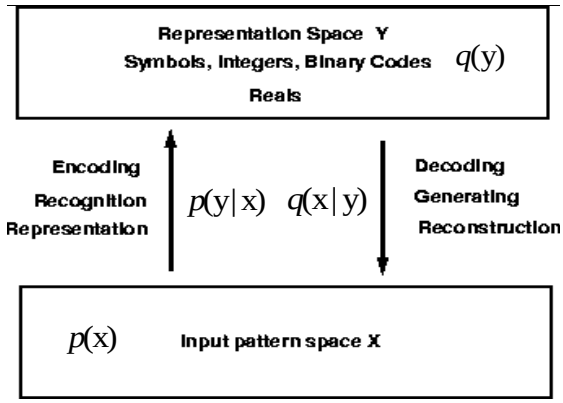


Figure 1. *Bayesian Ying-Yang System*

years.

As shown in Fig.1, we consider the joint distribution of an observation x and its inner representation in a learning system y from two complement aspects. On one hand, each x is interpreted as generated via a backward path $q(x|y)$ from an inner distribution $q(y)$ in a structure subject to certain learning tasks, i.e.,

$$q(x) = \int q(x|y)q(y)dy. \tag{27}$$

On the other hand, each x is interpreted as being mapped into an inner representation y via a forward path $p(y|x)$

$$p(y) = \int p(y|x)p(x)dx \tag{28}$$

to match the target density $q(y)$. The two aspects reflect the two types of Bayesian decomposition of the joint density $q(x|y)q(y) = q(x,y) = p(x,y) = p(x)p(y|x)$. Without any constraint, the two should be theoretically identical. However, the two types of Bayesian decomposition may not necessarily equal in a practical consideration and we denote them by two different notations p, q. The differences may come from different biological/physical/structural constraints on the four components $p(y|x), p(x)$, $q(x|y)$, and $q(y)$. Thus, we usually have two different but complementary Bayesian representations:

$$p(x,y) = p(y|x)p(x),\ q(x,y) = q(x|y)q(y), \tag{29}$$

which compliments to the famous Chinese ancient Ying-Yang philosophy with $p(x,y)$ called Yang machine that consists of the observation space (or called Yang space) by $p(x)$ and the forward pathway (or called Yang pathway) by $p(y|x)$, and with $q(x,y)$ called Ying machine that consists of the invisible domain (or Ying space) by $q(y)$ and the Ying (or backward) pathway by $q(x|y)$. Such a pair of Ying-Yang models is called *Bayesian Ying-Yang (BYY) system*.

Usually, $p(x)$ is given by a Parzen window estimate:

$$p_{h_x}(x) = \frac{1}{N}\sum_{t=1}^{N} G(x|x_t, h_x^2 I), \tag{30}$$

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

$$p_0(x) = \frac{1}{N}\sum_{t=1}^{N}\delta(x - x_t),\ \text{where } \delta(x) \text{ is a } \delta\text{-function.} \tag{31}$$

The task of learning on a BYY system consists of specifying the rest three components of the system.

As shown in Fig.2, the architecture of a BYY system is featured by a combination of specific structures of $p(y|x), q(x|y)$. Depending on which one or both of $p(y|x), q(x|y)$ are parametric, there are three typical architectures, namely, Backward architecture, Forward architecture, and BI-directional architecture, or shortly B-architecture, F-architecture, and BI-architecture, respectively. Each of three can implement a learning task of the same nature, but from a different perspective and with a different performance.

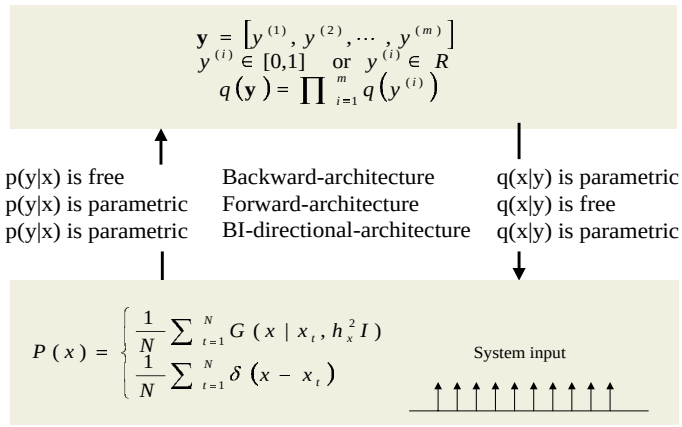


Figure 2. *Three Typical Architectures*

The structure of $q(y)$ describes the nature of the inner representation of x, and thus features the nature of learning tasks that a specific BYY system performs. Considering $q(y)$ given by eq.(3) as shown in Fig.2, we get an independence BYY system that performs learning tasks of ICA and various extensions. Moreover, considering $q(y)$ given by other structures, we will lead other typical learning tasks [94, 92, 93, 89, 90, 87].

Given a specific BYY architecture with $q(y)$ in a specific structure, we have two further tasks. One is parameter learning for determining the value of θ that consists of all the unknown parameters in $p(y|x)$, $q(x|y)$, $q(y)$ as well as h_x (if any). The other is selecting the dimension m of the vector $y = [y^{(1)}, \cdots, y^{(m)}]^T$. We call the second task as *model selection* since a collection of specific BYY systems with different values of m corresponds to a family of specific models that share a same system configuration but in different scales.

3.2. BYY Harmony Learning

The basic learning principle is to make the Ying machine and Yang machine be best harmony in a twofold sense: Mathematically, this principle can be implemented by [119, 92, 93, 89].

$$\max_{\theta,m} H(\theta, m), \; H(\theta, m) = H(p\|q), \text{ with } p, q \text{ being the Ying-Yang pair by eq.(29),} \tag{32}$$

where $H(p\|q)$ is a harmony measure as follows:

$$H(p\|q) = \int p(u) \ln q(u) du - \ln z_q. \tag{33}$$

This measure comes from the following cross entropy

$$H(p\|q) = \sum_{t=1}^{N} p_t \ln q_t, \tag{34}$$

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

$$p(u) = \sum_{t=1}^{N} p_t \delta(u - u_t), \; \sum_{t=1}^{N} p_t = 1, \; \delta(u) = \begin{cases} \lim_{\delta u \to 0} \frac{1}{\delta u}, & u = 0, \\ 0, & \text{otherwise,} \end{cases} \tag{35}$$

where δu is the infinitesimal volume at u. It can be observed that maximizing $H(p\|q)$ by eq.(34) has the following two interesting natures:

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

$$q_t = p_t, \quad \text{for all } t. \tag{36}$$

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

$$p(u) = \delta(u - u_\tau), or \; p_t = \bar{\delta}_{t,\tau}, \; with \; \tau = arg\max_t \; q_t, \; \bar{\delta}_{i,j} = \begin{cases} 1, & i = j, \\ 0, & \text{otherwise.} \end{cases} \tag{37}$$

As discussed in [92, 89], eq.(37) is a kind of the least complexity form from a statistical perspective. In other words, the maximization of the functional $H(p\|q)$ indeed implements the above harmony learning (shortly HL) principle mathematically.

Given a set $\{u_t\}_{t=1}^N$ of samples, either a discrete or continuous density $p(u), q(u)$ can be represented in the form of eq.(35) via the following normalization:

$$p_t = p(u_t)/z_p,\ z_p = \sum_{t=1}^N p(u_t),\ q_t = q(u_t)/z_q,\ z_q = \sum_{t=1}^N q(u_t). \tag{38}$$

Considering an infinitesimal volume δu, we approximately have $z_p \delta u = \sum_{t=1}^N p(u_t)\delta u \approx \int p(u)du = 1$ when N is large enough. Thus $\sum_{t=1}^N (p(u_t)/z_p)\ln q(u_t) = \sum_{t=1}^N p(u_t)\delta u \ln q(u_t) \approx \int p(u)\ln q(u)du$. It further follows from eq.(34) that we are lead to eq.(33). Clearly, it returns to eq.(34) when $p(u)$ and $q(u)$ are discrete.

In a simplest special case that We know $p(u)$ in eq.(35) with $p_t = 1/N$, the least complexity nature by eq.(37) becomes no use. Moreover, as discussed in [92, 93, 89], the matching nature by eq.(36) becomes equivalent to implementing the maximum likelihood (ML) learning principle on a parametric model $q(u)$ if the role of $-\ln z_q$ is ignored by simply setting $z_q = 1$. In general cases, however, the HL learning differs from the ML learning with two extra features.

First, the term $-\ln z_q$ takes a role that regularizes learning on a finite size of samples. As will be further discussed in Sec.3.3, this type of regularization is shortly called z-regularization which further consists of two regularization techniques, namely data-smoothing and normalization.

Second, with p, q being the Ying-Yang pair by eq.(29) where only $p(x)$ is known directly from a given set of samples by eq.(30), the least complexity nature by eq.(37) does bring in another big different role that makes model selection on m become possible. A detailed discussion can be found in [89]. In implementation, the model selection on m can be made either during parameter learning on θ or after parameter learning in a conventional two stage style. Specifically, we have the following two implementation strategies for eq.(32).

Parallel Implementation As discussed in [92, 93, 89], letting the variance of $q(y^{(j)})$ to be zero is equivalent to removing the j-th dimension (i.e., reducing the dimension m by one), while the least complexity nature by eq.(37) does push $q(y^{(j)})$ towards a δ-density for any extra dimension. Thus, we can set m large enough and then implement:

$$\max_{\theta}\ H(\theta), \quad H(\theta) = H(\theta, m), \tag{39}$$

which makes θ take a specific value such that m is effectively reduced to an appropriate one, i.e., model selection is made automatically in parallel to parameter learning. As demonstrated in Fig.3(b), the learning by eq.(39) will push a set of extra parameters $\theta_2 = \theta_2^*$ such that a large m becomes effectively m^*, that is, a smallest value of m at which the maximum of $H(\theta)$ is reached.

This feature is not shared by the existing approaches in literature. By the conventional approaches, parameter learning and model selection are made in a two-phase style. First, parameter learning is made usually under the maximum likelihood principle. Then, model selection is made by a different criterion, e.g., AIC, BIC, and MDL, etc [1, 13, 67, 68]. These model selection criteria are usually not good for parameter learning, while the maximum likelihood criterion is not good for model selection, especially on a small size of training samples.

Two Stage Implementation If we want, the problem eq.(32) can also be implemented in two stages as follows:

Stage I enumerate m from a small value incrementally, and at each value we perform parameter learning eq.(39) to get the best parameter value θ^*. A direct implementation of eq.(39) will lead to the above discussed case of Fig.3(b) again, which can be avoided by simply assuming that $q(y)$ comes from a family with equal variances among components. That is, the covariance matrix is $b_0 I$. E.g., $b_0 = 1$ for a real $y^{(j)}$ and $b_0 = 0.25$ for a binary $y^{(j)}$ [89].

Stage II select a best m^* by

$$\min_{\{1 \le m \le m^u\}}\ J(m),\ \ J(m) = -H(\theta^*, m). \tag{40}$$

where m^u is an upper bound. As demonstrated in Fig.3(a), the learning by eq.(32) should be made at every m as it is enumerated from in a possible range of m. With $q(y)$ constrained to a fixed covariance matrix $b_0 I$, $J(m)$ will takes a shape as shown in Fig.3(a). The maximum of $H(\theta)$ will be reached at one value m^* as m increases.

To get an insight, we consider the classic factor analysis by eq.(4) and eq.(5) with $\Sigma = \sigma^2 I$ with $z_q = 1$. In this case, we have [89]:

$$J(m) = 0.5[d\ln|\sigma^2| + m(\ln 2\pi + 1)]. \tag{41}$$

Tab.2 Typical Techniques for Implementing Learning

(A) **Find** $y_t = arg\max_y[G(x_t|Ay+\mu,\Sigma)\prod_{j=1}^m q(y^{(j)})]$

y_t is obtained by $\begin{cases}\text{running step (a) \& step (b) iteratively,} & \text{(i) for NFA,}\\ \text{running step (a) \& step (b) only once,} & \text{(ii) all the other cases.}\end{cases}$

Step (a): $\lambda_j^2 = \begin{cases}1, & \text{(a) for FA with } q(y^{(j)}) = G(y^{(j)}|0,1) \\ & \quad \text{and TFA-I with } q(y^{(j)}) = G(y^{(j)}|b_j y_{t-1}^{(j)},1),\\ \lambda_j^2, & \text{(b) for Orthogonal FA \& TFA-II with } q(y^{(j)}) = G(y^{(j)}|0,\lambda_j^2),\\ \sum_r \frac{p_{j,r}}{\lambda_{j,r}^2}, & \text{(c) for NFA with } q(y^{(j)}) = \sum_i \beta_{ji} G(y^{(j)}|m_{ji},\lambda_{ji}^2).\end{cases}$

$f_j = \begin{cases}0, & \text{(a) for FA with } q(y^{(j)}) = G(y^{(j)}|0,1) \\ & \quad \text{and for Orthogonal FA \& TFA-II with } q(y^{(j)}) = G(y^{(j)}|0,\lambda_j^2),\\ b_j y_{t-1}^{(j)}, & \text{(b) for TFA-I with } q(y^{(j)}) = G(y^{(j)}|b_j y_{t-1}^{(j)},1),\\ \sum_r \frac{p_{j,r} m_{j,r}}{\lambda_{j,r}^2}, & \text{(c) for NFA with } q(y^{(j)}) = \sum_i \beta_{ji} G(y^{(j)}|m_{ji},\lambda_{ji}^2);\end{cases}$

where we get $p_{j,r} = \frac{\beta_{j,r} G(y^{(j)\ old}|m_{j,r},\lambda_{j,r}^2)}{\sum_i \beta_{ji} G(y^{(j)\ old}|m_{j,i},\lambda_{j,i}^2)}$ for NFA.

Step (b): $y_t = (A^T\Sigma^{-1}A + diag[\lambda_1^2,\cdots,\lambda_m^2]^{-1})^{-1}(A^T\Sigma^{-1}(x_t-\mu) + [f_1,\cdots,f_m]^T)$.

(B) Updating Constrained Parameters

(a) One-Direction Updating with $\eta_t > 0$

Update $\Sigma^{new} = (1-\eta)\Sigma^{old} + \eta_0\eta_t(h_x^2 I + e_t e_t^T)$ for increasing $\eta_t \ln G(e_t|0,\Sigma)$ subject to $\Sigma > 0$.

Update $q_j^{new} = (1-\eta)q_j^{old} + \eta_0\eta_t y_t^{(j)}$ for increasing $\eta_t \ln[q_j^{y^{(j)}}(1-q_j)^{1-y^{(j)}}]$ subject $0 \le q_j \le 1$.

Update $\alpha_j^{new} = \frac{\alpha_j^{old} + \eta_0\eta_t\gamma_{t,j}}{1+\eta_0\eta_t}$ for increasing $\eta_t \sum_{j=1}^k \gamma_{t,j}\ln\beta_j$ subject $0 \le \alpha_j \le 1, \sum_{j=1}^k \alpha_j = 1$.

(b) Two-Direction Updating with η_t being negative or positive

Update $S^{new} = S^{old} + \eta_0\eta_t G_\Sigma^{old} S^{old}, G_\Sigma = \Sigma^{-1}(h_x^2 I + e_t e_t^T)\Sigma^{-1} - \Sigma^{-1}$

for increasing $\eta_t \ln G(e_t|0,\Sigma)$ such that $\Sigma^{new} = S^{new} S^{new\ T}$ remains to be positive definite.

Update $c_j^{new} = (1-\eta)c_j^{old} + \eta_0\eta_t(y_t^{(j)} - c_j^{old})$

for increasing $\eta_t \ln[q_j^{y^{(j)}}(1-q_j)^{1-y^{(j)}}]$ such that $q_j = 1/(1+e^{-c_j})$ always satisfies $0 \le q_j \le 1$.

Update $a_j^{new} = a_j^{old} + \eta_0\eta_t(\gamma_{t,j} - \alpha_j\sum_{r=1}^k \gamma_{t,r})$

for increasing $\eta_t \sum_{j=1}^k \gamma_{t,j}\ln\alpha_j$ such that $\alpha_j = e^{a_j}/\sum_{r=1}^k e^{a_r}$ satisfies $0 \le \alpha_j \le 1, \sum_{j=1}^k \alpha_j = 1$.

(C) An Approximation of $\int p(u)T(u)du$ (Xu, 1999a; Xu, 2000a)

$T(u) \approx T(\bar{u}) + \nabla_u T(\bar{u})^T(u-\bar{u}) + 0.5(u-\bar{u})^T H(\bar{u})(u-\bar{u})$, $\bar{u} = \int up(u)du$, $H(u) = \frac{\partial^2 T(u)}{\partial u \partial u^T}$,

$\int p(u)T(u)du = T(\bar{u}) + 0.5Tr[\Sigma_u^{-1}H(\bar{u})]$ with $\Sigma_u = \int(u-\bar{u})(u-\bar{u})^T p(u)du$.

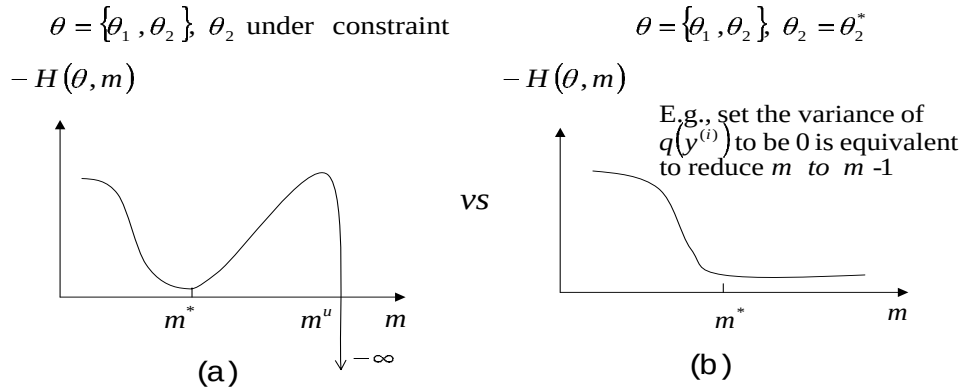


Figure 3. *(a) Model selection made after parameter learning on every m in a given interval $[m_d, m_u]$, (b) Automatic model selection with parameter learning on a value m of large enough.*

Given a finite set of samples, σ^2 decreases with m and the 1st term of $J(m)$ decreases with m, while the second term trades off this decreasing such that $J(m)$ first decreases with m and reaches a minimum, and then increases. Moreover, as m further increases, σ^2 will finally become zero which brings $J(m)$ drops rapidly towards $-\infty$. That is, $J(m)$ generally has an inverse N shape as shown in Fig.3(a). We use m^u to denote the smallest value of m that makes $J(m)$ rapidly tend to $-\infty$. The best dimension m^* is selected by eq.(40) within $1 \leq m \leq m^u$, when the model by eq.(4) is generally considered with noise (i.e., $\sigma^2 \neq 0$). Particularly, the best dimension should be selected at m^u in a case of eq.(4) without noise (i.e., $\sigma^2 = 0$).

Alternatively, we can make parameter learning in Stage I with eq.(39) replaced by

$$\min_{\theta} \; KL(\theta) = \int p(y|x)p(x) \ln \frac{p(y|x)p(x)}{q(x|y)q(y)} dxdy, \tag{42}$$

which has been systematically investigated in the early study of the BYY learning [119], which is shortly called KL learning hereafter.

Particularly, on a B-architecture, the minimization of the above $KL(\theta)$ with respect to a free $p(y|x)$ results in

$$p(y|x) = \frac{q(x|y)q(y)}{q(x)}, \; q(x) = \int q(x|y)q(y)dy, \; KL(\theta) = \int p(x) \ln \frac{p(x)}{q(x)} dx, \tag{43}$$

which becomes equivalent to ML learning on $q(x)$ when $p(x) = p_0(x)$ is given by eq.(31) [113, 119]. In this case, we actually implement the ML learning in the first phase and then model selection by eq.(40) in the second phase.

Without the least complexity nature by eq.(37), the implementation of eq.(42) will not lead to a case of Fig.3(b), and thus there is no need to impose the assumption that $q(y)$ comes from a family with equal variances among components.

3.3. Regularization vs Model Selection: (I) ML, HL, and z-Regularization

Regularization and model selection are two different strategies for tackling the problem of a finite size of samples. Model selection prefers a model of least complexity for which a compact inner representation y is aimed at such that extra representation space can be released. In contrast, regularization is imposed on a model with a fixed scale of representation space (e.g., the dimension of y) that is larger than needed such that inner representation can spread as uniformly as possible over all the representation space with a distribution that is as simple as possible, which thus becomes equivalent to a model with a reduced complexity.

The harmony learning by eq.(39) attempts to compress the representation space via the least complexity by eq.(37). On a B-architecture, it is demonstrated during the maximization of $H(\theta)$ with respect to a free $p(y|x)$. It results in

$$p(y|x) = \delta(y - y(x)), \; y(x) = arg \max_{y} [q(x|y)q(y)], \tag{44}$$

which concentrates to the peak point of $p(y|x)$, that is, we get a winner-take-all (WTA) competition variant of eq.(43). As discussed in [94, 92, 93, 89, 90, 87], it is this least complexity nature that makes the BYY learning by eq.(32) aim at a compact inner representation with an automatic model selection by discarding extra representation space during parameter learning. However, there is no free lunch. The WTA operation by eq.(44) locally per sample will make the learning become sensitive to the initialization of parameters and the way that samples are presented, resulting in that samples are over-aggregated in a small representation space. It usually leads to a local maximum solution for eq.(39). Pre-specifying a uniform inner representation (e.g., imposing that $q(y)$ has a unity covariance matrix I) can regularizes the WTA operation. However, the feature of automatic model selection is also lost since the representation space scale is already fixed. Thus, model selection should be made by eq.(40) in the second phase.

With a soft competition by eq.(43) to replace the WTA competition by eq.(44), the ML learning or equivalently the KL learning by eq.(42) with a B-architecture and $p(x) = p_0(x)$ by eq.(31) is regularized with a more spread inner representation that improves the local maximum problem. Again, there is no free lunch. The model selection ability becomes considerably weaken, especially on a small size of samples. Thus, a model selection by eq.(40) is needed in the second phase too.

Instead of the two phase style, regularization to the WTA by eq.(44) may also be imposed to the harmony learning by eq.(39) such that automatic model selection still occurs via either some external help or certain internal mechanism.

Externally, we can combine the KL learning by eq.(42) with the harmony learning by eq.(39), in the following three ways:

- The simplest way is to make the KL learning by eq.(42) with the resulted parameters as the initialization of the harmony learning by eq.(39).
- The other way suggested in [93] is to let $H(\theta)$ in eq.(39) replaced with $(1-\lambda)H(\theta) - \lambda KL(\theta)$ with an appropriate λ.
- Moreover, with $\lambda > 0$ gradually reducing towards zero from a given value such that the regularization role by eq.(42) takes effect at the beginning and then gradually decades as learning goes. That is, we combine KL and HL in a simulated annealing way such that KL is implemented in an early period of learning and is gradually switched to HL as learning goes [93, 88]. The disadvantage is that the computing cost is very high since parameter learning has to be repeatedly conducted.

Internally, regularization to the WTA by eq.(44) may also be imposed during the harmony learning by eq.(39) via either a BI-architecture or the role of z_q.

Instead of letting $p(y|x)$ free to be decided by eq.(44), we consider a BI-architecture with $p(y|x)$ designed in a structure such that it is not able to become the WTA by eq.(44). Specifically, $p(y|x)$ can be a structure that is either bundled with the Ying machine $q(x|y)q(y)$ (e.g., via the posteriori estimation by eq.(43) [93, 46]) or decoupled from the Ying machine to facilitate computation. For the latter case, as suggested in [99, 96, 94], when y is a binary vector we have

$$p(y|x) = \prod_{j=1}^{m} \pi_j(x)^{y^{(j)}} (1-\pi_j(x))^{1-y^{(j)}}, \ \pi(x) = [\pi_1(x), \cdots, \pi_m(x)]^T = s(Wx+c), \tag{45}$$

where $s(y)$ is same as in eq.(22). When y is a real vector, we may have

$$p(y|x) = \sum_{j=1}^{n} \beta_j(x) G(y|f_j(x|\theta_j), \Sigma_j), \ \sum_{j=1}^{n} \beta_j(x) = 1, \ \beta_j(x) \geq 0. \tag{46}$$

In the case of eq.(46), the harmony learning by eq.(39) will push it towards the following form of least complexity [94, 92, 89]

$$p(y|x) = \sum_{j=1}^{n} \beta_j(x) \delta(y - f_j(x, \theta_j)), \ \sum_{j=1}^{n} \beta_j(x) = 1, \ \beta_j(x) = 0 \ or \ 1, \tag{47}$$

unless extra constraints are imposed to prevent $\Sigma_j \to \varepsilon^2 I$ and $\varepsilon^2 \to 0$.

Moreover, eq.(44) is simplified into

$$p(y|x) = \delta(y - y(x)), \ y(x) = f_{j^*(x)}(x|\theta_{j^*(x)}), \ j^*(x) = arg \max_j [q(x|y)q(y)]_{y=f_j(x|\theta_j)}, \tag{48}$$

where the maximum is searched by simply enumerating n possibilities. Thus, regularization can also be observed from the perspective that the number of local maximums considerably reduces in comparison with eq.(44). However, there is no free lunch too. The problem is transferred to the difficulty of per-specifying the function form of each $y = f_j(x|\theta_j)$. If it is too simple, the representation ability of $p(y|x)$ is limited and is far from the optimal one. If it is too complicated with too much free parameters, it creates certain problems that needs regularization to be imposed too.

Regularization to the WTA by eq.(44) may also be internally imposed by the z-regularization. This type of regularization can be implemented by either data smoothing or normalization via z_q.

For data smoothing regularization, one simple way is considering smoothing on x via $p(x) = p_{h_x}(x)$ by eq.(30) with $z_q = \sum_{t=1}^{N} p_{h_x}(x_t)$. As discussed in [89, 93], the regularization is made via $h_x^2 > 0$ while h is determined in help of $-\ln z_q$. Moreover, a smoothing can be imposed on y via replacing $p(y|x) = \delta(y - y(x))$ in eq.(44). E.g., in the case of only one object (i.e., $k = 1$), we let $p(y|x) = G(y|y_t, h_y^2 I)$ and $z_q = (2\pi h_y)^{-0.5m} \sum_{t=1}^{N} p_h(x_t)$.

For normalization regularization, we have also different choices for implementing z_q. Strictly speaking, we should consider

$$z_q = \textstyle\sum_{t=1}^{N} q(x_t), \quad q(x_t) = \int q(x_t|y)q(y)dy. \tag{49}$$

This $q(x_t)$ can be accurately computed when y takes either a discrete value $1, \cdots, k$ or is a binary vector $y = [y^{(1)}, \cdots, y^{(m)}]$. In this case, the integral over y becomes a summation. The integral can also be analytically solved when $q(x|y)$ and $q(y)$ are both Gaussian. In other cases, this integral is difficult to compute. Even when it becomes a computable summation for a binary vector $y = [y^{(1)}, \cdots, y^{(m)}]$, the computing cost will increase exponentially with m.

One solution is to let the integral over the entire domain of y to be approximated by a summation on a set Y_t that consists of a few number of samples y_τ as follows:

$$q(x_t) = \gamma_t \sum_{y_\tau \in Y_t} q(x|y_\tau) q(y_\tau), \ \gamma_t = \begin{cases} 1, & \text{(a)}, \\ \frac{1}{\sum_{y_\tau \in Y_t} q(y_\tau)}, & \text{(b)}. \end{cases} \tag{50}$$

Specifically, γ_t in the choice (b) makes $q(y_\tau)/\sum_{y_\tau \in Y_t} q(y_\tau)$ represent discrete probabilities that weight $q(x|y_\tau)$ such that $q(x_t)$ is closer to a marginal density, while the γ_t in the choice (a) provides a crude simplification that we no longer have $\int q(x)dx = 1$.

As will be further discussed later on the Yang step in Sec.3.5, the set Y_t can be obtained according to $p(y|x)$. One way is randomly picking a set of samples of y according to $p(y|x)$. The other two ways are getting only one $y_t = y(x_t)$ for each x_t via either the peak point (e.g., by eq.(44)) or the mean point (e.g., $y(x)$ by eq.(48)) of $p(y|x)$. In these cases, there is only one sample y_t in Y_t and thus

$$z_q = \begin{cases} \sum_{t=1}^N q(x_t|y_t) q(y_t), & \text{choice (a)}, \\ \sum_{t=1}^N q(x_t|y_t), & \text{choice (b)}. \end{cases} \tag{51}$$

Further with $p(x) = p_{h_x}(x)$ given by eq.(30), $H(p\|q)$ by eq.(33) either on a B-architecture with eq.(44) or on a BI-architecture with eq.(48)) can be unified into the following representation

$$\begin{aligned} H(p\|q) &= \frac{1}{N} \sum_{t=1}^N \int \delta(y - y(x_t)) \ln [q(x_t|y) q(y)] dy - \ln z_q + 0.5 {h_x}^2 \pi_q, \\ \pi_q &= \frac{1}{N} \sum_{t=1}^N Tr[\frac{\partial^2 \ln q(x|y_t)}{\partial x \partial x^T}]_{x=x_t}, \end{aligned} \tag{52}$$

where $\nabla_\theta H(p\|q)$ denotes the gradient of H with respect to θ, which takes the following form

$$\begin{aligned} \nabla_\theta H(p\|q) &= \tfrac{1}{N} \sum_{t=1}^N \int \eta_t(y) \nabla_\theta \ln [q(x_t|y) q(y)] dy + 0.5 {h_x}^2 \nabla_\theta \pi_q, \\ \eta_t(y) &= \begin{cases} \delta(y - y(x_t)), & \text{for } z_q = 1, \\ \delta(y - y(x_t)) - \bar{\delta}(h_x) \sum_{y_\tau \in Y_t} \frac{q(x_t|y_\tau) q(y_\tau)}{z_q} \delta(y - y_\tau), & \text{for } z_q \text{ given by eq.(49)}, \\ \delta(y - y(x_t)) - \bar{\delta}(h_x) \sum_{y_\tau \in Y_t} \gamma_t \frac{q(x_t|y_\tau) q(y_\tau)}{z_q} \delta(y - y_\tau), & \text{for } z_q \text{ given by eq.(50)}, \end{cases} \\ \bar{\delta}(h) &= \begin{cases} 1, & \text{for } h = 0, \\ 0, & \text{for } h \neq 0. \end{cases} \end{aligned} \tag{53}$$

For the case $h_x > 0$, we have $\eta_t(y) = \delta(y - y(x_t))$ for all the cases since $\bar{\delta}(h_x) = 0$. The data smoothing regularization is imposed via $0.5 {h_x}^2 \nabla_\theta \pi_q$ and an appropriate regularization strength h_x^2 is determined via maximizing

$$J(h_x) = -\ln z_q + 0.5 {h_x}^2 \pi_q, with \ z_q = \sum_{t=1}^N p_{h_x}(x_t). \tag{54}$$

The details are referred to [92, 93, 89, 87].

For $h_x = 0$, we have $0.5 {h_x}^2 \nabla_\theta \pi_q = 0$ and $\bar{\delta}(h_x) = 1$. In this case, the normalization regularization is imposed via $-\ln z_q$, which can be observed via the difference of $\eta_t(y)$ in eq.(53). It introduces a degree of conscience de-learning on each updating direction $\nabla_\theta \ln [q(x_t|y) q(y)]$ to avoid over-fitting on each sample pair x_t, y_τ. With and without $-\ln z_q$ in its role, $\nabla_\theta H(p\|q)$ takes the same format and also adaptive updating can be made in the form of $\eta_t(y) \nabla_\theta \ln [q(x_t|y) q(y)]$ per sample x_t.

3.4. Regularization vs model selection: (II) KL-λ- HL Spectrum

The KL learning by eq.(42) on a BYY system is not limited to just the ML learning only. Even on a B-architecture with $p(y|x)$ determined by eq.(43) but $p(x) = p_h(x)$ by eq.(30), the KL learning by eq.(42) still performs a regularized ML learning.

Moreover, the KL learning by eq.(42) on a BI-architecture was suggested in [119] with $p(y|x)$ in a given parametric family $\mathcal{P}^S_{y|x}$. If the posteriori estimation by eq.(43) is contained in this family $\mathcal{P}^S_{y|x}$, the situation will be equivalent to the KL learning by eq.(42) with a B-architecture; if not, the posteriori estimation by eq.(43) will be approximated by the closest one within the family $\mathcal{P}^S_{y|x}$. This architecture leads to an advantage that the computing difficulty on the integral in $p(y|x)$ by eq.(43) is avoided by an easy implementing parametric model. In its sprit, this is equivalent to those approaches called variational approximation to the ML learning on $q(x)$ [73].

Beyond the approximation purpose, studies on the KL learning by eq.(42) on a BI-architecture were also made along two directions since [119]. One is to design a parametric model that makes the inner representation more spreading than that of $p(y|x)$ by eq.(43) such that ML learning is further regularized. The other is to design a parametric model that makes the inner representation more concentrated such that it tends to facilitate automatic model selection. One family of of such designs is as follows:

$$p(y|x) = \frac{\psi(q(x|y), q(y))}{\int \psi(q(x|y), q(y))dy}, \tag{55}$$

which returns to $p(y|x)$ by eq.(43) for the ML learning when $\psi(\xi, \eta) = \xi\eta$. It makes the inner representation either more spreading for a regularized ML learning (e.g., when $\psi(\xi, \eta) = \lambda_1 \ln \xi + \lambda_2 \ln \eta, \lambda_1 \geq 0, \lambda_2 \geq 0$), or more concentrated to facilitate model selection (e.g., when $\psi(\xi, \eta) = e^{\lambda_1 \xi} + e^{\lambda_2 \eta}, \lambda_1 \geq 0, \lambda_2 \geq 0$ that will make $p(y|x)$ tend to eq.(44) as $\lambda_1 = \lambda_2 \to \infty$). A simply form can even be $\psi(\xi, \eta) = (\xi\eta)^\lambda$ which varies from spreading cases to concentrated cases as λ increases from 0 to ∞.

As discussed in the previous subsection, the harmony learning with the WTA by eq.(44) on a B-architecture will also be regularized by a BI-architecture with $p(y|x)$ in a more spreading representation. Except those extreme cases that become equivalent to the WTA by eq.(44) (e.g., when $\psi(\xi, \eta) = (\xi\eta)^\lambda$ with $\lambda \to \infty$), $p(y|x)$ by eq.(55) generally leads to a regularized harmony learning. Even when $\psi(\xi, \eta) = \xi\eta$, we will not be lead to the ML learning but to a harmony learning regularized from a B-architecture with a free $p(\mathbf{y}|\mathbf{x})$ replaced by a posteriori estimation by eq.(43).

From the above discussion, we observe that the KL learning by eq.(42) and the harmony learning by eq.(39) become closely related via appropriately designing $p(y|x)$. The difference lays in the following term E_p:

$$E_p = -\int p(y|x)p(x) \ln [p(y|x)p(x)]dxdy + \ln z_p. \tag{56}$$

When $p(y|x) = \delta(y - y(x))$ is deterministic and $p(x) = p_0(x)$ given by eq.(31), we have $E_p = 0$. That is, the KL learning by eq.(42) and the harmony learning by eq.(39) becomes equivalent on this special BI-directional architecture. Moreover, the harmony learning by eq.(39) on a B-architecture results in $p(\mathbf{y}|\mathbf{x})$ by eq.(44). With this particular $p(\mathbf{y}|\mathbf{x})$, the harmony learning by eq.(39) and the KL learning by eq.(42) become equivalent.

Furthermore, the KL learning by eq.(42) and the harmony learning by eq.(39) are also related for those $p(y|x)$ such that $E_p = c \neq 0$ becomes a constant irrelevant to any unknown parameters in θ, e.g., with $p(x) = p_0(x)$ given by eq.(31) we have

$$E_p = 0.5m \ln (2\pi h_y^2) - \ln N, \ for\ p(y|x) = G(y|y(x), h_y^2 I). \tag{57}$$

Thus, the KL learning by eq.(42) and the harmony learning by eq.(39) are no longer equivalent when h_y, m are unknown to be determined. In the special case that h_y, m are prefixed in advance, the KL learning by eq.(42) further becomes equivalent to

$$\max_{\theta,\ s.t.\ E_p = c \neq 0} H(\theta). \tag{58}$$

The above discussions also apply to BYY systems with a F-architecture, with a free $p(x|y)$ decided by

$$p(x|y) = p(y|x)p(x)/p(y),\ p(y) = \int p(y|x)p(x)dx, \tag{59}$$

such that we have

$$KL(\theta) = \int p(y) \ln \frac{p(y)}{q(y)} dy,\ H(\theta) = -KL(\theta) - E_p, \tag{60}$$

with E_p given in eq.(56). When $q(y)$ is a uniform distribution, minimizing this $KL(\theta)$ becomes equivalent to maximizing the entropy by eq.(16), which maximizes the information transfer from input data to its inner representation via the forward path. Generally, $-KL(\theta)$ describes the incremental of information contained in the

representation of y after this information transfer. Thus, minimizing this $KL(\theta)$ is equivalent to making this incremental maximized via maximizing this information transfer against upon the information already in the inner representation. Particularly, when $p(y|x) = \delta(y - Wx)$, $q(y)$ by eq.(3), and $p(x) = p_0(x)$ by eq.(31), both the KL learning by eq.(42) and the harmony learning by eq.(39) become equivalent to the minimum mutual information approach for ICA that was previously discussed after eq.(3). All these cases are featured by maximum information transfer and thus shortly called as the Max-Inform approach.

Generally, the KL learning by eq.(42) and the harmony learning by eq.(39) are different for those $p(\mathbf{y}|\mathbf{x})$ that do not satisfy $E_p = c$. This difference can also be observed from the learning results in those cases that the KL learning by eq.(42) results in only $p(\mathbf{x}, \mathbf{y}) = q(\mathbf{x}, \mathbf{y})$, which has not the least complexity nature, while the harmony learning by eq.(39) results in not only $p(\mathbf{x}, \mathbf{y}) = q(\mathbf{x}, \mathbf{y})$ but also a minimized entropy

$$-\int q(\mathbf{x}, \mathbf{y}) \ln q(\mathbf{x}, \mathbf{y}) d\mathbf{x}d\mathbf{y} \; or \; equivalently \; -\int p(\mathbf{x}, \mathbf{y}) \ln p(\mathbf{x}, \mathbf{y}) d\mathbf{x}d\mathbf{y}, \tag{61}$$

which makes model towards a least complexity.

In a summary, the family of KL learning by eq.(42) and the family of harmony learning by eq.(39) do share an intersection that consists of interesting models. However, two families are different with each containing useful models outside this intersection. The union of the two families consists of a spectrum of learning models, ranging from regularized ML or Max-Inform versions to the original ML or Max-Inform versions, and then reaching regularized versions of harmony learning and finally to the harmony learning. In addition, as discussed previously in Sec.3.3, certain regularized versions of ML or Max-Inform and the harmony learning are also obtainable by the role of z_p and z_q via either data smoothing or normalization [88, 93, 94, 102].

This spectrum can be extended via a convex combination $\lambda KL(\theta) + (1 - \lambda)H(\theta),\ 0 \leq \lambda \leq 1$. Its minimization is equivalent to the KL learning when $\lambda = 1$ and then tends to the harmony learning as λ decreases from 1 to 0. As λ varies from 0 to 1, the harmony learning is regularized towards to the KL learning.

The combination may go beyond the above spectrum, which can be observed by considering a B-architecture with $p(y|x)$ free. It follows from $H(\theta) = -KL(\theta) - E_p$ that

$$\lambda KL(\theta) - (1 - \lambda)H(\theta) = \lambda[KL(\theta) - (\frac{1}{\lambda} - 1)H(\theta)] = \lambda[E_p + \frac{1}{\lambda}H(\theta)]. \tag{62}$$

Ignoring the regularization role of z_p and z_q by setting $z_p = 1, z_q = 1$, we can further get

$$\begin{aligned} E_p + \frac{1}{\lambda}H(\theta) &= \int p(y|x)p(x) \ln \frac{p(y|x)p(x)}{[q(x|y)q(y)]^{\frac{1}{\lambda}}} dxdy \\ &= \int p(y|x)p(x) \ln \frac{p(y|x)}{p_Q(y|x)} dxdy + \int p(x) \ln \frac{p(x)}{\hat{q}(x)} dxdy, \\ p_Q(y|x) &= [q(x|y)q(y)]^{\frac{1}{\lambda}}/\hat{q}(x),\ \hat{q}(x) = \int [q(x|y)q(y)]^{\frac{1}{\lambda}} dy. \end{aligned} \tag{63}$$

which was firstly proposed in [108]. Its minimization with respect to a free $p(y|x)$ will lead to

$$p(y|x) = p_Q(y|x) = \frac{[q(x|y)q(y)]^{\frac{1}{\lambda}}}{\hat{q}(x)},\ E_p + \frac{1}{\lambda}H(\theta) = \int p(x) \ln \frac{p(x)}{\hat{q}(x)} dx, \tag{64}$$

where $p(y|x)$ here is a special case of eq.(55) with $\psi(\xi, \eta) = (\xi\eta)^{\frac{1}{\lambda}}$ that makes the inner representation more concentrated than $p(y|x)$ by eq.(43). Minimizing $\int p(x) \ln \frac{p(x)}{\hat{q}(x)} dx$ is different from both the KL learning by eq.(42) with $p(y|x) = p_Q(y|x)$ and from the ML learning on $\hat{q}(x)$ since $\int \hat{q}(x)dx \neq 1$.

The spectrum can be further extended by considering a linear combination $\lambda KL(\theta) - (1 - \lambda)H(\theta)$ with $\lambda > 1$, which is no longer a convex combination since $1 - \lambda < 0$. However, it is still meaningful by observing $E_p + \frac{1}{\lambda}H(\theta)$, and eq.(64) still applies. The difference is that $1/\lambda < 1$ makes the inner representation more spreading than that of the ML learning, with the regularization strength increasing as λ increases. However, as λ becomes too large, a too strong regularization will make the system finally loose the ability of adapting input data.

3.5. Ying-Yang Alternative Procedure for Parameter Learning

As discussed in (Xu, 1995), learning on the Yang machine $p(x, y)$ and the Ying machine $q(x, y)$ can be implemented alternatively by

Ying-step: fixing $p(x, y)$, update unknowns in $q(x, y)$,

$$\text{Yang-step: fixing } q(x,y), \text{update unknowns in } p(x,y), \tag{65}$$

such that $-H(\theta)$ by eq.(39) or $KL(\theta)$ by eq.(42) decreases gradually until becomes converged. The details are discussed as follows.

Ying-step With $p(y|x)$ fixed and the regularization term z_q ignored, both eq.(39) and eq.(42) share a same updating format as follows:

$$\max_{\theta_{x|y}} L(\theta_{x|y}),\ \max_{\theta_y} L(\theta_y),\ L(\theta_{x|y}) = \int p(y|x)p(x)\ln q(x|y)dxdy,\ \ L(\theta_y) = \int p(y|x)p(x)\ln q(y)dxdy, \tag{66}$$

where $\theta_{x|y}, \theta_y$ consist of unknown parameters in $q(x|y), q(y)$, respectively. We make the updates $\delta\theta_{x|y}, \delta\theta_y$ such that $L(\theta_y + \delta\theta_y), L(\theta_{x|y} + \delta\theta_{x|y})$ either reach a local maximum or increase for certain extent. Typically, an updating $\delta\theta$ that increases an index $J(\theta)$ is a stepsize along a gradient based direction $g_\theta = \nabla_\theta J(\theta)$, i.e.,

$$\delta\theta = \eta T g_\theta, \tag{67}$$

where $\eta > 0$ is a small positive number that defines a stepsize. This $\delta\theta$ is along either the gradient direction when $T = I$ is a unit matrix or a direction that has a positive projection on the gradient direction when T is a positive definite matrix. Particularly, $\delta\theta$ is the well known natural gradient direction when T is the inverse of the metric tensor of $J(\theta)$.

When $p(x)$ is given by eq.(31), the integral over x in both eq.(32) and eq.(42) will disappear. When $p(x)$ is given by eq.(30), this integral over x can also be removed in help of the approximation by Tab.2(C) [92, 89].

Similar to the previous cases around eq.(50), the integrals over y is either analytically solved when $q(x|y)$ and $q(y)$ are both Gaussian or becomes a computable summation when y takes one of discrete values $1, \cdots, k$ or is a binary vector $y = [y^{(1)}, \cdots, y^{(m)}]$. Otherwise, these integrals are difficult to compute. Still, even when it becomes a computable summation for a binary vector $y = [y^{(1)}, \cdots, y^{(m)}]$, the computing cost will increase exponentially with m.

The problem is tackled via letting the integral over the entire domain of y to be approximated by a summation on a set Y_t of a finite number of samples of y, which are obtained according to $p(y|x)$ in the Yang step. One typical case is that Y_t consists of only one sample $y_t = y(x_t)$. In this case, we can further make

$$\text{updating } \delta\theta_{x|y}, \delta\theta_y \text{ in the form of eq.(67) with } \ln q(x_t|y_t) \text{ and } \ln q(y_t) \text{ in the place of } J(\theta), \text{ respectively.} \tag{68}$$

Considering eq.(53) with the z_q-regularization, we have that both the harmony learning by eq.(32) and the KL learning by eq.(42) share a same format in the following gradients:

$$\begin{aligned} &g_{\theta_{x|y}} = \tfrac{1}{N}\textstyle\sum_{t=1}^N \int \eta_t(y)\nabla_{\theta_{x|y}} \ln q(x_t|y)dy + 0.5{h_x}^2 \nabla_{\theta_{x|y}}\pi_q,\ g_{\theta_y} = \tfrac{1}{N}\textstyle\sum_{t=1}^N \int \eta_t(y)\nabla_{\theta_y}\ln q(y)dy, \\ &\eta_t(y) = \begin{cases} p(y|x_t) \text{ by eq.(43)}, & \text{for the KL learning by eq.(42)}, \\ \text{as in eq.(53)}, & \text{for the harmony learning by eq.(32)}, \end{cases} \end{aligned} \tag{69}$$

which can be further put in eq.(67) for updating.

Yang-step It could be implemented in one of four typical choices, according to not only whether eq.(39) or eq.(42) is used for learning but also whether a B-architecture or a BI-architecture is in consideration. The details are given as follows:

(1) In implementing the KL learning on a B-architecture with eq.(42) for parameter learning or equivalently the ML learning on $q(x)$, the Yang-step is to get the posteriori estimation by eq.(43). This is exactly the E-step by the well known EM algorithm [24, 65], with the Ying-step being a generalized M-step. In other words, the EM algorithm is a specific case of the Ying-Yang alternative procedure by eq.(65). An integral over y has to be encountered for getting $q(x)$, which is either analytically solvable when $p(y|x)$, $p(x|y)$ and $p(y)$ are all Gaussian densities or becomes a computable summation when y takes one of discrete values $1, \cdots, k$ or is a binary vector $y = [y^{(1)}, \cdots, y^{(m)}]$. Moreover, even when it is computable for a binary vector $y = [y^{(1)}, \cdots, y^{(m)}]$, the computing cost will increase exponentially with m. In other cases, the integral is difficult to compute. It was as previously suggested in 1997 to be implemented approximately via a computational expensive Monte Carlo simulation. That is, Y_t is obtained via randomly picking a set of samples of y according to $p(y|x_t)$, (see the choice (1) of Tab.2(C) in [109] and the choice (a) in Eqn.(24) and Sec. 3.1 in [99]) or even in a rough approximation according to $q(y)$ (see Step 1 in Tab.II(B) in [109]).

(2) In implementing the KL learning on a BI-architecture with eq.(42) for parameter learning, the Yang-step is to update $\theta_{y|x}$ that consists of all the parameters in $p(y|x)$ by either eq.(46) or eq.(45). The update $\delta\theta_{y|x}$ is made

such that $KL(\theta_{y|x} + \delta\theta_{y|x})$ either reaches a local minimum or reduces for a certain extent. Here, the integral over y for getting $q(x)$ by eq.(43) is avoided. However, we have to encounter not only the integrals in eq.(66) but also the following one

$$H(\theta_{y|x}) = \int p(y|x)p(x)\ln p(y|x)dxdy. \tag{70}$$

It was also firstly suggested in 1997 under the name of the mean field approximation (see the choice (3) of Tab.2(C) in [109] and the choice (c) in Eqn.(24) of [99]) that we get Y_t consisting of only one mean point

$$y_t = \int yp(y|x_t)dy \tag{71}$$

which is computable for a $p(y|x)$ given by either one of eq.(45), eq.(46), eq.(47), and eq.(48), but not applicable to a B-architetcure with $p(y|x)$ given by eq.(43) since another intergral has to be encountered to get $q(x)$.

(3) In implementing the harmony learning with eq.(39) for parameter learning, the Yang-step is simply getting Y_t that consists of only one peak point

$$y_t = arm\max_y p(y|x_t). \tag{72}$$

which was firstly suggested again in 1997 (see the choice (2) of Tab.2(C) in [109] and the choice (b) in Eqn.(24) of [99]) and then further encountered in eq.(44) on a B-architecture. This nonlinear optimization is implemented in help of an iterative procedure:

$$y^{new}(x_t) = ITER(y^{old}(x_t)). \tag{73}$$

Specific algorithms of this type are proposed in [92, 89] to suit typical structures of $q(x|y)$ and $q(y)$. E.g., we consider a Ying machine in the following typical structure:

$$q(x|y) = G(x|Ay + \mu, \Sigma),\ q(y) \text{ in eq.(3).} \tag{74}$$

Specifically, when $y^{(j)}$ is a real random number that comes from a Gaussian mixture:

$$q(y^{(j)}) = \sum_i \beta_{ji} G(y^{(j)}|m_{ji}, \lambda_{ji}^2), \sum_i \beta_{ji} = 1, 0 \le \beta_{ji} \le 1, \tag{75}$$

we can solve the following nonlinear optimization

$$y_t = arg\max_y [G(x_t|Ay + \mu, \Sigma) \prod_j \sum_i \beta_{ji} G(y^{(j)}|m_{ji}, \lambda_{ji}^2)], \tag{76}$$

by an iterative algorithm called the fixed posterior approximation [92], as summarized in Tab.2(A).

When $y^{(j)}$ is a binary random number that comes from a Bernoulli distribution:

$$q(y^{(j)}) = q_j^{y^{(j)}}(1 - q_j)^{1-y^{(j)}}, \tag{77}$$

the counterpart of eq.(76) becomes:

$$y_t = arg\max_y [G(x_t|Ay + \mu, \Sigma) \prod_j q_j^{y^{(j)}}(1 - q_j)^{1-y^{(j)}}]. \tag{78}$$

The complexity of making a nonlinear optimization is considerably less than that of making the integrals over y. Moreover, we usually need only a few iterations by eq.(73) instead of waiting it to converge. This is another salient advantage that the least complexity nature of eq.(44) provides us, in addition to making model selection possible.

(4) In implementing the harmony learning on a BI-architecture, the Yang-step consists of two parts. One is simply implementing eq.(72). For a $p(y|x)$ given by eq.(46), we can get $y_t = y(x_t)$ by eq.(48) via a simple comparison that reduces significantly the computational complexity for making a nonlinear optimization by eq.(44). For a $p(y|x)$ given by eq.(45), we can get either

$$y^{(j)} = \pi_j(x_t),\ or\ \ y^{(j)} = \begin{cases} 1, & \pi_j(x_t) \ge 0.5, \\ 0, & \text{otherwise.} \end{cases} \tag{79}$$

The second part attempts to increase $\ln[q(\mathbf{x}|\mathbf{y})q(\mathbf{y})]_{\mathbf{y}=f_{j^*(x)}(\mathbf{x}|\theta_{j^*(x)})}$ or $\ln[q(\mathbf{x}|\mathbf{y})q(\mathbf{y})]_{\mathbf{y}=\pi(x_t,\theta)}$. In help of the chain rule for gradient, it can be implemented via an update $\delta\theta_{x|y,j^*(x)}$ in the form of eq.(67) as follows

$$\begin{aligned} &\delta\theta_{j^*(x)} = \eta_t(y) g_{\theta_{j^*(x)}}^T[\psi(y) + \phi(y)],\ \psi(y) = \frac{\partial \ln q(x|y)}{\partial y},\ \phi(y) = \frac{\partial \ln p(y)}{\partial y}, \\ &g_{\theta_{j^*(x)}} = \nabla_{\theta_{j^*(x)}} f_{j^*(x)}(x|\theta_{j^*(x)}),\ or\ \ g_\theta = \nabla_\theta \pi(x|\theta). \end{aligned} \tag{80}$$

3.6. Regularization vs Model Selection: (III) From $\ln(r)$ to Convex Function

The function $\ln(r)$ is a typical example from the following convex function family

$$F_c = \{f(r) : \frac{d^2 f(r)}{d^2} < 0,\ f(r) \text{ monotonically increases with } r.\} \tag{81}$$

It is interesting to investigate what will happen when $\ln(r)$ is replaced with any $f(r) \in F_c$.

Let us return back to consider such an extension from eq.(34). That is,

$$H_f(p\|q) = \sum_{t=1}^{N} p_t f(q_t), \tag{82}$$

It can be observed that the maximization of this $H_f(p\|q)$ with respect to p_t still have the same least complexity nature as eq.(37) while the match nature by eq.(36) is modified into

$$q_t = f'(\frac{1}{p_t}) / \sum_{t=1}^{N} f'(\frac{1}{p_t}),\ f'(r) = df(r)/dr, \tag{83}$$

which returns to eq.(36) when $f'(r) = 1/r$. We can further classify other $f(r) \in F_c$ according to whether its $f'(r)$ decreases with r in a rate slower than $1/r$. If yes, it is said to be super-ln; otherwise it is said to be sub-ln.

A typical family of super-ln is the so called α-function [104, 110, 113, 114] as follows:

$$f(r) = r^{\alpha}, 0 \le \alpha \le 1,\ and\ thus\ f'(r) = \alpha / r^{1-\alpha}. \tag{84}$$

In this case, eq.(83) can be rewritten as

$$q_t = p_t^{1-\alpha} / \sum_{t=1}^{N} p_t^{1-\alpha}, \tag{85}$$

which returns to eq.(36) with $q_t = p_t$ when $\alpha = 0$ and becomes $q_t = 1/N$ when $\alpha = 1$. In other cases, p_t attempts to take q_t with lower probabilities increased but higher probabilities decreased in a certain degree that is controlled by α. That is, p_t becomes more spreading than q_t as α increases from 0 to 1. It becomes uniform when $\alpha = 1$. In other words, a super-ln function leads to a conscience de-learning regularization.

An example family of sub-ln is the following inversed and negated α-function as follows:

$$f(r) = -r^{\alpha}, \alpha < 0,\ and\ thus\ f'(r) = |\alpha| / r^{1+|\alpha|}. \tag{86}$$

which leads to

$$q_t = p_t^{1+|\alpha|} / \sum_{t=1}^{N} p_t^{1+|\alpha|}. \tag{87}$$

Now, p_t adopts q_t with higher probabilities increased but lower probabilities further decreased in a certain degree. That is, p_t becomes more concentrated than q_t as $|\alpha|$ increases. In other words, a sub-ln function leads to a competition effectively similar to the least complexity nature.

The above discussions also apply to the continuous case by eq.(33), which becomes:

$$H(p\|q) = \int p(u) f(q(u)/z_q) du. \tag{88}$$

Now, not only the term z_q provides a z-regularization as previously discussed, but also z_q and a super-ln $f(r)$ jointly introduce the above discussed regularization.

Even without z_q (e.g., let $z_q = 1$), a super-ln $f(r)$ will also perform a regularization role. With $p(u)$ given by eq.(31), eq.(88) becomes a f-likelihood function $L_f = \frac{1}{N} \sum_{t=1}^{N} f(q(u_t))$ which differs from the log-likelihood $L = \frac{1}{N} \sum_{t=1}^{N} \ln q(u_t)$ in that $f(q(u_t))$ takes the position of $\ln q(u_t)$. Those unlikely samples with small $q(u_t)$ (e.g., outliers) locate within a drastic varying range of $\ln q(u_t)$ and thus contribute a big portion to affect L. That is, the ML learning is vulnerable to the disturbance by outliers. In contrast, for a super-ln $f(r)$ such as given by eq.(84), the varying range of $f(q(u_t))$ with a small $q(u_t)$ is much smaller than that of $\ln q(u_t)$. Thus, L_f will be much less affected by those outliers with a small $q(u_t)$. In other words, we get a type of robust ML learning [110, 113] .

The Kullback divergence can also be extended. There are two ways. One is simply replace $\ln r$ with $f(r)$, resulting in

$$KL_f(p\|q) = \int p(u) f(\frac{p(u)/z_p}{q(u)/z_q}) du. \tag{89}$$

The other is considering its equivalent form of maximizing $H_{KL}(p\|q) = \int p(u) \ln \frac{q(u)/z_q}{p(u)/z_p} du$ and then replace $\ln r$ with $f(r)$, which results in the maximization of

$$H_{KL_f}(p\|q) = \int p(u) f(\frac{q(u)/z_q}{p(u)/z_p}) du. \tag{90}$$

Being different from eq.(89), it will return to exactly eq.(88) when $p(u)$ is given by empirical density eq.(31).

In the special case that $z_p = z_q$, we have

$$KL_f(p\|q) = \int p(u) f(\frac{p(u)}{q(u)}) du, \; H_{KL_f}(p\|q) = \int p(u) f(\frac{q(u)}{p(u)}) du. \tag{91}$$

Detailed studies can be further made by considering certain specific features of $f(r)$. E.g., for those $f(r) \in F_c$ that satisfy

$$(a)\; f(ab) = f(a)f(b), \quad (b)\; f(a^{-1}) = f^{-1}(a), \tag{92}$$

e.g., for the case of eq.(84) we can rewrite eq.(88) into

$$H(p\|q) = f^{-1}(z_q) \int p(u) f(q(u)) du, \tag{93}$$

and eq.(90) into

$$H_{KL_f}(p\|q) = \frac{f(z_p)}{f(z_q)} \int p(u) \frac{f(q(u))}{f(p(u))} du. \tag{94}$$

All the above results can be applied to a BYY system, what needs to do is simply put eq.(29) into eq.(88), eq.(89), eq.(90), eq.(93), and eq.(94). First, maximizing $H(p\|q)$ by eq.(88) with respect to a free $p(y|x)$ will still lead to eq.(44). Also, both minimizing $KL_f(p\|q)$ by eq.(89) and maximizing $H_{KL_f}(p\|q)$ by eq.(90) with respect to a free $p(y|x)$ will lead to eq.(43) too. Second, we are also lead to what discussed in Sec. 3.3 on trading off the strength of regularization and the ability of model selection in help of designing a parametric model for $p(y|x)$. Third, a new perspective we get here is that such a trading off may also be achieved via choosing $f(r)$ to be super-ln or sub-ln, in the matching process of a Ying machine $q(x|y)q(y)$ to a Yang machine $p(x|y)p(y)$ with $p(y) = \int p(y|x)p(x)dx$ and $p(x|y) = p(y|x)p(x)/p(y)$.

Learning can still be implemented by the Ying-Yang alternative procedure in Sec.3.5. For the Ying step, we no longer have the separated integral format as in eq.(66). However, we can still get the gradient format similar to those in eq.(69). The only difference is that $\eta_t(y)$ is replaced by $\eta_t^f(y)\eta_t(y)$ with

$$\eta_t^f(y) = \begin{cases} f'(\frac{q(x_t|y)q(y)}{z_q}) \frac{q(x_t|y)q(y)}{z_q}, & \text{for the harmony learning by eq.(88),} \\ f'(\frac{p(u)/z_p}{q(u)/z_q}) \frac{p(u)/z_p}{q(u)/z_q}, & \text{for the } KL_f \text{ learning by eq.(89),} \\ f'(\frac{q(u)/z_q}{p(u)/z_p}) \frac{q(u)/z_q}{p(u)/z_p}, & \text{for the } KL_f \text{ learning by eq.(90),} \end{cases} \tag{95}$$

where $f'(r) = df(r)/dr$.

For the Yang step, the discussions on a B-architecture in Sec. 3.5 remains the same. While for a BI-architecture, $\eta_t(y)$ in eq.(80) is also replaced by $\eta_t^f(y)\eta_t(y)$ with $\eta_t^f(y)$ given by eq.(95).

The rest of this paper will concentrate on a special case of BYY learning with $q(y)$ constrained to satisfy eq.(3). As shown in Fig.2, this special case is called BYY independence learning, in a sense that each x is interpreted either as generated via $q(x|y)$ from the inner independent factors, or being mapped via $p(y|x)$ to match the independent factors. This BYY independence learning will provide a unified view on solving ICA and extensions.

4. $L(W)$ Based ICA Studies

4.1. Marginal Densities: Prefixed vs Learned

We return to eq.(8) to continue our overview on ICA studies. Idealistically, if we know the exact density function form of $p(y^{(j)})$ and let $q(y^{(j)}) = p(y^{(j)})$, a solution W that satisfies eq.(3) will make $L(W)$ reach its global maximum. This point can be clearly observed from its equivalence to minimizing the mutual information

$$KL(W) = \int p(y) \ln \frac{p(y)}{\prod_{j=1}^m q(y^{(j)})} dy \geq 0 \tag{96}$$

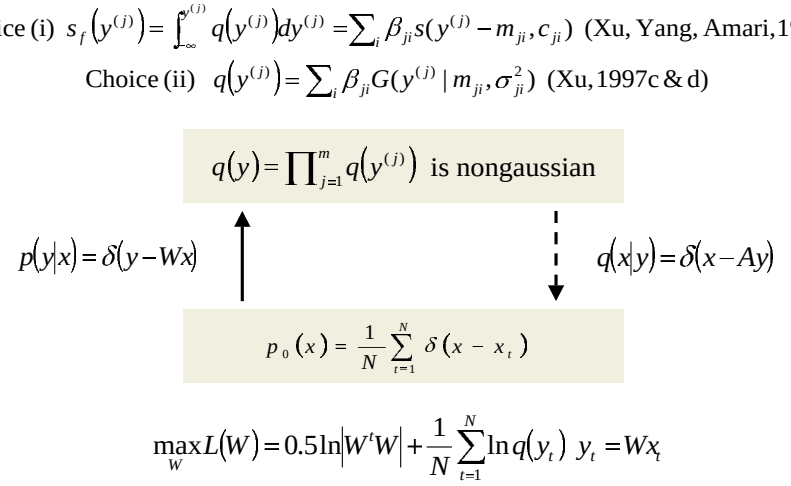


Figure 4. *LPM-ICA as A Special Case of BYY Learning*

which reaches 0 when $q(y^{(j)})$ is the true density of $y^{(j)}$. Due to this, early efforts have been made towards accurately estimating $p(y^{(j)})$. One typical example is using a nonparametric estimator [63]. However, at each updating on W this nonparametric estimate should be made on all the samples, which is computationally very expensive.

Alternatively, the truncated Gram-Charlier expansion is suggested by [22] and the truncated Edgeworth expansion is suggested by [6]. Even simply, in the work of [10], their use of sigmoid neurons is equivalently to estimating $p(y^{(j)})$ by the derivative of a simple sigmoid function. In these approaches, $q(y^{(j)})$ is prefixed during learning W, for which a typical algorithm is the well known natural gradient [6]:

$$W^{new} = W^{old} + \eta[I + \phi(y_t)(W^{old}x_t)^T]W^{old}, \ \phi(y) \text{ is given by eq.(80).} \tag{97}$$

Here $q(y^{(j)})$ is prefixed, all these approaches work only on the cases that the components of y_t are either all sub-Gaussians (e.g., for the way by [6]) or either all super-Gaussians (e.g., for the way by [10]).

In implementing histogram equalization, a mixture of sigmoid neurons is used to estimate the cumulated distribution function (CDF) more flexibly [41]. Such a type of CDF mixture is further suggested by [115] to model $q(y^{(j)})$. That is, the CDF of $q(y^{(j)})$ is represented by the following mixture

$$s_f(y^{(j)}) = \sum_i \beta_{ji} s(y^{(j)} - m_{ji}, c_{ji}), \ \sum_i \beta_{ji} = 1, \tag{98}$$

with $0 \leq \beta_{ji} \leq 1$ and $s(r, c)$ being a simple sigmoid scalar function, e.g., $s(r, c) = 1/(1 + e^{-cr})$. During learning on W, $q(y^{(j)}) = ds_f(y^{(j)})/dy^{(j)}$ is also learned via maximizing $\ln q(y^{(j)})$ with the parameters $\{\beta_{ji}, m_{ji}, c_{ji}\}$ adapted, such that whether a source $p(y^{(j)})$ is super-Gaussian or sub-Gaussian can be automatically detected and adapted by a model $q(y^{(j)})$ during learning. As a result, we get a so called Learned Parametric Mixture based ICA (shortly LMP-ICA) algorithm that is shown experimentally to work well on any combination of super-Gaussian or sub-Gaussian components of y_t. Moreover, an EM algorithm has been proposed for learning $q(y^{(j)})$ together with learning on W, with $q(y^{(j)})$ modeled via either a mixture of sigmoid based CDFs [111, 112, 106] or the finite mixture (especially Gaussian mixture) by eq.(75) in [109, 110].

The idea of modeling $q(y^{(j)})$ via a mixture of CDFs in help of sigmoid functions is also suggested by [62], but they did not clearly target at working on any combination of super-Gaussian or sub-Gaussian components of y_t. Latter, both Gaussian mixture and the EM algorithm were revisited for implementing ICA via the noiseless case of implementing noisy ICA in [8].

As shown in Fig.4, the above discussed studies can be also viewed from a special case of BYY learning by eq.(42) with a special BI-architecture:

$$p(y|x) = \delta(y - Wx), \ q(x|y) = \delta(x - Ay), \tag{99}$$

and with the empirical density $p(x) = p_0(x)$ given by eq.(30) at $h_x = 0$. It has been shown in [116] that eq.(8) can be revisited from eq.(42) with eq.(99) substituted in. With $q(y^{(j)})$ modeled via the CDF by eq.(98), the above

mentioned LPM-ICA is revisited, actually which was motivated from such a perspective [115]. Also shown in Fig.4 and to be further discussed in Sec.4.3, we can get an extension of eq.(8) with $\ln|W^TW|$ in place of $\ln|W|$ and thus becomes workable to the cases that n is less than the dimension of x, which is recently called under-complete ICA [45, 66]. Furthermore, with $q(y^{(j)})$ modeled via a finite mixture eq.(75), it follows from the Ying step by eq.(66) that θ_y can be updated by an EM algorithm as in [109, 110]. Also, the Yang step by eq.(80) leads to eq.(97).

In addition, from BYY harmony learning by eq.(32) on this special BI-architecture eq.(99), we have been also lead to two variants of ICA [89]. Especially, by simply setting $z_q = 1$, we can be lead to

$$\max_W L,\ L = \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{m}\ln q(y_t^{(j)})|_{y_t=Wx_t},\ s.t.\ \frac{1}{N}\sum_{t=1}^{N}Wx_tx_t^TW^T = I. \tag{100}$$

If x further goes through a pre-whitening, we will be lead to eq.(10) and its subsequent algorithm again.

4.2. Loosely Matching, 1-Bit-Conjecture, and Open Issues

Another interesting point is that the work by [10] also demonstrates that eq.(8) works even when modeling $q(y^{(j)})$ roughly via the derivative of a single sigmoid function such as $\tanh(x)$. This result echoes the successes of the early efforts of nonlinear Hebbian learning eq.(6) on separating sources, where it was also a simple sigmoid non-linearity in action [29, 35]. Thus, it remaindered the community that a pursuit on accurately estimating $p(y^{(j)})$ may not be really necessary [116] and certain simple non-linearity is already enough to yield a solution that satisfies eq.(3). Moreover, a generalized sigmoid function has been also suggested in [10] such that the sigmoid function may flexibly change to loosely match sources by altering two parameters during learning.

Actually, before and after the work [10], the phenomenon that a loosely matching of sources may lead a correct ICA performance in certain experiments have been observed by many researchers. Seeking the nature of this loose matching, it has been found experimentally that a model $q(y^{(j)})$ of super-Gaussian usually works on a source of super-Gaussian while a model $q(y^{(j)})$ of sub-Gaussian usually works on a source of sub-Gaussian [112], which indirectly reconfirms the success of some previous studies that separate blind sources via a contrast function that bases on kurtosis [77, 32, 17]. This observation is further summarized in [107] into the following so called one-bit-matching conjecture

"*all the sources can be separated as long as there is an one-to-one same-sign-correspondence between the kurtosis signs of m sources and the kurtosis signs of $q(y^{(j)})$, $j = 1, \cdots, m$. That is, there is at least one way to pair all the sources and $\{q(y^{(j)})\}$ such that for each pair the corresponding source and $q(y^{(j)})$ have a positive product in their kurtosis.*"

Unfortunately, there still lack theoretical results that prove or disprove this conjecture. In a special case that has only two sources of sub-Gaussians, it has been mathematically proved in [112, 20] that a correct ICA performance can be guaranteed with only a simple non-linearity $\phi(r) = r^3$ as in eq.(6) through showing that all the correct solutions of W are stable local maximums of $L(W)$ while no other local maximums exist. Unfortunately, a result of this type is not available yet for the cases of more than two sources. Alternatively, theoretical analysis has provided by [5] on the conditions for a non-linearity $\phi(r)$ such that all the correct solutions of W are stable local maximums of $L(W)$. However, under which conditions such a learning can converge to one correct solution still remains as an open challenge.

Instead, such an one-bit-conjecture has been experimentally demonstrated repeatedly. For examples, the experiments in [112, 106] have shown that modeling $q(y^{(j)})$ by eq.(98) with only two sigmoid functions can already result in a good ICA performance. Moreover, eq.(8) is implemented in [27] with a success by modeling $q(y^{(j)})$ via directly switching between a sub-Gaussian one and a super-Gaussian one through detecting the kurtosis of $y = Wx$ during learning. Furthermore, such a switching technique is also implemented in [43] by modeling $q(y^{(j)})$ via a mixture of two component densities such that $q(y^{(j)})$ switches between sub-Gaussian and super-Gaussian by only altering one parameter from one value to the other. This switching-based technique is actually closely related to the above LMP-ICA by considerably simplifying the mixture density into simply switching between two prefixed densities.

Here, we report a further result. We approximate $p(y^{(j)})$ by its truncated Gram-Charlier expansion:

$$p(y^{(j)}) \approx G(y^{(j)})[1 + \frac{1}{2}(m_2 - 1)H_2(y^{(j)}) + \frac{m_3}{6}H_3(y^{(j)}) + \frac{m_4 - 6m_2 + 3}{24}H_4(y^{(j)})], \tag{101}$$

where $G(y^{(j)})$ denotes the standard Gaussian density, m_n the nth-order moment of $p(y^{(j)})$, and $H_n(y^{(j)})$ the nth-order Chebyshev-Hermite polynomials. Also, we can approximate the corresponding parametric model $q(y^{(j)})$ similarly. Then we can further mathematically prove that the one-bit-conjecture as follows:

Theorem [44] Assume that the 3rd order moment of every parametric model $q(y^{(j)})$ is zero and ignore all the moments that are higher than the 4th order, the global minimum W^* of $KL(W)$ by eq.(96) will make $y = W^*x$ become independent among components as long as the sign of kurtosis of every $q(y^{(j)})$ is same as the sign of kurtosis of every corresponding source $y^{(j)}$.

This result is one step forward on the one-bit conjecture. It says that only considering the 4th order moments, the condition that makes ICA workable is the satisfaction of the one bit conjecture. However, not only it is unclear whether the algorithm by eq.(97) will lead to a solution W that make $y = Wx$ become independent among components since it may converge to a local minimum of $KL(W)$ by eq.(96), but also it remains unclear whether ICA surely does not work without the satisfaction of the one bit conjecture. Several problems remain to be explored. The first is what condition should be satisfied by $\phi(r)$ such that all the local maximums of $L(W)$ correspond to the correct solutions of W that makes eq.(3) satisfied. The second is how to control the learning process to enter those correct local maximums in the cases there are some local maximums of $L(W)$ that correspond to the correct solutions of W while others correspond to the wrong solutions. The third is how to get an effective algorithm that is able to search the global maximum. Also, how the 3rd order moments can be used to help the problem solving.

According to the above theorem, we can model each $q(y^{(j)})$ by

$$\begin{array}{l} q(y^{(j)}) = \beta q_1(y^{(j)}) + (1-\beta)q_2(y^{(j)}), \ \ where \\ q_1(y^{(j)}) \text{ is subGaussian and } \int y^{(j)} q_1(y^{(j)})dy^{(j)} = 0, \ \int (y^{(j)})^3 q_1(y^{(j)})dy^{(j)} = 0, \\ q_2(y^{(j)}) \text{ is superGaussian and } \int y^{(j)} q_2(y^{(j)})dy^{(j)} = 0, \ \int (y^{(j)})^3 q_2(y^{(j)})dy^{(j)} = 0. \end{array} \tag{102}$$

We have many choices to pre-specify such $q_1(y^{(j)}), q_2(y^{(j)})$ that have no unknown parameters. It follows directly from eq.(102) that $\int y^{(j)} q(y^{(j)})dy^{(j)} = 0, \ \int (y^{(j)})^3 q_(y^{(j)})dy^{(j)} = 0$. Thus, in the above theorem, the remaining condition to be satisfied is to match the kurtosis signs. By varying the parameter β from 0 to 1, the kurtosis of $q(y^{(j)})$ will vary from negative to positive smoothly. Thus, during the learning on W by eq.(97), we can learn an appropriate value of β to adapt the kurtosis sign of the source $y^{(j)}$ via maximizing $\ln q(y^{(j)})$. That is, we have the following new variant of the LPM-ICA algorithm by eq.(97) with a much simplified updating on $\phi(y)$ together:

$$\begin{array}{l} \beta^{new} = (1-\eta_0)\beta^{old} + \eta_0 p_t^{(j)}, \ p_t^{(j)} = \dfrac{\beta^{old} q_1(y^{(j)})}{\beta^{old} q_1(y^{(j)}) + (1-\beta^{old}) q_2(y^{(j)})}, \\ \phi(y) = \beta\phi_1(y^{(j)}) + (1-\beta)\phi_2(y^{(j)}), \ \phi_1(y) = \dfrac{\partial \ln q_1(y)}{\partial y}, \ \phi_2(y) = \dfrac{\partial \ln q_2(y)}{\partial y}, \end{array} \tag{103}$$

where $\eta_0 > 0$ are a small learning step size.

4.3. Natural Gradient, Non-invertible Matrix, and Other Extensions

Another issue is how efficiently a learning process converges to a solution. A well known result on this issue is the natural gradient algorithm by [6] that provides a considerable improvement on convergence and computing efficiency over the classic gradient algorithm. Still, more studies can be made along this direction. E.g., it is interesting to relate the natural gradient algorithm to the Newton algorithm, quasi-Newton algorithms, and various super-linear gradient based algorithms in the literature of nonlinear optimization. Also, it is useful to compare a number of existing typical ICA algorithms by mathematically analyzing their convergence rates.

Eq.(8) works for an ICA problem eq.(1) in the case that A is invertible. However, all the results obtained in this case can be extended to the cases that A is non-invertible but full rank, i.e., $m < d$. In [110], under the name of BYY learning based ICA model, another special case of eq.(42) with the following special BI-architecture

$$p(y|x) = G(y|Wx, \sigma^2 WW^T), q(x|y) = G(x|Ay, \sigma^2 I), \tag{104}$$

has been studied. Moreover, it is further shown in [110] that its special case with $\sigma^2 = 0$ leads to

$$L(W) = 0.5 \ln |WW^T| + \frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{m} \ln q(y_t^{(j)})|_{y_t = Wx_t}, \tag{105}$$

which replaces eq.(8) in the case of $m < d$. This result can also obtained by directly considering eq.(99). Observing that $\delta(x - Ay) = \delta(A(y - A^+x)) = \delta(y - A^+x)/|A^TA|^{0.5}$ and $A^+ = (A^TA)^{-1}A$ is a pseudo inverse of A, we insert eq.(99) into eq.(42) and get

$$KL(\theta) = \int p(x) \ln p(x) dx + 0.5 \ln |A^TA| - \int \delta(y - Wx) p(x) \ln q(y) dx dy,$$

$$KL_\delta(W,A) = \int \delta(y - Wx)p(x)\ln\frac{\delta(y-Wx)}{\delta(y-A^+x)}dxdy. \quad (106)$$

Moreover, its minimization will further push $KL_\delta(W,A) = 0$ at $A^+ = (A^TA)^{-1}A = W$ from which we see that $(A^TA)^{-1} = WW^T$. Also, the 1st term can be ignored since it is irrelevant to the parameters to be learned. With $p(x) = p_0(x)$ by eq.(31), the minimization of the last two terms become equivalent to eq.(105), which has also be recently studied under the name of under-complete ICA [45, 66].

It is interesting to observe that what needs to do is simply replacing $\ln|W|$ in eq.(8) with $0.5\ln|WW^T|$. Both become equivalent when $d = m$. Interestingly, the natural gradient algorithm on eq.(105) is found [107] to still keep its original form as in [6]. Moreover, eq.(8) and eq.(105) have been further extended with $\ln(r)$ replaced by a generalized convex function $f(r)$ as discussed in Sec.3.6,and the resulted ICA algorithm has been shown experimentally to be more robust to outliers [110].

Another ICA extension is to consider a probabilistic mapping $p(y|x)$ in place of the deterministic mapping $\delta(y - Wx)$. One effort was made in [117] under the name of Maximum Balanced Mapping Certainty (Max-BMC) principle that maximizes $J_c - J_b$ with

$$J_c = \int p(x)p(y|x)f(p(y|x))dydx,\ J_b = \int p(y)f(p(y))dy,\ p(y) = \int p(x)p(y|x)dx, \quad (107)$$

where $f(r)$ is strictly monotonic increasing on $[0,1]$ and two typical examples are $f(r) = r$ and $f(r) = \ln r$. E.g., for $f(r) = \ln r$ we have

$$J_c - J_b = \int p(x)p(y|x)\ln\frac{p(y|x)}{p(y)}dydx = \int p(x)p(y|x)\ln\frac{p(y|x)p(x)}{p(y)p(x)}dydx, \quad (108)$$

which represents the mutual information that is maximized with the information transferred to y maximally preserved.

If $p(y|x)$ is free, the above maximization will push $p(y|x)$ to become $p(y|x) = \delta(y - g(x))$ and J_c becomes an infinite large number that is irrelevant to $g(x)$. Thus, the problem becomes equivalent to maximizing $J_b = -\int p(y)\ln p(y)dy$ that is the information transferred via $y = g(x)$. Particularly, when $g(x) = s(Wx)$, it is equivalent to the INFOMAX approach [10]. The Max-BMC principle in [117] extends this INFOMAX based ICA in three aspects. First, the deterministic mapping $p(y|x) = \delta(y - g(x))$ is extended to a probabilistic mapping when $p(y|x)$ is given by eq.(43). Second, the function $\ln r$ is extended to a more general function $f(r)$. Third, it provides an information transfer based learning for unsupervised classification with a minimum mis-classification. Moreover, it can also be used for supervised learning. E.g., the principle by eq.(108) in the special case with $f(r) = \ln(r)$ has been also used on training the hidden units of three layer networks and self-organizing map [37].

Efforts have also been made on extending the ICA problem eq.(1) to the nonlinear cases. E.g., it has been extended by [82] to the post-nonlinear system. Also, a nonlinear blind source separation has been attempted by self-organizing map [60], Moreover, from a special case of BYY learning by eq.(42) with $p(y|x) = \delta(y - f(x,\theta))$, we are lead to the following general cost for implementing nonlinear ICA [99, 89]:

$$L = \frac{1}{N}\sum_{t=1}^N[0.5\ln|W(x_t)W^T(x_t)| + \sum_{j=1}^m \ln q(y_t^{(j)})],\ y_t = f(x_t,\theta),\ W(x) = \frac{\partial f(x,\theta)}{\partial x^T}. \quad (109)$$

Except certain specific nonlinear extension such as in [82], extensions of ICA to a nonlinear case usually no longer retain the nature that it recovers the waveform of each $\{y_t^{(j)}\}$ series subject to unknown scales. In fact, the waveform of each $\{y_t^{(j)}\}$ may undergo an arbitrary nonlinear distortion. Thus, generally speaking, nonlinear ICA is not well defined.

Instead, a nonlinear mapping $y = f(x,\theta)$ to a uniform distribution is well defined when the dimensions of x and y are same. In this case, an estimate $q(x)$ on the density of x is given as follows [91].

$$q(x) = |W(x)W^T(x)|^{0.5},\ W(x) = \frac{\partial y}{\partial x^T},\ y = f(x,\theta). \quad (110)$$

One early example is given in [41] for the case $d = m = 1$, with $y = f(x,\theta) = s_f(Wx)$ and $s_f(r)$ given by a mixture of CDFs as in eq.(98). Another early example is given in [70, 50] for a multi-dimensional density, with $y = f(x,\theta)$ given roughly by a simple sigmoid function. Both examples can be regarded as a special case of the BYY learning by eq.(42) with $p(y|x) = \delta(y - s(Wx))$, where each $s(r)$ is either a simple sigmoid function as in

[70] or a mixture of CDFs as in eq.(98). Denoting $\hat{y} = Wx$ and considering that $s_j(r)$ is monotonic with r, we have

$$|\frac{\partial y}{\partial x^T}| = |\frac{\partial y}{\partial \hat{y}^T}||\frac{\partial \hat{y}}{\partial x^T}| = |W| \prod_j q(\hat{y}^{(j)}), \tag{111}$$

which is actually equivalent to the case of eq.(8) with $q(y^{(j)}) = \frac{ds_j(r)}{dr}|_{r=y^{(j)}}$.

Moreover, the above discussions can be extended to a noisy mapping $y = f(x,\theta) + \varepsilon$ and thus $W(x) = \frac{\partial f(x,\theta)}{\partial x^T} + \frac{\partial \varepsilon}{\partial x^T}$. When ε is independent of x, we have $\frac{\varepsilon}{\partial x^T} = 0$ and thus $W(x)$ becomes equivalent to the noiseless case. However, ε will affect the learning in a sense of regularization.

Extensions can also be made to the cases that the dimension d of x is larger than the dimension m of y simply via an augmented vector $y^{'T} = [y^T, z^T]$ with z being a vector of the dimension $d-m$. Given $y = f(x,\theta)$ and $z = g(x)$, if y' has a uniform distribution on $[0,1]^d$, it is similar to eq.(110) that

$$\begin{aligned} q(x) &= |U(x)U^T(x)|^{0.5} = \left| \begin{array}{cc} W(x)W^T(x) & V(x)W^T(x) \\ W(x)V^T(x) & V(x)V^T(x) \end{array} \right|^{0.5}, \\ U(x) &= \frac{\partial y'}{\partial x^T},\ W(x) = \frac{\partial f(x,\theta)}{\partial x^T},\ V(x) = \frac{\partial g(x)}{\partial x^T}, \end{aligned} \tag{112}$$

which can be further simplified into

$$q(x) = |W(x)W^T(x)|^{0.5}|V(x)V^T(x)|^{0.5}, \tag{113}$$

by choosing $g(x)$ such that $V(x)W^T(x) = 0$, i.e., the subspace spanned by $V(x)$ is orthogonal to the subspace spanned by $W(x)$.

The use of eq.(113) can be made in two ways. One is that the ML learning on $q(x)$ can be estimated via its maximization with respect to $y = f(x,\theta)$ and $z = g(x)$. The other is that $V(x)$ is chosen such that $V(x)W^T(x) = 0$ and $V(x)V^T(x)$ becomes irrelevant to $W(x)$. It follows from $\ln q(x) = 0.5\ln|W(x)W^T(x)| + 0.5\ln|V(x)V^T(x)|$ that the maximization on $\ln q(x)$ with respect to $W(x)$ or $y = f(x,\theta)$ is equivalent to the maximization on $0.5\ln|W(x)W^T(x)|$ with respect to $W(x)$ or $y = f(x,\theta)$ such that the mapping $y = f(x,\theta)$ makes y become a uniform distribution on $[0,1]^m$ that are mutually independent among its components. Particularly, it becomes equivalent to eq.(105) for $y = f(x,\theta) = s_f(Wx)$ with $s_f(r)$ by eq.(98).

5. Extensions of ICA to Noisy Environment

5.1. Non-gaussian Factor Analysis, Algebraic Equation, and ML Learning

We further extend the ICA problem eq.(1) to noisy environments by considering an independent noise being added to the observation. We focus on considering eq.(4) subject to satisfying eq.(3) in the following two cases:

$$x = Ay + e,\ A = \left\{ \begin{array}{ll} \text{a general } d \times m \text{ matrix}, & \text{(a)} \\ \text{an orthogonal matrix, i.e., } A^TA = I, & \text{(b).} \end{array} \right. \tag{114}$$

subject to the satisfying eq.(3), where e usually consists of i.i.d. samples that is independent of y.

We start at the classic factor analysis (FA) model with eq.(5). It has been widely applied to various data analyses. Unfortunately, the FA suffers the so called rotation indeterminacy. Considering a rotation $y' = \phi y, \phi\phi^T = I$, we have $A' = A\phi^T$ and $Ey'y'^T = \phi E[yy^T]\phi^T = I$, that is

$$x = A'y' + e \text{ in a same form of eq.(114)},\ y' \sim G(y'|0,I),\ e \sim G(e|0,\Sigma). \tag{115}$$

As a result, y is not able to be recovered up to unknown scales but with a rotation indeterminacy.

This rotation indeterminacy can be solved when y comes from a non-Gaussian density (e.g., a Gaussian mixture as shown in Fig.5). In this case, we call eq.(114) non-Gaussian Factor Analysis (NFA) [92, 89], which is also previously called independent factor model (IFM) and BKYY dependence reduction (BKYY-DR) [99], as well as independent factor analysis (IFA) [8]. Here we prefer the name NFA for addressing its difference from the classic FA that is also a type of IFM or IFA in a 2nd order sense. NFA can be further classified according to specific types of $y^{(j)}$ in eq.(3). E.g., we get Binary or Bernoulli NFA (shortly BFA) when $y^{(j)}$ is a binary value with the Bernoulli distribution by eq.(77), and get a real NFA (shortly NFA) when $y^{(j)}$ is a real value with Gaussian mixture by eq.(75).

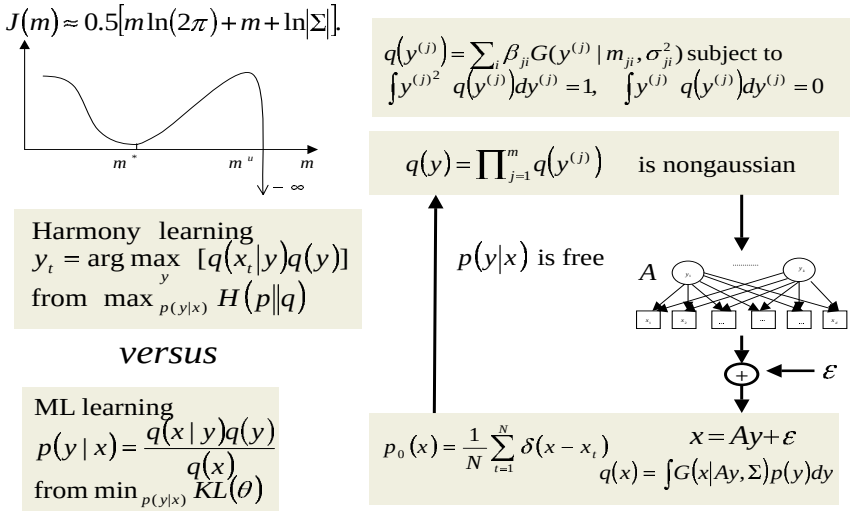


Figure 5. *ML equivalent learning versus harmony learning on a BYY B-architecture for NFA*

Usually, we consider the cases that e comes from Gaussian. More generally, we can also consider e from a nonGaussian density. Furthermore, eq.(114) may also be further extended into the following nonlinear model:

$$x = g(y, A) + e, \tag{116}$$

where e is still a white noise that is independent from y.

Examining the solving approaches in Sec.2, we can find that approaches of nonlinear Hebbian learning type, MMI type, and INFOMAX type are not applicable to the cases of eq.(114) or eq.(116), since they are all based on a given forward function that does not take the noise e in consideration.

However, the following two approaches are still applicable:

(1) *Solving algebraic equations* From eq.(114) or eq.(116), in help of eq.(3) we can also get a set of algebraic equations in term the matrix A, the statistics of e, and higher-order-statistics of y, x. Using those techniques developed in the literature of nonlinear algebraic equations, at least conceptually we may solve all the unknowns. However, there are not many studies made along this line.

(2) *The ML approach* Both eq.(114) and eq.(116) are special cases of a generative density $q(x|y)$. With $q(y)$ satisfying eq.(3), we have $q(x)$ by eq.(27) as shown in Fig.5. Thus, the problems of estimating $q(x|y)$ and $p(y)$ can be made via the ML learning on $q(x)$ on a set of samples of x, which is also equivalent to making the parameter learning by eq.(42) on a BYY backward architecture via eq.(43), by which the optimal forward mapping is made to match $q(y)$ that satisfies eq.(3).

For making BFA, the integral in eq.(27) becomes summation and eq.(43) can be made via enumeration. The well known EM algorithm has been suggested to effectively implement the ML learning [11, 28, 99]. Also, learning can be alternatively made via transforming the problems into a clustering problem that is solved by an existing clustering algorithm [61]. Using RPCL learning for making this clustering [21], the dimension m of y can be determined too.

However, for making NFA, the EM algorithm is not directly applicable since the integral in eq.(27) can not be solved analytically. A straight forward way is solving integrals via an approximation method such as a Monte Carlo random sampling approach [99]. However, the computation is usually very involved.

5.2. BYY Harmony Learning Based NonGaussian FA

We have a quite different scenario when the BYY harmony learning by eq.(32) is used for making BFA and NFA on eq.(114). As shown in Fig.5, the difficulty on making the integral in eq.(27) is avoided by getting $y_t = y(x_t)$ per a sample x_t in help of either eq.(48) on a BI-architecture or eq.(44) on a B-architecture via a nonlinear optimization by an iterative procedure eq.(73).

The learning is implemented by the Ying-Yang alternative procedure eq.(65). For simplicity, we do not consider data smoothing regularization in the rest of this paper by simply letting $h_x = 0$ in eq.(53). Readers are referred to [87, 88, 89, 92] for details about implementing data smoothing regularization.

It follows from eq.(53) and eq.(66) that the Ying step is equivalent to maximize $H(p||q)$ that becomes [92, 89]

$$\nabla_\theta H(\theta) = \frac{1}{N}\sum_{t=1}^N \eta_t^{x|y}\nabla_{\theta_{x|y}} \ln G(x_t|Ay_t+\mu,\Sigma) + \frac{1}{N}\sum_{t=1}^N \eta_t^y \nabla_{\theta_y} L_t^y,$$
$$L_t^y = \ln q(y_t) = \sum_{j=1}^m \ln q(y_t^{(j)}),\ z_q = \begin{cases} 1, & \text{(a) its role ignored,} \\ \sum_{t=1}^N q(x_t|y_t)q(y_t), & \text{(b) by eq.(49),} \\ \sum_{t=1}^N q(x_t|y_t), & \text{(c) by eq.(50),} \end{cases}$$
$$\eta_t^{x|y} = \begin{cases} 1, & z_q \text{ of case (a),} \\ 1-\frac{q(x_t|y_t)q(y_t)}{z_q}, & z_q \text{ of case (b),} \\ 1-\frac{q(x_t|y_t)}{z_q}, & z_q \text{ of case (c),} \end{cases}$$
for the case that Y_t contains only one sample y_t, we have
$$\eta_t^y = \begin{cases} 1, & z_q \text{ of case (a) \& case (b),} \\ 1-\frac{q(x_t|y_t)q(y_t)}{z_q}, & z_q \text{ of case (c),} \end{cases} \tag{117}$$

where $\theta_{x|y}$ consists of A, μ, Σ and θ_y consists of parameters of $q(y)$. With y_t obtained from the Yang step, the Ying step has the following detailed form:

The Ying Step

$updating\ q(x|y)$
$$e_t = x_t - A^{old}y_t - \mu^{old},\ \mu^{new} = \mu^{old} + \eta_0\eta_t e_t,\ \text{get } \Sigma^{new} \text{ as in Tab.2(B),}$$
$$A^{new} = \begin{cases} A^{old} + \eta_0\eta_t e_t y_t^T, & \text{(a) a general } A, \\ \text{by either of two approaches in Tab.1 via } g_A, & \text{(b) an orthogonal } A, \end{cases}$$
$$where\ g_A = \nabla_A \ln G(x|Ay+\mu,\Sigma) = y_t e_t^T \Sigma^{old\ -1}. \tag{118}$$

$updating\ q(y)$
For BFA with eq.(77), get q_j^{new} as in Tab.2(B),
For NFA with eq.(75), we update $\beta_{j,r}, m_{j,i}, \lambda_{j,i}^2$ by eq.(119) given below,

$discarding\ y^{(j)}$
$$get\ \lambda_j^2 = \begin{cases} q_j^{new}(1-q_j^{new}), & \text{for BFA with eq.(77),} \\ \sum_{r=1}^n \beta_{j,r}^{new}\lambda_{j,r}^{2\ new}, & \text{for NFA with eq.(75),} \end{cases}$$
if λ_j^2 tends zero constantly, discard $y^{(j)}$ and delete the j-th colum of A,

where $\eta_0 > 0$ is a small stepsize. The first two parts above was were previously presented in Tab.2(B) of [109] and Eqns.(35)&(35) in [99]) at the special cases without the term z_q.

Specifically, for NFA with eq.(75) and either $z_q = 1$ or z_q given by eq.(50), we update

$$p_{j,r} = \frac{\beta_{j,r}G(y^{(j)}|m_{j,r},\lambda_{j,r}^2)}{\sum_i \beta_{ji}G(y^{(j)}|m_{j,i},\lambda_{j,i}^2)},\ \beta_{j,r}^{new} = (1-\eta_0)\beta_{j,r}^{new} + \eta_0 p_{j,r},$$
$$\hat{m}_{j,r}^{new} = m_{j,r}^{old} + \eta_0 p_{j,r}(y^{(j)} - m_{j,r}^{old}),\ \hat{\lambda}_{j,r}^{2\ new} = (1-\eta_0 p_{j,r})\lambda_{j,r}^{2\ old} + \eta_0 p_{j,r}(y^{(j)} - m_{j,r}^{old})^2. \tag{119}$$

While for NFA with eq.(75) and z_q given by eq.(49), we update $\beta_{j,r}^{new}$ and $\hat{\lambda}_{j,r}^{2\ new}$ in the way as in Tab.2(B).

The Yang step is different for a B-architecture and a BI-architecture. As firstly suggested again in 1997 (see the choice (2) of Tab.2(C) in [109] and the choice (b) in Eqn.(24) of [99]), we can get y_t by eq.(72) or particulary by eq.(44) on a B-architecture and by eqs.(48)&(79) on a BI-architecture. Specifically, the Yang step has the following details:

The Yang Step
$$y_t = \begin{cases} \text{by eq.(44), e.g., run Tab.2(A) in a few iterations,} & \text{(a) for B-architecture,} \\ \text{by eqs.(48)\&(79),} & \text{(b) for BI-architecture;} \end{cases}$$
$$p(y|x)\ is\ updated\ via \begin{cases} \text{no action ,} & \text{(a) for B-architecture,} \\ \text{updating } \theta_{j^*(x)} \text{ or } \theta \text{ by eq.(80),} & \text{(b) for BI-architecture.} \end{cases} \tag{120}$$

For a BI-architecture, instead of searching a best forward nonlinear mapping $y(x)$ via an iterative procedure eq.(73), $y(x)$ is approximated by a forward parametric structure in the form of eq.(47) with each function form of $f_j(x|\theta_j)$ pre-specified. Thus, it is better to get some knowledge on what is the optimal $y(x)$.

It follows from eq.(44) that the optimal $y(x)$ should satisfy $\Phi(y(x)) = 0$ with

$$\Phi(y) = \phi(y) + 0.5A^T\Sigma^{-1}(x-\mu-Ay),\ \phi(y) = \nabla_y \ln q(y). \tag{121}$$

Given a specific form of $q(y)$, we can approximately solve $\Phi(y(x)) = 0$ to estimate $y(x)$. When $y(x)$ is an exact root of $\Phi(y(x)) = 0$, it returns to eq.(44) and thus becomes equivalent to the case of a B-architecture.

In the case that $\Sigma = \sigma^2 I$, we have $\Phi(y) = \phi(y) + 0.5\sigma^{-2}[A^T(x-\mu) - A^T Ay]$ and thus the optimal solution is

$$y(x) = G^{-1}(\frac{0.5}{\sigma^2}A^T(x-\mu)),\ G(y) = \frac{0.5}{\sigma^2}A^T Ay - \phi(y) \tag{122}$$

that takes the non-linearity $\phi(y)$ in consideration. When σ^2 is large enough, the 2nd term of $G(y)$ dominates and thus $G^{-1}(.)$ is given by the inverse function of $\phi(y)$. Thus, it is intuitively justified that we can select the following post-linear function

$$y(x) = s(Wx+\nu), s(y) = [s_1(y^{(1)}), \cdots, s_m(y^{(m)})]^T, \tag{123}$$

where $s_j(r)$ is a scalar nonlinear function. Particularly, we have $s_j(r) = r$ when there is no noise; while we have $s_j(r) = \phi_j^{-1}(r)$ and $\phi(y^{(j)}) = \frac{d \ln p(y^{(j)})}{dy^{(j)}}$ when σ^2 is large.

When there is no observation noise on x, i.e., we have $\Sigma = \sigma^2 I$ and $\sigma^2 = 0$, it follows from either eq.(121) or from Step (b) in Tab.2(A) that

$$y(x) = (A^T A)^{-1} A^T (x-\mu). \tag{124}$$

When $Ex = \mu = 0$, we get the linear mapping $y_t = Wx_t$ and $W = (A^T A)^{-1}A^T$ is simply the pseudo inverse of A. The BI-architecture is degenerated into eq.(99). That is, we revisit the LPM-ICA as shown in Fig.4.

5.3. Model Selection vs Automatic Model Selection

With the above Ying-Yang alternative learning, both the two types of model selection on Fig.3 apply but act differently for BFA, FA, and NFA.

(1) Model Selection for BFA During learning by eq.(118), maximizing $\ln q(y^{(j)})$ will push its variance λ_j^2 constantly towards 0, when the j-th dimension is extra. In this case, we can discard $y^{(j)}$ and delete the j-th column of A. In other words, we get the automatic model selection as shown in Fig.3(b).

Alternatively, we may also implement BFA in a two stage style. At the first stage, parameter learning can be made either by the ML learning [11, 28, 99] or by the above Ying-Yang alternative learning with every q_j simply fixed at 0.5. Then, at the second stage we further select a best number m of factors by eq.(40) with its detailed form given in the last row of Tab.3. The type of criteria at the special case of $h_x = 0$ was firstly proposed in 1997 (See Eqn.(8.10) and Eqn.(8.13) in [108]).

(2) Model Selection for NFA On eq.(114) with a real value vector y and a general A, however, we suffer a type of scaling indeterminacy. That is, any scaling transform $y' = \Lambda y$ with a diagonal matrix Λ will result in a density $p(y')$ that still satisfies eq.(3). To solve the problem, the constraint $Ey^{(j)} = 0, E(y^{(j)})^2 = 1$ is imposed on eq.(74) such that NFA is actually based on

$$\begin{aligned} &q(x|y) = G(x|Ay+\mu, \Sigma),\ q(y|\theta_y) = \prod_{j=1}^{m} q(y^{(j)}|\theta_y), \\ &subject\ to \int y^{(j)} q(y^{(j)}|\theta_y) dy^{(j)} = 0,\ \int (y^{(j)})^2 q(y^{(j)}|\theta_y) dy^{(j)} = 1. \end{aligned} \tag{125}$$

In implementation, learning by eq.(119) should be followed by a normalization

$$m_j = \textstyle\sum_{r=1}^{n} \beta_{j,r}^{new} \hat{m}_{j,r}^{new},\ m_{j,r}^{new} = (\hat{m}_{j,r}^{new} - m_j)/\lambda_j,\ \lambda_{j,r}^{2\ new} = \hat{\lambda}_{j,r}^{2\ new}/\lambda_j^2. \tag{126}$$

for ensuring $Ey^{(j)} = 0, E(y^{(j)})^2 = 1$. Moreover, model selection has to be made as shown in Fig.3(a) with a best m selected by eq.(40) that has a detailed form given in the 3rd row of Tab.3. The type of criteria at the special case of $h_x = 0$ was firstly proposed also in 1997 (See Eqn.(10.9) in [108]).

Since $\Lambda A^T A \Lambda = \Lambda^2$ no longer satisfies $A^T A = I$, this scaling indeterminacy can also be removed by imposing a constraint that A is orthogonal, with no need of eq.(126) to ensure $Ey^{(j)} = 0, E(y^{(j)})^2 = 1$. Moreover, an automatic model selection will also apply in help of the last step in eq.(118).

(3) Model Selection for FA In the special case by eq.(5), we are lead back to the classic factor analysis [7, 48], with both eq.(119) and eq.(126) becoming unnecessary. The Yang step by eq.(120) becomes simply

$$y_t = [I + A^T \Sigma^{-1} A]^{-1} A^T \Sigma^{-1}(x_t - \mu), \tag{127}$$

Tab. 3 Model Selection Criteria $Min_m J(m)$ on $x = Ay + e$

$J(m) = 0.5\ln\|\Sigma\| + 0.5h_x^2 Tr[\Sigma^{-1}] + J_y(m)$	• $h_x = 0$, $z_q = 1$ for empirical learning. • $0.5\ln\|\Sigma\| = 0.5d\ln\sigma^2$ when $\Sigma = \sigma^2 I$.	
Gaussian Factors (FA+TFA)	$J_y(m) = \begin{cases} 0.5m[1+\ln(2\pi)], & \text{(a) for a general } A \text{ \& after harmony learning,} \\ 0.5m[1+\ln(2\pi)+\ln\|\Lambda\|], & \text{(b) for an orthogonal } A \text{ \& after ML learning.} \end{cases}$ $\ln\|\Lambda\| = \ln\|A^TA\| = \sum_{j=1}^m \ln\lambda_j^2$	
Non-Gaussian Factors (NFA+ TNFA)	$J_y(m) = \begin{cases} -\frac{1}{N}\sum_{t=1}^N\sum_{j=1}^m \ln q(y_t^{(j)}\|\Omega), & \text{(a) for a general } A \text{ \& after harmony learning,} \\ 0.5\ln\|\Lambda\| - \frac{1}{N}\sum_{t=1}^N\sum_{j=1}^m \ln q(V\Lambda^{-0.5}y_t^{(j)}\|\Omega), & \text{(b) for an orthogonal } A \text{ \& after ML learning.} \end{cases}$ *subject to* $Ey_t^{(j)} = 0, E(y_t^{(j)})^2 = 1.$	
Binary Factors (BFA+ TBFA+ LMSER+ TLMSER)	**without time** $J_y(m) = \begin{cases} m\ln 2, & \text{(a) } q_j = 0.5 \text{ for all } j, \\ \sum_{j=1}^m E_j, & \text{(b) for } q_j \text{ learned via ML learning,} \end{cases}$ $E_j = -[q_j\ln q_j + (1-q_j)\ln(1-q_j)].$	**with time** $J_y(m) = \sum_{k=1}^m [\pi^{(k)}E_1^{(k)} + (1-\pi^{(k)})E_0^{(k)}]$ $E_1^{(k)} = -\begin{cases} \pi_{01}^{(k)}\ln\pi_{01}^{(k)} + \pi_{11}^{(k)}\ln\pi_{11}^{(k)}, & (a) \\ \ln\pi^{(k)}, & (b) \end{cases}$ $E_0^{(k)} = -\begin{cases} \pi_{00}^{(k)}\ln\pi_{00}^{(k)} + \pi_{10}^{(k)}\ln\pi_{10}^{(k)}, & (a) \\ \ln(1-\pi^{(k)}), & (b) \end{cases}$

and the Ying step is given by the part of updating $q(x|y)$ in eq.(118). Actually, this Ying-Yang procedure provides an adaptive algorithm that equivalently implements the ML learning. Then, model selection on m can be made by eq.(40) with its detailed form given in the 2nd row of Tab.3 [92, 89]. The type of criteria at the special case of $h_x = 0$ was firstly proposed in 1997 too (see Eqn.(9.13) in [108], Eqn.(56) in [100], as well as Eqn.(13) and Eqn.(18b) in [105]).

Another interesting special case is

$$q(x|y) = G(x|Ay + \mu, \Sigma),\ A^TA = I,\ q(y|\theta_y) = G(y|0, \Lambda),\ \Lambda = diag[\lambda_1^2, \cdots, \lambda_m^2], \tag{128}$$

we get a density $q(x) = G(x|\mu, A\Lambda A^T + \Sigma)$ that is the same as given by eq.(114) and eq.(5) with a general A. In the ML learning sense, both the two cases lead back to the classic factor analysis [7, 48], and can be implemented by the EM algorithm [72] or adaptive EM algorithm [100, 101]. Still, model selection on m can be made by eq.(40) with its detailed form given in the 2nd row of Tab.3.

However, the two cases are different in the sense of BYY harmony learning by eq.(32). Instead of eq.(127), the Yang step becomes

$$y_t = [\Lambda^{-1} + A^T\Sigma^{-1}A]^{-1}A^T\Sigma^{-1}(x_t - \mu). \tag{129}$$

Though the Ying step includes the part of updating $q(x|y)$ in eq.(118) with A updated via g_A by either of two approaches in Tab.1, what is different is that Λ is updated via maximizing $\eta_t \ln G(y|0, \Lambda)$ in the way as in Tab.2(B), during which λ_j^2 will be pushed towards zero if the dimension $y^{(j)}$ is extra. In other words, model selection is made automatically during parameter learning by eq.(39).

(4) Model Selection for Uncorrelated NFA For NFA on eq.(114) with a general A, it is also possible to have automatic model selection via turning eq.(125) into

$$q(x|y) = G(x|Ay + \mu, \Sigma),\ q_u(y) = |\Lambda|^{-0.5}q(\xi|\theta_y),\ \xi = V\Lambda^{-0.5}y,\ A^TA = I,\ V^TV = I. \tag{130}$$

in help of singular value decomposition of a general matrix. That is, $x = (A\Lambda^{0.5}V^T)\xi + e = Ay + e$ with $y = \Lambda^{0.5}V^T\xi$. Again, eq.(125) and eq.(130) are equivalent in the sense of the ML learning on $q(x)$. Moreover, the case of eq.(130) still falls in the paradigm of the conventional factor analysis [48] and thus we call it uncorrelated NFA, since the covariance matrix of $q_u(y)$ is still the diagonal matrix Λ. Being different from the conventional factor analysis, the matrix D and V are also learned such that an invertible mapping $\xi = VD^{-1}y$ further transfer the uncorrelated factors of y further into the independent factors of ξ such that the rotation indeterminacy is removed. Strictly speaking, it goes beyond the BYY independence learning since $q_u(y)$ in eq.(130) may no longer satisfy eq.(3).

However, it can be still learned in the sense of the BYY harmony learning by eq.(32). Because the mapping $\xi = VD^{-1}y$ is invertible, the joint density $q(x|y)q(y)$ given by eq.(125) differs from that by eq.(130) only in $|\Lambda|$, which has no effect on getting y_t by eq.(44). So, the Yang step for a B-architecture is still the same as in eq.(120). While for a BI-architecture, the Yang step is still in the same format as in eq.(120) but the difference is that $q(x|y), q(y)$ by eq.(130) is used for $\psi(y), \phi(y)$ in eq.(80). Moreover, the Ying step consists of the part of updating $q(x|y)$ in eq.(118) with A updated by either of two approaches in Tab.1 in help of g_A and the part of updating $q(y)$ in eq.(118) with y_t replaced by ξ_t, plus the following updating

$$\begin{aligned}
&\xi_t = VD^{-1}y_t,\ D = \Lambda^{0.5},\ \phi(\xi) = \frac{\partial \ln q_0(\xi)}{\partial \xi},\ g_V = \nabla_V \ln q(\xi_t|\theta_y) = \phi(\xi_t) y_t^T D^{-1}, \\
&\text{update } V^{old} \text{ into } V^{new} \text{ by either of two approaches in Tab.1,} \\
&from\ D^{-1} g_D D^{-1} = D^{-1} \frac{\partial[\ln q_0(VD^{-1}y_t|\theta_y) - \ln|D|]}{\partial D} D^{-1} = -diag[V^T \phi(\xi_t) y_t^T] - \ln|D|, \\
&update\ D^{new} = D^{old} - \eta_0 (diag[V^T \phi(\xi_t) y_t^T] + D^{old}), \\
&\text{If } \lambda_j^2 \text{ approaches 0 constantly, discard } y_t^{(j)}, \text{ delete the } j\text{-th column of } A \text{ and the } j\text{-th element of } \Lambda.
\end{aligned} \tag{131}$$

During learning, maximizing $\ln q_u(y_t) = -0.5 \ln|\Lambda| + \ln q(\xi_t|\theta_y)$ will push the variance λ_j^2 towards zero if the dimension $y_t^{(j)}$ is extra. That is, model selection on m happens automatically.

Finally, ξ_t acts as the recovered independent factors though y_t is not. That is, $x_t \to \xi_t$ performs an independence mapping that takes in consideration of the noise e_t.

(5) Model Selection for ICA In the special case that there is no observation noise on x, i.e., we have $\Sigma = \sigma^2 I$ with $\sigma^2 = 0$. It follows from eq.(124) that the mapping $x \to y$ becomes linear and we are lead back to the LPM-ICA as shown in Fig.4. Moreover, by Tab.3 we observe that $\sigma^2 = 0$ makes $J(m)$ becomes $-\infty$. We can decide m until

$$\sigma^2 = \frac{1}{dN} \sum_{t=1}^{N} \|x_t - W^T (WW^T)^{-1} W x_t\|^2, \tag{132}$$

becomes zero but is not nonzero at $m-1$. That is, as firstly proposed in Eqn.(55) of [97] or Eqn.(54) in [95], we can decide m by checking this σ^2 in eq.(132). This is actually equivalent to choosing m_u as shown in Fig.3 as well as its subsequent discussions. In a practical problem, there is always certain noise. Thus, choosing m^* in Fig.3 by the criteria given in Tab.3 are much useful.

5.4. Special Cases: LMSER and Auto-association

The previously discussed LMSER learning by eq.(23) actually performs ICA via a sigmoid post-linear mapping $y = s(Wx)$ via a constraint $W = A^T$ that attempts to minimize the mean square error $\|x - Ay\|^2 = \|e\|^2$. Though the role of $y = s(Wx)$ was originally interpreted as extracting features in [123], it has been further experimentally demonstrated that the LMSER learning performs ICA with a performance superior to a nonlinear Hebbian learning that does not consider noise [38]. Also, it has been successfully used on implementing a binary ICA with noise [130].

Relaxing the constraint $A = W^T$, eq.(23) becomes

$$\sigma^2 = \tfrac{1}{dN} \textstyle\sum_{t=1}^{N} \|x_t - As(Wx_t)\|^2, \tag{133}$$

which is referred as auto-association learning via three layer net and can be trained in the same way as training a three layer net by the Back-propagation technique [71]. It was experimentally demonstrated that the sigmoid layer $s(Wx_t)$ makes feature extraction, though it was not noticed that it actually performed an ICA [12].

As shown in Fig.6(a), the above LMSER and auto-association learning can be understood from a special case of the BYY learning with a BI-architecture.

$$q(x|y) = G(x|Ay + \mu, \sigma^2 I),\ p(y|x) = \delta(y - s(Wx + \nu)). \tag{134}$$

Considering $y = s(Wx + \nu)$ with each sigmoid function $0 < s(r) < 1$ that can be either simply $s(r, c) = 1/(1 + e^{-cr})$ or a mixture of CDFs by eq.(98), we expect that x is mapped to match $q(y)$ in a uniform density on $[0, 1]^m$ and that x is interpreted as generated from this uniform distribution via $Ay + \mu$ plus an observation noise e. In this case, $\ln q(y_t)$ becomes zero and maximizing $H(\theta)$ along the direction $\nabla_\theta H(\theta)$ by eq.(117) reduces to being

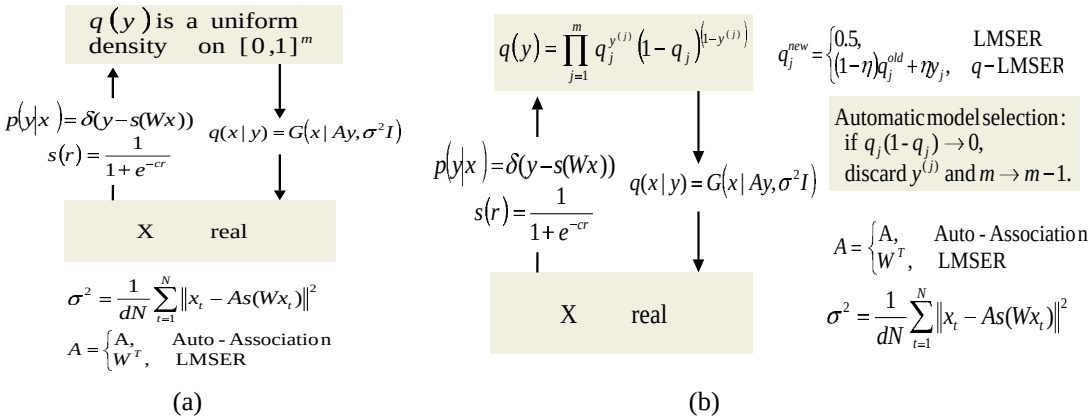


Figure 6. *A BYY BI-architecture that performs LMSER and auto-association learning*

equivalent to minimizing σ^2 in eq.(23). In other words, LMSER and auto-association learning attempt to perform both a specific ICA that maps x into a uniform density and a specific NFA that fits x via independent factors from a uniform density.

Moreover, the heuristic use of $A = W^T$ and $y = s(Wx + \nu)$ with each sigmoid function $0 < s(r) < 1$ can also be justified by $y(x)$ in eq.(122). That is, $y(x) = G^{-1}(\lambda W(x - \mu))$ with a scaling by $\lambda = 0.5/\sigma^2$. Moreover, if $q(y)$ is a uniform density on $[0,1]^m$, $-\phi(y) = -\nabla_y \ln q(y)$ consists of an impulse toward $-\infty$ at $y = 0$ and an impulse toward ∞ at $y = 1$. After superposing a linear function $\lambda A^T Ay$, it follows from eq.(122) that $G^{-1}(r)$ is a function that is linear between $[0,1]$ and takes either 0 on $(\infty, 0)$ or 1 on $(1, \infty)$. Thus, a sigmoid function $0 < s(r) < 1$ acts as a reasonable approximation to such a $G^{-1}(r)$.

As shown in Fig.6(b), the above LMSER and auto-association learning can also be understood from the BI-architecture by eq.(134) but with $q(y)$ by eq.(77). When the sigmoid function $0 < s(r) < 1$ is stiff enough such that $s(r)$ for any r value becomes either near 1 with a probability q_j or 0 with a probability $1 - q_j$. Particularly when $q_j = 0.5$, we have $\ln q(y) = -m \ln 2$, which is a constant that is only relevant to m. Again, maximizing $H(\theta)$ along the direction $\nabla_\theta H(\theta)$ by eq.(117) reduces to being equivalent to minimizing σ^2 in eq.(23).

Beyond just providing new insights on the LMSER and auto-association learning, the perspective of BYY harmony learning in Fig.6 also leads to several new results.

First, it follows from eq.(40) and eq.(117) that we get the criterion $J(m)$ in which the above term $\ln q(y) = -m \ln 2$ is no longer an ignorable constant but a balance force for determining an appropriate m. That is,

$$\min_m J(m),\ J(m) = 0.5d \ln \sigma^2 + m \ln 2, \tag{135}$$

which can be used after learning by either the algorithm in [123] or the following new algorithm by eq.(138) with each $q_j = 0.5$ fixed. Moreover, after a learning by the algorithm in [123], we can further estimate

$$q_j = \tfrac{1}{N} \textstyle\sum_{t=1}^{N} y_t^{(j)},\ y_t = s(W\xi_t + \nu), \tag{136}$$

Then, eq.(135) can be improved as follows

$$\min_m J(m),\ J(m) = 0.5d \ln \sigma^2 - \textstyle\sum_{j=1}^{m} [q_j \ln q_j + (1 - q_j) \ln (1 - q_j)]. \tag{137}$$

The above connection between BYY learning and LMSER learning, as well as the above criteria by eq.(135) and eq.(137) were firstly developed in (See Eqn.(8.10) and Eqn.(8.13) in [108]).

Second, it follows from eq.(65) that we can get a new adaptive algorithm in place of the learning algorithm in [123] for implementing LMSER and auto-association learning. Specifically, its Ying step is still given by eq.(118), and its Yang step consists of simply getting y_t by eq.(79) and updating W, ν as follows

$$\begin{aligned} &W^{new} = W^{old} + \eta D_s[\psi(y) + \phi(y)]x^T,\ \nu^{new} = \nu^{old} + \eta D_s[\psi(y) + \phi(y)], \\ &D_s = diag[d_s(y^{(1)}), \cdots, d_s(y^{(m)})],\ d_s(r) = \frac{ds(r)}{dr}, \\ &\psi(y) = A^T(x - Ay - \mu),\ \phi(y) = [\ln \frac{q_1}{1 - q_1}, \cdots, \ln \frac{q_m}{1 - q_m}]^T. \end{aligned} \tag{138}$$

Particularly, we have simply $\phi(y) = 0$ when $q_j = 0.5$. Early studies on this learning also started in 1997 (See Tab. 2 in [109] and also Eqn.(49) in [99]).

Third, as shown in Fig.6(b), automation selection on m can be made during the above harmony learning via detecting whether $(1 - q_j)q_j$ tends to 0. If so, it means that the dimension $y^{(j)}$ is extra.

Furthermore, extensions can also be made from the following aspects:

(a) With $s(r)$ given by a mixture of CDFs in eq.(98), the parameters $\{\beta_j, c_j, m_j\}$ are also be adapted during learning via $\min \sigma^2$ such that a best sigmoid mapping $y = s(Wx)$ is sought.

(b) We consider that each $s_j(r)$ is no longer limited to be a sigmoid function and each $q(y^{(j)})$ is a parametric model that is adapted via its parameters. Correspondingly, $H(\theta)$ becomes

$$\begin{aligned} &H(\theta) = -0.5d \ln \sigma^2 + \tfrac{1}{N} \textstyle\sum_{t=1}^{N} \sum_{j=1}^{m} \ln q(y^{(j)}(x_t)) - \ln z_q, \\ &y(x) = s(Wx + \nu),\ \sigma^2 = \tfrac{1}{dN} \textstyle\sum_{t=1}^{N} \|x_t - Ay(x_t) - \mu\|^2, \end{aligned} \tag{139}$$

which combines the feature of eq.(23) and the feature of eq.(105) with the role of $0.5 \ln |WW^T|$ replaced by $-0.5d \ln \sigma^2$. This can also be understood as an ICA extension to noise situation under the name of Principal ICA (P-ICA). This type of learning started from 1997 (See Eqn.(10.9), Eqn.(8.10) and Eqn.(8.13) in [108], Eqn.(40) and Eqn.(41) in [101], Eqn.(20b) and Eqn.(20c) in [105]). Particularly, when $s(Wx) = Wx$, $\min \sigma^2$ becomes equivalent to making either PCA with $A = W^T$ or singular value decomposition. In this case, eq.(139) degenerates to Eqn.(10.9) in [108].

(c) With $q(x|y) = G(x|g(y, A), \sigma^2 I)$ and thus

$$\sigma^2 = \tfrac{1}{dN} \textstyle\sum_{t=1}^{N} \|x_t - g(y(x_t), A)\|^2, \tag{140}$$

we get that $H(\theta)$ by eq.(117) becomes Eqn (10.8) in [108].

Before closing this subsection, it deserves to mention that all the above studies can be extended to not only the case that the noise e is nonGaussian by considering $q(x|y)$ in the form of eq.(46) with $g(y, A)$ in the place of all the $f_j(v|\theta_{u|v,j})$, but also the cases of a general $q(x|y)$ in the form of eq.(46). The learning can still be implemented by the Ying-Yang alternative procedure eq.(65), with the Ying and Yang steps modified accordingly.

5.5. Approximate ML, Exact ML, and EM Algorithm

As discussed in Sec.3.3, $p(y|x)$ given by eq.(43) performs a best ICA mapping under the noise e in the sense of implementing the ML learning on $q(x)$ by eq.(27). Its substitution by eq.(44) means a deviation from handling noise in this ML sense. This deviation will be reduced by considering the BYY learning by minimizing the Kullback divergence eq.(42) with a non-δ parametric density for $p(y|x)$, which is equivalent to making a type of variational approximation to the ML learning on eq.(27), as discussed previously in Sec.3.3.

Specifically, with $p(y|x)$ in the form of eq.(46) and a mean field approximation $q(x|y) = q(x|E[y|x])$ (i.e., y is replaced with its conditional mean $E[y|x]$, we can also get rid of the integrals in getting $g_{\theta_{x|y}}$ and g_{θ_y} in eq.(66) and $H(\theta_{y|x})$ in eq.(70), and thus are able to implement learning in the form of eq.(67) [99].

This mean field implementation of variational approximation is useful even when y is discrete. Though eq.(27) and eq.(43) can be implemented without approximation in this case, the computational cost is very high, especially when m is large. A typical example is that $p(y|x)$ by eq.(45) and $q(x|y) = G(x|g(y, A), \sigma^2 I)$. In this case, eq.(42) becomes [108]

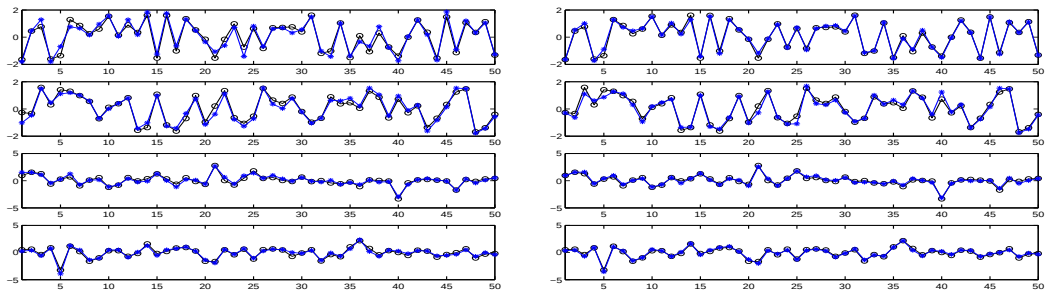
$$\begin{aligned} &L = \ln \sigma^2 + \tfrac{1}{N} \textstyle\sum_{t=1}^{N} \sum_{j=1}^{m} C_p(x_t),\ C_p(x_t) = \pi_j(x) \ln \dfrac{\pi_j(x)}{q_j} + (1 - \pi_j(x)) \ln \dfrac{1 - \pi_j(x)}{1 - q_j}, \\ &\sigma^2 = \tfrac{1}{dN} \textstyle\sum_{t=1}^{N} \|x_t - g(\pi(x_t), A)\|^2,\ \pi(x) = [\pi_1(x), \cdots, \pi_m(x)]^T, \end{aligned} \tag{141}$$

which includes the special case $g(\pi(x_t), A) = A\pi(x_t)$. As a result, a summation over all the 2^m values of y has been avoided. The detailed learning algorithm for implementing eq.(141) is given by Table 2 in [109].

Efforts of implementing the ML learning on eq.(27) have also been made by a number of other research groups. In [33], a so called joint maximum likelihood is considered to be maximized with respect to A and $y(x_t)$:

$$L = \tfrac{1}{N} \textstyle\sum_{t=1}^{N} [\sum_{j=1}^{m} \ln q(y^{(j)}(x_t)) - e_t^T \Sigma^{-1} e_t],\ e_t = x_t - Ay_t, \tag{142}$$

which closely relates to eq.(117) but with two key different points. First, instead of using an iterative procedure eq.(73), a rough solution $y(x)$ is used in [33] via expanding $\nabla_y L = 0$ by the first order approximation. Second, Σ



(a) (b)

Figure 7. The recovered 4 factors (denoted by 'o') and the original sources (denoted by '$*$'), each factor is normalized into zero mean and unit variance. (a) by IFA with the mean square errors being 0.056, 0.057, 0.070, 0.052 per each factor and the average being 0.059, and (b) by NFA with the mean square errors being 0.021, 0.042, 0.040, 0.026 per each factor and the average being 0.032

given in eq.(117) is the natural result of the BYY learning by eq.(32). While Σ in eq.(142) must be pre-given, e.g., $\Sigma = \sigma^2 AA^T$ is used in [33] but with the problem of how to specify the scalar σ^2 remaining as an open problem.

In [52], an approach that exactly implements ML learning for the model eq.(114) with $p(x|y) = G(x|Ay, \Sigma)$ was firstly proposed. Similar to [109], they considered the independence product eq.(3) with each $q(y^{(j)})$ modeled by a Gaussian mixture eq.(75). However, a key difference is that they dealt with the product of Gaussian mixtures via introducing a set of random variable $z^{(j)}, j = 1, \cdots, m$ such that each $z^{(j)}$ stochastically takes a number among $\{1, \cdots, n_j\}$ and each number indicates a component in the j-th mixture. Thus, it follows from $q(y) = \sum_{\{z^{(j)}\}} q(y, z)$ that the product of m summations in eq.(3) is equivalently exchanged into a summation of $\prod_j n_j$ products. As a result, the integral in eq.(27) becomes a summation of the $\prod_j n_j$ analytically computable integrals on Gaussians, which results in that $q(x)$ becomes a mixture of $\prod_j n_j$ Gaussians. For this reason, they were able to implement an exact ML learning on the problem eq.(114) by an exact EM algorithm. The same results have been also published in [8] under the name of independent factor analysis.

However, a summation of $\prod_j n_j$ terms has to be computed at each step of such an EM algorithm. The complexity increases exponentially with the number m of factors, i.e., $O(n^m)$ with $n = \max_j n_j$. In contrast, the implementation of NFA via BYY harmony learning by eq.(118) and eq.(120) turns the integral over y in the whole domain R^m into the problem of a nonlinear optimization by eq.(44). When solved by an iterative algorithm, it searches within the domain R^m in a trace that is usually within a much lower order subspace.

E.g., shown in Fig.7 are experiments on comparing the NFA implemented with the Yang step by eq.(120) and Ying step by eq.(118), and the EM algorithm based exact ML learning algorithm [52] (shortly denoted by IFA in [8]). Two data sets come from a model $x = Ay + e$ of the four factors and six factors, respectively. With each Gaussian mixture by eq.(75) consisting of three Gaussian components, both the NFA and IFA work well. Shown in Fig.7 are the recovered factors of the 4-factor model in comparison with the corresponding original sources. Moreover, it can be observed from both the recovered factors in the figures and the corresponding mean square errors that NFA outperforms IFA.

Moreover, as shown in Fig.8, with the number m of factors increasing from 4 to 6, the time used by NFA increase from $18.16s$ to $28.58s$ (about 57.4%) while that consumed by IFA increases from $64.77s$ to $634.63s$ (about $3^{(6-4)}$ times). That is, we found empirically that while the time complexity of NFA increases linearly with the number m of factors while IFA increases exponentially with m. That is, NFA outperforms IFA significantly in the aspect of complexity.

Also, it has been found in [85] that a major computing load can be removed by making pre-whitening on x such that $E(xx^T) = I$ and it only needs to consider an orthogonal matrix A. As a result, the exponential complexity with m can be reduced to be linear with m.

In [8], a product $p(y|x) = \prod_{j=1}^m p(y^{(j)}|x)$, with each $p(y^{(j)}|x)$ given by a mixture of Gaussians, is used as an alternative for $p(y|x)$ in the E step of the EM algorithm in [52]. The parameters of $p(y|x)$ are updated in help of variational approximation and mean-field approximation, such that the computing cost on estimating an exact posteriori density can be significantly reduced. This effort is similar to the case of the BYY learning eq.(42) with $p(y|x)$ by eq.(46). However, $p(y|x)$ by eq.(46) is not constrained to be in the form of $p(y|x) = \prod_{j=1}^m p(y^{(j)}|x)$. In a contrary, it follows from eq.(43) that with $q(y)$ satisfying eq.(3) the $p(y|x)$ that performs a best ICA mapping is usually not in the form of $p(y|x) = \prod_{j=1}^m p(y^{(j)}|x)$.

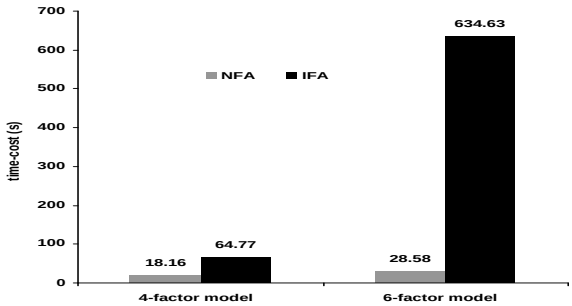


Figure 8. Time complexity comparison between NFA and IFA

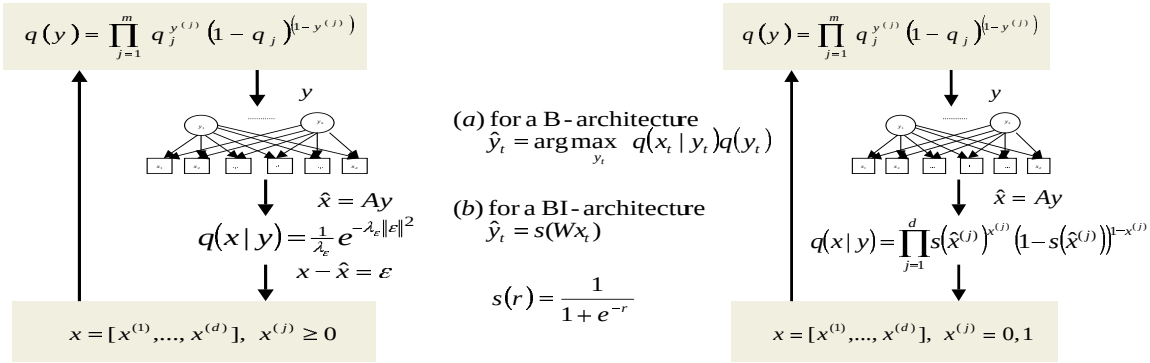


Figure 9. *Independence analysis on binary and non-negative data*

In [8], it is also discussed that the ML learning on a NFA with $q(y) = G(y|0, I)$ and $q(x|y) = G(x|Ay, \sigma^2 I)$ becomes equivalent to performing PCA when $\sigma^2 = 0$. However, requiring $\sigma^2 = 0$ in [8] is completely not necessary, such an equivalence has already been previously proved in [101] for any $\sigma^2 > 0$.

5.6. Non-negative Data Analyses and Supervised Learning on Three Layer Net

In many applications, the observed data $x = [x^{(1)}, \cdots, x^{(d)}]^T$ will take non-negative value $x^{(j)} \geq 0$ or binary value $x^{(j)} = 0$ or 1. As shown in Fig.9, a NFA on such data can be considered via the BYY learning on either a B-architecture with y_t given by the choice (a) or a BI-architecture with y_t given by the choice (b), in help of letting $q(x|y)$ in an appropriate format.

For a case of non-negative value $x^{(j)} \geq 0$, we can consider an exponential density as shown in Fig.9(a). All the previous results on $q(x|y) = G(x|Ay + \mu, \sigma^2 I)$ still apply directly with σ^2 replaced by λ_x that is updated simply by $\lambda_x^{new} = (1-\eta)\lambda_x^{new} + \eta\|\varepsilon\|^2$, when the case of $z_q = 1$ is considered.

For the case of binary value $x^{(j)} = 0$ or 1 that is regarded as generated from an inner binary code y, many studies have been made in the literature. One example that explores along this direction is the early work called Multiple Cause Mixture [74]. It models each bit $\hat{x}^{(j)} = 1 - \prod_i (1 - y_i a_{ij})$ via binary a_{ij} and then matches the observed bit $x^{(j)}$ with a heuristic cost function. Learning becomes a combinatorial optimization problem that searches the values of $a_{i,j}$. Also, the maximum likelihood learning is proposed on this model [23], with each binary code x interpreted as Bernoulli via defining the probability that $x^{(j)} = 1$ in help of a generating model $1 - \prod_i (1 - y_i a_{ij})$.

As shown in Fig.9(b), we can make model selection via modifying $J(m)$ in Fig.7(b) with $\ln \sigma^2$ replaced by $-\frac{1}{N}\sum_{t=1}^{N}\sum_{j=1}^{d}[x_t^{(j)} \ln s(\hat{x}_t^{(j)}) + (1 - x_t^{(j)}) \ln (1 - s(\hat{x}_t^{(j)}))]$. Also, we can get an adaptive algorithm from eq.(65), which is given in Tab.2 of [87]. Again, automation selection on m can be made during learning via detecting whether $(1 - q_j)q_j$ tends to 0 constantly.

As shown in Fig.10, the least square learning on a conventional three layer forward networks can also be revisited from the auto-association learning in Fig.6(b). Decomposing $x = [\xi, \zeta]$ into two parts, we consider a

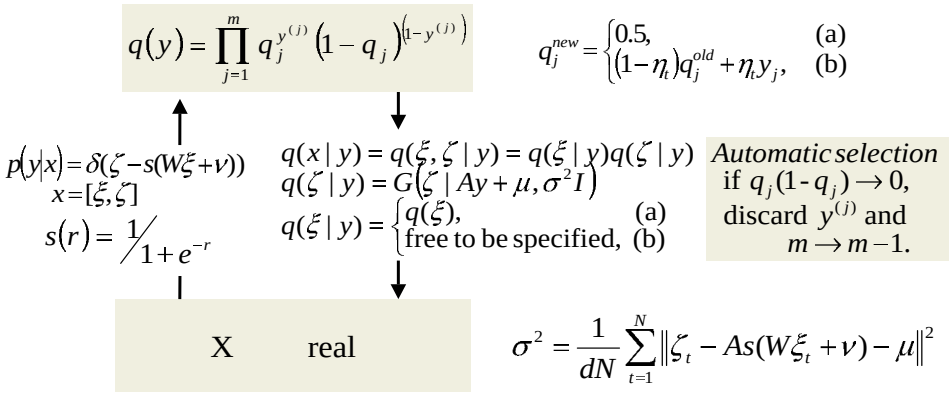


Figure 10. *BYY supervised learning on three layer net versus NFA on a BYY B-architecture*

special BI-architecture with $p(y|x)$, $q(y)$, and $q(x|y)$ given as in Fig.6(b), where the use of $q(\xi|y)$ comes from $q(\xi|\zeta,y)$ under the cascade mapping $\xi\to y\to\zeta$. In this case, $H(\theta)$ by eq.(139) becomes

$$\begin{aligned}
&H(\theta)=-0.5d_\zeta\ln\sigma^2+H_q+L(W),\ \sigma^2=\tfrac{1}{dN}\textstyle\sum_{t=1}^{N}\|\zeta_t-\mu-As(W\xi_t+\nu)\|^2,\\
&H_q=\tfrac{1}{N}\textstyle\sum_{t=1}^{N}\sum_{j=1}^{m}[y_t^{(j)}\ln q_j+(1-y_t^{(j)})\ln(1-q_j)]=\sum_{j=1}^{m}[q_j\ln q_j+(1-q_j)\ln(1-q_j)],\\
&y_t=s(W\xi_t+\nu),\ q_j=\tfrac{1}{N}\textstyle\sum_{t=1}^{N}y_t^{(j)},\\
&L(W)=\tfrac{1}{N}\textstyle\sum_{t=1}^{N}\ln q(\xi_t|y_t)=0.5\ln|W^TW|+\tfrac{1}{N}\sum_{t=1}^{N}\ln|D_s(y_t)|,\quad D_s \text{ as in eq.(138)},
\end{aligned}\tag{143}$$

where d_ζ is the dimension of ζ. The maximization of the first term is equivalent to $\min_{W,A}\sigma^2$, i.e., the least square learning on the widely studied three layer net $\xi\to y\to\zeta$ by $p(\zeta|\xi)=\int q(\zeta|y)p(y|\xi)dy=G(\zeta|As(W\xi+\nu)+\mu,\sigma^2 I)$.

The maximization of H_q is equivalent to minimize the entropy of the hidden units for more compact inner representations. A hidden unit with $(1-q_j)q_j\to 0$ constantly can be regarded as extra and thus can be discarded. The third term $L(W)$ is the likelihood of $q(\xi|y)$ that describes an inverse of $\xi\to y$. The optimal $q(\xi|y)$ can be searched via maximizing $H(\theta)$. As shown in [87], we can get $L(W)$ as given in eq.(143). Regarding $s(r)$ as a CDF, $|D_s(y_t)|$ actually describes a product of independent densities. Having a same format as $L(W)$ in eq.(105), maximizing $L(W)$ makes $y=W\xi$ becomes independent among its components.

Jointly, the convention least square error learning of minimizing σ^2 is improved by not only maximizing H_q that makes automatic selection on hidden units possible but also maximizing $L(W)$ that further regularizes learning via pushing the hidden units of the three layer nets to become independent.

Instead of making learning by the conventional back-propagation technique, it follows from eq.(65) that we can get a new adaptive algorithm with

$$\begin{aligned}
&Yang\ step && \text{updating } A,\mu,\sigma^2,q_j \text{ as in eq.(118) with } \zeta_t \text{ in the place of } x_t,\\
&Ying\ step && \text{getting } y_t=s(W\xi_t+\nu) \text{ and updating } W \text{ and } \nu \text{ by eq.(138), with } \xi_t \text{ in the place of } x_t,
\end{aligned}\tag{144}$$

during which each extra hidden unit is discarded via detecting $(1-q_j)q_j\to 0$ constantly.

The above learning can be simplified with each $q_j=0.5$ fixed. An appropriate m^* can be selected by eq.(135) via enumerating a number of m. Approximately we can also make $\min_{W,A}\sigma^2$ by the conventional back-propagation technique and select m^* by the criteria in eq.(137) and eq.(136), which were previously studied for both binary stochastic hidden units [109, 102, 103] (e.g., see Eqn. (56) in [100]) and deterministic real hidden units (see Eqn(88) and Eqn(89) in [97]). Details of recent studies are referred to [93] and [87].

5.7. Minimizing Fitting Error vs Enforcing Factor Independence

In fact, all the discussions in Sec. 5 deal with a twofold effort that targets at both minimizing the fitting error between a sample x_t and its reconstruction $\hat{x}_t$ and imposing the independence eq.(3) among the components of y for regularizing the indeterminacy of modeling.

Instead of directly imposing independence by eq.(3) throughout learning, such an independence may also be gradually approached during learning by optimizing a cost $\min Q_I$ with the independence eq.(3) satisfied at its minimum. Two typical examples are

$$Q_I = \begin{cases} \int p(y) \ln p(y) dy, & \text{(a)}, \\ -\int p(y) \ln p(y) dy, & \text{(b)}. \end{cases} \tag{145}$$

For the choice (a), pushing Q_I to its minimum, or equivalent maximizing the entropy of $p(y)$ as discussed in eq.(16), makes $p(y)$ become a uniform distribution that is one extreme case of the independence family by eq.(3). For the choice (b), pushing Q_I to its minimum, or equivalently minimizing the entropy of $p(y)$, pushes $p(y)$ to become an arbitrary δ distribution $\delta(y - y_0)$ that locates at any point y_0. Each $\delta(y - y_0)$ satisfies eq.(3) too and actually becomes each example of another extreme type of the independence family.

In the same time, minimizing the fitting error is also reached by optimizing another cost $\min Q_L$ with a learning process. Typical examples that have been discussed previously include

$$Q_L = \begin{cases} -\frac{1}{N} \sum_{t=1}^{N} \ln q(x_t|y_t), & \text{(a)}, \\ 0.5 d \ln \sigma^2, \ \ \sigma^2 = \frac{1}{dN} \sum_{t=1}^{N} \|x_t - A y_t\|^2, & \text{(b)}. \end{cases} \tag{146}$$

Making $\min Q_L$ alone usually suffers the indeterminacy of infinite many solutions. The joint implementation of both $\min Q_L$ and $\min Q_I$ will eliminate the indeterminacy and lead to an appropriate solution. It can be observed that an appropriate combination of $\min Q_L$ and $\min Q_I$ and the independence eq.(3) lead us to those previously discussed typical models. E.g., we have

(1) with Q_I by the case (b) in eq.(145) and Q_L by the case (a) in eq.(146) plus imposing the independence eq.(3), $\min(Q_L + Q_I)$ is actually equivalent to the harmony learning by eq.(52) with $z_q = 1$.

(2) In the above case with Q_L by the case (b) in eq.(146) instead of the case (a), $\min(Q_L + Q_I)$ is actually equivalent to the extended LMSER learning by eq.(139) with $z_q = 1$.

(3) If there is no need to consider noise, we can ignore $\min Q_L$. Then, $\min Q_I$ with Q_I by the case (b) in eq.(145) plus imposing the independence eq.(3) will lead us to eq.(8), eq.(19), eq.(100), eq.(105), and eq.(109), etc, respectively.

Moreover, other ways of combining $\min Q_L$ and $\min Q_I$ may result in different variants that deserve to be explored too. Different combinations comes from different choices of Q_I and Q_L. E.g., one class of variants may be obtained from replacing Q_I by the case (b) in eq.(145) with that by the case (a). Instead of targeting at a family of δ distribution $\delta(y - y_0)$, the regularization role of $\min Q_I$ is enhanced by targeting at a unique uniform distribution. However, this helps only when the number m of factors is pre-specified. In contrast, $\min Q_I$ with Q_I by the case (b) in eq.(145) enhances the harmony learning by eq.(52) or eq.(139) by pushing an extra dimension out of modeling such that an automatic selection of factors is implemented. Hence, we prefer Q_I by the case (b) in the cases with a unknown number m to be decided. In addition to the choices on Q_I, more choices other than those in eq.(146) can be used as Q_L. E.g., either $KL(\theta)$ by eq.(42) or $-H(\theta)$ by eq.(39) as well as their specific forms such as eq.(117) and eq.(139).

Still, differences come from the way that $\min Q_L$ and $\min Q_I$ are combined. In addition to $\min(Q_L + Q_I)$, we can also consider the following the three combining manners:

(a) We can first make $\min Q_I$ and then make $\min Q_L$. E.g., for a $q(y)$ given by eq.(3) and eq.(77), $\min Q_I$ results in $q_j = 0.5$ and $\min Q_L$ with Q_L by the case (a) becomes equivalent to the LMSER learning by one of eq.(18), eq.(23), and eq.(140);

(b) We can first make $\min Q_L$ and then make $\min Q_I$. E.g., $\min Q_L$ with Q_L by the case (a) results in $\sigma^2 = \sigma^2_{min}$, and then we make $\min Q_I$ with Q_I by the case (a) in eq.(145). That is, we have the following constrained learning

$$\min_{s.t. \ \sigma^2 = \sigma^2_{min}} \int p(y) \ln p(y) dy. \tag{147}$$

Specifically, when $\sigma^2_{min} = 0$, it becomes

$$\min_{s.t. \ Ax_t = y_t, \forall t} \int p(y) \ln p(y) dy, \tag{148}$$

which is equivalent to the Lagrange Constrained Neural Network (LCNN) in [81]. Eq.(147) extends eq.(148) to the cases with observation noises. Extensions can also be made with Q_L in other choices. When Q_I by the case (a) is replaced with Q_I by the case (b), we can further get a counterpart of eq.(147) as follows:

$$\min_{s.t. \ \sigma^2 = \sigma^2_{min}} -\int p(y) \ln p(y) dy, \tag{149}$$

which has a selection ability on m similar to various previously discussed types of harmony learning.

(c) More generally, we can $\min g(Q_L, Q_I)$ with $g(x, y)$ being monotonically increasing with both x, y. This $g(x, y)$ is called a combination formula. Further studies may deserve to be further conducted.

6. Extensions of ICA with Temporal Dependence

6.1. Typical Approaches for ICA with Temporal Dependence

Instead of considering independence among higher order statistics, the indeterminacy of the linear system eq.(1) or $y_t = Wx_t$ can also be removed by considering temporal dependence among samples of x_t (or equivalently of y_t) at different times. The key feature is that the 2nd order temporal dependence may be already enough for removing the indeterminacy, except some degenerated cases.

A number of approaches have been studied for exploring temporal dependences on solving the ICA problems. Roughly, we can again classify them into following three major types:

Joint diagonalization We use the term 'the j-channel' to denote a series of the j-component $\{y_t^{(j)}\}$ as time t goes. We say that being de-correlated across channels means the correlation matrices $E[y_t y_\tau^T] = R_{t,\tau}^y$ for $t \geq \tau$ should be diagonal. It follows from $y_t = Wx_t$ that $E[y_t y_\tau^T] = WE(x_t x_\tau^T)W^T = WR_{t,\tau}^y W^T$. That is, a matrix W should be determined to jointly diagonalize all the correlation matrices $R_{t,\tau}^x = E(x_t x_\tau^T)$ for $t \geq \tau$. This W can usually be determined from two or more such correlation matrices that are not identical. A solution W may either remain not varying with time for stationary signals or varies with time for non-stationary signals. Several techniques have been studied to implement such joint diagonalization. Readers are referred to [51, 47, 129, 40, 54].

Solving algebraic equations Similar to the situations discussed in Sec.2.1, by examining joint statistics of the time series $\{x_t\}$ and $\{y_t\}$ across different time delays, the constraint of $y_t = Wx_t$ and the constraint of requiring independence across channels will lead us to a number of algebraic equations. As a result, W can be determined by jointly solving these equations [15].

Optimizing a cost function Conceptually, any cost can be used for this purpose as long as it reaches its minimum (or equivalently maximum) when independence across channels is satisfied. There are many ways to get such a cost. First, we can get a cost by allowing errors from either the above joint diagonalization equations or the joint statistics based algebraic equations. Second, costs can be obtained from high order statistics (especially fourth-order cumulants) [129, 22, 78]. Third, a cost may also be heuristically motivated [2, 80]. E.g., with a belief that the temporal predictability of any signal mixture is less than or equal to that of any of its component source signals, a measure of temporal predictability is suggested in [80] via a short term and a long term moving average. Then, the measure is maximized to determine $y_t = Wx_t$. Usually, these costs consider independence across channels only in a sense of statistics up to a finite order (e.g., the 4 th order) .

In contrast, the cost by eq.(8) considers independence among component densities and thus conceptually covers statistics up to all the orders. Several efforts have been made to extend eq.(8) to cover temporal dependence.

Under the name of context sensitive ICA [62], $q(y_t^{(j)})$ is replaced by $q(\varepsilon_t^{(j)})$ with $y_t = Wx_t$ and an AR model $\varepsilon_t^{(j)} = y_t^{(j)} - \sum_{i=1}^{n_j} a_i y_{t-i}^{(j)}$. The AR coefficients a_i are updated together with the parameters of $q(\varepsilon_t^{(j)})$, basing on the past estimates $y_{t-i}^{(j)}, i \geq 1$. What is imposed here is actually $\prod_{j=1}^m q(y_t^{(j)}|y_{t-i}^{(j)})$, which says that components of y_t are independent, conditioning on the past values. This actually satisfies neither independence across channels nor $\prod_{j=1}^m q(y_t^{(j)})$.

As a special case of the temporal BYY learning with a F-architecture, a so called temporal ICA has been proposed [98, 96, 94], which extends eq.(8) with (i) $y_t = Wx_t$ replaced by $y_t = Wx_t + Ky_{t-1} + d$ such that the current estimate y_t is directly improved via temporal dependence; and (ii) $q(y_t^{(j)})$ is replaced by $q(y_t^{(j)}|y_{t-1}^{(j)})$ such that a non-linear dependence between $y_t^{(j)}$ and $y_{t-1}^{(j)}$ can also be covered. Moreover, a simplified algorithm that explores the 2nd order independence only is also given [92].

Moreover, we can also extend studies on the noise case of eq.(114) or eq.(116) by considering temporal dependence.

First, considering that e_t is not correlated with y_t, i.e., $Ee_t e_\tau^T = \Sigma$ when $t = \tau$ and $Ee_t e_\tau^T = 0$ when $t \neq \tau$, we have $E(x_t x_t^T) = AE[y_t y_t^T]A^T + \Sigma$ when $t = \tau$ and $R_{t,\tau}^x = E(x_t x_\tau^T) = E(Ay_t + e_t)(Ay_\tau + e_\tau)^T = AE[y_t y_\tau^T]A^T = AR_{t,\tau}^y A^T$ when $t > \tau$. Thus, we come to a similar situation of using the joint diagonalization approach for solving A, Σ.

Second, with the constraint $y_t = Wx_t$ replaced by the constraint $x_t = Ay_t + e_t$, we also come to the approach of using joint algebraic equations for solving A, Σ.

However, being different from the noiseless case, we also need to set up the inverse mapping $y = Wx$ to implement ICA, for which we need a model for y_t. In a special case of the temporal BYY learning, the first auto-regressive (AR) is used for modeling y_t. Together with eq.(114), we are lead to the well known linear state space model:

$$y_t = By_{t-1} + \varepsilon_t,\ x_t = Ay_t + e_t. \tag{150}$$

In the literature of control theory, this model has been extensively studied together with the Kalman filter that estimates y_t from the current x_t, under the following conditions

$$\varepsilon_t \sim G(\varepsilon_t|0,\Sigma),\ e_t \sim G(e_t|0,\Sigma),\ E(y_{t-1}\varepsilon_t^T) = 0,\ E(y_t e_t^T) = 0,\ E(e_t\varepsilon_t^T) = 0. \tag{151}$$

In these studies, however, A, B, Σ, Λ should be already known, and the number of states or the dimension of y should also be given. This is not practical in many cases. In [98, 96, 94, 92], as a special case of the temporal BYY learning with a B-architecture, an alternative state space model is suggested with the independence condition eq.(3) extended over all the times, i.e.,

$$q(\mathbf{y}) = \textstyle\prod_{j=1}^m q(\mathbf{y}^{(j)}),\ \mathbf{y} = \{y_t\}_{t=1}^T,\ \mathbf{y}^{(j)} = \{y_t^{(j)}\}_{t=1}^T, \tag{152}$$

such that A, B as well as Σ and Λ can remain unknowns to be specified. This alternative state space model can also be understood as an extension of the classical factor analysis by eq.(4) that has been widely studied in the literature of statistics for several decades [48]. With temporal dependence $y_t = By_{t-1} + \varepsilon_t$ added and the independence constraint by eq.(152) imposed, we call this extension as temporal factor analysis (TFA) [92].

Moreover, temporal BYY learning also provides a general guidance for extending Binary FA, NFA, LMSER and its supervised variant to temporal cases, as to be introduced in the rest of this section.

6.2. Two Types of Temporal BYY Systems

Conceptually, the Bayesian Ying-Yang can be used to learn temporal dependence via eq.(32) or eq.(42) by inserting a pair of temporal processes $\mathbf{x} = \{x_t\}_{t=1}^T$ and $\mathbf{y} = \{y_t\}_{t=1}^T$ in the place of the variables x, y in eq.(29), which thus can be called a *temporal Bayesian Ying-Yang (TBYY)* process system. In implementation, this situation is usually too complicated to be handled directly and further simplification is needed. First, given a sample x_t we let $p(x_t|\mathbf{x},\mathbf{y}) = \delta(x_t - \bar{x}_t)$ since 'knowing $x_t = \bar{x}_t$ definitely' means being irrelevant to any environment. Second, we adopt the causal assumption that x_t, y_t only depends on their values at past $\tau < t$ but not on any future $\tau > t$. Under these assumptions, the form eq.(32) becomes the following summation:

$$H(p\|q) = \textstyle\sum_{t=1}^N H_t - \ln z_q, \tag{153}$$

where z_q takes a role similar to that in eq.(52) and eq.(53).

That is, the harmony measure of the whole process is uniformly contributed from each time t as follows

$$H_t = \textstyle\int p(\omega_t) H_t(p\|q,\omega_t) d\omega_t,\ H_t(p\|q,\omega_t) = \int p(y_t|x_t,\omega_t) \ln\left[q(x_t|y_t,\omega_t) q(y_t|\omega_t)\right] dy_t, \tag{154}$$

where ω_t is a set that consists of all of or a part of the past samples $\{x_\tau, y_\tau, \tau < t\}$. Considering Tab. 2(C), we further have approximately

$$H_t = H_t(p\|q,\bar{\omega}_t),\ \bar{\omega}_t = \textstyle\int \omega_t p(\omega_t) d\omega_t. \tag{155}$$

Further imposing the Markovian assumption that x_t, y_t depend on only a finite number of their past sample values, ω_t can be a finite dimensional vector and thus $p(y_t|x_t,\omega_t), q(x_t|y_t,\omega_t), q(y_t|\omega_t)$ can be designed in regular structures. In the special cases of the 1st order Markovian, ω_t relates to y_{t-1}, x_{t-1} only. Also knowing y_{t-1} implies that y_t becomes independent from x_{t-1} since y_{t-1} is an inner code of x_{t-1}. Thus, ω_t simply relates to y_{t-1} only. Similarly, knowing y_t implies that x_t becomes independent from y_{t-1}. As a result, we are suffice to consider the following 1st order BYY state space system:

$$p(x_t, y_t|y_{t-1}) = p(y_t|x_t, y_{t-1})\delta(x_t - \bar{x}_t),\ q(x_t, y_t|y_{t-1}) = q(x_t|y_t) q(y_t|y_{t-1}), \tag{156}$$

to be inserted into H_t in eq.(154).

Thus, if it follows from eq.(154) that we have

$$H(p\|q) = \sum_{t=1}^{N} H_t - \ln z_q,\ H_t = \int p(y_t|x_t, \bar{y}_{t-1}) \ln [q(x_t|y_t)q(y_t|\bar{y}_{t-1})]dy_t,$$
$$\bar{y}_{t-1} = \int y_{t-1} p(y_{t-1}|\bar{x}_{t-1}, \bar{y}_{t-2})dy_{t-1}. \quad (157)$$

When $p(y_t|x_t, \bar{y}_{t-1})$ is free, e.g., for a B-architecture, it follows from maximizing $H(p\|q)$ that

$$p(y_t|x_t, \bar{y}_{t-1}) = \delta(y_t - \bar{y}_t),\ \bar{y}_t = \max_{y_t}[q(x_t|y_t)q(y_t|\bar{y}_{t-1})]. \quad (158)$$

Alternatively, we can also get a different extension of BYY system to temporal dependence. We consider the pair in eq.(29) with x, y being simply x_t, y_t at time t instantaneously, subject to the satisfaction of the following temporal dependence:

$$q(y_t) = \begin{cases} \int q(y_t|\omega_t)q(\omega_t)d\omega_t, & \text{(a) general,} \\ \int q(y_t|y_{t-1})q(y_{t-1})dy_{t-1}, & \text{(b) the 1st order.} \end{cases} \quad (159)$$

That is, temporal dependence is considered via this constraint, instead of being imposed in the Ying-Yang matching learning. This BYY system can be called the TBYY instantaneous system. Given $p(x_t) = \delta(x_t - \bar{x}_t)$, it follows from eq.(32) at $z_q = 1$ that the harmony measure $H_t = \int p(y_t|x_t)\delta(x_t - \bar{x}_t) \ln [q(x_t|y_t)q(y_t)]dx_t dy_t$ becomes simply

$$H_t = \int p(y_t|\bar{x}_t) \ln [q(\bar{x}_t|y_t)q(y_t)]dy_t, \text{ subject to the constraint of eq.(159).} \quad (160)$$

Particularly, for a B-architecture it follows from $\max_{p(y_t|x_t)} H_t$ that we have

$$p(y_t|x_t) = \delta(y_t - \bar{y}_t),\ \bar{y}_t = \max_{y_t}[q(x_t|y_t)q(y_t)],\ H_t = \ln [q(\bar{x}_t|\bar{y}_t)q(\bar{y}_t)],\ subject\ to\ eq.(159). \quad (161)$$

Further considering the contributions from each time t uniformly, the harmony measure of the entire process is simply the summation by eq.(153) again.

When y_t is discrete or Gaussian, the integral of eq.(159) will either become a summation or be analytically solved such that it can be directly inserted into $H(p\|q)$ for implementing learning. However, when y_t is a real nonGaussian random variable, the integral over y in eq.(159) is difficult to handle. In help of Tab. 2(C) again, it follows from eq.(159) that we can have approximately

$$q(y_t) = \int q(y_t|\bar{y}_{t-1})q(y_{t-1})dy_{t-1} = q(y_t|\bar{y}_{t-1}). \quad (162)$$

Moreover, $p(y_t|x_t)$ will be decided via maximizing $H(p\|q)$ by eq.(160), resulting in

$$p(y_t|x_t) = \delta(y_t - \bar{y}_t),\ \bar{y}_t = \max_{y_t}[q(x_t|y_t)q(y_t|\bar{y}_{t-1})]. \quad (163)$$

Putting eq.(158) into eq.(157) and eq.(163) into eq.(160), we can observe that the TBYY process system under the approximation in eq.(154) and TBYY instantaneous system under the approximation in eq.(162) become equivalent.

However, the two types of TBYY systems are conceptually different. The TBYY process system considers a best harmony of two representations in the form of the joint distribution for two entire temporal processes of $\mathbf{x}$ and $\mathbf{y}$, including all the temporal dependences in consideration. In contrast, the TBYY instantaneous system only emphasizes a best harmony of two representations of the joint distribution for only the pair x_t, y_t instantaneously at each time t, not directly including temporal dependences in this harmony. Instead, the temporal dependence is considered via the constraint eq.(159).

For implementation, we can also get $\nabla_\theta H(p\|q)$ in a way similar to eq.(53) and with $z_q = 1$ in consideration. For $q(x|y) = G(x|Ay + \mu, \Sigma)$, we can further get its specific form that is given by eq.(117) either directly on a TBYY instantaneous system or with all the appearances of $q(y_t)$ replaced with $q(y_t|\bar{y}_{t-1})$ for a TBYY process system.

In addition to making the harmony learning as above discussed, we can also conduct learning in two stages and make parameter learning in Stage I by eq.(42). In a similar way, we can get

$$KL(p\|q) = \sum_{t=1}^{N} KL_t,$$
$$KL_t = \begin{cases} \int p(y_t|x_t, \bar{y}_{t-1}) \ln \frac{p(y_t|x_t, \bar{y}_{t-1})}{q(x_t|y_t)q(y_t|\bar{y}_{t-1})} dy_t, & \text{TBYY process system,} \\ \int p(y_t|x_t) \ln \frac{p(y_t|x_t)}{q(x_t|y_t)q(y_t)} dy_t,\ subject\ to\ eq.(159), & \text{TBYY instantaneous system.} \end{cases} \quad (164)$$

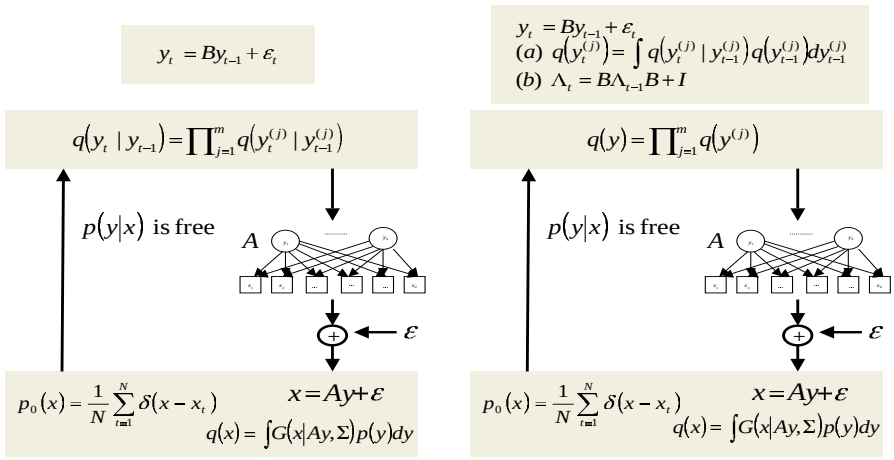


Figure 11. *Two Types of Temporal NFA*

Again, an estimate $\bar{y}_{t-1}$ is already available from the system at time $t-1$ when H_t or KL_t is considered at time t. Particularly, for a B-architecture, minimizing $KL(p\|q)$ leads to

$$\begin{aligned} &(a) \quad p(y_t|x_t, \bar{y}_{t-1}) = \frac{q(x_t|y_t)q(y_t|\bar{y}_{t-1})}{\int q(x_t|y_t)q(y_t|\bar{y}_{t-1})dy_t}, \text{ for a TBYY process system,} \\ &(b) \quad p(y_t|x_t) = \frac{q(x_t|y_t)q(y_t)}{\int q(x_t|y_t)q(y_t)dy_t}, \text{ for a TBYY instantaneous system.} \end{aligned} \tag{165}$$

Moreover, putting eq.(165) into eq.(164), we observe again that $KL(p\|q)$ for both the temporal BYY types becomes equivalent under the approximation in eq.(162).

Imposing the independence condition eq.(152) on both types of TBYY systems, we are lead to TBYY independence learning. Further recalling $H(p\|q)$ in eq.(32) and $KL(p\|q)$ in eq.(42), we can find that one key difference here from the BYY independence learning introduced in Sec.3 is that an additional structure has been imposed on $q(y)$ via either directly a regression between y_t and y_{t-1} or assembly the marginal density relation eq.(159). Correspondingly, the forward path $p(y|x)$ should also be modified with its dependence to y_{t-1} added in. These changes further result in changes in parameter learning and model selection for making TBYY independence learning, which will be introduced in the rest of this section.

6.3. TFA, Temporal NFA, and Space Dimension Selection

As shown in Fig.11(a), we start at extending the B-architecture given in Fig.5 into a TBYY B-architecture as follows:

$$q(x|y) = G(x|Ay + \mu, \Sigma),\ y_t = By_{t-1} + \varepsilon_t,\ \ B \text{ is a diagonal matrix,} \tag{166}$$

where a temporal structure is imposed on y_t and its stability is guaranteed by $\|B\| < 1$. $\|B\|$ is the norm of B, i.e., the positive root of the largest eigen-value of BB^T. For an example, we can consider

$$B = diag[b_1, \cdots, b_m],\ b_j = \frac{e^{c_j} - e^{-c_j}}{e^{c_j} + e^{-c_j}},\ C = diag[c_1, \cdots, c_m], \tag{167}$$

where $|b_j| < 1$ is always satisfied.

Moreover, ε_t is a white process in a strict sense that ε_t is independent from both y_{t-1} and ε_τ for any $\tau \neq t$. Also, the components of $\varepsilon_t = [\varepsilon_t^{(1)}, \cdots, \varepsilon_t^{(m)}]^T$ are independent. Two typical examples are a Gaussian and a Gaussian mixture as follows:

$$q(\varepsilon_t|\theta_\varepsilon) = \begin{cases} G(\varepsilon_t|0, I), & \text{(a) Gaussian,} \\ \prod_{j=1}^{m} q(\varepsilon_t^{(j)}|\theta_\varepsilon^{(j)}), & \text{(b) } q(\varepsilon_t^{(j)}|\theta_\varepsilon^{(j)}) \text{ by eq.(75) with } y^{(j)} \text{ replaced by } \varepsilon_t^{(j)}. \end{cases} \tag{168}$$

With the above setting, the process $\{y_t\}_{t=1}^T$ will satisfy eq.(152) as long as the components of y_0 are also initialized to be independent. Specifically, eq.(166) with the above case (a) is a temporal extension of the classic factor analysis and thus is called temporal factor analysis (TFA). As shown in [92] and to be further elaborated in the next subsection, one key feature of TFA is that the rotation indeterminacy by eq.(115) of the classic FA has been removed. Similarly, eq.(166) with the above case (b) is a temporal extension of the NFA in Fig.5 and thus is called temporal NFA.

The learning can still be implemented by the Ying-Yang alternative procedure eq.(65).

For the Ying step, $q(x|y)$ is still updated in the same way as in eq.(118), while the updating on $q(y)$ should be modified. In the implementation of a TBYY process system, it is updated to increase $\eta_t \ln q(y_t|y_{t-1})$ as follows:

$$\begin{aligned}
&updating\ B: && \varepsilon_t = y_t - By_{t-1},\ C^{new} = C^{old} + \eta_0\eta_t(I-B^2)\phi(\varepsilon_t)y_{t-1}^T,\ \text{and get } B \text{ by eq.(167);}\\
&where && \phi(\varepsilon) = \begin{cases} -\varepsilon, & \text{(a) TFA,}\\ \frac{\partial q(y_t|y_{t-1})}{\partial \varepsilon}, & \text{(b) temporal NFA;}\end{cases}\\
&updating\ \theta_\varepsilon: && \begin{cases} no\ action, & \text{(a) TFA,}\\ \text{by eq.(119) and eq.(126) with } y^{(j)} \text{ replaced by } \varepsilon_t^{(j)}, & \text{(b) temporal NFA.}\end{cases}
\end{aligned} \tag{169}$$

where the updating on B comes from

$$\begin{aligned}
&G_B = \eta_t \nabla_B \ln q(y_t|y_{t-1}) = \eta_t \phi(\varepsilon_t)\hat{y}_{t-1}^T, G_C = \eta_t \nabla_C \ln G(y_t|B\hat{y}_{t-1}, I),\\
&dB = (I-B^2)dC,\ Tr[G_B^T dB] = Tr[G_B^T(I-B^2)dC] = Tr[G_C^T dC],\ G_C = (I-B^2)G_B.
\end{aligned} \tag{170}$$

In the implementation of a TBYY instantaneous system, it is difficult to handle temporal NFA due to the integral over y in eq.(159). However, to handle TFA we simply have

$$q(y_t) = G(y_t|0,\Lambda_t), \Lambda_t = B\Lambda_{t-1}B + I,\ \Lambda_t = E[y_t y_t^T],\ \Lambda_{t-1} = E[y_{t-1}y_{t-1}^T]. \tag{171}$$

and thus B is updated by increasing $\eta_t \ln q(y_t)$ as follows:

$$\begin{aligned}
&C^{new} = C^{old} + \eta_0\eta_t(I - B^{old\ 2})G_\Lambda B^{old}\Sigma_{t-1}^y,\ G_\Lambda = \Lambda^{old\ -1}y_t y_t^T \Lambda^{old\ -1} - \Lambda^{old\ -1},\\
&\Lambda^{new} = B\Lambda^{old}B + I,\ \text{and get } B \text{ by eq.(167).}
\end{aligned} \tag{172}$$

The updating comes from $G_\Lambda = \eta_t \nabla_\Lambda \ln G(y_t|0,\Lambda) = \eta_t[\Lambda^{-1}y_t y_t^T \Lambda^{-1} - \Lambda^{-1}]$, $d\Lambda = dB\Sigma^y B^T + B\Sigma^y dB^T$, and thus $G_B = G_\Lambda B\Sigma^y$.

For the Yang step, y_t is obtained with a different degree of difficulty for performing TFA and temporal NFA, respectively. For TFA, it follows from eq.(158) and eq.(161) that we directly get

$$y_t = \begin{cases} [I + A^T\Sigma^{-1}A]^{-1}(A^T\Sigma^{-1}(x_t-\mu) + By_{t-1}), & \text{(a) TBYY process system,}\\ [(B\Sigma_{t-1}^y B + I)^{-1} + A^T\Sigma^{-1}A]^{-1}A^T\Sigma^{-1}(x_t-\mu), & \text{(b) TBYY instantaneous system.}\end{cases} \tag{173}$$

For temporal NFA, it is difficult to get y_t on a TBYY instantaneous system due to the integral over y in eq.(159). However, we can still get y_t on a TBYY process system by solving the following nonlinear optimization

$$y_t = arg\max_y [G(x_t|Ay+\mu,\Sigma)\prod_j \sum_i \beta_{ji} G(y^{(j)}|m_{ji} + b_j y_{t-1}^{(j)}, \lambda_{ji}^2)], \tag{174}$$

via running Tab.2(A) in a few iterations, with all the appearances of m_{jr} replaced by $m_{jr} + b_j y_{t-1}^{(j)}$.

In the special case that $B = 0$ (thus $C = 0, b_j = 0$), TFA returns back to FA and temporal NFA returns back to NFA, respectively. E.g., both the cases in eq.(173) return back to eq.(127) for FA and eq.(174) returns back to eq.(76) for NFA.

In all the above discussions, each dimension $\varepsilon_t^{(j)} = y_t^{(j)} - By_{t-1}^{(j)}$ has a fixed variance of 1. To select the best value m of the space dimension, we can enumerate m for a number of values and make parameter learning as above.

On a TBYY process system, we make the selection by eq.(40) with $J(m)$ in Tab.3. For simplicity, we can approximately use $J(m)$ in Fig.5 for both TFA and temporal NFA. While on a TBYY instantaneous system, we make the selection by directly using eq.(40) for temporal NFA in Tab.3.

However, this enumerating based selection costs extensively. Similar to eq.(130), it follows from the singular value decomposition of a general matrix that we consider a linear mapping $\Lambda^{0.5}V^T$ on y_t and ε_t with the linear state space eq.(150) turned into the following form

$$y_t = DB_vD^{-1}y_{t-1} + \varepsilon_t,\ E(\varepsilon_t\varepsilon_t^T) = D^2 = \Lambda,\ B_v = V^TBV,\ B \text{ is given by eq.(167)},$$
$$x_t = Ay_t + e_t,\ A^TA = I,\ E(e_te_t^T) = \sigma^2 I. \tag{175}$$

In this alternative state space, maximizing H_t by eq.(157) will, via maximizing $\eta_t \ln G(y_t|DB_vD^{-1}y_{t-1}, \Lambda)$, push the variance λ_j^2 towards zero constantly if the j-th dimension is extra. As a result, model selection on m happens automatically during parameter learning via simply discarding the dimension j. That is, discarding the components $y_t^{(j)}, y_{t-1}^{(j)}, \varepsilon_t^{(j)}$, eliminating the j-th column of A, and also eliminating the j-th column and the j-th row of V and D. In implementation, this discarding is made when λ_j^2 approaches near 0 but before becoming zero to avoid the computing difficulty on D^{-1}. Since B_v is no longer diagonal, eq.(175) no longer directly implements the TFA but still equivalent to the TFA.

Specifically, an adaptive learning algorithm can be obtained. For its Yang Step, we can simply replace the case (a) of eq.(173) with

$$\hat{y}_t = (\Lambda + \sigma^2 I)^{-1}\Lambda[A^T(x_t - \mu) + \Lambda^{0.5}B_v\Lambda^{-0.5}\hat{y}_{t-1}]. \tag{176}$$

Moreover, the Ying step consists of the part of updating $q(x|y)$ in eq.(118) with A updated in help of g_A by either of two approaches in Tab.1 and the part of updating $q(y)$ in help of the gradients of $\eta_t \ln G(y_t|DB_vD^{-1}y_{t-1}, \Lambda)$:

$$\begin{aligned}
&G_C = \eta_t \frac{\partial \ln G(y_t|DB_vD^{-1}y_{t-1}, \Lambda)}{\partial C} = \eta_t(I - B^2)diag[VRV^T],\\
&R = D^{-1}\varepsilon_t y_{t-1}^T D^{-1},\ \varepsilon_t = y_t - DB_vD^{-1}y_{t-1},\\
&G_V = \eta_t \frac{\partial \ln G(y_t|DB_vD^{-1}y_{t-1}, \Lambda)}{\partial V} = \eta_t B(V^TR + VR^T),\\
&G_D = \eta_t \frac{\partial \ln G(y_t|DB_vD^{-1}y_{t-1}, \Lambda)}{\partial D} = \eta_t D^{-1}diag[B_vR + RB_v + D^{-1}\varepsilon_t\varepsilon_t^T D^{-1} - I].
\end{aligned} \tag{177}$$

We can directly update $D^{new} = D^{old} + \eta_0 G_D$, $C^{new} = C^{old} + \eta_0 G_C$, and get B by eq.(167). Also, we use G_V to update V subject to $V^TV = I$ by any of two approaches in Tab. 1.

6.4. Temporal ICA, Kalman Filter, and Identifiable State Spaces

We consider the special case that there is no observation noise on x i.e., we have $\Sigma = \sigma^2 I$ with $\sigma^2 = 0$. For both types of temporal BYY learning, it follows from eq.(173) and eq.(174) that we get $y_t = Wx_t$ with $W = (A^TA)^{-1}A^T$ being simply the pseudo inverse of A, which is independent of time $t-1$. Again, we are lead to the degenerated case by eq.(99) and get a temporal extension of ICA.

Moreover, this temporal ICA is implemented in the same format as in eq.(105) with

(1) W is updated by eq.(97) or eq.(103);

(2) $q(y_t^{(j)})$ is updated by eq.(169) in the implementation of a TBYY process system. When $q(y_t|y_{t-1})$ is nonGaussian, it implements a temporal ICA. Even when $q(y_t|y_{t-1})$ is Gaussian, as firstly shown in [94], the rotation indeterminacy by eq.(115) can be removed when the elements of the diagonal matrix B are different, which can be observed from a fact that $y_t' = \phi B\phi^T y_{t-1} + \phi\varepsilon_t$ after a rotation $y_t' = \phi y_t, \phi\phi^T = I$, with $\phi_B = \phi B\phi^T y_{t-1}$ no longer being diagonal and thus the independence condition eq.(152) broken.

(3) $q(y_t^{(j)})$ is updated by eq.(172) in the implementation of a TBYY instantaneous system. Again, it is difficult to handle the case of nonGaussian $q(y_t^{(j)}|y_{t-1}^{(j)})$ due to the integral over y in eq.(159). However, the case of a Gaussian $q(y_t|y_{t-1})$ can be handled, with the rotation indeterminacy by eq.(115) removed too, which can be observed from a fact that Λ_t in eq.(171) will become $Ey_t'y_t'^T = \phi Ey_ty_t^T\phi^T$ that does not remain diagonal after a rotation $y_t' = \phi y_t, \phi\phi^T = I$.

Furthermore, we can select an appropriate dimension m for y_t as discussed at the end of Sec. 5.3.

Another interesting special case of eq.(150) has been widely studied for decades in the literature of control theory. Usually, e_t, ε_t are further assumed to come from the following Gaussian processes:

$$G(e_t|0, \Sigma),\ Ee_t = 0,\ Ey_te_t^T = 0,\ G(\varepsilon_t|0, \Lambda),\ E\varepsilon_t = 0,\ Ey_{t-1}\varepsilon_t^T = 0,\ Ee_t\varepsilon_t^T = 0. \tag{178}$$

Also, the parametric matrices A, B as well as the variances of Gaussian e_t, ε_t have to be known. The task is to estimate $\hat{y}_t$ upon observing x_t and then get $\hat{x}_t = A\hat{y}_t$ as the result after filtering out the noise in x_t. The optimal solution is the well known Kalman filter [36].

As shown in [14], the estimate y_t given by the Kalman filter is equivalent to

$$y_t = E[y_t|x_t] = \int y_t p(y_t|x_t)dy_t,\ p(y_t|x_t) = \frac{q(x_t|y_t)q(y_t)}{\int q(x_t|y_t)q(y_t)dy_t}, \quad \text{subject to eq.(159).} \tag{179}$$

Alternatively, it can be observed that $\min_{p(y_t|x_t)} KL_t$ with KL_t by eq.(164) also results in the above $p(y_t|x_t)$. Moreover, since e_t, ε_t (and thus $q(x_t|y_t), q(y_t)$) are all Gaussians, we have

$$E[y_t|x_t] = \hat{y}_t = \max_{y_t}[q(x_t|y_t)q(y_t)]. \tag{180}$$

That is, y_t obtained by eq.(179) is equivalent to $\hat{y}_t$ by eq.(161). On the instantaneous Bayesian Ying-Yang system with the special settings by eq.(150) and eq.(178), both the learning by maximizing $H(p\|q)$ and the learning by minimizing $KL(p\|q)$ are equivalent to the well known Kalman filter on estimating y_t, which is given by the cases (b) in eq.(173).

More than providing a different insight, we can also get some new results from maximizing $H(p\|q)$. Two examples are introduced as follows:

(1) In the literature of the Kalman filter studies, knowing the dimension m of y_t is implied in requiring that the parametric matrices A, B are known. Little studies have been made on how to determine an appropriate m. In some applications, there may be two or more sets of models with different dimensions. We may choose one that is the smallest in $J(m)$ by Tab.3.

(2) When ε_t is nonGaussian, we get a generalized Kalman filter by approximating the density of ε_t via a Gaussian density and then estimates y_t as the cases in eq.(178). In help of eq.(174) and Tab.2(A), we can also estimate y_t with the density of ε_t modeled by a Gaussian mixture.

Requiring that the matrices A, B are known is too restrictive in many temporal modeling problems. However, the problem of estimating all the unknowns y_t, A, B, as well as Σ and Λ only from $\{x_t\}$ will suffer intrinsic indeterminacy, which can be observed from the perspective of making a linear mapping $y'_t = \phi y_t$, with a general matrix ϕ. It follows from eq.(150) that we get the same form $x_t = A'y'_t + e_t, y'_t = B'y'_{t-1} + \varepsilon'_t$ with

$$A' = A\phi^{-1},\ B' = \phi B\phi^{-1},\ \varepsilon'_t = \phi\varepsilon_t, \tag{181}$$

and ε'_t remains to be a Gaussian in the form of ε_t as in eq.(178). That is, y_t is not identifiable in this state space model by eq.(150) and eq.(178).

This type of indeterminacy can be removed by imposing certain constraints on A or B or ε_t such that A' or B' or ε'_t becomes no longer satisfying the constraints. In the Kalman filter, A should remain unchanged at a given one and thus the only possibility is $\phi = I$. While in TFA, the satisfaction of eq.(152) on eq.(178) actually means that both B, Λ are diagonal matrices. It follows from eq.(181) that a linear mapping $y'_t = \phi y_t$ with $B \in \mathcal{D}^*$ and $B' \in \mathcal{D}^*$ means that $\phi = \Pi D$ with $D \in \mathcal{D}, \Pi \in \mathcal{P}$, where $\mathcal{D}^*, \mathcal{D}, \mathcal{P}$ are the following families:

$$\begin{aligned} &\mathcal{D} = \{D : D \text{ is diagonal}\},\ \mathcal{P} = \{\Pi : \Pi \text{ is a permutation matrix}\}, \\ &\mathcal{D}^* = \{D \in \mathcal{D} :\ \text{all diagonal elements of } D \text{ are different}\}. \end{aligned} \tag{182}$$

Observing that $\phi = \Pi D$ or equivalently

$$y'_t = \Pi D y_t, \tag{183}$$

there is an one-to-one correspondence between the m series $\{y_t^{'(j)}\}_{t=1}^T, j = 1, \cdots, m$ and the m series $\{y_t^{(i)}\}_{t=1}^T$, $i = 1, \cdots, m$. That is, for each series $\{y_t^{(i)}\}_{t=1}^T$ there must be one and only one series $\{y_t^{'(j)}\}_{t=1}^T$ such that both the series have a same waveform with difference only in a constant scale and a permutation (i.e., an order change $i \neq j$). In many applications, this type of difference is ignorable, and thus we also say that y_t is identifiable if we are able to specify y_t via y'_t in a sense of eq.(183).

Other types of indeterminacy exist also. E.g., on the additive decomposition $\Sigma_x = A\Lambda_t A^T + \Sigma$ from $x_t = Ay_t + e_t$ and on the additive decomposition $\Lambda = B\Lambda B^T + I$ from $y_t = By_{t-1} + \varepsilon_t$. Generally, the identifiable problem should be studied from the perspective of uniquely estimating all the parameters of A, B, as well as Σ and Λ subject to all the constraints in the state space model by eq.(150) and eq.(178). E.g., on a set of samples of a large size, the constraints consist of :

$$\Sigma_x = A\Lambda A^T + \Sigma,\ \Lambda = B\Lambda B^T + I,\ \Sigma_x \geq 0,\ \Lambda \geq 0,\ \Sigma \geq 0,\ \|B\| < 1,\ R_1^y = B\Lambda\ R_1^x = AB\Lambda A^T. \tag{184}$$

The last line comes from $R_1^y = E[y_t y_{t-1}^T] = B\Lambda$ and $R_1^x = E[x_t x_{t-1}^T] = AB\Lambda A^T$, where $\Lambda = E[y_t y_t^T] = E[y_{t-1} y_{t-1}^T]$ as well as $R_1^x = E[x_t x_{t-1}^T] = E[(Ay_t + e_t)(Ay_{t-1} + e_{t-1})^T] = AR_1^y A^T$ with consideration of $Ey_t e_{t-1}^T = 0, E[e_t y_{t-1}^T] = 0, E[e_t e_{t-1}^T] = 0$.

6.5. Temporal BFA, Temporal LMSER, and Supervised Recurrent Net

We further consider the cases that y_t is a binary vector. We start at extensions of Bernoulli NFA (BFA) to temporal BFA. Considering that y_t is a binary vector $[y_t^{(1)}, \cdots, y_t^{(m)}]^T$ that satisfies eq.(152) with each $y_t^{(j)}$ taking either 1 or 0, the key point is how to extend the Bernoulli distribution by eq.(77) to encoding temporal relations.

The simplest case is considering the first order temporal relation as follows:

$$q(y_t^{(k)} = j | y_{t-1}^{(k)} = i) = \pi_{ji}^{(k)}, \ i, j = 0, 1, \tag{185}$$

which describes the probability for $y_t^{(k)}$ to transfer from i to j. The process $\{y_t\}_{t=1}^T$ by the above eq.(185) consists of m independent Markovian chains of the 1st order, respectively, which are not directly observable. As in Fig.12, the observation process $\{x_t\}_{t=1}^T$ are generated from $\{y_t\}_{t=1}^T$ via the Ying path $q(x_t|y_t)$. Thus, we encounter a variant of the classic Hidden Markov Model (HMM) [64, 9], with one variable that takes m discrete values being replaced by a binary vector $[y_t^{(1)}, \cdots, y_t^{(m)}]^T$ that satisfies eq.(152). We called this type of HMM as the independent HMM. Especially, we call it the independent binary HMM when each $y_t^{(j)}$ only takes either 1 or 0.

In literature, learning on a HMM is made under the maximum likelihood (ML) principle by the well known Baum-Welch algorithm [64], which is a type of EM algorithm. However, in addition to a high computing cost by the Baum-Welch algorithm, it is well known that the maximum likelihood principle is usually weak on making model selection (i.e., deciding the number m), especially in a case of a small size of training samples. Usually, this number m is assumed to be known in advance. In contrast, learning on independent HMM can be implemented in help of TBYY harmony learning, with model selection made either automatically during an adaptive learning or subsequently after learning via a new class of model selection criteria.

Specifically, the TBYY harmony learning is implemented by the Ying-Yang alternative procedure eq.(65), with its detailed form modified as follows.

The Yang step is obtained from eq.(78) simply with q_j replaced either by $\pi^{(j)}$ in the implementation of temporal BYY instantaneous system or by

$$q_j = (1 - y_{t-1}^{(j)})\pi_{10} + y_{t-1}^{(j)}\pi_{11} \tag{186}$$

in the implementation of a TBYY process system.

For the Ying step, $q(x|y)$ is still updated in the same way as in eq.(118), while the updating on $q(y)$ should be modified. For a case that either $z_q = 1$ or z_q is given by eq.(50), we have $\eta_t = 1$ and thus update $q(y)$ to increase $\ln q(y_t|y_{t-1})$ as follows

$$\begin{aligned}
&(a) \quad \pi_{ji}^{(k)\ new} = \frac{\pi_{ji}^{(k)\ old} + \eta \bar{\delta}_{j y_t^{(k)}} \bar{\delta}_{i y_{t-1}^{(k)}}}{1 + \eta \bar{\delta}_{i y_{t-1}^{(k)}}}, \ i, j = 0, 1, \ for\ \forall k; \\
&(b) \quad \pi^{(k)\ new} = \pi_{11}^{(k)\ new} \pi^{(k)\ old} + \pi_{10}^{(k)\ new}(1 - \pi^{(k)\ old}), \ for\ \forall k; \\
&(c) \quad \text{if } \pi^{(k)}(1 - \pi^{(k)}) \text{ approaches 0 constantly,} \\
&\qquad \text{we discard the state } j \text{ as well as all the related parameters, and let } m = m - 1,
\end{aligned} \tag{187}$$

where $\eta\bar{\delta}_{ij}$ is same as given in eq.(37).

While in the implementation of temporal BYY instantaneous system by eq.(160) subject to eq.(159), the step (a) is replaced by:

$$\begin{aligned}
(a) \qquad \pi_{j1}^{(k)\ new} &= \frac{\pi_{j1}^{(k)\ old} + \eta \bar{\delta}_{j y_t^{(k)}} \frac{\pi^{(k)\ old}}{\pi^{*(k)}}}{1 + \eta \frac{\pi^{(k)\ old}}{\pi^{*(k)}}}, \ for\ \forall k, \ j = 0, 1, \\
\pi_{j0}^{(k)\ new} &= \frac{\pi_{j0}^{(k)\ old} + \eta \bar{\delta}_{j y_t^{(k)}} \frac{1 - \pi^{(k)\ old}}{1 - \pi^{*(k)}}}{1 + \eta \frac{1 - \pi^{(k)\ old}}{1 - \pi^{*(k)}}}, \ for\ \forall k, \ j = 0, 1.
\end{aligned} \tag{188}$$

With a large initial value, m will automatically reduce to an appropriate one during the above learning due to step (c) in eq.(187). Alternatively, we can also make parameter learning and model selection sequentially in two stages. At Stage I, parameter learning can be made by eq.(42) or equivalently by the ML learning in help of the Baum-Welch algorithm [64, 9]. Then, the number m is decided at Stage II in help of eq.(40) that takes a detailed form given in Tab.3:

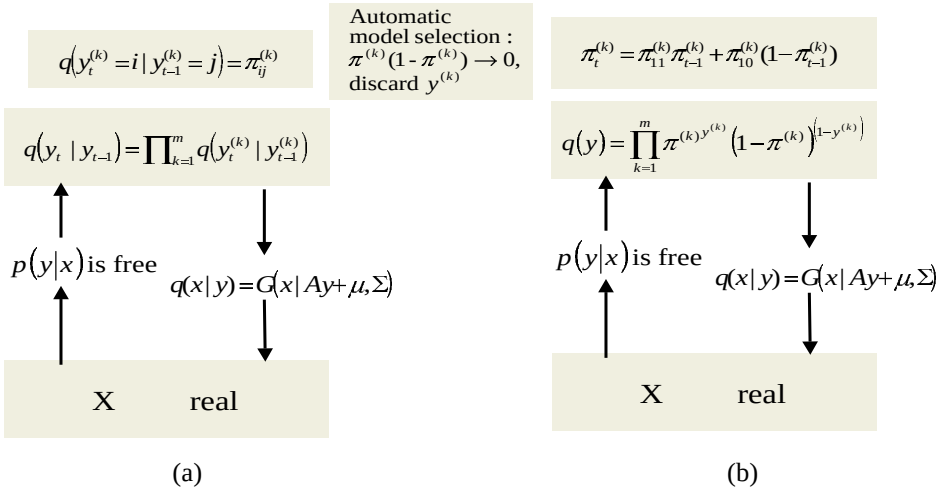


Figure 12. *Two Types of Temporal BFA*

Historically, the use of eq.(40) for selecting m on the classic HMM was firstly suggested in [98] and then in [94] but with a cumbersome form that directly uses $J(m) = -H$. Also, selecting m on the independent HMM was firstly suggested in [96] via a direct use of eq.(40) without any simplification. Then, a simplified form $J(m) = 0.5 \ln |\Sigma| + J_y(m)$ was obtained in [92] but still with a tedious second term $J_y(m)$. In contrast, the criteria in Tab.3, firstly proposed in this paper, are not only compact in their representation but also much simple to be computed accurately.

Furthermore, we can go beyond the first order temporal relation by defining $q(y_t^{(k)} = j|\omega_t^{(k)})$ with a vector $\omega_t^{(k)}$ consisting of past samples up to the order $\kappa \geq 1$. However, the number of free parameters will increases with an order $m2^{\kappa-1}$. Alternatively, we can define the following transfer probability for the purpose:

$$q(y_t^{(k)} = j|\omega_t) = s(\beta_k^T \omega_t^{(k)} + b_k)^{y_t^{(k)}}[1 - s(\beta_k^T \omega_t^{(k)} + b_k)]^{1-y_t^{(k)}}. \tag{189}$$

An adaptive learning algorithm that implements the learning by eq.(164) is given in [92] on page 839. Similar to making temporal NFA, we can also make temporal BFA via harmony learning as described by eq.(157).

Next, we consider extensions of LMSER to temporal LMSER. Similar to the relation between BFA and LMSER, the key point is that $p(y|x)$ is given directly by a forward mapping instead of solving a nonlinear maximization.

In the implementation of a TBYY instantaneous system, this $p(y|x)$ remains same as given in eq.(134). The Ying step is the same as above discussed and the Yang step is still given by eq.(138) simply with q_j replaced by $\pi^{(j)}$. Similarly, the criteria $J(m)$ are still those given by Tab.3.

In the implementation of a TBYY process system, eq.(134) is modified into

$$q(x|y) = G(x|Ay + \mu, \sigma^2 I),\ p(y|x) = \delta(y - s(Wx + Ky_{t-1} + \nu)), \tag{190}$$

for being consistent to its format in eq.(163). That is, the effect of the past y_{t-1} needs to be taken in consideration. Still, the Ying step is the same as above discussed and the criteria $J(m)$ are given by Tab.3. However, eq.(138) should be modified with not only q_j given by eq.(186) but also the following updating on K added in:

$$K^{new} = K^{old} + \eta_0 D_s[\psi(y) + \phi(y)]y_{t-1}^T. \tag{191}$$

Decomposing $x = [\xi, \zeta]$ into two parts, a supervised learning based three layer net by eq.(143) can also be extended to the temporal versions, as shown Fig.13.

In the implementation of a TBYY instantaneous system, eq.(143) remains the same. The Ying step is the same as above discussed but with ζ_t in the place of x_t, and the Yang step is still given by eq.(138) simply with q_j replaced by $\pi^{(j)}$ and with ξ_t replaced by x_t. Similarly, the criteria $J(m)$ are still those given by Tab.3.

In the implementation of a TBYY process system, eq.(143) can be implemented with

$$y_t = s(W\xi_t + Ky_{t-1} + \nu),\ \sigma^2 = \frac{1}{dN}\sum_{t=1}^{N}\|\zeta_t - \mu - As(W\xi_t + Ky_{t-1} + \nu)\|^2. \tag{192}$$

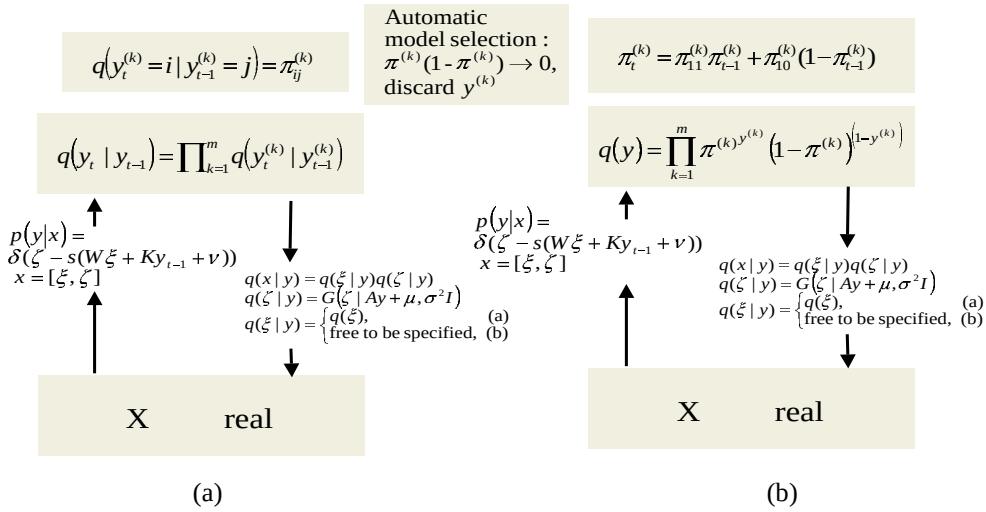


Figure 13. *Two Types of Recurrent Networks*

Still, the Ying step is the same as above discussed but with ζ_t in the place of x_t, and the criteria $J(m)$ are given by Tab.3. However, the Yang step consists of both eq.(138) with q_j given by eq.(186) and with eq.(191) for updating K.

As shown in Fig.13, the above two types of temporal extensions of supervised learning based three layer nets actually are types of recurrent networks with supervised learning. The one with the implementation of a TBYY instantaneous system is a three layer net with its hidden units being recurrently input by its past values. While the one with the implementation of a TBYY process system is a three layer net with the past values of the hidden units recurrently input to not only the hidden units but also the input units together with ξ_t. In other words, the above discussed Ying-Yang learning procedures also provide new tools to handle the two types of recurrent networks.

7. Conclusion

Studies on the ICA problem have become quite popularized in the literature of neural networks. Typical approaches for solving ICA problems have been summarized. Particularly, advances on the ICA studies that consider hybrid sources of both subGaussians and superGaussians and on the ICA extensions that consider noise and temporal dependence have been overviewed. Instead of targeting at a complete survey on all the main existing results, for which there are already several survey papers in the literature, the present paper has made a systematic sketch from the perspective of Bayesian Ying-Yang independence learning. Not only new insights are provided on existing results in the literature, but also a number of further results are presented. For clarity, these new results are summarized as follows:

- A Stiefel manifold based nonlinear Hebbian learning (see Sec.2.2).
- A simplified LPM-ICA algorithm that works on hybrid sources (see Sec.4.2).
- A forward mapping model for estimating density $q(x)$ (see Sec.4.3).
- Regularized adaptive learning on FA and NFA with criteria for selecting factors and with algorithms that make automatic selection (see Sec. 5.2 & Sec..5.3).
- Extensions of LMSER learning with hidden factors selected either by criteria or automatically during parameter learning (see Sec. 5.4).
- Adaptive learning on a three layer net with an ICA type algorithm in place of the conventional back-propagation algorithm, as well as with hidden units selected either by an easy implemented criterion or automatically during parameter learning (see Sec. 5.6).
- A comparative understanding on two types of temporal BYY systems.

- Modified algorithms for TFA and temporal NFA with states selected automatically during parameter learning (see Sec.6.3).
- Algorithms for variants of HMM models that are extended from BFA and LMSER, with states selected either by criteria or automatically during parameter learning (see Sec. 6.5).

Acknowledgment: The work described in this paper was fully supported by a grant from the Research Grant Council of the Hong Kong SAR (Project No: CUHK 4184/03E).

References

[1] Akaike, H. (1974), "A new look at the statistical model identification," *IEEE Tr. Automatic Control*, **19**, 714-723.
[2] Abed-Meraim, K., Moulines, E., & Loubaton, P. (1997), "Prediction error method for second-order blind identification," *IEEE Trans. on Signal Processing, 45(3)*, 694-705.
[3] Amari, S. (1999), "Natural Gradient Learning for Over- and Under-Complete Bases in ICA", *Neural Computation 11*, 1875-1883.
[4] Amari, S. & Cichocki, A. (1998), "Adaptive blind signal Processing: neural network approaches", *Proc. of IEEE,Vol. 86(10)*, 2026-2048.
[5] Amari, S., Chen, T.-P., & Cichocki, A. (1997), "Stability analysis of adaptive blind source separation", *Neural Networks, 10*, 1345-1351.
[6] Amari, S., Cichocki, A., & Yang, H. (1996), "A new learning algorithm for blind signal separation", *Advances in NIPS, 8* , MIT Press, 757-763.
[7] Anderson, T.W., & Rubin, H. (1956), "Statistical inference in factor analysis", *Proc. Berkeley Symp. Math. Statist. Prob. 3rd 5*, UC Berkeley, 111-150.
[8] Attias, H. (1999), "Independent factor analysis", *Neural Computation, 11*, 803-851.
[9] Baldi, P., & Brunak, S. (2001), *Bioinformatics: The Machine Learning Approach*, The MIT Press, Cambridge, MA.
[10] Bell, A. & Sejnowski, T. (1995), "An information maximization approach to blind separation and blind deconvolution", *Neural Computation, 17*, 1129-1159.
[11] Belouchrani, A., & Cardoso, J.-F. (1995), "Maximum likelihood source separation by the expectation-maximization technique: deterministic and stochastic implementation", *Proc. NOLTA95*, 49-53.
[12] Bourlard, H. & Kamp, Y. (1988), "Auto-association by multilayer Perceptrons and singular value decomposition", **Biol. Cyb. 59**, 291-294.
[13] Bozdogan, H. (1987) "Model Selection and Akaike's Information Criterion: The general theory and its analytical extension", *Psychometrika*, **52**, 345-370.
[14] Brown, R.G., & Hwang, P.Y.C., *Introduction to random signals and Applied Kalman Filtering*, John Wiley & Sons, Inc., 1997.
[15] Cardoso, J.-F. & Laheld, B. (1996), "Equivariant adaptive source separation", *IEEE Trans. on Signal Processing., 45(2)*, 434-444.
[16] Cardoso, J.-F. & Comon , P. (1996), "Independent Component Analysis: A Survey of Some Algebraic Methods", *Proc. IEEE ISCAS96*, Vol2, 93-96.
[17] Cardoso, J.-F. (1998), "Blind signal separation: Statistical Principles", *Proc. of IEEE,Vol. 86(10)*, 2009-2025.
[18] Cao, X.-R. & Liu, R.-W. (1996), "General approach to blind source separation," *IEEE Trans. on Signal Processing, 44*, pp562-571, Mar. 1996.
[19] Chen, T.P., & Amari, S. (2001), "Unified Stabilization Approach to principal and minor components extraction algorithm," *Neural Networks, 14*, 1377-1387.
[20] Cheung, C.C. & Xu, L. (1999), "Some Global and Local Convergence Analysis on The Information-Theoretic Independent Component Analysis Approach", *Neurocomputing, 30*, 79-102.
[21] Cheung, Y. M. & Xu, L. (2000), "Rival Penalized Competitive Learning Based Approach for Discrete-valued Source Separation", *Intl. J. of Neural Systems, 10(6)*, 483-490.
[22] Comon, P. (1994), "Independent component analysis: a new concept? " *Signal Processing, 36*, 287-314,
[23] Dayan, P. & Zemel, R. S., (1995), "Competition and multiple cause models", *Neural Computation 7*, 565-579.
[24] Dempster, A. P., Laird, N. M., & Rubin, D. B., (1977), "Maximum- likelihood from incomplete data via the EM algorithm", *J. of Royal Statistical Society*, **B39**, 1-38.
[25] Edelman, A., Arias, T.A., & Smith, S.T., (1998), "The geometry of algorithms with orthogonality constraints," *SIAM J. Matrix Anal. APPL., vol. 20, no. 2*, pp. 303-353.
[26] Gaeta, M., & Lacoume, J.-L. (1990). "Source separation without prior knowledge: the maximum likelihood solution", *Proc. EUSIPO 90*, 621-624.

[27] Girolami, M. (1998), "An alternative perspective on adaptive independent component analysis algorithms", *Neural Computation, 10*, 2103-2114.
[28] Grellier, O. & Comon, P. (1998), "Blind Separation of Discrete Sources," *IEEE Signal Processing Letters, 5(8)*, 212-214. .
[29] Herault, J. & Jutten, C. (1986), "Space or time adaptive signal processing by neural network models", In J. S. Denker (Ed.), Neural networks for computing: Proceedings of AIP Conference (pp. 206-211), American Institute of Physics.
[30] Hinton, G. E., Dayan, P., Frey, B.J., & Neal, R.N (1995), "The wake-sleep algorithm for unsupervised learning neural networks", *Science 268*, 1158-1160.
[31] Hu, X. L. & Xu, L. (2003), "A Comparative Study of Several Cluster Number Selection Criteria", *Proc. of IDEAL03*, Lecture Notes in Computer Science, LNCS 2690, Springer-Verlag, pp195-202.
[32] Hyvarinen, A. & Oja, E. (1997), "A fast fixed-point algorithm for independent component analysis", *Neural Computation, 9*, 1483-1492.
[33] Hyvarinen, A., (1998), "Independent component analysis in the presence of Gaussian noise by maximizing joint likelihood", *Neurocomputing 22*, 49-67.
[34] Hyvarinen, A. (1999), "Survey on Independent Component Analysis", *Neural Computing Surveys 2*, 94-128.
[35] Jutten, C. & Herault, J. (1991), "Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture", *Signal Processing, 24*, 1-20.
[36] Kalman, R.E., "A New Approach to linear filtering and prediction problems", *Trans. ASME-J Basic Engineering*, March, 35-45, 1960.
[37] Kamimura, R. & Kamimura, K, (2001), "Competitive Learning by Mutual Information Maximization", *Proc. of IJCNN01*, Washington, DC, July 15-19, 2001, 926-931.
[38] Karhunen, J. & Joutsensalo, J. (1994), "Representation and separation of signals using nonlinear PCA type learning", *Neural Networks, 7(1)*, 113-127.
[39] Karhunen, J., Pajunen, P., & Oja, E. (1998), "The nonlinear PCA criterion in blind source separation: Relations with other approaches ", *Neurocomputing 22(1-3)*, 5-20.
[40] Kawamoto, M., Matsuoka, K., & Ohnishi, N. (1998), "A method of blind separation for convolved non-stationary signals", *Neurocomputing 22(1-3)*, 157-171.
[41] King, I & Xu, L. (1995), "Adaptive contrast enhancement by entropy maximization with a 1-K-1 constrained network", *Proc. ICONIP95*, 703-706.
[42] Kormylo, J. & J. Mendel, J., (1983), "Maximum likelihood seismic deconvolution," *IEEE Trans. Geosci. Remote Sensing, 21*, 72-81.
[43] Lee, T., Girolami, M., & Sejnowski, T. (1999). "Independent component analysis using an extended infomax algorithm for mixed sub-Gaussian and super-Gaussian sources". *Neural Computation, 11*, 409-433.
[44] Liu, Z.Y., Chiu, K.C., & Xu, L. (2003), "The One-bit-Matching Conjecture for Independent Component Analysis", in press, *Neural Computation*.
[45] Lu, W., & Rajapakse, J. (2000), "A neural network for under complete Independent Component Analysis", In *Proc. 8th European Symposium on Artificial Neural Networks*, Bruges, Belgium.
[46] Ma, J., Wang, T., & Xu, L. (2003), "A gradient BYY harmony learning rule on Gaussian mixture with automated model selection", in press, *Neurocomputing* .
[47] Matsuoka, K., Ohya, M., & Kawamoto, M. (1995), "A neural net for blind separation of non-stationary signals", *Neural Networks, 8(3)*, 411-419.
[48] McDonald, R, (1985), *Factor Analysis and Related Techniques*, Lawrence Erlbaum, Hillsdale, NJ, 1985.
[49] Mendel, J. (1983), *Optimal Seismic Deconvolution: An Estimation Based Approach*, New York: Academic, 1983.
[50] Miller, G. & Horn, D. (1998), "Probability Density Estimation Using Entropy Maximization", *Neural Computation, 10*, 1925-1938.
[51] Molgedey, L. & Schuster, H. (1994). "Separation of a mixture of independent signals using time delayed correlations", *Physical Review Letters, 72(23)*, 3634- 3637.
[52] Moulines, E., Cardoso, J., & Gassiat, E. (1997), "Maximum likelihood for blind separation and deconvolution of noisy signals using mixture models", *Proc. ICASSP97*, 3617-3620.
[53] Moreau, E., & Macchi, O. (1996), " High order contrasts for self-adaptive source separation", *International Journal of Adaptive Control and Signal Processing, 10*, pp19-46.
[54] Moreau, E. (2001), "A generalization of joint diagonalization criteria for source separation," *IEEE Trans. on Signal Processing, 49(3)*, pp.530-541.
[55] Oja, E. (1982), "A simplified neuron model as a principal component analyzer", *Journal of Mathematical Biology, 16*, 267-273.
[56] Oja, E. (1989). Neural networks, principal components, and subspaces", *International Journal of Neural Systems, 1*, 61-68.

[57] Oja, E. Ogawa, H. & Wangviwattana, J. (1991), " Learning in nonlinear constrained Hebbian networks", *Proc. ICANN'91*, 385-390.
[58] Oja, E. (1992), "Principal components, minor components, and linear neural networks", *Neural Networks, 5*, 927-935, 1992.
[59] Oja, E. (1997), "The nonlinear PCA learning rule in independent component analysis", *Neurocomputing, 17*, 25-45.
[60] Pajunen, P., Hyvarinen, A. , & Karhunen, J. (1996), "Nonlinear blind source separation by self-organizing Maps", *Proc. ICONIP96*, 1207-1210.
[61] Pajunen, P. (1997), "A competitive learning algorithm for separating binary sources," *Proc. ESANN'97*, 255-260.
[62] Pearlmutter, B., & Parra, L. (1996). "A context-sensitive generalization of ICA", *Proc. ICONIP96*, 151-157.
[63] Pham, D. T., Garrat, P., & Jutten, C. (1992), "Separation of a mixture of independent sources through a maximum likelihood approach.", *Proc. EUSIPCO92*, 771-774.
[64] Rabiner, L. & Juang, B.H., *Fundamentals of Speech Recognition*, Prentice Hall, Inc., 1993.
[65] Redner, R.A. & Walker, H.F. (1984), "Mixture densities, maximum likelihood, and the EM algorithm", *SIAM Review*, **26**, 195-239.
[66] Ridder, D.D., Duin, D., & Kittler, R (2002), " Texture description by independent components", in *Proc. the Joint International workshop on syntactical and structural pattern recognition*, Windsor, Canada.
[67] Rissanen, J. (1986), "Stochastic complexity and modeling", *Annals of Statistics*, **14**(3), 1080-1100.
[68] Rissanen, J. (1999), "Hypothesis selection and testing by the MDL principle", *Computer Journal*, **42** (4), 260-269.
[69] Robinson, E., (1976), "Predictive decomposition of time series with application to seismic exploration," *Geophysics., 32*, 418-484, 1967.
[70] Roth, Z. & Baram, Y. (1996), "Multidimensional Density Shaping by Sigmoids", *IEEE Trans on Neural Networks, 7(5)*, 1291-1298.
[71] Rumelhart, D.E., Hinton, G.E., & Williams, R.J., (1986), "Learning internal representations by error propagation", *Parallel Distributed Processing*, **1**, MIT press.
[72] Rubi, D. & Thayer D. (1976), "EM algorithm for ML factor analysis", *Psychometrika 57*, 69-76.
[73] Saul, L., & Jordan, M. I. (1995), "Exploiting tractable structures in intractable Networks", *Advances in neural information processing systems, 8*, MIT Press , 486-492.
[74] Saund, E., (1995), "A multiple cause mixture model for unsupervised learning", *Neural Computation*, Vol.7, pp51-71.
[75] Sato, Y. (1975), "A Method for Self-Recovering Equalization for Multilevel Amplitude Modulation System," *IEEE Trans on Communication, 23*, 679-682.
[76] Schwarz, G. (1978), "Estimating the dimension of a model", *Annals of Statistics*, **6**, 461-464.
[77] Shalvi , O. & Weinstein, E. (1990), "New Criteria for Blind Deconvolution of Non-minimum Phase Systems Channels", *IEEE Trans on Information Theory, 36(2)*, 312-321.
[78] Simon, C., Loubaton, P., Vignat, C., Jutten, C., & D'urso, G. (1998), "Blind source separation of convolutive mixtures by maximization of fourth-order cumulants : the non i.i.d. case", *ICASSP98* , 12-15.
[79] Sorouchyari, E. (1991). "Blind separation of sources, part III: Stability analysis" *Signal Processing, 24*, 21-30.
[80] Stone, J. V. (2001), "Blind Source Separation Using Temporal Predictability", *Neural Computation, 13*, 1559-1574.
[81] Szu, H., & Kopriva, I., (2002), "Comparison of Lagrange constrained neural network with traditional ICA methods", *Proc. of IJCNN02*, May 12 - 17, 2002, Honolulu, Hawaii, pp466-471.
[82] Taleb, A. & Jutten, C. (1997), "Nonlinear source separation: The post-nonlinear Mixtures", *Proc. ESANN97*, 279-284.
[83] Tong, L., Inouye, Y., & Liu, R. (1993) "Waveform-preserving blind estimation of multiple independent sources", *IEEE Trans. on Signal Processing 41*, 2461-2470.
[84] Tong, L. & Perreau, S. (1998), "Blind signal separation: Statistical Principles", *Proc. of IEEE, Vol. 86(10)*, 1951-1068.
[85] Welling, M. & Weber, M. (2001), "A Constrained EM Algorithm for Independent Component Analysis", *Neural Computation 13*, 677-689 .
[86] Wiggins, R. (1978), "Minimum entropy deconvolution," *Geoexploration, 16*, 21-25.
[87] Xu, L. (2003a), "BYY Learning, Regularized Implementation, and Model Selection on Modular Networks with One Hidden Layer of Binary Units", *Neurocomputing*, Vol.51, p227-301.
[88] Xu, L. (2003b), "Data smoothing regularization, multi-sets-learning, and problem solving strategies", *Neural Networks, Vol. 15, Nos. 5-6*, 817-825.

[89] Xu, L. (2002a), "BYY Harmony Learning, Structural RPCL, and Topological Self-Organizing on Mixture Models", *Neural Networks, Vol. 15, Nos. 8-9*, 1125-1151.
[90] Xu, L., (2002b), "Bayesian Ying Yang Harmony Learning", *The Handbook of Brain Theory and Neural Networks*, Second edition, (M.A. Arbib, Ed.), Cambridge, MA: The MIT Press, pp1231-1237.
[91] Xu, L. (2002c). "Mining Dependence Structures from Statistical Learning Perspective", in H. Yin et al., eds., *Proc. IDEAL2002: Lecture Notes in Computer Science, 2412*, Springer-Verlag, 285-306.
[92] Xu, L. (2001a), "BYY Harmony Learning, Independent State Space and Generalized APT Financial Analyses", *IEEE Trans on Neural Networks, 12(4)*, 822-849.
[93] 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.
[94] Xu, L. (2000a), "Temporal BYY Learning for State Space Approach, Hidden Markov Model and Blind Source Separation", *IEEE Trans on Signal Processing 48*, 2132-2144.
[95] Xu, L.(2000b), "BYY Learning System and Theory for Parameter Estimation, Data Smoothing Based Regularization and Model Selection", *Neural, Parallel and Scientific Computations*, Vol. 8, pp55-82.
[96] Xu, L. (1999a), "Temporal Bayesian Ying-Yang Dependence Reduction, Blind Source Separation and Principal Independent Components", *Proc. IJCNN'1999*, July 10-16, Washington, D.C., Vol.2, pp1071-1076.
[97] Xu, L. (1999b), "Bayesian Ying-Yang Unsupervised and Supervised Learning: Theory and Applications", *Proc. of 1999 Chinese Conf. on Neural Networks and Signal Processing*, pp12-29, Shantou, China, 1999.
[98] Xu, L. (1998), "Bayesian Ying-Yang System and Theory as a Unified Statistical Learning Approach :(V) Temporal Modeling for Temporal Perception and Control", *Proc. ICONIP'98*, October 21-23, 1998, Kitakyushu, Japan, Vol.2, pp877-884.
[99] Xu, L. (1998a), "Bayesian Kullback Ying-Yang Dependence Reduction Theory", *Neurocomputing 22 (1-3)*, 81-112, 1998.
[100] Xu, L. (1998b), "RBF Nets, Mixture Experts, and Bayesian Ying-Yang Learning", *Neurocomputing*, Vol. 19, No.1-3, 223-257.
[101] Xu, L.(1998c), "Bayesian Ying-Yang Learning Theory For Data Dimension Reduction and Determination", *Journal of Computational Intelligence in Finance*, Finance & Technology Pub., Vol.6, No.5, pp 6-18, 1998.
[102] Xu, L., (1998d), "Bayesian Ying-Yang System and Theory as A Unified Statistical Learning Approach (VII): Data Smoothing", *Proc. ICONIP'98*, Oct. 21-23, 1998, Kitakyushu, Japan, Vol.1, pp243-248.
[103] Xu, L, (1998e), "BKYY Three Layer Net Learning, EM-Like Algorithm, and Selection Criterion for Hidden Unit Number", *Proc. ICONIP'98*, Oct. 21-23, 1998, Kitakyushu, Japan, Vol.2, pp631-634.
[104] Xu, L, (1998g), "Bayesian Ying-Yang System and Theory as A Unified Statistical Learning Approach :(VI) Convex Divergence, Convex Entropy and Convex Likelihood", *Proc. IDEAL98*, pp1-12.
[105] Xu, L, (1998j), "BKYY Dimension Reduction and Determination", *Proc. of IEEE-INNS International Joint Conference on Neural Networks (IJCNN98)*, May 5-9, 1998, Anchorage, Alaska, Vol.III, pp1822-1827.
[106] Xu, L., Cheung, C.C., & Amari, S.-I. (1998a) "Learned Parametric Mixture Based ICA Algorithm", *Neurocomputing 22 (1-3)*, 69-80.
[107] Xu, L., Cheung, C.C., & Amari, S.-I. (1998b), "Further Results on Nonlinearity and Separation Capability of A Linear Mixture ICA Method and Learned Parametric Mixture Algorithm", *Proc. I&ANN'98*, Feb. 9-10, 1998, Tenerife, Spain, pp39-44.
[108] Xu, L. (1997a),"Bayesian Ying-Yang System and Theory as A Unified Statistical Learning Approach: (I) Unsupervised and Semi-Unsupervised Learning", in S. Amari and N. Kassabov eds., *Brain-like Computing and Intelligent Information Systems*, Springer-Verlag, pp.241-274.
[109] Xu, L. (1997c), "Bayesian Ying-Yang System and Theory as A Unified Statistical Learning Approach (III): Models and Algorithms for Dependence Reduction, Data Dimension Reduction, ICA and Supervised Learning", in K.M. Wong, et al, eds, *Theoretical Aspects of Neural Computation: A Multidisciplinary Perspective*, Springer-Verlag, pp43-60.
[110] Xu, L. (1997d), "Bayesian Ying-Yang Learning Based ICA Models", *Proc. 1997 IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing VII*, Sept. 24-26, 1997, Florida, pp476-485.
[111] Xu, L., Cheung, C.C., Yang, H.H., & Amari, S.-I. (1997), "Independent Component Analysis by The Information-Theoretic Approach with Mixture of Density", *Proc. IJCNN97*, Vol. III, 1821-1826.
[112] Xu, L., Cheung, C.C., Ruan, J., & Amari, S.-I. (1997), "Nonlinearity and Separation Capability: Further Justification for the ICA Algorithm with A Learned Mixture of Parametric Densities", *Proc. ESANN97*, Bruges, April 16-18, 1997, pp291-296.
[113] Xu, L., (1997g), "Bayesian Ying-Yang Machine, Clustering and Number of Clusters", *Pattern Recognition Letters 18*, No.11-13, 1167-1178.

[114] Xu, L., (1997h), "New Advances on Bayesian Ying-Yang Learning System with Kullback and Non-Kullback Separation Functionals", *Proc. of 1997 IEEE-INNS International Joint Conference on Neural Networks (IJCNN97)*, June 9-12, 1997, Houston, TX, USA, Vol. III, pp1942-1947.
[115] 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)*, organized by S.Amari, S.Tanaka & A.Cichocki, RIKEN, Japan, April 10, 1996.
[116] Xu, L., & Amari,S.-I. (1996), "A general independent component analysis framework based on Bayesian-Kullback Ying-Yang Learning", *Proc. ICONIP96*, 1253-1240.
[117] Xu, L., (1996c), "A Maximum Balanced Mapping Certainty Principle for Pattern Recognition and Associative Mapping", *Proc. of INNS WCNN96*, Sept. 15-18, 1996, San Diego, CA, pp.946-949.
[118] Xu, L., & Jordan, M.I. (1996), "On convergence properties of the EM algorithm for Gaussian mixtures", *Neural Computation, 8*, No.1, 1996, 129-151.
[119] Xu, L., (1995a), "Bayesian-Kullback Coupled YING-YANG Machines: Unified Learnings and New Results on Vector Quantization", *Proc. ICONIP95*, Oct 30-Nov.3, 1995, Beijing, China, pp.977-988.
[120] Xu, L. (1994a), "Beyond PCA Learning: From Linear to Nonlinear and From Global Representation to Local Representation", *Proc. ICONIP94*, Oct. 17-20, 1994, Seoul, Korea, Vol.2, pp.943-949.
[121] Xu, L. (1994b),"Theories for Unsupervised Learning: PCA and Its Nonlinear Extensions", Invited Talk, *Proc. of IEEE ICNN94*, June 26-July 2, 1994, Orlando, Florida, Vol.II, pp.1252-1257.
[122] Xu, L., Oja, E., & Suen, C.Y. (1992), "Modified Hebbian learning for curve and surface fitting", *Neural Networks, 5*, pp393-407.
[123] Xu, L. (1991&93) "Least mean square error reconstruction for self-organizing neural-nets", *Neural Networks 6*, 627-648, 1993. Its early version on *Proc. IJCNN91'Singapore*, 2363-2373, 1991.
[124] Xu, L., Krzyzak, A., & Oja, E. (1991), "A Neural Net for Dual Subspace Pattern Recognition Methods", *International Journal of Neural Systems, 2(3)*, 169-184.
[125] Xu, L., Yan, P.F., & Chang, T. (1987a), "Phase Information is Richly Contained in the End Point of Finite length Digital Signal", *Proc. IEEE ISCAS87*, Vol.3, 851-854.
[126] Xu, L., Yan, P.F., & Chang, T. (1987b), "Almost Unique Specification of Discrete Finite Length Signal: From Its End Point and Fourier Transform Magnitude", *Proc. of ICASSP87*, Vol.3, 2097-2100.
[127] Xu, L., Yan, P.F., & Chang, T. (1987c), "Semi-Blind Deconvolution of Finite Length Sequence: (I) Linear Problem & (II) Nonlinear Problem ", *SCIENTIA SINICA*, Series A, Vol.XXX (12) , 1318-1344.
[128] Xu, L., Yan, P.F., & Chang, T. (1987d), "Investigation on Semi-Blind deconvolution Using Nonlinear Programming", *Science Bulletin, 24*, 1900-1903.
[129] Yellin, D. & Weinstein, E. (1996), "Multi-channel signal separation: Methods and analysis", *IEEE Trans. on Signal Processing, 44*, 106-118.
[130] Zhang, B. L., Xu, L., & Fu, M. Y. (1996), "Learning Multiple Causes by Competition Enhanced Least Mean Square Error Reconstruction", *Intl J. of Neural Systems, 7(3)*, 223-236.

Lei Xu (IEEE Fellow & IAPR Fellow) is a chair professor of Computer Science and Engineering in Chinese University of Hong Kong (CUHK), and is also a guest professor in four universities in PRC and UK. After receiving his Ph.D from Tsinghua Univ. in early 1987, he joined Peking Univ. where he became one of ten university-level exceptionally promoted young associate professors in 1988 and further been exceptionally promoted to a full professor in 1992. During 1989-93, he worked at several universities in Finland, Canada and USA, including Harvard and MIT. He joined CUHK in 1993 as a senior lecturer, became a professor in 1996 and then took the current position since 2002. Prof. Xu has published over 260 academic papers, with a number of them well cited in literature. He has given numerous keynote/plenary/invited/tutorial talks in intl. major Neural Networks (NN) conferences, such as WCNN, IEEE-ICNN, IJCNN, ICONIP, etc. He is on the Governor Board of Intl. NN Society, the chair of Computational Finance Technical Committee of IEEE Neural Networks Society, a past president of Asia-Pacific NN Assembly, and an associate editor for six Intl. journals on NN, including Neural Networks, IEEE Trans. Neural Networks. He was a ICONIP'96 program committee chair and a general chair of IDEAL'98, IDEAL'00, IEEE CIFER'03 as well as the joint-ICANN-ICONIP03 Program Committee co-chair. Also, he has served as program committee members in intl. major NN conferences in the past decade, including IJCNN (97,99,00,01,02,03), WCNN(95,96), IEEE-ICNN (96), etc. He has received several Chinese national prestigious academic awards (including the National Nature Science Prize) and also some international awards (including an 1995 INNS Leadership Award). Prof. Xu is a Fellow of IEEE, Fellow of IAPR (International Association for Pattern Recognition), member of European Academy of Sciences, also a member of Sigma Xi and a member of American Association for the Advancement of Science.